Machine Learning: Assignment 1

Group #1

Classification Problems: Abalone, Online News Popularity, Cardiotocography

Regression Problems: Advanced House Pricing

ID	Student Name	E-mail
196280	Ashraf Adel	ashraf196280@bue.edu.eg
194233	Farah Aymen	farah194233@bue.edu.eg
206562	Jacinta Samir	jacinta206562@bue.edu.eg

Table of Contents

Classification Problem 1: Abalone Dataset

Problem Description

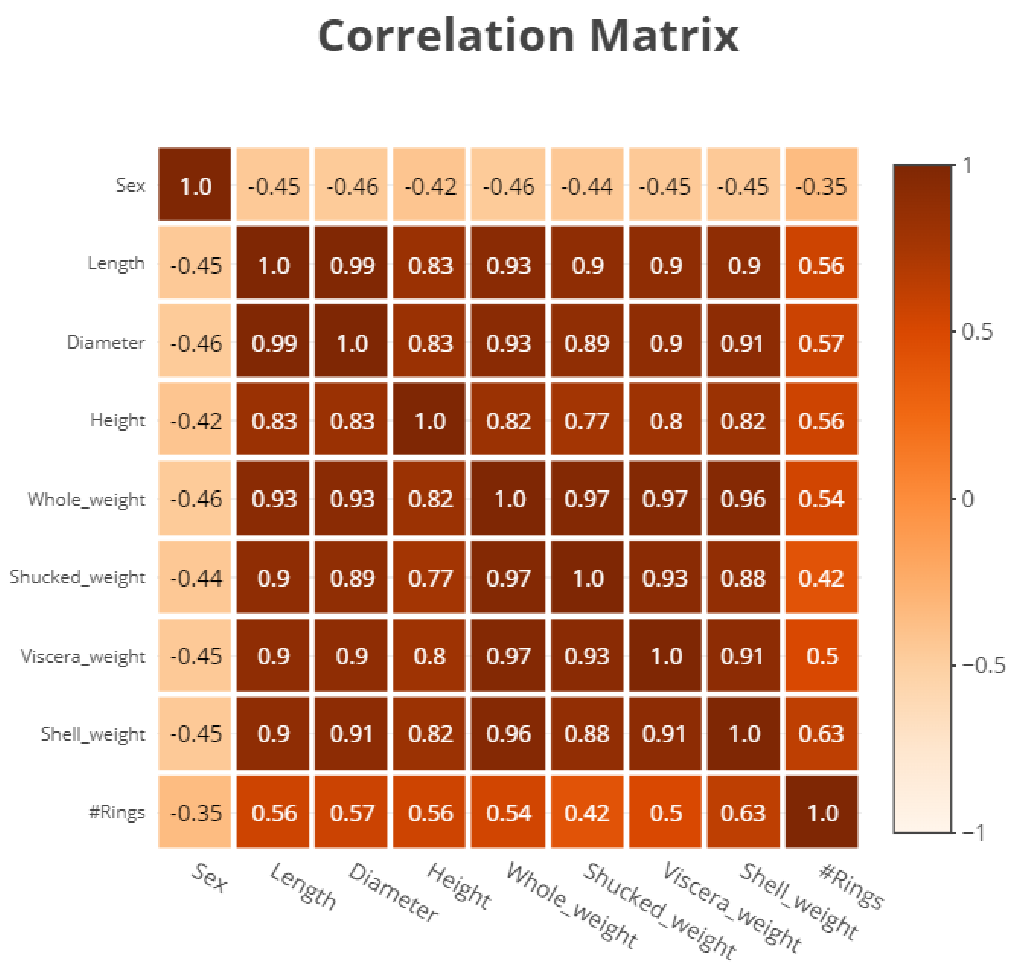
The objective of the abalone dataset is to create and train an ML model that Given the details (features) of an Abalone, it can predict the number of rings, which will give an indication to the age of that Abalone (rings + 1.5 is an approximation of the age of abalone). Note that an abalone is a marine snail.

Dataset Description and Visualisation

Given description of the columns (features):

Name	Data Type	Measurement Unit	Description
Sex	Nominal	…	Male, Female, Infant
Length	Continuous	mm	Longest shell measurement
Diameter			perpendicular to length
Height			with meat in shell
Whole weight		grams	whole abalone
Shucked weight			weight of meat
Viscera weight			gut weight (after bleeding)
Shell weight			after being dried
Rings	Integer	…	+1.5 gives the age in years
	Total rows (examples): 4177		

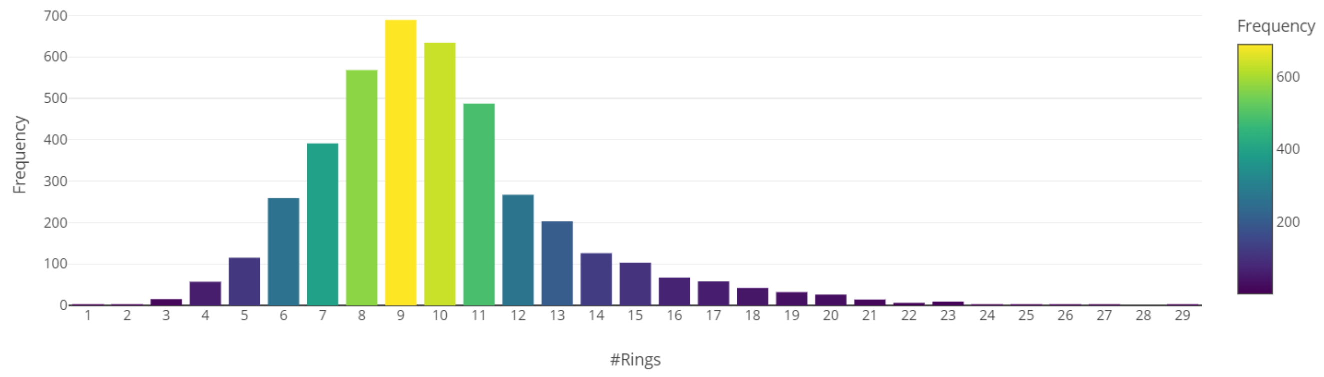
Correlation Matrix:



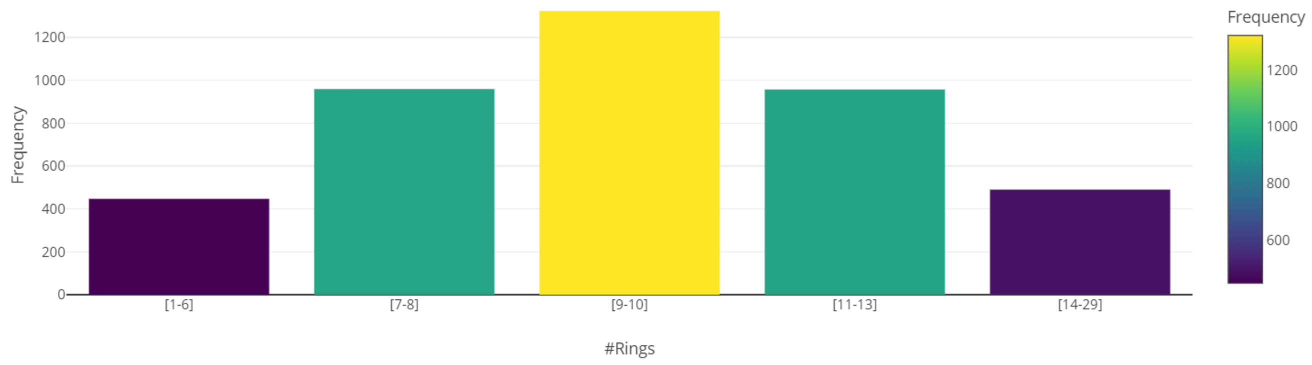
	Sex	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscera_weight	Shell_weight	#Rings
Sex	1.0	-0.45	-0.46	-0.42	-0.46	-0.44	-0.45	-0.45	-0.35
Length	-0.45	1.0	0.99	0.83	0.93	0.9	0.9	0.9	0.56
Diameter	-0.46	0.99	1.0	0.83	0.93	0.89	0.9	0.91	0.57
Height	-0.42	0.83	0.83	1.0	0.82	0.77	0.8	0.82	0.56
Whole_weight	-0.46	0.93	0.93	0.82	1.0	0.97	0.97	0.96	0.54
Shucked_weight	-0.44	0.9	0.89	0.77	0.97	1.0	0.93	0.88	0.42
Viscera_weight	-0.45	0.9	0.9	0.8	0.97	0.93	1.0	0.91	0.5
Shell_weight	-0.45	0.9	0.91	0.82	0.96	0.88	0.91	1.0	0.63
#Rings	-0.35	0.56	0.57	0.56	0.54	0.42	0.5	0.63	1.0

General notes regarding the dataset:

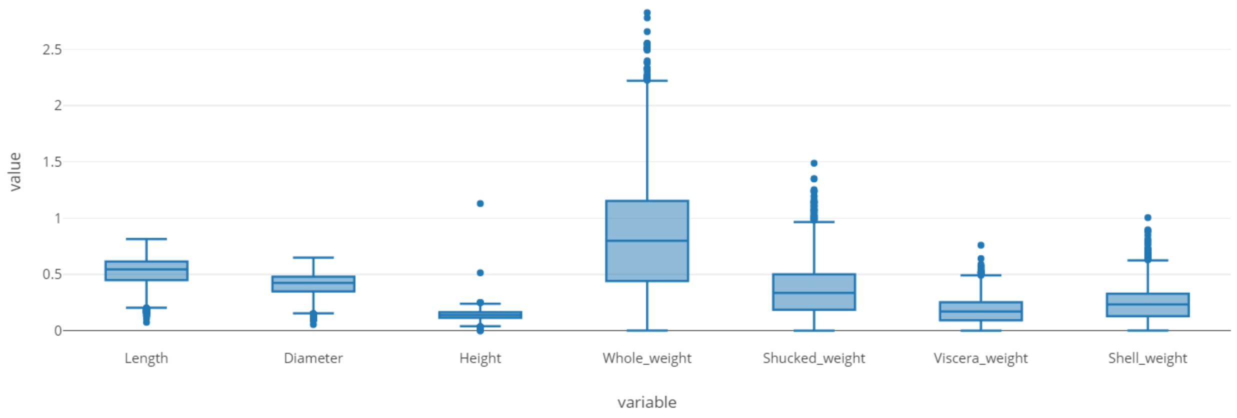
1. The domain of the dataset is relevant; people who are interested in finding snail characteristics that indicate their lifespan will know that the number of rings in it's shell can be used to approximate its age, so they will be interested in knowing the number of shell rings without having to cut the snail.
2. No missing values or duplicates. In addition, there are few outliers; the data has been preprocessed by the author
3. There are 8 features, so the data has relatively low dimensionality
4. The target classes are distributed in the following way (from 1 to 29 excluding 28):



Which shows that the data is highly imbalance. Therefore, instead of getting rid of target classes, we binned them in the following manner:



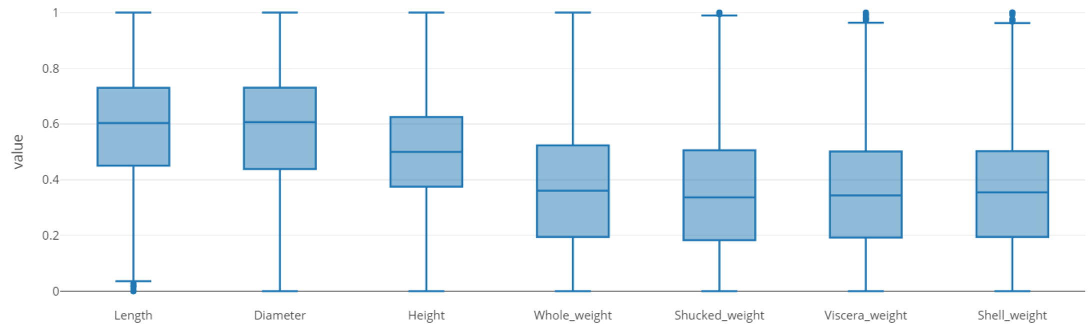
5. Regarding Encoding Features:
    a. Since an ML model is preferred to train on numbers, not strings (objects), therefore encoding needs to be done. Luckily, the only feature that is string is the "Sex" feature. Therefore, Target-Encoding is applied to convert "M", "F", "I" to weighted averages of the feature classes
6. Regarding feature scaling:
   The data is not normalized; the author of the dataset mentioned that the data is scaled by dividing by 200. However, normalization, which is mapping values to be in the range of [0, 1] or [-1, 1], didn't take place, as we can clearly see values higher than 1:



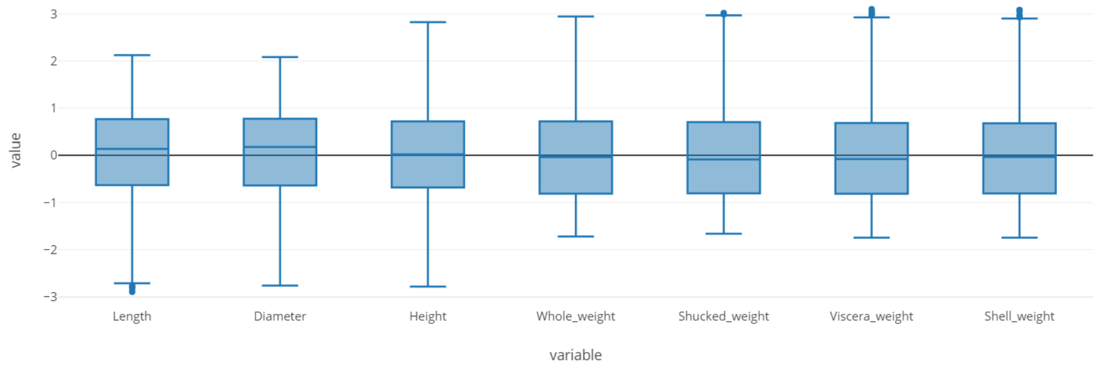
Therefore, normalization or standardization can be applied. Given the resources that I've read, I've decided to:

a. Use the original dataset for the decision tree (DT) model

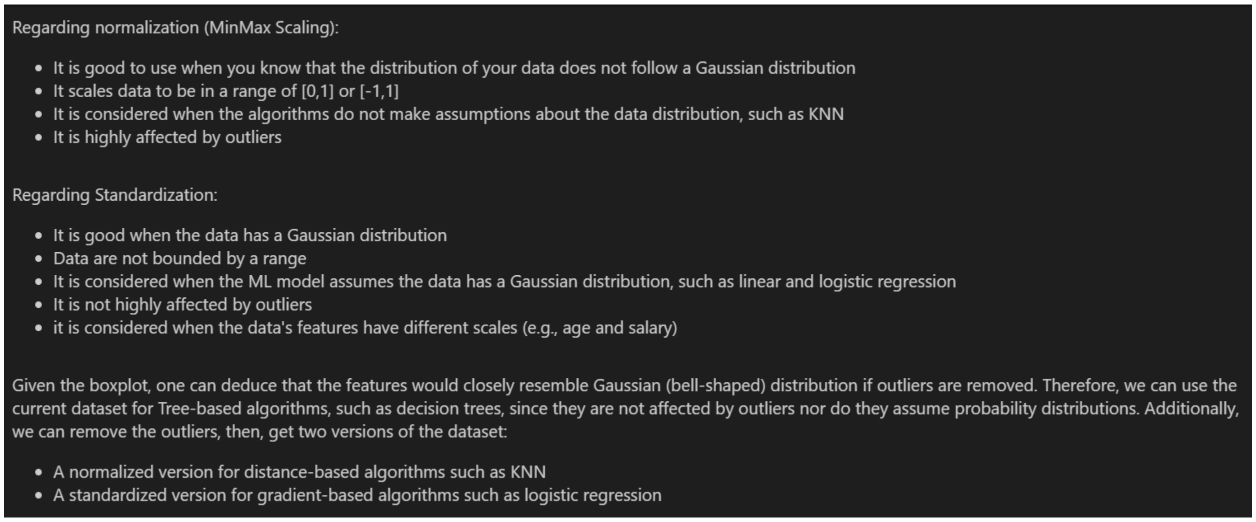
b. Use a normalized version of the dataset for the K-Nearest Neighbours (KNN) model



c. Use a standardized version of the dataset for the Logistic Regression (LR) model



The summary of why these choices were made is here:



Regarding normalization (MinMax Scaling):

- It is good to use when you know that the distribution of your data does not follow a Gaussian distribution
- It scales data to be in a range of [0,1] or [-1,1]
- It is considered when the algorithms do not make assumptions about the data distribution, such as KNN
- It is highly affected by outliers

Regarding Standardization:

- It is good when the data has a Gaussian distribution
- Data are not bounded by a range
- It is considered when the ML model assumes the data has a Gaussian distribution, such as linear and logistic regression
- It is not highly affected by outliers
- it is considered when the data's features have different scales (e.g., age and salary)

Given the boxplot, one can deduce that the features would closely resemble Gaussian (bell-shaped) distribution if outliers are removed. Therefore, we can use the current dataset for Tree-based algorithms, such as decision trees, since they are not affected by outliers nor do they assume probability distributions. Additionally, we can remove the outliers, then, get two versions of the dataset:

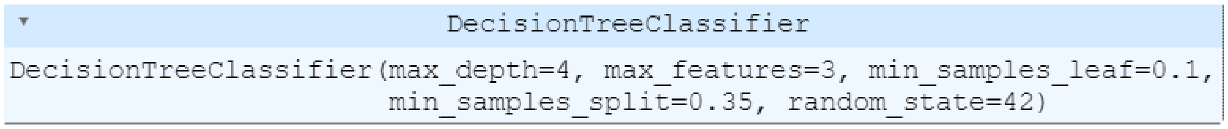
- A normalized version for distance-based algorithms such as KNN
- A standardized version for gradient-based algorithms such as logistic regression

7. Regarding feature selection:

   a. We'll use a grid search on each of the 3 models, such that each model (at each iteration of the grid search) will be trained on each combination (subset) of features generated by EFS, then the best features selected will be based on the the model that performs the best (within one grid search iteration). At the end, the GridSearchCV module will pick the best resulted estimator (model) from all the cross-validated models

Machine Learning Models Used

Decision Tree (DT)



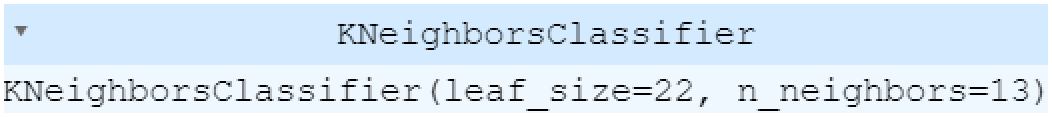
```
DecisionTreeClassifier
DecisionTreeClassifier(max_depth=4, max_features=3, min_samples_leaf=0.1,
                       min_samples_split=0.35, random_state=42)
```

The corresponding features selected for the regular dataset:

Sex	Length	Diameter	Whole_weight	Shucked_weight	Shell_weight	#RingsRange

Notice how "Height" and "Viscera_Weight" have been removed.

K-Nearest Neighbours (KNN)



```
KNeighborsClassifier
KNeighborsClassifier(leaf_size=22, n_neighbors=13)
```

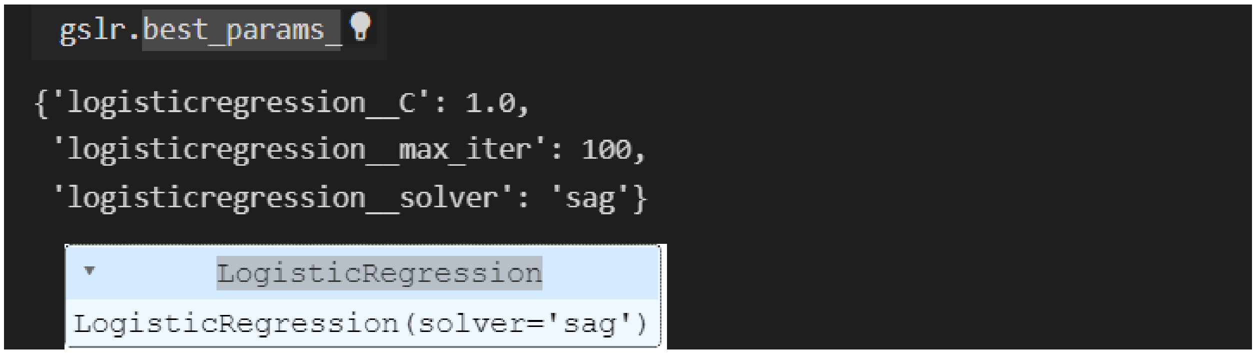
And p = 2, which is equivalent to using ecludian distance.

The corresponding features selected for the normalized dataset:

Sex	Length	Diameter	Shucked_weight	Viscera_weight	Shell_weight	#RingsRange

Notice how "Height" and "Whole_Weight" have been removed

Logistic Regression (LR)



```
gslr.best_params_

{'logisticregression__C': 1.0,
 'logisticregression__max_iter': 100,
 'logisticregression__solver': 'sag'}

LogisticRegression
LogisticRegression(solver='sag')
```

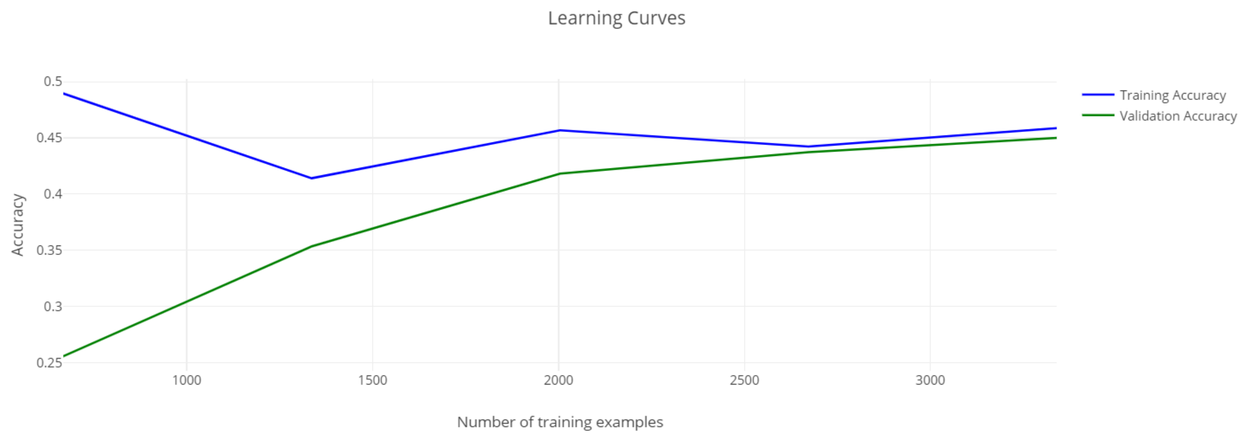
The corresponding features selected for the standardized dataset:

Sex	Height	Whole_weight	Shucked_weight	Viscera_weight	Shell_weight	#RingsRange

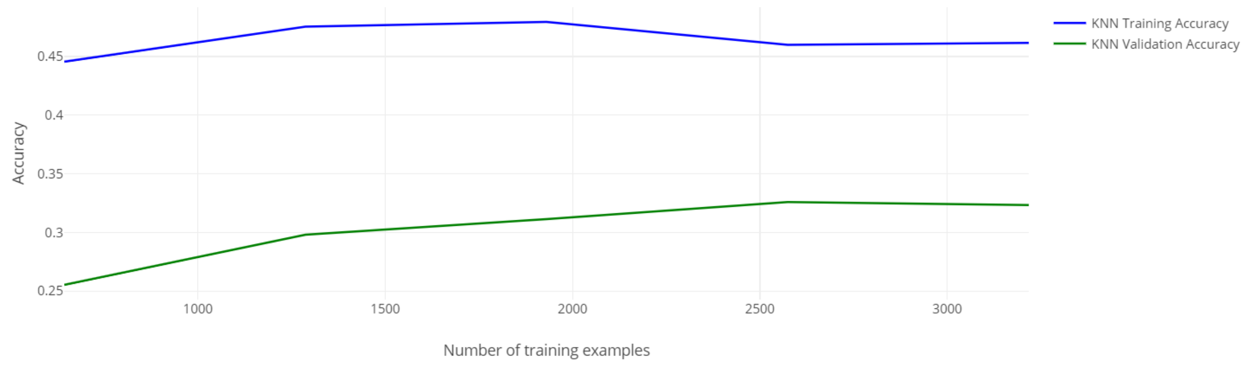
Notice how "Length" and "Diameter" have been removed

Learning Curves

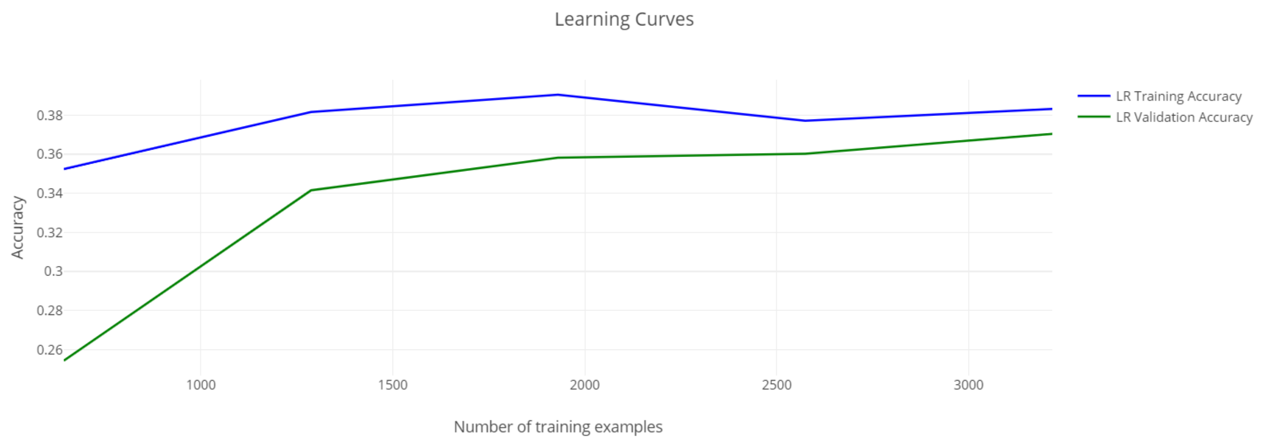
Decision Tree



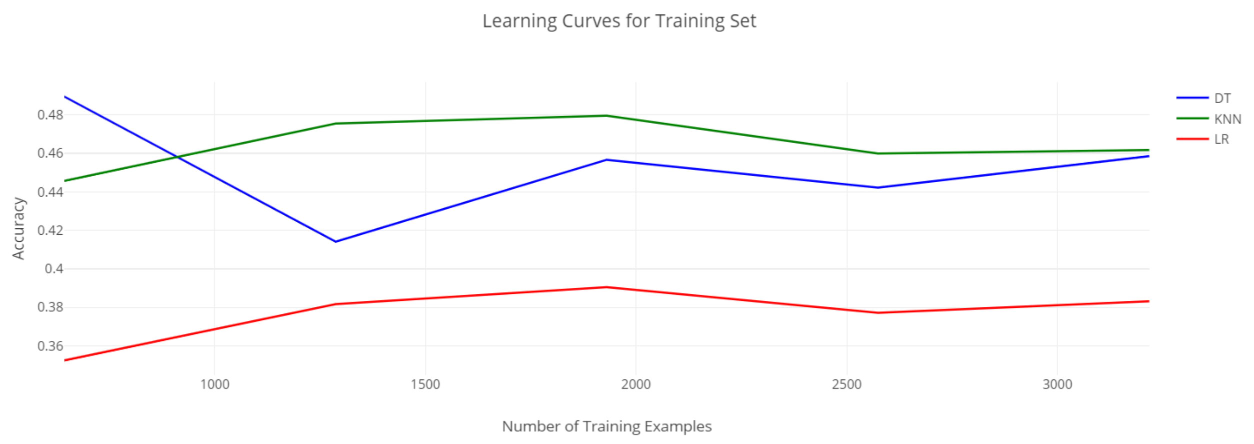
KNN

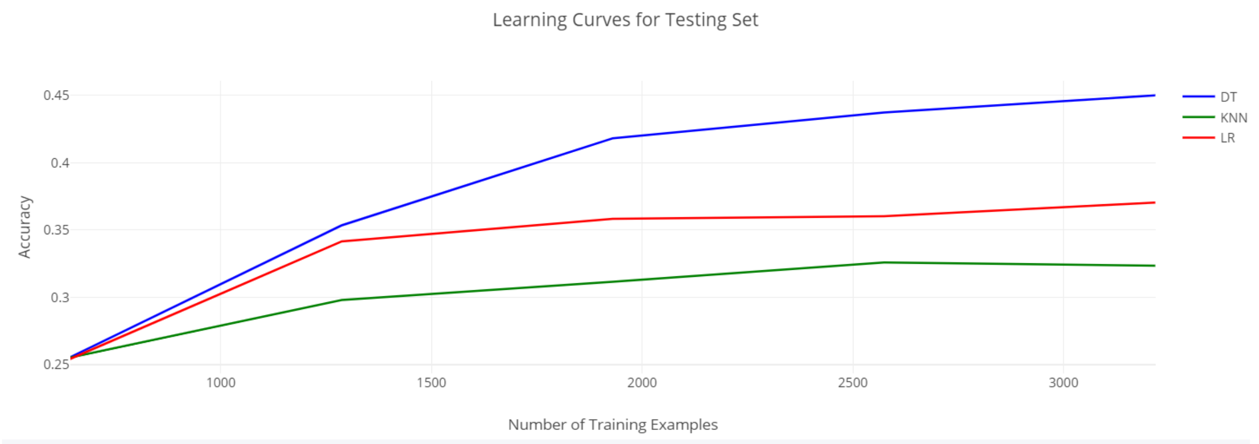


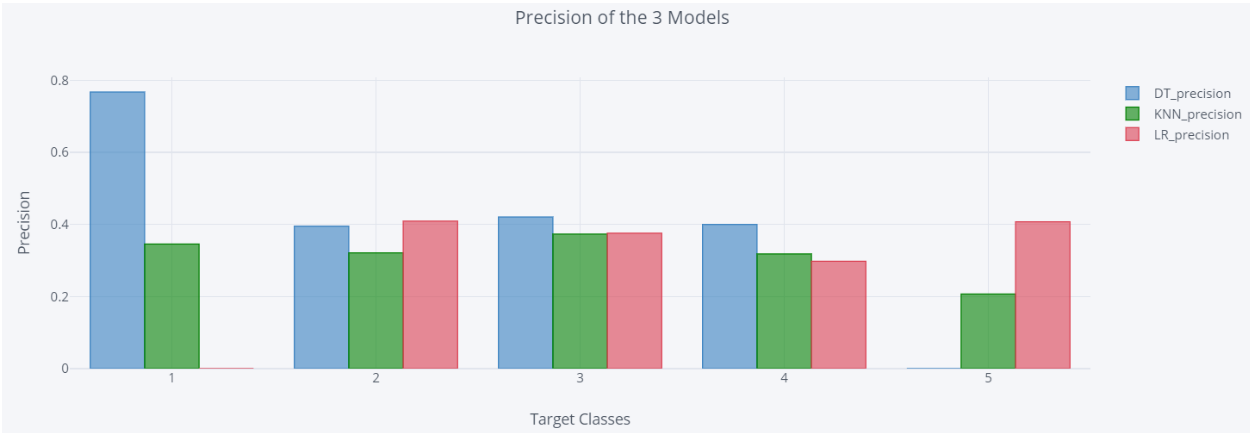
LR

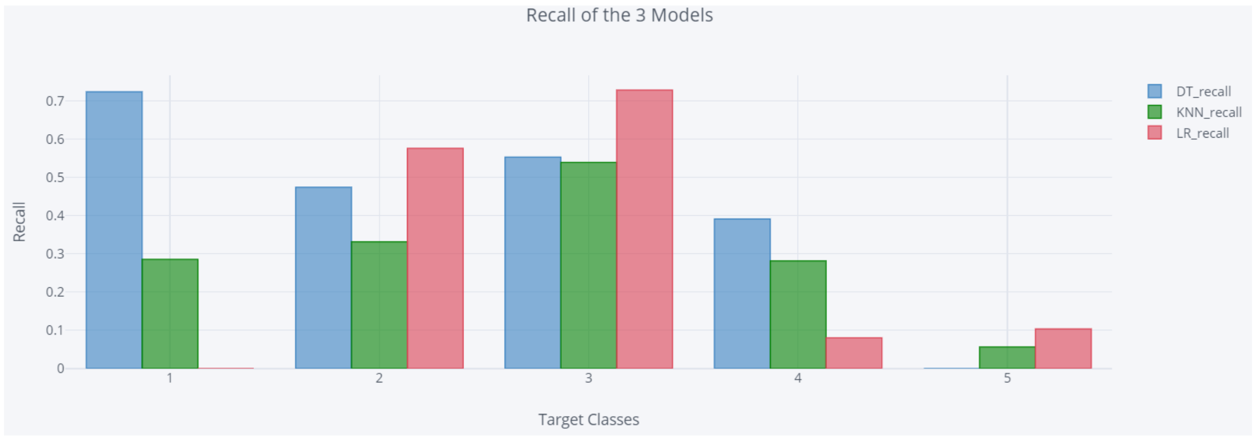


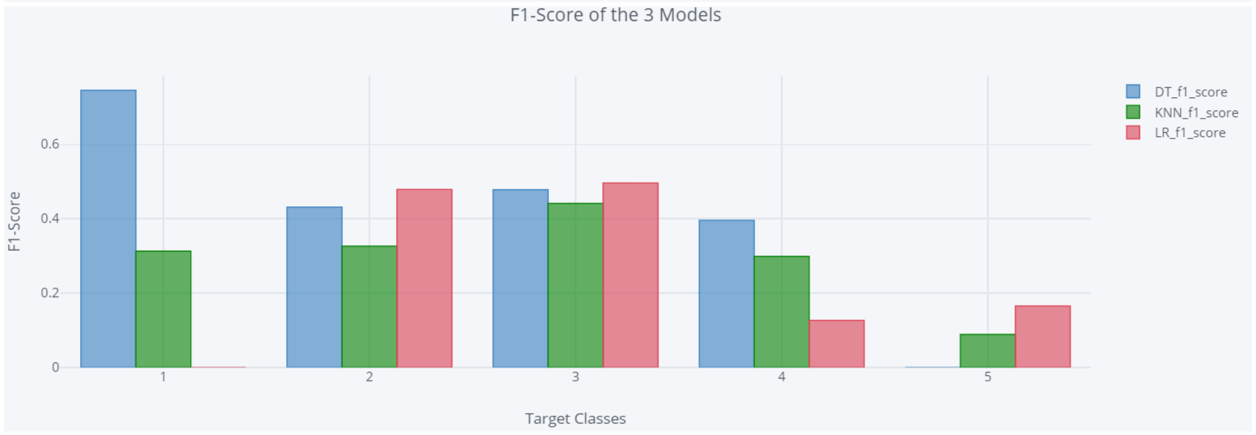
Comparative Analysis

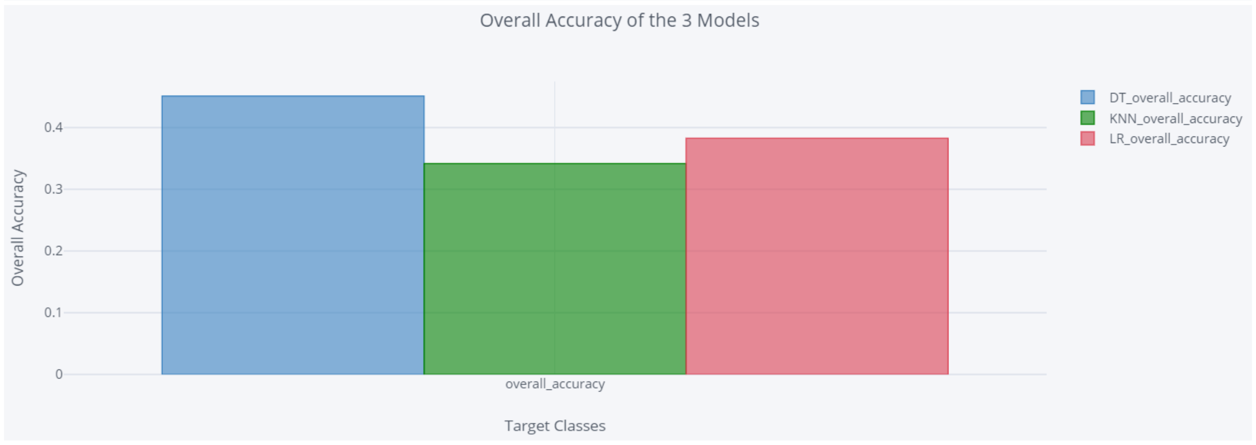












Interpretation of Results

Observations:

In general, the learning curves indicate that all the models are underfitting (given the training accuracy around 40%). This indicates that the data could've been pre-processed differently to enhance the results of the learning curves.
Even though the accuracy of KNN model is the least among the three, the f1-score bar chart tells us that the knn model is the only model that can actually predict all the 5 classes. Therefore, the KNN model is the most suitable model if we want to include all the target classes.
(it should be noted that the KNN model was run on the non-normalized model, yielding slightly higher learning curves of around 60% accuracy, but the testing accuracy was still low. Therefore, further modifications need to be done on the dataset before training the models on it)

Classification Problem 2: Online News Popularity

Problem Description

The problem is represented as to predict the number of shares that a news article can get. The problem with the dataset was that the target (Number of shares) consists of continuous values which cannot be applied in a classification model. Therefore, the number of shares need to be discritized into bins and considers these bnis as the classes that the models will classify on.

Dataset Description and Visualisation

0. url:	URL of the article
1. timedelta:	Days between the article publication and the dataset acquisition
2. n_tokens_title:	Number of words in the title
3. n_tokens_content:	Number of words in the content
4. n_unique_tokens:	Rate of unique words in the content
5. n_non_stop_words:	Rate of non-stop words in the content
6. n_non_stop_unique_tokens:	Rate of unique non-stop words in the content
7. num_hrefs:	Number of links
8. num_self_hrefs:	Number of links to other articles published by Mashable
9. num_imgs:	Number of images
10. num_videos:	Number of videos
11. average_token_length:	Average length of the words in the content
12. num_keywords:	Number of keywords in the metadata
13. data_channel_is_lifestyle:	Is data channel 'Lifestyle'?
14. data_channel_is_entertainment:	Is data channel 'Entertainment'?
15. data_channel_is_bus:	Is data channel 'Business'?
16. data_channel_is_socmed:	Is data channel 'Social Media'?
17. data_channel_is_tech:	Is data channel 'Tech'?
18. data_channel_is_world:	Is data channel 'World'?
19. kw_min_min:	Worst keyword (min. shares)
20. kw_max_min:	Worst keyword (max. shares)

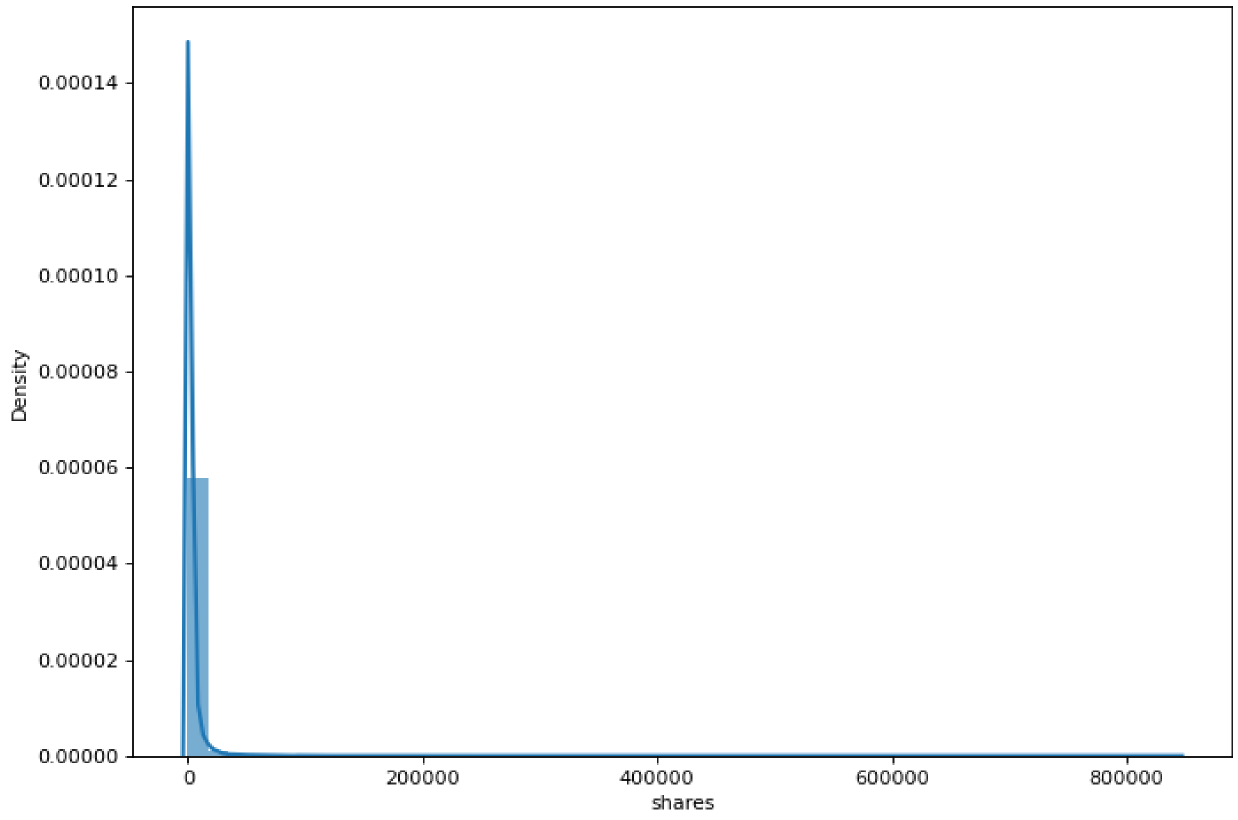
21. kw_avg_min:	Worst keyword (avg. shares)
22. kw_min_max:	Best keyword (min. shares)
23. kw_max_max:	Best keyword (max. shares)
24. kw_avg_max:	Best keyword (avg. shares)
25. kw_min_avg:	Avg. keyword (min. shares)
26. kw_max_avg:	Avg. keyword (max. shares)
27. kw_avg_avg:	Avg. keyword (avg. shares)
28. self_reference_min_shares:	Min. shares of referenced articles in Mashable
29. self_reference_max_shares:	Max. shares of referenced articles in Mashable
30. self_reference_avg_sharess:	Avg. shares of referenced articles in Mashable
31. weekday_is_monday:	Was the article published on a Monday?
32. weekday_is_tuesday:	Was the article published on a Tuesday?
33. weekday_is_wednesday:	Was the article published on a Wednesday?
34. weekday_is_thursday:	Was the article published on a Thursday?
35. weekday_is_friday:	Was the article published on a Friday?
36. weekday_is_saturday:	Was the article published on a Saturday?
37. weekday_is_sunday:	Was the article published on a Sunday?
38. is_weekend:	Was the article published on the weekend?
39. LDA_00:	Closeness to LDA topic 0
40. LDA_01:	Closeness to LDA topic 1
41. LDA_02:	Closeness to LDA topic 2
42. LDA_03:	Closeness to LDA topic 3
43. LDA_04:	Closeness to LDA topic 4
44. global_subjectivity:	Text subjectivity
45. global_sentiment_polarity:	Text sentiment polarity
46. global_rate_positive_words:	Rate of positive words in the content
47. global_rate_negative_words:	Rate of negative words in the content
48. rate_positive_words:	Rate of positive words among non-neutral tokens
49. rate_negative_words:	Rate of negative words among non-neutral tokens
50. avg_positive_polarity:	Avg. polarity of positive words
51. min_positive_polarity:	Min. polarity of positive words
52. max_positive_polarity:	Max. polarity of positive words
53. avg_negative_polarity:	Avg. polarity of negative words
54. min_negative_polarity:	Min. polarity of negative words
55. max_negative_polarity:	Max. polarity of negative words
56. title_subjectivity:	Title subjectivity
57. title_sentiment_polarity:	Title polarity
58. abs_title_subjectivity:	Absolute subjectivity level
59. abs_title_sentiment_polarity:	Absolute polarity level
60. shares:	Number of shares (target)

Cleaning the data

- After reading the dataset into a dataframe it was seen that the names of the featires had a leading space so it had to be removed.
- The data had no null values nor duplicates
- The features URL and Timedelta do not contribute to th data and are irrelevant when it comes to predicting the number of shares, so they will be dropped.

Visualizing the data

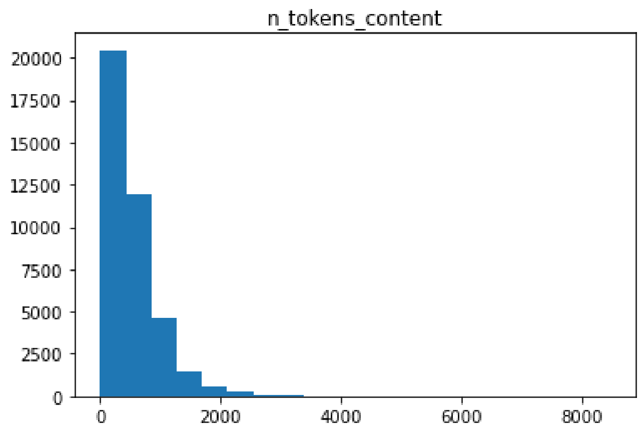
It is seen that the number of shares (target) itself is skewed and accumulated at a side which will cause complications while training.

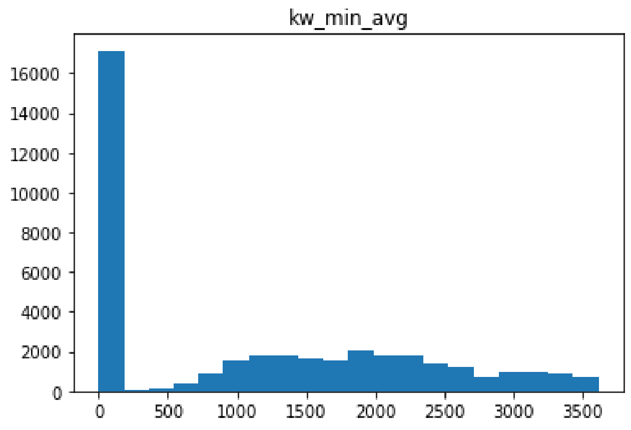


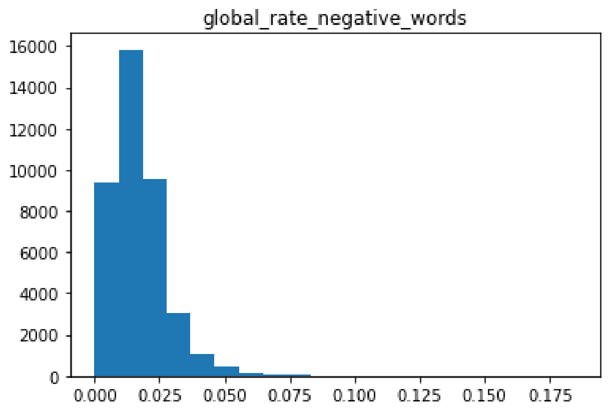
Dealing with Outliers:

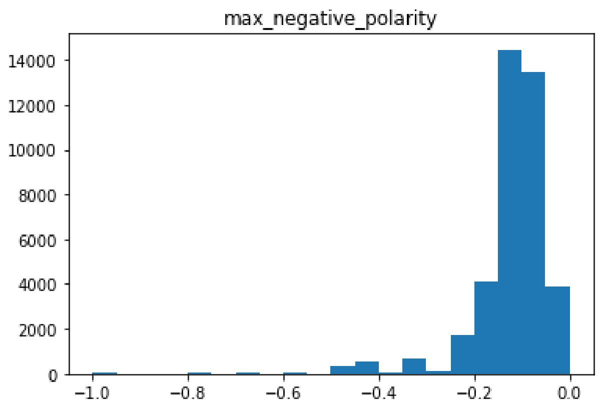
As seen in the shares graph, most of the data is skewed and most of it will be considered as outliers while they are in fact anomalies so we cannot trim any data. The IQR method was applied but it trimmed around 90% of the data and the training models did not perform well on these data because they accumulated a narrow range of values, so the model did not learn much.

Looking further into the distribution of the rest of the features, it is seen that most of the continuous data is not normally distributed.

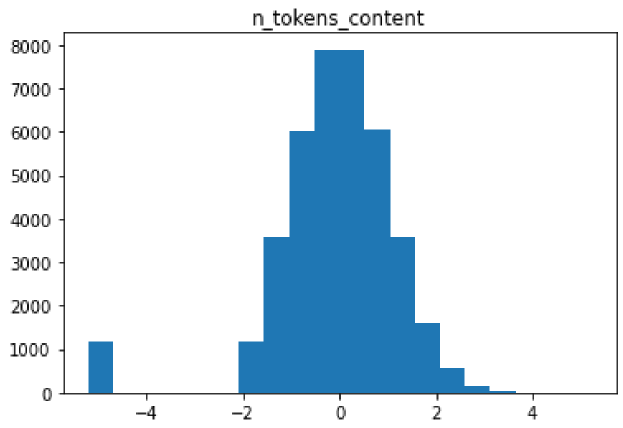


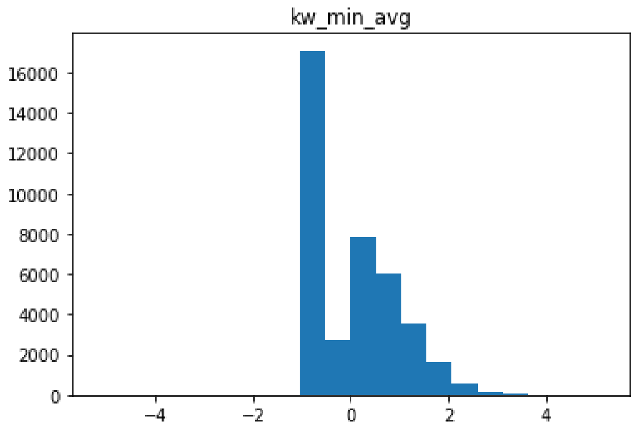


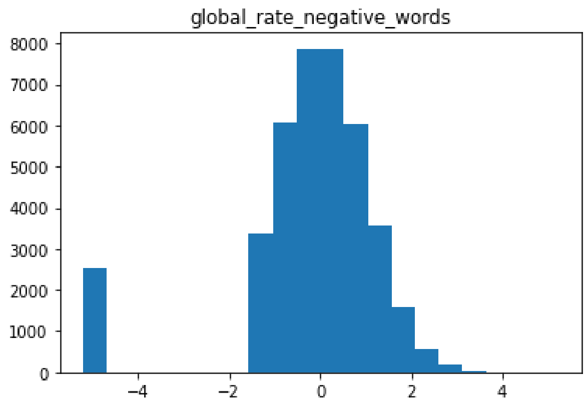


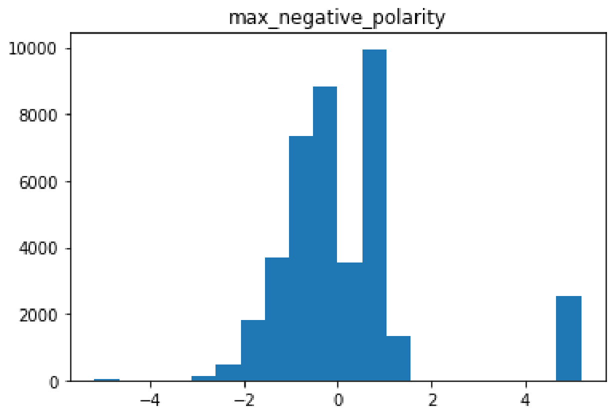


To solve this problem, th data will be normalized so we can adjust its distribution. Using function quartile transformer, it will normalize the data somehow and adjust it.









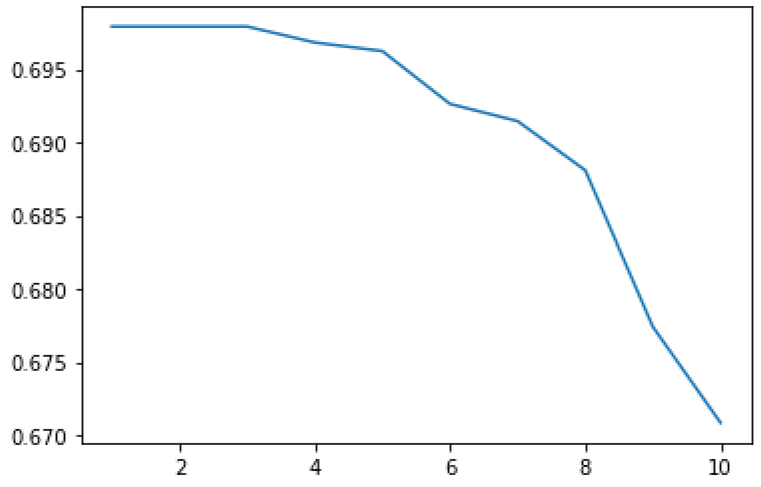
Next, we discretize the target to 5 bins using the KBinsDiscritizer. The function helped into distributing the data uniformly into appropriate ranges

Machine Learning Models Used

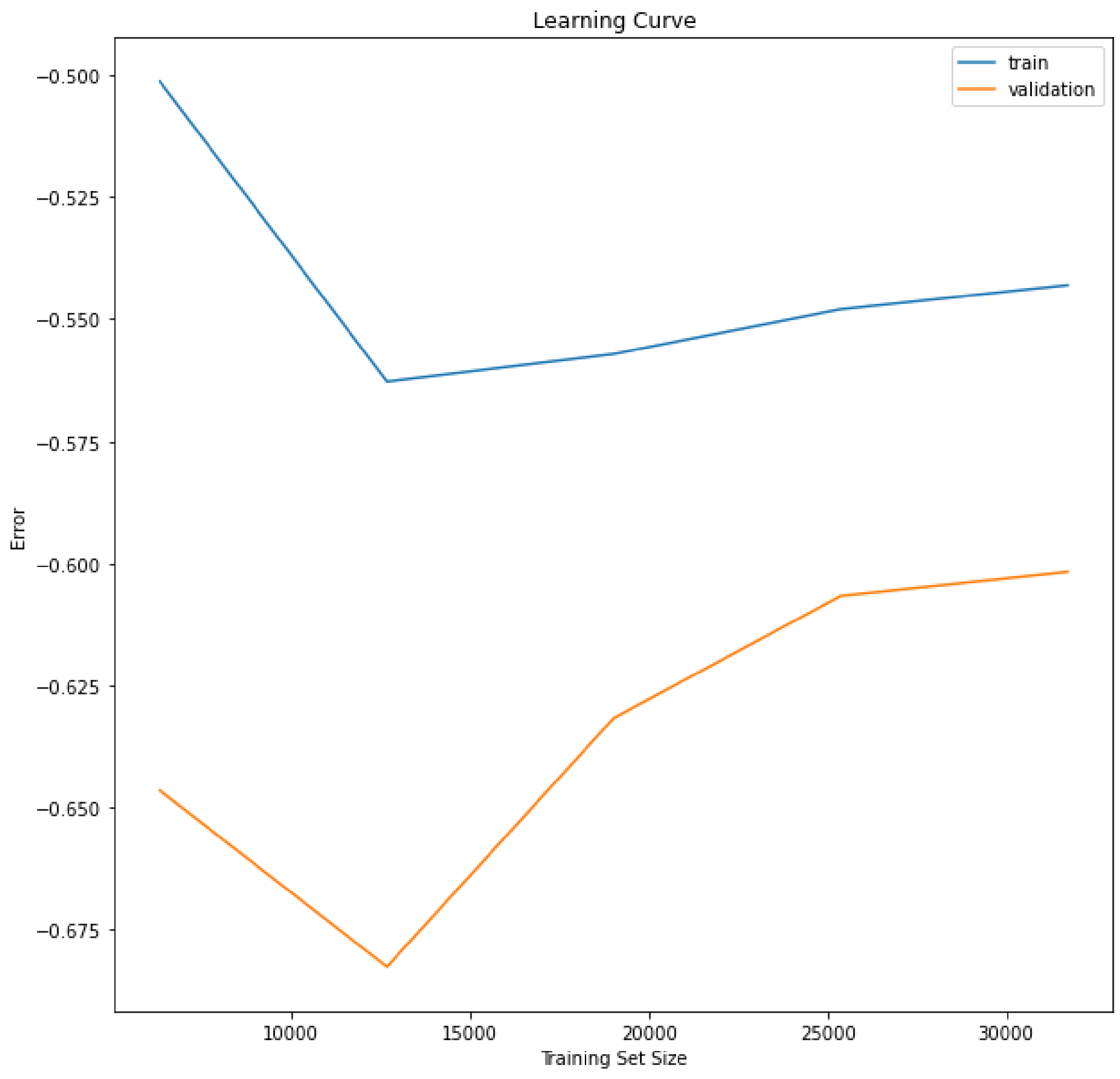
Now that the data is clean, standardised and ready for training, we split the data and feed it to the following models.

Decision Trees

Applying the decision trees, we first plot the data, then we tried different parameters for the decision trees. A maximum depth was put for the model to prune the tree in case overfitting happens.



After ploting the results we will find that the maximum depth that will produce best accuracy is at level 5. Then, we will train the final model.
On training the model, it gave us training accuracy of 70.5% and testing accuracy 69.6%


Learning Curve
train
validation
Error
−0.500
−0.525
−0.550
−0.575
−0.600
−0.625
−0.650
−0.675
10000
15000
20000
25000
30000
Training Set Size

Classification Report:

```
precision    recall  f1-score   support

        0.0       0.00      0.00      0.00         9
        1.0       0.47      0.03      0.06      1808
        2.0       0.70      0.97      0.82      8301
        3.0       0.34      0.06      0.10      1770
        4.0       0.00      0.00      0.00         6

    accuracy                           0.69     11894
   macro avg       0.30      0.21      0.19     11894
weighted avg       0.61      0.69      0.59     11894
```

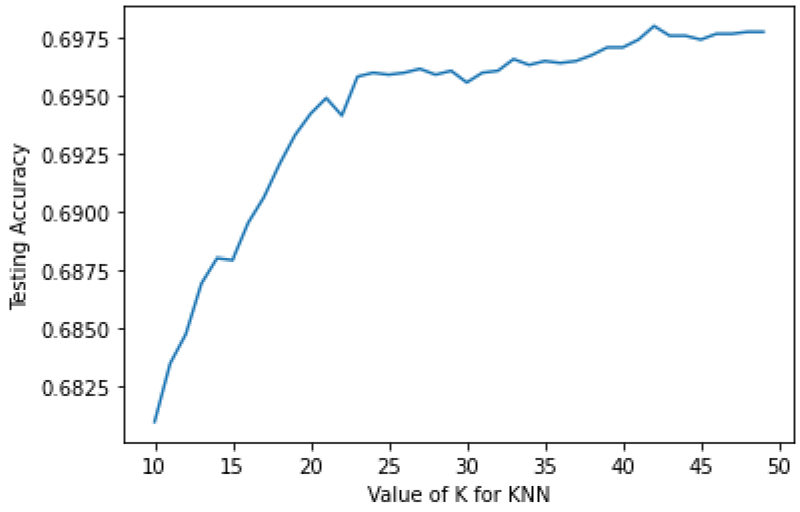
Applying Feature selection

Both Forward and Backward feature selection were applied so we can compare between them.
The accuracy that came after from the backward and forward was around the same accuracy that was done before the feature selection. But the features election selected around 15 features that got the same accuracy so it succeeded at its purpose.

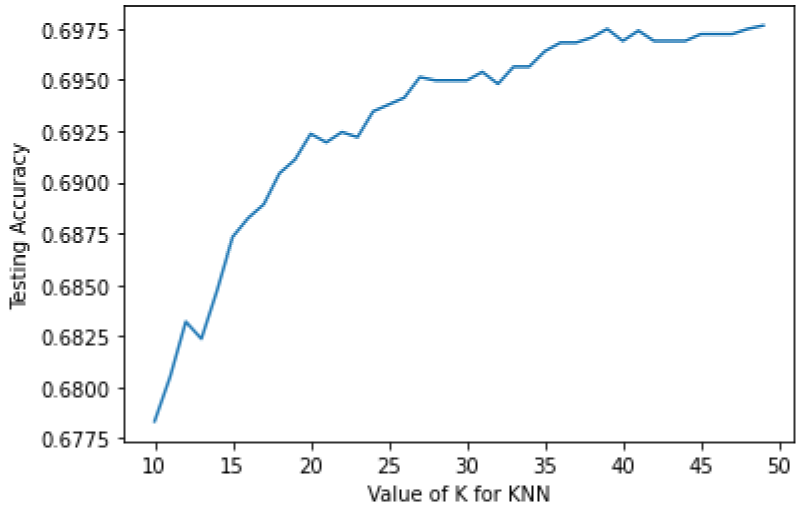
K-nearest Neighbour

For the K-nearest neighbour, many metrics were tested to see the best metric that fit the data. It is seen that the minkowski was the best.

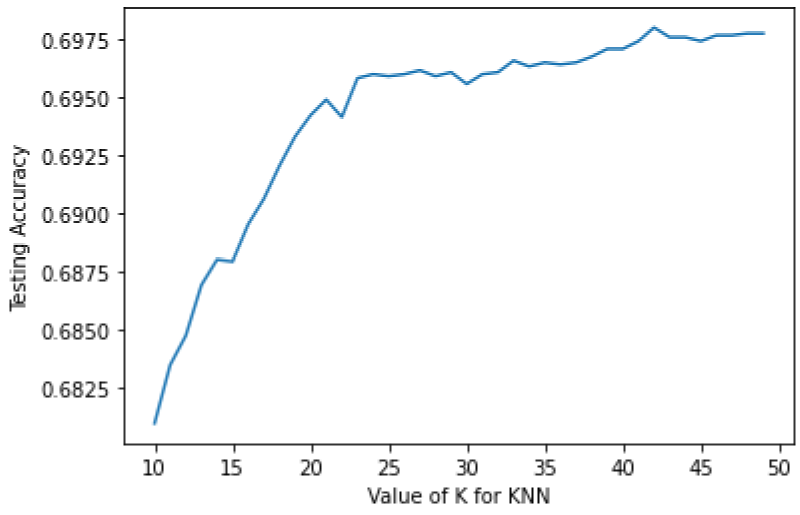
Using euclidean:



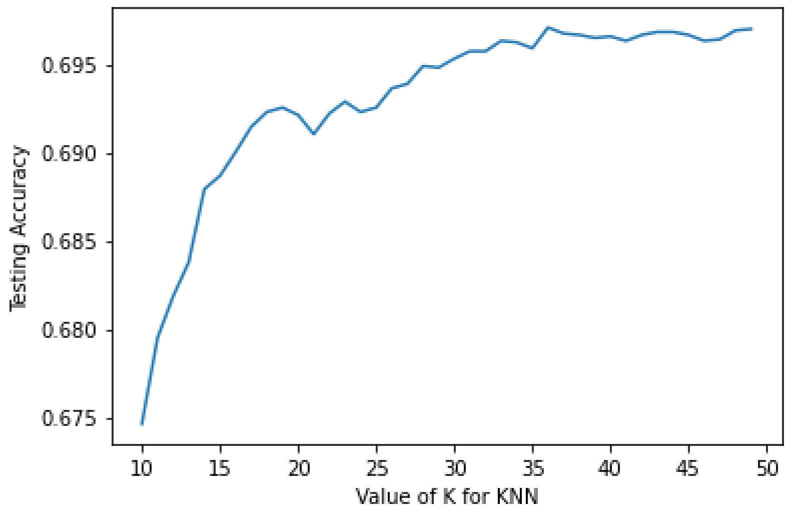
Using Manhattan:



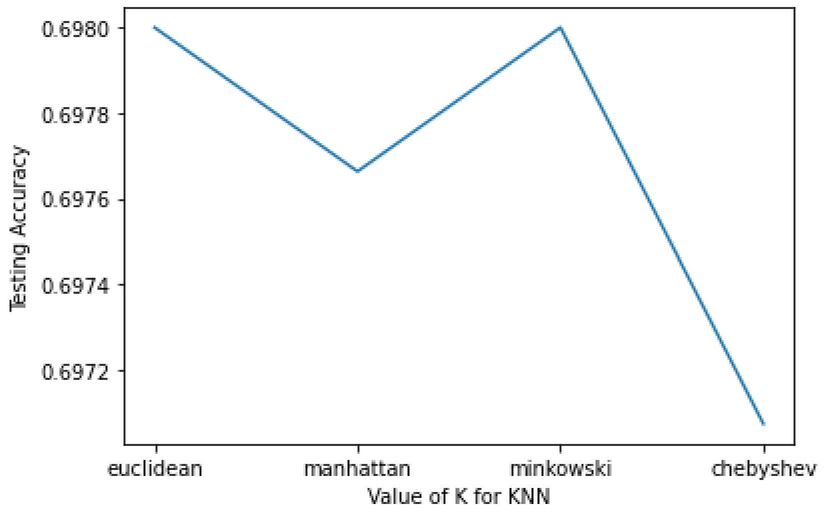
Using minkowski:



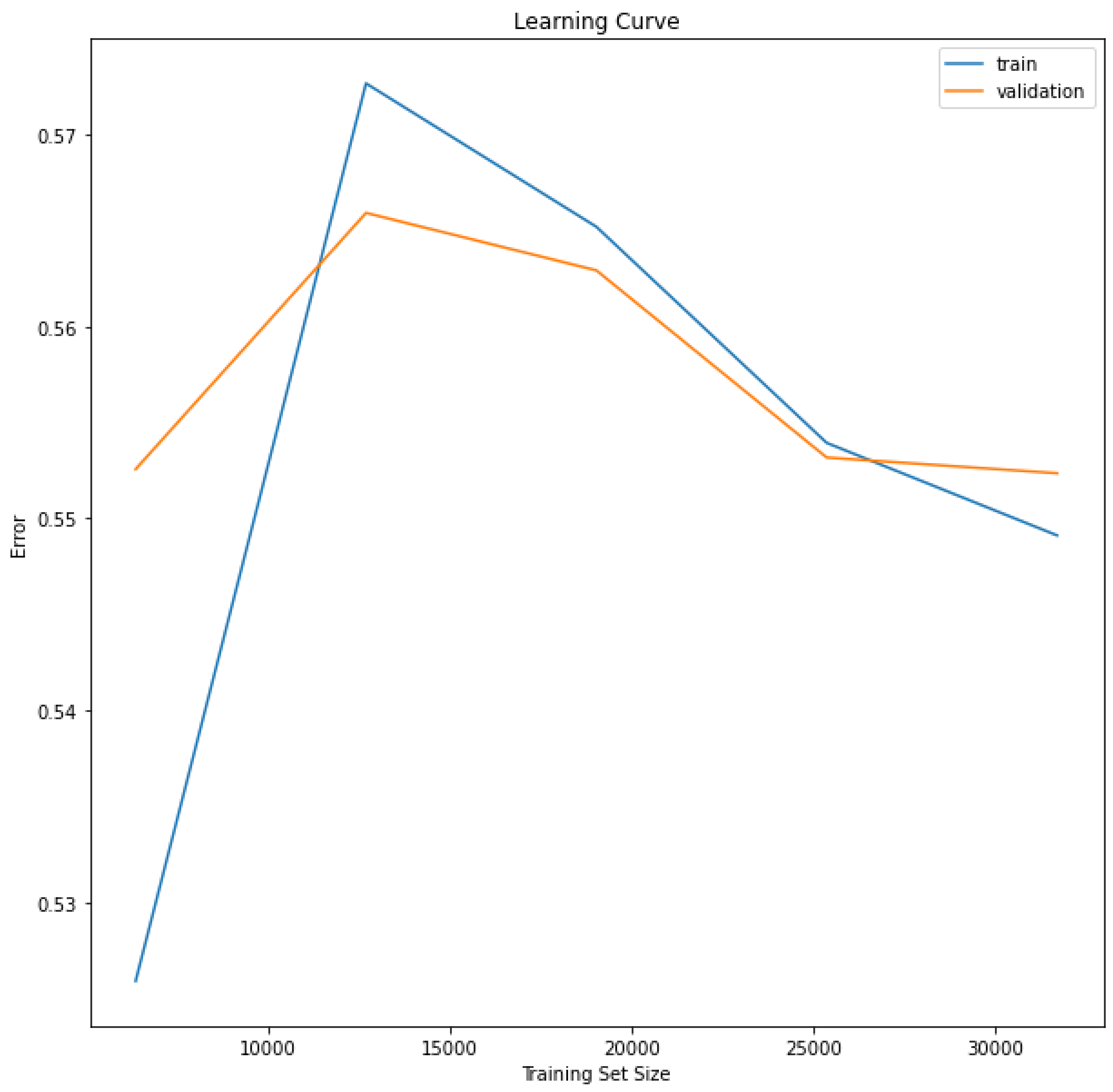
Using chebyshev:



Comparing between Matrics:



Therefore, we will look back at the graph for the minkowski and select the best k to train on which is around 37.



The accuracy that was hit was 70% for training and 69% for testing which is the same results as the decision trees.

Classification Report

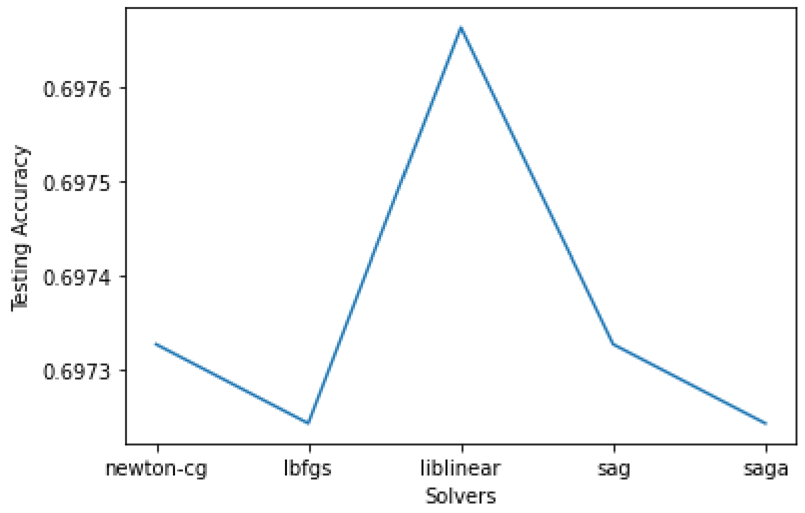
```
precision    recall  f1-score   support

         0.0       0.00      0.00      0.00         9
         1.0       0.38      0.01      0.02      1808
         2.0       0.70      1.00      0.82      8301
         3.0       0.10      0.00      0.00      1770
         4.0       0.00      0.00      0.00         6

    accuracy                           0.70     11894
   macro avg       0.23      0.20      0.17     11894
weighted avg       0.56      0.70      0.58     11894
```

Applying Feature selection

Both Forward and Backward feature selection were applied so we can compare between them.
The accuracy that came after from the backward and forward was around the same accuracy that was done before the feature selection. But the features election selected around 15 features that got the same accuracy so it succeeded at its purpose.

Logistic Regression

For the K-nearest neighbour, many metrics were tested to see the best metric that fit the data. It is seen that the liblinear scorer was the best.

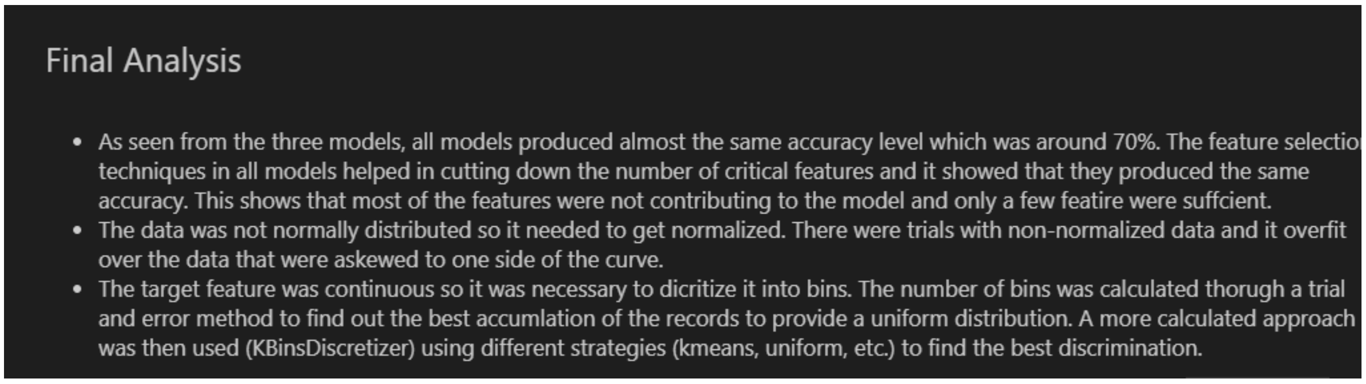


On training the model, it gave us training accuracy of 70.5% and testing accuracy 69.6%

Applying Feature selection

Both Forward and Backward feature selection were applied so we can compare between them.
The accuracy that came after from the backward and forward was around the same accuracy that was done before the feature selection. But the feature selection selected around 15 features that got the same accuracy so it succeeded at its purpose.

Interpretation of Results



Final Analysis

- As seen from the three models, all models produced almost the same accuracy level which was around 70%. The feature selectio techniques in all models helped in cutting down the number of critical features and it showed that they produced the same accuracy. This shows that most of the features were not contributing to the model and only a few featire were suffcient.
- The data was not normally distributed so it needed to get normalized. There were trials with non-normalized data and it overfit over the data that were askewed to one side of the curve.
- The target feature was continuous so it was necessary to dicritize it into bins. The number of bins was calculated thorugh a trial and error method to find out the best accumlation of the records to provide a uniform distribution. A more calculated approach was then used (KBinsDiscretizer) using different strategies (kmeans, uniform, etc.) to find the best discrimination.

Classification Problem 3: Cardiotocography Dataset

Problem Description

Cardiotocography or CTG is a method of using ultrasound to monitor fetal heartbeats to produce fetal heart rate (FHR) signals. The patterns depicted within these signals can provide vital information about the fetus's state, vitals and overall health.

Dataset Description and Visualisation

The dataset consists of the following features:
LB - FHR baseline (beats per minute)
AC - # of accelerations per second
FM - # of fetal movements per second
UC - # of uterine contractions per second
DL - # of light decelerations per second
DS - # of severe decelerations per second
DP - # of prolonged decelerations per second
ASTV - percentage of time with abnormal short term variability
MSTV - mean value of short term variability
ALTV - percentage of time with abnormal long term variability
MLTV - mean value of long term variability
Width - width of FHR histogram
Min - minimum of FHR histogram
Max - Maximum of FHR histogram
Nmax - # of histogram peaks
Nzeros - # of histogram zeros
Mode - histogram mode
Mean - histogram mean
Median - histogram median
Variance - histogram variance
Tendency - histogram tendency
CLASS - FHR pattern class code (1 to 10)
NSP - fetal state class code (1 to 3)

In which the feature CLASS consists of 10 labels as follows:
A - calm sleep
B - REM sleep
C - calm vigilance
D - active vigilance
SH - shift pattern (A or Susp with shifts)
AD - accelerative/decelerative pattern (stress situation)
DE - decelerative pattern (vagal stimulation)

LD - largely decelerative pattern
FS - flat-sinusoidal pattern (pathological state)
SUSP - suspect pattern

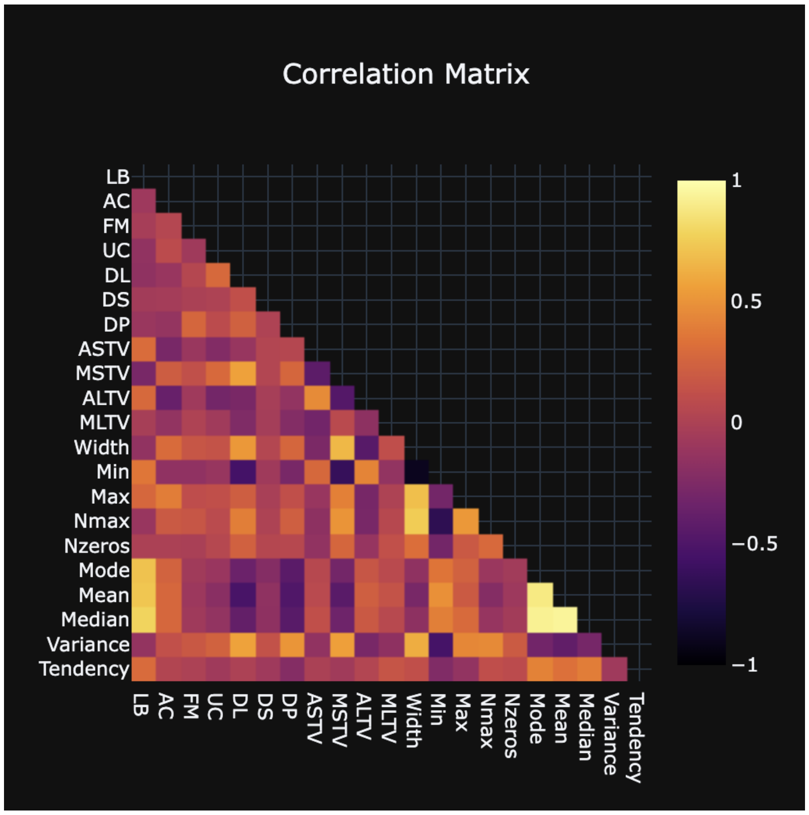
And NSP:
N - normal
S - suspect
P - pathologic

It is of note that the dataset houses two target classes, however only one will be used to build the model (CLASS and NSP). All data involved is numerical.

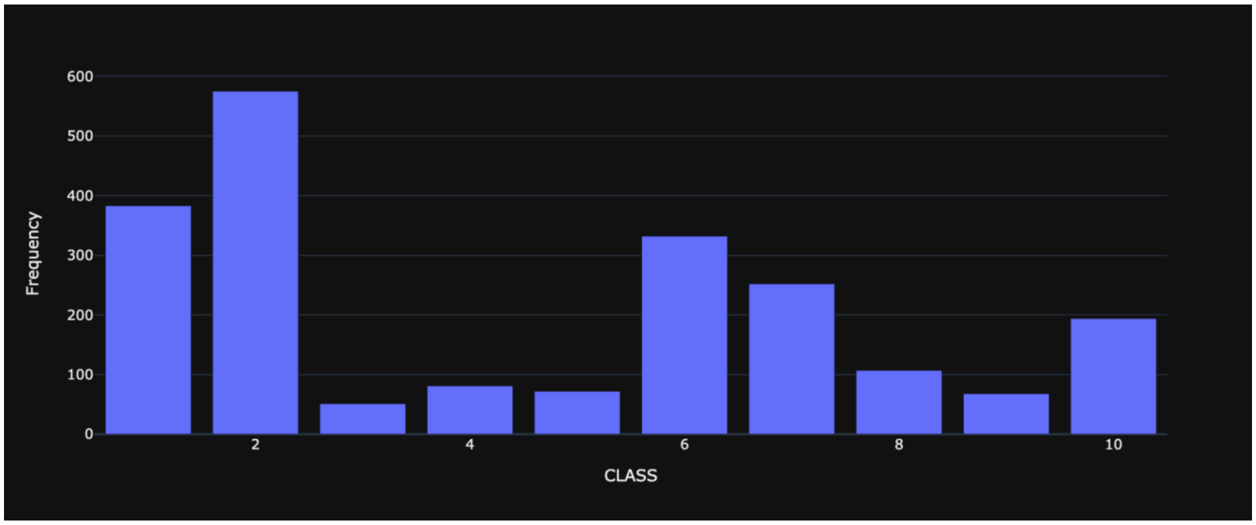
The dataset was first prepared as such:

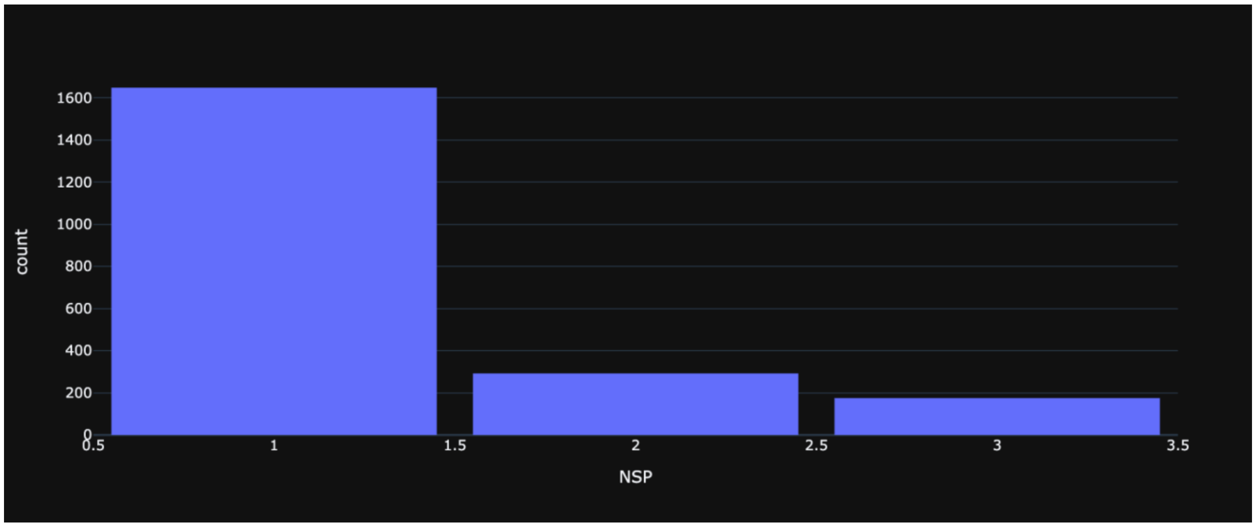
- Column headers fixed
- Empty columns (all null) removed
- Nulls removed
- Duplicates eliminated
- Checked for outliers
- Correcting data types
- Excess columns discarded (such as the extra target class and its encoding)

The correlation of the remaining features can be seen using a correlation matrix.



To determine which target class to work on, the distributions of both were visualized.





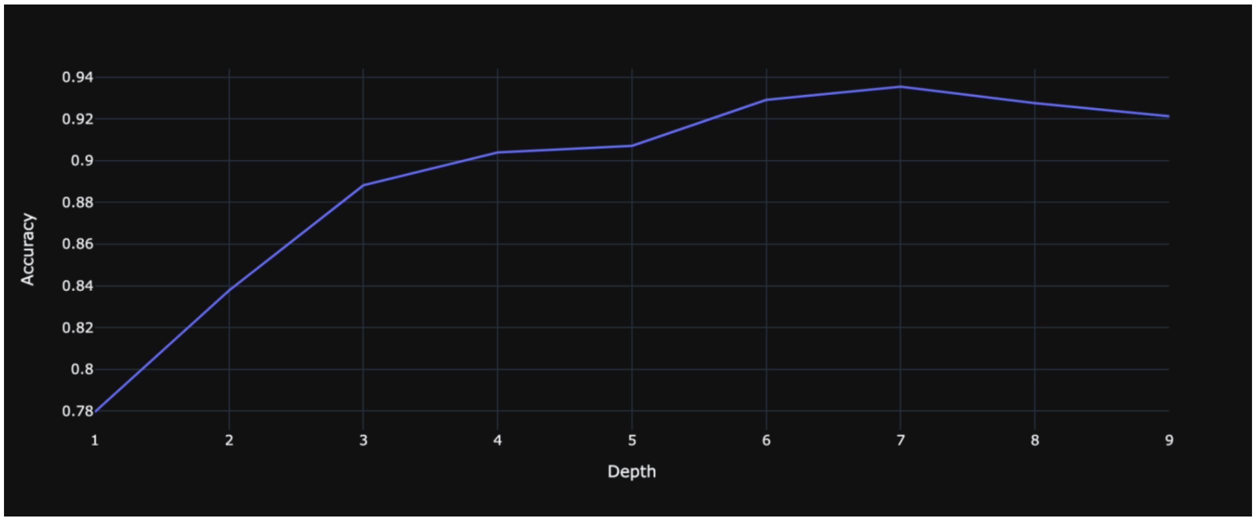
Since both are quite unbalanced, the choice between the two was made with the aim of overall applicability. The NSP target class will be utilized, as it determines the overall state of the fetus's health, rather than classifying the morphological pattern of the FHR signal. By using the NSP class, the data can be used to produce a more understandable and decisive observation regarding the fetus.

Machine Learning Models Used

The data was split into 70% training and 30% testing. Before implementing any models, feature selection via filtration was performed. The metric used to do so was ANOVA (analysis of variance) f-test, which involves calculating the ratio of variances, in cases with a categorical target class and numerical input data. The features with a score below 40 were removed. These were: FM, DS, Max, Nmax, Nzeros, and Tendency.

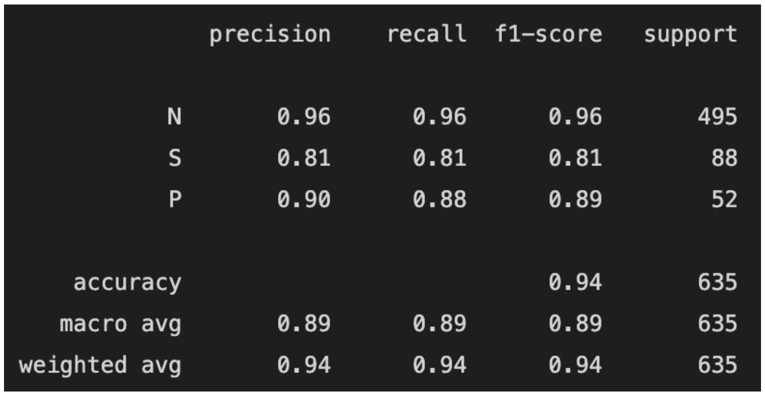
Decision Tree

Wrapper methods can now be applied, and Exhaustive Feature Selection (EFS) was used to refine the remaining features and locate the most relevant ones. These were found to be: LB, AC, UC, DL, DP, ASTV, ALTV, Mode, Mean, and Variance.
To figure out the best depth for the tree, multiple possibilities were tested and plotted.



It was found that the best depth for the tree classifier was 7.

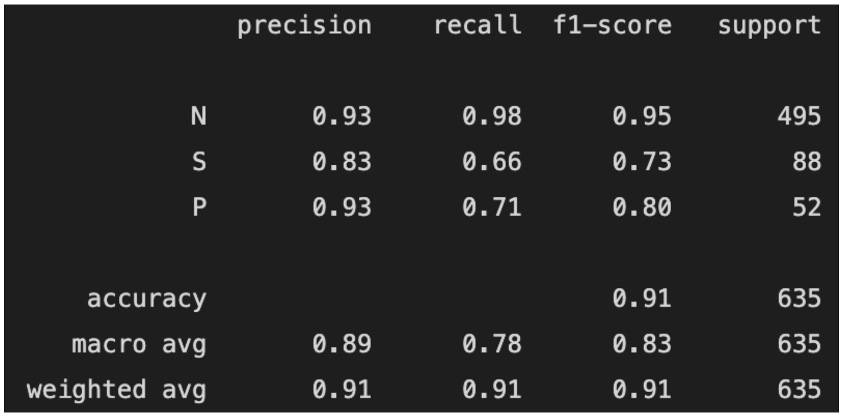
The best features and depth were then used to build and fit the model. The accuracy on the training set was 0.968, and that of the test set was 0.935. The classification report was as such:



```
              precision    recall  f1-score   support

           N       0.96      0.96      0.96       495
           S       0.81      0.81      0.81        88
           P       0.90      0.88      0.89        52

    accuracy                           0.94       635
   macro avg       0.89      0.89      0.89       635
weighted avg       0.94      0.94      0.94       635
```

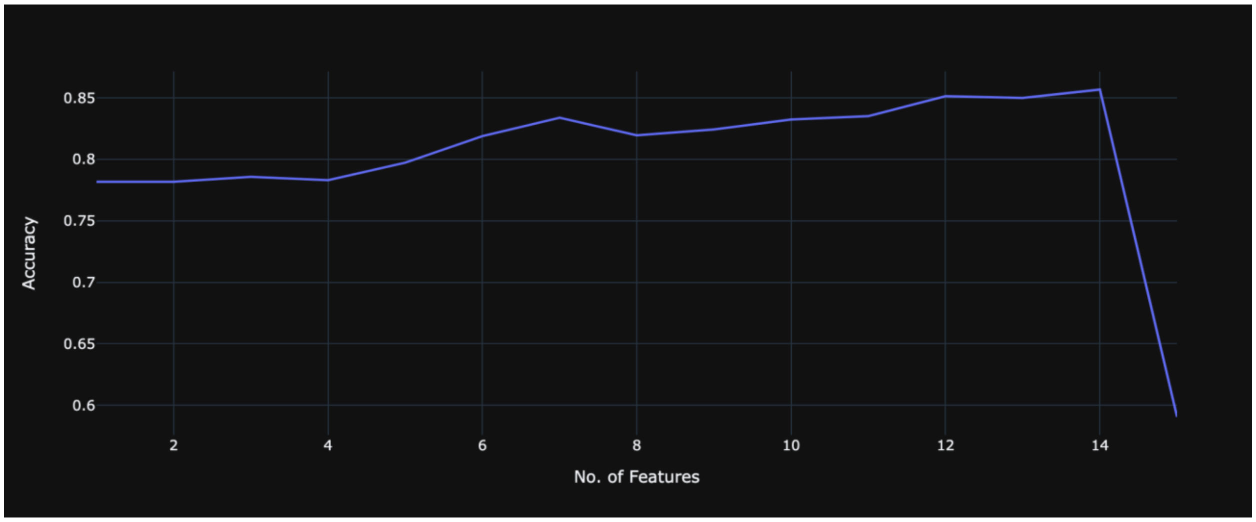
k-Nearest Neighbour

EFS was also implemented on the KNN classifier, producing this set of best features: LB, DL, DP, ASTV, ALTV, and Mode. Once again, hyperparameters were iterated on to determine which would be the best. The best number of neighbours to use was found to be 2, and the best metric was manhattan distance.
Accuracy on training set: 0.947, Accuracy on test set: 0.915, Classification Report:

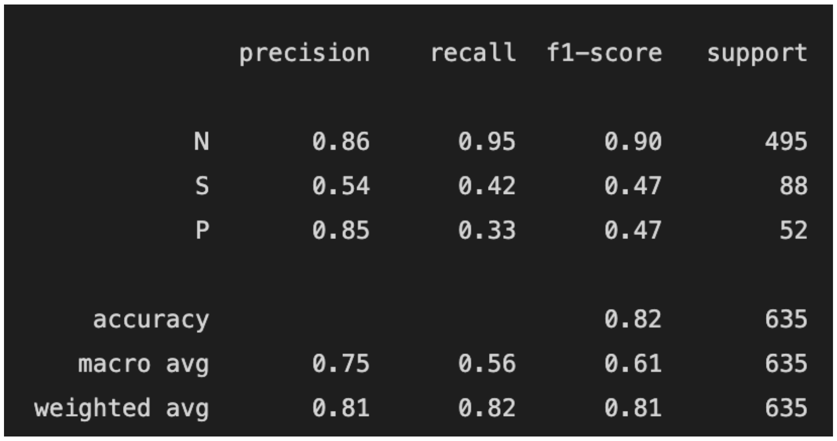

```
              precision    recall  f1-score   support

           N       0.93      0.98      0.95       495
           S       0.83      0.66      0.73        88
           P       0.93      0.71      0.80        52

    accuracy                           0.91       635
   macro avg       0.89      0.78      0.83       635
weighted avg       0.91      0.91      0.91       635
```

Rule-Based Classifier

The RIPPER (Repeated Incremental Pruning to Produce Error Reduction) algorithm was used as a rule-based classifier, taking on a One-versus-Rest approach for the multiclass nature of the problem.
Step Backwards Selection (SBS) was used to select a set of features. Even though it does not produce the optimal feature set, it is more computationally efficient, and will maintain a larger pool of features than its SFS counterpart. This set of features consisted of: LB, UC, DL, DP, ASTV, MSTV, ALTV, MLTV, Width, Min, Mode, Mean, Median, and Variance.

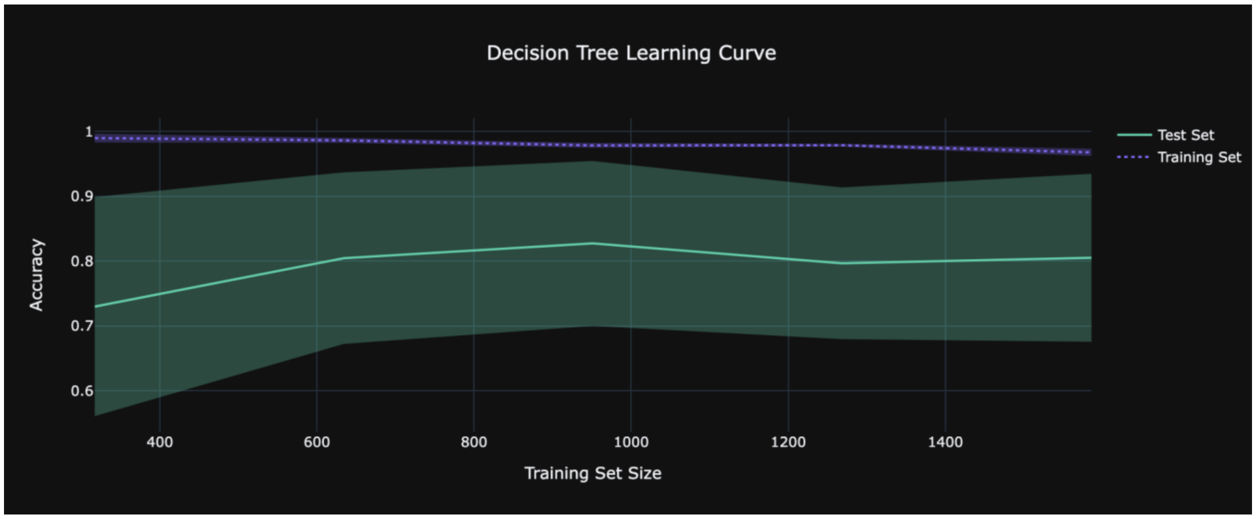


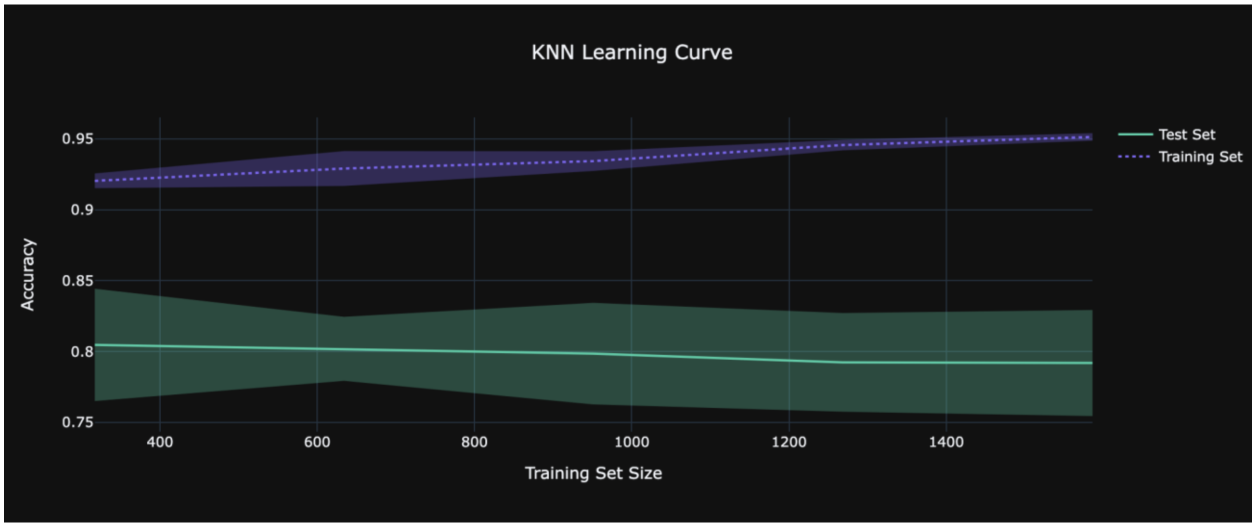
Different prune sizes and optimization run values were iterated on. The best were found to be 0.33 and 2 respectively. Accuracy on training set: 0.822, Accuracy on test set: 0.824, Classification Report:

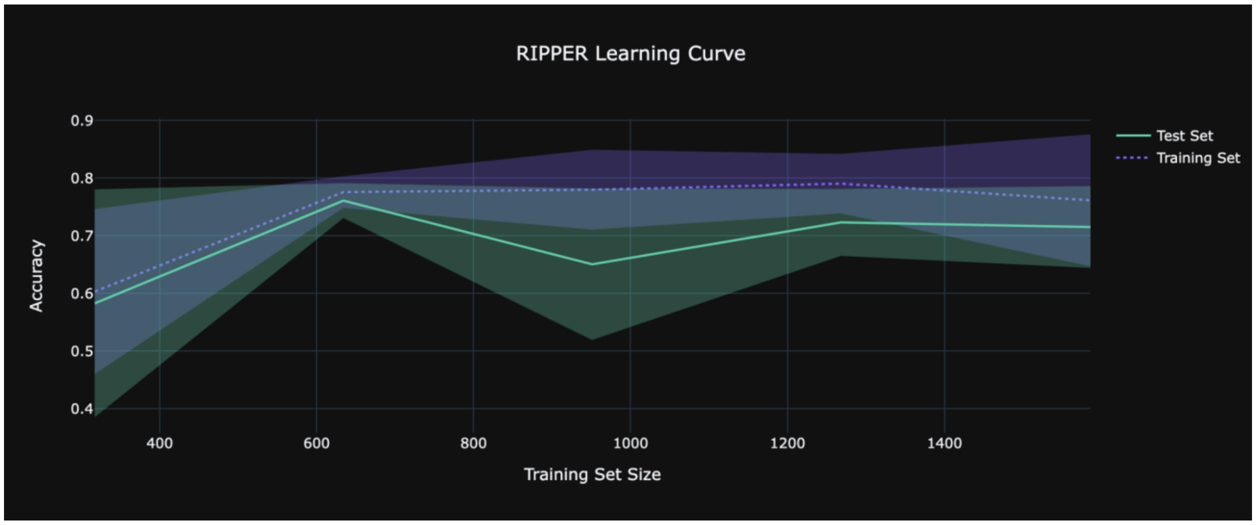


```
              precision    recall  f1-score   support

           N       0.86      0.95      0.90       495
           S       0.54      0.42      0.47        88
           P       0.85      0.33      0.47        52

    accuracy                           0.82       635
   macro avg       0.75      0.56      0.61       635
weighted avg       0.81      0.82      0.81       635
```

Learning Curves

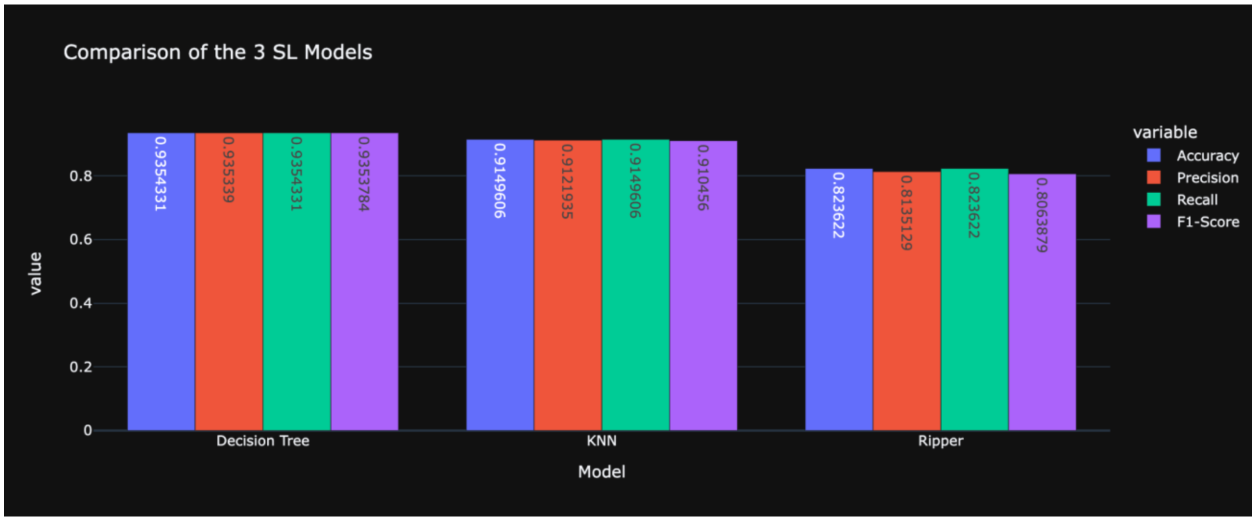
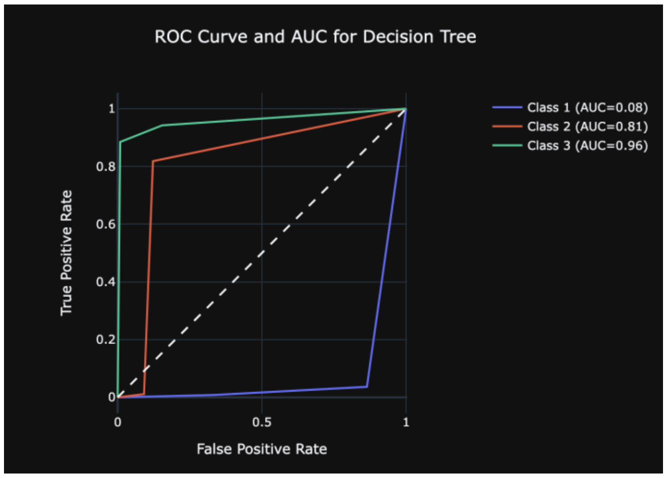
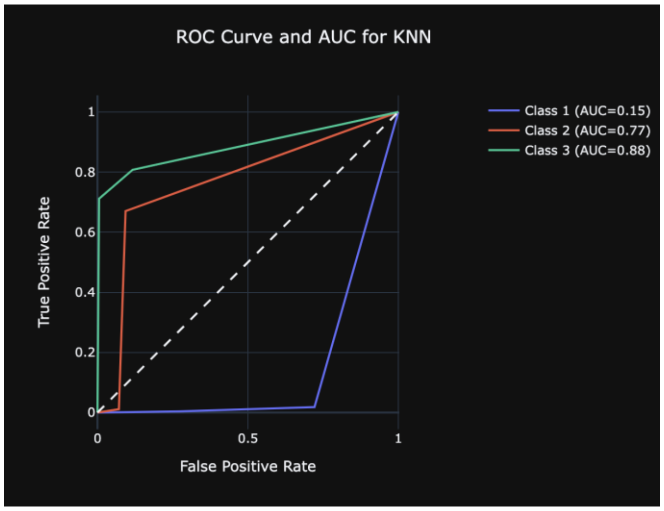


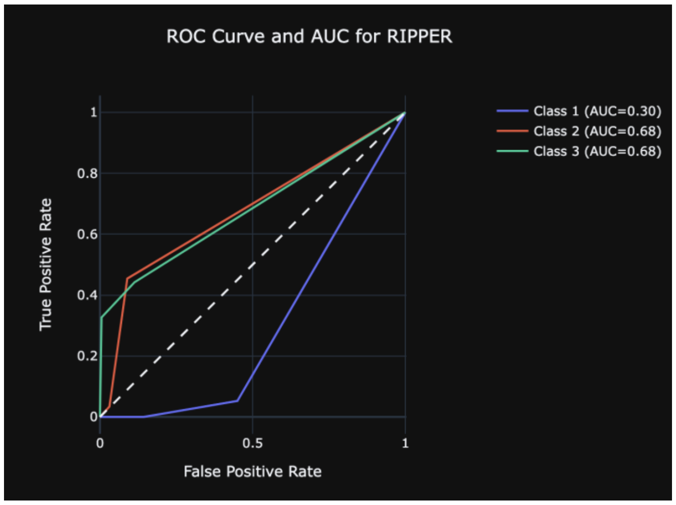




Comparative Analysis

The models were compared using four various metrics (Accuracy, Precision, Recall, and F1-Score). ROC (Receiver Operating Characteristics) curves (which plot the True Positive Rate against the False Positive Rate) were also drawn per class for each model, and the respective AUCs (Area Under Curve) calculated, which is said to ‘provide an aggregate measure of performance across all possible classification thresholds’.


Comparison of the 3 SL Models
value
Model
Decision Tree
KNN
Ripper
variable
Accuracy
Precision
Recall
F1-Score
0.9354331
0.935539
0.9354331
0.9353784
0.9149606
0.9121935
0.9149606
0.910456
0.823622
0.8135129
0.823622
0.8063879

ROC Curve and AUC for Decision Tree
True Positive Rate
False Positive Rate
Class 1 (AUC=0.08)
Class 2 (AUC=0.81)
Class 3 (AUC=0.96)

ROC Curve and AUC for KNN
True Positive Rate
False Positive Rate
Class 1 (AUC=0.15)
Class 2 (AUC=0.77)
Class 3 (AUC=0.88)



Interpretation of Results

The decision tree had the highest score for all of the four performance metrics, but was closely followed by the KNN classifier. The RIPPER rule-based classifier was the poorest of the three.
Likewise, when looking at the ROC curves, the decision tree classifier had the greatest AUC for both Class 2 (S) and Class 3 (P), and was negligibly smaller for Class 1 (N) than the KNN classifier. The rule-based classifier was once again the worst-performed overall, but intriguingly produced the greatest AUC for Class 1.

When looking at the learning curves, the KNN classifier showed signs of overfitting, as there was a discrepancy between the training accuracy and that of the testing. The RIPPER appeared to be a good fit, but had the poorest accuracy on both training and testing of the three. The learning curve of the decision tree classifier showed slight convergence towards the end, hinting that it could make a good fit if implemented on a larger dataset.

For the aforementioned reason, it is recommended that the decision tree be used in solving the CTG classification problem. The decision tree is also advantageous for this application, as it can provide a coherent justification behind how the classification occurred, which is often crucial in medical diagnosis.

Regression Problem: Advanced House Prices

Problem Description

The problem is represented in trying to predict the house price according to features that describe the house. Using regression models, we can predict the price.

Dataset Description and Visualisation

This dataset consists of a total of 80 features and the target with 1406 rows.

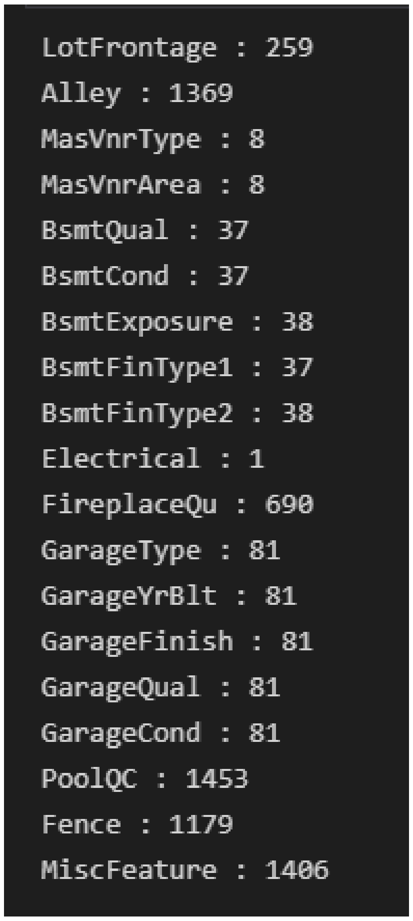
A Brief description of each feature:

- **SalePrice -** the property's sale price in dollars. This is the **target** variable that we are trying to predict.
- **MSSubClass**: The building class
- **MSZoning**: The general zoning classification
- **LotFrontage**: Linear feet of street connected to property
- **LotArea**: Lot size in square feet
- **Street**: Type of road access
- **Alley**: Type of alley access
- **LotShape**: General shape of property
- **LandContour**: Flatness of the property
- **Utilities**: Type of utilities available
- **LotConfig**: Lot configuration
- **LandSlope**: Slope of property
- **Neighborhood**: Physical locations within Ames city limits
- **Condition1**: Proximity to main road or railroad
- **Condition2**: Proximity to main road or railroad (if a second is present)
- **BldgType**: Type of dwelling
- **HouseStyle**: Style of dwelling
- **OverallQual**: Overall material and finish quality
- **OverallCond**: Overall condition rating
- **YearBuilt**: Original construction date
- **YearRemodAdd**: Remodel date
- **RoofStyle**: Type of roof
- **RoofMatl**: Roof material
- **Exterior1st**: Exterior covering on house
- **Exterior2nd**: Exterior covering on house (if more than one material)
- **MasVnrType**: Masonry veneer type
- **MasVnrArea**: Masonry veneer area in square feet
- **ExterQual**: Exterior material quality
- **ExterCond**: Present condition of the material on the exterior
- **Foundation**: Type of foundation
- **BsmtQual**: Height of the basement
- **BsmtCond**: General condition of the basement
- **BsmtExposure**: Walkout or garden level basement walls
- **BsmtFinType1**: Quality of basement finished area
- **BsmtFinSF1**: Type 1 finished square feet
- **BsmtFinType2**: Quality of second finished area (if present)
- **BsmtFinSF2**: Type 2 finished square feet

- **BsmtUnfSF**: Unfinished square feet of basement area
- **TotalBsmtSF**: Total square feet of basement area
- **Heating**: Type of heating
- **HeatingQC**: Heating quality and condition
- **CentralAir**: Central air conditioning
- **Electrical**: Electrical system
- **1stFlrSF**: First Floor square feet
- **2ndFlrSF**: Second floor square feet
- **LowQualFinSF**: Low quality finished square feet (all floors)
- **GrLivArea**: Above grade (ground) living area square feet
- **BsmtFullBath**: Basement full bathrooms
- **BsmtHalfBath**: Basement half bathrooms
- **FullBath**: Full bathrooms above grade
- **HalfBath**: Half baths above grade
- **Bedroom**: Number of bedrooms above basement level
- **Kitchen**: Number of kitchens
- **KitchenQual**: Kitchen quality
- **TotRmsAbvGrd**: Total rooms above grade (does not include bathrooms)
- **Functional**: Home functionality rating
- **Fireplaces**: Number of fireplaces
- **FireplaceQu**: Fireplace quality
- **GarageType**: Garage location
- **GarageYrBlt**: Year garage was built
- **GarageFinish**: Interior finish of the garage
- **GarageCars**: Size of garage in car capacity
- **GarageArea**: Size of garage in square feet
- **GarageQual**: Garage quality
- **GarageCond**: Garage condition
- **PavedDrive**: Paved driveway
- **WoodDeckSF**: Wood deck area in square feet
- **OpenPorchSF**: Open porch area in square feet
- **EnclosedPorch**: Enclosed porch area in square feet
- **3SsnPorch**: Three season porch area in square feet
- **ScreenPorch**: Screen porch area in square feet
- **PoolArea**: Pool area in square feet
- **PoolQC**: Pool quality
- **Fence**: Fence quality
- **MiscFeature**: Miscellaneous feature not covered in other categories
- **MiscVal**: $Value of miscellaneous feature
- **MoSold**: Month Sold
- **YrSold**: Year Sold
- **SaleType**: Type of sale
- **SaleCondition**: Condition of sale

The data had no duplicates but it had many nulls.

Dealing with Nulls

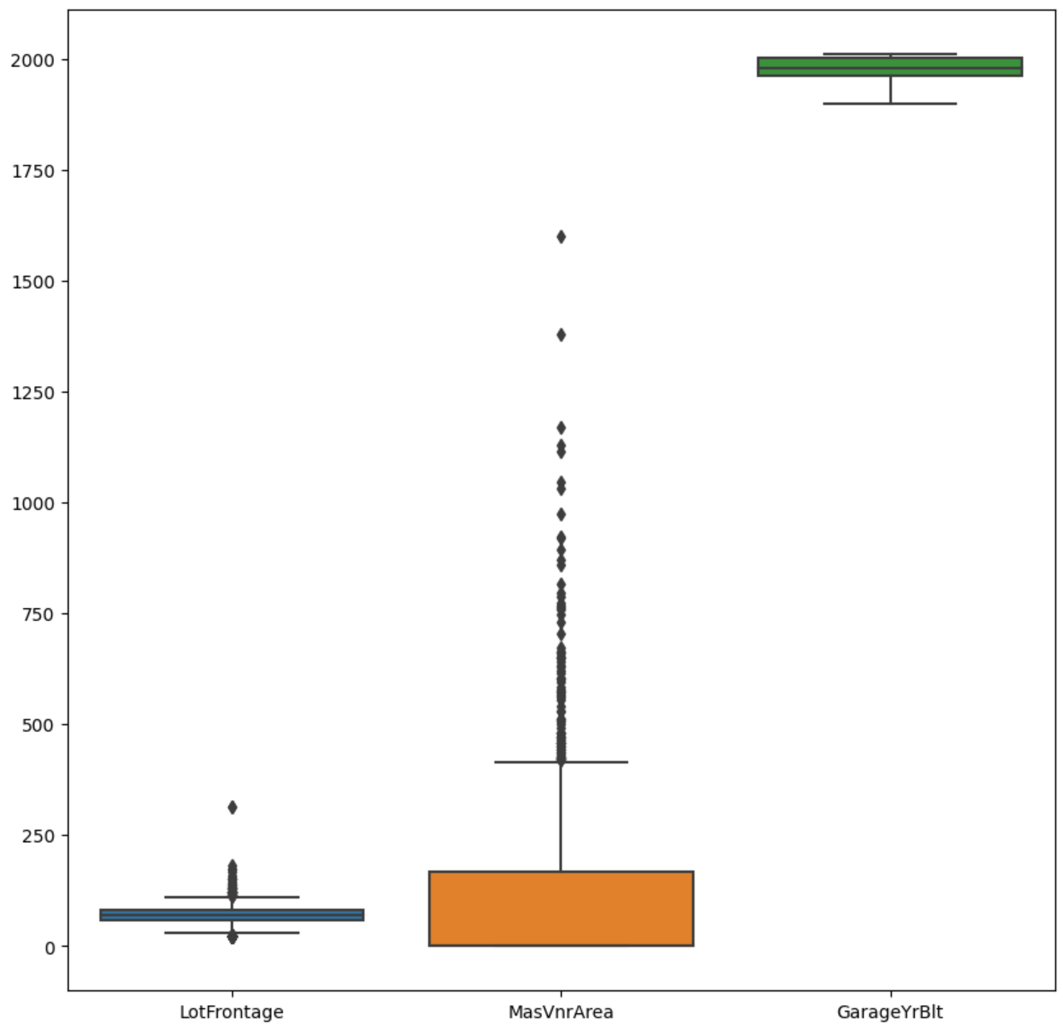


```
LotFrontage : 259
Alley : 1369
MasVnrType : 8
MasVnrArea : 8
BsmtQual : 37
BsmtCond : 37
BsmtExposure : 38
BsmtFinType1 : 37
BsmtFinType2 : 38
Electrical : 1
FireplaceQu : 690
GarageType : 81
GarageYrBlt : 81
GarageFinish : 81
GarageQual : 81
GarageCond : 81
PoolQC : 1453
Fence : 1179
MiscFeature : 1406
```

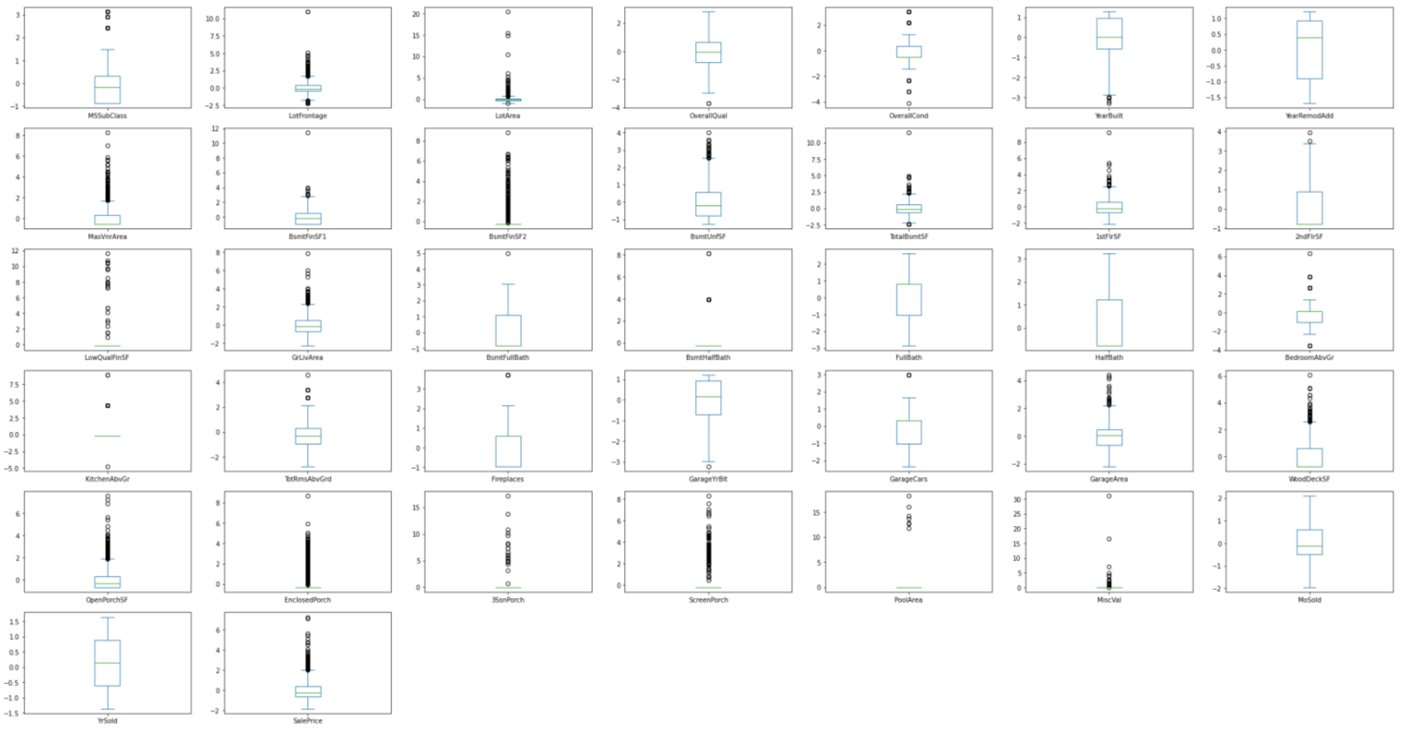
First we checked the sum of nulls in each feature to find some features that had more than 93% of the values are nulls. So, these attributes were dropped. The features that had less nulls were imputed based on their type. For example, the features that are continuous that had too many missing values (GarageType, GarageYeBlt, GarageFinish, etc..) were imputed with the mean of the mode. While the continuous features that didn’t have many nulls (MasVnrArea, MasVnrType, etc.) were imputed using the mean of the top five mode values.

Also, all categorical features were transformed to ordinal values (0,1, 2, etc.). This was done because based on the data description, it was found that most of the categorical data was ranked.

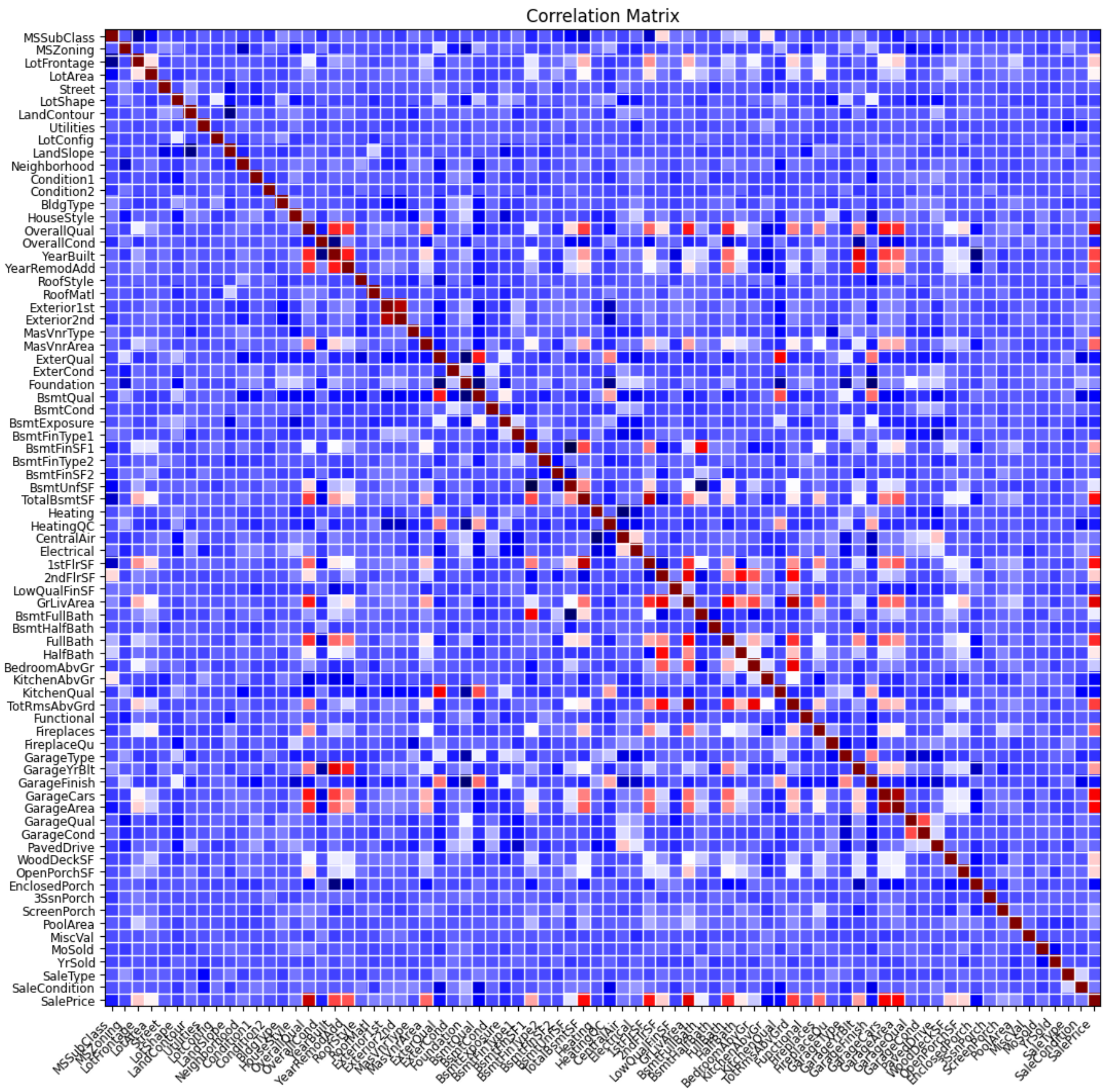
First, we visualized the continuous features that had many nulls to determine how to deal with them and any outliers they had.

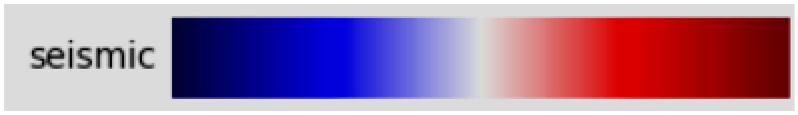


Visualizing the Data



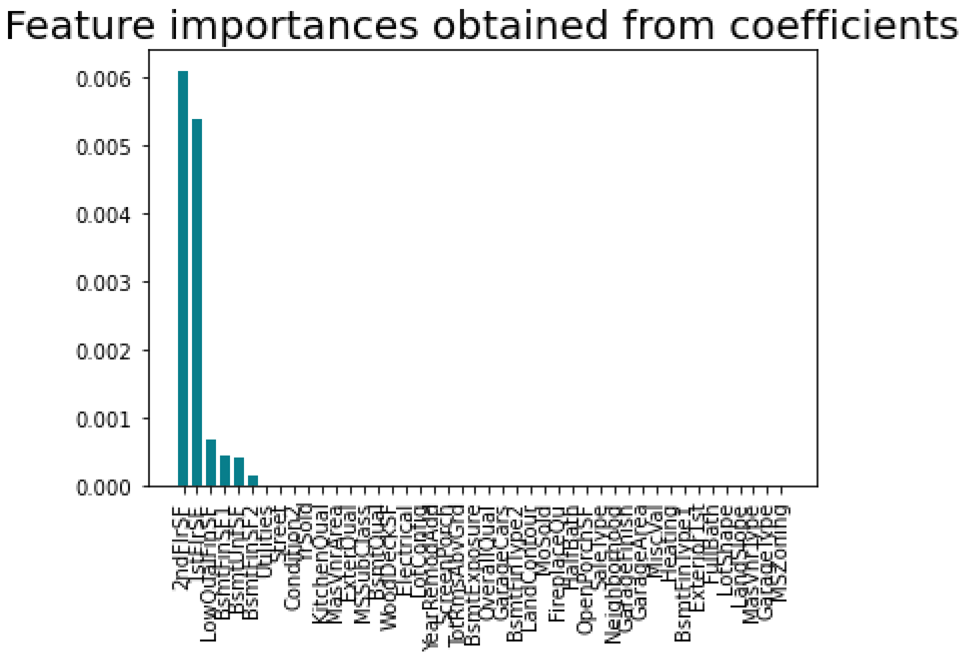
Measuring correlation between the features


Correlation Matrix
MSSubClass
MSZoning
LotFrontage
LotArea
Street
LotShape
LandContour
Utilities
LotConfig
LandSlope
Neighborhood
Condition1
Condition2
BldgType
HouseStyle
OverallQual
OverallCond
YearBuilt
YearRemodAdd
RoofStyle
RoofMatl
Exterior1st
Exterior2nd
MasVnrType
MasVnrArea
ExterQual
ExterCond
Foundation
BsmtQual
BsmtCond
BsmtExposure
BsmtFinType1
BsmtFinSF1
BsmtFinType2
BsmtFinSF2
BsmtUnfSF
TotalBsmtSF
Heating
HeatingQC
CentralAir
Electrical
1stFlrSF
2ndFlrSF
LowQualFinSF
GrLivArea
BsmtFullBath
BsmtHalfBath
FullBath
HalfBath
BedroomAbvGr
KitchenAbvGr
KitchenQual
TotRmsAbvGrd
Functional
Fireplaces
FireplaceQu
GarageType
GarageYrBlt
GarageFinish
GarageCars
GarageArea
GarageQual
GarageCond
PavedDrive
WoodDeckSF
OpenPorchSF
EnclosedPorch
3SsnPorch
ScreenPorch
PoolArea
MiscVal
MoSold
YrSold
SaleType
SaleCondition
SalePrice


seismic

Feature Selection

We used the feature importance measurement to determine which features are important and which are not:
Alot of coefficients came out so we will be plotting the ones that are > 0

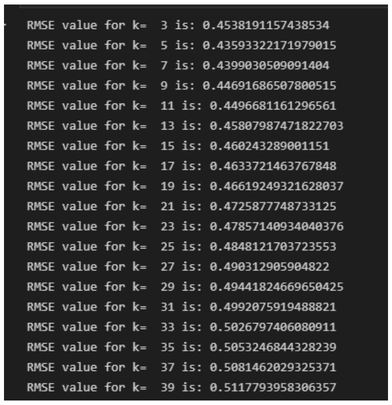


Machine Learning Models Used

We will be using decision trees for regression, Linear regression and K-nearest neighbour as regression models, then we will compare the results of each model.

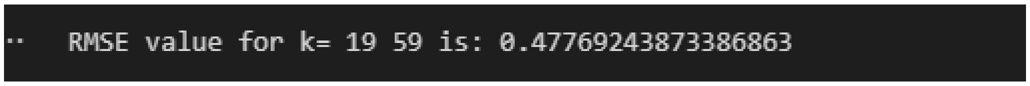
Decision Trees

Before Feature Selection:



```
RMSE value for k=  3 is: 0.4538191157438534
RMSE value for k=  5 is: 0.43593322171979015
RMSE value for k=  7 is: 0.4399030509091404
RMSE value for k=  9 is: 0.44691686507800515
RMSE value for k=  11 is: 0.4496681161296561
RMSE value for k=  13 is: 0.45807987471822703
RMSE value for k=  15 is: 0.460243289001151
RMSE value for k=  17 is: 0.4633721463767848
RMSE value for k=  19 is: 0.46619249321628037
RMSE value for k=  21 is: 0.4725877748733125
RMSE value for k=  23 is: 0.47857140934040376
RMSE value for k=  25 is: 0.4848121703723553
RMSE value for k=  27 is: 0.490312905904822
RMSE value for k=  29 is: 0.49441824669650425
RMSE value for k=  31 is: 0.4992075919488821
RMSE value for k=  33 is: 0.5026797406080911
RMSE value for k=  35 is: 0.5053246844328239
RMSE value for k=  37 is: 0.5081462029325371
RMSE value for k=  39 is: 0.5117793958306357
```

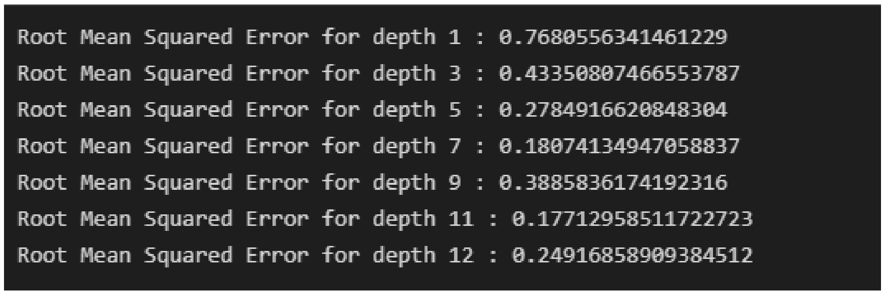
AfterFeature Selection:



```
··  RMSE value for k= 19 59 is: 0.47769243873386863
```

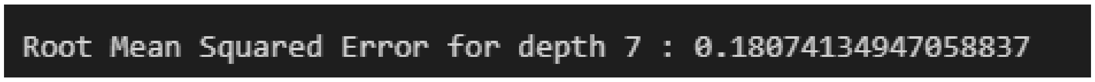
KNN

Before Feature Selection:



```
Root Mean Squared Error for depth 1 : 0.7680556341461229
Root Mean Squared Error for depth 3 : 0.43350807466553787
Root Mean Squared Error for depth 5 : 0.2784916620848304
Root Mean Squared Error for depth 7 : 0.18074134947058837
Root Mean Squared Error for depth 9 : 0.3885836174192316
Root Mean Squared Error for depth 11 : 0.17712958511722723
Root Mean Squared Error for depth 12 : 0.24916858909384512
```

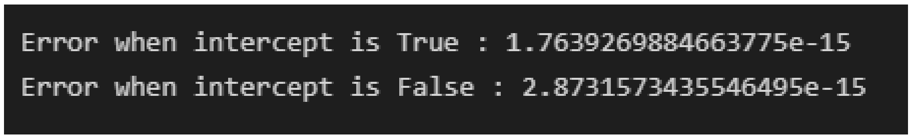
AfterFeature Selection:



```
Root Mean Squared Error for depth 7 : 0.18074134947058837
```

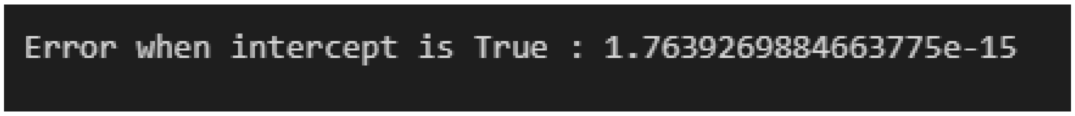
Linear Regression

Before Feature Selection:



```
Error when intercept is True : 1.7639269884663775e-15
Error when intercept is False : 2.8731573435546495e-15
```

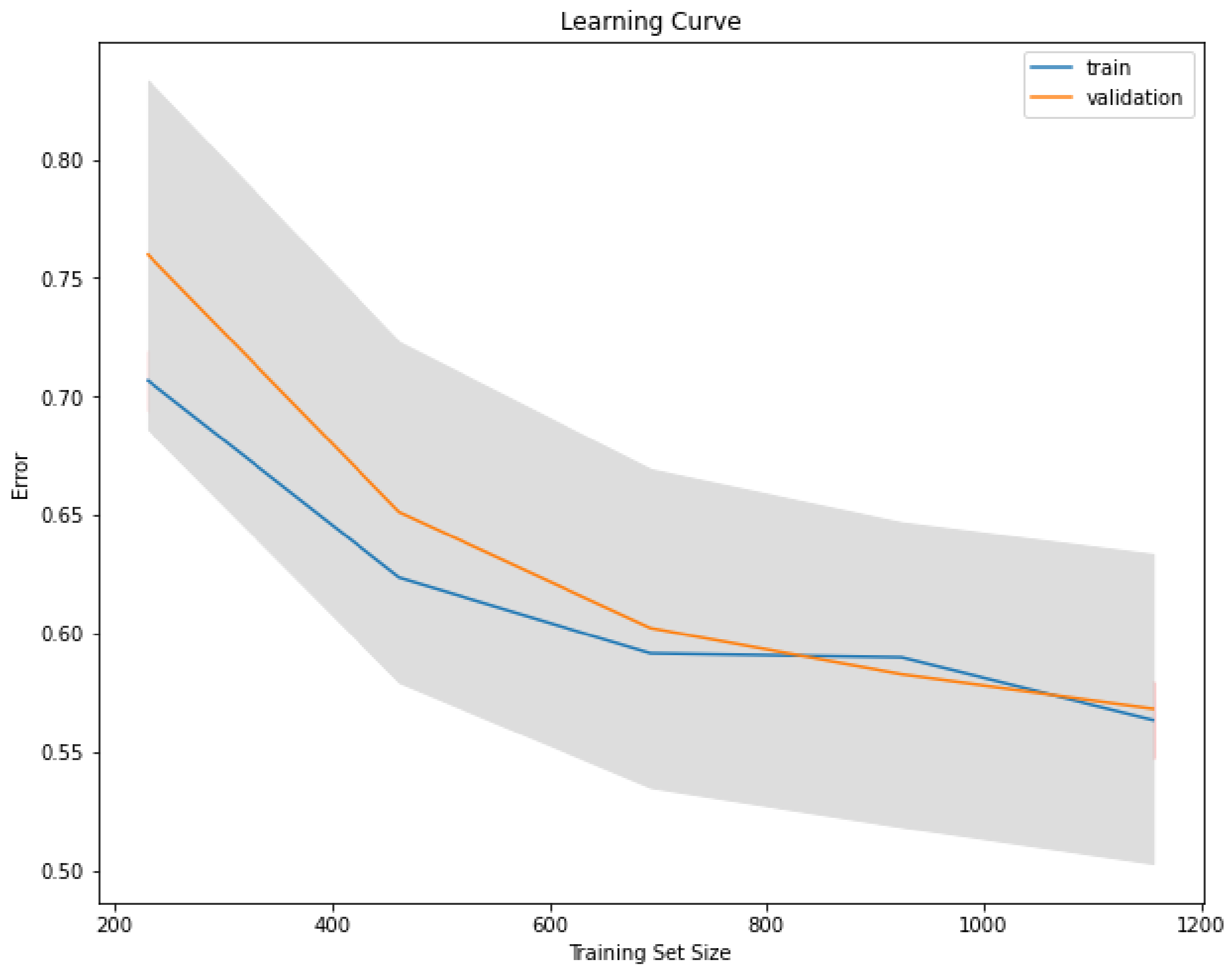
AfterFeature Selection:



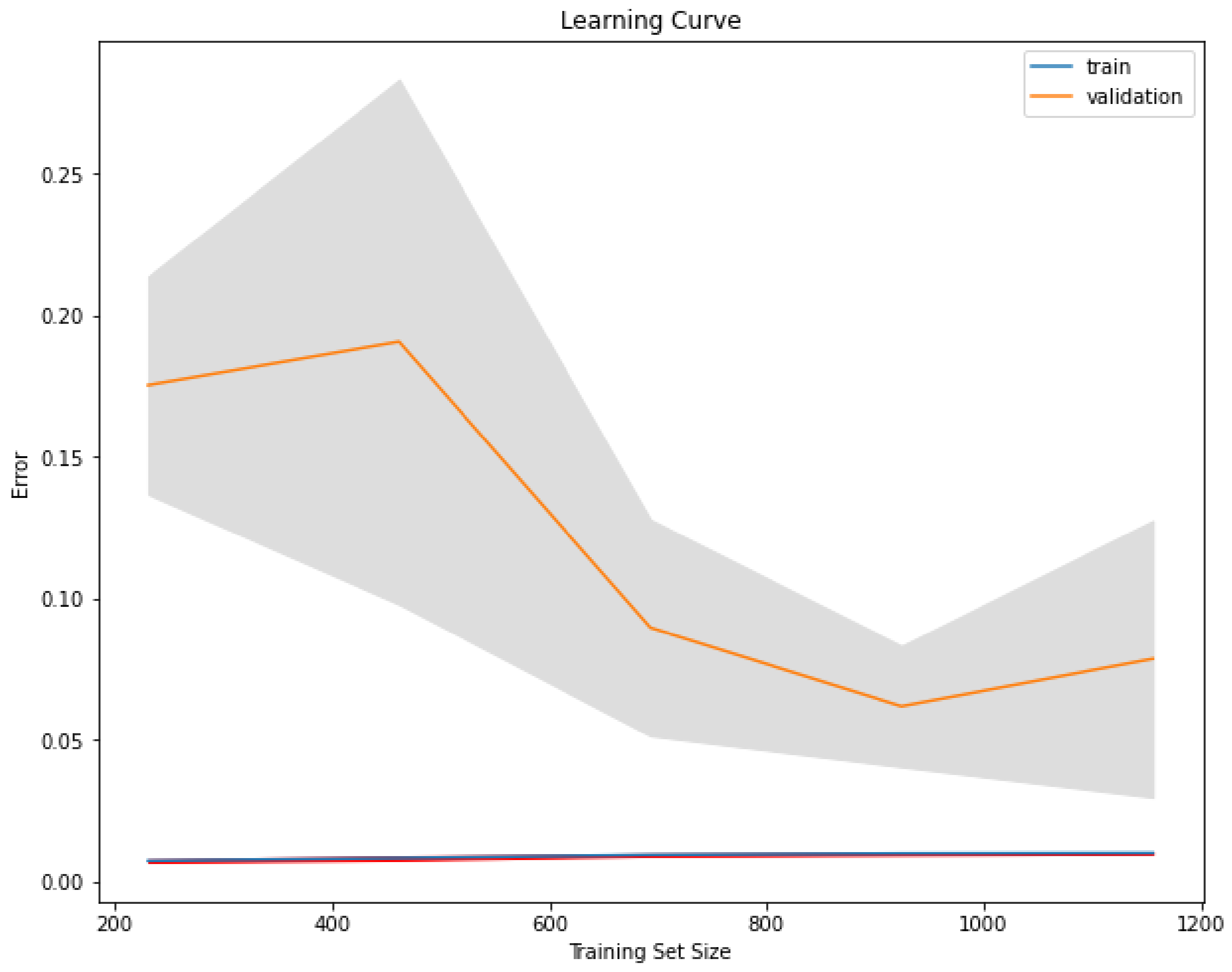
```
Error when intercept is True : 1.7639269884663775e-15
```

Learning Curves

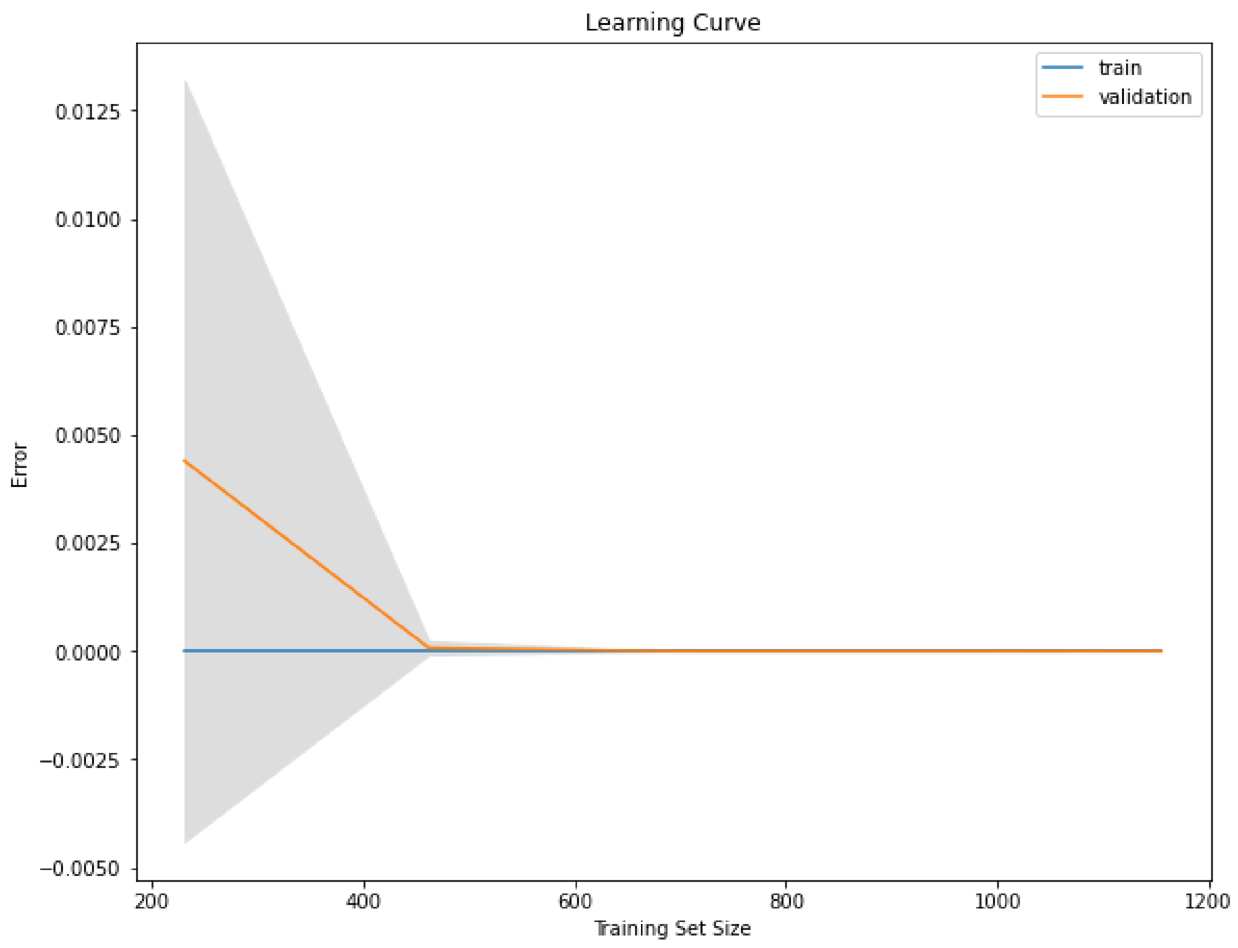
KNN:



Decision Trees:



Linear Regression:



Comparative Analysis

As seen, the Decision trees had the best learning curve because it did not overfit. Meanwhile the KNN and the linear regression faced overfitting which affected the results. The most reliable results are the ones produced by the decision trees especially after the feature selection.