Shortest Path Analysis in the Wikipedia Graph: Machine Learning Approaches to Wikirace

Department of Politics and Public Administration

Wikipedia Data in Computational Social Science

Jannik Wirtheim (1398359)

`jannik.wirtheim@uni-konstanz.de`

Social and Economic Data Science (3. Semester)

GitHub Repository: `https://github.com/wirthy21/WDCSS-Project.git`

Date: 30.03.2025

1 Introduction

WikiRace is an online competitive game played on Wikipedia. The game involves participants to navigate from one Wikipedia page to another using only the hyperlinks in the articles (Wikipedia-Contributors, 2024b). Players start at a randomly selected Wikipedia article and aim to reach a predetermined destination article. The goal is to navigate to the destination by clicking on as few links as possible to find the shortest path or by reaching it in the shortest time. While the game's rules have remained consistent over time, the playing field continues to evolve dynamically, with articles constantly being created, updated, or deleted (Figure 1). As of December 2024, Wikipedia hosts 6,930,544 articles comprising a total of 4.7 billion words (Wikipedia-Contributors, 2024a). The compressed size of all articles (excluding media) in the current version (October 2024) is approximately 24.05 GB. The platform grows by about 14,000 articles per month (as of January 2024), with an annual text-based size increase of approximately 1 GB since 2006.

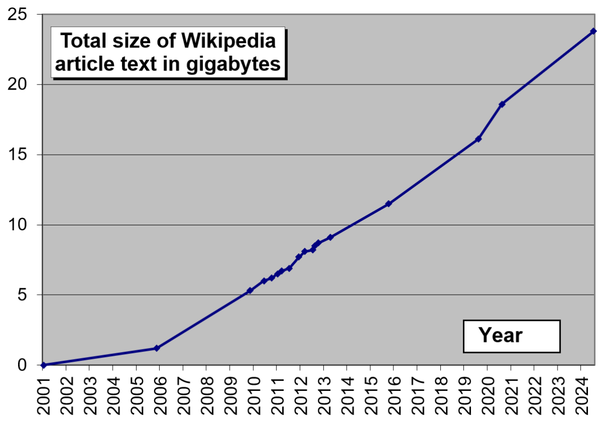


Figure 1: Growth of Wikipedia (Wikipedia-Contributors, 2025)

But not only players are interested in the game. Researchers have investigated how humans navigate Wikipedia links, focusing on the strategies they use. While most players rely on intuition and semantic knowledge of topics, some more experienced employ specific patterns, such as prioritizing high-traffic nodes as part of their strategy (Abreu, 2009; West, 2010; Lahti, 2014). With the rise of machine learning (ML), a third contributor has entered the WikiRace community. Some researchers have shifted their focus to evaluating how effectively ML algorithms can predict navigation paths in the WikiRace (Barron and Swafford, 2015; Cheng et al., 2024) or within the Wikipedia graph (Feijen and Schafer, 2021). Some approaches to predicting link distances in networks rely on embeddings as input features (Rizi et al., 2018; Barron and Swafford, 2015), while others use graph-based features (Feijen and Schafer, 2021; Aktılav and Öz, 2022) or centrality measurements to investigate shortest path distances (Vadivu, 2020). While studies directly related to WikiRace (Barron and Swafford, 2015; Cheng et al., 2024) use embeddings as features, this study investigates a graph-based metrics approach. To keep the study within the scope of available computational resources, a selection of computationally feasible metrics was identified and used. The research design was derived from the work of Barron and Swafford (2015), formulating the task as a classification problem to train models on. To evaluate shortest path distances, they predicted distances from random articles to the destination article "Stanford University." Similarly, in this study, shortest paths are calculated to "University of Konstanz." Due to hardware limitations, Barron and Swafford (2015) constrained their analysis to the "Simple English Wikipedia," a smaller, simplified version designed for educational purposes. In contrast, this study uses the main English Wikipedia, as it is where most WikiRace tournaments take place. Additionally, Wikipedia has evolved significantly over the past decade, making past models potentially outdated for larger networks. In their future work, Barron and Swafford (2015) mentioned investigating deep learning (DL) models, such as Multi-Layer Perceptron (MLP), to evaluate their performance on the same task. This leads to the following research questions:

• What is the actual structure of Wikipedia in terms of WikiRacing?
• How effective are different ML models at predicting shortest paths using graph-based metrics?
• Is Deep Learning necessary for this task?
• How do these models perform compared to a completely random classifier?

2 Research Design

2.1 Data Collection and Cleaning

Due to its open-source nature, the Wikimedia Foundation provides open access to backups of its platforms (Wikimedia Foundation, 2025). These Wikipedia dumps contain various types of data extracted from the Wikipedia database, including page content, metadata, revision history, and structured relationships between pages. Regularly updated, they serve as a common data source for research. The "Latest pagelinks" dataset (enwiki-latest-pagelinks.sql) stores information about hyperlinks between Wikipedia pages. Making it particularly well-suited for the objectives of this study.

First, the SQL file containing the dataset was converted into a dataframe to facilitate further preprocessing (see Code "WDCSS Project – Data"). Due to the immense size of the data, this step was performed in smaller batches. The final dataframe consists of 1.6 billion rows, including the page ID, the type of the page (e.g., article, talk, or discussion page), and the linked article. To reduce complexity and data size, only articles (pages with type ”0”) were retained. Additionally, self-loops, dead ends, and orphaned articles were removed to enhance computational feasibility. Self-loops are hyperlinks that point to the page itself, providing no additional information and offering no benefit for players. Dead ends are articles that lack outgoing links, if these were randomly selected as starting pages, players wouldn't be able to traverse away from them, rendering the game unplayable. Orphaned articles, which lack both incoming and outgoing hyperlinks, unnecessarily increase the complexity of the network without offering any benefit as well. After these preprocessing steps, the final graph was built. It consists of 4,520,178 nodes and a total of 101,881,676 edges. The Giant Component Ratio (GCR) was used as a metric to further validate the graph structure. A GCR of 0.4824 indicates that approximately half of the graph is part of the largest connected subgraph. This suggests that the graph has a significant, though not dominant, largest component, with many nodes still in smaller components or isolated. To extract only the nodes of the largest connected subgraph related to the article "University of Konstanz," the entire network was filtered again. The reduction in size was minimal, resulting in the UniKN-Graph, which now consists of 4,515,338 nodes and 101,879,136 edges.

Feature engineering involves transforming raw data into meaningful features that improve the performance of machine learning models (IBM, 2025). For this purpose, a dataframe was created containing both the features and the dependent variable to facilitate classification. The criterion was computed by calculating the shortest paths between every article node and the destination article "University of Konstanz" using Dijkstra's Algorithm. The graph search algorithm finds the shortest path between a source node and all other nodes with non-negative edge weights (Dijkstra, 1959). It works by iteratively selecting the unvisited node with the smallest tentative distance, updating the distances to its neighbors, and marking nodes as visited once their shortest distance is determined. These computed shortest paths were distributed in a range from the minimum of 1 to the maximum of 15 (Figure 3b). To select meaningful predictors, the trade-off between computational resources and insightfulness was considered. For instance, Eccentricity, which measures the maximum shortest path distance from a node to any other node in the graph, was deemed unsuitable due to its computational complexity of $O(V \cdot (V + E))$. Therefore, the following features were chosen (Table 1):

Feature	**Description**
PageRank	Measures the relative importance of a node by distributing scores based on its incoming neighbors' ranks (Page et al., 1999). Originally designed for ranking web pages.
In-Degree and Out-Degree	In-degree counts the number of incoming links to a node, while out-degree counts outgoing links, indicating a node's connectivity (Developers, 2025).
Hub Score	Evaluates a node's importance based on how well it links to influential nodes, similar to how WikiRace players navigate through high-traffic pages (GeeksforGeeks, 2023).
Authority Score	Measures how many important hubs link to a given node, closely related to the Hub Score (GeeksforGeeks, 2023).
Eigenvector Centrality	Assesses a node's influence by considering the centrality of its neighbors, derived from the eigenvector of the adjacency matrix (NetworkX Developers, 2025b).
Betweenness Centrality	Counts how often a node appears on shortest paths between other nodes, indicating its role as a bridge in the network (NetworkX Developers, 2025a).

Table 1: Summary of Selected Features

The correlation matrix of the features shows, that most of them are uncorrelated (Appendix A, Figure 10). A strong relationship between *In-Degree Source* and *Eigenvector Centrality* (0.93) exists. Additionally, *Hub Score* and *Out-Degree Source* show moderate correlation (0.67) as well. Highly correlated features can introduce redundancy in models, potentially leading to multicollinearity issues. This problem could be addressed using Principal Component Analysis (PCA) to reduce the dataset's dimensionality. However, this would make the analysis of feature importance unfeasible. Being aware of these limitations, all predictors are included in the model to assess their individual importance and effectiveness in handling the task.
Due to class imbalance in the dataset, classification performance suffered. For example, shortest path lengths of 14 and 15 appeared only 16 and 5 times, respectively, whereas a length of 5 occurred 1,052,684 times. To improve model performance, minority classes with fewer than 10,000 samples were removed. Specifically, classes 1, 2, 10, 11, 12, 13, 14 and 15 were dropped, reducing the dataset by only 12,406 observations, which accounted for just 0.004% of the total dataset. Resampling the dataset was not feasible due to the extreme imbalance and computational resource limitations. Finally, all feature variables were standardized, and the dataset was split into training and testing sets in an 80-20 proportion.

2.2 Model Architectures

To evaluate how well different ML models predict shortest paths in the Wikipedia network using graph-based parameters, three models were trained (see Code ”WDCSS Project - ML”). These models represent distinct approaches to classification and interpretability (Figure 2). While Multi-Layer Perceptron (MLP) offer high performance as a black-box approach, they suffer from a lack of interpretability, whereas k-Nearest Neighbors (KNN) is more interpretable. Extreme Gradient Boosting (XGBoost) was chosen as the third model because it allows for investigating the importance of individual predictors.

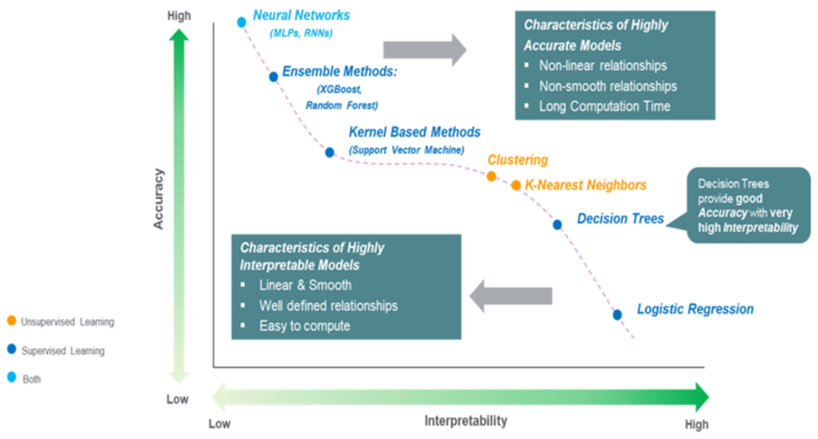


Figure 2: Model Selection based on Interpretability Accuracy Trade-off (Rane, 2018)

The KNN model assigns a data point to a class based on local probabilities (Taunk et al., 2019). The key hyperparameter k determines the number of closest neighbors considered for classification. To find the optimal k, 10-fold cross-validation (CV) was applied to the training set. In this method, the model was trained on ten different subsets of the dataset, with nine folds used for training and one fold for validation in each iteration. This approach identified an optimal k-value of 10 for this task. The best model was evaluated using the weighted F1-score, which reflects the harmonic mean of precision and recall, taking dataset imbalance into account (GeeksforGeeks, 2025).

XGBoost constructs an ensemble of weak decision trees, where each tree corrects the errors of the previous ones by iteratively minimizing a loss function (Chen and Guestrin, 2016). Due to hardware limitations, automatic hyperparameter tuning was not feasible, so parameters were tuned manually. The process began with 300 boosting rounds, a learning rate of 0.1, and a maximum depth of five. The classifier’s performance improved as these metrics were adjusted for more fine-grained training. Some parameters remained unchanged throughout training, including the minimum sum of weights of the child node, gamma for partitioning, and lambda for L2 regularization. Multi-class logarithmic loss was used as the loss function, as it is a common choice for multi-class classification tasks.

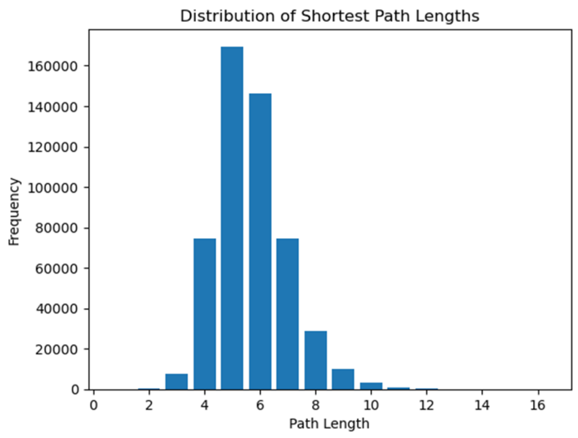
To determine whether deep learning was necessary for this task, a MLP was trained. Due to the long training time (8 hours), the model was evaluated only three times, and hyperparameters were tuned as well manually. To make the Network “deep” five Hidden Layers with decreasing neuron count were implemented. After each layer, the ReLU activation function was applied to introduce non-linearity into the model. For training, cross-entropy loss (CEL) was used. By default, CEL applies the softmax activation function to the output for multi-class classification. The Adam optimizer was chosen, as it combines

Stochastic Gradient Descent with momentum, leading to faster convergence (Kingma and Ba, 2014). To prevent overfitting, early stopping was implemented with a patience of ten epochs, meaning training would halt if no significant improvement occurred within that period. The model was trained on a 20% validation set, and early stopping was triggered after 63 epochs.

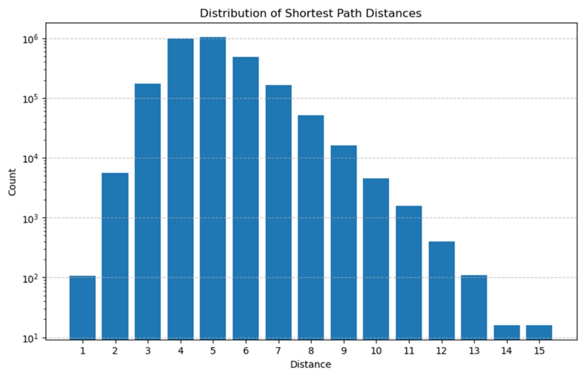
3 Analysis and Results

3.1 Exploratory Analysis

The aim of the exploratory analysis is to investigate the current graph structure of Wikipedia in terms of WikiRacing (see Code ”WDCSS Project - Explorative Analysis”). The results showed that the average shortest path length between nodes in the graph is 5.689, indicating that it typically takes about six steps to navigate between two randomly selected articles (Figure 3a). The distribution of shortest path lengths exhibited a peak between 5 and 7, aligning with the average and indicating that most articles are relatively close to each other within the graph. Only a few paths extended beyond a length of 10, suggesting that exceptionally long connections are uncommon. A similar trend is observed in the distribution of shortest path lengths to the article ”University of Constance” (Figure 3b). While most paths cluster around 5, those exceeding 10 become increasingly rare.



(a) Distribution of Shortest Path Lengths in Wikipedia Graph



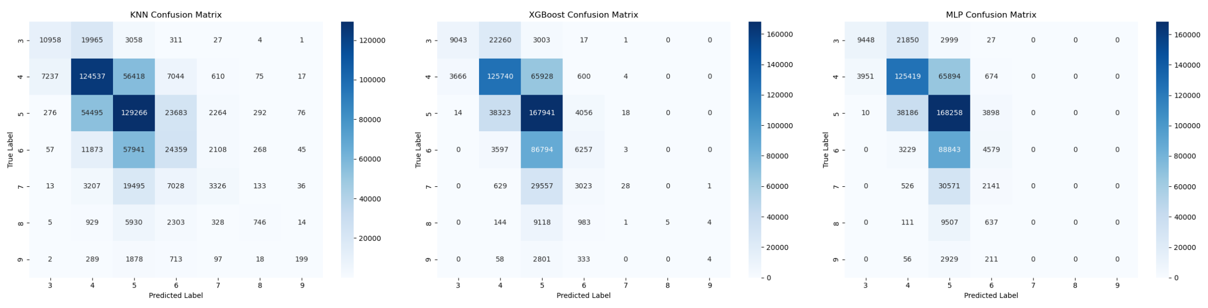
(b) Distribution of Shortest Path Lengths to ”University of Constance”

Figure 3: Shortest Path Distributions

The descriptive analysis of the top-ranked nodes with most in- and outgoing connections lead to surprising results (Appendix A, Figure 5 and 6). The Wikipedia articles with the most incoming nodes are lists with a collection of a specific topic. Therefore, the node with the most incoming hyperlinks is a collection of bird names. But also political ideologies and Irelend-related topics offer a lot of incoming connections. The result of the top-ranked nodes with outgoing hyperlinks is a bit strange. But also after double-checking, the results stay consistent. ”Blow and go,” which relates to the Wikipedia article on ”Emergency ascent” for divers, is the article with the most outgoing connections. However, a manual review of the article does not confirm this result, so the analysis should be interpreted with skepticism. Similarly, the PageRank analysis, like the results for the most incoming nodes, primarily highlights lists with high influence (Appendix A, Figure 7). The most influential node in this case is a list of Hieracium species, with the list of bird names also appearing again. While WikiRace research of the Cornell University (2022) identified that most of the connections of traversing the Wikipedia graph almost always consisted of accessing a specific date, this finding can't be replicated. A analysis of 1,000,000 shortest paths showed that the nodes are more or less visited arbitrary than with a underlying logic (Appendix A, Figure 8). Redoing the analysis without setting an seed lead to different results as well. Furthermore a analysis was conducted to see how removing the most-influential nodes iteratively (10,000 at each step) affects the average path length of traversing the graph (Appendix A, Figure 9). Based on intuition it should be logically that with the removal of nodes the complexity of the network decreases and therefore the average shortest paths length should decrease. After 50 iterations and a total removal of 500,000 nodes there is a clear linear trend visible (Appendix A). With the deletion of 50k most-influential nodes, based on PageRank measurement, the average shortest path length steadily increases from around 5.6 to over 7.0 indicating the crucial role these nodes play in preserving the graph's efficiency and connectivity.

3.2 Model Comparison

Due to class imbalance, interpreting accuracy, precision, and recall without considering class distribution would lead to biased results. To address this imbalance, macro and weighted averages of these metrics must be considered. Macro averaging calculates each metric separately for each class and then averages the results across all classes (Sklearn Developers, 2025). Weighted averaging, on the other hand, assigns weights to each class based on its proportion in the dataset, meaning that classes with more observations contribute more to the final metric compared to less frequent classes. Since weighted metrics prioritize majority classes, macro averaging penalizes classifiers that perform well on large classes but poorly on smaller ones. Therefore, weighted metrics are used. The effect of class imbalance is evident in the accuracy metric. For example, while the KNN classifier achieves an accuracy of 48%, this value is misleading, as seen in the confusion matrix. The model performs well in predicting the most frequent classes, such as class 4 and class 5, but struggles significantly with class 9.



(a) Confusion Matrix KNN (b) Confusion Matrix XGBoost (c) Confusion Matrix MLP

Figure 4: Confusion Matrices of different ML Models

Comparing the results of the three classifiers, no significant differences emerge (Table 2). Interpreting the weighted precision, recall, and F1-score, the study shows that the XGBoost model performs best, achieving the highest overall precision (0.55) while matching the recall (0.53) and accuracy of the MLP. The weakest model is KNN, with an accuracy of 50%, meaning it predicts the class correctly or incorrectly about half of the time. Its precision (0.49) and recall (0.50) are slightly worse in comparison to the other models. Examining the confusion matrix reveals interesting insights (Figure 4). While the stronger models, XGBoost and MLP, struggle with the classification of minority classes, KNN performs notably well in these cases. MLP was unable to classify connections with shortest paths longer than six, and XGBoost only correctly classified a few beyond this threshold. This suggests that both models achieve higher classification metrics overall but fail to accurately classify minority classes. KNN, on the other hand, excels in handling minority classes but performs more worse in high-frequency classes like class 5. Despite its lower generalization across all classes, its ability to capture minority class patterns more effectively makes it a noteworthy contender. Interestingly, the most complex model, MLP, was unable to fully handle the task. Comparing the models to a purely random classifier, the study reveals that graph-based metrics as features are suitable predictors for shortest path predictions. The random classifier showed overall similar performance, with approximately 0.27 for all metrics, indicating that all trained classifiers perform better than random guessing.

Table 2: Results of different ML models

Model	Accuracy	Precision	Recall	F1-Score
KNN	50%	49%	50%	48%
XGBoost	53%	55%	53%	47%
MLP	53%	49%	53%	46%
Random	27%	27%	27%	27%

Analyzing the final XGBoost tree, the importance of different predictors is evident (Appendix C). The x-axis represents the feature importance scores, the higher the value, the greater the influence of the predictor. Among the features, the *Out-Degree Nodes* has the highest importance score, surpassing all others. This suggests that this feature plays the most critical role in the model's decision-making process. It is followed by *Eigenvector Centrality* and *In-Degree Nodes*, which, while less influential, still play a significant role in the model. In comparison, *Authority Score*, *Hub Score*, *Page Rank*, and *Betweenness Centrality* have negligible influence.

To assess the variability of the results, uncertainty measurements were performed on the bootstrapped test set (n = 200) for each model. Interpreting the boxplots and the density plot of the KNN model, only small variability is observable (Appendix B). While the boxes in the boxplots are compact and small, the density plot exhibits a sharp peak, indicating low uncertainty. In comparison, the XGBoost (Appendix C) and the MLP model (Appendix D) shows greater uncertainty. While Accuracy, Recall, and F1-Score all display compact boxplots, Precision exhibits a larger variability in both models. This is also reflected in the density plot, where Precision has a flatter distribution. Although F1-Score and Recall exhibit a more pronounced peak, they still demonstrate greater variability compared to KNN.

From a practical perspective, deep learning was not necessary for this task. In most applications, using deep learning requires a trade-off between performance and computational complexity. However, in this analysis, the MLP did not provide a significant performance advantage over the other models, making this trade-off unnecessary. Furthermore, the MLP performed the worst in classifying minority classes, making it unsuitable for efficiently handling WikiRacing. XGBoost, combining the interpretable structure of decision trees and random forests, delivered the best overall results with low training times. However, like MLP, it failed to capture low-frequency classes effectively. While KNN had slightly worse overall classification performance, its ability to handle minority classes. the stability of the results and its interpretability make it the most suitable choice for WikiRacing.

4 Conclusion

This study provided valuable insights into predicting the shortest paths in the Wikipedia graph, as in the WikiRacing game, using only graph-based metrics. The findings address the research question (Section 1) as follows:

The analysis of Wikipedia's network in terms of the game shows that articles are highly interconnected, with most being reachable in just 5 to 6 steps. The most important pages are often lists that group related topics, acting as central hubs that keep the network well-connected. Removing these key pages significantly increases the average shortest path length, showing their crucial role in maintaining Wikipedia's structure. Without these influential nodes, navigation within Wikipedia would become noticeably less efficient, requiring users to take more steps to find related content.

Applying ML models to predict shortest paths using only graph-based metrics provided insightful results as well. By comparing the features based on their importance (Appendix C, Figure 17), the most influential features in the model could be identified. In particular, the number of *Out-Degree* and *In-Degree Nodes*, as well as *Eigenvector Centrality*, were the most important for the model's decision-making, enabling models to correctly classify approximately half of the shortest paths. While this confirms the utility of these metrics, it also highlights their limitations in fully capturing the complexity of Wikipedia's network, limiting their effectiveness for WikiRacing.

The MLP model did not significantly outperform traditional machine learning models such as XGBoost or KNN, despite its higher computational cost. Furthermore, MLP struggled with classifying longer shortest paths, making it unsuitable for handling class imbalances. XGBoost emerged as the best-performing model in terms of weighted precision and recall but still failed to accurately classify minority classes. KNN, while slightly weaker in overall accuracy, performed better in these rare cases, suggesting its potential for applications where minority classes hold particular importance.

A random classifier achieved a classification performance of approximately 0.27 across all metrics, confirming that all trained models performed significantly better than random guessing. This reinforces the validity of graph-based metrics as meaningful predictors for shortest path estimation.

5 Critique

While this study provided valuable insights in how ML could handle shortest paths in WikiRacing, several limitations must be acknowledged. One major issue was the imbalance in shortest path distributions, as paths leading to the target article "University of Konstanz" were not evenly distributed. As a result, path lengths beyond 9 were entirely excluded from the dataset. Even after attempts to mitigate class imbalance through removal of minority classes, the remaining skew still affected model performance, particularly for MLP. Additionally, the study was constrained to computationally feasible graph metrics, which, while effective, may not fully capture the network's structure. More complex features, such as embeddings from Barron and Swafford (2015) or Cheng et al. (2024), could enhance predictive performance. Future research could explore hybrid approaches that integrate these embeddings with graph metrics. Furthermore, the relevance of different features should be investigated first to develop a more interpretable and comprehensible model. Another notable limitation was the lack of systematic hyperparameter tuning due to hardware constraints. Instead, parameters were adjusted manually, which may have prevented models from achieving optimal performance. This limitation likely impacted MLP and XG-Boost the most, as both are highly sensitive to hyperparameter adjustments. Furthermore, the study could have benefited from additional model comparisons, such as Logistic Regression, which was identified as an effective model by Barron and Swafford (2015). Given its lower complexity and higher interpretability, including it would have provided a useful benchmark. To address these limitations, future research should focus on improving the classification of minority path lengths. Strategies such as resampling techniques, alternative target articles, or refined preprocessing of the graph structure could help balance the dataset. Moreover, the application of deep learning methods specifically tailored to graph data, such as Graph Neural Networks (GNNs), should be explored, as they may be more suitable for capturing the relationships within Wikipedia's network. Lastly, testing these models in real-world applications, such as predicting WikiRace navigation patterns beyond shortest path estimation, could further validate their effectiveness in competitive settings. Expanding the analysis to tournament-based strategies or training human players could provide additional insights into the interplay between human navigation behavior and algorithmic predictions.

Overall, while XGBoost and MLP demonstrated strong classification performance on common path lengths, KNN proved to be the most interpretable and effective for handling rare cases. The findings highlight the importance of balancing interpretability, computational efficiency, and classification accuracy when selecting machine learning models for graph-based predictions.

A Appendix: Exploratory Analysis

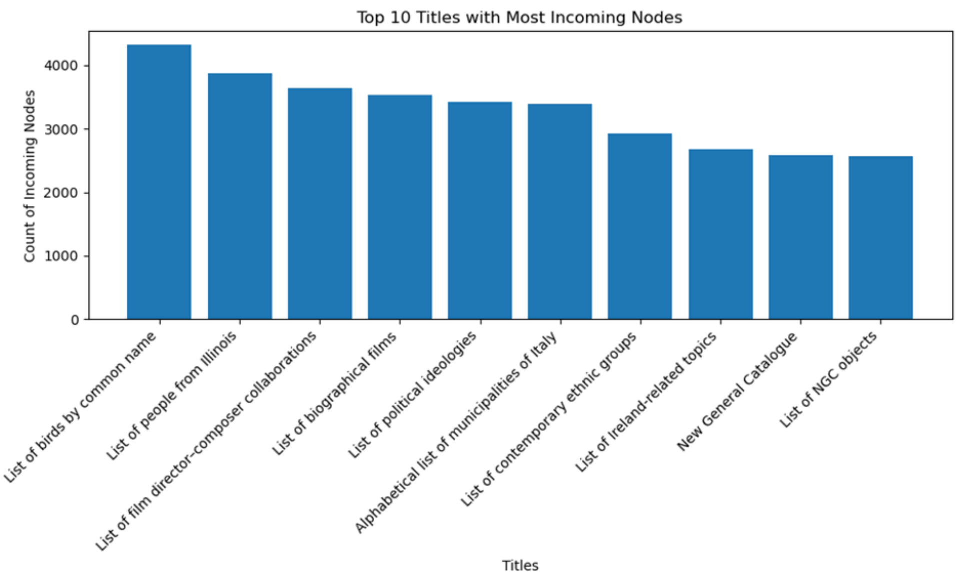


Figure 5: Top 10 Titles with Most Incoming Nodes

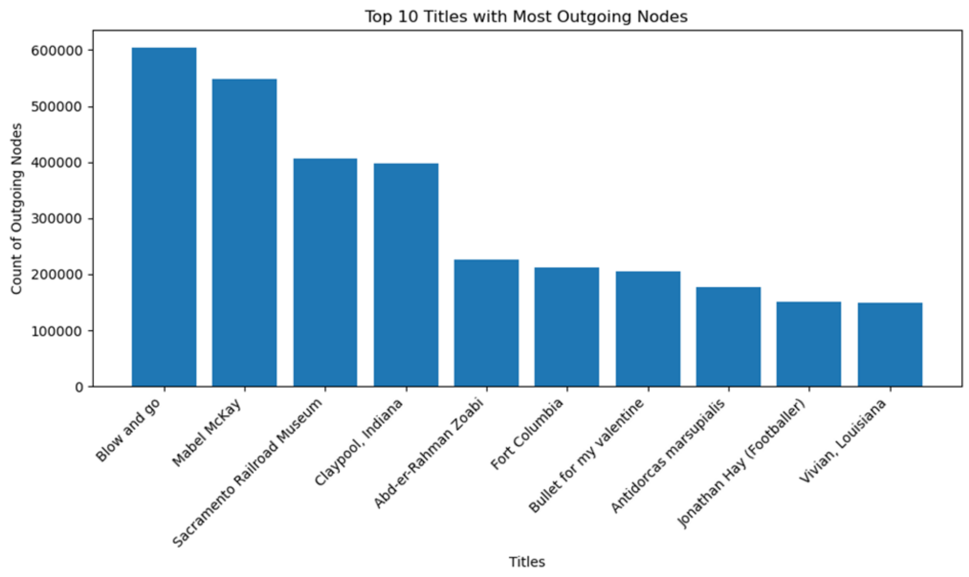


Figure 6: Top 10 Titles with Most Outgoing Nodes

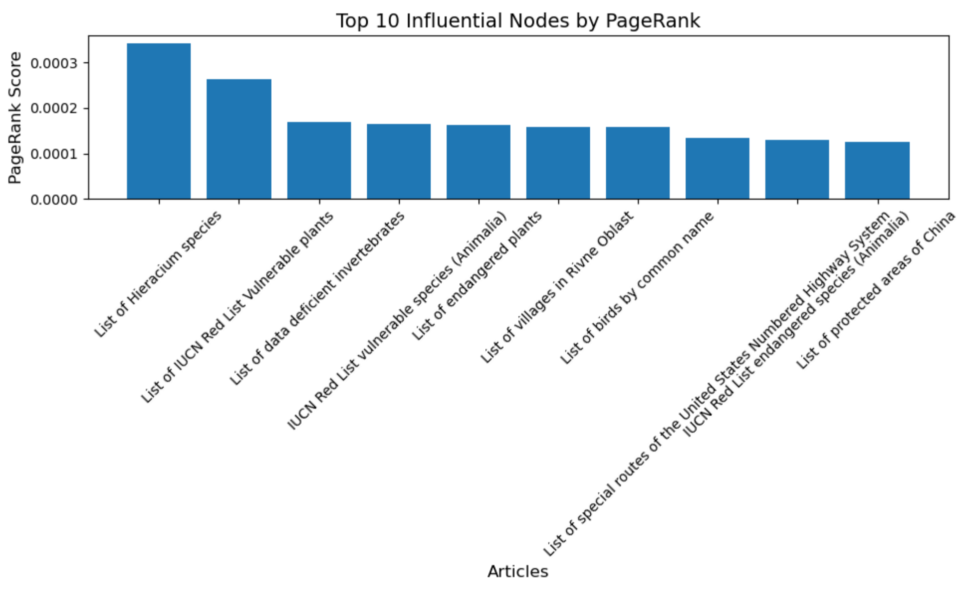


Figure 7: PageRank Analysis

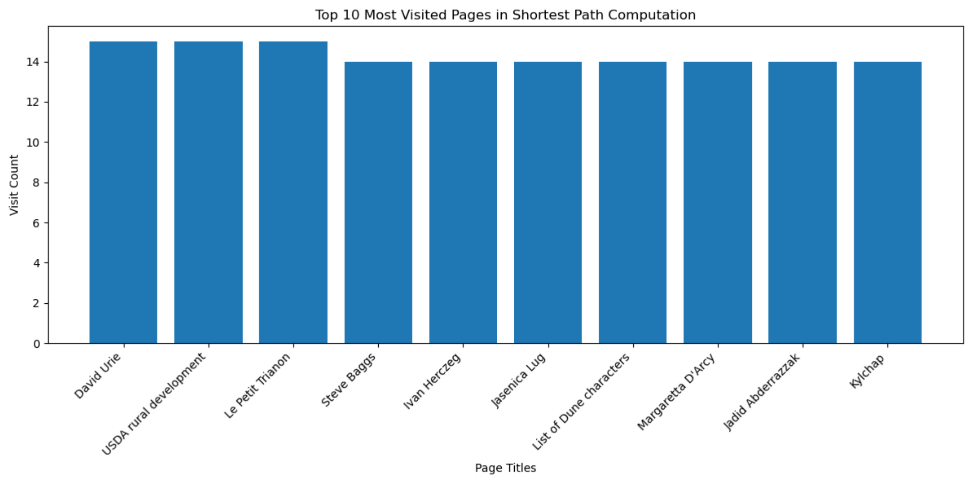


Figure 8: Top 10 Nodes in Shortest Path Computation

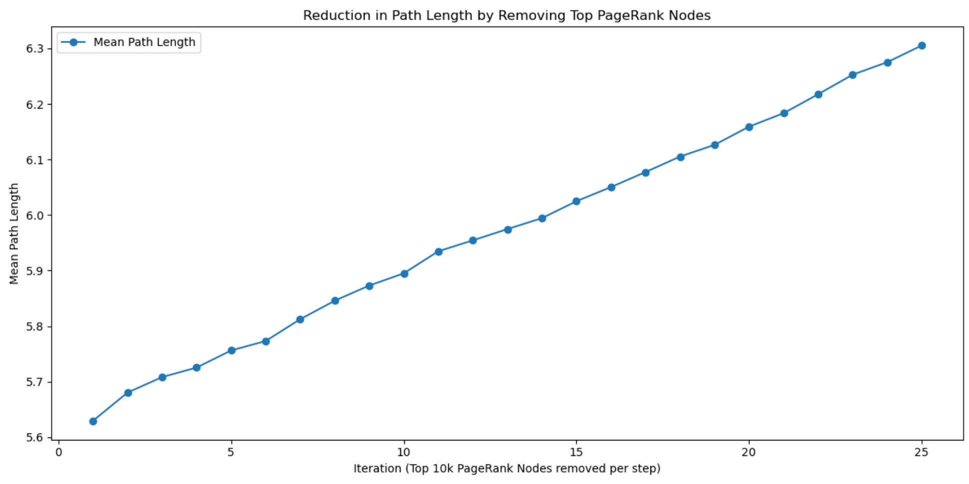


Figure 9: Reduction of Average Path Length by Removing of Top PageRank Nodes

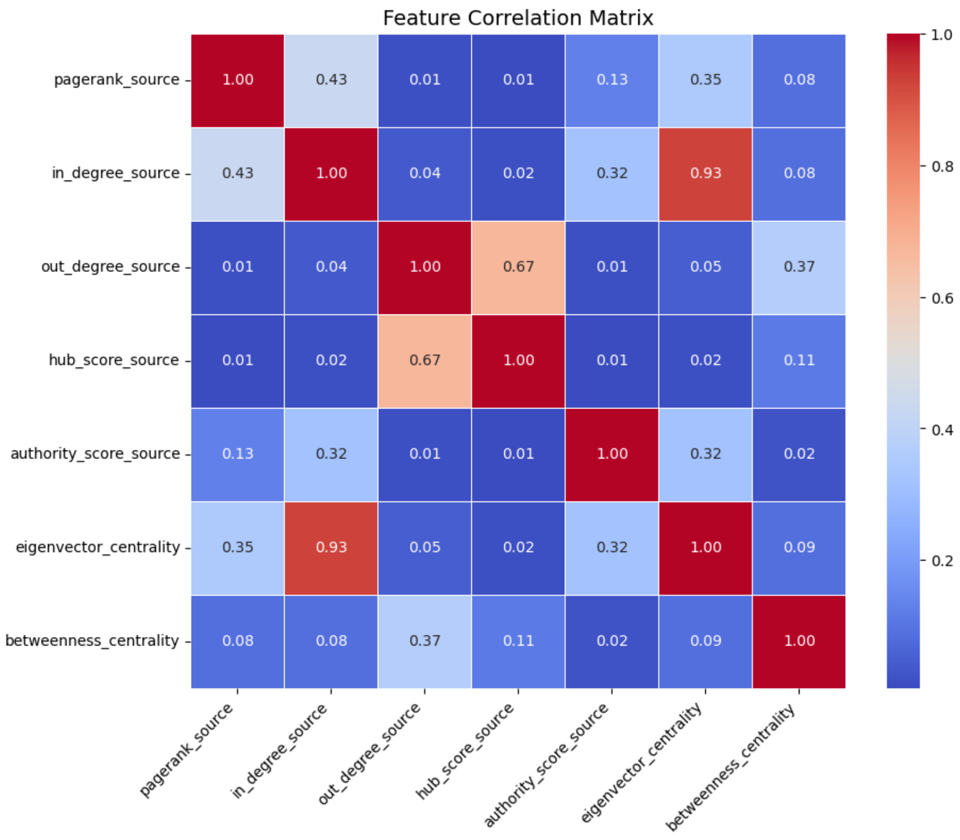


Figure 10: Feature Correlation Matrix

B Appendix: KNN Evaluation

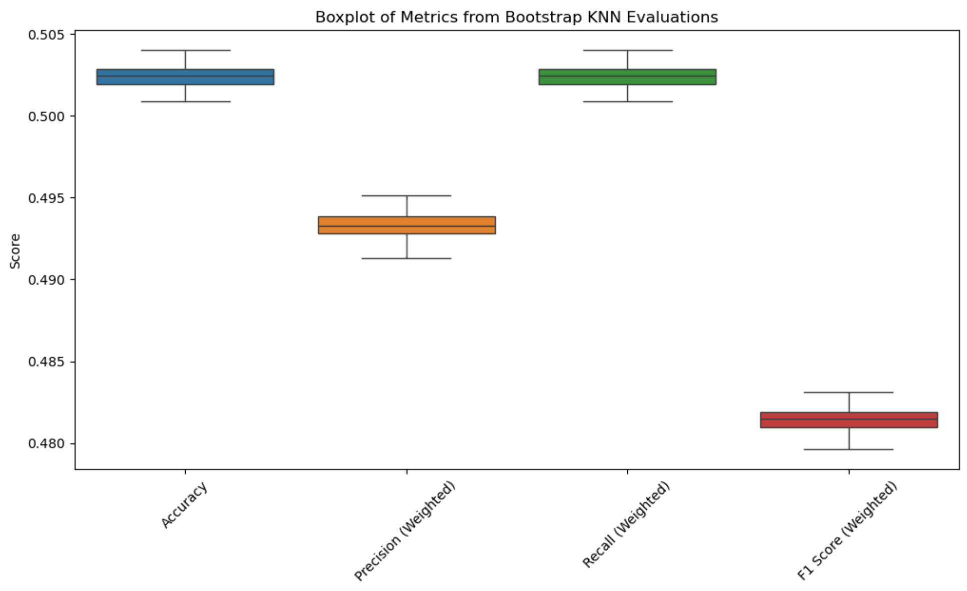


Figure 11: KNN Boxplots

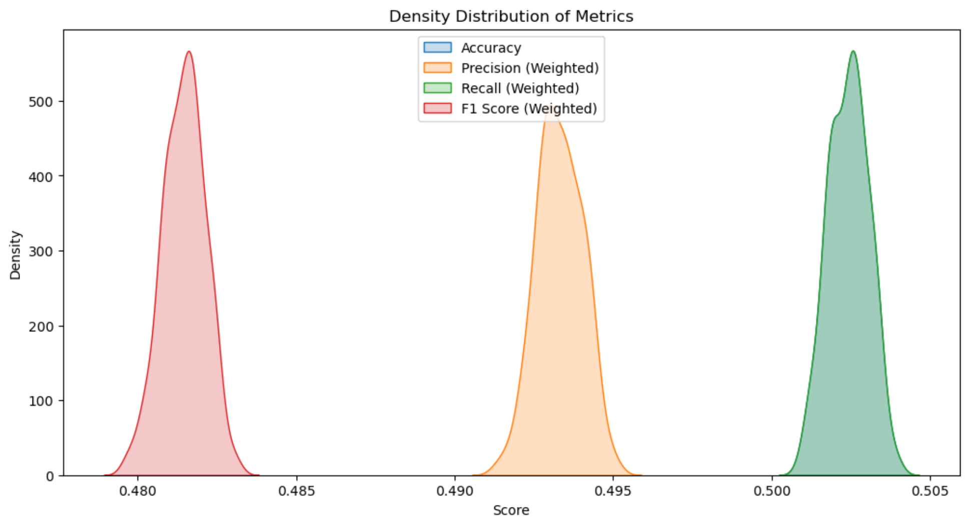


Figure 12: KNN Density Plot - Accuracy seems to be missing, but weighted recall is equal to the accuracy results (User Community, 2018; ResearchGate Community, 2020) (unfortunately, I couldn't find better references).

C Appendix: XGBoost Evaluation

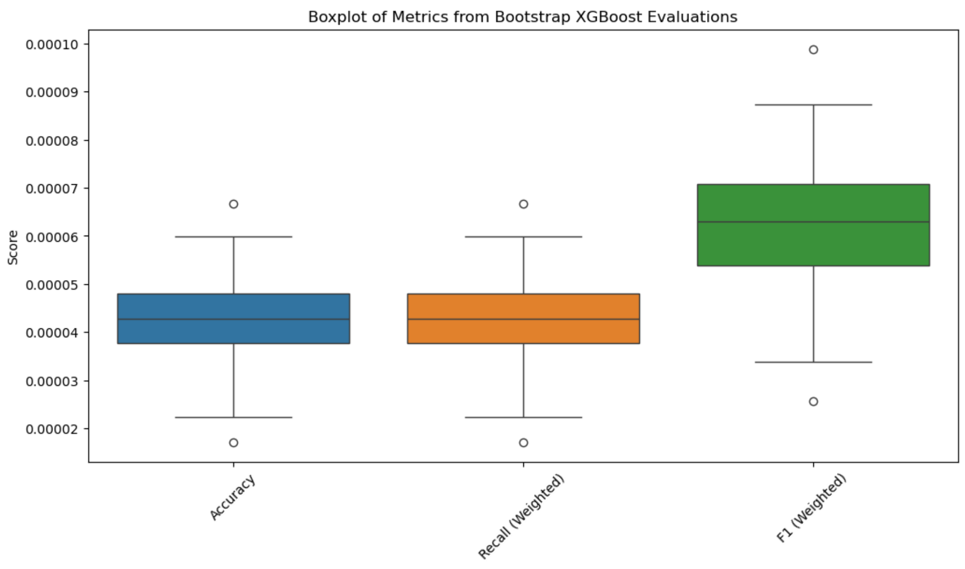


Figure 13: XGBoost Boxplots without Precision

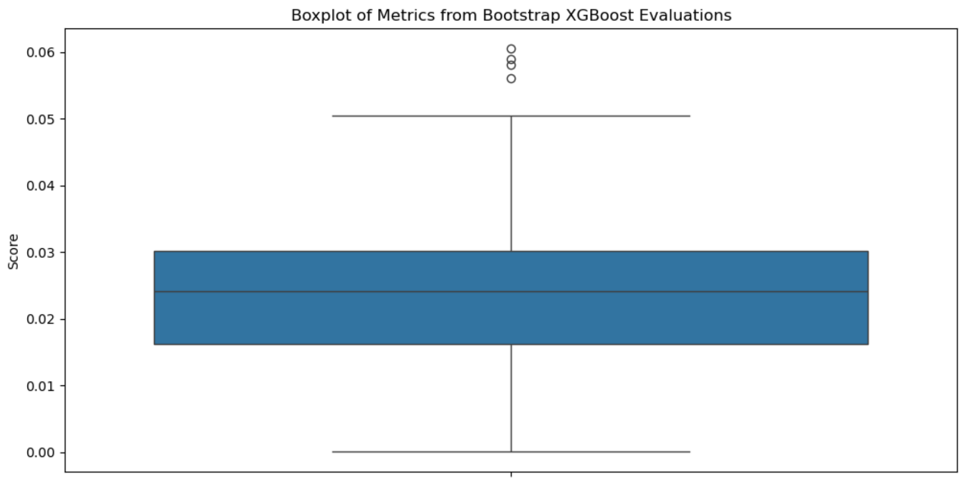


Figure 14: XGBoost Precision Boxplot

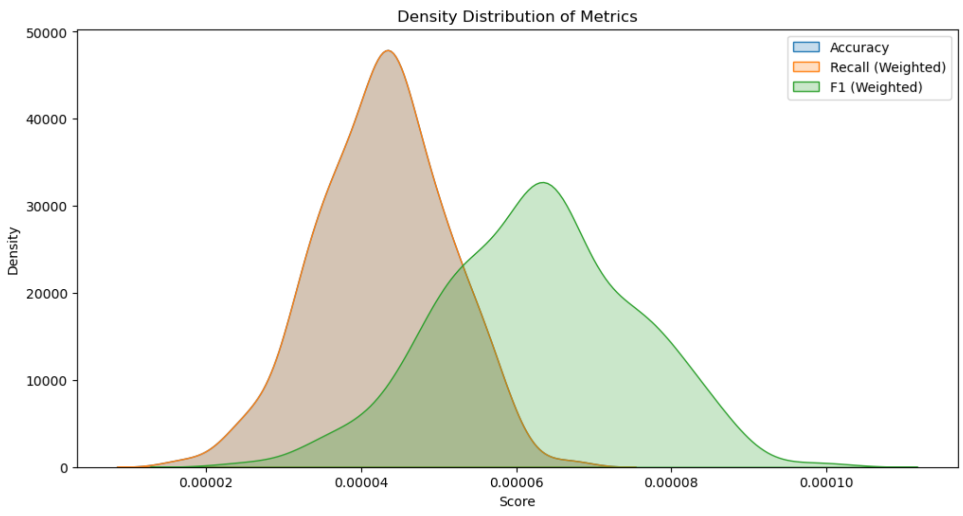


Figure 15: XGBoost Density Plot without Precision - Accuracy seems to be missing, but weighted recall is equal to the accuracy results (User Community, 2018; ResearchGate Community, 2020) (unfortunately, I couldn't find better references).

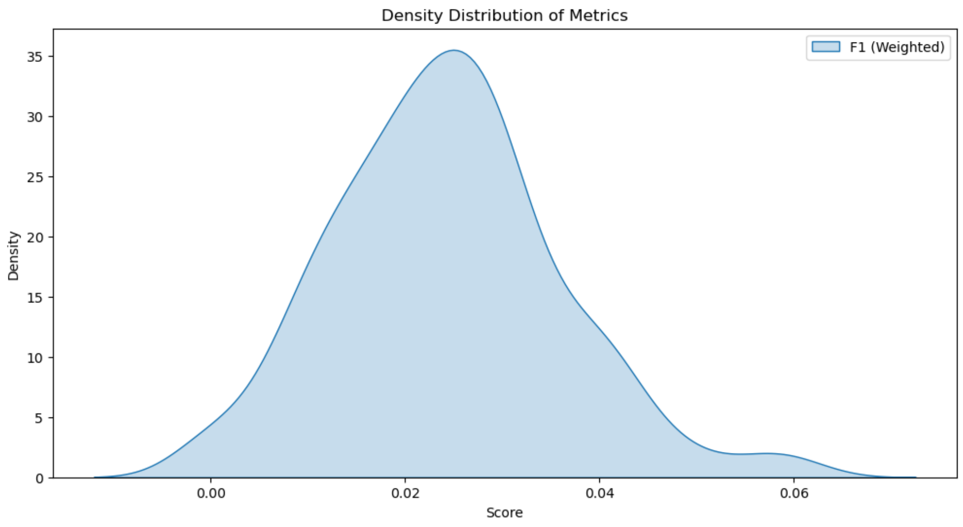


Figure 16: XGBoost Precision Density Plot

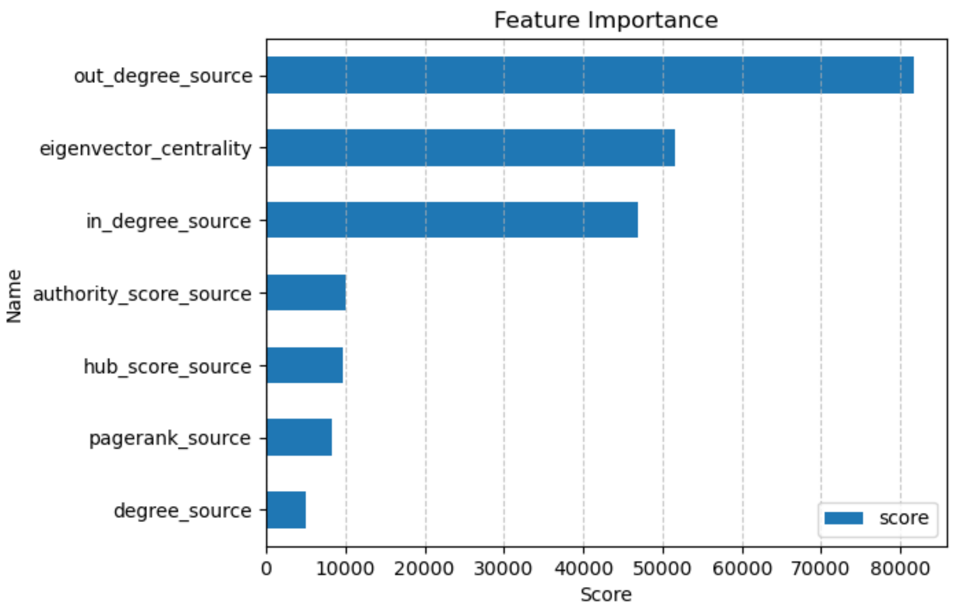


Figure 17: Feature Importance

D Appendix: MLP Evaluation

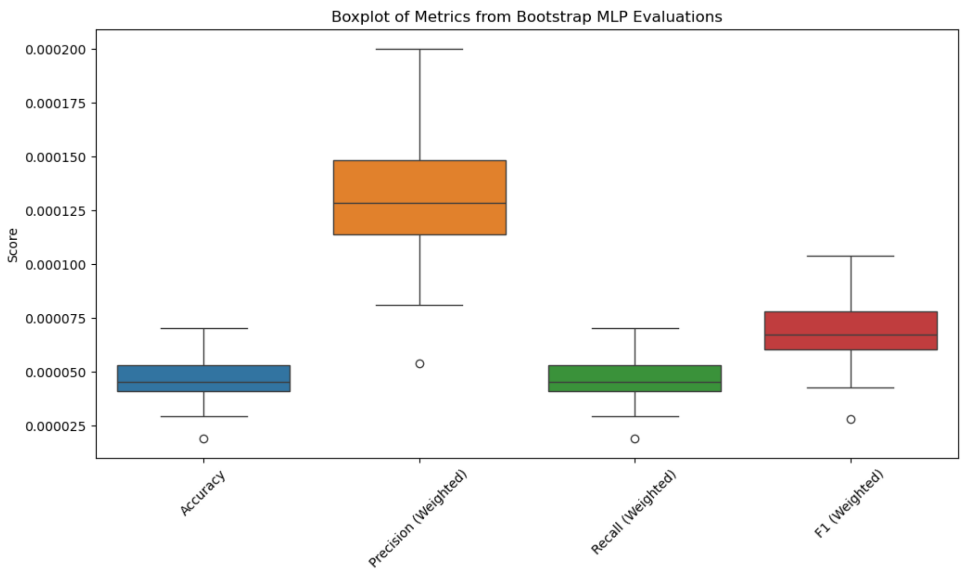


Figure 18: MLP Boxplots

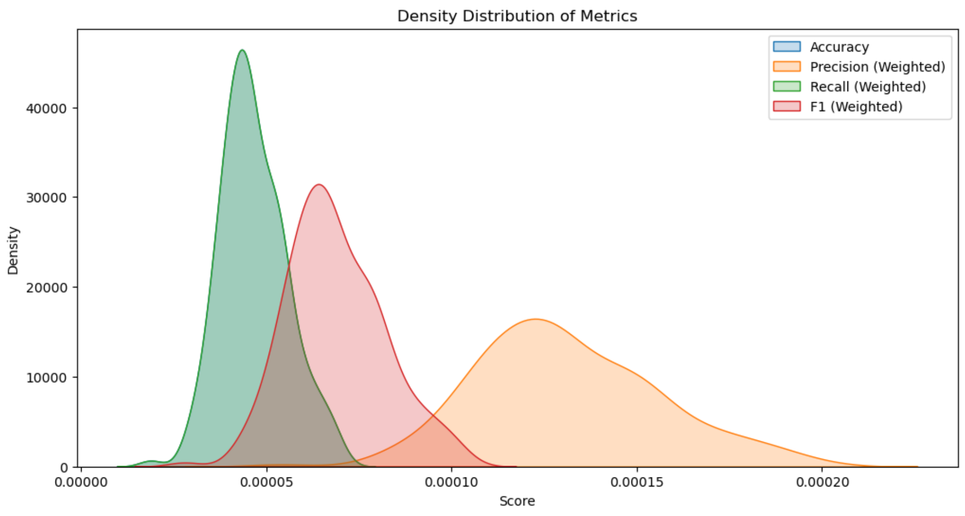


Figure 19: MLP Density Plot - Accuracy seems to be missing, but weighted recall is equal to the accuracy results (User Community, 2018; ResearchGate Community, 2020) (unfortunately, I couldn't find better references).

References

Abreu, A. (2009). Wikipaths: Re-imagining information space through semantic play. Accessed: 2025-02-14.

Aktılav, B. and Öz, I. (2022). Performance and accuracy predictions of approximation methods for shortest-path algorithms on gpus. *Parallel Computing*, 113:102942. Accessed: 2025-02-14.

Barron, A. and Swafford, Z. (2015). An ai for the wikipedia game.

Chen, T. and Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794.

Cheng, G., Wang, O., Adams, A., Biehl, M., Caspar, L., and Witkowski, O. (2024). Interacting llms: A dive into collaborative ai. In *2024 IEEE 18th International Conference on Semantic Computing (ICSC)*, pages 152–155.

Cornell University (2022). An analysis of wikiracing and the web as a network. Accessed: 2025-02-14.

Developers, N. (2025). Digraph.out_degree — networkx 3.0 documentation. Accessed: 2025-02-14.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271.

Feijen, W. H. J. and Schafer, G. (2021). Dijkstras algorithm with predictions to solve the single-source many-targets shortest-path problem.

GeeksforGeeks (2023). Hyperlink-induced topic search (hits) algorithm using networkx module | python. Accessed: 2025-02-14.

GeeksforGeeks (2025). F1 score in machine learning. Accessed: 2025-03-10.

IBM (2025). Feature engineering. Accessed: 2025-03-10.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lahti, L. (2014). Experimental evaluation of learning performance for exploring the shortest paths in the hyperlink network of wikipedia. In *World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education 2014*, pages 1069–1074, New Orleans, Louisiana, USA.

NetworkX Developers (2025a). networkx.algorithms.centrality.betweenness$_centrality. Accessed : 2025 - 02 - 14$.

NetworkX Developers (2025b). networkx.algorithms.centrality.eigenvector$_centrality. Accessed : 2025 - 02 - 14$.

Page, L., Brin, S., Motwani, R., and Winograd, T. (1999). The pagerank citation ranking : Bringing order to the web. In *The Web Conference*.

Rane, S. (2018). The balance: Accuracy vs. interpretability. Accessed: 2025-03-12.

ResearchGate Community (2020). Multiclass classification: micro (weighted) recall equals accuracy? Accessed: 2025-03-12.

Rizi, F. S., Schlötterer, J., and Granitzer, M. (2018). Shortest path distance approximation using deep learning techniques. *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 1007–1014.

Sklearn Developers (2025). sklearn.metrics.f1_score. Accessed: 2025-03-10.

Taunk, K., De, S., Verma, S., and Swetapadma, A. (2019). A brief review of nearest neighbor algorithm for learning and classification. In *2019 International Conference on Intelligent Computing and Control Systems (ICCS)*, pages 1255–1260.

User Community (2018). Recall equals to accuracy but different to precision. Accessed: 2025-03-12.

Vadivu (2020). Shortest path of a graph using centrality measures. *International Journal of Advanced Trends in Computer Science and Engineering*, 9:3898–3903.

West, R. (2010). *Extracting Semantic Information from Wikipedia Using Human Computation and Dimensionality Reduction*. Ph.d. thesis, McGill University.

Wikimedia Foundation (2025). English Wikipedia Database Dump. Accessed: 2025-02-14.

Wikipedia-Contributors (2024a). Wikipedia: Size of wikipedia. Accessed: 2025-02-14.

Wikipedia-Contributors (2024b). Wikipedia: Wiki game. Accessed: 2025-02-14.

Wikipedia-Contributors (2025). Wikipedia: Size of Wikipedia - Article Size in Gigabytes. [Online; accessed 11 March 2025].