# Assignment 1: Getting Started with Machine Learning

## COMP 551

January 31st, 2025

Androw Abd El Malek (261050237)
Kevin Luo (261038261)
Radu Petrescu (261051351)

**Abstract**

In this assignment, we implemented and evaluated weighted K-Nearest Neighbors (KNN) and Decision Trees (DT) on the Heart Disease and Penguin datasets for binary and multiclass classification, respectively. Both models were designed using object-oriented programming, incorporating various distance and cost functions. We preprocessed the datasets by handling missing values, encoding categorical features, removing unnecessary features, and scaling numerical features to improve model performance. The best hyperparameters were determined before evaluation. Model performance was assessed using accuracy, ROC curves, and AUROC values. KNN outperformed DT on the Heart Disease dataset, while both models showed comparable AUROC and accuracy on the Penguin dataset. To validate results, the data was split into training, validation, and testing sets.

**Introduction**

The assignment explores the implementation of two different machine learning models: K-Nearest Neighbors (KNN) and Decision Trees (DT) using class material and Kevin P. Murphy's *Probabilistic Machine Learning: An Introduction* [1]. Both models were built using object-oriented programming, allowing flexible distance functions for KNN and cost functions for DT. KNN was evaluated with Euclidean, Manhattan, and Cosine similarity, while DT was tested with misclassification rate, entropy, and Gini index. Various hyperparameters were explored to optimize performance. The models were benchmarked on the Heart Disease dataset [2], which predicts heart disease based on 13 features, and the Penguin dataset [3], which classifies penguin species using six features. Both datasets were preprocessed, including feature scaling, to improve training.

**Methods**

KNN is a machine learning model that is used for both classification and regression tasks. KNN is a lazy learner. In the training phase, KNN stores the training set as a reference, but no computations are made in this phase. In the prediction phase, KNN identifies the "k" nearest neighbours of a new datapoint. It then analyzes the most probable/common labels from the "k" neighbours, which will be the predicted label for the new datapoint. In our implementation of KNN, we opted to use a KNN, where we tie a higher weight for closer neighbours to prioritize their labels, improving prediction accuracy by giving more influence to nearby points.

DT is a machine learning model also used for classification and regression problems. DT creates a model during the training phase. In the training phase, DT recursively splits the dataset based on feature values to create subsets where target values are as pure as possible. DT uses a cost function at each node, such as misclassification rate, entropy and Gini index, to determine the best threshold and split for the children nodes. These cost functions measure the impurity of these splits. The smaller the cost function result, the better the split. This split continues until the maximum depth is reached or if we achieve a pure enough split. In the prediction phase, DT traverses the tree from the root to the leaves by evaluating the feature values of the new data point against the thresholds at each node. When a leaf is reached, the predicted value is determined by the most common class at the leaf or the average of the values at the leaf.

**Datasets**

The Heart Disease dataset contains patient information in 303 rows and 13 features, including age, sex, cholesterol, and chest pain type. The target variable indicates whether a person has a heart disease, where 0 represents no heart disease, and 1 to 4 represents a patient with heart disease. The dataset contained missing values; four were in the "ca" column, and six were in the "thal" column. We opted to drop the rows containing the missing values rather than replace them with the column's mean. Furthermore, we validated the data by ensuring no negative values in numerical data columns to maintain consistency and integrity. As for categorical columns, we confirmed no abnormally labelled categories. We performed feature scaling on the dataset as features were measured in different scales. To determine a feature's relevance, we grouped each feature into positive and negative groups, computed the feature's mean in each group and computed their mean squared difference (MSD). A high MSD suggests that the feature significantly impacts distinguishing positive and negative cases, while a lower difference indicates a minor impact. Figure 1 shows the MSD we found for our features. We opted to drop columns with a MSD of less than 0.1, taking into consideration

the domain knowledge. We also observed that 67% of the dataset were male patients, making it biased for male patients. Therefore, predictions on female patients might be skewed. As for the labels, 45% of patients had heart diseases, and 55% of patients did not have a heart disease, making it a fair distribution.

The Penguin dataset contains penguin information with 334 records and six features: sex, island, body_mass_g, flipper_length_mm, culmen_depth_mm and culmen_length_mm. This dataset aims to classify each penguin into three species: Adelie, Chinstrap, and Gentoo. The dataset contained multiple null values and required cleaning. Similarly to the preprocessing in the heart disease dataset, we dropped rows that contained null values. When validating the consistency of the dataset, we observed a row with "." in the sex column. We interpreted it as dirty and malformed data and dropped the row. When validating the other rows, we did not observe any inconsistencies. We one-hot-encoded categorical features using OneHotEncoder from Sklearn and encoded the target values using LabelEncoder. When observing the distribution of the labels, 43% of the penguins are Adelies, 35% are Gentooes, and 20% are Chinstraps. We also used a MSD approach to compute a feature's relevance. However, since we are working with a multiclass dataset, we calculated the average MSD across all classes for each feature. We observed that sex has little impact in determining a penguin's species, while the other features have a similar relevance as shown in Figure 2.

| Feature | MSD |
|---|---|
| thal | 1.116090 |
| ca | 0.863352 |
| oldpeak | 0.723620 |
| thalach | 0.722818 |
| exang | 0.714446 |
| cp | 0.672979 |
| slope | 0.446364 |
| sex | 0.312046 |
| age | 0.207497 |
| restecg | 0.111348 |
| trestbps | 0.094805 |
| chol | 0.025938 |
| fbs | 0.000040 |

*Figure 1. Feature MSD between positive and negative groups on the Heart Disease dataset*

| Feature | MSD |
|---|---|
| culmen_length_mm | 0.703579 |
| flipper_length_mm | 0.696351 |
| culmen_depth_mm | 0.661806 |
| body_mass_g | 0.649437 |
| sex_MALE | 0.000142 |
| sex_FEMALE | 0.000142 |

*Figure 2. Mean MSD between species on the Penguin Dataset*

**Results**

When comparing the AUROC of KNN and DT on the heart disease dataset, we observed an AUROC of 0.9734 for KNN with a K of 3 using Euclidean distance and 0.9022 for DT with a max depth of 3 using entropy cost. In the penguin dataset, KNN had an accuracy of 100%, while DT had an accuracy of 99%. In this experiment, we attempted different hyperparameters and functions for both, and KNN has a higher AUROC score and accuracy than DT overall. However, we will discuss this further when testing different cost functions and in the ROC curve comparison experiment.

In KNN, tuning the hyperparameter K impacts the accuracy of the model. In our experiment, we opted to test odd K values from 1 to 30 to eliminate ties when predicting. We are also using euclidean distance as our distance function. As illustrated in Figure 3, when tuning K on the heart disease dataset, our KNN's accuracy starts at 86% and increases to 93%. It peaks when K is 19 with an accuracy of 95% and drops to around 91% accuracy. However, when tuning KNN's hyperparameter on the penguin dataset, we observed an accuracy of 100% through all Ks tested as show in Figure 4.

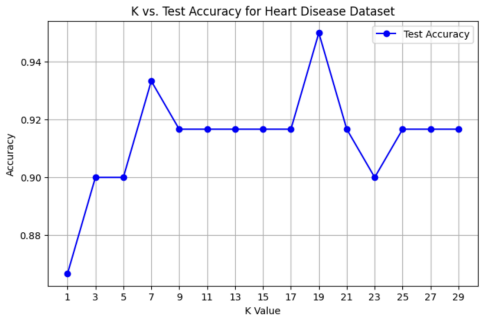


*Figure 3. Test Accuracy of KNN for Different Values of K on the Heart Disease Dataset*

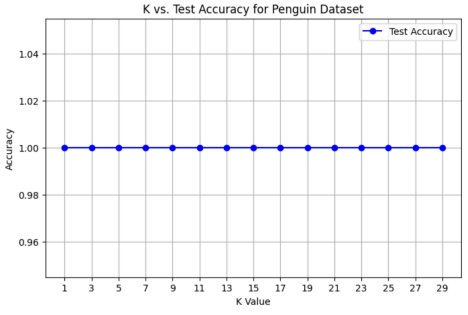


*Figure 4. Test Accuracy of KNN for Different Values of K on the Penguin Datase*

In DT, tuning the max depth also impacts the accuracy. In this experiment, we evaluate depths from 1 to 20 using the misclassification rate cost function. When tuning the max depth on the heart disease dataset, DT's accuracy starts at 69%. From depths 3 to 7, the accuracy increases to 75% and remains constant from 7 to 10. It then drops down to around 71.5%. The best depths for this experiment are 7, 8, 9 or 10, with a 75% accuracy as observed in Figure 5. As

for the penguin dataset, at depth 1, DT has an accuracy of around 81%, and drastically, it increases to 98% at depth 2. It then peaks at an accuracy of 100% from depth 10 to 12 and decreases back to 98% as observed in Figure 6.

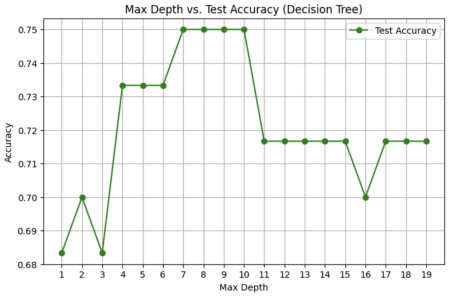


*Figure 5. Test Accuracy of DT for Different Values of Depth on the Heart Disease Dataset*

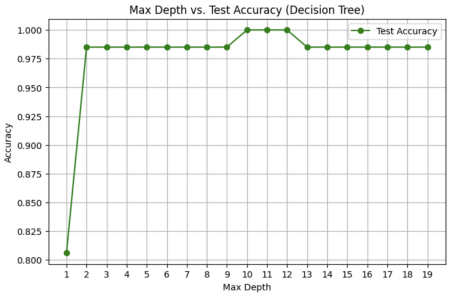


*Figure 6. Test Accuracy of DT for Different Values of Depth on the Penguin Dataset*

When testing different distance/cost functions, we decided to evaluate the function's performance on their best k. In KNN, we observed that the cosine distance function has the best accuracy at its best k of 9, with an accuracy of 97% on the heart disease dataset, as seen in Figure 7. Euclidean distance had the second-best accuracy, with an accuracy of 95% and a k of 19. Manhattan distance ranked the worst, with its best accuracy being 91.5% with a k of 23. As for the penguin dataset, all three distance functions have a best accuracy of 100% on various Ks as seen on Figure 8. However, the Manhattan distance decreases its accuracy to 98.5% when K is 23.

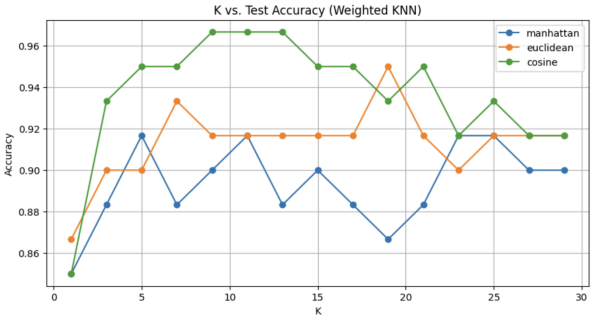


*Figure 7. Test Accuracy at different K's for KNN for Heart Disease Dataset*

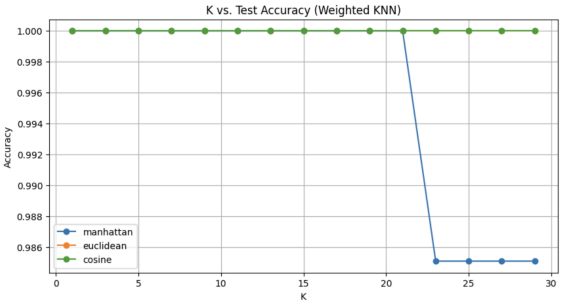


*Figure 8. Test Accuracy at different K's for KNN for Penguin Dataset*

When evaluating the different cost functions on the heart disease dataset for DT, we observe that, generally, cost entropy had the highest accuracy across various depths, as seen in Figure 9. The Gini index and entropy have the highest accuracy at a max depth of 2, with an accuracy of 87%. The misclassification rate tends to be less accurate than the other two cost functions, as seen across different depths. However, when we evaluate the cost function on the penguin dataset, we observe that all three functions have the same highest accuracy across various max depths. However, the misclassification rate tends to be less accurate across different depths, as seen in Figure 10.

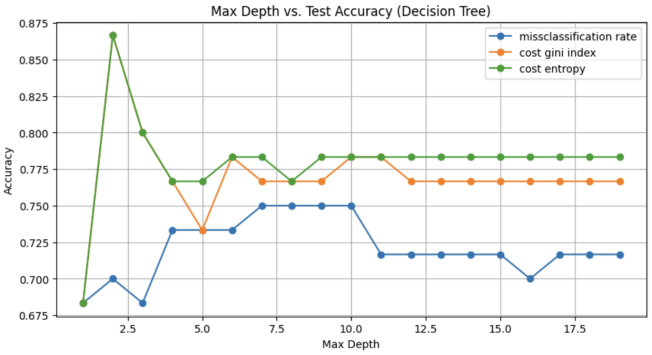


*Figure 9. Test Accuracy at different depths for DT for Heart Disease Dataset*

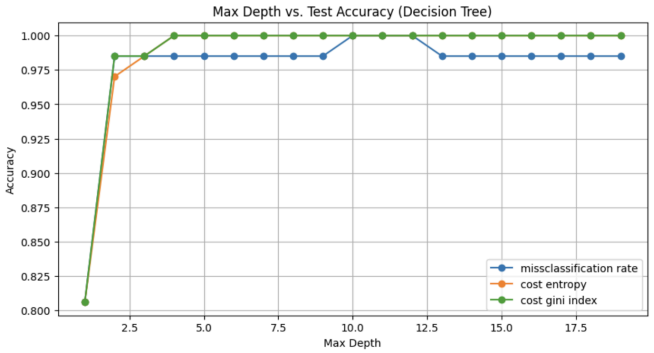


*Figure 10. Test Accuracy at different depths for DT for Penguin Dataset*

When comparing the ROC curves of both KNN and DT, we observed that KNN has a significantly higher AUROC score than DT. KNN achieves a high True Positive Rate (TPR) even at low False Positive Rates (FPR), while DT's performance degrades more quickly as FPR increases as observed in Figure 11. To compute these, we decided to use the best model found in the previous experiment. Since these models were evaluated on accuracy, it might not have the most optimised AUROC results.

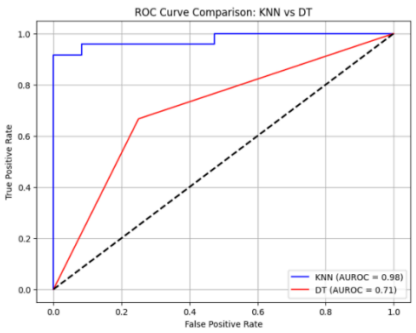


*Figure 11. ROC Curve KNN vs. DT for HT Dataset*

In order to determine feature importance used in KNN we computed the correlation of each feature with the target variable for both heart disease and penguins datasets. Features with an absolute correlation greater than 0.2 were selected, while those with a correlation of 0.2 or less were removed. The method "corrwith" implicitly uses the Pearson Correlation Coefficient to measure the linear relationship, hence the selected features are expected to have a stronger relationship with targets and will be retained for KNN, whereas the removed features showed weaker correlations and were dropped to avoid adding noise to the model. The results are described in Table 1 below. Removed features for heart disease include trestbps, chol and fbs, while for penguins the sex is of low importance.

*Table 1. Feature importance for KNN*

| | |
|---|---|
| Selected features Heart Disease (>0.2) | ['age', 'sex', 'cp', 'restecg', 'thalach', 'exang', 'oldpeak', 'slope', 'ca', 'thal'] |
| Removed Features Heart Disease (<0.2) | ['trestbps', 'chol', 'fbs'] |
| Selected features Penguins (>0.2) | ['island_Biscoe', 'island_Dream', 'island_Torgersen', 'culmen_length_mm', 'culmen_depth_mm', 'flipper_length_mm', 'body_mass_g'] |
| Removed Features Penguins (<0.2) | ['sex_FEMALE', 'sex_MALE'] |

As for the Decision Tree method, counting the number of times a feature is used on a node split is indicative of its importance since this reflects the lowest cost, and best selected feature at that time in our greedy_node algorithm. The most important features for each dataset were found to be as shown in Table 2. Our results indicate that on the first dataset, features do not have significant importance, as the most important feature 'cp' only occurs twice. However, for the penguins dataset, 'culmen_length_mm' is a very strong indicator, being used 59 times.

*Table 2. Feature importance for DT*

| | |
|---|---|
| Top 5 Heart Disease features and their count | ('cp', 2), ('ca', 1), ('thal', 1), ('thalach', 1), ('oldpeak', 1) |
| Top 5 Penguins features and their count | ('culmen_length_mm', 59), ('culmen_depth_mm', 9), ('sex_FEMALE', 6), ('flipper_length_mm', 1), ('body_mass_g', 1) |

Another important observation is that while the KNN method discarded 'sex_FEMALE', the DT method ranks it fairly highly in its top 3. Overall, the features importance varies from method to method, even though some features are found to be important in both techniques.

Our additional experiments introduce the validation set and K-fold validation. We decided to split the dataset into training, validating, and testing sets so we could test our tuned hyperparameters on unseen data and provide an unbiased estimate. K-Fold validation allows our models to be trained on as many data points as possible, improving its training. When using the validation set to test different distance/cost functions in our model we obtain similar but improved distributions from the ones in Fig. 7 & 8, as can be seen in Figure 12 for the heart disease set, and Figure 13 for the penguins set. In KNN, we observed that the cosine distance function has the best accuracy at its best k of 9, with an accuracy of 97% on the heart disease dataset, as seen in Figure 12. Euclidean distance and Manhattan had varying accuracies for different K's, with often unpredictable behavior. As for the penguin dataset, all three distance functions have a best accuracy of 100% on various Ks as seen on Figure 13. However, the Manhattan distance decreases its accuracy to 98.5% when K is 23.

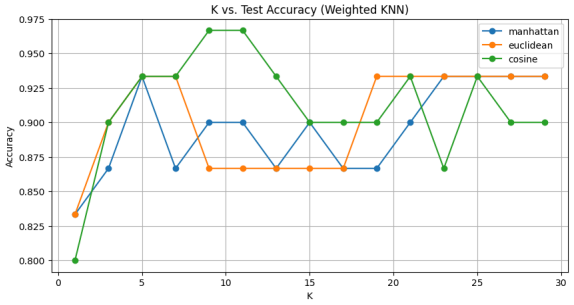


*Figure 12. Test Accuracy with Validation set at different K's for KN for Heart Disease Dataset*

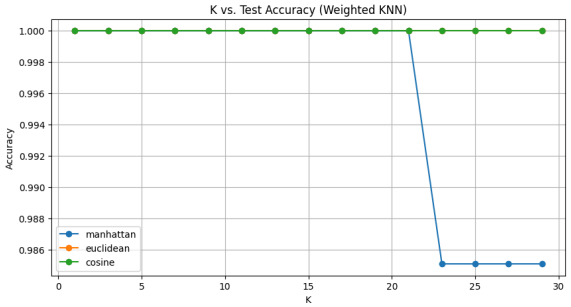


*Figure 13. Test Accuracy with Validation set at different K's for KN for Penguin Dataset*

Similarly, the introduction of the validation set slightly modifies our previous results from figures 9 and 10. Evaluating the different cost functions on the heart disease dataset for DT, we observe that, generally, cost entropy had the highest accuracy across various depths, as seen in Figure 14, while the gini index and entropy give highest accuracy (87%) at depth of 2 but steeply decrease after. The misclassification rate tends to be less accurate than the other two cost functions, as seen across different depths. However, when we evaluate the cost function on the penguin dataset, we observe that all three functions have similar, near 100% accuracy rates from dept 3 and on. Once more, the gini index and entropy tie for best cost function as can be seen in Figure 15.

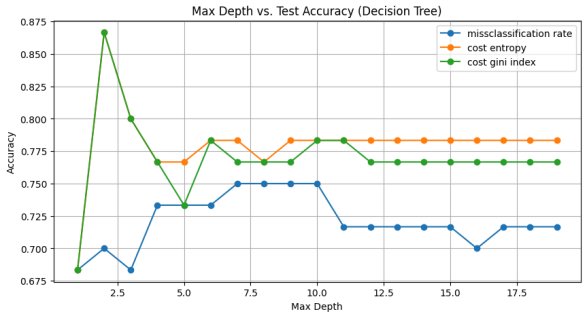


*Figure 14. Test Accuracy with Validation set at different depths for DT for Heart Disease Dataset*

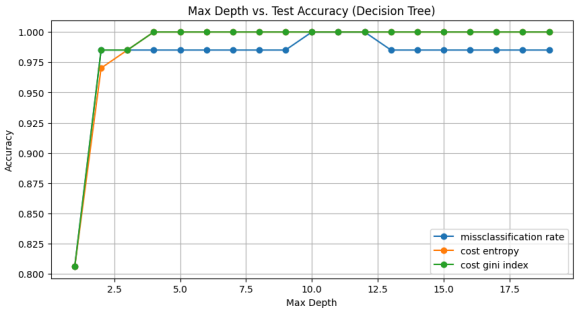


*Figure 15. Test Accuracy with Validation set at different depths for DT for Penguin Dataset*

Finally, the results for the K-fold cross-validation show that the KNN classifier with Manhattan distance on the penguins dataset achieved the highest cross-validation accuracy (99.39%), suggesting that this distance metric is highly effective for this dataset. Similarly, the KNN model using cosine similarity on the heart disease dataset performed well (85.49%), indicating that this metric captures important relationships in the data. As for decision trees, the entropy-based model for the penguins dataset achieved strong performance (97.29%), while the heart disease dataset had a lower accuracy (79.53%), likely due to the dataset's complexity or feature distribution.

**Discussion and Conclusion**

In this assignment, we implemented and evaluated two fundamental machine learning models - weighted K-Nearest Neighbors (KNN) and Decision Trees (DT) - on binary and multiclass classification tasks. Our experiments revealed that KNN generally outperformed DT on both datasets, achieving an AUROC of 0.9643 on the heart disease dataset and 100% accuracy on the penguin dataset, compared to DT's 0.9598 AUROC and 97% accuracy respectively. The choice of distance metrics significantly impacted KNN's performance, with cosine similarity proving most effective for the heart disease dataset (97% accuracy), while all distance metrics performed equally well on the penguin dataset. For DT, entropy and Gini index cost functions consistently outperformed the misclassification rate, particularly at lower tree depths. Feature importance analysis revealed interesting insights, with 'cp' being the most significant feature for heart disease classification in DT, while 'culmen_length_mm' was crucial for penguin species classification. Additional experiments revealed the improvement of both models' predictions when adding the validation set and our K-fold cross-validation results further validated these findings.

While we went beyond and implemented additional methods to validate and test the data such as k-fold validation, future investigations could explore several promising directions. First, for feature selection, implementing weighted feature importance detection algorithms could help us better understand the discrepancies between findings using KNN. Second, exploring adaptive K selection methods for KNN could help optimize the model's performance across different regions of the feature space. Third, investigating the impact of different feature scaling techniques could provide insights into model robustness. Finally, testing these implementations on larger, more complex datasets would help validate their scalability and generalization capabilities.

**Statement of Contribution**

Each team member contributed to the coding and writing part rather equally. Pair programming sessions were held where all members contributed to ensure full understanding of the models and implementation before going on to produce results (graphs) and writing the report.

## References

[1] Murphy, K. P. (2022). *Probabilistic machine learning: An introduction*. MIT Press. Retrieved from http://probml.github.io/book1

[2] Dua, D., & Graff, C. (2017). UCI Machine Learning Repository: Heart Disease Dataset. University of California, Irvine. Retrieved from https://archive.ics.uci.edu/ml/datasets/Heart+Disease

[3] Horst, A. M., Hill, A. P., & Gorman, K. B. (2020). Palmer Penguins: A dataset for data science & machine learning. Retrieved from https://allisonhorst.github.io/palmerpenguins/