CSCI 630
Foundations of Intelligent Systems

Project 2 Report

Prepared By:
Akash Desai(ad3059), Vaibhav Joshi(vj3470)

TABLE OF CONTENTS:

1

INTRODUCTION

Goal:
The goal for this project is build a machine learning agent that can predict which subreddit an unlabeled post comes from.

Environment:
The environment of this project uses majority components from Praw package, Python's SciKit, Pandas, NumPy and MatPlotLib. The whole project is designed and tested on PyCharm IDE.

Scikit(sklearn) Library: It provides various algorithms for the supervised and unsupervised Learning and provides tools for data analysis and data mining tasks. It is build upon NumPy, SciPy and matplotlib. We have used this for creating feature vectors, training the models, prediction, calculating accuracy and generating learning curves.

NumPy and Pandas : These two are used primarily for dealing with the data. They have useful method which are helpful in splitting, shuffling, reshaping the data.

Praw API: PRAW, an acronym for "Python Reddit API Wrapper". It is a python package that allow simple access to Reddit's API. PRAW aims to be easy to use and internally follows all of Reddit's API rules. We use this to authenticate with Reddit and firing request for fetching the posts from subreddits.

Setting up the environment :
Since Python is the main building arena for this project and everything is to be installed on Python using the **pip** command.

Open Pycharm terminal and type the following commands :

Install sklearn :

> pip install sklearn

Install numpy and pandas:

> pip install numpy
> pip install pandas

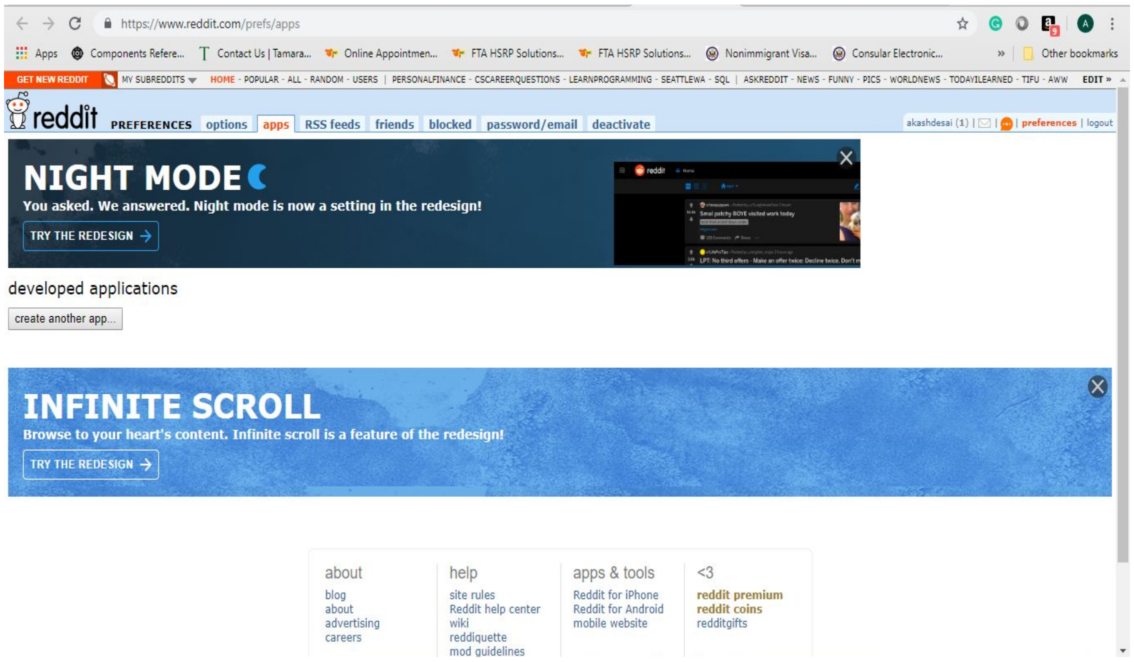
Reddit Application:

Three types of application:
(1) Web Application
(2) Installed Application
(3) Script Application

For this project, we will use script application. Script application is basically used for the personal application. It is used for developer use.

Steps to create a reddit script application:

(1) Create a [script application](https://www.reddit.com/prefs/apps):



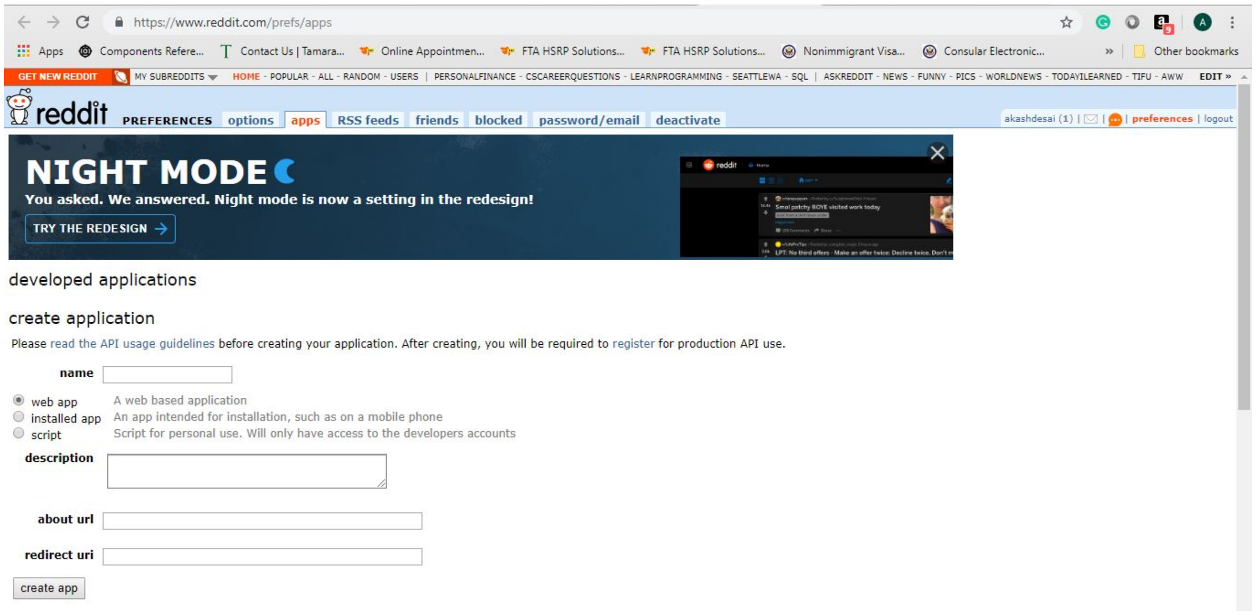
You asked. We answered. Night mode is now a setting in the redesign!

developed applications

create another app...

Browse to your heart's content. Infinite scroll is a feature of the redesign!

(2) Click on the ‘Create another app’.


https://www.reddit.com/prefs/apps
reddit
PREFERENCES
options
apps
RSS feeds
friends
blocked
password/email
deactivate
akashdesai (1) | preferences | logout
NIGHT MODE
You asked. We answered. Night mode is now a setting in the redesign!
TRY THE REDESIGN

developed applications

create application

Please read the API usage guidelines before creating your application. After creating, you will be required to register for production API use.

name	
[✓ web app]	A web based application
installed app	An app intended for installation, such as on a mobile phone
script	Script for personal use. Will only have access to the developers accounts
description	
about url	
redirect uri	

create app

Name: Name of the application
Application Type: Script
Description: Information about application
About url: https://www.cs.rit.edu/~cmh/iis.html
Redirect uri: https://www.cs.rit.edu/~cmh/iis.html

2

METHODS

STEPS:

- Collecting data using Praw API
- Data Fetching
- Data Preprocessing

Collecting data using praw API :

```
> reddit = praw.Reddit(client_id=API_client,
        client_secret=API_secret,
        user_agent=user_agent,
        username=user,
        password=passwd)
```

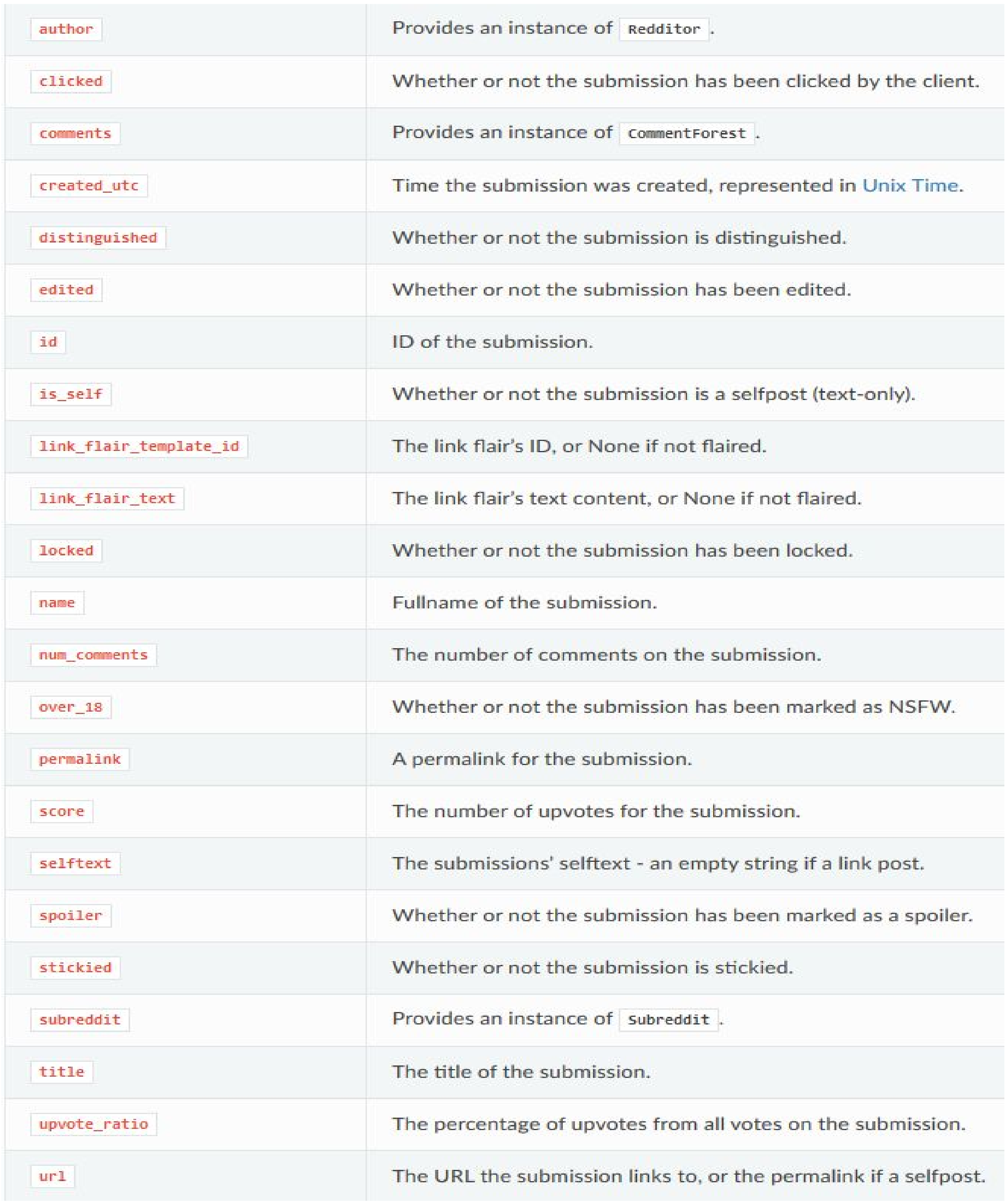
The above will create an authorized Reddit instance. Using this instance, we can perform all the tasks which user can perform.

Data Fetching :-

```
>sub_reddit = reddit.subreddit('soccer').top(limit = 900)
```

The above line of code will fetch top 900 posts from the r/soccer subreddit. We can change this code to get posts from any subreddit by replacing the subreddit name. We can also fetch the new, rising or hot posts by replacing with top accordingly. Limit is as per user requirement. We have taken 900 posts.

Using the *sub_reddit* instance from the previous code snippet, we can use that to extract what attributes we need for our task. The sub_reddit instance has attributes such as :-



`author`	Provides an instance of `Redditor`.
`clicked`	Whether or not the submission has been clicked by the client.
`comments`	Provides an instance of `CommentForest`.
`created_utc`	Time the submission was created, represented in Unix Time.
`distinguished`	Whether or not the submission is distinguished.
`edited`	Whether or not the submission has been edited.
`id`	ID of the submission.
`is_self`	Whether or not the submission is a selfpost (text-only).
`link_flair_template_id`	The link flair's ID, or None if not flaired.
`link_flair_text`	The link flair's text content, or None if not flaired.
`locked`	Whether or not the submission has been locked.
`name`	Fullname of the submission.
`num_comments`	The number of comments on the submission.
`over_18`	Whether or not the submission has been marked as NSFW.
`permalink`	A permalink for the submission.
`score`	The number of upvotes for the submission.
`selftext`	The submissions' selftext - an empty string if a link post.
`spoiler`	Whether or not the submission has been marked as a spoiler.
`stickied`	Whether or not the submission is stickied.
`subreddit`	Provides an instance of `Subreddit`.
`title`	The title of the submission.
`upvote_ratio`	The percentage of upvotes from all votes on the submission.
`url`	The URL the submission links to, or the permalink if a selfpost.

And many more.

Here for the purpose of our project we will pick the title attribute. This attribute will form the feature using which we will be predict the subreddit post.

Data Preprocessing:-

For each subreddit, all the posts, we

Dataframe has two columns : Title and Target
Title: Stores the title of post
Target: 0 for subreddit-1, 1 for subreddit-2, 2 for subreddit-3,.....

for post in allPosts:
    (1) Stores each attribute of post in respective field of data frame
    (2) Value in target field according to the subreddit

Splitting data into Training Set, Development Set and Test Test:

Total data will be divided in 50% Training set, 25% Development Set and 25% Test Set. Each set contains equal number of posts from each subreddit but order will be random.

Feature Extraction:

To perform classification on the text data, first task is the generate feature vector from the text.
Two methods can be used for feature extraction:
(1) Count Vectorise() - It counts no of times each word appears in the text. But, this method will perform well for larger text.
(2) TF-IDF (Term Frequency times Inverse Document Frequency) :
TF is the no of times word appear in the document.
IDF is the weight of the rare words in the document.
Final result is the multiplication of TF and IDF.

To implement TF-IDF in python:

(1) Import TF-IDF from the sklearn library.
    from sklearn.feature_extraction.text import TfidfVectorizer
(2) First step is to remove stop words. Stop words are common words like the, a, an, is , are, you etc. tfidfVectorizer method is used to remove stop words.ENGLISH_STOp_WORDS is provided as input to the stop_words parameter.

(3) tfidf_transformer.fit_transform method is used for feature vector generation. Todense() method is used on the feature vector to flatten it.

3

LEARNING ALGORITHMS

We have preprocessed the data. Now we will train this data using different learning algorithms. They are :

- **Naive Bayes :**

  - The Naive Bayes algorithm is an intuitive method that uses the probabilities of each attribute belonging to each class to make a prediction. It is the supervised learning approach you would come up with if you wanted to model a predictive modeling problem probabilistically.

  - Naive bayes simplifies the calculation of probabilities by assuming that the probability of each attribute belonging to a given class value is independent of all other attributes. This is a strong assumption but results in a fast and effective method.

  - To make a prediction Naive Bayes calculates probabilities of the instance belonging to each class and selects the class value with the highest probability.

- **Implementing Naive Bayes in Python :**

  - Pythons SciKit(sklearn) library has inbuilt modules for Naive Bayes model. We import it using the following command :-

    from sklearn.naive_bayes import MultinomialNB

- ➢ Below snippet shows a basic way to implement the Naive Bayes in Python

Train_NaiveBayes(train_X, train_Y, test_X, test_Y) :

```
1) Using the Naive Bayes instance [MutinomialNB()] fit the training
   data( train_X and train_Y) using sklearns fit method
2) Perform prediction  using the testing data test_X. (Use sklearns
   predict method)
3) Using the result from step 2 compute the accuracy by comparing
with test_y which has the class labels. This can be done using metrics
module from sklearn library
```

- ❖ **Random Forest :**
  - ➢ It uses multiple decision trees on the sample dataset and uses the average of result to improve predictive power of the algorithm. It also reduces over fitting**.**

- ❖ **Implementing Random Forest in Python :**
  - ➢ Pythons SciKit(sklearn) library has inbuilt modules for Random Forest model. We import it using the following command :-

from sklearn.naive_bayes import MultinomialNB

- ➢ Below snippet shows a basic way to implement the Random Forest in Python

Train_NaiveBayes(train_X, train_Y, test_X, test_Y) :

```
3) Using the Naive Bayes instance [MutinomialNB()] fit the training
   data( train_X and train_Y) using sklearns fit method
4) Perform prediction  using the testing data test_X. (Use sklearns
   predict method)
3) Using the result from step 2 compute the accuracy by comparing
with test_y which has the class labels. This can be done using metrics
module from sklearn library
```

- **Support Vector Machine :**
    - The objective of support vector classifier is to find hyperplane that distinctively c lassifies the data points. It is a supervised learning algorithm, because according to the labels data it outputs hyperplane which classifies the new data.
    - Data points on the different side of hyperplane is said to be in different classes.
    - Dimension of hyperplane depends on the number of features of the input data.
    -
- **Three hyperparameters:**
    - **Kernel:** Value of it specifies the type of kernel that will be used in the algorithm.
    - **Regularization:** It defines how much data misclassification of data can be avoided.For higher value of C, algorithm tries to find simple algorithm to classify the data. So, it will give higher accuracy for the new data. This is called underfitting.
    - **Gamma:** It is used for ‘rbf’,’poly’,’sigmoid’ kernel type. High value of gamma represents, points close to the separation line will be considered and low value represents, points far to the separation line will be considered.

- **Implementing Support Vector Classifier in Python :**

    - Pythons SciKit(sklearn) library has inbuilt modules for Support Vecot Classifier model. We import it using the following command :-

```
from sklearn.svm import SVC
```

    - Below snippet shows a basic way to implement the Support Vector Classifier in Python

```
train_SVC(train_X, train_y, test_X, test_Y)
```

5) *Using the Naive Bayes instance [MutinomialNB()] fit the training data(train_X and train_Y) using sklearns fit method*

6) *Perform prediction using the testing data test_X. (Use sklearns predict method)*

3) Using the result from step 2 compute the accuracy by comparing with test_y which has the class labels. This can be done using metrics module from sklearn library

❖ Logistic Regression :

- ➢ Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables.Like all regression analyses, the logistic regression is a predictive analysis. Logistic Regression is used when the dependent variable(target) is categorical.

- ➢ Logistic Regression computes models the probability that a response falls into a specific category.

- ➢ It uses a sigmoid function internally which gives output **0** and **1 for** respective class labels.

❖ Implementing Logistic Regression in Python :

- ➢ Pythons SciKit(sklearn) library has inbuilt modules for Logistic Regression model. We import it using the following command :-

  from sklearn.linear_model import LogisticRegression

- ➢ Below snippet shows a basic way to implement the Naive Bayes in Python

  Train_LogisticRegresssion(train_X, train_Y, test_X, test_Y) :

  1)Using the LogisticRegression instance [LogisticRegression()] fit the training data(train_X and train_Y) using the fit method

2)Perform prediction using the testing data test_X. (Using the predict method)
3) Using the result from step 2 compute the accuracy by comparing with test_y which has the class labels. This can be done using metrics module from sklearn library

❖ **Long term short memory Network:**

➢ It is a type of recurrent neural network. It means output of neural network is feeded back to the input after some time. In this type of network, previous information can be used for the present task.
➢ A common LSTM cell is composed of a cell, an input gate, an output gate and a forget gate. The cell remembers values over arbitrary time intervals and the three *gates* regulate the flow of information into and out of the cell. It is basically used for classifying , regression and making prediction on the time series data.

Implementing LSTM in Python :

➢ Pythons SciKit(sklearn) library has inbuilt modules for Support Vecot Classifier model. We import it using the following command :-

```
from keras.models import Sequential
from keras.layers import Dense,LSTM, Dropout
```

➢ Below snippet shows a basic way to implement the Support Vector Classifier in Python

```
train_LSTM(train_X, train_y,dev_X,dev_Y,test_X,
test_Y)
```

(1) To create a sequential mode, it uses instance of Sequential()
(2) Add method is used to add layer in the network. First layer will be LSTM which takes no of hidden units , train data as an input data, And development data as validation data in input.

(3) Using, dense it will randomly remove some feature.
(4) Next layer will be LSTM.
(5) Final Layer will be sigmoid activation function. For binary classification, ‘Sigmoid’ activation function will be used.
(6) Using evaluate method, the accuracy of model will be tested on the classification, ‘Sigmoid’ activation function will be used.

4

ANALYSIS/RESULTS

We have trained the models using the training data. Now we visualize the results and compare all the models. We also plot the learning curves to analyse the training accuracy using the development split data.

For this project we have used the following subreddits: -

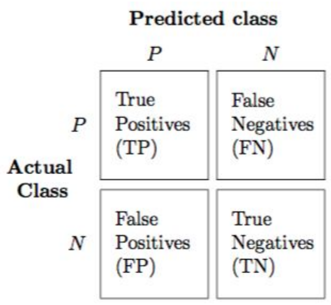
- r/soccer : A community which contains posts regarding soccer matches, news, etc all over the world.
  r/britishproblems : A community which contains mostly humorous posts regarding the problems faced by British people in their routine lives.

- r/science : A subreddit containing posts from the world of science.
  r/biology : A subreddit containing posts from the world of biology

We will compare the results of r/soccer and r/britishproblems (very different and very less correlation) and r/science and r/biology (more similarity and more correlation)

ACCURACY PRECISION AND RECALL :

Consider the following confusion matrix :



		Predicted class	
		P	*N*
Actual Class	*P*	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

Here the cell we are focusing is True Positive (TP) and True Negative(TN). TP states that the actual and predicted class is same. TN states that the prediction is correctly determined to NOT be in the actual class.

Metrics :

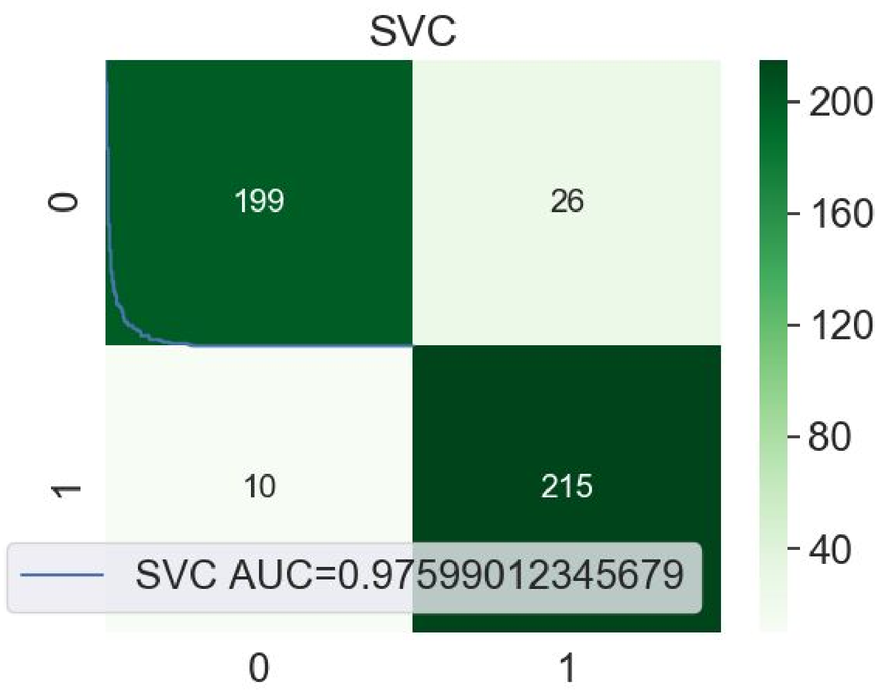
Precision : Proportion of positive identifications that was actually correct
Precision = TP/TP+FP

Recall : Proportion of actual positives that was identified correctly.
Recall = TP/TP+FN

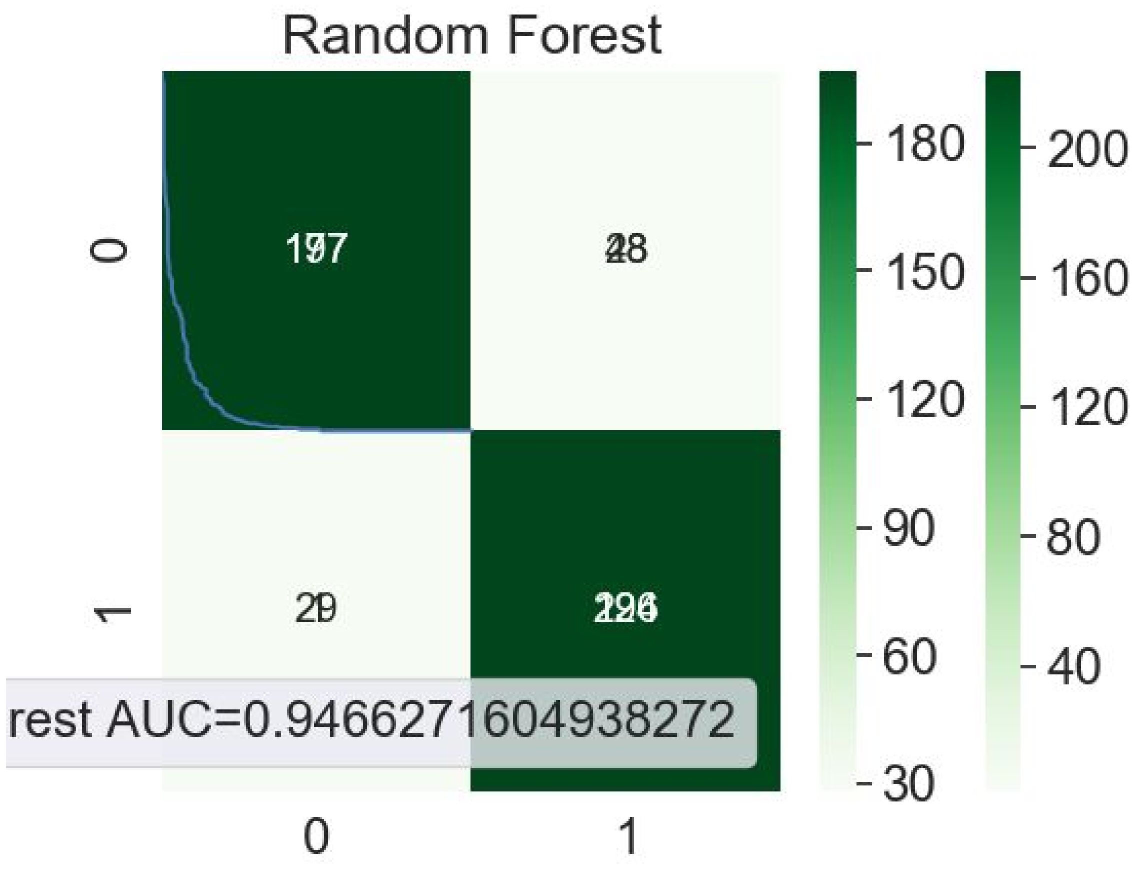
Accuracy : The fraction of predictions our model got right.
Accuracy : No of correct predictions/Total predictions = TP + TN / TP+TN+FN+FP

Confusion Matrix:

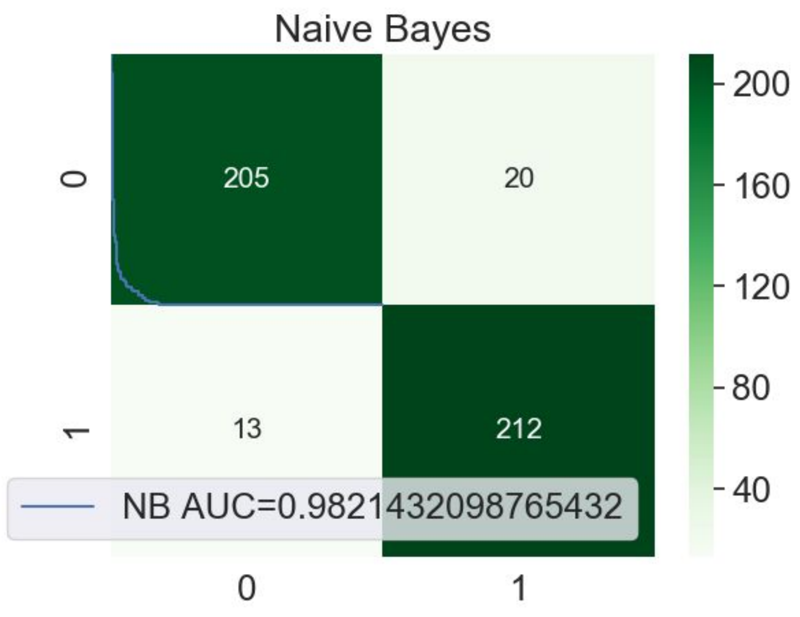
(1) SVC



(2) Random Forest:

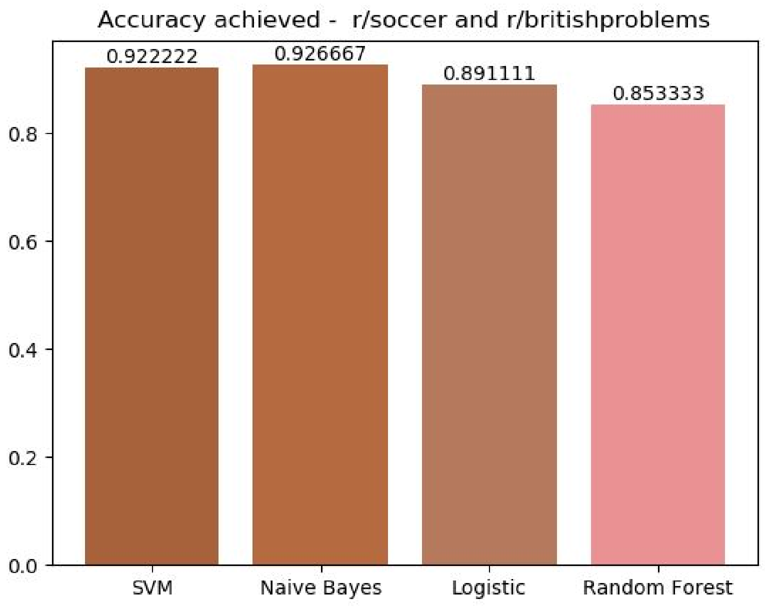


(3) Naive Bayes:

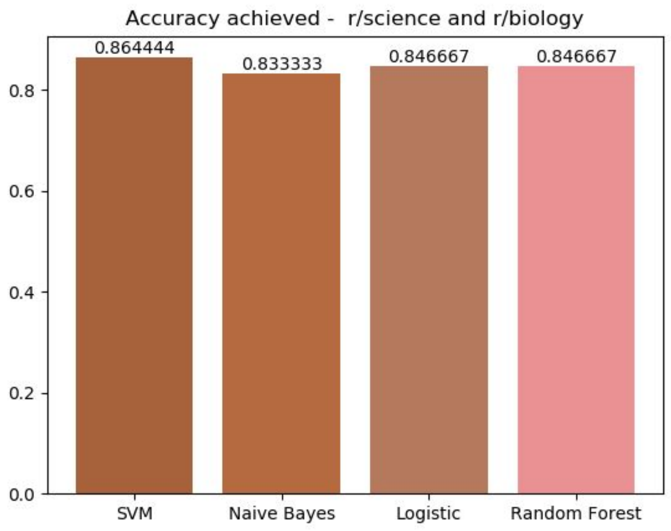


Computing accuracy for each model:

1) **r/soccer and r/britishproblems :**



2) **r/science and r/biology**

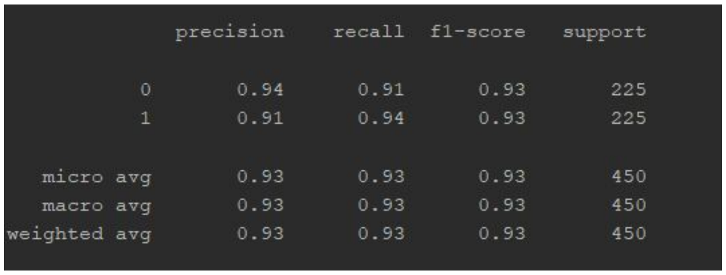


Here we see that the first pair of subreddits is different and hence we get more accuracy compared to the other pair since they are more similar.

Precision and Recall :

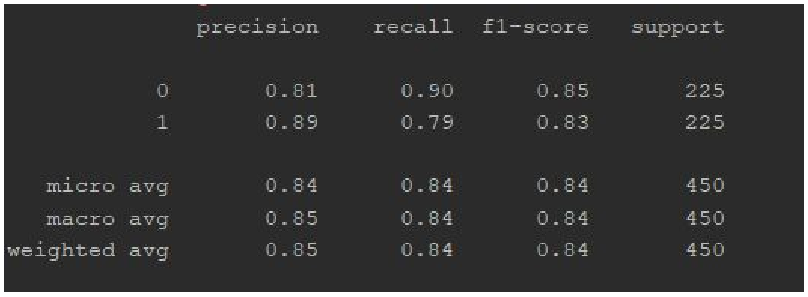
1) **r/soccer and r/britishproblems :**

- **SVC :**



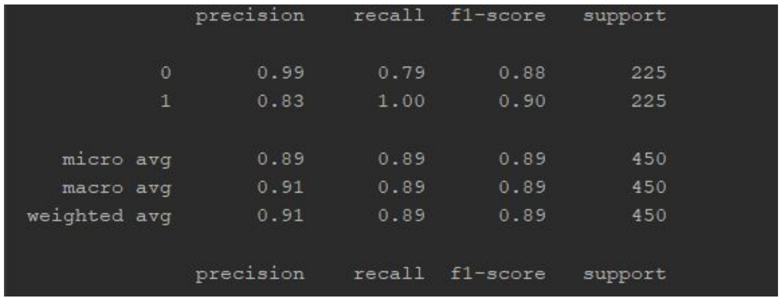
```
              precision    recall  f1-score   support

           0       0.94      0.91      0.93       225
           1       0.91      0.94      0.93       225

   micro avg       0.93      0.93      0.93       450
   macro avg       0.93      0.93      0.93       450
weighted avg       0.93      0.93      0.93       450
```

- **Naive Bayes :**



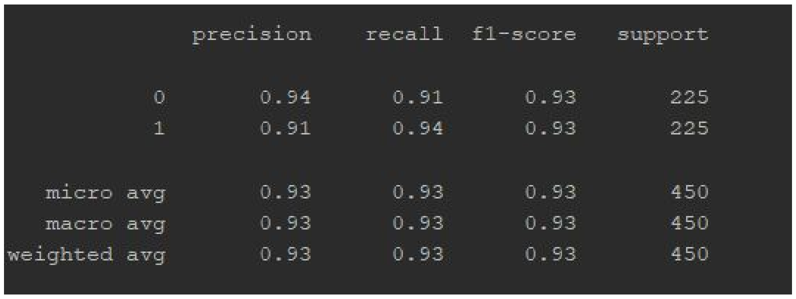
```
              precision    recall  f1-score   support

           0       0.81      0.90      0.85       225
           1       0.89      0.79      0.83       225

   micro avg       0.84      0.84      0.84       450
   macro avg       0.85      0.84      0.84       450
weighted avg       0.85      0.84      0.84       450
```

- **Logistic Regression :**



```
              precision    recall  f1-score   support

           0       0.99      0.79      0.88       225
           1       0.83      1.00      0.90       225

   micro avg       0.89      0.89      0.89       450
   macro avg       0.91      0.89      0.89       450
weighted avg       0.91      0.89      0.89       450

              precision    recall  f1-score   support
```

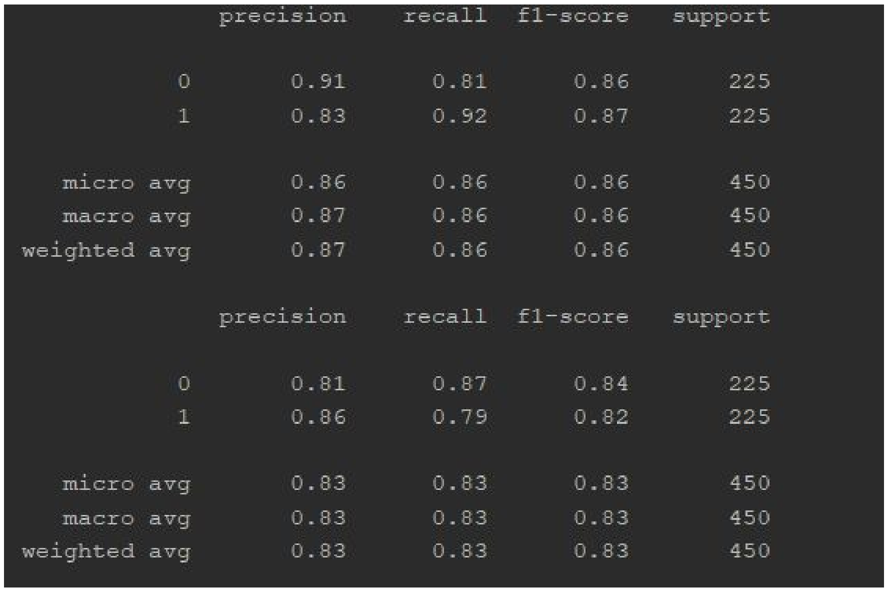
- **Random Forest :**



```
              precision    recall  f1-score   support

           0       0.94      0.91      0.93       225
           1       0.91      0.94      0.93       225

   micro avg       0.93      0.93      0.93       450
   macro avg       0.93      0.93      0.93       450
weighted avg       0.93      0.93      0.93       450
```

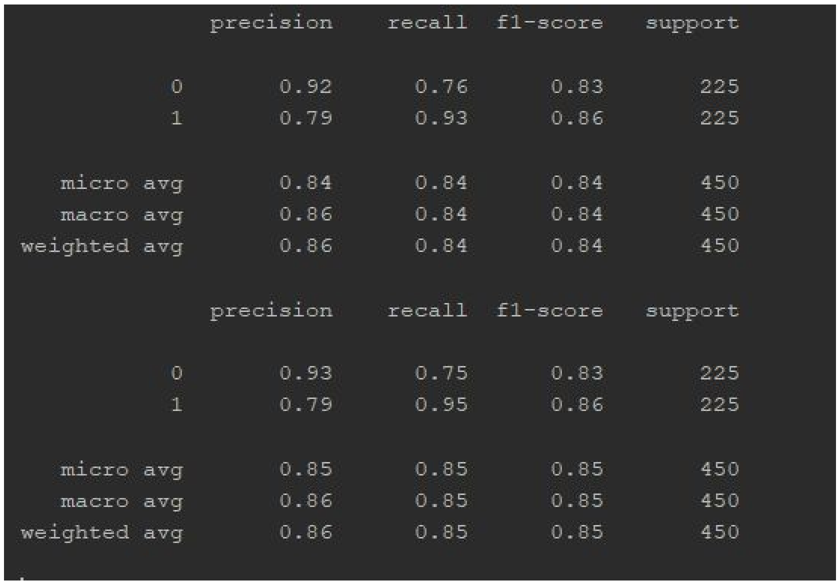
2) **r/science and r/biology :**

- **SVC and Naive Bayes (Top and Bottom)**



```
              precision    recall  f1-score   support

           0       0.91      0.81      0.86       225
           1       0.83      0.92      0.87       225

   micro avg       0.86      0.86      0.86       450
   macro avg       0.87      0.86      0.86       450
weighted avg       0.87      0.86      0.86       450

              precision    recall  f1-score   support

           0       0.81      0.87      0.84       225
           1       0.86      0.79      0.82       225

   micro avg       0.83      0.83      0.83       450
   macro avg       0.83      0.83      0.83       450
weighted avg       0.83      0.83      0.83       450
```

- **Logistic and Random Forest :(top and bottom)**



```
              precision    recall  f1-score   support

           0       0.92      0.76      0.83       225
           1       0.79      0.93      0.86       225

   micro avg       0.84      0.84      0.84       450
   macro avg       0.86      0.84      0.84       450
weighted avg       0.86      0.84      0.84       450

              precision    recall  f1-score   support

           0       0.93      0.75      0.83       225
           1       0.79      0.95      0.86       225

   micro avg       0.85      0.85      0.85       450
   macro avg       0.86      0.85      0.85       450
weighted avg       0.86      0.85      0.85       450
```

LEARNING CURVES :

- Graph that compares the performance of a model on training and testing data over a varying number of training instances
- Performance generally improves as the number of training points increases
- Learning curve allows us to verify when a model has learning as much as it can about the data. When this situation occurs the performances on the training and testing sets reach a plateau and there is a consistent gap between the two error rates

Plotting Learning Curves in Python :

- Using sklearns learning curve function we can plot the learning for the models we have trained.
- The below snippet shows how to plot learning curve in python :

  learning_curve(estimator, X, y, cv=cv, n_jobs=n_jobs, scoring =scoring train_sizes=train_sizes)

- Here estimator is our model fitted with the training data, ie, Naive Bayes, SVM etc. X and y will be our training set however we will use the development data as well the training data. N_jobs is used to specify the number of cores to be used. Cv stands for cross validation and it indicates the number of splits to be done for each training size. The scoring attribute determines the metric used. Here we will use accuracy. Training size can be a list of values such as [100,200,500,900] where each value is the training and test split ratio.

- For our project we are using 4 subreddits in a pair of two:-
  1) r/soccer and r/britishproblems : These two are very different in content.
  2) r/science and r/biology : These two have more number of common posts and are more similar.
- We will take 900 subreddits from each subreddit and plot learning curves for each pair. The split is 50%/25%/25% -> train/development/test.

- Naive Bayes Learning Curve :

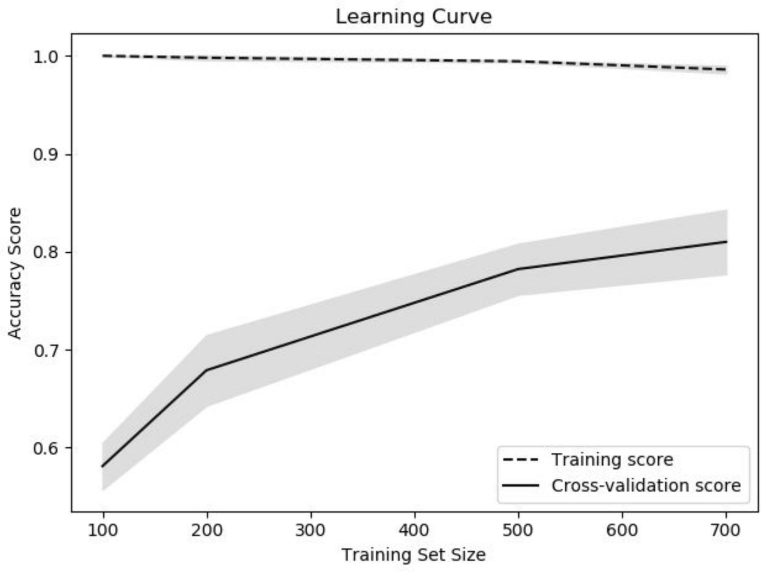


Fig. Learning curve for r/biology and r/science

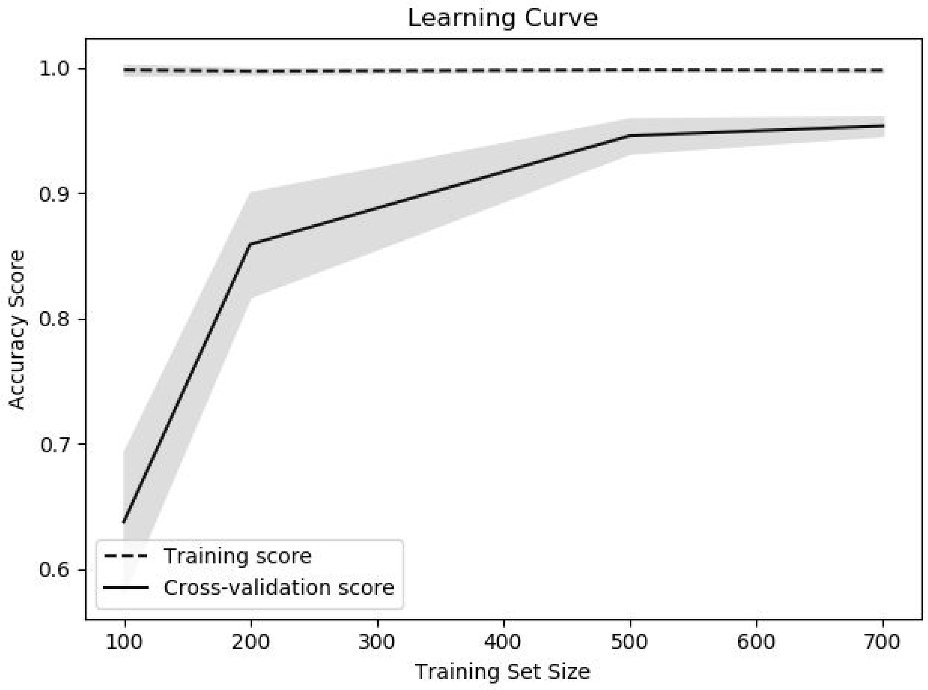
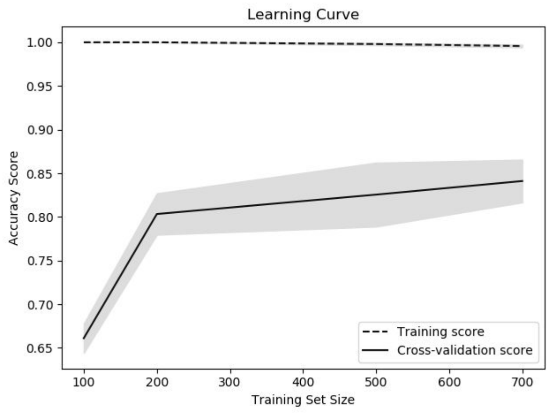


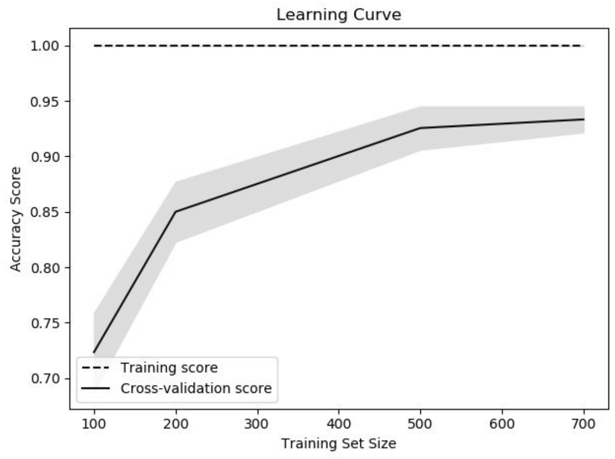
Fig. Learning curve for r/soccer and r/britishproblems

- Here we see that the training data accuracy is high as expected while the validation set accuracy gradually improves. It reached a plateau at about 84 percent accuracy. The shadow around the validation curve show the mean and variance of the accuracy.

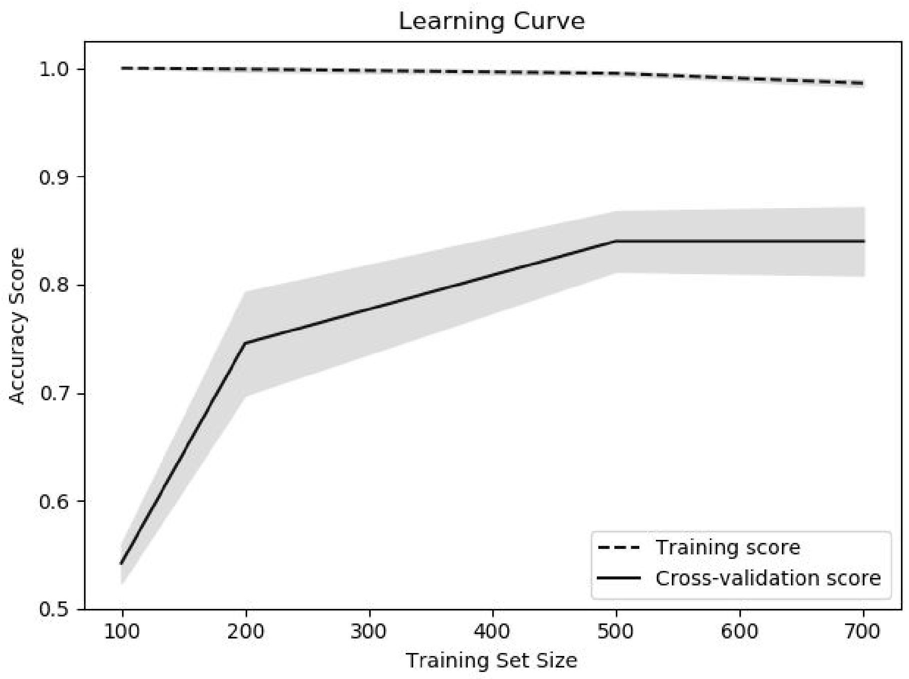
Similarly the learning curve for remaining models : (Top shows r/biology and r/science and bottom shows r/soccer and r/britishproblems

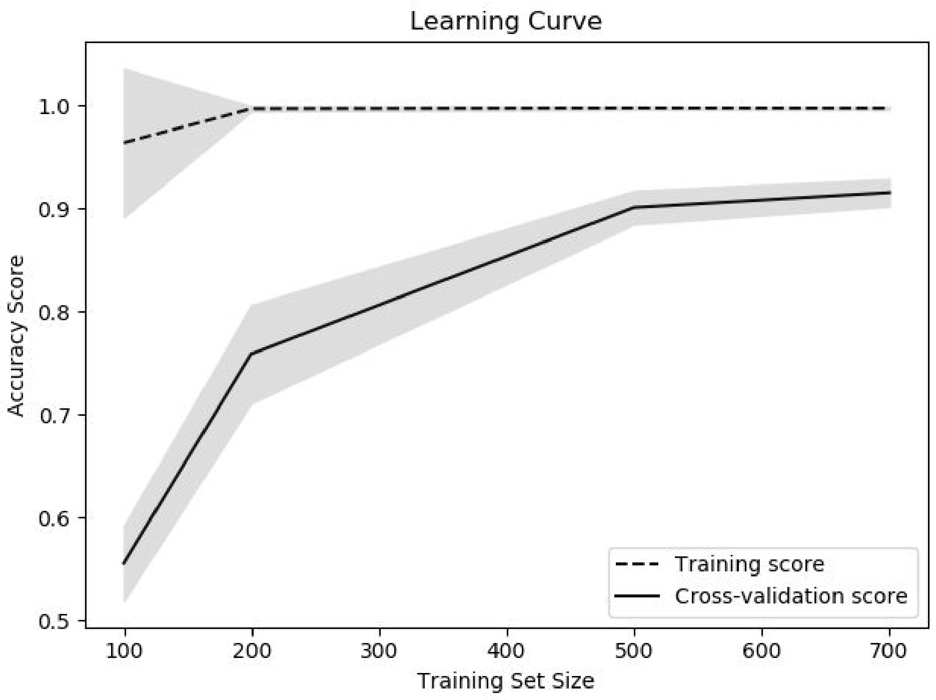
- SVM Learning Curve :



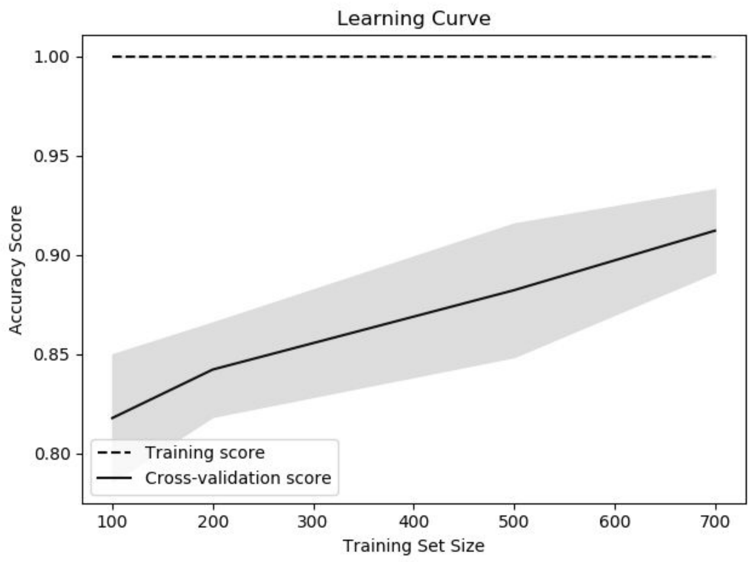


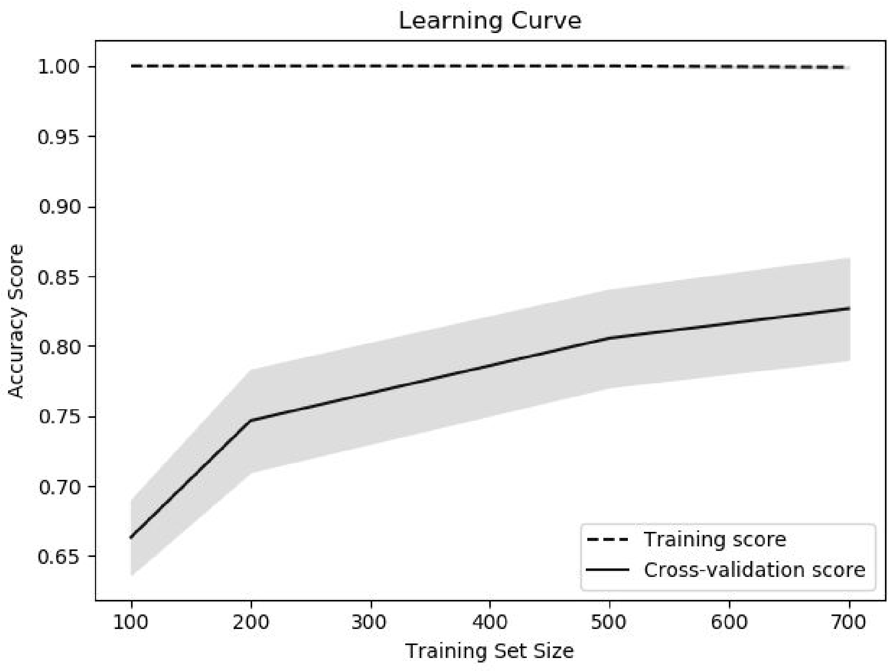
- Logistic Regression Learning Curve :





- Random Forest Learning Curve :





FURTHER VISUALIZATION : ROC Curves

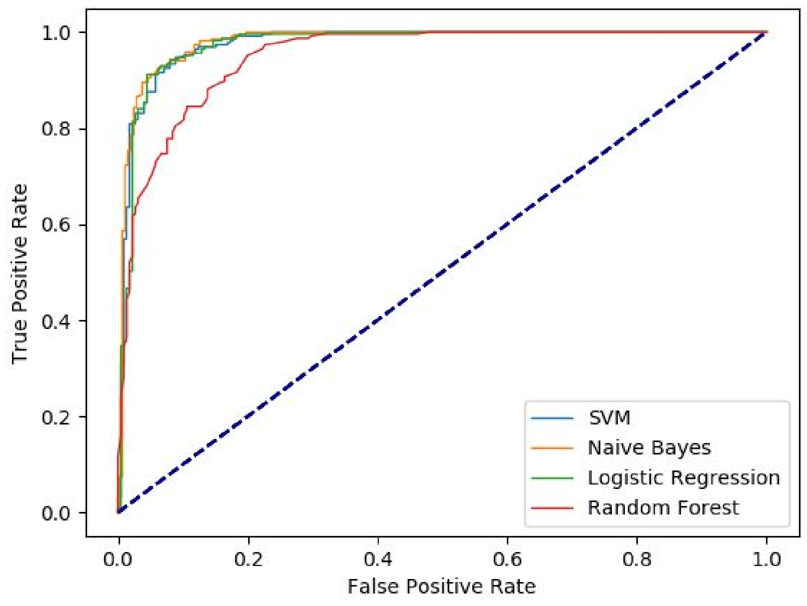
- ROC(Receiver Operating Characteristic) curves are commonly used to characterize the sensitivity/specificity tradeoffs for a binary classifier. Different threshold values give different levels of sensitivity and specificity.

- The ROC curve plots true positive rate against false positive rate.

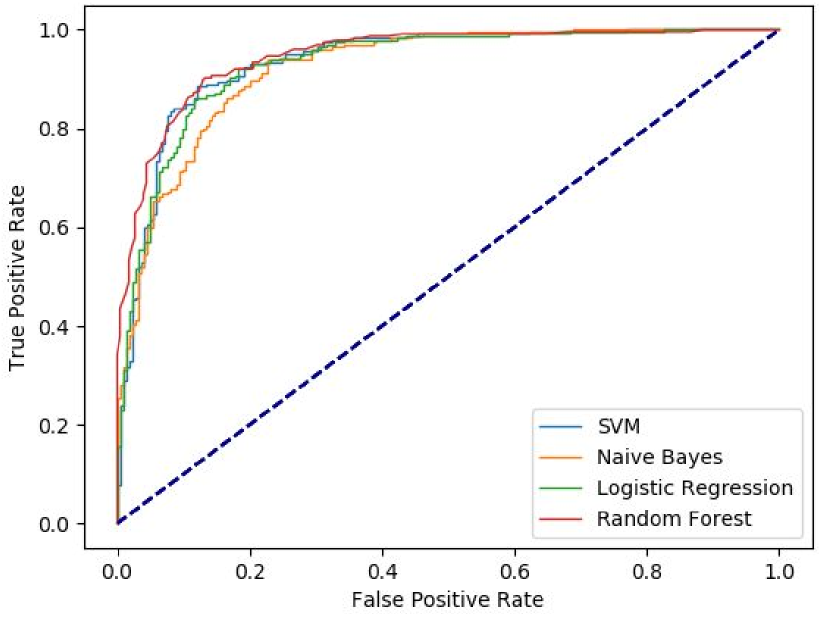
- The closer the curve follows the left-hand border and then the top border of the ROC space, the more accurate the test.

- The closer the curve comes to the 45-degree diagonal of the ROC space, the less accurate the test.

ROC Curve for r/soccer and r/britishproblems for all models



ROC Curve for r/science and r/biology for all models



4

CONCLUSION/ FUTURE EXTENSION

CONCLUSION:

- In this project, we used various python libraries to classify post according to it's subreddit. We have used praw API, sklearn library, numpy library.
- Data is fetched from the reddit using praw API and it is divided into training set, development test and test test.
- Models used for the classification are SVC, Naive Bayes, Logistic Regression, Random Forest and LSTM.
- Models performed well for binary classification rather than multiclass classification.
- More and more data can build more efficient models but may take longer to train
- Accuracy of model increases as we take more data for the training.
- Selection of features is important for the accuracy of the models.
- Metrics such as accuracy can be useful for smaller tasks but for bigger tasks such as multiclass classification we may need to consider other metrics.

FUTURE EXTENSION:

- Tune hyperparameters to increase the accuracy of the models.
- Improve feature selections and LSTM network design to increase the accuracy.
- Perform other feature extraction methods to improve accuracy.
- Design and choose models that perform well for the multiclass classification.