Assignment1: Getting Started with Machine Learning
COMP 551

Howard Y., Negin N.
McGill University
School of Computer Sciences

January 31th, 2024

1 Abstract

In this assignment we investigated the performance of two machine learning models on two benchmark datasets, the National Health and Nutrition Examination Survey Age Prediction Subset and the Breast Cancer Wisonsin (Original) dataset. We found that the Decision Tree approach achieved worse accuracy than K - Nearest Neighbour due to the fact that K - Nearest Neighbour perform better on smaller dataset and because they are better at managing the noisy features.

2 Introduction

In this assignment, we implement and compare two fundamental classification techniques in machine learning: K-Nearest Neighbors (KNN) and Decision Trees (DT). Our objective is to gain hands-on experience with programming for machine learning, understand how to properly store the data, run the experiments, and compare different methods.

Specifically, we aim to analyze the NHANES (National Health and Nutrition Examination Survey) Age Prediction Subset, which provides a comprehensive snapshot of the health and nutritional status of individuals in the United States. In addition, we will also analyze the Breast Cancer Wisconsin (Original) dataset, which is widely used for breast cancer diagnosis and comprises of features extracted from digitized images of breast masses.

Through this assignment, we learned that the KNN algorithm performs significantly better than Decision tree algorithm when dealing with small datasets. This proves consistent with previous studies that shows that KNN is best fit for small datasets[**raikwal2012performance**]. By comparison, the decision tree algorithm often suffered from overfitting due to the low sample size of both datasets.

3 Methods

In this study we used two main algorithms which will be explained further in this section.

3.1 K nearest neighbors(KNN)

The K-Nearest Neighbors (KNN) algorithm is a widely-used technique in machine learning, based on the assumption that similar data points tend to share similar labels or values. Like many machine learning algorithms, KNN consists of two major phases: training and testing. However, KNN is considered a lazy learner because it memorizes the training data and postpones processing until classification is required during testing. During testing, it decides the label for a test data point based on the labels of its nearest neighbors in the memorized training data.
, To implement this method, we followed the Object-Oriented Programming (OOP) paradigm and defined a KNN class object with attributes such as K (number of neighbors) and the distance function. In the training phase, we simply received the training data points as inputs. During the testing phase or prediction step, we assigned the most probable class among the K nearest data points to the input test data points.

To evaluate the performance of our classification, we measured accuracy and AUC (Area Under the ROC Curve), defined as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$TPR = \frac{TP}{TP + FN}$$

$$FPR = \frac{FP}{FP + TN}$$

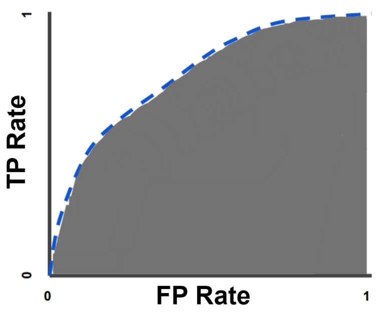


Figure 1: Area under ROC curve

3.2 Decision Tree (DT)

The decision tree algorithm is a foundational machine learning method employed for both classification and regression tasks. Its core principle involves recursively partitioning the input space into smaller regions to establish decision rules for accurately predicting the target variable. Similar to KNN, the decision tree algorithm utilizes both training and testing data points.

Decision trees possess the capability to handle missing data and exhibit robustness against outliers. However, they are susceptible to overfitting, particularly when the tree is deep and lacks proper pruning.

Following the Object-Oriented Programming (OOP) paradigm, we instantiated the decision tree class object with attributes such as DT (decision tree) and cost functions. Model training involves utilizing a predetermined number of data points, while the remaining data is allocated for testing purposes. To assess the model's performance, it is evaluated across a range of tree depths to ascertain the optimal tree complexity,balancing between model simplicity (to prevent overfitting) and predictive accuracy.

Datasets

In this study, we applied the machine learning classification techniques to two distinct health datasets: the National Health and Nutrition Health Survey 2013-2024 (NHANES) Age prediction Subset [**misc_national_health_and_nutrition_h** and the Breast Cancer Wisconsin dataset[**misc_breast_cancer_wisconsin_(original)_15**]. During the pre-processing stage, rigorous data cleaning and standardization procedures were employed to rectify missing or malformed features and ensure compatibility across datasets.

The NHANES data set encompasses 7 features, 1 value response corresponding to the respondent's age group (senior/non-senior), comprising 2278 instances. It was designed to collect extensive health and nutritional information from a diverse U.S. population, and the selected features were hypothesized to have strong correlations with age. By ranking the squared difference of the group means, LBXGLT emerged as the top feature associated with the target variable, reflecting the respondent's oral health status. Additionally, data points with erroneous values were systematically excluded, notably instances where values fell outside permissible boundaries. For instance, a data point featuring a value of 7 for the "PAQ605" parameter was deemed invalid, as this parameter only admits values of 1 or 2. One ethical concern can arise if the demographic data is used to discriminate individuals predicted to be older based on their health conditions, which can have an effect on employment or insurance.

The Breast Cancer Wisconsin dataset encompasses 9 features, 1 value response determining tumor malignancy, encompassing 699 instances. The instances were classified based on the 9 features, valued from a scale of 1 to 10, collectively contributing to the correct classification of tumors as benign or malignant. Utilizing a same methodology of ranking based on squared differences of group means, features such as uniformity of cell shape, uniformity of cell size, and bare nuclei emerged as primary predictors associated with the target variable. Notably, despite slight variations, the dataset demonstrated a relatively small disparity among features, with 8 out of 9 features exhibiting squared differences exceeding 2. Mitoses emerged as the sole exception, with a squared difference of 0.79, indicating its comparatively weaker association with the target variable. One ethical concern can arise if the dataset is not representative of the broader population, or if it contains biases, which can lead to unfair or discriminatory outcomes.

Results

3.3 Results of different settings for KNN models

3.3.1 Cleaning of outliers and duplicates

Outliers and duplicates are widely recognized as detrimental to the machine learning process due to their potential to significantly impact model performance and compromise the generalization ability of algorithms. Outliers have the capacity to distort model training, influence decision boundaries, and bias parameter estimates. Similarly, duplicates introduce redundant information, leading to overfitting and biased training.

In the datasets analyzed in this assignment, we observed outliers in the NHANES dataset but not in the BCW dataset. Conversely, we identified 236 duplicated data points in the BCW dataset, equivalent to 33.76

To assess the impact of removing outliers and duplicates, we constructed KNN models with five different K values and the Euclidean distance function, once on the datasets including outliers or duplicates, and once again on the same data after outlier and duplicate removal. The resulting test accuracy and AUC are presented in Table 1 and 2, and Figures 2 to 6.

3.3.2 Significant Features Selection

In this section, we examined the impact of feature selection. As KNN is known to be sensitive to noisy features, incorporating numerous features unrelated to our labels can diminish the model's performance. To identify significant features, we employed two methods. Firstly, we assessed the squared difference between the means of each feature for the positive and negative groups. To ensure comparability across features, we standardized them before calculation, although this did not alter the key features. Figures 4 and 5 depict the key features of the NHANES and BCW datasets based on this metric. Secondly, we evaluated the correlation between each feature and the target classes, aiming to identify features with minimal correlation to the targets. Figures 6 and 7 illustrate the outcomes of this approach on the NHANES and BCW datasets. It is evident from the figures that both methods yielded similar suggestions for both datasets. Consequently, we opted to exclude the Mitoses feature from the BCW dataset, while the RIAGENDR feature in the NHANES dataset appeared to exhibit slight correlation with the corresponding targets and some capability to differentiate between positive and negative classes. Figures 8 and 9 shows the effect of omitting this feature on the test accuracy and AUC.

3.3.3 Effect of the parameter K

We tested the KNN model on both datasets for 9 different K values using a single distance function of ecludcian. Figures 10 and 11 and table 5 and 6 show the effect of K on the test accuracy and AUC.

3.3.4 Choosing different distance functions

In this section, we implemented 5 different distance function including Manhattan, Euclidean, Minkowski (p = 3), Cosine dissimilarity and Correlation. We tested each of the functions on our model, considering a constant k = 10 which according to previous parts seemed to have a good performance. Figures 12 and 13 and tables 7 and 8 show the effect of K on the test accuracy and AUC.

3.3.5 Simultaneous effect of the parameters

To see what how the two parameters K and distance function can effect our KNN model testing score, we plotted the accuracy and AUC metrics for a wide range of K values ($1 <= K <= 200$) and all the five distance functions. The results on both datasets are presented in figures 10 and 11.

3.3.6 Weighted KNN

As we see in the previous sections, KNN performed poorly on the NHANES dataset. Our hypothesis was that this is due to due fact that this dataset has uneven distribution of data for two classes. As we mention, NHANES has only 364 points with the class label senior while it has 1914 points for the Adult class. This is not the same with BCW because in this dataset two classes were well-balanced. Weighted kNN can potentially perform better on datasets with imbalanced classes because weighted k-NN assigns different weights to each neighbor based on their distances, giving more influence to closer neighbors. Therefor, in this section we used weighted KNN with $weights = \frac{1}{distances}$ to allow for unevenly distributed dataset to perform better in terms of AUC and accuracy metrics. The results on both datasets are presented in tables 9, 10 and figures 16.

3.4 Results Decision Tree models

3.4.1 NHANES Dataset

3.4.2 Performance of decision tree algorithm on a 50/50 train/test split

When evaluating the performance of the decision tree algorithm on a 50/50 training and testing split on the NHANES dataset, we generated the AUC curve of the model using 3 different cost functions: misclassfication cost function, gini index cost function, and the entropy cost function.

As shown in Figure 18 found in the appendix, our observation revealed initial similarities in model accuracy across all cost functions, followed by a consistent decline as soon as a tree depth of 4. While the model's accuracy decline suggests that the model is overfitting the data, early signs of overfitting suggest that the dataset is small, which causes the decision tree to memorize the training instances rather than generalize new instances. In Figure 17, we see a substantial discrepancy in the AUC values between the misclassification cost function and the Gini index and entropy counterparts, which signifies that the classes are imbalanced. This is further substantiated by an examination of the dataset's distribution, revealing 1914 instances classified as adults and 364 as seniors.

Despite its better accuracy, the misclassification-based model struggles to properly classify minority instances, resulting in a lower AUC value. Therefore, while accuracy may appear high, the model's performance on minority classes remains inadequate due to the majority class bias. In such case, gini index and entropy will provide better discrimination as it is more sensitive to the distribution of classes and the structure of the decision tree. Hence, we chose to proceed with the gini index due to the misclassfication's propensity to favor the majority class. In the future, using the precision-recall metric would be beneficial to measure the performance on imbalanced data.

3.4.3 Hyperparameter tuning using validation test

To optimize hyperparameter tuning and prevent overfitting, we use validation test on the decision tree model. While we still split the dataset into the training subset and the testing subset, we will separate half of the test data into a validation subset. In this experiment, the training set is used to build the decision tree, while the validation set evaluates its effectiveness in predicting outcomes on unseen data. This process helps gauge the model's generalization ability, ensuring it doesn't overfit to the training data.

As shown in figure 19 and 20, we ran 2 different validation tests of splits 80-10-10 and 70-15-15 depicting the tree depth with the highest accuracy as well as the train curve and the validation curve. We notice that the validation curve and the tree curve deviate starting from tree depth of 5, which indicates that the model is overfitting. In our experiment, the validation test split of 70-15-15 came out with a marginally higher accuracy of 85.35% and a tree depth of 3. Despite the use of validation tests, we were unable to meaningfully address overfitting in the decision tree.

3.4.4 feature importance score

By performing a depth-first search, we computed a rough feature importance score by counting the number of the non-leaf node where a feature is used. For the NHANES dataset, we found out that the top five most commonly used features are (in descending order): BMXBMI with 662 instances, RIAGENDR with 66, LBXGLU with 57, PLQ605 with 47,LBXGLT with 33. Despite LBXGLT having the biggest mean difference as explained in the dataset section, BMXBMI came out on top by a wide margin. This suggests that LBXGLT does not have as strong of a correlation as we have previously thought, and that BMXBMI has highest impact on the model's predictive capabilities. With such disparities, the effect that BMXBMI has on the decision tree model needs to analyze more thoroughly in the future.

3.4.5 Breast Cancer Wisconsin Dataset

3.4.6 Performance of decision tree algorithm on a 50/50 train/test split

Similarly to the NHANES dataset, when evaluating the performance of the decision tree algorithm on a 50/50 training and testing split on the Breast Cancer Wisconsin dataset, we generated the AUC curve of the model using 3 different cost functions: misclassfication cost function, gini index cost function, and the entropy cost function.

As shown in figure 21 and 22, our observation revealed the gini index cost function performed better than the other 2 both in terms of average accuracy as well as AUC. After removing 263 duplicate instances from the dataset, a 33% decrease in sample size (from 699 instances to 463), we noticed in figure 23 and 24 that the curves of all cost function are now relatively similar. Additionally, all curves stabilize sooner than if the duplicates remained. This signifies that the model is learning from a more focused and relevant subset of the data, and that the duplicates present in the dataset affect the model's prediction for both the entropy and misclassification cost function. With the dropped duplicates, the new accuracy and AUC graphs indicate that the model's prediction are reliable and balanced across various evaluation criteria. With both the highest AUC curve and accuracy curve, we chose the gini index for the rest experiment. However, it is important to note that the sample size of the dataset is significantly smaller than before, which can lead the model to quickly overfitting the dataset.

3.4.7 Hyperparameter tuning using validation test

To optimize hyperparameter tuning and prevent overfitting, we use validation test on the decision tree model. Similarly to the experimentation made on the NHANES dataset, the training set is used to build the decision tree, while the validation set evaluates its effectiveness in predicting outcomes on unseen data. However, due to the dataset having considerably lower sample size, we avoided splits where test samples and validation samples where less than 15%.

As such, we ran 2 different validation tests of splits 60-20-20, 70-15-15 as seen in figure 25 and 26, depicting the tree depth with the highest accuracy as well as the train curve and the validation curve. We notice that the validation curve and the tree curve deviate starting from tree depth of 6, which indicates that the model is overfitting very early. In our experiment, the validation test split of 70-15-15 came out with a higher accuracy of 93.75% and a tree depth of 5. However, we also notice that the testing accuracy is significantly lower than the validation curve, which indicates a small training size, suggesting that the model is not able to learn enough patterns to generalize well to unseen data.

Unfortunately, due to the datasets small sample size, we are not able to increase the training set as the validation set and test set will be too small. As shown in figure 27 and figure 28, performance estimates at a 80-10-10 split and 90-5-5 split are unreliable and highly sensitive to random fluctuations in the data due to small validation and testing set samples. In further experiments, techniques like k-fold cross-validation should be used for model evaluation as it leverages the available data more effectively, reducing the variability associated with small validation and testing sets.

3.4.8 feature importance score

Similarly to the feature importance score in the NHANES dataset, we performed a depth-first search to compute a rough feature importance score by counting the number of the non-leaf node where a feature is used. For the Breast Cancer Wisconsin dataset, we found out that the top five most commonly used features are (in descending order): clump thickness with 60 instances, uniformity of cell size with 45, uniformity of cell shape with 45, marginal adhesion with 26, and single epithelial cell size with 19. In this case, the top features all have similar importance, signifying that the feature distribution of the model is relatively uniform, contributing to the model's robustness and reliability.

3.5 Comparison between KNN and DT

According to the experimental results described above, we manually chose the settings with the "best" test accuracy and AUC for both datasets. For both datasets, KNN with cosine disimilarity distance outperforms Decision Tree with gini index when $15 <= K <= 60$. We picked K = 60 for NHANES and K = 25 for BCW as NHANES was a much larger dataset. For DT the best parameters chosen was d = 3 using gini-index cost function with NHANES and d = 5 using gini-index cost function for BCW. The results on both datasets are presented in tables 9, 10 and figures 12, 13.

4 Discussion and Conclusion

4.1 BCW has better accuracy and AUC than NHANES

Both KNN and decision trees (DT) showed better performance on the Breast Cancer Wisconsin (BCW) dataset compared to the NHANES dataset for several reasons. Firstly, the BCW dataset had a better class distribution, which allowed both KNN and decision trees to make more accurate predictions without being biased towards the majority class. Additionally, the features found in BCW demonstrate stronger correlations with the target variable, providing more discriminative power for both algorithms. In contrast, NHANES features had much lower correlations between target variable, which hindered predictive performance on both algorithms. Furthermore, the bigger feature set in BCW provides both KNN and DT with more information to distinguish patterns within the data, enhancing their predictions. These factors all improved the performance of both KNN and DT on the BCW dataset compared to NHANES.

4.2 KNN has better performance than Decision Tree

As shown in Figure 9 and 10, we also concluded that KNN performed better than DT for both dataset for several reasons. Firstly, KNN is often favored over decision Trees for smaller datasets such as the NHANES dataset and the BCW dataset due to its ability to capture complex decision boundaries with fewer instances. In fact, our results showed that decision quickly overfitted both dataset while KNN was not affected as much. Additionally, KNN performed better in the presence of noise, as it considers multiple neighboring points rather than relying on a single split as in DT. Overall, KNN's local instance-based approach and noise tolerance make it a suitable choice for both small datasets and noise-prone datasets.

4.3 Weighted KNN not able to improve model performance on NHANES

As we mentioned our hypothesis for KNN poor performance on the NHANES dataset that this dataset has uneven distribution of data for two classes. We assumed that Weighted kNN can perform better than KNN on this datset because of it's imbalanced classes. However, figure 16 shows that the performance got even worse. This can be due to the fact that this dataset behaves very nonlinear in features space. It can be like predicting grains of salt among the other class because of both being less in amount and to sparse. The majority of neighbor for each point has the opposite class making for weighted KNN difficult to make the correct prediction.

5 Statement of Contributions

Teammate workload was seperated equally, with Howard working on the decision tree section and Negin working on the KNN section.

Appendix

5.1 KNN Appendix

	k = 1	k = 2	k = 5	k = 10	k = 20
Accuracy outliers removed	79.1%	73%	83.4%	83.8%	83.9%
Accuracy main data	78.2%	72.1%	83.1%	83.6%	84.7%

Table 1: Test accuracy measurements on NHANNES before and after outlier removal

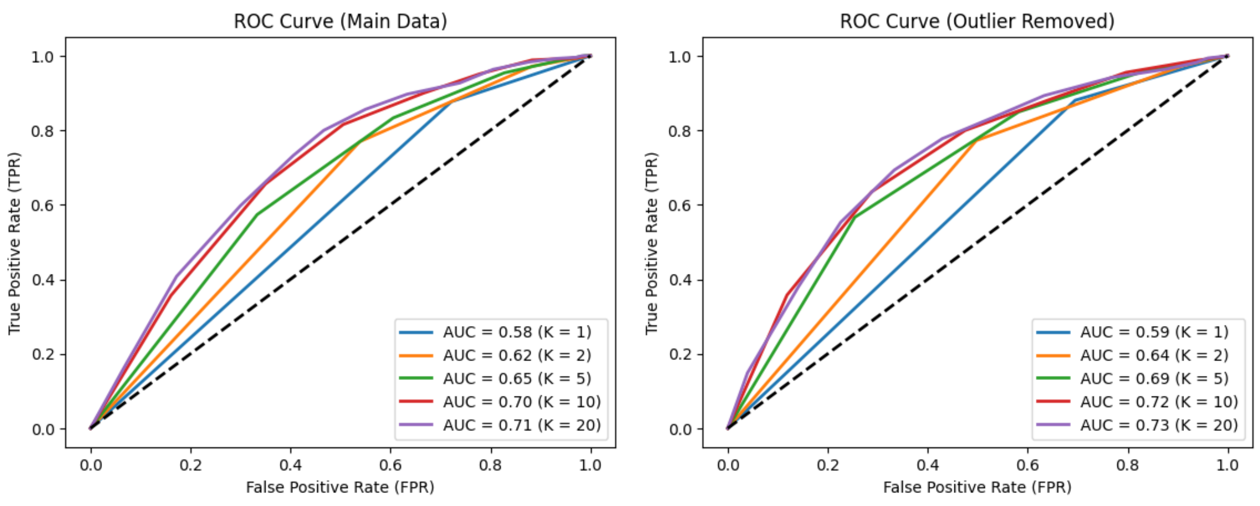


Figure 2: ROC plotting and AUC measurements on NHANNES before and after outliers removal

	k = 1	k = 2	k = 5	k = 10	k = 20
Accuracy duplicates removed	91.8%	90.9%	93.5%	92.7%	93.1%
Accuracy main data	95.1%	92.9%	96.3%	96.3%	96.6%

Table 2: Test accuracy measurements on BCW before and after duplicates removal

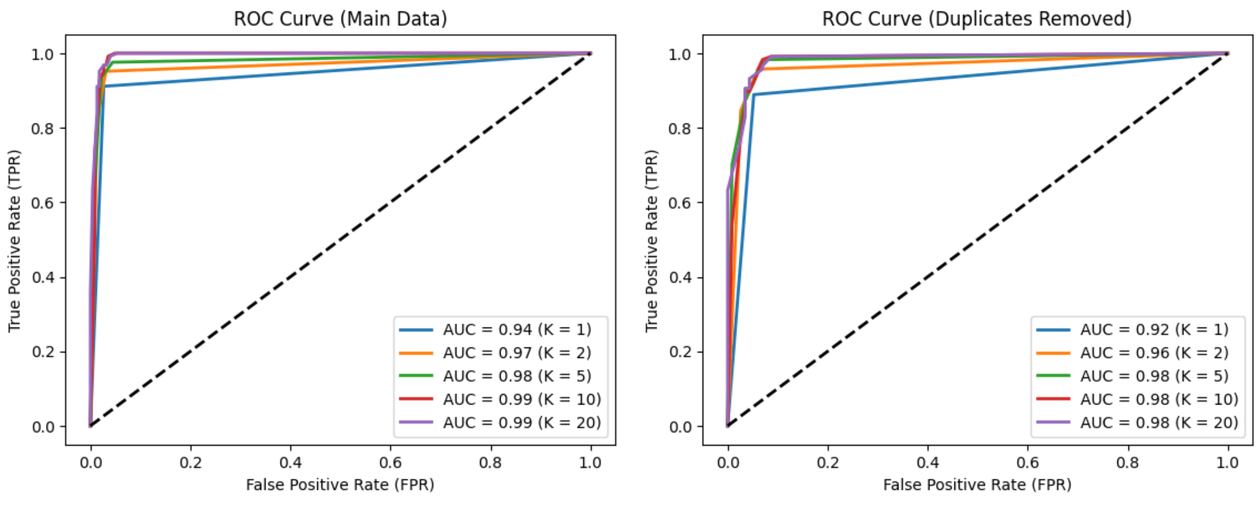
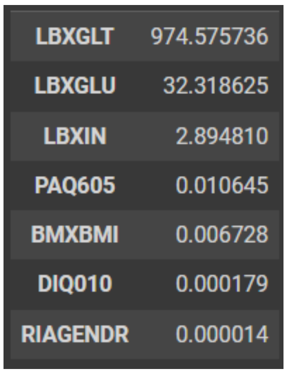
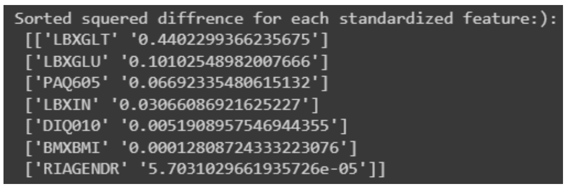


Figure 3: ROC plotting and AUC measurements on BCW before and after duplicates removal



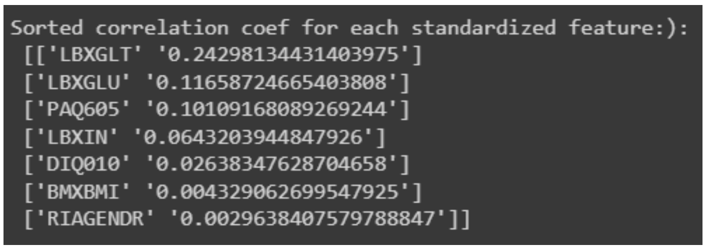
LBXGLT	974.575736
LBXGLU	32.318625
LBXIN	2.894810
PAQ605	0.010645
BMXBMI	0.006728
DIQ010	0.000179
RIAGENDR	0.000014

Figure 4: Squared difference between the mean of each feature of NHANNES for the positive group and the negative group



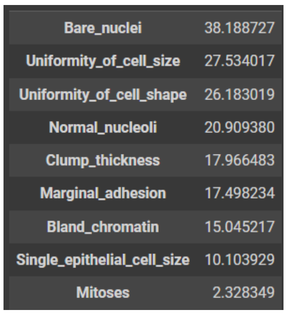
```
Sorted squered diffrence for each standardized feature:):
 [['LBXGLT' '0.4402299366235675']
 ['LBXGLU' '0.10102548982007666']
 ['PAQ605' '0.06692335480615132']
 ['LBXIN' '0.03066086921625227']
 ['DIQ010' '0.005190895754694435']
 ['BMXBMI' '0.00012808724333223076']
 ['RIAGENDR' '5.7031029661935726e-05']]
```

(a) Standarized squared difference between the mean of each feature of NHANNES for the positive group and the negative group



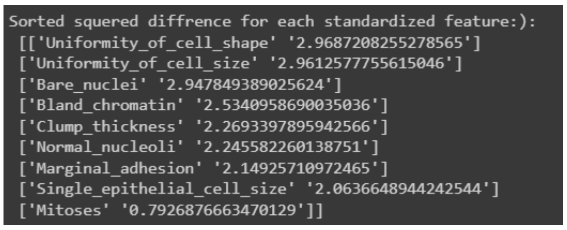
```
Sorted correlation coef for each standardized feature:):
 [['LBXGLT' '0.24298134431403975']
 ['LBXGLU' '0.11658724665403808']
 ['PAQ605' '0.10109168089269244']
 ['LBXIN' '0.0643203944847926']
 ['DIQ010' '0.02638347628704658']
 ['BMXBMI' '0.004329062699547925']
 ['RIAGENDR' '0.0029638407579788847']]
```

(b) Correlation of each each feature of NHANNES with labels



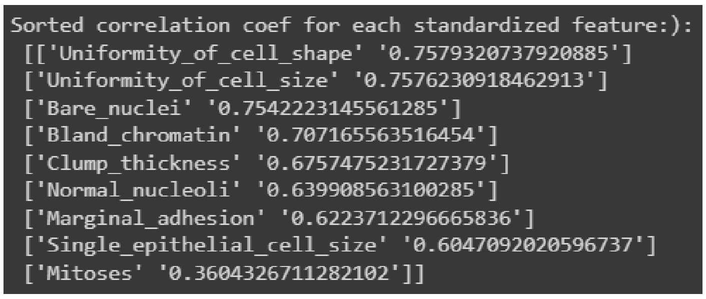
Bare_nuclei	38.188727
Uniformity_of_cell_size	27.534017
Uniformity_of_cell_shape	26.183019
Normal_nucleoli	20.909380
Clump_thickness	17.966483
Marginal_adhesion	17.498234
Bland_chromatin	15.045217
Single_epithelial_cell_size	10.103929
Mitoses	2.328349

Figure 6: Squared difference between the mean of each feature of BCW for the positive group and the negative group



```
Sorted squered diffrence for each standardized feature:):
 [['Uniformity_of_cell_shape' '2.9687208255278565']
 ['Uniformity_of_cell_size' '2.9612577755615046']
 ['Bare_nuclei' '2.947849389025624']
 ['Bland_chromatin' '2.5340958690035036']
 ['Clump_thickness' '2.2693397895942566']
 ['Normal_nucleoli' '2.245582260138751']
 ['Marginal_adhesion' '2.14925710972465']
 ['Single_epithelial_cell_size' '2.0636648944242544']
 ['Mitoses' '0.7926876663470129']]
```

(a) Standarized squared difference between the mean of each feature of BCW for the positive group and the negative group



```
Sorted correlation coef for each standardized feature:):
 [['Uniformity_of_cell_shape' '0.7579320737920885']
 ['Uniformity_of_cell_size' '0.7576230918462913']
 ['Bare_nuclei' '0.7542223145561285']
 ['Bland_chromatin' '0.707165563516454']
 ['Clump_thickness' '0.6757475231727379']
 ['Normal_nucleoli' '0.639908563100285']
 ['Marginal_adhesion' '0.6223712296665836']
 ['Single_epithelial_cell_size' '0.6047092020596737']
 ['Mitoses' '0.3604326711282102']]
```

(b) Correlation of each each feature of BCW with labels

	k = 1	k = 2	k = 5	k = 10	k = 20
Accuracy after feature selection	77.5%	70.9%	83.4%	84.7%	84.1%
Accuracy before feature selection	79.1%	73%	83.4%	83.8%	83.9%

Table 3: Test accuracy measurements on NHANNES before and after feature removal

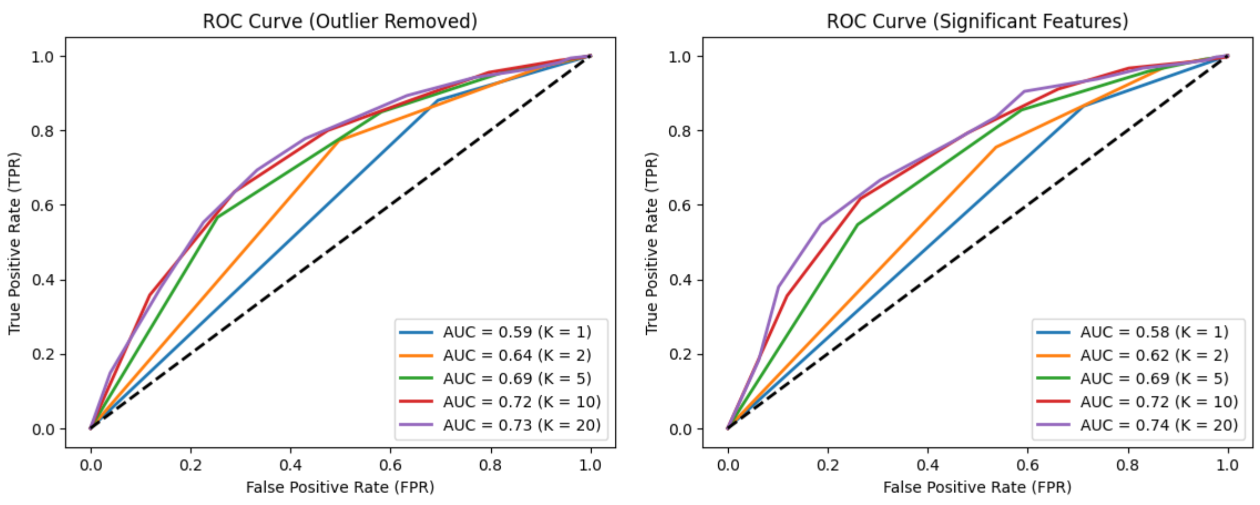


Figure 8: ROC plotting and AUC measurements on NHANNES before and after feature removal

	k = 1	k = 2	k = 5	k = 10	k = 20
Accuracy after feature selection	92.7%	91.4%	93.5%	93.1%	94.0%
Accuracy before feature selection	91.8%	90.9%	93.5%	92.7%	93.1%

Table 4: Test accuracy measurements on BCW before and after feature removal

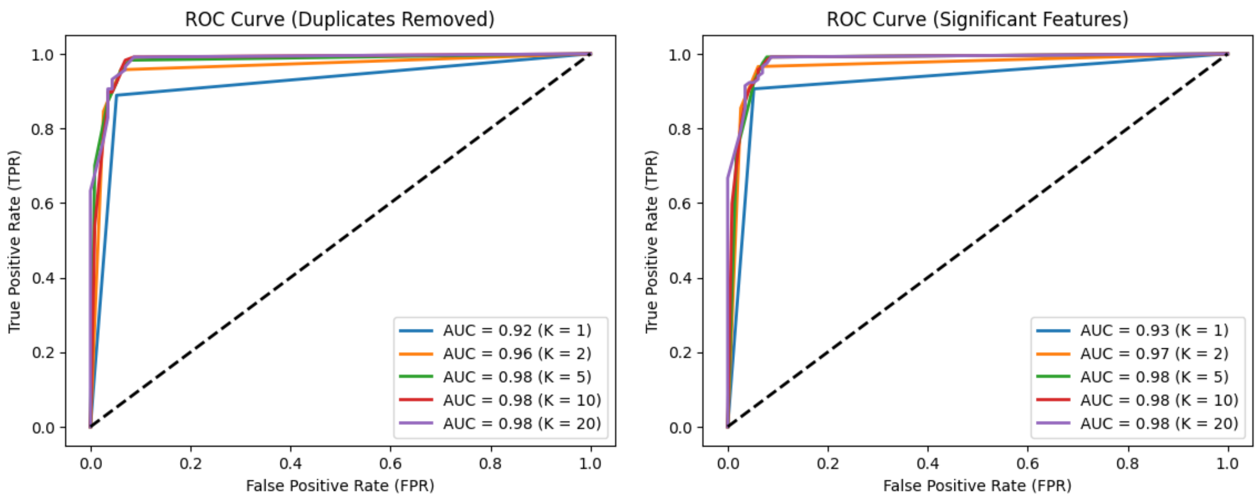


Figure 9: ROC plotting and AUC measurements on BCW before and after feature removal

	k = 1	k = 2	k = 5	k = 10	k = 20	k = 50	k = 80	k = 100	k = 200
Accuracy	77.5%	70.9%	83.4%	84.7%	84.1%	84.5%	84.5%	84.5%	84.5%

Table 5: Test accuracy measurements on NHANNES for different K

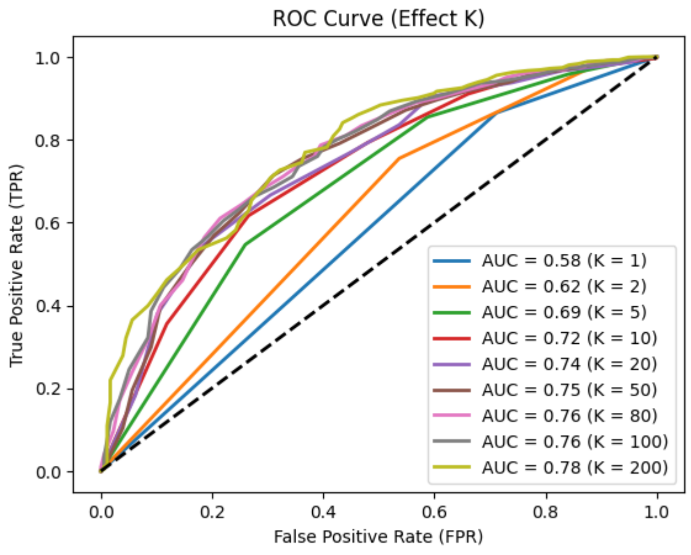


Figure 10: ROC plotting and AUC on NHANNES for different K

	k = 1	k = 2	k = 5	k = 10	k = 20	k = 50	k = 80	k = 100	k = 200
Accuracy	92.7%	91.4%	93.5%	93.1%	94.0%	93.1%	90.9%	90.9%	85.8%

Table 6: Test accuracy measurements on BCW for different K

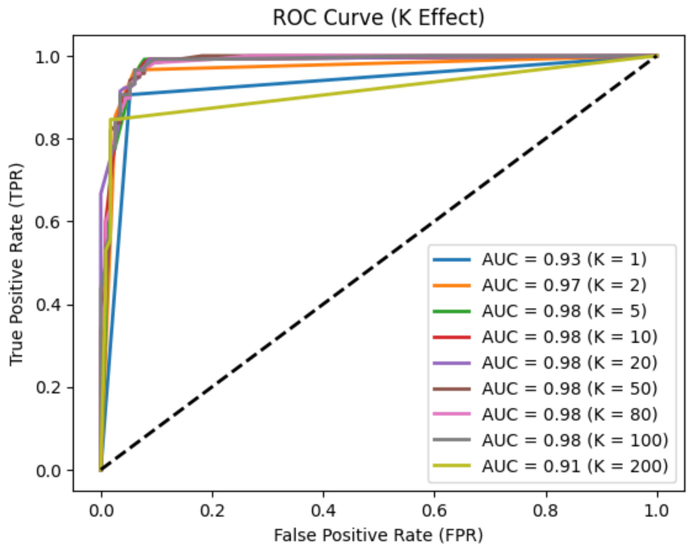


Figure 11: ROC plotting and AUC on BCW for different K

	manhattan	euclidean	minkowski	cosine	correlation
Accuracy	84.0%	84.7%	84.5%	85.3%	84.9%

Table 7: Test accuracy measurements on NHANNES for different distance functions (k = 10)

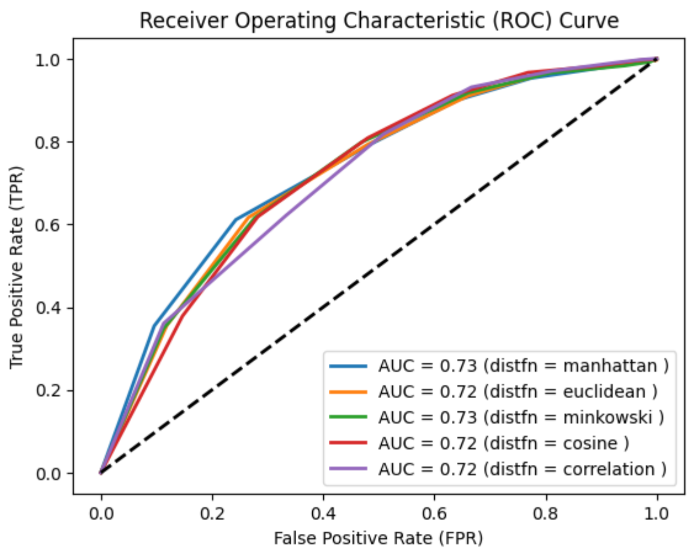


Figure 12: ROC plotting and AUC on NHANNES for different distance functions (k = 10)

	manhattan	euclidean	minkowski	cosine	correlation
Accuracy	93.5%	93.1%	92.7%	94.4%	79.3%

Table 8: Test accuracy measurements on BCW for different distance functions (k = 10)

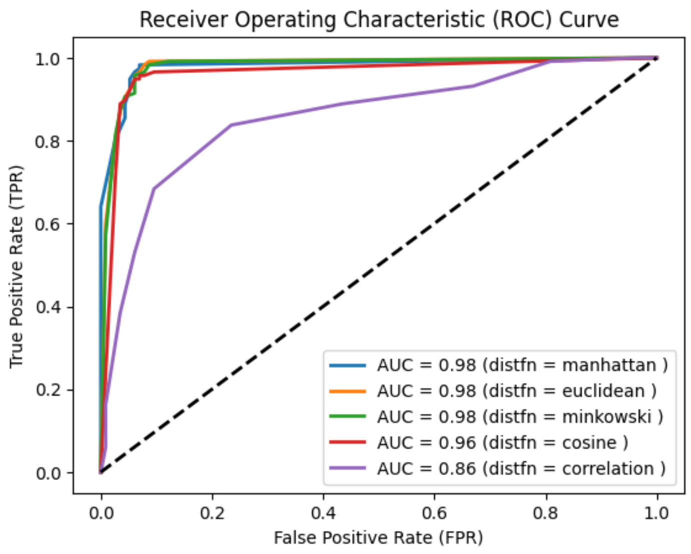


Figure 13: ROC plotting and AUC on BCW for different distance functions (k = 10)

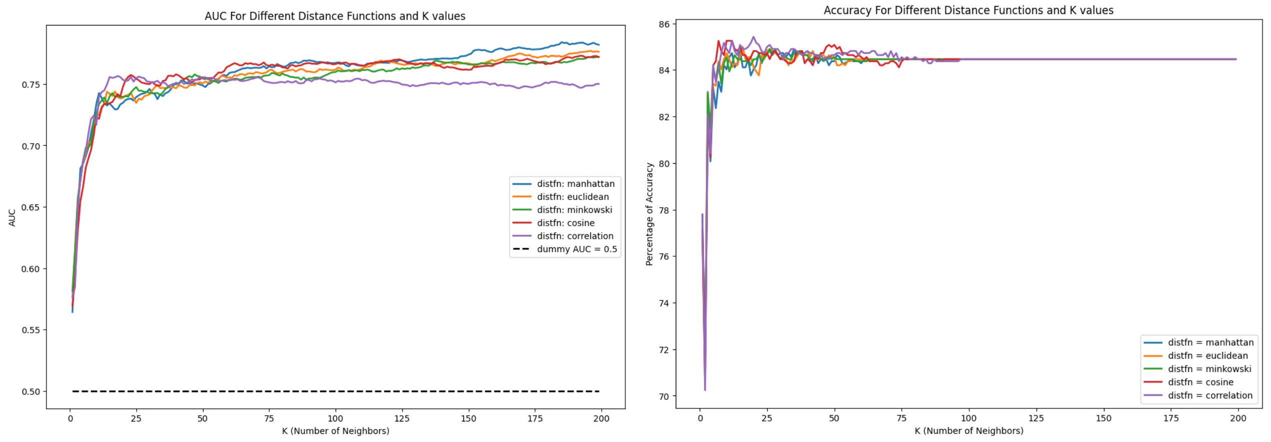


Figure 14: AUC measurements and accuracy scores as a function of K for 5 distance functions on NHANNES dataset

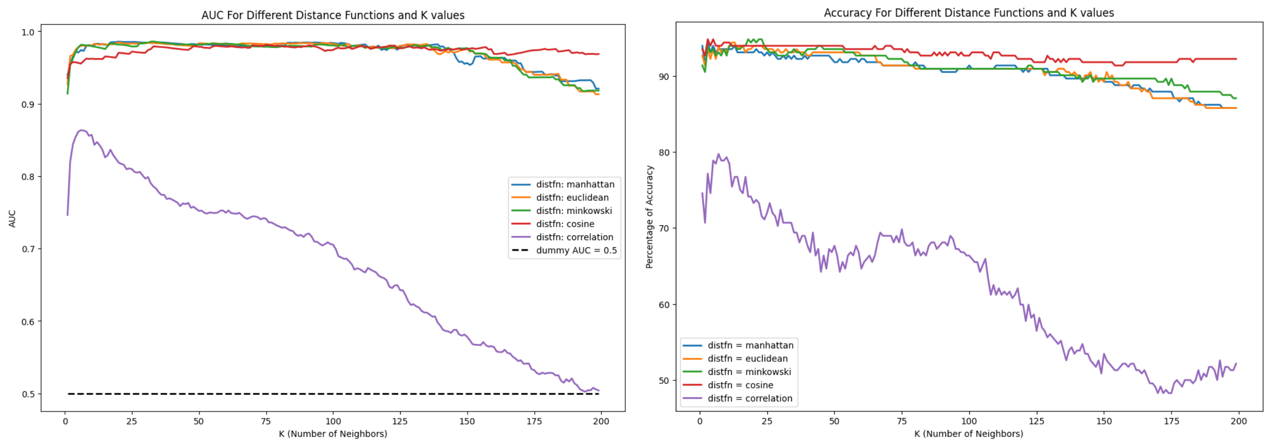


Figure 15: AUC measurements and accuracy scores as a function of K for 5 distance functions on BCW dataset

	k = 1	k = 2	k = 5	k = 10	k = 20	k = 50	k = 80	k = 100	k = 200
Accuracy	77.5%	77.5%	83.8%	84.5%	84.6%	84.6%	84.5%	84.4%	84.5%

Table 9: Test accuracy measurements on NHANNES for different K using weighted KNN

	manhattan	euclidean	minkowski	cosine	correlation
Accuracy	84.5%	84.5%	84.3%	84.5%	84.0%

Table 10: Test accuracy measurements on NHANNES for different distance functions using weighted KNN

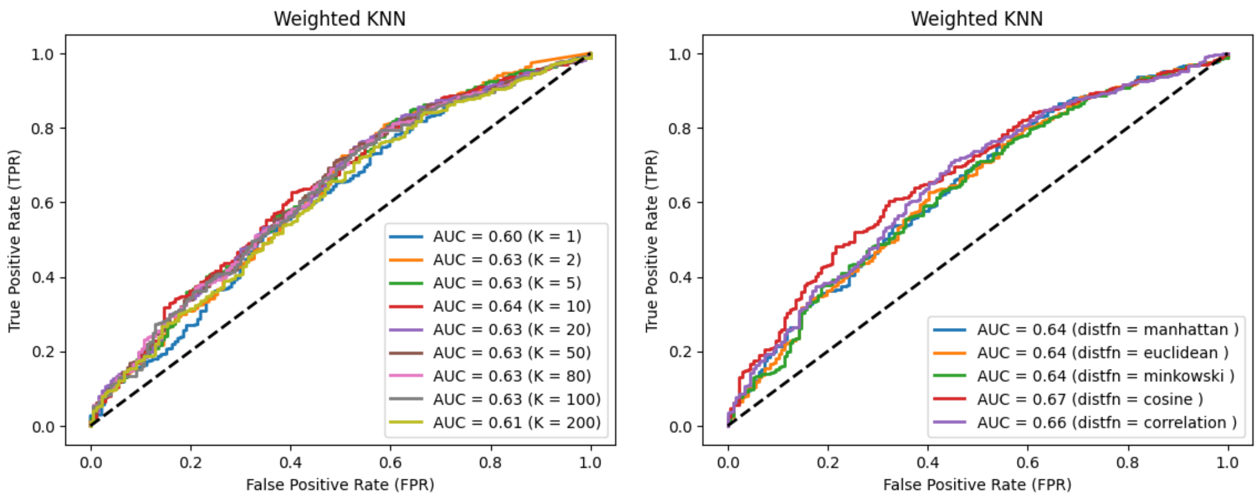


Figure 16: ROC plotting and AUC measurements on NHANNES using weighted KNN (for the left figure euclidean is used as a distance function and for the right figure K is set to 10)

5.2 Decision Tree Appendix

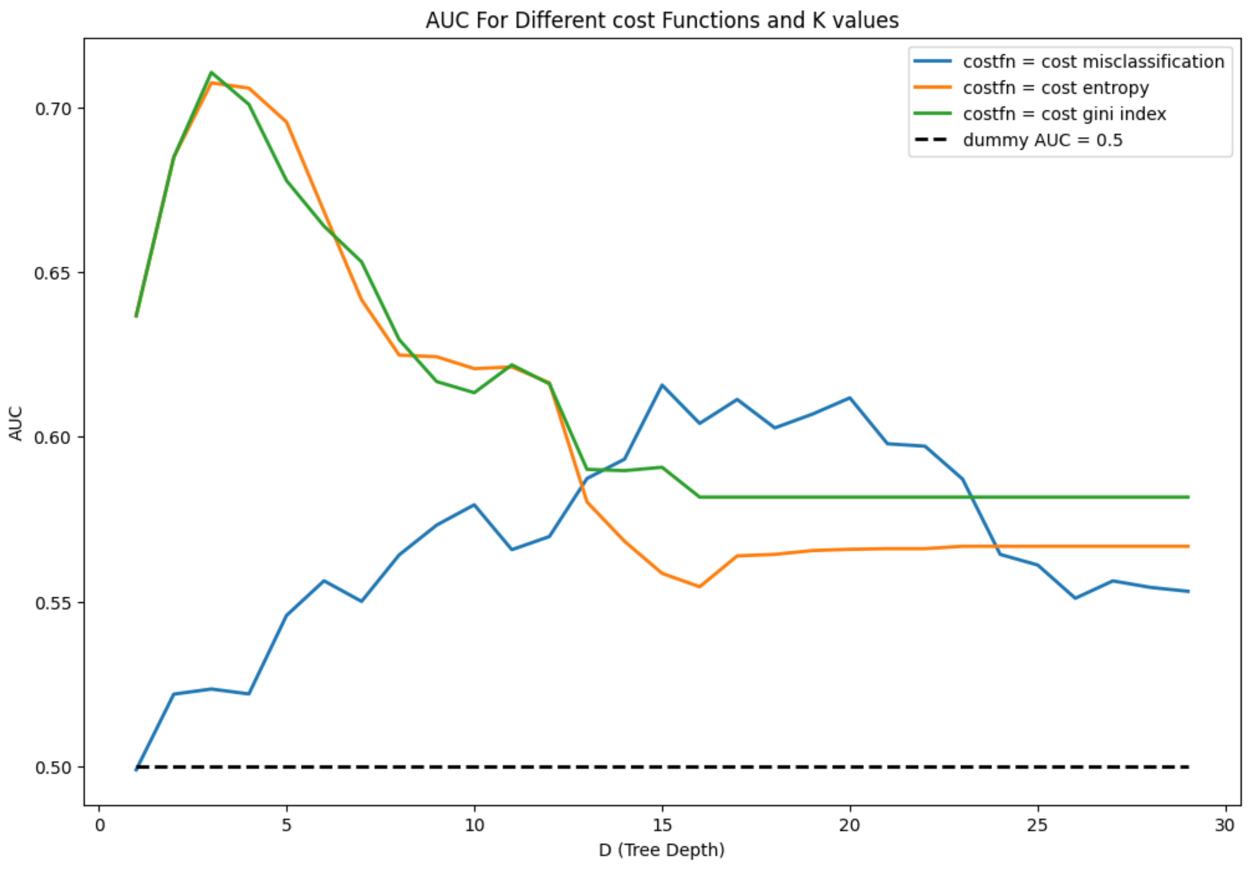


Figure 17: AUC curves for all cost functions for NHANES dataset

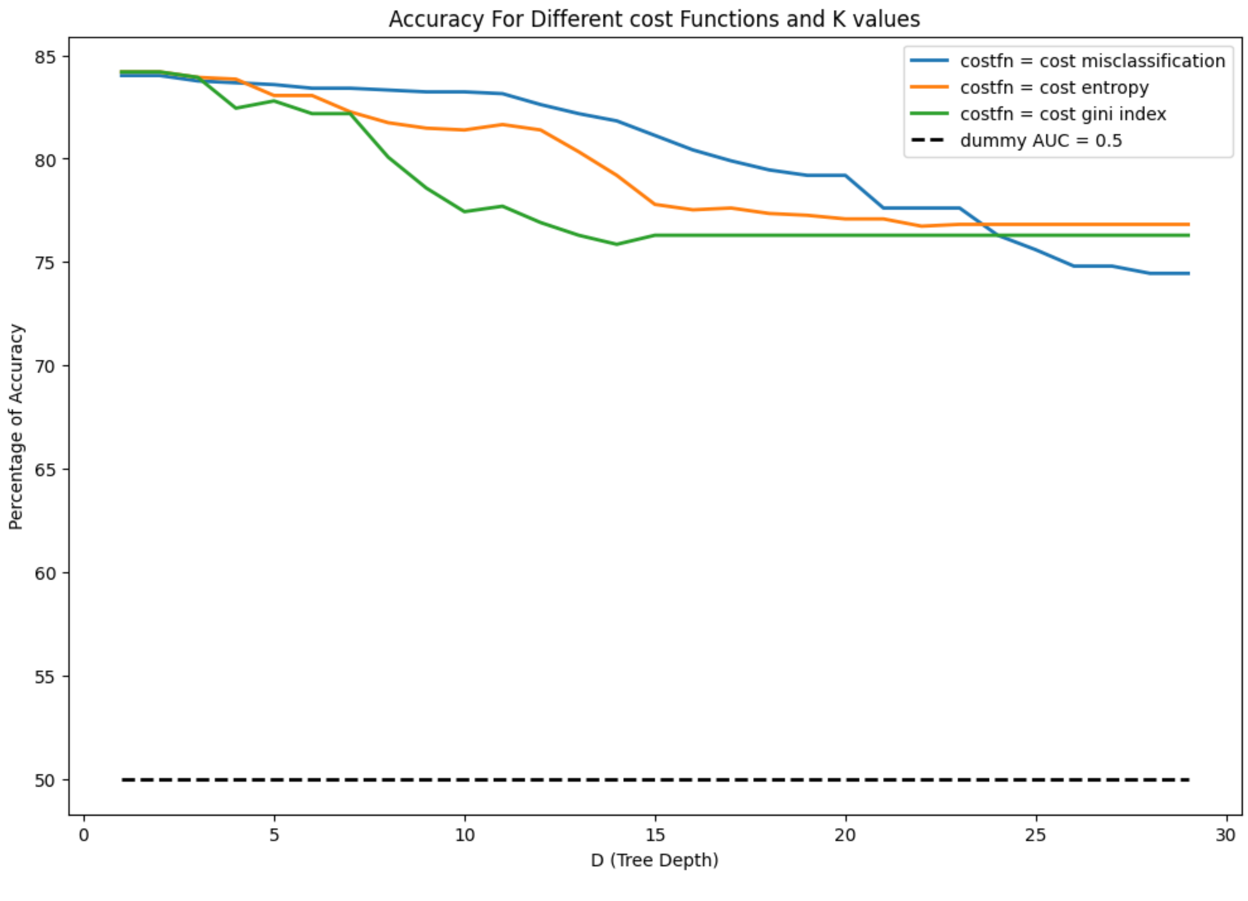


Figure 18: accuracy for all cost functions for NHANES dataset

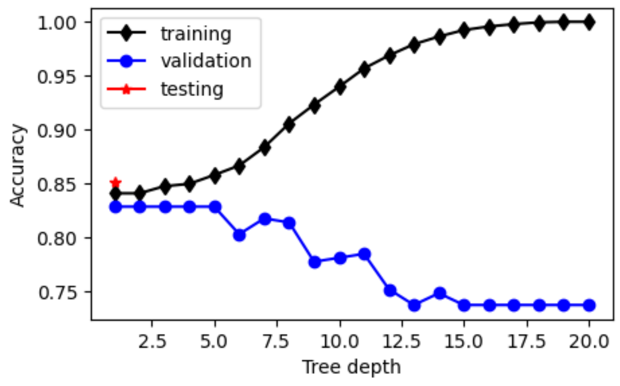


Figure 19: NHANES dataset validation test with 80% train 10% validation and 10% testing with an accuracy of 85.16%

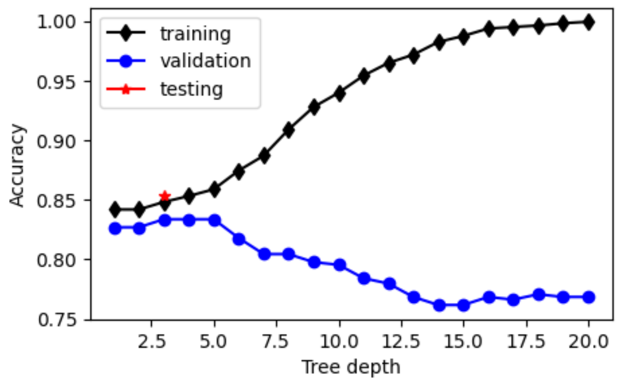


Figure 20: NHANES dataset validation test with 70% train 15% validation and 15% testing with an accuracy of 85.35%

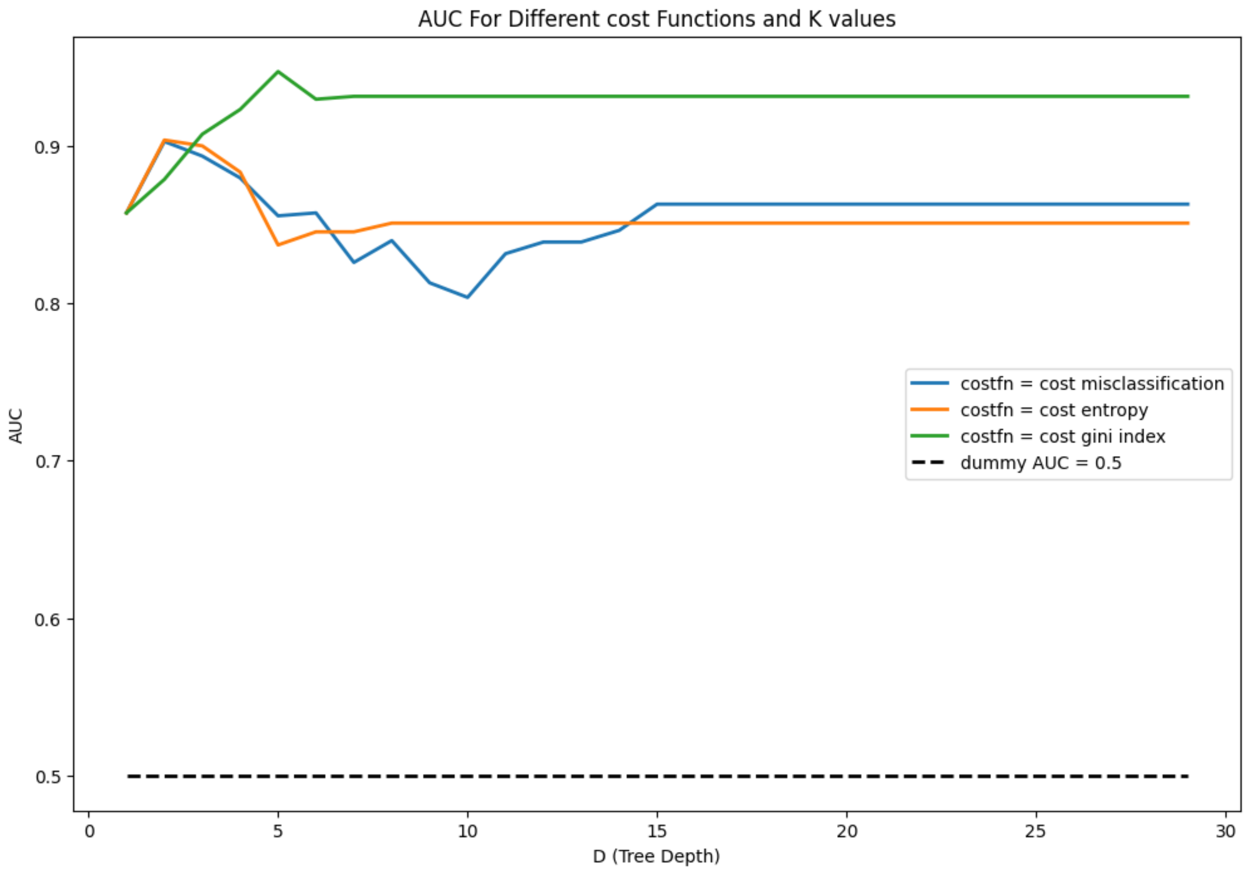


Figure 21: Breast Cancer Wisconsin AUC curve with duplicate

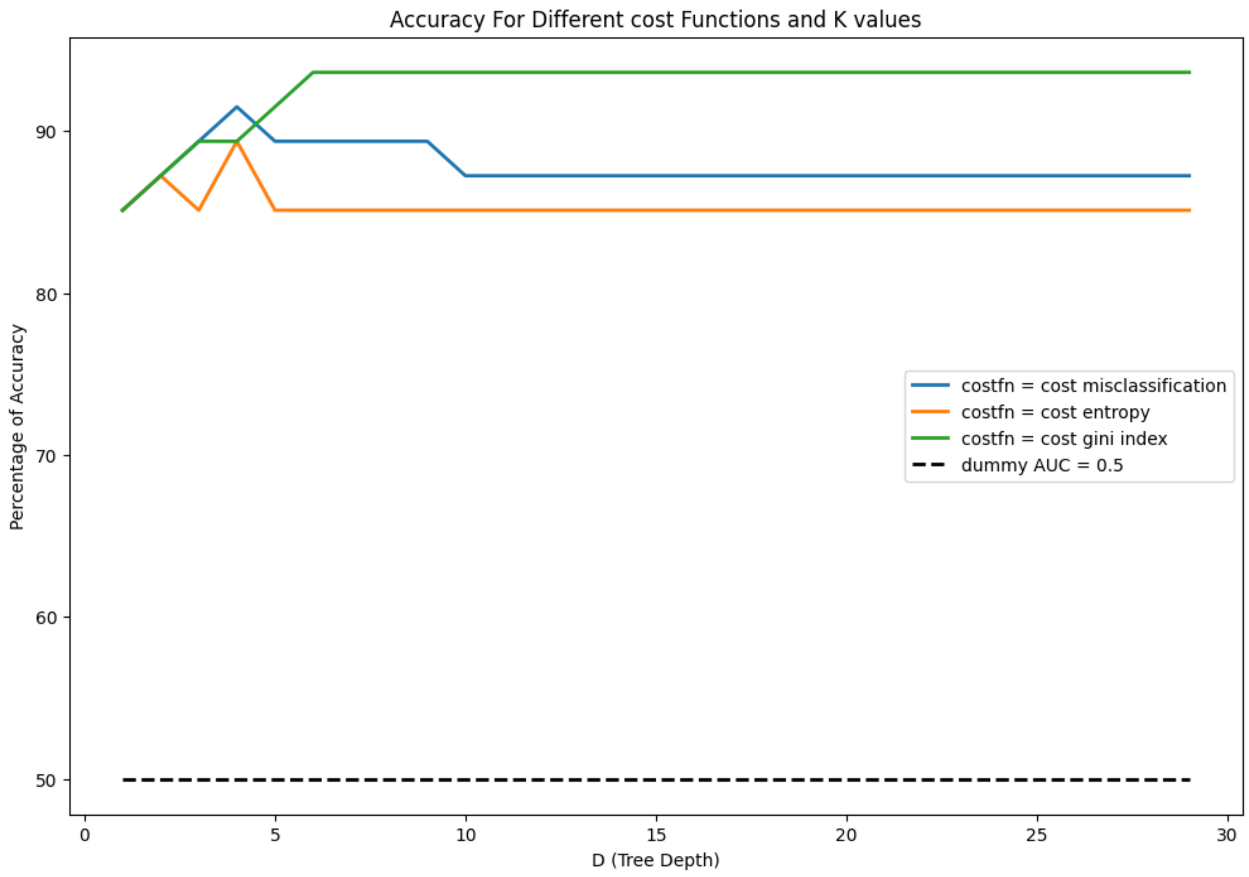


Figure 22: Breast Cancer Wisconsin accuracy curve with duplicate

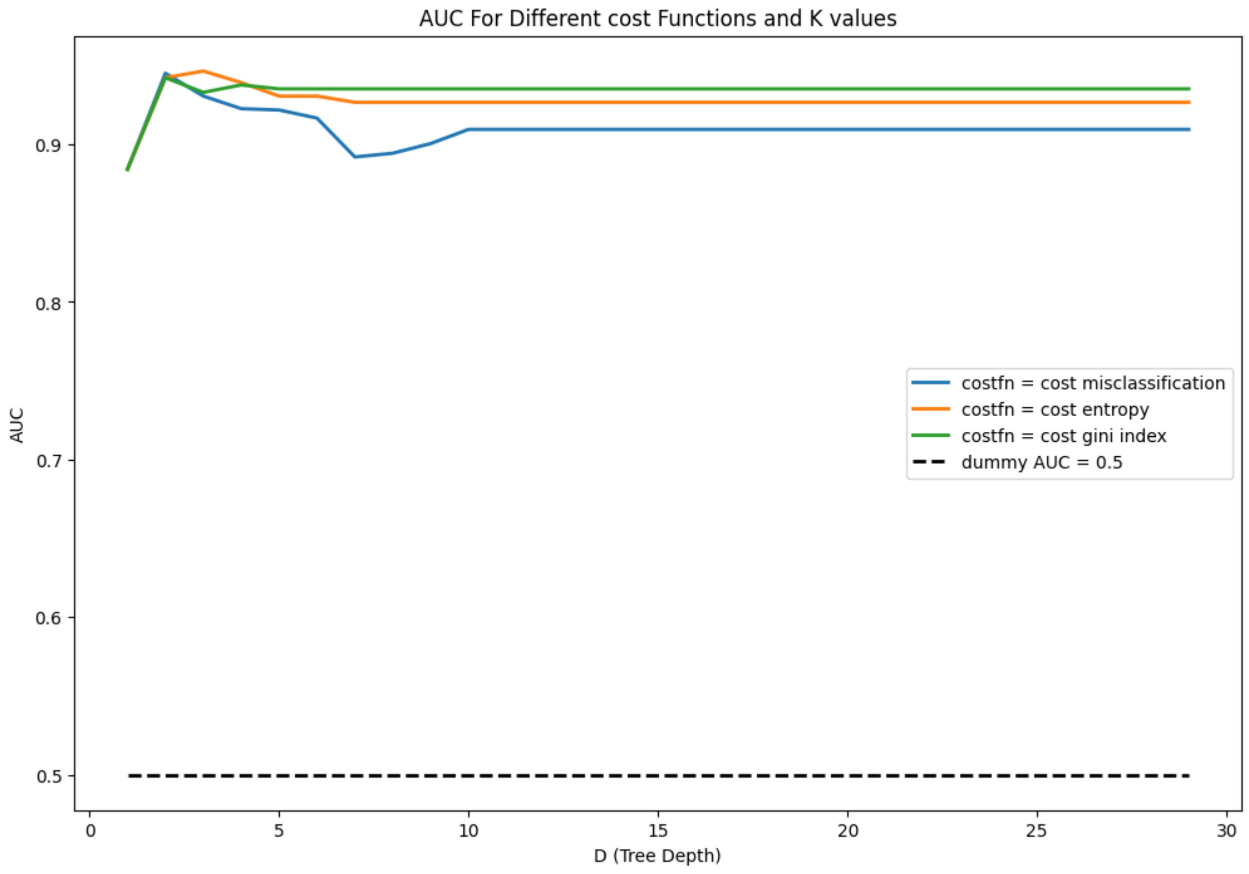


Figure 23: Breast Cancer Wisconsin AUC curve with no duplicate

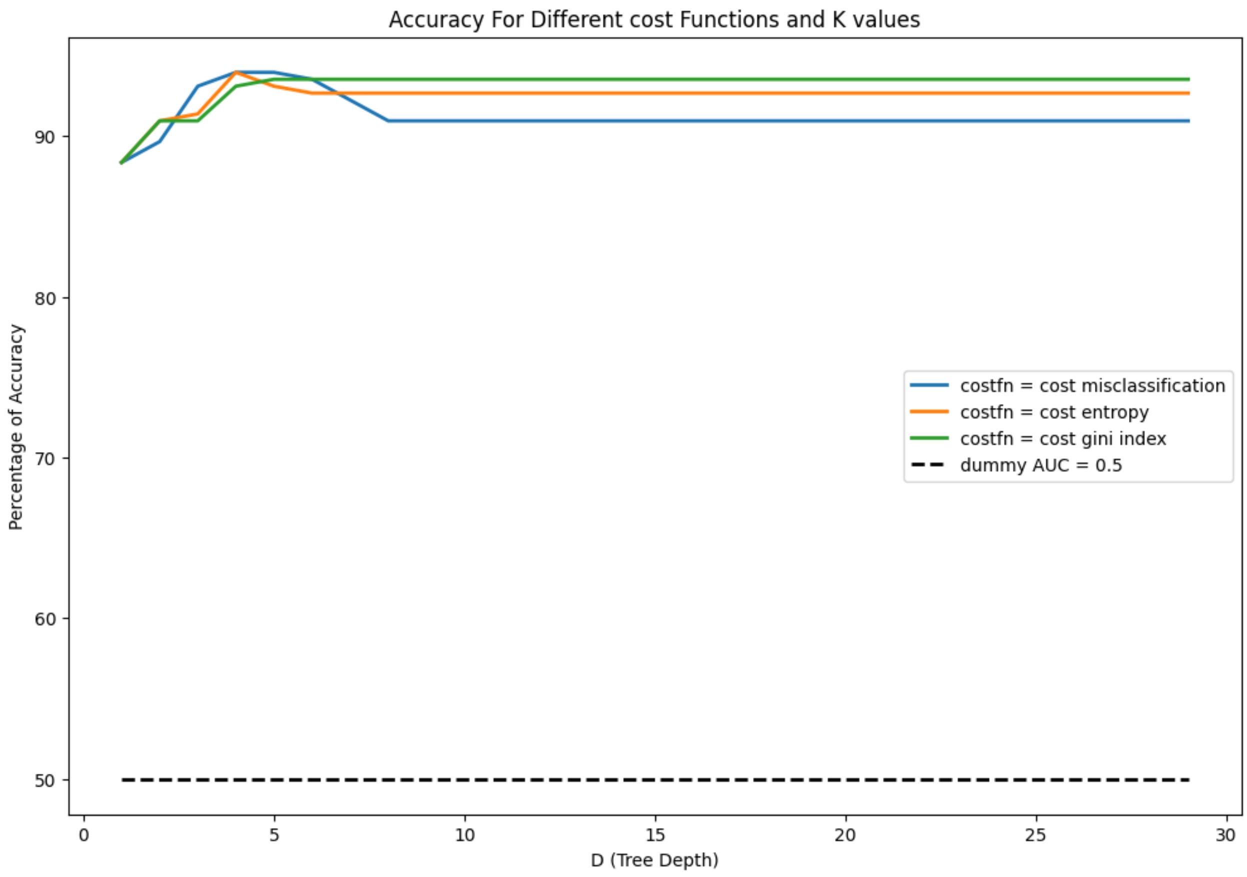


Figure 24: Breast Cancer Wisconsin accuracy curve with no duplicate

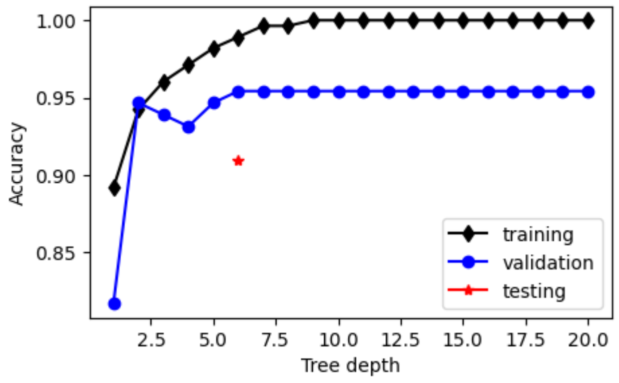


Figure 25: Breast cancer Wisconsin validation test with 80% train 10% validation and 10% testing with an accuracy of 85.16%

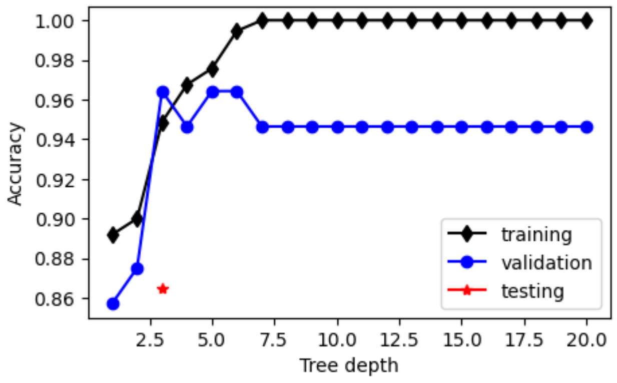


Figure 26: Breast cancer Wisconsin validation test with 90% train 5% validation and 5% testing with an accuracy of 85.29%

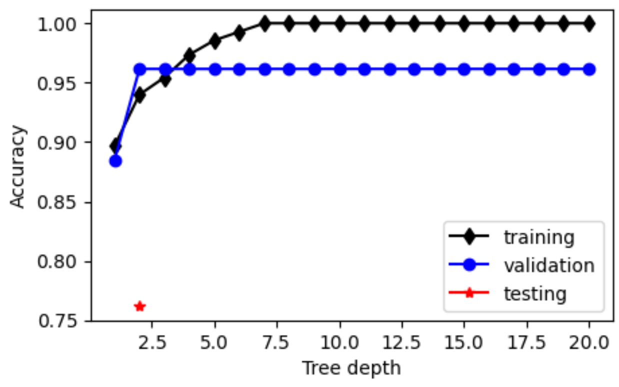


Figure 27: Breast cancer Wisconsin validation test with 90% train 5% validation and 5% testing with an accuracy of 85.16%

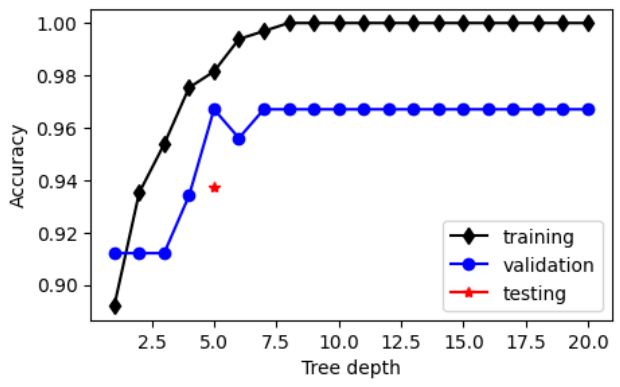


Figure 28: Breast cancer Wisconsin validation test with 90% train 5% validation and 5% testing with an accuracy of 85.29%