Final project report

CIS 520 Machine Learning

Professor Lyle Unger

**1.** **Introduction**

This project is to predict male or female from a tweeter’s words and profile picture using different types of models. In this report, we will discuss and analyze the result of all the different method we have tried and provide some data visualization to the project. For all the method, we evaluated the result by **10-fold cross validation accuracy** as well as the **test set accurac**y based on the given **validate dataset**.

The data we are given consist of training set (n = 4998) and test set (n = 4997) for the following:

1) How often each user used each of the 5,000 most frequent words

2) The raw images 100\*100\*RGB

3) A set of extracted features for each user's image

4) The gender outcome

**2.** **Testing**

To test our model, we used a 10-fold cross validation scheme. The training data was divided into N equally sized (as close to equal as possible) “fold”. All the words in *i*-th were assigned a number *i* and switch. Each model was trained on N-1 folds of the 5000 frequent words. The remaining held out fold was used to test the accuracy of our predictions.

**3.** **Experiment and Result**

We divided all the model we have tried into 2 checkpoints:

1. The 1st baseline (86% accuracy)

2. The 2nd baseline (88% accuracy)

**3.1** **Models in 1st baseline:**

**Linear Regression**

The first method we tried is linear regression. Since the gender vector is a binary data as 1 for female and 0 for male, we have to transform the vector into -1 and 1 so that we can train the coefficient vector **w** as the following:

Then to predict the test with the coefficient vector **w**:

However, with 10-fold cross validation, we obtain an accuracy of 0.8245 which obviously failed the first baseline. One of the reason might be some of the words are strongly correlated or redundant. Therefore result in very large variance in the final parameter estimates which cause overfitting. We then considering using PCA feature selection before linear regression to reduce variance and overfitting. Although we are getting a slightly better result with 82.45% accuracy on 10-fold cross validation and 70.22% on test set accuracy, the model still failed the 86% baseline.

**Support Vector Machine with Intersection**

A multi class, L2-regularized, L2-loss SVM is also implemented from MATLAB package LIBSVM. The model is also train on how many time each user used the 5000 most frequent words. With 10-fold cross validation produced accuracy of %. Test set accuracy is 0.8926%. The parameter used in SVM might provide an over-fitted model by using most of the training dataset. However this over-fitting is not fatal and thus may still positively contribute to the final model. The training time increased at an approximately quadratic rate as number of observation trained.

**Logistic Regression**

We then considered Logistic Regression given the binary outcome of the gender. We have implemented the L2-regularized logistic regression model using the built-in MATLAB package LIBLINEAR. The model was trained directly on how many time each user used the 5000 most frequent words. The 10-fold cross validation accuracy is 0.8641%. The test set accuracy was 85.84%. This model perform a lot better which beats the threshold given the simplicity of the model itself. However the training time growth in an approximate quadratic rate as training observation increase.

**3.2** **Models in 2nd baseline:**

**Kmean**

We have implement the K\_mean model based on the built-in MATLAB function as

[IDX,C] = kmeans(train\_x, K, 'MaxIter', 500)

We train the model with K=25. The model consisted of the centroid of the cluster and the predicted gender for each cluster. By taking the minimum Euclidean distance between each test point and the cluster the model predicts the final result. Kmean is one of the simplest model we created and with a very efficient running time. However the result turned out to be fairly poor with a 10-fold cross validation accuracy of 40% and the test set accuracy of 40.54%. We have tried different K values, however none of them seem to work significantly better. This is probably because our cluster do not always have a well defined centers in this particular dataset, which is 5000 most frequent words. Also the different initial partitions can result in different cluster.

**PCA Logistic Regression**

Although logistic model generates a relatively high accuracy, there are still some problem associated with it. The logistic model has a high dimension given there are more features (p) than observations (n) which require time and space. The multicollinearity of the parameter will also affect the estimation and performance of the model. Therefore we decide to use PCA to do a feature selection before our logistic regression model to reduce the dimension and thus improve the estimation of our parameters. However the ensemble gave us a % of 10-fold cross validation accuracy and 57.28% of the test set accuracy, which is significantly lower than before. This is probably because the frequency of each words and some image features such as age are strongly correlated with each other. Also this tells us there might not be a lot of redundant in the word dataset. Therefore performing PCA will remove some of the highly dependent features will reduce the prediction accuracy.

**Naïve Bayes with PCA**

In order to meet the test speed performance, we tried to implement Naïve Bayes model base on the built-in MATLAB since the model is easy to build and without any complicated iterative parameter estimation. The model did not perform very well producing % 10-fold cross validation accuracy and 70.44% of the test set accuracy. We have tried to perform PCA feature selection before Naïve Bayes however the result is not significantly improved. One of the possible reason might be the Naïve Bayes assumption: linearly independent of the predictor isn’t well satisfied. Since words frequency are often depends on each other.

**K Nearest Neighbor**

This model is also implemented based on the built-in MATLAB function *fitcknn.*30 nearest neighbor were used in predicting results. It is also a very simple algorithm as to implement and compute. However, The choice of K is essential in building the KNN model. And K can be one of the most important factors of the model that can strongly influence the quality of predictions. We have tried K value as 1,5,10,20,30,40,50 respectively and found that K =30 we have the best prediction result. The model has % 10-fold cross validation accuracy and 75.62% of the test set accuracy.

**A summary table of the result of each method developed**

|  |  |  |  |
| --- | --- | --- | --- |
| Model | 10-fold Cross Validation Accuracy | Test set Accuracy (validate data ) | Iteration/Second or running time? |
| Linear Regression |  | **0.7022** |  |
| Logistic Regression |  | **0.8584** |  |
| Support Vector Machine |  | **0.8926** |  |
| PCA Logistic |  | **0.5728** |  |
| Naïve Bayes with PCA |  | **0.7044** |  |
| K Nearest Neighbor |  | **0.7562** |  |
| K Mean |  | **0.4054** |  |

**4.** **Conclusion**

We performed well enough with SVM with intersection and Logistic regression.

Our performance is fairly consistent among cross-validation and test set, which means that overfitting is probably not raising an issue in our model. Feature selection such as PCA is one of the powerful method to improve our result as our dataset has relative high dimension given limited number of observation. However the running time of PCA is one of the big concern.

Finally, implemented stemming to group the work with same meaning contextually before compute the gram matrix will probably improve our model as well. For example, as we can see from the 5000 most frequent words, “aha”, “ahahaha”, “hahaha” or “ah” and “ahhhh” with the last letter repeating for multiple times would all be stemmed to be the same word. We can implement a “Has-multiple-repeated letters” feature when we stemmed the words. This is also helpful because female tend to use more repeated letter more than males. Stemming words would also reduce dimension and increase the counts as it combines features (i.i.d columns), therefore may lead to a better model on the gram matrix.

man\_most\_word\_top50\_words\_sort =

'her'

'happy'

'oh'

'she'

'thank'

'morning'

'even'

'little'

'feel'

'because'

'yes'

'things'

'look'

'always'

'hope'

wman\_most\_word\_top50\_words\_sort =

'1'

'video'

'twitter'

'world'

'4'

'check'

'game'

'man'

'5'

'into'

's'

'down'

'free'

'any'

'where'