CIS 520: Final Project Report

PizzaPlanet
Brandon Duick, Chris Jordan, Stephen McGill

November 25, 2009

Results

Blogs

AdaBoost on "Significant" Words

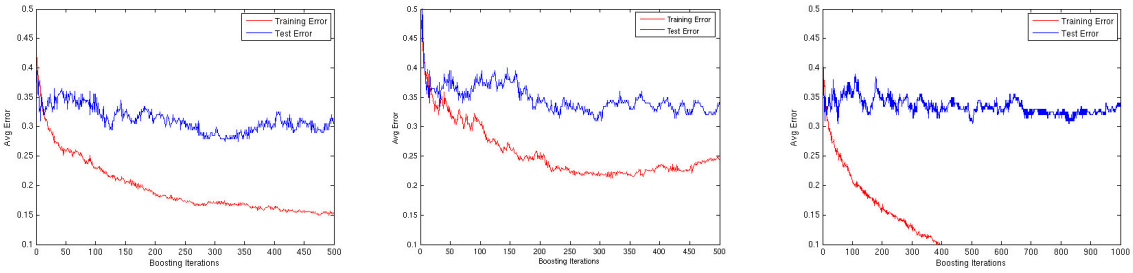

Figure 1: Test error vs. training error for Adaboost implementation on blogs. (a) 500 "significant" words and 500 iterations splitting on binary existence of words. (b) 500 words and 500 iterations splitting on number word appearances. (c) 20000 words and 1000 iterations splitting on existence of words.

Fold	Baseline Test Error for Gender	Adaboost Test Error for Gender
1	0.6794	0.6794
2	0.6294	0.6529
3	0.6088	0.6853
4	0.7029	0.6353
5	0.7324	0.6912
Avg Test Error	0. 6706	0.6688

Table 1: Cross-validation of Adaboost with 500 "significant" words on 350 iterations:

Word Stemming

One of the suggestions on the CIS520 wiki indicated that it would be a smart idea to apply a word stemming algorithm to the dictionary and blog posts in order to collect words of the same type for analysis. The idea here is that "dance," "dancing," "danced" all describe the same topic, and can be considered a single feature. By collating these features together, the occurrence of the stem, "danc" provides a stronger indication of the blog topic - and therefore a strong indication of gender and age.

We passed our stemmed dictionary and word set to the Naive Bayes trainer and tester. After running cross validation, we found that the word stemmer achieved an accuracy of 68.12% for gender and 67.41% for age. This was around 1.5 and .5 percentage points better than the baseline Naive Bayes, respectively.

	Gender		Age	
Fold	Baseline	Stemmed	Baseline	Stemmed
1	0.6235	0.6471	0.6676	0.6647
2	0.6941	0.7000	0.6529	0.6471
3	0.6324	0.6500	0.6647	0.6735
4	0.6853	0.7176	0.7000	0.7088
5	0.6912	0.6912	0.6676	0.6765
Average	0.6653	0.6812	0.6706	0.6741

Table 2: Cross-validation Accuracy Levels of the stemmed word base

Average Word Size

We added an average word size feature in order to give an extra layer of discretion to the Naive Bayes predictor. The training accuracy after appending the probabilities for word length were astonishingly low, with age accuracy being 0.5367 and gender accuracy being 0.5587. For this reason, we abandoned using word length as a feature.

Faces

	Age Accuracy	Gender Accuracy	Average Runtime (secs)
No virtual samples	0.439	0.718	525.0
Images scaled by 1.1 and 0.9	0.444	0.721	753.4
Images scaled by 1.2 and 0.8	**0.462**	0.711	741.7
Images scaled by 1.4 and 0.6	0.419	**0.741**	745.4

Table 3: Results Using Scaled Virtual Samples

	Age Accuracy	Gender Accuracy	Average Runtime (secs)
No virtual samples	0.444	0.719	447.0
Images mirrored	**0.455**	**0.743**	627.5

Table 4: Results Using Mirrored Virtual Samples

	Age Accuracy	Gender Accuracy	Average Runtime (secs)
No virtual samples	**0.455**	0.701	446.0
Images translated by 2 pixels	0.438	0.716	639.3
Images translated by 4 pixels	0.437	**0.733**	633.9

Table 5: Results Using Translated Virtual Samples

Weak Classifiers	Age Accuracy	Gender Accuracy	Average Runtime (secs)
25	0.442	0.743	502.4
30	0.442	0.747	555.1
35	0.445	0.740	609.2
40	**0.445**	0.751	663.2
45	0.443	**0.755**	716.7

Table 6: Results Using Different Numbers of Boosting Iterations

Analysis

Blogs

AdaBoost

Significant words were found by analyzing the test set and determining which words had the greatest difference in percentage usage between men and women. Percentage usage is defined as the number of times a word appeared in a male or female blog divided by the number of male or female blogs.

Adaboost was then applied to the data with the feature spaced modified to only specify the existence (or lack thereof) of the "significant" words. It was also modified to split based on the number of occurences of the words. Various combinations of boosting iterations and the number of words used were tried. To get an idea of the proper number of iterations plots comparing the training error to the test error were created. Some example plots are shown in Figure 1.

The cross-validation on 5 folds of one of the better combinations (500 words and 350 iterations) is shown in Table 2. This cross-validation failed to beat the baseline tested on the same folds, and all submissions to the nightly checkpoint failed to pass.

The results show convincingly that the dataset is much better suited for naive bayes than a boosting approach. With the number of features and the amount of overlap between any single feature it is difficult for boosting to make effective splits. Naive bayes on the other hand is able to accumulate slight differences over the many features to make reasonably more effective classifications.

Cluster Weight	Gender Accuracy
0.0	**0.766**
0.05	0.762
0.1	0.760
0.15	0.756
0.2	0.755
0.25	0.753

Table 7: Results Using Cluster Voting for Borderline Gender Classification

Word Stemming

In order to accomplish the word stemmer, we downloaded the Porter Stemmer Algorithm, which is available online[1]. We used the MATLAB script available on the site. Unfortunately, the stemmer would fail on some of the words present in blog set. For instance, the word 'oing' was present in the dictionary, and the porter stemmer would fail, since there was not enough letters in the stem - no single letter is a word stem.

To fix this error, we modified the stemmer code to just output the full word, in this case 'oing,' as the word stem, rather than outputting 'o.' This took a few iterations of debugging, but was worth the patience. It was interesting to see how misspellings affected our machine learning system.

We used the word stemming as an unsupervised method in our project. After stemming, the size of the dictionary decreased from 89182 words to 72221 words.

Average Word Size

We probably achieved such low accuracies when adding the word length options because we did not normalize, or find a better weighting for these features, as they were just "tacked on" as extra logarithms. We used a linear combination, although by changing these constants randomly, we saw minimal improvement, and hardly performance that beat baseline. The confusion matrix also showed minimal choosing of old people and young people, but mostly middle aged. This could indicate an implementation bug. Due to such problems, we abandoned using this method.

Faces

Implementation

For gender classification on faces our base system used AdaBoost on Haar-like features. For age classification we used a similar set up and trained a one vs. all classifier for each of the age classes and took the classifier with the maximum score as the label. For our implementation we cropped our faces to 128x128 pixels at the center of the image. Our implementation uses

[1]http://tartarus.org/ martin/PorterStemmer/

haar-like features and the integral image technique to extract these features efficiently from our images. Our features were spaced throughout the image in steps of 8 pixels and were restricted to those with an area of 2^{10}, 2^{11} , 2^{12}, or 2^{14} pixels (in order to make calculations simpler, feature types composed of 3 or 9 box parts we used respectively 2/3rds or 8/9ths of the area instead of the actual area). We also restricted the dimensions of each feature to be a multiple of 32 in order to further reduce the number of features. We calculated these features over 4 image channels: grayscale intensity, red from the RGB color space, a from the CIELAB color space, and saturation from the HSV color space. These channels were chosen because they resulted in the best gains over cross validation tests. This resulted in a set of 2176 features which allowed for training of a classifier with 30 to 45 decision stumps from 390 training samples in roughly 1.5 to 2 minutes. For training we used the GML AdaBoost library[2]. In our final implementation we used the full training set and 40 decision stumps for the age classifiers and 45 weak classifiers for our gender classifier. Visualizations of the top 5 features for each of the four classifiers can be seen in Figure 2.

Virtual Samples

We experimented with virtual samples to increase the size of our training set. In order to test different experimental set ups, we used cross validation with 5 folds and ran 3 trials and took the average test accuracy and run-time of the cross validation test. We tried generating virtual samples by adding the mirror of our training images, adding small translations of our images, and by scaling our images slightly. We selected randomly from the generated set of virtual samples so that the number of images added to the training set was roughly half the number of images originally included in the training set. Although a number of virtual sample set ups led to gains during cross validation, all of our most promising set ups led to decreased scores on the checkpoint so we did not using any virtual samples in our final submission. Cross validation results on different virtual sample set ups can be seen in Tables 3, 4, and 5. We also cross validated boosting with different numbers of weak classifiers. The results from these trials can be seen in Table 6.

K-means Clustering

In order to improve results on borderline cases of our gender classifier, we attempted to combine a secondary classification score with the score from our boosted classifier. To do this we cropped images to 196x196 pixels and computed 1701 hog features from blocks of size 64, 32, and 16 using publicly available code from the INRIA Pedestrian detector[3]. We then used k-means to group our images into 20 clusters using k-means with Euclidean distance based on these features. We then gave each cluster a vote based on the average label of each member of the cluster times some constant clustering weight. For borderline gender cases, we found which cluster the sample was closest to and added the vote to the score from the boosted classifier. Unfortunately, with cross validation we found the best clustering weight

[2]http://graphics.cs.msu.ru/ru/science/research/machinelearning/adaboosttoolbox

[3]http://www.cs.berkeley.edu/ smaji/projects/ped-detector/

to be 0, so this method did not help us improve classification of borderline cases. Results from cross validation can be seen in Table 7.

Visualizations

Blogs

When trying to determine features to select, we tested word length, Sentence Length, number of Capital Letters, and a linear combination of all these features. We hoped to separate the different age classes, but for the most part, the old folks were a subset of the younger folks.

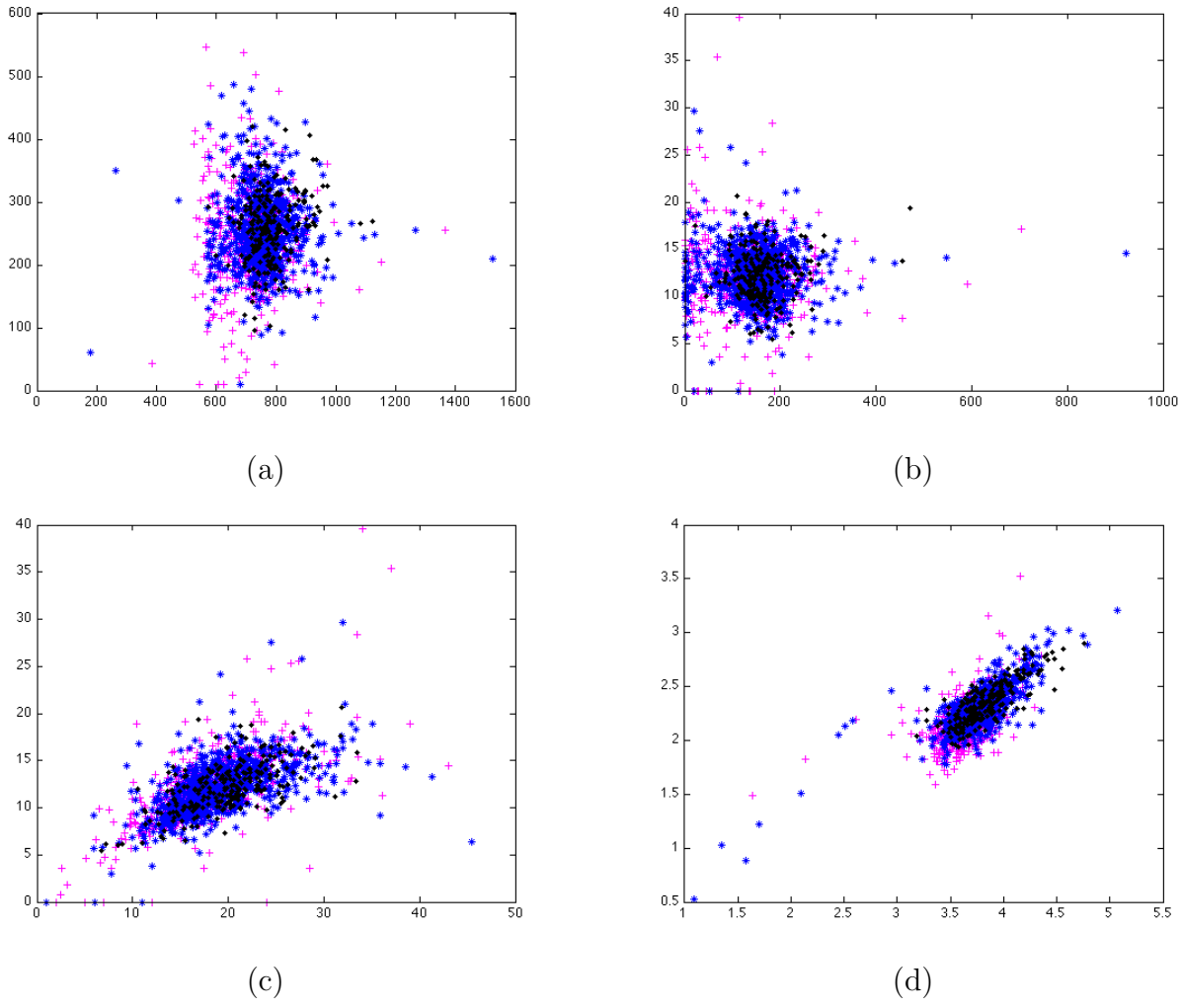

Table 8: (a) Linear combination of word length, capitilizations, and sentence length vs. the respective standard deviations. (b) Capitalizations mean vs. standard deviation. (c) Sentences length mean vs. standard deviation. (d) Word length mean vs. standard deviation.

Faces

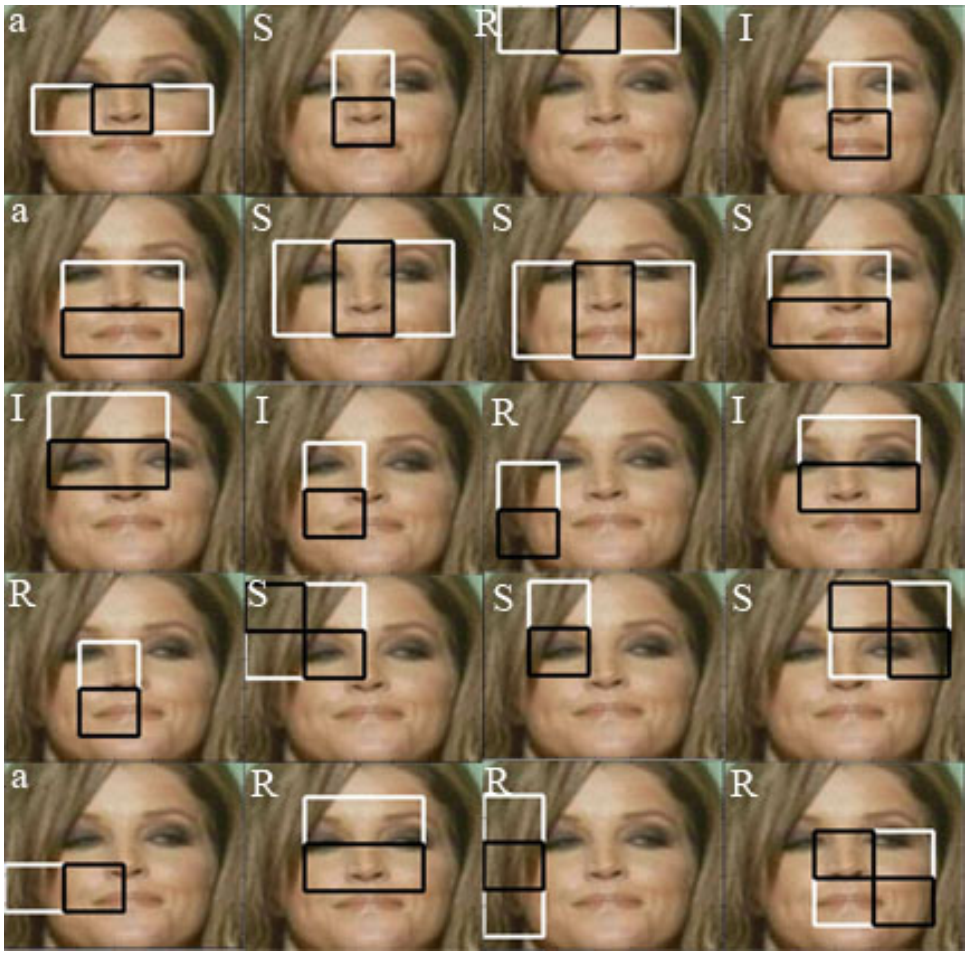

Figure 2: Visualization of Haar features chosen for weak classifiers: The top weak classifiers are in the first row, 2nd in the second row, and so on. The columns from left to right are the gender classifier, the young classifier, middle aged classifier, and old classifier. The image channel of the feature is designated by a letter in the upper left corner.