U.PORTO

FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

INTELIGÊNCIA ARTIFICIAL

PREDICTION OF FOOTBALL EUROPEAN TEAMS GAME OUTCOME

Cláudia Mamede – 201604832@fe.up.pt
João Macedo - 201704464@fe.up.pt
Raúl Viana - up201208089@fe.up.pt

28-05-2020

INTELIGÊNCIA ARTIFICIAL – TRABALHO 2

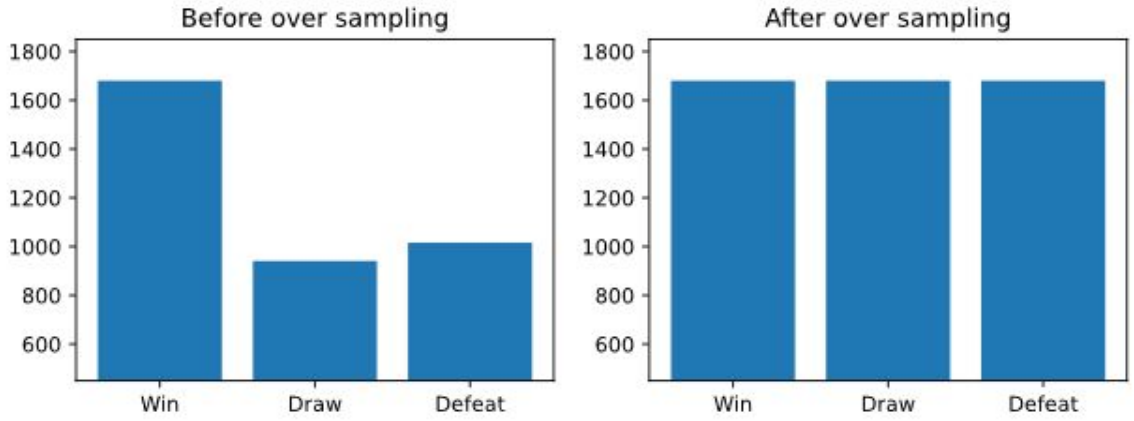
PRÉ PROCESSAMENTO

Removeram-se **inúmeros atributos irrelevantes** para o problema (por exemplo: *season, country, …*). Posteriormente, procedeu-se à **agregação de alguns atributos**: definiram-se os atributos *overall_rating_home, overall_rating_away* que condensam a informação relativa à *performance* dos jogadores de cada equipa; *bet_away e bet_draw* com a média das apostas. Calcularam-se também os **resultados das equipas nos 5 últimos jogos** para se poder compreender melhor a evolução da mesma.

Encontraram-se também algumas entradas na base de dados com ***missing values*** **em atributos bastante importantes** (relativos aos jogadores) o que **impossibilitou a utilização dessas mesmas entradas**.

No final deste pré processamento inicial, verificou-se que os dados não estavam balanceados e segui-se uma estratégia de **over sampling** para ultrapassar este problema.

Uma vez que os dados variam muito em unidades e algoritmos como o KNN são bastante sensíveis a isso optamos por **normalizar os dados**.



Para escolher o conjunto de treino e conjunto de teste, atualmente, optou-se por um **random split** de 80/20%.

INTELIGÊNCIA ARTIFICIAL – TRABALHO 2

MODELOS DE APRENDIZAGEM

Os modelos escolhidos são: Árvore de Decisão, Rede Neuronal e *K-Nearest Neighbor*. Para todos eles procurou-se encontrar a melhor parametrização para o conjunto de treino, utilizando o sklearn.model_selection.GridSearchCV (com cv=5).

Árvore de decisão

Para este modelo geralmente **não é necessário normalizar os dados** mas para tornar a comparação entre os 3 algoritmos mais fidedigna, optou-se por utilizar o mesmo *input* (comparou-se a *accuracy* para dados normalizados e não normalizados e não havia diferença significativa).

Relativamente ao balanceamento dos dados, também se poderia ter optado por utilizar o parâmetro *class_weight='balanced'* mas esta estratégia difere um pouco da escolhida.

Parâmetros a variar:

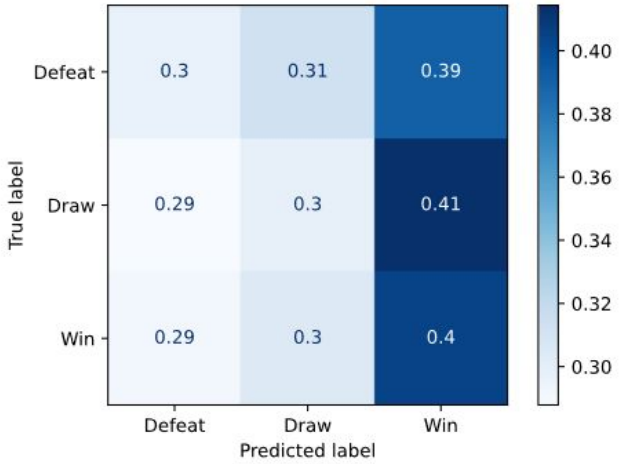
- 'max_depth' (5 - 150)
- 'criterion' ('gini' ou 'entropy')
- 'splitter' ('random' ou 'best')

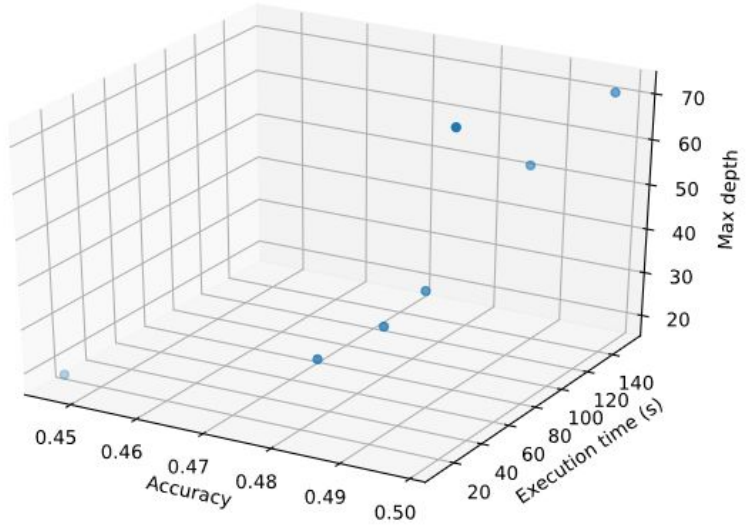
Parâmetros selecionados:

- 'max_depth': 55
- 'criterion': gini
- 'splitter': random

Best score: `0.4946428571428571`

Accuracy: `0.3470173187940988`





Comparação *accuracy, execution time* e *max_depth:* como expectável, um aumento da profundidade da árvore leva a um aumento da *accuracy e* do tempo de execução do modelo.

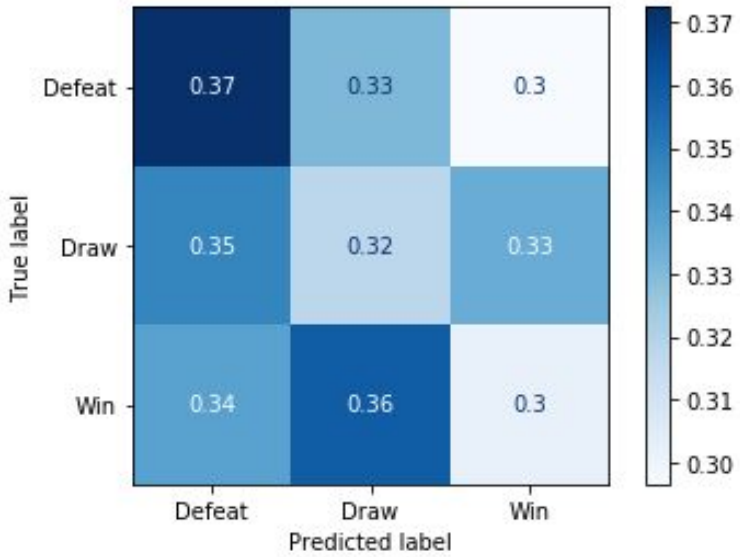
INTELIGÊNCIA ARTIFICIAL – TRABALHO 2

K-nearest neighbor (KNN)

O modelo KNN pode ser afinado com diferentes parâmetros, que melhoram os resultados em função das características dos dados.

- peso da distância: é dado um maior peso a vizinhos mais próximos;
- métrica: forma de cálculo da distância entre os vizinhos
- número de vizinhos: o número de vizinhos a considerar para a classificação.

Os melhores hyperparametros encontrados pelo sklearn.model_selection.GridSearchCV foram: metric - 'euclidean'; número vizinhos - ‘23’; peso - 'distance’



Accuracy: 0.3258499037844772

Precision: 0.33001335610520505

Parâmetros a variar:

- metric: Minkowski, Manhatan, Euclidean
- n_neighbors: (5:23)
- weight: distance, uniform

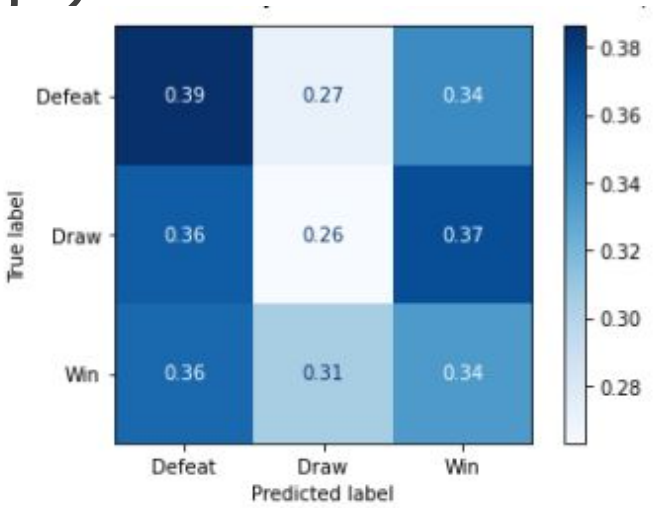
Parâmetros selecionados:

- metric: Euclidean
- n_neighbors: 23
- weight: distance

INTELIGÊNCIA ARTIFICIAL – TRABALHO 2

Rede Neuronal

O modelo de rede neuronal para previsão do vencedor de jogos que pretendemos utilizar será um **Multi-layer Perceptron (MLP)**, utilizando **scikit-learn**, particularmente a classe **MLPClassifier** do módulo sklearn.neural_network.
Os parâmetros a que se testaram vão muito para além do que se encontra no notebook, tendo sido maioritariamente utilizado RandomizedSearchCV nessa fase para reduzir o espaço de procura ao que se encontra neste momento no notebook.
Os restantes parâmetros variados estão listados no Notebook entregue.



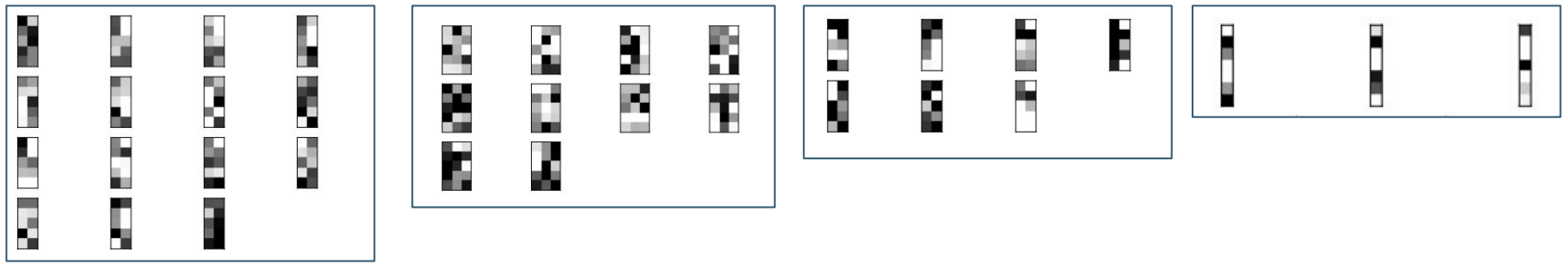
Parâmetros a variar:

- 'hidden_layer_sizes': (10,7),(15,10,7)
- 'activation': 'tanh', 'relu'
- 'alpha': 0.0001, 0.05

Parâmetros selecionados:

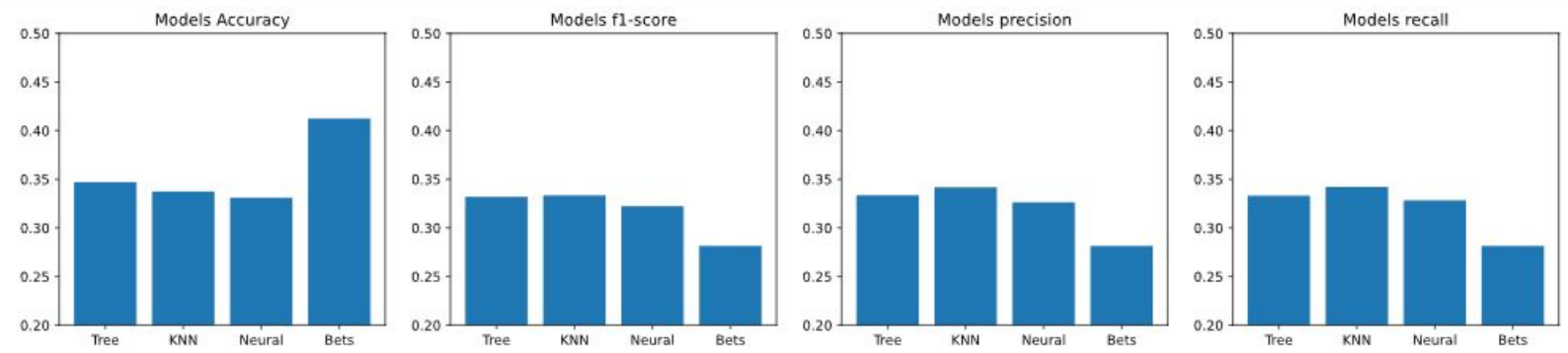
- 'hidden_layer_sizes': (15,10,7)
- 'activation': "relu'
- 'alpha': 0.05

***Best score*:** `0.383531746031746`
***Accuracy*:** `0.33098139833226425`



Visualização dos Coeficientes do *MLPClassifier* com melhores resultados*:* Cada uma das imagens refere-se aos coeficientes de cada uma das camadas da rede.

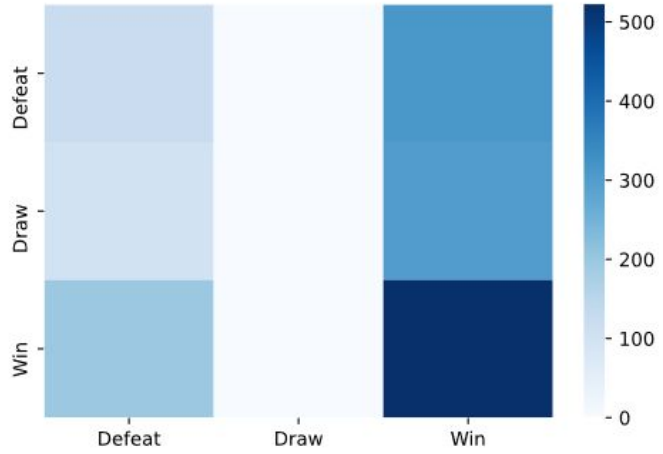
COMPARAÇÕES



De um modo geral, os **algoritmos apresentam todos resultados muito semelhantes**, com a *performance* a ser mais influenciada pela escolha do *dataset* do que pelo modelo.

Para este caso específico, a Decision Tree apresentou melhores resultados contudo é importante referir que estamos a utilizar um *random splitter* para dividir os dados em conjuntos de treino e teste pelo que os resultados oscilam um pouco entre execuções.

Não podemos também esquecer que o futebol é um jogo imprevisível e que ainda que existam alguns fatores bastante "pesados" como a qualidade da equipa, evolução da mesma ao longo da época, etc..., é sempre muito complicado prever com total rigor os resultados de um jogo.



Análise das *odds* "profissionais": decidimos também analisar, com recurso aos dados das *odds* da tabela *match*, a capacidade dos profissionais em prever os resultados dos jogos.