Utilização de algoritmos de inteligência artificial na previsão de resultados de partidas de futebol

Olesio Gardenghi Neto [1]

Resumo: O futebol é uma modalidade esportiva disputada por duas equipes, cujo objetivo é transpor uma bola entre as extremidades – chamadas de baliza - utilizando basicamente toques com o pé. O vencedor da partida será o time que conseguir atingir o objetivo mais vezes no jogo. O interesse de tantas pessoas no mundo pelo futebol gera não apenas telespectadores, mas também muitas movimentações financeiras em torno desse esporte. Sistemas computacionais que trabalham com a previsão de resultados e que auxiliam a minimizar os riscos e maximizar os lucros tornam-se então uma importante ferramenta de trabalho para o dia a dia do futebol. O objetivo deste projeto é testar e comparar o desempenho de alguns classificadores na previsão de resultados de partidas de futebol dos campeonatos: Premier League, Serie A e Brasileirão Série A. Os classificadores utilizados foram regressão logística, árvore de decisão, floresta aleatória, knn, svm e mlp. Foram utilizados para divisão dos dados os métodos treino e teste e de validação cruzada. Como resultado tem-se a acurácia de cada classificador.
Palavras-chave: Futebol. Previsão de Resultados. Classificadores. Inteligência Artificial. Aprendizado de Máquina.

Use of artificial intelligence algorithms in predicting soccer match results

Abstract: Soccer is a sport played by two teams, whose goal is to transpose a ball between extremities - called goalposts - basically utilizing foot strokes. The winner of the match will be the team that can reach the goal more often in the game. The interest of so many people in the world for soccer generates not only viewers, but also many financial transactions around this sport. Computer systems that work on predicting results and helping to minimize risks and maximize profits become an important day-to-day work tool for soccer. The objective of this project is to test and compare the performance of some classifiers in predicting soccer matches results for Premier League, Serie A and Brasileirão Serie A. The classifiers used were logistic regression, decision tree, random forest, knn, svm and mlp. Training and testing and cross-validation methods were used for data division. The result is the accuracy of each classifier.
Keywords: Soccer. Results prediction. Classifiers. Artificial Intelligence. Machine Learning.

[1]Estudante de Ciência da Computação, IFTM, Campus Ituiutaba, netogardenghi@gmail.com

1 INTRODUÇÃO

O futebol é uma modalidade esportiva disputada por duas equipes de 11 jogadores cada um, cujo objetivo é transpor uma bola entre as extremidades – chamadas de baliza - utilizando basicamente toques com o pé. A baliza, mais conhecida como trave, é um retângulo formado por duas traves verticais, perpendiculares ao solo, e uma paralela ao solo. Entre as traves posiciona-se o goleiro o qual é o único jogador que pode colocar as mãos na bola defendendo o gol. O vencedor da partida será o time que conseguir atingir o objetivo mais vezes no jogo (SFEIR, 2011). O fácil entendimento, o baixo custo e a dinâmica empolgante fizeram dessa modalidade uma verdadeira febre mundial. A última copa do mundo jogada na Rússia no ano de 2018 atingiu uma audiência televisiva de mais de 3,5 bilhões de pessoas (FIFA, 2018).

O interesse de tantas pessoas no mundo pelo futebol gera não apenas telespectadores, mas também muitas movimentações financeiras em torno desse esporte. A copa do mundo de 2018 gerou de lucro para a FIFA cerca de 5,35 bilhões de dólares (FIFA, 2019). A movimentação financeira envolve patrocínios a clubes e seleções nacionais, venda de ingressos e produtos licenciados. Além é claro de transações de transferências envolvendo jogadores. Esse fluxo de movimentação financeira atrai uma série de investidores, que sempre visam lucrar. Sistemas computacionais que trabalham com a previsão de resultados e que auxiliam a minimizar os riscos e maximizar os lucros tornam-se então uma importante ferramenta de trabalho para o dia a dia do futebol (PERIN; VUILLEMOT; FEKETE, 2013).

A utilização de técnicas de inteligência artificial para realizar predição e previsão de resultados de partidas de futebol é um problema bastante explorado na literatura (PERIN; VUILLEMOT; FEKETE, 2013). Em Martins et al. (2017) foi realizado um estudo com a predição de resultados de partidas de futebol envolvendo dados do Campeonato Brasileiro, Espanhol e Inglês. A pesquisa demonstrou a eficiência de algoritmos de Inteligência Artificial para demonstrar quais as características mais relevantes na predição de resultados de partidas de futebol. Tal trabalho deixou em aberto a possibilidade de utilização desses algoritmos para realizar a previsão de resultados de partidas de futebol.

O resultado de uma partida de futebol é o personagem central de inúmeros estudos científicos. Existe um esforço muito grande para melhorar as táticas do jogo e as características de uma determinada equipe. Na literatura, existem estudos que se concentram nas previsões de jogos de futebol (CONSTANTINOU; FENTON; NEIL, 2013). A previsão em partidas de futebol é composta por resultado de uma partida (vitória, empate e derrota) que pode ser utilizada para

várias finalidades, incluindo apostas. Muitos esforços foram dedicados para a compreensão do futebol a partir da perspectiva dos resultados preditivos. Prever os resultados é um problema difícil devido ao grande número de fatores que devem ser levados em consideração e que nem sempre podem ser representados em valores quantitativos (HUCALJUK; RAKIPOVIć, 2011). Por exemplo, uma equipe pode dominar completamente uma partida sob alguns aspectos, como o número de finalizações certas, o número de passes certos ou a posse de bola e não conseguir marcar um gol a mais do que a equipe adversária para vencer uma partida (BROOKS; KERR; GUTTAG, 2016).

Na literatura é possível encontrar estudos que envolvam predição de resultados de futebol e também de outros esportes coletivos. Ulmer e Fernandez (2013) estudaram as técnicas Baseline, Gaussian Naive Bayes, Hidden Markov Model, Multimodal Naive Bayes, SVM, RBF, One vs All SGD para prever resultados utilizando como parâmetros os gols marcados por cada time em 10 temporadas (da temporada 2002-03 à temporada 2011-12) do Campeonato Inglês. Hucaljuk e Rakipović (2011) pesquisaram a previsão de resultados para a UEFA Champions League também através de gols marcados com o seguintes algoritmos: Naive Bayes (NB), Redes Bayesianas, Logitboost, K-Nearest Neighbours (KNN), RF e Redes Neurais Artificiais. Com o classificador SVM, Igiri (2015) estudou os dados relacionados aos resultados de partidas do Campeonato Inglês.

Com uma quantidade muito maior de características, Owramipur, Eskandarian e Mozneb (2013) utilizaram dados obtidos através de scout e também de fisiologia para analisar o time de futebol do Barcelona no Campeonato Espanhol. Nesse trabalho, os autores descreveram uma abordagem de redes bayesianas para previsão de resultados de partidas de futebol com o software NETICA. Apesar dos bons resultados obtidos, o modelo considera apenas uma equipe para realizar a previsão do resultado das partidas. Tax e Joustra (2015) usaram os seguintes algoritmos de classificação: CHIRP, LogitBoost, DTNB, FURIA, HiperPipes, J48, Naive Bayes, Perceptron e RF. Para a seleção, eles aplicaram Relief, CfsSubsetEval, e PCA para tentar determinar dentre as 65 características levantadas quais as mais relevantes para aumentar a acurácia dos classificadores. Entretanto, eles não conseguiram determinar quais seriam estas características. O estudo levou em consideração jogos do Campeonato Holandês.

Duarte, Soares e Teixeira (2015) pesquisaram os classificadores: C5.0, JRip, RF, KNN, SVM e NB para prever as partidas do Campeonato Português com as informações obtidas através de scout e também dados psicológicos dos jogadores das equipes o que, no entanto,

não melhorou o desempenho em termos de acurácia. Por esse motivo, é um desafio investigar informações e estratégias que facilitem a previsão dos resultados dos jogos.

Outros estudos foram desenvolvidos para prever o resultado de partidas de futebol utilizando algoritmos de aprendizagem de máquinas. Esses algoritmos são ferramentas que recebem como entrada um conjunto de características e fornece como saída a previsão do resultados (vitória, empate e derrota). Existem algoritmos que podem fornecer uma resposta mais adequada ao problema (PENDHARKAR; KHOSROWPOUR; RODGER, 2000).

O objetivo deste trabalho é explorar algoritmos de inteligência artificial como ferramenta de previsão de resultados de partidas de futebol. Como resultado comparar e demonstrar a eficácia para os classificadores utilizados.

2 DESENVOLVIMENTO

2.1 MATERIAIS

Este capítulo tem a finalidade de apresentar as tecnologias utilizadas.

2.1.1 Python

Lançada por Guido Van Rossum em 1991, Python é uma linguagem de programação dinâmica, interpretada, robusta, multiplataforma e multiparadigma. Seus objetivos eram produtividade e legibilidade. Foi utilizada como linguagem de programação deste projeto.

2.1.2 NumPy

NumPy é uma biblioteca para linguagem Python que contém vetores e matrizes de N dimensões além de fornecer um grande conjunto de funções e operações matemáticas. Foi utilizada neste projeto para facilitar o uso de operações matemáticas e como dependência do Matplotlib.

2.1.3 Pandas

Pandas é uma biblioteca para linguagem Python para trabalhar com dados de forma rápida e intuitiva. Fornece estruturas de dados rápidas, flexíveis e expressivas, projetadas para facilitar o trabalho com dados estruturados e de séries temporais. Possui o objetivo de se tornar a

ferramenta de análise/manipulação de dados de código aberto mais poderosa e flexível (PYPI, 2019). Foi utilizada neste projeto para lidar com a análise e manipulação da base de dados.

2.1.4 Matplotlib

Matplotlib é uma biblioteca de plotagem de gráficos em Python. Com esta biblioteca é possível gerar histogramas, gráficos de barras, de erros, de dispersão e etc. com poucas linhas de código. Foi utilizada neste projeto para a plotagem de gráficos.

2.1.5 Seaborn

Seaborn é uma biblioteca para linguagem Python feita para plotar gráficos estatísticos baseada na Matplotlib. Ela oferece estilos e cores diferentes além de prover integração com Pandas. Foi utilizada neste projeto para a plotagem de gráficos.

2.1.6 Scikit-learn

Scikit-learn é uma biblioteca de aprendizado de máquina para linguagem Python. Inclui algoritmos de classificação, regressão e agrupamento e também possui métodos de pré-processamento de dados. Começou a ser desenvolvida em 2007 e foi lançada em 2010 (PEDREGOSA et al., 2011). Foi utilizada neste projeto para fazer o pré-processamento, treino, teste e previsão dos dados.

2.1.7 Jupyter Notebook

Jupyter Notebook é um ambiente computacional web originado do IPython que suporta dezenas de linguagens de programação. O documento Jupyter contém uma lista ordenada de células que podem conter código, texto, fórmulas matemáticas, plotagens e imagens. Foi utilizado neste projeto como ambiente de desenvolvimento.

2.2 METODOLOGIA

A metodologia utilizada é apresentada na figura 1, a seguir, a figura apresenta o fluxograma do sistema proposto.

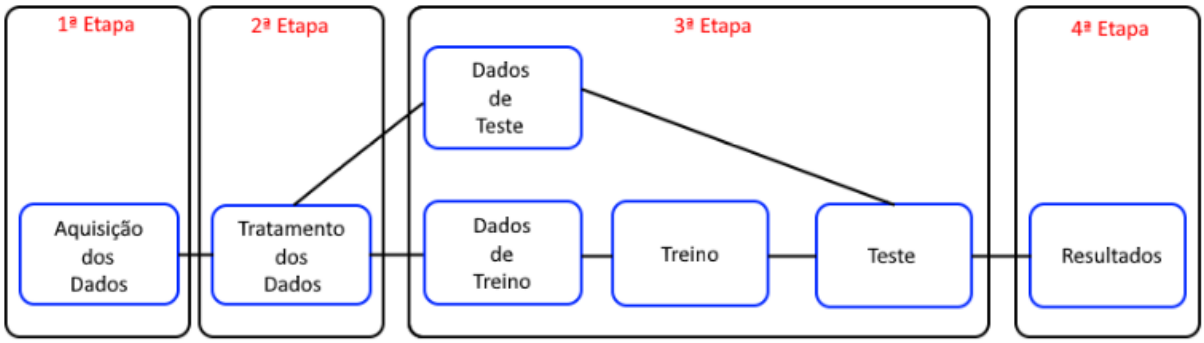

Figura 1: Fluxograma do sistema proposto

2.2.1 1ª Etapa - Aquisição dos dados

A primeira etapa envolve a escolha e a montagem de uma base de dados, algumas bases públicas estão disponíveis em https://www.football-data.co.uk e foram objetos de estudo em Ulmer e Fernandez (2013). Foram escolhidas deste mesmo site as seguintes bases de dados:

- Premier League (Temporada 2018/2019);
- Serie A (Temporada 2018/2019).

Foi escolhida também a base de dados do Brasileirão 2012 coletada do UOL Sports e disponível em https://github.com/zandree/statsBrazilianLeagueChampionship. Todas as bases estão no formato CSV.

As bases da Premier League e Serie A foram feitas como cada linha sendo um jogo e esta linha possui os dados do time de casa e o time de fora. O resultado é dado como casa, empate ou fora. A base do Brasileirão tem sua construção diferente. Cada linha também é um jogo porém esta linha só possui os dados de um dos times, ou seja, cada jogo ocorrido terá 2 linhas uma para o time de casa e uma para o de fora. O resultado é dado com vitória, empate ou derrota.

2.2.2 2ª Etapa - Tratamento dos dados

Esta etapa envolve todo o pré-processamento de dados. Primeiramente foi criado um Data-Frame utilizando o Pandas para ler as bases de dados. A base da Premier League possui 380 linhas e 62 colunas enquanto a da Serie A possui 380 linhas e 61 colunas e ambas as bases não possuem dados faltantes. A base do Brasileirão possui 760 linhas, 50 colunas e 4 linhas com dados faltantes. As linhas com dados faltantes foram removidas e a base passou a ter 756 linhas.

O próximo passo foi selecionar as características para a previsão. Nas bases Premier League e Serie A foram utilizadas as seguintes características:

- FTR - Resultado da partida (Casa, Empate, Fora);
- HS - Chutes do time da casa;
- AS - Chutes do time de fora;
- HST - Chutes do time da casa no alvo(gol);
- AST - Chutes do time de fora no alvo(gol);
- HF - Faltas cometidas pelo time da casa;
- AF - Faltas cometidas pelo time de fora;
- HC - Escanteios do time da casa;
- AC - Escanteios do time de fora;
- HY - Cartões Amarelos do time da casa;
- AY - Cartões Amarelos do time de fora;
- HR - Cartões Vermelhos do time da casa;
- AR - Cartões Vermelhos do time de fora;
- HTHG - Gols marcados pela equipe da casa até o intervalo;
- HTAG - Gols marcados pela equipe de fora até o intervalo.

Na base de dados do Brasileirão foram utilizadas as seguintes características:

- assistances - Assistências;
- receivedBalls - Bolas recebidas;
- recoveredBalls - Bolas recuperadas;
- lostBalls - Bolas perdidas;
- yellowCards - Cartões amarelos;

- redCards - Cartões vermelhos;
- receivedCrossBalls - Cruzamentos recebidos;
- missedCrossBalls - Cruzamentos perdidos;
- defenses - Defesas;
- sucessfulTackles - Desarmes com sucesso;
- unsucessfulTackles - Desarmes sem sucesso;
- sucessfulDribles - Dribles com sucesso;
- unsucessfulDribles - Dribles sem sucesso;
- givenCorners - Escanteios dados;
- receivedCorners - Escanteios recebidos;
- receivedFouls - Faltas recebidas;
- committedFouls - Faltas cometidas;
- goodFinishes - Boas finalizações;
- badFinishes - Finalizações ruins;
- ownGols - Gols marcados;
- offsides - Impedimentos;
- sucessfulLongPasses - Passes longos com sucesso;
- unsucessfulLongPasses - Passes longos sem sucesso;
- sucessfulPasses - Passes com sucesso;
- unsucessfulPasses - Passes sem sucesso;
- win - Vitória;
- draw - Empate;
- defeat - Derrota.

Em Premier League e Serie A os dados de FTR são categóricos e como FTR é o alvo da previsão foi preciso transformá-los para numéricos. Casa passou a ser 2, empate passou a ser 1 e fora passou a ser 0. No Brasileirão as características win, draw e defeat foram mescladas em uma só, chamada FTR, que também é o alvo da previsão, sendo 2 - vitória, 1 - empate e 0 - derrota.

Com as características definidas o próximo passo foi padronizar os dados. Os dados foram padronizados com o *StandardScaler* que está presente no Scikit-learn. O *StandardScaler* transforma os dados de forma que sua distribuição tenha um valor médio de 0 e o desvio padrão 1. Cada valor no conjunto de dados será subtraído pela média das amostras e dividido pelo desvio padrão de todo o conjunto de dados (LEARN, 2019). O motivo da padronização foi para que classificadores como KNN possam performar melhor.

2.2.3 3ª Etapa - Treino e teste

Foi definido qual era o alvo da previsão. Em todas as bases o alvo é a característica FTR a qual contém o resultado da partida. As demais características foram usadas para a previsão de FTR. Nesta etapa foram utilizados dois métodos diferentes de divisão de dados.

O primeiro método foi o de divisão de treino e teste. Este método embaralha os dados e os divide em uma porcentagem para treino e outra para teste. Foi dividido 70% para treino e 30% para teste. A divisão dos dados gera quatro novos conjuntos de dados, sendo eles: características para treino, características para teste, alvo para treino e alvo para teste.

O segundo método foi o de validação cruzada k-*fold*. Neste método os dados de entrada são divididos em subconjuntos de dados k(chamados de *folds*). O modelo é treinado com k-1 subconjuntos e depois avaliado no subconjunto que não foi utilizado para treinamento. O procedimento é repetido k vezes e em cada vez um subconjunto diferente é reservado para avaliação (SOUZA, 2019). Foram utilizados 10 *folds*.

Para previsão dos resultados foram utilizadas diferentes abordagens. Foi feito a previsão com a base completa e com os dados do alvo agrupados de 2 em 2. Nas bases Premier League e Serie A foram feitas previsões com as seguintes abordagens:

- Casa, Empate e Fora;
- Casa e Fora;
- Casa e Empate;

- Fora e Empate.

Na base do Brasileirão as abordagens foram:

- Vitória, Empate e Derrota;
- Vitória e Derrota;
- Vitória e Empate;
- Derrota e Empate.

O próximo passo foi a criação dos modelos de previsão. Os classificadores utilizados foram:

- Regressão logística;
- Árvore de decisão;
- Floresta aleatória;
- K Nearest Neighbours(KNN);
- Support vector machine(SVM);
- Multi Layer Perceptron(MLP).

A regressão logística é uma técnica estatística que tem como objetivo modelar, a partir de um conjunto de observações, a relação "logística" entre uma variável resposta dicotômica e uma série de variáveis explicativas numéricas e/ou categóricas (CABRAL, 2013). Define-se uma variável dependente (Y), ou variável de saída, e procura-se verificar a influência de uma ou mais variáveis ditas variáveis independentes (X's) sobre esta variável dependente (ZANINI, 2007). A regressão logística busca calcular ou prever a probabilidade da variável dependente assumir um determinado valor em função de outras variáveis. Regressão logística é um algoritmo de aprendizagem supervisionada.

Árvores de decisão são modelos estatísticos para a classificação e previsão de dados. Formada por um conjunto de nós de decisão uma árvore começa com um nó, que se divide em possíveis resultados. Cada resultado leva a nós adicionais, que se ramificam em outras possibilidades. Uma árvore de decisão é essencialmente uma série de declarações if-elses, que quando aplicados a um registro de uma base de dados, resultam na classificação daquele registro (CARACIOLO, 2009). Árvore de decisão é um algoritmo de aprendizagem supervisionada.

Floresta aleatória é um classificador que consiste em uma combinação de árvores de decisão. Cada árvore depende dos valores de vetores aleatórios amostrados de forma independente e distribuídos igualmente para todas as árvores na floresta (BREIMAN, 2001). Uma floresta aleatória é construída por um simples voto de árvores de decisão usando o algoritmo Bagging. Todas as árvores treinadas são combinadas em uma composição com uma votação simples, usando o erro médio de todas as amostras (DMITRIEVSKY, 2018). Floresta aleatória é um algoritmo de aprendizagem supervisionada.

K vizinhos mais próximos, do inglês *K nearest neighbours(KNN)* é um algoritmo de aprendizagem supervisionada. O KNN classifica um elemento com base na proximidade de seus k vizinhos. O algoritmo calcula a distância do elemento alvo aos demais e as ordena da menor para o maior. Após ordenadas são selecionados apenas os k mais próximos vizinhos deste elemento alvo. Ele será classificado com as mesmas características do tipo de vizinho que está em maior quantidade dos k selecionados.

Máquina de vetores de suporte, do inglês *Support vector machine(SVM)* é um algoritmo de aprendizagem supervisionada, cujo objetivo é classificar determinado conjunto dados que são mapeados para um espaço de características multidimensional usando uma função kernel. Nela, o limite de decisão no espaço de entrada é representado por um hiperplano em dimensão superior no espaço (BOSER; GUYON; VAPNIK, 1992). O que o SVM faz é encontrar um hiperplano entre dados de duas classes buscando maximizar a distância entre os pontos mais próximos em relação a cada uma das classes.

Perceptron multicamadas, do inglês *Multi Layer Perceptron(MLP)* é uma rede neural com camadas ocultas com um número indeterminado de neurônios. O MLP utiliza métodos derivados do gradiente no ajustes de seus pesos por retropropagação(*backpropagation*). A rede consiste em uma camada de entrada, uma ou mais camadas escondidas e uma ou mais camadas de saída. Um sinal de entrada é propagado da entrada para saída e a saída propaga um sinal em caminho reverso alterando os pesos, para uma nova validação (AMARAL JOSE MARIA CELESTINO DE LIMA, 2014).

O scikit-learn possui todos esses classificadores. Para fazer a previsão primeiro é preciso instanciar o classificador, depois chamar a função de treino e ao final a função de previsão. Os modelos que serão apresentados a seguir valem para as 3 bases de dados. A abordagem a ser tratada é a da base de dados completa, a abordagem de 2 em 2 segue do mesmo raciocínio.

Conforme a figura 2 a seguir, foi instanciado o classficador de regressão logística, depois

chamada a função *fit* responsável por treinar o modelo passando como parâmetro as características de treino e o alvo de treino. A função *predict* é responsável pela previsão e nela é passada como parâmetro o alvo de teste, ou seja, com base no treinamento feito com as bases de treino o classificador vai tentar prever uma base ainda não vista, no caso a do alvo de teste. Todo esse procedimento foi feito com as bases do método de divisão 70-30. Para a validação cruzada é utilizado o *cross_val_score* passando como parâmetros o classificador, o vetor de características, o alvo da previsão, e a quantidade de *folds*.

```
logistic_regression = LogisticRegression(solver='lbfgs', multi_class='auto')
logistic_regression.fit(X_train, y_train)
predict_logistic_regression = logistic_regression.predict(X_test)
reg_log_all = logistic_regression.score(X_test, y_test) * 100
cross_log_all = max(cross_val_score(logistic_regression, X, y, cv=10)) * 100
```

Figura 2: Modelo de regressão logística

A árvore de decisão segue do mesmo raciocínio, conforme ilustrado na figura 3 a seguir.

```
decision_tree = DecisionTreeClassifier()
decision_tree.fit(X_train, y_train)
predict_decision_tree = decision_tree.predict(X_test)
dec_tree_all = decision_tree.score(X_test, y_test) * 100
cross_dec_tree_all = max(cross_val_score(decision_tree, X, y, cv=10)) * 100
```

Figura 3: Modelo de árvore de decisão

Para os modelos de floresta aleatória e KNN é preciso definir o número de árvores para um e o número de vizinhos para o outro. A fim de encontrar qual o valor que gere a melhor previsão é feito um "método cotovelo". Este método consiste em fazer previsões alterando os valores dentro de um intervalo. Ao terminar as previsões é feito a média das diferenças entre o valor previsto e o valor real e este valor é salvo em um vetor. O melhor número para utilizar é aquele que representa a menor diferença entre o valor real e o previsto. A floresta aleatória e o KNN seguem do mesmo raciocínio que os demais.

```
#Método do cotovelo
error_rate = []

for i in range(1,200):
    random_forest = RandomForestClassifier(n_estimators=i)
    random_forest.fit(X_train, y_train)
    predict_random_forest = random_forest.predict(X_test)
    error_rate.append(np.mean(predict_random_forest!=y_test))
```

Figura 4: Método cotovelo floresta aleatória

```
random_forest = RandomForestClassifier(n_estimators=error_rate.index(min(error_rate)))
random_forest.fit(X_train, y_train)
predict_random_forest = random_forest.predict(X_test)
rand_for_all = random_forest.score(X_test, y_test) * 100
cross_rand_for_all = max(cross_val_score(random_forest, X, y, cv=10)) * 100
```

Figura 5: Modelo de floresta aleatória

```
error_rate = []

for i in range(1,40):
    knn = KNeighborsClassifier(n_neighbors=i)
    knn.fit(X_train, y_train)
    predict_knn = knn.predict(X_test)
    error_rate.append(np.mean(predict_knn!=y_test))
```

Figura 6: Método cotovelo KNN

```
knn = KNeighborsClassifier(n_neighbors=error_rate.index(min(error_rate)))
knn.fit(X_train, y_train)
predict_knn = knn.predict(X_test)
knn_all = knn.score(X_test, y_test) * 100
cross_knn_all = max(cross_val_score(knn, X, y, cv=10)) * 100
```

Figura 7: Modelo KNN

Para o modelo de SVM é utilizado o *GridSearchCV*. O *GridSearchCV* é um método onde é testado diferentes valores para alguns parâmetros do classificador a fim de descobrir qual combinação gera a melhor previsão. Primeiro é definido qual o classificador depois, uma lista com os parâmetros que vão ser testados e seus respectivos valores e a quantidade de *folds*. Para o modelo de SVM não foi utilizado o método de divisão 70-30.

```
param_grid = {'C':[0.1,1,10,100,1000],'gamma': [1,0.1,0.01,0.001,0.001], 'kernel':['rbf']}
grid = GridSearchCV(SVC(),param_grid,refit=True,cv=10,iid=False)
grid.fit(X, y)
predict_svm = grid.predict(X_test)
svm_all = grid.score(X_test, y_test) * 100
```

Figura 8: Modelo SVM

O modelo de MLP segue do mesmo raciocínio de regressão logística e árvore de decisão, conforme ilustrado na figura a seguir.

```
mlp_classifier = MLPClassifier(hidden_layer_sizes=(1,8), activation='logistic', solver='adam', max_iter=10
00)
mlp_classifier.fit(X_train,y_train)
predict_mlp_classifier = mlp_classifier.predict(X_test)
mlp_all = mlp_classifier.score(X_test, y_test) * 100
cross_mlp_all = max(cross_val_score(mlp_classifier, X, y, cv=10)) * 100
```

Figura 9: Modelo MLP

2.2.4 4ª Etapa - Resultados

Para avaliar a precisão dos classificadores foi utilizado o método *score* que esta contido em cada classificador. Este método retorna a acurácia ou o coeficiente de determinação de cada classificador. Os resultados obtidos foram:

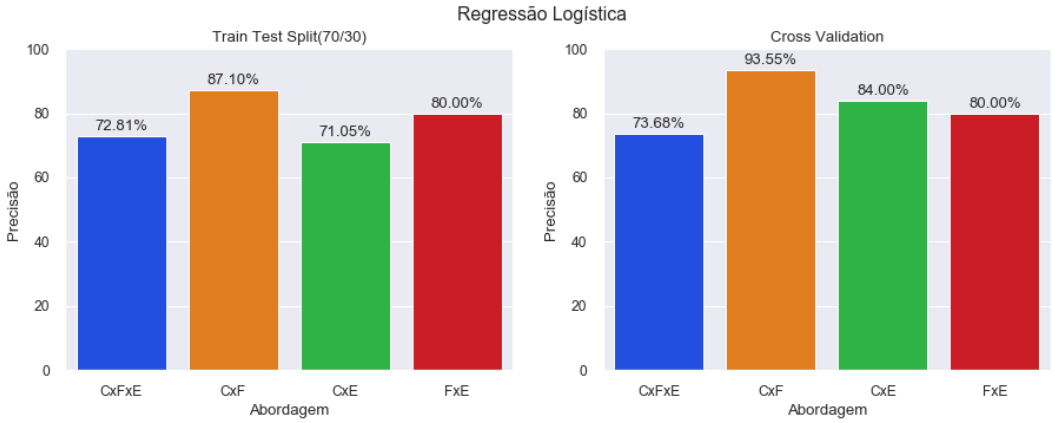

Figura 10: Regressão logística Premier League

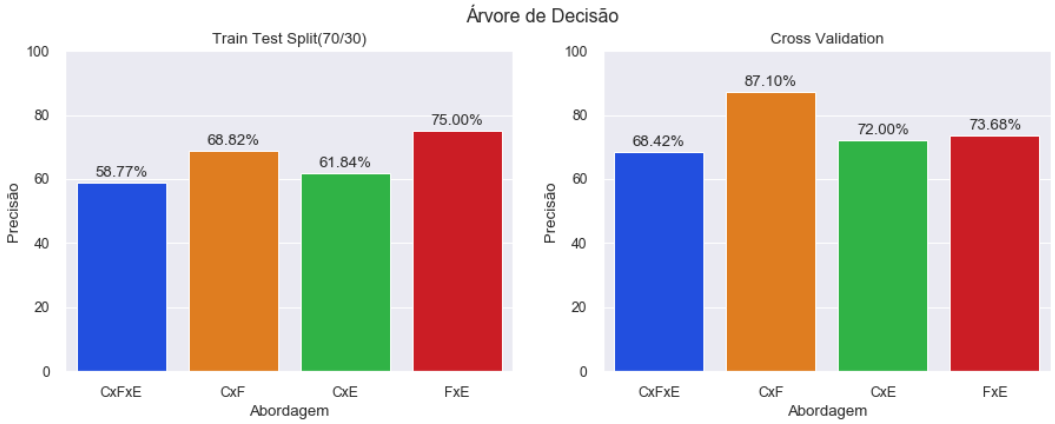

Figura 11: Árvore de decisão Premier League

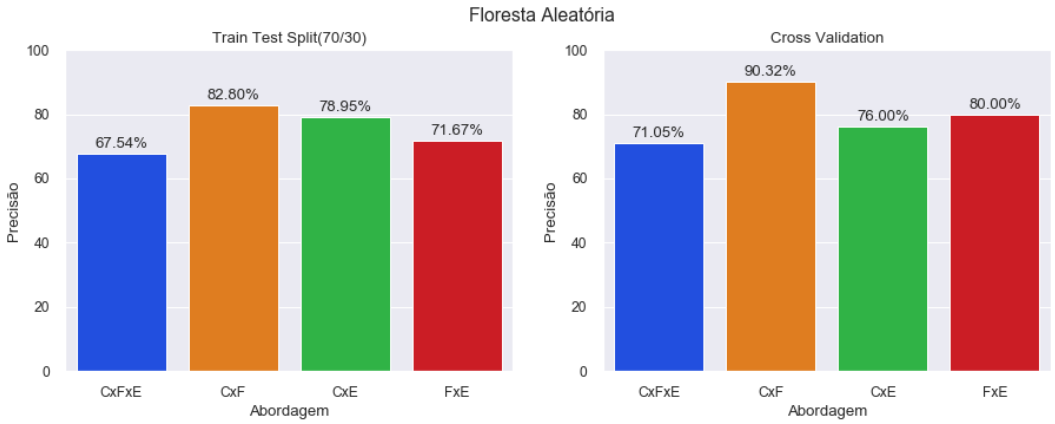

Figura 12: Floresta aleatória Premier League

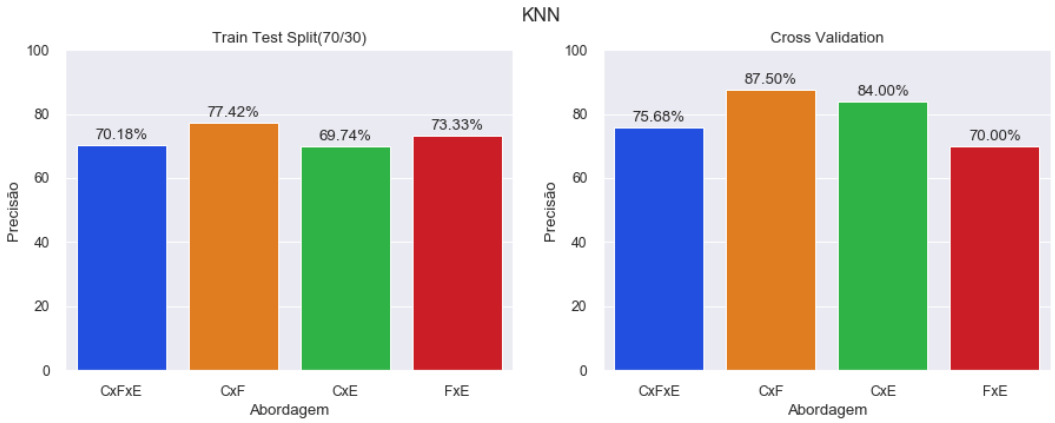

Figura 13: KNN Premier League

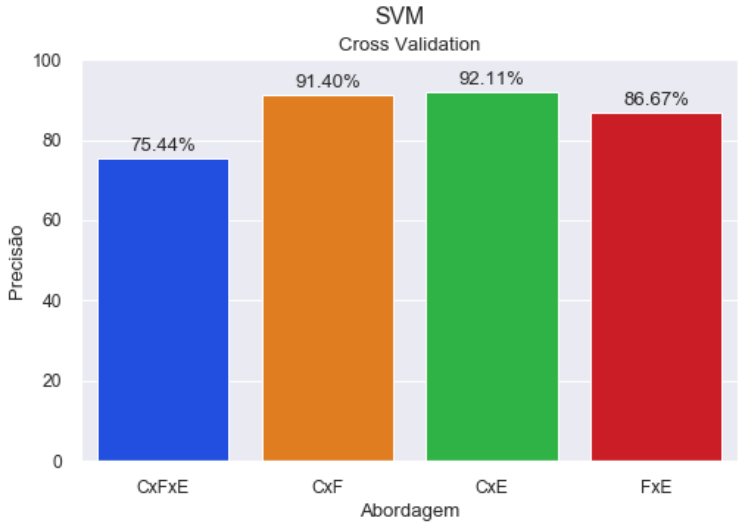

Figura 14: SVM Premier League

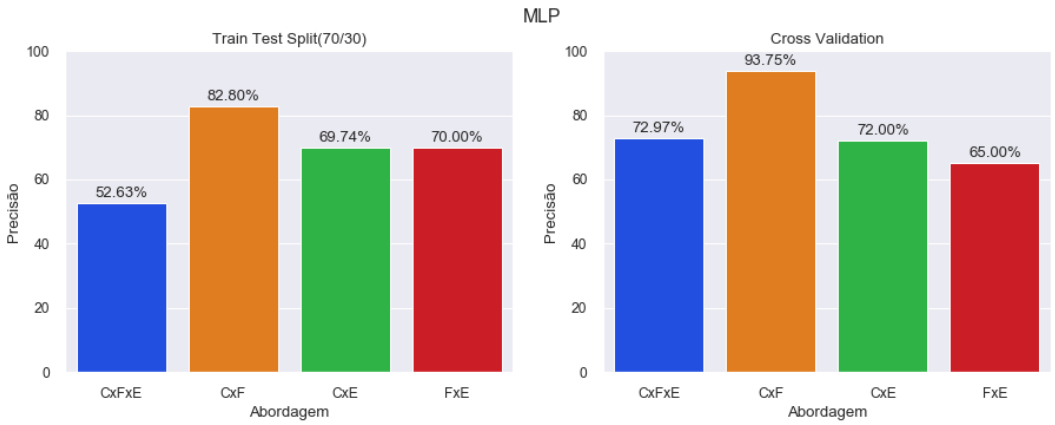

Figura 15: MLP Premier League

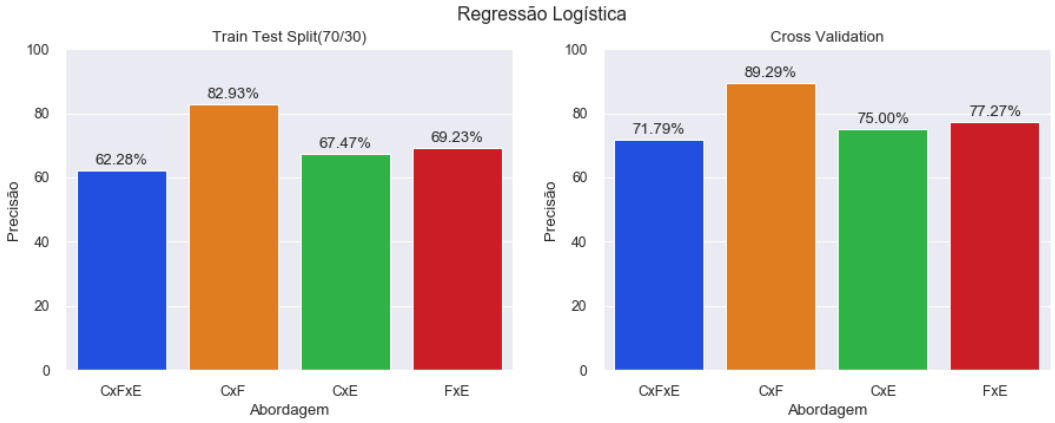

Figura 16: Regressão logística Serie A

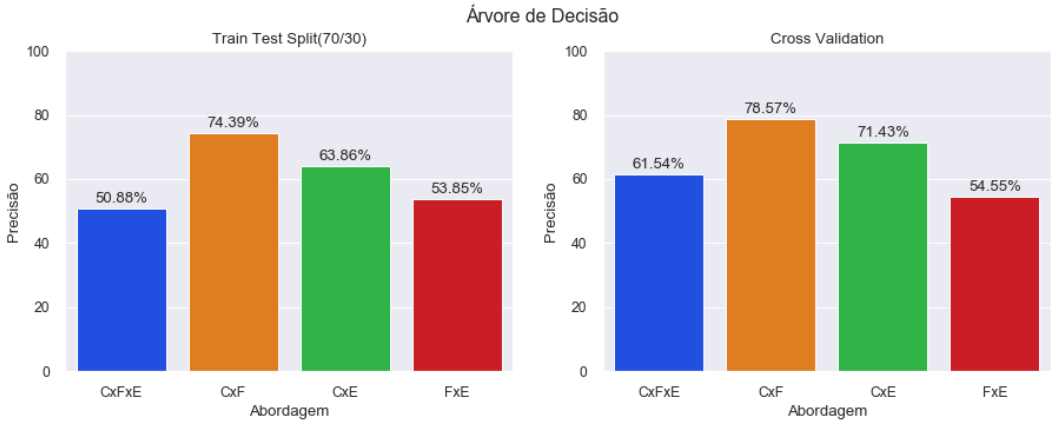

Figura 17: Árvore de decisão Serie A

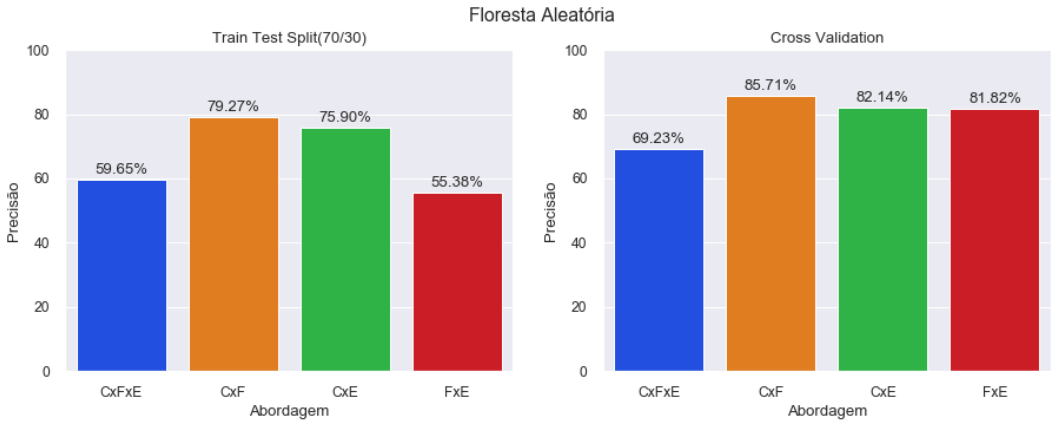

Figura 18: Floresta aleatória Serie A

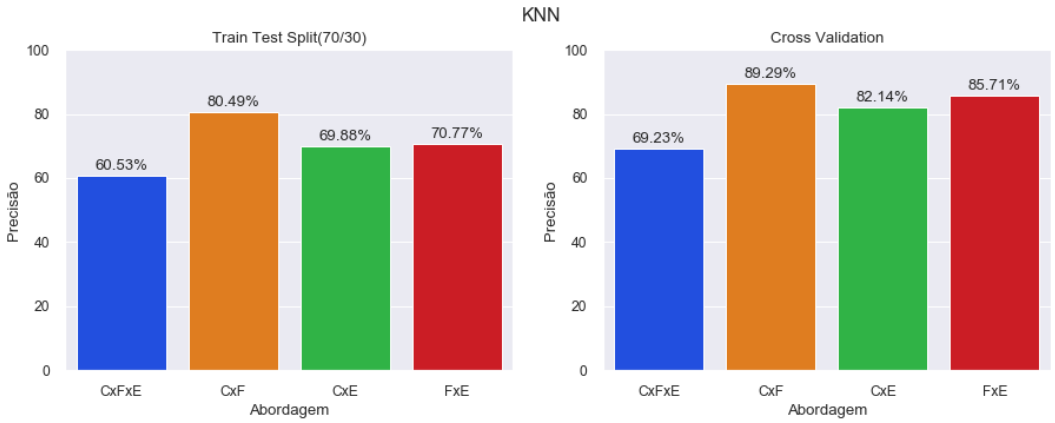

Figura 19: KNN Serie A

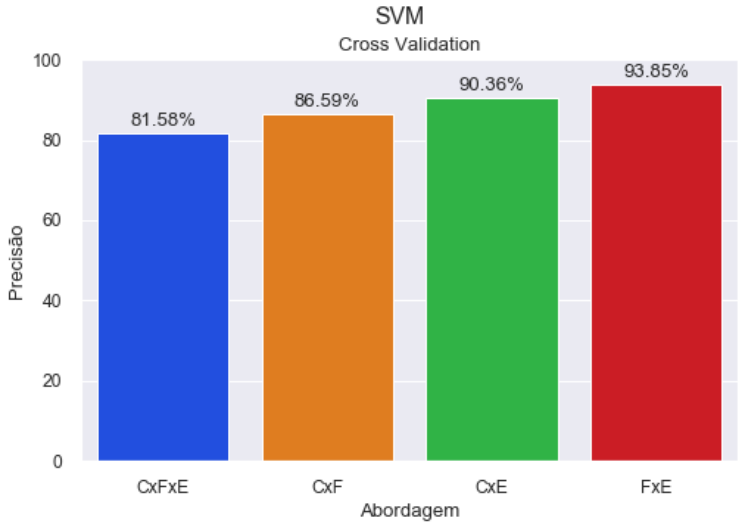

Figura 20: SVM Serie A

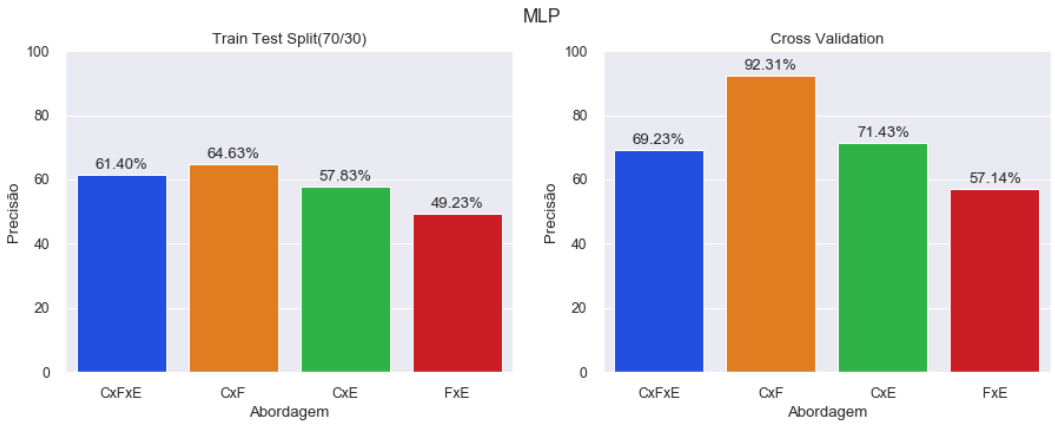

Figura 21: MLP Serie A

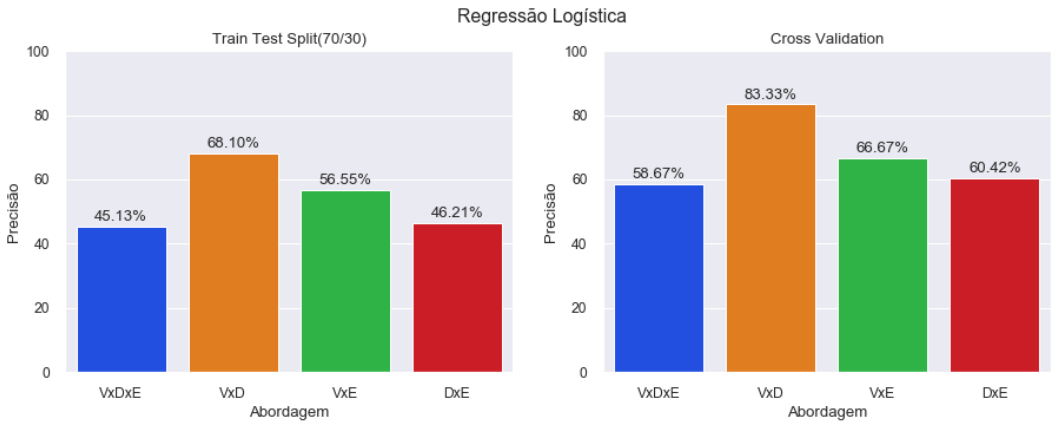

Figura 22: Regressão logística Brasileirão

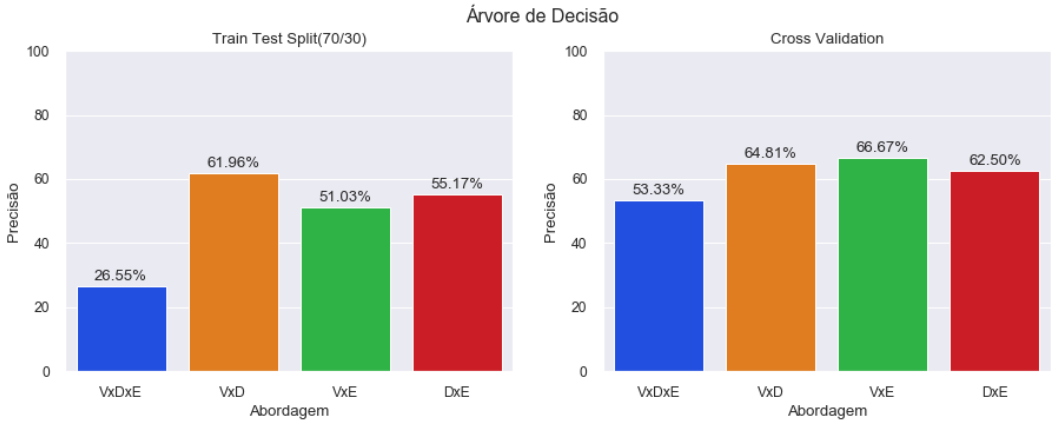

Figura 23: Árvore de decisão Brasileirão

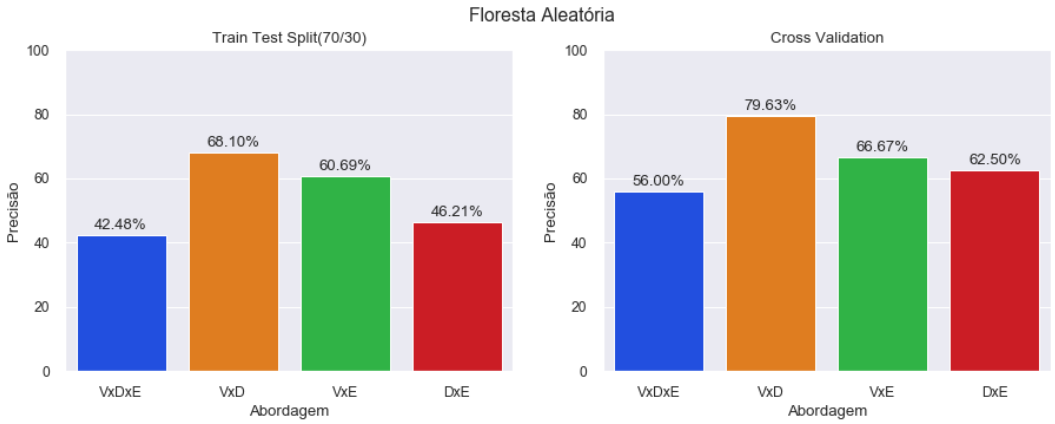

Figura 24: Floresta aleatória Brasileirão

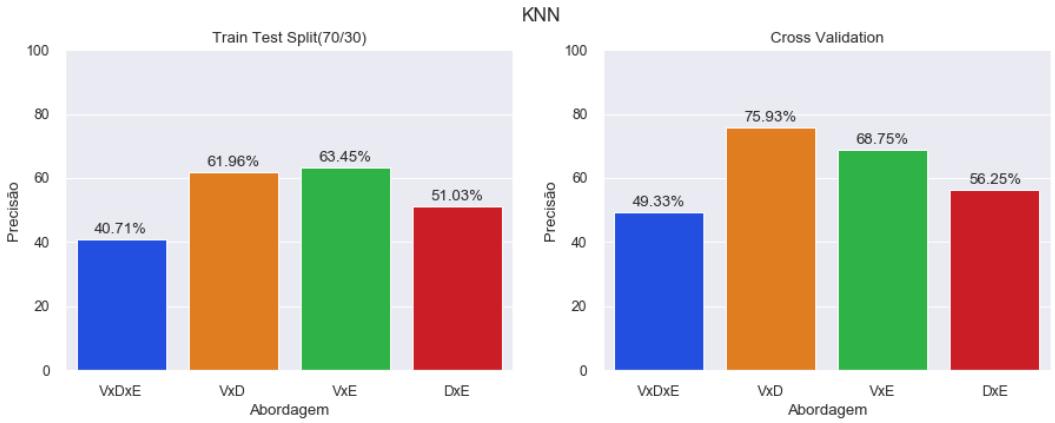

Figura 25: KNN Brasileirão

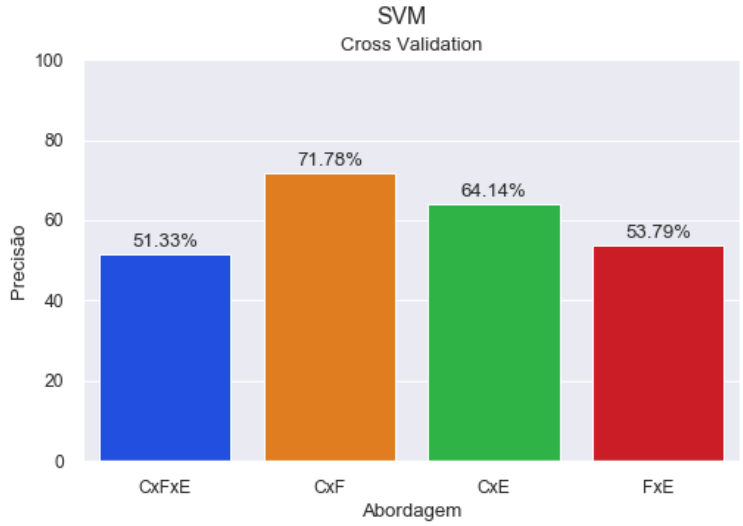

Figura 26: SVM Brasileirão

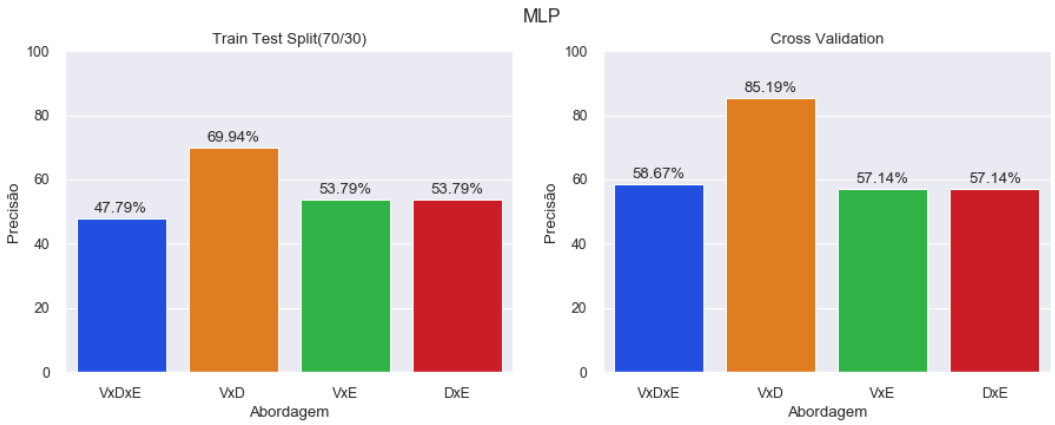

Figura 27: MLP Brasileirão

3 CONCLUSÃO

Conclui-se que em questão de dados para os modelos é mais difícil prever o resultado de Casa x Fora x Empate e Vitória x Empate x Derrota. Os modelos performam melhor quando o alvo da previsão é binário (2 classes). Dentre as bases com 2 classes Fora x Empate e Derrota x Empate foram as que tiveram menor performance e Casa x Fora e Vitória x Derrota foram as que tiveram melhor performance. Podemos concluir então que os modelos performam melhor em prever resultados sem empate.

Levando em consideração os métodos de divisão das bases de dados concluimos que o método de validação cruzada gera uma performance melhor do que o método de divisão 70% treino 30% teste.

A tabela abaixa mostra quais classificadores obtiveram uma melhor performance em cada método de divisão e em cada abordagem.

70-30			
	Premier League	**Serie A**	**Brasileirão**
CxFxE	Regressão Logística	Regressão Logística	MLP
CxF	Regressão Logística	Regressão Logística	MLP
CxE	Floresta Aleatória	Floresta Aleatória	KNN
FxE	Regressão Logística	KNN	Árvore de Decisão
Validação Cruzada			
CxFxE	KNN	SVM	Regressão Logística
CxF	MLP	MLP	MLP
CxE	SVM	SVM	KNN
FxE	SVM	SVM	Árvore de Decisão/Floresta Aleatória

Tabela 1: Classificadores com melhor performance em cada método de divisão e abordagem.

Podemos concluir que os algoritmos que obtiveram a melhor performance mais vezes foram nesta ordem: regressão logística, SVM e MLP, KNN, floresta aleatória e árvore de decisão.

Todos os algoritmos abordados neste trabalho foram implementados em Python e encontram-se dísponíveis em um repositório público do GitHub.

Referências

AMARAL JOSE MARIA CELESTINO DE LIMA, L. E. S. d. O. Cesar do. *Rede Neural Supervisionada Multicamadas - Árvore de decisão plataforma WEKA*. 2014. Disponível em: ⟨https://pt.slideshare.net/cesardoamatal/a-rede-neural-supervisionada-chamada-perceptron-multicamadas⟩. Acesso em: 26 nov 2019.

BOSER, B. E.; GUYON, I. M.; VAPNIK, V. N. A training algorithm for optimal margin classifiers. In: *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*. New York, NY, USA: ACM, 1992. (COLT '92), p. 144–152. ISBN 0-89791-497-X. Disponível em: ⟨http://doi.acm.org/10.1145/130385.130401⟩.

BREIMAN, L. *Machine Learning*, Springer Nature, v. 45, n. 1, p. 5–32, 2001.

BROOKS, J.; KERR, M.; GUTTAG, J. Using machine learning to draw inferences from pass location data in soccer. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, Wiley, v. 9, n. 5, p. 338–349, jun 2016.

CABRAL, C. I. S. *Aplicação do Modelo de Regressão Logística num Estudo de Mercado*. Dissertação (Mestrado) — Universidade de Lisboa, 2013. Disponível em: ⟨https://repositorio.ul.pt/bitstream/10451/10671/1/ulfc106455_tm_Cleidy_Cabral.pdf⟩.

CARACIOLO, M. P. *Introdução a Árvores de decisão para classificação e mineração de dados*. 2009. Disponível em: ⟨http://aimotion.blogspot.com/2009/04/artigo-introducao-arvores-de-decisao.html⟩. Acesso em: 24 nov 2019.

CONSTANTINOU, A. C.; FENTON, N. E.; NEIL, M. Profiting from an inefficient association football gambling market: Prediction, risk and uncertainty using bayesian networks. *Knowledge-Based Systems*, Elsevier BV, v. 50, p. 60–86, sep 2013.

DMITRIEVSKY, M. *Floresta de decisão aleatória na aprendizagem por reforço*. 2018. Disponível em: ⟨https://www.mql5.com/pt/articles/3856⟩. Acesso em: 24 nov 2019.

DUARTE, L. M. da S.; SOARES, C.; TEIXEIRA, J. *Previsão de resultados de jogos de futebol*. Dissertação (Mestrado) — Faculdade da Engenharia da Universidade do Porto, 2015.

FIFA. *More than half the world watched record-breaking 2018 World Cup*. 2018. Disponível em: ⟨https://www.fifa.com/worldcup/news/more-than-half-the-world-watched-record-breaking-2018-world-cup⟩. Acesso em: 15 jun 2019.

FIFA. *Financial Report 2018*. 2019. Disponível em: ⟨https://resources.fifa.com/image/upload/xzshsoe2ayttyquuxhq0.pdf⟩. Acesso em: 15 jun 2019.

HUCALJUK, J.; RAKIPOVIć, A. Predicting football scores using machine learning techniques. *2011 Proceedings of the 34th International Convention MIPRO*, may 2011.

IGIRI, C. P. Support vector machine–based prediction system for a football match result. *IOSR Journal of Computer Engineering*, v. 17, n. 3, p. 21–26, jun 2015.

LEARN scikit. *sklearn.preprocessing.StandardScaler*. 2019. Disponível em: ⟨https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html⟩. Acesso em: 24 nov 2019.

MARTINS, R. G. et al. Exploring polynomial classifier to predict match results in football championships. *Expert Systems with Applications*, Elsevier BV, v. 83, p. 79–93, oct 2017.

OWRAMIPUR, F.; ESKANDARIAN, P.; MOZNEB, F. S. Football result prediction with bayesian network in spanish league-barcelona team. *International Journal of Computer Theory and Engineering*, IACSIT Press, p. 812–815, 2013.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011.

PENDHARKAR, P. C.; KHOSROWPOUR, M.; RODGER, J. A. Application of bayesian network classifiers and data envelopment analysis for mining breast cancer patterns. *Journal of Computer Information Systems*, v. 40, n. 4, p. 127–132, 2000.

PERIN, C.; VUILLEMOT, R.; FEKETE, J.-D. SoccerStories: A kick-off for visual soccer analysis. *IEEE Transactions on Visualization and Computer Graphics*, Institute of Eletrical and Electronics Engineers (IEEE), v. 19, n. 12, p. 2506–2515, dec 2013.

PYPI. *Pandas*. 2019. Disponível em: ⟨https://pypi.org/project/pandas/⟩. Acesso em: 19 nov 2019.

SFEIR, M. N. Laws of the game (adapted from fifa 2010-11). *World Literature Today*, v. 85, n. 3, p. 38–39, may 2011.

SOUZA, A. *Validação Cruzada: Conceito e Exemplo em R*. 2019. Disponível em: ⟨https://pessoalex.wordpress.com/2019/04/16/validacao-cruzada-conceito-e-exemplo-em-r/⟩. Acesso em: 24 nov 2019.

TAX, N.; JOUSTRA, Y. Predicting the dutch football competition using public data: A machine learning approach. Unpublished, 2015.

ULMER, B.; FERNANDEZ, M. *Predicting Soccer Match Results in the English Premier League*. Tese (Doutorado) — Stanford University, 2013.

ZANINI, A. *Regressão Logística e Redes Neurais Artificiais: Um Problema de Estrutura de Preferência do Consumidor e Classificação de Perfis de Consumo*. Dissertação (Mestrado) — Faculdade de Economia e Administração da Universidade Federal de Juiz de Fora, 2007. Disponível em: ⟨http://www.ufjf.br/poseconomia/files/2010/01/td_007_20071.pdf⟩.