School of Informatics and Computing

Indiana University

Applied Machine Learning

Term-project

Yatin Sharma (yatins@indiana.edu)
Diksha Yadav (yadavd@umail.iu.edu)

This report contains two tasks performed:
Task 1:Predicting gender of the person from his/her tweets.
Task2: Comparison of algorithms' performance on balanced and Imbalanced dataset.

Task1: Predicting gender of the person from his/her tweets.

Abstract

We attempt to identify the gender of the tweet's author using linguistic evidence. We use twitter's data to construct dataset labeled with gender and explore different classifiers on this dataset to predict the gender of uncharacterized Twitter users. Prediction accuracies up to 94% are achieved suggesting the applicability of these classifiers to gender pre[1]diction

Overview and motivation:

The strategy was to build train an algorithm that could predict the gender of the twitter user, using methods that could be used by online recommender systems to provide targeted advertisement. For e.g.: If we are able to predict the user to be a female our recommender system could suggest female oriented items to her or at the least refrain from suggesting male products to her. This project originated as an International competition on author profiling by PAN1 and is closed now.

Data

The dataset used in this project is constructed from twitter, a social media platform whose users post short messages or tweets. The data set available to us consisted of XML files with each XML containing a series of document lines with URL to the tweet and an empty space for the content that would be populated by downloading the text of the tweets using html requests. In total we had ~ 42,000 tweets from 450 users with labeled gender (average 1000 tweets per user).This data was difficult to process because we had to parse through each of the xml files separating by XML elements to get useful text/tweets out of it.

Preprocessing

Tweets are often tagged with some kind of meta-data, including time-stamps, icons, images etc. Being user input, they also exhibit noise such as repeated words or spelling mistakes. To handle all this we restricted ourselves to only textual part of the tweets and also got rid of the noise by following pre-processing steps:

- URLs in all the tweets were replaced the string URL
- All the tweets were converted to lower case.
- Everything except alphanumeric characters was removed.

Features

We tried two techniques to classify the user according to the gender: Text-Based and Style based classification.

Text-Based Classification

In this approach the vocabulary of the tweet forms the feature set. We take the set of distinct words in each tweet to be the feature set. Our features does not count the

[1] http://pan.webis.de/clef16/pan16-web/author-profiling.html

number of occurrence of each word because initial experiments with features containing count values showed no significant improvement in performance. So our features are quite simple with each feature indicating absence or presence of the particular word

Style based features

In this approach we extracted stylistic features from the tweets and used them as our feature set. Average tweet length, Word length are some of the stylistic features that were extracted.

Table 1. showing Stylistic features used in the experiment

Features	**Description**	**Possible feature values**
Tweet length	Average tweet length	Low, Medium, High
Average word length	Average word length	Low, Medium, High
Distinct words	No of distinct words	Low, Medium, High

Above mentioned numerical features were converted into categorical features as per following logic:

Classification of Low/Medium/High was based on following logic:

Low→Anything less than 25 percentile of entire data.

Medium→Anything between 25 percentile and 75 percentile of entire data.

High→Anything greater than 75 percentile of entire data.

We also built a regular expression tagger that chooses a part of speech tag for each word in the tweet. Tagger was built by us by using the last 1,2,3 suffixes of each word.

For example:

If word was "entertaining", the last 3 suffix would give us "ing" which could be classified as an adjective or a verb. Once we have the most common suffixes, we train our classifiers to find out which suffixes are most informative in the prediction task.

Experimental Setup

We performed our experiment using following classifier types: Naïve Bayes, Support Vector Machines. We also ran the data set on our implementation of Bagging/Boosting Algorithm developed as a part of Programming Assignment#2. 10-fold cross validation was used in all experiments. We used different combination of features (such as only word unigram, or only tweet length, word unigram + tweet length, etc.) and analyzed their effects on the performance.

Results

Table 2 and 3 shows the result of each classifier on various combinations of features, all of which outperform the baseline method, which predicts each user to be Male. We used accuracy as our evaluation metric because our data-set was evenly distributed (230Males and 220 Females)

Table 2. Comparison of effect of Word Based feature on accuracy in each algorithm

Features	**Naïve Bayes**	**SVM**	**Our Bagging/Boosting algorithm**
Word Unigram	57.39%	74.29%	67.61%
Word Unigram + POS	58.25%	77.12%	68.4%
Word Unigram + average word+ No. of distinct words	58.96%	74.76%	65.72%

Table 3. Comparison of effect of style based features on accuracy in each classifier

Features	**Naïve Bayes**	**SVM**	**Our Boosting/Bagging algorithm**
Part of speech tagging (POS)	51.88%	54.00%	58.39%
Average Word length	56.60%	50.47%	56.7375%
No. Of Distinct Words	51.41%	51.41%	60.283%
Average + Distinct	60.14%	60.37%	60.283%

Since SVM classifier gave the highest accuracy, we further explored the SVM Classifier by trying different Kernels. Finally, maximum accuracy of 94.33% was achieved by using PUK (Pearson universal kernel) kernel function. In style based classification, SVM classifier achieved best results with an accuracy of 60.37%.

Conclusion

In this project we analyzed different configuration of classifiers for predicting the gender over a large twitter data set. The experiment also led to finding that word usages as well as writing style of users of opposite sex vary significantly. Table 4 shows a sample of discriminative words among men and women.

Table 4. Word usage according to the gender of the user.

Male	**Female**
Should	please
google	people
people	stories
really	credit

It is apparent that males tend to use more decisive word where as females tend to use more possessive words. The style-based analysis also shows that, in general, women tend to use longer words than men do. This clearly shows that word usage in a tweet can be used to discriminate the gender of the user.

Task 2: Comparison Of Algorithms' Performances On Balanced And Imbalanced Dataset

Abstract

In this project, we have compared performances of some selected algorithms on balanced and imbalanced datasets. We have used feature selection and under sampling in our balanced and imbalanced datasets respectively to improve the result.

Introduction

We have taken two datasets:
1) **Loan Dataset** (*balanced dataset*) - from lending club website.
2) **Credit Card fraud Dataset** (*highly imbalanced dataset*)- from UCI website.

- We have started with the loan dataset, performed **"feature selection"** using trial and error and finalized features giving best performance metric.
- Then, we applied Decision Tree, Bagging, Boosting, Support Vector Machine, K-nearest neighbor, Logistic Regression and Naïve Bayes classification algorithms on the finalized loan dataset and compared their performances.
- With imbalanced dataset, we have performed **"under sampling"** to increase the percentage of minority class from 2% to 50% then applied Random Forest, Decision Tree, Naïve Bayes, K Nearest Neighbor, Bagging and boosting classification algorithms and compared their performances.

Dataset Description

Loan Dataset

Before feature selection:
- Initial number of features – 68
- Initial record count – 122608

After feature selection:
- Selected features – 11
- Length of training set –35173
- Length of test set -3908

Credit Card Fraud Dataset

Before under sampling:
- Initial number of features – 23
- Initial record count – 30001
- Percentage of minority class in actual dataset- 2% approximately

After under sampling:
- Percentage of minority class sampling – 50% approximately
- Length of training set–13334
- Length of test set -5100

Methodology

After running 5 selected algorithms in Weka, we got following results,

Loan Dataset:

Performance Measurement using training and test set.

ML Algorithms	Decision Tree	Naïve Bayes	KNN 1	KNN 2	KNN 3	SVM	Logistic Regression
Accuracy %	99.31	95.05	86.42	97.02	93.24	100	99.87

Performance Measurement using 10 fold cross validation:

ML Algorithms	Decision Tree	Naïve Bayes	KNN1	SVM	Logistic Regression
Accuracy%	85.80	83.98	77.26	85.95	85.75

Analyses:
From above results, we analyzed that,

- SVM and Logistic Regression performed best on the loan dataset with highest accuracies.
- KNN (K=1) and Naïve Bayes performed worst on the loan dataset with lowest accuracies
- KNN with K=2 performed much better than KNN with K=1.
- Decision tree gave an average performance.

Credit Card Dataset

Since the credit card dataset was imbalanced, we performed under sampling on training set and then applied the following algorithms on this dataset. Also, as we know that accuracy is not a correct measure to test the performance of an algorithm in imbalanced dataset, we have computed other factors listed below.

Accuracy:
Accuracy = (TP + TN) / (TP + TN + FP + FN)

Precision:
Precision is the number of True Positives divided by the number of True Positives and False Positives.
Precision= TP / TP+FP
In this dataset, it tells how many positive cases (fraud cases) were correct.

Recall:
Recall is the number of True Positives divided by the number of True Positives and the number of False Negatives.
Recall=TP / TP+FN
In this dataset, it tells how many positive cases (fraud cases) were caught.

Area under ROC:
It gives the area under ROC curve.

F-measure:
The F-measure is the 2*((precision*recall) / (precision +recall)). It is also called the F Score or the F1 score.
F1 score conveys the balance between the precision and the recall.

Kappa Statistic:
It is described as the amount of agreement correct by the agreement expected by chance.
It compares an **Observed Accuracy** with an **Expected Accuracy** (random chance).

ML Algorithms	Decision Tree	Random Forest	Naïve Bayes	KNN 1	KNN 2	KNN 3
Accuracy%	86.07	68.82	70.09	71.03	72.92	70.11
Precision	0.98	1	1	1	0.99	0.99
Recall	0.86	0.68	0.69	0.70	0.72	0.69
Area under ROC	0.72	0.95	0.94	0.87	0.83	0.83
Kappa statistic	0.09	0.07	0.08	0.08	0.06	0.06

F-measure	0.92(0)
	0.12(1)
		0.81(0)
		0.11(1)
			0.82(0)
			0.11(1)
				0.82(0)
				0.11(1)
					0.84(0)
					0.09(1)
						0.82(0)
						0.09(1)

Analyses

- Since our data is imbalanced, there are many factors to observe before we can say which algorithm performed best on the dataset.
- According to our analyses, Decision tree should be considered as best performing algorithm on this dataset because,
  - ➢ In this prediction problem the most important point is to be able to catch maximum of the fraud cases i.e., algorithm which is giving maximum recall should be considered better than the other.

    - Precision also matters here but cost of predicting non fraud cases as frauds is lower than not being able to catch actual fraud cases. Hence, We will give more importance to recall than precision in this case.
    - Third most important factor here is kappa statistic; its value is too close to perfect with decision tree.
    - F-measure value for class 1(fraud cases) is highest with decision trees.
- Second best performing algorithm according to us is K Nearest Neighbors but with low value of K, which is obvious, as our data is imbalanced so large value of K will definitely give bad result.
- Performance of Naïve Bayes and Random Forest is very similar on this dataset with almost equal values for different [2]factors considered by us.

Conclusion

After analyses of both balanced and imbalanced datasets with different machine learning algorithms, we have come to the conclusion that comparing the performance of different algorithms on a balanced dataset is much easier than comparing it on imbalanced dataset because for former we just need to look at the accuracy and give the best algorithm tag to one giving highest accuracy but when it comes to making decision for imbalanced dataset, we need to look at a number of factors including precision, recall, F-measure etc. along with the own parameters of the algorithms. Still there is no hard and fast rule even if we consider all these factors. It depends on the kind of problem we are dealing with. For example, in our problem (credit card fraud detection), we have considered recall as the best factor because the cost of missing a fraud case is very high. In other problems, other factors can be more important. So, having a good understanding of the problem along with the various costs related to it is the first step before moving to the analyses part.

NOTE: We have attached screenshots of Weka outputs along with code folder uploaded on canvas.