Machine Learning Engineer Nanodegree

Capstone Project

Rakan Alkheliwi

Aug 8, 2018

I. Definition

Project Overview

Much of the delay in serving customers through customer service reps is due to the need for levels of support to be provided prior to reaching the correct level of support. For instance, if the user report a complex problem in an application on his Windows machine, a first level support in the form of a Windows support need to arrive and attempt to resolve the problem. If not, he will assign the ticket to the next level of support.

Often, organizations provide their employees with templates on how to resolve these problems. They would need to ask a couple of questions to analyze the problem before sending it to the appropriate level. This process can be automated and optimized which I will discuss in this report.

Most research on text classification uses SVM or naïve Bayes but the problem with these methods are data sparsity where data do not have a fully defined region of answer [1]. In this project, I will be utilizing deep learning methods to avoid this problem and to analyze the problem using the new trend.

Problem Statement

In this paper, I will perform a text classification analysis to articles. The purpose of this activity is to develop a potential Chabot that can from the user description and later interaction be able to arrive to the quickest solution and provide support with all the needed results from the agent analysis. I will be utilizing deep neural networks in this problem and use bag of words as a feature selection mechanism.

Metrics

The metrics is the ability of the agent to correctly classify testing data to be of the correct class or in other words accuracy. If the agent is able to predict with great accuracy what are the results we will be able to develop a potential Chabot that can solve users' problems. The result is in the form of probabilities where maximum probability indicate class since I am using a deep neural network.

II. Analysis

Data Exploration

The dataset has been obtained from UCI and it contains articles from twenty online newsgroups. The data has been cleaned out, however, the data is encoded in latin-1 instead of utf-8 that caused some problems in the analysis. To mitigate that, I encoded the data as latin-1 throughout my experimentation. In addition, the data contains a lot of sequence of numbers and special characters that are not English words. Thus, these characters need to be cleaned out and removed.

The data contain random words and are part of online newsgroups. Thus, we expect a lot of stop words in these articles. Therefore, I have attempted to clean out common stop words from these datasets prior to my actual analysis.

As part of the project, I attempted to analyze IT helpdesk tickets to classify customers into segments. A lot of processing has been conducted but it is not worth reporting, as the data didn't fit with the assumption and the problem context I attempted to pursue. However, the results of this analysis are provided with the code.

Simply, I segmented customers based on their frequency of satisfaction and length of time it takes to solve their problem. Best K-means model showed two segments; however, when we draw the graph it shows a linear curve between frequency of satisfaction and length of time, which just show that unsupervised learning is not a good predictor of customer class. Same results were obtained when we attempted to segment employees.

Below is a sample of the dataset:

Path:
cantaloupe.srv.cs.cmu.edu!crabapple.srv.cs.cmu.edu!fs7.ece.cmu.edu!europa.eng.gtefsd.com!howland.reston.ans.net!noc.near.net!uunet!pipex!bnr.co.uk!marble.uknet.ac.uk!uknet!gdt!aber!news

From: azw@aber.ac.uk (Andy Woodward)

Newsgroups: rec.motorcycles

Subject: Re: Cultural Enquiries

Message-ID: <1993Apr5.094451.8144@aber.ac.uk>

Date: 5 Apr 93 09:44:51 GMT

References: <1993Mar10.165837.3224@drycas.club.cc.cmu.edu> <C3pKw6.J3p@javelin.sim.es.com> <1993Mar17.115603.28712@aber.ac.uk> <Stafford-310393095530@stafford.winona.msus.edu>

Sender: news@aber.ac.uk (USENET news service)

Organization: University College of Wales, Aberystwyth

Lines: 39

Nntp-Posting-Host: 144.124.112.30

In article <Stafford-310393095530@stafford.winona.msus.edu> Stafford@Vax2.Winona.MSUS.Edu (John Stafford) writes:

>In article <1993Mar17.115603.28712@aber.ac.uk>, azw@aber.ac.uk (Andy

>Woodward) wrote:

>>

>> Two questions that fascinate me:-

> You are easily fascinated.

>

>> 1) Why are rednecks called rednecks?

> Why are you called a Welch?

> OK, it's because they are often south or southeastern farmers

> who's necks are permanently damaged from sunburn. The sun;

> you know what that is, it never sets on the British Empire

> and never shines in Wales.

>

This is a despicable LIE! It was sunny on 3rd July 1958 from 11.23am

to 11 37am. I made a note of it. Diaries are never wrong.

>> 2) Why do they ride Harleys?

> They don't. They drive in pick-up trucks and shoot bikers.

>

>> Please enlighten me. When I visited last, the only answers I got

>> were incoherent splutterings.

>   You deserve more?

>

>===

>John Stafford   Minnesota State University @ Winona

>   All standard disclaimers apply.

Do you, by any chance ride a Harley? (just a feeling...) How is your

neck? Calamine lotion is good, I'm told.

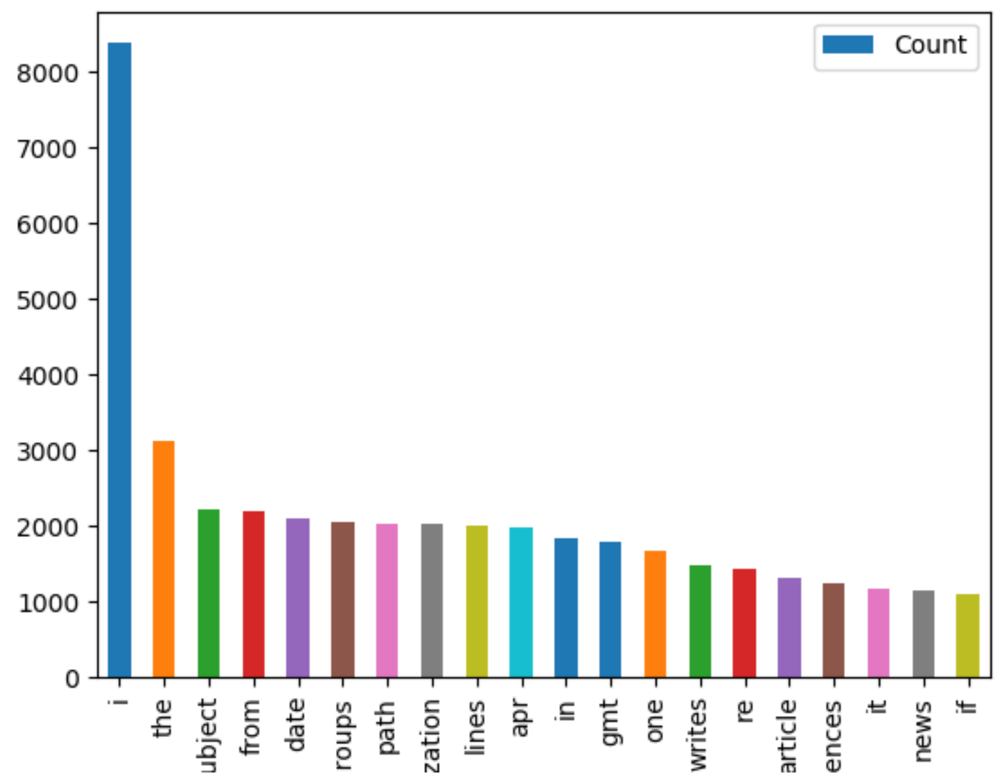
I am getting bored with winding up Americans. Its like bombing fish

in a barrel.

Haaaaaaaaaaave a Niiiiiiiiiiiiiice Daaaaaaaaaaaaaaaaaaaaaay

Andy

Exploratory Visualization



The graph above shows the distribution of words in our text corpus. As you see, a lot of words in our text corpus are common words which I tried to remove but still managed to sneak in. The most common word is ‘I’. The rest of the words are repetitive words with no value. As an improvement, we need a way to remove all of these words from the corpus in a more efficient manner.

Algorithms and Techniques

The algorithm I used is a combination of bag of words and deep neural networks. Bag of words is an algorithm where we count the number of occurrences of a word in a sentence regardless of its position. In the context of IT problems, the position of the word I would argue is not essential as problem descriptions are not usually verbose.

The other technique is deep neural networks. This algorithm is beneficial in discovering hidden patterns in the text; however, its biggest advantage in this instance is that it assigns probabilities rather than a score compared to Naïve Bayes algorithm. This is beneficial in determining what is the likelihood that the problem is either a network or a database problem but not an application issue for instance.

I used NLTK library to remove stop words and tensor flow to speed up the processing time for the training. However, due to graphical card limitation, I used CPU version of Tensor flow.

Here, we used RNN neural networks. Experimentation with the neural network was the name of the game. In neural networks, practice is ahead of theory. Thus, I based my solution on trial and error where I used tanh as an activation function, dropout layers, and a fully connected layer at the end.

Benchmark

One naïve way to determine if a problem is in a particular category is to see if the subject name is contained in the article or not. For instance, articles about sports will probably contain the word sports. This method is easier done for organizations dealing with applications problems, as the user will probably mention the name of the application in the article. However, there could be different reason for the problem such as a network problem or database problem. The benchmark algorithm had an accuracy of 0 so we had to improvise a new benchmark.

My benchmark is whether the algorithm performs better than random. If the algorithm has a score equal or less than 50%, then it's simply guessing the solution rather than solving it through an intelligent manner.

III. Methodology

Data Preprocessing

Preprocessing includes downloading NLTK API and tokenizing words contained in the articles. In the middle of this process, we had to decode articles in Latin-1 format. Also, we took out stop words contained in NLTK library and removed words that contain numbers or special characters. This process consumed a significant amount of time, which required the use of pickle library.

Implementation

I used NLTK, TFlearn, pandas and numpy as libraries. I started by tokenizing articles using NLTK library. I then counted all occurring words except for stop words and words that only contain alpha words. Then, I ran bag of words algorithms and built an array of classes that are 20.

I split the training and testing with a size of test equal to 33%. I then ran X and Y on a deep neural network using Tensor flow and TFLearn and using tanh activation function. This resulted in an array of probabilities the size of Y. I received around 95% accuracy but this could be due to overfitting as the testing result initially was 50%, which I ran it and got the class with highest probability.

Some of the challenges I faced is making the layers of neural network compatible. This is dependent on the size of the variables, which need to be paid attention to when reproducing this solution.

Refinement

Initial solution used 1000 epochs and batch size of 8. This resulted in overfitting the results as accuracy of test evaluation was 50%. So, I dropped to 500 epochs to undermine overfitting and increased batch to 16 from 8 for one hidden layer. This improved the result 5 points but not much. So, I further dropped epochs to 100, and made the batch size into 16. I changed the neural network to accept input then 64 entries, then 32 entries, then the final categorization. This increased the score to 62%.

IV. Results

Model Evaluation and Validation

The problem has an accuracy score on the test data that are 62%. This is a good result for a starter but it shows some limitations. The results would probably be better for a model that tries to predict IT problems rather than lifestyle problems as they lifestyle problems tend to be verbose with similar words while IT problems include unique words for each subject. In addition, the model gives probabilities for different results so we could use it to feed it to a reinformcent-learning agent that will decide next course of action.

However, we are long away from having a robust model. The problem with this model is mostly overfitting and the fact that is using a slightly dirty data. Thus, improvement is needed on this model.

Justification

Our A/B test is whether the algorithm performs better than random. So, our hypothesis is the following:

Null hypothesis: The algorithm is guessing with an accuracy test of 50%

Alternative hypothesis: The algorithm is making intelligent predictions with test accuracy higher than 50%.

The null hypothesis mean is 50% while our sample has 62% mean. The sample size is (test sample) equal to 2000/3 = 667. From the blog below:

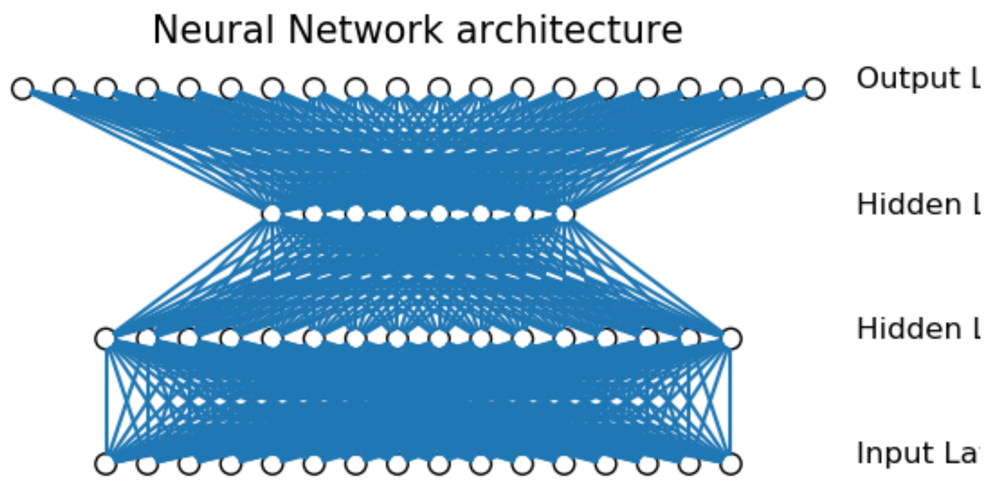
https://machinelearningmastery.com/report-classifier-performance-confidence-intervals/

Standard deviation is equal to the following where error equal to 1-.62 = .38. 1.96*sort((error * (1-error))/n) = .04. Plug those numbers to get a z-score of 77.48 which is in rejection region. Thus, we reject null hypothesis and accept alternative hypothesis that our algorithm performs better than random.

V. Conclusion

Free-Form Visualization

Below I am displaying an approximate architecture of the neural network. I accepted a text of tokens the length of more than 200 entries. Then, I used a batch size of 8 but entered a hidden layer that accept 64 then another hidden layer that accept 32. Finally, the article will be classified as one of 20 classes.



Reflection

This wasn’t an easy problem to tackle and my expectations was that I will be able to develop a good model using unsupervised learning and reinformcnet learning with the IT tickets. However, I realized from my unsupervised learning that the data doesn’t provide any good results and the problem I’m trying to solve using reinformcnet learning is not worth the time as management would usually choose the correct person to assign the ticket to with ease.

But the text classification problem was interesting and I arrived at the idea of using it and Chabot prior to reading about catboats and after submitting my proposal. Thus, I would like to get a chance to skip the original proposal and focus on text classification.

A problem with neural networks is that they require a high end GPU for processing which I lacked in my machine. However, I learned a great deal about neural networks doing this algorithm.

Improvement

The choice of implementing a deep learning algorithm was done for the sole purpose of challenging myself in this new field to me. However, from the articles I read it seems that Naïve Bayes algorithm is a better solution to this. So potentially, I could develop a neural network that utilizes a combination of Naïve Bayes, bag of words, DNN or CNN to obtain a probability for which problem and solution the problem (or in this context article) belongs to. Also, next step is to develop a reinforcing learning algorithm that will take the problem or article and based on it interact with user and ask him questions to determine correct action or group to send the problem to.

References:

[1] Siwei Lai, Liheng Xu, Kang Liu, Jun Zhao, Recurrent Convolutional Neural Networks for Text Classification