University of Minho
School of Engineering

APRENDIZAGEM E EXTRAÇÃO DE CONHECIMENTO

Caroline Rodrigues
PG39285

Hugo da Gião
PG41073

Rafaela de Pinho
PG41095

15 de Dezembro de 2019

Resumo

O principal objetivo deste trabalho prático é aprender os vários procedimentos utilizados em Projetos de Extração de Conhecimento.

Conteúdo

Capítulo 1

Introdução

Inteligência artificial (IA) é a inteligência demonstrada pelas máquinas, é frequentemente usada para descrever máquinas que imitam funções "cognitivas" ligadas à mente humana, como "aprendizagem". A aprendizagem tem como objetivo de adquirir ou incorporar o conhecimento resultante de experiências passadas, através da construção de um novo caso.

Para este trabalho foi fornecido um dataset que apresenta informações referentes a funcionários de uma empresa de transporte, onde são analisados os motivos associados à ausência ao trabalho. Este possui 740 caso de estudo com 20 features distintas adquiridas de 36 funcionários diferentes.

Neste relatório vamos mostrar as diferentes fazes do processo de aprendizagem aplicadas e este problema especifico,são elas o tratamento dos dados,a "feature selection",a "feature extration",os modelos usados e o processo de treino e escolha de parâmetros dos mesmos.

Por fim, mostraremos quais os melhores modelos e a sua hiper parametrização

Capítulo 2

Tratamento dos dados

2.1 Visualização e exploração do dataset

A primeira fase deste trabalho consiste numa análise e familiarização com o dataset de estudo. Isto inclui analisar o tipo da dados presente no mesmo e as modificações necessárias a realizar nestes,o seu tamanho,relações entre os dados presentes no mesmo e outras métricas e características que nos permitam fazer melhor uso do mesmo.

	ID	Reason for absence	Month of absence	Day of the week	Seasons	Transportation expense	Distance from Residence to Work	Service time	Age	Work load Average/day
0	1	26	7	3	1	289	36	13	33	239,554
1	2	0	7	3	1	118	13	18	50	239,554

Figura 2.1: primeiros dois elementos do dataset original,primeiras 10 colunas

	Hit target	Disciplinary failure	Education	Son	Social drinker	Social smoker	Pet	Weight	Height	Body mass index	Absent
0	97	0	1	2	1	0	1	90	172	30	1
1	97	1	1	1	1	0	0	98	178	31	0

Figura 2.2: primeiros dois elementos do dataset original ultimas 11 colunas

2.1.1 Tipos de dados presentes no dataset

O primeiro passo na visualização do dataset consistiu em verificar quais os tipos dos dados presentes no dataset.Obtemos esta informação olhando ao types do nosso dataset lido pelo pandas.

ID	int64
Reason for absence	int64
Month of absence	int64
Day of the week	int64
Seasons	int64
Transportation expense	int64
Distance from Residence to Work	int64
Service time	int64
Age	int64
Work load Average/day	object
Hit target	int64
Disciplinary failure	int64
Education	int64
Son	int64
Social drinker	int64
Social smoker	int64
Pet	int64
Weight	int64
Height	int64
Body mass index	int64
Absent	int64

Figura 2.3: tipos dos dados presentes no dataset

A partir dos resultados obtidos notamos que será necessário converter os dados na coluna 'Wokr load Average/day' para inteiros.

2.1.2 Tamanho do dataset

Obtemos que o tamanho inicial do dataset de treino é de 500.

2.1.3 Informação sobre os diferentes campos do dataset original

Utilizamos a função decribe da biblioteca pandas para obter informação relativa ao dataset original.

	ID	Reason for absence	Month of absence	Day of the week	Seasons	Transportation expense	Distance from Residence to Work	Service time	Age
count	500.000000	500.000000	500.000000	500.00000	500.000000	500.000000	500.000000	500.000000	500.000000
mean	250.500000	19.288000	6.614000	3.88000	2.450000	223.640000	29.978000	12.650000	36.660000
std	144.481833	8.543245	3.343555	1.43587	1.165425	67.323155	15.068498	4.036345	6.137731
min	1.000000	0.000000	1.000000	2.00000	1.000000	118.000000	5.000000	3.000000	27.000000
25%	125.750000	13.000000	3.750000	3.00000	1.000000	179.000000	16.000000	10.000000	33.000000
50%	250.500000	23.000000	7.000000	4.00000	2.000000	225.000000	26.000000	13.000000	37.000000
75%	375.250000	26.000000	9.000000	5.00000	4.000000	260.000000	50.000000	16.000000	40.000000
max	500.000000	28.000000	12.000000	6.00000	4.000000	388.000000	52.000000	29.000000	58.000000

Figura 2.4: descrição das 10 primeiras colunas do dataset

	Hit target	Disciplinary failure	Education	Son	Social drinker	Social smoker	Pet	Weight	Height	Body mass index	Absent
count	500.000000	500.000000	500.000000	500.000000	500.000000	500.000000	500.000000	500.000000	500.000000	500.000000	500.000000
mean	94.168000	0.064000	1.204000	1.086000	0.620000	0.076000	0.628000	79.698000	172.098000	26.870000	0.790000
std	3.912338	0.244998	0.561261	1.178721	0.485873	0.265264	1.071406	12.605101	6.234913	4.151092	0.407716
min	81.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	56.000000	163.000000	19.000000	0.000000
25%	92.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	69.000000	169.000000	24.000000	1.000000
50%	95.000000	0.000000	1.000000	1.000000	1.000000	0.000000	0.000000	83.000000	170.000000	25.000000	1.000000
75%	97.000000	0.000000	1.000000	2.000000	1.000000	0.000000	1.000000	89.000000	172.000000	31.000000	1.000000
max	100.000000	1.000000	3.000000	4.000000	1.000000	1.000000	5.000000	108.000000	196.000000	38.000000	1.000000

Figura 2.5: descrição das ultimas 11 colunas do dataset

2.1.4 Análise e exploração dos dados

Após obtermos a informação anterior sobre o dataset decidimos criar gráficos e explorar diferentes métricas que nos permitam ganhar familiaridade com o dataset.

Correlação com o campo Absent

Criamos gráficos que nos apresentam a correlação entre os diferentes e o campo que pretendemos prever.

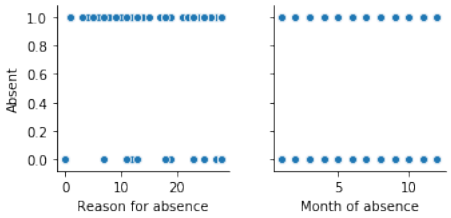


Figura 2.6: Relações das diferentes classes com a class Absent

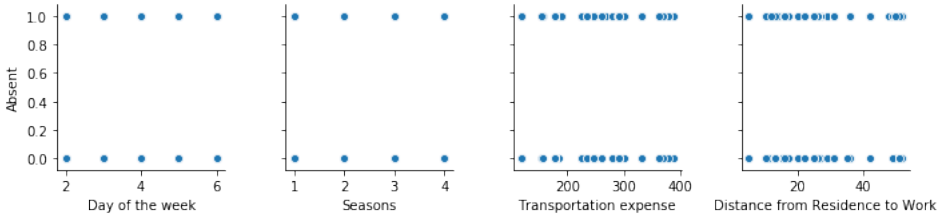


Figura 2.7: Relações das diferentes classes com a class Absent

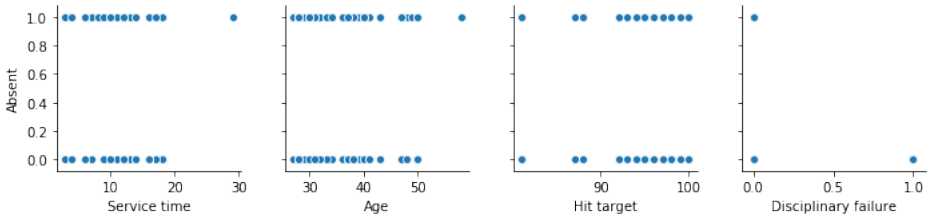


Figura 2.8: Relações das diferentes classes com a class Absent

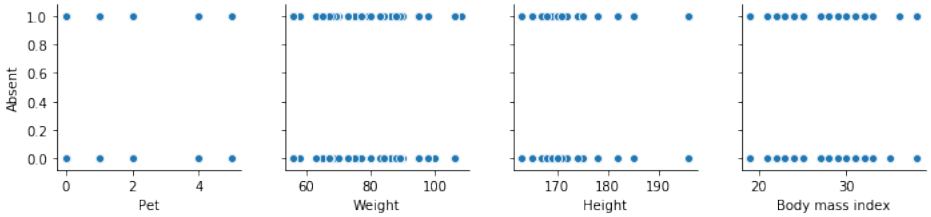


Figura 2.9: Relações das diferentes classes com a class Absent

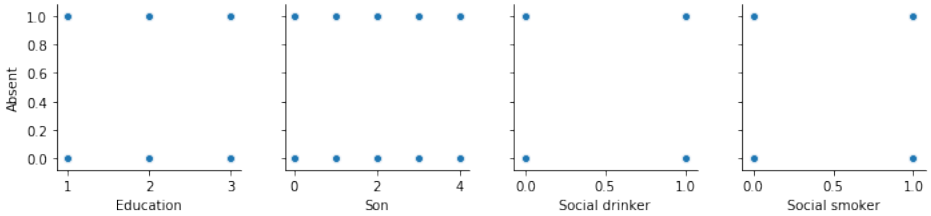


Figura 2.10: Relações das diferentes classes com a class Absent

Relação entre os campos

Criamos alguns gráficos que nos permitem averiguar relações entre diferentes campos.

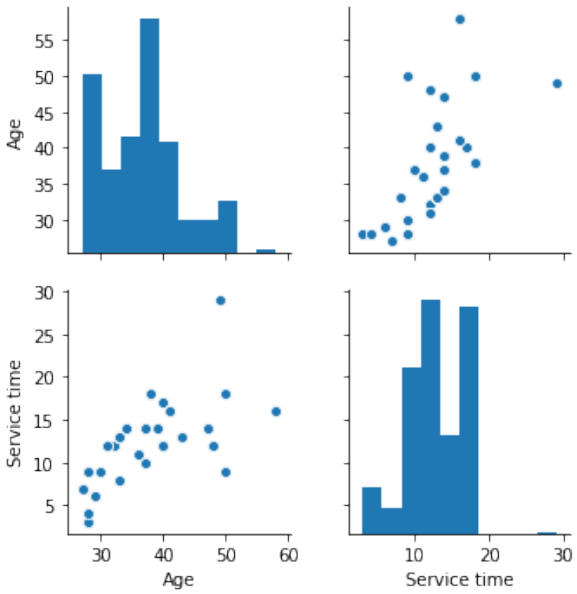


Figura 2.11: Relações entre os campos idade e tempo de serviço

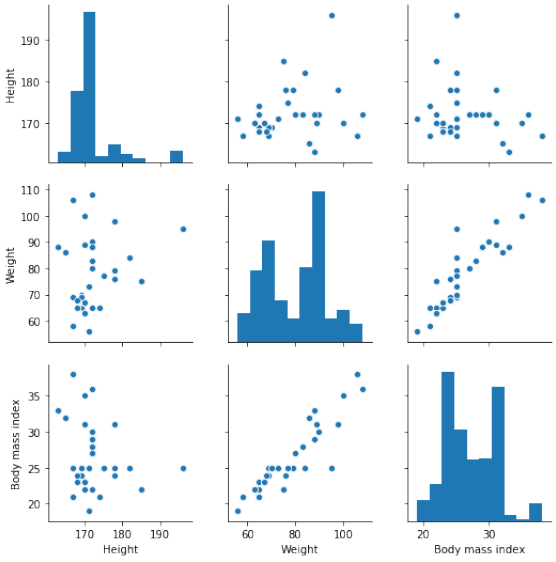


Figura 2.12: Relações entre os campos altura peso e índice de massa corporal

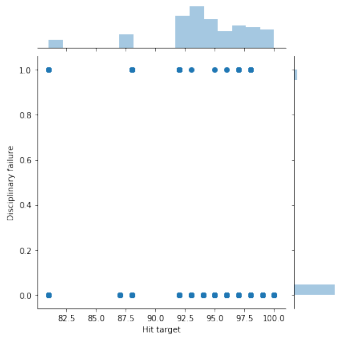


Figura 2.13: Relação entre Hit Target e disciplinary failure

Distribuição de dados

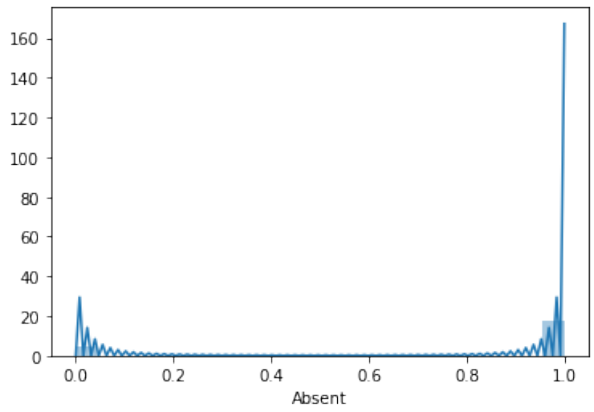


Figura 2.14: Distribuição dos dados da coluna Absent

2.2 Normalização — Binarização — Estandardização

2.2.1 Normalização

Normalização consiste numa técnica de pré-processamento de dados que permite obter a partir de um dataset com valores numéricos um dataset em que todos os valores estão entre 0 e 1.

Existem diferentes técnicas de normalização mas para este trabalho optamos por utilizar o MinMaxScaller da biblioteca sklearn.

	Reason for absence	Month of absence	Day of the week	Seasons	Transportation expense	Distance from Residence to Work	Service time	Age	Work load Average/day	Hit target
0	0.928571	0.545455	0.25	0.0	0.633333	0.659574	0.384615	0.193548	0.194471	0.842105
1	0.000000	0.545455	0.25	0.0	0.000000	0.170213	0.576923	0.741935	0.194471	0.842105

Figura 2.15: dataset normalizado 10 primeiras colunas

	Disciplinary failure	Education	Son	Social drinker	Social smoker	Pet	Weight	Height	Body mass index
0	0.0	0.0	0.50	1.0	0.0	0.2	0.653846	0.272727	0.578947
1	1.0	0.0	0.25	1.0	0.0	0.0	0.807692	0.454545	0.631579

Figura 2.16: dataset normalizado ultimas 11 colunas

2.2.2 Binarização

Binarização consiste no processo de transformar os elementos de um dataset num conjunto de 0 e 1 de modo a tornar o processo de treino mais eficiente.

Para tal utilizamos o Binarizer da biblioteca sklearn.

	Reason for absence	Month of absence	Day of the week	Seasons	Transportation expense	Distance from Residence to Work	Service time	Age	Work load Average/day	Hit target
0	1	1	1	1	1	1	1	1	1	1
1	0	1	1	1	1	1	1	1	1	1

Figura 2.17: dataset binarizado 10 primeiras colunas

	Disciplinary failure	Education	Son	Social drinker	Social smoker	Pet	Weight	Height	Body mass index
0	0	1	1	1	0	1	1	1	1
1	1	1	1	1	0	0	1	1	1

Figura 2.18: dataset binarizado ultimas 11 colunas

2.2.3 Estandardização

O processo de Estandardização consiste em fazer com que os valores de um dataset tenham uma media de 0 e um desvio padrão de 1.

Para implementar esta funcionalidade utilizamos o StandartScaler da biblioteca Sklearn.

	Reason for absence	Month of absence	Day of the week	Seasons	Transportation expense	Distance from Residence to Work	Service time	Age	Work load Average/day	Hit target
0	0.786437	0.115562	-0.613483	-1.245427	0.971812	0.400042	0.086799	-0.596909	-0.858446	0.724589
1	-2.259951	0.115562	-0.613483	-1.245427	-1.570720	-1.127850	1.326784	2.175618	-0.858446	0.724589

Figura 2.19: dataset estandardizado 10 primeiras colunas

	Reason for absence	Month of absence	Day of the week	Seasons	Transportation expense	Distance from Residence to Work	Service time	Age	Work load Average/day	Hit target
0	0.786437	0.115562	-0.613483	-1.245427	0.971812	0.400042	0.086799	-0.596909	-0.858446	0.724589
1	-2.259951	0.115562	-0.613483	-1.245427	-1.570720	-1.127850	1.326784	2.175618	-0.858446	0.724589

Figura 2.20: dataset estandardizado ultimas 11 colunas

2.3 Balancear o dataset

Os diferentes modelos de machine learning são criados com intuito de maximizar a accuracy. Isto apresenta-se como desfavorável em vários casos em que datasets não estão balanceados podendo ter valores de accuracy favoráveis mas não conseguindo identificar nenhum dos casos menos frequentes,por isso surge a necessidade de balancear o dataset para que o treino seja feito de modo a distinguir as diferentes classes.

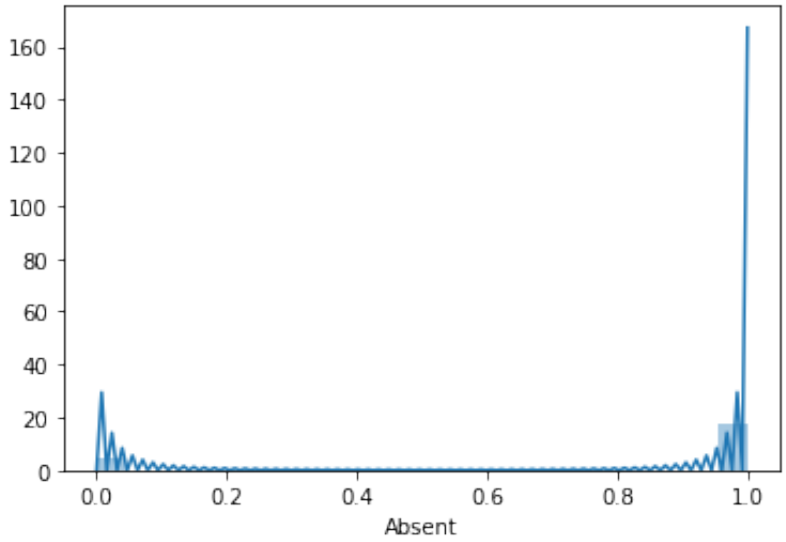


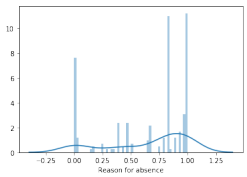
Figura 2.21: Distribuição do campo ausente no dataset de treino

A partir desta figura podemos denotar que os casos presentes no dataset estão bastante desbalanceados o que poderá levar a problemas futuros.

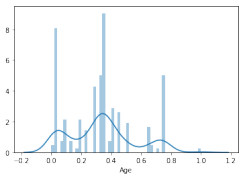
Para implementar esta funcionalidade no nosso caso primeiro obtemos o numero de elementos ausentes e não ausentes separamos o dataset separamos estes elementos em dois datasets. Pegamos no maior dos datasets fazemos shuffle dos seus elementos e removemos os n últimos de modo aos dois datasets terem o mesmo tamanho.Por fim juntamos os dois datasets e fazemos shuffle dos seus elementos. Experimentamos realizar esta operação com diferentes percentagens de cada ausentes e não ausentes no dataset.

2.4 Remoção de outliers

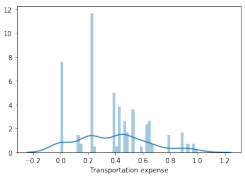
Neste processo começamos por realizar uma visualização da distribuição de dados das diferentes colunas do dataset.



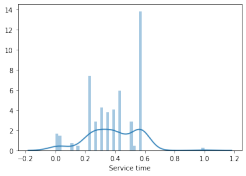
(a) outliers reason of absence



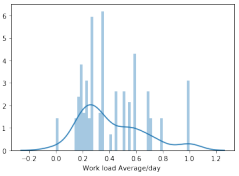
(b) outliers age



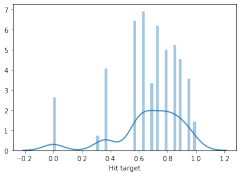
(c) outliers transportation expense



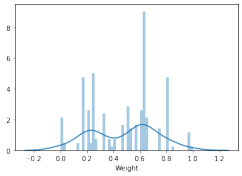
(d) outliers service time



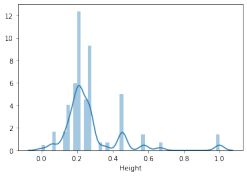
(e) outliers Work load average day



(f) outliers Hit Target



(g) outliers weight



(h) outliers height

Figura 2.22: Outliers para diferentes colunas do dataset

A partir dos gráficos acima e da análise do dataset optamos por remover os outlieer das seguintes colunas 'Reason for absence',"Age","Transportation expense", "Service time","Work load Average/day ","Hit target", "Weight","Height"e "Body mass index".

Após realizar esta visualização definimos uma função que calcula os 0.25 percentis de todas as colunas e remove as linhas do dataset com elementos que façam parte deste grupo nas colunas selecionadas anteriormente.

Capítulo 3

Feature Selection + Extraction

3.1 Feature Selection

3.2 KBest

Um dos métodos utilizados no nosso trabalho para realizar a feature selection é o KBest.Este algoritmo utiliza uma função de teste estatístico para determinar quais são as melhores features do dataset no nosso caso optamos por utilizar chi^2.

	Reason for absence	Seasons	Age	Hit target	Disciplinary failure	Son	Weight	Body mass index	Absent
0	0.928571	0.0	0.193548	0.842105	0.0	0.50	0.653846	0.578947	1
1	0.000000	0.0	0.741935	0.842105	1.0	0.25	0.807692	0.631579	0

Figura 3.1: Dataset após aplicar KBest

3.3 RFE

O método de Recursive Feature elimination dado um numero de features a obter e um dataset,este treina um estimador e elimina a feature com menor importância recursivamente ate ter o numero de features do dataset ser igual ao pretendido.

	Service time	Age	Disciplinary failure	Education	Pet	Weight	Height	Body mass index	Absent
0	0.384615	0.193548	0.0	0.0	0.2	0.653846	0.272727	0.578947	1
1	0.576923	0.741935	1.0	0.0	0.0	0.807692	0.454545	0.631579	0

Figura 3.2: Dataset após aplicar RFE

3.4 Feature extraction

O processo de feature extraction consiste em a partir de features existentes criar novas mais adaptadas ao processo de treino.

No nosso caso criamos uma função que percorre a lista de features e calcula a sua correlação,ao encontrar um grupo features com um grau de correlação superior a 0.5 remove-as do dataset original e cria uma nova feature que consiste numa amalgação das mesmas e remove as features originais.

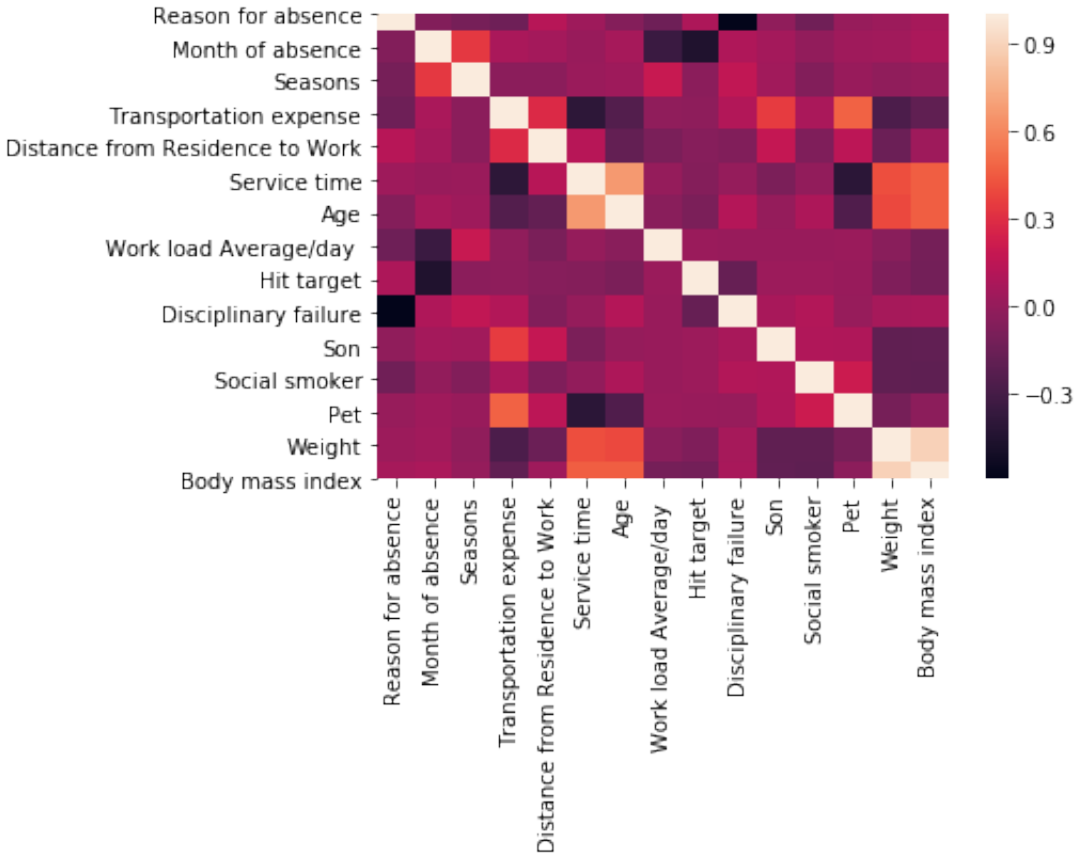


Figura 3.3: Correlações entre as diferentes colunas

	Reason for absence	Month of absence	Day of the week	Seasons	Transportation expense	Distance from Residence to Work	Service time + Age	Work load Average/day	Hit target	Disciplinary failure
0	0.928571	0.545455	0.25	0.0	0.633333	0.659574	0.289082	0.194471	0.842105	0.0
1	0.000000	0.545455	0.25	0.0	0.000000	0.170213	0.659429	0.194471	0.842105	1.0

Figura 3.4: 10 primeiras colunas do dataset após aplicar FE

	Education	Son	Social drinker	Social smoker	Pet	Weight + Body mass index	Height	Absent
0	0.0	0.50	1.0	0.0	0.2	0.616397	0.272727	1
1	0.0	0.25	1.0	0.0	0.0	0.719636	0.454545	0

Figura 3.5: 8 ultimas colunas do dataset após aplicar FE

Capítulo 4

Modelos de Machine Learning

4.1 LinearRegression

A regressão linear é uma técnica de análise estatística de dados. Determina se há uma relação linear entre uma variável dependente e uma ou mais variáveis independentes.

Utilizamos o modelo da biblioteca SKLEARN com os parâmetros fit_intercept=False, normalize = True, copy_X = False.

Pré-processamento realizado	**Accuracy**
Nenhum	80.4
Normalização	80.8
Estandardização	18
Binarização	80.8
Normalização + balanceamento 50/50	69.1
Normalização + balanceamento 50/50 + remoção de outliers	69.1
Normalização + Kbest	79.5
Normalização + RFE	80.8
Normalização + Feature Extraction	80

A partir dos teste realizados podemos verificar que a regressão linear temo bons resultados a nível de accuracy excepto para casos onde ocorre o balanceamento dos dados e remoção dos outliers, isto poderá dever-se a falta de dados.O processo de estandardização também leva a resultados fracos mas isto poderá ser corrigido obtendo o valor oposto do previsto pelo modelo.

4.2 LogisticRegression

A regressão logística é usada onde uma saída discreta é esperada, como a ocorrência de algum evento. Normalmente, a regressão logística usa uma função para comprimir os valores para um determinado intervalo.

Utilizamos o modelo da biblioteca SKLEARN com os parâmetros solver = 'saga'.

Pré-processamento realizado	**Accuracy**
Nenhum	80.8
Normalização	80.8
Estandardização	79.5
Binarização	80.4
Normalização + balanceamento 50/50	71.6
Normalização + balanceamento 50/50 + remoção de outliers	71.6
Normalização + Kbest	80.8
Normalização + RFE	80.8
Normalização + Feature Extraction	80.8

Os resultados da regressão logística são geralmente elevados a exceção do balanceamento e remoção de outliers.

4.3 Dummy Classifier

O Dummy Classifier usa apenas regras simples para classificar os dados e não gera nenhuma perceção sobre estes. Este modelo é independente dos dados de treino e usa umas da estratégias para prever o rótulo da classe.

Algumas da estratégias usadas por este modelo são:

- Mais frequente
- Estratificação
- Uniforme
- Constante

Utilizamos o modelo da biblioteca SKLEARN com os parâmetros strategy= 'most_frequent'.

Pré-processamento realizado	**Accuracy**
Nenhum	81.67
Normalização	81.67
Estandardização	81.67
Binarização	81.67
Normalização + balanceamento 50/50	81.67
Normalização + balanceamento 50/50 + remoção de outliers	81.67
Normalização + Kbest	81.67
Normalização + RFE	81.67
Normalização + Feature Extraction	81.67

Em geral este modelo da os melhores valores,isto deve-se ao facto de em todos os casos este modelo devolver o caso mais frequente neste caso sempre 1 excepto no caso do balanceamento que são iguais.

4.4 KMeans

k-means é um método de Clustering que tem como objetivo particionar n observações dentre k grupos onde cada observação pertence ao grupo mais próximo da média. Isso resulta numa divisão do espaço de dados.

Utilizamos a biblioteca de SKLEARN com os parâmetros n_clusters=2.

Pré-processamento realizado	**Accuracy**
Nenhum	23.75
Normalização	50
Estandardização	47.5
Binarização	44.5
Normalização + balanceamento 50/50	50
Normalização + balanceamento 50/50 + remoção de outliers	48.75
Normalização + KBest	47.5
Normalização + RFE	54.1
Normalização + Feature Extraction	50

Os resutlados deste modelo no dataset são bastante fracos.

4.5 KNN

Este é um algoritmo simples que prevê pontos de dados desconhecidos com os seus vizinhos mais próximos. O valor de k é um indicador crítico aqui quanto à precisão da predição. Ele determina o mais próximo calculando a distância, usando funções básicas como Euclidean.

Utilizamos a biblioteca de SKLEARN com os parâmetros n_neighbors=2.

Pré-processamento realizado	**Accuracy**
Nenhum	45.8
Normalização	68.3
Estandardização	64.5
Binarização	70.4
Normalização + balanceamento 50/50	46
Normalização + balanceamento 50/50 + remoção de outliers	45.8
Normalização + KBest	64.1
Normalização + RFE	65.8
Normalização + Feature Extraction	68.75

Os resultados deste modelo no dataset são bastante fracos.

4.6 SVM

O SVM realiza a separação de um conjunto de objetos com diferentes classes, ou seja, utiliza o conceito de planos de decisão. O SVM propõe a classificação de novos objetos (teste) com base em dados disponíveis (treino).

Utilizamos a biblioteca de SKLEARN com os parâmetros gamma = 'auto'.

Pré-processamento realizado	**Accuracy**
Nenhum	81.6
Normalização	80.8
Estandardização	80.8
Binarização	80
Normalização + balanceamento 50/50	80.8
Normalização + balanceamento 50/50 + remoção de outliers	80.8
Normalização + KBest	80.8
Normalização + RFE	80.8
Normalização + Feature Extraction	80.8

No geral este modelo apresenta bons resultados especialmente sem processamento.Neste caso as mudanças da estrutura dos dados resultante do processamento dos dados traduz-se a piores resultados.

4.7 NaiveBayes

Naive Bayes pertence à família dos classificadores probabilístico, baseados na aplicação do teorema de Bayes. Não existe um algoritmo único para o treino desses classificadores, mas um conjunto de algoritmos em que cada para de recursos classificados é independente do outro.

Utilizamos a biblioteca SKLEARN com os parâmetros por defeito.

Pré-processamento realizado	Accuracy
Nenhum	66.6
Normalização	80.8
Estandardização	-
Binarização	80.8
Normalização + balanceamento 50/50	80.4
Normalização + balanceamento 50/50 + remoção de outliers	80
Normalização + KBest	80.8
Normalização + RFE	80.8
Normalização + Feature Extraction	80.8

Temo bons resultados especialmente com o pré-processamento dos dados.

4.8 Perceptron

O Perceptron é um tipo de rede neural artificial. É um classificador binário onde função pode decidir se uma entrada pertence ou não a alguma classe específica.

Utilizamos a biblioteca de SKLEARN com os parâmetros tol=1e-3, random_state=0.

Pré-processamento realizado	Accuracy
Nenhum	81.67
Normalização	76.25
Estandardização	77.92
Binarização	80.42
Normalização + balanceamento 50/50	24.58
Normalização + balanceamento 50/50 + remoção de outliers	27.92
Normalização + Kbest	26.25
Normalização + RFE	54.17
Normalização + Feature Extraction	76.67

Tem bons resultados sem processamento, mas os resultados pioram bastante quando se faz pré-processamento dos dados especialmente balanceamento remoção de outliers e features.

4.9 Multilayer percetron

A Multilayer percetron (MLP) é uma rede neural semelhante à perceptron, composta por, pelo menos, 3 camadas de neurónios, uma de entrada, outra oculta e uma de saída ligadas entre si por sinapses com pesos.

Para o seu , usa um método de supervised learning chamado de backpropagation .

Utilizamos a biblioteca de SKLEARN com os parâmetros com os parâmetros solver='lbfgs', alpha=1e-5, hidden_layer_sizes=(5, 2), random_state=1.

Pré-processamento realizado	Accuracy
Nenhum	18.3
Normalização	79.1
Estandardização	75
Binarização	80.4
Normalização + balanceamento 50/50	65.4
Normalização + balanceamento 50/50 + remoção de outliers	67.08
Normalização + KBest	80.8
Normalização + RFE	80
Normalização + Feature Extraction	79.5

Melhores resultados do que o percetron com menor tamanho de dataset piores sem pré-tratamento de dados.

4.10 Decision Trees

O propósito da árvore de decisão é fazer diversas divisões dos dados em subconjuntos, de tal forma que os subconjuntos vão ficando cada vez mais puros.

Utilizamos a biblioteca de SKLEARN com os parâmetros random_state=0.

Pré-processamento realizado	**Accuracy**
Nenhum	74.5
Normalização	74.5
Estandardização	75.4
Binarização	80.4
Normalização + balanceamento 50/50	62.08
Normalização + balanceamento 50/50 + remoção de outliers	58
Normalização + KBest	69.5
Normalização + RFE	80.8
Normalização + Feature Extraction	63.5

Resultados pobres geralmente,binarização do dataset e remoção de features dão bons resultados.

4.11 Random Forest Classifier

O classificador Random Forest é um conjunto de árvores de decisão. Cada árvore do conjunto obtém uma classificação, ou seja um voto. Através desses escolhemos a classificação mais votada.

Utilizamos a biblioteca de SKLEARN e os parâmetros n_estimators = 200.

Pré-processamento realizado	**Accuracy**
Nenhum	78.33
Normalização	77.5
Estandardização	52.92
Binarização	80.42
Normalização + balanceamento 50/50	57.5
Normalização + balanceamento 50/50 + remoção de outliers	24.58
Normalização + Kbest	80.83
Normalização + RFE	80.83
Normalização + Feature Extraction	80.83

Mais uma vez geralmente pobres resultados mas bons resultados com remoção de features e binarização.

4.12 Passive Agressive Classifier

Este algoritmo é um algoritmo utilizado para classificar streams de data.Para cada dado recebido este algoritmo mantém o modelo treinado se a predição for correta ou modifica-o se for incorrecta, não guardado este dado por isso torna-se útil para processar grandes quantidades de dados que não possam ser guardados em memoria.

Utilizamos os parâmetros max_iter=1000, random_state=0, tol=1e-3.

Pré-processamento realizado	Accuracy
Nenhum	81.67
Normalização	21.67
Estandardização	70.42
Binarização	80.42
Normalização + balanceamento 50/50	20.42
Normalização + balanceamento 50/50 + remoção de outliers	38.75
Normalização + Kbest	25.0
Normalização + RFE	25.83
Normalização + Feature Extraction	22.08

Pobres resultados com pré-tratamento de dados.

4.13 Nearest Centroid

É um classificador simples, mas poderoso. Este modelo atribui às observações o rótulo da classe das amostras de treino, onde média está mais próxima da observação.

Utilizamos os parâmetros por defeito.

Pré-processamento realizado	Accuracy
Nenhum	68.75
Normalização	61.25
Estandardização	65.83
Binarização	80.42
Normalização + balanceamento 50/50	70.42
Normalização + balanceamento 50/50 + remoção de outliers	69.17
Normalização + Kbest	68.33
Normalização + RFE	25.83
Normalização + Feature Extraction	71.67

Maus resultados excepto com binarização.

4.14 Radius Neighbors classifier

Este modelo é semelhante ao modelo K-Neighbors, simplesmente difere em dois pontos. Um dos pontos é especificar o raio para saber se a observação é um vizinho, a outro é fornecer o outlier que indica qual classe não tem observações dentro do raio.

Utilizamos os parâmetros radius=1.9.

Pré-processamento realizado	Accuracy
Nenhum	-
Normalização	81.67
Estandardização	-
Binarização	-
Normalização + balanceamento 50/50	80.83
Normalização + balanceamento 50/50 + remoção de outliers	81.25
Normalização + Kbest	81.67
Normalização + RFE	81.67
Normalização + FE	81.67

Apesar de existirem casos onde o pré-processamento dos dados levar a que não existam vizinhos para determinados elementos do dataset quando e possível treinar o modelo este da face a situação enfrentada resultados bastante satisfatórios.

Capítulo 5

Ensemble Learning

A aprendizagem por conjunto combina as decisões de vários modelos para melhorar o desempenho geral da classificação. As principais causas de erro nos modelos de aprendizagem são devidas a ruídos e variância. Os ensemble methods ajudam a minimizar esses fatores. Projetado para melhorar a estabilidade e a precisão dos algoritmos de aprendizado de máquina.

5.1 Bagging

Bagging é uma técnica usada para reduzir a variância das previsões. A diversidade é obtida com o uso de diferentes subconjuntos de dados aleatoriamente criados com reposição. Cada subconjunto é usado para treinar um classificador do mesmo tipo. As saídas dos classificadores são combinadas por meio do voto maioritário com base em suas decisões. Para uma dada instância, a classe que obtiver o maior número de votos será então a resposta.

Utilizamos os parâmetros base_estimator=SVC(gamma='auto'), n_estimators=10, random_state=0.

Pré-processamento realizado	**Accuracy**
Nenhum	81.6
Normalização	80.8
Estandardização	80.8
Binarização	80.8
Normalização + balanceamento 50/50	80.8
Normalização + balanceamento 50/50 + remoção de outliers	80.8
Normalização + KBest	80.8
Normalização + RFE	80.8
Normalização + Feature Extraction	80.8

Geralmente bons resultados.

5.2 AdaBoost

É uma versão mais genérica do algoritmo de boosting original. O AdaBoost gera um conjunto de hipóteses e combina-as por meio da votação ponderada. As hipóteses são geradas por meio do treinamento de classificadores usando uma distribuição dos dados iterativamente ajustada.

Utilizamos os parametros n_estimators = 100, random_state=0.

Pré-processamento realizado	Accuracy
Nenhum	81.6
Normalização	80.8
Estandardização	80.8
Binarização	80.8
Normalização + balanceamento 50/50	80.8
Normalização + balanceamento 50/50 + remoção de outliers	80.8
Normalização + KBest	80.8
Normalização + RFE	80.8
Normalização + Feature Extraction	80.8

5.3 Voting classifier

Este algoritmo consiste em treinar diferentes algoritmos e juntar os seus resultados. Este algoritmo funciona em diferentes modos Hard Voting, neste modo a classe de um elemento é prevista de acordo com a maioria dos votos dos modelos de treino e o Soft Voting em que a classificação de um elemento e dada a partir da media das classificações dos modelos. Este algoritmos também aceita como parâmetro os pesos dos diferentes modelos.

Utilizamos uma regressão logística, um random forest classifier e um classificador naive bayes.

Pré-processamento realizado	Accuracy
Nenhum	81.6
Normalização	80.8
Estandardização	80.8
Binarização	80.8
Normalização + balanceamento 50/50	80.8
Normalização + balanceamento 50/50 + remoção de outliers	80.8
Normalização + KBest	80.8
Normalização + RFE	80.8
Normalização + Feature Extraction	80.8

Geralmente bons resultados.

5.4 Extra tress Classifier

Este algoritmo cria um número de binary tree classifiers dando a cada um deles um subset do dataset de treino e utiliza bagging nos mesmos.

Pré-processamento realizado	Accuracy
Nenhum	77.92
Normalização	80.83
Estandardização	79.17
Binarização	80.42
Normalização + balanceamento 50/50	61.25
Normalização + balanceamento 50/50 + remoção de outliers	40.83
Normalização + Kbest	80.83
Normalização + RFE	80.83
Normalização + Feature Extraction	80.83

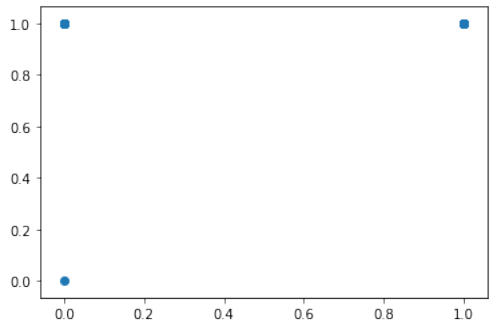
Resultados não são maus mas ficam atrás dos outros métodos de ensemble learning.

Capítulo 6

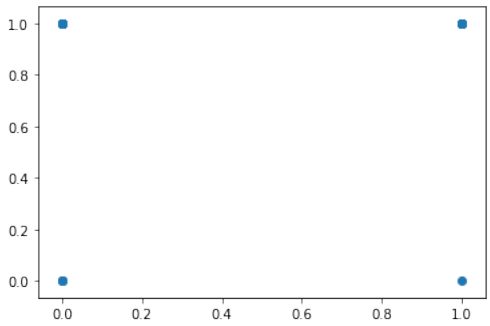
Melhores modelos e hyper parametrization

Após realizar testes das accuracy com diferentes processos de pré-processamento dos dados optamos por realizar mais testes e hyper parametrização dos Modelos SVM e Radius Neighboors classifier e Adaboost.

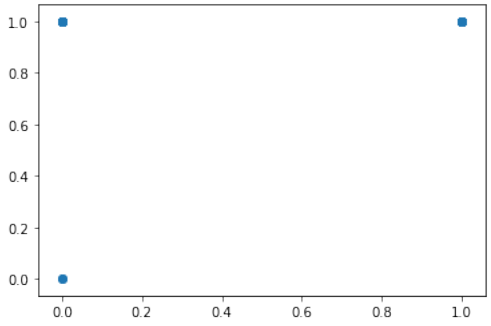
Pré-processamento realizado	**Accuracy**	**Modelo**	**parâmetros**
Normalização + RFE 15 features	82.5	Radius Neighbors classifier	'algorithm': 'auto', 'leaf_size': 15, 'radius': 1.9
Normalização + remoção de outliers	82.5	SVM	'C': 1, 'gamma': 1, 'kernel': 'rbf'
Normalização + RFE 15 features	82.08	Adaboost	'algorithm': 'SAMME', 'learning_rate': 0.1, 'n_estimators': 1000



(a) ADABOOST predictions



(b) RTC predictions



(c) SVM predictions

Figura 6.1: Predições dos diferentes modelos face a classe actual

Podemos notar nestes casos e noutros que o modelo tem tendência a acertar nos casos do tipo 1 e falhar nos do tipo 0,nos experimentamos balancear o dataset com diferentes valores mas notamos que em casos em que o valor de tipo 0 fosse superior aos de tipo 1 o oposto acontecia e mesmo balanceamentos moderados não obtínhamos melhores resultados,com o balanceamento perfeito obtínhamos resultados ainda piores do que sem realizar balanceamento nenhum.

Isto pode dever-se pela falta de casos de estudo no dataset para casos de empregados terem faltado e por não permitirem caracterizar um caso geral dum empregado que falte.

Capítulo 7

Conclusão

Este trabalho permitiu-nos praticar e aprofundar os nossos conhecimentos sobre os modelos dados nas aulas desta UC e outros modelos que existem para aprendizagem e extração de conhecimento.

Achamos que o tratamento dos dados encontra-se como uma parte fundamental deste processo. Ao longo da realização deste trabalho notamos que dataset de treino é importante mesmo quando são aplicadas diferentes técnicas de machine learning e tratamento de dados mesmo que estas sejam complexas e extensivas , quando os dados não são suficientes ou simplesmente não permitem aos modelos chegar a conclusões, os nossos modelos dificilmente irão chegar a resultados aceitáveis.

Na realização deste trabalho notamos pouca dificuldade a chegar a accuracy de 81.6 % no entanto passar deste valor envolveu uma experimentação extensiva de vários modelos, diferentes métodos de tratamento de dados e experimentação com diferentes parâmetros dos modelos. Achamos que a accuracy não é uma métrica perfeita, na maioria dos casos visualizados, visto que como a maioria dos casos no dataset de treino e dataset de avaliação, os elementos são não *absents*, um modelo que preveja que um elemento será sempre não *absent* terá uma accuracy elevada, o que foi a motivação para incluirmos o modelo Dummy Classifier neste projeto.

Visto que fizemos alguma experimentação com diferentes subsets do dataset original, com diferentes proporções das classes a ser avaliadas, e que nenhuma destas nos dá resultados desejáveis podemos averiguar que algumas das causas destes problemas são resultantes de dificuldades em distinguir os elementos destas duas classes. Contudo poderíamos ter experimentado mais técnicas que nos permitissem extrair mais informação do dataset, por exemplo podíamos ter experimentado treinar diferentes modelos de utilizados no ensemble learning com diferentes datasets manualmente,utilizado outros modelos de aprendizagem e mais parâmetros nos modelos utilizados.

Contudo como dito anteriormente este projeto permitiu-nos ganhar mais familiaridade com o processo de extração de conhecimento a partir de um dataset e pretendemos utilizar os conhecimentos adquiridos ao longo deste projeto em trabalhos futuros.