ONLINE NEWS POPULARITY ANALYSIS

EE 559 PROJECT

APRIL 30, 2018

TAMOGHNA CHATTOPADHYAY
tchattop@usc.edu

Abstract

The Online News Popularity dataset provides several attributes from different articles published by Mashable. Our goal with this dataset was to predict if an article is popular or not. The labels (popular/not popular) are based on the number of shares the article has received. The dataset contained 60 features which were a mixture of different types. The problem was divided into two parts: a two class problem and a multi class problem. For the two class problem, we defined values in the shares column > median as 'popular' and those below the median as 'not popular'. For the multiclass problem, a set of conditions was used on the shares column to decide different levels of popularity. The dataset was divided into two sets – training and testing – with 90% of the data was kept in training data and 10% in test data. Before the training data was every given to the model the data needed to be cleaned and pre-processed and various measures such as treatment of missing values, outlier detection, and removal, categorical data merging, feature dimensionality expansion or reduction and normalization were used. The pre-processed data was then sent to various types of classifiers to train them for the incoming testing dataset. The results were compared on the basis of Classification accuracy and F1 score. Also, the confusion matrix was shown for each classifier result. Based on the F1 scores, the KNN Classifier gave the best result whereas the Perceptron Classifier gave the worst result. The test accuracy for all the classifiers was around 0.55-0.65.

Task and Methodology:

Task 1 - The task of this project is to find a model that can classify new articles into class popular or unpopular (divided by 'shares' = median value) using the input features.

Task 2 - The task of this project is to find a model that can classify new articles into various classes of popularity (divided by conditions on column 'shares') using the input features.

Task1:

Pre-Processing Steps:

Among the features given to us are:

- **58 predictive input attributes** for example count of stop words, length of stop word, number of links, images and video on the webpage.
- **2 non-predictive features** which are webpage URL and publish time;
- **1 target feature** namely the shares

So while reading the csv file, we can drop the columns having URL and Publish time, i.e., timedelta from the dataframe. Then, we found out the median value in the shares column which was 1400. Thus, we can use this value to label the shares. If value of shares is greater than 1400, we set it as popular and otherwise we set it as unpopular. Here, popular is characterized by 1 and unpopular is characterized by 0.

After dropping the two input features, there are 58 input features remaining.
Data types: All the input features have numerical values, but by counting the unique values, we can tell some of them are actually binary. And some of them are related in content. So we can merge them into multinomial features encoded by integers to reduce dimension.
Feature scales: The range of these features are quite different. Standardization/Normalization are needed before fitting models on them.
Outliers: If we define values greater than 5 standard from mean as outliers, some features have considerable amount of outliers thus outlier removal is needed as well in preprocessing.

We find that the features :

'data_channel_is_lifestyle', 'data_channel_is_entertainment', 'data_channel_is_bus', 'data_channel_is_socmed', 'data_channel_is_tech', 'data_channel_is_world'

Can all be combined into one feature encoded by integers.

Similarly,

'weekday_is_monday','weekday_is_tuesday','weekday_is_wednesday','weekday_is_thursda y','weekday_is_friday','weekday_is_saturday', 'weekday_is_sunday'

Can all be combined into one feature encoded by integers.
This was done to reduce feature dimension. After this step, outlier removal was a necessary step. An outlier is a data point that statistically inconsistent with rest of the data set. Outlier removal should be done before standardization. We define any value greater than 5 standard from the mean as an outlier value. Once an outlier value was found in a feature, the data point was

removed. The outliers are only removed from features having continuous values as categorical features have been converted to indicator feature space. To standardize the features, the min-max scaler was applied and then the data points were centered by subtracting the mean value. Standardization is done so that the entire dataset is within a defined range. Un-normalized data was also run on various classifiers and it was found that the classifiers were very sensitive to vast differences in the data set values or skewed data and hence needed to be normalized.

The class labels were divided into 0 and 1 according to the median = 1400 as discussed before. Then after all the preprocessing steps were done, the data was divided into two sets: training data and testing data with 90% of the data in the former and the rest in the latter.

After that I tried feature reduction using PCA (Principal Component Analysis). The PCA was applied to both the training and testing data.

Classifiers used:

I used four different classifiers to fit to the training data and to analyze which one is the best for this data. The four classifiers used are:

- SVM Classifier
- Naïve Bayes Classifier
- KNN Classifier
- Perceptron Classifier

The first classifier which I implemented was SVM classifier. The advantages of support vector machines are:

- Effective in high dimensional spaces.
- Still effective in cases where number of dimensions is greater than the number of samples.
- Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
- Versatile: different Kernel functions can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels.

The disadvantages of support vector machines include:

- If the number of features is much greater than the number of samples, avoid over-fitting in choosing Kernel functions and regularization term is crucial.
- SVMs do not directly provide probability estimates, these are calculated using an expensive five-fold cross-validation.

I used the rbf kernel for the SVM Classifier. Then I applied the classifier for various values of gamma and C. Then selecting the best pair from those I applied it to the test data to find the accuracy.

The second classifier which I implemented was Naïve Bayes Classifier. I used the Gaussian Naïve Bayes Classifier. They are a set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of independence between every pair of features. Naive Bayes learners and classifiers can be extremely fast compared to more sophisticated methods. They require a small amount of training data to estimate the necessary parameters. In Gaussian Naïve Bayes Classifier, the likelihood of the features is assumed to be Gaussian. I applied the Gaussian Naïve Bayes and found the accuracy on the test dataset.

The third classifier which I implemented was KNN Classifier. The principle behind nearest neighbor methods is to find a predefined number of training samples closest in distance to the new point, and predict the label from these. The number of samples is a user-defined constant as seen in k-nearest neighbor learning. The distance can, in general, be any metric measure: standard Euclidean distance is the most common choice. Being a non-parametric method, it is often successful in classification situations where the decision boundary is very irregular

KNeighborsClassifier implements learning based on the k nearest neighbors of each query point, where k is an integer value specified by the user. he optimal choice of the value k is highly data-dependent: in general a larger k suppresses the effects of noise, but makes the classification boundaries less distinct.

The fourth classifier which I implemented was Perceptron. The Perceptron is another simple algorithm suitable for large scale learning. By default:

- It does not require a learning rate.
- It is not regularized (penalized).
- It updates its model only on mistakes.

I set the Perceptron classifier for 1000 iterations and a tolerance value of 0.0001 and fit it to the dataset. Then I found out the test accuracy.

This was a basic description of the four classifiers which I used.

Performance Evaluation Techniques used:

The performance evaluation was done on the testing data set that which would go through the same pre-processing steps as the training data did. The testing data was then fed into different classification models and predicted labels were obtained from testing the test data on the trained classifiers. The test accuracies were found from this data. Also, F1 score and confusion matrix was found for the test dataset. A confusion matrix is a table that is used to describe the performance of classification model on a set of testing data set whose true label test is known. he F1 score can be interpreted as a weighted average of the precision and recall, where an F1 score

reaches its best value at 1 and worst score at 0. The relative contribution of precision and recall to the F1 score are equal. The classification performance of each classifier is shown below.

Results for Task1:

For SVM Classification,

The best pair is C = 0.464158883361 and gamma = 0.464158883361

The mean Cross-Validation Accuracy for the best pair = 0.662072403034

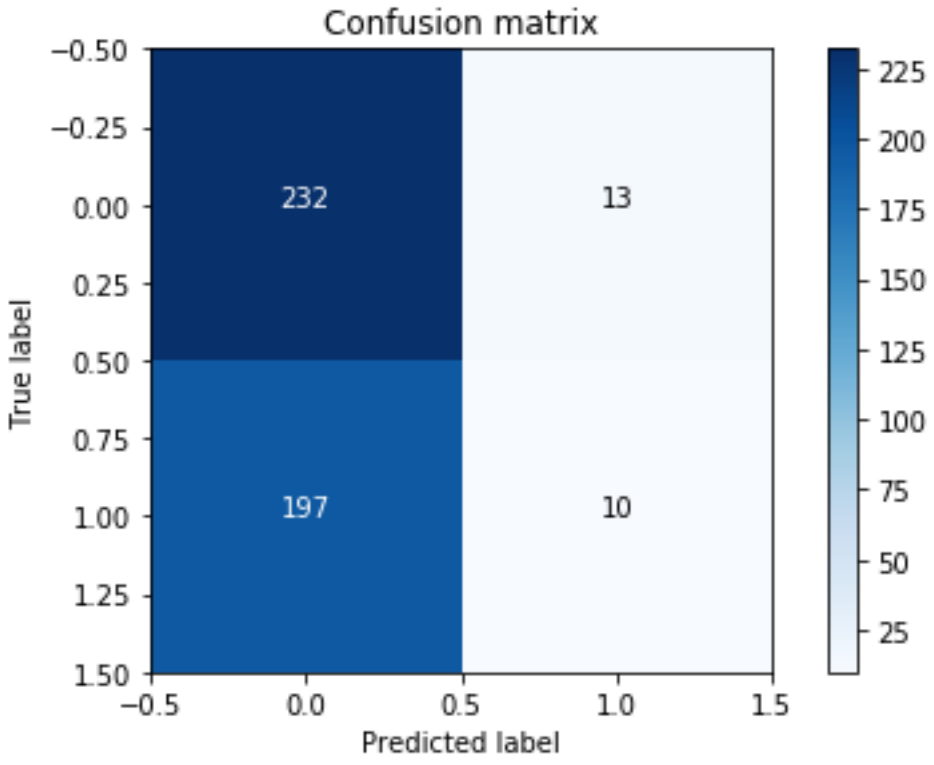
The Standard deviation for the best pair = 0.0103258692375

Test accuracy = 0.637168141593

F1 Score = 0.616885801703

Confusion Matrix = [[196 49], [115 92]]

Confusion matrix, without normalization:



For Naïve Bayes Classifier,

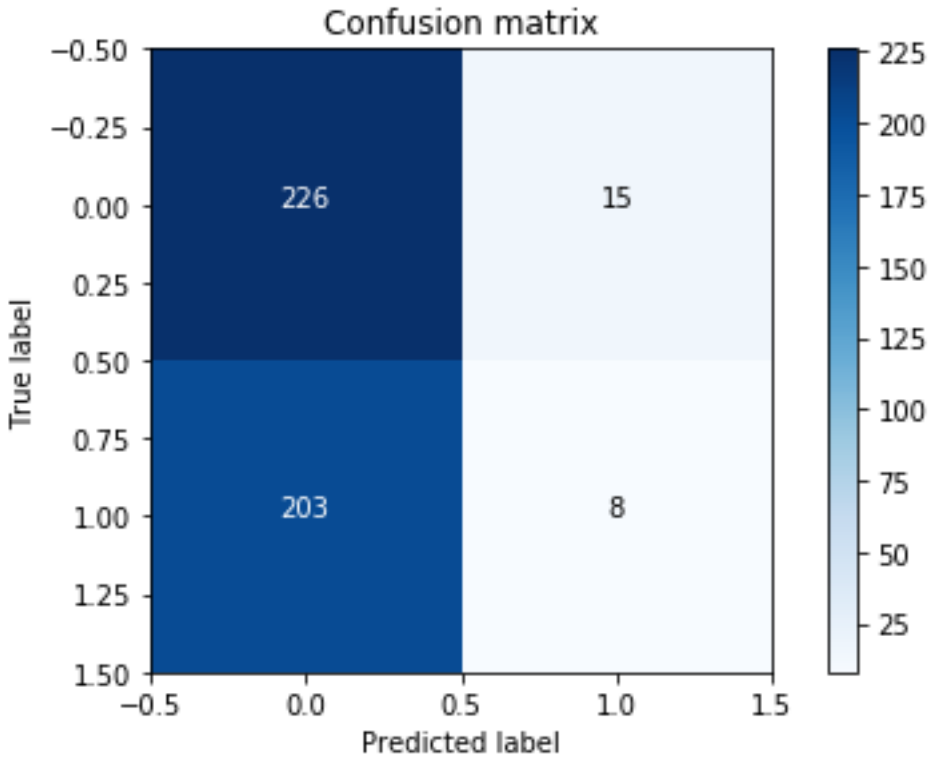
The best Cross-Validation Accuracy is = 0.635467980296

Test accuracy = 0.621681415929

F1 Score = 0.604957706166

Confusion Matrix = [[187 54], [117 94]]

Confusion matrix, without normalization:



For KNN Classifier,

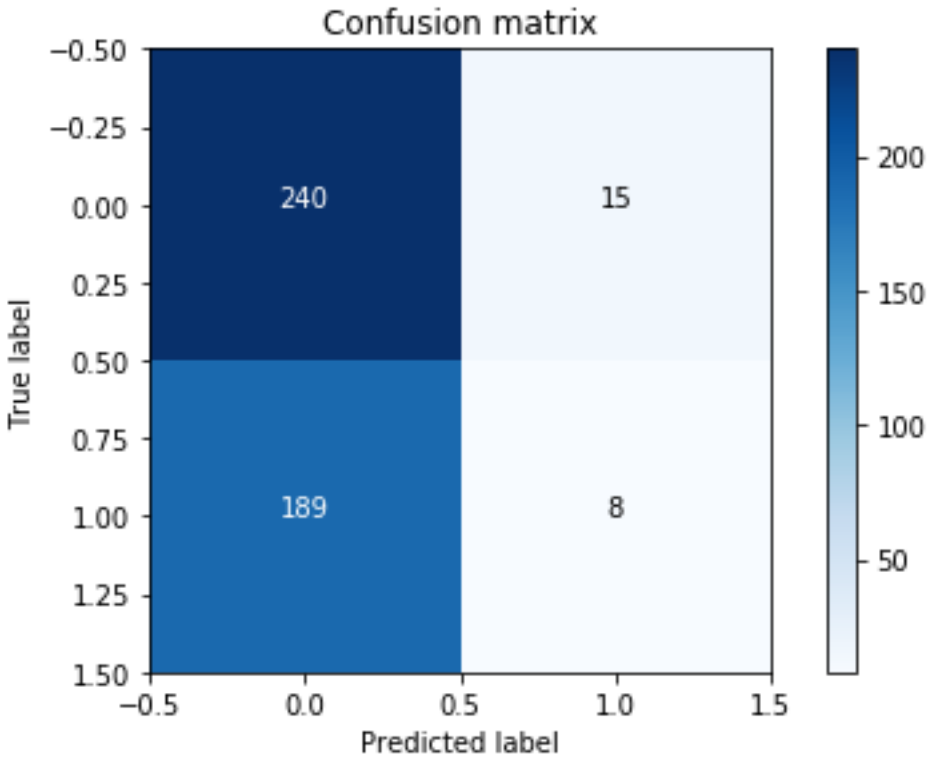
The best Cross-Validation Accuracy (PCA) is = 0.60270602706

Test accuracy = 0.592920353982

F1 Score = 0.625434554127

Confusion Matrix = [[184 71], [93 104]]

Confusion matrix, without normalization:



For Perceptron Classifier,

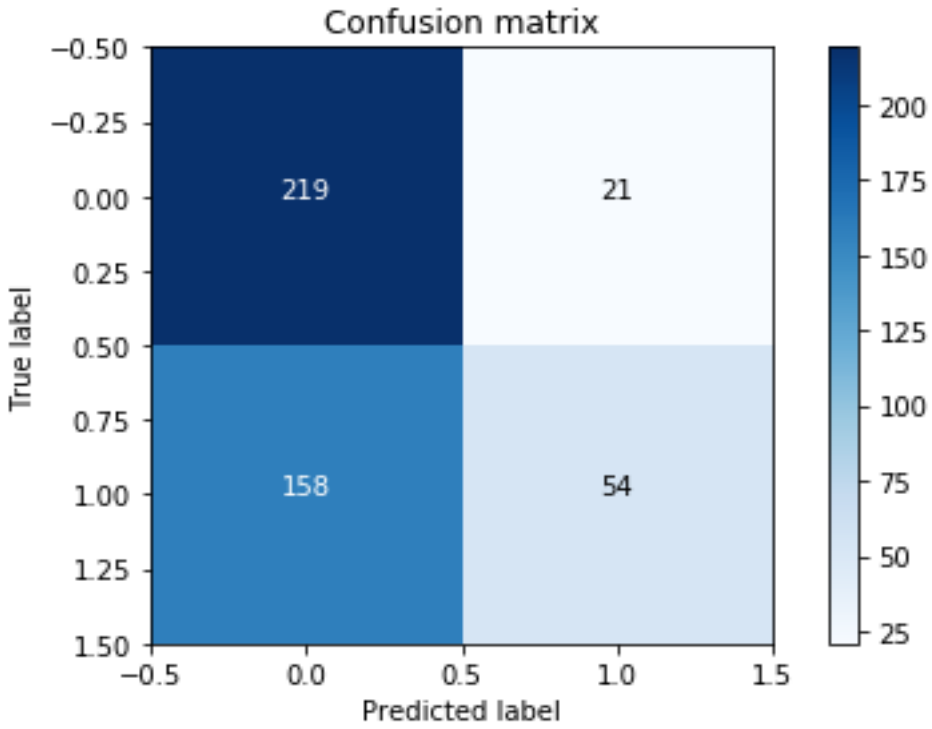
Final Weights = [[3.07334052 -1.6692933 -0.37578256 0.61644576 1.49009311 -0.14883494
  0.03484295 1.31320846 -0.68608751 -0.14372272 -0.05314438 0.44708251
 -0.20389333 1.2455336   1.1933848   2.05316415 0.07925163 3.12108806
 -0.44820357 1.03935281 -0.82416343 -0.61437721 -2.34744916 0.64586745
  0.94142856 -0.84074334 -0.64519375 0.61883049 -0.87191462 0.66096348
 -0.79760958 1.42563113 -0.70481396 1.00001546 -0.69144986 0.7344162
 -1.12216477 -0.40802541 -2.77686418 -0.72537432 -2.07785578 -0.64882411
 -0.15457473 -0.47943469 0.16732484 -1.12805559 -0.31341515]]

Test accuracy = 0.603982300885

F1 Score = 0.543096584011

Confusion Matrix = [[219 21], [158 54]]

Confusion matrix, without normalization:



Conclusion for Task1:

- Pre-processing of data is the most important part of the entire project on improving the accuracy of the classification models. It has been inferred that some features contain more relevant data as compared to others and the pre-processing steps helped in identifying such features.
- Normalization helps in speeding up the classification process and also helps in increasing the accuracy on the test dataset.
- Based on the F1 Scores got from application of the different classifiers, KNN Classifier has the best result in classification and Perceptron has the worst classification accuracy. The more the F1 score, the better.
- Application of PCA and setting the n_components attributes to different values doesn't bring about much changes to the test accuracy.
- During the preprocessing, if we set the threshold to the median rather than 1400, we get better test accuracy.
- The overall test accuracies for different classifiers is actually pretty low, just around 55-65%. None of the models give very good test accuracies.

Task2:

The difference between the previous case and this is the number of classes for classification. Whereas the previous one was a binary classifier, in this case, the problem is a multiclass problem. We do this by dividing the shares according to the following cases:

Class	Popularity	No. of Shares
S1	1	S<=900
S2	2	900<s<=1200
S3	3	1200<s<=1600
S4	4	1600<s<=3400
S5	5	3400<s

Here, the same preprocessing steps were carried out as before like Task-1 The only difference was the class labels assignment based on the above table.

The same preprocessing steps were followed. The classifiers were also kept the same. Thus, I used four different classifiers to fit to the training data and to analyze which one is the best for this data. The four classifiers used are:

- SVM Classifier
- Naïve Bayes Classifier
- KNN Classifier
- Perceptron Classifier

The performance evaluation was done on the testing data set that which would go through the same pre-processing steps as the training data did. The testing data was then fed into different classification models and predicted labels were obtained from testing the test data on the trained classifiers. The test accuracies were found from this data. Also, F1 score and confusion matrix was found for the test dataset. A confusion matrix is a table that is used to describe the performance of classification model on a set of testing data set whose true label test is known. he F1 score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst score at 0. The relative contribution of precision and recall to the F1 score are equal. The classification performance of each classifier is shown below.

Results for Task2:

For SVM Classification,

The best pair is C = 2.15443469003 and gamma = 0.464158883361

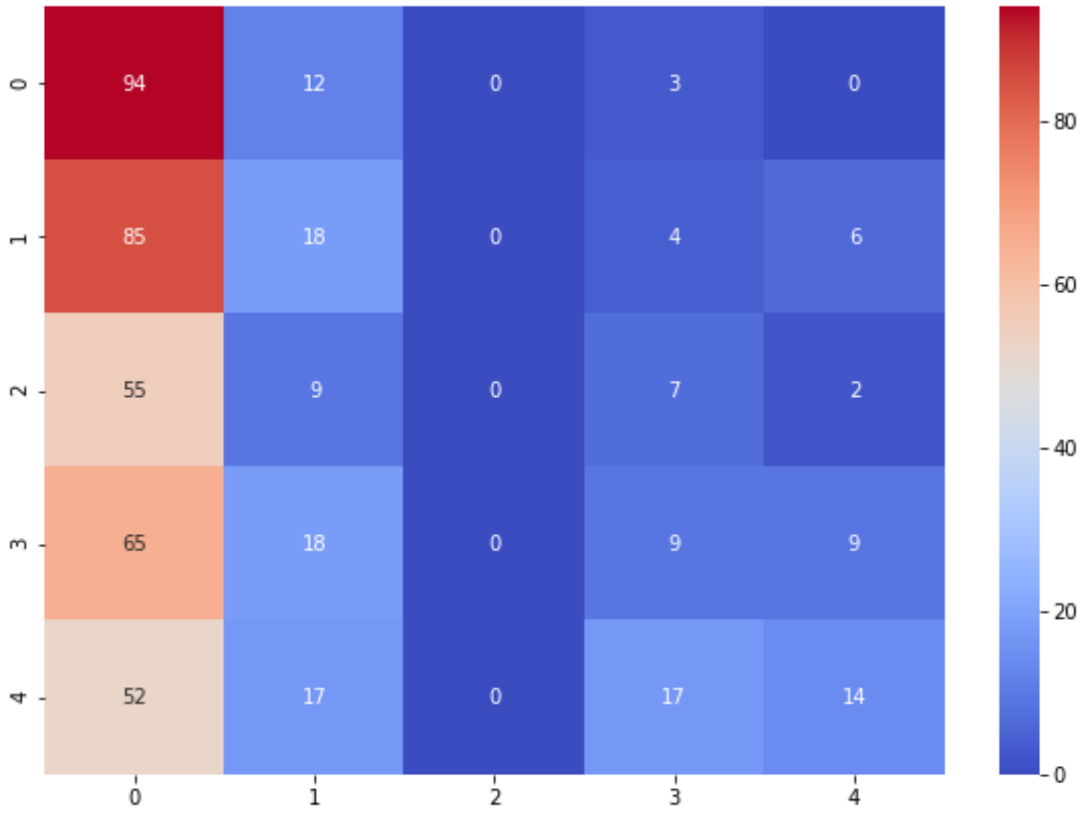
The mean Cross-Validation Accuracy for the best pair = 0.339841284688

The Standard deviation for the best pair = 0.0114936438504

Test accuracy = 0.272177419355

F1 score = 0.188521810728

Confusion Matrix = [[94 12 0 3 0], [85 18 0 4 6], [55 9 0 7 2], [65 18 0 9 9], [52 17 0 17 14]]



The x axis is the predicted label and the y axis is the true label for the above heatmap.

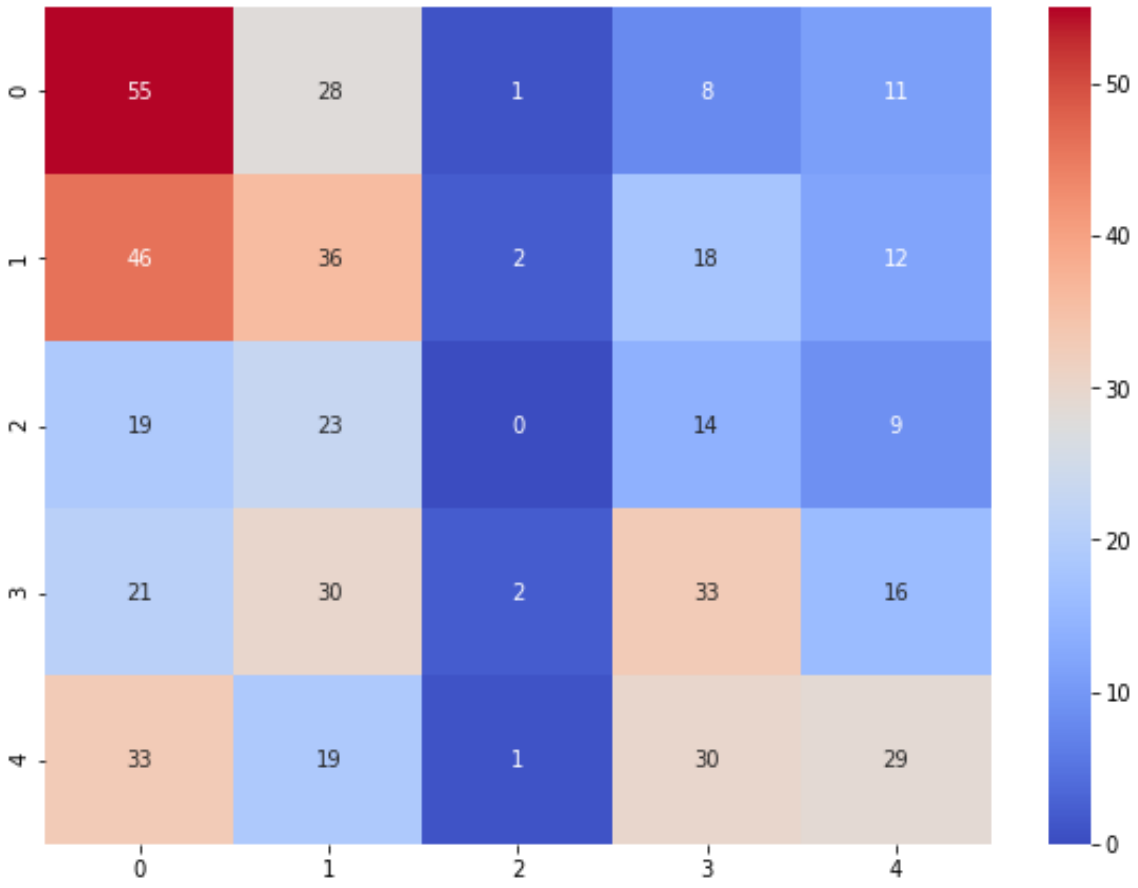
For Naïve Bayes Classifier,

The best Cross-Validation Accuracy is = 0.296752519597

Test accuracy = 0.308467741935

F1 Score = 0.26278828795

Confusion Matrix = [[55 28 1 8 11], [46 36 2 18 12], [19 23 0 14 9], [21 30 2 33 16], [33 19 1 30 29]]



The x axis is the predicted label and the y axis is the true label for the above heatmap.

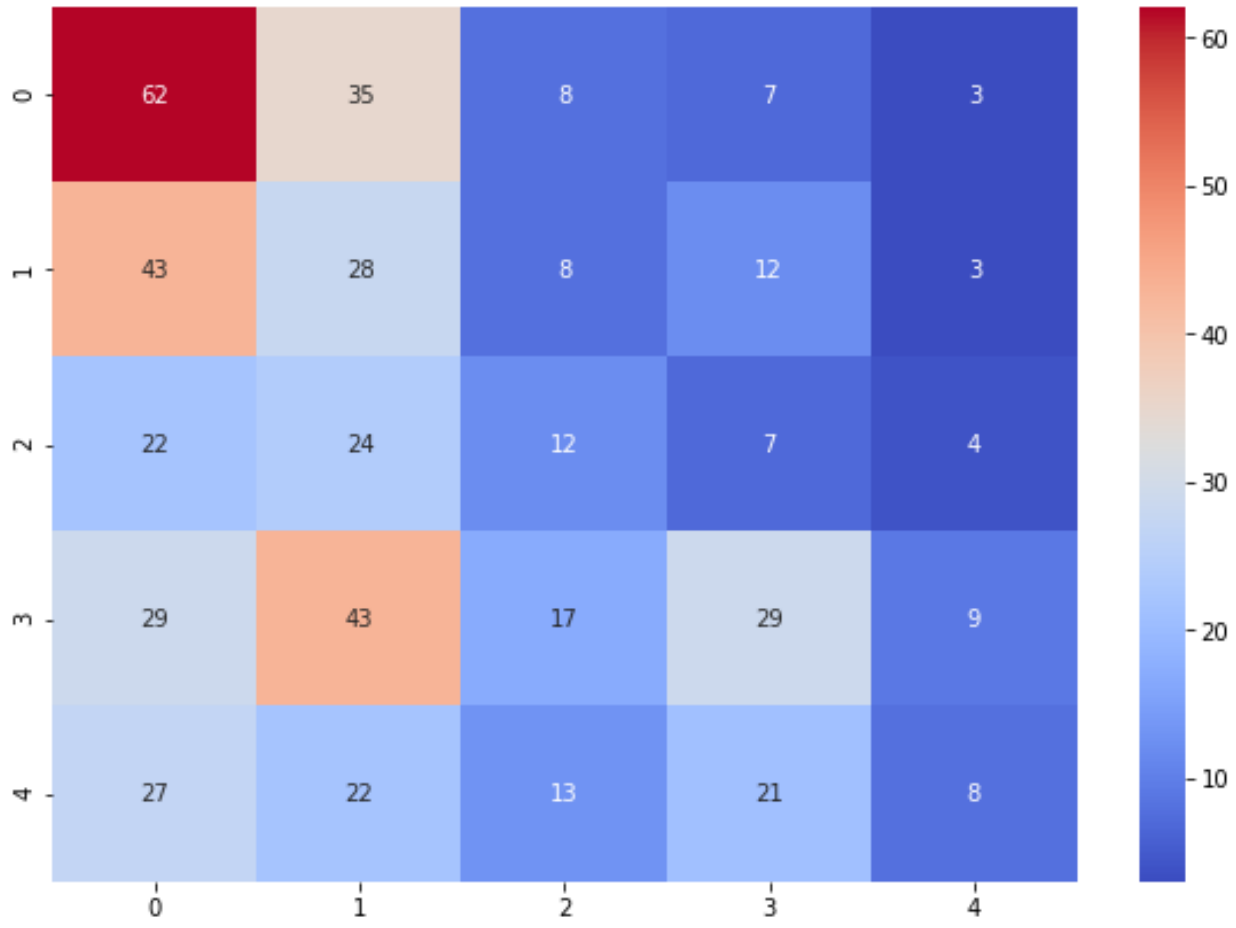
For KNN Classifier,

The best Cross-Validation Accuracy is = 0.247757847534

Test accuracy = 0.280241935484

F1 Score = 0.250806708596

Confusion Matrix = [[62 35 8 7 3], [43 28 8 12 3], [22 24 12 7 4], [29 43 17 29 9], [27 22 13 21 8]]



The x axis is the predicted label and the y axis is the true label for the above heatmap.

For Perceptron Classifier,

Final Weights:

```
[[  3.53783895e-01   7.86608998e-01  -3.73165178e-01  -2.97043392e-01
   -1.97752205e+00  -2.05002125e-01   2.32945561e+00   1.61185195e+00
   -4.99736053e-01  -9.62302563e-01   1.60088573e+00   2.10185471e+00
   -9.95743865e-01  -6.70190357e-01  -9.27943423e-02  -1.59901519e-01
   -6.66890484e-01  -3.39090861e-01   8.71111892e-01  -6.37296735e-01
   -4.40479298e-02   2.48197167e-01  -1.06205920e+00  -1.59839688e+00
    7.25959206e-01   5.01747234e-01  -5.05050610e-02   1.91316821e+00
   -1.30815579e+00  -1.16641035e+00  -5.45037845e-01  -1.09826301e+00
    8.20649939e-01  -4.26271622e-01   1.31062592e+00  -3.72232647e+00
    1.41649602e+00   2.17271615e+00  -1.23535703e+00  -1.48637094e+00
    4.86328078e+00  -2.19285910e+00  -5.28285219e-01  -1.21526979e+00
    2.35232393e+00  -3.74510889e-01  -4.99106907e-02]
 [ -1.27767607e+00  -7.21695029e-01   3.00831662e-01   7.49415882e-01
   -1.48063862e+00  -6.37989036e-01   2.19779305e-01   6.47797722e-01
   -1.83300946e+00   6.33549922e-01  -1.30472931e-01   5.88065279e-01
   -2.64857822e-01  -5.88340156e-01   1.21965620e-02   7.31375024e-01
    1.08880225e+00   3.46605854e-01  -4.80458671e-01  -2.81260443e-01
   -1.29757090e+00   5.51743336e-01  -6.91520826e-01  -5.77210729e-01
   -5.29454650e-01  -1.35902852e-01   7.63396097e-01   1.73034223e+00
    1.53404162e-02  -2.12293877e-01  -6.92014748e-02   6.44870055e-01
    1.11425749e+00   1.48339786e+00  -4.16959135e-01  -1.82410214e+00
    3.48833675e-01   1.36647276e+00  -3.45405258e-01   2.33884075e-01
    1.18232716e+00  -2.46089124e+00   1.79096600e+00  -1.95066778e+00
    4.60693785e-01   2.62757051e-01   9.39877645e-01]
 [ -4.92291875e-01   2.31759139e-01  -7.65253267e-01  -3.50491174e-02
    6.00289782e-01  -1.47153900e-01   1.08198158e+00   1.51969371e+00
```

-2.56077369e-01 1.00550466e+00 1.28717836e+00 4.01899203e-01
 5.70646774e-01 -1.19525925e+00 -7.64035527e-01 2.13157811e+00
 8.49269508e-02 -7.25257957e-01 8.42225036e-01 1.17017384e+00
 1.64764672e-01 -4.96154019e-01 -8.20941040e-01 -7.29877129e-01
-6.61742408e-01 8.28612063e-01 -2.09036537e-01 2.76551896e-01
-1.09548824e+00 6.97742050e-02 1.53301767e-01 2.93266546e-01
-9.97632126e-01 -5.43976828e-01 -1.13101912e-01 -6.44556561e-01
-4.78418492e-01 -1.87957227e-01 5.29226650e-01 -1.37771306e+00
-7.72628922e-01 -5.24355312e-01 4.32609015e-01 4.52597329e-01
 1.15317904e+00 5.32639744e-02 -1.30104624e+00]
[-1.77097069e-01 -1.07553832e+00 1.88123590e-01 4.80075329e-01
 1.55445500e+00 -1.16210798e+00 -1.91062129e+00 8.61615221e-01
-1.40194707e+00 -6.43582762e-01 -1.92368628e-01 -1.41041767e-01
 1.45266327e+00 4.34143375e-01 -2.42769044e+00 -6.68719192e-01
-8.62218318e-01 3.32698254e-01 2.23703849e-02 -1.05160902e-01
-3.54086217e-01 2.10659207e-01 -8.10586054e-01 5.81398019e-01
 2.68484078e-01 1.73146828e-01 1.89177717e-01 -1.18079282e+00
-7.27559876e-01 1.52236072e+00 5.67521642e-01 9.86282339e-01
 1.95064039e-01 8.50806438e-02 1.26504563e+00 6.56203652e-01
-1.37061204e+00 -1.42986397e-01 1.84202367e-01 1.90248650e+00
-3.05522898e-01 2.62929349e+00 -2.38029938e-01 1.25948681e+00
-2.01414636e+00 -3.68157180e-01 2.78535399e-01]
[2.54518035e+00 -1.01080619e+00 1.76680560e+00 4.04746603e-01
 1.15224315e+00 -3.95686833e-01 -2.07853421e+00 3.86323719e-02
-1.37339496e+00 3.66812728e-01 -1.02568367e+00 -1.04459325e+00
 1.13788752e+00 -2.13590750e-01 -4.55456508e-01 5.33789596e-01
 1.27932854e+00 1.86011480e-01 -1.50455360e+00 3.85094790e-02
-1.94151622e-02 8.08294969e-01 -1.39956584e+00 2.55079985e+00

-8.89340319e-01 -2.82618354e-01 -1.40660854e-01 -1.25834767e+00

1.81410106e+00 -4.98003379e-01 1.20274403e+00 2.45514950e-01

-8.91041774e-01 -6.87584714e-01 -4.88362187e-01 6.15639126e-01

5.59071093e-01 -1.38782784e+00 1.08532333e-01 2.83364712e+00

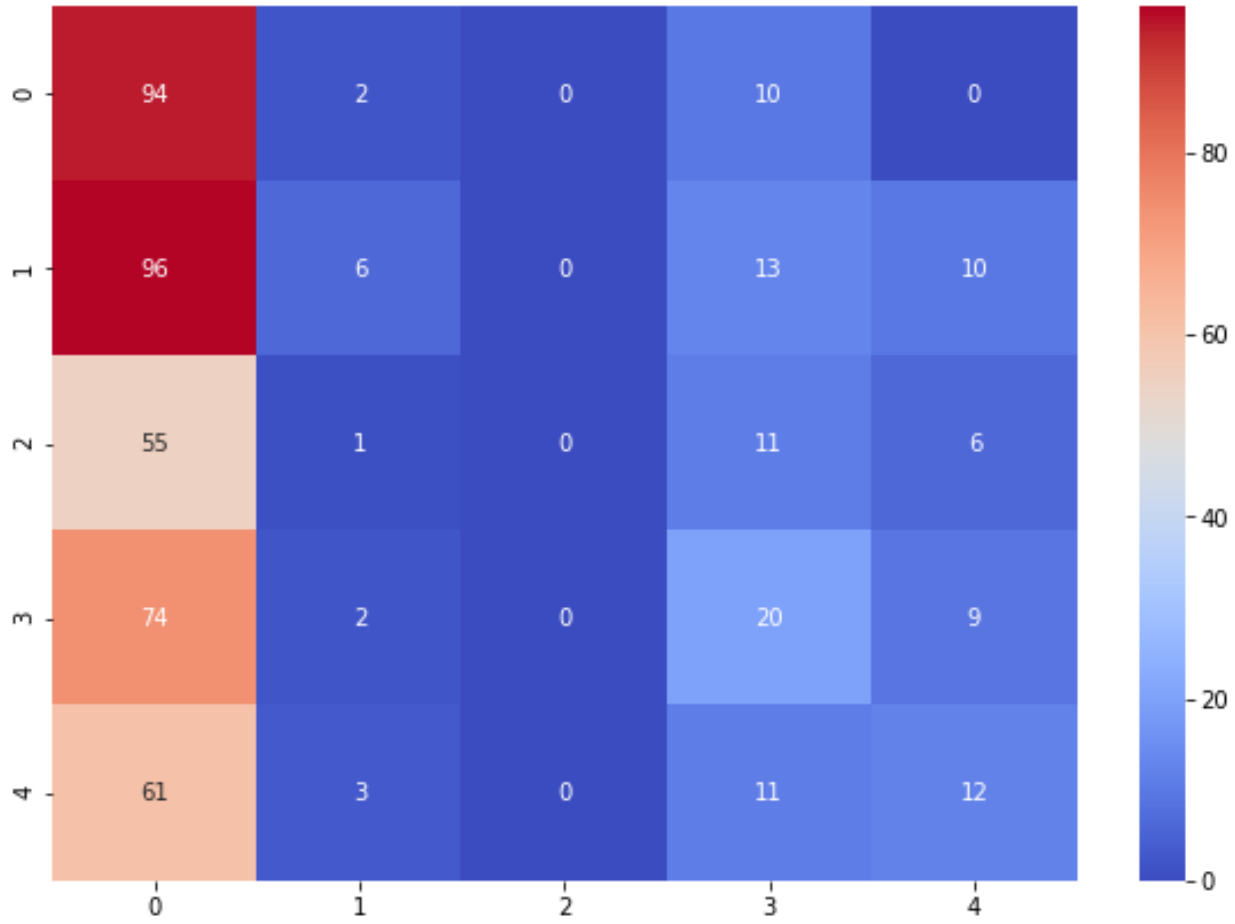
-3.60382875e+00 2.22792626e+00 8.57066468e-01 9.07034359e-04

-1.47894177e+00 -1.36166927e+00 2.15316152e-01]]

Test accuracy = 0.266129032258

F1 Score = 0.180400943143

Confusion Matrix = [[94 2 0 10 0], [96 6 0 13 10], [55 1 0 11 6], [74 2 0 20 9], [61 3 0 11 12]]



The x axis is the predicted label and the y axis is the true label for the above heatmap.

Conclusion for Task2:

- Pre-processing of data is the most important part of the entire project on improving the accuracy of the classification models. It has been inferred that some features contain more relevant data as compared to others and the pre-processing steps helped in identifying such features.
- Normalization helps in speeding up the classification process and also helps in increasing the accuracy on the test dataset.
- Based on the F1 Scores got from application of the different classifiers, Naïve Bayes Classifier has the best result in classification and Perceptron has the worst classification accuracy. The more the F1 score, the better.
- Application of PCA and setting the n_components attributes to different values doesn't bring about much changes to the test accuracy.
- The overall test accuracies for different classifiers is actually pretty low, just around 20-30%. None of the models give very good test accuracies.

The main result we found out is that the test accuracy for all classifiers is more for this dataset in the case of two class problem (Task-1) than in multiclass problem(Task-2)

References:

- http://scikit-learn.org
- ***Outliers*** **-** John M. Cimbala, Penn State University http://www.mne.psu.edu/cimbala/me345/Lectures/Outliers.pdf
- ***A Comprehensive Guide to Data Exploration –*** Sunil Ray http://www.analyticsvidhya.com/blog/2016/01/guide-data-exploration/#three
- https://stackoverflow.com/
- Is Your Story Going to Spread Like a Virus? Machine Learning Methods for News Popularity Prediction - Xuandong Lei, Xiaoti Hu , Hongsheng Fang – Stanford University
- Tatar, Alexandru, et al. "A survey on predicting the popularity of web content." Journal of Internet Services and Applications 5.1 (2014): 1-20
- Tsagkias, Manos, Wouter Weerkamp, and Maarten De Rijke. "Predicting the volume of comments on online news stories." Proceedings of the 18th ACM conference on Information and knowledge management. ACM, 2009
- Fernandes, Kelwin, Pedro Vinagre, and Paulo Cortez. "A Proactive Intelligent Decision Support System for Predicting the Popularity of Online News." Progress in Artificial Intelligence. Springer International Publishing, 2015. 535-546.
- Bandari, Roja, Sitaram Asur, and Bernardo A. Huberman. "The Pulse of News in Social Media: Forecasting Popularity." ICWSM. 2012.
- James, Gareth, et al. An introduction to statistical learning. New York: springer, 2013.
- Class Notes and Discussion Notes from USC EE 559 course.