Machine Learning Engineer Nanodegree

Capstone Proposal

Garrison Jensen

February 21st, 2017

Domain Background

Is it possible to predict the star rating of an Amazon review given the text of that review? I want to try to answer that question using a collection of 400,000 Amazon reviews of unlocked mobile phones as my dataset. I think it is interesting to see if the sentiment of a review can be decided from the text of the review itself.

The problem of determining the sentiment of text is not new. Research was done to try to extract sentiment from movie reviews [1]. They tried to predict whether the review was positive or negative based on the text. They found that one of the main difficulties with sentiment extraction is that it requires more understanding of the text than traditional topic-based categorization. Topic-based categorization relies on finding keywords in text, however sentiment can be expressed more subtly.

Problem Statement

Given a collection of Amazon reviews for mobile phones is it possible to predict the star rating of that review? Solving this problem requires us to extract features from the review and then categorize the reviews into five categories. One potential solution is to use the TF-IDF[2] method for extracting features and a Bayesian model for learning.

Datasets and Inputs

PromptCloud[3] extracted 400,000 reviews of unlocked mobile phones from Amazon and provided the data on Kaggle.com[4]. The data was extracted on December, 2016.

The data set includes the following fields:

- Data Fields:

[1] http://www.cs.cornell.edu/home/llee/papers/sentiment.home.html
[2] http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html
[3] https://www.promptcloud.com/
[4] https://www.kaggle.com/PromptCloudHQ/amazon-reviews-unlocked-mobile-phones

- Product Title: Title of product
- Brand: Brand of phone
- Price: Price of phone
- Rating: Rating 1-5 stars
- Review text: The written description of the review
- Number of people who found the review helpful: Up votes of the review

The two fields I will focus on are the **Review Text** and **Rating** fields. My goal is to predict **Rating** by analyzing **Review Text**.

Solution Statement

To extract features from the text we can count the occurrences of words in the text and use the word-count pair as a feature. This is basically what the TF-IDF algorithm does, however it offsets words that appear often in all the samples. This is important because words that are very frequent in all the samples are probably not very important. We can then train a Bayesian algorithm on those features. If predicting five categories is too challenging I might have to simplify the problem by categorizing the reviews into fewer groups (For example, we could try two categories: low-rated 1-3 and high-rated 4-5).

Benchmark Model

A simple baseline model can be built using the 'bag of words' method for feature extraction and a simple linear regression for learning.[5]

Evaluation Metrics

I plan to calculate the F1 score to evaluate the model.

```
F1 = 2 * (precision * recall) / (precision + recall)
```

```
precision = True positives / (True positives + False positives)
```

```
recall = True positives / (True positives + False negatives)
```

Project Design

1. Explore the data

   First, I plan to explore the data. I want to see the distribution of star reviews (Are there more 5-star reviews than 3-star reviews?). I also want to look at the distribution of review lengths.

[5] http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

2. Clean the data

   We first need to remove punctuation, numbers, and common words from the text. Also, we should make all the text lowercase.

3. Extract features from the text

   Using the TF-IDF method we can create features to use for training.

4. Split the data

   We need to split the data into a training set and a testing set. I plan to use sklearns `train_test_split` method.

5. Train the model

   For my first attempt, I want to try the `GaussianNB` and `MultinomialNB` algorithms. As I iterate on the design I might try different algorithms.

6. Evaluate/compare models

   Using confusion matrices and F1 scores I will compare the models. Depending on the results I will try to find tweaks, or try different algorithms to iterate on the design. If the problem is too difficult, I've considered simplifying the problem by categorizing the reviews into fewer categories. Perhaps I could try categorizing reviews into lower-rated vs high-rated reviews. I'm not sure how difficult this problem is so I will wait until I've tried solving this more difficult problem before trying a simpler one.