# **Classification of Sentiments of Somerville Residents**

**Contributors**:

1) Sumit Dhaundiyal

2) Suraaj Shrestha

**Data Mining goal**:

Our goal is to predict the sentiment of the residents of Somerville based on the inputs given by the residents during the survey. We will classify if the residents are happy or unhappy based on 28 inputs out of 54 from the survey.

We want to find the most relevant columns with respect to our class attribute so that we can predict the class attribute more efficiently using the reduced dataset which contains only the most important columns. Hence, reducing the runtime of the algorithm.

We are also trying to find the most efficient algorithm (KNN, Decision Tree, Random Forest, Naïve Bayes, Neural Networks) to predict the class attribute for our dataset using cross fold validation and testing various scenarios based on the reduced dataset that we are generating using attribute selection methods (Pearson Correlation, Chi-Squared, Recursive Feature Elimination, L1-based feature selection, Tree-based feature selection).

**Dataset Description-**

A happiness survey is sent out by the City of Somerville every two years to a random sample of Somerville residents. Residents are asked to score their own happiness, well-being, and satisfaction with City services as part of the survey. The survey responses from 2011 to 2019 are included in this consolidated dataset. The Somerville Happiness Survey Responses Dataset contains rating given by residences based on happiness, how satisfied they are with Somerville as a place to live, neighborhood, how proud they are as Somerville residents, availability of information about comity services, coat of housing, public schools, local police, maintenance of streets and sidewalks, availability of social community events, How safe do residents feel walking in neighborhood at night , How satisfied are residents with the beauty or physical setting of residents neighborhood , How satisfied are residents with the appearance of parks and squares in residents neighborhood , gender , Age , race or ethnicity , Do residents have children age 18 or younger who live with residents , housing status in Somerville , Do residents plan to move away from Somerville in the next two years , How long have residents lived here, annual household income , Are residents a student , How anxious did resident feel yesterday , How satisfied are residents with the quality and number of transportation options available to residents , Language , What languages do residents speak at home.

The following columns are categorical: gender, race or ethnicity, housing status, Are residents a student, Language, What languages do residents speak at home. Rest of the columns are numerical data.

**Data Mining Tools Used-**

**Python –**

Python has grown to become one of the central languages in data mining offering both a general programming language and libraries specifically targeted numerical computations. Researchers have noted a number of reasons for using Python in the data science area (data mining, scientific computing)

1. **Simplicity** - Programmer’s regard Python as a clear and simple language with a high readability. Even non-programmers may not find it too difficult. The simplicity exists both in the language itself as well as in the encouragement to write clear and simple code prevalent among Python programmers.
2. **Platform-independent** - Python will run on the three main desktop computing platforms Mac, Linux and Windows, as well as on a number of other platforms.
3. **Various Libraries** - Perhaps very few languages in the field of data science have this number of libraries. With libraries such as TensorFlow, NumPy, Pandas, and many more, Python programming language has implemented various algorithms with very high quality and prepared them for user use. These libraries are provided free of charge to Python programmers and data mining professionals. Using Python libraries reduces the time and cost of data mining and machine learning projects and generally speeds up programming.
4. **Interactive program**- With Python you get an interactive prompt with REPL (read-eval-print loop) like in Matlab and R. The prompt facilitates exploratory programming convenient for many data mining tasks, while you still can develop complete programs in an edit-run-debug cycle. The Pythonderivatives IPython and Jupyter Notebook are particularly suited for interactive programming.

**Description of Classification Algorithms used-**

**1. K Nearest Neighbors**

K-Nearest Neighbors is a [machine learning](https://www.unite.ai/what-is-machine-learning/) technique and algorithm that[can be used for both regression and classification tasks](https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=6&cad=rja&uact=8&ved=2ahUKEwjT8emWsubnAhUHKKwKHbeYCKIQFjAFegQIAxAB&url=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FK-nearest_neighbors_algorithm&usg=AOvVaw2YaNgXyrE3Vga4aiLiYcGm). It classifies the data based on the labels of its neighbors.

[K-Nearest Neighbors](https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/) examines [the labels of a chosen number of data points](https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761) surrounding a target data point, in order to make a prediction about the class that the data point falls into. K-Nearest Neighbors (KNN) is a conceptually simple yet very powerful algorithm, and for those reasons, it’s one of the most popular machine learning algorithms.

**2.** **Decision Tree**

**Decision Tree Analysis is a general, predictive modelling tool with applications spanning several different areas. In general, decision trees are constructed via an algorithmic approach that identifies ways to split a data set based on various conditions. It is one of the most widely used and practical methods for supervised learning. Decision Trees are a non-parametric supervised learning method used for both classification and regression tasks. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.**

**The decision tree classifier creates the classification model by building a decision tree. Each node in the tree specifies a test on an attribute, each branch descending from that node corresponds to one of the possible values for that attribute.**

**3. Random Forest**

**Random Forest is a robust**[**machine learning**](https://deepai.org/machine-learning-glossary-and-terms/machine-learning)**algorithm that can be used for a variety of tasks including regression and classification. It is an ensemble method, meaning that a random forest model is made up of a large number of small**[**decision trees**](https://deepai.org/machine-learning-glossary-and-terms/decision-tree)**, called**[**estimators**](https://deepai.org/machine-learning-glossary-and-terms/estimator)**, which each produce their own predictions. The random forest model combines the predictions of the estimators to produce a more accurate prediction.**

**Standard decision tree**[**classifiers**](https://deepai.org/machine-learning-glossary-and-terms/classifier)**have the disadvantage that they are prone to overfitting to the training set. The random forest's ensemble design allows the random forest to compensate for this and generalize well to unseen data, including data with missing values. Random forests are also good at handling large datasets with high dimensionality and heterogeneous feature types (for example, if one column is categorical and another is numerical).**

**Random forests are very good for classification problems but are slightly less good at regression problems. In contrast to**[**linear regression**](https://deepai.org/machine-learning-glossary-and-terms/linear-regression)**, a random forest regressor is unable to make predictions outside the range of its training data.**

**Random forests are also black boxes: in contrast to some more traditional machine learning algorithms, it is difficult to look inside a random forest classifier and understand the reasoning behind its decisions. In addition, they can be slow to train and run, and produce large file sizes.**

**4. Naïve Bayes**

A Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature.

Naive Bayes is the most straightforward and fast classification algorithm, which is suitable for a large chunk of data. Naive Bayes classifier is successfully used in various applications such as spam filtering, text classification, sentiment analysis, and recommender systems. It uses Bayes theorem of probability for prediction of unknown class.

**5. Neural Network - MLP Classifier :**

The [multilayer perceptron (MLP)](https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html) is a feedforward artificial neural network model that maps input data sets to a set of appropriate outputs. An MLP consists of multiple layers and each layer is fully connected to the following one. The nodes of the layers are neurons with nonlinear activation functions, except for the nodes of the input layer. Between the input and the output layer there may be one or more nonlinear hidden layers.

**Brief description of attribute selection methods being used**

The attribute selection methods used are :

***1. Pearson Correlation :*** We calculated the Pearson's correlation between the Class variable and predictors. We keep the top 10 features which are strongly correlated with the class variables.

***2. Chi-Squared:*** The chi-square metric between the class variable and the predictors is calculated in this procedure, and only the variable with the highest chi-squared values is chosen.

***3. Recursive Feature Elimination:***

RFE is a feature selection method that fits a model and removes the weak feature (or features) until the provided number of features is reached. The model's coef\_ or feature importances\_ attributes rank features, and RFE seeks to minimize dependencies and collinearity in the model by recursively eliminating a small number of features per loop.

We have used DecisionTreeClassifier()as Estimator instance

***4. L1-based feature selection:***

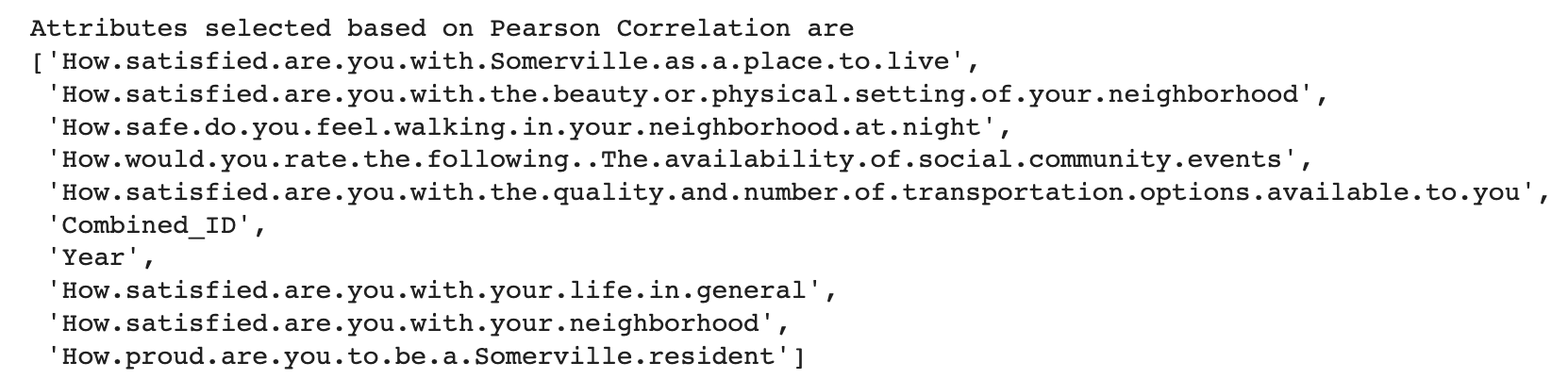
Many of the estimated coefficients in linear models penalized with the L1 norm are zero, resulting in sparse solutions. They can be used in conjunction with SelectFromModel to choose non-zero coefficients when the purpose is to reduce the dimensionality of the data for use with another classifier. The Lasso for regression and LogisticRegression and LinearSVC for classification are examples of sparse estimators effective for this purpose.

***5. Tree-based feature selection:***

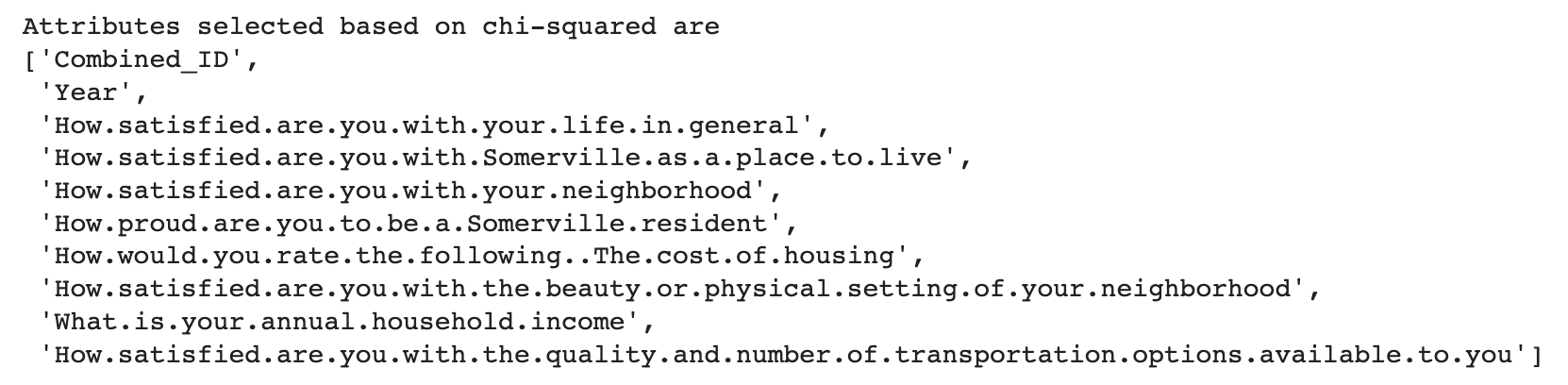
Impurity-based feature importances can be computed using tree-based estimators (see the sklearn.tree module and forest of trees in the sklearn.ensemble module), which can then be used to reject irrelevant features (when coupled with the SelectFromModel meta-transformer). We are using ExtraTreesClassifier model for the selection.

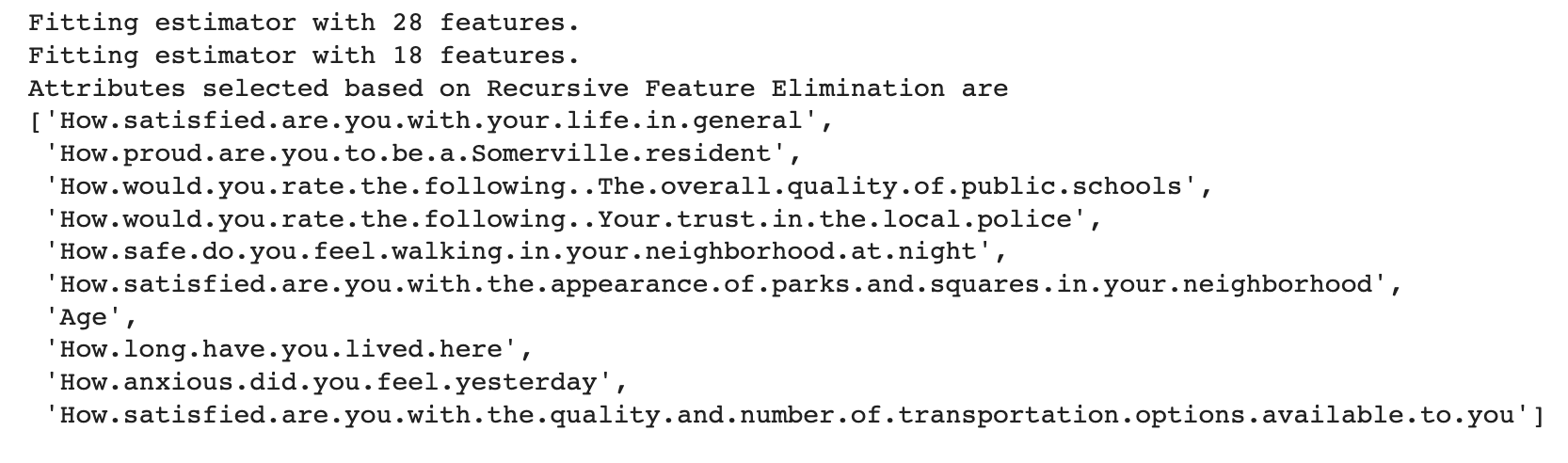
**The set of attributes selected by each attribute selection method-**

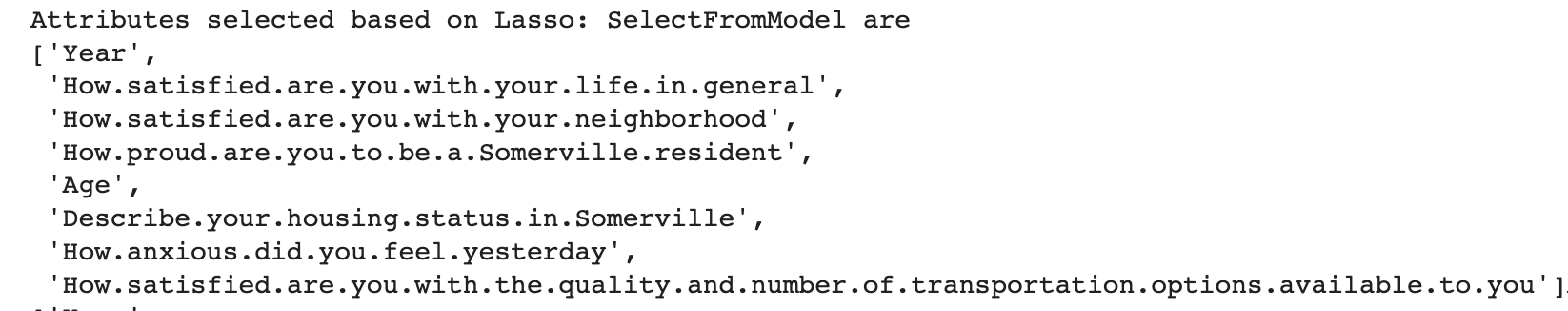
* Attribute selected from Method 1 (***Pearson Correlation***):



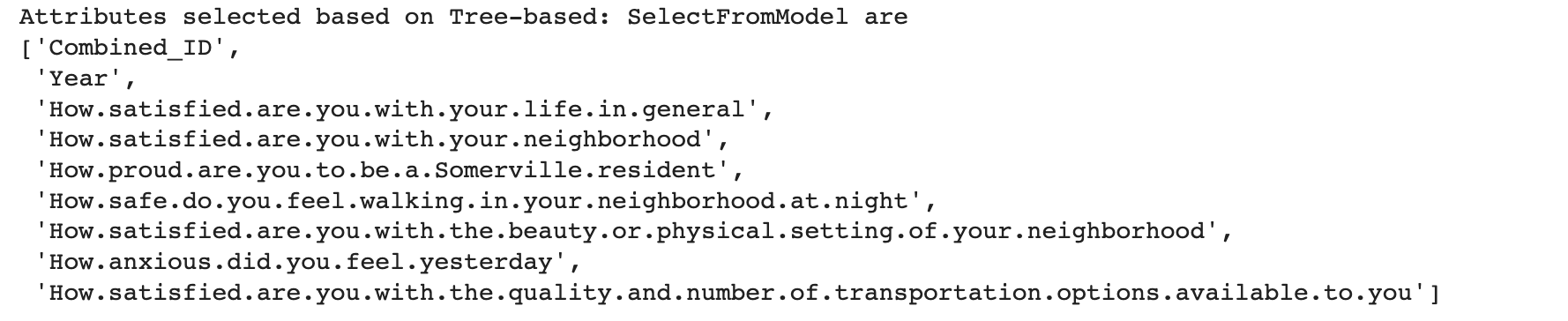
* Attribute selected from Method 2 (***Chi-Squared***):



* Attribute selected from Method 3 (Recursive Feature Elimination):
* Attribute selected from Method 4 ( ***L1-based feature selection***) :



* Attribute selected from Method 5 ( ***Tree-based feature selection***) :

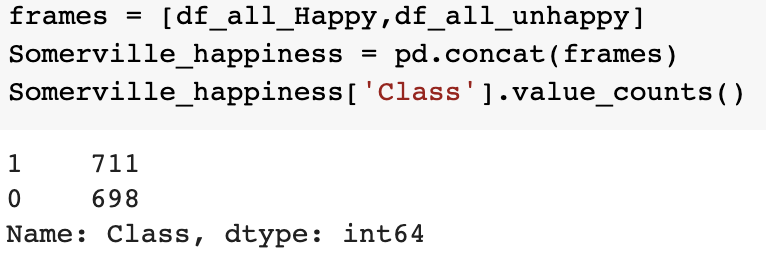


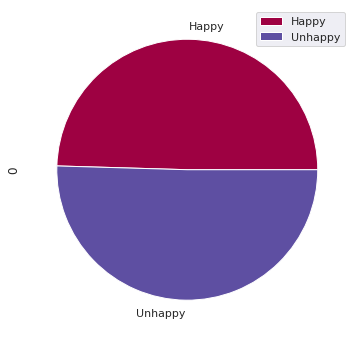
**Detailed description of data mining procedure (the procedure you actually followed) including all data preprocessing you performed-**

**Pre-processing:**

1. There were 8886 rows in total, some of which contained all null values, therefore we filtered data on the basis of following steps:

* For classification purpose we are considering the resident with 5 and above rating in the column 'How.happy.do.you.feel.right.now' as happy resident and unhappy otherwise. Hence, we added a new column named ‘Class’ which contains 0, if rating given is more than 4 and 1 otherwise
* Since we are using Random Forest algorithm (which requires a lot of data to train inorder to get good results), we checked the count of target variable, i.e., number of responses (Happy and Unhappy) since the count of ‘Unhappy’ responses are very low as compared to ‘Happy’ responses. We dropped rows that has at least 28 out of 29 column values empty.
  + The dataset after filtering contains the following class count:



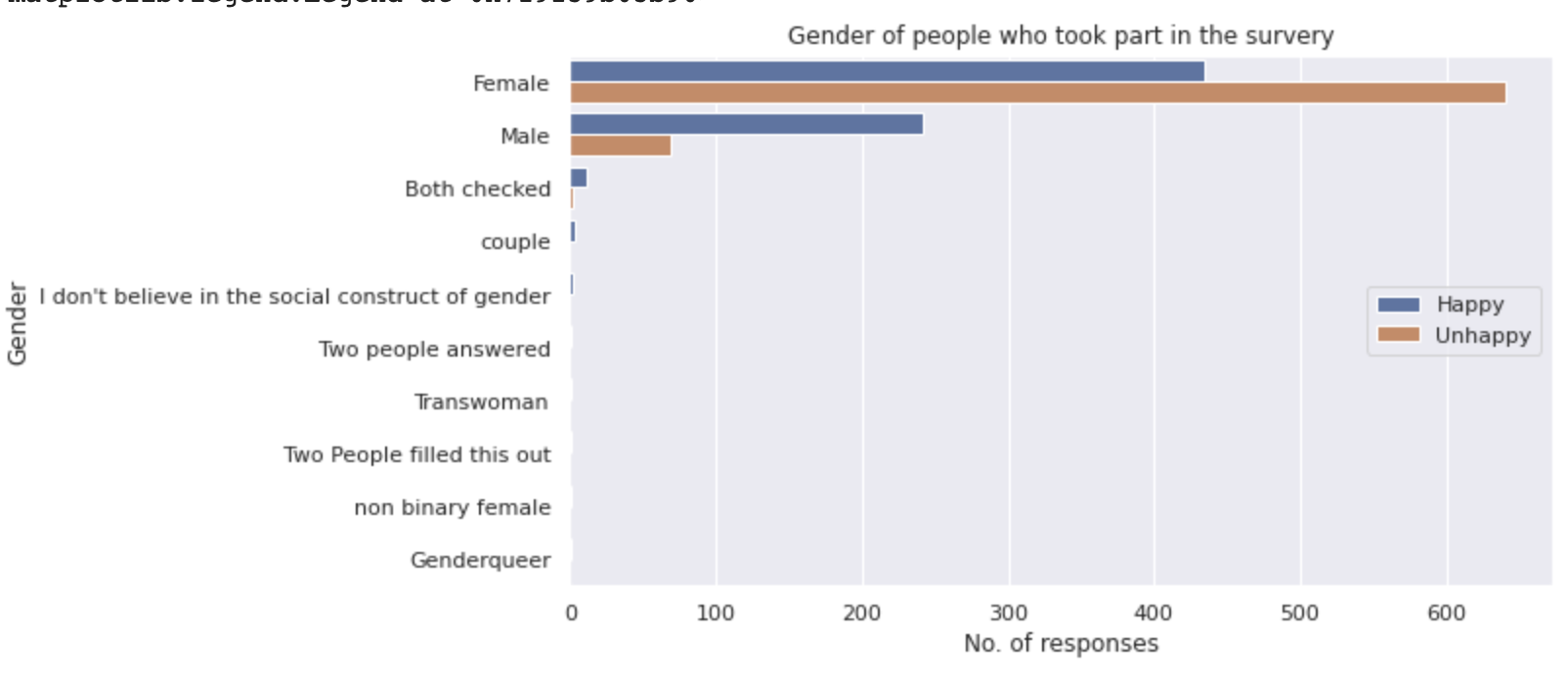


Where 0 represents Happy Class and 1 represents Unhappy class.

After the above filtration the class was almost balanced.

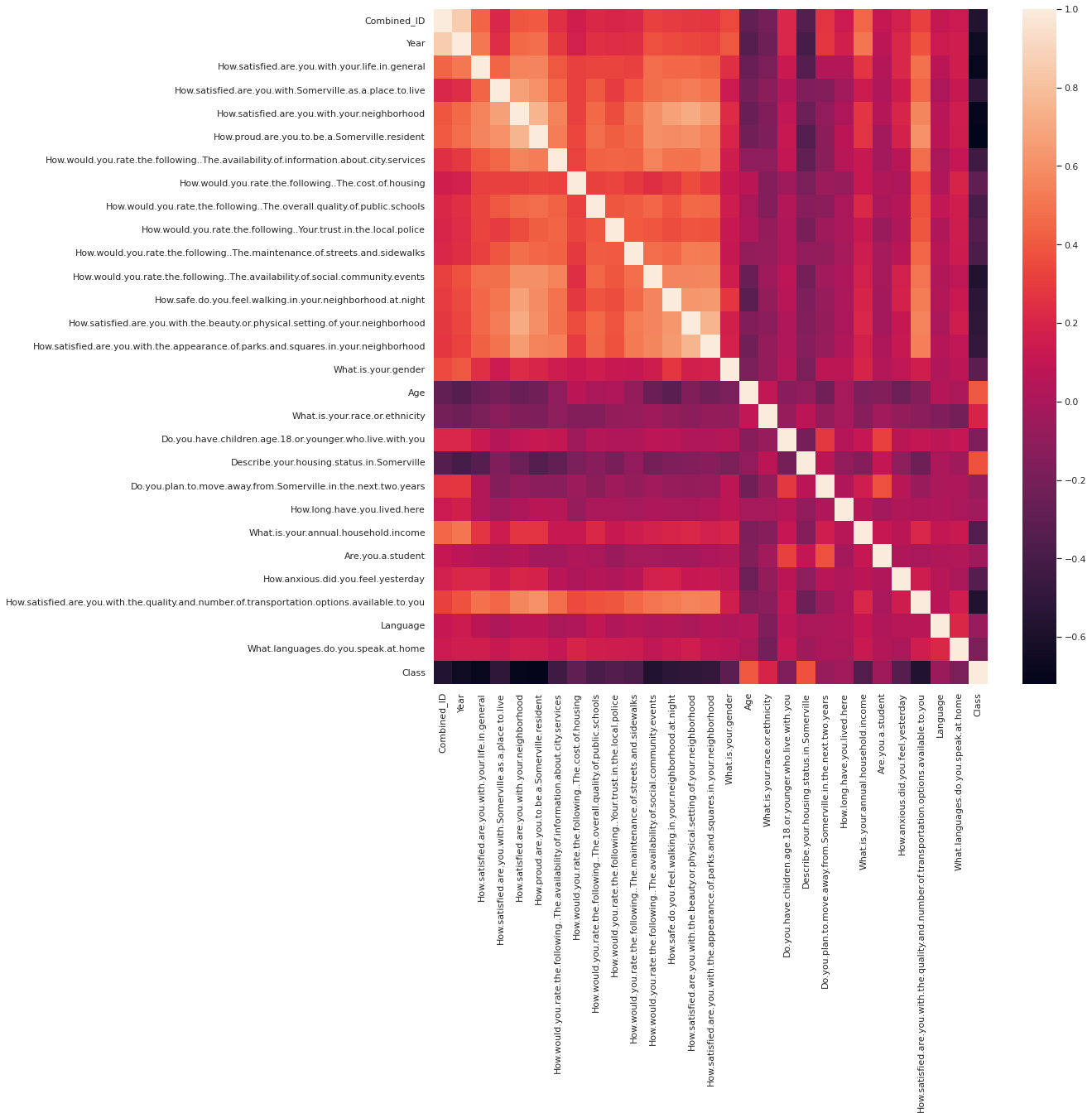
* There are some rows where value is null, we fill the null values class wise, using methods below (we divided the data on the basis of target class value and then filled the empty data):
* Numerical Data : we will replace null values with the mean value of the entire column for that class.
* Categorical Data: We will replace the null values with value with mode value of the entire column for that class.
* For classification purpose we are considering the resident with 5 and above rating in the column 'How.happy.do.you.feel.right.now' as happy resident and unhappy otherwise
* Convert categorical to numeric data
* Column 'How.long.have.you.lived.here' contains words like year/years/ 1 year or more, we remove unwanted characters and converted the categorical column into numerical column
* Row containing gender contains many redundant values, hence we replaced the repeated values with anyone.

The response given by people of various gender are below:

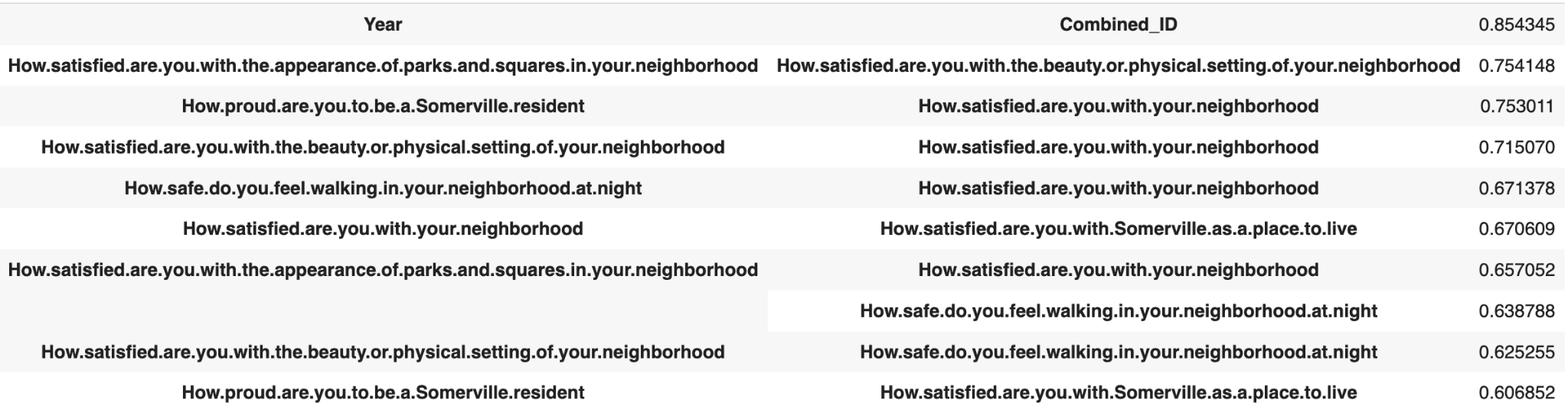


**Data Mining:**

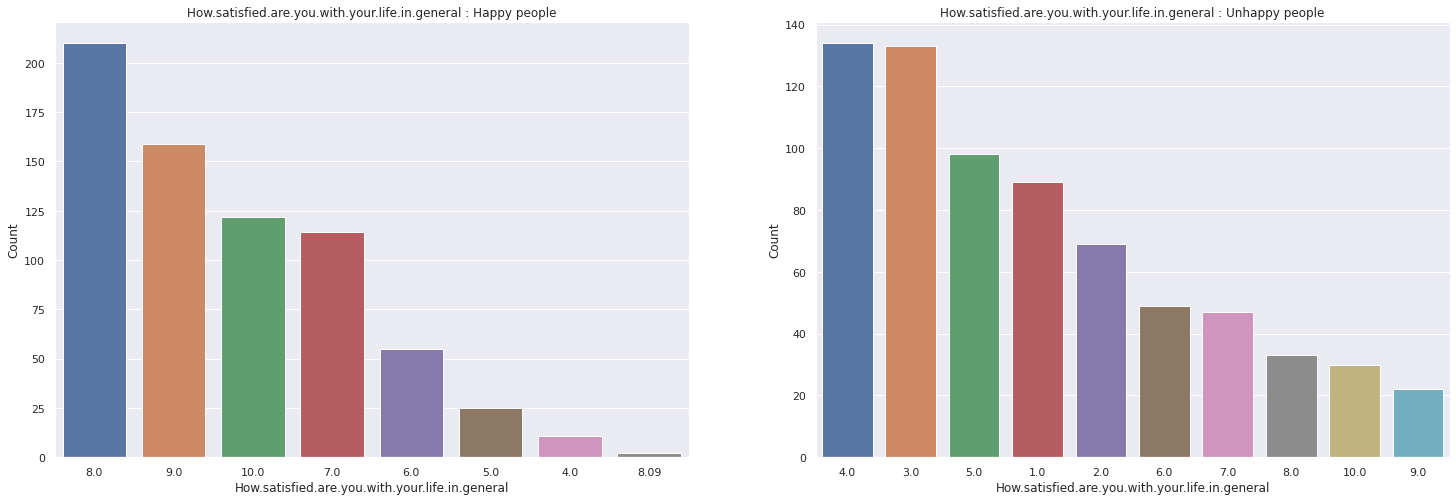
We looked at some of the variables that might give us a better insight of on which basis the people are rating their overall experience hence we looked at the confusion matrix.

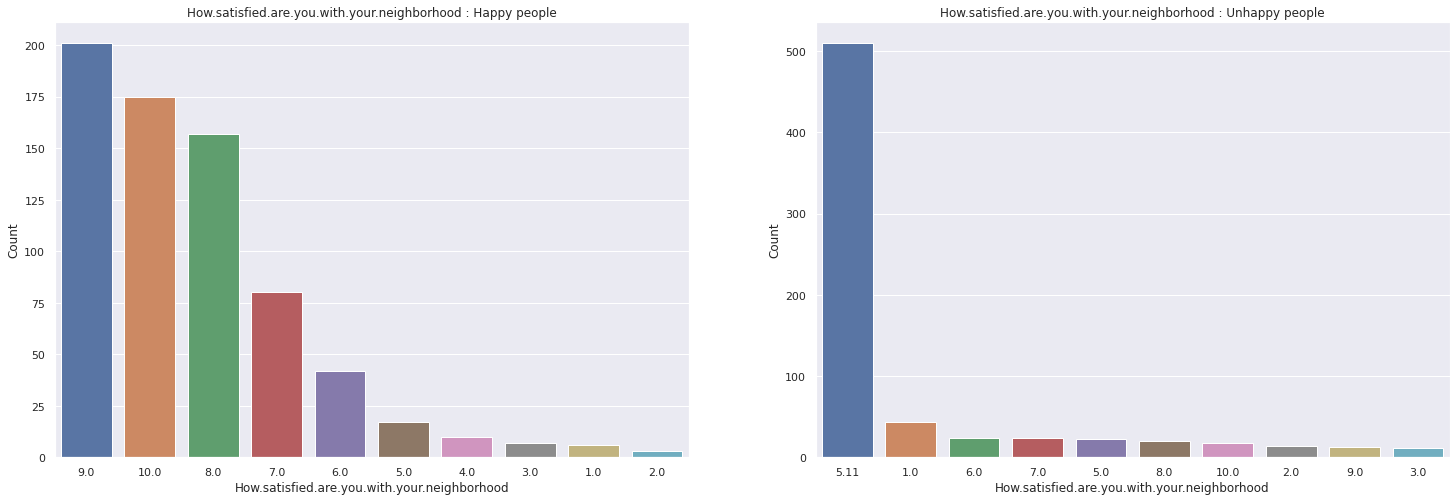


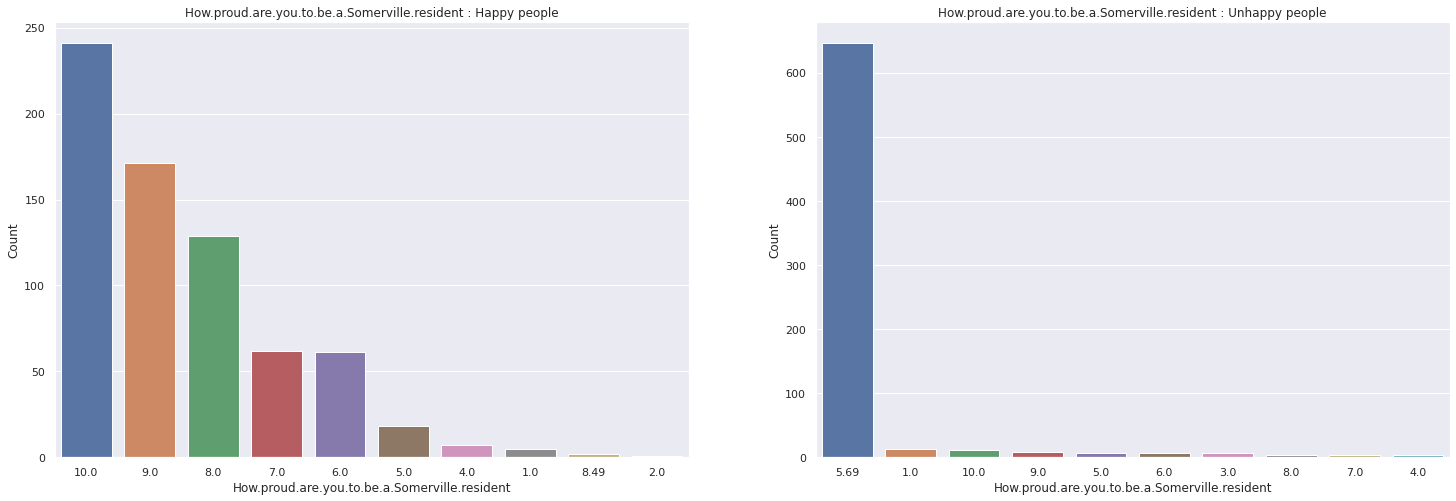
The following columns have a strong correlation coefficient with each other:

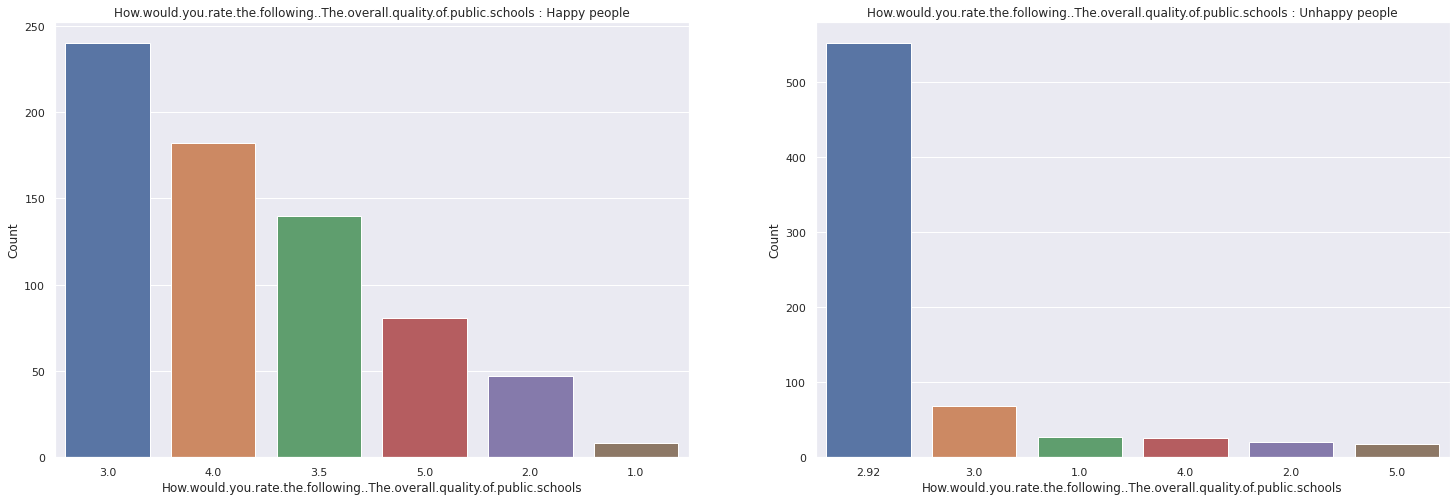


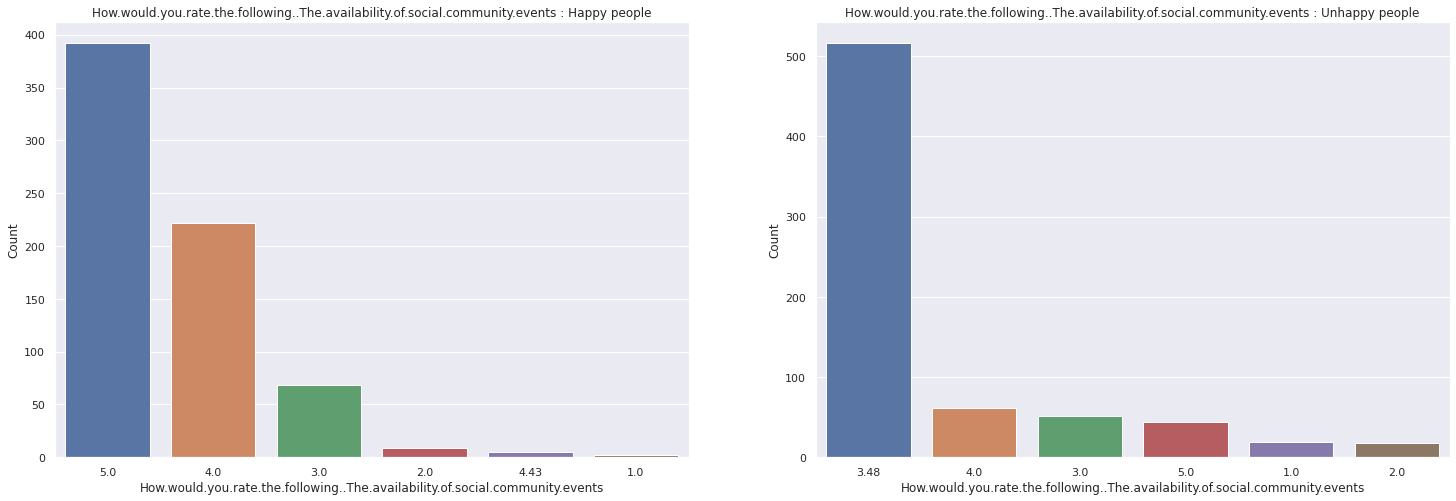
**We have done exploratory data analysis on various columns for better understanding:**

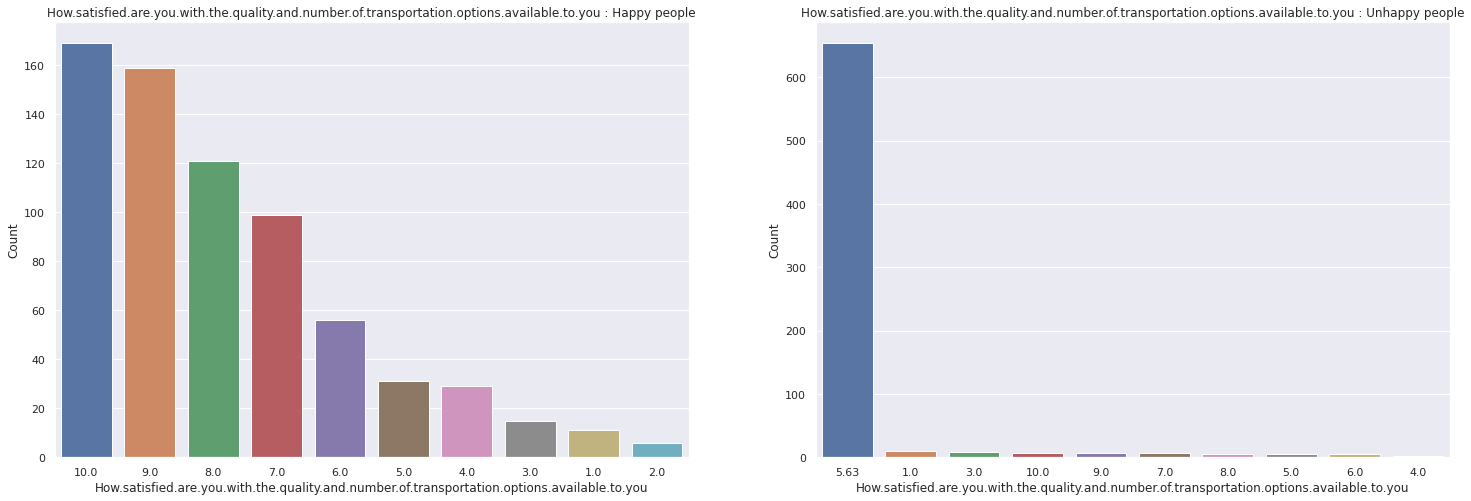


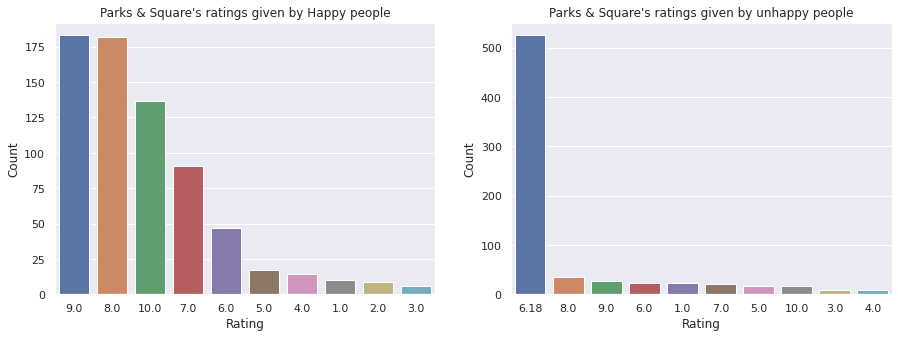


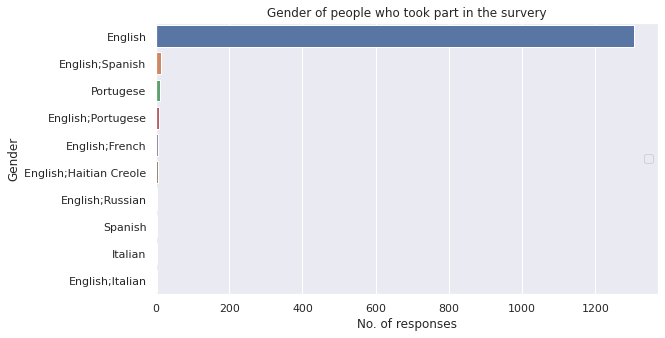




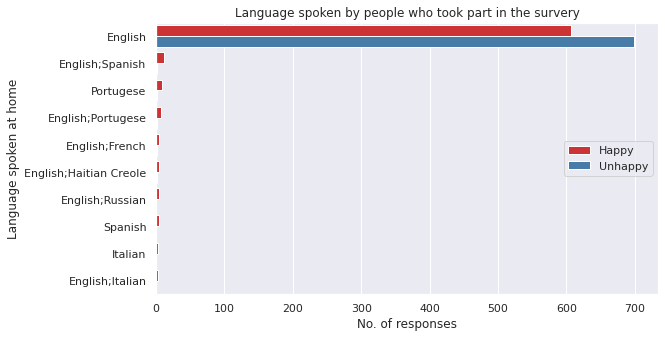








We can observe that the maximum number of people who took part in survey speak English at home and are also the people who gave maximum response as unhappy.



**Data mining result and evaluation:**

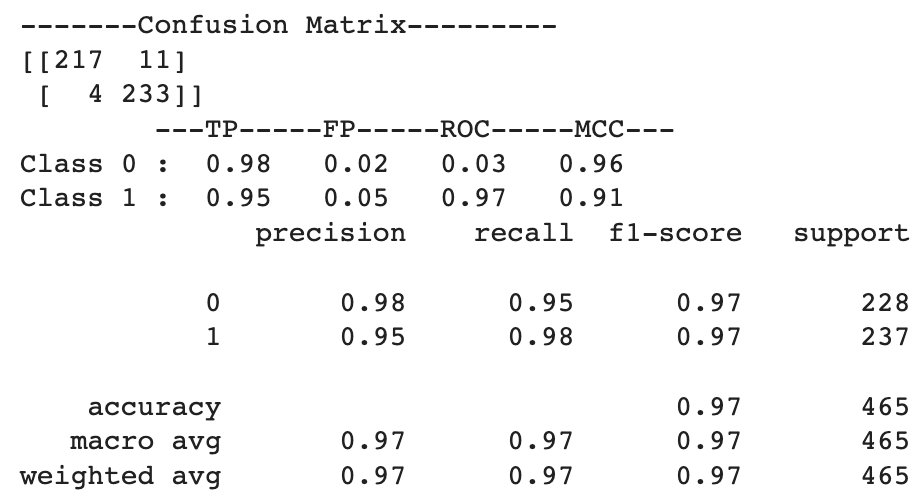
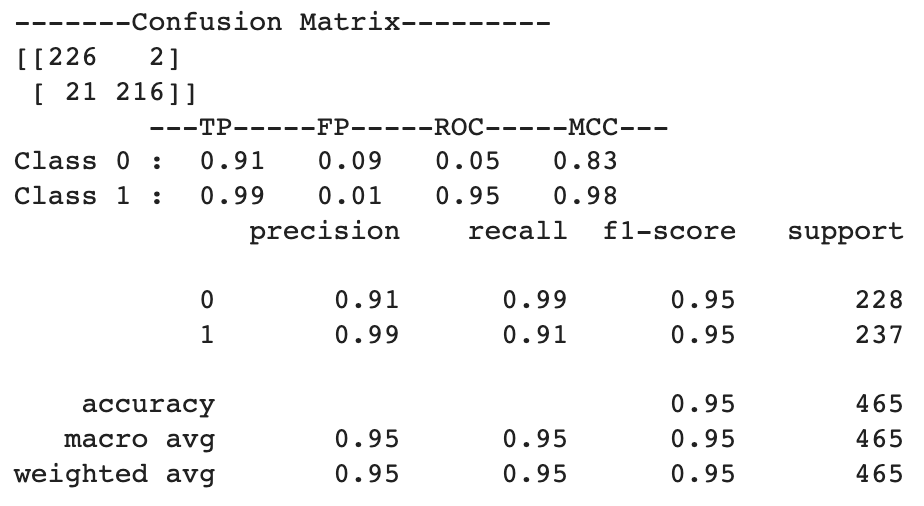
**Abbreviations-**

1. **TP is same as TP Rate in our confusion matrix**
2. **FP is same as FP Rate in our confusion matrix FP Rate**
3. **f1- Score is same as F-Measure in our confusion matrix**

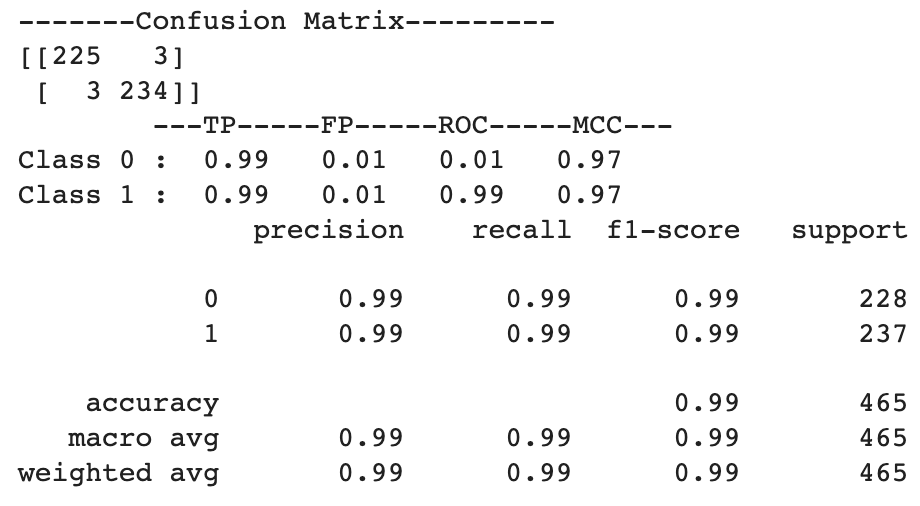
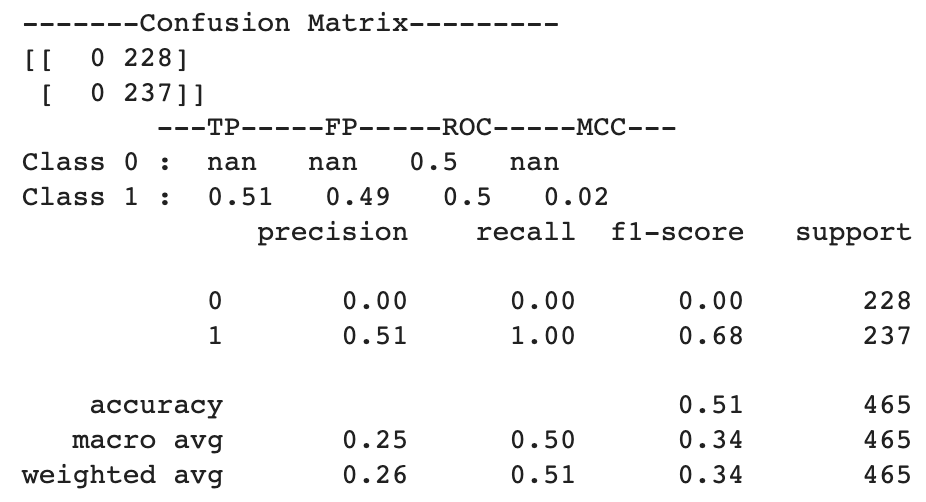
**Performance measures of all 25 models, including all 25 confusion matrices-**

Performance Measure of various algorithms on **Filtered Training data 1**:

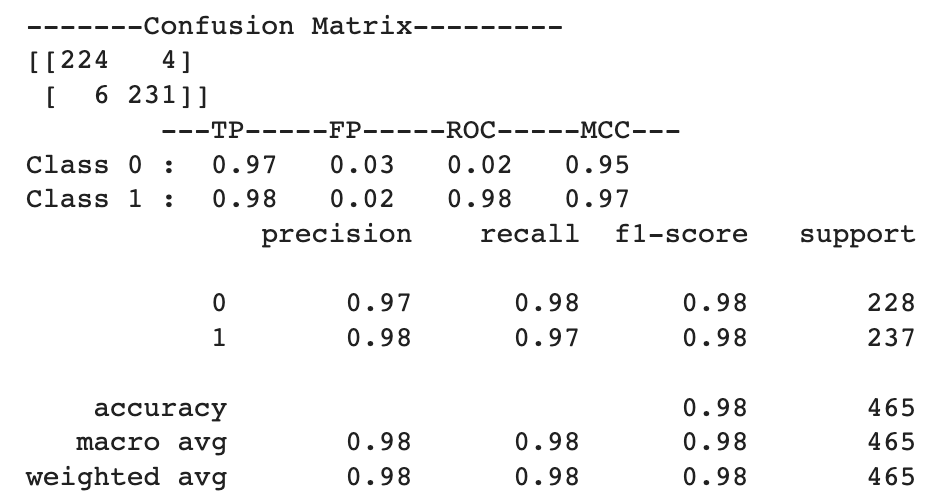
KNN Algorithm Decision Tree



Naive Bayes Random Forest

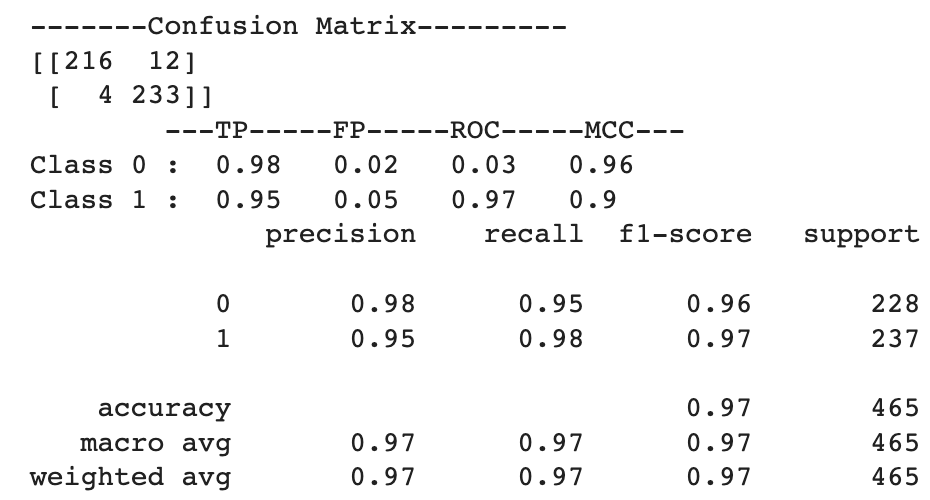
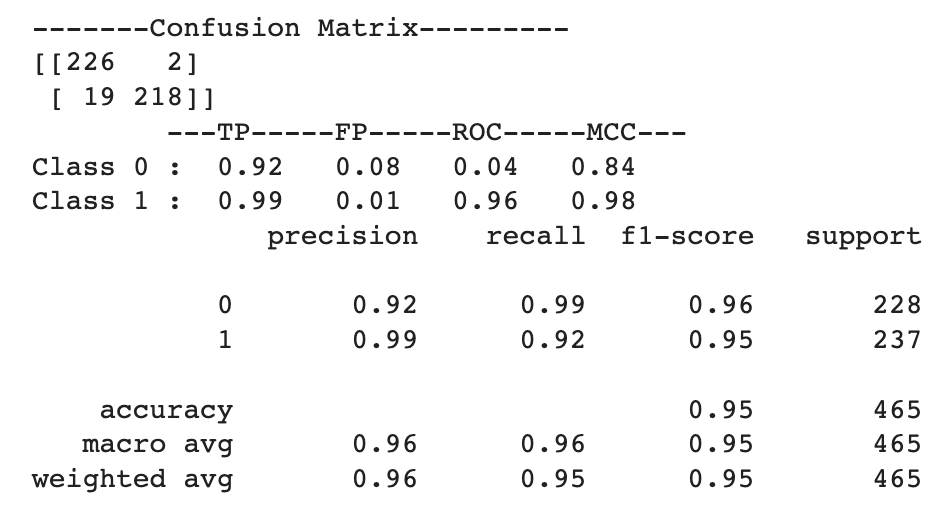


Neural Network

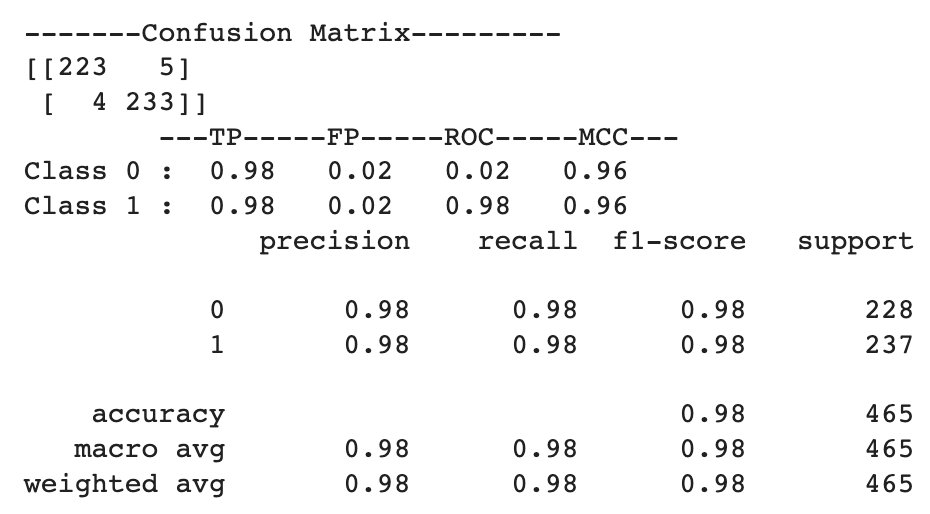
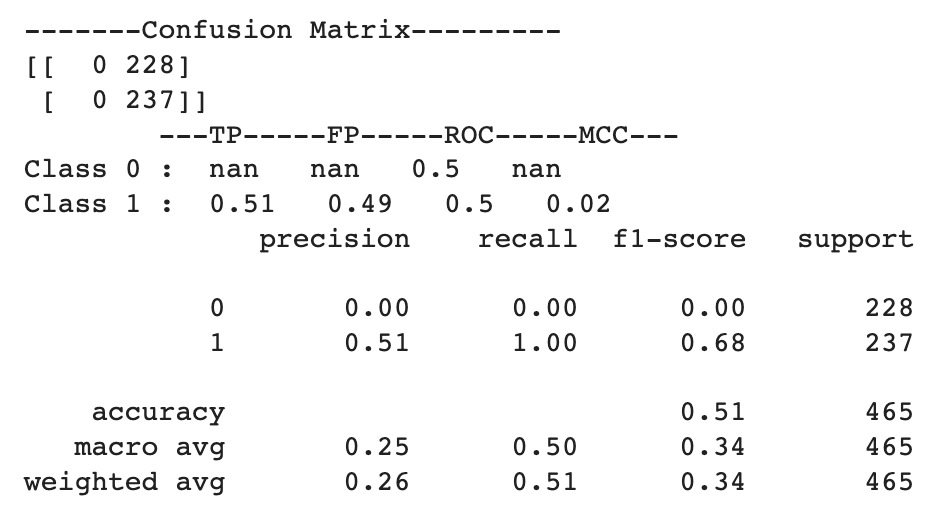


Performance Measure of various algorithms on **Filtered Training data 2**:

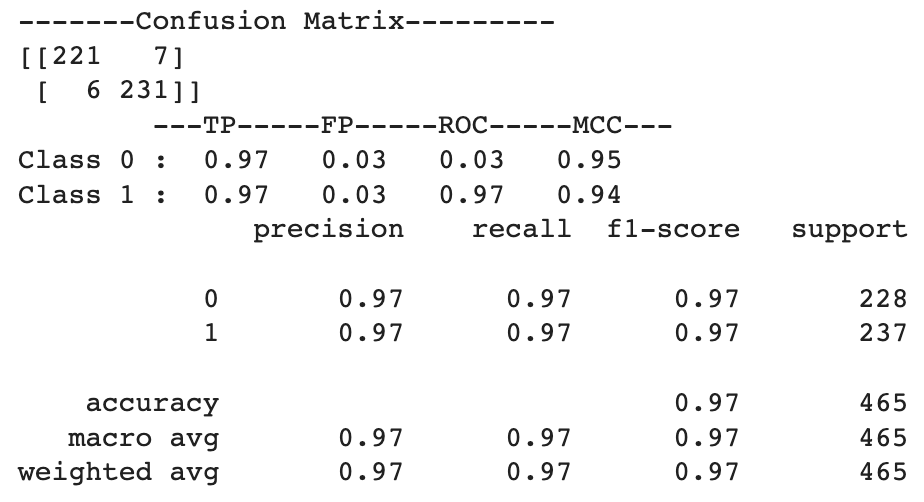
KNN Algorithm Decision Tree



Naive Bayes Random Forest

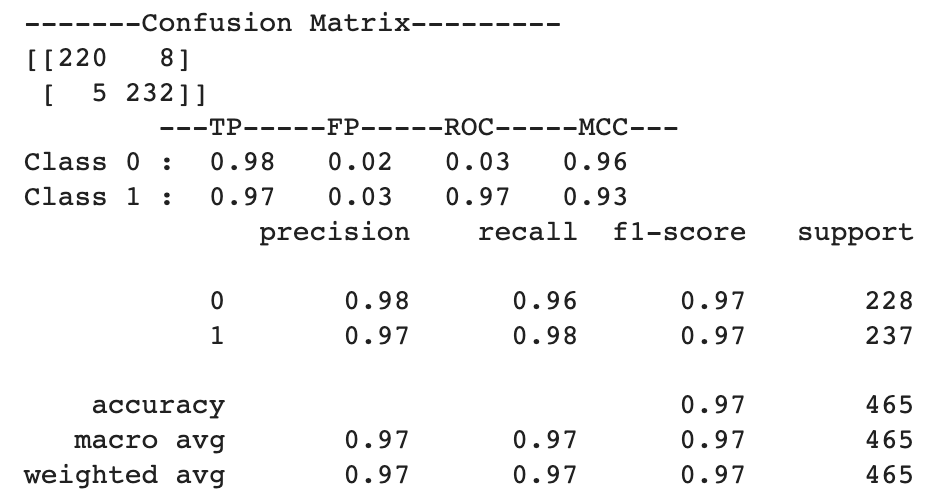
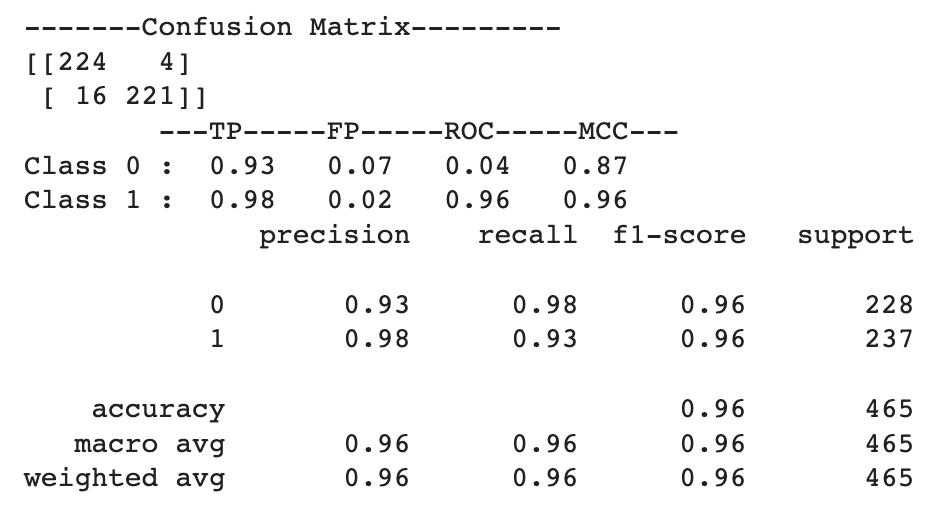


Neural Network

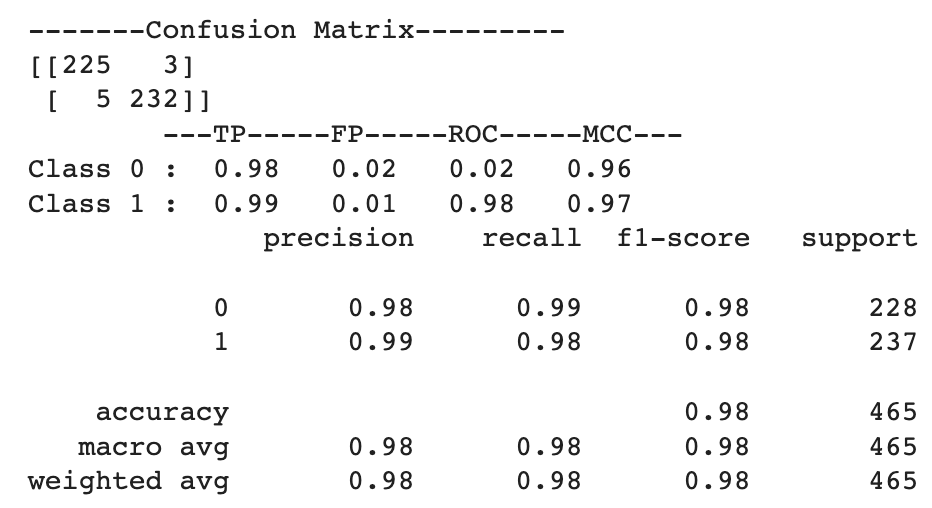
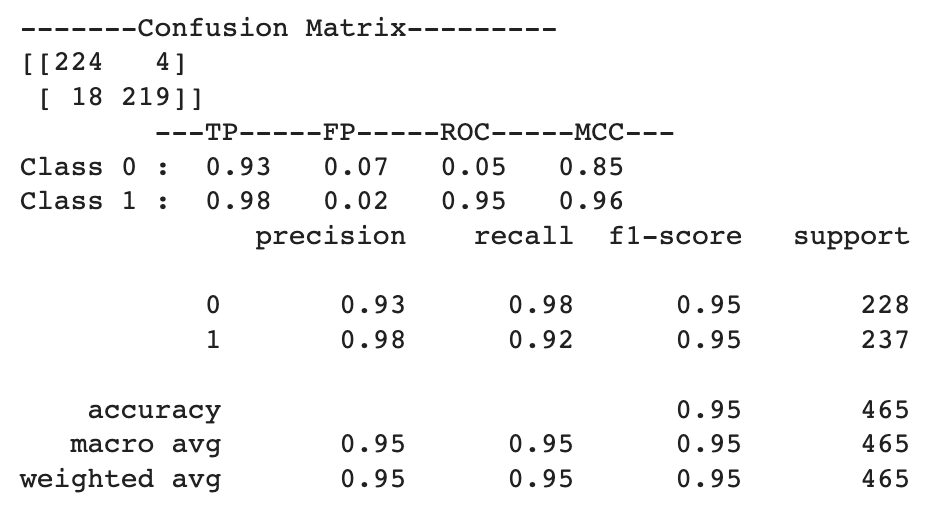


Performance Measure of various algorithms on **Filtered Training data 3**:

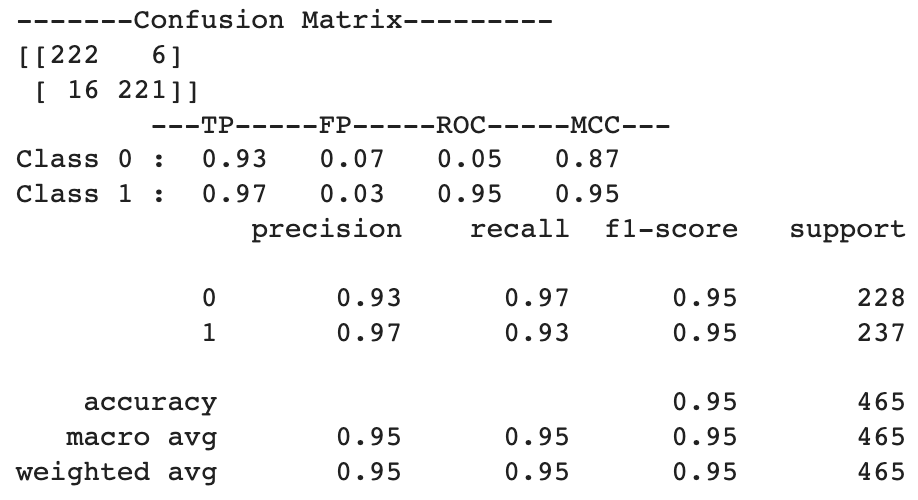
KNN Algorithm Decision Tree



Naive Bayes Random Forest

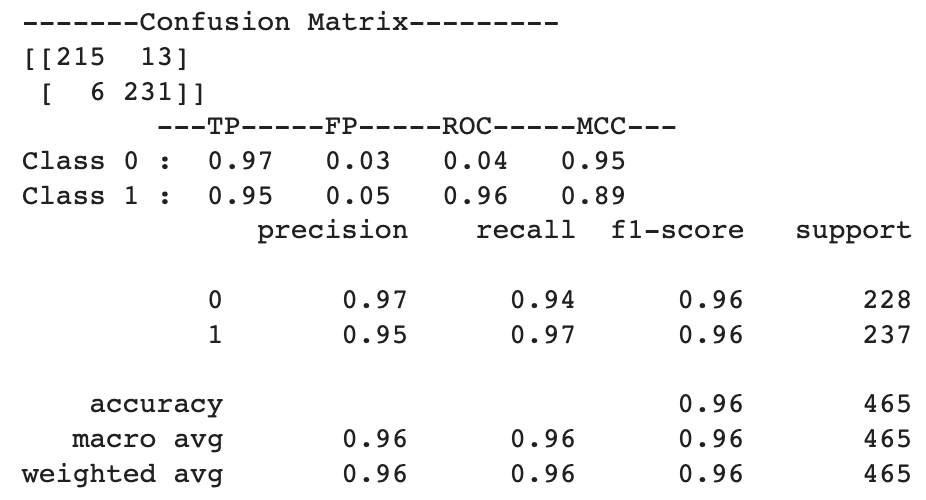
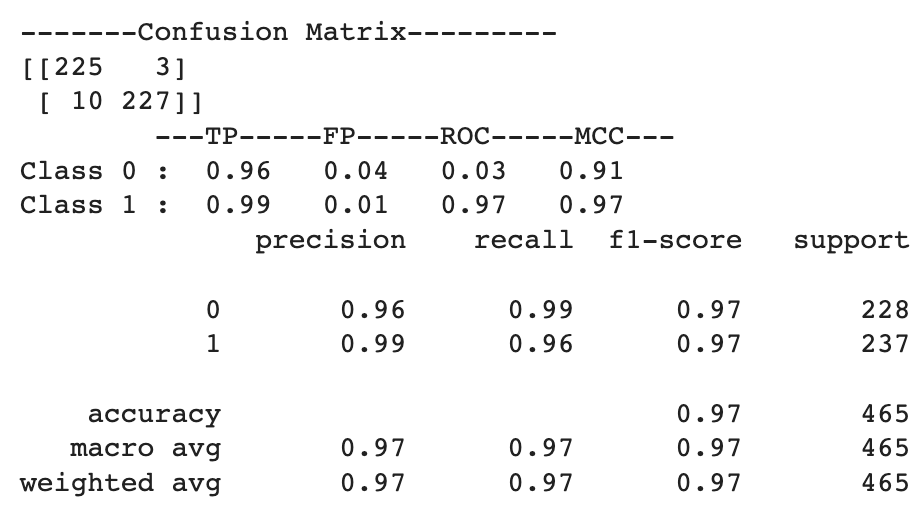


Neural Network

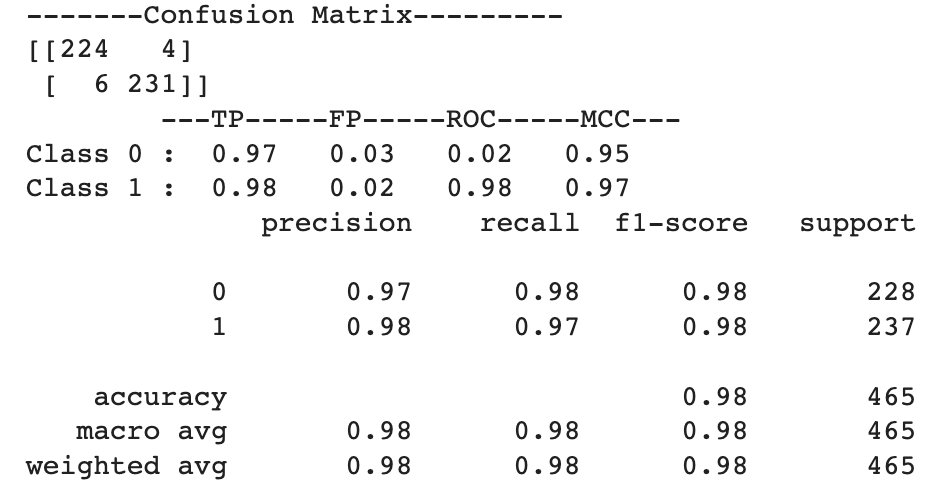
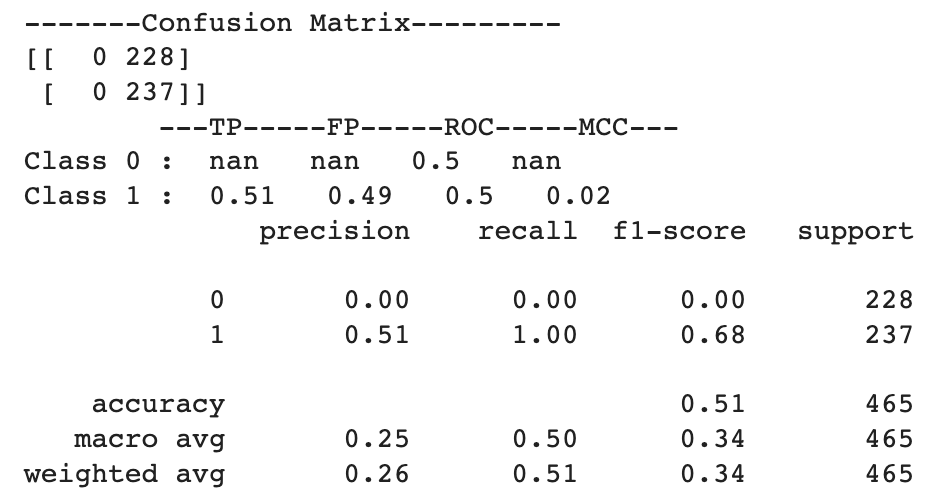


Performance Measure of various algorithms on **Filtered Training data 4**:

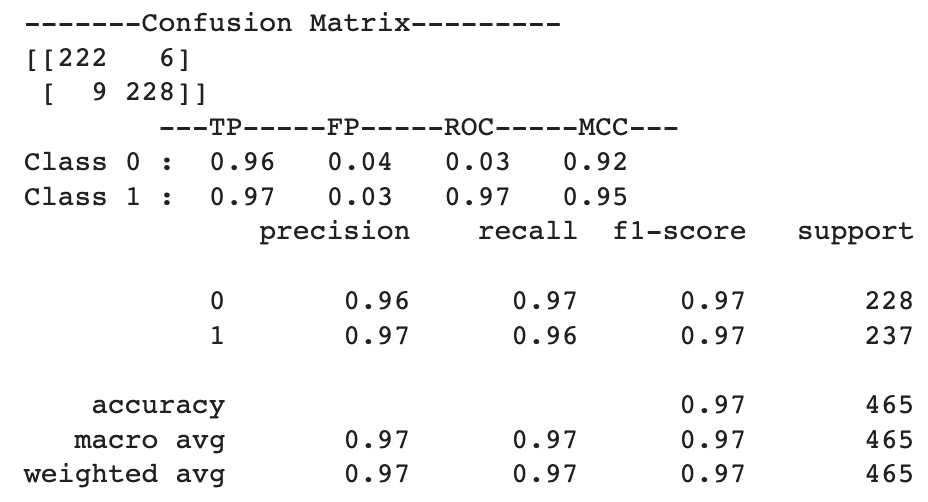
KNN Algorithm Decision Tree



Naive Bayes Random Forest

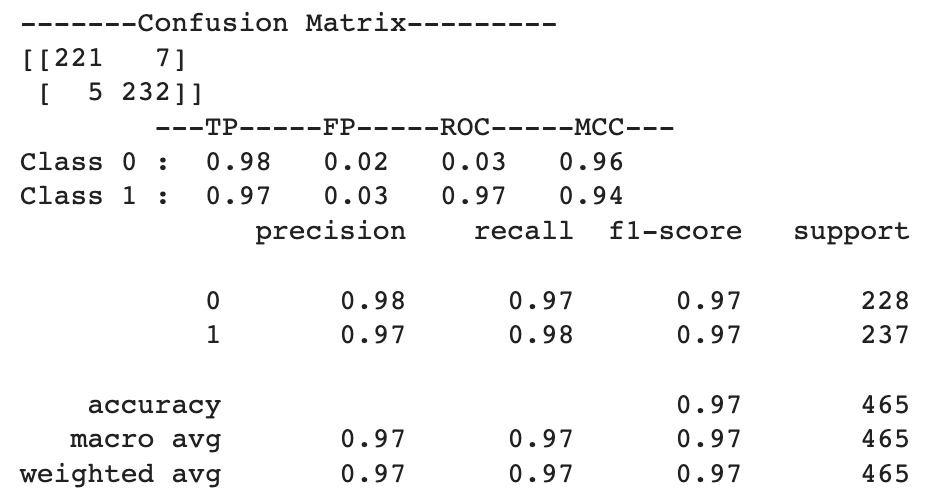
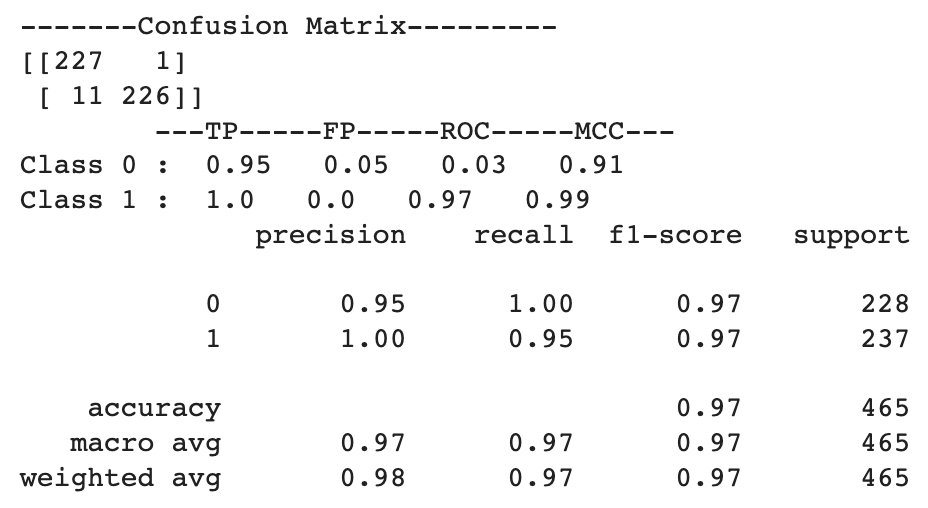


Neural Network

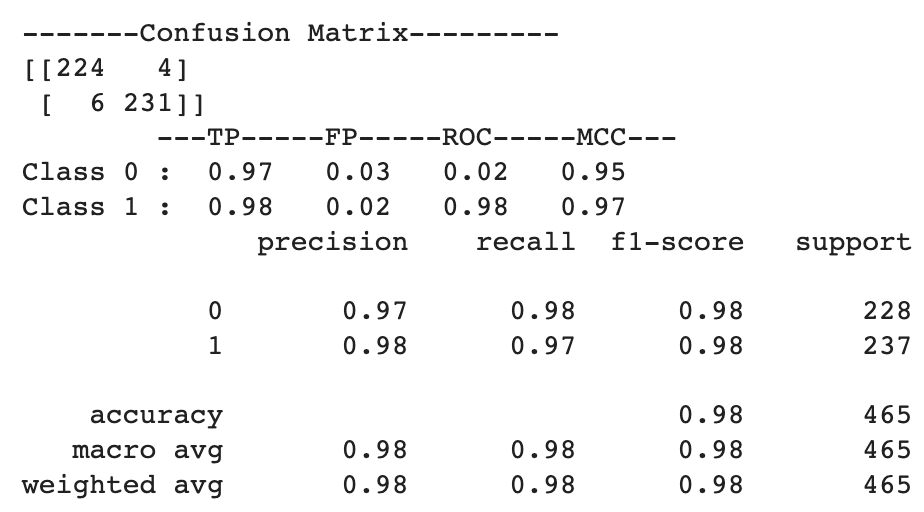
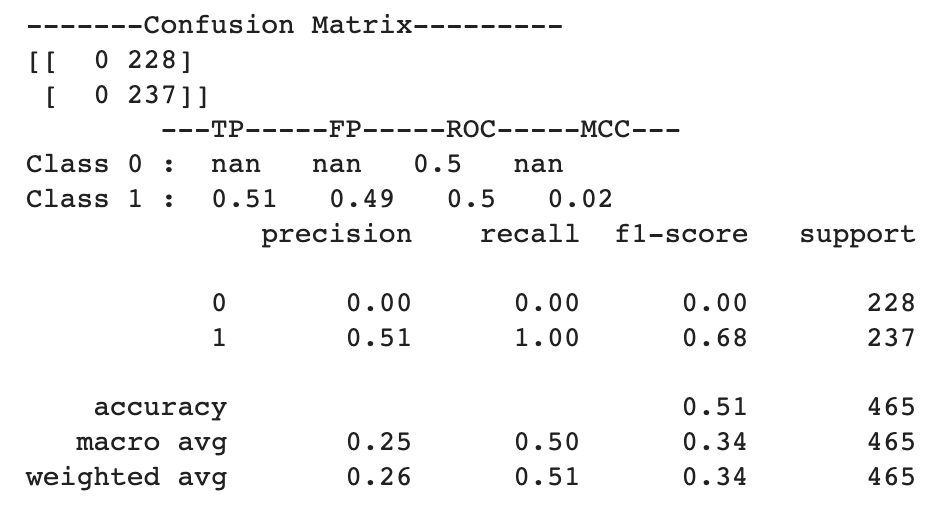


Performance Measure of various algorithms on **Filtered Training data 5**:

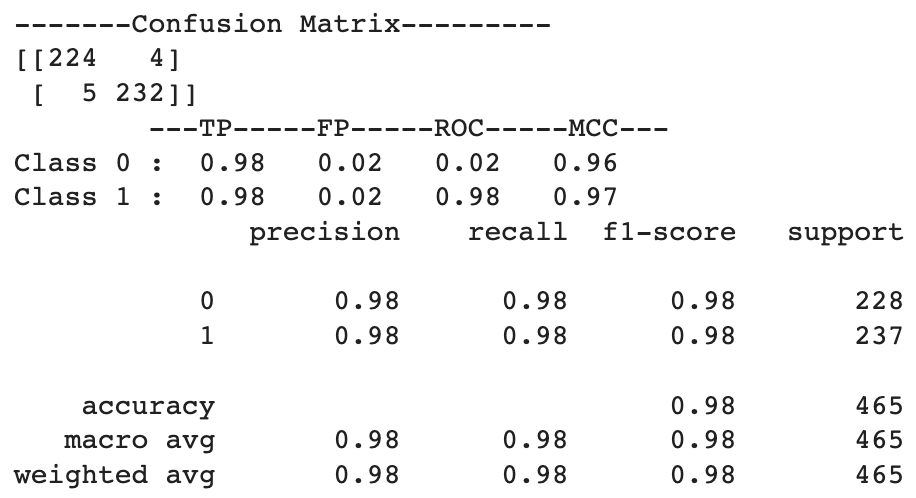
KNN Algorithm Decision Tree



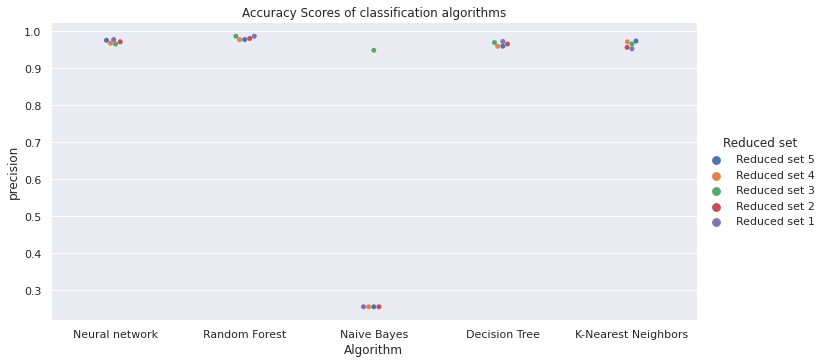
Naive Bayes Random Forest



Neural Network



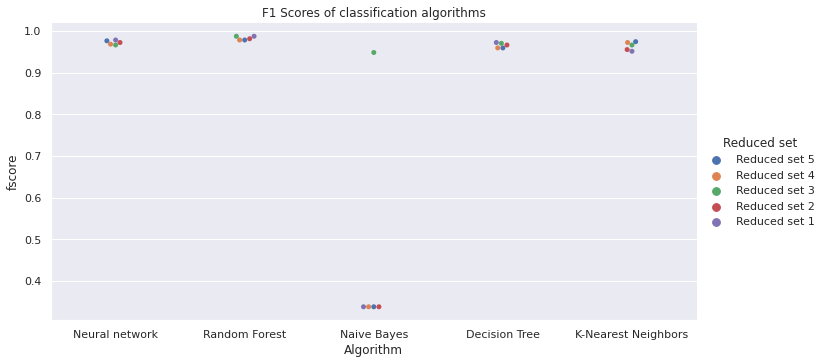
The accuracy of different algorithms on different reduced training set but same test set is given below:



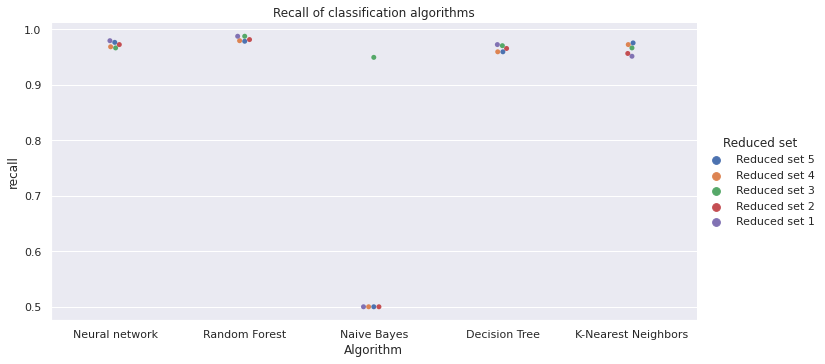
The accuracy score of the models indicates that 4 out of 5 algorithm’s performance is very good as compared to Naive Bayes which only performs good for reduced set 3 and poorly for all the other reduced set.

Looking more closely we can see that even for reduced set 3, Naive Bayes’s recall is higher for one class as compared to other. Hence, we can assume that Naive Bayes is not a good choice of classification algorithm for our dataset.

The F1 scores of different algorithms on different reduced training set but same test set is given below:



Since our classes are slightly imbalanced and performance of our models for different classes varies as well, therefore we will look at the F1 score and Recall as well. The F1 score doesn’t give us much insight as the graph is very similar to precision.



**Justification for the selection of the best model:**

The accuracy and F1 Score indicates that out of the 5 algorithms, Random forest, Decision tree and Neural network performs better. Decision tree’s performance is slightly lower as compared to Random forest and the algorithm is slightly biased towards one of the classes for some models, hence we will only consider Random forest and Neural network as contenders for best model.

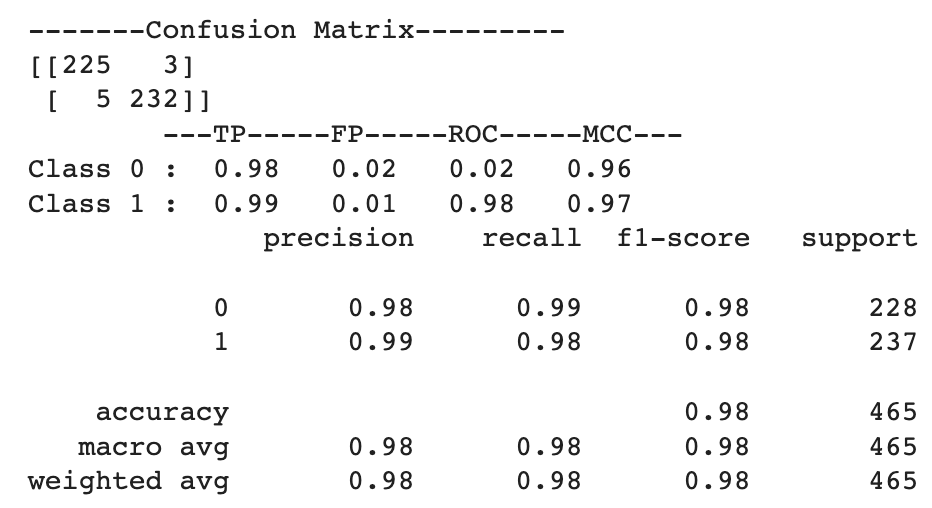
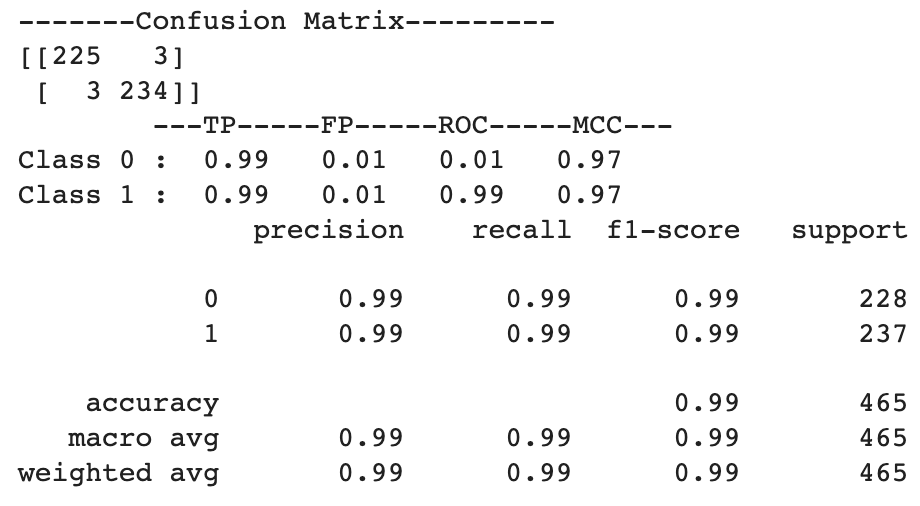
The recall of the Random forest algorithm is higher as compared to other algorithms. Since all the other parameters are almost similar for algorithms (Neural network, Random forest, Decision Tree, K-Nearest Neighbors).

We also, looked at class wise comparison of parameters for Neural network and Random forest :

For Reduced set 3, neural network have higher recall for class 1 as compared to class 0 and for the rest of the models the performance is almost similar. If we see the overall performance of Random forest and Neural network for all the models, we can say that Random forest outperforms Neural network but with a small margin.

For Random forest algorithm we have 5 models, out of which 2 performs best:

model trained on filtered set 1 & model trained on filtered set 3



The first model is more balanced, less biased and performs better hence we can say that Random forest trained on Filtered dataset 1 is the best model for our dataset.

**Discussion and conclusion, including what you learned from this project:**

The attribute selection method helped boost the performance of all the models.

The attribute selection method - Pearson Correlation is only choosing the attributes that are strongly correlated with class attribute hence it is boosting the performance of all the models except Naive Bayes which is unexpected. It is same for the chi square method as well. An attribute selection method - Recursive Feature Elimination is selecting attributes based on DecisionTreeClassifier, hence it was to be expected that it will boost the performance of tree based algorithms. Which it is doing. The attribute selection method - Lasso: SelectFromModel is boosting the performance of all models but it is less for tree-based methods as compared to others. The last selection method is Tree-based: SelectFromModel, we were expecting a boost in random forest and decision tree but it is not the case which is unexpected.

**Team Members individual contribution to the project**

1) Sumit Dhaundiyal (U26474764)

1. Pre-Processing of data set
2. Random Forest Classifier
3. Decision Tree Classifier
4. Neural Network Classifier

2) Suraaj Shrestha (U05652372)

1. Exploratory Data Analysis
2. K Nearest Neighbors Classifier
3. Naïve Bayes Classifier
4. Conclusion

**References:**

<https://towardsdatascience.com/the-5-feature-selection-algorithms-every-data-scientist-need-to-know-3a6b566efd2>

<https://scikit-learn.org/stable/modules/tree.html>

<https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html>

<https://scikit-learn.org/stable/modules/neural_networks_supervised.html>

<https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html>

<https://scikit-learn.org/stable/modules/naive_bayes.html>

<https://scikit-learn.org/stable/modules/feature_selection.html>