CIS520 Final Project Report

He Chen, Henri Maxime Demoulin, Gabrielle De Micheli

December 11, 2016

Methods Accuracy Report

First, let us explain some of the preprocessing steps that we have taken. For the word count preprocessing, we deleted all the English, French, and Spanish common stop words, words containing only one character (punctuation?), unicode words starting with $\backslash u$, and words that are http and rt links. All of these preprocessing steps were tested one by one to confirm that it indeed does increase the cross validation accuracy.

Another thing that we have done to try and see which words are good to delete in the preprocessing was iterating through all 10,000 of the word columns, and if deleting that column yielded significantly better cross validation accuracy (1.5 percent of better), then we deleted that column in the preprocessing step. However, that did not work and we ended up with more cross validation error (2-3 percent) than we started with.

Generative Methods

Naive Bayes

We start by using Naive Bayes only on the words. This method gave us around 80% testing accuracy on the words without preprocessing. If we do the preprocessing described above, the testing accuracy rises to 81.42%. In order for Naive Bayes to work, we got rid of all the words that do not appear in any tweets, since $fitncb()$ had issues dealing with words of probability 0. In order to use a variation of the bag of word model, the words are set to 1 if they appear in a tweet and zero otherwise. We have also tried and tested $\log(w_i + 1)$ and $\sqrt(w)$, but doing that made the accuracy of Naive Bayes go bad $(22 - 35)$ percent error. So we concluded that the best way was simply checking if the word exists or doesn't exist (0 or 1). From our experience, Naive Bayes is both extremely accurate and extremely fast when used on the word counts.

Discriminative Methods

Linear Regression

We primarily did linear regression on only the CNN features. For each of the linear regression methods we have tried, we have tried it on the raw CNN features, the CNN features max pooled to various degrees, and the CNN features mean pooled to various degrees. At first we tried elastic net regression, which got around 65 percent cross validation accuracy give or take 3 percent for all of the different pooling methods. Then we tried L1 regression, which got pretty identical results as elastic net. Then we tried L2 regression which was slightly better than elastic net and L1, with about 68 percent cross validation accuracy. In all 3 of these regression methods, we were not able to get much better than 68 percent cross validation accuracy, which combined with the bag of words Naive Bayes, resulted in a 78-80 percent cross validation accuracy. Our conclusion was that linear regression was not appropriate for the CNN features, and so we did not incorporate linear regression in our leaderboards model.

Random Forest

"We tried random forest only on the words but we observed the following: with 20 trees, we had a cross validation error of 0.2167 and with 100 trees a cross validation error of 0.2116. We use the same preprocessing process that with Naive Bayes".(DELETE MAYBE)
For 100 trees, the cross validation error is 0.0533. For 500 trees, we get cross validation error of 0.0520 and for 1000 trees, we get a cross validation error of 0.0509. For the last case, the cross validation accuracy is 0.7776.

SVM

If we use SVM with an RBF (Radial Basis Function) kernel function, we have a cross validation error of 0.3813. Without RBF, the cross validation error is 0.26. The linear kernel is twice as accurate as the RBF function. However, if we use RBF but this time without the word-count, we have a cross validation error of 0.3776, which is greater than the error obtained with the two previous combinations we tried. If we try to use SVM without the word-count and without RBF, the cross validation error is 0.2626. We also try SVM only on words, and notice that it performs better than KNN.

Perceptron

TO DO

Instance based Methods

KNN

We use KNN on all four datasets and get a traning error (cross-validation) of 0.3556.

Regularization Method

our own regularization method other that the standard L_i penalty.... TO DO

Semi-supervised dimensionality reduction

PCA

We try using PCA on the *image dataset*. First, we use PCA with all the PCs combined with SVM which gives a cross validation error of 0.0324. We do the same again but only with 1000 PCs. This gives an error of 0.1198. Moreover, using 570 PCs (corresponding to 50MB), we get a cross validation error of 0.1509.

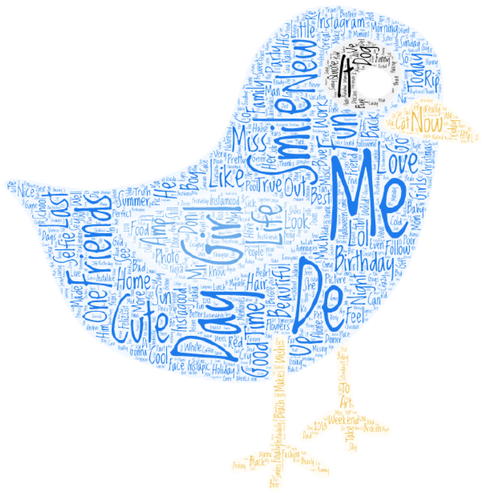
On the *word dataset*, we also try using PCA with various numbers of PCs (all, 500 ...) and this time, combined with random forest. This gives a cross validation error of 0.0064. The test-accuracy is 0.7491.

Methods Analysis

TO DO

Visualization

We consider the well-classified words and we want to find which words are the most important. In order to visualize this, we compute a word cloud.



Me
De
Day
Girl
Smile
New
Fun
Cute
Friends
Love
Life
Like
Miss
Birthday
Home
Instagram
Today
Lol