Aprendizagem Automática
Laboratório 5: **Avaliação e Generalização**

N.º:90007 Nome: Alice Rosa
N.º:90026 Nome: Aprígio Malveiro

Turno: 3ª feira - 14h00

2.1 Tarefa de Classificação

Com base nos conjuntos de treino e de teste fornecidos vai-se proceder ao treino e teste de diferentes algoritmos classificadores, com a finalidade de analisar qual deles tem uma melhor prestação na classificação dos dados de teste. Os classificadores que irão ser estudados são os seguintes, SVM (Máquinas de Vetores de Suporte) e *Naive Bayes*.

Podem ser utilizadas várias métricas para avaliar a performance de um determinado modelo. Nesta parte irão ser utilizadas as seguintes: exatidão, precisão, exatidão balanceada, *F-Measure*. Para o cálculo destas medidas recorre-se à matriz de confusão, uma vez que ela nos dá os valores dos termos TP (*true positive*), TN (*true negative*), FP (*false positive*) e FN (*false negative*). Seguidamente especifica-se o que cada um dos parâmetros de avaliação representa:

- **Exatidão:** Taxa de previsões realizadas correctamente relativamente a todas as previsões realizadas.

$$\text{Exatidão} = \frac{TP + TN}{TP + FP + TN + FN} \tag{1}$$

- **Precisão:** Taxa de previsões realmente positivas relativamente a todas as previsões consideradas positivas.

$$\text{Precisão} = \frac{TP}{TP + FP} \tag{2}$$

- **Exatidão balanceada:** Média aritmética da sensibilidade (taxa de positivos reais) e especificidade (taxa de negativos reais).

$$\text{Exatidão balanceada} = \frac{1}{2}\left(\frac{TP}{TP + FN} + \frac{TN}{TN + FP}\right) \tag{3}$$

- ***F-Measure:*** Média harmónica entre a precisão e sensibilidade.

$$F - Measure = \frac{2 \times precisão \times sensibilidade}{precisão + sensibilidade} \tag{4}$$

2.1.1 Implementação do classificador SVM

Para o treino e escolha dos hiperparâmetros do classificador SVM fez-se uso da função *GridSearchCV*, presente na biblioteca *sklearn.model_select*. Esta função é utilizada para identificar qual o melhor conjunto de hiperparâmetros que resultam numa exatidão superior para um determinado tipo de classificador. Isto é, esta função tem como argumento de entrada o tipo de classificador que se está a desenvolver, os parâmetros que queremos testar e ainda o número de partições que o conjunto de treino vai ser subdividido. Esta divisão permite fazer a escolha dos melhores hiperparâmetros para o classificador, a partir do método de validação cruzada.

A partir dos dados de treino fornecidos, procedeu-se à procura exaustiva dos melhores hiperparâmetros para o classificador. Neste sentido, foram testado os seguintes kernels: linear, polinomial e RBF (*Radial Basis Function*),

para diferentes valores dos hiperparâmetros: C, grau (p) para o kernel polinomial e γ para o kernel RBF e uma subdivisão do conjunto dos dados de treino de 6. Seguidamente, utiliza-se o conjunto de dados de teste para a avaliar a prestação do classificador. Os melhores hiperparâmetros obtidos assim como os resultados obtidos na avaliação estão presentes na tabela 1.

Tabela 1: Hiperparâmetros e avaliação do classificador SVM

Melhor kernel	RBF
Melhor C	500
Melhor γ	0.0616
Exatidão	0.875
Precisão	0.75
Exatidão Balanceada	0.9
F-Measure	0.857

A matriz de confusão obtida, a partir da qual se calculam as métricas de performance apresentadas, é representada na tabela 2.

Tabela 2: Matriz de Confusão (SVM)

	Negativo previsto	Positivo previsto
Negativo real	9	0
Positivo Real	3	12

2.1.2 Implementação do classificador *Naive Bayes*

De seguida, passou-se ao estudo do modelo Gaussiano de *Naive Bayes* (NB), implementado a partir da função *GaussianNB*, importada da biblioteca *sklearn.naive_bayes*. Começou-se por treinar o modelo com o conjunto de treino e depois avalia-se a sua performance a partir do conjunto de teste. Os resultados obtidos da avaliação estão presentes na tabela 3.

Tabela 3: Resultados da avaliação do classificador NB

Exatidão	0.625
Precisão	0.7
Exatidão Balanceada	0.(6)
F-Measure	0.5

A matriz de confusão obtida é representada na tabela 4.

Tabela 4: Matriz de Confusão (NB)

	Negativo previsto	Positivo previsto
Negativo real	9	0
Positivo real	9	6

2.1.3 Análise e Conclusões dos classificadores

Relativamente ao classificador de NB, tem-se de ter atenção que este assume que todas as características (*features*) dos dados são condicionalmente independentes. Tal pode justificar o facto de se ter obtido piores resultados para todas as métricas analisadas para este classificador relativamente ao SVM. O largo número de características e o reduzido número de dados de treino e de teste levam a que o classificador de NB não tenha uma boa prestação.

Para o classificador SVM verificou-se um valor de C elevado o que significa que o classificador penaliza de forma rígida o custo de falha na classificação. Por outro lado, o valor de γ reduzido traduz-se uma "baixa curvatura" da linha limite de decisão (*decision boundarie*). A escolha destes hiperparâmetros C e γ não é fácil uma vez que se se testarem demasiados valores o tempo computacional é muito elevado e também basta mudar ligeiramente os valores para os resultados já serem bastantes diferentes. Neste sentido é mais fácil implementar o classificador NB que o SVM.

No entanto, também se verificou que para os valores certos dos hiperparâmetros de SVM obtêm-se muitos melhores resultados em todas as métricas testadas com o classificador. Deste modo, conclui-se que para estes conjuntos de dados pequenos e com muitas características o melhor classificador a utilizar-se para se obter melhores resultados é o SVM.

2.2 Tarefa de regressão

Nesta questão, de modo a resolver o problema de regressão foi utilizado uma Rede Neuronal *Multilayer Perceptron* (MLP) e um o método de Mínimos Quadrados como comparação.

De modo a optimizar os resultados no treino da rede neuronal, começamos por normalizar os dados de treino utilizando a função *min_max_scaler* da biblioteca *sklearn.preprocessing*

Esta função analisa *feature* a *feature* calculando o seu máximo e mínimo e posteriormente utiliza-as para normalizar os dados seguindo a seguinte equação, onde X representa as *features*.

$$X_{std} = \frac{X - min(X)}{max(X) - min(X)}$$

Depois do pré-processamento dos dados ter sido realizada, construiu-se a seguinte rede neuronal composta por três camadas *Dense* sendo a última linear.

Tabela 5: Sumário do modelo

Layer (type)	**Output Shape**	**Paramters**
Inicial (Dense)	(None, 13)	182
Hidden (Dense)	(None, 13)	182
Output (Dense)	(None, 1)	14
Total Paramters	**Trainable Paramters**	**Non-trainable Paramters**
378	378	0

No processo de construção testou-se várias dimensões e número de camadas sendo a estrutura apresentada na tabela 5 a qual se obteve melhores resultados. Os resultados obtidos são representados na figura 1. Em que a perda (*Loss*) é calculada utilizando o erro absoluto médio (*Mean Absolute Error*).

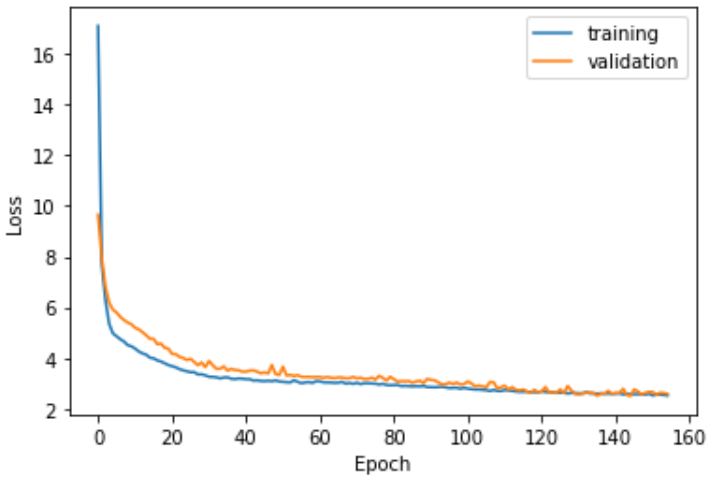

Figura 1: MLP-Regression Loss

Depois de se treinar o modelo, segue-se a validação do mesmo a partir dos dados de teste que foram previamente normalizados com os valores de treino. Para a avaliação da validade do modelo utilizam-se cinco critérios:

- Erro Máximo (*Maximmum Error*)
- Erro Médio Quadratico (*Mean Squared Error*)
- Erro Absoluto Médio (*Mean Absolute Error*)
- Resultado da Variância de Regressão (*Explained Variance Regression Score*)
- Função de Resultado da Regressão (*Regression Score Function*)

As duas últimas métricas são uma forma de avaliar que parte dos resultados são explicados pelos dados de entrada. As suas fórmulas são dadas por:

Regression Score Function:

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

onde $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ e $\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} \epsilon_i^2$

Explained Variance Regression Score:

$$explained\_variance(y, \hat{y}) = 1 - \frac{Var\{y - \hat{y}\}}{Var\{y\}}$$

Passamos a treinar e testar também o *Least Squared Method* e registamos os resultados na tabela abaixo.

Tabela 6: Metrics Values of MLP Regression and Least Squared

Metrics	**MLP**	**LS**
Maximmum Error	12.40	14.59
Mean Squared Error	12.16	20.47
Mean Absolute Error	2.50	3.30
Explained Variance Regression Score	0.84	0.74
Regression Score Function	0.84	0.74

Analisando os valores obtidos para os dois métodos verifica-se que a rede neuronal ajusta-se melhor ao contexto do problema tendo valores melhores para todos os parâmetros de métrica testados. Tal era de esperar tendo em conta o seu elevado número de parâmetros (378) comparativamente aos do LS (14). Apesar de tudo, é possível comprovar que a Rede Neuronal não se ajusta tão bem como seria de esperar aos dados do problema.

Isto acontece, possivelmente porque os dados relativos a uma imobiliária são muitas vezes pouco objectivos, levando há possível existência de *outliers*. O erro máximo de 12.40 num conjunto de soluções onde o valor máximo ronda os 45 é algo preocupante. Analisando o erro quadrático médio e o erro médio absoluto à luz do erro máximo, é possível verificar que os dados que dão origens ao valor de erro máximo constituem um desses *outliers*. O facto de que apenas 84% dos resultados é explicado pelos dados de entrada é mais uma evidência da possível existência de *outliers*. Outra possível explicação é o número reduzido de dados de treino.

Concluindo tendo em conta a natureza do problema e as limitações dos dados, podemos considerar que o modelo do MLP adapta-se melhor ao problema que o método LS.