Kyosuke Tanaka
Mark Díaz

EECS 349 Final Report

Method

In order to create our full dataset, we collected data from 3 sources. We first obtained an archive of Threadless design images and data from design competitions between February 2010 and December 2011. In addition to average score, number of votes, and number of votes per rating option (0-5), the dataset features the standard deviation for votes for a particular design, the z-score for a design's average rating in the competition, and the number of votes flagged by Threadless as spam. We then took the images from the dataset and ran them through the DeepSentiBank image recognition software to generate the list of semantic word pairs for each image as well as a confidence score for each word pair. Additionally, we used the python Pillow package (formerly Python Image Library) to extract the average rgb values, percent white values, percent black values, and resolution for each image. Because relatively few pixels in the t-shirt design images were pure white (rgb values = (255,255,255)) or pure black (rgb values = (0,0,0)) we created a threshold value. RGB values greater than 230 were determined to be "white" and values less than 30 were determined to be "black". The combined Threadless data, DeepSentiBank data, and extracted image data formed our final dataset. In total our feature space amounted to 2,109 features.

We decided to predict a binary "good" or "bad" output for designs. For our training set we defined a good design as having an average score in the top 2,500 scores of t-shirt designs. We took 2,500 images with the highest user scores and labeled them as good designs, and took the 2500 images with the lowest user scores as bad designs. To create our test dataset, we randomly sampled 5,000 data points from the remaining 34,952 data points. We did not use the full dataset of images because the vast feature space made testing and training quite slow.

Testing/Training

Our main interest in testing and training our model was to determine which semantic concepts (adjective-noun pairs) were most predictive of t-shirt design. We trained multiple machine learning models using a variety of machine learning algorithms including KNN, SVM, and Naive Bayes. Our dataset provided challenges because of its massive feature space, which proved to be too computationally intensive for some approaches. The adjective-noun pairs generated by DeepSentiBank alone amounted to over 2,000 features, in addition to voting score statistics and image details including average RGB values and resolution. Furthermore, the adjective-noun pair probability scores for each design tended to hover extremely close to zero, with the exception of a few pairs for most images. This made our feature space not only large, but also very sparse.

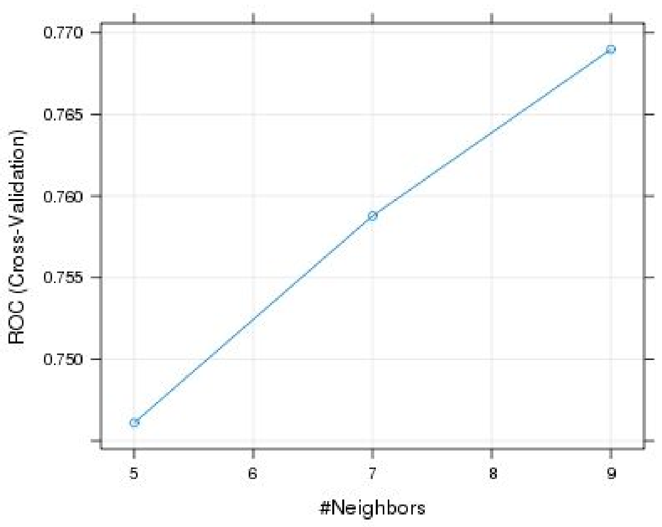
We used R statistical software, to train our data and predict the success of T-shirt designs. Particularly, using the “caret”, “knn”, and “e1071” packages, we employ k-Nearest Neighbors (KNN), Support Vector Machine (SVM) and Naive Bayes. We used a KNN learner, a SVM learner, as well as a Naive Bayes learner to train different learning models. The type of features in our dataset, particularly the semantic word pairs, are commonly seen in network and text data. We chose these algorithms because they are commonly used to train on these similar datasets. Naive Bayes in particular, runs efficiently and is able to handle the large feature space well.

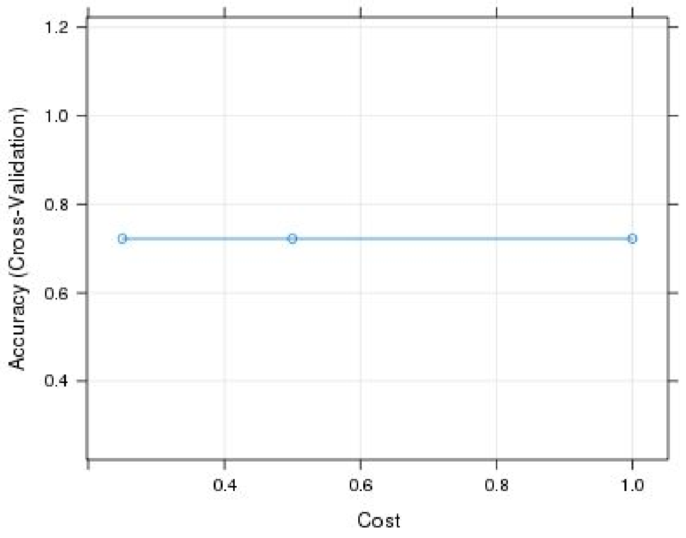
Results

We trained multiple machine learning models using a variety of machine learning algorithms including KNN, SVM, and Naive Bayes. Our dataset provided challenges because of its massive feature space. Furthermore, the adjective-noun pair probability scores for each design tended to hover extremely close to zero, with the exception of a few pairs. This made our feature space not only large, but also very sparse.

Cross-Validation

With 5-fold cross-validation, we trained our models using KNN and SVM with different parameters. The left figure shows results for the relationship between ROC and the number of neighbors. It indicates that as we increase the number of neighbors, training set accuracy increases. Training set accuracy was best, 78.56%, in our experiment, when we specified k = 9.





The right figure illustrates how different cost specification changes training accuracy in SVM with a linear kernel and gamma of 0.01. The results show that there is no difference in terms of training accuracy. Training accuracy remained at 72.24%, regardless the cost parameter value for SVM with a linear kernel.

Finally, we used Naive Bayes to train our training set with 5-fold cross-validation. Because the “caret” package currently does not provide Naive Bayes, we used the “e1071” package to implement it. Our Naive Bayes returned 68.54% accuracy on our training set.

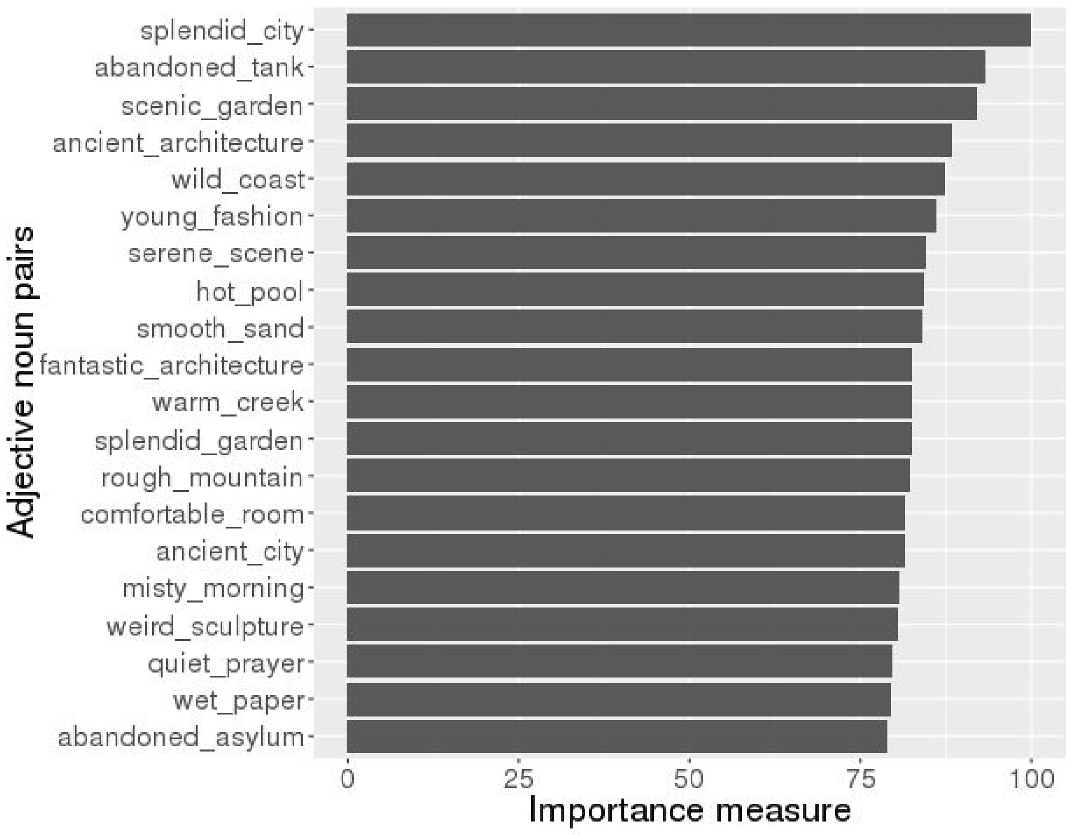
Testing on Our Test Set

In sum, our results for the test set show that the Naive Bayes learner's accuracy is slightly better than the other two algorithms. The table below shows the summary of accuracy rates for each algorithm. This suggests that the KNN and SVM learners may be overfitting more on the training set compared to Naive Bayes.

Learner	**KNN**	**SVM**	**Naive Bayes**
Accuracy	51.72%	50.58%	53.62%

Findings

Using variable importance evaluation functions, we looked at which features played a major role in predicting successful designs. *Figure 3* shows 20 features with highest important scores. Unfortunately, it is difficult to make a general statement about the top 20 list of features, but conventional adjective noun pairs such as beautiful flower, clean water and cute cat are not on the list. This may imply that some novel or unique concepts can help designs gain a higher vote score. This may also be an artifact of the fact that many t-shirt designs are abstract in nature and may not be easily categorized by image recognition software.



Limitations and Future Work

One of limitations of our models is that we could not find good feature selection criteria to reduce the number of features. Given the size of our training set, we wanted to reduce the number semantic word pairs to delete noisy pairs that did not provide helpful information. But we did not do so because we wanted to determine which word pairs were most predictive of t-shirt success. We were also unsure of how many word pairs to keep. In future work we would like to find a proper means by which to reduce features.

Another limitation of the project was the variable t-shirt design images. Some of the images showed the designs on a t-shirt, some showed the designs on a shirt with a human model, and some showed a combination of the design on a model plus a close-up crop. Presumable, the different presentation styles influenced the output of the image recognition software. In future work, we would standardized the image set or divide the dataset to determine whether the image styles have a strong influence on DeepSentiBank's output.

Member Contributions
Kyosuke Tanaka: Data Collection, Implementation of models in R, Report writing
Mark Díaz: Image data extraction, Dataset combining, Website, Report writing