Carnegie Mellon University Africa

COURSE 18-785: DATA, INFERENCE & APPLIED MACHINE LEARNING

ASSIGNMENT 6

Nchofon Tagha Ghogomu

ntaghagh

November 11, 2024

LIBRARIES:

The following libraries were used:

- Pandas[1]
- Numpy[2]
- Pyplot from Matplotlib [3]
- Stats from Scipy
- Seaborn
- Linear model from Sklearn
- Linear Regression from Sklearn Linear model
- Logistic Regression from Sklearn Linear model

- import pandas as pd # Pandas
- import numpy as np # Numpy
- from scipy import stats # Scipy - for statistics
- import matplotlib.pyplot as plt # Matplotlib - for ploting
- import seaborn as sns # Seaborn - for ploting
- from sklearn import linear_model
- from sklearn.linear_model import LinearRegression
- from sklearn.linear_model import LinearRegression
- from sklearn.linear_model import LogisticRegression
- from sklearn.metrics import confusion_matrix, classification_report
- from sklearn.model_selection import train_test_split
- import statsmodels.api as sm
- from tabulate import tabulate
- from sklearn.tree import DecisionTreeClassifier
- from sklearn.metrics import classification_report, accuracy_score
- from sklearn.model_selection import cross_val_score
- from sklearn.preprocessing import StandardScaler
- from sklearn.neighbors import KNeighborsClassifier
- from sklearn.linear_model import LassoCV
- from sklearn.neighbors import KNeighborsRegressor

Programming Language:

- Python

INTRODUCTION

This assignment was made of 4 practical questions that portray real application of data analytics on world data. This assignment was centred around:

- Linear regression with one or multiple explanatory data
- Model fitting and estimation.
- Hypothesis test for correlation.

This assignment had one open study for us to assess and study the trend in the transport domain of transportation.

SOLUTIONS

Question 1: NONLINEARITY.

1.1 Nonlinearity and necessity.

Nonlinearity refers to relationships that cannot be expresses in the form of a straight line, that is, in the form of a first order equation, according to the course notes. Nonlinear relationships can take the form of a curve, exponential function, higher degree polynomials, and other more complex equations. The portray relations that are not proportional. Some nonlinear functions include:

- The sigmoid function
- The Logarithm
- Exponential function

In the real world, most events and applications don not follow a linear pattern. Also, there are events that cannot be generalised to a linear model. Reasons why nonlinearity in important include:

i) Intricate pattern capturing: In healthcare for example, the progression of a medical conditions might not vary proportionately with the patient's biomedical parameters like blood sugar, BMI and perhaps blood pressure. As a result, nonlinear models will be well placed to capture these complex patterns thereby providing a more accurate prediction than a linear model. Also, some partens are not obvious. As

ii) Flexibility: By this, we refer to the model's ability to adapt to changes in shifts and represent broader relationships in data. In commerce, when trying to predict weather or not a customer will churn, a linear model will not be well suited for this relationship since it will not capture small amounts of dissatisfaction. This accumulates and eventually drives away the customer.

iii) Parsimony: Some nonlinear models are apt at closely modelling relationships while involving fewer parameters. In this case, a linear model might tend to be more complex or not properly fit the model.

1.2 Nonlinear model and application example.

Let us explore the exponential growth or decay model. The formula is given by

$$\boldsymbol{y(t) = y_0 e^{kt}},$$

Where y(t) is the value at current time, y_0 is the initial value and k, the decay rate. On of the most practical application of this function is in growth rate and epidemiology. For example, at the early stage of COVID-19, the spread of the virus was predicted to model an exponential function. This information helped health practitioners know what measures to take that will best curb the spread of the virus.

1.3 Nonlinear model parsimony versus linear models.

Yes. Nonlinear models can sometime be more parsimonious than linear models. This is because sometimes, nonlinear models tend to fit certain occurrences more accurately than linear models, even using fewer parameters. Using growth rate as and example, a linear model that depicts growth rate can be represented using the following equation:

$$\boldsymbol{P_t = P_0 + \lambda t + \varepsilon}.$$

This will be a straight line, but will not accurately represent an exponential growth rate that can be expressed in the form bellow:

$$\boldsymbol{P_t = P_0 e^{-\lambda t}}$$

Therefore, nonlinear models can be mor parsimonious than linear models.

1.4 Surrogate data and nonlinearity.

Surrogate data is generated that keeps the baseline properties of a distribution like the mean, variance, and correlation structure, but might leave out specific patterns, and is used to test for nonlinearity. This test is done by comparing surrogates to the original data, and on the basis of a given hypothesis, verted for linearity. Therefore, the characteristics preserved by during surrogate generation include:

- The distribution: Properties that define the distribution like the mean and variance are preserved.
- Auto correlation: This is the relationship data points of a distribution have with past data points in a time series.

By preserving these characteristics, we ensure that the surrogates have the base line representation of the model they were generated from.

Two surrogate techniques include:

i) Random shuffle: This is a technique that creates surrogates by shuffling the original data. The original distribution structure is maintained; however the linear correlation is destroyed.

ii) Random Phases: This technic is used to preserve linear correlation and is obtained by applying the inverse furrier transform on the modules of the furrier transform of the original data wit new uniform random phases.

1.5 Information, entropy and Mutual information.

Information in the context of information theory refers to the degree of knowledge gained by observing an outcome, or the how much an outcome reduces uncertainty. It can be represented by the following formular below.

$$\boldsymbol{I(p) = -\log(p)}$$

where p is the probability of an event and I, the information obtained with the occurrence of that event, where $I(1) = 0$ and $I(p) \geq 0$.

Entropy is a measure of the degree of disorderliness, or unpredictability of a system or random variable. It can be represented using the equation below.

$$\boldsymbol{H(x) = -\sum_{i=0}^{n} P(x) lnP(x)}$$

Entropy is very useful in determining regularity or predictability in a system. A system with lower entropy tends to be more predictable or follows a pattern compared to system with higher entropy reflecting much randomness. Entropy is very useful in financial markets for risk mitigation. For example, stock prices with lower entropy are relatively stable and predictable whereas, high entropy stock prices are volatile and random. This enhances well informed risk mitigation decision making.

Mutual information measures the amount of information provided by a given variable about another variable. It is expressed in the following equation.

$$\boldsymbol{I(x,y) = H(x) + H(y) - H(x,y)}$$

Mutual information permits us determine the dependence between features, and between target variable and features. Mutual information can be used to assess how relevant a feature is as features with high mutual information with target carriable will provide more valuable information to a model than one with lower mutual information.

Question 2: CLASSIFICATION TREES

2.1 Decision Tree Components.

A decision tree[4] is a tree-like decision making structure achieved by using observed values from features in a data set. It has the following structure:

- Nodes: These are the point on the tree where decisions are made. We have the root node with is the very first point of decision making in a decision tree, and it represents the entire data set. The internal node are the nodes after the root node that perform a decision based on one of the parameters of the data set.
- Branches: These are the paths or outcomes taken after a decision at a node has been made. Branches may lead to another node (internal or leafy node)
- Leafy nodes: These are special kind of node representing the final decision taken by the model. No further splitting is done here.

In decision trees, pruning is the act of reducing a tree's size by cutting down the number of nodes to ensure that only representative nodes are kept that provides getter predictability power to the system[5].

Pruning trees enhances model parsimony by ensuring just features that greatly contribute the model are retained. It also enhances model generalisation as thick trees might perform well on training data but not as when exposed to independent data. In pruning, the tree can be simplified and can lead to an increase in accuracy.

Decision trees suit several practical applications due to the following reasons:

- It is simple and easy to understand. As a result, it is valuable in areas that require the decision made to be explainable like in hospitals, finance, to list a few.
- It is versatile: It can be used for both classification and regression purposes.
- It can handle complex nonlinear relationships.

2.2 Rule based classification

To implement a data driven classifier[6], we:

- Begin with the entire data and identify all the possible binary decision that can be made on every predictor:
- Now, we select the decision or splits with the best optimization criterion.
- We impose the split and repeat the same process for the child node
- At this point, we set a large value for the maximum number of splits to get a big tree.

To test the validity of the newly implemented model, we can make use of the following methods:

i) Cross Validation[7]: This is a technic for assessing the performance of model by splitting data into two for training and testing the model. The average performance of the model on the testing data gives use a cross-validation score which tells us how accurate our model is.

ii)

2.3 Decision tree on the Titanic Dataset:

To construct a decision tree based on the Titanic dataset, we:

- Import our dataset
- Extract independent variables (gender, age, pclass) dependent variable(survived)
- Clean data. Here, we replaced the missing age values with the mean age and transformed sex data into categorical data (0 for male and 1 for female)
- Employed sklearn's DecisionTreeClassifier and fit our X and y variables to it.
- Using plot_tree from sklearn, we obtained the following tree before pruning, with a depth set at 20

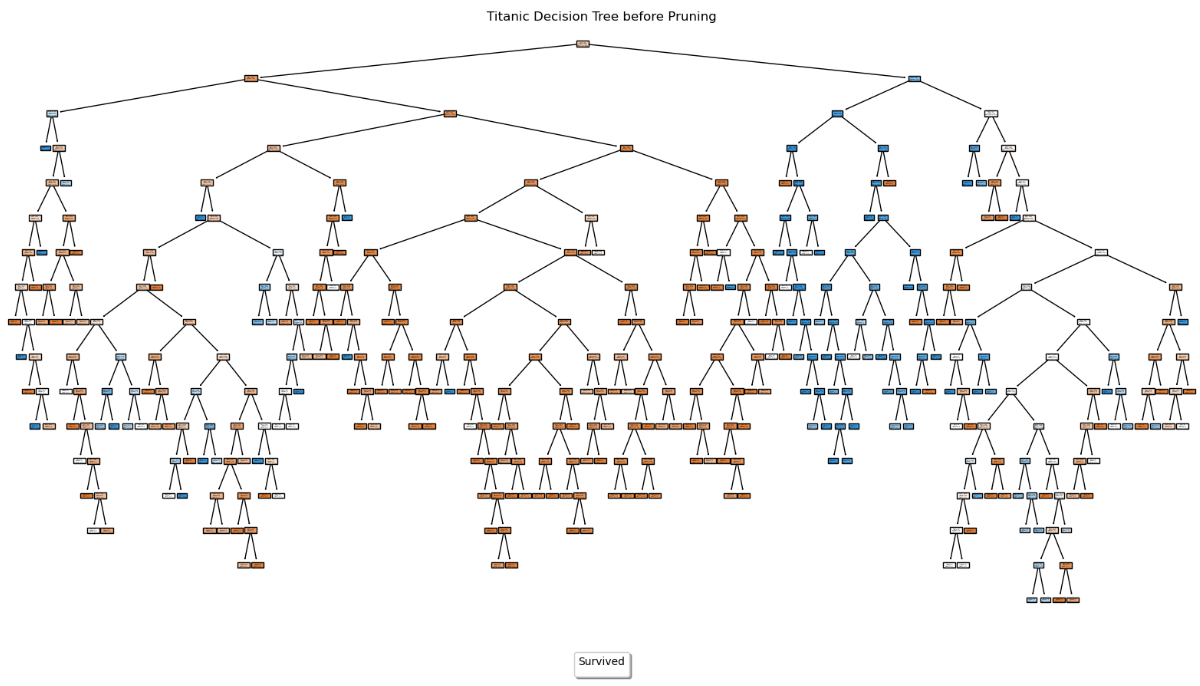
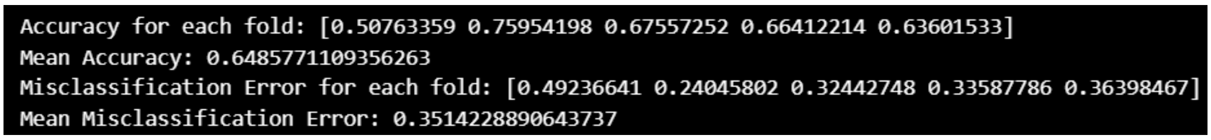


Figure 1 Decision tree of Titanic dataset before pruning

2.4 Model evaluation before and after pruning:

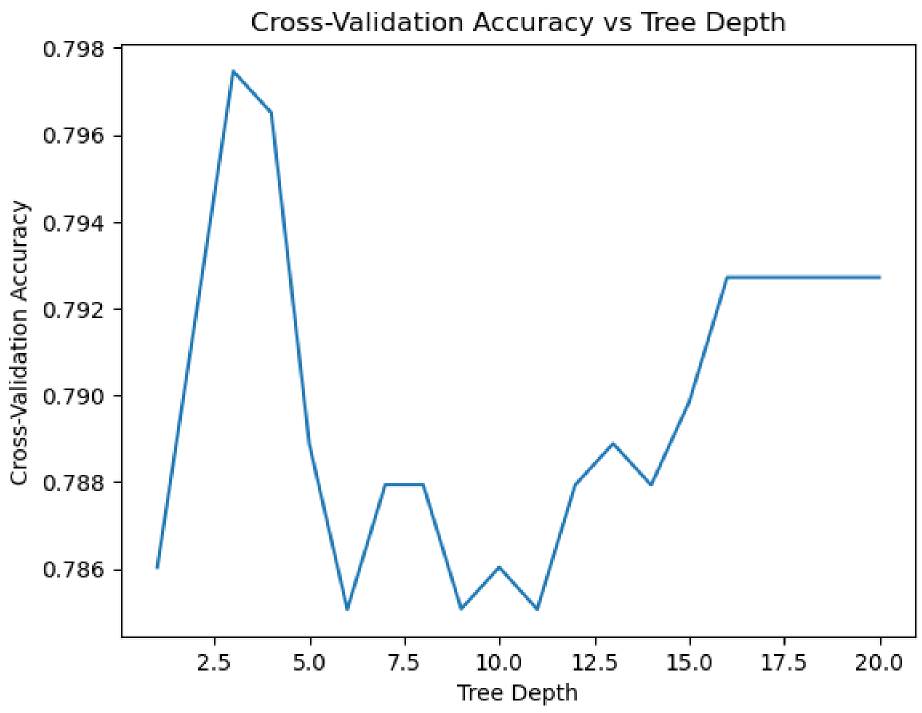
Before pruning, five-fold cross validation was performed on the data and the results can be seen bellow, with a mean misclassification error of 0.35



```
Accuracy for each fold: [0.50763359 0.75954198 0.67557252 0.66412214 0.63601533]
Mean Accuracy: 0.6485771109356263
Misclassification Error for each fold: [0.49236641 0.24045802 0.32442748 0.33587786 0.36398467]
Mean Misclassification Error: 0.3514228890643737
```

We observe that for each fold, the accuracy is different. We obtained a mean accuracy: 0.6458. We obtain the misclassification error by subtracting the accuracy score from one.

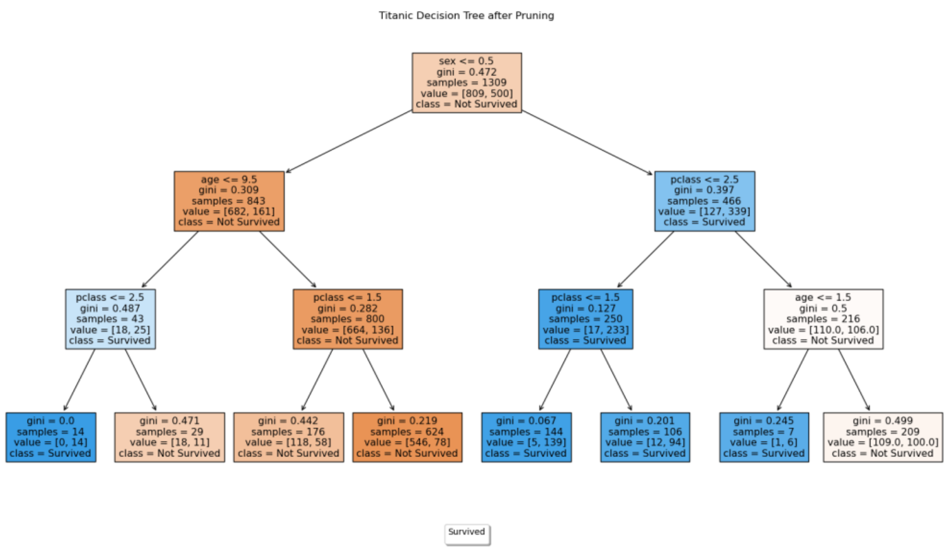
Pruning was performed and we examined the cross-validation accuracy with tree depth and the following results was achieved:



By using sklearn’s DecisionTreeClassifier, we extract the tree with the best depth. Our results tell us that 3 is the best tree depth with a cross-validation accuracy of 0.7974:

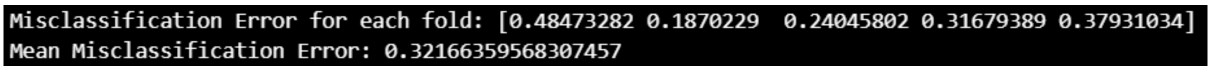
```
Best Tree Depth: 3, Cross-Validation Accuracy: 0.7974663932558669
```

Finally, after fitting our model with the best tree parameter, we obtain the following tree:



Titanic Decision Tree after Pruning
sex <= 0.5 gini = 0.472 samples = 1309 value = [809, 500] class = Not Survived
age <= 9.5 gini = 0.309 samples = 843 value = [682, 161] class = Not Survived
pclass <= 2.5 gini = 0.397 samples = 466 value = [127, 339] class = Survived
pclass <= 2.5 gini = 0.487 samples = 43 value = [18, 25] class = Survived
pclass <= 1.5 gini = 0.282 samples = 800 value = [664, 136] class = Not Survived
pclass <= 1.5 gini = 0.127 samples = 250 value = [17, 233] class = Survived
age <= 1.5 gini = 0.5 samples = 216 value = [110.0, 106.0] class = Not Survived
gini = 0.0 samples = 14 value = [0, 14] class = Survived
gini = 0.471 samples = 29 value = [18, 11] class = Not Survived
gini = 0.442 samples = 176 value = [118, 58] class = Not Survived
gini = 0.219 samples = 624 value = [546, 78] class = Not Survived
gini = 0.067 samples = 144 value = [5, 139] class = Survived
gini = 0.201 samples = 106 value = [12, 94] class = Survived
gini = 0.245 samples = 7 value = [1, 6] class = Survived
gini = 0.499 samples = 209 value = [109.0, 100.0] class = Not Survived
Survived

The misclassification error for the pruned tree was calculated to be:



```
Misclassification Error for each fold: [0.48473282 0.1870229  0.24045802 0.31679389 0.37931034]
Mean Misclassification Error: 0.32166359568307457
```

We observe a slight decrease in misclassification error of the pruned tree from 0.35 to 0.32. Therefore, we can conclude that in this model, the pruned tree is less liable to misclassification. In addition, this model is parsimonious.

2.5 Model comparison:

The accuracy of both the decision tree and the linear regression model are roughly the same. But base on the decision tree's parsimony nature, I would prefer it for the Kaggle competition.

Question 3: CLASSIFICATION USING KNN

3.1 Decision Tree Components.

The k-nearest neighbours (KNN) algorithm is a non-parametric, supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point[8]. It works under the principles that elements with the same propertied or features are close together. Therefore, this algorithm simply seeks to calculate the distance between points and classify points close together under a given category. Two example areas where this algorithm can be applied include:

3 Pattern recognition, in the case of identifying handwritten texts and related content
4 Recommendation engines, where an item is suggested to a customer based on a group of items a customer purchased.

The general approach to implement this algorithm involves finding the closed query point to a given data point and attributing this data point to that class.

The step-by-step approach would be:

1) Selecting k, the optimal number of neighbours to be considered to when predicting. A value of 1 suggest that we simply assume the class to which the nearest neighbour belongs to. For multiple points, we select the class where most of the nearest points are.
2) Calculate the distances between nearest points. Algorithms like the Euclidean, Manhattan, Minkowski distance to name a few can be used to calculate the distances between points.
3) Choosing nearest neighbour. We chose the neighbour with the majority vote of least distance to this data point.

3.2 Data preparation/transformation in the current challenge.

To transform the data[9]:

5 NaN values for ages were filled with mean ages
6 Convert categorical data for gender was converted to binary
7 Perform dummy encoding on the passenger class feature using pandas get_dummies class.
8 Scale features to reduce the impact large feature values can have on smaller ones

At this point, we can be sure of having a clean data to work with.

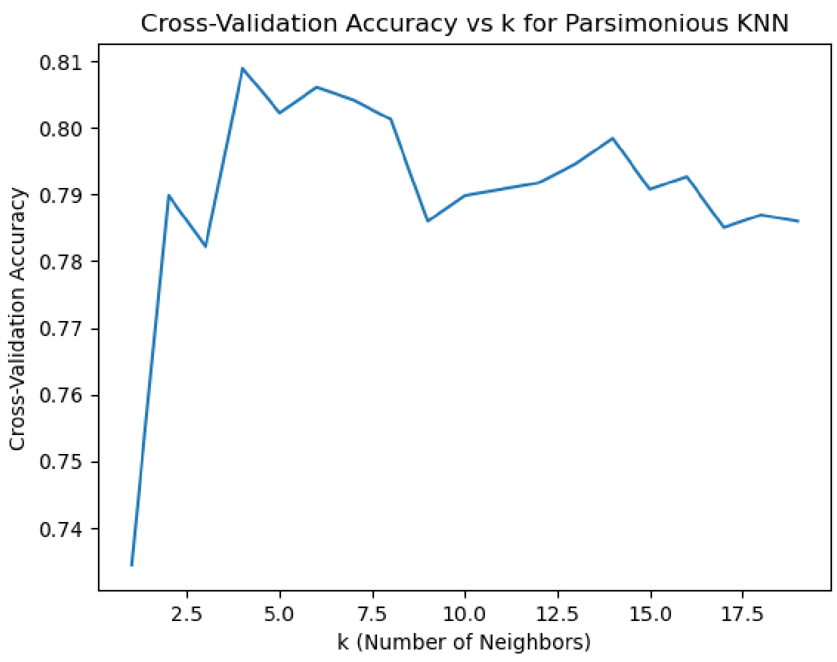
3.3 Classifier performance versus number of neighbours.

To examine the classifier performance versus the number of neighbours we:

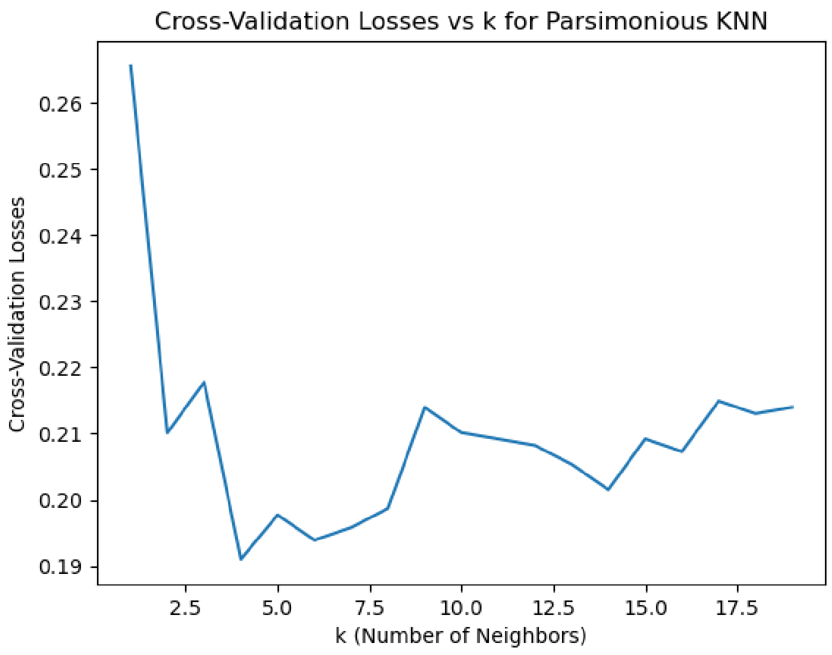
i) Created and empty list cross validation scores.
ii) Set a range of k, in this case 1 to 20.

iii) For each value of k, we compute KNN.
iv) Calculate the accuracy (performance score) using cross-validation.
v) The score is appended to the list.

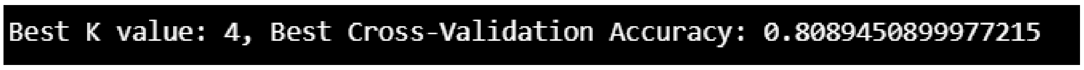
Plotting a graph of performance against k, we obtain the following:



The losses is computed by simply subtracting the accuracy from 1:



The optimal number of neighbours is obtained by extracting k with the highest accuracy or list losses. In this case, the value for k obtained was 4.



```
Best K value: 4, Best Cross-Validation Accuracy: 0.8089450899977215
```

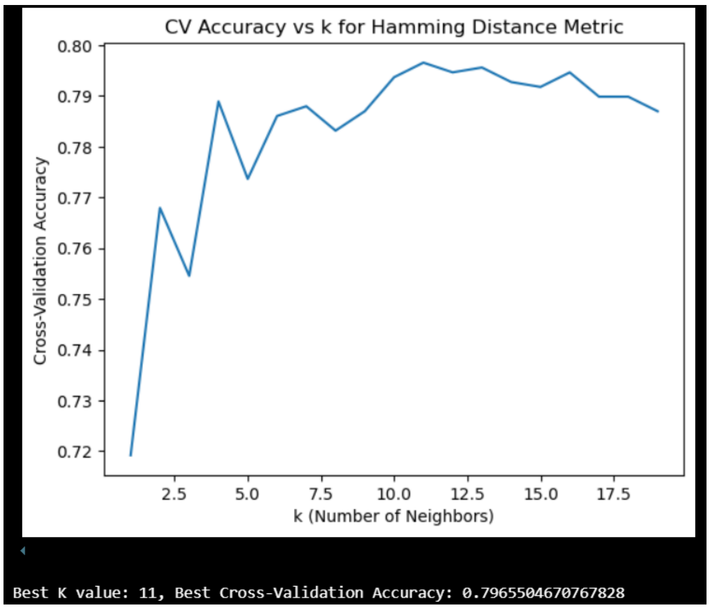
3.4 Why some matrices are sensitive to the kind of features used.

Some distance metrics are sensitive to the kind of features used because the features tend to hold a variety of data types, while matrices were designed to work on specific data types. For example, Euclidian, Manhattan, and Mahalanobis are best suited for continuous data but sensitive to scaling while Hamming distance matrix best suits categorical/binary data and hybrid will suit a mixture of these data types.

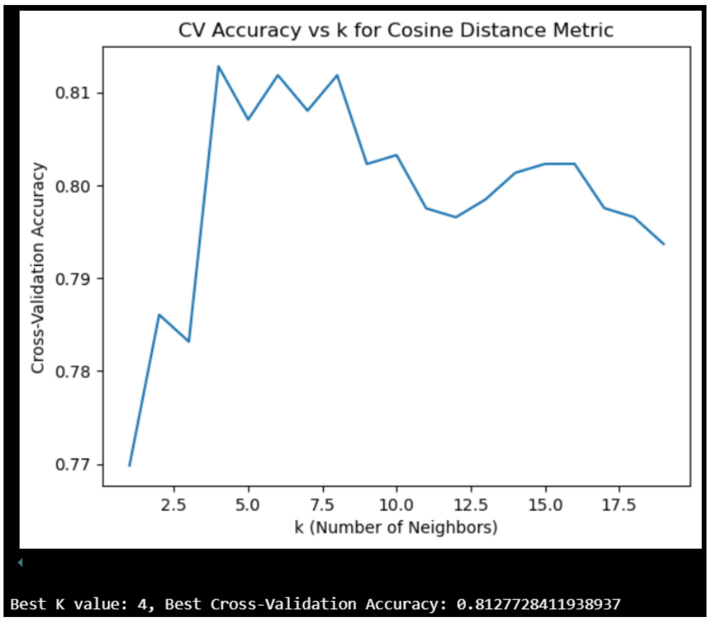
To compare the performance of the model using different distance matrices, we used the:

- Euclidian distance matrix
- Harming distance matrix
- Cosine distance matrix

We obtained the following performance plots.

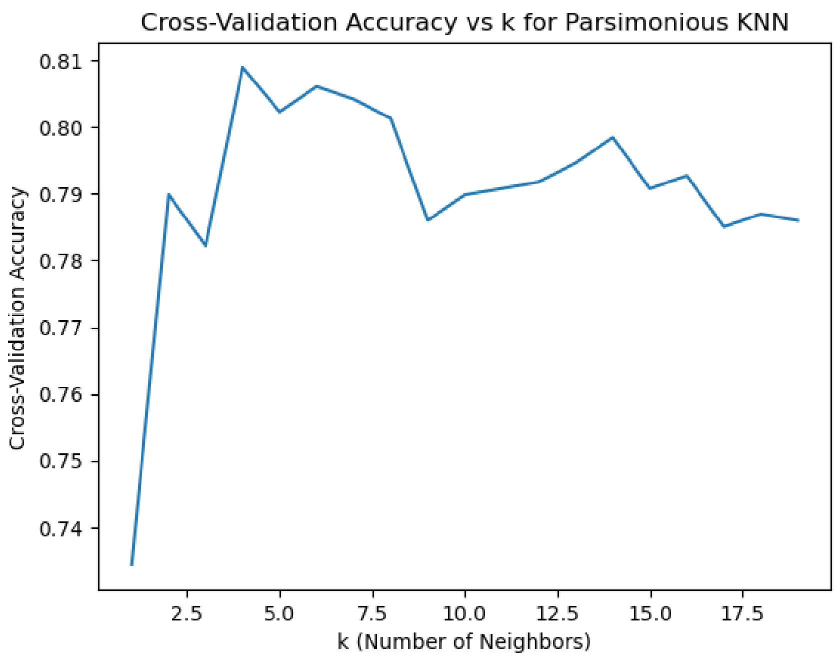


```
Best K value: 11, Best Cross-Validation Accuracy: 0.7965504670767828
```



```
Best K value: 4, Best Cross-Validation Accuracy: 0.8127728411938937
```

The firs plot was done using the Euclidian distance matric by default.



We obtain the following accuracy:

- Euclidian: 0.8089
- Hamming: 0.797
- Cosine: 0.8128

Based on this comparison, The cosine matric has the best accuracy.

3.5 Comparing KNN with regression.

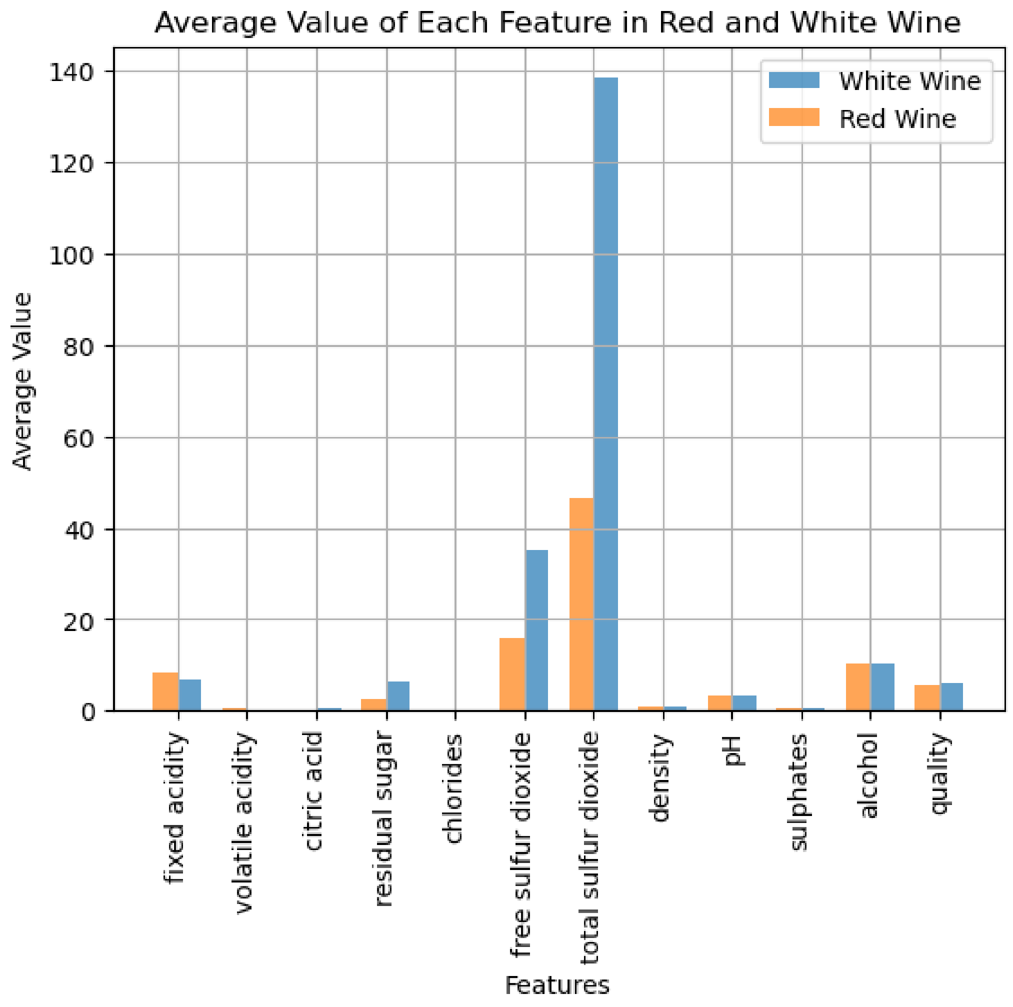
Comparing KNN with the logistic regression, KNN performs better when using the Euclidian and Cosine Matrix. KNN is good for it's simplicity, adaptability, and interpretability[10], but may suffer high computational cost, and sensitivity to irrelevant feature. It is best to employ KNN when data set is small, when features present complex correlation between each other and when data is reasonably well scaled. On the other hand, linear regression is simple and efficient, but it assumes linearity, it is sensitive to outliers.

For the Kaggle competition, KNN is best suited due to its higher accuracy.

Question 4: REGRESSION – WINE QUALITY

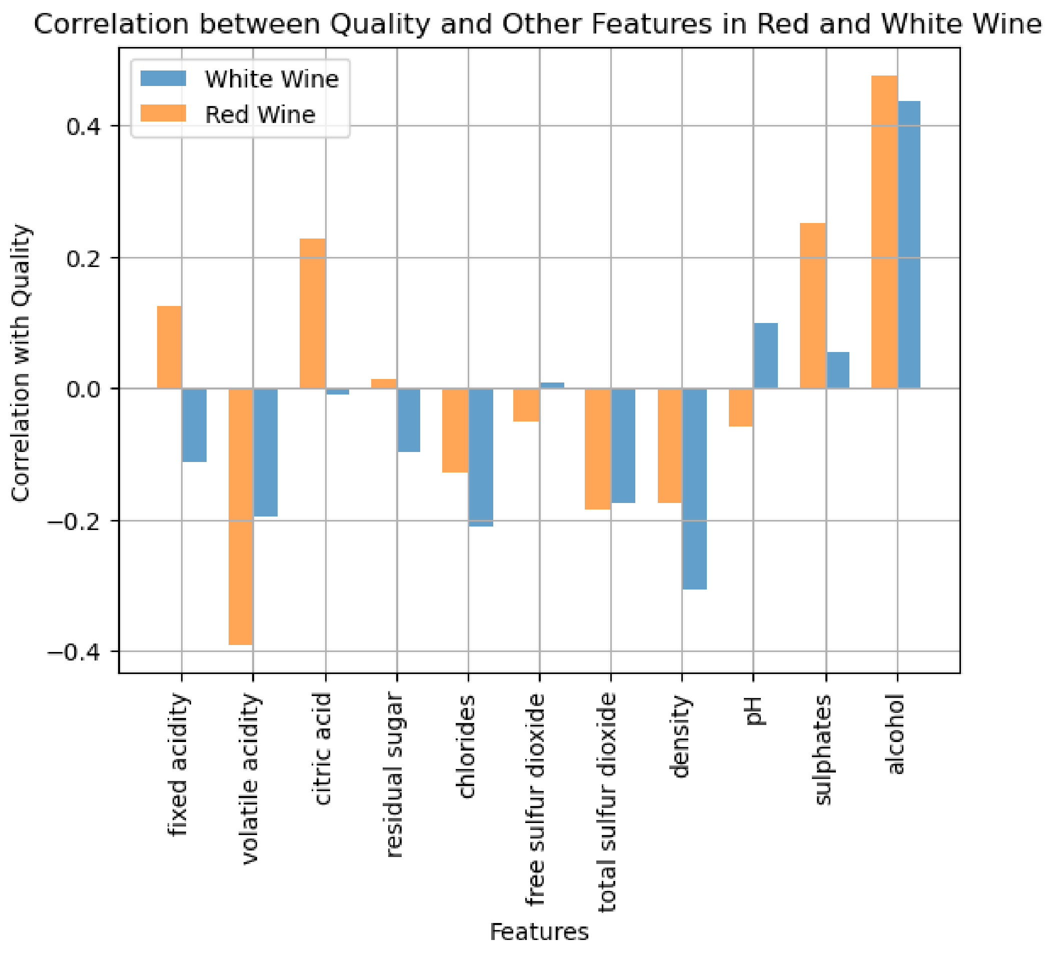
4.1 Feature average comparison and interpretation:

The average of each feature for red wine and white wine were computed and represented in the bar chat bellow. We observe that the feature with the greatest average for both white type is total sulfurdioxide. In basic common sense, the total sulfur dioxide will probabliy contribute greatly to the wine quality of both types, followed by free sulfur dioxide.



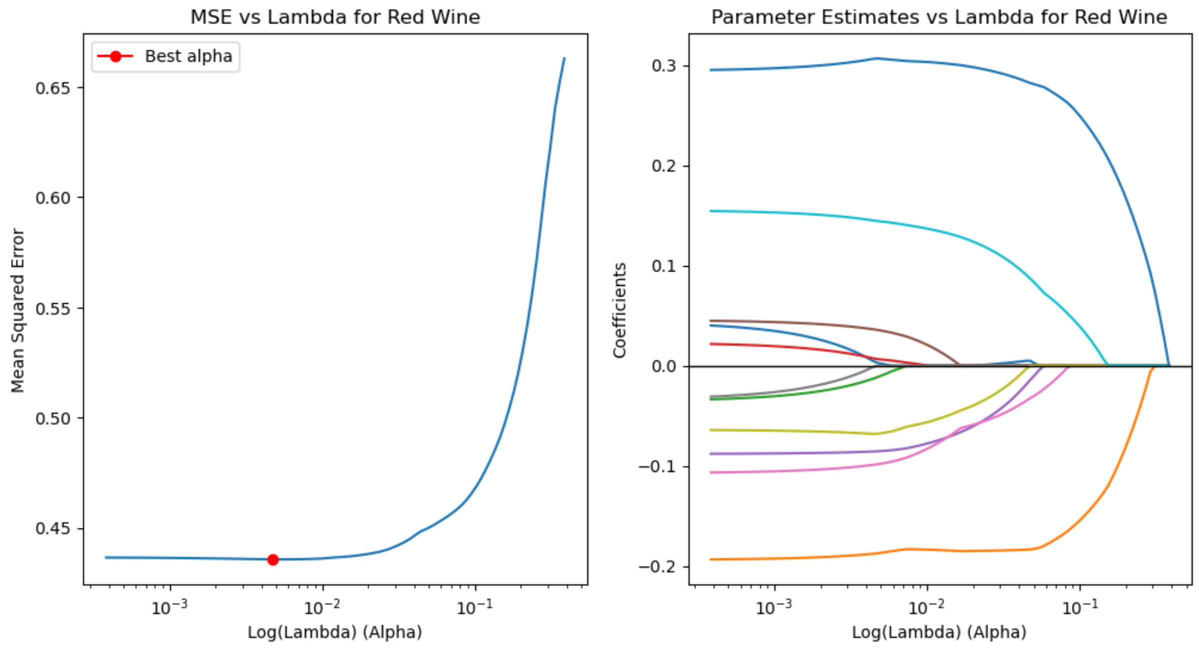
4.2 Feature correlation with wine quality

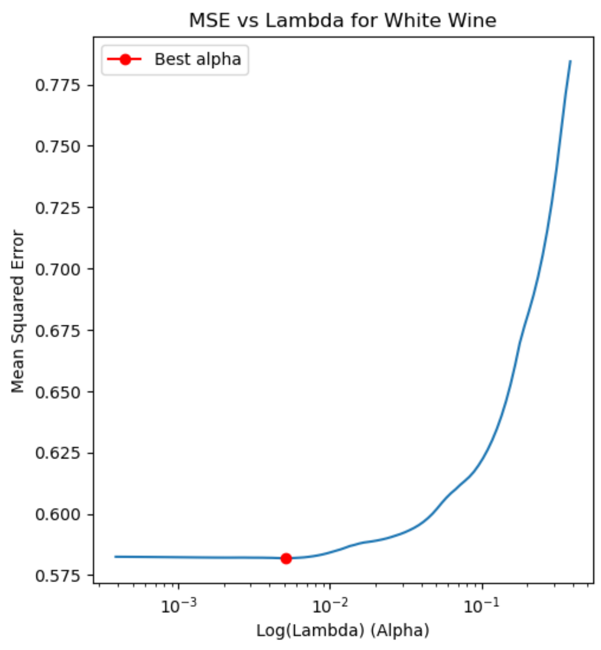
To obtain the correlation between features and qualities, we use the .corr() function. The correlations obtained was represented using the bar graph bellow. For both white and red wine, the most relevant feature was **alcohol.** This means, there is a great relationship between the quantity of alcohol in wine and the wine quality.

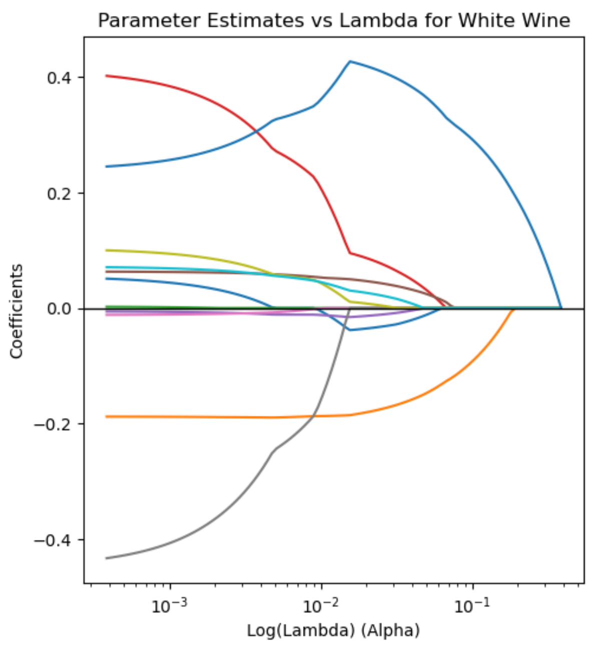


4.3 Use Lasso and cross-validation to provide a plot of MSE against lambda and the parameter estimates versus lambda for white and red wine

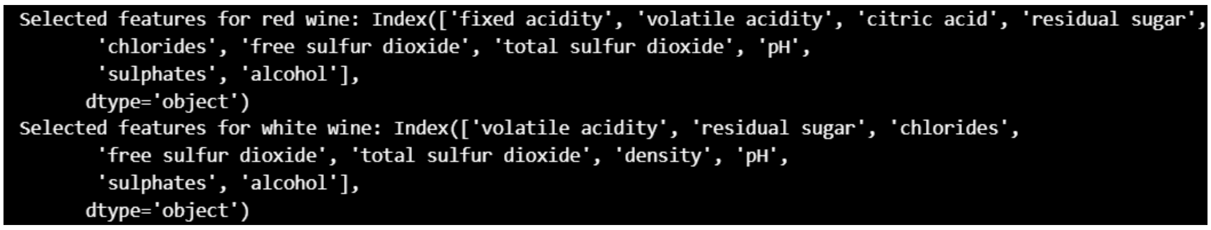
The following graph for MSE and Lambda was obtained after using lassso and cross-validation.





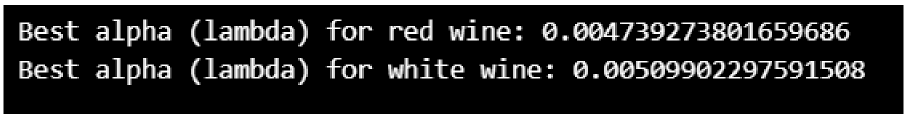


Ten and nine features were selected from red and white wine respectively as seen bellow:



```
Selected features for red wine: Index(['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
       'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'pH',
       'sulphates', 'alcohol'],
      dtype='object')
Selected features for white wine: Index(['volatile acidity', 'residual sugar', 'chlorides',
       'free sulfur dioxide', 'total sulfur dioxide', 'density', 'pH',
       'sulphates', 'alcohol'],
      dtype='object')
```

The best lambda was computed to be:



```
Best alpha (lambda) for red wine: 0.004739273801659686
Best alpha (lambda) for white wine: 0.00509902297591508
```

Remarks:

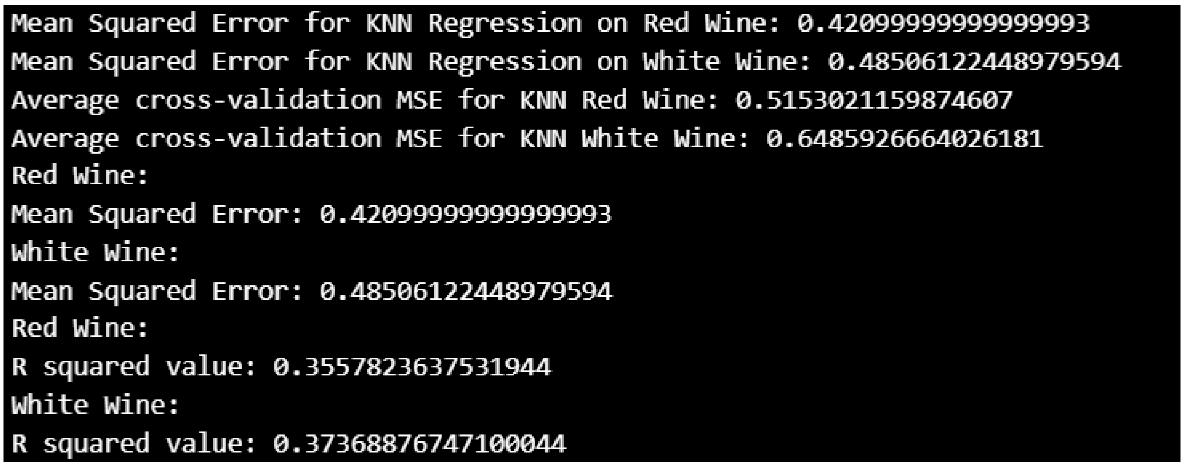
After analysis, we can infer that LASSO is relatively a more robust feature selection approach since it performs regularisation during feature selection. This can go a long way to prevent overfitting. Lasso can easily deal with multicollinearity between features. On the other hand, absolute correlation is very rigid.

*4.4 **KNN regression** model for Red Wine using selected features from LASSO*

To construct a KNN regression model, we simply use the KNeighborsRegressor and fit it to our data.

4.5 Assessing model performance.

The model performance on the bases of MSE and R-squared values was obtained after KNN Regression to be:


```
Mean Squared Error for KNN Regression on Red Wine: 0.42099999999999993
Mean Squared Error for KNN Regression on White Wine: 0.48506122448979594
Average cross-validation MSE for KNN Red Wine: 0.5153021159874607
Average cross-validation MSE for KNN White Wine: 0.6485926664026181
Red Wine:
Mean Squared Error: 0.42099999999999993
White Wine:
Mean Squared Error: 0.48506122448979594
Red Wine:
R squared value: 0.3557823637531944
White Wine:
R squared value: 0.37368876747100044
```

References

[1] "pandas documentation — pandas 2.2.2 documentation." Accessed: Sep. 02, 2024. [Online]. Available: https://pandas.pydata.org/pandas-docs/stable/index.html

[2] "NumPy -." Accessed: Sep. 02, 2024. [Online]. Available: https://numpy.org/

[3] "Matplotlib documentation — Matplotlib 3.9.2 documentation." Accessed: Sep. 02, 2024. [Online]. Available: https://matplotlib.org/stable/

[4] "Decision Tree," GeeksforGeeks. Accessed: Nov. 16, 2024. [Online]. Available: https://www.geeksforgeeks.org/decision-tree/

[5] "Pruning decision trees," GeeksforGeeks. Accessed: Nov. 19, 2024. [Online]. Available: https://www.geeksforgeeks.org/pruning-decision-trees/

[6] "Building and Implementing Decision Tree Classifiers with Scikit-Learn: A Comprehensive Guide," GeeksforGeeks. Accessed: Nov. 19, 2024. [Online]. Available: https://www.geeksforgeeks.org/building-and-implementing-decision-tree-classifiers-with-scikit-learn-a-comprehensive-guide/

[7] "3.1. Cross-validation: evaluating estimator performance," scikit-learn. Accessed: Nov. 19, 2024. [Online]. Available: https://scikit-learn/stable/modules/cross_validation.html

[8] "What is the k-nearest neighbors algorithm? | IBM." Accessed: Nov. 19, 2024. [Online]. Available: https://www.ibm.com/topics/knn

[9] "K-Nearest Neighbor(KNN) Algorithm," GeeksforGeeks. Accessed: Nov. 19, 2024. [Online]. Available: https://www.geeksforgeeks.org/k-nearest-neighbours/

[10] C. Story, "KNN vs. Linear Regression: How To Choose The Right ML Algorithm," Medium. Accessed: Nov. 19, 2024. [Online]. Available: https://medium.com/@skytoinds/knn-vs-linear-regression-how-to-choose-the-right-ml-algorithm-4f6bf01a4202