IT 461 Practical Machine Learning
Fall 2020

Project Part II

Name: Raghad Aloraini
Main Topic of Problem: Prediction of application rating

I. Introduction

a) Introducing the topic and why is it useful to use ML algorithms on it.

The dataset is https://www.kaggle.com/lava18/google-play-store-apps and we are using this dataset to make prediction of ratings of Apps, hence we use ML algorithms on it.

b) What is the problem you want to solve and what are you proposing to solve it with?
The Play Store Apps data has enormous potential to drive app-making business to success. Hence by drawing meaningful insights from data by doing analysis it can really help developers for their work and also give insights to users based on what app is popular. By using this dataset, we are trying to make predictions of the ratings of the apps, using features such as Category, Reviews, Size, Installs, Type, Price, Content Rating, Genres, Last Updated, Current Ver, Android Ver.

II. Data Preprocessing

a) Show what preprocessing techniques you will be using
Checking for missing values.
Dropping missing values.
One hot encoding categorical variables.
Scaling the values.

b) Why did you choose to apply these techniques

The Preprocessing techniques that I had used, were I first found the missing values in the dataset. The ratings column had most missing values and I decided to drop those na values. Also, I added 2 more columns in the data set by spliting the last updated attribute, by doing this we find that in which year apps are added or updated on playstore. Also, I hot encoded the categorical variables Category, Type and Content Rating because the

ml models work better with numerical values, and then I dropped the columns App, Size, Price, Genres, Last Updated, Current Ver, Android Ver because they didn't add any value to our predictions of ratings. We are scaling the data so that the distribution has a value of 0 and standard deviation of 1.

c) Provide examples of preprocessing steps

```
#check missing values
df.isnull().sum()
```

```
App                0
Category           1
Rating          1474
Reviews            0
Size               0
Installs           0
Type               1
Price              0
Content Rating     0
Genres             1
Last Updated       0
Current Ver        8
Android Ver        2
dtype: int64
```

```
[79] df.dropna(inplace=True)
```

```
[80] print('Number of rows after deleting: ', df.shape[0])

     Number of rows after deleting:  9360
```

```
[81] df["Last Updated"] = pd.to_datetime(df['Last Updated'])
     df['year_added']=df['Last Updated'].dt.year
     df['month_added']=df['Last Updated'].dt.month
```

```
[84] #one hot encode categorical variables
     catgry=pd.get_dummies(df['Category'],prefix='catg',drop_first=True)
     typ=pd.get_dummies(df['Type'],prefix='typ',drop_first=True)
     cr=pd.get_dummies(df['Content Rating'],prefix='cr',drop_first=True)
     frames=[df,catgry,typ,cr]
     df=pd.concat(frames,axis=1)
     df.drop(['Category','Installs','Type','Content Rating'],axis=1,inplace=True)
```

```
[85] #drop columns because we need to make prediction of rating and these dont add value for making predictions
     df.drop(['App','Size','Price','Genres','Last Updated','Current Ver','Android Ver'],axis=1,inplace=True)
```

```
[86] df.head(2)
```

	Rating	Reviews	year_added	month_added	catg_AUTO_AND_VEHICLES	catg_BEAUTY	catg_BOOKS_AND_REFERENCE	catg_BUSINESS	catg_COMICS	catg_COMM
0	4.1	159	2018	1	0	0	0	0	0	
1	3.9	967	2018	1	0	0	0	0	0	

III. Methodology

a) Explain the ML algorithm used

I tried three different ML algorithms:

Logistic Regression:

Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. In regression analysis, logistic regression (or logit regression) is estimating the parameters of a logistic model (a form of binary regression)

SVM:

As the name suggest the SVR is an regression algorithm, so we can use SVR for working with continuous Values instead of Classification which is SVM

Random Forest:

The random forest is a classification algorithm consisting of many decision trees. It uses bagging and feature randomness when building each individual tree to try to create an uncorrelated forest of trees whose prediction by committee is more accurate than that of any individual tree

b) Implementation (with screenshots)

```
#LogisticRegression
lr_c=LogisticRegression(random_state=0, solver='lbfgs',max_iter=400 )
lr_c.fit(X_train,y_train)
lr_pred=lr_c.predict(X_test)
lr_cm=confusion_matrix(y_test,lr_pred)
lr_ac=accuracy_score(y_test, lr_pred)
print('Accuracy of Logistic Regression Model: ', lr_ac)
```

```
Accuracy of Logistic Regression Model:  0.7622863247863247
```

```
[107] #Random Forest
      rdf_c=RandomForestClassifier(n_estimators=10,criterion='entropy',random_state=0)
      rdf_c.fit(X_train,y_train)
      rdf_pred=rdf_c.predict(X_test)
      rdf_ac=accuracy_score(y_test, rdf_pred)
      print('Accuracy of Random Forest: ', rdf_ac)

      Accuracy of Random Forest:  0.7355769230769231
```

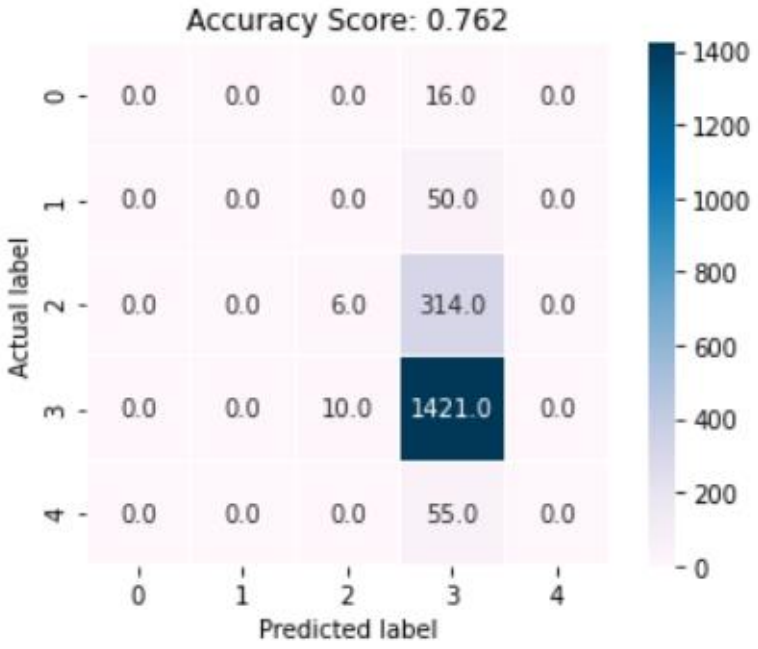
```
[102] svc_r=SVC(kernel='rbf', random_state=0)
      svc_r.fit(X_train,y_train)
      svr_pred=svc_r.predict(X_test)
      svr_ac=accuracy_score(y_test, lr_pred)
      print('Accuracy of SVM Model: ', svr_ac)

      Accuracy of SVM Model:  0.7622863247863247
```

c) What are the initial parameters of the model if any
Setting random_state of all models to 0 so that the model results can be reproducible.
Logistic Regression parameters: solver= 'lbfgs', max_iter= 400
SVM: kernel= 'rbf',.
Random Forest: n_estimators= 10, criterion='entropy'

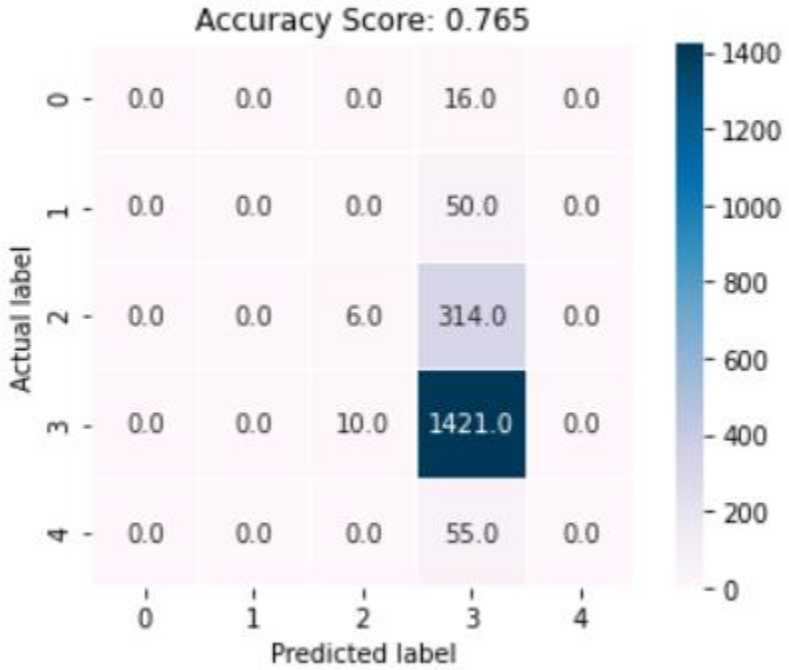
IV. Evaluation and Results

a) Show your results (tables, figure, confusion matrices …etc)

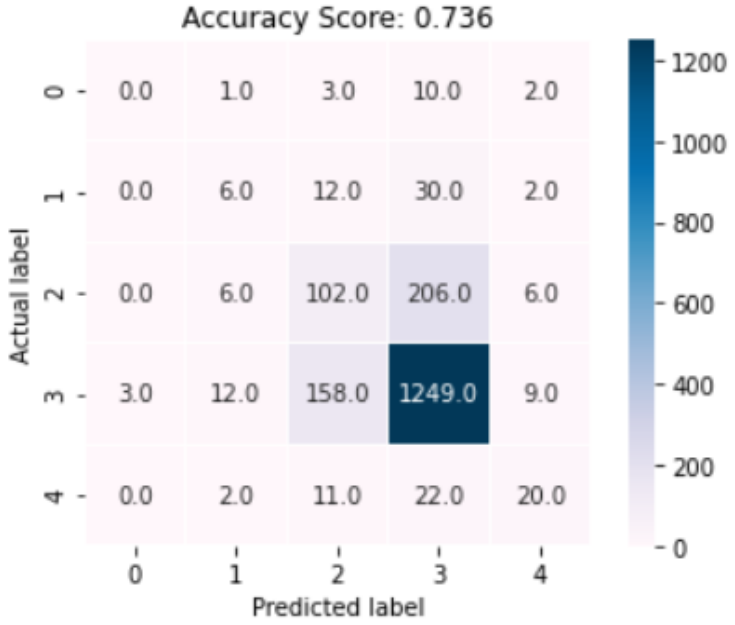
Confusion Matrix for Logistic Regression Classifier



Confusion Matrix for SVM



Confusion Matrix for Random Forest Classifier

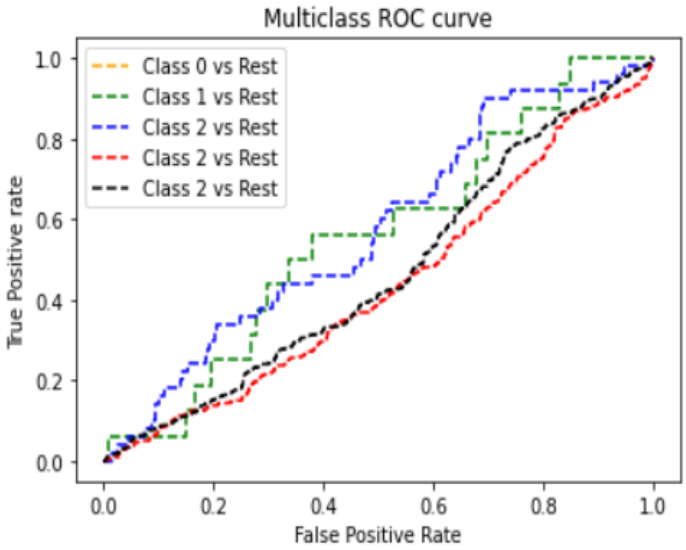


b) Plot your output (visualization of error)
I have plotted the ROC curve for SVM, since SVM performs best in terms of accuracy among the three algorithms. Since this is a multiclass problem as we have converted ratings column to int, we use a multiclass ROC curve. The ROC curve tells us how good the model is at distinguishing classes. The SVM has a roc auc score of 50%.

```
from sklearn.preprocessing import LabelBinarizer
multiclass_roc_auc_score(y_test, svr_pred)
```

```
0.5007015817495383
```

```
/usr/local/lib/python3.6/dist-packages/sklearn/metrics/_ranking.py:808: Un
  UndefinedMetricWarning)
```



c) Which evaluation techniques is better and why?
The key to a fair comparison of machine learning algorithms is ensuring that each algorithm is evaluated in the same way on the same data. You can achieve this by forcing each algorithm to be evaluated on a consistent test harness. We use accuracy as the evaluation technique for the models. Comparing the accuracy of the three models, we conclude that SVM has the highest accuracy of 76.5 followed by Logistic Regression 76.2 and Random Forest 73.6

V. Analysis and conclusion

a) Explain your results in analytical way instead of numbers. Meaning, why do you think one algorithm performed that way and the other didn't?
SVM performed better than the other models. I think SVM performed better because there were not many features in the dataset, whereas Logistic regression require the dataset to have a lot more features for a better accuracy and also, the number of training samples were large. Since Logistic Regression work better with small training samples (10-1000) while SVM can work with many training samples in the range of (10-10000). SVM is also less prone to overfitting in comparison to Logistic Regression

VI. References (if Any)