Text Classification

COMP 551: Applied Machine Learning
Kaggle Team: Chris-HP

Christopher Glasz
christopher.glasz@mail.mcgill.ca
260720944

Truong Hoai Phuoc
phuoc.truong2@mail.mcgill.ca
123456789

I. Introduction

We are to predict participation and performance of past participants of the Montreal marathon in the 2016 version of the event using linear learning models.

II. Problem Representation

We chose to represent the text data as vectors of TF-IDF scores. We initially implemented our own code to calculate these feature vectors, but eventually turned to the NLTK library to produce them for us, to speed up the calculation time.

At the outset, we had planned on using the 2,000 most common unigrams and the 200 most common bigrams as features, but this proved to be ineffective. We improved our classification accuracy (at the cost of feature preprocessing time) by increasing to 90,000 unigram and 10,000 bigram TF-IDF scores, for a total of 100K features. Along the way we tried 22K and 55K features (both with a 10/1 unigram-bigram ratio), and found that the performance of our models just continued to increase as we added more features, so we settled on a nice even 100K, as that's just about how many unique terms exist in the training corpus.

Our performance was further improved by lemmatizing the input tokens using NLTK's WordNetLemmatizer. This prevented terms carrying the same information (e.g. "authenticate" and "authenticated") from being treated as two separate features. By tagging words by their part of speech before passing them to the lemmatizer, we were able to further improve our features (by reducing terms like "is", "are", and "am" to their common meaning of "be"). This process greatly increased the computation time, but was well worth it, as we only needed to calculate the feature vectors once.

Another decision we made over the course of the project was from where to extract the features. Early on, we were very strict about not using the text in the test corpus to select features, as we did not want to inadvertently influence the training of the model with information from set-aside data. However, as we increased the number of features, the difference between the set of most common words in the training corpus and in the combined training and testing corpora became negligible. Eventually we decided to produce 4 datasets:

1) *X_trn*: 80% of the training examples, with features drawn only from that corpus
2) *X_val*: The remaining 20% of the training examples, with features drawn only from that corpus
3) *X_all*: All examples in the training corpus, with features drawn from the words in both corpora
4) *X_tst*: All examples in the test corpus, with features drawn from words in both corpora

X_trn was used to find optimal hyper-parameters for our models, and used to train before prediction on the validation set. *X_val* was our validation set, and was only used to calculate our validation accuracy. *X_all* was used for final training before predicting labels on the test set, and was also the set over which we performed cross-validation. *X_tst* was, as one would expect, the dataset used for predicting labels on the test set.

All of these sets are saved as compressed sparse row matrices to save space. They can be read by using the `load_sparse_csr` function in `utils.py`

III. Training Method

A. Linear Classifier

B. Nonlinear Classifier

IV. Results

A. Logistic Regression

Our first serious attempt at a solution was the first scheme in the last section (training and validating in 2014, testing on 2015). We used 30-fold cross validation to chose alpha and lambda parameters with all thirteen features available. We got 90% accuracy on the training set (2014) and 94% accuracy on the test set (2015). Too good to be true indeed. We tried to learn on the same data using Scikit Learn and got the same output. Taking a look a little bit closer, we remarked the precision was incredibly low (around 1-2%) on both training and test set.

We predicted almost only class zero (no participation).This can be explained by the severe imbalance in our data : the return rate of participants from the last two years is of only around 5% to 10%. The logistic regression failed to learn the distribution of the minority class (one). We tried many subsets of our features (inspecting the weights to see which ones could matter most), and even polynomial transformations of our features. Nothing could get our precision up.

We arrived at the conclusion that we had to counter this severe imbalance in order to be able to learn the minority class. We tried three different schemes :

- Lower the decision boundary (e.g predict $y = 1$ for $\sigma(Xw) \leq a$ where $a \leq 0.5$). We tried cross-validation on different values of a, but it only reduced the accuracy without getting the precision above 50
- Undersampling the majority class. But we were left with a few hundred examples only and the functions we learned overfit severely.
- Oversampling the minority class. We reproduced randomly examples from the minority class in order to change the proportion of our training set. Obviously we left the test set untouched.

The third scheme worked best. We observed a trade-off between accuracy and precision of ones. As we increased the proportion of class one examples in our data set artificially, the precision of ones got better (reaching around 75% in training set and 60% in test set with a data set with equal proportions of class one and zero). But as a counterpart, the accuracy got worse (down to around 67% in training and test set with the same data set).

We tried to reach a sweetspot in this tradeoff. But we came to the conclusion that it was not worthwhile in terms of accuracy to try to learn the class one too well. We introduce too much bias by oversampling, and losing a few points in precision for the class zero means losing a lot of accuracy. Since the true distribution will most likely be class zero heavy, we finally decided to go with no undersampling at all.

Inspecting the final weight vector chosen for prediction, large posive weights on past participations to the Montreal marathon confirm our initial intuition that they should be positively correlated with the future participation.

B. Naïve Bayes

Below is the prediction output for 3 different randomly chosen training and testing sets. For each of the cases, training is based on 70% of the data, whereas testing is on rest of the 30%. For the first prediction result, False Positive Rate (FPR) is 16.09% and True Positive Rate (TPR) is 82.21%. FPR and TPR for the secoond table is 17.74% and 80.80% respectively. For the last table FPR and TPR is 17.32% and 81.72% respectively. Since, for each of the prediction results, TPR is significantly higher that FPR, we can come to the conclusion that our Naive Bayes model is a good choice for prediction of participation.

TP = 698	FN = 151
FP = 284	TN = 1481

TP = 682	FN = 162
FP = 162	TN = 1456

TP = 702	FN = 157
FP = 304	TN = 1451

C. Linear Regression

The validation results for regression were reasonably good. The linear model learned a very large positive weight on participants' average time, as expected, and a significantly smaller positive weight (10% of the size of the largest) on the time since their last marathon. The weight on participants' age was about half that (5% of the size of the largest).

Our mean squared error was 0.0094855 on the normalized training data and 0.01137544 on the normalized validation data. Our cross-validated error is also consistently in the same range, indicating that our model generalizes well and that our estimated error is accurate.

V. Discussion

A. Classification

We are confident about the accuracy of our logistic regression prediction, but we do not have the same kind of confidence about the precision of this prediction. The Naive Bayes should give a better precision, but probably a lower accuracy, given that the return rate of runners should be low as it has been in the past few years.

B. Regression

We are confident that the times predicted for the 2016 Montreal Marathon will be fairly accurate. The biggest detractor from our results is a lack of additional data. In particular, long distance running times are highly dependent on weather and temperature. It would have been possible to get this data for previous years, but we did not feel comfortable with trusting a weather forecast from two days before the event, so our trained model would be missing parameters for prediction.

We were surprised by how well simple regression works on this data, but are also confident that other model types would work much better. A deep neural net, for example, would be able to learn complex relationships between features, allowing us to add other useful features that would impact the linear regression model negatively (number of incomplete marathons, number of marathons attended each year, etc.)

VI. Statement of Contributions

A. Christopher Glasz

I was primarily responsible for the linear regression implementation and analysis, but also contributed a significant portion of the codebase that each model was built on, and the translation of our intermediate representation of the data to our final datasets. Sumana and Theo are relatively new to Python, so a lot of my contribution involved bugfixing and working to make the code more "Pythonic." I worked closely with Sumana to get her implementation of the Naïve Bayes classifier implemented, and also helped Theo with speeding up the logistic regression function.

B. Theophile Gervet

My main responsibility was the logistic regression implementation and analysis. I wrote the first version of the code to sanitize the data and convert it to the intermediate representation, before Chris perfected it. I also worked on cross-validation and the design on the learning scheme for the logistic regression, and implemented the cross validation algorithm we've been using for classification.

C. Sumana Basu

I was primarily responsible for the Naive Bayes implementation and analysis. I also worked closely with Chris and Theo to come up with the features we kept for the intermediate representation.

We hereby state that all the work presented in this report is that of the authors.