Machine Learning Engineer Nanodegree

Capstone Proposal - Sentiment Analysis of Yelp review dataset from Kaggle

January 11, 2017

Proposal

Domain Background:

Natural language processing (or NLP) serves numerous use cases when dealing with text or unstructured text data. My project will examine Yelp review dataset from Kaggle.https://www.kaggle.com/c/yelp-recsys-2013#description

The application of sentiment analyser are endless with twitter tweet analysis, Amazon reviews , Hotel reviews , Restaurant reviews. Given just words we would like to determine if the statement is positive or negative. The crux of the project is an algorithm to predict ratings and the requirement is to create a model to predict the rating a user would assign to a restaurant.

1. Opinion Mining and Sentiment Analysis
    a. http://www.cs.cornell.edu/home/llee/omsa/omsa-published.pdf
2. Evolution on sentiment analysis
    a. http://www.cs.cornell.edu/home/llee/omsa/omsa.pdf

Problem Statement

Attempt to classify Yelp reviews based on text content. Each observation is review by a particular user. Higher the star better the review

We aim to tackle the problem of sentiment polarity categorization, which is one of the fundamental problems of sentiment analysis.

This type of problem is challenging because you usually can't solve it by looking at individual words. No single word is going to tell you whether the review is positive or negative. And that is the main reason for choosing this problem, I believe that helping to solve this task might bring some enlightenment to other critical NLP tasks.

My goal is to build a sentiment analyser model that predicts whether a user liked a local business or not, based on their review on Yelp.

Datasets and Inputs

https://www.kaggle.com/c/yelp-recsys-2013#description

Our data is a detailed dump of Yelp reviews, businesses, users, and checkins for the Phoenix, AZ metropolitan area. We will use the Review dataset and attempt to classify them

Our data contains 10,000 reviews, with the following information for each one:

Data Format

{

'type': 'review',

'business_id': (encrypted business id),

'user_id': (encrypted user id),

'stars': (star rating),

'text': (review text),

'date': (date, formatted like '2012-03-14', %Y-%m-%d in strptime notation),

'votes': {'useful': (count), 'funny': (count), 'cool': (count)}

}

Summary Statistics:

In the training set:

- 11,537 businesses
- 8,282 checkin sets
- 43,873 users
- 10000 reviews

In the testing set:

- 1,205 businesses
- 734 checkin sets
- 5,105 users
- 10000 reviews to predict

Review

Our data contains 10,000 reviews, with the following information for each one:

business_id	date	review_id	stars	text	type	user_id	cool	useful	funny

Since we are trying to predict how a user will rate a business, the only information in a Review object is the user_id and business_id.

Ratings	Count of Ratings
1	749
2	927
3	1461
4	3526
5	3337
Grand Total	**10000**

We split into a training and a test set using train_test_split from Scikit-learn. We will use **30%** of the dataset for testing.

Solution Statement

NLTK is a standard library in Python library for text processing, which has many useful features. The objective of this project is to apply sentiment analysis techniques(NLP) on the Yelp reviews and assess whether they can correctly identify the reviews as positive or negative.

For this project I'm proposing a Naive Bayes classification problem, and I am going to use the Yelp Reviews dataset from a Kaggle competition. The data is a detailed dump of Yelp reviews, businesses, users, and checkins for the Phoenix, AZ metropolitan area. We will use Multinomial NB classifier to predict a given review is positive or negative , which is a classic NLP task. We can evaluate the performance using precision and recall from the confusion matrix and use Pandas, NLTK, scikit -learn and seaborn libraries as a part of our analysis

Benchmark Model

This is a dataset available in Kaggle and lot of results have been published. Teams have used rule based Sentiment analysis tools and various natural language tool kits. We will use a NB classifier and compare with other models such as Sentiwordnet . Recently, teams have experimented with end-to-end deep learning solutions.

For benchmarking, against a secondary algorithm, Logistic Regression is an algorithm that can be trained as a classifier and uses a linear decision boundary as a means of classifying a set of features.

Logistic Regression is derived from Linear Regression and has a very similar cost function. Additionally, Logistic Regression is a great first classification algorithm to learn and can be used to classify multivariate data as well as fit polynomial terms to find a decision boundary.

Evaluation Metrics

The evaluation metrics for this model will be the confusion matrix, precision,

recall, and error rate.

The terms are defined as follows:

• True positives: Entries that are correctly labelled

• False positives: Entries that a wrongly identified with a given label

• False negatives: Entries for a given label that are wrongly identified with

other labels. In the general case, a confusion matrix is simply a matrix illustrating the mapping from the true labels to the predicted labels. Elements along the diagonal represent a correct classification, whereas the off-diagonal represent a misclassification.

Precision is the result of the number of true positives divided by the sum of true positives and false negatives. This can be given by the following equation,

precision = tp / (tp + fp) where tp and fp stand for true positive and false positive respectively.

Recall is the result of dividing the true positives by the sum of true positives with false negatives. This can be given as follows,

recall = tp / (tp + fn) where tp and fn stand for true positive and false negative respectively. From this, the error rate or rate at which the classifier wrongly classifies the samples can be derived,

error rate = 1 - (tp / (tp + fn)) = fn / (tp + fn) which is also known as the false negative rate. The error rate per digit could be derived similarly, but on a digit-by-digit basis, as given by the confusion matrix above.

The error rate as well as all the other metrics discussed in this section will be calculated using the SciKit-Learn metrics.classification_report and metrics.confusion_matrix methods.

Project Design

This project will follow a typical machine workflow, starting from the dataset acquisition, then preprocessing, model generation, and then an evaluation and optimization process

Load data and perform exploratory analysis

1. Read the yelp.csv file , perform an exploratory analysis (i.e, reading info(), describe methods() and number of entries in the csv etc) using seaborn. Identify if there is co-relation between text length and reviews (stars).
2. Check if there are correlations between columns (cool, useful and funny) and create a heat map.This will provide correlation between each column w.r.t other column.

Pre-processing steps:

- **a. Text- Preprocessing:**
  - a. The simplest way to convert a corpus to a vector format is the **bag-of-words** approach, where each unique word in a text will be represented by one number
- **b. Tokenization:**
  - a. We can use Scikit-learn's CountVectorizer to convert the text collection into a matrix of token counts.
- **c. Improving accuracy:**
  - a. To improve accuracy, I am also looking at including Parts of Speech Tagging, Stopword Removal, Stemming.

Classification task:

Grab reviews 1 and 5 stars so that we can predict if a review is good or bad. Use bag of words approach to convert each word to a number , remove stopwords, punctuations using NLTK library . We will need a feature vector to perform the classification task.

a. Data to vectors: We can use Count vectorizer to convert text into matrix of tokens
b. Use MultinomialNB (Naive bayes) to train and classify and compare against logistic regression
c. Use classification report and confusion matrix to evaluate the result

I am planning to use logistic regression to contrast against Naive Bayes. Logistic Regression is an algorithm that can be trained as a classifier and uses a linear decision boundary as a means of classifying a set of features. The model cannot generate new data from a distribution, however, it can be used to predict the value of an independent variable from a single or multiple dependent variables

What we have done and how can we improve the model?:

References:

1. Evolution on sentiment analysis
    a. http://www.cs.cornell.edu/home/llee/omsa/omsa.pdf
2. Opinion Mining and Sentiment Analysis
    a. http://www.cs.cornell.edu/home/llee/omsa/omsa-published.pdf
3. scikit-learn - working with text data
    a. http://scikit-learn.org/stable/tutorial/text_analytics/working_with_text_data.html
4. https://radimrehurek.com/data_science_python/