Mestrado Integrado em Engenharia Biomédica

Reconhecimento de Padrões

Relatório Final
Human Activity Recognition using Smartphones

Adriana Costa, 2013140059
Filipa Costa, 2013148125

May 29, 2017

ÍNDICE

LIST OF FIGURES

LIST OF TABLES

1 Resumo

Este projeto está inserido no âmbito da cadeira de Reconhecimento de Padrões e a nossa tarefa consiste em desenvolver classificadores com vista a reconhecer padrões em dados referentes à atividade humana. Deste modo, objetivo principal consiste em explorar todo o processo de classificação de um dataset que nos foi fornecido: "Human Activity Recognition using Smartphones". Para isso, fizemos um trabalho exploratório que envolveu variadíssimas etapas, desde pré-processamento dos dados, análise de métodos de seleção e redução de features e exploração de vários métodos de classificação. Por fim, avaliámos os resultados de performance e chegámos à combinação de métodos ótima para o contexto do problema.

2 INTRODUÇÃO

Hoje em dia, com o envelhecimento da população, há uma grande necessidade de sistemas inteligentes de assistência capazes de monitorar pessoas idosas, pessoas com deficiências fisicas, pessoas que sofrem de diabetes e obesidade ou até para monitorizar certas atividades fisicas. Assim, o reconhecimento automático de atividade humana (HAR) em Ambientes de Vida Assistida (AAL) é de grande importância. A necessidade de sistemas que não sejam muito intrusivos, em tempo real e fáceis de usar é necessária e tem sido objecto de muitos estudos recentes de investigação no domínio HAR.
Neste trabalho vamos considerar dois cenários distintos de classificação:

- Cenário A - Problema Binário: pretendemos apenas distinguir se uma pessoa está a andar ou está parada.
- Cenário B - Problema Multiclass: pretendemos distinguir seis atividades distintas, são elas: andar, subir escadas, descer escadas, sentar, parado e deitado.

Vamos então proceder à aplicação de técnicas de pré-processamento, redução e seleção de features e classificação com vista a atingir os objetivos acima referidos. Para este trabalho optámos por utilizar a linguagem Python recorrendo a bibliotecas como *Pandas* e *scikit-learn*.

3 DESCRIÇÃO DOS DADOS

No âmbito deste Projeto foi fornecido um Dataset que contém informação de 30 voluntários, entre os 19 e os 48 anos. A cada participante foi pedido para executar atividades simples como:

- Andar
- Subir escadas
- Descer escadas
- Sentar
- Parado, em pé
- Deitado

Os participantes usaram um smartphone na cintura. Foram usados os seus acelerômetro e giroscópio embutidos, que permitem a aquisição de aceleração linear 3-axial e 3-axial velocidade angular a uma taxa de amostragem de 50Hz. A anotação foi realizada manualmente com base em um vídeo gravado em paralelo com os dados do telemóvel. O conjunto de dados obtido foi dividido aleatoriamente em dois conjuntos, onde 70% dos voluntários foram selecionados para gerar os dados de treino e 30% dos dados do teste. O conjunto de dados que nos foi fornecido contém então dados de treino e dados de teste pelo que iremos usar os dados de teste unicamente para testar a performance do classificador após este já estar construído e, com isso, retirar as devidas conclusões. Iremos dividir os dados de treino em grupo de treino (70%) e grupo de validação (30%) e iremos usar ambos os grupos na construção dos classificadores. Os dados de validação servirão unicamente para avaliarmos a qualidade dos classificadores no contexto do problema.

4 PRÉ-PROCESSAMENTO

Começámos por analisar a distribuição das *samples* pelos vários labels com vista a aferir o estado de balanceamento do dataset.

- Andar - Label 1: 1226
- Subir escadas - Label 2: 1073
- Descer escadas - Label 3: 986
- Sentar - Label 4: 1286
- Parado em pé - Label 5: 1374
- Deitado - Label 6: 1407

Como podemos ver, o dataset parece ser balanceado uma vez que o número de samples por label não difere muito.
Foi necessário construir um array que contivesse os labels utilizados no cenário de classificação binário, definimos então:

- Parado - Label 0: 3285
- A andar - Label 1: 4067

Posteriomente, normalizámos os dados de treino obtendo uma nova matriz de dados com média nula e desvio padrão unitário. Separadamente, procedemos também à normalização dos dados de teste utilizando os mesmo pesos que resultaram da normalização dos dados de treino. Para isso recorremos à função preprocessing.StandardScaler() fazendo o fit aos dados de treino, posteriormente aplicamos a mesma aos dados de teste. Caso tivéssemos feito a normalização do dataset completo, isto é, dados de treino e dados de teste estariamos a ter em conta características do conjunto de teste para o pré-processamento dos dados de treino e tal não se pretende pois estaríamos a enviesar os resultados finais.

5 Seleção de Features

Introduzir num classificador um conjunto de características muito elevado, pode necessitar de um poder computacional e tempo de treino demasiado elevado e portanto uma estratégia que deve ser aplicada é a seleção de features.

Nesta secção do trabalho pretendemos então procurar quais as features que possuem um maior poder discriminatório, isto é, aquelas que têm um melhor desempenho a diferenciar os labels existentes. É então fundamental haver uma grande pesquisa no que toca a métodos de seleção de features de modo a perceber qual o que melhor se adequa ao contexto do problema.

Esta parte do trabalho é essencial, uma vez que estamos a retirar informação que não traz vantagens para obter a solução do problema, conseguindo também diminuir o custo computacional, visto que estamos a manipular uma menor quantidade de dados. Tendo em conta que o dataset utilizado é composto por 561 features haverá, certamente, uma grande quantidade de informação redundante que não constitui uma mais valia aquando da construção dos classificadores.

Existem 3 tipos de métodos para seleção de features:

- *Filter Methods* - selecionam as features consoante os scores dos testes estatísticos aplicados;
- *Wrapper Methods* - selecionam diferentes subsets de features sendo que cada um deles é utilizado para construir um modelo que é treinado com um algoritmo de classificação escolhido. O melhor subset de features é escolhido tendo em conta a performance de teste do classificador criado;
- *Embedded Methods* - são uma combinação dos dois modelos atrás descritos.

De modo selecionar caracteristicas mais discriminativas é usual avaliar as features individualmente ou a pares o que, se torna extremamente difícil tendo 561 features. Caso o dataset tivesse uma menor dimensão poderíamos, por exemplo, avaliar cada feature separadamente de modo a perceber se esta separava os labels em questão; poderíamos também fazer scatterplots 2D de duas features (fazendo para todas, duas a duas) de modo a perceber se ambas continham a mesma informação e, consequentemente excluir uma delas. No entanto, para este problema estas técnicas não são robustas pelo que recorremos a outros que passaremos a analisar.

Alguns dos métodos que utilizámos não selecionam, por assim dizer, as melhores features, apenas fazem um ranking. Deste modo, optámos por fazer o seguinte teste, de modo a averiguar qual o número de features que deveríamos selecionar deste ranking: escolhemos um métodos de seleção, o Kruskal Wallis e um método de classificação, o SVM; seguidamente, representámos graficamente a accuracy do classificador em função do número de features selecionadas. No caso do problema binário, o gráfico era praticamente constante (e, por isso, não é apresentado); já para o problema multiclasse, podemos verificar algumas variações nos valores da

accuracy. Optámos por selecionar 20 features, sempre que usamos um modelo de seleção que nos dá um ranking, apesar de ignorarmos grande quantidade de informação consideramos que é importante também ter em conta que existe bastante redundância entre features, deste modo, conseguimos também reduzir a necessidade de elevado poder computacional.

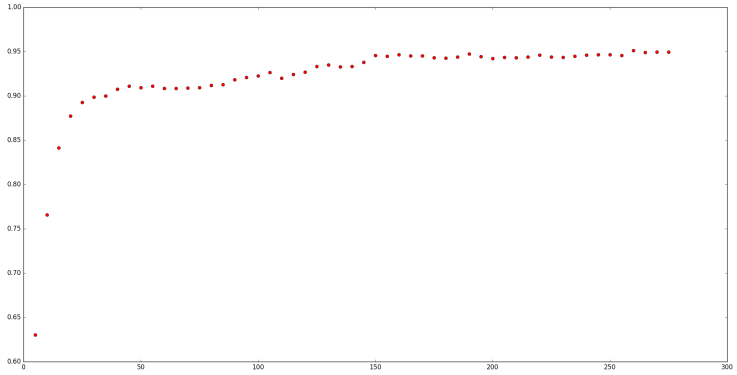

Figure 5.1: Representação da accuracy de um modelo SVM para os diferentes números de features selecionadas.

5.1 Matriz de Correlação

Começámos por calcular uma matriz de correlação que continha os coeficiente de correlação entre cada par de features. Pensámos que seria interessante analisar cada coeficiente em particular de modo a perceber quais os pares de features que apresentavam maior redundância. No entanto, percebemos que esta abordagem não seria a melhor devido à quantidade de features existente. Optámos então por calcular a média de cada linha da matriz de correlação, deste modo, obtínhamos a média dos coeficientes de correlação de uma determinada feature com todas as outras. Talvez esta abordagem não seja a mais correta mas pareceu-nos a mais eficiente no contexto do problema.
Através do vetor que continha a média dos coeficientes de correlação para cada feature, estabelecemos um threshold de 0,03 e considerámos que as features selecionadas correspondiam a todas aquelas que tinham coeficiente de correlação inferior a 0.01.

Table 5.1: Média dos coeficientes de correlação obtidos para cada feature. Deste método foram selecionadas **18** features.

Feature	Coeficiente de Correlação (média)
tBodyAcc-mean()-X	0.00128
tGravityAcc-arCoeff()-Y,4	0.00509
tGravityAcc-arCoeff()-Z,4	0.00137
tBodyAccJerk-mean()-X	0.00810
tBodyAccJerk-mean()-Y	0.00027
tBodyAccJerk-mean()-Z	0.00828
tBodyAccJerk-arCoeff()-X,2	0.00529
tBodyGyro-mean()-Y	0.00835
tBodyGyroJerk-mean()-Y	0.00722
tBodyGyroJerk-mean()-Z	0.00554
tBodyGyroJerk-arCoeff()-X,4	0.00170
tBodyGyroJerk-arCoeff()-Y,4	0.00327
tBodyGyroJerk-correlation()-X,Z	0.00055
tBodyGyroJerk-correlation()-Y,Z	0.00632
tBodyGyroMag-arCoeff()	0.00098
fBodyGyro-maxInds-X	0.00467
fBodyAccMag-meanFreq()	0.00724
angle(tBodyAccJerkMean),gravityMean)	0.00889

5.2 Correlação entre feature e label

Decidimos também calcular os valores de correlação entre cada feature e os labels, tanto para abordagem binária como para a abordagem multiclasse. Definimos um threshold para o coeficiente de correlação e selecionámos todas as features que estavam acima desse threshold. Um valor alto de correlação indica que a feature é capaz de discriminar entre classes.

Table 5.2: Correlação entre features e label para o cenário de classificação binário (A), utilizando um threshold de 0.94, tendo sido selecionadas **22** features

Feature	Coeficiente de Correlação
tBodyAcc-std()-Y	0.9424
tBodyAcc-sma()	0.9447
tBodyAccJerk-entropy()-X	0.9673
tBodyAccJerk-entropy()-Y	0.9611
tBodyAccJerk-entropy()-Z	0.9613
tBodyGyroJerk-entropy()-Z	0.9469
tBodyAccMag-mean()	0.9417
tBodyAccMag-sma()	0.9417
tGravityAccMag-mean()	0.9417
tGravityAccMag-sma()	0.9417
tBodyAccJerkMag-entropy()	0.9683
fBodyAcc-mean()-Y	0.9406
fBodyAcc-mad()-Y	0.9448
fBodyAcc-sma()	0.9436
fBodyAcc-entropy()-X	0.9664
fBodyAcc-entropy()-Y	0.9527
fBodyAcc-entropy()-Z	0.9441
fBodyAccJerk-entropy()-X	0.9773
fBodyAccJerk-entropy()-Y	0.9719
fBodyAccJerk-entropy()-Z	0.9564
fBodyAccMag-entropy()	0.9486
fBodyBodyAccJerkMag-entropy()	0.9641

Para os labels multiclasse procurámos ajustar o nosso threshold de modo a obter o mesmo número de features selecionadas do que no cenário binário.

Table 5.3: Correlação entre features e label para o cenário de classificação multiclasse (B), utilizando um threshold de 0.811, tendo sido selecionadas **22** features.

Feature	Coeficiente de Correlação
tBodyAcc-std()-Y	0.8179
tBodyAcc-sma()	0.8175
tBodyAccJerk-mad()-Y	0.8169
tBodyAccJerk-iqr()-Y	0.8134
tBodyAccJerk-entropy()-Y	0.8575
tBodyAccJerk-entropy()-Z	0.8534
tBodyGyroJerk-entropy()-Z	0.8594
tBodyAccJerkMag-entropy()	0.8429
tBodyGyroJerkMag-entropy()	0.8476
fBodyAcc-mean()-Y	0.8121
fBodyAcc-std()-Y	0.8133
fBodyAcc-mad()-Y	0.8138
fBodyAcc-entropy()-X	0.8160
fBodyAcc-sma()	0.8249
fBodyAcc-entropy()-Y	0.8398
fBodyAcc-entropy()-Z	0.8258
fBodyAccJerk-entropy()-X	0.8447
fBodyAccJerk-entropy()-Y	0.8536
fBodyAccJerk-entropy()-Z	0.8324
fBodyGyro-entropy()-Z	0.8223
fBodyBodyAccJerkMag-entropy()	0.8295
fBodyBodyGyroJerkMag-entropy()	0.8119

5.3 Teste de Kruskal Wallis

Aplicámos também o método Kruskal-Wallis implementando-o em Python. Aplicámos duas vezes o método, tanto para o cenário A como para o cenário B.
O método de Kruskal-Wallis, é um método não paramétrico (pertencendo por isso aos *Filter Methods*) que testa se duas ou mais features têm mediana igual e dá-nos um valor de P. Deste modo, a hipótese nula é que as duas features têm mediana igual. Se o valor de P estiver próximo de 0 significa que a feature apresenta informação discriminativa, caso contário essa feature não é selecionada. Da aplicação no método obtemos um ranking de features consoante a sua capacidade discriminativa.
Iremos apresentar dois valores obtidos pelo teste: o score, que corresponde ao *Kruskal-Wallis H statistic*; o p-Value, que testa a hipótese nula.

Table 5.4: Scores obtidos pelo Kruskal-Wallis para o cenário de classificação binária (A), selecionando as 20 primeiras features mais discriminativas.

Feature	Score	p-Value
fBodyAccJerk-entropy()-X	639.8716	3.5634e-141
fBodyGyro-entropy()-X	626.7263	2.5752e-138
tBodyGyroMag-entropy()	618.3131	1.7403e-136
BodyGyroJerk-entropy()-X	617.7203	2.3419e-136
fBodyAcc-entropy()-X	604.4305	1.8202e-133
fBodyAcc-entropy()-Y	598.5105	3.5300e-132
tBodyAccJerk-entropy()-X	589.5131	3.1974e-130
tBodyAccJerk-entropy()-Y	554.3488	1.4254e-122
angle(Y,gravityMean)	531.9295	1.0744e-117
tBodyAccMag-entropy()	491.7001	6.0797e-109
tGravityAccMag-entropy()	491.7001	6.0797e-109
tBodyAccJerkMag-entropy()	448.9841	1.2000e-099
GravityAcc-correlation()-Y,Z	442.7595	2.7156e-098
angle(Z,gravityMean)	434.4981	1.7056e-096
tBodyGyroJerkMag-entropy()	401.6305	2.4321e-089
tGravityAcc-correlation()-X,Y	378.1004	3.2249e-084
tGravityAcc-energy()-X	267.0018	5.1092e-060
tGravityAcc-min()-X	188.2858	7.5249e-043
tGravityAcc-mean()-X	181.2981	2.5233e-041
tGravityAcc-max()-X'	179.5719	6.0102e-041

Table 5.5: Scores obtidos pelo Kruskal-Wallis para o cenário de classificação multiclasse (B), selecionando as 20 primeiras features mais discriminativas.

Feature	Score
fBodyAccJerk-entropy()-Z	9247.6415
fBodyGyro-entropy()-X	9228.1588
fBodyBodyAccJerkMag-entropy()	9222.7484
fBodyAccMag-entropy()	9216.5473
tBodyAccMag-entropy()	9199.6617
tGravityAccMag-entropy()	9199.6617
tGravityAcc-min()-Y	9192.2211
angle(X,gravityMean)	9185.1329
tGravityAcc-max()-Y	9185.0188
tBodyGyroJerk-entropy()-Z	9183.1922
tGravityAcc-mean()-Y	9177.3569
tBodyAccJerk-entropy()-Z	9175.8759
fBodyAccJerk-entropy()-Y	9081.9451
fBodyAcc-entropy()-Y	9081.3269
BodyAccJerk-entropy()-X	9074.4488
fBodyAcc-entropy()-X	9016.3991
tBodyAccJerk-entropy()-X	8972.2503
tBodyAccJerk-entropy()-Y	8952.3559
tBodyGyroJerkMag-entropy()	8915.3478
BodyAccJerkMag-entropy()	8786.9384

Relativamente ao cenário os valores para o p-Value devolvidos eram todos nulos daí não serem apresentados.

5.4 Seleção de Features recorrendo a Extra Trees Classifier

Trata-se de um método embbedded para seleção de features, isto é, é utilizado um algoritmo de classificação, neste caso, o Extra Trees Classifier, para selecionar as features que apresentem maior poder discriminatório na classificação.

Table 5.6: Nomes das **20** features selecionadas recorrendo a Extra Trees Classifiers tanto para o problema binário (A) como para o problema multiclasse (B).

Features selecionada	
Problema Binário	**Problema Multiclasse**
fBodyAccJerk-mean()-Y	tGravityAcc-max()-X
fBodyBodyAccJerkMag-mad()	fBodyAcc-max()-X
fBodyAcc-mean()-Y	tGravityAcc-mean()-Y
tBodyAcc-std()-X	tGravityAcc-mean()-X
fBodyBodyAccJerkMag-std()	tGravityAcc-min()-Y
fBodyGyro-entropy()-Y	tGravityAcc-min()-X
fBodyAcc-bandsEnergy()-1,16	fBodyAcc-mean()-X
fBodyAcc-mean()-X	tGravityAcc-energy()-Y
tBodyAcc-iqr()-Y	tBodyAcc-iqr()-X
tBodyGyroJerkMag-sma()	fBodyAcc-entropy()-Y
tBodyAccMag-arCoeff()1	fBodyAcc-entropy()-Z
fBodyAccJerk-bandsEnergy()-1,24	tBodyGyro-std()-Y
fBodyAccJerk-mean()-X	fBodyBodyAccJerkMag-sma()
fBodyAcc-energy()-X	fBodyGyro-bandsEnergy()-1,16
tBodyAccJerk-max()-X	fBodyAcc-iqr()-Y
tBodyGyro-min()-Z	tBodyAccJerkMag-iqr()
tBodyAccJerk-iqr()-Z	tGravityAcc-min()-Z
fBodyGyro-sma()	tGravityAcc-energy()-Z
fBodyAccJerk-entropy()-X	tBodyAcc-std()-X
fBodyBodyGyroMag-iqr()	tBodyAccMag-mad()

Este método dá-nos um ranking de features daí termos selecionado as 20 primeiras mais discriminativas.

5.5 LASSO CV

O método LASSO - Least Absolute Shrinkage and Selection Operator - está também incluído nos métodos embedded para seleção de features. A abordagem deste método consiste em construir um modelo linear que penaliza os coeficientes da regressão com um L1 *penalty* , fazendo com que muitos deles tomem o valor de zero. As features com coeficientes diferentes de zero são selecionadas.

Table 5.7: Nomes das features selecionadas ao método LASSO CV tanto para o problema binário (A) como para o problema multiclasse (B).

Features selecionada	
Problema Binário	**Problema Multiclasse**
tBodyAcc-std()-X	tGravityAcc-min()-X
tBodyAcc-std()-Y	tGravityAcc-energy()-Y
tBodyAccJerk-entropy()-X	tGravityAcc-energy()-Z
tGravityAcc-std()-X	tBodyAccJerk-entropy()-X
tBodyAccMag-energy()	tBodyAccJerk-entropy()-Z
tBodyGyroMag-mean()	tBodyGyroJerk-mad()-X
tBodyGyroJerkMag-entropy()	tBodyGyroJerk-mad()-Z
fBodyAcc-std()-X	tBodyGyroJerk-entropy()-X
fBodyAcc-std()-Y	tBodyAccMag-std()
fBodyAcc-std()-Z	fBodyAcc-std()-Y
fBodyAcc-mad()-X	fBodyAcc-std()-Z
fBodyAcc-mad()-Y	fBodyAcc-bandsEnergy()-1,8
fBodyAcc-energy()-X	fBodyGyro-std()-Y
fBodyAcc-energy()-Y	fBodyGyro-entropy()-X
fBodyAcc-bandsEnergy()-1,24	fBodyBodyAccJerkMag-std()
fBodyAccJerk-entropy()-X	fBodyBodyGyroJerkMag-entropy()
fBodyBodyGyroJerkMag-entropy()	angle(Y,gravityMean)

Neste método, fizemos variar o threshold de modo a selecionar um número de features idêntico ao que havíamos selecionado pelos restantes métodos, isto porque, o método nos dá as features mais discriminativas do conjunto de dados.

5.6 RFE - Recursive Feature Elimination com SVM

Support Vector Machine- Recursive Feature Elimination é um método de seleção de features que pertence ao grupos do wrappers e que, é referido na literatura como sendo uma técnica bastante eficiente. Este método começa por treinar um modelo SVM com todas as features, calculando o peso de cada uma no modelo. As features com menores pesos são eliminadas e é construido novo modelo. Este processo é repetido até que seja selecionado um determinado número de features. Como é de esperar, este método exige um elevado poder computacional e, no contexto do problema, torna-se num processo bastante demorado tendo em conta o número de features que dispomos pelo que não foi utilizado no nosso estudo.

5.7 ROC - Area Under Curve (AUC)

Para a abordagem binária é ainda possível aplicar uma técnica que seleciona as features tendo em conta a área da curva ROC (Receiving Operating Characteristic). Esta curva consiste numa representação dos valores da sensibilidade (SS - True positive rate) em função de 1-Especificidade (1-SP False positive rate). Estes valores são calculados mediante a variação de

um threshold que indica qual a *classe 0* e a *classe 1*.
Esta técnica é utilizada assumindo que as amostras pertence a um conjunto de um variável contínua aleatória sendo, por isso, possível recorrer a funções de densidade de probabilidade.

$$Sensibilidade = \frac{TP}{TP+FN} * 100\%$$

$$Especificidade = \frac{TN}{TN+FP} * 100\%$$

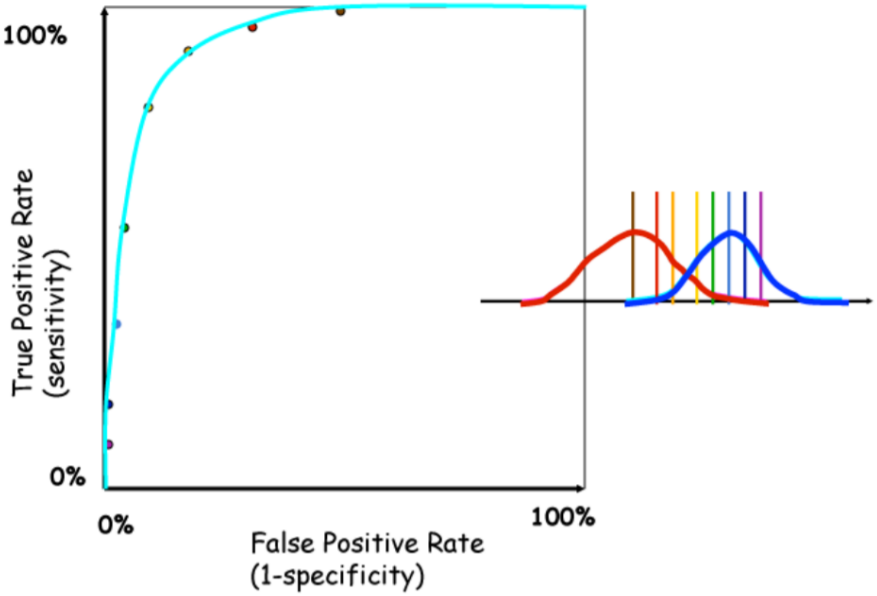

Figure 5.2: Exemplo ilustrativo do processo de construção da curva ROC.

Table 5.8: Valores de AUC obtidos para o cenário de classificação binária (A), selecionando as 20 primeiras features mais discriminativas.

Feature	AUC
fBodyAccJerk-bandsEnergy()-1,16	0.99999955
fBodyAccJerk-max()-X	0.9999982
fBodyAccJerk-bandsEnergy()-1,8	0.99999648
tBodyAccJerk-mad()-X	0.99999528
tBodyAccJerk-iqr()-X	0.99999499
fBodyAccJerk-bandsEnergy()-1,24	0.99999349
fBodyAccJerk-std()-X	0.99999274
tBodyAccJerk-energy()-X	0.99998997
fBodyAccJerk-energy()-X	0.9999899
tBodyAccJerk-std()-X	0.9999899
fBodyAccJerk-entropy()-X	0.99998915
fBodyAccJerk-mad()-X	0.9999854
fBodyAccJerk-mean()-X	0.99998301
tBodyGyroJerk-iqr()-Z	0.99998271
tBodyAccJerkMag-sma()	0.99997515
tBodyAccJerkMag-mean()	0.99997515
tBodyAccJerk-sma()	0.99997246
fBodyAccJerk-bandsEnergy()-9,16	0.99997118
tBodyAccJerkMag-entropy()	0.99997096
tBodyAcc-max()-X	0.99996534

6 Redução de Dimensão

Esta etapa do trabalho tem como objetivo reduzir a dimensionalidade de um grande conjunto de dados. Quando o dataset que temos disponível apresenta uma grande dimensionalidade podemos recorrer a técnicas deste tipo de modo a diminuir o poder computacional requerido, eliminando informação redundante.

Outra vantagem referente à utilização de métodos de redução de dimensão prende-se com a possibilidade de vizualizar a padrões a duas ou três dimensões. É, por isso, nesta secção que iremos apresentar representações gráficas do dataset após este ter sido reduzido.

Aplicámos então três técnicas de redução de dimensão: PCA (Principal Component Analysis), LDA (Linear Discriminant Analysis) e MDS (Multi Dimensional Scaling).

6.1 PCA - Principal Component Analysis

O PCA é utilizado para encontrar as melhores projeções, de forma, a obter a melhor representação da estrutura de dados, retendo o máximo de informação, medida em termos da variabilidade dos dados, num número mínimo de dimensões, que são escolhidas mudando o sistema de eixos. Para isso, esta técnica encontra o eixo onde a variância apresenta maiores valores e de seguida procura os eixos perpendiculares onde a variância apresenta também valores elevados.

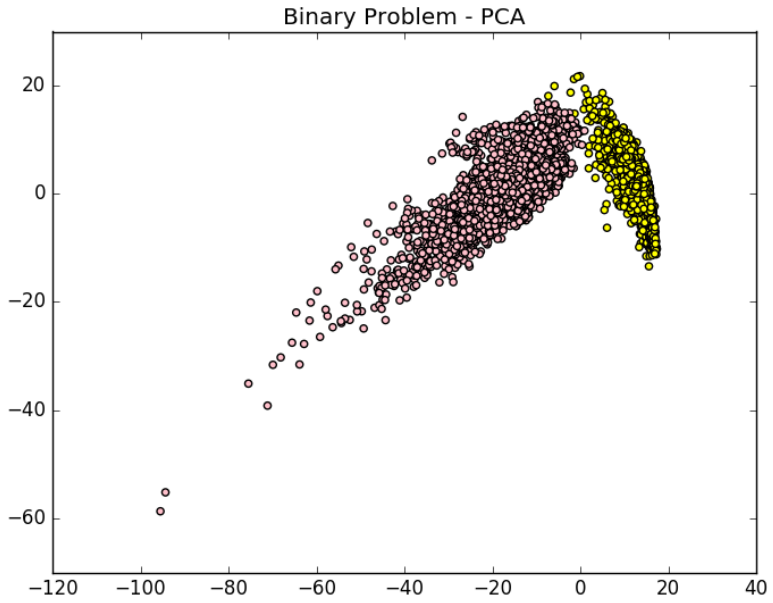

Figure 6.1: Resultado da aplicação do PCA com duas dimensões, no caso do cenário de classificação binária.

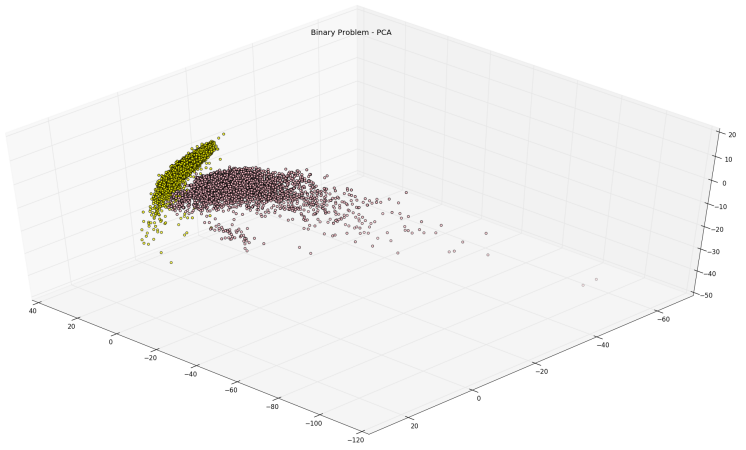

Figure 6.2: Resultado da aplicação do PCA com três dimensões, no caso do cenário de classificação binária.

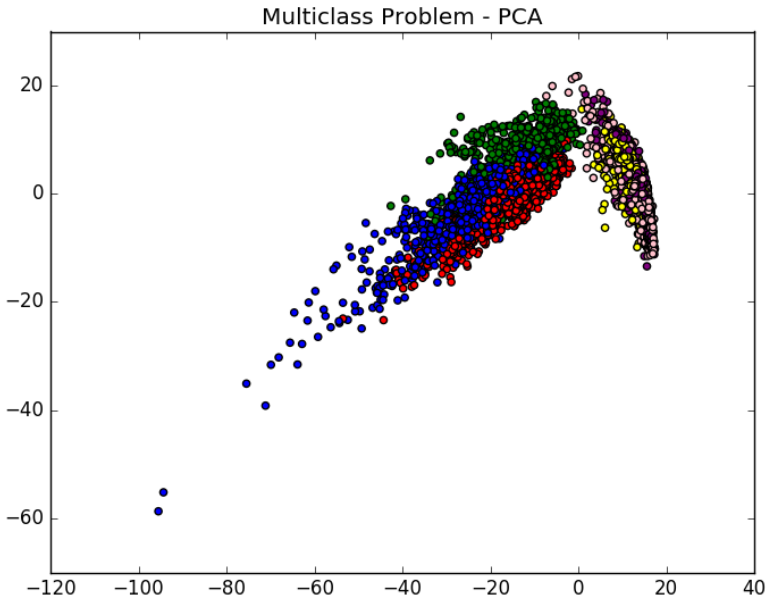

Figure 6.3: Resultado da aplicação do PCA com duas dimensões, no caso do cenário de classificação multiclasse.

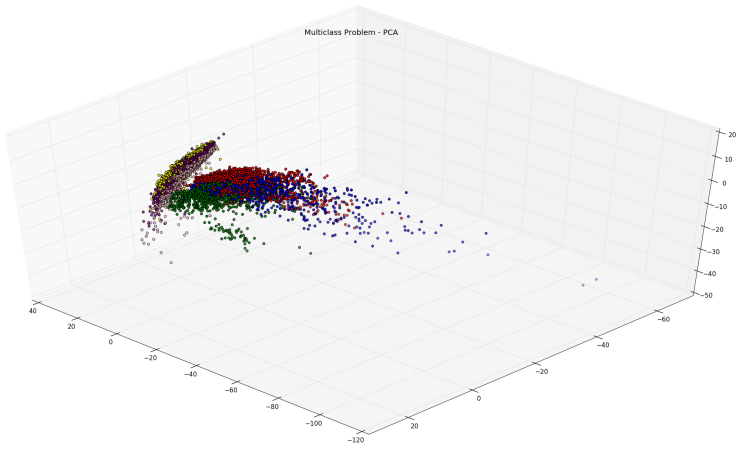

Figure 6.4: Resultado da aplicação do PCA com três dimensões, no caso do cenário de classificação multiclasse.

Relativamente às representações gráficas dos resultados do PCA podemos ver que no caso do problema binário conseguimos separar perfeitamente os dois grupos. Já no problema multiclasse existe sobreposição de alguns pontos de classes distintas o que poderá gerar algumas complicações na fase da classificação.

Com o objetivo de estudar qual o número de dimensões que deveríamos retirar do método PCA, recorremos à função *pca.explained_variance_ratio_* que nos dá quantidade de informação contida em cada dimensionalidade, em percentagem.
É necessário ter em conta que o PCA lida apenas com as features, isto é, consiste num método de redução de dimensionalidade não supervisionado e, por isso, procurámos também interpretar os seus resultados tendo em conta ambos os cenários de classificação que estamos a estudar. Para isso, recorremos a um modelo de regressão linear que aplicámos a cada componente do PCA, individualmente, e ao array de labels que estávamos a estudar. Para cada iteração foi calculado o MSE e esses valores foram representados graficamente.

Table 6.1: Quantidade de informação presente em cada dimensão consoante o número de dimensões escolhidas.

Dimensionalidade	Quantidade de informação (%)
1	50.78
2	57.36
3	60.17
4	62.67
5	64.56
6	66.28
7	67.65
8	68.85
9	69.85
10	70.82
11	71.68
12	72.48
13	73.24
14	73.89
15	74.52
16	75.12
17	75.71
18	76.29
19	76.86
20	77.39
21	77.89
22	78.38
23	78.86
24	79.33
25	79.78
26	80.2
27	80.62
28	81.03
29	81.42
30	81.81

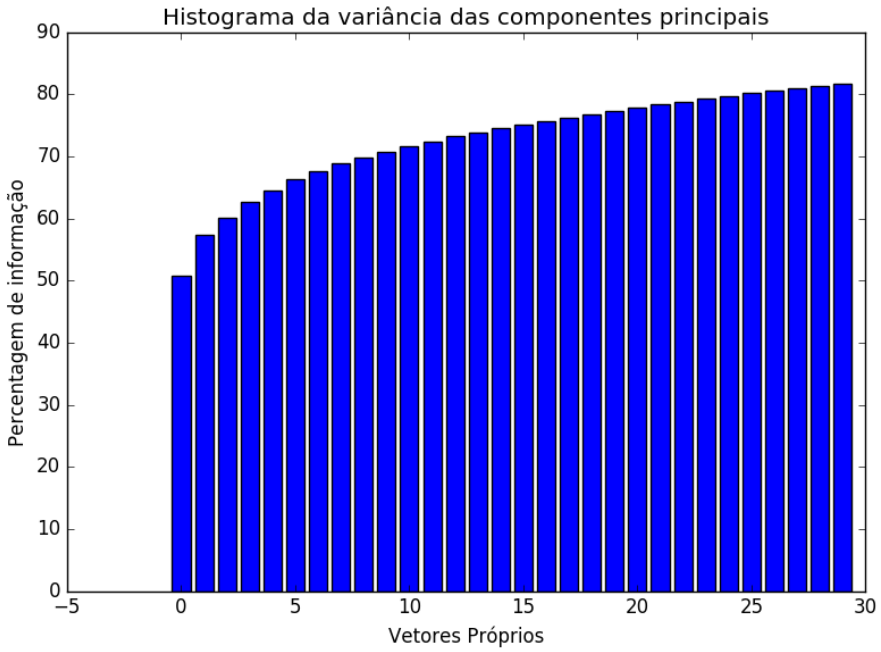

Figure 6.5: Representação gráfica da quantidade de informação presente em cada dimensão consoante o número de dimensões escolhidas.

Figure 6.6: Representação dos valores obtidos para o MSE aquando da aplicação de um modelo de regressão linear a 100 componentes principais e às labels.

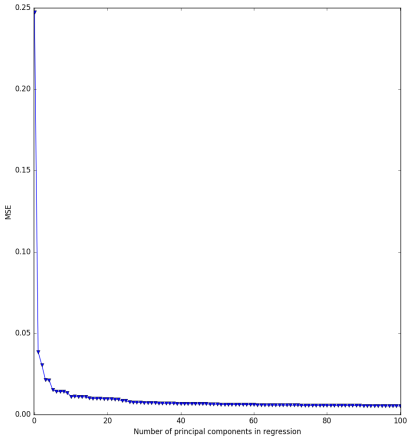

Figure 6.7: Problema binário.

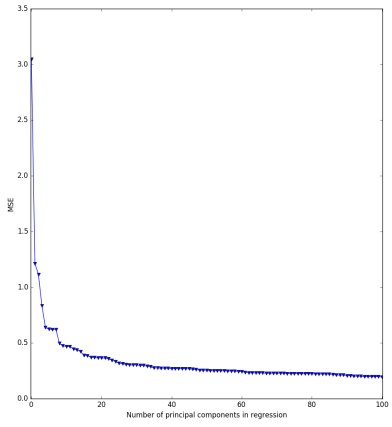

Figure 6.8: Problema multiclasse.

Figure 6.9: Representação dos valores obtidos para o MSE aquando da aplicação de um modelo de regressão linear a 30 componentes principais e às labels.

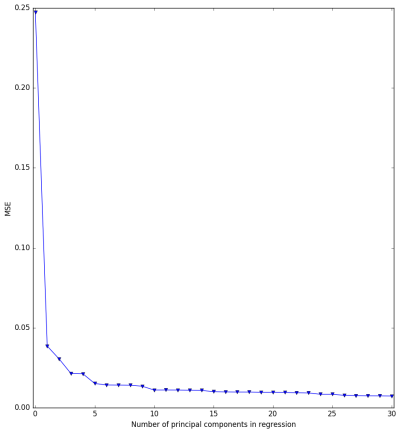

Figure 6.10: Problema binário.

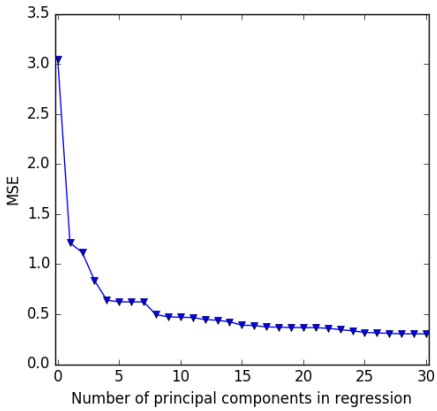

Figure 6.11: Problema multiclasse.

6.2 MDS - MULTIDIMENSIONAL SCALING

Nesta técnica são usadas definições de distâncias. As distância exprimem semelhanças ou dissemelhanças. Ao visualizarmos os dados pretendemos aferir quão próximos ou distantes estão os pontos uns dos outros. O MDS produz então uma representação dos dados num número reduzido de dimensões tendo por base as distâncias emparelhadas. Neste caso, utilizámo o MDS métrico que cria uma representação visual do padrão das semelhanças entre o conjunto de pontos. As semelhanças podem-se exprimir pelas distâncias pelo que podemos dizer que esta duas são proporcionais.
As distâncias emparelhadas podem ser calculadas usando várias definições: euclediana, city block, mahalanobis e chebychev. Neste caso utilizámos a distância euclediana. Existem ainda outros critérios que podem ser variados, sendo que o stress é o mais utilizado.
Este método apresenta uma grande vantaem que se prende com o elevado poder computacional que requer, apesar disso, conseguimos implementá-lo.

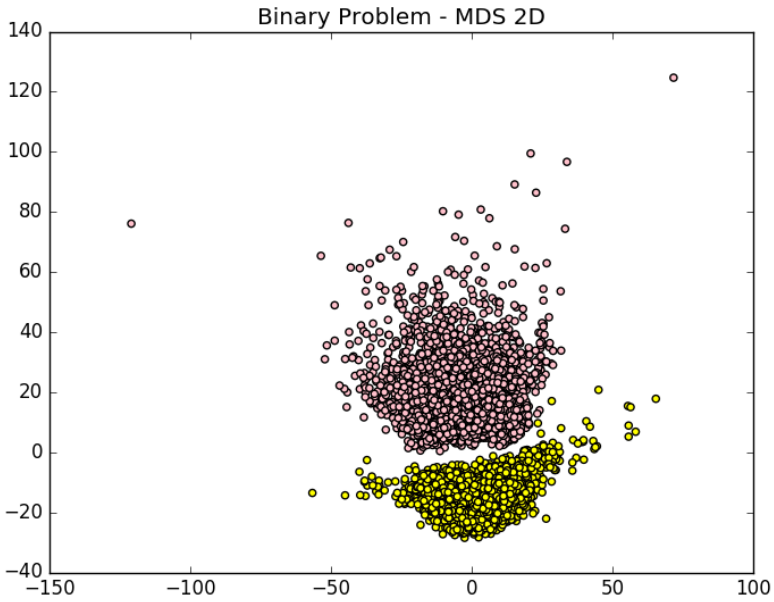

Figure 6.12: Resultado da aplicação do MDS com duas dimensões, no caso do cenário de classificação binária.

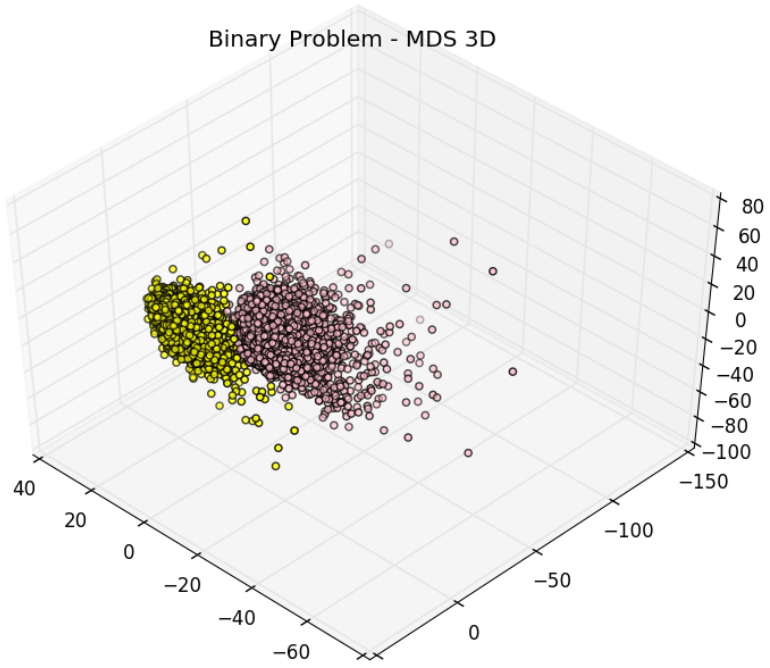

Figure 6.13: Resultado da aplicação do MDS com três dimensões, no caso do cenário de classificação binária.

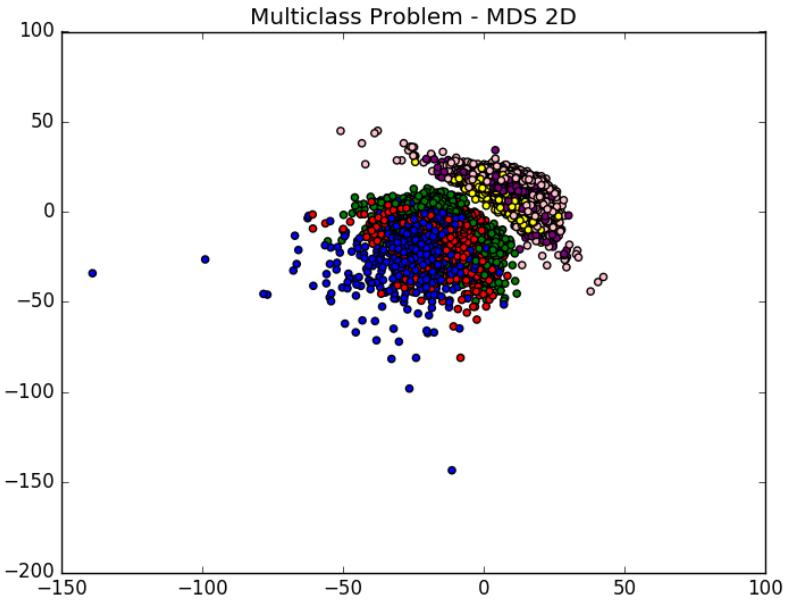

Figure 6.14: Resultado da aplicação do MDS com duas dimensões, no caso do cenário de classificação multiclasse.

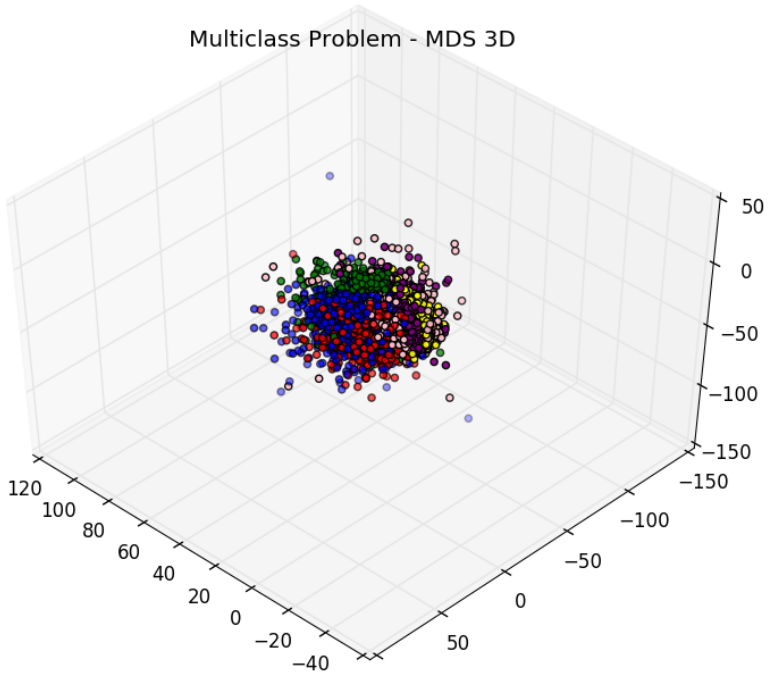

Figure 6.15: Resultado da aplicação do MDS com três dimensões, no caso do cenário de classificação multiclasse.

6.3 LDA E QDA - LINEAR AND QUADRATIC DISCRIMINANT ANALYSIS

Outros métodos que podem ser usados para redução de dimensionalidade consistem em métodos de análise discriminante, como por exemplo, Análise Discriminante Linear (LDA) e Análise Discriminante Quadrática (QDA).
Estes métodos fazem redução de dimensionalidade e, posteriormente, classificam com base numa função de decisão. Deste modo, os resultados dos mesmos serão apenas apresentados na próxima secção.
Relativamente à aplicação do LDA, aplicámo-la apenas com duas dimensões. O LDA consiste numa análise discriminativa que devolve a probabilidade das samples pertencerem à primeira ou à segunda classe, daí apenas resultar um vetor. Também com este método conseguimos separar os dados em dois grupos, no cenário binário. Assim como no PCA, no cenário multiclasse existe alguma mistura de pontos de classes distintas.

7 ANÁLISE DOS MÉTODOS DE SELEÇÃO DE FEATURES E REDUÇÃO DE DIMENSÃO

Table 7.1: Métricas de classificação quando não é aplicado qualquer método de seleção de features, para o problema binário.

Problema	Classificador	Accuracy	Precisão	Sensibilidade	Especificidade
Binário	MDC-Euclideana	0.998	0.996	0.997	1
				1	0.997
	MDC-Mahalanobis	0.999	0.999	0.999	0.999
				0.999	0.999
	kNN	0.999	0.999	0.999	1
				1	0.999
	Naive Bayes	0.990	0.980	0.982	1
				1	0.982
	Random Forest	0.999	0.998	0.998	1
				1	0.998
	LDA	0.999	0.999	0.999	1
				1	0.999
	QDA	0.988	0.975	0.978	1
				1	0.978
	SVM	0.999	0.999	0.999	1
				1	0.999

Table 7.2: Métricas de classificação quando não é aplicado qualquer método de seleção de features, para o problema multiclasse.

Problema	Classificador	Accuracy	Precisão	Sensibilidade	Especificidade
Multidimensional	MDC-Euclideana	0.532	0.539	0	1
				0	0.999
				1	0.612
				0.409	0.965
				0.780	0.969
				0.993	0.902
	MDC-Mahalanobis	0.839	0.851	0.935	0.962
				0.915	0.960
				0.630	0.993
				0.759	0.954
				0.887	0.937
				0.872	1
	kNN	0.865	0.875	0.984	0.999
				0.864	0.957
				0.705	0.962
				0.807	0.998
				0.878	0.975
				0.924	0.947
	Naive Bayes	0.712	0.779	0.859	0.964
				0.962	0.946
				0.612	0.979
				0.849	0.797
				0.677	0.974
				0.344	0.994
	Random Forest	0.924	0.924	0.951	0.976
				0.892	0.979
				0.861	0.992
				0.898	0.983
				0.921	0.979
				1	1
	LDA	0.964	0.965	0.992	0.995
				0.979	0.992
				0.959	1
				0.894	0.990
				0.957	0.979
				1	1
	QDA	0.829	0.851	0.558	0.989
				0.996	0.931
				0.859	0.943
				0.664	0.993
				0.891	0.941
				1	0.999
	SVM	0.954	0.955	0.995	0.990
				0.968	0.989
				0.921	0.999
				0.868	0.991
				0.962	0.974
				1	1

Table 7.3: Métricas de classificação quando é feita seleção de features através da matriz de correlação para o problema binário.

Problema	Classificador	Accuracy	Precisão	Sensibilidade	Especificidade
Binário	MDC-Euclideana	0.546	0.516	0.509	0.588
				0.588	0.509
	MDC-Mahalanobis	0.546	0.516	0.521	0.576
				0.575	0.520
	kNN	0.834	0.990	0.994	0.653
				0.653	0.994
	Naive Bayes	0.955	0.949	0.955	0.955
				0.955	0.955
	Random Forest	0.975	0.972	0.975	0.975
				0.975	0.975
	LDA	0.555	0.550	0.782	0.299
				0.299	0.782
	QDA	0.956	0.943	0.948	0.965
				0.965	0.948
	SVM	0.586	0.994	0.999	0.122
				0.122	0.999

Table 7.4: Métricas de classificação quando é feita seleção de features através da matriz de correlação para o problema multiclasse.

Problema	Classificador	Accuracy	Precisão	Sensibilidade	Especificidade
Multiclasse	MDC-Euclideana	0.319	0.285	0.564	0.739
				0.957	0.568
				0.171	0.952
				0	1
				0.037	0.987
				0.171	0.929
	MDC-Mahalanobis	0.268	0.286	0.300	0.773
				0.119	0.958
				0.259	0.895
				0.409	0.756
				0.289	0.840
				0.225	0.897
	kNN	0.495	0.545	0.615	0.901
				0.556	0.954
				0.248	0.983
				0.477	0.805
				0.598	0.813
				0.441	0.934
	Naive Bayes	0.542	0.566	0.875	0.942
				0.817	0.935
				0.469	0.971
				0.126	0.934
				0.876	0.678
				0.101	0.984
	Random Forest	0.677	0.675	0.923	0.953
				0.753	0.949
				0.650	0.958
				0.479	0.896
				0.573	0.909
				0.689	0.946
	LDA	0.340	0.337	0.366	0.833
				0.503	0.873
				0.180	0.944
				0.216	0.877
				0.374	0.829
				0.379	0.849
	QDA	0.572	0.601	0.830	0.947
				0.794	0.929
				0.559	0.964
				0.175	0.927
				0.849	0.736
				0.240	0.979
	SVM	0.289	0.382	0.542	0.802
				0.637	0.838
				0.193	0.944
				0.181	0.917
				0.334	0.898
				0.425	0.863

Table 7.5: Métricas de classificação quando é feita seleção de features através do método *Feature Importance using Tree Classifiers* para o problema binário.

Problema	Classificador	Accuracy	Precisão	Sensibilidade	Especificidade
Binário	MDC-Euclideana	0.998	0.997	0.997	1
				1	0.997
	MDC-Mahalanobis	0.999	0.999	0.999	1
				1	0.999
	kNN	1	1	1	1
				1	1
	Naive Bayes	0.991	0.982	0.984	1
				1	0.984
	Random Forest	1	1	1	1
				1	1
	LDA	1	1	1	1
				1	1
	QDA	0.992	0.984	0.985	1
				1	0.9485
	SVM	1	1	1	1
				1	1

Table 7.6: Métricas de classificação quando é feita seleção de features através do método *Feature Importance using Tree Classifiers* para o problema multiclasse.

Problema	Classificador	Accuracy	Precisão	Sensibilidade	Especificidade
Multidimensional	MDC-Euclideana	0.307	0.095	0	1
				0	1
				1	0.609
				0.989	0.571
				0	1
				0	1
	MDC-Mahalanobis	0.255	0.558	0.083	0.998
				0.072	0.999
				0.024	0.999
				0.629	0.592
				0.492	0.758
				0.179	0.749
	kNN	0.658	0.676	0.927	0.943
				0.815	0.956
				0.648	0.991
				0.538	0.853
				0.577	0.899
				0.471	0.945
	Naive Bayes	0.524	0.521	0.643	0.953
				0.843	0.904
				0.621	0.968
				0.656	0.744
				0.415	0.885
				0.046	0.974
	Random Forest	0.744	0.751	0.953	0.951
				0.779	0.964
				0.738	0.988
				0.546	0.907
				0.692	0.914
				0.756	0.966
	LDA	0.665	0.685	0.964	0.956
				0.887	0.979
				0.759	0.996
				0.462	0.871
				0.647	0.838
				0.326	0.957
	QDA	0.603	0.614	0.893	0.979
				0.837	0.962
				0.807	0.971
				0.238	0.886
				0.806	0.746
				0.101	0.980
	SVM	0.721	0.747	0.958	0.977
				0.907	0.984
				0.864	0.989
				0.529	0.869
				0.659	0.866
				0.464	0.976

Table 7.7: Métricas de classificação quando é feita seleção de features através do método *LASSO* para o problema binário.

Problema	Classificador	Accuracy	Precisão	Sensibilidade	Especificidade
Binário	MDC-Euclideana	0.995	0.990	0.991	1
				1	0.991
	MDC-Mahalanobis	0.997	0.993	0.993	1
				1	0.993
	kNN	1	1	1	1
				1	1
	Naive Bayes	0.997	0.994	0.995	1
				1	0.995
	Random Forest	1	1	1	1
				1	1
	LDA	0.999	0.999	0.999	1
				1	0.999
	QDA	0.998	0.996	0.997	1
				1	0.997
	SVM	0.999	0.999	0.999	1
				1	0.999

Table 7.8: Métricas de classificação quando é feita seleção de features através do método *LASSO* para o problema multiclasse.

Problema	Classificador	Accuracy	Precisão	Sensibilidade	Especificidade
Multidimensional	MDC-Euclideana	0.313	0.416	0	1
				0	1
				1	0.607
				0.974	0.581
				0.008	1
				0.037	0.997
	MDC-Mahalanobis	0.336	0.533	0.421	0.948
				0.116	0.993
				0.067	1
				0.605	0.709
				0.617	0.596
				0.136	0.949
	kNN	0.655	0.668	0.855	0.904
				0.573	0.947
				0.669	0.982
				0.543	0.888
				0.611	0.908
				0.679	0.956
	Naive Bayes	0.481	0.511	0.857	0.872
				0.446	0.942
				0.569	0.977
				0.020	0.993
				0.932	0.608
				0.069	0.977
	Random Forest	0.714	0.718	0.883	0.919
				0.651	0.953
				0.674	0.983
				0.523	0.923
				0.693	0.919
				0.838	0.959
	LDA	0.659	0.682	0.867	0.919
				0.760	0.953
				0.645	0.994
				0.515	0.902
				0.695	0.861
				0.484	0.959
	QDA	0.602	0.632	0.905	0.959
				0.819	0.960
				0.742	0.983
				0.126	0.937
				0.909	0.696
				0.153	0.984
	SVM	0.729	0.742	0.954	0.950
				0.800	0.974
				0.800	0.994
				0.497	0.909
				0.695	0.879
				0.646	0.965

Table 7.9: Métricas de classificação quando é feita seleção de features através do método Kruskal Wallis, para o problema binário.

Problema	Classificador	Accuracy	Precisão	Sensibilidade	Especificidade
Binário	MDC-Euclideana	0.957	0.917	0.919	1
				1	0.919
	MDC-Mahalanobis	0.981	0.961	0.964	1
				1	0.964
	kNN	0.999	0.997	0.997	1
				1	0.997
	Naive Bayes	0.999	0.997	0.997	1
				1	0.997
	Random Forest	1	1	1	1
				1	1
	LDA	0.999	0.999	0.999	1
				1	0.999
	QDA	0.999	0.999	0.999	1
				1	0.999
	SVM	0.999	0.996	0.999	1
				1	0.999

Table 7.10: Métricas de classificação quando é feita seleção de features através do método *Kruskal Wallis* para o problema multiclasse.

Problema	Classificador	Accuracy	Precisão	Sensibilidade	Especificidade
Multidimensional	MDC-Euclideana	0.354	0.283	0.113	0.973
				0	1
				0.988	0.644
				0.073	0.988
				0	1
				1	0.622
	MDC-Mahalanobis	0.499	0.592	0.718	0.721
				0.391	0.889
				0.436	0.903
				0.491	0.934
				0.511	0.952
				0.436	1
	kNN	0.773	0.795	0.821	0.852
				0.597	0.960
				0.431	0.975
				0.769	0.985
				0.927	0.954
				1	1
	Naive Bayes	0.771	0.799	0.822	0.904
				0.658	0.981
				0.636	0.949
				0.491	0.992
				0.957	0.897
				1	1
	Random Forest	0.808	0.812	0.729	0.929
				0.786	0.935
				0.607	0.974
				0.815	0.969
				0.859	0.962
				1	1
	LDA	0.825	0.850	0.897	0.891
				0.626	0.987
				0.690	0.977
				0.721	0.993
				0.962	0.943
				1	1
	QDA	0.677	0.693	0.667	0.863
				0.641	0.889
				0.145	0.968
				0.495	0.996
				0.979	0.896
				1	0.999
	SVM	0.844	0.851	0.885	0.929
				0.739	0.971
				0.676	0.971
				0.778	0.987
				0.939	0.955
				1	1

Table 7.11: Métricas de classificação quando é feita seleção de features através do método *AUC score* para o problema binário.

Problema	Classificador	Accuracy	Precisão	Sensibilidade	Especificidade
Binário	MDC-Euclideana	1	1	1	1
				1	1
	MDC-Mahalanobis	0.997	1	1	0.994
				0.994	1
	kNN	1	1	1	1
				1	1
	Naive Bayes	0.993	0.986	0.987	1
				1	0.987
	Random Forest	1	1	1	1
				1	1
	LDA	0.999	0.999	0.999	1
				1	0.999
	QDA	0.997	0.993	0.993	1
				1	0.993
	SVM	1	1	1	1
				1	1

Table 7.12: Métricas de classificação quando é feita seleção de features através do método correlação entre label e feature para o problema binário.

Problema	Classificador	Accuracy	Precisão	Sensibilidade	Especificidade
Binário	MDC-Euclideana	0.996	0.992	0.993	1
				1	0.993
	MDC-Mahalanobis	0.999	0.999	0.999	1
				1	0.999
	kNN	1	1	1	1
				1	1
	Naive Bayes	0.991	0.981	0.983	1
				1	0.983
	Random Forest	1	1	1	1
				1	1
	LDA	0.999	0.998	0.998	1
				1	0.998
	QDA	0.981	0.961	0.963	1
				1	0.963
	SVM	1	1	1	1
				1	1

Table 7.13: Métricas de classificação quando é feita seleção de features através do método correlação entre label e feature para o problema multiclasse.

Problema	Classificador	Accuracy	Precisão	Sensibilidade	Especificidade
Multidimensional	MDC-Euclideana	0.307	0.095	0	1
				0	1
				1	0.609
				0.988	0.571
				0	1
				0	1
	MDC-Mahalanobis	0.454	0.655	0.452	0.991
				0.263	0.998
				0.188	0.998
				0.613	0.735
				0.647	0.694
				0.495	0.922
	kNN	0.800	0.811	0.935	0.955
				0.837	0.973
				0.795	0.992
				0.713	0.933
				0.840	0.925
				0.687	0.980
	Naive Bayes	0.544	0.545	0.675	0.957
				0.885	0.918
				0.619	0.964
				0.796	0.705
				0.316	0.938
				0.058	0.970
	Random Forest	0.844	0.845	0.915	0.971
				0.883	0.976
				0.843	0.986
				0.733	0.955
				0.859	0.953
				0.834	0.971
	LDA	0.853	0.854	0.989	0.988
				0.947	0.989
				0.948	0.999
				0.676	0.941
				0.808	0.938
				0.775	0.964
	QDA	0.712	0.763	0.867	0.998
				0.925	0.968
				0.907	0.952
				0.741	0.825
				0.626	0.919
				0.287	0.993
	SVM	0.868	0.869	0.991	0.982
				0.911	0.995
				0.957	0.997
				0.719	0.945
				0.863	0.952
				0.782	0.969

Table 7.14: Métricas de classificação quando é feita redução de features através do método PCA, para 2 e 3 componentes, para o problema binário.

Problema	Classificador	Accuracy		Precisão		Sensibilidade		Especificidade	
		3	2	3	2	3	2	3	2
Binário	MDC-Euclideana	0.998	0.997	0.995	0.994	0.996	0.994	1	1
						1	1	0.996	0.994
	MDC-Mahalanobis	0.999	0.999	0.999	0.999	0.999	0.999	0.999	0.999
						0.999	0.999	0.999	0.999
	kNN	1	0.998	1	0.999	1	0.999	1	0.997
						1	0.997	1	0.999
	Naive Bayes	0.990	0.991	0.980	0.981	0.982	0.983	1	1
						1	1	0.982	0.983
	Random Forest	1	0.999	1	1	1	1	1	0.999
						1	0.999	1	1
	LDA	0.999	0.999	0.999	0.991	0.998	0.992	0.998	1
						1	1	1	0.992
	QDA	0.994	0.991	0.988	0.982	0.909	0.984	1	1
						1	1	0.989	0.984
	SVM	1	0.999	1	0.999	1	0.999	1	1
						1	1	1	0.999

Table 7.15: Métricas de classificação quando é feita redução de features através do método PCA, para 2 e 3 componentes, para o problema multiclasse.

Problema	Classificador	Accuracy		Precisão		Sensibilidade		Especificidade	
		3	2	3	2	3	2	3	2
Multidimensional	MDC-Euclideana	0.326	0.325	0.152	0.181	0	0	1	1
						0	0	0.999	0.999
						1	1	0.611	0.609
						0.018	0.018	0.989	0.996
						0	0	1	0.999
						0.993	0.985	0.596	0.590
	MDC-Mahalanobis	0.568	0.498	0.584	0.466	0.653	0.472	0.911	0.922
						0.794	0.866	0.929	0.872
						0.369	0.495	0.956	0.923
						0.348	0.057	0.882	0.969
						0.6503	0.443	0.832	0.877
						0.5679	0.659	0.969	0.833
	kNN	0.671	0.523	0.681	0.535	0.849	0.854	0.890	0.869
						0.758	0.690	0.959	0.972
						0.395	0.331	0.961	0.955
						0.549	0.422	0.888	0.843
						0.614	0.298	0.918	0.854
						0.816	0.924	0.977	0.931
	Naive Bayes	0.595	0.712	0.593	0.779	0.867	0.859	0.877	0.964
						0.873	0.962	0.955	0.946
						0.269	0.612	0.982	0.979
						0.097	0.849	0.937	0.797
						0.863	0.677	0.797	0.974
						0.549	0.344	0.964	0.994
	Random Forest	0.662	0.516	0.670	0.525	0.830	0.841	0.895	0.877
						0.734	0.679	0.967	0.971
						0.421	0.381	0.955	0.953
						0.517	0.332	0.895	0.862
						0.625	0.522	0.904	0.841
						0.797	0.342	0.977	0.912
	LDA	0.661	0.574	0.662	0.496	0.772	0.766	0.910	0.920
						0.826	0.839	0.967	0.954
						0.474	0.490	0.954	0.957
						0.439	0	0.901	1
						0.658	0.613	0.898	0.854
						0.764	0.719	0.961	0.800
	QDA	0.706	0.538	0.721	0.549	0.887	0.889	0.885	0.875
						0.847	0.809	0.976	0.974
						0.383	0.352	0.979	0.975
						0.409	0.228	0.945	0.923
						0.852	0.810	0.884	0.734
						0.797	0.136	0.976	0.959
	SVM	0.713	0.557	0.720	0.573	0.875	0.877	0.895	0.889
						0.807	0.771	0.983	0.982
						0.450	0.431	0.966	0.963
						0.437	0.008	0.938	0.997
						0.812	0.427	0.899	0.897
						0.841	0.804	0.974	0.734

8 Discussão - Seleção de features

Decidimos apresentar a secção anterior de modo a prever qual o melhor método para a seleção de features. É importante referir que, para os resultados intermédios apresentados na secção anterior utilizámos as features que foram apresentadas na secção de seleção de features, isto porque, apesar de serem poucas podemos fazer uma análise intermédia que não exige tanto poder computacional para o treino dos classificadores. No entanto, o nosso objetivo consiste fazer uma seleção intermédia de métodos de seleção (escolher apenas 1) e de métodos de classificação para que possamos apresentar resultados finais mais fundamentados.
Nas tabelas 7.1 e 7.2 podemos verificar que, sem que haja qualquer seleção de features, os resultados da classificação são satisfatórios. Para o problema binário, os classificadores apresentam uma ótima performance. Já no problema multiclasse, podemos verificar que os classificadores com melhore resultados são aqueles obtidos pelos métodos LDA, SVM e Random Forest.
Já nas tabelas 7.3 e 7.4 podemos observar os resultados de classificação obtidos pelo método de análise dos coeficientes de correlação entre features. Neste caso, os resultados não são os mais satisfatórios, principalmente no problema multiclasse. Podemos então concluir que este método não é o mais adequado para a seleção de features. No entanto, é importante referir que o método de classificação que apresenta melhor performance é o Random Forest.
Nas tabelas 7.6 e 7.7 encontram-se os resultados obtidos para o método *Feature Importance using Tree Classifiers*. Existe, mais uma vez, uma grande diferença entre os resultados obtidos no problema binário e no problema multiclasse, pelo que não consideramos que este método de seleção seja o mais adequado. Resta realçar que, novamente, são os algoritmos de classificação SVM e Random Forest que mostram melhor performance.
Nas tabelas 7.8 e 7.9 são apresentados os resultados do método de seleção LASSO. Novamente, não nos parece o método mais adequado uma vez que, no problema multiclasse as performances dos classificadores ficam muito aquém das expectativas. Os algoritmos de classificação com melhor performance voltam a ser o SVM e Random Forest.
Nas tabelas 7.9 e 7.10 apresentamos os resultados da aplicação do método Kruskal Wallis. No contexto do problema binário, continuamos a ter ótimos resultados, para todos os algoritmos de classifcação. No contexto do problema multiclasse podemos verificar uma melhoria nos resultados de quase todos os classificadores, sendo que os melhores são obtidos por SVM, LDA e Random Forest.
A aplicação do método AUC score foi feita apenas no contexto do problema binário. Na tabela 7.11 podemos, mais uma vez, observar ótimos resultados.
Por fim, nas tabelas 7.12 e 7.13 são apresentados os resultados do método de seleção que analisa a correlação entre cada feature e os seus labels. Este método, apresenta resultado razoáveis para o problema multidimensional, relativamente aos restantes métodos apresentados e, assim como o método de Kruskall Wallis, tem em consideração as labels, o que poderá constituir um impedimento à sua utilização quando estamos perante um problema não supervisionado. Neste caso, os métodos de classificação que apresentaram melhores valores de performance foram SVM, Random Forest, LDA e kNN.
Iremos então continuar o nosso estudo tendo por base as features selecionadas pelo método Kruskal Wallis e pelo método que avalia a correlação entre cada feature e o seu label.

9 Discussão - Redução de Dimensão

São apenas apresentados resultados quando utilizado o método PCA uma vez que, como já foi referido, o método MDS exige um elevado poder computacional resultando por isso num processo bastante demorado.
Os resultados presentes na tabela 7.14 referem-se à redução da dimensão a 2 e a 3 dimensões, utilizando PCA, no problema binário. Podemos verificar que os resultados são bastante bons apesar de reduzirmos 561 features a 2 e a 3 componentos. Já na tabela 7.15 podemos observar que os resultados obtidos para o problema multiclasse não são satisfatórios, no entanto, é necessário ter em conta que devemos utilizar mais componente na redução, nesta fase utilizámos 2 e 3 por uma questão de representação dos resultados.
Nesta fase já foi feita uma análise dos métodos de seleção, tendo em conta os resultados de performance de teste obtidos pelos vários algoritmos de classificação. Resta agora, recuar novamente no trabalho e avaliar quais os melhores modelos no contexto do problema.
Resta referir que nesta secção iremos utilizar dois datasets diferente: um resulta da seleção das 100 features mais discriminativas pelo método Kruskal Wallis e posterior redução a 20 dimensões com o PCA; o outro apenas difere na medida em que utiliza para a seleção de features o método que avalia a correlação entre cada feature e os labels.

10 Classificação com validação cruzada

Nesta fase pretendemos avaliar a qualidade dos modelos produzidos por cada método de classificação. Para isso vamos utilizar o método de seleção de features Kruskal Wallis, fazendo a seleção das 100 primeiras features mais discriminativas, de seguida aplicamos o método PCA para reduzir este conjunto a 20 dimensões.
Cada modelo é avaliado pelo média dos scores obtidos em cada fold da validação cruzada, sendo que iremos utilizar 10-fold. Para a validação cruzada utilizámos 70% do dataset de treino para treinar o modelo e 30% para o validar.

Table 10.1: Scores obtidos em cada fold fazendo 10-fold cross validation, para o modelo binário, usando Kruskal Wallis como método de seleção.

fold	1	2	3	4	5	6	7	8	9	10	
SVM											
score	0.990	1	0.995	1	1	1	1	1	0.995	0.995	0.998
kNN											
score	1	1	1	0.995	1	1	1	0.995	1	1	0.999
LDA											
score	1	1	1	0.995	1	1	1	0.991	1	1	0.998
QDA											
score	1	1	1	1	1	1	1	1	1	0.995	0.999
Naive Bayes											
score	1	0.995	1	1	1	1	1	0.991	1	1	0.999
Random Forest											
score	1	1	1	1	1	1	1	0.995	1	1	0.999
DMC - Euclediana											
score	0.980	0.979	0.983	0.985	0.979	0.980	0.978	0.976	0.981	0.985	0.980
DMC - Mahalanobis											
score	0.977	0.983	0.978	0.979	0.984	0.979	0.976	0.983	0.983	0.980	0.980

Table 10.2: Scores obtidos em cada fold fazendo 10-fold cross validation, para o modelo multi-classe, usando Kruskal Wallis como método de seleção.

fold	1	2	3	4	5	6	7	8	9	10	
SVM											
score	0.852	0.874	0.869	0.860	0.829	0.864	0.859	0.894	0.857	0.889	0.864
kNN											
score	0.825	0.812	0.851	0.809	0.882	0.859	0.809	0.831	0.798	0.806	0.828
LDA											
score	0.822	0.811	0.829	0.812	0.822	0.794	0.813	0.776	0.835	0.872	0.820
QDA											
score	0.888	0.869	0.816	0.864	0.809	0.877	0.836	0.845	0.881	0.838	0.853
Naive Bayes											
score	0.815	0.811	0.793	0.769	0.796	0.851	0.8	0.836	0.881	0.781	0.813
Random Forest											
score	0.838	0.780	0.819	0.778	0.760	0.773	0.805	0.782	0.804	0.825	0.796
DMC - Euclediana											
score	0.535	0.522	0.514	0.542	0.519	0.523	0.499	0.509	0.523	0.521	0.520
DMC - Mahalanobis											
score	0.670	0.660	0.663	0.650	0.658	0.673	0.664	0.669	0.658	0.667	0.663

11 Discussão- Classificação com validação cruzada

Na secção em cima recorreu-se ao método de validação cruzada para avaliar os modelos de classificação, de modo a perceber quais os métodos de classificação mais indicados para o nosso problema.
Perante os resultados obtidos nas tabelas 10.1 e 10.2, podemos observar que no caso da classifação binária, praticamente todos os modelos de classificação demonstram ser adequados para a resolução do nosso problema, isto porque existe uma clara separação entre classes e, desse modo, qualquer método que seja aplicado irá ter uma ótima performance.

Na classificação multiclasse, apesar de no geral todos os modelos terem bons resultados, os modelos que aparentam ser mais adequados são o SVM e o QDA.
Relativamente aos classificadores que fazem também redução de dimensionalidade, isto é, LDA e QDA, podemos observar que os resultados são semelhantes, no entanto, em média o QDA apresenta melhores resultados de performance.
Fazendo uma análise geral verificamos que o SVM apresenta a melhor performance. Poderíamos ter feito um estudo mais aprofundado neste ponto com vista a encontrar os melhores valores para o parâmetro C que na melhor AUC. No entanto, por falta de tempo, e como nos focámos bastante na parte de seleção de features não tivemos oportunidade de o fazer.

Relativamente aos classificadores de minima distância verificamos que a sua performance fica um bocado aquém das expectativas. Estes apenas resultam em boa performance quando são aplicados no contexto do problema binário. Em média, e como nós esperávamos, o modelo que usa a distância de Mahalanobis apresenta uma melhor performance.
O kNN no contexto binário apresente ótimos resultados, uma vez que os dados poder ser facilmente separados em dois clusters. No problema multiclasse, a performance deste método de classificação diminui, no entanto, os resultados são satisfatórios.
Relativamente aos classificador Naives e Random Fores, estes também apresenta um valor de score médio para a validação cruzada aceitável, no entanto, o SVM superá-los.

12 Classificação com teste

12.1 SVM

Table 12.1: Matriz confusão para o teste do modelo obtido pelo SVM, para o problema binário.

	0	1
0	1558	2
1	0	1387

Accuracy: 0.998
Precisão: 0.996
Sensibilidade: [0.997 1.]

Especificidade: [1. 0.997]

Table 12.2: Matriz confusão para o teste do modelo obtido pelo SVM, para o problema multi-classe.

	1	2	3	4	5	6
1	452	7	37	0	0	0
2	67	332	72	0	0	0
3	106	46	268	0	0	0
4	0	5	0	355	133	0
5	0	1	0	72	460	0
6	0	0	0	2	0	535

Accuracy: 0.816
Precisão: 0.819
Sensibilidade: [0.911 0.705 0.638 0.723 0.865 0.996]
Especificidade: [0.929 0.977 0.957 0.970 0.945 1.]

12.2 kNN

Table 12.3: Matriz confusão para o teste do modelo obtido pelo kNN, para o problema binário.

	0	1
0	1555	5
1	0	1387

Accuracy: 0.998
Precisão: 0.996
Sensibilidade: [0.997 1.]
Especificidade: [1. 0.997]

Table 12.4: Matriz confusão para o teste do modelo obtido pelo kNN, para o problema multi-classe.

	1	2	3	4	5	6
1	436	20	40	0	0	0
2	65	360	46	0	0	0
3	0	5	0	392	4	0
4	0	5	0	346	141	0
5	0	1	0	157	374	0
6	0	0	0	10	1	536

Accuracy: 0.783
Precisão: 0.789
Sensibilidade: [0.879 0.764 0.521 0.798 0.703 0.980]
Especificidade: [0.916 0.966 0.966 0.932 0.961 1.000]

12.3 LDA

Table 12.5: Matriz confusão para o teste do modelo obtido pelo LDA, para o problema binário.

	0	1
0	1553	7
1	0	1387

Accuracy: 0.998
Precisão: 0.995
Sensibilidade: [0.996 1.]
Especificidade: [1. 0.996]

Table 12.6: Matriz confusão para o teste do modelo obtido pelo LDA, para o problema multi-classe.

	1	2	3	4	5	6
1	411	27	58	0	0	0
2	47	376	48	0	0	0
3	86	56	278	0	0	0
4	0	5	0	346	141	0
5	0	1	0	83	447	0
6	0	0	0	0	0	537

Accuracy: 0.813
Precisão: 0.813
Sensibilidade: [0.829 0.798 0.662 0.705 0.840 1.]
Especificidade: [0.946 0.964 0.958 0.966 0.942 1.]

12.4 QDA

Table 12.7: Matriz confusão para o teste do modelo obtido pelo QDA, para o problema binário.

	0	1
0	1556	4
1	0	1387

Accuracy: 0.999
Precisão: 0.998
Sensibilidade: [0.998 1.]
Especificidade: [1. 0.998]

Table 12.8: Matriz confusão para o teste do modelo obtido pelo QDA, para o problema multi-classe.

	1	2	3	4	5	6
1	447	6	43	0	0	0
2	67	334	70	0	0	0
3	87	59	274	0	0	0
4	0	5	0	346	142	1
5	0	1	0	75	457	0
6	0	0	0	0	0	533

Accuracy: 0.811
Precisão: 0.814
Sensibilidade: [0.902 0.709 0.652 0.705 0.859 0.993]
Especificidade: [0.937 0.973 0.955 0.968 0.941 0.999]

12.5 Naive Bayes

Table 12.9: Matriz confusão para o teste do modelo obtido pelo Naive Bayes, para o problema binário.

	0	1
0	1554	6
1	1	1386

Accuracy: 0.998
Precisão: 0.996
Sensibilidade: [0.996 0.999]
Especificidade: [0.999 0.996]

Table 12.10: Matriz confusão para o teste do modelo obtido pelo Naive Bayes, para o problema multiclasse.

	1	2	3	4	5	6
1	434	18	44	0	0	0
2	105	317	49	0	0	0
3	99	39	282	0	0	0
4	0	4	0	379	106	2
5	1	0	0	119	412	0
6	0	0	0	7	1	529

Accuracy: 0.798
Precisão: 0.805
Sensibilidade: [0.875 0.673 0.671 0.772 0.774 0.985]
Especificidade: [0.916 0.975 0.963 0.949 0.956 0.999]

12.6 Random Forest

Table 12.11: Matriz confusão para o teste do modelo obtido pelo Random Forest, para o problema binário.

	0	1
0	1555	5
1	3	1384

Accuracy: 0.997
Precisão: 0.996
Sensibilidade: [0.997 0.998]
Especificidade: [0.998 0.997]

Table 12.12: Matriz confusão para o teste do modelo obtido pelo Random Forest, para o problema multiclasse.

	1	2	3	4	5	6
1	284	27	41	0	0	0
2	84	331	54	1	1	0
3	133	52	235	0	0	0
4	0	4	0	351	135	1
5	0	1	0	139	392	0
6	0	0	0	17	3	517

Accuracy: 0.765

Precisão: 0.771
Sensibilidade: [0.863 0.703 0.559 0.715 0.737 0.963]
Especificidade: [0.911 0.966 0.962 0.936 0.942 0.999]

12.7 DMC - EUCLEDIANA

Table 12.13: Matriz confusão para o teste do modelo obtido pelo DMC - Euclediana, para o problema binário.

	0	1
0	1520	40
1	0	1387

Accuracy: 0.986
Precisão: 0.972
Sensibilidade: [0.974 1.]
Especificidade: [1. 0.974]

Table 12.14: Matriz confusão para o teste do modelo obtido pelo DMC - Euclediana, para o problema multiclasse.

	1	2	3	4	5	6
1	11	114	371	0	0	0
2	0	402	69	0	0	0
3	0	73	347	0	0	0
4	0	38	0	166	15	272
5	1	54	1	169	151	156
6	0	6	0	0	0	531

Accuracy: 0.546
Precisão: 0.658
Sensibilidade: [0.022 0.853 0.826 0.338 0.284 0.989]
Especificidade: [0.999 0.885 0.825 0.931 0.994 0.822]

Table 12.15: Matriz confusão para o teste do modelo obtido pelo DMC - Mahalanobis, para o problema binário.

	0	1
0	1520	40
1	0	1387

Accuracy: 0.986
Precisão: 0.972
Sensibilidade: [0.974 1.]
Especificidade: [1. 0.974]

Table 12.16: Matriz confusão para o teste do modelo obtido pelo DMC - Mahalanobis, para o problema multiclasse.

	1	2	3	4	5	6
1	370	18	50	39	19	0
2	41	292	57	41	40	0
3	59	31	269	33	28	0
4	36	59	7	306	83	0
5	37	42	6	83	364	0
6	83	23	8	58	18	347

Accuracy: 0.661
Precisão: 0.689
Sensibilidade: [0.746 0.620 0.640 0.623 0.684 0.646]
Especificidade: [0.896 0.930 0.949 0.897 0.922 1.]

13 DISCUSSÃO - CLASSIFICAÇÃO COM TESTE

Ao observar as métricas de cada classificador, obtidas pela comparação dos dados obtidos com o data set de teste, concluímos que o melhor classificador é o SVM, quer para o problema binário, quer para o multiclasse. Os classificadores LDA e QDA também obtiveram resultados muito bons, concluindo que estes também são bons classificadores para aplicar ao nosso problema.

14 INTERFACE GRÁFICA

O nosso trabalho foi complementado com a criação de uma interface gráfica que permite a exploração dos vários métodos de pré-processamento, seleção e/ou redução de features e

classifcação de novos dados do mesmo tipo, introduzidos pelo o utilizador.
Esta interface mostra ao utilizador as métricas da classificação obtida pelos parâmetros escolhidos, tais como, Accuracy, Matriz de Confusão e AUC Score e a curva ROC (para o problema binário).

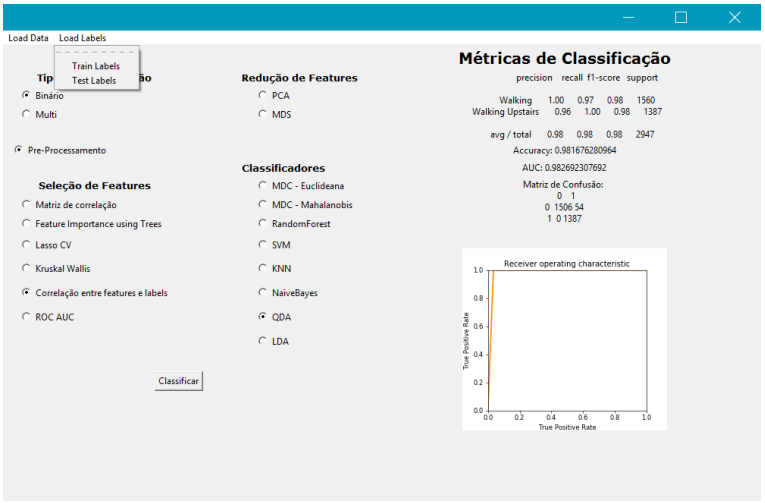

Figure 14.1: Exemplo da apresentação dos resultados de uma classificação binária.

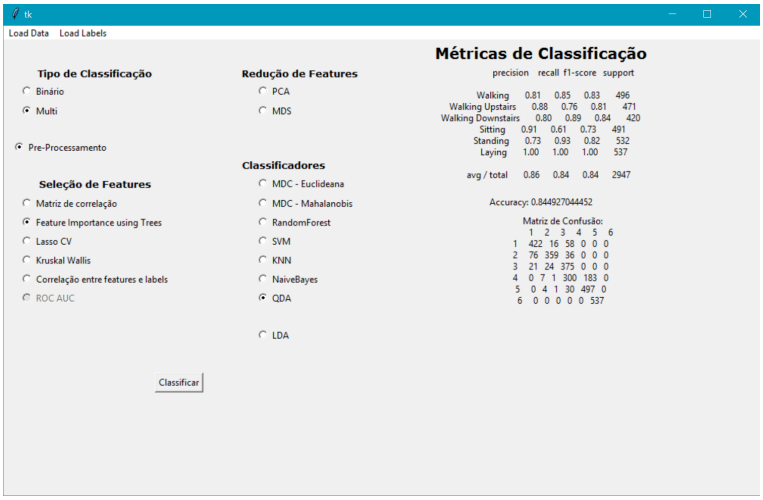

Figure 14.2: Exemplo da apresentação dos resultados de uma classificação multi-classe.

15 Notas Finais

Em primeiro lugar, achamos que é importante referir que propusemos a nós mesmas o desafio de aprofundar os nossos conhecimentos de Python, já que antes deste projeto não tínhamos

muita experiência com essa mesma linguagem.
Relativamente ao problema em si, focámo-nos bastante em procurar métodos de seleção de features que se adequassem aos nossos dados e essa tarefa levou-nos muito tempo a desenvolver. Achámos que era importante focarmo-nos neste tópico uma vez que tínhamos um dataset com uma elevada dimensionalidade o que, caso este não fosse processado, exigiria um elevado custo computacional aquando da classificação.
Relativamente à classificação, poderíamos ter estudado mais aprofundadamente os classificadores que utilizámos, isto é, poderíamos ter variado os seus parâmetros, no entanto, não nos foi possível.
Apesar de tudo acreditamos que conseguimos obter bons resultados e aprofundar os nossos conhecimentos no campo do machine learning.

REFERENCES

[1] http://napitupulu-jon.appspot.com/posts/feature-selection-ud120.html

[2] http://machinelearningmastery.com/feature-selection-machine-learning-python/

[3] http://wtlab.iis.u-tokyo.ac.jp/ wataru/lecture/rsgis/rsnote/cp11/cp11-6.htm

[4] http://blog.datadive.net/selecting-good-features-part-iv-stability-selection-rfe-and-everything-side-by-side/

[5] http://www.few.vu.nl/nl/Images/werkstuk-fonti_tcm243-836234.pdf

[6] http://www.sciencedirect.com/science/article/pii/S1570023212002929