k-NN Algorithm Report

Jess Nguyen 000747411

# Data Description

Data set that was used in this test is Website phishing data <https://archive.ics.uci.edu/ml/datasets/Website+Phishing>. It contained 9 features name: 'SFH', 'popUpWindow', 'SSLfinal\_State', 'Request\_URL', 'URL\_of\_Anchor', 'web\_traffic', 'URL\_Length', 'age\_of\_domain', and 'having\_IP\_Address'. The result of each row would be classified as -1, 0, or 1 which transcribes to be Phishy, Suspicious, and Legitimate. These number and names are also used in the columns data.

# The Tests

## Part 1: k-Nearest Neighbour

I consistently choose the same k value of 4, 8, and 12 for each parameter. The Euclidean distance with the 3 different k values, Euclidean distance with z-score and then Manhattan distance with z-score.

For each kNN algorithm I had them in a method that took in parameters. Euclidean method had taken in a row from the testing data, the whole training data, the whole training label, and a k value. With those values it would get the closest point distance for that testing row from the training data and find the highest number of points with the same label to also label that testing label the same one.

Euclidean distance with z-score would use the Euclidean method but have the dataset be adjusted. Before using the Euclidean running it through the z-score algorithm. The z-score will take the mean of data (testing and training), subtract the data mean with data and then divide by the standard deviation.

Finally, for the Manhattan distance with z score I created a method to take in the testing data row, training data, training labels and a k value. Inside the method is similar to the Euclidean but it has a different distance algorithm. Manhattan will take the absolute of test data row subtracting training data then sum it.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Average | Run 1 | Run 2 | Run 3 | Run 4 | Run 5 |
| K=4, Euclidean Distance | 87.75 | 88.46 | 88.17 | 88.46 | 85.21 | 88.46 |
| K=8, Euclidean Distance | 87.34 | 87.28 | 86.09 | 87.87 | 86.69 | 88.76 |
| K=12, Euclidean Distance | 86.39 | 84.32 | 87.57 | 87.28 | 84.62 | 88.17 |
| K=4, Euclidean Distance, Normalized z-score | 84.97 | 84.62 | 84.91 | 85.5 | 83.14 | 86.69 |
| K=8, Euclidean Distance, Normalized z-score | 85.98 | 86.69 | 84.32 | 85.8 | 84.32 | 88.76 |
| K=12, Euclidean Distance, Normalized z-score | 84.56 | 84.91 | 83.73 | 82.84 | 82.25 | 89.05 |
| K=4, Manhattan Distance, Normalized z-score | 87.51 | 88.46 | 86.98 | 89.05 | 86.98 | 86.09 |
| K=8, Manhattan Distance, Normalized z-score | 87.81 | 90.83 | 84.62 | 87.28 | 87.87 | 88.46 |
| K=12, Manhattan Distance, Normalized z-score | 87.63 | 90.83 | 84.32 | 88.76 | 88.46 | 85.8 |

## Chart, bar chart Description automatically generated

## Part 2: Decision Trees

I choose 4 different parameter max depth, criterion, min sample leaf, and max leaf nodes. For max depth I used 4 and 20, criterion gini and entropy, min sample leaf of 5 and 40, max leaf node of 4 and 40.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Average | Run 1 | Run 2 | Run 3 | Run 4 | Run 5 |
| Max Depth: 4, ENTROPY, Min Sample Leaf: DEFAULT, Max Leaf Nodes: DEFAULT | 82.13 | 84.02 | 84.02 | 81.66 | 81.66 | 79.29 |
| Max Depth 20, GINI, Min Sample Leaf: DEFAULT, Max Leaf Nodes: DEFAULT | 87.04 | 86.69 | 87.87 | 86.69 | 86.39 | 87.57 |
| GINI, Max Depth: DEFAULT, Min Sample Leaf: DEFAULT, Max Leaf Nodes: DEFAULT | 88.4 | 88.17 | 87.87 | 89.64 | 88.46 | 87.87 |
| ENTROPY, Max Depth: DEFAULT, Min Sample Leaf: DEFAULT, Max Leaf Nodes: DEFAULT | 87.93 | 86.98 | 88.17 | 87.87 | 87.28 | 89.35 |
| Min Sample Leaf: 5, ENTROPY, Max Depth: DEFAULT, Max Leaf Nodes: DEFAULT | 87.45 | 89.35 | 86.09 | 85.5 | 87.57 | 88.76 |
| Min Sample Leaf: 40 GINI, Max Depth: DEFAULT, Max Leaf Nodes: DEFAULT | 82.01 | 84.02 | 81.07 | 81.66 | 79.88 | 83.43 |
| Min Sample Leaf: 40 GINI, Max Depth: DEFAULT, Max Leaf Nodes: DEFAULT | 81.48 | 80.47 | 81.95 | 81.36 | 80.47 | 83.14 |
| Max Leaf Nodes: 40, GINI, Max Depth: DEFAULT, Min Sample Leaf: DEFAULT | 90.06 | 86.39 | 89.35 | 91.72 | 92.6 | 90.24 |

Diagram

Description automatically generated

# Discussion

**PART 1:**

There isn’t a clear winner in kNN since all the averages are around 80%. But looking closer with the runs in the report I see that `K=8, MANHATTAN DISTANCE, NORMALIZED Z-SCORE` has the best average. The worst algorithm results are Euclidean with z-score. I experimented with running the program multiple times and the results of which algorithm is best differs between Euclidean distance and Manhattan distance with z-score. This could mean the z-score work with Manhattan algorithm instead of the Euclidean. Therefore, I would recommend this configuration for any future kNN usage with this dataset.

**PART 2:**

The clear winner in Decision Trees was setting the max depth to 40. It was consistently getting an average of around 90%. The highest average I’ve seen with max depth 40 was 93%. Decision Trees performed better since I was able to can a consistent average of around 90% compared to kNN of around 86-87%.

After testing with parameters and value in Decision Trees I found some sweet spots in the values. Max depth 40 was one of them. If the number extended passed 40 it seems like it was stretching and caused it to overfit to the predictions. In most cases, it seemed better to slightly generalize. I would also recommend using gini over entropy in criterion. Even though they sort of always get around 86-88%, gini appears to get an 88% accuracy with another parameter more frequently. Setting the Min sample leaf to 5 was also getting its best accuracy for that parameter. It does well with other parameters too. I would recommend using Decision Trees over kNN.

# Future Work

I would explore with different data sets and revise if the algorithm was still the best and try out different parameters and value for decision tree for those new datasets. The dataset used was limited to the number of features, to the values and classification being the same. My prediction would be that with a different dataset normalization would work better and be more accurate.