OUTSTANDING PROJECT 2

Mudit Mathur – mm7692@srmist.edu.in

Problem: Rating Prediction of Stack Overflow Questions

DESCRIPTION OF DATASET: Dataset contains Id, Title, Body, Creation Date, Quality of Question(Y) and tags.

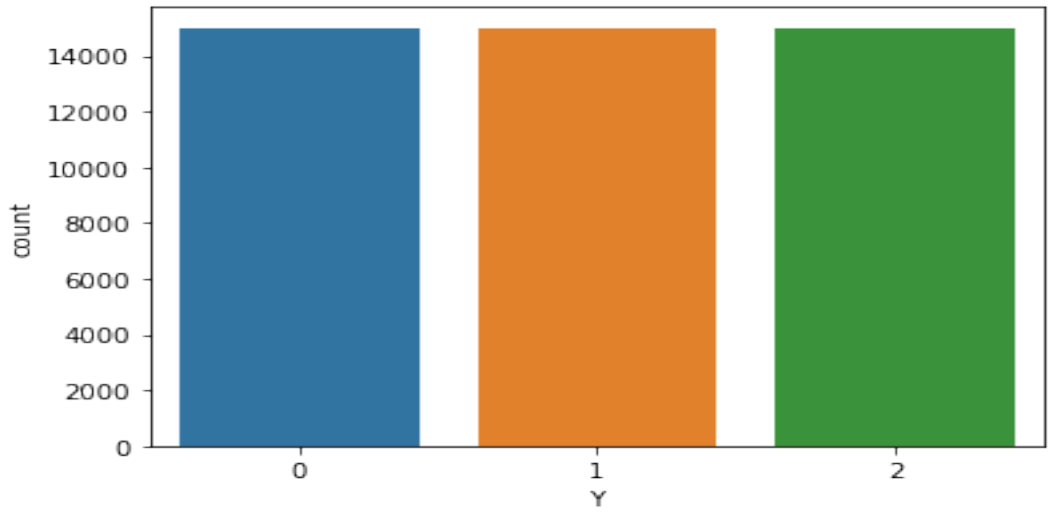
Total number of Entries are 45000 and 6 Features(Columns).

LIBRARIES USED:
Pandas, Numpy, Matplotlib, Seaborn, Sci-kit learn

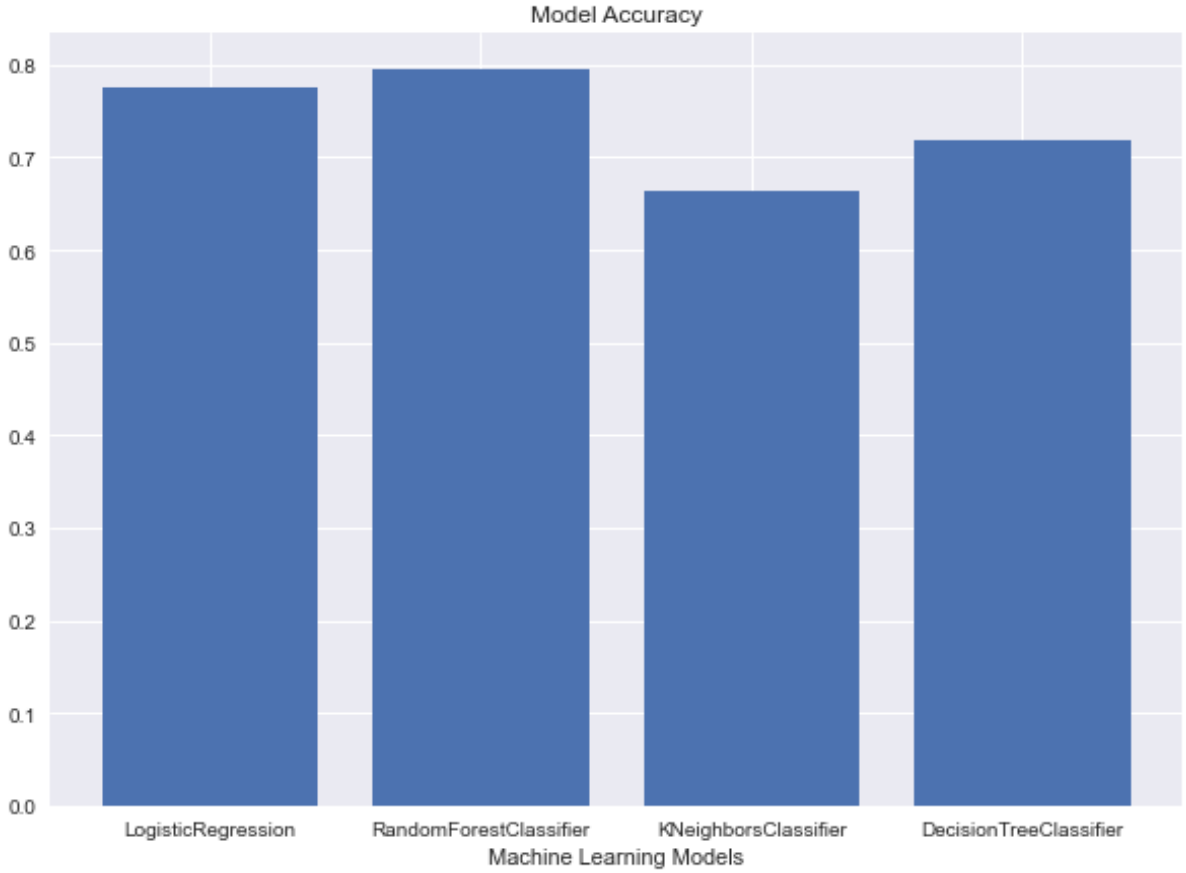
DATA CLEANING:

1. Feature Selection of Body and Y(Target Variable).
2. Removing the <p> in Body of every row.
3. Importing String and removing punctuations and converting into lower.
4. Converting Y to Numerical Data using Dictionary.
5. Converting Y into INT data.
6. Importing nltk.Stopwords and removing them from Body.
7. Count-vectorizing the text of the Body

EXPLORATORY DATA ANALYSIS (EDA):
The Number of Quality of Question VS the count of them



Model Accuracy



MACHINE LEARNING AND ALGORITHMS:

Using Natural Language Processing, Modeling with,

1. Logistic Regression
2. Random Forest Classifier
3. KNN
4. Decision Tree Classifier

LOGISTIC REGRESSION: (ACCURACY 78%)

```
lr=LogisticRegression()
lr.fit(X_train,y_train)
predlr=lr.predict(X_test)
print(confusion_matrix(y_test,predlr))
print("\n")
print(classification_report(y_test,predlr))
```

```
[[1019  235  207]
 [ 129 1368   34]
 [ 310   79 1119]]
```

	precision	recall	f1-score	support
0	0.70	0.70	0.70	1461
1	0.81	0.89	0.85	1531
2	0.82	0.74	0.78	1508
accuracy			0.78	4500
macro avg	0.78	0.78	0.78	4500
weighted avg	0.78	0.78	0.78	4500

RANDOM FOREST (ACCURACY 80%)

```
rf=RandomForestClassifier()
rf.fit(X_train,y_train)
predrf=rf.predict(X_test)
print(confusion_matrix(y_test,predrf))
print("\n")
print(classification_report(y_test,predrf))
```

```
[[1026  198  237]
 [ 130 1365   36]
 [ 244   69 1195]]
```

	precision	recall	f1-score	support
0	0.73	0.70	0.72	1461
1	0.84	0.89	0.86	1531
2	0.81	0.79	0.80	1508
accuracy			0.80	4500
macro avg	0.79	0.80	0.79	4500
weighted avg	0.80	0.80	0.80	4500

KNN (ACCURACY 66%)

```
knn=KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train,y_train)
predknn=knn.predict(X_test)
print(confusion_matrix(y_test,predknn))
print("\n")
print(classification_report(y_test,predknn))
```

```
[[1029  197  235]
 [ 406 1083   42]
 [ 558   95  855]]
```

	precision	recall	f1-score	support
0	0.52	0.70	0.60	1461
1	0.79	0.71	0.75	1531
2	0.76	0.57	0.65	1508
accuracy			0.66	4500
macro avg	0.69	0.66	0.66	4500
weighted avg	0.69	0.66	0.66	4500

DECISION TREE (ACCURACY 72%)

```
dt=DecisionTreeClassifier()
dt.fit(X_train,y_train)
preddt=dt.predict(X_test)
print(confusion_matrix(y_test,preddt))
print("\n")
print(classification_report(y_test,preddt))
```

```
[[ 937  211  313]
 [ 229 1209   93]
 [ 333   84 1091]]
```

	precision	recall	f1-score	support
0	0.63	0.64	0.63	1461
1	0.80	0.79	0.80	1531
2	0.73	0.72	0.73	1508
accuracy			0.72	4500
macro avg	0.72	0.72	0.72	4500
weighted avg	0.72	0.72	0.72	4500

CONCLUSION:

The Best Machine Learning Model is **Random Forest Classifier (80% Accuracy)**