Ministério da Educação
Universidade Tecnológica Federal do Paraná
Diretoria de Graduação e Educação Profissional

Preditor de desempenho de um time da Premier League
Sistemas Inteligentes

Vinicius Bertuol
Engenharia de Computação
Prof. Daniel Cavalcanti Jeronymo

CAMPUS TOLEDO, 2023

Sumário

1. Problema

Para o seminário final da matéria de Sistemas Inteligentes, resolvi criar um preditor de desempenho de um time do campeonato inglês de futebol: Premier League. A predição é feita baseando-se nas estatísticas atuais do clube e na base de dados disponíveis do campeonato.

A minha inspiração foi o meu trabalho realizado na Atividade 3 da matéria: “Árvore do Conhecimento do Bem e do Mal”, onde utilizei o método de árvore de decisão de aprendizado supervisionado.

Na atividade 3, o objetivo era o algoritmo receber um filme e suas informações relevantes e responder se aquele filme pode ser considerado um filme “bom” ou “ruim” baseado no banco de dados de filmes assistidos por mim.

O banco de dados possuía na Atividade 3 um campo dedicado a receber minhas notas pessoais para cada filme assistido, assim um novo campo binário era criado para classificar os filmes como “bom” ou “ruim” a partir de uma nota arbitrária usada como fronteira de decisão.

No algoritmo criado para a Premier League o objetivo é ter como resultado final um intervalo que representa as possíveis posições finais de um time baseando-se nas principais estatísticas do clube em determinada rodada do campeonato.

2. Abstração

2.1 Base de Dados

O primeiro passo do trabalho é achar um banco de dados que satisfaça as necessidades do projeto.

O banco de dados mais adequado encontrado veio do site “Transfermarkt”, que possui informações dos principais campeonatos de futebol nos últimos anos.

Uma única tabela do banco de dados foi utilizada, a tabela possui todos os jogos disputados em determinados campeonatos há alguns anos. Meu primeiro filtro foi deixar apenas os jogos da Premier League. Os dados da Premier League na tabela começam em 2012 e vão até os dias atuais, porém o ano de 2023 não é considerado, pois o campeonato ainda não terminou. Todavia, as informações presentes na tabela não eram o que eu precisava diretamente.

As informações que eu precisava, eram as estatísticas de cada time a cada rodada ano a ano relacionadas a posição final do time no campeonato daquele ano. Então, através de códigos Python e do pacote pandas consegui montar uma nova tabela que possui exatamente os dados necessários: ano, rodada, pontos, vitórias, empates, derrotas, gols marcados (gm), gols sofridos(gc), saldo de gols(sg), posição final no campeonato, posição na rodada e aproveitamento de pontos.

O campo timeId da nova tabela criada era utilizado para relacionar as estatísticas da tabela original.

2.2 Abstração Computacional

2.2.1 Problema

Desenvolver um sistema preditivo para estimar as possíveis posições finais de um time na Premier League, com base em estatísticas atuais do clube e dados históricos do campeonato.

2.2.2 Solução computacional

2.2.2.1 Entrada de Dados

1)Estatísticas atuais do clube, incluindo número de vitórias, empates, derrotas, gols marcados, gols sofridos, posição na tabela, entre outros.

2)Dados históricos do campeonato, como resultados anteriores, pontos ganhos e posição final.

2.2.2.2 Inspiração na Atividade 3 - Árvore de Decisão

Utilização do método de árvore de decisão, semelhante ao aplicado na Atividade 3, onde o algoritmo decidia se um filme era "bom" ou "ruim" com base nas notas pessoais.

2.2.2.3 Banco de Dados de Filmes vs. Dados da Premier League

Analogia entre o campo de notas pessoais dos filmes e as estatísticas atuais do clube na Premier League.

Substituição da classificação "bom" ou "ruim" por uma estimativa de posições finais na tabela da Premier League.

2.2.2.4 Fronteira de Decisão na Atividade 3 vs. Intervalo de Posições na Premier League

Na Atividade 3, uma fronteira de decisão era definida por uma nota arbitrária para classificar os filmes.

No contexto da Premier League, a fronteira de decisão é representada por uma faixa ou intervalo de posições possíveis.

2.2.2.5 Desenvolvimento do Algoritmo

Utilização de uma árvore de decisão para modelar as relações entre as estatísticas do clube e a posição final na tabela.

Treinamento do modelo com base nos dados históricos disponíveis da Premier League.

2.2.2.6 Saída do Sistema

Intervalo de posições finais estimadas para o time na tabela da Premier League.

3. Conceituação

3.1 Árvore de decisão

3.1.1 Definição Geral

Uma árvore de decisão é um modelo de representação gráfica e estruturada de decisões e suas possíveis consequências. Ela é construída em forma de uma árvore invertida, com raiz no topo representando a decisão inicial e os ramos se ramificando para baixo representando escolhas e resultados possíveis.

3.1.2 Contexto Computacional

Em aprendizado de máquina e ciência de dados, uma árvore de decisão é um modelo preditivo que toma decisões com base em condições e resultados observados nos dados de treinamento.

A estrutura de árvore representa um conjunto de regras de decisão, onde cada nó interno representa uma condição sobre um atributo, cada ramo representa o resultado dessa condição, e cada folha representa a decisão final ou a saída prevista.

3.1.3 Componentes Básicos

Nó Raiz: Representa a decisão inicial a ser tomada com base em um atributo.

Nós Internos: Representam condições que levam a diferentes caminhos da árvore com base em valores de atributos.

Ramos: Conexões entre nós que representam os diferentes resultados possíveis das condições.

Folhas: Representam as decisões finais ou as saídas previstas.

3.2 sklearn

O scikit-learn, frequentemente abreviado como sklearn, é uma biblioteca em Python projetada para oferecer ferramentas eficientes e simples para análise de dados e modelagem estatística, incluindo aprendizado de máquina (machine learning). Ela é construída sobre outras bibliotecas científicas populares, como NumPy, SciPy e Matplotlib, e é amplamente utilizada na comunidade de ciência de dados.

4. Proposta de solução

O código utiliza a biblioteca scikit-learn para criar um modelo de árvore de decisão e prever a posição final de um time na tabela da Premier League, com base em estatísticas atuais e históricas.

4.1 Importação de Bibliotecas

pandas para manipulação de dados.

numpy para operações numéricas.

Simple Imputer da scikit-learn para tratar valores ausentes.

tree da scikit-learn para criar a árvore de decisão.

os para manipulação do ambiente.

4.2 Leitura do Conjunto de Dados

O conjunto de dados da Premier League é lido e armazenado em um Data Frame e definição das estatísticas atuais do time em variáveis

4.3 Laço de Decisão

Dois laços serão iterados, um representando o limite inferior e o outro representando o limite superior.

Um loop for é utilizado para iterar sobre posições possíveis na tabela (de 1 a 20).

Para cada iteração, uma nova coluna chamada position é criada no DataFrame, indicando se a posição é menor ou igual ao valor atual da iteração.

Um modelo de árvore de decisão é treinado usando os dados da coluna position como rótulo (y) e as estatísticas como características (X).

Se o modelo prediz que o time está na posição desejada, a posição é armazenada na lista de posições e o loop é encerrado.

4.4 Laço Reverso

Um segundo loop for é utilizado para ajustar finamente a posição, começando do final da tabela.

Similar ao primeiro loop, um novo modelo de árvore de decisão é treinado.

Se o modelo prediz que o time não está na posição desejada, a posição é ajustada e adicionada à lista de posições.

4.5 Impressão das Posições Estimadas

A lista de posições é impressa, indicando as posições finais estimadas para o time.

4.6 Solução do Problema

O código utiliza um modelo de árvore de decisão para prever a posição final do time com base nas estatísticas disponíveis.

A iteração sobre diferentes posições permite encontrar um intervalo de possibilidades.

A impressão das posições estimadas fornece insights sobre a possível colocação do time na tabela da Premier League, considerando as estatísticas atuais e históricas.

5. Resultados

Os resultados obtidos podem variar desde intervalos grandes de até oito posições possíveis ou casos em que o intervalo aponta apenas uma posição.

Para testar o algoritmo utilizei a tabela do campeonato de 2011, pois ela é a primeira abaixo dos campeonatos utilizados no treinamento e possui os resultados finais, diferente de 2023 por exemplo, que ainda não acabou.

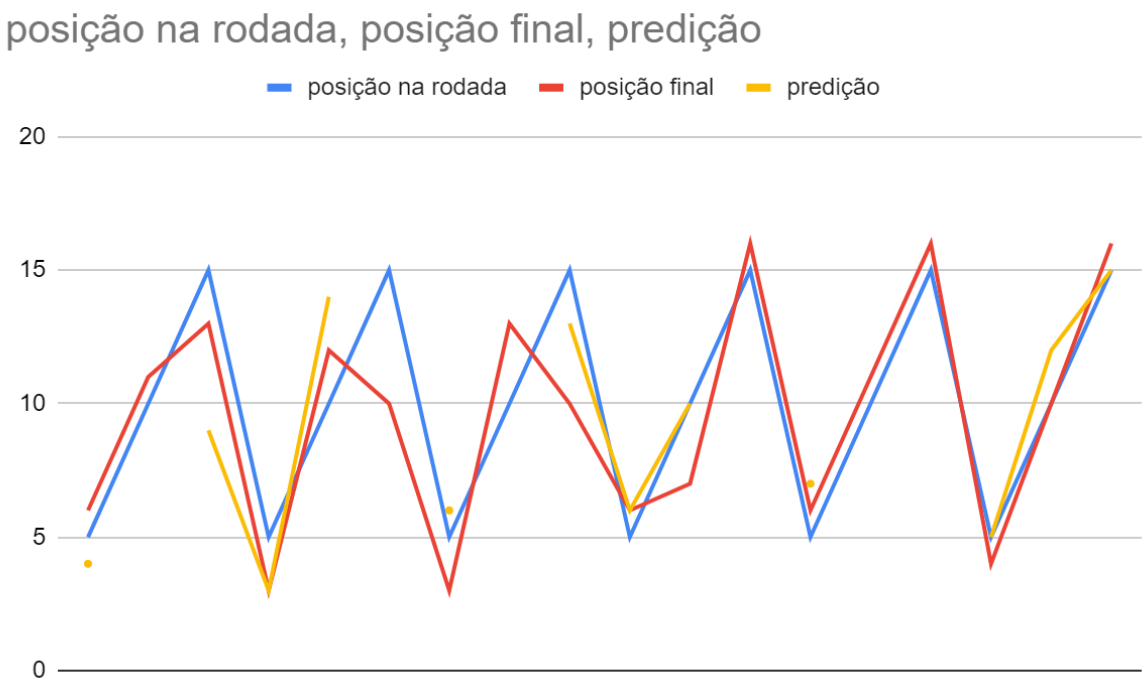
Um fator importante para a qualidade da previsão é o quão adiantado o campeonato está. Quanto menos rodadas restantes maior a probabilidade de acertar a previsão pois não há espaço para grandes mudanças na tabela do campeonato.

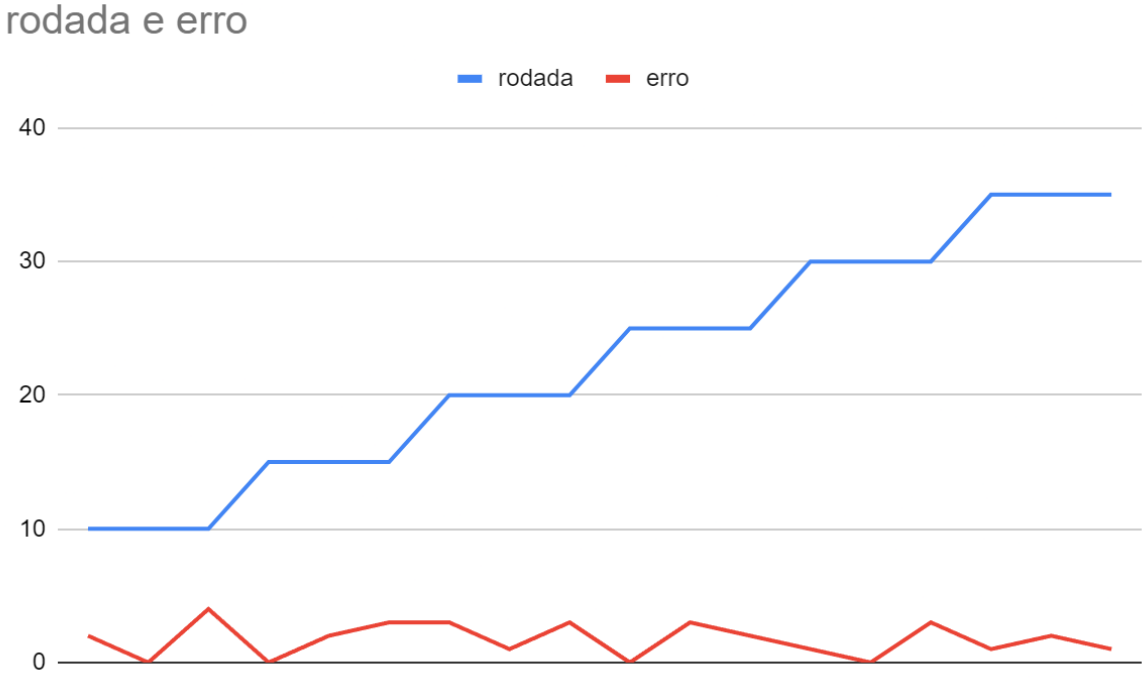
Para variar os testes utilizei 3 casos diferentes para 6 momentos diferentes do campeonato. Os casos são os times que ocupam as posições 5, 10 e 15 na tabela nas rodadas 10, 15, 20, 25, 30 e 35, considerando que ao todo são 20 times e 38 rodadas.

5.1 Tabela de Resultados

rodada	posição na rodada	posição final	predição	erro
10	5	6	4	2
10	10	11	11-14	0
10	15	13	9	4
15	5	3	3	0
15	10	12	14	2
15	15	10	13-20	3
20	5	3	6	3
20	10	13	6-12	1
20	15	10	13	3
25	5	6	6	0
25	10	7	10	3
25	15	16	12-14	2
30	5	6	7	1
30	10	11	10-13	0
30	15	16	10-13	3
35	5	4	5	1
35	10	10	12	2
35	15	16	15	1

5.2 Gráficos de resultados

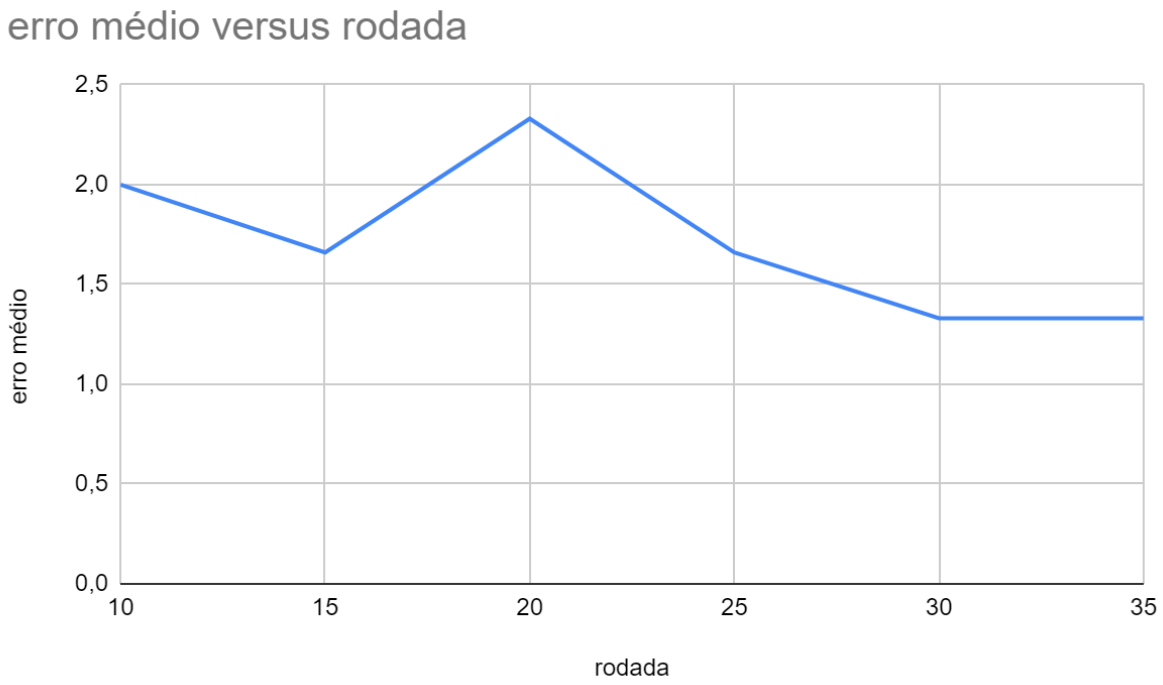




5.3 Tabela de Erros Médios por rodada testada

rodada	erro médio
10	2
15	1,66
20	2,33
25	1,66
30	1,33
35	1,33

5.4 Tabela de Erros Médios por rodada testada



5.5 Resultados finais

O erro médio total e final encontrado após os 18 testes foi de: 1,722. Considerando que o total de possibilidades de posição final de um time são 20 posições, 1,722 representa um erro de 8,61%, ou seja, conseguimos um percentual de acerto de 91,39%.

5.6 Possíveis melhorias

Existem duas melhorias que poderiam melhorar bastante o desempenho da predição do algoritmo:

5.6.1 Mais Dados

A base de dados disponível e utilizada contempla as temporadas de 2012 a 2022, porém o campeonato inglês é disputado no formato atual desde 1991, então poderiam ser adicionadas mais 21 temporadas ao banco de dados para melhorar o aprendizado.

5.6.2 Times específicos

O algoritmo atual leva em consideração um time genérico na hora de predizer o resultado. Uma possível melhoria é levar em consideração o retrospecto individual do time que irá ser analisado.

5.6.3 Considerar todos os times

Por fim, a principal melhoria do preditor seria poder passar a situação atual de todos os times na rodada escolhida para que o algoritmo possa levar em consideração todas as estatísticas da tabela.