Assignment 3

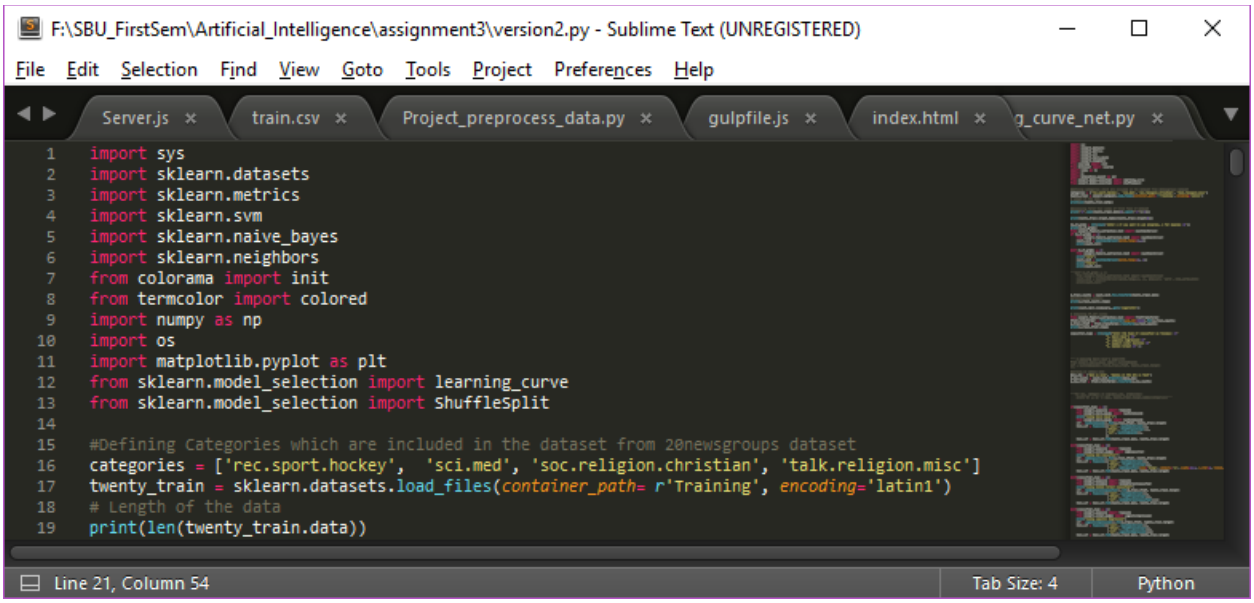
Hemant Pandey
110828730

INTRODUCTION AND COMPARISON

The purpose of this assignment is to investigate the utility of different learning algorithms for a text classification task. We have used the implementations available in the scikit python library:

http://scikit-learn.org/stable/supervised_learning.html#supervised-learning

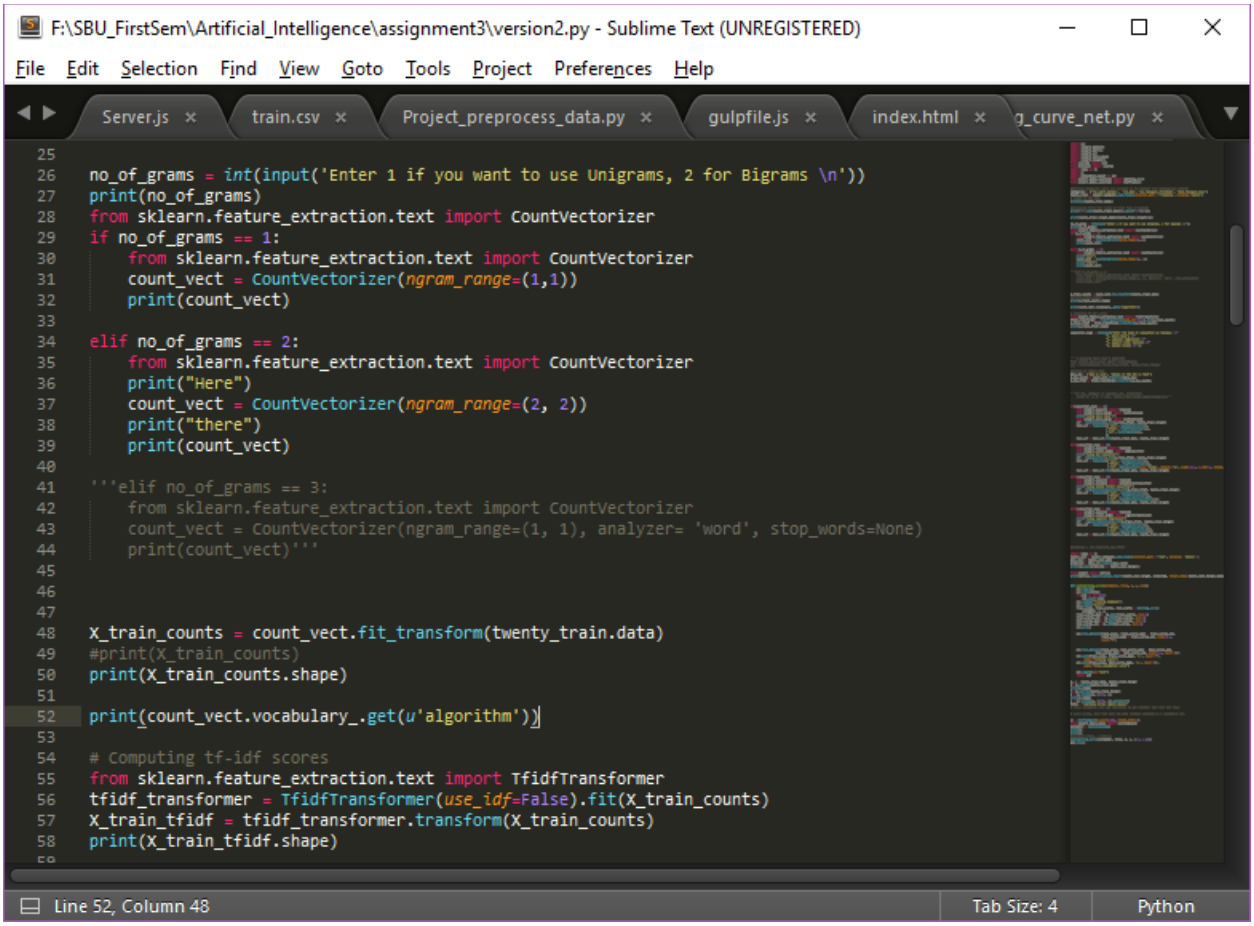
We are using the dataset from 20 newsgroup which contains up to 20 categories. Out of these 20, we have been using 4 categories which we will be using as classes. We are provided with training and test data of all the cases. Using the load_files() method of sklearn.datasets, we are loading the files.



Now, I have used the CountVectorizer() method of sklearn.feature_selection.text and set the range of n_grams to deal with unigrams and bigrams.

Then, I have used the TfidfTransformer() method of sklearn.feature_selection.text and set the desired parameters.

The code can be seen in the below screenshot :



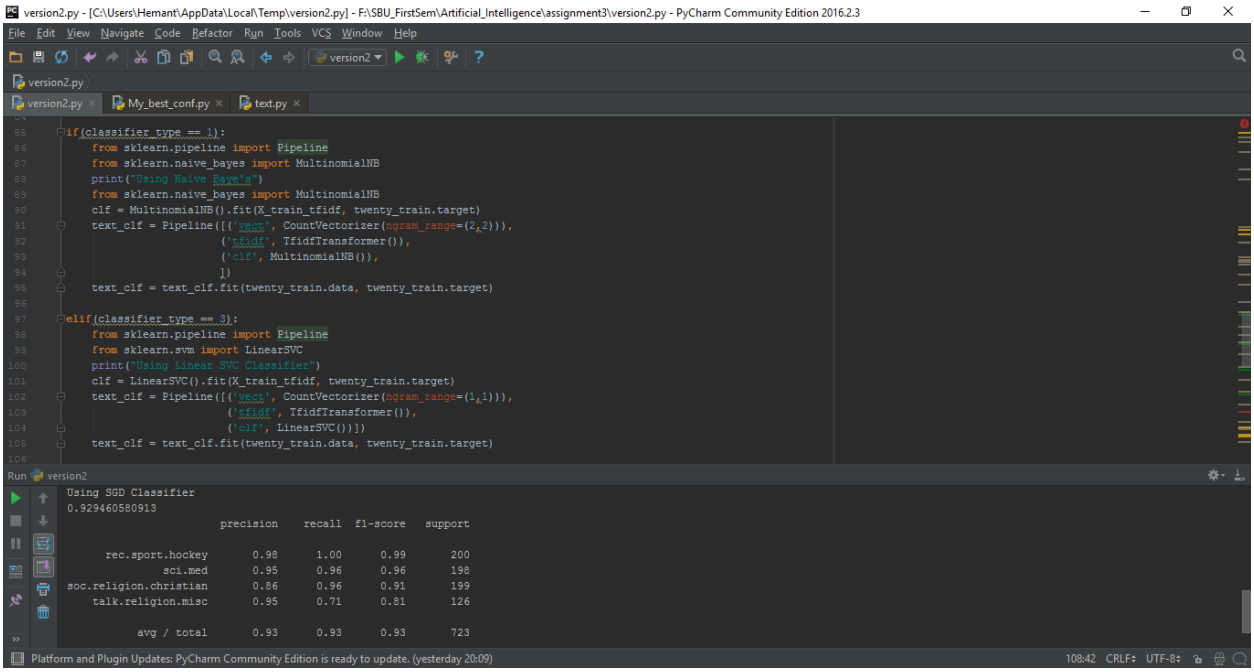
Pipeline() function from sklearn.pipeline is used for four different classifiers :

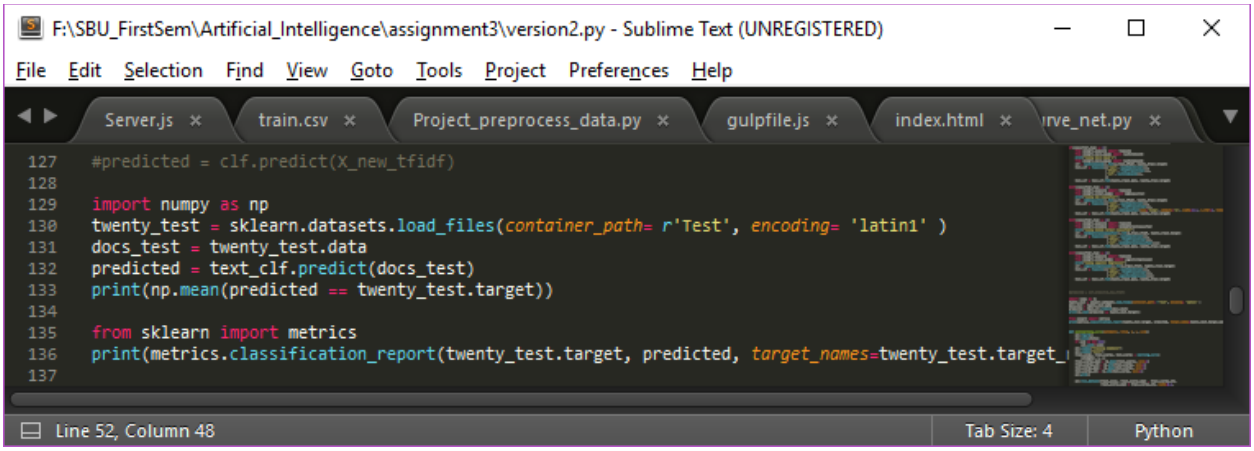
1.) Naïve Baye's
2.) Logistic Regression
3.) Stochastic Gradient Descent
4.) Random Forest

The code can be seen in the screenshot on the next page.

The test data is loaded in similar way and observations are made.

Then the precision recall metrics of each classifier using both unigram and bigram is made and following observations are made.





For Unigrams
Naïve Baye's
Mean=0.816044260028

Class	Precision	Recall	F1- Score
Rec.sport.hockey	0.98	1.00	0.99
Sci.med	0.97	0.86	0.91
Soc.religion.christian	0.61	0.99	0.76
Talk.religion.misc	1.00	0.17	0.29
Avg/Total	0.88	0.82	0.78

Logistic Regression
Mean=0.907330567082

Class	Precision	Recall	F1- Score
Rec.sport.hockey	0.98	1.00	0.99
Sci.med	0.90	0.98	0.94
Soc.religion.christian	0.83	0.97	0.89
Talk.religion.misc	0.99	0.54	0.70
Avg/Total	0.92	0.91	0.90

Support Vector
Mean=0.915629322268

Class	Precision	Recall	F1- Score
Rec.sport.hockey	0.98	1.00	0.99
Sci.med	0.95	0.96	0.96
Soc.religion.christian	0.86	0.96	0.91
Talk.religion.misc	0.95	0.71	0.81
Avg/Total	0.93	0.93	0.93

Random Forest
Mean=0.763485477178

Class	Precision	Recall	F1- Score
Rec.sport.hockey	0.83	0.94	0.88
Sci.med	0.74	0.84	0.78
Soc.religion.christian	0.73	0.78	0.75
Talk.religion.misc	0.72	0.35	0.47

Avg/Total	0.76	0.76	0.75

For Bigrams

Naïve Baye's
Mean=0.831044260028

Class	Precision	Recall	F1- Score
Rec.sport.hockey	0.96	0.98	0.97
Sci.med	0.94	0.81	0.87
Soc.religion.christian	0.66	0.99	0.79
Talk.religion.misc	0.98	0.37	0.53
Avg/Total	0.88	0.83	0.82

Logistic Regression
Mean=0.861687413555

Class	Precision	Recall	F1- Score
Rec.sport.hockey	0.92	0.98	0.95
Sci.med	0.90	0.87	0.88
Soc.religion.christian	0.76	0.97	0.85
Talk.religion.misc	0.98	0.47	0.63
Avg/Total	0.88	0.86	0.85

Support Vector
Mean=0.892116182573

Class	Precision	Recall	F1- Score
Rec.sport.hockey	0.95	0.98	0.97
Sci.med	0.92	0.89	0.91

Soc.religion.christian	0.86	0.98	0.91
Talk.religion.misc	0.95	0.71	0.81
Avg/Total	0.91	0.91	0.91

Random Forest
Mean=0.721991701245

Class	Precision	Recall	F1- Score
Rec.sport.hockey	0.71	0.88	0.78
Sci.med	0.71	0.68	0.69
Soc.religion.christian	0.71	0.87	0.78
Talk.religion.misc	0.97	0.31	0.47
Avg/Total	0.75	0.72	0.70

The plot for the four classifiers are as :

Red : Naïve Baye's

Blue : Logistic Regression

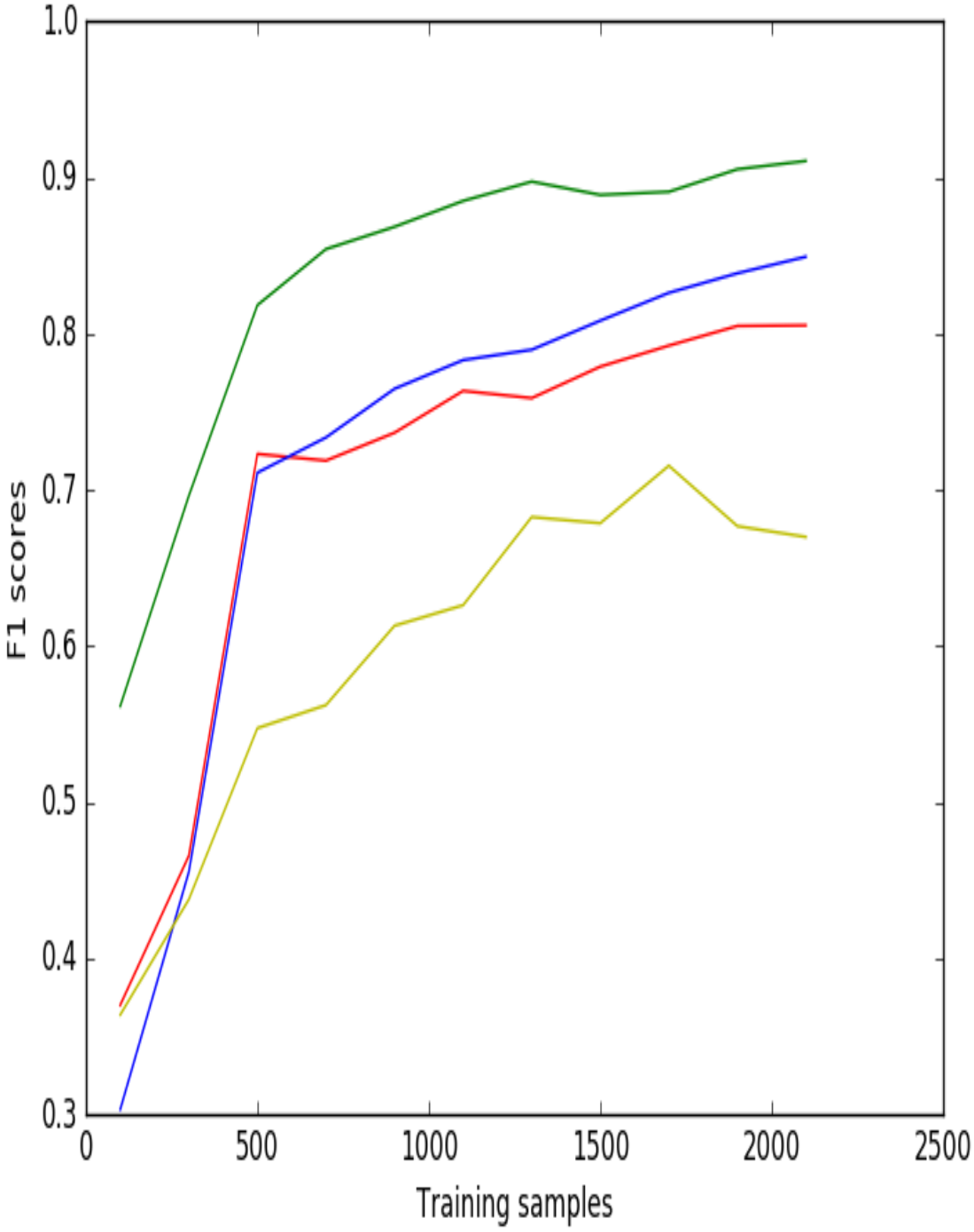
Green : Support Vector Machine

Yellow : Randomized Forest

SVM classifier is suitable in cases where there is a reasonable amount of data. Since SVMs use overfitting protection, which does not necessarily depend on the number of features, they have the potential to handle these large feature spaces.

Naive Bayes classifiers are a family of humble probabilistic classifiers based on applying Bayes' theorem with strong (naive) independence conventions between the features. Random classifier is an ensemble learning method for classification that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. **Logistic regression** is a statistical method for analyzing a dataset in which there are one or more independent variables that determine an outcome. The outcome is measured with a dichotomous variable (in which there are only two possible outcomes).

Since, we got better results in Linear SVC, I haven't extended it to radial dimensions. For a good amount of training dataset and a good number of feature spaces, SVM suits the given dataset much appropriate than the other classifier methods.


1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
F1 scores
0
500
1000
1500
2000
2500
Training samples

Since, the F1 score i.e. the harmonic mean of precision and recall is maximum i.e. 0.91 in case of Support Vector Machines with unigrams, I have used the LinearSVC classifier in my best configuration model.

I have implemented and tested the following 9 configurations

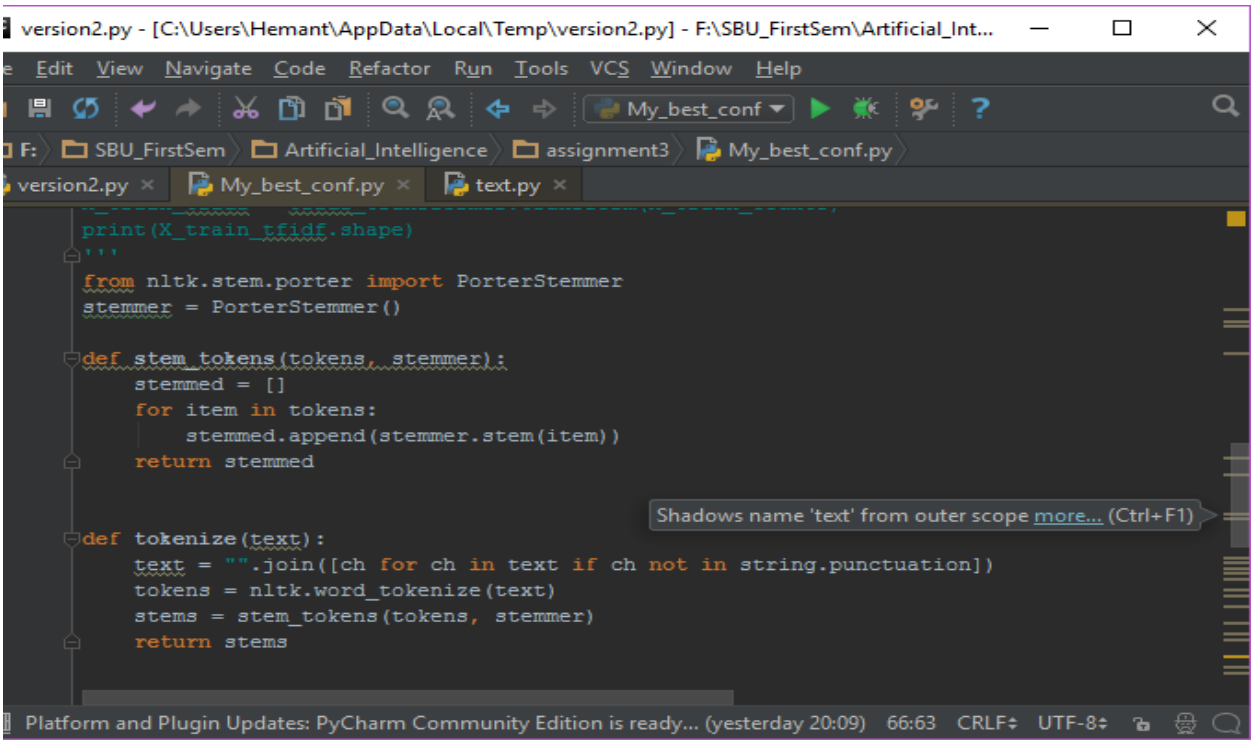
1. Removed Stop words
2. Porter Stemmed
3. Removed Stop words + Porter Stemmed
4. Neither stop words removed nor stemming done
5. L2 + Both Unigrams and Bigrams
6. Both unigrams and bigrams – L2 + Univariate Feature Selection (Chi2)
7. Porter Stemming + Stop Words + L2 + Univariate Feature Selection (Chi2)
8. Only stop words + L2

The last one has given the best results.

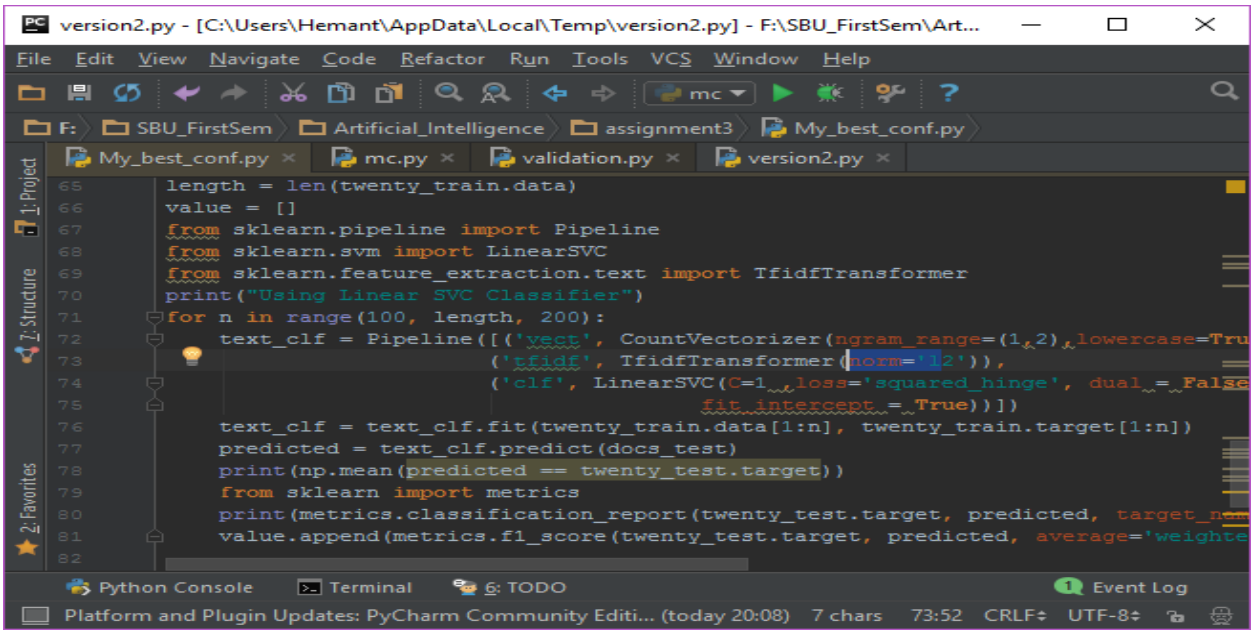
9. Adding penalty parameter + L2 + Stop Words

Following are the snapshots of code snippets for different features

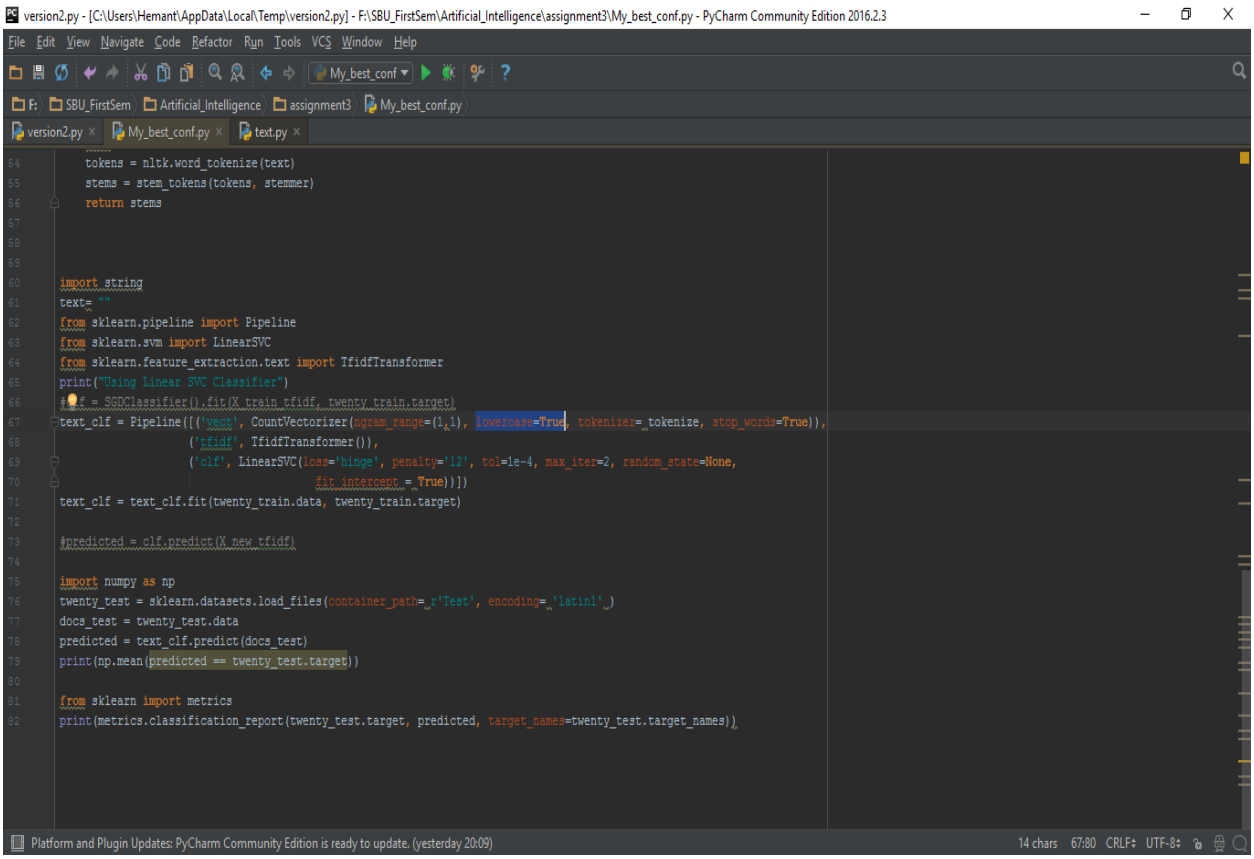
I have used the following code which builds a callable analyzer into the Count Vectorizer functions and performs stemming of words and removal of punctuation marks.



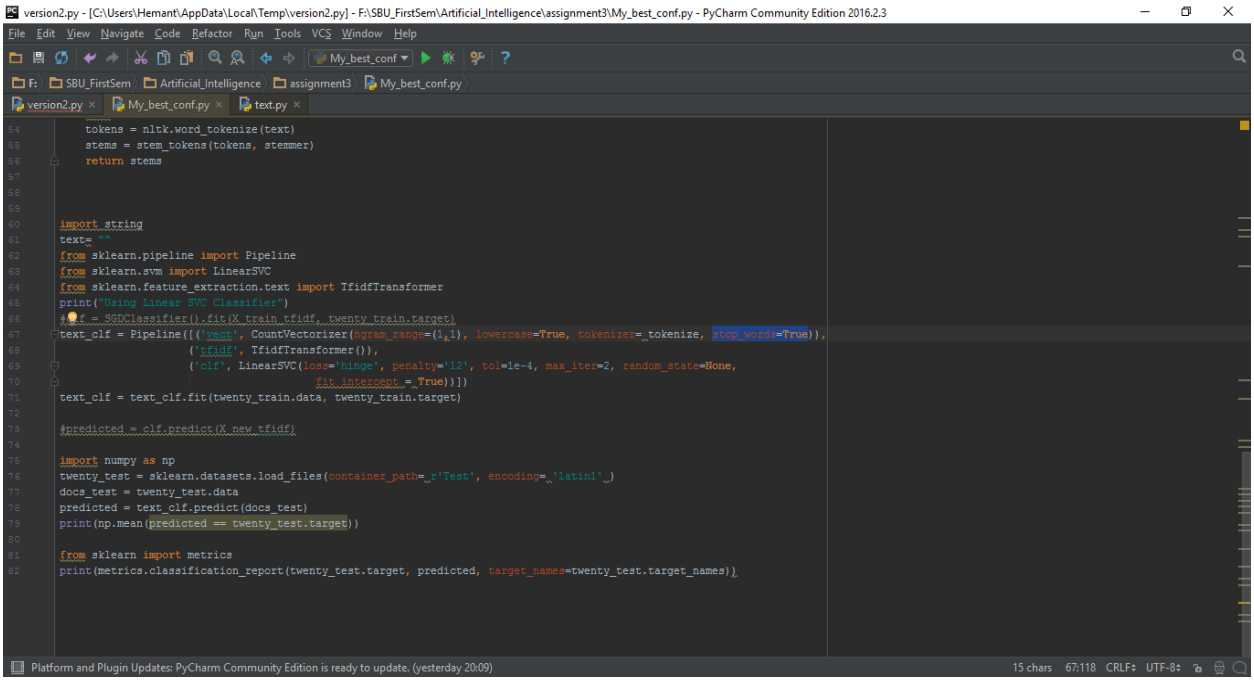
The ‘L1’ and ‘L2’ regularization constant parameter can be modified by changing the penalty parameter in the LinearSVC classifier and adding norm parameter to tfidftransformer.



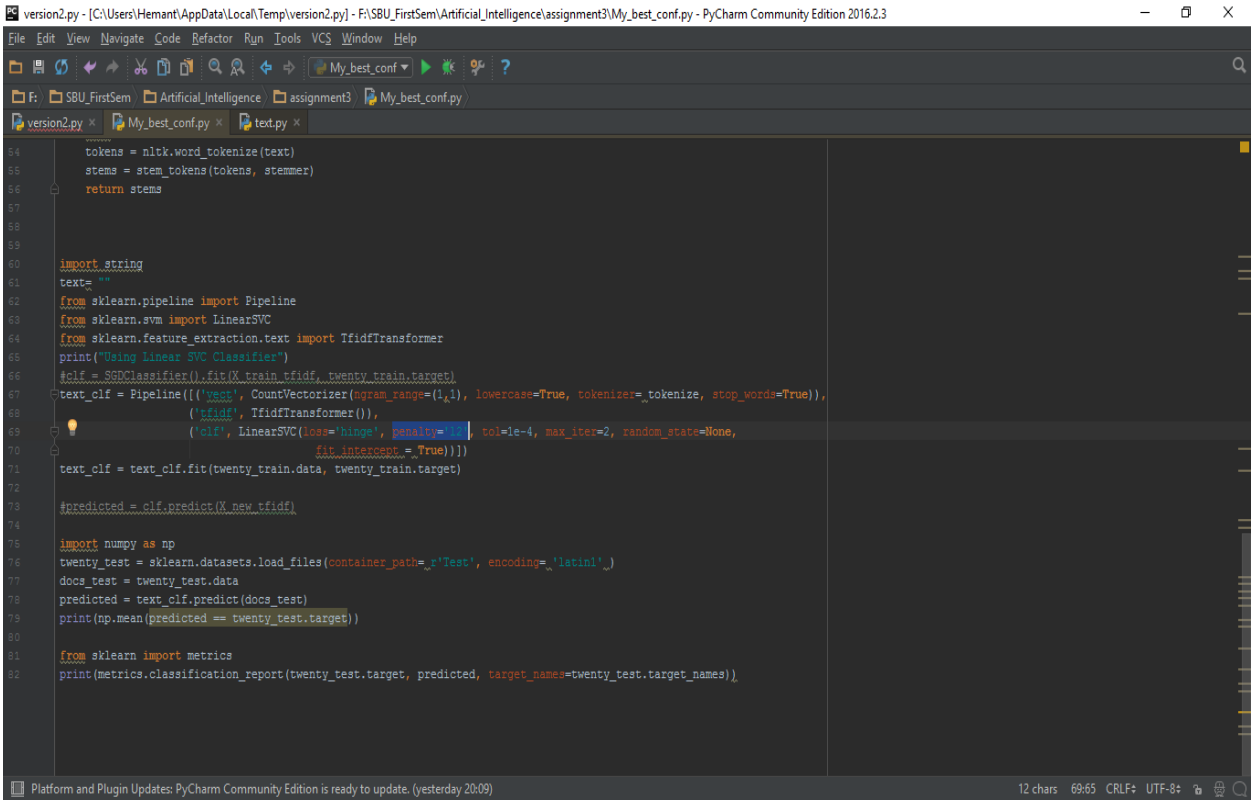
To make all the capital letters into smaller case, we make lowercase = True in CountVectorizer.



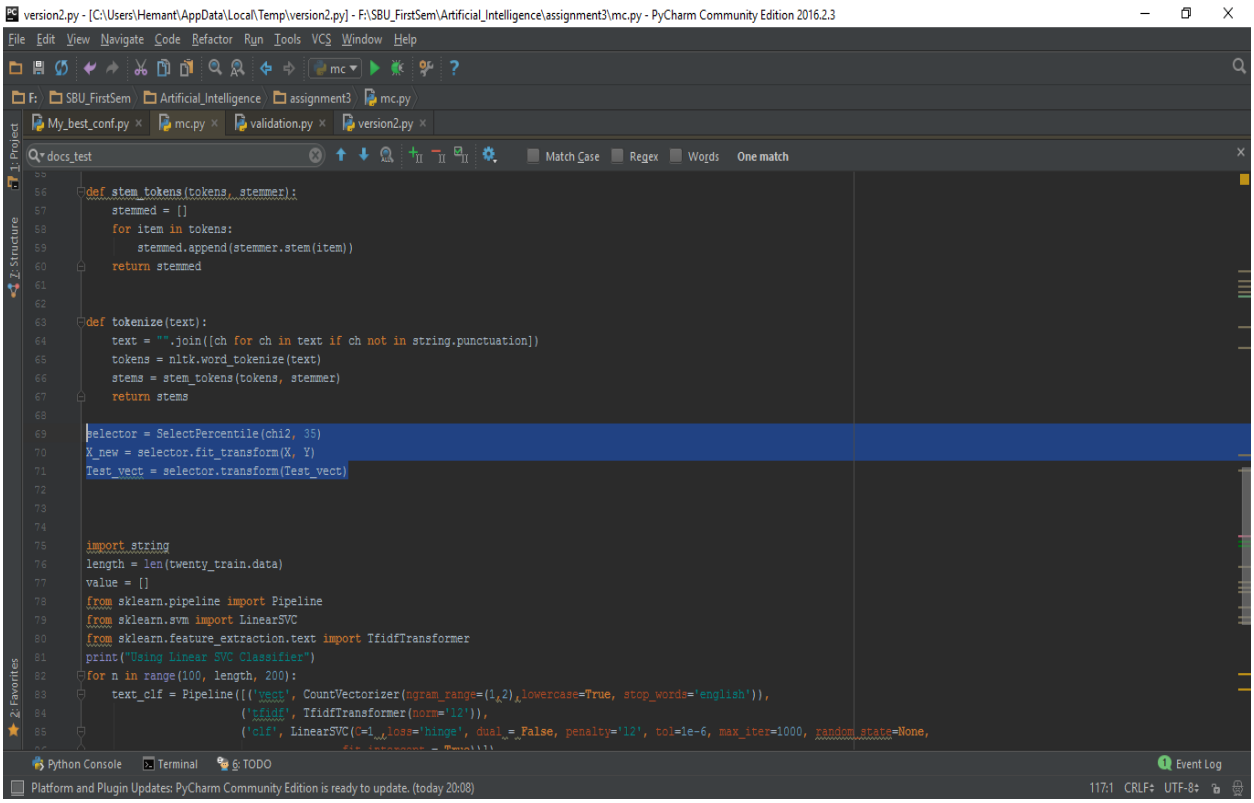
To remove all the stop words.



To change the regularization parameter,



For univariate feature selection,



Now,

1.) Removed Stop words but Porter stemmed

Mean=0.918395573997

Class	Precision	Recall	F1- Score
Rec.sport.hockey	0.98	0.99	0.99
Sci.med	0.95	0.96	0.96
Soc.religion.christian	0.83	0.95	0.89
Talk.religion.misc	0.92	0.67	0.78
Avg/Total	0.92	0.92	0.92

2.) Both Porter Stemmed and Stop words

Mean=0. 923928077455

Class	Precision	Recall	F1- Score
Rec.sport.hockey	0.98	0.99	0.98
Sci.med	0.96	0.97	0.96
Soc.religion.christian	0.85	0.95	0.90
Talk.religion.misc	0.93	0.71	0.80
Avg/Total	0.93	0.92	0.92

3.) Removed both stop words and porter stemmed

Mean=0.922544951591

Class	Precision	Recall	F1- Score
Rec.sport.hockey	0.98	1.00	0.99
Sci.med	0.95	0.96	0.96
Soc.religion.christian	0.84	0.96	0.89
Talk.religion.misc	0.94	0.67	0.79
Avg/Total	0.93	0.92	0.92

4.) No Stemming but stop words removed

Mean=0.929460580913

Class	Precision	Recall	F1- Score
Rec.sport.hockey	0.98	1.00	0.99
Sci.med	0.95	0.96	0.96
Soc.religion.christian	0.86	0.95	0.90

Talk.religion.misc	0.92	0.74	0.82
Avg/Total	0.93	0.93	0.93

So the best results are obtained in case 4 when stop words were removed but porter stemmer was not used. Now, I am using the same arguments for count vectorizer and will be manipulating the parameters of LinearSVC to obtain much better results.

5.) Taking both unigrams and bigrams and applying L2 Regularization

Mean=0.9377593361

Class	Precision	Recall	F1- Score
Rec.sport.hockey	0.99	1.00	0.99
Sci.med	0.95	0.97	0.96
Soc.religion.christian	0.88	0.96	0.92
Talk.religion.misc	0.95	0.75	0.84
Avg/Total	0.94	0.94	0.94

6.) Taking both unigrams and bigrams without L2 Regularization and using Univariate Feature Selection (Chi2)

Mean=0.914246196404

Class	Precision	Recall	F1- Score
Rec.sport.hockey	1.00	1.00	1.00
Sci.med	0.91	0.96	0.93
Soc.religion.christian	0.87	0.95	0.91
Talk.religion.misc	0.86	0.64	0.74
Avg/Total	0.91	0.91	0.91

7.) Porter Stemming + Stop Words removed + L2 + Univariate Feature Selection (Chi2)

Mean=0.934993084371

Class	Precision	Recall	F1- Score
Rec.sport.hockey	0.99	1.00	1.00
Sci.med	0.96	0.99	0.98
Soc.religion.christian	0.87	0.95	0.91
Talk.religion.misc	0.92	0.71	0.80
Avg/Total	0.94	0.93	0.93

8.) Only Stop words removed and L2 regularization applied

Mean=0.9377593361

Class	Precision	Recall	F1- Score
Rec.sport.hockey	0.99	1.00	0.99
Sci.med	0.95	0.97	0.96
Soc.religion.christian	0.88	0.96	0.92
Talk.religion.misc	0.95	0.75	0.84
Avg/Total	0.94	0.94	0.94

9.) Adding penalty parameter + L2 Regularization + Only Stop words

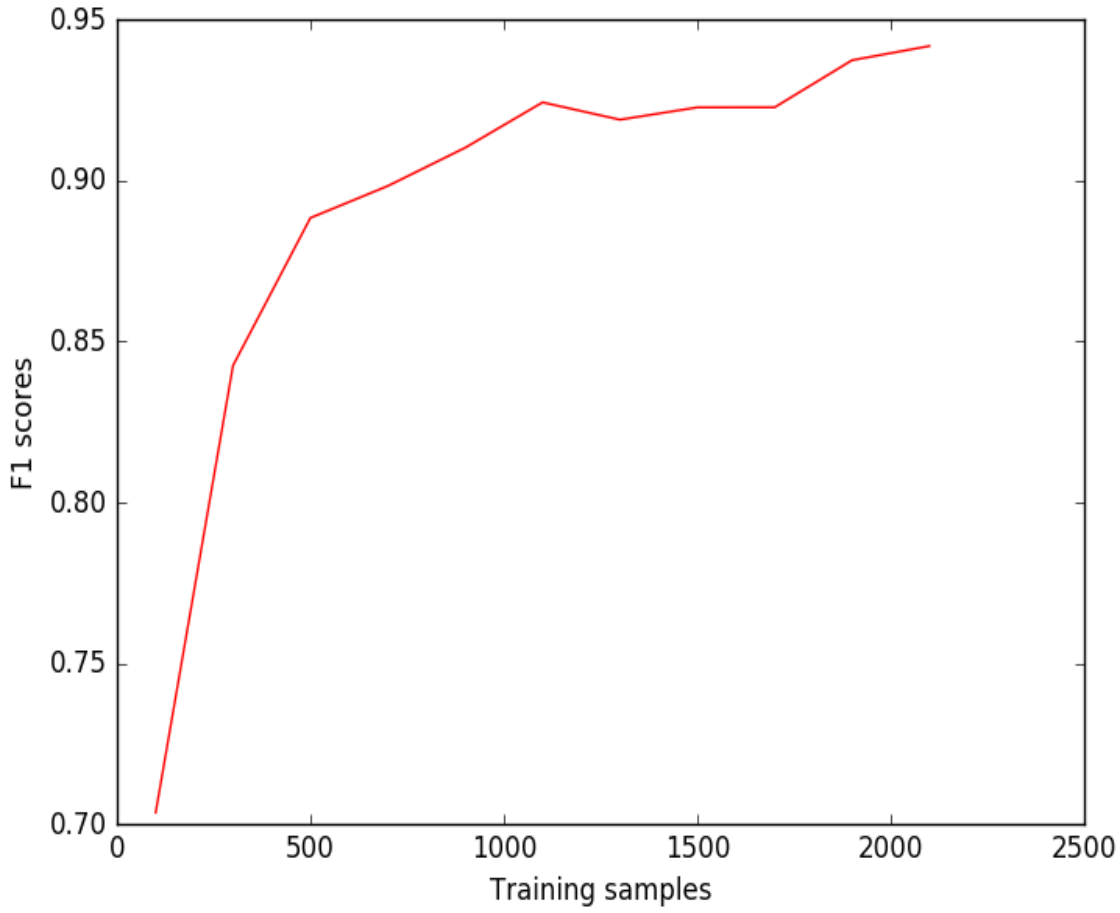
Mean=0.943291839557

Class	Precision	Recall	F1- Score
Rec.sport.hockey	0.98	1.00	0.99
Sci.med	0.96	0.97	0.96

Soc.religion.christian	0.89	0.97	0.93
Talk.religion.misc	0.96	0.77	0.85
Avg/Total	0.95	0.94	0.94

Thus, observing all the possible configurations made the best results were obtained with stop words removal and L2 regularization i.e. F1 score **0.94 and** precision **0.95**.

The learning curve is as below:



The changes made to my best configuration are :

```
text_clf = Pipeline([('vect', CountVectorizer(ngram_range=(1,2),lowercase=True,
stop_words='english')),
                     ('tfidf', TfidfTransformer(norm='l2')),
                     ('clf', LinearSVC(C=1 ,loss='squared_hinge', dual = False,
penalty='l2', tol=1e-6, max_iter=1000, random_state=None,
                                       fit_intercept = True))])
```

Stop words have been removed.
Using PorterStemmer didn't improved my result.
Adding L2 Regularization Parameter to TfidfTransformer.
Adding penalty parameter improved my result well.
Changing the ngram_range to (1,2)

1.) Parameters of CountVectorizer()
ngram_range defines the range of grams:
(1,1) is used for Unigrams
(2,2) is used for Bigrams
(1,2) is used for both unigrams as well as bigrams

Lowercase = True:
This converts all capital letters to lower case letter.

Stop_words = 'english':
This removes all the stop words.

2.) Parameters for TfidfTransformer()
Norm used to normalize term vectors. None for no normalization.
We have used L2.

3.) Parameters for LinearSVC()
Loss : Specifies the loss function. 'hinge' is the standard SVM loss (used e.g. by the SVC class) while 'squared_hinge' is the square of the hinge loss. We have used L2.

Dual: Select the algorithm to either solve the dual or primal optimization problem. Prefer dual=False when n_samples > n_features.

C : Penalty parameter C of the error term

Tol: Tolerance for stopping criteria.

Max_iter: The maximum number of iterations to be run.

Random_state : The seed of the pseudo random number generator to use when shuffling the data.

My code will run for the datasets at particular intervals, show the report at every step and then finally displays the precision recall table for complete dataset and plot the learning curve.

References

1.) http://scikit-learn.org/stable/supervised_learning.html#supervised-learning Sklearn Documentation.

2.) http://scikit-learn.org/stable/tutorial/text_analytics/working_with_text_data.html Sklearn Text Documentation