CSE 484-654 HW3 REPORT

Furkan Kalabalık 161044001

I. Introduction

In this project, we've built a classifier that classify movie comments from beyazperde dataset using Naive Bayes classifier with word2vec created vector features. We trained our word vectors by combining Turkish Wikipedia dump, train and test data of movie reviews. After training and obtained word vectors by the help of word2vec, we obtained vectors for every comment using word2vec created vectors. After that we proposed two different algorithm for Naive Bayes classification method. Using these algorithms and train set, we trained our models, then we run it over test set as well as on random review sentence. Then we show results and accuracy of our models.

II. Preprocessing of Corpus

First of all, to train and get word vectors, we used word2vec framework. Word2vec is an NLP technique to get word vectors according to word's context with preserving their semantic meaning and associations. In this purpose, we created our big corpus. The reason of we include all test and train in this vector creating process, when we need to get a one feature vector for every comment, we need all vector representations of every word that constitute review. If one of the word of review is not in word2vec representation, we cannot end up with a feature vector for review and we cannot estimate its positiveness or negativeness.

In this manner, we processed our Wikipedia dump. We remove all punctuation from text. Than we do case-folding by lower all characters. After that, we remove words that length less than 3. The reason of that, words that has length less than 3, cannot serve meaning to our model and they usually inflection or conjunction. After doing these, we saved new created corpus to file. This is the process of Wikipedia dump.

Then we process train and test set. We get comments from dataset, then we combine all reviews into one single corpus. Now, we have one big corpus just like Wikipedia dump. We make all operations on that review corpus just like Wikipedia dump. Then we add review corpus to end of Wikipedia dump. From there we have clean and ready to process corpora that is created from train, test and wikipedia set. We have no unnecessary word and no punctuation.

III. Word2Vec

In Word2Vec stage, we switch to Word2Vec project, and we run this C code on our big corpus. We use default of Word2Vec values, generally. We only changed the minimum count parameter to 1 and vector size to 300. 300 is recommended by users of Word2Vec algorithm and it is commonly used. Minimum count needed to requirement of every word in train set must have vector representation. Other than that we used CBOW approach while we obtain word vectors, our windows size is 5 and we don't use hierarchical softmax function.

IV. Word2Vec to Review2Vec

From Word2Vec framework, it outputs word vectors that is needed to classification. We can load this vector file to Python by using **gensim** module. From there, we can get review vectors by using word vectors. To do that, we read our comments into a array. Every index corresponds a comment and we also kept labels associated with that comment. Comments are in raw form, they are not processed but we give processed words to word2vec framework. To find vector representation of words in a review, we process comment just like preprocessing stage. After this process, we have words just like input to word2vec. Now, when we try to get a word vector of a word in a review we can get it.

For every comment, we need to obtain a one feature vector by using word vectors. There different kind of methods:

- We can take element-wise minimum or maximum of every word vector that constitutes comment.
- We can sum all word vectors that constitutes comment then we can take average.
- We can use tf-idf scores and multiply it word vectors to get review vector.

We used element-wise maximum method because it performed well on our dataset. Then using maximum method, for every comment by using its words, we get vector representation of words from word2vec and we end up with one feature vector for every review on dataset that has same dimension with word vectors. Then we put these vectors inside a **numpy** array and we have our input matrix for classifying that every row represents a comment and we have labels that associated with every row of feature matrix.

We do these operations on test set also to use it in training step. After that, we have training set and labels for them. We can proceed to classification stage.

V. Naive Bayes

In project, we used Naive Bayes classifier. Actually, Naive Bayes is consist of set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of conditional independence between every pair of features given the value of the class variable.Bayes' theorem states that for given class y and dependent features from x_1

to x_n:

$$P(y|x_1, x_2, ..., x_n) = \frac{P(y)P(x_1, x_2, ..., x_n|y)}{P(x_1, x_2, ..., x_n)}$$

By the independence assumption:

$$P(y|x_1, x_2, ..., x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i|y)}{P(x_1, x_2, ..., x_n|y)}$$

Since $P(x_1, x_2, ..., x_n)$ is same for given feature vector:

$$P(y|x_1, x_2, ..., x_n) = P(y)\prod_{i=1}^{n} P(x_i|y)$$

and we can classify given features to a class by result of this probability score for particular class. Maximum probability of this calculation for a class given class of given feature vector:

$$\hat{y} = \underset{y}{\operatorname{argmax}}\, P(y)\prod_{i=1}^{n} P(x_i|y)$$

Different naive bayes classifiers, differs from calculation $P(x_i|y)$ according to distribution of data. In case our word vectors we have a continuous data, our features are continuous floating point numbers and they have positive and negative values inside features. Naive Bayes classifiers have different methodologies by dealing with continuous data. The most known Naive Bayes Classifiers are Gaussian, Multinomial and Bernoulli. Multinomial and Bernoulli are deal with discrete features. Especially, Bernoulli deals with binary values.

To calculate above equation and maximize a class' probability we need two things. First we need class prior probabilities then we need a calculation method for $P(x_i|y)$. By using these, we can calculate $\hat{y}$.

To calculate class prior probabilities we can use count method. By using total number of training data count and frequencies of class we can calculate class prior probabilities as follows:

$$\text{prior of a class} = \frac{\text{\# of samples in the class}}{\text{total \# of samples}}$$

Since our data continuous, we need different methods:

1) We can directly assume that distribution is a normal distribution and we can use Gaussian Naive Bayes method for calculation.
2) To discretize it and estimate the probability distribution of the discretized variable by means of a multinomial probability distribution.

A. Gaussian Naive Bayes

In this approach we calculate $P(x_i|y)$ values by using normal distribution function.

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} e^{\left(-\frac{(x_i-\mu_y)^2}{2\sigma_y^2}\right)}$$

Here is the calculation for particular feature. Using that function and prior probability, we can calculate probability for particular class for a review and bigger indicates negativeness or positiveness of given comment. As you can see, our dataset is consist of feature vectors for every comment. Also every comment associates with a label. In the equation we have values of μ_y and σ_y. These are mean and variance for a particular class. To calculate them we use all features of particular class and after calculation we put them into equation for probability calculation. As we said, all features of particular class is used for calculation.

From there for any given film review, we can derive comment vector by word vectors. After that by using above formula and prior probability, we can understand class of given comment and we can classify. By calculating probability, bigger than indicates class of given comment, positive or negative.

B. Multinomial Naive Bayes with Discretization

Mostly used paradigm for classification is using Multinomial Naive Bayes on discredited variables.This paradigm only handles discrete variables and if a continuous variable is present, it must be discretized with the tolarance of loss of information.

By doing discretization of features, we used bining method.We find whole interval by using maximum and minimum of features and we divide this interval into smaller intervals. These intervals called bins. According to value of feature, it putted to a bin. After that these bins represent values of features. By doing that we can discretize our continuous values into discrete values.

These discrete values provide us to use Multinomial Naive Bayes Classification method which is widely used in text classification. Previous step we do discretization, the reason of that is to use Multinomial Naive Bayes, because Multinomial Naive Bayes can only deal with discrete values.

The distribution is for each feature in a sample putted in a vector that every element corresponds $P(x_i|y)$ for that particular feature that belong to class y and of course this vector has size of feature vector corresponds total feature number in a sample. There are features counted. As we can see, there can be holes in this probability vector because not all feature values can be presented in a sample. To prevent zero probability, we need smoothing. Here, I used Laplace(Add-one) smoothing.

$$P(x_i|y) = \frac{N_{yi} + \alpha}{N_y + \alpha n}$$

Here α is 1, since we used Laplace smoothing. N_{yi} is the number of feature i appears in a sample that belongs to class y in training set. N_y total count of features that belongs to class y. n is number of feature in one sample which is 300 in our case.

With that probability calculation, we can calculate probability of features individually, then with the assumption of independence, we can calculate a sample's probability with respect to a class. Bigger probability for a class indicates that sample belongs to this class.

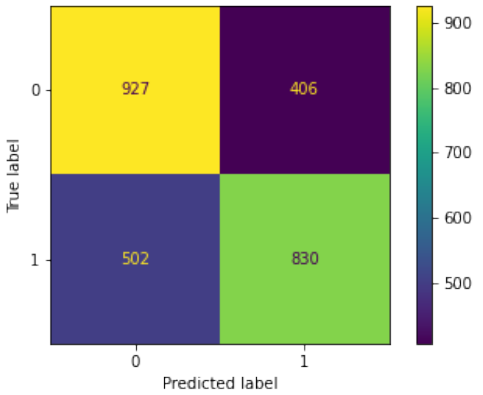
In movie reviews, after discretization, we have features that countable and Multinomial NB can be applicable. We can count features and we can decide whether a review positive or negative.

VI. Results

After the train of classifiers, we run test set on this classifiers. Here is the results with in confusion matrix.

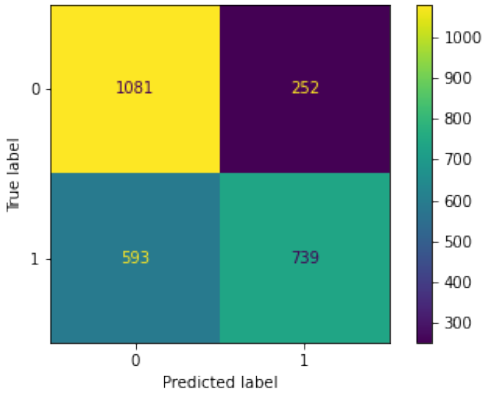
A. Gaussian Naive Bayes

Fig. 1. Confusion Matrix for Gaussian NB



B. Multinomial Naive Bayes with Discretization

Fig. 2. Confusion Matrix for Multinomial NB



VII. Lastly

Project output can be examined on following colab link. Project written on using Jupyter Notebook. Both python file and Jupyter Notebook can be used to examine. Also train and test files included, because some changes made on them manually. Wikipedia dump used from previous HW's link. Colab Link.