INTERNATIONAL DOCTORAL
SCHOOL OF THE USC

Heba
Basheer Saeed Alateyat

PhD Thesis

Two-dimensional visualization of classification and regression problems. Automatic prediction of behavior from sensory data in autism spectrum disorder

Santiago de Compostela, 2023

Doctoral Programme in Information Technology Research

TESE DE DOUTORAMENTO

TWO-DIMENSIONAL VISUALIZATION OF CLASSIFICATION AND REGRESSION PROBLEMS. AUTOMATIC PREDICTION OF BEHAVIOR FROM SENSORY DATA IN AUTISM SPECTRUM DISORDER

Heba Basheer Saeed Alateyat

ESCOLA DE DOUTORAMENTO INTERNACIONAL
DA UNIVERSIDADE DE SANTIAGO DE COMPOSTELA
PROGRAMA DE DOUTORAMENTO EN
INVESTIGACIÓN EN TECNOLOXÍAS DA INFORMACIÓN

SANTIAGO DE COMPOSTELA
2023

DECLARACIÓN DA AUTORA DA TESE

Dona **Heba Basheer Saeed Alateyat**

Título da tese: **Two-dimensional visualization of classification and regression problems. Automatic prediction of behavior from sensory data in autism spectrum disorder**

Presento a miña tese, seguindo o procedemento adecuado ao Regulamento, e declaro que:

1. A tese abarca os resultados da elaboración do meu traballo.
2. De ser o caso, na tese faise referencia ás colaboracións que tivo este traballo.
3. Confirmo que a tese non incorre en ningún tipo de plaxio doutros autores nin de traballos presentados por min para a obtención doutros títulos.
4. A tese é a versión definitiva presentada para a súa defensa e coincide a versión impresa coa presentada en formato electrónico.

E comprométome a presentar o Compromiso Documental de Supervisión no caso de que o orixinal non estea na Escola.

En Santiago de Compostela, Febreiro de 2023

Asdo. Heba Basheer Saeed Alateyat

Autora da tese

AUTORIZACIÓN DOS DIRECTORES DA TESE

Don **Manuel Fernández Delgado** e Dona **Eva Cernadas García**, Profesores Titulares de Universidad da Área de Ciencias da Computación e Intelixencia Artificial, Departamento de Electrónica e Computación, Universidade de Santiago de Compostela

En condición de: **co-director e co-directora da tese**

Título da tese: **Two-dimensional visualization of classification and regression problems. Automatic prediction of behavior from sensory data in autism spectrum disorder**

INFORMAN:

Que a presente tese correspóndese co traballo realizado por Dona **Heba Basheer Saeed Alateyat**, baixo a nosa dirección, e autorizamos a súa presentación, considerando que reúne os requisitos esixidos no Regulamento de Estudos de Doutoramento da USC, e que como co-directores desta non incorre nas causas de abstención establecidas na Lei 40/2015.

En Santiago de Compostela, Febreiro de 2023

Asdo. Manuel Fernández Delgado
Co-director da tese

Asdo. Eva Cernadas García
Co-directora da tese

I dedicate this work to my father's soul, may God have mercy on him,
who always dreamed of obtaining me a Ph.D.,
to my dear mother who always supported and stood by me,
to my beloved son, the hope of my life,
who endured my absence from him during my studies,
to my sister and brothers,
and Fadia, my dear friend,
thank you all for your support.

قال تعالى:

﴿يَرْفَعِ اللَّهُ الَّذِينَ آمَنُوا مِنكُمْ وَالَّذِينَ أُوتُوا الْعِلْمَ دَرَجَاتٍ وَاللَّهُ بِمَا تَعْمَلُونَ خَبِيرٌ﴾

﴿ Allah will raise those who have believed among you and those who were given knowledge, by degrees. And Allah is Aware of what you do ﴾

Holy Quran

Acknowledgments

Getting Ph.D. was a goal that turned into a dream with the time of many challenges I faced. But I kept trying to convert it into reality. Leaving my family and my son for the first time was difficult. But when I reached Spain and met great supervisors, Mr. Manuel and Ms. Eva, who became my new family who learned and guided me in my research and kept listening to my complaints, questions, crying, and the heartbreak of longing after being away from my son.

No words would be enough to thank Mr. Manuel and Ms. Eva.

This work has received financial support from the Consellería de Cultura, Educación e Universidade (accreditation 2019-2022 ED431G-2019/04) and the European Regional Development Fund (ERDF), which acknowledges the CiTIUS - Centro Singular de Investigación en Tecnoloxías Intelixentes da Universidade de Santiago de Compostela as a Research Center of the Galician University System.

Febreiro de 2023

Resumo en galego

Esta tese sitúase no ámbito da aprendizaze automática, que se podería definir como un conxunto de técnicas (tamén chamadas modelos) para a predicción de valores a partir de datos empregando exemplos. Estes exemplos inclúen o valor a predecir (chamado normalmente saída do modelo) e os datos empregados para realizar esta predicción (entradas do modelo). As entradas son vectores multi-dimensionais (patróns) compostos por valores numéricos que poden ser continuos ou discretos. Se a saída a predecir é continua, fálase dun problema de regresión, e o modelo chámase regresor. Se o valor é discreto, falamos de problema de clasificación e de clasificadores. Os exemplos empregados para construir o modelo de aprendizaxe automática compoñen o que normalmente se chama "conxunto de entrenamento".

A dimensionalidade do patrón é en xeral elevada, e isto dificulta entender a dependencia entre cada entrada e a saída, xa que cada entrada influe na saída de forma distinta dependendo das outras entradas. Por esta razón, sería interesante dispor dun mapa bi-dimensional onde se representase, por exemplo mediante un código de cores, a saída que se debería predecir nas distintas rexións do espazo de entrada (é decir, facer unha predicción para cada un dos posibles valores dos patróns). Nos problemas de clasificación, este mapa sería similar aos mapas xeográficos políticos, onde cada clase sería como un país no mapa. Coñecendo a localización dun patrón neste mapa poderiamos predecir a qué clase pertence. Ademais, poderiamos saber que clases están adxacentes ou cercanas, caso no cal sería fácil confundir os seus patróns porque son parecidos, ou lonxanas, caso no cal sería difícil confundir os seus patróns porque son moi diferentes. Un mapa deste tipo permitiría tamén saber se existe sobreposición ou coincidencia entre clases, e entre que clases se produce esta sobreposición. Similarmente, en problemas de regresión este mapa sería similar aos mapas de temperaturas, ou aos mapas topográficos de elevacións do terreo, nos que se representa a temperatura cun código de cores, e novamente sabendo a localización dun patrón obteriamos o valor a predecir. Isto permitiría saber en que direccións medra ou diminúe este valor, con que velocidade, e cais son as rexións con valores baixos, medios ou altos.

A tese formula algoritmos para a visualización bi-dimensional (en 2D) de problemas de clasificación e regresión. Estes algoritmos, chamados **ultra-fast 2D classifier** (UF2DC) e **ultra-fast 2D regressor** (UF2DR), deseñaronse co obxectivo de realizar a clasificación ou regresión sobre o propio espazo 2D. Para isto, emprégase a técnica clásica de análise dis-

criminante linear (LDA) para proxectar os datos multi-dimensionais orixinais ao espazo 2D. Seleccionouse esta técnica entre as moitas que existen para a reducción da dimensionalidade debido aos seus bos resultados para a clasificación en 2D, demostrados en traballos de investigación anteriores, e pola súa elevada eficiencia computacional. Aínda que o LDA se emprega para problemas de clasificación e reducción da dimensionalidade, nesta tese ampliamos o seu uso a problemas de regresión, discretizando a saída en niveis manexados como clases e empregando LDA para a súa proxección ao espazo bi-dimensional.

Os patróns de entrada proxectados ao espazo 2D úsanse para crear un mapa de clasificación ou regresión. Emprégase o mesmo rango de valores nas coordenadas horizontal e vertical deste espazo, definindo así un mapa cadrado. Este mapa divídese nun número de cadrados (ou pixeles, se consideramos o mapa como unha imaxe), cada un dos cais se asociará a un valor da saída que representa unha etiqueta de clase en problemas de clasificación, ou un valor numérico en problemas de regresión. Deste modo, para predecir a saída ante un patrón de entrada multi-dimensional abonda con proxectar este patrón ao espazo 2D, localizar o cadrado do mapa ao cal se proyectou este patrón e proporcionar como predicción o valor do mapa nese cadrado.

Para executar este proceso, o primeiro paso consiste en seleccionar un número axeitado de cadrados en base ao tamano do conxunto de entrenamento. Porque, dependendo deste tamano, será necesario un mapa máis o menos detallado, dito noutras palabras, con máis ou menos resolución. É importante destacar que nos métodos propostos este número de cadrados está acotado superiormente, o cal resulta importante para garantir a súa eficiencia en problemas con moitos patróns de entrenamento. Considerando este número de cadrados e o rango de valores das coordenadas do mapa, é posíbel determinar os índices horizontal e vertical do cadrado ao que se proxectou un patrón de entrada para o cal se desexa realizar unha predicción. Neste punto, o deseño dun procedemiento eficiente require lixeiras diferenzas entre os problemas de clasificación e regresión.

1. Nos problemas de **clasificación**, defínese a clase asociada a un cadrado como a máis votada entre tódolos patróns de entrenamento que son proxectados a este cadrado. Con este fin, créase para cada clase unha matriz de poblacións que indica o número destes patróns que se proxectaron a cada cadrado. A predicción asociada a un cadrado defínese como a clase con meirande poboación no mismo (é decir, faise unha votación). Como en xeral o número de cadrados é meirande que o de patróns, existirán moitos cadrados sen patróns nos que non se pode realizar esta votación. Para resolver este problema de

forma eficiente, esténdese o ámbito espacial (normalmente chamado "máscara") sobre o que se realiza a votación. Con este fin, pártese do cadrado baleiro orixinal e auméntase o tamano da máscara, sempre conservando unha forma cadrada, ata incluir cadrados con patróns.

2. Nos problemas de **regresión**, a votación é substituida por un promedio da saída, que é un valor numérico, sobre tódolos patróns de entrenamento que foron proxectados a este cadrado. Con este fin, créanse dúas matrices. A matriz de saída contén elementos cuxo valor é a suma das saídas de tódolos patróns de entrenamento que se proxectaron ao cadrado asociado a este elemento. A matriz de poboación contén elementos cuxo valor é o número destes patróns de entrenamento proxectados ao devandito cadrado. Cada unha destas dúas matrices contén un valor para tódolos cadrados do mapa. O valor a predecir para un patrón proxectado a un cadrado será o cociente entre os seus elementos correspondentes nas matrices de saída e de poboación. Similarmente ao caso de clasificación, este procedemento non pode realizarse con cadrados baleiros. Por isto, o promedio esténdese a unha máscara centrada no cadrado considerado, co tamano mínimo necesario para incluir cadrados aos que se proxectou algún patrón de entrenamento.

A extensión da máscara require efectuar unha busca que pode estenderse máis ou menos sobre os píxeles do cadrado 2D ao que se proxectan os patróns de entrenamento. Este proceso pode resultar custoso computacionalmente. Por esta razón, a tese inclúe un procedemento eficiente para poder empregar UF2DC e UF2DR en problemas de clasificación y regresión grandes, tanto en número de patróns de entrenamento como de entradas ou clases. O pseudocódigo deste procedemento inclúese tamén na tese.

O mapa de clasificación codifícase na matriz de mapa, onde cada elemento é a predicción (etiqueta de clase para problemas de clasificación, ou valor da función en problemas de regresión) asociada ao seu cadrado correspondente no mapa. O tamano desta matriz é relativamente reducido para garantir a eficiencia dos métodos propostos, tanto en velocidade como en requerimentos de memoria RAM. Nembargantes, este tamano resulta escaso para unha visualización detallada do mapa de clasificación/regresión, que debe ter unha resolución superior. Por isto, a partir desta matriz créase a matriz de imaxe, de meirande tamano, que codifica a imaxe que se visualizará do mapa, mentres que a matriz de mapa usarase somentes para formular unha predicción. A matriz de imaxe créase: 1) aumentando a matriz de mapa e rea-

lizando un promedio cun núcleo de forma rectangular; e 2) realizando na imaxe un suavizado que aplica la función moda estatística por grupos de cadrados sobre a imaxe.

O traballo experimental ilustra gráficamente o funcionamiento de UF2DC e UF2DR en exemplos de dúas e máis dimensións. Neles móstrase a súa capacidade para elaborar representacións espaciais separadas e coherentes das distintas clases, ou dos distintos valores da función a predecir en problemas de regresión. Os experimentos de validación estatística realizan una comparativa dos métodos propostos sobre 26 problemas de clasificación, algúns deles binarios, e 23 problemas de regresión. Todos estes problemas están considerados como de referencia na literatura, están dispoñíbeis nos almacéns públicos de datos e son usados constantemente no ámbito da aprendizaxe automática. Inclúense problemas de grande tamano, ata 5 millóns de patróns e 719 entradas en clasificación, e ata 4 millóns e 482 entradas en regresión.

Na clasificación, a comparativa inclúe o algoritmo SVM (*support vector machine* con núcleo gausiano), aplicado nos patróns multi-dimensionais orixinais; LSVM (ídem usando un núcleo linear) nos patróns orixinais; PNN (*probabilistic neural network*) sobre os patróns orixinais, bi-dimensionais e h-dimensionais, sendo h=número de clases menos 1; e WSVM (proxección dos datos orixinais a 2D usando unha rede neuronal RBF, *radial basis funcion*, deseñada para optimizar a clasificación dunha SVM con cerne linear dacordo coa aproximación denominada *wrapper* na literatura). Incluíronse os clasificadores LSVM e PNN na comparativa pola súa eficiencia computacional en problemas grandes. O acerto acadado por UF2DC clasificando os patróns en 2D é similar aos obtidos polos outros métodos usando os patróns orixinais na maioría dos problemas (20 de 26), sendo a diferencia apreciábel somentes nos 6 problemas restantes. No tocante á velocidade de execución, UF2DC é entre 10 e 1000 veces máis rápido que os outros clasificadores considerados, e a diferencia aumenta moito co tamano dos datos. De feito, no problema máis grande, con 5 millóns de patróns, UF2DC e LSVM son os únicos clasificadores capaces de executarse, obtendo acertos similares. Nembargantes, o primero tarda so 3 minutos mentres que o segundo tarda 30 horas, a pesar de estar deseñado específicamente para datos grandes. O traballo experimental tamén avalía o tiempo empregado por UF2DC para crear a matriz de imaxe, tempo que non supera o minuto en ningún problema de clasificación considerado.

A experimentación de UF2DR en problemas de regresión inclúe unha comparativa con SVR (*support vector regression* con núcleo gausiano), aplicado aos patróns orixinais; LSVR (ídem con núcleo linear), aplicado aos patróns orixinais; e GRNN (*generalized regression*

neural network), aplicada aos patróns orixinais, bi-dimensionais e *h*-dimensionais. Tanto LSVR como GRNN foron incluidos pola súa velocidade en problemas grandes. A correlación acadada por UF2DR cos patróns 2D é moi similar á obtida usando os restantes métodos en 19 de 23 problemas. Canto á velocidade, o método proposto executouse en tódolos problemas, e no máis grande (4.1 millóns de patróns) tarda 2.3 minutos mentres LSVR necesita 53 minutos. Avaliouse tamén a sensibilidade da predicción a respeito do número de niveis usados para cuantificar a saída nos problemas de regresión. Os experimentos demostran que esta sensibilidade non é excesivamente elevada, e que o número de niveles (10) empregado nos experimentos é unha elección axeitada que proporciona uns resultados moi cercanos aos óptimos.

Hai que destacar que UF2DC e UF2DR, ademais de executar a predicción en 2D con fiabilidades similares a outras aproximacións competitivas na literatura, permiten visualizar a estructura dos datos en problemas de clasificación e regresión, proporcionando explicacións ás características e propiedades destes problemas. Como se comentou anteriormente, estas propiedades incluyen, nos problemas de clasificación, superposición e relacións de cercanía ou lonxanía entre clases, e nos problemas de regresión, rexións de valores baixos, medios e altos, e direccións de aumento e diminución da función a predecir.

A segunda parte da tese aborda a predicción automática de comportamentos en base a información sensorial para o tratamiento da desorde de espectro autista na infancia e adolescencia. Esta investigación realizouse en colaboración co Grupo de Medicina Xenómica[1] da Universidade de Santiago de Compostela. As persoas que experimentan a desorde de espectro autista sufren problemas de procesamiento sensorial do entorno (interese inusual, ben hiper- ou hipo-reactivo, cara estímulos visuais, auditivos, olfactivos ou táctiles) e tamén problemas de comportamento (aillamento e falla de interacción social, problemas na comunicación non verbal, comportamentos repetitivos e restrinxidos, irritabilidade, ansiedade e depresión). As alteracións sensoriais detéctanse a idades moito máis temperás, xa nos primeiros meses de vida, que os problemas de comportamento, que se manifiestan a idades posteriores e aínda na adolescencia, tendo un grande impacto no desenvolvemento persoal e as relacións sociais da persoa afectada.

A literatura demostrou a existencia de relacións entre alteracións sensoriais e problemas de comportamento. Ámbolos dous analízanse empregando sendos cuestionarios: *sensory profile-2* (SP2) e *child behavior check-list* (CBCL), que proporcionan medidas estandarizadas

[1] http://www.xenomica.eu

do desenvolvemento persoal e guías para intervencións clínicas. Os métodos de aprendizaxe automática aportan interesantes posibilidades para o estudo das relacións entre ambos e chegar así a saber en que medida un deles é útil para predecir o outro. Específicamente, nesta tese o obxectivo é usar as respostas do cuestionario SP2, que avalía problemas sensoriais, para predecir as respostas no cuestionario CBCL, que avalía problemas de comportamento. No cuestionario SP2 empréganse normalmente 6 grupos de respostas, cada unha delas con valores entre 0 e 5, chamados en inglés *avoiding*, *registration*, *seeking*, *sensitivity*, *total* and *touch*. Cada grupo inclúe unha serie de respostas do cuestionario, e *total* inclúe tódalas respostas. No cuestionario CBCL empréganse usualmente 11 indicadores, que toman valores entre 0 e 100. Cada un deles calcúlase a partir dos valores dun grupo de respostas. Estes indicadores son: *anxious/depressed*, *withdrawn/depressed*, *somatic complaints*, *social problems*, *thought problems*, *attention problems*, *rule-breaking behavior*, *aggressive behavior*, *internalizing problems*, *externalizing problems* e *total* (este último indicador calcúlase a partir de tódalas respostas do cuestionario). Como os valores destes indicadores do cuestionario CBCL son numéricos, a súa predicción constitúe un problema de regresión. Empregáronse os cuestionarios de 72 participantes con idades entre 6 e 14 anos.

Para a predicción dos 11 grupos do cuestionario CBCL empregouse unha colección de 26 métodos de aprendizaxe automática para regresión implementados nas linguaxes de programación R, Octave, Matlab e Python. Estes regresores pertencen a unha ampla variedade de familias de algoritmos de regresión, incluindo: regresión linear (familia que inclúe a 4 regresores) e linear regularizada (4 regresores), *support vector regression* (3), *regression trees* (3), *ensembles* (9, incluindo *bagging*, *random forest*, *adaboost* e *gradient boosting machine*), redes neuronais *multi-layer perceptron* (1) e outros métodos como *Gaussian process regression* e *kernel ridge regression*. Estes modelos foron seleccionados polos seus bos resultados nunha exhaustiva comparativa desenvolvida nun traballo de investigación previo a esta tese doutoral. Avaliouse a fiabilidade da regresión usando o coeficiente de correlación linear R (que debería ser R=1 se os valores verdadeiro e predito coincidiran exactamente), *root mean squared error* (RMSE), *mean squared error* (MSE) e *weighted absolute percentage error* (WAPE). Probáronse tódalas combinacións das 11 saídas CBCL e os 6 grupos de entradas, e estudouse a influencia do xénero da persona participante na predicción.

Os resultados son moi desiguais dependendo da saída considerada. Isto era algo esperado, porque dacordo coa literatura algunhas combinacións de entradas SP2 e saídas CBCL non deberían estar relacionadas. Concretamente, a predicción máis fiable acadouse para a

saída *externalizing problems*, cun R moi perto do ideal (0.98), usando tódalas entradas SP2 (grupo *total*) e o método de regresión linear. O regresor *adaboost* acadou unha predicción relativamente fiábel para a saída *anxious/depressed* usando o grupo de entradas *avoiding*, e o regresor *gradient boosting machine* (gbm) acadou tamén un bo resultado para a saída *social problems* empregando a entrada *registration*, en ambos casos con R=0.72. O regresor gbm tamén acadou un bo R=0.68 para a saída *total* empregando a entrada *touch*. Outras saídas onde se acadou menor correlación, sendo aínda aceptábel, foron *withdrawn/depressed*, *thought problems*, *attention problems*, *aggressive behavior* e *internalizing*. As restantes saídas acadaron prediccións menos fiábeis ($R < 0.5$). O regresor que acadou a mellor posición promedio sobre tódalas saídas foi a regresión linear regularizada *ridge*, seguida pola *Gaussian process regression*, *gradient boosting machine* (*gbm*) e regresión linear regularizada *lasso*, aínda que deles so *gbm* acadou o mellor resultado nalgunha saída.

Dado que a fiabilidade da predicción é prácticamente igual incluindo ou non o xénero entre as entradas, a súa influencia non resultou significativa. A partires da literatura, esperabase que existisen relacións entre algunhas das entradas *seeking*, *avoiding* ou *touch* con algunhas das saídas *anxious/depressed*, *withdrawn/depressed*, *rule-breaking*, *aggressive behavior*, *internalizing* ou *externalizing*. A relación entre unha entrada e unha saída confírmase cando esa saída acada o mellor resultado (a predicción máis fiábel) con esa entrada. Nembargantes, de entre estas relacións esperadas somentes se confirmaron as da entrada *avoiding* coas saídas anteriores agás *externalizing problems*, que acadou o mellor resultado coa entrada *total*.

A predicción de cada indicador no cuestionario CBCL tamén se formulou en termos de clasificación entre participante normal ou que require un tratamento clínico, definindo así 2 clases (normal e clínico). Adicionalmente, tamén se consideraron 3 clases: participante normal, pre-clínico ou clínico. A clase determínase umbralizando o indicador CBCL correspondiente. Con 2 clases, o participante considérase normal cando o indicador é inferior a 60, e clínico no caso contrario. Con 3 clases, un participante normal ten un indicador inferior a 60; se está entre 60 e 70, considérase pre-clínico; e se está por riba de 70, considérase clínico. Neste caso, a calidade mídese coa estatística kappa, en %. A respeito da clasificación en 2 clases, a mellor saída, *externalizing*, acadou un kappa de 93.9% cun acierto do 97.2%, unha clasificación prácticamente perfecta sen falsos negativos e case sen falsos positivos. Outras saídas con bos resultados foron *anxious/depressed* (kappa=59.3%), *internalizing* (57%), *total* (47.1%), *withdrawn/depressed* (40.4%) e *thought problems* (39.7%). Considerando 3

clases, a lista inclúe: *externalizing* (kappa=91.9%), *internalizing* (42.4%), *total* (39.4%), *anxious/depressed* (35.7%) e *withdrawn/depressed* (35.1%).

Os resultados (regresión, clasificación en 2 e 3 clases) indican que para os seguintes indicadores do cuestionario CBCL (saídas) a predicción usando o grupo sensorial correspondente é fiábel, confirmando a relación entre ambos: *externalizing* e *total*; *anxious/depressed* e *avoiding*; *social problems* e *registration*; *total problems* e *touch*. Dado que o cuestionario SP2 cúbrese a idades más temperás que o CBCL, a metodología proposta permitiría deseñar intervencións clínicas e terapias personalizadas moito antes no tempo. Estes resultados coinciden con traballos previos que suxiren que as alteracións sensoriais, especialmente táctiles, provocan dificultades na resposta ao entorno e a perda de ocasións para aprender, que é a base de procesos máis complexos como a regulación emocional e as interaccións sociais. Os efectos en cascada destas dificultades maniféstanse na vida diaria das persoas afectadas a través de hiper-actividade, impulsividade, comportamentos estereotipados e repetitivos, e angustia emocional e social. Ademais, a sobrecarga que para estas persoas supón la xestión dos estímulos sensoriais condúceos con frecuencia a estados de ansiedade ou depresión.

A investigación futura contempla estender os métodos de proxección 2D propostos para superar as limitacións da análise discriminante linear en problemas de clasificación con moitas clases. Tamén se traballa na síntese de patróns para que a súa clase ou función a predecir tome un determinado valor, situando os patróns nas distintas rexións definidas polo mapa. Pola outra banda, preténdese estender o estudo sobre autismo a un meirande número de participantes, incluindo persoas adultas, para aumentar a validez das conclusións acadadas. Tamén se traballa no deseño de clasificadores que operen sobre respostas numéricas a cuestionarios, un tipo de datos amplamente empregado en eidos da Psicoloxía tais como trastorno obsesivo-compulsivo (TOC) e trastorno por déficit de atención e hiperactividade (TDAH).

Palabras chave: aprendizaxe automática, visualización 2D, clasificación, regresión, desorde de espectro autista, comportamento, procesamento sensorial.

Resumen en castellano

Esta tesis se sitúa en el ámbito del aprendizaje automático, que se podría definir como un conjunto de técnicas (también denominadas modelos) para la predicción de valores a partir de datos utilizando ejemplos. Estos ejemplos incluyen el valor a predecir (llamado normalmente salida del modelo) y los datos utilizados para efectuar esta predicción (entradas del modelo). Las entradas son vectores multi-dimensionales (patrones) compuestos por valores numéricos que pueden ser continuos o discretos. Si la salida a predecir es continua, se habla de un problema de regresión, y el modelo se llama regresor. Si el valor es discreto, hablamos de problema de clasificación y de clasificadores. Los ejemplos utilizados para construir el modelo de aprendizaje automático componen lo que usualmente se denomina "conjunto de entrenamiento".

La dimensionalidad del patrón es en general elevada, y esto dificulta entender la dependencia entre cada entrada y la salida, ya que cada entrada influye en la salida de forma distinta dependiendo de las otras entradas. Por esta razón, sería interesante disponer de un mapa bi-dimensional donde se representase, por ejemplo mediante un código de colores, la salida que se debería predecir en las distintas regiones del espacio de entrada (es decir, efectuar una predicción para cada uno de los posibles valores de los patrones). En los problemas de clasificación, este mapa sería similar a los mapas geográficos políticos, donde cada clase sería como un país en el mapa. Conociendo la localización de un patrón en este mapa podríamos predecir a qué clase pertenece. Además, podríamos saber qué clases están adyacentes o cercanas, en cuyo caso sería fácil confundir sus patrones porque son parecidos, o lejanas, de modo que fuese difícil confundir sus patrones porque son muy diferentes. Un mapa de este tipo permitiría también saber si existe superposición entre clases, y entre qué clases se produce esta superposición. Similarmente, en problemas de regresión este mapa sería similar a los mapas de temperaturas, o a los mapas topográficos de elevaciones del terreno, en los que se representa la temperatura con un código de colores, y nuevamente sabiendo la localización de un patrón obtendríamos el valor a predecir. Esto permitiría saber en qué direcciones crece o decrece este valor, con qué velocidad, y cuáles son las regiones con valores reducidos, medios o elevados.

La tesis formula algoritmos para la visualización bi-dimensional (en 2D) de problemas de clasificación y regresión. Estos algoritmos, denominados **ultra-fast 2D classifier** (UF2DC) y

ultra-fast 2D regressor (UF2DR), se han diseñado con el objetivo de realizar la clasificación o regresión sobre el propio espacio 2D. Para ello, se emplea la técnica clásica de análisis discriminante lineal (LDA) para proyectar los datos multi-dimensionales originales al espacio bi-dimensional. Se ha seleccionado esta técnica entre las muchas que existen para la reducción de dimensionalidad debido a sus buenos resultados para la clasificación en 2D, demostrada en trabajos de investigación anteriores, y por su elevada eficiencia computacional. Aunque el LDA se emplea para problemas de clasificación y reducción de la dimensionalidad, en esta tesis ampliamos su uso a problemas de regresión, discretizando la salida en niveles manejados como clases y utilizando LDA para su proyección al espacio bi-dimensional.

Los patrones de entrada proyectados al espacio 2D se utilizan para crear un mapa de clasificación o regresión. Se emplea el mismo rango de valores en las coordenadas horizontal y vertical de este espacio, definiendo así un mapa cuadrado. Este mapa se divide en un número de cuadrados (o píxeles, si consideramos el mapa como una imagen), cada uno de los cuales se asociará a un valor de salida que representa una etiqueta de clase en problemas de clasificación, o un valor numérico en problemas de regresión. De este modo, para predecir la salida ante un patrón de entrada multi-dimensional es suficiente con proyectar este patrón al espacio 2D, localizar el cuadrado del mapa al cual se proyectó este patrón y proporcionar como predicción el valor del mapa en ese cuadrado.

Para ejecutar este proceso, el primer paso consiste en seleccionar un número apropiado de cuadrados en base al tamaño del conjunto de entrenamiento. Porque, dependiendo de este tamaño, será necesario un mapa más o menos detallado, dicho en otras palabras, con más o menos resolución. Es importante destacar que en los métodos propuestos este número de cuadrados está acotado superiormente, lo cual resulta importante para garantizar su eficiencia en problemas con muchos patrones de entrenamiento. Considerando este número de cuadrados y el rango de valores de las coordenadas del mapa, es posible determinar los índices horizontal y vertical del cuadrado al que se ha proyectado un patrón de entrada para el cual se desea realizar una predicción. En este punto, el diseño de un procedimiento eficiente requiere ligeras diferencias entre los problemas de clasificación y regresión.

1. En problemas de **clasificación**, se define la clase asociada a un cuadrado como la más votada entre todos los patrones de entrenamiento que son proyectados a este cuadrado. Con este fin, se crea para cada clase una matriz de poblaciones que indica el número de estos patrones que se proyectaron a cada cuadrado. La predicción asociada a un cuadrado se define como la clase con mayor población en el mismo (es decir, se realiza

una votación). Como en general el número de cuadrados es mayor que el de patrones, existirán muchos cuadrados sin patrones en los que no se puede realizar esta votación. Para resolver este problema de forma eficiente, se extiende el ámbito espacial (usualmente denominado "máscara") sobre el cual se realiza la votación. Con este fin, se parte del cuadrado vacío original y se aumenta el tamaño de dicha máscara, siempre conservando una forma cuadrada, hasta incluir cuadrados con patrones.

2. En problemas de **regresión**, la votación es sustituida por un promedio de la salida, que es un valor numérico, sobre todos los patrones de entrenamiento que fueron proyectados a este cuadrado. Con este fin, se crean dos matrices. La matriz de salida contiene elementos cuyo valor es la suma de las salidas de todos los patrones de entrenamiento que se proyectaron al cuadrado asociado a ese elemento. La matriz de población contiene elementos cuyo valor es el número de estos patrones de entrenamiento proyectados a dicho cuadrado. Cada una de estas dos matrices contiene un valor para todos los cuadrados del mapa. El valor a predecir para un patrón proyectado a un cuadrado será el cociente entre sus elementos correspondientes en las matrices de salida y de población. Similarmente al caso de clasificación, este procedimiento no puede realizarse con cuadrados vacíos. Por ello, el promedio se extiende a una máscara centrada en el cuadrado considerado, con el tamaño mínimo necesario para incluir cuadrados a los que se proyectó algún patrón de entrenamiento.

La extensión de la máscara requiere efectuar una búsqueda que puede extenderse más o menos sobre los píxeles del cuadrado 2D al que se proyectan los patrones de entrenamiento. Este proceso puede resultar costoso computacionalmente. Por esta razón, la tesis incluye un procedimiento eficiente para poder emplear UF2DC y UF2DR en problemas de clasificación y regresión grandes, tanto en número de patrones de entrenamiento como de entradas o clases. El pseudocódigo de este procedimiento se incluye también en la tesis.

El mapa de clasificación se codifica en la matriz de mapa, donde cada elemento es la predicción (etiqueta de clase para problemas de clasificación, o valor de la función en problemas de regresión) asociada a su cuadrado correspondiente en el mapa. El tamaño de esta matriz es relativamente reducido para garantizar la eficiencia de los métodos propuestos, tanto en velocidad como en requerimientos de memoria RAM. Sin embargo, este tamaño resulta escaso para una visualización detallada del mapa de clasificación/regresión, que debe tener una resolución superior. Por ello, a partir de esta matriz se crea la matriz de imagen, de tamaño

mayor que la matriz de mapa, que codifica la imagen que se visualizará del mapa, mientras que la matriz de mapa se usará sólo para realizar la predicción. La matriz de imagen se crea: 1) aumentando la matriz de mapa y realizando un promediado con un núcleo de forma rectangular; y 2) realizando en la imagen un suavizado que aplica la función moda estadística por grupos de cuadrados sobre la imagen.

El trabajo experimental ilustra gráficamente el funcionamiento de UF2DC y UF2DR en problemas de dos y más dimensiones. En ellos se muestra su capacidad para elaborar representaciones espaciales separadas y coherentes de las distintas clases, o de los distintos valores de la función a predecir en problemas de regresión. Los experimentos de validación estadística realizan una comparativa de los métodos propuestos sobre 26 problemas de clasificación, algunos de ellos binarios, y 23 problemas de regresión. Todos estos problemas están considerados como de referencia en la literatura, están disponibles en los almacenes públicos de datos y son usados constantemente en el ámbito del aprendizaje automático. Se incluyen problemas de gran tamaño, hasta 5 millones de patrones y 719 entradas en clasificación, y hasta 4 millones y 482 entradas en regresión.

En clasificación, la comparativa incluye el algoritmo SVM (*support vector machine* con núcleo gausiano), aplicado en los patrones multi-dimensionales originales; LSVM (ídem usando un núcleo lineal) en los patrones originales; PNN (*probabilistic neural network*) sobre los patrones originales, bi-dimensionales y h-dimensionales, siendo h=número de clases menos 1; y WSVM (proyección de los datos originales a 2D usando una red neuronal RBF, *radial basis funcion*, diseñada para optimizar la clasificación de una SVM con núcleo lineal de acuerdo con la aproximación denominada *wrapper* en la literatura). Los clasificadores LSVM y PNN se incluyeron en la comparativa por su eficiencia computacional en problemas grandes. El acierto obtenido por UF2DC clasificando los patrones en 2D es similar a los obtenidos por los otros métodos usando los patrones originales en la mayoría de los problemas (20 de 26), siendo la diferencia apreciable sólo en los 6 problemas restantes. En cuanto a velocidad, UF2DC es entre 10 y 1000 veces más rápido que los otros clasificadores considerados, y la diferencia aumenta mucho con el tamaño de los datos. De hecho, en el problema más grande, con 5 millones de patrones, sólo UF2DC y LSVM son capaces de ejecutarse, obteniendo aciertos similares. Sin embargo, el primero tarda sólo 3 minutos mientras que el segundo tarda 30 horas, a pesar de estar diseñado específicamente para datos grandes. El trabajo experimental también evalúa el tiempo requerido por UF2DC para crear la matriz de imagen, tiempo que no supera el minuto en ninguno de los problemas de clasificación considerados.

La experimentación de UF2DR en problemas de regresión incluye una comparativa con SVR (*support vector regression* con núcleo gausiano), aplicado a los patrones originales; LSVR (ídem con núcleo lineal), aplicado a los patrones originales; y GRNN (*generalized regression neural network*), aplicada a los patrones originales, bi-dimensionales y *h*-dimensionales. Tanto LSVR como GRNN fueron incluidos por su velocidad en problemas grandes. La correlación obtenida por UF2DR con los patrones 2D es muy similar a la obtenida usando los restantes métodos en 19 de 23 problemas. En cuanto a la velocidad, el método propuesto se ejecuta en todos los problemas, y en el más grande (4.1 millones de patrones) tarda 2.3 minutos mientras LSVR necesita 53 minutos. También se evaluó la sensibilidad de la predicción con respecto al número de niveles usados para cuantificar la salida en los problemas de regresión. Los experimentos demuestran que esta sensibilidad no es excesivamente elevada, y que el número de niveles (10) utilizado en la validación experimental es una elección adecuada que proporciona unos resultados muy cercanos a los óptimos.

Hay que destacar que UF2DC y UF2DR, además de ejecutar la predicción en 2D con fiabilidades similares a otras aproximaciones punteras en la literatura, permiten visualizar la estructura de los datos en problemas de clasificación y regresión, proporcionando explicaciones a las características y propiedades de estos problemas. Como se comentó anteriormente, estas propiedades incluyen, en problemas de clasificación, superposición y relaciones de cercanía o lejanía entre clases, y en problemas de regresión, regiones de valores bajos, medios y altos, y direcciones de incremento y decremento de la función a predecir.

La segunda parte de la tesis aborda la predicción automática de comportamientos en base a información sensorial para el tratamiento del desorden de espectro autista en la infancia y la adolescencia. Esta investigación fue realizada en colaboración con el Grupo de Medicina Xenómica[2] de la Universidad de Santiago de Compostela. Las personas que experimentan desorden de espectro autista se caracterizan tanto por sufrir problemas de procesamiento sensorial del entorno (interés inusual, bien hiper- o hipo-reactivo, en estímulos visuales, auditivos, olfativos o táctiles) como por problemas de comportamiento (aislamiento y déficit de interacción social, problemas en la comunicación no verbal, comportamientos repetitivos y restringidos, irritabilidad, ansiedad y depresión). Las alteraciones sensoriales se detectan a edades mucho más tempranas, incluso en los primeros meses de vida, que los problemas de comportamiento, los cuales se manifiestan a edades posteriores e incluso en la adolescencia,

[2]http://www.xenomica.eu

teniendo un gran impacto en el desarrollo personal y las relaciones sociales de la persona afectada.

La literatura ha demostrado la existencia de relaciones entre alteraciones sensoriales y problemas de comportamiento. Ambos se analizan utilizando sendos cuestionarios: *sensory profile-2* (SP2) y *child behavior check-list* (CBCL), que proporcionan medidas estandarizadas del desarrollo personal y guías para intervenciones clínicas. Los métodos de aprendizaje automático aportan interesantes posibilidades para el estudio de las relaciones entre ambos y llegar así a saber en qué medida uno de ellos es útil para predecir el otro. Específicamente, en esta tesis el objetivo es usar las respuestas del cuestionario SP2, que evalúa problemas sensoriales, para predecir las respuestas en el cuestionario CBCL, que evalúa problemas de comportamiento. En el cuestionario SP2 se emplean normalmente 6 grupos de respuestas, cada una de ellas con valores entre 0 y 5, denominados en inglés *avoiding*, *registration*, *seeking*, *sensitivity*, *total* and *touch*. Cada grupo incluye una serie de respuestas del cuestionario, y *total* incluye todas las respuestas. En el cuestionario CBCL se emplean usualmente 11 indicadores, que toman valores entre 0 y 100. Cada uno de ellos se calcula a partir de los valores de un grupo de respuestas. Estos indicadores son: *anxious/depressed*, *withdrawn/depressed*, *somatic complaints*, *social problems*, *thought problems*, *attention problems*, *rule-breaking behavior*, *aggressive behavior*, *internalizing problems*, *externalizing problems* y *total* (este último indicador se calcula a partir de todas las respuestas del cuestionario). Como los valores de estos indicadores del cuestionario CBCL son numéricos, la predicción de los mismos constituye un problema de regresión. Se emplearon cuestionarios de 72 participantes con edades entre 6 y 14 años.

Para la predicción de los 11 grupos del cuestionario CBCL se empleó una colección de 26 métodos de aprendizaje automático para regresión implementados en los lenguajes de programación R, Octave, Matlab y Python. Estos regresores pertenecen a una amplia variedad de familias de algoritmos de regresión, incluyendo: regresión lineal (4 regresores) y lineal regularizada (4), *support vector regression* (3), *regression trees* (3), *ensembles* (9, incluyendo *bagging*, *random forest*, *adaboost* y *gradient boosting machine*), redes neuronales *multi-layer perceptron* (1) y otros métodos como *Gaussian process regression* y *kernel ridge regression*. Estos modelos fueron seleccionados por sus buenos resultados en una exhaustiva comparativa desarrollada en un trabajo de investigación previo a esta tesis doctoral. La fiabilidad de la regresión se evaluó usando el coeficiente de correlación lineal R (que debería ser R=1 si los valores verdadero y predicho coincidieran exactamente), *root mean squared error* (RMSE),

mean squared error (MSE) y *weighted absolute percentage error* (WAPE). Se probaron todas las combinaciones de las 11 salidas CBCL y los 6 grupos de entradas, y se estudió la influencia del género de la persona participante en la predicción.

Los resultados son muy desiguales dependiendo de la salida considerada. Esto era algo esperado, porque de acuerdo con la literatura algunas combinaciones de entradas SP2 y salidas CBCL no deberían estar relacionadas. Concretamente, la predicción más fiable se consiguió para la salida *externalizing problems*, con un R muy cercano al ideal (0.98), usando todas las entradas SP2 (grupo *total*) y el método de regresión lineal. El regresor *adaboost* consiguió una predicción relativamente fiable para la salida *anxious/depressed* con la entrada *avoiding*, y el regresor *gradient boosting machine* (gbm) obtuvo también un buen resultado para la salida *social problems* utilizando la entrada *registration*, en ambos casos con R=0.72. El regresor gbm también obtuvo un buen R=0.68 para la salida *total* empleando la entrada *touch*. Otras salidas para las cuales se obtuvo una menor correlación, pero todavía aceptable, fueron *withdrawn/depressed*, *thought problems*, *attention problems*, *aggressive behavior* e *internalizing*. Las restantes salidas obtuvieron predicciones menos fiables ($R < 0.5$). El regresor que obtuvo la mejor posición promedio sobre todas las salidas fue la regresión lineal regularizada *ridge*, seguida por *Gaussian process regression*, *gradient boosting machine* (*gbm*) y regresión linear regularizada *lasso*, aunque de ellos sólo *gbm* obtuvo el mejor resultado en alguna salida.

Dado que fialibilidad de la predicción es prácticamente igual incluyendo o no el género entre las entradas, su influencia no resultó significativa. A partir de la literatura, se esperaba que existiesen relaciones entre algunas de las entradas *seeking*, *avoiding* o *touch* con algunas de las salidas *anxious/depressed*, *withdrawn/depressed*, *rule-breaking*, *aggressive behavior*, *internalizing* o *externalizing*. La relación entre una entrada y una salida se confirma cuando esa salida obtiene el mejor resultado (la predicción más fiable) con esa entrada. Sin embargo, de entre estas relaciones esperadas sólo se confirmaron las de *avoiding* con las salidas anteriores excepto *externalizing problems*, cuyo mejor resultado fue usando la entrada *total*.

La predicción de cada indicador en el cuestionario CBCL también se formuló en términos de clasificación entre participante normal o que requiere un tratamiento clínico, definiendo así 2 clases (normal y clínico). Adicionalmente, también se consideraron 3 clases: participante normal, pre-clínico o clínico. La clase se determina umbralizando el indicador CBCL correspondiente. Con 2 clases, el participante se considera normal cuando el indicador es inferior a 60, y clínico en caso contrario. Con 3 clases, un participante normal tiene un indicador inferior a 60; si está entre 60 y 70, se considera pre-clínico; y si está por encima de 70, se considera

clínico. En este caso, la calidad se mide con la estadística kappa, en %. Respecto a la clasificación en 2 clases, la mejor salida, *externalizing*, obtuvo un kappa de 93.9% con un acierto del 97.2%, una clasificación prácticamente perfecta sin falsos negativos y casi sin falsos positivos. Otras salidas con buenos resultados fueron *anxious/depressed* (kappa=59.3%), *internalizing* (57%), *total* (47.1%), *withdrawn/depressed* (40.4%) y *thought problems* (39.7%). Considerando 3 clases, la lista incluye: *externalizing* (kappa=91.9%), *internalizing* (42.4%), *total* (39.4%), *anxious/depressed* (35.7%) y *withdrawn/depressed* (35.1%).

Los resultados (regresión, clasificación en 2 y 3 clases) indican que para los siguientes indicadores del cuestionario CBCL (salidas) la predicción usando el grupo sensorial correspondiente es fiable, confirmando la relación entre ambos: *externalizing* y *total*; *anxious/depressed* y *avoiding*; *social problems* y *registration*; *total problems* y *touch*. Dado que el cuestionario SP2 se cubre a edades más tempranas que el CBCL, la metodología propuesta permitiría diseñar intervenciones clínicas y terapias personalizadas mucho antes en el tiempo. Estos resultados coinciden con trabajos previos que sugieren que las alteraciones sensoriales, especialmente táctiles, provocan dificultades en la respuesta al entorno y la pérdida de ocasiones para aprender, que es la base de procesos más complejos como la regulación emocional y las interacciones sociales. Los efectos en cascada de estas dificultades se manifiestan en la vida diaria de las personas afectadas a través de hiper-actividad, impulsividad, comportamientos estereotipados y repetitivos, y angustia emocional y social. Además, la sobrecarga que para estas personas supone la gestión de los estímulos sensoriales los conduce con frecuencia a estados de ansiedad o depresión.

La investigación futura contempla extender los métodos de proyección 2D propuestos para superar las limitaciones del análisis discriminante lineal en problemas de clasificación con muchas clases. También se trabaja en la síntesis de patrones para que su clase o función a predecir tome un determinado valor, situando los patrones en las distintas regiones definidas por el mapa. Por otra parte, se pretende extender el estudio sobre autismo a un mayor número de participantes, incluyendo personas adultas, para aumentar la validez de las conclusiones obtenidas. También se trabaja en el diseño de clasificadores que operen sobre respuestas numéricas a cuestionarios, que son ampliamente utilizados en campos de la Psicología tales como trastorno obsesivo-compulsivo (TOC) y trastorno por déficit de atención e hiperactividad (TDAH).

Palabras clave: aprendizaje automático, visualización 2D, clasificación, regresión, desorden de espectro autista, comportamiento, procesamiento sensorial.

Contents

CHAPTER 1

OBJECTIVES

The current PhD. Thesis has been developed in the scope of machine learning (ML), dealing specifically with the automatic prediction of certain pre-defined values. In order to perform this prediction, the ML algorithms use a collection of data examples (named training set) for which both the input data (used to perform the prediction) and the true (or desired) values to be predicted, are available. The examples are expected to be a set complete enough to include all the possible situations, so that the prediction is still reliable for new data not using in the training set. The prediction can be performed for discrete or categorical values, leading to a "classification" problem, or for continuous numeric values, leading to a "regression" problem.

The first objective of this Thesis is to design and evaluate methods for the visualization of classification and regression problems. This visualization is expected to provide a visual map of the dataset that has been learnt by the ML algorithm. This map should allow a certain "understanding" of the problem at hand, both in terms of regions associated to each class label, for classification problems, and in terms of regions with low, middle and high values of the continuous function to be predicted, in regression problems. In order to achieve this objective, two methods are proposed in this Thesis, named "ultra-fast 2D classifier" (UF2DC) and "ultra-fast 2D regressor", for classification and regression problems, respectively. These methods have been designed to map the original high-dimensional data to the 2D space and to perform the automatic prediction in this space, providing a "map" of the classification or regression problem that allows to explain the available data and the relations between the input data and the different class labels or numeric values to be predicted. Both algorithms UF2DC and UF2DR are very fast both in the creation of the map and on the classification of new

data, not seen during training. The proposed methods have been validated on classification and regression problems with very large data. The results prove that they provide intuitive explanations of the data structure and its relation to the values to be predicted. Besides, its performance in the automatic prediction is very near to ML methods performing on the original high-dimensional dataset. Finally, the methods are very fast both for small and large data.

The second objective is to apply regression ML models for the automatic prediction of behavior problems in the context of autism spectrum disorder (ASD) in children and teenagers. The prediction is performed using as input data the numeric answers to questions in a questionnaire that evaluates sensory abnormalities. Analogously, the values predicted are the answers to questions in another questionnaire about behavior problems. The regressors are trained to predict behavior problems, such as anxiety/depression, somatic complaints, social problems, aggressive behavior and attention problems, among others. The prediction is performed using sensory abnormalities in touch processing, search of exciting sensory experiences, hyper- or hypo-sensitivity to sensory inputs compared to normal individuals, and avoidance of sensory experiences. This research work was performed in collaboration with the Group of Genomic Medicine of the University of Santiago de Compostela. The objective is to predict automatically behavior problems, that are evident at higher ages, from sensory information, that are registered at much lower ages. This might allow to raise early alerts, and apply treatments in order to reduce the impact of ASD in children's life. We used a large and variated collection of ML regressors to perform this prediction. The statistical results prove that a good reliability in prediction is achieved for some behavior problems using certain sensory data, defining clear relations between certain sensory abnormalities and behavior problems. In the remaining cases, the relations are not so definite, and the prediction is much less reliable.

CHAPTER 2

INTRODUCTION

The following sections 2.1 and 2.2 introduce the objectives of this Thesis. The materials, methods and experimental work for each objective will be developed in the following chapters of the thesis.

2.1 Dimensionality reduction and data visualization

Machine learning (ML) encloses a set of techniques that allow to model a system by reproducing its behavior in some sense. This reproduction is performed by predicting a value, named "output", that is of interest to identify the system, and that for some reason can not be, or it is difficult to be, measured directly. Thus, it is appealing to estimate this output indirectly. The prediction is issued by using other values, named "inputs", from the system, that are indead available, or at least that are more easy to acquire than the output. Both terms "inputs" and "output" get their names because they are the input data and output result, respectively, for the ML model that performs the prediction. The objective of this model is to learn how to synthesize the output using the inputs. In order to perform this task, the model has two resources:

1. A collection of examples, named "training set", composed by the values of inputs and output in several examples or cases. This collection should be large and representative enough to describe the system behavior. The hypothesis is that it is possible to learn how to predict the output from the inputs using only the relation between the inputs and

the output values in this collection. Thus, the collection is used to create the ML model in order to perform this prediction.

2. A mathematical method to perform the prediction of the output, that is a number, as a calculation that uses the inputs. This method is different for different ML models, and is named "training algorithm".

The prediction issued by the ML model will be in general different from the true output values. This difference is expected to be low enough to consider the prediction as reliable or accurate. The model performance is inversely related to this difference: the larger difference, the poorer performance. The difference must be calculated using examples not included in the training set. Otherwise, the model might be just "memorizing" the true output that corresponds to the inputs of each example, so the difference on the training set would be very low or even zero. However, the difference would probably be much higher for examples not seen during training. This phenomenon is known as "overtraining", and leads to an optimistic biasing in the measurement of the model performance. The set of examples used to measure the performance is named as "test set", and can not include training examples.

The output to be predicted can have discrete or continuous values. The first case corresponds to classification, where the model must predict a class, that is a discrete value, for each example. In some classification problems, the classes can be "sorted" in some way, so there is a relation of order between them, and they are named "ordinal classification" problems. The remaining classification problems are known as "nominal classification". The prediction of continuous values is known as regression, and in these cases there is always an order relation between the values to be predicted. Classification and regression are important fields in machine learning, although the former has received more interest from the research community because very often what we need is to discriminate between two or more cases, instead of predict a numerical value. In fact, we can always quantify the continuous output of a regression problem into a set of discrete levels, achieving an ordinal classification problem.

The ML models are considered as black-box methods because they perform a prediction (class or number), but most of them do not explain nor justify why that prediction is issued. In the ML framework, a classification or regression problem can be formulated as the synthesis of a function, the output, that gives discrete or continuous values and uses several variables (inputs), i.e., operates on the high-dimensional input space. Thus, each example is a vector in this space, named "pattern" containing the input values. The function is unknown, excepting

for its values on a collection of points in the high-dimensional space, that are known and correspond to the inputs of the training examples.

In classification tasks, the function gives a class label, and it is expected that each class is related to a certain region of the input space (i.e., the patterns from that region belong to that class). In regression tasks, it is also expected that the higher or lower values of the continuous output can be assigned to specific regions of this space. In fact, if the patterns of several classes or function values were completely mixed, these expectations would not fulfilled, and we can expect that no ML method would be able to perform a reliable prediction. In such case, to achieve a reliable prediction would require to include additional inputs, with different values for each class or function value. In other words, the available inputs would not be enough for a ML method to learn the problem.

Therefore, it is expected that both prediction tasks (classification and regression) allow a geometrical representation where the different output values are assigned to different regions of the high-dimensional input space. This representation, that must be performed in low-dimensional spaces, is often used to visualize data in classification or regression problems, so visualization methods have become a popular tool to explain and justify the machine learning models trained on these data.

Visualization of classification and regression problems has also been tightly related in the literature to dimensionality reduction (DR). The objective of DR methods is to transform or map the data from the original high-dimensional space where the data lie, into a transformed space with lower dimensionality while conserving most of the intrinsic information that are present inside the data. In fact, most DR methods use some criterion defining the information to be kept under the mapping, e.g., distances between patterns, so the mapped data optimize the difference between this information on the original high-dimensional space and on the transformed low-dimensional space.

Unsupervised DR methods are oriented to discover and visualize low-dimensional structures in high-dimensional unlabeled data [52], and then map the data into this low-dimensional space while preserving the data structure. These methods include principal component analysis (PCA) [86], Sammon mapping [44], kernel PCA [2], multi-dimensional data scaling (MDS) [58], stochastic neighbor embedding (SNE) [40], locally linear embedding (LLE) [49] and Isomap [63], among others. There are also supervised DR methods which have been applied to classification, such as the classical linear discriminant analysis (LDA) [4], neighborhood component analysis (NCA) [55], large margin nearest neighbor (LMNN) met-

ric learning [60], supervised LLE [87], supervised Isomap [33] and dubbed multi-manifold discriminant Isomap (MMD-Isomap) [84]. Recent approaches include manifold learning [77], canonical kernel DR [71], feature unionization [42], joint dictionary learning [83] and probabilistic structure learning DR [80].

Dimensionality reduction has also been used for data visualization in 2D, usually oriented to explain the classification problems in terms of class distributions, and of borders and overlapping among classes. When DR is combined to classification, the classical "filter" approach, that consists of training a classifier using the low-dimensional patterns, is often followed. The alternative "wrapper" strategy, that designs the DR method to optimize in the low-dimension space the performance of some classifier, has also been used with ML methods such as support vector machines [81], neural networks [70], genetic algorithms [69] or nearest neighbor classifiers [79], among others. Some visualization methods have also been applied to regression, such as supervised PCA [7] and LMNN [5].

The previous study [3] developed a comparison of 34 DR methods, including most of the previous approaches, in the 2D visualization on a collection of 71 classification datasets. The visualization was performed over two dimensions in order to display the classification problem for human inspection and understanding, which is more difficult in one or three, and not possible with more, dimensions. This study used 2D classification maps defined by a Gaussian kernel support vector classifier (GSVC), trained on the 2D mapped patterns using the filter approach, in order to visualize the classification problem. The agreement between the original high-dimensional and the 2D classification problems was measured by the performance of a GSVC on both problems.

In this research, the LDA-based methods achieved the highest performances and lowest times, although below the performance levels obtained using the original high-dimensional data. As well, the previous study [3] was limited to classification. Thus, in the current PhD. Thesis we pursue the following objectives:

1. Extend the LDA mapping to regression problems and use the 2D map to explain and visualize the landscape of the function to be predicted.

2. Propose a method to define the 2D classification or regression map without the need of training any standard ML classifier or regressor, such as GSVC in [3].

3. Use this map to perform predictions in the same way as any other ML method.

4. Provide a visual explanation about the structure and geometry of the data in:

a) Classification problems: relative positions of classes, class overlapping, shape of the borders between classes, neighboring and far classes.

b) Regression problems: regions with low, medium and high values of the function to predict, directions and magnitudes of changes in function values.

5. Develop the proposed methods in an efficient way in order to visualize and predict almost instantaneously any large-scale classification or regression problem.

The methods that are proposed in the current PhD. Thesis, called **ultra-fast 2D classifier** (UF2DC) for classification and **ultra-fast 2D regressor** (UF2DR) for regression, will be described in chapter 3 and evaluated experimentally in chapter 4.

2.2 Behavior prediction in autism spectrum disorder

The autism spectrum disorder (ASD) is a neurodevelopmental condition with a consistently high prevalence worldwide [16] that poses a serious burden to the society and affected families. An early diagnosis of ASD is crucial for implementing interventional approaches on individuals with this disorder [17]. Multiple risk factors contribute to the ASD phenotype, including genetic, biological, psycho-social and environmental contributors [59, 21]. This disorder is characterized by deficits in social interactions, communication and restricted/repetitive behavior, including sensory processing problems [6].

Impairments in sensory processing are described as unusual interests in sensory aspects of the environment (e.g., visual, auditory, or tactile stimuli) to which individuals with ASD frequently display atypical responses [76]. These responses can be organized and classified as hyper-reactivity (tendency to respond at lower intensity thresholds quicker and more intensely or longer) or hypo-reactivity (tendency to respond with "indifference", unawareness, or slowly) to sensory inputs [72] that seem to be underpinned by specific neurophysiological markers (see [53] for a detailed review).

Atypical processing patterns have been observed in ASD across all sensory modalities, including visual, auditory, olfactory, proprioceptive, somatosensory, or interoceptive stimulation and multisensory integration, regardless of age and symptom's severity in children and adolescents [9]. Sensory alterations have an early onset and are one of the first signs of ASD, as early as observed in the first months of life [41]. In consequence, these alterations can

impact behavior and social functioning of children and adolescents and may be at the root of social deficits during development [74].

Indeed, altered sensory responsivity in ASD cascades into social and behavioral impairments [74]. The relationship between hypo-responsiveness and hyper-responsiveness to sensory stimuli and maladaptive behavior has been documented, with atypical processing of stimuli being correlated with social, cognitive and communicative impairments [43] and the presence of repetitive and restricted interests and behaviors [30]. Sensory abnormalities were also linked to isolation, reactivity to change, disinterest and indifference, self-aggression, irritability, or emotional lability [36]. Additionally, altered sensory processing has been related to anxiety [78] and depressive [10] states in children and adolescents with ASD, with a great impact on their adaptive behaviors [75, 47, 85]. One of the earliest and most common sensory alterations described in ASD is related to abnormalities in tactile processing, for example, food texture [57].

Tactile contact is considered a precursor for the development of social and communication abilities, and impaired touch processing has been linked to emotional and social distress early in life, as it imposes limits on environmental learning opportunities [57]. Moreover, evidence suggests that increased difficulties in touch processing are associated with behavioral impairments (e.g., difficulties in inhibitory control) in children and adolescents with ASD [62, 61]. In addition, altered touch processing was related to increased social and communication deficits [30, 56], to non-verbal communication impairments and repetitive behaviors [30], as well as anxious/depressed states and repetitive/obsessive behavior [29]. Therefore, it is important to further clarify the relationship between sensory processing and behavior in children and adolescents with ASD, by exploring to what extent sensory processing is predictive of behavioral outcomes in this population.

The Sensory Profile 2, SP2 [25] and the Child Behavior Check-List, CBCL [1], are two widely used tools to measure sensory and behavioral competencies, respectively. These questionnaires, usually filled out by parents, provide standardized measures of child's development and offer guidance for future clinical interventions. In line with the relationship between sensory and behavioral characteristics previously discussed, the scores of different SP2 items and CBCL subscales have already shown to be correlated [56]. New tools– like machine learning (ML) techniques– could offer novel insights in how constructs measured by both questionnaires are intertwined and how they are useful to predict each other. Some works in the literature showed the interest of using ML in the study of autism to analyze continuous/dimensional

or categorical/qualitative variation between and within individuals [50]. In addition, ML can offer improved diagnostic timing, precision and quality, allowing clinicians to provide more robust diagnosis and intervention programs [73], as well as providing evidence on possible altered processes (e.g., reactivity to sensorial stimuli) for implementing early personalized care therapies and strategies. Machine learning models have provided satisfactory solutions to medical and non-medical applications due to its ability to extract information and make predictions from large datasets [13].

SP2 quadrant	CBCL outcome	
Seeking	Anxious/depressed	Aggressive behavior
Avoiding	Withdraw/depressed	Internalizing
Touch	Rule-breaking	Externalizing

Table 2.1: Relations that are expected between SP2 quadrants and CBCL outcomes.

In this thesis, the objective is to use machine learning methods to evaluate the relation between the sensory and behavioral domains in ASD participants. This research was developed in collaboration with the Group of Genomic Medicine[1] of the University of Santiago de Compostela. Specifically, a wide collection of supervised ML algorithms, described in chapter 5 of this Thesis, was applied including: multi-variate linear an regularized linear regression, support vector machine, gradient boosting machine and regression tree-based ensembles such as gradient boosting machine and random forest, among others. With these algorithms, we sought to examine how SP2 quadrant scores (seeking, avoiding, sensitivity and registration) and touch processing (given the evidence suggesting that tactile impairments are related to the onset of ASD and being the most common sensory alteration in this population) are predictive of CBCL outcomes: anxious/depressed, withdrawn/depressed, somatic complaints, social problems, thought problems, attention problems, rule-breaking behavior, aggressive behavior and the two empirically derived internalizing and externalizing broadband scales. It is expected that hyper-responsive and hypo-responsive sensory experiences assessed using the SP2, i.e., items corresponding to seeking and avoiding quadrants, as well as touch processing, can be used to predict ASD behavioral problems such as anxious/depressed, withdrawn/depressed, rule-breaking behavior and aggressive behavior, as well as in internalizing and externalizing domains (see Table 2.1). Our results, in chapter 6 of this Thesis, will allow us to relate SP2 and CBCL questionnaires, in order to know which groups of items in SP2

[1] http://www.xenomica.eu

can be most useful in predicting CBCL scores in other categories. These results can help to classify different phenotypes in ASD and relate alterations in sensory mechanisms with impairments in behavioral manifestations.

CHAPTER 3

TWO-DIMENSIONAL CLASSIFICATION AND REGRESSION

The proposed methodology first describes the 2D mapping using the linear discriminant analysis (LDA, section 3.1). Then, it introduces the 2D map (section 3.2) and describes how the map is defined in UF2DC and UF2DR for classification and regression (sections 3.3 and 3.4), respectively. Finally, section 3.5 summarizes the pseudo-code of the proposed methods.

3.1 Projection of the high-dimensional patterns in 2D

Let $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})$, with $i = 1, \ldots, N$, be the original high-dimensional training patterns (row vectors), being p the number of inputs and N the number of training patterns (dataset population). In classification problems, let C be the number of classes, $c_i \in \{1, \ldots, C\}$ the true class label of $\mathbf{x}_i$ and N_j, with $j = 1, \ldots, C$, the population of class j. In regression problems, let o_i be the true output for $\mathbf{x}_i$, where o_i is a continuous value. According to the theory of LDA, three matrices are defined:

1. The total covariance matrix $\mathbf{T}$ (of size $p \times p$):

$$\mathbf{T} = \frac{(\mathbf{X} - \bar{\mathbf{X}})^T (\mathbf{X} - \bar{\mathbf{X}})}{N - 1} \tag{3.1}$$

   where the matrix $\mathbf{X}$, of size $N \times p$, contains the training patterns $\mathbf{x}_i$, with $i = 1, \ldots, N$, by rows. The matrix $\bar{\mathbf{X}}$, of the same size as $\mathbf{X}$, contains the vector $\bar{\mathbf{x}}$ (mean over all the patterns) repeated in each row:

$$\bar{\mathbf{x}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i \tag{3.2}$$

2. The matrix $\mathbf{W}$, also of size $p \times p$, is calculated by averaging the within-class covariance matrices of the C classes:

$$\mathbf{W} = \frac{1}{C}\sum_{j=1}^{C}\frac{\left(\mathbf{X}_j - \bar{\mathbf{X}}_j\right)^T\left(\mathbf{X}_j - \bar{\mathbf{X}}_j\right)}{N_j - 1} \tag{3.3}$$

   Both $\mathbf{X}_j$ and $\bar{\mathbf{X}}_j$ are matrices of size $N_j \times p$. The matrix $\mathbf{X}_j$ contains, by rows, the N_j training patterns $\mathbf{x}_i$ of class j, thus being $i \in \mathscr{A}_j$ with $\mathscr{A}_j = \{i = 1,\ldots,N : c_i = j\}$. Each row of matrix $\bar{\mathbf{X}}_j$ is equal to $\bar{\mathbf{x}}_j$ (mean over patterns of class j):

$$\bar{\mathbf{x}}_j = \frac{1}{N_j}\sum_{i\in\mathscr{A}_j}\mathbf{x}_i \tag{3.4}$$

3. The between-class covariance matrix $\mathbf{B}$, of size $p \times p$, is the difference between the total covariance matrix $\mathbf{T}$ and the within-class covariance matrix $\mathbf{W}$:

$$\mathbf{B} = \mathbf{T} - \mathbf{W} \tag{3.5}$$

Once the matrices $\mathbf{W}$ and $\mathbf{B}$ are calculated, the matrix $\mathbf{A}$ of size $2 \times p$, which defines the mapping from $\mathbb{R}^p$ (original high-dimensional space) to $\mathbb{R}^2$ (two-dimensional space), is composed by the two (row) eigenvectors associated to the largest eigenvalues (henceforth named "two main eigenvectors") of the matrix product:

$$(\mathbf{W} + \alpha\mathbf{I}_p)^{-1}\mathbf{B} \tag{3.6}$$

where $\mathbf{I}_p$ is the identity matrix of size p and $\alpha = 0.001$ regularizes the matrix $\mathbf{W}$, that is often singular. Finally, LDA maps linearly the original pattern $\mathbf{x}$ into the 2D pattern:

$$\mathbf{y} = \mathbf{A}(\mathbf{x} - \bar{\mathbf{x}}) \tag{3.7}$$

where $\bar{\mathbf{x}}$ is defined in eq. 3.2. In regression problems, where the continuous output o_i replaces the class label c_i for a training pattern $\mathbf{x}_i$, the previous methodology can also be used by replacing the classes by levels of output values, i.e., by quantifying the output values in a set of levels. The number $\mathscr{L}$ of output levels should not be very low, because a certain number

of levels are required to quantify ouput values, nor very high, because many within-level covariance matrices $\mathbf{X}_j$ and $\bar{\mathbf{X}}_j$ (eq. 3.3) should be calculated. A reasonable value might be $\mathscr{L} = 10$, which provides a trade-off between ability to quantify the output values and excessive number of levels. Note that $\mathscr{L}$ is not the number of possible output values given by UF2DR (see section 3.4), but the number of classes or output levels used to define the LDA mapping in regression problems, so that in these problems $C = \mathscr{L}$. The number l of patterns per level is calculated as $l = \lfloor N/\mathscr{L} \rfloor$, where $\lfloor \cdot \rfloor$ is the integer `floor` function. Sorting the true output values $o_1, \ldots, o_N$ and denoting by $\sigma_i \in \{1, \ldots, N\}$ the position of pattern $\mathbf{x}_i$ according to this sorting, a class label $c_i = \max(1, \lfloor \sigma_i / l \rfloor) \in \{1, \ldots, \mathscr{L}\}$ is assigned to pattern $\mathbf{x}_i$, with $i = 1, \ldots, N$, and LDA can be applied analogously to the classification case, achieving an ordinal classification problem.

In the literature, LDA is usually applied for classification by taking the $C-1$ main eigenvectors and mapping the high-dimensional patterns into the $(C-1)$-dimensional space generated by them. Using LDA for visualization, only the two main eigenvectors are used. Obviously, when information is transformed from $p > 2$ dimensions into a 2D space some information might be lost, so the performance is expected to be lower on the two-dimensional mapped dataset than on the original high-dimensional dataset. To evaluate the impact of this loss of information in the classification and regression performance is one of the objectives of the current work. Note that the proposed method is designed for 2D data visualization, classification and regression, and is not specifically oriented to dimensionality reduction.

On the other hand, the use of LDA for regression problems through categorization of the value to predict may seem an artificial approach, given that a plethora of methods do exist specifically for regression. However, this categorization allows us to extend the LDA mapping, and the methods proposed to develop a 2D map, for regression problems in an efficient way. The experimental work (section 4.2) will perform an empirical evaluation of this efficiency and the performance loss compared to existing regression methods working on the original p-dimensional data.

3.2 Definition of the 2D map

Once the 2D training patterns are projected from $\mathbb{R}^p$ to $\mathbb{R}^2$ and standardized (i.e., scaled and translated to have zero mean and standard deviation one in each dimension), the proposed methods UF2DC or UF2DR create a map, defined by the square matrix $\mathbf{M}$, of the squared

region $\mathscr{R}$ of $\mathbb{R}^2$ where the mapped patterns are located. Let us consider the range of values of $\mathscr{R}$ in each dimension as divided in s equal-length intervals, in such a way that $\mathscr{R}$ is divided in s^2 squares (this squares will be pixels in the image map, see below). The matrix $\mathbf{M}$, that defines the map of region $\mathscr{R}$, must be of size $s \times s$, being s referred as "the map size". The element M_{nm}, with $n,m = 1,\ldots,s$, is the output value associated to the n,m-th square in $\mathscr{R}$. Thus, a new p-dimensional test pattern $\mathbf{x}_t$ can be projected to a 2D pattern $\mathbf{y}_t = \mathbf{A}(\mathbf{x}_t - \bar{\mathbf{x}})$, where $\bar{\mathbf{x}}$ is defined by eq. 3.2 as the mean of the N training patterns. Then, the output z_t predicted by UF2DC or UF2DR for $\mathbf{y}_t$ will be $z_t = M_{nm}$, where n and m are the indices of the square where $\mathbf{y}_t$ is located.

In the previous work [3], the 2D patterns mapped by LDA are used to train a GSVC which afterwards classifies a 2D pattern for each square in $\mathscr{R}$. This procedure, based on the "filter" approach, is very intuitive and easy to understand, but it has several drawbacks:

1. The classification map is conditioned by the classifier, so different classifiers will define different maps. However, a 2D map of a classification problem should be intrinsic to the problem, i.e., unique, and should not be different for each, or depend on the, classifier used.

2. Some classifiers might be limited defining complex 2D maps. An example is random forest: despite being a state-of-the-art classifier, it splits binarily each input dimension, so in 2D it only would learn class borders parallel to both axes. This is a limiting factor to define complex geometries for classification or regression. Other ML methods might exhibit other restrictions.

3. Despite of training with 2D patterns, the classifier training and testing may be slow with large and even with small datasets, because it must be tested on all the s^2 squares of the map to achieve their output values. Specifically, the GSVC training in 2D may be slow when the high-dimensional classification problem is already difficult or when the patterns mapped to 2D exhibit high overlapping among classes.

As a consequence of the previous drawbacks, it would be useful to define a fast method that allows to define 2D maps using only the mapped training patterns without train any standard ML classifier. Such a method is proposed in sections 3.3 and 3.4 for classification and regression, respectively.

3.3 Classification map in 2D with UF2DC

In this section we will describe the ultra-fast 2D classifier, whose pseudo-code is listed in section 3.5. Let $\mathbf{y}_i = (y_{i1}, y_{i2})$, with $i = 1, \ldots, N$, be the standardized 2D mapped training patterns (i.e., with zero mean and standard deviation one), and let L_q, U_q be the lower and upper bounds in the dimension $q = 1, 2$, of the squared region $\mathscr{R}$ which will be covered by the classification map. Since the 2D patterns are standardized, it is expectable that $-2 \leq y_{i1}, y_{i2} \leq 2$ for most patterns, so $L_q = -2$ and $U_q = 2$ for $q = 1, 2$, should be a good choice. In those datasets with many 2D patterns outside the square $[-2, 2] \times [-2, 2]$, the values L_q and U_q might be set adequately, and they even might depend on q, so the region $\mathscr{R}$ and the $s \times s$ squares might indeed be rectangles. The explanation will however proceed considering squares.

The 2D classification map will cover the square $\mathscr{R} = \{(y_1, y_2) : L_q \leq y_q \leq U_q, q = 1, 2\}$, which is divided in a grid of s^2 squares. The map size s is selected as:

$$s = \max(S_L, \min(\lfloor \sqrt{\eta N} \rfloor, S_U)) \tag{3.8}$$

which is increasing with the number N of training patterns and with the parameter η, a pre-specified ratio between the number of squares and training patterns, so that $\eta \approx s^2/N$. For classification, we use $\eta = 1$ to have roughly so many squares as training patterns. In order to avoid excesively large or small s when N is too high or too low, a lower and an upper bound ($S_L = 10$ and $S_U = 300$, respectively) are set. The upper bound S_U limits the size of the $\mathbf{M}$ matrix, and the computational cost of the procedure to define the map, to $S_U^2 = 90,000$ squares in large classification datasets where $N > S_U^2/\eta = 90,000$ patterns.

The $\mathbf{M}$ matrix defines UF2DC because each element M_{nm} is the class prediction for the test patterns located in the square with indices n and m, with $n, m = 1, \ldots, s$. This square will henceforth be denoted by $\langle n, m \rangle$. The prediction M_{nm} is selected as the class with the highest relative population in $\langle n, m \rangle$, i.e., the class with the highest proportion of training patterns located in that square. In order to create $\mathbf{M}$, a matrix $\mathbf{P}^j$, named "population matrix", of size $s \times s$, is defined for each class $j = 1, \ldots, C$, in such a way that P_{nm}^j, with $n, m = 1, \ldots, s$, is the relative population of class j in $\langle n, m \rangle$. Given $\mathbf{y}_i = (y_{i1}, y_{i2})$, in order to calculate P_{nm}^j we define:

$$u_{iq} = \max\left[1, \min\left(\frac{s(y_{iq} - L_q)}{U_q - L_q}, s\right)\right], \quad q = 1, 2 \tag{3.9}$$

The value u_{iq} is the index of the interval where y_{iq} is located considering a set of s intervals between L_q and U_q, so that $\mathbf{y}_i$ is located in $\langle u_{i1}, u_{i2}\rangle$ and P^j_{nm} is calculated as:

$$P^j_{nm} = \frac{1}{N_j}\sum_{i=1}^{N}\delta(n,u_{i1})\delta(m,u_{i2})\delta(j,c_i) \tag{3.10}$$

where $\delta(a,b) = 1$ when $a = b$ and zero otherwise. Since the value inside the sum is only non-zero when $u_{i1} = n$, $u_{i1} = m$ and $c_i = j$, it follows that P^j_{nm} is the rate of training patterns of class j that are located in the square $\langle n,m\rangle$.

The matrix $\mathbf{P}^j$ is fairly noisy because their elements are calculated separately, without any kind of spatial averaging. This noise can be reduced by applying a smoothing process performed e.g. by a 2D median filter [37]. Specifically, we used the Matlab comand:

$$\mathbf{P}^j = \texttt{medfilt2}(\mathbf{P}^j, [K,K]) \tag{3.11}$$

which applies a median filter with a square mask of size K, padding the matrix with zeros on the edges. A value used frequently in this kind of filtering is K=10. When $\langle n,m\rangle$ is not empty, i.e., exists at least one class with training patterns in $\langle n,m\rangle$, the element M_{nm} is defined as:

$$M_{nm} = \underset{j=1,\ldots,C}{\arg\max}\{P^j_{nm}\}, \quad \text{when } \exists j \in \{1,\ldots,C\} \text{ with } P^j_{nm} > 0 \tag{3.12}$$

Otherwise, when $\langle n,m\rangle$ is empty, so there are no training patterns on it and $P^j_{nm} = 0$ for $j = 1,\ldots,C$, the calculation of M_{nm} must consider a set of squares, a set with squared shape, centered in $\langle n,m\rangle$, in order to perform the arg max function in eq. 3.12. This set of squares is usually referred in the literature as a "mask" centered in $\langle n,m\rangle$. For reasons of computational efficiency, the mask should be of the smallest size required to be non-empty. In order to be centered in $\langle n,m\rangle$, the mask size in each dimension must be $2k+1$, being k the semi-size of the mask. Thus, k must be the least value in the set $\{1,\ldots,s\}$ such that the mask includes some square $\langle r,v\rangle$ with $P^j_{rv} > 0$ for some class $j \in \{1,\ldots,C\}$.

Since the mask is centered on $\langle n,m\rangle$ and has size $2k+1$, in order to be inside the map, that has size s, it must be $r \in \{r_{nk},\ldots,R_{nk}\}$, with $r_{nk} = \max(1,n-k)$ and $R_{nk} = \min(n+k,s)$. Analogously, it must be $v \in \{v_{mk},\ldots,V_{mk}\}$, with $v_{mk} = \max(1,m-k)$ and $V_{mk} = \min(m+k,s)$. Defining:

$$\phi(a,b) = \max(1,a-b), \qquad \Phi(a,b) = \min(a+b,s) \tag{3.13}$$

the lower and upper bounds for r can be written as:

$$r_{nk} = \phi(n,k), \qquad R_{nk} = \Phi(n,k) \tag{3.14}$$

while the bounds for v are given by:

$$v_{mk} = \phi(m,k), \qquad V_{mk} = \Phi(m,k) \tag{3.15}$$

The value of the mask semi-size k is thus defined as:

$$k = \underset{l=1,\ldots,s}{\arg\min} \left\{ \exists j \in \{1,\ldots,C\}, r \in \{r_{nl},\ldots,R_{nl}\}, v \in \{v_{ml},\ldots,V_{ml}\} : P^j_{rv} > 0 \right\} \tag{3.16}$$

Once k is known, the element M_{nm} of matrix $\mathbf{M}$ is the class j with the largest sum of P^j_{rv} over the squares of the mask with size $2k+1$:

$$M_{nm} = \underset{j=1,\ldots,C}{\arg\max} \{ \mathscr{P}^j_{nm} \}, \quad \mathscr{P}^j_{nm} = \sum_{r=r_{nk}}^{R_{nk}} \sum_{v=v_{mk}}^{V_{mk}} P^j_{rv} \tag{3.17}$$

$$\text{when } P^j_{nm} = 0, \quad \forall j \in \{1,\ldots,C\}$$

As an efficient procedure to calculate M_{nm} (see item 15 and Algorithm 2 in section 3.5), the value of k is set to 1 in the beginning of each row n of the $\mathbf{M}$ matrix (when $m = 1$). This value is increased by 1 until $\mathscr{P}^j_{nm} > 0$ for some class j. At this point, the increase of k stops, and M_{nm} is calculated. Next, we must calculate the matrix value for the next square $\langle n, m+1 \rangle$. Note that the current value of k already provides at least one $P^j_{rv} > 0$ (non-empty square) inside the mask. Thus, in order to calculate $M_{n,m+1}$ for $\langle n, m+1 \rangle$ in the next column, the value $k-1$ also gives some $P^j_{rv} > 0$, so the starting value of k for $\langle n, m+1 \rangle$ is selected as $k = \max(1, k-1)$. Here, the max function avoids $k = 0$. Finally, when $P^j_{nm} > 0$ for some $j = 1,\ldots,C$ (non-empty square, eq. 3.12), the value of k is reset to 1, because $\langle n, m+1 \rangle$ will only need at most a mask of size 3 ($= 2k+1$ with k=1) to include non-zero P^j_{rv} values.

Once the matrix $\mathbf{M}$ has been calculated, the classification of a new test pattern $\mathbf{x}_t \in \mathbb{R}^p$ requires:

1. To project $\mathbf{x}_t$ to $\mathbf{y}_t = (y_{t1}, y_{t2}) = \mathbf{A}(\mathbf{x}_t - \bar{\mathbf{x}})$.
2. To calculate the indices $n = u_{t1}$ and $m = u_{t2}$ of the square where $\mathbf{y}_t$ is located (eq. 3.9).
3. To assign the test pattern $\mathbf{x}_t$ to the class $z_t = M_{nm}$.
4. If the test pattern $\mathbf{x}_t$ is mapped to a point $\mathbf{y}_t$ outside $\mathscr{R}$, the predicted class is $z_t = M_{nm}$, where $n = 1$ when $y_{t1} < L_1$, $n = s$ for $y_{t1} > U_1$, $m = 1$ when $y_{t2} < L_2$ and $m = s$ when $y_{t2} > U_2$.

The squared matrix $\mathbf{M}$ is a low-resolution map of size $s \leq S_U = 300$, which is enough to predict the class for a 2D pattern. However, an adequate data visualization requires a larger image matrix $\mathbf{I}$ of size $s' > s$. This matrix is defined in two steps:

1. The resampled map matrix $\mathbf{M}'$ is defined by augmenting $\mathbf{M}$ to a size s' and processing it with a box-shaped kernel using e.g. with the Matlab command:

$$\mathbf{M}' = \texttt{imresize}(\mathbf{M}, \gamma, \texttt{box}) \tag{3.18}$$

   The scale $\gamma > 1$ is defined as $\gamma = \min(\Gamma, S/s)$, where $\Gamma = 10$ is the maximum scale and $S = 1{,}000$ is the maximum size of $\mathbf{M}'$ in each dimension, so that $s' = \min(\Gamma s, S)$.

2. The image matrix $\mathbf{I}$ is a smoothed version of $\mathbf{M}'$ defined by applying the `mode` function over sliding squares (columnwise) of size $\xi = \lfloor 3\gamma/2 \rceil$ in each dimension ($\lfloor \cdot \rceil$ is the nearest integer function), using the Matlab command:

$$\mathbf{I} = \texttt{colfilt}(\mathbf{M}', [\xi, \xi], \texttt{sliding}, \texttt{mode}). \tag{3.19}$$

Note that the calculation of the larger $\mathbf{M}'$ and $\mathbf{I}$ matrices, both of size s', is only necessary for visualization, but not for classification, that only requires the matrix $\mathbf{M}$, so the time required to the calculation of $\mathbf{M}'$ and $\mathbf{I}$ can be saved. The whole pseudocode of UF2DC is listed in section 3.5.

Figure 3.1 shows the image matrix $\mathbf{I}$ of the classification map defined by UF2DC and the 2D mapped training patterns (small squares) for the UCI classification dataset *Synthetic control chart time series* (abbreviated as `synthetic`), with 6 classes (see section 4.1). The map provides a coherent 2D representation that shows which classes are adjacent or nearby. The color of the training patterns is given by their class labels, so the figure also shows their distribution over the regions occupied by each class, and the level of class overlap, given by the number of patterns of one class located in regions of other classes. In this dataset, most patterns are located in the region of their own class, so the the class overlap is low and UF2DC discriminates fairly well among classes.

3.4 Two-dimensional regression map with UF2DR

In this section we will describe the ultra-fast 2D regressor, whose pseudo-code is also listed in 3.5. In regresion problems the output is continuous, so the map matrix $\mathbf{M}$ must be larger

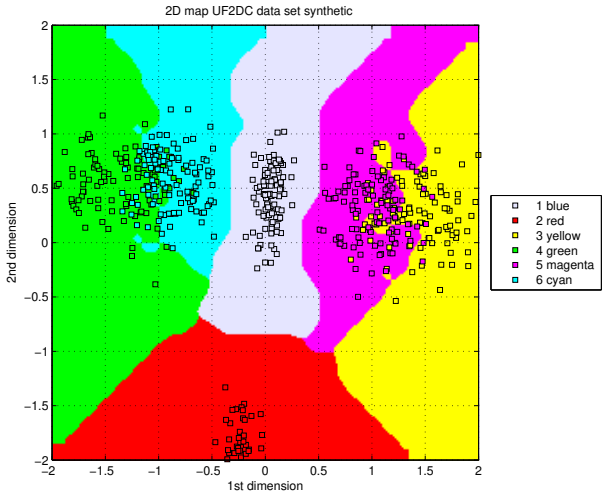

Figure 3.1: Classification map defined by UF2DC and 2D training patterns for the UCI dataset `synthetic`, with 6 classes.

compared to UF2DC in order to codify a high number of output values. Therefore, a higher η value is recommendable to achieve a higher resolution map. We used $\eta = 5$ so the number s^2 of squares in the region $\mathscr{R}$ is five times larger than the number N of training patterns. The map size s is calculated from η and N as in UF2DC (eq. 3.8).

In order to calculate the elements M_{nm}, with $n,m = 1,\ldots,s$, of the map matrix $\mathbf{M}$, the selection of the most populated class in classification is replaced by an output averaging in regression, similarly to many other ML approaches. Two squared matrices, $\mathbf{P}$ (population) and $\mathbf{O}$ (output), both of size s, are defined. The element P_{nm} of the population matrix $\mathbf{P}$ is the number of training patterns inside $\langle n,m\rangle$, that can be calculated as:

$$P_{nm} = \sum_{i=1}^{N} \delta(n,u_{i1})\delta(m,u_{i2}) \tag{3.20}$$

where u_{i1}, u_{i2} and $\delta(a,b)$ are defined in section 3.3. The element O_{nm} of the output matrix $\mathbf{O}$ is the sum of the true outputs o_i over the mapped training patterns $\mathbf{y}_i$ located inside $\langle n,m\rangle$:

$$O_{nm} = \sum_{i=1}^{N} \delta(n,u_{i1})\delta(m,u_{i2})o_i \tag{3.21}$$

For a non-empty square $\langle n,m\rangle$, the element $P_{nm} > 0$, so that the predicted output M_{nm} is the average of the true output o_i over the training patterns located inside this square:

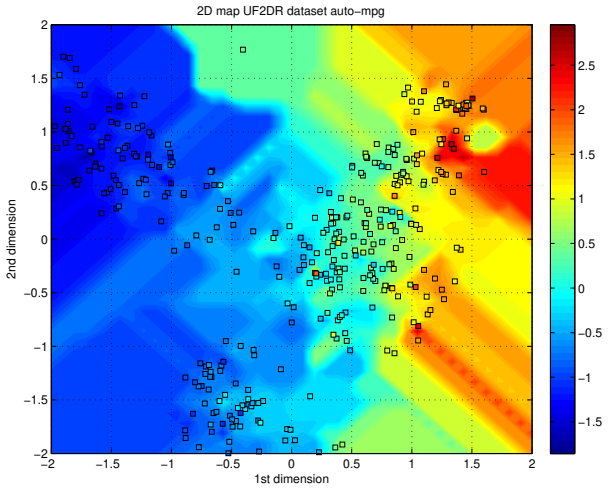

Figure 3.2: Regression map defined by UF2DR and 2D training patterns for the UCI dataset `auto-mpg`.

$$M_{nm} = \frac{O_{nm}}{P_{nm}}, \quad \text{when } P_{nm} > 0 \tag{3.22}$$

With empty squares ($P_{nm} = O_{nm} = 0$), the output M_{nm} is defined, analogously to the classification case, as the average of O_{rv} over the values of r, v inside the squared mask of size $2k+1$ centered in $\langle n, m \rangle$. The limits r_{nk}, R_{nk}, v_{mk} and V_{mk}, for the square indices inside the mask are defined as in section 3.3. The mask semi-size k is the smallest value in $\{1, \ldots, s\}$ for which the mask has non-empty squares:

$$k = \arg\min_{l=1,\ldots,s} \left\{ \exists r_{nl} \leq r \leq R_{nl}, v_{ml} \leq v \leq V_{ml} : P_{rv} > 0 \right\} \tag{3.23}$$

and M_{nm} is given by:

$$M_{nm} = \frac{\sum_{r=r_{nk}}^{R_{nk}} \sum_{v=v_{mk}}^{V_{mk}} O_{rv}}{\sum_{r=r_{nk}}^{R_{nk}} \sum_{v=v_{mk}}^{V_{mk}} P_{rv}}, \quad \text{when } P_{nm} = O_{nm} = 0 \tag{3.24}$$

Finally, the output predicted by UF2DR for a test pattern $\mathbf{x}_t$ mapped to $\mathbf{y}_t = \mathbf{A}(\mathbf{x}_t - \bar{\mathbf{x}})$ is $z_t = M_{nm}$, where $n = u_{t1}$ and $m = u_{t2}$ similarly to UF2DC (see the definition of u_{iq} in eq. 3.9). When $\mathbf{x}_t$ is mapped outside $\mathscr{R}$, the predicted output z_t is as with UF2DC (section 3.3).

The median filtering of the **P** matrix is no longer required in UF2DR because the averaging already filters the output values in the **O** matrix. Similarly, the resampled map $\mathbf{M}'$ and image **I** matrices are no longer necessary because **M** is already of large size, due to the higher η value, and can also be used for visualization. The whole pseudocode of UF2DR is listed in section 3.5.

Figure 3.2 shows an example of regression map defined by UF2DR for the UCI dataset *Auto MPG* (see section 4.2), alongside with the training patterns (small squares), whose color codifies its true output value according to the color bar on the right of the plot. Note that most squares have the same color as the background below them, which means a good agreement between the predicted and true output. This regression map is a meaningful and understandable 2D representation of the unknown function which has to be predicted and of the training pattern distribution. The map plots in the color scale the function values in each region of the space. Also, it is possible to see how the function behaves going to a certain direction: whether it increases, decreases or remains constant. Thus, the map provides an visual explanation of the prediction issued.

3.5 Pseudo-code of UF2DC and UF2DR

This section compiles the algorithm for 2D classification and regression using UF2DC and UF2DR, respectively, following the descriptions presented in the previous sections 3.3 and 3.4. First, the training stage is described. Then, the information that identifies the trained UF2DC or UF2DR is listed. Finally, we describe the steps of the test stage for both methods.

Training stage:

1. **Input (training) data**:
   a) $\{\mathbf{x}_i\}_{i=1}^N$, with $\mathbf{x}_i \in \mathbb{R}^p$: input patterns.
   b) Output:
      i. **UF2DC**: $\{c_i\}_{i=1}^N$, with $c_i \in \{1,\ldots,C\}$: class labels.
      ii. **UF2DR**: $\{o_i\}_{i=1}^N$: output values.
2. **Parameters**:
   a) **Lower and upper number of intervals in each dimension**: $S_L = 10, S_U = 300$.

b) **Ratio between map squares and training patterns**: $\eta = 1$ (for UF2DC) and $\eta = 5$ (for UF2DR).

c) **Map size**: number of intervals in each dimension $s = \max(S_L, \min(\lfloor\sqrt{\eta N}\rfloor, S_U))$.

3. **Total and class means**: $\bar{\mathbf{x}}$ is the mean over all the patterns; $\bar{\mathbf{x}}_j$, with $j = 1,\ldots,C$, are the means over the patterns of each class:

$$\bar{\mathbf{x}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i, \quad \bar{\mathbf{x}}_j = \frac{1}{N_j}\sum_{i\in\mathscr{A}_j}\mathbf{x}_i, \quad j = 1,\ldots,C \tag{3.25}$$

4. **Total covariance matrix T**: squared of size p defined by:

$$\mathbf{T} = \frac{(\mathbf{X}-\bar{\mathbf{X}})^T(\mathbf{X}-\bar{\mathbf{X}})}{N-1} \tag{3.26}$$

The matrices $\mathbf{X}$ and $\bar{\mathbf{X}}$, both of size $N \times p$, contain the training patterns $\mathbf{x}_i$ by rows, and the vector $\bar{\mathbf{x}}$ repeated in each row, respectively.

5. **Within-class covariance matrix W**: squared of size p, defined as:

$$\mathbf{W} = \frac{1}{C}\sum_{j=1}^{C}\frac{\left(\mathbf{X}_j-\bar{\mathbf{X}}_j\right)^T\left(\mathbf{X}_j-\bar{\mathbf{X}}_j\right)}{N_j-1} \tag{3.27}$$

The matrices $\mathbf{X}_j$ and $\bar{\mathbf{X}}_j$ of size $N_j \times p$ contain, by rows, the N_j training patterns $\mathbf{x}_i$ of class j, and N_j rows equal to $\bar{\mathbf{x}}_j$, respectively.

6. **Between-class covariance matrix B**: squared of size p, calculated as $\mathbf{B} = \mathbf{T} - \mathbf{W}$.

7. **Mapping matrix A**: of size $2 \times p$, its rows are the 2 main eigenvectors of $(\mathbf{W} + \alpha\mathbf{I}_p)^{-1}\mathbf{B}$, being α=0.001 and $\mathbf{I}_p$ the squared identity matrix of size p.

8. **Mapping of patterns to 2D**: $\mathbf{y}_i = (y_{i1}, y_{i2}) = \mathbf{A}(\mathbf{x}_i - \bar{\mathbf{x}})$, with $i = 1,\ldots,N$.

9. **Standardization in 2D**: $\mathbf{y}_i = (\mathbf{y}_i - \bar{\mathbf{y}})/\tilde{\mathbf{y}}$, with $i = 1,\ldots,N$ (vector operations are assumed), where $\bar{\mathbf{y}} = (\bar{y}_1, \bar{y}_2)$ and $\tilde{\mathbf{y}} = (\tilde{y}_1, \tilde{y}_2)$ are the mean and standard deviation over the 2D mapped patterns.

10. **Map lower and upper limits in each dimension**: $L_q = -2, U_q = 2$ with $q = 1,2$. These values should be set appropiately when the number of training patterns outside the square $[-2,2] \times [-2,2]$ is high.

11. **Population matrices $\mathbf{P}^j$** (only for UF2DC): squared of size s, with $j = 1,\ldots,C$, and elements P^j_{nm}, for $n,m = 1,\ldots,s$, defined as:

$$P^j_{nm} = \frac{1}{N_j} \sum_{i=1}^{N} \delta(n,u_{i1})\delta(m,u_{i2})\delta(j,c_i) \tag{3.28}$$

where $\delta(a,b) = 1$ when $a = b$ and zero otherwise, while u_{iq}, with $q = 1,2$, are defined as:

$$u_{iq} = \max\left[1, \min\left(\frac{s(y_{iq} - L_q)}{U_q - L_q}, s\right)\right], \quad q = 1,2 \tag{3.29}$$

12. **Population matrix P** (only for UF2DR): squared of size s, with elements P_{nm}, with $n,m = 1,\ldots,s$, defined as:

$$P_{nm} = \sum_{i=1}^{N} \delta(n,u_{i1})\delta(m,u_{i2}) \tag{3.30}$$

13. **Output matrix O** (only for UF2DR): squared of size s, whose elements O_{nm}, with $n,m = 1,\ldots,s$, are:

$$O_{nm} = \sum_{i=1}^{N} \delta(n,u_{i1})\delta(m,u_{i2})o_i \tag{3.31}$$

The population and output matrices can be efficiently calculated from $\{u_{i1},u_{i2}\}_{i=1}^{N}$ using algorithm 1.

Algorithm 1: Population and output matrices.

```
for i=1:N do
    n = u_{i1}; m = u_{i2};
    j = c_i; P^j_{nm} = P^j_{nm} + 1/N_j; #UF2DC;
    P_{nm} = P_{nm} + 1; O_{nm} = O_{nm} + o_i; #UF2DR;
```

14. **Median filtering of population matrices** (only for UF2DC): with mask size K=10, using the Matlab command:

$$\mathbf{P}^j=\texttt{medfilt2}(\mathbf{P}^j,[K,K]), \text{ with } j=1,\ldots,C$$

15. **Map matrix M** (only for UF2DC): squared of size s. In order to calculate M_{nm}, with $n,m=1,\ldots,s$, when $\exists j \in \{1,\ldots,C\}$ such that $P^j_{nm}>0$, the n,m-th square, denoted as $\langle n,m\rangle$, is not empty and:

$$M_{nm}=\underset{j=1,\ldots,C}{\arg\max}\{P^j_{nm}\} \tag{3.32}$$

When $P^j_{nm}=0, \forall j=1,\ldots,C$, the square $\langle n,m\rangle$ is empty. The semi-size k of the squared mask is calculated as:

$$k=\underset{l=1,\ldots,s}{\arg\min}\Big\{\exists j\in\{1,\ldots,C\}, r\in\{r_{nl},\ldots,R_{nl}\}, v\in\{v_{ml},\ldots,V_{ml}\}: P^j_{rv}>0\Big\} \tag{3.33}$$

where $r_{nl}=\phi(n,l)$, $R_{nl}=\Phi(n,l)$, $v_{ml}=\phi(m,l)$ and $V_{ml}=\Phi(m,l)$, being $\phi(a,b)=\max(1,a-b)$ and $\Phi(a,b)=\min(a+b,s)$. The element M_{nm} is calculated as:

$$M_{nm}=\underset{j=1,\ldots,C}{\arg\max}\{\mathscr{P}^j_{nm}\}, \quad \mathscr{P}^j_{nm}=\sum_{r=r_{nk}}^{R_{nk}}\sum_{v=v_{mk}}^{V_{mk}}P^j_{rv} \tag{3.34}$$

16. **Map matrix M** (only for UF2DR): when $\langle n,m\rangle$ is not empty, M_{nm} is given by:

$$M_{nm}=\frac{O_{nm}}{P_{nm}}, \quad n,m=1,\ldots,s \tag{3.35}$$

When $\langle n,m\rangle$ is empty, the mask semi-size k is calculated as:

$$k=\underset{l=1,\ldots,s}{\arg\min}\Big\{\exists r\in\{r_{nl},\ldots,R_{nl}\}, v\in\{v_{ml},\ldots,V_{ml}\}: P_{rv}>0\Big\} \tag{3.36}$$

where r_{nl}, R_{nl}, v_{ml} and V_{ml} are defined as above for UF2DC. The element M_{nm} is calculated as:

$$M_{nm}=\frac{\sum_{r=r_{nk}}^{R_{nk}}\sum_{v=v_{mk}}^{V_{mk}}O_{rv}}{\sum_{r=r_{nk}}^{R_{nk}}\sum_{v=v_{mk}}^{V_{mk}}P_{rv}} \tag{3.37}$$

17. **Efficient calculation of M**: the values M_{nm} for UF2DC and UF2DR can be efficiently calculated using Algorithm 2.

Algorithm 2: Efficient calculation of the **M** matrix from P^j_{nm} (resp. O_{nm} and P_{nm}) for UF2DC (resp. UF2DR).

for n=*1:s* **do**
  $k_0 = 1$;
  for m=*1:s* **do**
    $P_{nm} = \sum_{j=1}^{C} P^j_{nm}$; #UF2DC;
    if $P_{nm} > 0$ **then**
      $M_{nm} = \arg\max_{j=1,\dots,C}\{P^j_{nm}\}$; #UF2DC;
      $M_{nm} = O_{nm}/P_{nm}$; #UF2DR;
      $k_0 = 1$;
    else
      for k=k_0*:s* **do**
        $r_{nk} = \phi(n,k)$;$R_{nk} = \Phi(n,k)$;
        $v_{mk} = \phi(m,k)$;$V_{mk} = \Phi(m,k)$;
        for j=*1:C* **do**
          $\mathscr{P}^j_{nm} = \sum_{r=r_{nk}}^{R_{nk}} \sum_{v=v_{mk}}^{V_{mk}} P^j_{rv}$; #UF2DC;
        $\mathscr{P}_{nm} = \sum_{j=1}^{C} \mathscr{P}^j_{nm}$; #UF2DC;
        $\mathscr{P}_{nm} = \sum_{r=r_{nk}}^{R_{nk}} \sum_{v=v_{mk}}^{V_{mk}} P_{rv}$; #UF2DR;
        if $\mathscr{P}_{nm} > 0$ **then**
          $M_{nm} = \arg\max_{j=1,\dots,C}\{\mathscr{P}^j_{nm}\}$; #UF2DC;
          $M_{nm} = \frac{1}{\mathscr{P}_{nm}} \sum_{r=r_{nk}}^{R_{nk}} \sum_{v=v_{mk}}^{V_{mk}} O_{rv}$; #UF2DR;
          $k_0 = \max(1, k-1)$; **break**;

18. **Resampled map matrix M′** (only for UF2DC, only for visualization): squared of size $s' > s$. Let S=1,000 be the maximum size of the resampled image and Γ=10 the maxi-

mum scale. Resize $\mathbf{M}$ by a scale γ=min$(\Gamma, S/s)$ to the size $s' = \min(\Gamma s, S)$ and apply a `box` kernel to achieve $\mathbf{M}'$, using the Matlab command:

$$\mathbf{M}' = \texttt{imresize}(\mathbf{M},\gamma,\texttt{box})$$

19. **Image matrix I** (only for UF2DC, only for visualization): squared of size s'. Let $\xi = \lfloor \frac{3\gamma}{2} \rceil$ be the smoothing step, where $\lfloor \cdot \rceil$ is the nearest integer. Smooth the matrix $\mathbf{M}'$ applying the `mode` function to columnwise sliding neighborhoods with step ξ in order to achieve $\mathbf{I}$, using the Matlab command:

$$\mathbf{I}=\texttt{colfilt}(\mathbf{M}',[\xi,\xi],\texttt{sliding},\texttt{mode})$$

Trained UF2DC/UF2DR defined by:

1. Map size: s.
2. Map limits: L_q, U_q, with $q = 1, 2$.
3. Total mean: $\bar{\mathbf{x}} \in \mathbb{R}^p$.
4. Mapping matrix: $\mathbf{A}$, of size $2 \times p$.
5. 2D mean/deviation vectors: $\bar{\mathbf{y}}, \tilde{\mathbf{y}} \in \mathbb{R}^2$.
6. Map matrix: $\mathbf{M}$, squared of size s.
7. Image matrix (only for UF2DC, only for visualization): $\mathbf{I}$, squared of size s'.

Test stage: output prediction for a test pattern $\mathbf{x}_t \in \mathbb{R}^p$.

1. **Mapping**: $\mathbf{y}_t = \mathbf{A}(\mathbf{x}_t - \bar{\mathbf{x}}) \in \mathbb{R}^2$
2. **Standardization**: $\mathbf{y}_t = (y_{t1}, y_{t2}) = (\mathbf{y}_t - \bar{\mathbf{y}})/\tilde{\mathbf{y}}$, with vector operations.
3. **Prediction inside** $\mathscr{R}$: if $\mathbf{x}_t$ is mapped to a point $\mathbf{y}_t$ inside $\mathscr{R}$, i.e.

$$1 \leq \frac{s(y_{tq} - L_q)}{N_q - L_q} \leq s, \quad q = 1, 2 \tag{3.38}$$

then the output predicted for a test pattern $\mathbf{x}_t$ is $z_t = M_{nm}$, with $n = u_{t1}$, $m = u_{t2}$ and u_{tq} for $q = 1, 2$ is defined by eq. 3.29 replacing y_{iq} by y_{tq}.

4. **Prediction outside** $\mathscr{R}$: if $\mathbf{x}_t$ is mapped to a point $\mathbf{y}_t$ outside $\mathscr{R}$, i.e.:

$$\frac{s(y_{tq}-L_q)}{N_q-L_q} < 1 \text{ or } \frac{s(y_{tq}-L_q)}{N_q-L_q} > s, \text{ for } q=1 \text{ or } q=2 \tag{3.39}$$

then the predicted value is $z_t = M_{nm}$, where $n = 1$ when $y_{t1} < L_1$, $n = s$ for $y_{t1} > U_1$, $m = 1$ when $y_{t2} < L_2$ and $m = s$ when $y_{t2} > U_2$.

CHAPTER 4

RESULTS AND DISCUSSION OF 2D CLASSIFICATION AND REGRESSION

We evaluated UF2DC and UF2DR on a collection of classification and regression datasets, described in the following sections. We used 4-fold cross-validation in all the experiments, using 2, 1 and 1 folds in each trial for training, validation and test, respectively, for the hyper-parameter tuning. All the programs are executed in Matlab R2012a under a Linux Kubuntu 18.04 (64 bit) computer with Intel®Core™ i7-4790K CPU at 4GHz, equipped with 32GB of RAM. The classifiers and regressors are allowed a maximum run time of 192 hours (8 days).

4.1 Classification

Figure 4.1 (left panel) plots the simple 2D artificial dataset multiple-spirals-appart, generated by us with $N = 1,200$ patterns in $C = 6$ classes (spirals) with $N_j = 200$ patterns/class for $j = 1,\ldots,C$. The i-th point in the j-th spiral (x_{ji}, y_{ji}), with $j = 1,\ldots,C$ and $i = 0,\ldots,N_j$, is given by:

$$x_{ji} = \rho_{ji}\cos\theta_{ji}, \quad y_{ji} = \rho_{ji}\sin\theta_{ji}, \quad \rho_{ji} = 2\pi i/N_j, \quad \theta_{ji} = \rho_{ji} + j \tag{4.1}$$

The classification map defined by UF2DC in the right panel of Figure 4.1 shows the regions of the 6 classes, that are clearly separated, and the training patterns (small squares in the figure), that are placed in the regions of the right classes.

Table 4.1 compiles the 26 real classification datasets from the UCI machine learning repository [24], sorted by decreasing populations, that we used to evaluate the proposed approach

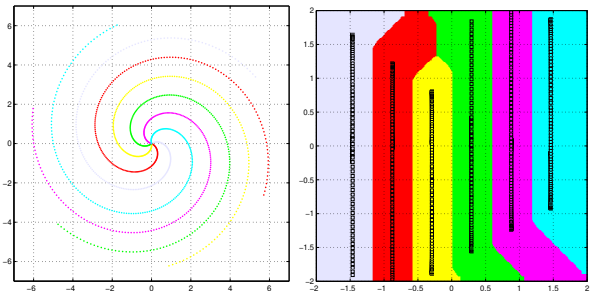

Figure 4.1: Data set multiple spirals appart (left) and classification map defined by UF2DC (right).

UF2DC for classification problems. Figure 4.2 shows an example of classification map defined by UF2DC in one of these datasets (`tic-tac`), that has two classes. The 2D mapped training patterns of both classes are very separated and they are placed inside the region of the classification map associated to its right class, excepting few patterns, so that UF2DC discriminates perfectly between both classes.

The performance of UF2DC on the datasets of Table 4.1 is compared with other six well-known classifiers. Those classifiers whose name starts with p are trained on the original p-dimensional patterns.

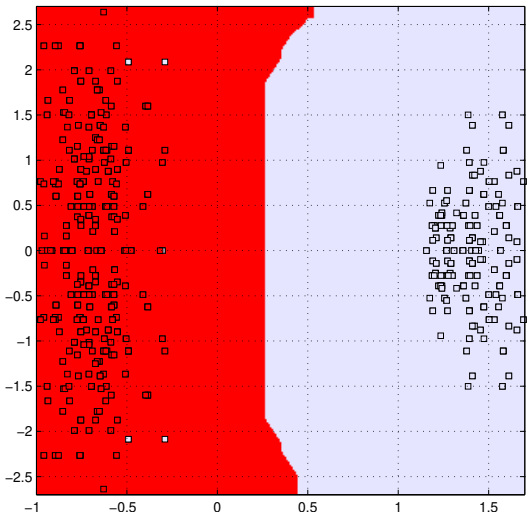

Figure 4.2: Classification map defined by UF2DC and 2D training patterns for dataset `tic-tac` with 2 classes.

Dataset	Name	N	p	C
Susy	susy	5,000,000	18	2
MiniBooNE particle identification	miniboone	130,064	50	2
Mnist	mnist	70,000	719	10
Connect-4	connect-4	67,557	42	3
Adult	adult	48,842	14	2
MAGIC Gamma Telescope	magic	19,020	10	2
Nursery	nursery	12,958	8	4
Abalone	abalone	4,177	9	3
Image segmentation	imseg	2,086	18	7
Tic-tac-toe endgame	tictac	958	18	2
Annealing	annealing	886	52	5
Vehicle silhouettes	vehicle	846	18	4
Energy efficiency	energy-heat	768	7	3
Pima indians diabetes	pima	768	8	2
Australian credit approval	australian	690	35	2
Mammographic mass	mammograph	642	15	2
Synthetic control chart time series	synthetic	600	60	6
Dermatology	dermatology	366	98	6
German credit	german	366	44	2
Ionosphere	ionosphere	350	33	2
US congressional voting records	voting	342	32	2
Statlog heart	heart	270	20	2
Seeds	seeds	210	7	3
Sonar, mines vs. rocks	sonar	208	60	2
Wine recognition	wine	178	13	3
Hepatitis domain	hepatitis	155	24	2

Table 4.1: List of the 26 UCI classification datasets.

1. *p***GSVC**: Gaussian kernel support vector machine classifier (LibSVM [15] library v. 3.22) on the original p-dimensional patterns. It was selected because it is a state-of-the-art classifier [22], although its training speed, tunable hyper-parameters (regularization parameter, with 11 values in the set $\{2^i\}$ with i from -5 to 15 with step 2, and inverse of the Gaussian kernel spread, with 10 values in the set $\{2^i\}$ with i from -15 to 3 with step 2) and memory requirements may hinder its use with large datasets.

2. *p***LSVC**: large-scale linear kernel support vector machine (LibLinear [27] library v. 2.3.20), working on the original p-dimensional patterns. It is selected due to its speed and ability to process large datasets. However, its performance is expected to be suboptimal compared to *p*GSVC because the former uses a linear kernel. The regularization parameter is tuned as *p*GSVC.

3. *p***WLSVC**: a linear SVC combined with dimensionality reduction in a wrapper approach [81], performed by a radial basis function (RBF) neural network trained to optimize the performance of the linear SVC (LibSVM v. 3.18), that is applied on the 2D

mapped patterns[1]. The spread of the RBF network is tuned with values 2^i for i=-7 to 7 with step 2, and the regularization parameter of the linear SVC is tuned as pGSVC.

4. p**PNN**: probabilistic neural network [68], implemented by the `newpnn` function in the Matlab Neural Network Toolbox[2], trained on the original p-dimensional patterns. This classifier was selected due to its speed, low memory requirements and simple tuning (only the spread of the Gaussian functions must be tuned with 19 values between 0.001 and 10), which allows to classify large datasets.

5. h**PNN**: the PNN trained on the h-dimensional patterns ($h = C-1$) mapped by LDA using a matrix $\mathbf{A}$ of size $h \times p$ composed by the h main eigenvectors of $(\mathbf{W}+\alpha\mathbf{I}_p)^{-1}\mathbf{B}$. This classifier was tuned as pPNN.

6. **2PNN**: the PNN trained on the 2D patterns mapped by LDA using the 2 main eigenvectors and tuned as pPNN.

Table 4.2 reports the Cohen kappa score [14], averaged over the 4 folds (standard deviations are not reported to save space and increase readability), achieved by the previous classifiers in each dataset, and the average kappa over all the datasets (lower row) for each classifier, excluding errors. The kappa is used instead of accuracy because it takes into account the class unbalancing, discarding the agreement by change between predicted and true class. The value of kappa (κ, in %) can be calculated as:

$$\kappa = 100\frac{p_a - p_e}{s - p_e}, \qquad p_a = \sum_{i=1}^{C} n_{ii}$$
$$p_e = \frac{1}{s}\sum_{i=1}^{C}\left(\sum_{j=1}^{C} n_{ij}\right)\left(\sum_{k=1}^{C} n_{ki}\right); \qquad s = \sum_{i=1}^{C}\sum_{j=1}^{C} n_{ij}$$

where n_{ij} is the number of patterns of class i that classifier predicts as of class j. Dashed (—) cells in Table 4.2 correspond to datasets where the corresponding classifier did not finish within 8 days (which will be referred as "time errors") or crashed due to lack of memory (referred as "memory errors"). This happens in the 4 largest datasets (`mnist`, `miniboone`, `susy` and `connect-4`) for pGSVC (time errors); in the same datasets plus `adult` for pWLSVC (time errors); and in dataset `susy` for pPNN, hPNN and 2PNN (memory errors).

[1] http://faculty.ucmerced.edu/mcarreira-perpinan/research/software/ldsvm.tar.gz
[2] http://www.mathworks.com

Dataset	*p*GSVC	*p*LSVC	*p*WLSVC	*p*PNN	*h*PNN	2PNN	UF2DC
wine	0.958	0.983	0.974	0.932	0.992	0.992	0.983
tictac	0.977	0.960	0.975	0.970	0.958	0.963	0.960
annealing	0.970	0.956	0.764	0.906	0.958	0.926	0.866
seeds	0.907	0.943	0.865	0.894	0.950	0.950	0.951
dermatology	0.959	0.939	0.622	0.932	0.946	0.736	0.666
nursery	1.000	0.866	0.732	0.917	0.931	0.875	0.836
voting	0.885	0.862	0.876	0.786	0.872	0.873	0.839
mnist	—	0.837	—	0.931	0.900	0.521	0.509
imseg	0.954	0.795	0.593	0.929	0.921	0.754	0.726
energy-heat	0.918	0.787	0.815	0.862	0.907	0.907	0.856
synthetic	0.990	0.782	0.494	0.960	0.966	0.778	0.756
miniboone	—	0.724	—	0.818	0.768	0.763	0.761
ionosphere	0.865	0.700	0.843	0.877	0.707	0.688	0.671
heart	0.686	0.692	0.667	0.713	0.658	0.673	0.617
australian	0.719	0.675	0.685	0.555	0.697	0.714	0.690
vehicle	0.769	0.672	0.531	0.609	0.707	0.650	0.617
mammograph	0.579	0.568	0.613	0.551	0.568	0.545	0.510
adult	0.556	0.556	—	0.488	0.521	0.525	0.503
susy	—	0.544	—	—	—	—	0.517
sonar	0.639	0.543	0.597	0.657	0.352	0.374	0.321
magic	0.708	0.524	0.710	0.621	0.497	0.574	0.592
pima	0.476	0.469	0.460	0.394	0.472	0.464	0.438
abalone	0.454	0.454	0.487	0.442	0.469	0.472	0.442
connect-4	—	0.422	—	0.428	0.451	0.450	0.408
hepatitis	0.489	0.348	0.500	0.454	0.367	0.454	0.373
german	0.281	0.313	0.179	0.143	0.198	0.248	0.286
Average	0.761	0.689	0.666	0.711	0.709	0.675	0.642

Table 4.2: Kappa achieved by *p*GSVC, *p*LSVC, *p*WLSVC, *p*PNN, *h*PNN, 2PNN and UF2DC.

The average kappa achieved by UF2DC (0.642) is only 0.119 below *p*GSVC (0.761) and 0.069 below *p*PNN, so the method proposed by us to define the 2D map is only slightly below the classification on the original p-dimensional space. The difference between *p*GSVC and UF2DC is below 0.15 in 15 of 26 datasets, it is always below 0.31 and only overcomes 0.2 in four datasets: `synthetic`, `dermatology`, `imseg` and `sonar`. In the first three datasets, the low performance of UF2DC is caused by retaining only 2 eigenvectors in LDA for 2D visualization, as shown by the fact that 2PNN achieves kappa similar to UF2DC, while *h*PNN (using $C-1$ eigenvectors) is similar to *p*GSVC. This also happens in dataset `mnist`, where *p*GSVC does not finish, but UF2DC and 2PNN achieve kappa fairly lower than *p*PNN and *h*PNN. In this dataset, this fact is emphasized by the high number of classes ($C = 10$), which is widely known that raises the number of required eigenvectors in the LDA. In dataset `sonar`, *h*PNN already is below *p*PNN and *p*GSVC, so retaining h eigenvectors also reduces the performance.

Comparing 2PNN and UF2DC, the difference never overcomes 0.1, so their performances are very similar. This means that the proposed method defines 2D classification maps simi-

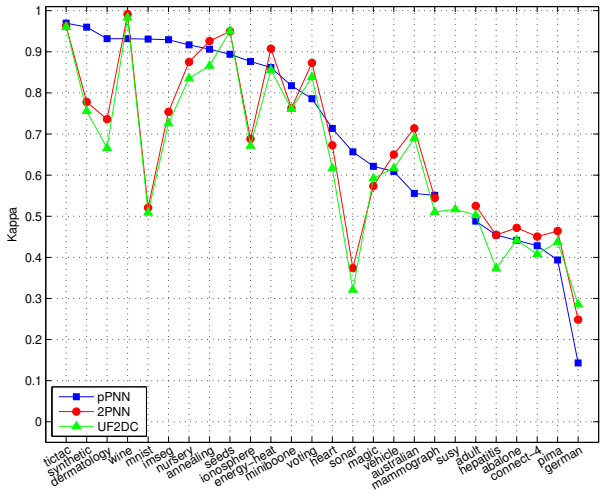

Figure 4.3: Values of kappa achieved using pPNN, 2PNN and UF2DC.

larly to PNN trained on the 2D mapped patterns (2PNN), but without training and tuning a classifier, so it is much faster, as we will see. Another remarkable fact is that pPNN and hPNN achieve similar average kappas (0.711 and 0.709, respectively), which shows that PNN combined with LDA using the h main eigenvectors (i.e., a mapping to $\mathbb{R}^h$ instead to $\mathbb{R}^2$) achieves performances similar to the original patterns. In 16 of 26 datasets hPNN overcomes pPNN, and pPNN only overcomes hPNN by more than 0.1 in 3 datasets.

The average kappa (0.689) of pLSVC is only 0.05 above UF2DC, because the former uses the original patterns, but the difference only overcomes 0.1 in 3 of 26 datasets, achieving its highest value in dataset `mnist`, because C is high and UF2DC uses only two eigenvectors. The pWLSVC is 0.095 below pGSVC and 0.0119 above UF2DC, but in 5 of 26 datasets it does not finish within 8 days.

Figure 4.3 shows that UF2DC follows 2PNN very nearly, and they follow pPNN except in: 1) datasets `synthetic`, `dermatology`, `mnist` and `imseg`, due to the use of only two eigenvectors; and 2) datasets `ionosphere` and `sonar`, where hPNN (see Table 4.2) is already below pPNN and pGSVC. The low difference between UF2DC and 2PNN can be considered a low price to pay in order to achieve a faster method to define the 2D classification map without the need of training a classifier. It is also remarkable that for 8 of 26 datasets (e.g. `australian` and `voting`) UF2DC overcomes pPNN. Globally, the use of

two eigenvalues for 2D visualization in UF2DC causes a performance loss (in kappa) compared to p-dimensions (pGSVC or pPNN) that is not significant for 20 of 26 datasets, being significant only for 6 of them, with an average loss (last line of Table 4.2) of 0.119 between pGSVC and UF2DC over all the datasets.

Dataset	pPNN	2PNN	pGSVC	pLSVC	pWLSVC	UF2DC	Map
susy	—	—	—	1.1e+05	—	175.48	62.97
mnist	2e+05	6896.84	1.9e+07*	2.1e+04	2.6e+07*	70.12	30.06
miniboone	5.9e+04	1.3e+04	3.8e+06*	849.71	—	92.19	14.45
connect-4	1.7e+04	1034.54	3.1e+06*	1698.22	3.8e+07*	13.99	11.13
adult	7073.83	584.05	5.2e+05	476.90	9.6e+06*	15.84	8.15
magic	127.55	85.34	1.6e+04	22.88	7.7e+05*	1.65	7.26
nursery	89.59	43.05	3908.50	45.27	2.7e+05	1.64	9.96
abalone	14.88	8.13	10.57	10.72	1.6e+05	0.51	4.26
annealing	5.37	3.21	21.74	4.86	1.3e+05	0.15	0.79
imseg	4.87	3.80	35.67	13.82	1.2e+05	0.25	1.60
synthetic	2.69	2.56	11.83	5.76	5.5e+04	0.12	0.61
tictac	2.54	2.37	14.12	0.75	4.2e+04	0.09	0.94
australian	2.49	2.37	10.14	1.06	3.8e+04	0.09	0.69
dermatology	2.46	2.55	8.87	1.99	3.6e+04	0.10	0.36
energy-heat	2.39	2.37	6.26	1.12	2.7e+04	0.07	0.78
vehicle	2.37	2.26	11.30	2.97	2.3e+04	0.09	0.82
mammograph	2.03	1.99	10.98	0.85	1.6e+04	0.05	0.66
german	2.02	2.06	4.65	0.83	1.3e+04	0.05	0.38
pima	2.02	1.99	10.34	0.70	1.1e+04	0.06	0.75
ionosphere	1.89	2.76	3.53	0.41	8204.34	0.04	0.30
voting	1.85	1.88	3.50	0.35	6781.85	0.05	0.33
sonar	1.79	1.87	2.89	0.35	5398.25	0.04	0.24
heart	1.68	1.71	2.31	0.31	4506.20	0.03	0.26
hepatitis	1.60	1.61	1.33	0.26	3381.11	0.03	0.16
seeds	1.51	1.61	1.12	0.22	2025.15	0.03	0.26
wine	1.51	1.57	1.22	0.14	953.14	0.03	0.19

Table 4.3: Elapsed time spent by pPNN, 2PNN, pGSVC, pLSVC, pWLSVC and UF2DC, and by the map definition.

Table 4.3 reports the times spent by pPNN, 2PNN, pGSVC, pLSVC, pWLSVC and UF2DC in each classification dataset, sorted by decreasing times of UF2DC. Values with asterisks (pGSVC in datasets `mnist`, `miniboone` and `connect-4`) are times estimated using the number of tuning trials finished by pGSVC within 8 days, because pGSVC had time errors (it did not finished within 8 days, as reported in Table 4.2). Dashed cells mean that no tuning trial was finished due to time or memory errors, so the time could not be estimated. The times of UF2DC are really tiny compared to the remaining classifiers, whose values raise very fastly with the dataset population, specially pWLSVC, which is the slowest, and pGSVC. The pPNN is slower than 2PNN, and both are much slower than UF2DC. Despite pLSVC is known to be very fast, it is 10-1000 times slower than UF2DC depending on the dataset size. Note that UF2DC only spends 1 and 3 minutes to classify large datasets as

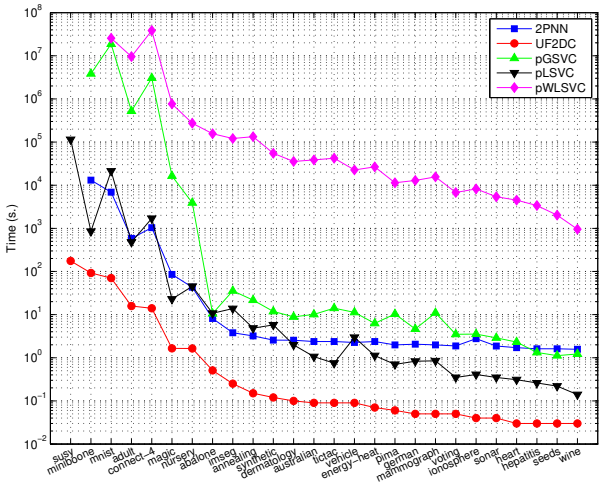

Figure 4.4: Times (in a logarithmic scale) spent by 2PNN, UF2DC, *p*GSVC, *p*LSVC and *p*WLSVC.

`mnist` (70,000 patterns, 719 inputs) and `susy` (5 million patterns), where *p*LSVC spends 6 and 32 hours, respectively. Remember that LDA requires to calculate the eigenvectors of the p-order squared matrix $(\mathbf{W}+\alpha\mathbf{I}_p)^{-1}\mathbf{B}$, which may be time-consuming for high p. However, the UF2DC classifier is fairly fast (70 s.) on the dataset with the highest dimensionality (`mnist`, p=719). Therefore it scales pretty well with the number of inputs compared to other classifiers, although it is oriented to data visualization and classification of general, and not specifically high-dimensional, datasets.

The last column (Map) of Table 4.3 reports the times spent by UF2DC to define the map of the whole dataset including the resampling time (see section 3.3). These times only overcome 10 s. in the 4 datasets with more than 50,000 patterns, and they overcome one minute, only slightly, in the largest dataset `susy`.

Figure 4.4 plots the times spent by 2PNN, UF2DC, *p*GSVC, *p*LSVC and *p*WLSVC. In the smaller datasets (right end of the plot), UF2DC is in general two orders of magnitude faster than *p*GSVC and 2PNN, six orders faster then *p*WLSVC and one order faster than *p*LSVC. The difference raises with the dataset size (left end of the plot) to six orders for *p*GSVC (*p*WLSVC has time errors) and two (resp. three) orders for 2PNN (resp. *p*LSVC), because UF2DC does not increase the size s of matrices $\mathbf{P}^j$ and $\mathbf{M}$, while the other classifiers strongly slow down with N and p.

Considering the times spent by UF2DC in each stage of the map definition, the calculation of the $\mathbf{P}^j$ matrices (section 3.3) spends a maximum time of 0.3 s. in the largest dataset `susy` (5 million patterns). The median filtering is also very fast, with times below 0.44 s. in all the datasets. The calculation of the $\mathbf{M}$ matrix also spends times below 1 s. except in the largest datasets `susy` (1.1 s.), `adult` (2.2 s.), `connect-4` (3.1 s.), `miniboone` (6.4 s.) and `mnist` (9.2 s). Remarkably, this time: 1) does not depend so much on N because the size of matrices $\mathbf{P}^j$ and $\mathbf{M}$ is upper bounded by S_U when $N > S_U^2/\eta$; and 2) depends on the percentage of empty squares, i.e., on the pattern distribution. This explains that the calculation of $\mathbf{M}$ in UF2DC is faster in dataset `susy`, with N=5,000,000 patterns, than in the other four datasets, where $N <$130,000 patterns. Therefore, UF2DC scales well with N, being suitable for large-scale classification problems. It also scales well with p, which does not influence the elapsed time after the LDA mapping, although the calculation of the covariance matrices $\mathbf{W}$ and $\mathbf{T}$, both of size p, in LDA might be slow for very high p.

Dataset	Name	N	p
Gas sensor array under dynamic gas	gas-co	4,178,504	17
Indiv. household electric power consum.	household	2,049,280	6
Dynamic features of virusshare execs.	dynamic	107,856	482
Relative location of CT slices	CT-slices	53,500	374
Appliances energy prediction	app-energy	19,735	26
Combined cycle power plant	combined-cycle	9,527	4
Communities and crime	com-crime	1,994	122
Airfoil self-noise	airfoil	1,503	5
Air quality	air-quality-NOx	1,230	8
Geographical origin of music	geo-long	1,059	72
Concrete compressive strength	compress-stren	992	8
Energy efficiency	energy-cool	768	7
Bike sharing	bike-day	731	31
Student performance	student-por	649	30
Istanbul stock exchange	stock-exchange	536	8
Facebook performance metrics	fb-metrics	500	20
Boston housing	housing	452	13
Auto MPG	auto-mpg	398	23
Yacht hydrodynamics	yacht-hydro	308	6
1985 Auto imports	automobile	194	66
Servo	servo	167	15
Concrete slump test	slump	103	7
Daily demand forecasting orders	daily-demand	60	12

Table 4.4: List of the 23 UCI regression datasets.

4.2 Regression

Table 4.4 compiles the 23 UCI regression datasets used to evaluate UF2DR, with N and p, which are high for some datasets (up to four million patterns and almost 500 inputs). In all the experiments, UF2DR uses $\eta = 5$ (five times more squares in the map than patterns) and $\mathscr{L} = 10$ (number of output levels in the LDA mapping, see section 3.1). The UF2DR was compared to the following five well-known regressors in the literature:

1. p**GSVR**: Gaussian kernel ε-support vector regression (LibSVM library v. 3.22), applied on the original p-dimensional patterns. This regressor was selected by its good performance [28]. The loss function uses ε=0.1 and the regularization parameter and inverse of the Gaussian kernel spread are tuned as pGSVC.

2. p**LSVR**: linear support vector regression (LibLinear library v. 2.3.20), also on the p-dimensional patterns, with L2 regularized L2-loss SVR (primal) solver and ε = 0.1. The regularization parameter is tuned as pGSVR. This regressor is included due to its speed, which makes it suited for large datasets.

3. p**GRNN**: generalized regression neural network (`newgrnn` function in the Matlab Neural Network Toolbox) on the p-dimensional patterns. This regressor was selected by its speed [67] and easy hyperparameter tuning, only the Gaussian spread, with 25 values between 0.001 to 50.

4. h**GRNN**: the GRNN trained on the h-dimensional patterns ($h = C-1$) mapped by LDA retaining the h main eigenvectors, tuned as pGRNN.

5. **2GRNN**: the GRNN trained on the 2D patterns mapped by LDA retaining only the two main eigenvectors, tuned as pGRNN.

In regression problems, the performance can be evaluated by several measurements. One of the most popular is the root mean squared error (RMSE):

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(o_i - z_i)^2} \tag{4.2}$$

where o_i is the true outcome, while z_i is the outcome predicted by the regressor, for the i-th data. The lower RMSE value, the more reliable prediction. Since the differences are

Dataset	*p*GSVR	*p*LSVR	*p*GRNN	*h*GRNN	2GRNN	UF2DR
bike-day	0.998	0.997	0.745	0.986	0.997	0.989
yacht-hydro	0.989	0.650	0.539	0.966	0.991	0.934
daily-demand	0.979	0.979	0.805	0.809	0.823	0.770
energy-cool	0.965	0.883	0.903	0.899	0.939	0.901
combined-cycle	0.947	0.928	0.943	0.931	0.941	0.911
fb-metrics	0.940	0.871	0.718	0.623	0.985	0.974
automobile	0.905	0.904	0.831	0.815	0.717	0.635
air-quality-NOx	0.872	0.800	0.856	0.834	0.835	0.796
auto-mpg	0.863	0.847	0.792	0.783	0.857	0.802
compress-stren	0.862	0.611	0.671	0.675	0.679	0.614
CT-slices	—	0.861	1.000	0.998	0.933	0.908
airfoil	0.858	0.508	0.781	0.782	0.638	0.616
housing	0.830	0.712	0.712	0.787	0.764	0.684
stock-exchange	0.785	0.785	0.708	0.730	0.772	0.651
servo	0.776	0.706	0.334	0.921	0.802	0.692
gas-co	—	0.687	—	—	—	0.737
com-crime	0.669	0.644	0.548	0.629	0.632	0.494
app-energy	0.406	0.161	0.373	0.262	0.187	0.052
slump	0.345	0.266	0.266	0.304	0.410	0.367
student-por	0.298	0.270	0.258	0.270	0.282	0.144
geo-long	0.272	0.160	0.231	0.208	0.104	0.033
household	—	0.248	—	—	—	0.243
dynamic	—	0.014	0.486	0.472	0.270	0.353
Average	0.766	0.630	0.643	0.699	0.693	0.622

Table 4.5: Squared correlation R^2 achieved by *p*GSVR, *p*LSVR, *p*GRNN, *h*GRNN, 2GRNN and UF2DR.

squared, the large difference values are weighted much more than the low values, so the RMSE penalizes the high differences. Note that RMSE measures the error instead of performance.

Another important performance metric is the Pearson correlation coefficient (denoted as R), defined as the sum of component-wise products of true and predicted outcomes, after subtracting their means, divided by the product of their variances:

$$R = \frac{\left[\sum_{i=1}^{N}(o_i-\overline{o})\right]\left[\sum_{i=1}^{N}(z_i-\overline{z})\right]}{\sqrt{\frac{1}{N}\sum_{i=1}^{N}(o_i-\overline{o})}\sqrt{\frac{1}{N}\sum_{i=1}^{N}(z_i-\overline{z})}}, \quad \overline{o} = \frac{1}{N}\sum_{i=1}^{N} o_i, \overline{z} = \frac{1}{N}\sum_{i=1}^{N} z_i, \tag{4.3}$$

The R values range from -1, that means that true and predicted outcomes are completely sign-reversed, to +1, that means perfect coincidence between true and predicted outcomes. The coincidence is calculated by discarding mean and variance factors. The correlation value is relevant because the dependence with the absolute values of the true outcome has been removed by subtracting the outcome mean and dividing by the outcome variance, so that R evaluates the coincidence between true and predicted outcomes independently from their specific values.

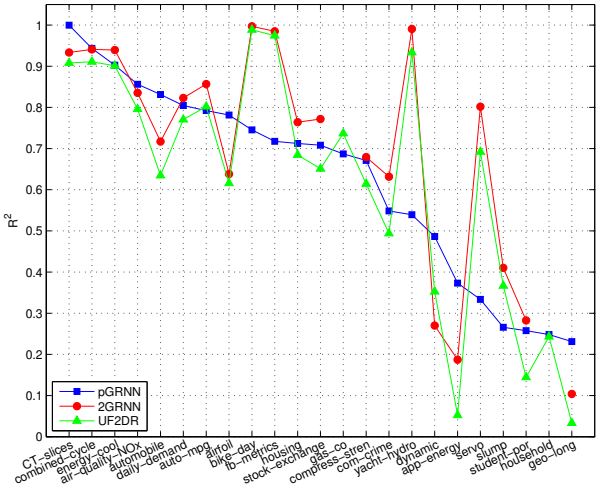

Figure 4.5: Values of R^2 achieved by *p*GRNN, 2GRNN and UF2DR.

Table 4.5 reports the squared correlation (R^2) between the true and predicted outputs achieved by the previous regressors and by UF2DR in each dataset, and the average R^2 over all the datasets excepting errors (last row). We used the R^2 instead of *RMSE* because it is a performance (not error) measurement, and because $0 \leq R^2 \leq 1$ while $0 \leq RMSE < \infty$, so that R^2 measures better how near the regressor is from the perfect prediction ($R^2 = 1$) in absolute terms. The dashed cells in Table 4.5 correspond to regressors and datasets which achieved memory (*p*GRNN, *h*GRNN and 2GRNN in `gas-co` and `household`) or time (*p*GSVR in `CT-slices`, `gas-co`, `household` and `dynamic`) errors.

The highest average R^2 is achieved by *p*GSVR (0.766). The *h*GRNN and 2GRNN achieve average R^2 about 0.69, outperforming *p*LSVR and *p*GRNN, which achieve about 0.63–0.64 using the original high-dimensional patterns. The average R^2 of UF2DR (0.622) is 0.144 below *p*GSVR, and the difference is below 0.15 in 11 of 19 datasets where *p*GSVR does not fail, so the 2D mapping does not reduce very much the performance. In the majority of the 8 remaining datasets (`daily-demand`, `automobile`, `compress-stren`, `airfoil`, `housing`, `app-energy`, `student-por` and `geo-long`), 2GRNN is also below *p*GSVR, but *h*GRNN is near *p*GSVR, so the decrease in performance is also caused by retaining 2 instead of *h* eigenvectors. Comparing UF2DR and *p*LSVR, the average R^2 is very similar (0.622 and 0.630, respectively).

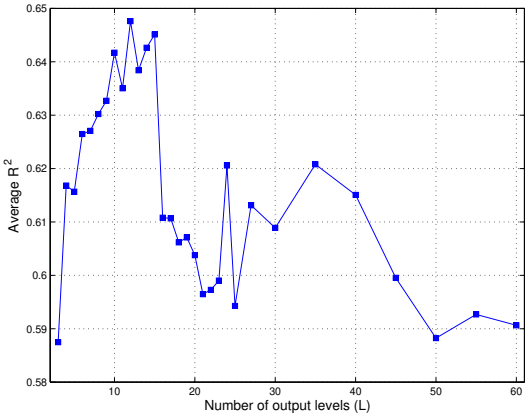

Figure 4.6: Average R^2 achieved by UF2DR using different numbers $\mathscr{L}$ of levels in the LDA mapping.

Figure 4.5 compares the R^2 achieved by *p*GRNN, 2GRNN and UF2DR. Since *p*GRNN and 2GRNN achieve memory errors in datasets `gas-co` and `household`, the associated circles in Figure 4.5 are missing, and the squares correspond to *p*LSVR. The 2GRNN and UF2DR are very near each other, and usually near *p*GRNN, and the exceptions (6 of 23 datasets) are due to retain only 2 eigenvectors. In some datasets (`app-energy`, `student-por` and `geo-long`), the R^2 of *p*GSVR is already very low, and the value of UF2DR is not much lower.

Figure 4.6 reports the average R^2 over all the datasets achieved by UF2DR using different numbers $\mathscr{L}$ of output levels (section 3.1). First, note that the vertical scale is very narrow (0.58–0.65), so the influence of $\mathscr{L}$ in the performance of UF2DR is low. Therefore, a default value can be used for $\mathscr{L}$ and a specific tuning is not required. With low $\mathscr{L}$ (about 3-10, left end of the plot), the average R^2 raises achieving the highest values in a narrow interval about $\mathscr{L}$=10-15 levels, so the selected default value $\mathscr{L} = 10$ is a good choice, whose performance is very near to the best one. The behavior fastly degrades for $\mathscr{L} > 16$ and does not raise again for larger $\mathscr{L}$, but the elapsed times increase because more covariance matrices must be calculated (eq. 3.3).

Table 4.6 reports the times spent by *p*GRNN, 2GRNN, *p*GSVR, *p*LSVR and UF2DR on the regression datasets. Only UF2DR and *p*LSVR are able to process all the datasets, while *p*GRNN and 2GRNN had memory errors in `gas-co` and `household`, and *p*GSVR had time errors in `gas-co`, `household`, `dynamic` and `CT-slices`. The times with asterisks are

estimated considering the number of tuning trials executed by *p*GSVR within 8 days (because it had time errors, see Table 4.5), although it could not be estimated in datasets `gas-co` and `household` (4 and 2 million patterns, respectively) because no tuning trial was finished. The times of UF2DR are much lower than the other regressors, and the difference with them raises fastly with N after dataset `com-crime` (1,994 patterns): 1-2 orders of magnitude with *p*LSVR, and 3-6 orders of magnitude with the remaining regressors.

Figure 4.7 plots the times spent by 2GRNN, UF2DR, *p*GSVR and *p*LSVR. In the smallest datasets (right part of the plot), the times of UF2DR are 1 and 2 orders lower than *p*LSVR and *p*GSVR-2GRNN, respectively. The difference raises in the largest datasets (left part of the plot) to 6 orders for *p*GSVR, 3 orders for 2GRNN, and 1-2 orders for *p*LSVR (2 orders in datasets `CT-slices` and `dynamic`, with high p). In the largest datasets `gas-co` and `household`, *p*LSVR is 10-15 times slower than UF2DR, which only spends 1-2 minutes. The time spent by UF2DR scales so well with $N > S_U^2/\eta$ because the map size s is upper bounded by $S_U = 300$, and because the time sensitivity with p is low, as discussed in section 4.1.

Dataset	*p*GRNN	2GRNN	*p*GSVR	*p*LSVR	UF2DR	Map
gas-co	—	—	—	1910.12	135.79	45.13
household	—	—	—	635.55	51.54	16.89
dynamic	2.1e+05	1.2e+04	9.2e+06*	5545.75	31.33	9.96
CT-slices	5.7e+04	1301.95	4.6e+06*	380.80	25.92	7.59
app-energy	305.00	132.94	1.7e+05	11.69	2.70	1.12
combined-cycle	37.15	34.22	4e+04	2.88	1.13	0.51
com-crime	18.29	9.89	182.96	5.33	0.47	0.18
geo-long	6.29	4.69	42.72	1.44	0.21	0.10
air-quality-NOx	3.52	3.38	279.54	0.55	0.22	0.12
airfoil	3.34	3.67	439.99	0.54	0.19	0.08
bike-day	3.11	2.93	7.96	0.50	0.11	0.05
student-por	2.94	2.74	17.07	0.49	0.10	0.04
compress-stren	2.84	2.83	112.40	0.42	0.16	0.06
energy-cool	2.49	2.46	93.58	0.32	0.12	0.07
fb-metrics	2.40	2.37	10.34	0.35	0.08	0.03
auto-mpg	2.29	2.33	7.21	0.30	0.06	2.68
stock-exchange	2.23	2.28	43.86	0.25	0.08	0.03
housing	2.21	2.24	14.45	0.26	0.07	0.03
automobile	2.20	2.48	2.85	1.13	0.06	0.02
yacht-hydro	1.98	2.00	6.97	0.15	0.04	0.02
servo	1.86	1.95	2.28	0.14	0.03	0.01
slump	1.79	1.86	1.30	0.10	0.02	0.01
daily-demand	1.78	1.83	0.40	0.07	0.02	0.01

Table 4.6: Elapsed time spent by *p*GRNN, 2GRNN, *p*GSVR, *p*LSVR and UF2DR, and by the map definition.

The last column (Map) in Table 4.6 reports the times spent by UF2DR to define the map of the whole dataset, with very low values which only overcome 10 s. in the two datasets

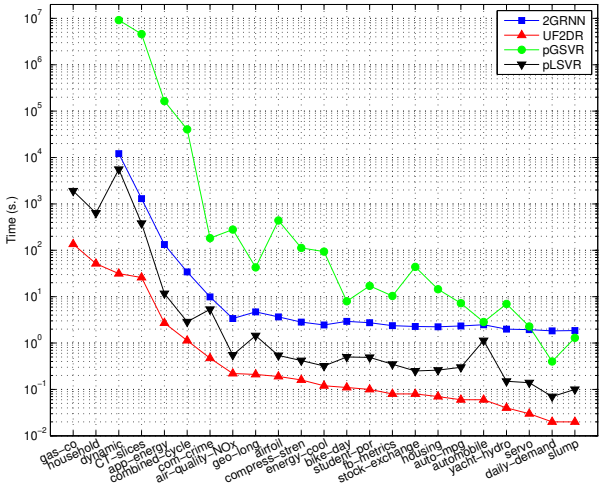

Figure 4.7: Times (log. scale) spent by 2GRNN, UF2DR, *p*GSVR and *p*LSVR.

with more than 2 million patterns (`gas-co` and `household`). These times are comparatively larger in some datasets (e.g. `auto-mpg`) with many empty squares, being lower than UF2DC in the smallest datasets due to the lack of median filtering, map resampling and image smoothing.

The two steps in the definition of the regression map are: 1) calculation of population **P** and output **O** matrices, which spend times below 0.1 excepting `household` and `gas-co`, with the longest time, 0.381 s.; and 2) calculation of the **M** matrix, which is also fast with times below 1 s. excepting `household` and `CT-slices`, spending only 0.25 s. in the largest dataset `gas-co` despite its high N.

4.3 Discussion

The proposed methods ultra-fast 2D classifier (UF2DC) and regressor (UF2DR) define a 2D map which visualizes the high-dimensional problem by:

1. Using linear discriminant analysis to project the data in 2D, and dividing in a grid of squares the region where the standardized 2D patterns are placed.

2. Defining the 2D map as a squared matrix where each element is the predicted value for the mapped patterns which are located in the square associated to that element.

In UF2DC, the predicted output for a square is the label of the class with the highest relative population of patterns located in that square, in order to manage possible class imbalances. In UF2DR, the predicted output for a square is the average output over the training patterns inside that square. In empty squares, the voting and average are performed over the smallest non-empty squared region.

The experimental work compares UF2DC and UF2DR with several well-known classifiers and regressors using high-dimensional and 2D mapped patterns with very large datasets (up to 5 million patterns and 791 inputs). The results show that:

1. The proposed methods perform an understandable 2D visualization of classification and regression datasets.

2. This visualization can be developed without a big loss in performance: comparing with the best method (pGSVC and pSVR), the loss is about 0.12 and 0.14 in kappa and R^2 in average, respectively, but only 0.02 and 0.008 comparing with pLSVC and pLSVR, that were the only classifier and regressor of the collection that could be executed on the largest datasets.

3. Both UF2DC and UF2DR are very fast, projecting datasets with 5 million patterns in just 1 minute and being 1,000 (resp. 10) times faster than the other classifiers (resp. regressors). Most of these methods required huge amounts of memory or could not process large datasets in a reasonable time.

CHAPTER 5

AUTOMATIC BEHAVIOR PREDICTION IN AUTISM SPECTRUM DISORDER

Machine Learning is an important topic in Artificial Intelligence that encloses methods (or models) for the automatic prediction of values using on data examples. A model learns to predict the output (or outcome) value as a function of the input data in a process called training, that uses a collection of pairs input-output examples. During training, the model changes the values of its parameters in order to predict an outcome near to the true value for the training data, i.e., to predict an output value for the training data near to the true value. The trained model is expected to generalize its predictions with reliability to new input data not used during training. In our case, each output to be predicted is a CBCL outcome (section 6.2), that have continuous values, so their prediction is named "regression" and the models used to predict them are named "regressors". Often, regressors have hyper-parameters, i.e. values that are not calculated during training but they must be set previously. These values may influence the reliability of the model prediction. Their values are usually set by trying several values and selecting the one with the most reliable prediction (see subsection 6.3 below) on a separate dataset, in a process named "hyper-parameter tuning".

As explained in section 6.2, for each CBCL outcome it is a common clinical practice to diagnose a participant as normal or pre-clinical/clinical, depending on whether the outcome value is below or above 60, or in normal, pre-clinical and clinical, when the value is below 60, between 60 and 70 and above 70, respectively. In order to measure the validity of the prediction to perform a diagnosis, the outcome value (S1-S8, internal, external and total)

predicted by the ML regressor is thresholded and converted in a class label: normal and clinical, thus defining a binary classification problem; or alternatively as normal, pre-clinical and clinical, thus considering three classes.

For the current study, we selected a collection of 26 regressors that provided the best performances in our previous exhaustive comparison [28]. The methods used in the current study are implemented in several languages:

1. The **R** Statistical and Computing Language [1], using regressors implemented by several packages [20].
2. The **Octave** Scientific Programming Language[2] .
3. The **Matlab** computing language, using the Statistics and Machine Learning Toolbox[3].
4. The **Python** programming language, with the `scikit-learn` Machine Learning module[4].

The hyper-parameter tuning was performed by trying several values and selecting the one with the lowest error (see subsection 6.3), using the so-called *grid-search* approach. For the regressors implemented in the R statistical computing language, the collection of values used for each hyper-parameter and data set were obtained by the function `getModelInfo` of the `caret`[5] R package [45]. These regressors (see Table 5.1) can be grouped in several families or groups of methods related, that are described in the following sections.

5.1 Linear regression

We tried several multi-variate linear regressors [8] as a baseline to be compared with the remaining ML regressors. The linear regressors calculate the output y to be predicted as the scalar product of a weight vector $\mathbf{w}$ and the input $\mathbf{x}$ pattern, plus a bias term b, so that $y(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$. The weight vector is calculated to minimize the least-mean-square error between the true and predicted outputs. Linear regressors have no tunable hyper-parameter. The regressors of this family that we use are the following:

[1] http://www.r-project.org
[2] http://octave.org
[3] http://mathworks.com
[4] http://scikit-learn.org
[5] http://topepo.github.io/caret

Family	Name	Regressor	Language
Linear	lreg	Linear regression	Octave
	lmreg		Matlab
	lm		R
	pylm		Python
Regularized	ridge	Ridge regression	Python
	lasso	Least absolute shrinkage and selection operator	Python
	enet	ElasticNet	Python
	sgd	Stochastic gradient descent	Python
Gaussian	gpr	Gaussian process regression	Python
Kernel	krr	Kernel ridge regression	Python
Support vector regression	svr	RBF kernel ε-SVR	Octave
	pysvr	RBF kernel SVR	Python
	lsvr	linear kernel SVR	Octave
Trees	M5	Regression tree	R
	tree		Python
	cubist	Tree + nearest neighbors	R
Ensembles	bagging	Bagging	Python
	rf	Random forest	R
	pyrf		Python
	ExtraTrees	Extremely randomized regression trees	R
	vote	Committee linear reg. + random forest	Python
	adaboost	Boosting	Python
	bstTree		R
	gbm	Gradient boosting machine	R
	pygbm		Python
Neural networks	pymlp	Multi-layer-perceptron neural network	Python

Table 5.1: List of the 26 regressors used to predict CBCL outcomes from SP2 groups.

– `lreg`, in Octave, implemented by the `LinearRegression` function of the `optim` package.

– `lmreg`, in Matlab, with the `fitlm` function of the Statistics and Machine Learning Toolbox.

– `lm`, in R with the `lm` function of the intrinsic `stats` package.

– `pylm`, in Python using the `LinearRegressor` object in the `sklearn.linear_model` module.

5.2 Regularized linear regression

This family of regressors uses a regularization term, alongside with the loss (difference between true and predicted output) that penalizes the complexity of the linear model. This term is usually measured as the norm $|\mathbf{w}|$, known as L_1, or squared norm $|\mathbf{w}|^2$, known as L_2, of the weight vector $\mathbf{w}$. Therefore, they have one tunable "regularization" hyper-parameter, that defines the relative weight of each component (loss and regularization). All the regressors of this family are implemented in Python by objects of the `sklearn.linear_model` module. This family includes the following regressors:

– `ridge`: ridge regression, implemented by the `Ridge` object. The regularization term is L_2, and the regularization hyper-parameter α is tuned with values from 0 to 500 with step 20.

– `lasso`: least absolute shrinkage and selection operator regression, that uses L_1 regularization, implemented by the `Lasso` object, tuning α with values from -1 to 3 with step 0.2.

– `enet`: elastic-net regression, implemented by the `ElasticNet` object, with both L_1 and L_2-norm regularization. So, two hyper-parameters α and l_1 are tuned, with values from 0.1 to 10 with step 0.5 and from 0 to 1 with step 0.2, respectively.

– `sgd`: stochastic gradient descent, implemented in Python by the `SGDRegressor` object of the `sklearn.linear_model` module. The regularization hyper-parameter α is tuned with 10 values equally spaced between 0.0001 and 0.5.

5.3 Support vector regression

These regressors are based on the well-known support vector machine [19] theory. We considered the following regressors of this family:

– `svr`: ε-support vector regression (ε-SVR), implemented by the Libsvm library [15] and accessed through its Octave binding as in the previous section 4.2. We used the

svr with radial basis function (RBF) kernel, tuning the regularization hyper-parameter C and the RBF kernel spread inverse γ, with values $\{2^i\}$ with i from -5 to 15 with step 2 and $\{2^i\}$ with i from -10 to 10 with step 2, respectively. The RBF kernel was used because it is the one that provides usually the best performance. The margin parameter ε, that defines the tolerance for the ε-insensitive loss function, takes the value ε=0.1. This parameter was not tuned because the SVR performance is not very sensitive to it.

- `pysvr`: ε-support vector regression implemented by the `SVR` object in the `sklearn.svm` module and tuning the same hyper-parameters as `svr` with the same values.
- `lsvr`: linear ε-support vector regression implemented by the Liblinear library [27] and accessed from Octave as in section 4.2. We tuned the regularization parameter C with values $\{2^i\}$, with i from -15 to 15 with step 2.

5.4 Regression trees

We included the following three variants of regression trees:

- `M5`: model tree [64] implemented by the Weka Data Mining Software [31] and accessed from R using the `M5Rules` function of the `RWeka` package [39], that provides an interface between R and Weka.
- `tree`: regression tree implemented in Python by the `DecisionTreeRegressor` object in the `sklearn.tree` module.
- `cubist`: M5 rule-based regressor with corrections based on nearest neighbors in the training set, implemented by the `Cubist` R package [46]. Its hyper-parameters are the number of training committees (with values 1, 10 and 20) and the number of neighbors for prediction (values 0, 5 and 9). These values are determined automatically by the `getModelInfo` of the `caret` package from the current dataset.

5.5 Ensembles

Several types of ensembles (collection of regressors) were also considered in this work. All of them are collections of regression trees, and include:

- `bagging`: bagging ensemble, whose trees are trained using different bootstrap samples of the training set. We used the Python implementation provided by the `BaggingRegressor` object of the `sklearn.ensemble` module, tuning the number of trees with values from 10 to 50 with step 5.
- **Random forest**, that trains each tree with a different boostrap sample of the training set considering a different subset of inputs, randomly selected [12]. Both properties add diversity to the ensemble, raising its performance compared to each single tree. We used two popular random forest versions:
  - `rf`, implemented in R by the `randomForest` package [48], tuning the number of randomly selected inputs (`mtry`) with 10 values between 2 and the number of inputs.
  - `pyrf`, implemented in Python by the `RandomForestRegressor` object in the `sklearn.ensemble` module, tuning the number of randomly selected inputs (`max_features`) with values from 2 to the number of inputs.
- `ExtraTrees`: ensemble of extremely randomized regression trees [35], implemented by the `extraTrees` R package. We tuned the number of trees (`n_trees`) with 10 values from 2 to the number of inputs.
- `vote`: voting committee composed by a linear regressor and a random forest, implemented in Python by the `VotingRegressor` object of the `sklearn.ensemble` module. Since this object is for regression problems, it replaces the voting by an averaging among the regressors in the ensemble.
- **Boosting**: ensemble of regression trees that are trained using a weighted training set, where the weight of each pattern is given by the error of the previous tree on that pattern. We used the following two versions:
  - `adaboost`: adaptive boosting ensemble of regression trees with linear loss, implemented in Python by the `AdaBoostRegressor` object of the `sklearn.ensemble` module, that implements the Adaboost.R2 algorithm [23]. We tuned the number of regression trees (`n_estimators`) and the learning rate (`learning_rate`) with values from 10 to 50 with step 10, and from 0.1 to 2 with step 0.5, respectively.

  - `bstTree`: boosting ensemble of componentwise linear, smoothing splines, regression trees implemented by the `bst` R package [82]. The training uses gradient boosting to optimize the squared error loss function and Gaussian family.

- **Gradient boosting machine** [32], including:

  - `gbm`: stochastic generalized gradient boosting machine implemented by the `gbm` R package [38] with a Laplace distribution. The tunable hyper-parameters are the maximum depth of input interactions (`interaction.depth`), with values from 1 to 10, and the number of trees for prediction (`ntrees`), with values from 50 to 500 with step 50.
  - `pybgm`: Python implementation using the `GradientBoostingRegressor` object in the `sklearn.ensemble` module tuning the number of trees (`n_estimators`), with values from 50 to 200 with step 50, and the maximum tree depth (`max_depth`), with values from 1 to 9 with step 3.

5.6 Neural networks

The only neural network that we considered in this research was `pylm`, the multi-layer perceptron neural network implemented in Python by the `MLPRegressor` object of the `sklearn.neural_network` module. We used one hidden-layer, tuning its number of neurons with values from 10 to 200 with step 20.

5.7 Other regressors

Additionally, we used the following methods:

- `gpr`: Gaussian process regression, implemented in Python by the `GaussianProcessRegressor` object in the `sklearn.gaussian_process` module, using a kernel achieved by suming `DotProduct` and `WhiteKernel` objects.
- `krr`: kernel ridge regression, implemented in Python by the `KernelRidge` object in the `sklearn.kernel_ridge` module. This regressor learns a function similar to ε-SVR, with a radial basis function (RBF) kernel, but using a closed-form expression to calculate the trainable parameters by minimizing the squared error loss with L_2

regularization instead of the ε-insensitive function of SVR. The hyper-parameters α (L_1-regularization) and σ (RBF kernel width) are tuned with values from 0.1 to 2 with step 0.5 and $\{2^i\}$ with i from -10 to 11 with step 2.

The whole collection of regressors listed above has been used to predict automatically the 11 CBCL outcomes listed in Table 6.2 using the 6 groups of SP2 items listed in Table 6.1, as we will report in the next chapter.

CHAPTER 6

RESULTS AND DISCUSSION OF BEHAVIOR PREDICTION

Before the presentation of results and discussion, we will describe the sample data (section 6.1) and the questionnaires used to perform the prediction (section 6.2). Then, we will describe the experimental settings (section 6.3) and the performance metrics used in the experimental work (section 6.4). The results will be presented in terms of reliability in regression in section 6.5, as well of classification into normal, pre-clinical and clinical diagnostics in section 6.6. The chapter finalizes in section 6.7 with the discussion of the results.

6.1 Participants

The available data include 72 children and adolescents (21 females), aged between 6 and 14 years (mean = 7.83 years; SD = 2.80 years), diagnosed with ASD following the criteria established by the Diagnostic and Statistical Manual of Mental Disorders in its revised fourth version (DSM-IV-TR) or fifth version (DSM-5). Participants were part of a larger research project, leaded by the Group of Genomic Medicine, studying the association between phenotype and genotype characteristics in ASD. Qualified clinicians who were part of the research team confirmed the ASD diagnosis for research purposes, using the following two questionnaires:

ADI-R: autism diagnostic interview revised [65], a clinical interview useful for the ASD diagnosis. It is focused on the assessment of core symptoms of the disorder, including

sensory processing. This parent-report questionnaire consists of 93 questions focused on three main domains related to ASD diagnosis: language/communication, reciprocal social interactions and restricted, repetitive, and stereotyped behaviors and interests. Questions are focused on the clinical history of the participant and their current situation (recent months prior to the assessment).

ADOS: autism diagnostic observation schedule-2 [51], used to assess clinical traits of ASD such as communication, social interaction and play or imaginative use of materials. It consists of five modules (T, 1, 2, 3 and 4) that are applied according to the language skills and chronological age of the participant. Each module is composed of a set of activities and, according to participant's performance, the evaluator observes and confirms the presence of social and communicative behaviors relevant to the ASD diagnosis.

All parents who agreed to voluntarily participate gave the written informed consent, obtained in accordance with the Declaration of Helsinki. The studies involving human participants were reviewed and approved by the Research Ethics Committee of Santiago-Lugo. Written informed consent to participate in this study was provided by the participants' legal guardian or next of kin.

6.2 Questionnaires

The **Sensory Profile 2** (SP2) child record form 3:0-14:11 (in Spanish language) assess the sensory processing patterns of the sample [25]. This is a parent-report questionnaire consistent in 86 items scored on a five-point Likert Scale (1 = "Almost Never", 2 = "Occasionally", 3 = "Half of the time", 4 = "Frequently", 5 = "Almost Always") and a Zero-point possibility when the item is not applicable for the child. Items can be classified into six different Sensory sections (Auditory Processing, Visual Processing, Touch Processing, Movement, Body Position and Oral Processing) and three Behavioral sections (Conduct, Social-emotional and Attentional) or into four different quadrants (Seeking, Avoiding, Sensitivity and Registration). Table 6.1 reports the list of items included by each set, alongside with the Touch group of items and the Total group, that includes all the items of the questionnaire.

The **Child Behavior Check-List** for ages 6-18 (CBCL/6-18) was used to assess behavioral and emotional problems in the sample [1]. It is a parent-completed questionnaire consistent in 113 items rated on a three-point Likert scale (0 = Not True (as far as you know), 1 =

Input	no.	Items in SP2 questionnaire
Avoiding	20	1,2,5,15,18,58,59,61,63,64,65,66,67,68,70,71,72,74,75,81
Registration	22	8,12,23,24,26,33,34,35,36,37,38,39,40,53,54,57,62,76,79,80,85,86
Seeking	19	14,21,22,25,27,28,30,31,32,41,48,49,50,51,55,56,60,82,83
Sensitivity	19	3,4,6,7,9,13,16,19,20,44,45,46,47,52,69,73,77,78,84
Total	86	1-86
Touch	11	16-26

Table 6.1: List of the 6 groups of items selected from the SP2 questionnaire: total, touch and quadrants avoiding, seeking, registration and sensitivity, with their numbers of items (no.) and the items of each set.

Sometimes or Somewhat True, or 2 = Very True or Often True). This instrument is composed of eight subscales (S1-S8), two main domains (Internal and External) and a principal scale of overall impairment (Total), listed in Table 6.2. The syndrome subscales "Anxious/Depressed" (subscale 1, S1), "Withdrawn/Depressed" (subscale S2) and "Somatic Complaints" (S3) conform the "Internalizing Domain" scale. The subscales "Rule Breaking" (S7) and "Aggressive Behavior" (S8) sum into the "Externalizing Domain" scale. On the other hand, "Social Problems" (S4), "Thought Problems" (S5) and "Attention Problems" (S6) quantify into the "Total Problems" scale with the remaining behavior problems. Raw scores for each scale were converted to standardized T-scores based on Spanish standards [26] and bounded between 0 and 100, considering the age (6–11 and 12–18 years) and gender of participants separately.

S1	Anxious/depressed	S5	Thought problems	Internal	Internalizing problems
S2	Withdrawn/depressed	S6	Attention problems	External	Externalizing problems
S3	Somatic complaints	S7	Rule-breaking behavior	Total	All the CBCL outcomes
S4	Social problems	S8	Aggressive behavior		

Table 6.2: List of the 11 groups of outcomes (or scales) selected from the CBCL questionnaire to be predicted using ML methods.

Figure 6.1 shows the box plots that represent the distributions of values for the 11 CBCL outcomes in Table 6.2 over the 72 participants. Although each outcome is a scale score from 1 to 100, the available data only exhibit values above 35. For each CBCL outcome, a value below 60 corresponds to a normative (normal) behavior, while values between 60 and 70 correspond to pre-clinical behavior (i.e., near to require clinical treatment) and values above 70 correspond to a clinical diagnosis, requiring a treatment. This is a case with three diagnostics or classes: normal, pre-clinical and clinical. As well, sometimes it is considered that an

outcome below 60 is normal and above 60 is pre-clinical and clinical, thus having only two classes.

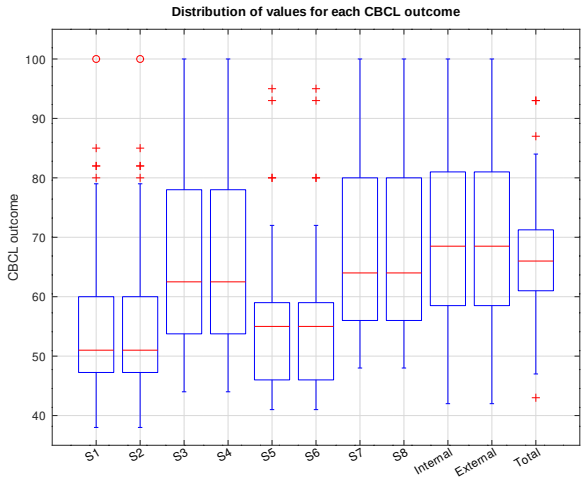

Figure 6.1: Box plots of the 11 CBCL outcomes to be predicted. The blue box represents the 25%-75% percentiles, the red horizontal line is the median, the blue whiskers are the data outside this interval and the red crosses are outliers.

The 6 groups of SP2 items in Table 6.1, combined with the 11 CBCL outcomes to be predicted in Table 6.2, give 66 datasets. We also study the influence of the gender, so each set was analyzed with and without gender and the final number of sets was 6x11x2=132 sets.

6.3 Experimental setting

In order to evaluate the performance of the trained regressors for the prediction of the CBCL outcomes, we used the classical K-fold cross validation. This method consists on dividing the available data into K partitions or folds. Some of them are devoted to train the model, and the remaining folds are used to test the trained model. Thus, the performance is evaluated over data that have not been used during training, being a realistic measurement of the reliability in the prediction and avoiding optimistic bias. The training/test cycle is repeated K times (often named as trials), what ensures that each fold is used once for test and $K-1$ times for training.

For those regressors with hyper-parameters, their tuning was performed by evaluating the regressor performance with several combinations of values for the hyper-parameters and selecting the combination that performs better. The evaluation of this performance requires a third dataset (different to training and test datasets) in order to avoid optimistic biasings. This dataset is often named "validation set". Overall, having K partitions, the process is repeated K trials: in the k-th trial, with $k = 1, \ldots, K$, the model is trained using a part of the K folds, that compose the training set on the current trial, with a certain combination of hyper-parameter values. Afterwards, the trained model is tested on one or several validation folds (that compose the validation set of this trial).

This process is repeated K trials, and the averaged performance over the K validation sets is calculated. Then, this training-test cycle is repeated for all the combinations of hyper-parameter values, and the combination with the best average performance over the validation sets is selected. Finally, the model is trained on the training and validation sets for the first trial using the selected combination of hyper-parameter values, and tested on the test set of the first trial (composed by the folds that were not included in the training nor in the validation sets), achieving a first test performance. The process is repeated for the K trials, and the average performance over the K trials is calculated. This is the performance of the model estimated by the K-fold cross-validation method after hyper-parameter tuning. This estimation is performed using test data not included on the training nor validation sets, and therefore they were not used nor for training nor for hyper-parameter tuning, thus discarding optimistic biasing caused by overfitting.

An example is 4-fold cross validation, where $K = 4$ and there are 2 training partitions, 1 validation and 1 test partition, with 4 trials. The first trial uses partitions 1,2 for training, partition 3 for validation and partition 4 for test. Second trial uses partitions 2,3 for training, 4 for validation and 1 for test, and so on. There are also alternative configurations, being also $K = 5$ and $K = 10$ other popular settings. When the amount of data is very small, as in our case (only 72 data, one for each participant), it is very difficult to split the data into K partitions, so an extreme case of K-fold cross-validation is used, the so-called leave-one-out (LOO) cross-validation, where $K = N$ being N the number of data. In this setting, there are N trials, and in each trial one data is devoted for test, and the remaining $N - 1$ data are splitted into training and validation sets for the hyper-parameter tuning. The process is repeated N times: each time the regressor, trained with the training and validation sets ($N - 1$ data) with the best combination of hyper-parameter values, gives a prediction for the remaining participant.

After repeating this process N trials, the performance is calculated by comparing the true and predicted CBCL outcomes for the N participants.

6.4 Performance metrics

Section 4.2 already described the two most relevant performance metrics for regression problems: root mean squared error (RMSE) and correlation coefficient (R). The classical definition of Colton [18] for the intervals of correlation values and their significance are listed in Table 6.3.

R	Correlation
$R < 0.15$	Nothing at all
$0.15 \leq R < 0.5$	Bad to moderate
$0.5 \leq R < 0.75$	Moderate to good
$0.75 \leq R$	Very good to excellent

Table 6.3: Intervals defined by Colton [18] for the correlation R.

Alongside with RMSE and R, we will also consider the mean absolute error (MAE), defined as the mean absolute difference between true and predicted outcomes:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|o_i - z_i| \tag{6.1}$$

Another popular error measurement is the weighted absolute percentage error (WAPE), that expresses in % the ratio of the absolute difference between true and predicted outcome divided by the true value.

$$WAPE(\%) = \frac{100\sum_{i=1}^{N}|o_i - z_i|}{\sum_{i=1}^{N} o_i} \tag{6.2}$$

In binary classification, such as the discrimination between normal and clinical participants (see section 6.2), is usual to consider one class as "positive" and the other as "negative". The classification process is formulated as the detection of positive cases. In our case, "clinical" and "normal" are the positive and negative classes, respectively. In order to evaluate performance metrics, it is usual define the number of:

– True negatives (TN): participants that are normal (negative class) and are predicted by the ML method as normal. Since the regressor prediction is right, this number is labeled as "true".

– False positives (FP): participants that are normal but are predicted as clinical (positive), so they are classification errors and the number is named "false".

– False negatives (FN): participants that are clinical but are predicted as normal (negative), being again errors (false).

– True positives (TP): participants that are clinical and are predicted as clinical (positive), so they are well predicted (true).

These four values are usually arranged in the so named "confusion matrix", shown in Table 6.4. The numbers of participants correctly classified are inside the matrix diagonal, and the numbers outside this diagonal count participants wrongly classified. Note also that rows identify the true class label, while columns identify the class label predicted by the ML method (in our case, achieved by thresholding in 60 the CBCL outcome predicted by the regressor). From the confusion matrix, it is very easy to calculate the most popular performance metric for classification: the accuracy (denoted as Acc), that measures the percentage of participants correctly classified. This metric (in %) can be calculated for a binary classification problem as:

	Normal	Clinical
Normal	True negative (TN)	False positive (FP)
Clinical	False negative (FN)	True positive (TP)

Table 6.4: Confusion matrix for the current binary classification problem in normal-clinical participants.

$$Acc(\%) = \frac{100(TN+TP)}{TN+FP+FN+FP} \tag{6.3}$$

The accuracy is a metric biased by the unbalance between classes, so that when one class has much more participants than the other, a classifier that predicts the majoritary class for all the patterns may still exhibit a high accuracy. This, however, does not mean that classifier behaves well, in fact its performance is fairly poor. In order to consider the class unbalance, the most popular metric is the Cohen kappa score [14], denoted as κ, that we already defined and used in section 4.1. This metric measures the agreement between the true class label and

the label predicted by the ML method discarding the agreement by chance, e.g. because the classes are unbalanced. For binary classification problems, the kappa statistic (in %) can be calculated alternatively to eq. 4.1 using the elements of the confusion matrix in the following way:

$$\kappa(\%) = \frac{100 \cdot 2(TP \cdot TN - FN \cdot FP)}{(TP+FP)(FP+TN)+(TP+FN)(FN+TN)} \tag{6.4}$$

Table 6.5 reports the kappa intervals defined by Blackman [11] and their meanings in terms of classification performance, analogously to the Colton intervals for the correlation R. We will analyze these results for each CBCL outcome. Other performance metrics for binary classification problems, that are very popular in the health care field, are:

– Sensitivity (Se): percentage of true positives (TP) with respect to the total number of positives (FN+TP, sum of second row in the confusion matrix), defined in % as:

$$Se(\%) = \frac{100TP}{FN+TP} \tag{6.5}$$

– Specificity (Sp): percentage of true negatives (TN) with respect to the total number of negatives (TN+FP, sum of first row in the confusion matrix), defined in % as.

$$Sp(\%) = \frac{100TN}{TN+FP} \tag{6.6}$$

Kappa (%)	Agreement
$\kappa < 20$	Poor
$20 \leq \kappa < 40$	Weak
$40 \leq \kappa < 60$	Moderate
$60 \leq \kappa < 80$	Good
$80 \leq \kappa$	Very good

Table 6.5: Intervals defined by Blackman [11] for the Cohen kappa score.

In the three-class classification of the CBCL outcomes as normal, pre-clinical and clinical, the sensitivity and specificity are not considered and kappa (defined for non-binary classification by eq. 4.1 in section 4.1), accuracy and confusion matrix are the only performance metrics considered.

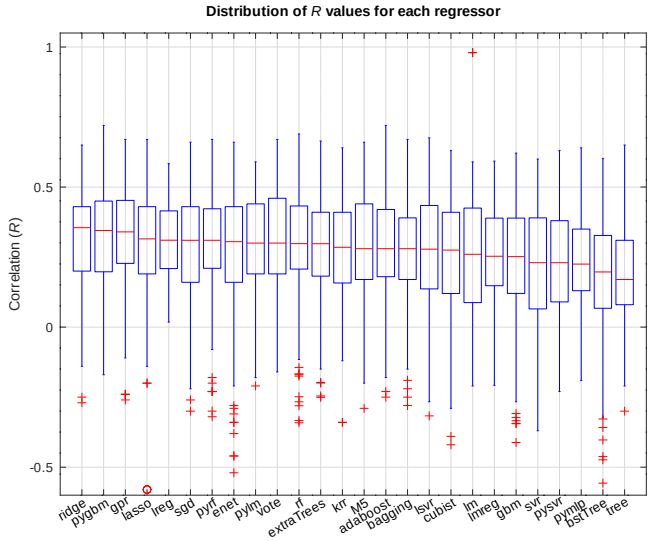

Figure 6.2: Box plots of the *R* values achieved by each regressor over all the groups of SP2 items and CBCL outcomes.

6.5 Prediction of CBCL outcomes

Figure 6.2 shows the box plots of the correlation (*R*) values achieved by the ML models listed in chapter 5 over all the datasets for the prediction of CBCL outcomes, sorted by decreasing medians (red line inside the blue box). All the regressors achieve low correlations, being the highest value about 0.6-0.7, with only an outlier (red cross) for regressor `lm` with *R*=0.98. As well, other methods in the linear and regularized linear regression families (`ridge`, `lasso`, `lreg`, `sgd` and `enet`) also achieve good results in general over all the outcomes and groups. Other regressors with good performance are Gaussian process regression (`gpr`) and two ensembles: `pygbm` and `pyrf`.

Table 6.6 reports the Friedman rank [66] of regressors for the prediction of CBCL outcomes. This rank is decreasing with performance, so the lower rank, the higher performance. The rank of a regressor is the average position achieved by it over a collection of datasets (in our case, 132 datasets=6 SP2 groups x 11 CBCL outcomes x 2 genders) when we sort the regressors by decreasing performance (in the regression case, by decreasing *R*) in each dataset.

Position	Regressor	Family	Rank	Position	Regressor	Family	Rank
1	ridge	Linear	8.2	14	krr	Linear	13.4
2	gpr	Kernel	8.5	15	lmreg	Linear	13.5
3	pygbm	Ensemble	9.9	16	lsvr	Support	13.6
4	lasso	Linear	10.4	17	ExtraTrees	Ensemble	13.7
5	lreg	Linear	10.6	18	M5	Tree	13.9
6	vote	Ensemble	10.7	19	bagging	Ensemble	14.3
7	pyrf	Ensemble	11.6	20	cubist	Ensemble	15.5
8	rf	Ensemble	11.8	21	pymlp	Network	16.3
9	pylm	Linear	12.8	22	gbm	Ensemble	16.7
10	lm	Linear	12.9	23	pysvr	Kernel	17.4
11	adaboost	Ensemble	13.0	24	svr	Kernel	17.5
12	enet	Linear	13.1	25	bstTree	Ensemble	18.7
13	sgd	Linear	13.2	26	tree	Tree	19.0

Table 6.6: Friedman rank of regressors for the R achieved in the prediction of CBCL outcomes.

Thus, the regressors that achieve the first (lower) positions achieve the higher R values and therefore the most reliable predictions.

For a given collection of classification problems, the best performance is not achieved by the same regressor in all the datasets of the collection. Therefore, usually the rank of the globally best regressor is not near 1. For example, if we have four regressors and three datasets, we sort the regressors by decreasing R for each dataset. Then, for each regressor we register its position on the sorted list for each dataset. The Friedman rank of a regressor is its average position over the three datasets. Specifically, let be the best regressor achieves position 1, 4 and 2 in the three datasets. Then, its Friedman rank is (1+4+2)/3=2.3 so in average over the three datasets, it achieves the 2.3th position. The rank is not 1, but is the lower rank, so it is the globally best regressor.

In Table 6.6, the best regressor (`ridge`) achieves in average over the 132 datasets the 8.2th position. The Gaussian process regressor (`gpr`) achieves a similar rank (8.5). There is a group of regressors with nearby ranks between 10 and 11 (see Figure 6.3): `pygbm`, `lasso`, `lreg` and `vote`, while `pyrf` and `rf` are about 12. The following group includes 11 regressors, from `pylm` to `bagging`, with ranks about 13-14. The last regressors in the ranking have ranks above 15, from `cubist` to `tree`.

Considering the regressor families, linear is the family with the best regressor (`ridge`), and the remaining linear models (`lasso`, `lreg`, `pylm`, `lm`, `enet`, `sgd`, `krr` and `lmreg`)

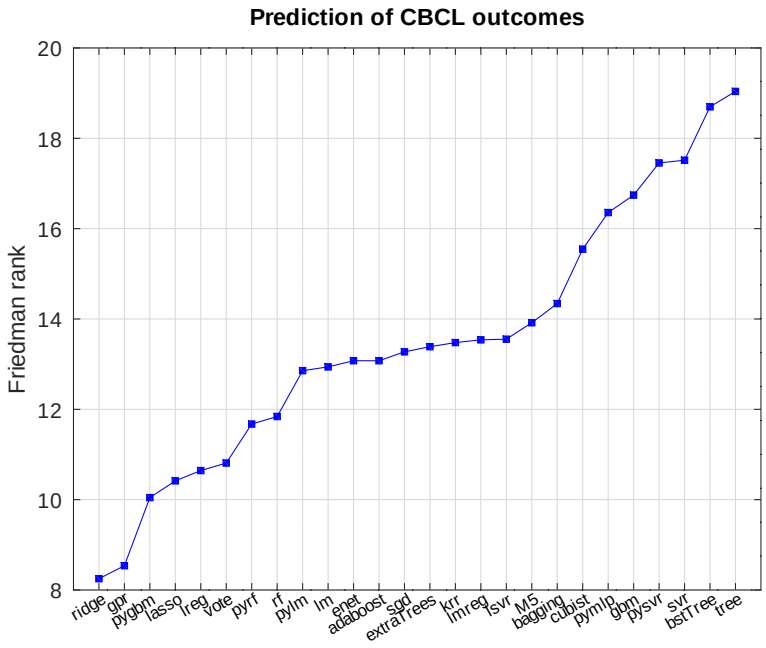

Figure 6.3: Plot of the regressor's Friedman rank.

achieve ranks in 10-14. The `pygbm` is the best ensemble (3rd position, with rank 9.9), outperforming `gbm` (16.7). Other good ensembles are `vote` (10.7), random forests (`pyrf` and `rf`, about 11) and `adaboost` (13). Surprisingly, the linear kernel support vector regression (`lsvr`) outperforms the RBF kernel version (`svr`). The bad ranks of `M5` (13.9) and `cubist` (15.5) are also surprising, given their good performance in [28].

Table 6.7 shows the best correlation R achieved by some regressor for each CBCL outcome (in columns, see Table 6.2) and for each SP2 group (in rows, see Table 6.1) with and without gender (G), highlighting in bold the best R for each outcome. Following the Colton criteria, these results can be considered "from very good to excellent" (VGE, that means $R > 0.75$) for the external outcome of the CBCL questionnaire, and "moderate to good" (MTG, $0.5 \leq R \leq 0.75$) for outcomes S1, S2, S4, S5, S6, S8, internal and total. The outcomes S3 and S7, with R under 0.5, can be considered "from bad to moderate" (BTM) correlations. Several comments may be performed by analyzing each outcome separately:

Input		CBCL outcome										
		S1	S2	S3	S4	S5	S6	S7	S8	Int.	Ext.	Total
Touch		0.53	0.30	0.21	0.68	**0.57**	0.46	0.27	0.47	0.48	0.40	**0.67**
	G	0.57	0.30	0.21	0.66	0.56	0.50	0.37	0.47	0.49	0.40	0.66
Seek.		0.11	0.09	**0.44**	0.34	0.39	0.50	0.24	0.48	0.16	0.42	0.34
	G	0.10	0.09	0.35	0.34	0.38	**0.51**	0.29	0.48	0.14	0.33	0.31
Avoid.		**0.72**	0.57	0.28	0.49	0.26	0.23	0.37	0.52	**0.63**	0.54	0.56
	G	0.65	0.57	0.28	0.48	0.26	0.12	**0.39**	**0.55**	**0.63**	0.51	0.45
Sens.		0.41	0.36	0.20	0.51	0.41	0.40	0.28	0.37	0.43	0.37	0.47
	G	0.43	0.36	0.26	0.50	0.41	0.37	0.35	0.36	0.43	0.36	0.54
Reg.		0.68	0.51	0.34	**0.72**	0.38	0.39	0.24	0.48	**0.63**	0.39	0.58
	G	0.69	**0.58**	0.35	**0.72**	0.37	0.43	0.32	0.48	0.62	0.42	0.61
Total		0.67	0.48	0.36	0.61	0.49	0.46	0.30	0.50	0.60	**0.98**	0.57
	G	0.68	0.51	0.36	0.63	0.51	0.45	0.30	0.44	0.61	**0.98**	0.56

Table 6.7: Best correlation (R) achieved by some regressor for each outcome of the CBCL questionnaire (by columns) and for each group of items in the SP2 questionnaire (touch, seeking, avoiding, sensitivity, registration and total, by rows) without and with gender (G). The best R for each outcome is in bold.

S1 (anxious/depressed). The best result is achieved by group avoiding (R=0.72) and, slightly lower, registration and total, but the remaining groups (touch and specially sensitivity and seeking) are not able to predict their values. Comparing R with and without gender, the difference is small excepting avoiding, where the gender reduces the performance, and touch, where the gender slightly raises the performance.

S2 (withdrawn/depressed). The highest R is 0.58, much below S1 but still "moderate to good", and is achieved using registration and avoiding groups, while the remaining groups are very far (total about 0.5, sensitivity and touch about 0.3 and seeking about 0.1). The gender does not change the results excepting by 0.07 in registration.

S3 (somatic complaints). The best prediction for this outcome (R=0.44) is already "from bad to moderate" (below 0.5) using seeking, while the remaining outcomes are even poorer: registration and total about 0.3 and touch, avoiding and sensitivity about 0.2. The gender has relevance in group's seeking and sensitivity, with differences of 0.09 and 0.06, respectively.

S4 (social problems). The prediction is "moderate to good" (R=0.72) using registration, while the remaining groups are much below: touch and total about 0.6 and avoiding and reg-

istration about 0.5, without influence of gender (the highest difference with and without gender is 0.02).

S5 (thought problems). The correlation is low (0.57) using touch, while the remaining groups are much below: total about 0.5, sensitivity about 0.4, seeking and registration about 0.3 and avoiding about 0.2, with no gender influence.

S6 (attention problems). The best performance is again low (0.51 using seeking), while the remaining groups are about 0.4 (touch and total), 0.3 (sensitivity) or 0.2 (avoiding, with high gender influence, 0.11).

S7 (rule-breaking behavior). The correlation is "from bad to moderate" (0.39 using avoiding), and the other groups about 0.2-0.3. The gender influences touch (0.1), registration (0.08), sensitivity (0.07) and seeking (0.05).

S8 (aggressive behavior). All the groups excepting sensitivity achieve correlations about 0.4-0.5, being the best value (0.55) achieved by avoiding. The gender only influences total (0.06).

Internal (internalizing). The groups avoiding (0.63), registration (0.63) and total (0.61) achieve values "moderate to good", while touch and sensitivity are about 0.4 and seeking performs poorly. The gender has no influence in any group.

External (externalizing). The best correlation (0.98), associated to an almost perfect prediction, is achieved by group total, while the remaining groups are very far 0.3-0.5. Gender only influences seeking (0.09).

Total (all the CBCL outcomes). Touch achieves a good correlation value (0.67), while the remaining groups are also about 0.3-0.5, with differences by gender in avoiding (0.09) and sensitivity (0.07).

Overall, the avoiding group is the most reliable in anxious/depressed (S1), aggressive behavior (S8), and internalizing problems (Table 6.7). The registration group best predicts withdrawn/depressed (S2), social problems (S4), and internalizing (same R as avoiding). Touch processing best predicts thought problems (S5), rule-breaking (S7, with low R=0.44), and total problems. The seeking group achieves low R=0.44 in somatic complaints (S3) and R=0.51 in attention problems (S6), so it is not very related to any of the problems. The sensitivity

group never gets the best reliability in prediction. Table 6.8 compiles the best SP2 group and R for each CBCL outcome, alongside with the second best group and R when it was near to the best value (e.g. S2 and internal). In 5 of 11 outcomes avoiding achieved the best result. Touch and registration are listed four and three times, respectively, while seeking is only the best in S3.

Outcome	Best SP2 group	R	Outcome	Best SP2 group	R
S1	Avoiding	0.72	S7	Avoiding	0.39
S2	Registration	0.58		Touch	0.37
	Avoiding	0.57	S8	Avoiding	0.55
S3	Seeking	0.44	Internal	Avoiding	0.63
S4	Registration	0.72		Registration	0.63
S5	Touch	0.57	External	Total	0.98
S6	Seeking	0.51	Total	Touch	0.67
	Touch	0.50			

Table 6.8: Best SP2 group and R for each CBCL outcome.

When a group achieves the best R for an outcome, we can think that they are related, more tightly when R is high. This allows to test whether the *a priori* expected relations between SP2 groups and CBCL outcomes (see Table 2.1) were indeed found in the experimental work. Table 6.9 tests the coincidence between expected and found relations by reporting the R when some SP2 group (by rows) achieved the best or second best R for a CBCL outcome (by columns), so that both are related. Empty cells are for pairs group-outcome with sub-optimal R. Of the three SP2 groups that were candidate to be related (avoiding, seeking and touch) with outcomes in Table 2.1 (S1, S2, S7, S8, internal and external), only the group avoiding was related in the 5 former outcomes, although in S2 it achieves the second best R. The R values report that avoiding was related mainly with S1 and internal, and less strongly to S2 and S8. Touch is related only in outcome S7 with low R value. Seeking is not related to any of the expected outcomes. Interestingly, external was found very related to the SP2 total group, with high R, instead to the SP2 candidates.

The bar plot in Figure 6.4 represents the best correlation achieved by some regressor for each CBCL outcome, sorted by decreasing values. External is the only outcome with correlation is "from very good to excellent" (above 0.75). The majority of the remaining outcomes (S1, S4, total, internal, S2, S5 and S8) are in the range 0.5-0.75 so they are "moderate to good". Finally, outcomes S3 and S7 are below 0.5, so they are "from bad to moderate".

CBCL \ SP2	R with expected outcome Avoiding	Seeking	Touch	Unexpected Total
S1	0.72			
S2	0.57			
S7	0.39		0.37	
S8	0.55			
Internal	0.63			
External				0.98

Table 6.9: Best R of the SP2 groups expectedly related to CBCL outcomes (see text for details).

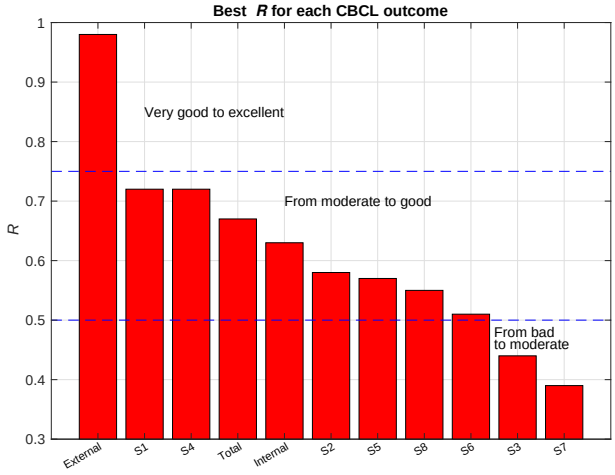

Figure 6.4: Best correlation achieved by some regressor for each CBCL outcome.

As we saw in Table 6.7 above, the influence of gender on the prediction is marginal, with very small differences (about 0.02) in the best R for each outcome with and without gender, although this influence might be implicit on the remaining items of the questionnaire. Table 6.10 reports that the difference in R between including or not gender among features for each SP2 group and CBCL outcome. The majority of the absolute values are below 0.05, and only in three cases (touch+S7, avoiding+S6 and avoiding+total) they overcome 0.1, achieving a maximum value of 0.11. Besides, the few cases where the difference is high are not the ones with the best performance. These values suggest that gender can not be considered a significant feature for the prediction of CBCL outcomes. We also performed a Wilcoxon

	CBCL outcome										
Input	S1	S2	S3	S4	S5	S6	S7	S8	Int.	Ext.	Total
Touch	0.04			-0.02		0.04	**0.10**		0.01		-0.01
Seek.	-0.01		**-0.09**		-0.01	0.01	0.05		-0.02	**-0.09**	-0.03
Avoid.	**-0.07**			-0.01		**-0.11**	0.02	0.03		-0.03	**-0.11**
Sens.	0.02		**0.06**			-0.03	**0.07**	-0.01		-0.01	**0.07**
Reg.	0.01	**0.07**			-0.01	0.04	**0.08**		-0.01	0.03	0.03
Total	0.01	0.02		0.02	0.02	-0.02		**-0.06**	0.01		

Table 6.10: Difference between the best correlation for each SP2 group (by rows) and CBCL outcome (by columns) with and without gender (positive values mean that gender provides higher *R*). The values above 0.05 are in bold, and the values below 0.01 are removed for clarity.

Outcome	*R*	Colton	MAE	WAPE	RMSE	Regressor	SP2 group
S1	0.72	MTG	6.39	11.90	8.56	adaboost	Avoiding
S2	0.58	MTG	12.44	18.93	16.30	pymlp	Registration+G
S3	0.44	BTM	7.71	14.22	10.05	pygbm	Seeking
S4	0.72	MTG	8.05	11.90	9.93	pygbm	Registration
S5	0.57	MTG	18.15	25.80	21.98	lreg	Touch
S6	0.51	MTG	6.60	10.07	8.67	pyrf	Seeking+G
S7	0.39	BTM	7.07	12.97	9.59	bstTree	Avoiding+G
S8	0.55	MTG	9.12	15.95	12.05	tree	Avoiding+G
Internal	0.63	MTG	7.79	13.18	9.89	ridge	Avoiding
External	0.98	VGE	1.07	1.86	2.32	lm	Total
Total	0.67	MTG	7.07	11.20	9.16	pygbm	Touch

Table 6.11: Best *R* (and valoration according to the Colton criterion), MAE, WAPE (in %), RMSE, best regressor and best SP2 group for each CBCL outcome.

ranksum test comparing the correlation *R* achieved by all the regressors using the datasets with and without gender under the null hypothesis that both distributions have the same mean. The *p*-value achieved by this test was 0.455, that does not allow to reject the null hypothesis, so the difference between using and discarding gender is not statistically significant.

Table 6.11 reports the results achieved by the best regressor and SP2 group for each CBCL outcome. These metrics include *R*, MAE, WAPE and RMSE. The best regressor and SP2 group are also reported. According to the Colton criteria, the prediction for external (*R*=0.98) is considered "very good to excellent" (VGE in Table 6.11), while the prediction of anxious/ depressed (S1) and social problems (S4), with *R*=0.72, is "moderate to good" (MTG) but close to "very good to excellent". The predictions of withdrawn/depressed (S2), thought problems

(S5), attention problems (S6), aggressive behavior (S8), internalizing and total problems are also "moderate to good" (MTG). Overall, the prediction is VGE and MTG in 1 and 8 outcomes, respectively. Finally, the predictions of somatic complaints (S3) and rule-breaking (S7) behaviors are "bad to moderate" (BTM), and therefore fairly unreliable.

The `pygbm` (gradient boosting machine) is the regressor that achieves the best R in more CBCL outcomes (see Table 6.11): somatic complains (S3), with low R, social problems (S4) and total, with higher R. Linear regression achieves the best R in thought (S5, regressor `lreg`), internalizing (`ridge`), and externalizing problems (`lm`, with high R). Regression trees and ensembles provide the best prediction for anxious/depressed (S1, `adaboost`); attention problems (S6, `pyrf`); rule-breaking (S7, `extraTrees`); and aggressive behavior (S8, `tree`). Neural networks are the best for withdrawn/depressed behavior (S2, `pymlp`).

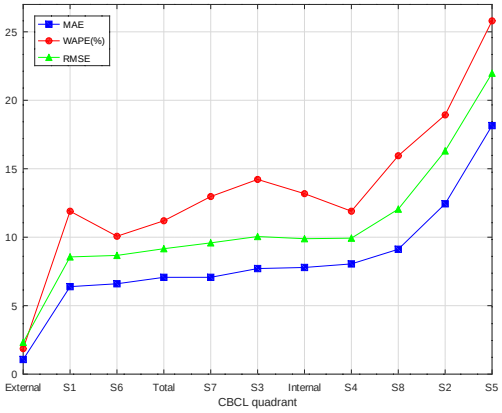

Figure 6.5: Values of MAE, WAPE and RMSE for the best regressor on each CBCL outcome.

The MAE in Table 6.11 is below 10 for all the CBCL outcomes, excepting withdrawn/depressed (S2) and thought problems (S5), so the uncertainty prediction value for most outcomes is below 20. The outcome values range approximately between 35 and 100 (see Figure 6.1), so these MAE values are comparatively small.

The WAPE is very low in externalizing problems (1.86%) and below 14% for most outcomes excepting withdrawn/depressed (S2), somatic complains (S3), thought problems (S5), and aggressive behavior (S8). Since percentage of differences between predicted and true

CBCL outcome are expected to be within 2×WAPE above and below zero, WAPE values below or about 10% can be considered as relatively good. This happens in outcomes S1, S4, S6, external and total, that can be considered as the outcomes where the prediction is "acceptable". These outcomes exhibit MAE values below or about 7. Note that S3 has low MAE (7.71) but high WAPE (18.93%) and low R (0.44), being one of the worst-performing outcomes.

Regarding RMSE, the values of these "good" outcomes (S1, S4, S6, external and total) are below or about 9, although S7 and internal also have RMSE about 9 being poorer outcomes. Figure 6.5 plots the three measures (MAE, WAPE and RMSE), that show a high coherence and identify as the best outcomes: external, S1, S6, total and S4. The outcomes S7, S3 and internal can be excluded due to their high RMSE values.

Figure 6.6 plots the predicted vs. observed (true) values for CBCL outcomes external and S1 for all the participants, the best groups (total and avoiding, respectively) and the best regressors `lm` and `adaboost`. The concordance between both values is very high, specially for outcome external (left panel), where the majority of the participants (blue points) are on the red line that means perfect prediction (WAPE below 2%). The right panel shows lower coincidence values (WAPE around 12%), although the predicted values somehow raise with the true values.

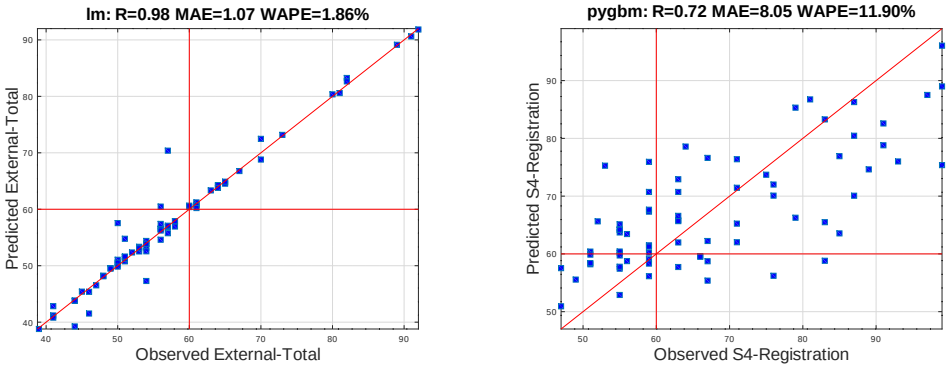

Figure 6.6: Scatterplot with observed (true) and predicted values for: SP2 group total, CBCL outcome external and regressor `lm` (left panel); SP2 group registration, CBCL outcome S4 (social problems) and regressor `pygbm` (right panel).

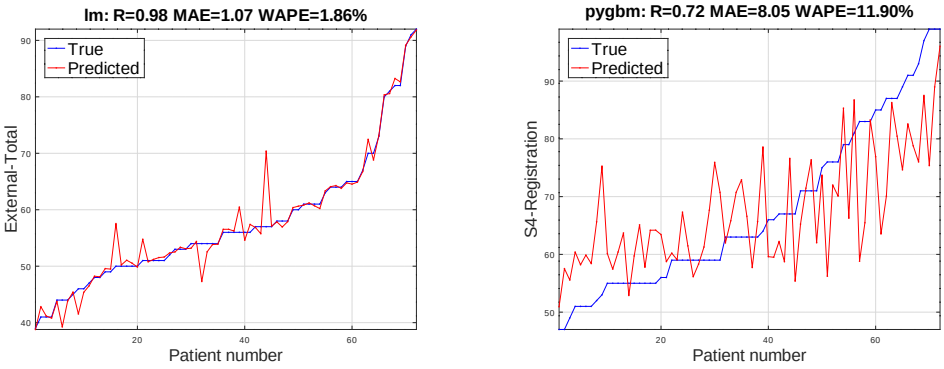

Figure 6.7: True values and values predicted by `lm` (left panel) and `pygbm` (right panel), sorted by increasing true values, for the SP2 groups and CBCL outcomes of figure 6.6.

Figure 6.7 reports the true and predicted values for the previous outcomes sorted by increasing true values. The coincidence is very high for the left panel, excepting few participants, and lower in the right panel.

6.6 Classification in normal, pre-clinical and clinical

Table 6.12 reports the best result for the binary classification into normal or clinical participants and the 11 CBCL outcomes. The table also reports the SP2 group and regressor that achieved the best R, as well as the confusion matrix, kappa, accuracy, sensitivity and specificity (in %) achieved by this regressor. The kappa values are further plotted as a bar diagram in Figure 6.8. According to the levels defined by the above Table 6.5, these values can be grouped as in Table 6.13. Only outcomes in levels "very good" and "moderate" can be considered as an acceptable classification, so 5 of 11 outcomes are "acceptably" classified: externalizing, anxious/depressed (S1), internalizing, total, withdrawn/depressed (S2) and thought problems (S5, that we included in the moderate level because its kappa 39.7% is very near to 40%). Finally, the worst-performing outcomes (weak level) are aggressive behavior (S8), somatic complaints (S3), social problems (S4), attention problems (S6) and rule-breaking behavior (S7).

Analyzing the best result for each CBCL outcome in Table 6.12, we can formulate the following considerations:

Output	Input	Regressor		N	C	Kappa	Acc	Se	Sp
S1	Avoid	adaboost	N	**54**	2	59.3	87.5	56.2	96.4
			C	7	**9**				
S2	Reg+G	pymlp	N	**20**	12	40.4	70.8	77.5	62.5
			C	9	**31**				
S3	Seeking	pygbm	N	**50**	5	35.7	79.2	41.2	90.9
			C	10	**7**				
S4	Reg	pygbm	N	**14**	17	29.2	66.7	82.9	45.2
			C	7	**34**				
S5	Touch	lreg	N	**17**	5	39.7	70.8	68.0	77.3
			C	16	**34**				
S6	Seeking+G	pyrf	N	**6**	9	29.2	77.8	87.7	40.0
			C	7	**50**				
S7	Avoid+G	bstTree	N	**52**	6	27.6	79.2	35.7	89.7
			C	9	**5**				
S8	Avoid+G	tree	N	**40**	8	38.7	73.6	54.2	83.3
			C	11	**13**				
Internal	Avoid	ridge	N	**35**	9	57.0	79.2	78.6	79.5
			C	6	**22**				
External	Total	lm	N	**46**	2	93.9	97.2	100.0	95.8
			C	0	**24**				
Total	Touch	pygbm	N	**24**	11	47.1	73.6	78.4	68.6
			C	8	**29**				

Table 6.12: Confusion matrix, kappa, accuracy, sensitivity and specificity (the four in %) for the classification into normal (N) and clinical (C) for each outcome and the best SP2 group.

Classification	Kappa interval	CBCL outcomes
Very good	$80 \leq \kappa$	external
Moderate	$40 \leq \kappa < 60$	S1, internal, total, S2, S5
Weak	$20 \leq \kappa < 40$	S8, S3, S4, S6, S7

Table 6.13: Outcomes in each kappa level according to the criterion of Table 6.5.

1. The best-performing outcome **external** achieves an almost perfect classification, with kappa of 93.9% ("very good", according to the Blackman criterion), and 97.2% of accuracy using the linear regression, `lm`. The confusion matrix is excellent, with the numbers outside the diagonal near zero, with 100% of sensitivity and specificity 96.4% because there are 2 false positives (normal participants classified as clinical).

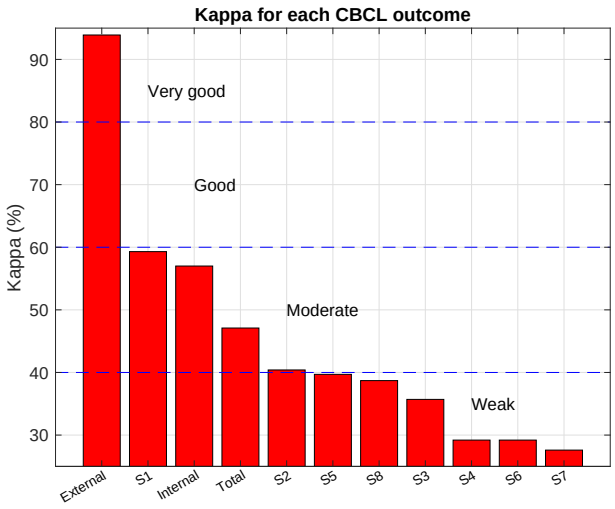

Figure 6.8: Best kappa (in %) for each CBCL outcome.

2. The second best classification is achieved by regressor `adaboost` for outcome **S1**, with kappa=59.3% ("moderate") and accuracy 87.5%, high specificity (96.4%) but low sensitivity (56.2%) because a significant number (7) of clinical participants are classified as normal, compared to 9 clinical participants correctly classified.

3. The outcome **internal** also achieves a "moderate" kappa (57%) and accuracy (79.2%) using `ridge` regressor, in this case with similar sensitivity and specificity (about 79%), and a good confusion matrix, because the numbers outside the diagonal (9 and 6) are much lower than the ones on the diagonal (35 and 22). This is because normal participants predicted as clinical (FP) are few compared to normal predicted as normal (TN), so that Sp is high, and also because clinical participants classified as normal (FN) are few compared to clinical predicted as clinical (TP), so that Se is high.

4. A similar case, although with slightly lower performance, is the outcome **total** with regressor `pygbm`. Kappa is lower (47.1%) with 73.6% of accuracy, but the confusion matrix is relatively good because the numbers outside the diagonal (11 and 8) are much lower than the diagonal terms (24 and 29). Both sensitivity and specificity are relatively high (above 68%), although the latter is 10% below the former because more normal participants (11) are classified as clinical.

5. The outcome **S2** also achieves a kappa (40.4%) very near to moderate using `pymlp`, with accuracy 70.8% and terms outside the diagonal (12 and 9) lower than inside the diagonal (20 and 31). The sensitivity and specificity are also relatively acceptable, both about 62-77%. The same accuracy is also achieved by outcome **S5** and `lreg`, with very similar kappa (39.7%), sensitivity and specificity (about 68-77%), although in this case there are more false negatives (16) than negatives (5), while in S2 there are more false positives than negatives (12 vs. 9).

6. The remaining outcomes (S3, S4, S6, S7 and S8) exhibit clearly lower performance, with kappa values in the range 27-38%, that correspond to weak classification. Either sensitivity or specificity are poor, but the other is high: e.g., S3 has a high specificity (90.9%) but very low sensitivity (41.2%), and the same happens with S4 and S6 (specificity about 40%), S7 (sensitivity 35.7%) and S8 (sensitivity 54.2%).

Overall, accuracy is above 70% in all the outcomes excepting social problems (S4). The accuracy values are close to or above 80% for anxious/depressed (S1), somatic complains (S3), attention problems (S6), rule-breaking (S7), externalizing and internalizing. Sensitivity achieves 100% in externalizing, overcomes 80% in social (S4) and attention (S6) problems, being close to or above 70% in the remaining CBCL outcomes excepting anxious/depressed (S1), somatic complains (S3), rule-breaking (S7), and aggressive behavior (S8). Specificity is near or above 80% in 8 of 11 outcomes, excepting withdrawn/depressed (S2), social problems (S4) and attention problems (S6).

Regarding Figure 6.6 in section 6.5, the CBCL values (blue squares) depicted on the right (resp. left) of the red vertical line at 60 are participants who scored in the clinical (resp. normal) ranges. Likewise, the squares above (resp. below) the horizontal red line at 60 are participants predicted by the best regressor as clinical (resp. normal). All the squares on the upper right and lower left areas defined by the horizontal and vertical red lines at 60 are predicted as the right class label, and only values on the upper-left and lower-right quadrants are predicted as the wrong class label. Thus, the number of classification errors is very low in the left panel (external-total, `lm`), with no false negatives (squares in the lower right quadrant, 100% sensitivity) and just two false positives (upper left quadrant). In the right panel (S4-registration, `pygbm`), the number of dots in the upper left and lower right quadrants is higher, but much lower than in lower left and upper right quadrants (participants well-classified), with

Output	Input	Regressor		N	PC	C	Kappa	Acc
S1	Avoid	adaboost	N	**54**	2	0	35.7	79.2
			PC	4	**1**	3		
			C	3	3	**2**		
S2	Reg+G	pymlp	N	**20**	4	8	35.1	58.3
			PC	8	**4**	4		
			C	1	5	**18**		
S3	Seeking	pygbm	N	**50**	5	0	22.1	73.6
			PC	9	**2**	0		
			C	1	4	**1**		
S4	Reg	pygbm	N	**14**	14	3	31.5	54.2
			PC	5	**5**	4		
			C	2	5	**20**		
S5	Touch	lreg	N	**19**	4	2	23.2	51.4
			PC	4	**1**	10		
			C	10	5	**17**		
S6	Seeking+G	pyrf	N	**7**	14	1	12.0	45.8
			PC	6	**20**	6		
			C	0	12	**6**		
S7	Avoid+G	bstTree	N	**52**	6	0	27.1	77.8
			PC	2	**4**	0		
			C	7	1	**0**		
S8	Avoid+G	tree	N	**43**	6	3	27.9	69.4
			PC	8	**3**	2		
			C	3	0	**4**		
Internal	Avoid	ridge	N	**35**	9	2	42.2	68.1
			PC	5	**9**	1		
			C	1	5	**5**		
External	Total	lm	N	**46**	1	1	91.9	95.8
			PC	0	**14**	0		
			C	0	1	**9**		
Total	Touch	pygbm	N	**24**	9	2	39.4	61.1
			PC	7	**9**	2		
			C	1	7	**11**		

Table 6.14: Confusion matrix, kappa and accuracy (in %) for the classification into normal (N), pre-clinical (PC) and clinical (C) for each outcome and the bet SP2 group.

an accuracy of 66.7%, so the impact of unreliability in prediction (R=0.72) over classification is relatively low for this outcome.

Table 6.14 reports the confusion matrix, alongside with the kappa and accuracy (in %) of the classification in 3 classes (normal, pre-clinical and clinical) for each outcome. Although the table reports the results for the 11 outcomes, it is obvious that those outcomes with weak concordance between true and predicted outcome with two classes will exhibit even lower concordance considering three classes. Specifically, the only outcomes with kappa above or about 40% (very good and moderate levels) are.

1. **External**, with kappa 91.9% (very good) and accuracy 95.8%. Thus, discrimination between normal, pre-clinical and clinical is very accurate, with very low terms (0 or 1) outside the diagonal of the confusion matrix.

2. **Internal**, with kappa 42.2% (moderate) and accuracy (68.1%) much lower than external. The confusion matrix has so many normal participants erroneously predicted pre-clinical (9) as pre-clinical participants classified as pre-clinical. Also, a high part of pre-clinical participants (5) are classified as normal.

3. **Total**, with kappa 39.4% (very near to moderate) and accuracy 61.1%. These low values are due to the high terms N-PC (9), PC-N (7) and C-PC (7).

The remaining outcomes have kappa values between 12% and 35%, with accuracies between 51% and 79%. The performance is specially poor in attention problems (S6, kappa =12%), somatic complaints (S3, kappa=22.1%), thought problems (S5, kappa=23.2%), and rule-breaking behavior (S7, kappa=27.1%), where 7 of 8 clinical participants are predicted as normal.

6.7 Discussion

The current study investigates how sensory processing alterations predict behavior problems in children and adolescents with ASD using ML models. This technique offers a new potential solution to classify ASD difficulties based on different dimensions, e.g., behavior, genetics, and neuroimaging data, among others [34]. These artificial intelligence tools can be an important and useful approach to further explain how sensory processing abnormalities (one of the earliest clinical alterations in ASD) are predictive of social, behavior, and emotional problems in this population.

Overall, results revealed high correlation between some true behavioral (CBCL) outcomes and the values predicted by the regressors using some SP2 processing groups. The highest correlation was observed for externalizing, total problems, anxiety/depression and social problems. The prediction was more reliable using certain SP2 groups, revealing strong relations with the predicted CBCL outcomes: CBCL externalizing problems and SP2 total group; CBCL anxious/depressed and SP2 avoiding group; CBCL social problems and SP2 registration group; and CBCL total problems outcome with SP2 touch group. With this prediction, early indicative signs of the overall behavioral difficulties of children with an ASD diagnosis, that occur at higher ages, can be obtained exclusively from their response to sensory stimuli, that is registered at very early ages. Thus, these results can be extremely useful for clinical interventions. Considering the predicted information, early signs of altered sensory functioning would allow for incorporating personalized and early sensory-based care therapies for children with an ASD diagnosis, hence minimizing the affected areas of behavior, as much as possible.

Results obtained by the ML methods are consistent with other findings, suggesting an association between sensory processing and behavior problems in children and adolescents with ASD [41]. Patterns of sensory processing alterations, including touch processing, may imply difficulties in response to environmental cues and missing opportunities to learn from them, which is the foundation for the development of more complex processes, such as emotional regulation and social interactions [43]. These difficulties have cascade effects in daily life activities in ASD, manifested through a range of behavior problems such as hyperactivity, impulsivity, stereotyped and repetitive behaviors, as well as emotional and social distress [74]. It may contribute to the emergence of anxious and depressive states, as dealing with sensory stimulation may be demanding for children and adolescents with ASD [29].

The results contribute to the existing evidence highlighting the detrimental impact of altered sensory processing on behavioral outcomes in children and adolescents with ASD [30, 78, 56]. These difficulties interfere with their adaptive functioning [54] and should be considered in interventional approaches with this population. Early intervention programs focused on addressing sensory difficulties may help prevent behavioral problems, alleviate behavioral difficulties and improve adaptive functioning of children and adolescents to environmental situations.

This work offers multiple strengths of which how ML methods contribute to clarify the association between sensory processing impairments and behavioral difficulties in children

and adolescents with ASD. These results can provide important clues for sensory-related intervention approaches within this population. However, this study has also some limitations. First is the reduced number of participants. Future studies should consider a larger sample to confirm the results. In addition, it would be important to replicate this investigation in ASD adult population to verify if similar outcomes are present. Also, it would be important that future studies address severity levels of ASD presentations, considering the high inter-individual variability observed in this population. Another limitation of this work refers to its cross-sectional design that does not allow for determining causal associations between the variables. Thus, future studies should consider assessing sensory processing and behavioral outcomes in a longitudinal manner to clarify the causal relationship between these dimensions. Also, future investigations should integrate brain imaging techniques to elucidate about the areas of the brain that pertain to sensory processing and/or behavior that are affected, which would strengthen these findings.

CHAPTER 7

CONCLUSIONS

The current PhD. Thesis develops efficient algorithms for classification and regression by projecting the original high-dimensional data in two-dimensions and creating a 2D map of the problem considered. The proposed methods, named ultra-fast two-dimensional classifier and regressor (UF2DC and UF2DR, respectively), perform a projection of the training patterns into the 2D plane using the linear discriminant analysis (LDA), i.e., multiplying the training pattern by the two main eigenvectors of the product $\mathbf{W}^{-1}\mathbf{B}$, where $\mathbf{W}$ and $\mathbf{B}$ are the within-class and between-class covariance matrices of the training set. These eigenvectors identify the directions that maximize (resp. minimize) the between-class (resp. within-class) covariance, that measures the separation between classes (resp. among patterns within the same class). The LDA was selected among the plethora of dimensionality reduction methods available in the literature due to its speed and its performance to keep the data structure in the reduced dimensionality space, as we tested in previous research works. This framework in the current PhD. Thesis is generalized to regression problems considering classes associated to the levels of the continuous output, and using LDA with the classes defined by these levels.

After the 2D projection, UF2DC and UF2DR create an image of the squared region that includes the projected patterns. The value of each pixel in this image identifies the class label (for classification) or the level value (for regression) associated to the region of the 2D space covered by this pixel (or square). This value is set by considering the true outputs for the training patterns that are placed in this square or, in case of empty squares, by the patterns placed within an environment of the current square. The image acts as a map of the classification or regression problem, showing the class label of level value of each pixel (or

point in the map), that provides a visual understanding of the landscape of the function that has been learned either for classification or regression. The map can be used as a classifier, because the output predicted for a new test pattern is given by the class label of the square to where the original pattern is projected. In regression problems, the map is used as a method that predicts, for a given input pattern, the value associated to the square where the pattern is projected by LDA.

The proposed UF2DC and UF2DR are evaluated on a collection of 26 classification and 23 regression datasets, some of them very large both in terms of number of patterns (up to 5 millions) and features (up to 719). These methods conserve most of the performance of state-of-the-art classifiers as the Gaussian kernel support vector machine (GSVC or GSVR). In classification, the average kappa of UF2DC is only 0.11 below GSVC, and in regression the average squared correlation R^2 of UF2DR is only 0.14 below GSVR. Interestingly, the difference is significant only in few datasets, 6 of 26 in classification and 4 of 23 in regression. Besides, UF2DC and UF2DR are between 10 and 1,000 times faster than the competing approaches, and the difference raises quickly with the dataset size. These methods are able to process large datasets where the other classifiers fail due to its low speed and high memory requirements. They also provide a way to explain and visualize the intrinsic structure of the data or the function to be learnt, in terms of class overlap and relative class positions for classification, and of function values, direction and speed of change in these values for regression problems.

The automatic prediction of behavior from sensory data in autism spectrum disorder using machine learning models is another work developed in this PhD. Thesis. Autism spectrum disorders are associated with sensory processing abnormalities that often lead to behavioral alterations. The current study investigates the relations between the scales of the sensory profile-2 (SP2) questionnaire, that evaluates sensory indicators through a collection of questions, and the outcomes in the Child Behavior Check-List (CBCL) questionnaire. The objective is to study the association between sensory abnormalities, that are detected at very early ages, and behavior problems, that emerge at higher ages. More specifically, we tried to predict automatically the CBCL outcomes using the SP2 items. We used a large collection of supervised machine learning algorithms oriented to the prediction of the CBCL outcomes, that are continuous numeric values. This collections is composed by 26 regressors of different families: linear and regularized linear regression; kernel and support vector regression; ensembles, in-

cluding bagging, adaboost, gradient boosting machine and random forest; regression trees; and neural networks.

The 11 behavior outcomes to be predicted are anxious/depressed, withdrawn/depressed, somatic complaints, social problems, thought problems, attention problems, rule-breaking behavior, aggressive behavior, internalizing problems, externalizing problems and total (all the CBCL outcomes). The 6 groups of items in the sensory questionnaire to be used as input data by the automatic predictions are the four quadrants of the SP2 questionnaire (seeking, avoiding, sensitivity and registration), alongside with the "touch processing" section and "total", that includes all the SP2 items. Using a sample of 72 participants, the reliability of the predictions issued by the automatic methods are "from good to excellent" for externalizing problems using linear regression with all the SP2 items. The reliability is "from moderate to good" for anxious/depressed using the adaboost ensemble with the avoiding sensory quadrant, and for social problems using the gradient boosting machine ensemble with the registration quadrant. The predictions can be qualified as "from moderate to good" for the remaining outcomes excepting somatic complaints and rule-breaking, where the predictions are "bad to moderate".

We also considered the binary classification into normal (outcome below 60) vs. clinical (above 60) participants for each CBCL outcome. In our experiments accuracy reaches 97.2% and 87.5% for externalizing and anxious/depressed, respectively. As well, somatic complaints and rule-breaking outcomes, with low predictive reliability, still achieve accuracies near 80%, alongside with attention problems and internalization, while the remaining outcomes achieve accuracies above or near 70%. Considering three classes (normal below 60, pre-clinical between 60 and 70, and clinical above 70), the accuracy achieves 95.8% and 79.2% in externalizing and anxious/depressed, respectively, while internalizing and total achieve 68.1% and 61.1%, respectively. These results show that an accurate prediction is only possible for 7 of 11 behavior outcomes: externalizing, anxious/depressed, internalizing and total, and in lower degree for social problems, somatic complaints and withdrawn/depressed. The remaining 4 outcomes perform much poorer and they can not be considered as reliably predicted.

Future work includes: 1) to extend the use of proposed mapping methods to overcome the limitations of LDA for classification problems with many classes; 2) to synthesize patterns with a given class label or function value depending on the region of the map where they are located; 3) to extend the study about autism for a larger number of participants, including adults, in order to increase the significance of the conclusions drawn; and 4) to

propose classification algorithms focused on numeric answers to questionnaires, that are widely used in fields of Psychology such as obssesive-compulsory disease (OCD) and attention deficit/hyperactivity disorder (ADHD).

Appendix A

Listado de publicacións

As publicacións asociadas a esta tese de doutoramento son as seguintes:

1. H. Alateyat, M. Fernández-Delgado, E. Cernadas, S. Barro. Ultra fast classification and regression of high-dimensional problems projected on 2D". *Neural Processing Letters* (2022). ISSN 1573-773X. DOI: https://doi.org/10.1007/s11063-022-11090-3. Índice de impacto 2.908 (2020), posición 63 de 139 en Computer Science-Artificial Intelligence (Q2).

2. H. Alateyat, S, Cruz, E, Cernadas, M, Tubío-Fungueiriño, A, Sampaio, A.J. González-Villar, A. Carracedo, M. Fernández-Delgado, M. Fernández Prieto. A machine learning approach in autism spectrum disorders: from sensory processing to behavior problems. *Frontiers in Molecular Neuroscience*, Vol 15, pp. 1–9, 2022. Número especial en "Molecular Advances and Applications of Machine Learning in Understanding Autism and Comorbid Psychiatric Disorders". ISSN 1662-5099. DOI: https://doi.org/10.3389/fnmol.2022.889641. Índice de impacto 5.639 (2020), posición 60 de 273 en Neurosciences (Q1).

3. M. Tubío-Fungueiriño, S. Cruz, E. Cernadas, H. Alateyat, A. Sampaio, A. González-Villar, S. Conde-Pumpido, M. Pozo, A. Carracedo, S. Barro, M. Fernández-Delgado, M. Fernández-Prieto. Predicción automática de comorbilidades de TDAH a partir de información sensorial en pacientes con TEA. Póster en el XXV Congreso Nacional de Psiquiatría. 17-19 de noviembre de 2022, Santiago de Compostela. Enlace: https://congresonacionaldepsiquiatria.es/cnps/expoposters-2022/posters/postersb_194.png

Os co-autores das publicacións Eva Cernadas e Senén Barro pertencen ao Departamento de Electrónica e Computación, Universidade de Santiago de Compostela. Os restantes co-autores están asociados ao Grupo de Xenómica da USC e actúan como expertos do dominio da aplicación, neste caso a desorde de espectro autista no ámbito da psicoloxía. Estas publicacións foron realizadas exclusivamente pola doutoranda, agás no que respecta ao coñecemento e datos da aplicación nas publicacións 1 e 3, proporcionados polo persoal investigador do Grupo de Xenómica. Polo tanto, a contribución da doutoranda nestas publicacións é orixinal e representa o 100% das mesmas.

Bibliography

[1] T.M. Achenbach and L.A. Rescorla. *Manual for the ASEBA School-Age Forms & Profiles: An Integrated System of Multi-Informant Assessment*. University of Vermont; Research Center for Children, Youth, & Families, 2001.

[2] M. Ahsan, M. Mashuri, H. Khusna, and Wibawati. Kernel principal component analysis (PCA) control chart for monitoring mixed non-linear variable and attribute quality characteristics. *Heliyon*, 8(6):e09590, 2022.

[3] S. Alawadi, M. Fernández-Delgado, D. Mera, and S. Barro. Polynomial kernel discriminant analysis for 2D visualization of classification problems. *Neural Comput & Appl*, 8:1–17, Nov 2017.

[4] M. Amouzgar, D.R. Glass, R. Baskar, I. Averbukh, S.C. Kimmey, A.G. Tsai, F.J. Hartmann, and S.C. Bendall. Supervised dimensionality reduction for exploration of single-cell data by HSS-LDA. *Patterns*, 3(8):100536, 2022.

[5] K. C. Assi, H. Labelle, and F. Cheriet. Modified large margin nearest neighbor metric learning for regression. *IEEE Signal Proc. Let.*, 21(3):292–296, 2014.

[6] American Psychiatric Association. *Diagnostic and statistical manual of mental disorders (DSM-5©)*. Am Psychiatric Pub, 2013.

[7] E. Barshan, A. Ghodsi, Z. Azimifar, and M.Z. Jahromi. Supervised principal component analysis: visualization, classification and regression on subspaces and submanifolds. *Pattern Recogn*, 44(7):1357–1371, 2011.

[8] D. M. Bates and J. M. Chambers. *Statistical Models*, chapter 10. Wadsworth & Brooks/Cole, 1992.

[9] A. Ben-Sasson, A.S. Carter, and M.J. Briggs-Gowan. Sensory over-responsivity in elementary school: prevalence and social-emotional correlates. *J Abnorm Child Psychol*, 37:705–716, 2009.

[10] V. Bitsika, C.F. Sharpley, and R. Mills. Are sensory processing features associated with depressive symptoms in boys with an asd? *J Autism Dev Disord*, 46:242–252, 2016.

[11] N. Blackman and J. Koval. Interval estimation for cohen's kappa as a measure of agreement. *Stat. Med.*, 19:723–741, 2020.

[12] L. Breiman. Random forests. *Mach Learn*, 45:5–32, 2001.

[13] J. Briscoe and O. Marín. Looking at neurodevelopment through a big data lens. *Science*, 369(6510):1–10, 2020.

[14] Jean Carletta. Assessing agreement on classification tasks: The kappa statistic. *Comput Linguistics*, 22(2):249–254, 1996.

[15] C.C. Chang and C.J. Lin. LIBSVM: a library for support vector machines. *ACM T Intel Syst Technol*, 2:27:1–27:27, 2011.

[16] F. Chiarotti and A. Venerosi. Epidemiology of autism spectrum disorders: a review of worldwide prevalence estimates since 2014. *Brain Sci*, 10:274–295, 2020.

[17] Z. Cidav, J. Munson, A. Estes, G. Dawson, S. Rogers, and D. Mandell. Cost offset associated with early start Denver model for children with autism. *J Am Acad Child Psychol*, 56:777–783, 2017.

[18] T. Colton. *Statistical in medicine*. Little Brown and Co., New York, NJ, 1974.

[19] C. Cortes and V. Vapnik. Support-vector networks. *Mach Learn*, 20:273–297, 1995.

[20] D. Dalpiaz. *R for Statistical Learning*. Springer, Berlin, 2021.

[21] B.K. Deb and H.S. Bateup. Modeling somatic mutations associated with neurodevelopmental disorders in human brain organoids. *Front Mol Neurosci*, 14:787243, 2022.

[22] M.F. Delgado, E. Cernadas, and S. Barro. Do we need hundreds of classifiers to solve real world classification problems? *J Mach Learn Res*, 14(3):3133–3181, 2014.

[23] Harris Drucker. Improving regressors using boosting techniques. In *Intl Conf Mach Learn*, 1997.

[24] D. Dua and C. Graff. UCI machine learning repository, 2017.

[25] W. Dunn. *Sensory Profile 2 Manual*. Pearson. The Psychological Corporation, 2014.

[26] Unitat d'Epidemiologia i de Diagnòstic en Psicopatologia del Desenvolupament. Norms of Achenbach's CBCL6-18 Forms in Spanish population, 2016.

[27] R.E. Fan, K.W. Chang, C.J. Hsieh, X.R. Wang, and C.J. Lin. LIBLINEAR: A library for large linear classification. *J Mach Learn Res*, 9:1871–1874, 2008.

[28] M. Fernández-Delgado, M. Sirsat, E. Cernadas, S. Alawadi, S. Barro, and M. Febrero-Bande. An extensive experimental survey of regression methods. *Neural Netw*, 111:11–34, 2019.

[29] M. Fernández-Prieto, C. Moreira, S. Cruz, V. Campos, R. Martínez-Regueiro, and M. Taboada et al. Executive functioning: a mediator between sensory processing and behaviour in autism spectrum disorder. *J Autism Dev Disord*, 51:2091–2103, 2021.

[30] J.H. Foss-Feig, J.L. Heacock, and C.J. Cascio. Tactile responsiveness patterns and their association with core features in autism spectrum disorders. *Res Autism Spectr Disord*, 6:337–344, 2012.

[31] E. Frank, M. Hall, and I. Witten. *The WEKA Workbench. Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques"*. Morgan Kaufmann, Fourth Edition, 2016.

[32] J.H. Friedman. Greedy function approximation: a gradient boosting machine. *Ann. Stat.*, 29:1189–1232, 2001.

[33] X. Geng, D.C. Xuan, and Z.H. Zhou. Supervised nonlinear dimensionality reduction for visualization and classification. *IEEE T Syst Man Cyb B*, 35(6):1098–1107, 2005.

[34] A.L. Georgescu, J.C. Koehler, J. Weiske, K. Vogeley, N. Koutsouleris, and C. Falter-Wagner. Machine learning to study social interaction difficulties in ASD. *Front Robot AI*, 6:132, 2019.

[35] P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees. *Mach Learn*, 63:3–42, 2006.

[36] C. Gonthier, L. Longuépée, and M. Bouvard. Sensory processing in low-functioning adults with autism spectrum disorder: distinct sensory profiles and their relationships with behavioral dysfunction. *J Autism Dev Disord*, 46:3078–3089, 2016.

[37] R.C. González, R.E. Woods, and S.L. Eddins. *Digital image processing using Matlab*. Prentice-Hall, 2004.

[38] B. Greenwell, B. Boehmke, J. Cunningham, and GBM Developers. gbm: Generalized boosted regression models, R package version 2.1.8, 2020. https://cran.r-project.org /package=gbm.

[39] K. Hornik, C. Buchta, and A. Zeileis. Comput stat. *Open-Source Machine Learning: R meets Weka*, 24(2):225–232, 2009.

[40] Y. Huang, K. Guo, X. Yi, J. Yu, Z. Shen, and T. Li. T-copula and Wasserstein distance-based stochastic neighbor embedding. *Knowl-Based Syst*, 243:108431, 2022.

[41] G. Iarocci and J. McDonald. Sensory integration and the perceptual experience of persons with autism. *J Autism Devel Disord*, 36:77–90, 2006.

[42] A. Jalilvand and N. Salim. Feature unionization: a novel approach for dimension reduction. *Appl Soft Comput*, 52:1253–1261, 2017.

[43] N. Kojovic, L. Ben Hadid, M. Franchini, and M. Schaer. Sensory processing issues and their association with social difficulties in children with autism spectrum disorders. *J Clin Med*, 8:1508, 2019.

[44] M. Kudelka, P. Kromer, M. Radvansky, Z. Horak, and V. Snasel. Efficient visualization of social networks based on modified sammon's mapping. *Swarm Evol Comput*, 25:63–71, 2015.

[45] M. Kuhn. Caret: Classification and regression training. R package version 6.0-86, 2020. https://CRAN.R-project.org/package=caret.

[46] M. Kuhn and R. Quinlan. Cubist: Rule- and instance-based regression modeling, R package version 0.2.3, 2020. https://CRAN.R-project.org/package=Cubist.

[47] A.E. Lane, R.L. Young, A.E. Baker, and M.T. Angley. Sensory processing subtypes in autism: association with adaptive behavior. *J Autism Dev Disord*, 40:112–122, 2010.

[48] A. Liaw and M. Wiener. Classification and regression by randomforest. *R News*, 2(3):18–22, 2002. https://CRAN.R-project.org/doc/Rnews.

[49] S. Liu, Z. Chen, and F. Jiao. Detection of maize seed germination rate based on improved locally linear embedding. *Comput Electron Agric*, 204:107514, 2023.

[50] M.V. Lombardo, M.C. Lai, and S. Baron-Cohen. Big data approaches to decomposing heterogeneity across the autism spectrum. *Mol Psychiatry*, 24:1435–1450, 2019.

[51] C. Lord, M. Rutter, P.C. DiLavore, S. Risi, K. Gotham, and S.L. Bishop et al. *ADOS. Escala de observación para el diagnóstico del autismo. (In Spanish).* TEA Editions., 2008.

[52] L. Maaten. An introduction to dimensionality reduction using Matlab. Technical Report 2579-2605, Universiteit Maastricht, 2007.

[53] E. Marco, L. Hinkley, S. Hill, and S. Nagarajan. Sensory processing in autism: a review of neurophysiologic findings. *Pediatr Res*, 69:48–54, 2011.

[54] R.L. McLean, A. Johnson-Harrison, E. Zimak, R.M. Joseph, and E.M. Morrow. Executive function in probands with autism with average IQ and their unaffected first-degree relatives. *J Am Acad Child Adolesc Psychol*, 53:1001–1009, 2014.

[55] O. Mendels, H. Stern, and S. Berman. User identification for home entertainment based on free-air hand motion signatures. *IEEE T Syst Man Cyb A*, 44(11):1461–1473, 2014.

[56] H.O. Miguel, A. Sampaio, R. Martínez-Regueiro, L. Gómez-Guerrero, C.G. López-Dóriga, and S. Gómez. Touch processing and social behavior in ASD. *J Autism Dev Disord*, 47:2425–2433, 2017.

[57] M. Mikkelsen, E.L. Wodka, S.H. Mostofsky, and N.A.J. Puts. Autism spectrum disorder in the scope of tactile processing. *Dev Cogn Neurosci*, 29:140–150, 2018.

[58] B. Ortega-Flores, L.A. Solari, and M. Martini. Multidimensional scaling (MDS): A quantitative approximation of zircon ages to sedimentary provenance with some examples from Mexico. *J S Am Earth Sci*, 110:103347, 2021.

[59] I. Parenti, L.G. Rabaneda, H. Schoen, and G. Novarino. Neurodevelopmental disorders: from genetics to functional pathways. *Trends Neurosc*, 43:608–621, 2021.

[60] E. Pasolli, H. L. Yang, and M. M. Crawford. Active-metric learning for classification of remotely sensed hyperspectral images. *IEEE T Geosci Remote*, 54(4):1925–1939, 2016.

[61] E.S. Piccardi, A.J. Begum, E.J.H. Jonesm, L. Mason, T. Charman, and M.H. Johnson et al. Behavioural and neural markers of tactile sensory processing in infants at elevated likelihood of autism spectrum disorder and/or attention deficit hyperactivity disorder. *J Neurodev*, 13:1, 2021.

[62] N.A. Puts, E.L. Wodka, M. Tommerdahl, S.H. Mostofsky, and R.A. Edden. Impaired tactile processing in children with autism spectrum disorder. *J Neurophysiol*, 111:1803–1811, 2014.

[63] H. Qu, L. Li, Z. Li, and J. Zheng. Supervised discriminant Isomap with maximum margin graph regularization for dimensionality reduction. *Expert Syst Appl*, 180:115055, 2021.

[64] R.J. Quinlan. Learning with continuous classes. In *5th Australian J Conf Artif Intel*, pages 343–348, 1992.

[65] M. Rutter, A. Le Couteur, and C. Lord. *ADI-R: entrevista para el diagnóstico del autismo-revisada (In Spanish)*. TEA Editions., 2006.

[66] D. Sheskin. *Handbook of parametric and nonparametric statistical procedures*. Chapman and Hall/CRC Press, 2006.

[67] D.F. Specht. A general regression neural network. *IEEE T Neural Netw*, 2:568–576, 1991.

[68] Donald F. Specht. Probabilistic neural networks. *IEEE T Neural Netw*, 3(1):109–118, 1990.

[69] D. Stathakis and K. Perakis. Feature evolution for classification of remotely sensed data. *IEEE Geosci Remote S*, 4(3):354–358, 2007.

[70] AJ. Tallón-Ballesteros, J.C. Riquelme, and R. Ruiz. Semi-wrapper feature subset selector for feed-forward neural networks: applications to binary and multi-class classification problems. *Neurocomputing*, 353:28–44, 2019.

[71] C. Tao and J. Feng. Canonical kernel dimensionality reduction. *Comput Stat Data An*, 107:131–148, 2017.

[72] T. Tavassoli, K. Bellesheim, M. Tommerdahl, J.M. Holden, A. Kolevzon, and J.D. Buxbaum. Altered tactile processing in children with autism spectrum disorder. *Autism Res*, 9:616–620, 2016.

[73] T. Thabtah. Machine learning in autistic spectrum disorder behavioral research: a review and ways forward. *Inform Health Soc Care*, 44:278–297, 2019.

[74] M.D. Thye, H.M. Bednarz, A.J. Herringshaw, E.B. Sartin, and R.K. Kana. The impact of atypical sensory processing on social impairments in autism spectrum disorder. *Dev Cogn Neurosci*, 29:151–167, 2018.

[75] S.D. Tomcheck and W. Dunn. Sensory processing in children with and without autism: a comparative study using the short sensory profile. *Am J Occup Ther*, 61:190–200, 2007.

[76] M.H. Tseng, C.P. Fu, S. Cermak, L. Lu, and J.Y. Shieh. Emotional and behavioral problems in preschool children with autism: relationship with sensory processing dysfunction. *Res Autism Spect Dis*, 5:1441–1450, 2011.

[77] C. Turchetti and L. Falaschetti. A manifold learning approach to dimensionality reduction for modeling data. *Inform Sci*, 491:16–29, 2019.

[78] M. Uljarević, A. Lane, A. Kelly, and S. Leekam. Sensory subtypes and anxiety in older children and adolescents with autism spectrum disorder. *Autism Res*, 9:1073–1078, 2016.

[79] A. Wang, N. An, G. Chen, L. Li, and G. Alterovitz. Accelerating wrapper-based feature selection with k-nearest-neighbor. *Knowl.-Based Syst.*, 83:81–91, 2015.

[80] L. Wang and Q. Mao. Probabilistic dimensionality reduction via structure learning. *IEEE T Pat Anal*, 41(1):205–219, 2019.

[81] W. Wang and M.A. Carreira-Perpinan. The role of dimensionality reduction in classification. In *AAAI Conf. Artif. Intel.*, pages 2128–2134, 2004. http://arxiv.org/abs/1405.6444.

[82] Z. Wang. bst: Gradient boosting, R package version 0.3-23, 2020. https://cran.r-project.org/package=bst.

[83] X. Wei, H. Shen, Y. Li, X. Tang, F. Wang, M. Kleinsteuber, and Y. L. Murphey. Reconstructible nonlinear dimensionality reduction via joint dictionary learning. *IEEE T Neur Netw Lear*, 30(1):175–189, 2019.

[84] B. Yang, M. Xiang, and Y. Zhang. Multi-manifold discriminant Isomap for visualization and classification. *Pattern Recogn*, 55:215–230, 2016.

[85] D.A. Zachor and P. Curatolo. Recommendations for early diagnosis and intervention in autism spectrum disorders: an Italian-Israeli consensus conference. *Eur J Paediatr Neurol*, 18:107–118, 2014.

[86] M. Zhang and X. Zang. Application of principal-component analysis to the interpretation of coal tar physico-chemical properties. *Fuel*, 338:127304, 2023.

[87] Z. Zhang, T. W. S. Chow, and M. Zhao. Trace ratio optimization-based semi-supervised nonlinear dimensionality reduction for marginal manifold visualization. *IEEE T Knowl Data Eng*, 25(5):1148–1161, 2013.

List of Figures

List of Tables

This thesis formulates methods to perform classification and regression by projecting high-dimensional patterns in two dimensions. These methods create a 2D classification or regression map to visualize the data as a political (for classification) or temperature (for regression) map, where each pixel in the map has an associated prediction. The thesis also uses 26 machine learning models for the automatic prediction of behavior in the treatment of autism spectrum disorder using sensory processing information. Behavior and sensory data are extracted from their respective questionnaires. Out of 11 behavior outcomes, the prediction of externalizing problems is very reliable and accurate enough in other 7 outcomes.