Classification model performance analysis for League of Legends 2022 competitive team matches using KNN and Decision Tree algorithms

Heitor F. Ferreira[1], **Guilherme A. Carvalho**[1]

[1]Faculdade de Computação - Universidade Federal de Uberlândia (UFU)

{heitor.ff,guilherme.092011}@ufu.br,

Abstract. *The following work aims to study and analyse classical data mining techniques to predict the result of League of Legends matches. In order to achieve this goal, classical algorithms like KNN and Decision Tree algorithms were used, and the results were compared using post processing metrics. It was possible to conclude that the techniques achieves the expected objectives, effectively classifying the match according to the result.*

Resumo. *Este trabalho propõe um estudo e uma análise de técnicas clássicas de mineração de dados, tentando prever o resultado de partidas de League of Legends. Para isto, foram usados algoritmos clássicos como o KNN e Árvore de Decisão, os resultados obtidos com os modelos foram comparados usando métricas de pós-processamento. É possível concluir que as técnicas atingem os objetivos esperados, classificando corretamente as partidas conforme seu resultado.*

1. Introdução

O League of Legends (LOL) é um jogo eletrônico extremamente popular que cativou milhões de jogadores ao redor do mundo. Desenvolvido pela Riot Games e lançado em 2009, o League of Legends rapidamente se tornou um fenômeno global, revolucionando o gênero de jogos online multiplayer de arena de batalha (MOBA, na sigla em inglês).

No jogo, jogadores assumem o papel de ”invocadores”, que controlam personagens fictícios em confrontos estratégicos. O objetivo principal é destruir a base inimiga enquanto defende a própria base dos ataques inimigos.

Os jogadores podem escolher entre mais de 160 personagens, cada um com habilidades únicas e estilos de jogo distintos. Os jogadores são divididos em cinco classes posições principais: topo, caçador, meio, carregador de dano, e suporte. Cada jogador possui um papel específico dentro do jogo, eles devem escolher campeões que se complementem para formar uma equipe equilibrada e capaz de vencer a partida.

Durante a partida, os jogadores acumulam ouro e experiência ao derrotar inimigos e monstros. O ouro pode ser utilizado para comprar itens que melhoram as habilidades do campeão, enquanto a experiência permite que o campeão evolua e desbloqueie novas habilidades. Os jogadores também podem utilizar runas e talentos para personalizar ainda mais o campeão, permitindo que ele se adapte a diferentes situações e estratégias.

O jogo se destaca pela sua cena competitiva, com torneios profissionais. As ligas regionais e os campeonatos mundiais atraem milhões de espectadores online e presenciais,

transformando jogadores habilidosos em ídolos e celebridades no mundo dos esportes eletrônicos.

Na área de ciência de dados, a classificação envolve o desenvolvimento de modelos preditivos que podem aprender a distinguir e categorizar diferentes classes de informações com base em um conjunto de dados de treinamento. Esse processo geralmente envolve a um tratamento prévio dos dados, seguida de uma fase de treinamento, onde algoritmos de aprendizado de máquina são aplicados para criar um modelo classificador. Uma vez treinado e validados segundo algumas métricas, esse modelo pode ser usado para fazer previsões em novos conjuntos de dados e classificar novas instâncias com base nas informações aprendidas durante o treinamento.

Dessa forma, modelos de classificação de dados podem ser usados em bases de dados de partidas esportivas para prever resultado com base em informações disponíveis. Utilizando dados de partidas anteriores, é possível treinar um modelo classificador capaz de identificar padrões e tendências que levam a certos resultados [Bahrololloomi et al. 2022]. Isso pode fornecer informações valiosas para times profissionais e amadores, ajudando-os a tomar decisões informadas e estratégicas antes das partidas.

2. Descrição do problema proposto

O problema em questão é encontrar maneiras de prever os resultados de partidas de League of Legends em vitória ou derrota, utilizando os dados dos times de cada partida e usando técnicas clássicas como KNN e Árvores de Decisão. A classificação é feita com base nos dados dos times de cada partida, e o resultado da partida é definido como vitória ou derrota.

3. Sobre o jogo

O League of Legends (LoL) é um jogo eletrônico multiplayer online de arena de batalha (MOBA) que se passa em um universo fictício chamado de Summoner's Rift. No jogo, duas equipes de cinco jogadores competem entre si, assumindo o papel de campeões, com o objetivo de destruir a base inimiga enquanto defendem a sua própria base.

O jogo pode ser definido a partir dos seguinte pontos centrais, os quais determinam de forma direta se um time irá ganhar ou não uma partida:

- Seleção de Campeões: Antes do início da partida, cada jogador escolhe um campeão para controlar. Existem mais de 140 campeões únicos disponíveis, cada um com habilidades e papéis diferentes, como assassinos, magos, tanques, atiradores e suportes. A seleção de campeões é estratégica e visa criar uma equipe equilibrada com diferentes habilidades.
- Mapa e Objetivos: O jogo se passa em um mapa simétrico chamado Summoner's Rift, que é dividido em três rotas principais: Top (superior), Mid (meio) e Bot (inferior). Cada rota é protegida por torres defensivas e leva à base inimiga. O objetivo principal é destruir o Nexus inimigo, uma estrutura localizada na base adversária.
- Habilidades e Itens: Cada campeão tem habilidades únicas que podem ser aprimoradas ao subir de nível durante a partida. Além disso, os jogadores podem

comprar itens com o ouro acumulado para fortalecer seus campeões. Os itens oferecem diferentes benefícios, como aumento de dano, resistência, velocidade de ataque, entre outros, e são cruciais para o desempenho dos campeões durante o jogo.

- Comunicação e Estratégia: A comunicação e a estratégia em equipe são essenciais no League of Legends. Os jogadores se comunicam através de chat de texto ou pings para coordenar movimentos, sinalizar objetivos ou alertar sobre ações inimigas. É importante planejar táticas, sincronizar ataques, proteger rotas e trabalhar em equipe para obter a vitória.
- Atualizações e Competições: O League of Legends é constantemente atualizado com novos campeões, ajustes de balanceamento e eventos especiais. Além do jogo casual, existem competições profissionais de alto nível, como o Campeonato Mundial de League of Legends, em que as melhores equipes de todo o mundo se enfrentam para conquistar o título.

Além dessas características, a partida do jogo também pode ser dividida em diferentes fases:

1. Fase de Rotação (Laning Phase): Nessa fase, os jogadores se dividem nas rotas e enfrentam os adversários correspondentes, tentando acumular ouro e experiência ao abater minions (soldados controlados pelo computador) e adversários. O objetivo é dominar as rotas, derrubar torres inimigas e ganhar vantagem para a próxima fase.
2. Fase de Meio de Jogo (Mid Game): Nessa fase, as equipes começam a se agrupar e se movimentar pelo mapa para lutar em equipe, buscar objetivos globais, como dragões e Heralds, e controlar a visão do mapa através de sentinelas. O foco é conquistar vantagens e preparar-se para as batalhas decisivas.
3. Fase de Final de Jogo (Late Game): Nessa fase, as equipes estão mais fortes e começam a se confrontar em batalhas em equipe mais intensas. O objetivo é ganhar lutas decisivas para avançar pelo mapa, destruir torres, inibidores (estruturas que fortalecem os minions) e, finalmente, o Nexus inimigo.

Todas essas características que compõem o jogo acabam determinando se um time irá vencer ou não a partida.

4. Desenvolvimento

4.1. Base de dados

A base de dados utilizada utilizada é de autoria de Tim Sevenhuysen da comunidade OraclesElixir.com e detém como amostras resultados de vitória e derrota, de jogadores e times, em partidas competitivas de League of Legends em diferentes ligas mundiais no ano de 2022.

O conjunto de dados possui 123 atributos e 149232 amostras, sendo que os atributos disponíveis possuem informações em relação ao player e também em relação ao time.

Há também estatísticas de jogo até quinze minutos de duração do mesmo, porém o foco do pré-processamento será classificar partidas de times de 2022 em vitória ou não, dependendo das variáveis dependentes selecionadas no pré-processamento.

4.2. Pré-processamento e técnicas usadas

O processamento de dados é o conjunto de técnicas e métodos utilizados para manipular e transformar informações brutas em dados significativos e úteis. Essas informações então são utilizadas como entrada no modelo de mineração de dados, o qual será capaz de extrair informações relevantes dos dados.

Dentre as técnicas utilizadas no pré-processamento da base de dados selecionada, podemos citar a seleção dos amostras com base nas seguintes colunas:

- Atributo 'position' igual a 'team': como na base de dados há dados misturados com informações de jogadores e de times, foi necessário selecionar as amostras com apenas times.
- Atributo 'datacompleteness' igual a 'complete': a existência dessa coluna emprega que haviam amostras com informações faltosas, dessa forma os mesmos foram removidos após a seleção utilizando esse critério.
- Atributo 'league' dentro das ligas profissionais do jogo: aqui foram selecionadas partidas de equipes de ligas profissionais, pois se torna mais confiável modelar utilizando dados de jogadores profissionais.
- Atributo 'year' igual a 2022: Apesar da base de dados utilizada dizer apresentar apenas informações de partidas do ano de 2022, foram encontradas outras amostras em diferentes anos, dessa forma foi necessário remover essas para a análise proposta.

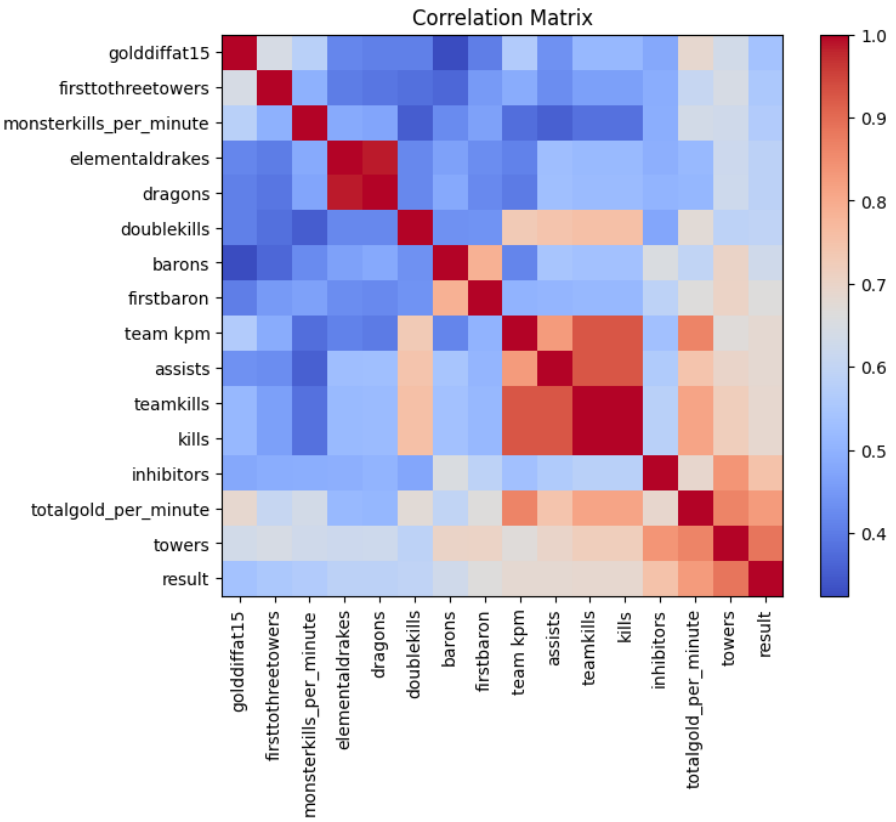

Figure 1. Matriz de correlação dos quinze elementos mais correlacionados com o atributo alvo 'result'

Após ocorrida a seleção de amostras, a redução de dimensionalidade é aplicada para remover colunas não interessantes, como atributos com informações de jogadores (após a seleção de amostras temos apenas times) e informações de identificação de partida, etc.

No final dos processos anteriores, parte da informação da base de dados original foi selecionada e os seguintes métodos são aplicados:

- Mapeamento das colunas 'totalgold', 'minionkills', 'monsterkills' e 'controlwardsbought' para um novo espaço, em dados por minuto, utilizando o atributo 'gamelenght' que se encontra em minutos
- Normalização dos atributos contínuos da base utilizando z-score, dado pela seguinte formula:

$$Z = \frac{x - \mu}{\sigma}$$

Finalmente, após uma visualização e compreensão melhor dos dados com as etapas de pré-processamento realizadas anteriormente, as quinze melhores correlações na Fig. 1 com o atributo alvo 'result' foram selecionadas como colunas de entrada para os modelos de classificação KNN e Árvore de Decisão.

4.3. Técnicas de mineiração usadas

As tarefas de mineração de dados referem-se às diferentes atividades e técnicas empregadas no processo de extração de informações relevantes. Essas tarefas têm como objetivo identificar padrões, relações e tendências nos dados, permitindo que estratégias mais eficazes sejam realizadas e uma compreensão melhor do comportamento dos dados seja atingida.

Entre as principais tarefas de mineração de dados estão:

- Análise de cluster: Agrupa objetos semelhantes em clusters, identificando a estrutura subjacente nos dados.
- Classificação: Atribui rótulos ou categorias a objetos com base em características e padrões previamente identificados.
- Regressão: Estima e modela a relação entre variáveis para prever valores numéricos futuros.
- Análise de associação: Descobre relações e correlações entre itens frequentemente co-ocorrentes em conjuntos de dados.
- Detecção de anomalias: Identifica padrões incomuns, desvios ou exceções nos dados que podem indicar eventos ou comportamentos anômalos.

Essas tarefas de mineração de dados são realizadas com o auxílio de algoritmos e técnicas estatísticas e de aprendizado de máquina, explorando as capacidades computacionais para lidar com grandes volumes de dados de forma eficiente.

Dessa maneira, para a resolução do problema proposto, as seguintes técnicas de mineração de dados são utilizadas, no escopo de classificação: o algoritmo KNN e Árvore de Decisão para estabelecer um comparativo entre essas duas técnicas.

O algoritmo K-Nearest Neighbors (KNN) é um método de aprendizado de máquina supervisionado utilizado para classificação e regressão. Ele é amplamente utilizado devido à sua simplicidade e eficácia em problemas de reconhecimento de padrões.

O KNN baseia-se na ideia de que objetos semelhantes tendem a estar próximos uns dos outros no espaço de características. Portanto, o algoritmo classifica um novo exemplo com base nos rótulos das instâncias vizinhas mais próximas a ele.

O seu funcionamento se dá na seguinte forma na resolução do problema:

- Preparação dos dados: Inicialmente, os dados de treinamento são preparados, consistindo em exemplos rotulados, onde cada exemplo possui um conjunto de características e uma classe associada. É importante normalizar ou padronizar as características para que todas tenham a mesma escala.
- Escolha do valor de K: O valor de K é um parâmetro importante no algoritmo KNN. Ele determina o número de vizinhos mais próximos que serão considerados para a classificação de um novo exemplo. A escolha adequada de K depende do conjunto de dados e pode ser determinada por meio de técnicas de validação cruzada ou outros métodos de seleção de hiperparâmetros.
- Medição da distância: O próximo passo é calcular a distância entre o exemplo a ser classificado e todos os exemplos de treinamento. A distância Euclidiana é frequentemente utilizada, mas outras medidas, como a distância de Manhattan ou a distância de Minkowski, também podem ser aplicadas, dependendo do problema e dos dados.
- Seleção dos vizinhos mais próximos: Com base nas distâncias calculadas, os K exemplos de treinamento mais próximos do exemplo a ser classificado são selecionados.
- Votação dos vizinhos: Para classificação, é realizada uma votação entre os vizinhos mais próximos. Cada vizinho contribui com um voto para sua classe correspondente. A classe mais frequente entre os vizinhos é escolhida como a classe do novo exemplo.
- Atribuição do rótulo: O exemplo a ser classificado é atribuído à classe prevista com base na votação dos vizinhos ou no valor regressão obtido.

O algoritmo KNN não requer uma etapa de treinamento explícita, pois a classificação é feita com base nas instâncias existentes no conjunto de treinamento. Isso torna o KNN um algoritmo ”preguiçoso” (lazy) em comparação com outros algoritmos de aprendizado de máquina que exigem um estágio de treinamento.

O KNN tem vantagens, como a simplicidade de implementação e a capacidade de lidar com dados não lineares e problemas complexos. No entanto, ele pode ser computacionalmente intensivo em conjuntos de dados grandes, pois requer o cálculo das distâncias para todos os exemplos de treinamento.

O algoritmo de árvore de decisão é um método de aprendizado de máquina supervisionado que é amplamente utilizado para tarefas de classificação e regressão. Ele é uma representação visual de decisões e suas possíveis consequências em forma de uma estrutura em forma de árvore.

Aqui está uma descrição detalhada do funcionamento do algoritmo de árvore de decisão:

- Preparação dos dados: Inicialmente, os dados de treinamento são preparados, consistindo em exemplos rotulados, onde cada exemplo possui um conjunto de características e uma classe ou valor associado. Esses dados são usados para construir a árvore de decisão.

- Seleção do atributo de divisão: O algoritmo começa escolhendo o atributo (característica) mais relevante para dividir os dados de treinamento. Existem várias métricas para avaliar a relevância dos atributos, sendo a mais comum o ganho de informação ou a redução da impureza.
- Divisão dos dados: Com base no atributo selecionado, os dados de treinamento são divididos em subconjuntos menores. Cada subconjunto representa um ramo da árvore e corresponde a um valor específico do atributo selecionado.
- Recursividade: Os passos 2 e 3 são repetidos para cada subconjunto, criando subárvores para cada ramo da árvore principal. Essa recursividade continua até que uma condição de parada seja atingida, como quando todos os exemplos pertencem à mesma classe ou quando não há mais atributos disponíveis para dividir os dados.
- Podagem (pruning): Após a construção completa da árvore de decisão, o processo de podagem pode ser aplicado para evitar o superajustamento (overfitting) aos dados de treinamento. A podagem remove os ramos da árvore que não contribuem significativamente para a melhoria da precisão da classificação em um conjunto de dados de validação separado.
- Classificação ou regressão: Para classificação, a árvore de decisão é usada para atribuir uma classe aos exemplos desconhecidos com base nas características fornecidas. Para regressão, a árvore é usada para prever um valor numérico para um novo exemplo.

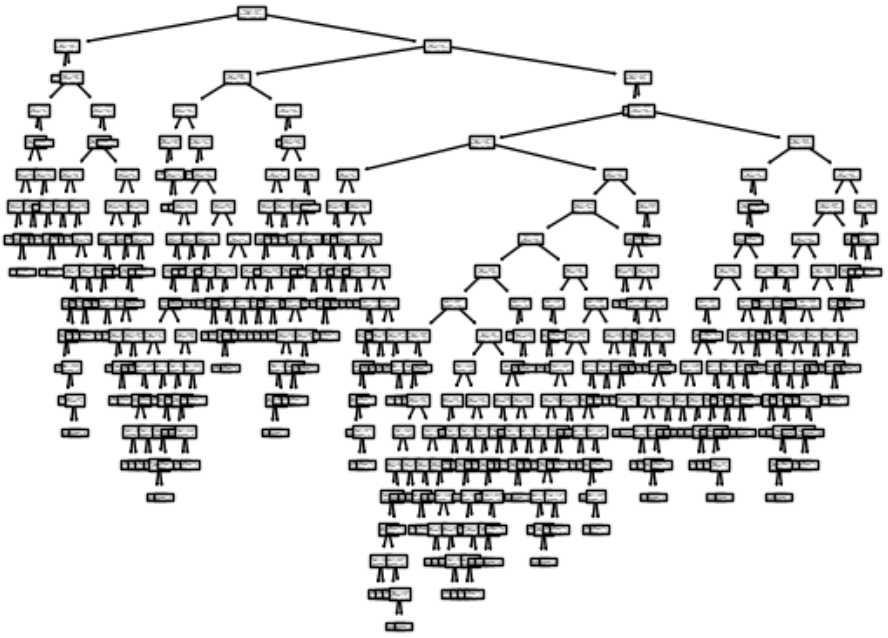

Figure 2. Árvore de Decisão gerada pela biblioteca scikit-learn usando os 15 atributos mais correlancionados

O algoritmo de árvore de decisão tem várias vantagens. Ele é fácil de entender e interpretar, permitindo visualizar e explicar as decisões tomadas pelo modelo. Além disso, as árvores de decisão podem lidar com dados numéricos e categóricos e são capazes de lidar com problemas de classificação e regressão.

No entanto, as árvores de decisão também podem ter algumas limitações. Elas podem ser sensíveis a pequenas variações nos dados de treinamento, resultando em árvores diferentes. Além disso, árvores de decisão complexas podem ser propensas ao superajustamento (overfitting) aos dados de treinamento, o que pode diminuir sua generalização para novos exemplos.

Dessa forma, o KNN em conjunto com a Árvore de Decisão são selecionados e implementados na linguagem na Python [Van Rossum and Drake Jr 1995], utilizando a biblioteca Pandas [pandas development team 2020] para manipulação da base de dados. Para o treinamento desses abordagens, foi utilizado o método Houldout com 90% dos dados para treinamento e 10% para testes.

Na implementação do algoritmo KNN, o mesmo é implementado manualmente sem a utilização de bibliotecas adicionais, realizando o algoritmo para diferentes K's e utilizando distancia de Manhattan, já a Árvore de Decisão utilizou a biblioteca Scikit-learn [Pedregosa et al. 2011]. Ambas técnicas tiveram como objetivo explorar o desempenho de algoritmos clássicos diante de um problema moderno, realizando a classificação de partidas competitivas utilizando os atributos selecionados no pré-processamento.

4.4. Pós-processamento e métricas usadas

No pós-processamento, técnicas como visualização por matriz de confusão e métricas de sensitividade e especificidade foram utilizadas, permitindo analisar os resultados do modelo KNN e Árvore de Decisão e comparar suas execuções.

A matriz de confusão é uma ferramenta que permite visualizar e calcular métricas, como sensibilidade (também chamada de taxa de verdadeiro positivo) e especificidade (também chamada de taxa de verdadeiro negativo).

A matriz de confusão organiza os resultados da classificação em uma tabela que mostra a quantidade de observações classificadas corretamente e erroneamente em cada classe. Geralmente, a matriz de confusão é uma matriz quadrada com duas classes: a classe positiva (ou verdadeira) e a classe negativa (ou falsa).

As células da matriz de confusão são preenchidas da seguinte maneira:

- Verdadeiro positivo (VP): Representa a quantidade de observações corretamente classificadas como positivas.
- Falso positivo (FP): Representa a quantidade de observações erroneamente classificadas como positivas quando, na verdade, são negativas.
- Falso negativo (FN): Representa a quantidade de observações erroneamente classificadas como negativas quando, na verdade, são positivas.
- Verdadeiro negativo (VN): Representa a quantidade de observações corretamente classificadas como negativas.

Com base nesses valores, podemos calcular a sensibilidade (ou taxa de verdadeiro positivo) e a especificidade (ou taxa de verdadeiro negativo).

- Sensibilidade (ou taxa de verdadeiro positivo): É a proporção de casos positivos corretamente identificados em relação ao total de casos positivos. É calculada pela fórmula: Sensibilidade = VP / (VP + FN). Essa medida indica a capacidade do modelo em detectar corretamente os casos positivos.

VP	FP	VN	FN	Sensitividade	Especificidade
568	27	578	17	0.9714	0.9546

Table 1. Valores da matriz de confusão e resultados de Sensitividade e Especificidade para diferentes K's do algoritmo KNN

- Especificidade (ou taxa de verdadeiro negativo): É a proporção de casos negativos corretamente identificados em relação ao total de casos negativos. É calculada pela fórmula: Especificidade = VN / (VN + FP). Essa medida indica a capacidade do modelo em identificar corretamente os casos negativos.

Como resultado do KNN, temos a Fig. 3, onde os resultados de sensitividade e especificidade gerados após a execução do KNN, com valores acima de 95%.

Já com o resultado da Árvore de Decisão, a árvore da Fig. 2 foi gerada e sua matriz de confusão foi gerada na Tabela 1 com sensitividade e especificidade com valores também acima de 95%.

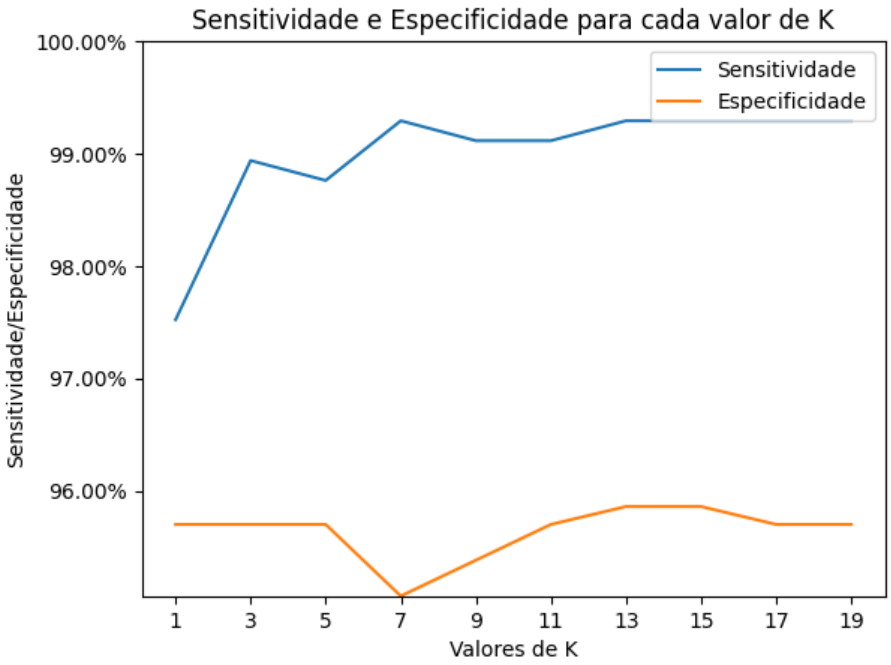

Figure 3. Resultados de Sensitividade e Especificidade para diferentes K's do algoritmo KNN

5. Conclusão e trabalhos futuros

Ambos os modelos tiveram resultados satisfatórios com relação à classificação da partida em vitória ou derrota usando os top 15 atributos mais relacionados com a variável alvo (resultado da partida), demonstrando valores de sensitividade e especificidade acima de 95%.

Pode-se afirmar que a etapa de pré-processamento se demonstrou uma etapa crucial para sucesso nas métricas de pós-processamento estabelecidas, pois de um dataset

com 123 atributos e 149232 amostras foi possível realizar a classificação com resultados consistentes.

Futuramente em outra pesquisa, a mesma base pode ser utilizada para agrupar os jogadores conforme seus dados individuais ao longo de todas as ligas e realizar a classificação para prever o resultado com os dados coletados no meio da partida (15 minutos de jogo).

References

Bahrololloomi, F., Sauer, S., Klonowski, F., Horst, R., and Dörner, R. (2022). A machine learning based analysis of e-sports player performances in league of legends for winning prediction based on player roles and performances. In *VISIGRAPP (2: HUCAPP)*, pages 68–76.

pandas development team, T. (2020). pandas-dev/pandas: Pandas.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Van Rossum, G. and Drake Jr, F. L. (1995). *Python reference manual*. Centrum voor Wiskunde en Informatica Amsterdam.