Report

Word count:1173

Introduction:

This report helps to analyze the tasks performed while handling two different data sets. In order to handle first data set, Classification models are implemented and best model from the executed models is used for prediction task. Similarly, for second data set, the best model from implemented Regression models is used for prediction tasks. The model comparison is done on various factors in both the data sets. Data preprocessing is carried out before the models are fitted on the datasets. Primarily sklearn and pandas libraries are used extensively throughout the tasks.

Implementation:

A) Task 2:

a) Part A:

- Data Preprocessing:

In task A, 'CE802_P2_Data.csv' data is imported into a dataframe in python. The dataset has 1000 rows and 22 columns however F21 column had 500 null values. "Eliminating null values using dropna" imputation method was not appropriate for the dataset as almost 50% of data from the dataset would have been removed if axis=0 i.e delete rows that have null values. If axis=1 then the whole column of F21 would have been eliminated which is not a good practice as 'F21' is crucial feature for target variable 'Class'. Thus, mean value = -10.18352 is used to replace null values. All the data types are checked of each feature and converted to 'float64' wherever necessary. The relationship of each feature is checked with the 'Class' target variable constant on X axis.

- Classifiers:

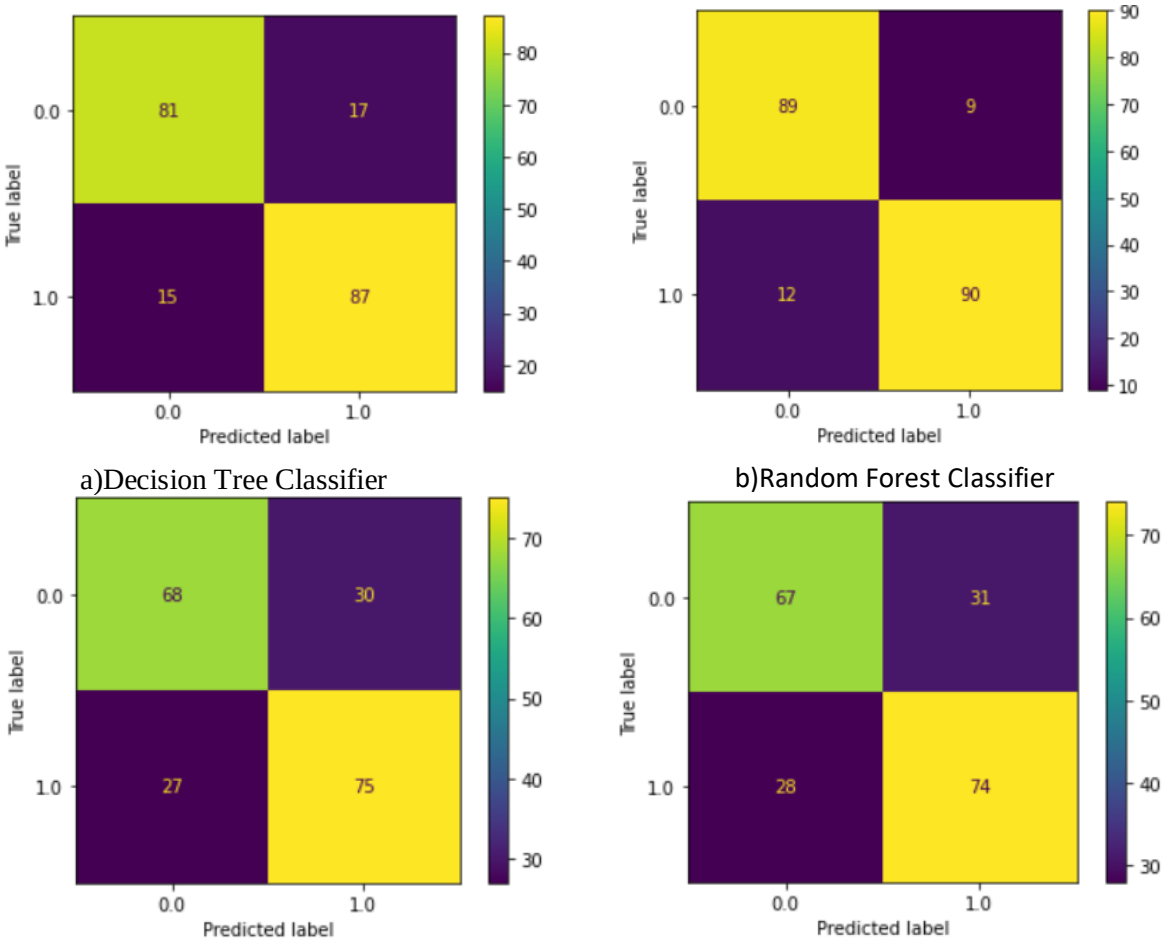
The input features (F1 to F21) and the target variable (Class) is separately stored in variables in order to use it further. In each of classifiers that are implemented the data is split into training and test data where the test data split acts as validation split in order to check the efficiency of the model. The following 4 classifiers are implemented and then compared:

1) Decision Tree Classifier:
   Decision Tree Classifier is implemented after splitting the data into 80:20 ratio split. The accuracy of classifier is 0.84. The number of mislabeled points were 32 out of 200.
2) Random Forest Classifier:
   Random Forest Classifier has also 80:20 split ratio. N_estimators is the attribute that helps to decide number of trees required before any average predictions are taken. This parameter is tuned to 155 due to which mislabeled points drop to 21 out of 200 and accuracy increases to 0.895.
3) Support Vector Machine Classifier:
   The kernel in support vector machine classifier is considered to linear. This gives the accuracy of 0.715 and mislabeled points of 57 out of 200
4) Naïve Bayes Classifier:
   GaussianNB model is imported which is used for implementing Naïve Bayes Classifier. After the 80:20 split, the mislabeled points of 59 out of 200 are

obtained here which is more that 25%. The lowest accuracy noted for this data set is 0.705 in this classifier.

- Comparison of models:

Confusion matrices of all the classifiers are shown below:



a)Decision Tree Classifier

b)Random Forest Classifier

c) SVM Classifier

d) Naïve Bayes Classifier

Fig: Confusion matrices of all models

The f1_scores of the classifiers are as follows:

Name of Classifier	F1_score
1)Decision Tree Classifier	0.8399
2)Random Forest Classifier	0.895
3)SVM Classifier	0.715
4)Naïve Bayes Classifier	0.705

Therefore, from the value of accuracy, f1_score and the confusion matrix it is clear that Random Forest Classifier provides optimum results in all the 4 implemented models. Thus, Random Forest Classifier is used for prediction task i.e Part B.

b)Part B:

In this part, 'CE802_P2_Test.csv' dataset is imported in a dataframe. Data preprocessing is performed as done in Part A. The dataset is checked for null values and the 'Replacement by mean value' imputation method is used . As mentioned above, Random Forest Classifier is used for prediction on

'test_data'. The predicted values are saved in last empty column of the csv file that is imported . This completed file is then exported with 'CE802_P2_Test_Predictions.csv' name. Here, the second task is completed.

B) Task 3:

a) Part A:

- Data Preprocessing:

  In this task, for part A 'CE802_P3_Data.csv' data set is imported that has 1500 rows and 37 columns. As this data set had no null values, int and object data types were transformed into float data type. Factorize method of data frame was used for transformation of categorical values into numerical values. From, 'F4' column, 4 unique values of UK, Europe, USA and Rest were encoded as 0.0, 1.0,2.0 and 3.0 respectively. Similarly, 5 unique values very low, low, high, medium, very high were encoded from 0.0 to 4.0 respectively. A plot of subplots where each subplot shows relation of particular attribute with 'Target' output variable is plotted. Each attribute depicts relationship with 'Target' due to which no attribute is dropped.

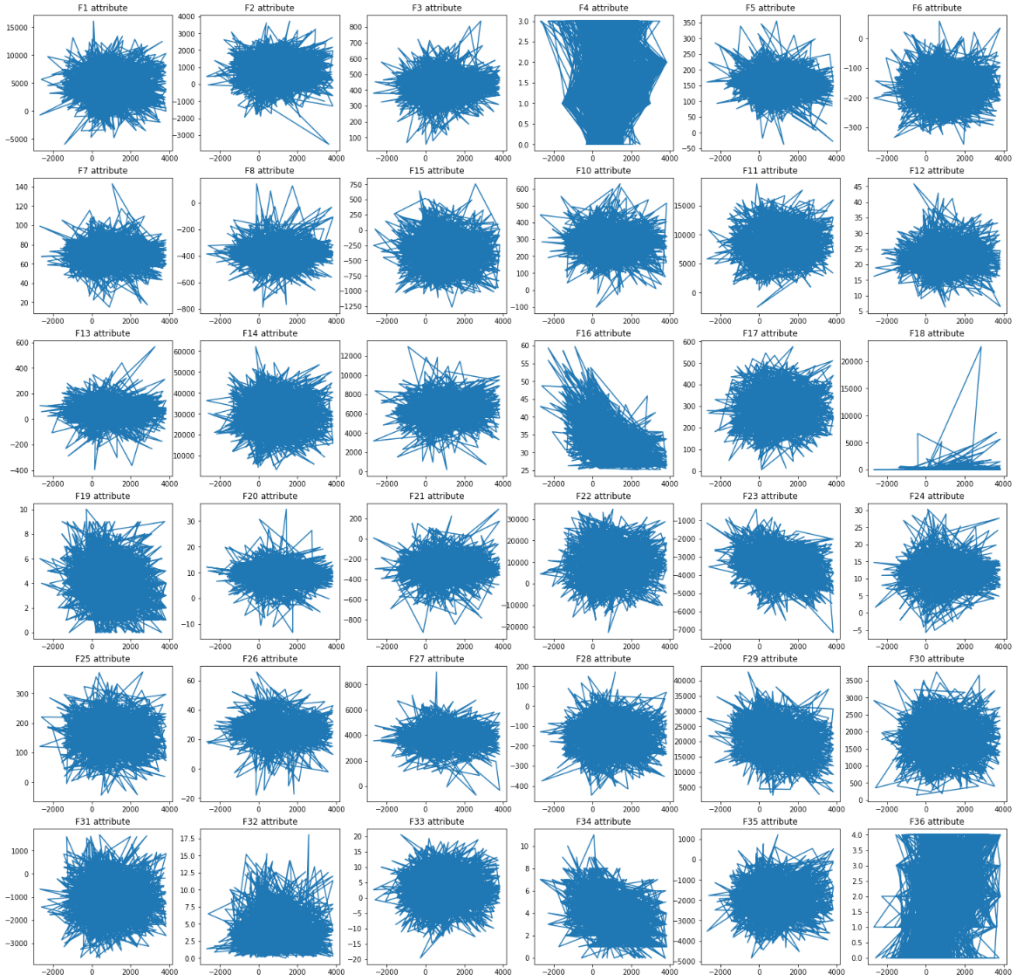


Fig: Relation of Features with Target

- Regression Models:

  The features (F1 to F36) and target variable (Target) is saved separately for use in the models. The data is split in 80:20 ratio for all regression models into training

and testing data as done in step 2. Here, as Regression models are used, r2_score and errors are obtained. The following regression models are implemented:

1)Linear Regression model:

The Linear Regression model is fitted on training data and predictions are obtained. The coefficients of linear regression model are printed. The RMSE(Root Mean Square Error) value is around 794.3744.

2)Lasso Regression model:

The step prior to model fitting in this model is tuning hyperparameters. Alpha , fit intercept and normalization methods are tuned. The RMSE value is around 787.667.

3)Random Forest Regression model:

In this model, the max_depth i.e the max_depth of the tree that is formed is tuned to 3. The RMSE value obtained is 934.0189.

4)Ridge Regression model:

Ridge regression gives coefficients and RMSE values 794.3685 after tuning alpha value to 1.0.

- OLS Regression results for the whole input and output variables is as follows:

OLS Regression Results

Dep. Variable:	Target	**R-squared:**	0.610
Model:	OLS	**Adj. R-squared:**	0.600
Method:	Least Squares	**F-statistic:**	63.46
Date:	Tue, 18 Jan 2022	**Prob (F-statistic):**	5.77e-269
Time:	21:49:46	**Log-Likelihood:**	-12086.
No. Observations:	1500	**AIC:**	2.425e+04
Df Residuals:	1463	**BIC:**	2.444e+04
Df Model:	36		
Covariance Type:	nonrobust		

- Comparison of models:

The Coefficient of Determination values, should range between 0 to 1, of all the regression models are as follows:

Name of Regressor	Coefficient of Determination(r2_score)
1)Linear Regression model	0.57003
2)Lasso Regression model	0.57726
3)Random Forest Regression model	0.40558
4)Ridge Regression model	0.57004

From r2_score values and RMSE values mentioned above it is clear that Lasso Regression model provides better results than the remaining models. Thus, Lasso Regression model is used for prediction task in Part B.

b) Part B:

In this part, 'CE802_P3_Test.csv' dataset is imported with last column of ‘Target’ to be predicted as empty. As mentioned above, Lasso regression model is used for prediction in this part. The object of trained model from Part A is used for prediction. However, before prediction, the data set is preprocessed. All the null values are checked and then the data types of ‘F19’ and ‘F34’ are changed to float. Along with that, ‘F4’ and ‘F36’ categorical

attributes are converted into numerical values. After prediction the predicted values are stored in 'Target' column and 'CE802_P3_Test_Predictions.csv' file is exported.

Conclusion:

In conclusion, for second question **Random Forest Classifier** is used for prediction with highest accuracy of 0.895. In the next task, **Lasso regressor** is used for prediction task with highest coefficient of Determination to be 0.5776.

References:

1) https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html?highlight=decision%20tree%20classifier#sklearn.tree.DecisionTreeClassifier
2) https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
3) https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
4) https://www.datacamp.com/community/tutorials/naive-bayes-scikit-learn
5) https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html
6) https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Lasso.html
7) https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html
8) https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html
9) https://www.analyticsvidhya.com/blog/2016/01/ridge-lasso-regression-python-complete-tutorial/
10) https://www.statsmodels.org/dev/examples/notebooks/generated/ols.html
11) https://matplotlib.org/stable/gallery/subplots_axes_and_figures/subplots_demo.html
12) https://pandas.pydata.org/docs/reference/api/pandas.factorize.html