Machine Learning Engineer Nanodegree

Capstone Project

Zhao Mingchuan
April 21st, 2017

I. Definition

Project Overview

Attitudes and opinions are always been an important piece of information for most of us during the decision-making process. Widespread Internet makes it possible to find out opinions and experiences in the vast pool of people which are not someone we are familiar with. And nowadays, more and more people search on the Internet to find recommended auto mechenics, popular publications and famous attractions. The user reliance on online advice and recommendations is one of the reasons why we need good and explicit online opinion service. Besides the consumption of goods and service, public issues like political information is also another motivation. For example, in a survey of over 2500 American adults, Rainie and Horrigan studied the 31% of Americans that were 2006 campaign internet users which means nearly 60 million people gathered information about the 2006 elections online. Opinion mining or sentiment analysis is a relatively new area which deals with the computational treatment of opinion, sentiment and subjectivity in text. In this project, I am trying to implement some machine learning techniques on the Internet Movie Database in order to achive a good sentiment prediction on the movie reviews.

The dataset IMDB comes from kaggle(https://www.kaggle.com/c/word2vec-nlp-tutorial/data (https://www.kaggle.com/c/word2vec-nlp-tutorial/data)) which contains three data sets labeledTrainData, unlabeledTrainData and testData. Here is an example of the data in labeledTrainData below:

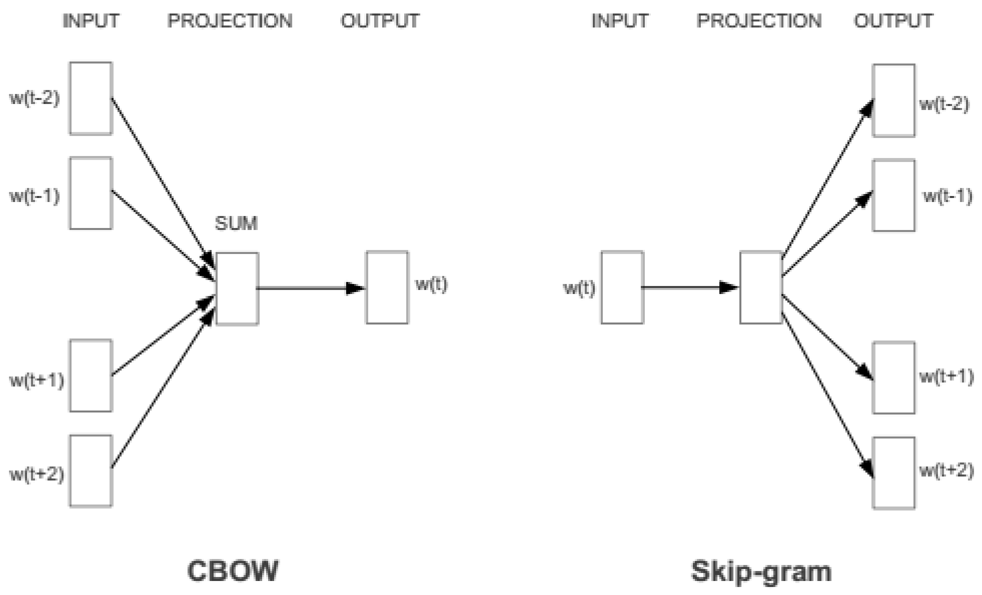
id sentiment review "5814_8" 1 "With all this stuff going down at the moment with MJ i've started listening to his music, watching the odd documentary here and there, watched The Wiz and watched Moonwalker again. Maybe i just want to get a certain insight into this guy who i thought was really cool in the eighties just to maybe make up my mind whether he is guilty or innocent. Moonwalker is part biography, part feature film which i remember going to see at the cinema when it was originally released. Some of it has subtle messages about MJ's feeling towards the press and also the obvious message of drugs are bad m'kay.Visually impressive but of course this is all about Michael Jackson so unless you remotely like MJ in anyway then you are going to hate this and find it boring. Some may call MJ an egotist for consenting to the making of this movie BUT MJ and most of his fans would say that he made it for the fans which if true is really nice of him.The actual feature film bit when it finally starts is only on for 20 minutes or so excluding the Smooth Criminal sequence and Joe Pesci is convincing as a psychopathic all powerful drug lord. Why he wants MJ dead so bad is beyond me. Because MJ overheard his plans? Nah, Joe Pesci's character ranted that he wanted people to know it is he who is supplying drugs etc so i dunno, maybe he just hates MJ's music.Lots of cool things in this like MJ turning into a car and a robot and the whole Speed Demon sequence. Also, the director must have had the patience of a saint when it came to filming the kiddy Bad sequence as usually directors hate working with one kid let alone a whole bunch of them performing a complex dance scene.Bottom line, this movie is for people who like MJ on one level or another (which i think is most people). If not, then stay away. It does try and give off a wholesome message and ironically MJ's bestest buddy in this movie is a girl! Michael Jackson is truly one of the most talented people ever to grace this planet but is he guilty? Well, with all the attention i've gave this subject....hmmm well i don't know because people can be different behind closed doors, i know this for a fact. He is either an extremely nice but stupid guy or one of the most sickest liars. I hope he is not the latter."

The data in labeledTrainData includes three columns which are id, sentiment and review. The data in unlabeledTrainData only contains id and review columns and the TestData is the dataset for testing which contains id and review. I will use labeledTrainData to train the classifier, the unlabeledTrainData is used to build word2vec model and the TestData is used to make predictions.

Problem Statement

In this project, I am trying to solve a problem about sentiment analysis on IMDB. IMDB refers to the Internet Movie Database which contains the reviews and sentiment on Internet movies. The reviews in labeledTrainData, unlabeledTrainData and TestData are selected from IMDB, the reviews in which are all on different movies. In order to reach the goal and make sentiment predictions on the reviews I need to first transfer the text data into digits so I choose two popular models which are bag of words model and word2vec model. Bag of words model is a model which is a simplifying representation used in natuaral language processing, in this model a text is represented as the bag of its words, disregarding grammar and word order but multiplicity. For example, suppose we have two text documents, after going through several data cleaning and text preprocessing approach we will get two text contains only words. The preprocessing procedure for each review includes removing HTML tags using BeautifulSoup, removing non-letters using re package in python, removing stop words by nltk and merge the words with the same stem using WordNetLemmatizer and SnowballStemmer in nltk. Then we pick out all the words in the two texts and count the frequency of every word appears in the two texts, then we go back to the two texts and mark every word with the frquency number then we will get two vectors and the elements in each of which represents the frequency of the word. Bag of words model is simple and easy to comopute but it loses the order of the words.

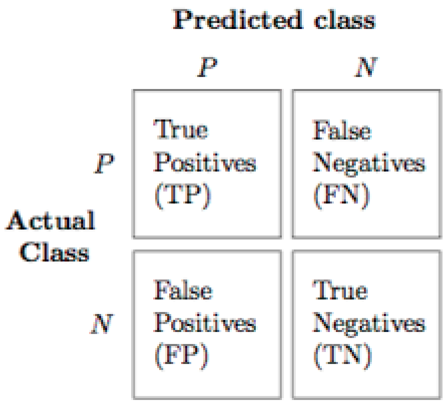
Another approach we will use is called word2vec, it is a model inspired by deep learning,which is a neural network implemetation to learn distributed representatins for words, it learns quickly relative to other models like recurrent neural network. Word2vec does not need labels to create meaningful representations. It takes a text corpus as input and produces the word vectors as output. It first constructs a vocabulary from the training text data and then learns vector representation of words. There are two main algorithms in word2vec which are continuous bag-of-words(CBOW) and continuous skip-gram. In CBOW the model uses the current word to predict the surrounding window of context words. In skip-gram architecture, the the model uses the current word to predict the surrounding window of context words.The skip-gram architecture weights nearby context words heavily than remote context words. In this project we will use skip-gram architecture.



After transfer the text data to digital data, I will push the data into some classifers implemented by some machine learning algorithms which includes randomforest, naive bayes and support vector machine. The reason to choose them will be discussed below and after the output predictions are made, I will compare the output of each classifier and find the best one of them.

Metrics

In this project, I use confusion matrix to evaluate the predictions. Confusion matrix is a technique for summarizing the performance of a classification algorithm. Classification accuracy can be misleading if you have an unequal number of observations in each class or if you have more than two classes so confusion matrix can give a better idea of whether the model performs good or bad. Confusion matrix is a matrix consists of four parts, true positive(prediction and fact are both positive),true negative(prediction and fact are both negative),false positive(prediction positive but fact is negative) and false negative(prediction is negative but the fact is positive). In this project, we will calculate the TP,TN,FP,FN and calculate the accuracy score by using (TP+TN)/(P+N).



For this project, the labels include two symbols 0 and 1 which indicates the negative attitude and positive attitude, so the output predictions can only be true positive, true negative, false positive and false negative and we will evaluate the result based on true positive and true negative, so using a confusion matrix to evaluate the predictions is proper and enough.

II. Analysis

Data Exploration

The data used in this project comes from three tsv files which are labeledTrainData, unlabeledTrainData and TestData. The data in the tsv files will be read into pands dataframe form. There are 25000 rows in labeledTrainData and TestData, each row contains three columns which are id(the id number of review),reveiws(the reviews of the movie) and sentiment (positive or negative sentiment on the movie reviews). In unlabeledTrainData, there are 50000 rows and each row contains two columns which are id and review. Here is an example of the input data in labeledTrainData and TestData:

id sentiment review "5814_8" 1 "With all this stuff going down at the moment with MJ i've started listening to his music, watching the odd documentary here and there, watched The Wiz and watched Moonwalker again. Maybe i just want to get a certain insight into this guy who i thought was really cool in the eighties just to maybe make up my mind whether he is guilty or innocent. Moonwalker is part biography, part feature film which i remember going to see at the cinema when it was originally released. Some of it has subtle messages about MJ's feeling towards the press and also the obvious message of drugs are bad m'kay.Visually impressive but of course this is all about Michael Jackson so unless you remotely like MJ in anyway then you are going to hate this and find it boring. Some may call MJ an egotist for consenting to the making of this movie BUT MJ and most of his fans would say that he made it for the fans which if true is really nice of him.The actual feature film bit when it finally starts is only on for 20 minutes or so excluding the Smooth Criminal sequence and Joe Pesci is convincing as a psychopathic all powerful drug lord. Why he wants MJ dead so bad is beyond me. Because MJ overheard his plans? Nah, Joe Pesci's character ranted that he wanted people to know it is he who is supplying drugs etc so i dunno, maybe he just hates MJ's music.Lots of cool things in this like MJ turning into a car and a robot and the whole Speed Demon sequence. Also, the director must have had the patience of a saint when it came to filming the kiddy Bad sequence as usually directors hate working with one kid let alone a whole bunch of them performing a complex dance scene.Bottom line, this movie is for people who like MJ on one level or another (which i think is most people). If not, then stay away. It does try and give off a wholesome message and ironically MJ's bestest buddy in this movie is a girl! Michael Jackson is truly one of the most talented people ever to grace this planet but is he guilty? Well, with all the attention i've gave this subject....hmmm well i don't know because people can be different behind closed doors, i know this for a fact. He is either an extremely nice but stupid guy or one of the most sickest liars. I hope he is not the latter."

And another example in unlabeledTrainData:

id review "9999_0" "Watching Time Chasers, it obvious that it was made by a bunch of friends. Maybe they were sitting around one day in film school and said, \"Hey, let's pool our money together and make a really bad movie!\" Or something like that. What ever they said, they still ended up making a really bad movie--dull story, bad script, lame acting, poor cinematography, bottom of the barrel stock music, etc. All corners were cut, except the one that would have prevented this film's release. Life's like that." "45057_0" "I saw this film about 20 years ago and remember it as being particularly nasty. I believe it is based on a true incident: a young man breaks into a nurses' home and rapes, tortures and kills various women. It is in black and white but saves the colour for one shocking shot. At the end the film seems to be trying to make some political statement but it just comes across as confused and obscene. Avoid."

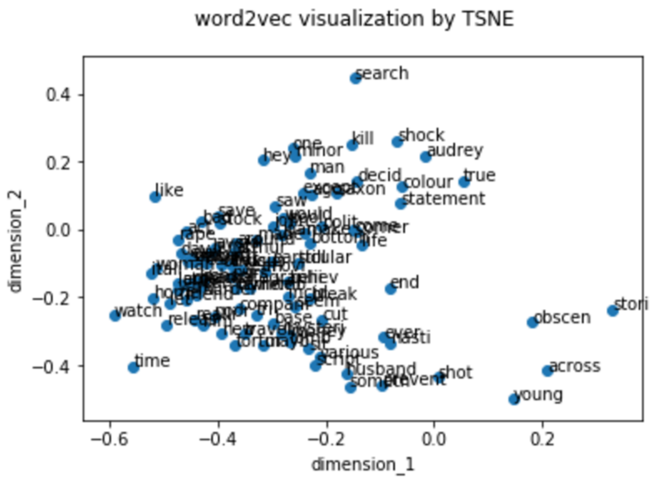
It is abvious that these data are all text which means the computer cannot read these data directly, so in order to put these data in the classifier, we need to trnasfer these data into digital form. But before that we find that there are some symbols or words which do not related to the sentiment of the reviews such as symbols "/?><.," or some tags from HTML, so before we tranform these text data into digits, we need to filter these data and pick out the meaningful part from it. The methods to filter the data include removing HTML tags by regular expression package in python, removing stop words by nltk and methods to extract features including vector counter in scikit-learn and distributed representation in word2vecc will be discussed below in details. There is one problem for the text dataset is how many words left for each review is better, the more words the better or less is better, is there an optimal number of words to make a good prediction? It is really hard to say but obviously we want the meaningful words left so that we use the methods to preprocess the text data which means the words left for each review differs from one and another because of the size of the review and the meaningful words it contains.

Exploratory Visualization

In this project, the features we extract all come from the words of the data, in bag of words, the features come from the frequency of the words appearance, in word2vec the features come from the word features extracted by the shallow neural network. So it is important to know what is the word features look like. I decide to visualize the word features come from word2vec model, there are several reasons, first one is unlike bag of words method which gives a onthot feature on each word, word2vec gives a distributed representation on words so that we can use the vector distance to measure the relations of the words. Second reason is we need to check if the features to represent the words are correct, take the word 'man' for example, if we find the result shows 'desk' is more similar to it than 'woman', we need to check if there is something going wrong in the dataset.

In order to finish the goal of visualize the result of the word2vec, I decide to choose TSNE to visualize the data. TSNE is a machine learning algorithm for dimensionality reduction developed by Geoffrey Hinton and Laurens van der Maaten. It is a nonlinea dimensionality reduction technique which is particularly well-suited for embedding high-dimensional data into space of two or three dimensions, it is composed by two main stages, the first one is to construct a probability distribution over pairs of high-dimensional objects in a way which gives similar objects a higher probability being picked. Second, TSNE defines a similar probability distribution over the points in the low-dimensional map and it minimizes the Kullback-Leibler divergence between the two distributions with respect to the localizations of the points in the map. TSNE is widely used in a wide range of applications, including computer security, music analysis, cancer search and bioinformatics.

Below shows some of the output points come from word2vec model, due to the limit of computation and in order to show them clearly, I choose to show 100 of the points.



Besides, I test some of the words and the result shows the relation of the words are properly defined in word2vec.

```
model_new.most_similar('money')
```

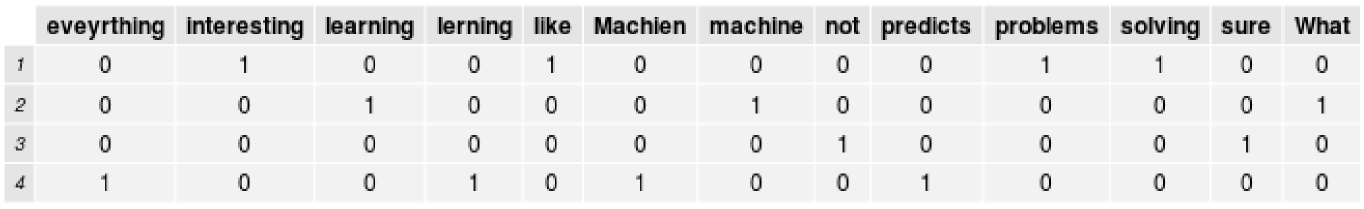
```
[('dollar', 0.6525331735610962),
 ('buck', 0.631759524345398),
 ('debt', 0.5932310819625854),
 ('cash', 0.5882231593132019),
 ('fund', 0.5747219324111938),
 ('fee', 0.5643592476844788),
 ('ticket', 0.5594334602355957),
 ('financ', 0.5379629135131836),
 ('expens', 0.5272642374038696),
 ('profit', 0.5272501111030579)]
```

```
model_new.doesnt_match('man woman watch school money'.split())
```

```
'watch'
```

Algorithms and Techniques

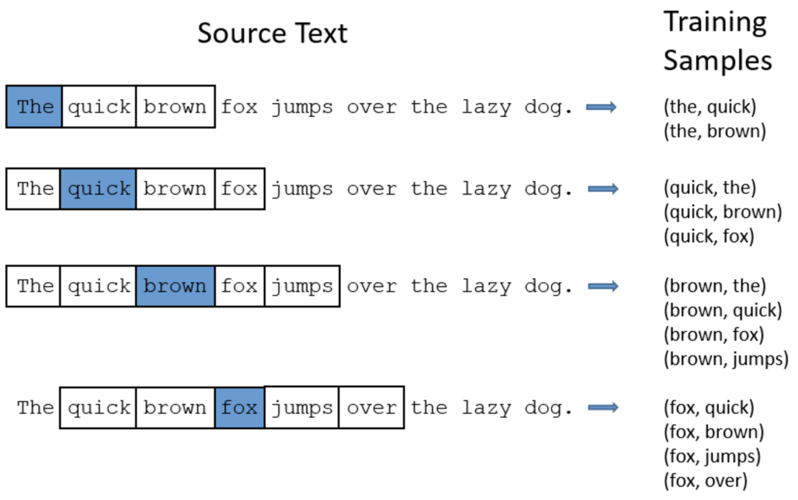
In this project, I am going to use countervectorizer in scikit-learn to implement bag of words algorithm. In bag of words algorithm, we need to select the frequency of several words to represent the reviews in dataset just like the table shows below:



	eveyrthing	interesting	learning	lerning	like	Machien	machine	not	predicts	problems	solving	sure	What
1	0	1	0	0	1	0	0	0	0	1	1	0	0
2	0	0	1	0	0	0	1	0	0	0	0	0	1
3	0	0	0	0	0	0	0	1	0	0	0	1	0
4	1	0	0	1	0	1	0	0	1	0	0	0	0

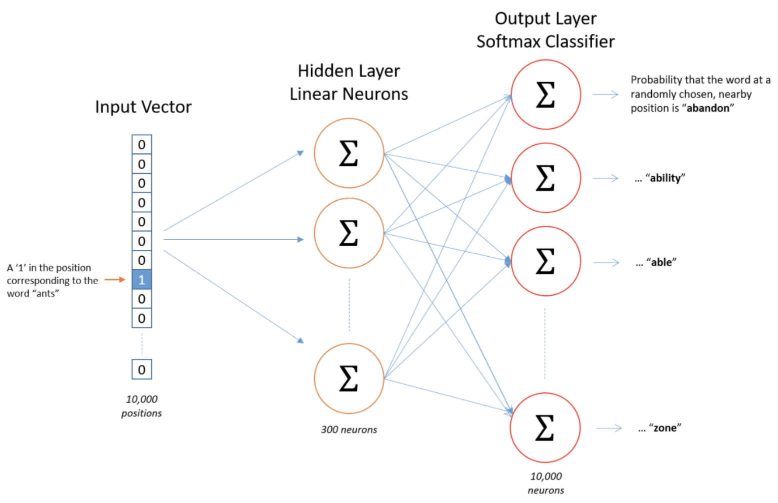
Each row in this table represents a review of a movie and the words in header are the selected words to represent the features. We need to calculate the frequency of each word on the header and use the row vector to represent a review. In countervectorizer there is one parameter needed to be set which is max_features, due to computation limit and the efficiency of the representation, I set the dimension to be 50 for the features.

As to word2vec model, there are several parameter needed to be set. They are num_features, min_word_count, num_workers,context.



For skip-gram in word2vec model, we need to use one word to predict the the probability of the words around it, so the context defines how many words to predict from a single input, the figure above shows what is the output when input a single word with differenct context size. The blue square represents an input and the white squares represent the outputs. In word2vec model, which I set context to 10 due to the computation limit and model efficiency.

The num_features represents how many features to represent a single word.



As the figure shown above, the word2vec model consists of two layers which are hidden layer and output layer. Every input word needs transfering into a one-hot vector, the figure gives an one-hot vector of 10000 elements to represent a word. After going through the hidden layer, each word has been changed into a vector of 300 features is a distributed representation. If we want to predict the probability of certain words around it, we just put the distributed vector into the output layer which is shown as the below figure.

```
model_new.most_similar('money')
```

```
[('dollar', 0.6525331735610962),
 ('buck', 0.631759524345398),
 ('debt', 0.5932310819625854),
 ('cash', 0.5882231593132019),
 ('fund', 0.5747219324111938),
 ('fee', 0.5643592476844788),
 ('ticket', 0.5594334602355957),
 ('financ', 0.5379629135131836),
 ('expens', 0.5272642374038696),
 ('profit', 0.5272501111030579)]
```

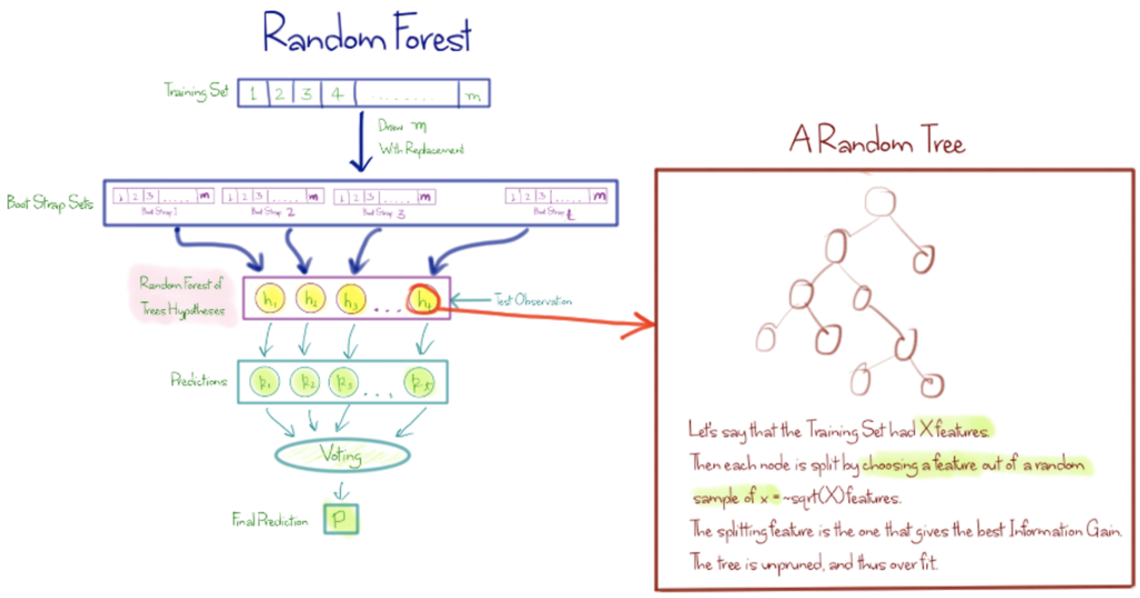
In the word2vec model, I set num_features to 200 which means one word is represented by a 200-dimension vector.

min_word_count is a limit on the occurence of the words in order to save the storage and computation ability, I set the min_word_count to 50 which means the leave the feature vector of the words which appears at least 50 times. num_workers refers to the parallel threads needed to work for the word2vec model, I set it to 6.

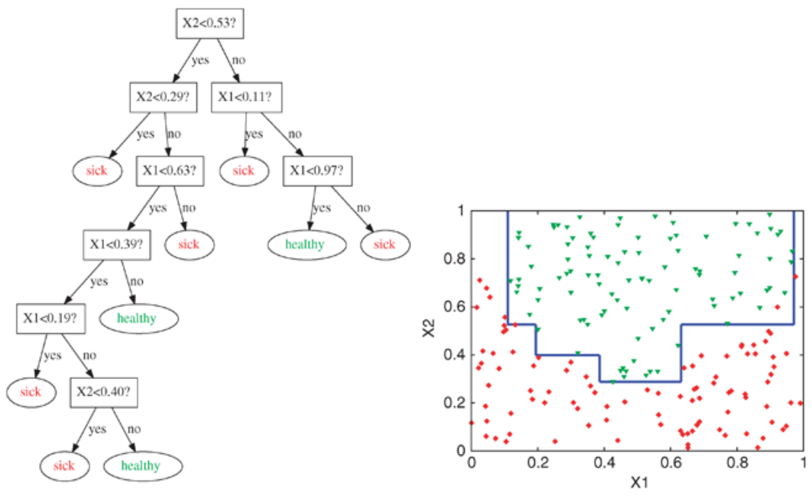
Now even though we have extracted features from the text, what we have are still just representations for words, what we need are the representations for each review. In the vector to paragraph procedure we use the words features to represent the review features. There are mainly two ways to solve it. In bag of words, we have already extract review features, so this procedure only works for word2vec model. The first one is calculate the average feature vector of the words in each review, another one is using kmeans clustering algorithm to extract some 'core' words and then to do representations using the occurence frequency of these words in reviews. Actually the second method is an improvement for the simple bag of words method.

After we get the review features, we come to the final step to do classifications on the input data. I have chosen three machine learning algorithms to achieve this goal. The first one is randomforest, second one is naive bayes and the third one is support vector machine.

Randomforest is an ensemble learning algorithm which operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes or mean prediction of the individual trees. The image below is a graph for the workflow of random forest trees. First we take an input data and copy the data several times then we put each data into one hypothesis function which refers to a random forest tree. After the data goes through each tree, the algorithm will gather all the predictions together and let them to vote for a final decision.

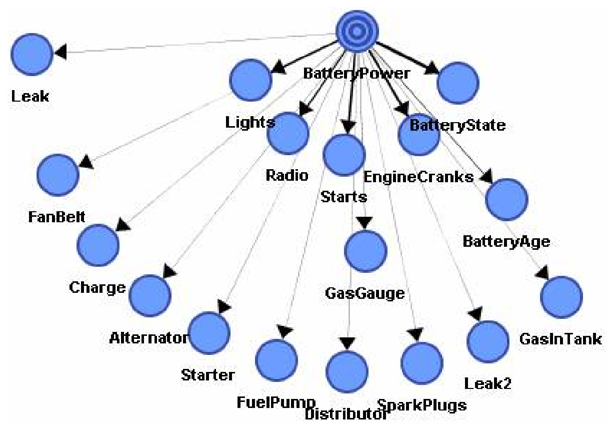

Random Forest
Training Set
Draw m With Replacement
Boot Strap Sets
Random Forest of Trees Hypotheses
Test Observation
Predictions
Voting
Final Prediction
A Random Tree
Let's say that the Training Set had X features.
Then each node is split by choosing a feature out of a random sample of x = ~sqrt(X) features.
The splitting feature is the one that gives the best Information Gain.
The tree is unpruned, and thus over fit.

What the ramdom forest tree will do in the hypothsis function is shown as the figure below, because random forest tree consists of lots of decision trees which means each tree in the random forest is a classifier and it can do nonlinear classifications.



Randomforest usually get a good score on such competition dataset,besides randomforest is an algorithm works as a large collections of decision trees which means it is nothing more than a bunch of decision trees so it is suitable for binary classification problems another reason is as for bagging structure, it is can handle the high dimensionality problems well and it is easy to implement and run fast. In this project I will set the number of trees to 100 which satisfy both efficency and accuracy.

The next algorithm I am trying to use is naive bayes,which is a family of simple probabilistic classifiers based on applying Bayes'theorem with naive independence assumptions between the features.



Abstractly, naive bayes is a conditional probability model, we see that there are four probability to form the formula which are posterior, prior, likelihood, evidence. Given a problem instance to be classified, reprensented by a vecter X, posterior assigns to this instance a probability to class C, the problem of this posterior model is that if the number of features of X is larger or if a feature can take on a large number of values, then basing such a model on probability table is infeasible that is why we use Bayes'theorem.

$$\overset{\text{posterior}}{P(C|\mathbf{x})} = \frac{\overset{\text{prior}}{P(C)}\,\overset{\text{likelihood}}{p(\mathbf{x}|C)}}{\underset{\text{evidence}}{p(\mathbf{x})}}$$

In practice, we only inteseted in the numerator of the fraction, because the denominator does not depend on C and the values of the features are given so that the denominator is effectively constant. The numerator is equivalent to the joint probability model as shown below:

$$p(C_k, x_1, \ldots, x_n)$$

Using chain rule the equation above van be shown as the formula below:

$$\begin{aligned} p(C_k, x_1, \ldots, x_n) &= p(x_1, \ldots, x_n, C_k) \\ &= p(x_1 \mid x_2, \ldots, x_n, C_k) p(x_2, \ldots, x_n, C_k) \\ &= p(x_1 \mid x_2, \ldots, x_n, C_k) p(x_2 \mid x_3, \ldots, x_n, C_k) p(x_3, \ldots, x_n, C_k) \\ &= \ldots \\ &= p(x_1 \mid x_2, \ldots, x_n, C_k) p(x_2 \mid x_3, \ldots, x_n, C_k) \ldots p(x_{n-1} \mid x_n, C_k) p(x_n \mid C_k) p(C_k) \end{aligned}$$

Now the naive assumption comes into play which assumes that all the features are independent which means:

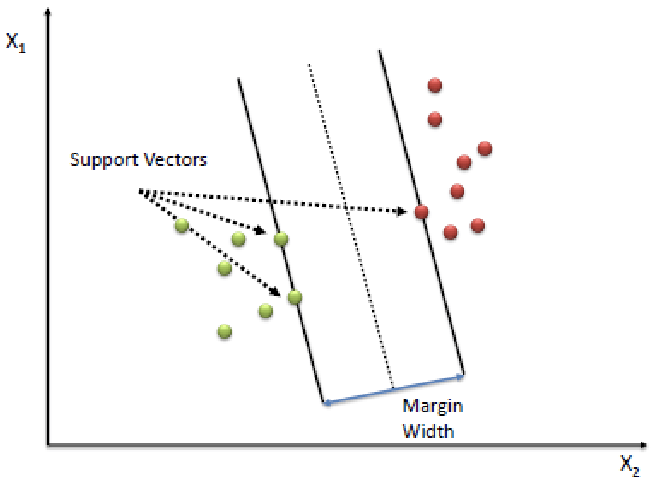
$$p(x_i \mid x_{i+1}, \ldots, x_n, C_k) = p(x_i \mid C_k)\,.$$

Thus, the joint model can be expressed as:

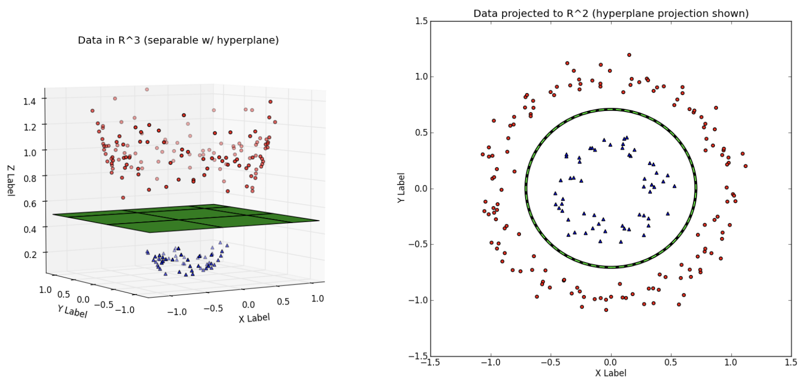
$$\begin{aligned} p(C_k \mid x_1, \ldots, x_n) &\propto p(C_k, x_1, \ldots, x_n) \\ &\propto p(C_k)\, p(x_1 \mid C_k)\, p(x_2 \mid C_k)\, p(x_3 \mid C_k)\, \cdots \\ &\propto p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k)\,. \end{aligned}$$

Though even if naive bayes is often blamed for its simple assumptions but it actually performs well especially in spam email classifcations, it may require more applications if the naive bayes work not well.

The last one I decide to choose support vector machine. As we know that support vector machine does a really good job in text classifications due to its high accuracy and theoretical guarantees regarding overfitting. SVM is an advanced method to solve classification problems which can automatically find the best line to separate the two groups of data, the best line means the line leaves the largest margin between the data as the image shows below:



The start point of SVM is simple. Suppose you want to draw a line to seperate the data into two groups, how will you draw? You must draw a line leaves a distance as far as possible to the two groups of data. Why? Because that will make the classification clearly and confidently. If a data is too near the line or the data just stand on the line, it will be hard to figure out which part it belongs to, for SVM, it is the same that why we need svm to find the maximum margin and that is why SVM works well. Additionally, in SVM we kust need to calculate several data to find the maximum margin, these data called support vector as shown above. Besides SVM can do nonlinear classification, the method it uses called kernel trick which is shown below:



We know that linear classification is easier because we just need a line. However kernel trick makes it possible to draw a line even if the data is nonlinear classifcation. So like the right image above, the data forms a circle and we need a curve to divide the data. But actually the two dimensional plan is not enough to show the relation of these data which means there is some hidden relation between the data like the distance to the circle center. So if we put a third dimension in and mark the distance of every data to the circle center on the third dimension, we will find the result like left image above and we find the data is linear seperable. So this is a brief example to show how kernel trick works and simplely kernel trick is to use a function to reveal the hidden information of the data and put the data on higher dimension in order to make the data linear seperatable.

In this project I will choose linear kernel to do bianry classification.

Benchmark

The benchmark of this project is bag of words model combined with random forest trees. Because bag of words model and random forest model are relative simple models which means we have a space to elaborate the accuracy of the output, besides the result from these two models is good but not enough which means we may find a better solution to solve the problem.

The prediction accuracy of the benchmark when trained with the 25000 reviews and sentiment labels in labeledTrainData and going through 25000 reviews in TestData is 70.46% which is much better than random choice.

III. Methodology

Data Preprocessing

The data used in this project comes from three tsv files which are labeledTrainData, unlabeledTrainData and TestData. The data in the tsv files will be read into pands dataframe form. There are 25000 rows in labeledTrainData and TestData, each row contains three columns which are id(the id number of review),reveiws(the reviews of the movie) and sentiment (positive or negative sentiment on the movie reviews). In unlabeledTrainData, there are 50000 rows and each row contains two columns which are id and review.

It is abvious that these data are all text which means the computer cannot read these data directly, so in order to put these data in the classifier, we need to trnasfer these data into digital form. But before that we find that there are some symbols or words which do not related to the sentiment of the reviews such as symbols "/?><.," or some tags from HTML, so before we tranform these text data into digits, we need to filter these data and pick out the meaningful part from it. The steps to preprocess the data including removing HTML tags by BeautifulSoup, removing non-letter symbols by regular expression package in python, removing step words by nltk, merging similar words by WordNetLemmatizer and Snowball Stemmer, divide sentence by PunketSentenceTokenizer. Then the preprocessed data will be sent to bag of words model and word2vec model to extract text features. The bag of words model will output review feature vectors formed by words frequency and word2vec model will output word feature vectors formed by distributed representations.

Implementation

In bag of words model, the output is review features which means we can send these features directly into the classification model and get results. However in word2vec model, the ouptut is word features which means we need to transfer these word features into review features. There are two ways to reach this goal, first one is average all the word vectors in a review and take the average vector to represent the review features, this is use distributed representation to represent the review features firestly and the second way is using kmeans clustering algorithm to find the 'core' words in all reviews and use bag of words method which means using frequency of these words in each reveiw to represent the reivew features and then normalize these vectors.

After we get the features extracted from bag of words model and word2vec model, we send these features into classifier. Three classifiers have been considered which are randomforest, naive bayes and support vector machine. The principle of these classification algorithms have been disccussed above. Briefly speaking, randomforest will build 100 decision trees to do classification for each input data and then selecting the result which reaches the most agreements of these trees. Naive bayes will using baye' theorem and chain rule to process the data and make the best predictions. Support vector machine will find the best margin to seperate the two groups of training data and make the best predictions on it.

At last we will use confusion matrix to evaluate the predictions and then compare all these predictions to find the best model.(The review features extracted by kmeans cluster must be normalized or there will be no output for the classifier.)

Refinement

During the refinement process, I have done alot with the parameter tuning. In data preprocessing step I have tried to use or not to use lemmatizer and snowballstemmer to merge the simialr words, the result shows depend ton the randomforest classifier, with similar wrods merging the prediction accuracy is 70.46% better than without similar words merging the accuracy of which is 61.06%.

In feature extraction procedure, I have improved the simple bag of words method to combining word2vec, clustering and bag of words method. Because in simple bag of words method, the 'core' words selected to represent the data is by random but in the improved method, the 'core' words are selected by kmeans clustering algorithm which means word2vec gives a distributed representation for all the words and then kmean clustering selected the best centers to reepresent the words, depending on the randomforest classifer the result shows that the accuracy with word2vec and clusters is 84.34% which is better than the result of simple bag of words model which is 70.46%.

In tuning word2vec model, I have chosen several parameters for feature_num which are 300,500, the results of which show depending on the randomforest algorihtm with no clustering are 76.49%,83.62%. Due to the limitation of the computation, I donot choose higher feature size.

In the classification procedure, I have tuning the parameter of the kernel tpye and get an accuracy of 85.94% with linear kernel and word2vec model only which is better that accuracy of 73.38% with rgb kernel. Due to the computation limit, I donnot use grid search to find the best parameter.

IV. Results

Model Evaluation and Validation

The final model I decide to use is combining support vector with word2vec and clustering. Because support vector amchine is a high accuracy and theoretical guarantees regarding overfitting and svm works really well text classifications. In parameter tuning process combining word2vec (num_features=500,context=10,min_count=50)with clustering(num_center=1000), support vector machine with linear kernel get an accuray of 86.73% on the 25000-row TestData which is betterthat 83.71% using rbg kernel. The accuracy from benchmark to final model has been improved nearly 20% which reaches the expectation, so the model and the parameter for the model are appropriate. However because there are only 25000 rows in training data which is about 30MB and 50000 rows in unlabeled taining data which is about 60MB, the training data is relatively small. Usually a text model needs about 50MB text data to process in order to work properly. However, the word2vec model is built by unlabeld traind data of 60MB which means the data size is big enough for word2vec model to extract proper features from the data. So I think the final model is robust enough besides the input test data from TestData file is absolutely different from the data in labeledTrainData and unlabeledTrainData, so the accuracy nearly 90% can be trusted and the model generilizes well to unseen data. Additionally, the table shown below is implement by train_test_split method on labeledTrainData with SVM, word2vec and clustering method which is shows an accuracy of 86.69%, 87.37% and 87.92% on validation dataset. I split the labeledTrainData with 40%, 30% and 20% to be the validation dataset 1,2 and 3, then train the model with the trainig data and evaluate with the validation data.

	Validation_1	Validation_2	Validation_3
SupportVM	86.69%	87.37%	87.92%

So I think the final model is robust enough besides the input test data from TestData file is absolutely different from the data in labeledTrainData and unlabeledTrainData, so the accuracy nearly 90% can be trusted and the model generilizes well to unseen data. Additionally, because features are extracted from unlabeledTrainData, so small change on train data to the classifier does not change the result.

Justification

	Bag of words	Word2vec	W2VCluster
Randomforest	70.46%	83.62%	84.34%
GaussianNB	69.28%	75.92%	76.18%
SupportVM	71.42%	85.94%	86.73%

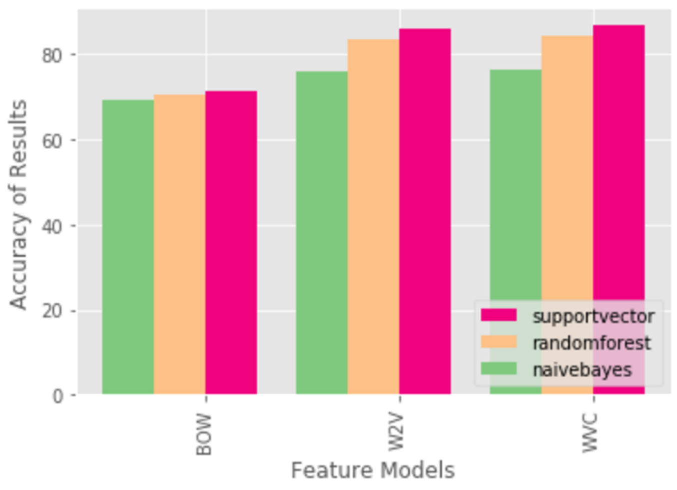
From the table above, we can see that the benchmark combining bag of words and randomforest gets an accuracy of 70.46%. The final model SVM with W2VCluster(combining word2vec with keams clustering) gets an accuracy of 86.73% which is about 16% higher. The best result of naive bayes implemented on the W2VCluster is 76.18% which is about 6% better than the benchmark. The average accuracy of the randomforest classifer implemented on above three data feature extraction methods is 79.47%, the average accuracy of the naive bayes classifier implemented on above three data feature extraction methods is 73.79%. The average accuracy of the support vetor machine classifeir implemented on above three data feature extraction methods is 81.36%. The median of the results of the three classifier lies in word2vec column which is 83.62% for eandomforest, 75.92% for naive bayes and 85.94% for support vector machine.

The final model is composed by SVM and W2VCluster which gets an accuracy of 86.73%. It is known that support vector machine is an algorithm of high accuracy and theoretical guarantees regarding overfitting. Support vector machine can find the best margin to fit and to divide the data, and support vector machine is good at doing binary classification problemss, so it is reasonable to see the best result. Besides the other reason to have a better model than benchmark model is using word2vec model to extract data features. Due to the distributed representaion of the words from word2vec model, we can extract semantic features from the text that is why we can evaluate the similarity and dissimilarity of the two words after going through word2vec model. However even though we get a good result from the above three feature extraction methods, there is a common problem for these three which is they all lose the order of the words. In bag of words model, we just count the occurence of the words in each review and in word2vec we just evaluate each word seperately in each review. So the next step to fill the gap between the 87% accuracy and the 100% accuracy is to extract the features from the words order which is a feature from sentence not words.

Actually the result from randomforest and support vector machine are enough to solve our problem. Suppose we have 100 reviews on different movies and there are about 87 reviews will be predicted correctly, if someone wants to find a good movie to watch on Internet, there is a high probability to select the right one, or if someone wants to know the overall reviews on one movie, it is possible to get the overall opinion on the movie because the majority of the reviews are prdeicted correctly.

V. Conclusion

Free-Form Visualization



We can see from the visualization of the models above. The word2vec model has an advantage towards the bag of words model, because for the three classifier implemented on word2vec model have a big increase in result of accuracy. And the results from word2vec is similar with word2vec with clustering which means using clustering to extract 'core' features from distributed vectors does not help a lot. For the classifiers, we can see that SVM is the best of these three, the accuracy gaps between these three classifier increase from bag of words model to word2vec model with clustering.

Reflection

In this project, we need to do a sentiment analysis on the IMDB. The procedure includes data preprocessing, data feature extraction and feed in classifier. In the first step we use BeautifulSoup, regular expression package, nltk to remove the HTML tags, remove non-letter symbols, remove stopwords, merge similar words, seperate sentences. After going through the first step, we use bag of words model, word2vec model to extract data features. Then we use word vectors average method and clustering method to recover the review features from word features. At last feeding thses review features into three classifiers and find the best result.

In this project, I find that how to extract the features of reviews is the most interesting and challenging thing. As we know there are many noise for text data such as stopwords, symbols and so on, besides sometimes some noise may have a meaning like symbol ^^ which indicates a positive sentiment, so how can we distinguish what is meaningful symbol and what is noise is an important thing to do. Another problem is how to keep the features of the reviews, as mentioned above, using bag of words model and word2vec model only extract the word features but loss the sentence features whic h is the order of the words, so how to keep sentence feature to be extracted is another important problem. So it makes sense that the difficult part of this project despite choosing the classifier is how to filter the text and extract the text features, should we remove all the symbols? or should we remove the stopwords? For feature extractions, the difficulty is how stay the features of reviews including word features, sentence features even paragraph features.

The final model solution fits my expectation because I know with no sentence features which is the words order, it is hard to achieve a 90% above accuracy. But actually 87% is enough to solve problems like sentiment analysis on movie review

Improvement

As mentioned above, the further improvement should be leaving meaningful symbols from reveiws and extract the lost features which is words order. Besides we can use gridsearch to tune support vector machine in order to reach a higher accuracy.

For leaving meaningful symbol, I think we can give a list of such symbols and check the reviews depend on it. For feature extractions I know there is a package in gensim called doc2vec which can extract the sentence features from the reviews and for classifier we can use gridsearch and try different kernels. AS I mentioned above, there should be better results because we do not extract all the features from the text.

Before submitting, ask yourself. . .

- Does the project report you've written follow a well-organized structure similar to that of the project template?
- Is each section (particularly **Analysis** and **Methodology**) written in a clear, concise and specific fashion? Are there any ambiguous terms or phrases that need clarification?
- Would the intended audience of your project be able to understand your analysis, methods, and results?
- Have you properly proof-read your project report to assure there are minimal grammatical and spelling mistakes?
- Are all the resources used for this project correctly cited and referenced?
- Is the code that implements your solution easily readable and properly commented?
- Does the code execute without error and produce results similar to those reported?

Type *Markdown* and LaTeX: α^2