Probability of a comment to receive a sarcastic reply.

for

Text Mining and Sentiment Analysis

by

Marticola No: 942091

Amanpreet singh

Università degli Studi di Milano

TABLE OF CONTENTS

Chapter 1

Probability of a comment to receive a sarcastic reply

1.1 Introduction

The Sarcasm on Reddit dataset provides comments posted on Reddit labeled as sarcastic (1) or not sarcastic (0). The task of the project is,to use only parent comment and the Reddit category (subreddit) and to predict the probability of a parent comment to receive a sarcastic comment.

1.1.1 Techinques and Methods Used

PCA
It is a method of summarizing data by reducing dimensions.Many of information in features could be redundant. If so, we should be able to summarize features with fewer characteristics.PCA looks for properties that show as much variation across data as possible.PCA looks for properties that allow to reconstruct the original data characteristics as well as possible.

K-Means Clustering
K means clustering is Centroid based clustering.K-means clustering calculates the centroids and shuffles until it finds optimal centroid. It assumes that the number of clusters are already known(K).K-means follow Expectation Maximization approach to solve the problem. Expectation step is used for grouping the data points to the closest cluster and the Maximization-step is used for computing the centroid of each cluster. The data points are assigned to a cluster in such a manner that the sum of the squared distance between the data points and centroid would be minimum. It is to be understood that less variation within the clusters will lead to more similar data points within same cluster.

NGRAMS
N-grams are simply all combinations of adjacent words or letters of length n that one can find text. As an example, the hello, world! text contains the following word-level bigrams hello, hello world, world The basic point of n-grams is that they capture the

language structure from the statistical point of view, like what letter or word is likely to follow the given one. The longer the n-gram (the higher the n), the more context you have to work with. Optimum length really depends on the application .If n-grams are too short, it may fail to capture important differences. On the other hand, if they are too long, it may fail to capture the "general knowledge" and only stick to particular cases.

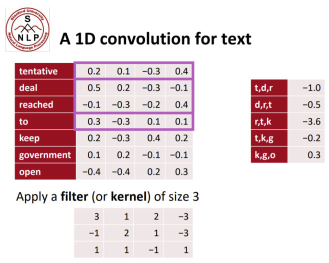
SVD and Truncated SVD
The Singular-Value Decomposition, is a matrix decomposition method for reducing a matrix to its constituent parts in order to make matrix calculations simpler. Data with a large number of features, such as more features could be reduced to a smaller subset of features that are most relevant to the prediction problem.To do this we can perform an SVD operation on the original data.In NLP, SVD can be applied on matrices of word occurances. Truncated Singular Value Decomposition (SVD) is a matrix factorization technique that factors a matrix M into the three matrices U, $\sum$
, and V. This is very similar to PCA, excepting that the factorization for SVD is done on the data matrix, whereas for PCA, the factorization is done on the covariance matrix. Typically, SVD is used under the hood to find the principle components of a matrix.

Logistic Regression
The idea behind the model is to estimate the conditional probability of some event of interest given some independent variables. Mostly they are the maximum likelihood estimators. Because the logistic model is non-linear model - the optimization is done numerically. Using Logistic model in order to perform classification rather than compute probabilities. As such, need to set a rule that converts probabilities into classification. The default rule is: if p(yi—Sarcasmi)>1/2 then give the label "sarcasm" to subject i.Logistic Regression uses the sigmoid function, that is a bit like a "cut-off" function (a function that is 0 up to a point, then at some X_0 very quickly goes to 1). This allows us to split values of x to two groups-classes: those less than X_0, and those more than X_0. At the same time, it is a bit better than a step function) because sigmoid is smooth and so can have a derivative. (Also, maybe if we are very close to the cut-off point we might not want to give 1 or 0 but rather a number)

Google Bert
BERT is pre trained model by google which enables Transfer Learning and implementation over personalised data and can be effective on Training a model purely on weak labels,which is infact our case,BERT has its own ability to train language models based on a set of all the words in a sentence or question (two-way training) rather than the traditional way of training word order (from left to right or combining left to right and left). BERT allows the language model to learn the context of words based on the surrounding words rather than just the previous or next word immediately.

1D Convolutional Layer
A 1D CNN is very effective when you expect to derive interesting features from shorter (fixed-length) segments of the overall data set and where the location of the feature within the segment is not of high relevance.1D convolution layers are translation invariant in the sense that because the same input transformation is performed on every patch, a pattern learned at a certain position in a sentence can later be recognized at a

different position.



A 1D convolution for text

tentative	0.2	0.1	−0.3	0.4
deal	0.5	0.2	−0.3	−0.1
reached	−0.1	−0.3	−0.2	0.4
to	0.3	−0.3	0.1	0.1
keep	0.2	−0.3	0.4	0.2
government	0.1	0.2	−0.1	−0.1
open	−0.4	−0.4	0.2	0.3

t,d,r	−1.0
d,r,t	−0.5
r,t,k	−3.6
t,k,g	−0.2
k,g,o	0.3

Apply a **filter** (or **kernel**) of size 3

3	1	2	−3
−1	2	1	−3
1	1	−1	1

Figure 1.1: Christopher Manning https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1194/slides
2019-lecture11-convnets.pdf

1.2 Research question and methodology

I implemented Two Approaches to the problem,considering with a thought of trying to recognize patterns in a sequence which can lead to a Sarcastic reply-

1.2.1 1st Approach

1- What are the Sentiments in a Comment?

Polarity of a comment-Emotions expressed in a sentence.
Subjectivity-Amount of personal feelings, views, or beliefs expressed in a sentence
For this I used package that calculates Polarity and subjectivity of a sentence on the basis of Lydia System.

$$\text{Polarity} = (\text{Pos}-\text{Neg})/(\text{Pos}+\text{Neg})$$
$$\text{Subjectivity} = (\text{Pos}+\text{Neg})/\text{count}(*)$$

and Classifying words from Harvard HIV-Dictionary which classifies words to Positive,Negative,Motivational etc.

2 - What are the POS Tags frequency in Comments which received Sarcastic replies and Comments which received Neutral replies?

To see the usage of Nouns,Adjectives,Sequence patterns in POS Tags effects a sarcastic or neutral reply.For example if a comment is using too many adjectives for Nouns or pronouns in an Sentence,does it leads to Sarcastic reply.
To find this ,I created Bag of words and counted frequency of each Pos Tags,and then Reduced dimensions to use them as features in Classifcation Model.PCA cannot be an alternative to Clustering as they both have different goals.Cluster analysis groups observations while PCA groups variables rather than observations.But they way I used

in sequence,different POS Tags are variables in PCA that are grouped together from 34 dimensions to 12,

3 - What are the different kind of words used in whole dataset and if the sentences in whole database can be grouped into Clusters?

I cleaned and pre processed the parent_comment and created bigrams (ngrams of 2) for each sentence.Then i tokenized the bigrams of words.In order to see if the they can be grouped together,first I further decided to reduce dimension of the sequences and I did PCA to keep components that explains 70 percent of the Data Variance.Then I found K means can find patterns and can group the parent_comment into 4 different clusters.

4 - Sarcasm is dependent on the person who gives answers or the framing or words of a question? Is it possible to predict if a question has more probability to get sarcastic reply from just the category of the question/parent reply and words and framing of the question/parent reply itself ?

I tried to used various techniques and selected Logistic regression with SVD dimension reduction as it has the best results with various other feature selections I did.More about results will be in the Result section.

1.2.2 2nd Approach

1 - Is Bert which is Pre Trained over massive corpus can get pattern and put weights on the words that can lead to a Sarcastic Reply?

BERT as a sentence encoder, which can accurately get the context representation of a sentence and it is possible when we train a model on a large text corpus, our model starts to pick up the deeper and intimate understandings of how the language works. BERT learns positions of the words in a sentence.It captures information from both the right and left context of a token ,and if a question has more polarity and someway uncertain and different in views in first part from second part ,or from back to front or front to back(recognizing sequence patterns),these are the few Human factors for giving Sarcastic replies.Bert embeddings are context dependent,and work like this;it will mask a token in the sentence and will try to predict that token to learn better about the sentence.

2-Improving Bert Results with BERT-CNN model By adding CNN to the work-specific layers of the BERT model, our model can get details of key fragments in the text. In addition, by fine tuning BERT and CNN detailed tests show that model gets competitive performance ,even on the tasks like predicting Sarcasm based on ,how a question or parent comment is formed.

1.2.3 Models,Hyper Paramters

Truncated Singular Value Decomposition and Logistic Regression

Evaluation strategy-Cross-validation.
SVD:

SVD is part of LSA topics modelling which helps in finding relationship between Document and words by dimension reduction by keeping the values that of our matrix emphasizes on the strong relationships and removes the noise. In other words, it makes the best possible reconstruction of the matrix with the least possible information. The trick in using SVD is in figuring out how many dimensions(concepts) to use when approximating the matrix (n-components)
In my Matrix,in order to decide which variables are noise and which variables are actual useful,or how much data is relevant and how much not,SVD can help by reducing dimensions. In order to decide best dimension,n_compnents,I used cross validation.
Logistic Regression:-
HyperParameters

Class_weight: It penalizes mistakes in samples of class[i] with class_weight[i] instead of 1. So higher classweight means to put more emphasis on a class.
class_weight="balanced", basically means replicating the smaller class until you have as many samples as in the larger one, but in an implicit way

tol: Parameter tells the optimization algorithm when to stop. If the value of tol is too big, the algorithm stops before it can converge.

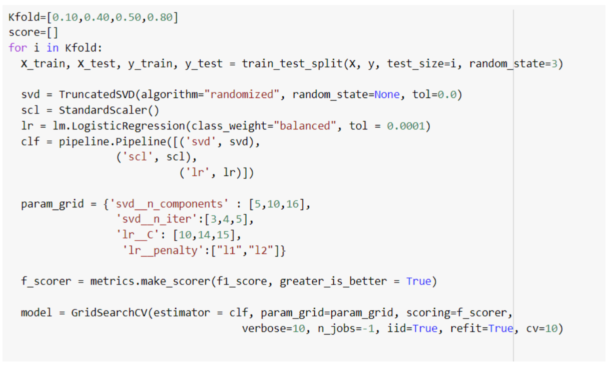
Inverse regularization parameter (lr_C)- Regularization is applying a penalty to increasing the magnitude of parameter values in order to reduce overfitting.g.A control variable that retains strength modification of Regularization by being inversely positioned to the Lambda regulator. The relationship, would be that lowering C - would strengthen the Lambda regulator.

Penalty(l1 and l2 regularization) This is also used to save overfitting of the data. A regression model that uses L1 regularization technique is called Lasso Regression and model which uses L2 is called Ridge Regression.Lasso shrinks the less important feature's coefficient to zero thus, removing some feature altogether. So, this works well for feature selection in case we have a huge number of features.Ridge regression adds "squared magnitude" of coefficient as penalty term to the loss function.

Cross Validation

KFold CV
In order to decide best train and test split ,I did K fold CV and found 0.60 for Train data helps the model to learn the best without overfitting.

HyperParameters and N-Component Selection
I used Gridsearch Cv,it is a library function that is a member of sklearn's model_selection package. It helps to loop through hyperparameters and fit estimator (model) on training set. So, in the end,we can select the best parameters from the listed hyperparameters.



```
Kfold=[0.10,0.40,0.50,0.80]
score=[]
for i in Kfold:
  X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=i, random_state=3)

  svd = TruncatedSVD(algorithm="randomized", random_state=None, tol=0.0)
  scl = StandardScaler()
  lr = lm.LogisticRegression(class_weight="balanced", tol = 0.0001)
  clf = pipeline.Pipeline([('svd', svd),
               ('scl', scl),
                         ('lr', lr)])

  param_grid = {'svd__n_components' : [5,10,16],
               'svd__n_iter':[3,4,5],
               'lr__C': [10,14,15],
                'lr__penalty':["l1","l2"]}

  f_scorer = metrics.make_scorer(f1_score, greater_is_better = True)

  model = GridSearchCV(estimator = clf, param_grid=param_grid, scoring=f_scorer,
                                     verbose=10, n_jobs=-1, iid=True, refit=True, cv=10)
```

BERT and 1D Convolutional Neural Network

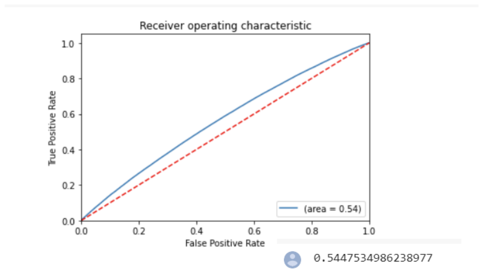
Evaluation strategy-Fine Tuning and Adjustments
For tokenizing the text ,i used BERT tkenizer,BERT embedding layer by importing the BERT model from hub.KerasLayer. The trainable parameter is set to False, which means that it will not be training the BERT embedding.Then I created BERT vocabulary file in the form a numpy array, Then I find the total number of batches by dividing the total records by 32. Next, 10 % of the data is left aside for testing. I created tf.keras.Model class named TEXT_MODEL. Inside the class I made 1D CNN model layers.I used 1d architecture from paper CNN Architectures for Large-Scale Audio Classification , and this architecture seems to work quite well for this Database also.
HyperParameters

Selected with Fine tuning and Adjustments
EMB_DIM = 200
CNN_FILTERS = 100
DNN_UNITS = 256
OUTPUT_CLASSES = 2
DROPOUT_RATE = 0.2
NB_EPOCHS = 5

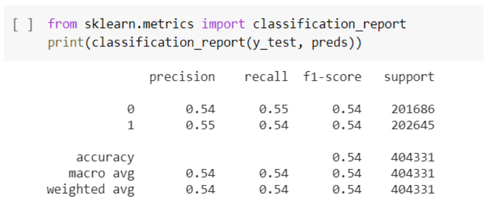
1.3 Experimental results

The results from first approach were not any better than tossing a coin and on an random deciding if certain parent comment will get sarcastic comment or not,As my approach was to find patterns in the Sequence of POS TAGS,and context of sentence in 2Grams ,but the algorithm was not able to find patterns and fit the best estimator. But going towards another approach,where I used BERT tokenizer,results were lot better,reason being,BERT embeddings are context-dependent. irrespective of Word2Vec - Ngrams.BERT generates contextual embeddings, the input to the model is a sentence rather than a single word.This is because the BERT model needs to know the context or the surrounding words before generating a word vector.BERT model explicitly takes as input the position of each word in the sentence before calculating its embedding and along with that 1d neural Networks,also used in Human Activity Recognition,are effective to extract features from sequences of observations and how to map the internal features to different activity types The model can learn an internal representation of the sequence

1st Approach Results

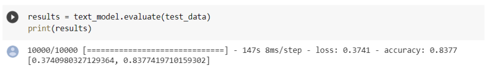


```
0.5447534986238977
```

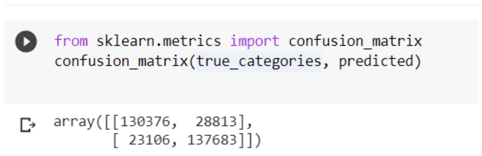


```
[ ] from sklearn.metrics import classification_report
    print(classification_report(y_test, preds))

              precision    recall  f1-score   support

           0       0.54      0.55      0.54    201686
           1       0.55      0.54      0.54    202645

    accuracy                           0.54    404331
   macro avg       0.54      0.54      0.54    404331
weighted avg       0.54      0.54      0.54    404331
```

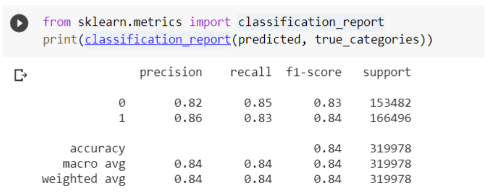
2nd Approach Results



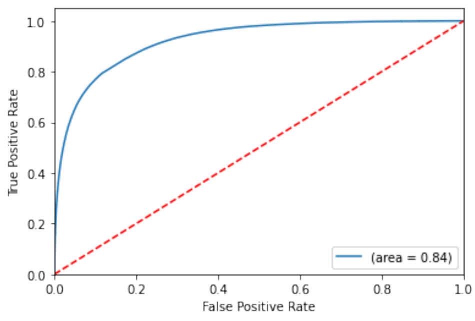
```
results = text_model.evaluate(test_data)
print(results)

10000/10000 [==============================] - 147s 8ms/step - loss: 0.3741 - accuracy: 0.8377
[0.3740980327129364, 0.8377419710159302]
```



```
from sklearn.metrics import confusion_matrix
confusion_matrix(true_categories, predicted)

array([[130376,  28813],
       [ 23106, 137683]])
```



```
from sklearn.metrics import classification_report
print(classification_report(predicted, true_categories))

              precision    recall  f1-score   support

           0       0.82      0.85      0.83    153482
           1       0.86      0.83      0.84    166496

    accuracy                           0.84    319978
   macro avg       0.84      0.84      0.84    319978
weighted avg       0.84      0.84      0.84    319978
```



Precision
Precision calculates, what fraction of the parent comment, model predicted as more probability to get sarcastic reply,actually got sarcastic reply.

Recall Recall tells, what fraction of all the parent comment that got sarcastic reply are were actually predicted with more probability that they will get sarcastic reply.

Making Decisions From Precision ,Recall Auc and Roc Curves The precision and recall give a better sense of how an algorithm is actually doing, especially when we have a highly skewed dataset or weak labels. If we predict 0 all the time and get 99.5% accuracy, the recall and precision both will be 0. Because there are no true positives. So, We know that classifier is not a good classifier. When the precision and recall both are high, that is an indication that the algorithm is doing very well. Auc-ROC Curves are used to see how well your classifier can separate positive and negative examples and to identify the best threshold for separating them. a higher X-axis value indicates a higher number of False positives than True negatives. While a higher Y-axis value indicates a higher number of True positives than False negatives.

1.4 Concluding remarks

The aim for the the project was to find the probabilities ,that if a comment will get a Sarcastic reply or not,But Sarcastic replies should not be dependent on how a comment is made or a question is formed but depends on the person who will give the reply itself.We could have checked that if certain subreddit has more Trolls or etc,but any implementation for this will be incomplete science as we are trying to predict causality from variables that may or may not have an influence over the replier. Though with this project,I tried to approach it the way,by finding patterns in the sequence and grouping them together,and in the first approach ,though if we see probability table,they were making sense according to my approach based on Pos tags,Length and Polarity of the sentence with different words Tokenized,but still the result was not good ,because of the missing context ,that a word could mean different things or if there are more adjectives or less adjectives for a Noun,but without context we cannot link it with it will get Sarcastic reply, Thus here BERT word embedding and 1D cnn came into action where,BERT was able to get context and tokenize a single word differently according to context and 1D Cnn were able to find patterns in those word sequence that can have weights over Sarcasm,But still it is an incomplete science and with saying ,the more you torture your data,it will give you results you want.

1.4.1 References and Citations

Text Mining and Sentiment Analysis ,Alfio Ferra -Giancarlo Manzi

Natural Language Processing with Deep Learning ,Christopher Manning
https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1194/slides/cs224n-2019-lecture11-convnets.pdf

Statistical Approach for Classifying Sentiment Reviews by Reducing Dimension using Truncated Singular Value Decomposition Asmaul Husna,Habibur Rahman,Emrana

Kabir Hashi

1D Convolutional Neural Network Models for Human Activity Recognition https://machinelearningmastery.com/cnn-models-for-human-activity-recognition-time-series-classification/

Khodak, M., Saunshi, N., Vodrahalli, K. (2017). A large self-annotated corpus for sarcasm. arXiv preprint arXiv:1704.05579.

Eke, C. I., Norman, A. A., Shuib, L., Nweke, H. F. (2019). Sarcasm identification in textual data: systematic review, research challenges and open directions. Artificial Intelligence Review, 1-44. link

Chris McCormick-BERT Word Embeddings Tutorial https://mccormickml.com/2019/05/14/BERT-word-embeddings-tutorial/

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding https://arxiv.org/abs/1810.04805

Joshi, A., Bhattacharyya, P., Carman, M. J. (2017). Automatic sarcasm detection: A survey. ACM Computing Surveys (CSUR), 50(5), 1-22. link