Student Number – 10518683
Course Title – MSc in Data Analytics
Lecturer Name – Abhishek Koushik
Module/Subject Title – Machine Learning
Assignment Title – Machine Learning CA2

Question 1 :

Data Pre-processing and its steps:

Data Pre-processing is an essential step in machine learning as the nature of the data and the quality information that can be extracted from it straightforwardly affects the capacity of the designed model. Therefore, it is critical that we pre-process our information before taking care of it into our model.

The steps which are involved in this are –

1) **Importing the libraries** – In this step, we import important libraries required in data pre-processing. We use the word “import” keyword for importing libraries. The following are important libraries –
    - Numpy – It is mostly used when we have to deal with complex mathematical calculations in machine learning like linear algebra, etc
    - Matplotlib – This library is used for plotting graphs like bar graph, pie chart, etc.
    - Pandas – This library is used for data manipulations.
    - Seaborn – This is the library which is used for data visualization.
2) **Importing the datasets** – Before we start with the pre-processing, we must have the dataset ready. Pandas are usually used to import the dataset. It would be easy to import CSV dataset because it is low in size and fast to process.
3) **Filling up the missing values in the dataset** – If we have huge datasets containing huge information, then we can delete the row which has the missing values. It will have negligible effect on getting the accuracy. Suppose if we have numerical dataset which has null values in it, then we can replace them with the calculated mean, median or mode.

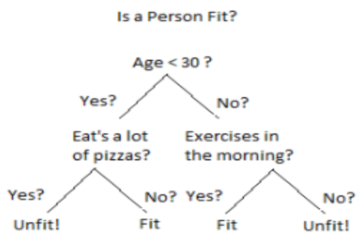
    Example - data['Column].fillna(method ='ffill', inplace = True)

4) **Modification of categorical or text values to numerical values** – As we know that in machine learning, all the values have to be numerical. Hence the LabelEncoder() class is used to transform string into numerical values.

    Example - data[Column] = converter.fit_transform(data[Column].astype(str))

Decision Tree - A decision tree is a flowchart-like structure in which each interior hub speaks to a test on an element, each leaf hub speaks to a class name and branches speak to conjunctions of highlights that lead to those class names. The ways from root to leaf speak to grouping rules.

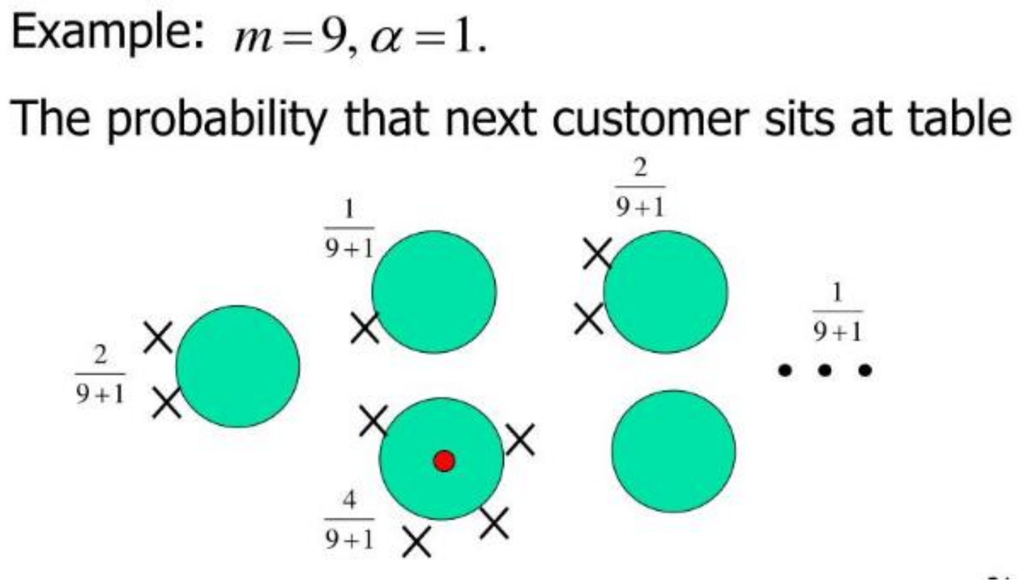
Example :



Information Gain – Information gain is a metric used to train decision trees. The information gain depends on the decline in entropy after an informational collection is part of a characteristic. Building a decision tree is tied in with discovering characteristics that profit the most noteworthy information gain. The dataset is first divided on the various characteristics. The entropy for each branch is determined. At that point it is included relatively, to get all out entropy for the split. The subsequent entropy is subtracted from the entropy before the split. The outcome is the Information Gain or decrease in entropy.

Entropy - A decision tree is constructed top-down from a root hub and includes portioning the information into subsets that contain occasions with comparative values. It actually effects how a decision tree draws its boundaries.

Chinese Restaurant Process - This work proposes a relaxing up work as a component of the earlier and updates the parameters with the probability function in terms of the consistency between the genuine mark data and anticipated outcome. This work presents two Gibbs testing calculations to perform posterior inference. Preparing huge volumes of gushing information in close constant is getting progressively significant as the Internet, sensor systems and system traffic develop. Online AI is a regular method for managing streaming information since it permits the characterization model to learn each occurrence of information in turn. Albeit numerous web-based learning strategies have been created since the improvement of the Perceptron algorithm, existing techniques expect that the quantity of classes is accessible ahead of the classification process. Be that as it may, this supposition is ridiculous for huge scope or streaming data sets. Hence, CRP is there to tackle this problem.



Question 2

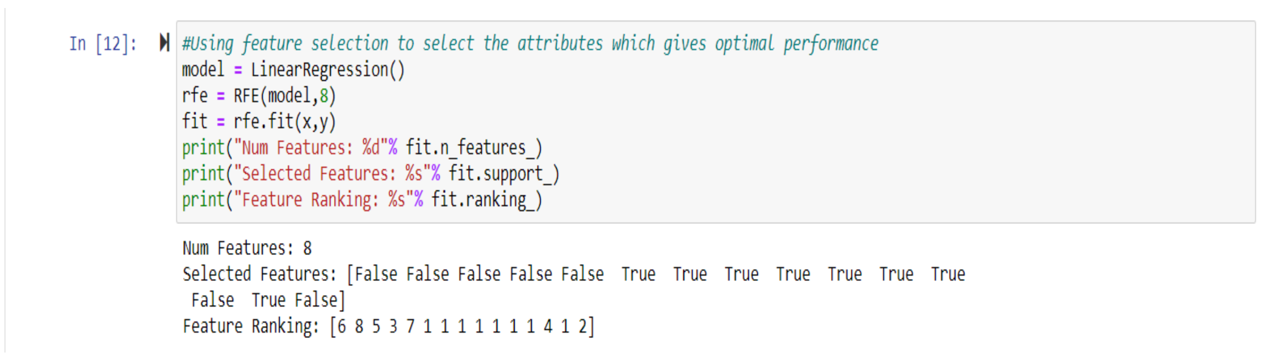
REGRESSION REPORT

For the Regression model, a dataset is chosen which is related to the mortality rate caused by drug poisoning. For the categorical values, the variable “Deaths” is chosen as it gives the information on the number of deaths occurred due to the drug poisoning.

Firstly, the pandas is imported to read the data from the local disk. OneHotEncoder is imported which encodes categorical features as one-hot numeric array. LabelEncoder is used which encodes target names with an incentive among 0 and n_classes-1. The contribution to this transformer ought to be a cluster like numbers or strings, indicating the qualities taken on by categorical features. A RandomForestRegressor is used which acts as a meta estimator that fits various decision tree classifiers on different sub-tests of the dataset and utilization, averaging to improve the precision and control over-fitting. The train_test_split is imported which splits arrays into irregular train and test subsets. The confusion_matrix is used to assess the exactness of a grouping. The accuracy_score is used to capacity subset exactness: the arrangement of labels anticipated for a sample should precisely coordinate the corresponding set of labels in y_true. ColumnTransformer is used which applies transformers to segments of an array or pandas DataFrame. SimpleImputer is imported to complete the missing values. The preprocessing is used which includes scaling, centering, normalization and binarization methods.

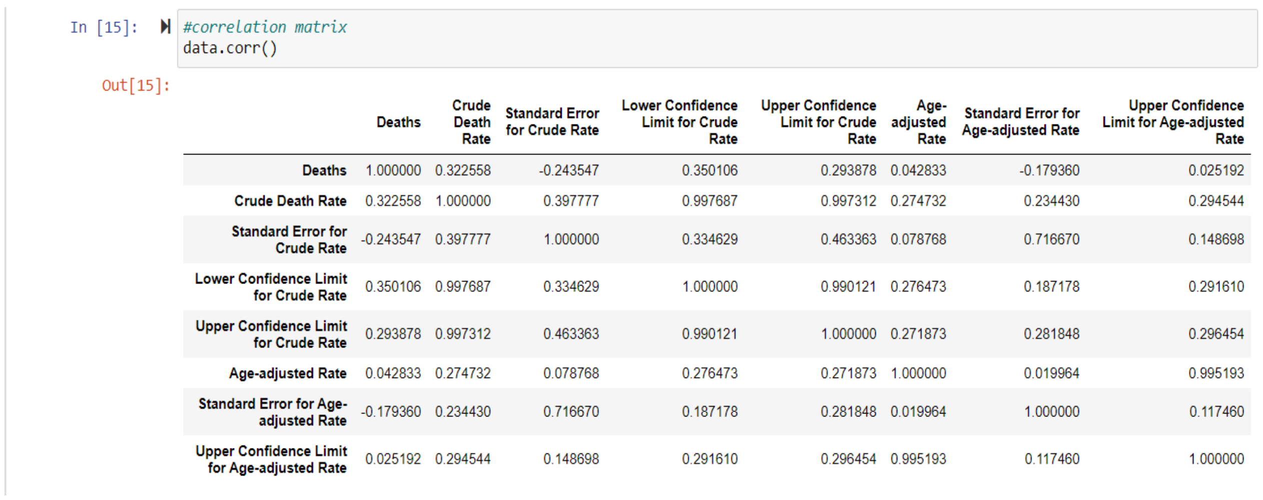
Firstly, the data is imported from the specified location. Then, the data is checked for the sum of null values in all the columns. Next, the variables which are unrelated to the selected target variable is removed using the in-built function drop and in this selected dataset, three variables are removed totally simply because it causes disturbance in performance of the model. Once the specified columns are dropped, the data types of all the variables are checked using “dtypes” data type object.The next step is that the categorical values are converted to numerical values using LabelEncoder. The resulted dataset columns are then converted to float type to fit into the feature selection process. Then, the null values are replaced using ‘ffill’ method. The resulted data is then assigned to x and y variable. X contains all the columns except for the target variable “Deaths”, and Y variable contains the target variable “Deaths”. After the X and Y variables are assigned, feature selection is implemented to retrieve the columns which gives optimal performance. Once, the appropriate columns are retrived, the rest of the columns are dropped. After the columns are retrieved, the correlation matrix is implemented. The dataset is then split into 80 percent training data and 20 percent testing data. The data is next segregated all the train data with all the independent variables into x_train and test data with all the dependant variables into y_train, and x_test and y_test having dependant variables of train and test data. The model RandomForestRegressor is used and x_train and y_train is then fit into this model. Next, the correctness of the model is predicted using x_test. Then, the accuracy of the model is predicted using model.score which returns the co-efficient of determination R-square of the prediction. And the accuracy for the regression model obtained is “98 percent”. Adding to the above predictions, the model is evaluated using mean absolute error and mean squared error.

The Screenshot for feature selection is as given below –



```
In [12]:  #Using feature selection to select the attributes which gives optimal performance
          model = LinearRegression()
          rfe = RFE(model,8)
          fit = rfe.fit(x,y)
          print("Num Features: %d"% fit.n_features_)
          print("Selected Features: %s"% fit.support_)
          print("Feature Ranking: %s"% fit.ranking_)

          Num Features: 8
          Selected Features: [False False False False False  True  True  True  True  True  True  True
           False  True False]
          Feature Ranking: [6 8 5 3 7 1 1 1 1 1 1 1 4 1 2]
```

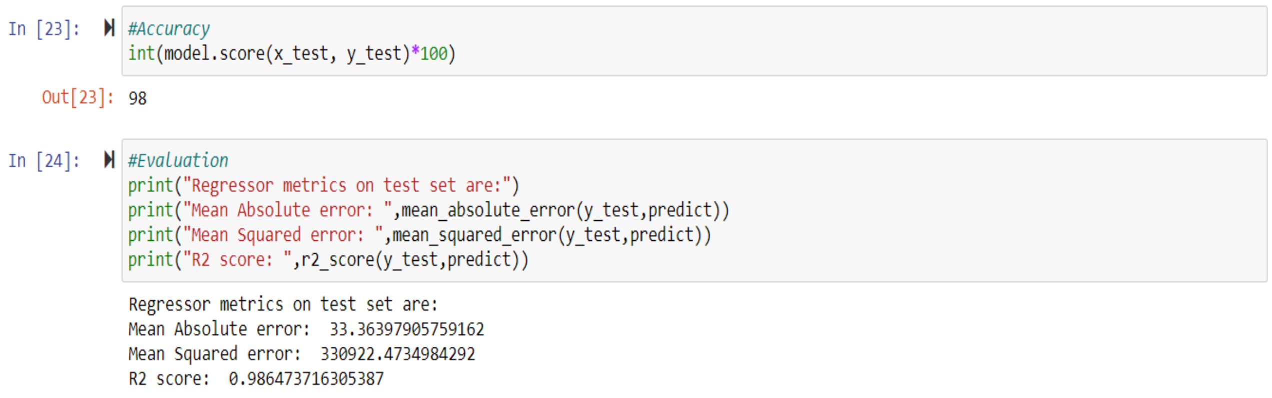
The Screenshot for correlation matrix is as given below –



```
In [15]:  #correlation matrix
          data.corr()

Out[15]:
```

	Deaths	Crude Death Rate	Standard Error for Crude Rate	Lower Confidence Limit for Crude Rate	Upper Confidence Limit for Crude Rate	Age-adjusted Rate	Standard Error for Age-adjusted Rate	Upper Confidence Limit for Age-adjusted Rate
Deaths	1.000000	0.322558	-0.243547	0.350106	0.293878	0.042833	-0.179360	0.025192
Crude Death Rate	0.322558	1.000000	0.397777	0.997687	0.997312	0.274732	0.234430	0.294544
Standard Error for Crude Rate	-0.243547	0.397777	1.000000	0.334629	0.463363	0.078768	0.716670	0.148698
Lower Confidence Limit for Crude Rate	0.350106	0.997687	0.334629	1.000000	0.990121	0.276473	0.187178	0.291610
Upper Confidence Limit for Crude Rate	0.293878	0.997312	0.463363	0.990121	1.000000	0.271873	0.281848	0.296454
Age-adjusted Rate	0.042833	0.274732	0.078768	0.276473	0.271873	1.000000	0.019964	0.995193
Standard Error for Age-adjusted Rate	-0.179360	0.234430	0.716670	0.187178	0.281848	0.019964	1.000000	0.117460
Upper Confidence Limit for Age-adjusted Rate	0.025192	0.294544	0.148698	0.291610	0.296454	0.995193	0.117460	1.000000

The Screenshot for Accuracy and Evaluation of the model is as given below –



```
In [23]:  #Accuracy
          int(model.score(x_test, y_test)*100)

Out[23]:  98

In [24]:  #Evaluation
          print("Regressor metrics on test set are:")
          print("Mean Absolute error: ",mean_absolute_error(y_test,predict))
          print("Mean Squared error: ",mean_squared_error(y_test,predict))
          print("R2 score: ",r2_score(y_test,predict))

          Regressor metrics on test set are:
          Mean Absolute error:  33.36397905759162
          Mean Squared error:  330922.4734984292
          R2 score:  0.986473716305387
```

Note : Please find the dataset in the link below :

https://catalog.data.gov/dataset/nchs-drug-poisoning-mortality-by-state-united-states

CLASSIFICATION REPORT

For the classification model, a dataset is chosen which is related to the Dams in Vermont. Here, For the categorical value, the variable “DamHazClass” is chosen as it gives the information on the status of the dams, that is whether the Dam is minimal, maximum or not hazardous at all.

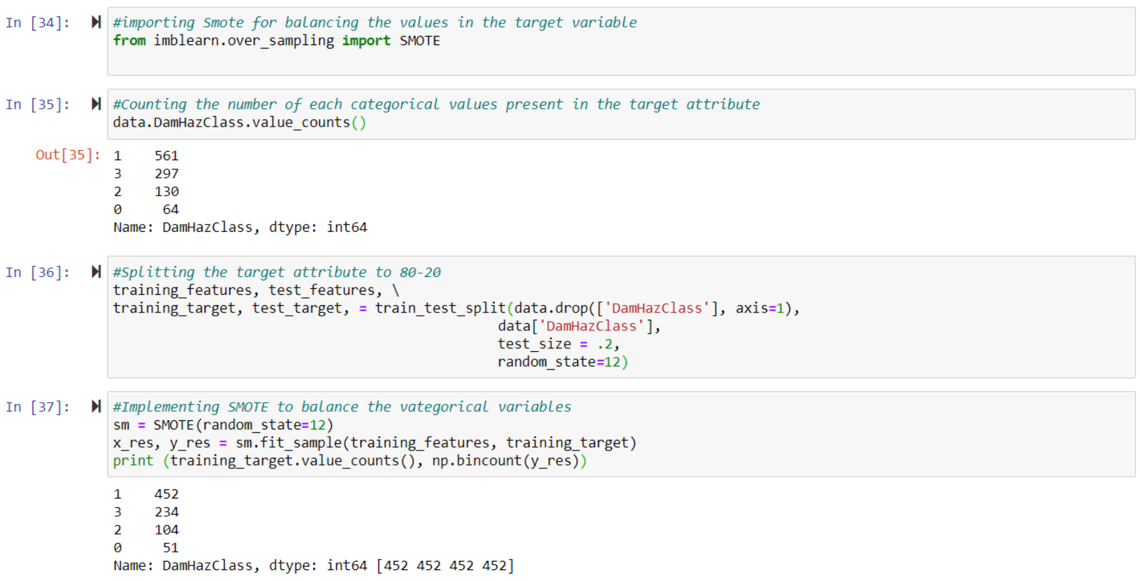
Firstly, the pandas is imported to read the data from the local disk. OneHotEncoder is imported which encodes categorical features as one-hot numeric array. LabelEncoder is used which encodes target names with an incentive among 0 and n_classes-1. The contribution to this transformer ought to be a cluster like numbers or strings, indicating the qualities taken on by categorical features. A RandomForestClassifier is used which acts as a meta estimator that fits various decision tree classifiers on different sub-tests of the dataset and utilization, averaging to improve the precision and control over-fitting. The train_test_split is imported which splits arrays into irregular train and test subsets. The confusion_matrix is used to assess the exactness of a grouping. The accuracy_score is used to capacity subset exactness: the arrangement of labels anticipated for a sample should precisely coordinate the corresponding set of labels in y_true. The classification_report report is used to assemble a book report indicating the fundamental classification measurements. The matplotlib.pyplot is imported which acts as a state-based interface to matplotlib. It gives a MATLAB-like method for plotting. And finally, numpy is imported to fill the null values. LogisticRegression and RFE is imported to implement the feature selection.

Firstly, the data is imported from the specified location. Next, the variables which are unrelated to the selected target variable is removed using the in-built function drop and in this selected dataset, fifteen variables are removed totally simply because it causes disturbance in performance of the model. Once the specified columns are dropped, the data types of all the variables are checked using “dtypes” data type object. Then the categorical features present in the dataset is converted into numerical features. Then, the data is checked for the sum of null values in all the columns. The null values are removed using ‘ffill’ method. Then the target value ‘DamHazClass’ is imbalanced. So, in order to balance it, oversampling is used using SMOTE. The resulted dataset columns are then assigned to X and Y variables. X contains all the columns except for the target variable “DamHazClass’”, and Y variable contains the target variable “DamHazClass’”. After the X and Y variables are assigned, feature selection is done in order to retrieve the columns which gives optimal performance. Once the columns are retrieved, the other features are dropped. The dataset is then split into 60 percent training data and 40 percent testing data. The data is next segregated all the train data with all the independent variables into x_train and test data with all the dependant variables into y_train, and x_test and y_test having dependant variables of train and test data. The model RandomForestClassifier is used and x_train and y_train is then fit into this model. Next, the correctness of the model is predicted using x_test. Then, the accuracy of the model is predicted using model.score which returns the co-efficient of determination R-square of the prediction. The accuracy of the model obtained is “82 percent”.

Note : Please find the link to the dataset below :

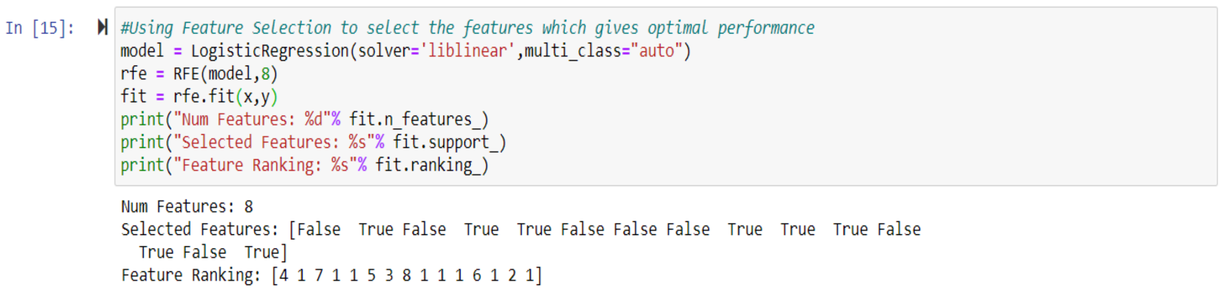
https://catalog.data.gov/dataset/dams-3816f

The Screenshot for the sampling of the target variable is as given below –



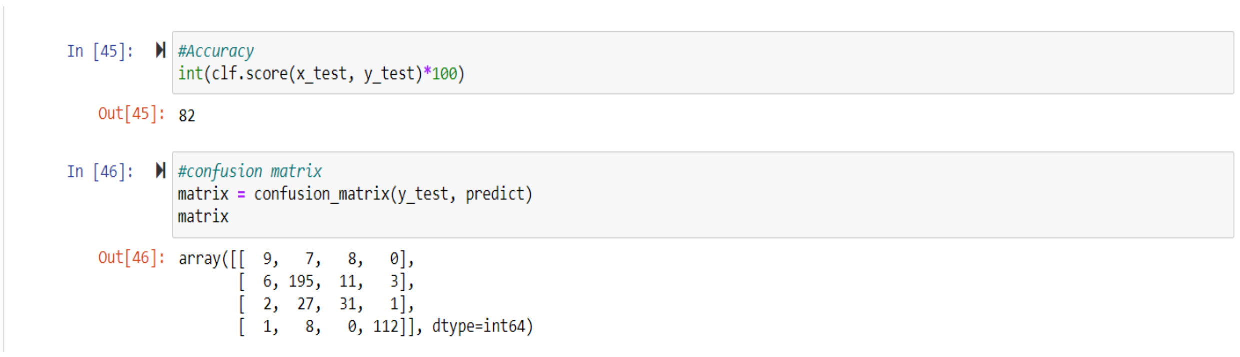
```
In [34]:  #importing Smote for balancing the values in the target variable
          from imblearn.over_sampling import SMOTE

In [35]:  #Counting the number of each categorical values present in the target attribute
          data.DamHazClass.value_counts()

Out[35]:  1    561
          3    297
          2    130
          0     64
          Name: DamHazClass, dtype: int64

In [36]:  #Splitting the target attribute to 80-20
          training_features, test_features, \
          training_target, test_target, = train_test_split(data.drop(['DamHazClass'], axis=1),
                                                           data['DamHazClass'],
                                                           test_size = .2,
                                                           random_state=12)

In [37]:  #Implementing SMOTE to balance the vategorical variables
          sm = SMOTE(random_state=12)
          x_res, y_res = sm.fit_sample(training_features, training_target)
          print (training_target.value_counts(), np.bincount(y_res))

          1    452
          3    234
          2    104
          0     51
          Name: DamHazClass, dtype: int64 [452 452 452 452]
```

The Screenshot of the Feature Selection is as given below –



```
In [15]:  #Using Feature Selection to select the features which gives optimal performance
          model = LogisticRegression(solver='liblinear',multi_class="auto")
          rfe = RFE(model,8)
          fit = rfe.fit(x,y)
          print("Num Features: %d"% fit.n_features_)
          print("Selected Features: %s"% fit.support_)
          print("Feature Ranking: %s"% fit.ranking_)

          Num Features: 8
          Selected Features: [False  True False  True  True False False False  True  True  True False
            True False  True]
          Feature Ranking: [4 1 7 1 1 5 3 8 1 1 1 6 1 2 1]
```

The Screenshot of Accuracy and the Confusion matrix is as given below –



```
In [45]:  #Accuracy
          int(clf.score(x_test, y_test)*100)

Out[45]:  82

In [46]:  #confusion matrix
          matrix = confusion_matrix(y_test, predict)
          matrix

Out[46]:  array([[  9,   7,   8,   0],
                 [  6, 195,  11,   3],
                 [  2,  27,  31,   1],
                 [  1,   8,   0, 112]], dtype=int64)
```

Takeaway : The main thing I learnt from this project is how to apply various models to get the optimised accuracy from the applied machine learning model. I had issues with applying the feature selection for both the models, but it was worth trying. Overall, I learnt how the machine learning model justifies the huge dataset coming from the various organization.