Machine Learning Engineer Nanodegree
Capstone Project

Ahmed S. Abdelmalek

May 30th, 2018

I.Definition

Project Overview

Social commentary is the act of using rhetorical means to provide commentary on issues in a society. This is often done with the idea of implementing or promoting change by informing the general populace about a given problem and appealing to people's sense of justice. Social commentary can be practiced through all forms of communication, from the printed form to conversations to computerized communication.

The abundance of public discussion spaces on the Internet has in many ways changed how we communicate with others. Whether it is a comments section for a controversial news article or a forum for discussing a particular video game, these online spaces allow us to easily share our own opinions and findings, as well as hear about the thoughts of others. Though these discussions can often be productive, the relative anonymity that comes with hiding behind a username has allowed people to post insulting or inappropriate comments. These posts can often create a hostile or uncomfortable environment for other users, one that may even discourage them from visiting the site. This problem is a serious one that website owners commonly face, and some researchers already got interested in that case and began to classify the insults using some machine learning models like SVM, Naïve Bayes and ensemble learning techniques like AdaBoost and random Forest, you can have a look here.

Problem Statement

One potential way to mitigate this problem is to build a system that can detect whether or not any given comment is insulting. With such a system, website owners would have a lot of flexibility in dealing with this problem. For instance, the owner could choose to automatically block or hide these insulting comments, or flag them so that they can more easily be found by site moderators. The objective of this project is to do just this: build a machine learning system that can accurately classify online posts and comments as insulting or not.

Metrics

Prediction results are evaluated on the log-loss and auc-roc between the predicted values and the ground truth. The ground truth is the set of labels that have been supplied by human experts. The ground truth labels are inherently subjective, as the true meaning of sentences can never be known with certainty. Human labeling is also a 'noisy' process, and reasonable people will disagree. As a result, the ground truth labels on this dataset should be taken to be 'informed' but not 100% accurate and may include incorrect labeling. We believe the labels, on the whole, to represent a reasonable consensus, but this may often not be true.

Since this is a Kaggle competition project, I will take the leaderboard score as my final evaluation. However I don't know exactly which datasets they have used, the leaderboard score is actually auc-roc of the test data, which are unseen to the public. However, for my training process, I will also use the model both auc-roc and log-loss as the screening metric to select the best training model, and then use F1-Score to fine tune the parameters.

Why these metrics?

1. F1 score which is a mixture between precision and recall where:

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

So the F1 score conveys the balance between the precision and the recall.

2. Area Under the Receiver Operating Characteristic Curve: It's the expected false positive rate if the ranking is split just after a uniformly drawn random positive.

3. Log loss function: The log-loss is a great metric for this project because we are predicting the probability or label being either 0 or 1 where:

$$-\log P(yt|yp) = -(yt \log(yp) + (1 - yt) \log(1 - yp))$$

II. Analysis

Data Exploration

All the data from the three files for this problem are provided here

- Train.csv as a train set (3,947 samples)
- Test with solutions as a test set (2,647 samples)
- impermium_verification_labels.csv (2,235 samples)

Let's look at the first 5 lines of training data:

	Insult	Comment
0	1	"You fuck your dad."
1	0	"i really don't understand your point.\xa0 It seems that you are mixing apples and oranges."
2	0	"A\\xc2\\xa0majority of Canadians can and has been wrong before now and will be again.\\n\\nUnless you're supportive of the idea that nothing is full proof or perfect so you take your chances and if we should inadvertently kill your son or daughter then them's the breaks and we can always regard you as collateral damage like in wartime - and sorry, but\\xc2\\xa0the cheques in the mail. "
3	0	"listen if you dont wanna get married to a man or a women DONT DO IT. what would it bother you if gay people got married stay in your lane do you let them do them. And your god is so nice but quick to judg if your not like him, thought you wasnt suppose to judge people."
4	0	"C\xe1c b\u1ea1n xu\u1ed1ng \u0111\u01b0\u1eddng bi\u1ec3u t\xecnh 2011 c\xf3 \xf4n ho\xe0 kh\xf4ng ? \nC\xe1c ng\u01b0 d\xe2n ng\u1ed3i cu\xed \u0111\u1ea7u chi\u1ee5 nh\u1ee5c c\xf3 \xf4n ho\xe0 kh\xf4ng ?\nC\xe1c n\xf4ng d\xe2n gi\u1eef \u0111\u1ea5t \u1edf V\u0103n Giang, C\u1ea7n Th\u01a1 c\xf3 \xf4n ho\xe0 kh\xf4ng ?\n.....\nR\u1ed1t cu\u1ed9c \u0111\u01b0\u1ee3c g\xec\xa0 th\xec ch\xfang ta \u0111\xe3 bi\u1ebft !\nAi c\u0169ng y\xeau chu\u1ed9ng ho\xe0 b\xecnh, nh\u01b0ng \u0111\xf4i khi ho\xe0 b\xecnh ch\u1ec9 th\u1eadt s\u1ef1 \u0111\u1ebfn sau chi\u1ebfn tranh m\xe0 th\xf4i.\nKh\xf4ng c\xf2n con \u0111\u01b0\u1eddng n\xe0o ch\u1ecdn kh\xe1c \u0111\xe2u, \u0111\u1eebng m\u01a1 th\xeam n\u01b0\xe3."

- From the first glance, the data is much unclean

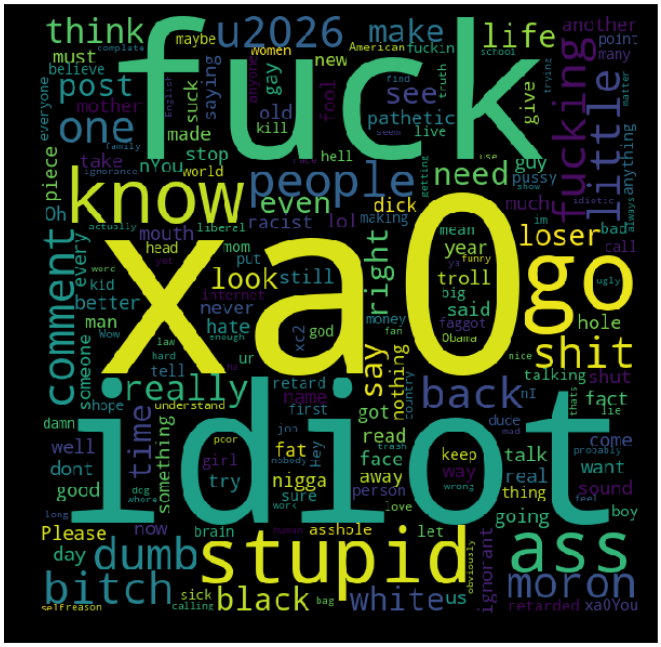
- The total number of comments is 8,829

- The neutral comments represent about 60% of the data(6,010) and Insulting comments represents about 40% of the data (2,819)

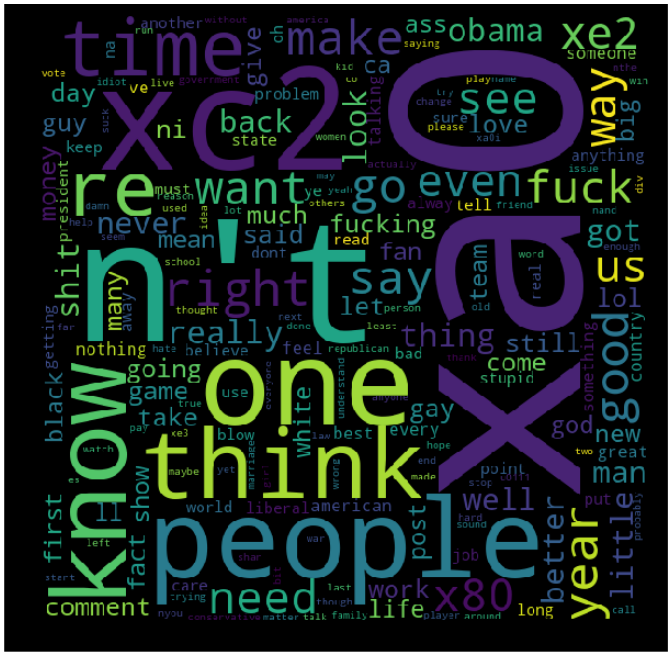
Exploratory Visualization

After splitting the data into train set and test set we could see further using visualization techniques like WordCloud , we could see both bad words in Insults and neutral words in neutral comments

Bad words:



Good words:



When we take a closer look at these two pictures, we notice that there are several words appearing in both clouds like (I'm sorry for any unpolite words here) "People", "fuck", "go" and lots of more words

And other words that appear frequently in only one group like “Know” and “Idiot”
Of course, there are a lot of these non-ascii and non-alphabet characters

Algorithms and Techniques

Since this is a binary classification problem, I will be using supervised learning algorithms.

I will extract features from the attribute I am given on the data. These features include 1. CountVectorizer 2. TfidfVectorizer 3. HashVectorizer. The algorithms I will be implementing are 1. Logistic regression 2. Naive Bayes
2.Support vector machine with linear kernel 4. Gradient boosting 5.Stochastic gradient descents and find the best one.

And then do fine tuning on the best algorithm to get optimal auc-roc and log-loss.

Logistic regression is a regression model where the outcome is binary, and the model predicts the probability (0 to 1) of the outcome
Naive Bayes classifier is a family of simple probabilistic classifiers based on applying Bayes’ theorem with naive independence assumptions between the features.
Support vector machine is an algorithm which outputs an optimal hyperplane to separate labeled training data.

Finally, gradient boosting model is similar to random forests, with the difference being how we train them. For gradient boosting, we assign prediction score in each leaf instead of a binary value, and during training, we add one new tree to the ensemble at a time and do the optimizing.
Linear SVM is the newest extremely fast machine learning, It is a linearly scalable routine meaning that it creates an SVM model in a CPU time which scales linearly with the size of the training dataset, It’s a simple yet very efficient approach to discriminative learning of linear classifiers under convex loss functions such as (linear) Support Vector Machines and Logistic Regression

Benchmark

The benchmark model in my problem is the Multinomial naïve Bayes. The reason is that it is very easy to implement and naive enough to be compared with other algorithms. There is no published result for this problem, but since it is a Kaggle competition, I can get a sense of how my model is performing by looking at the leaderboard score and ranking.

III. Methodology

Data Preprocessing

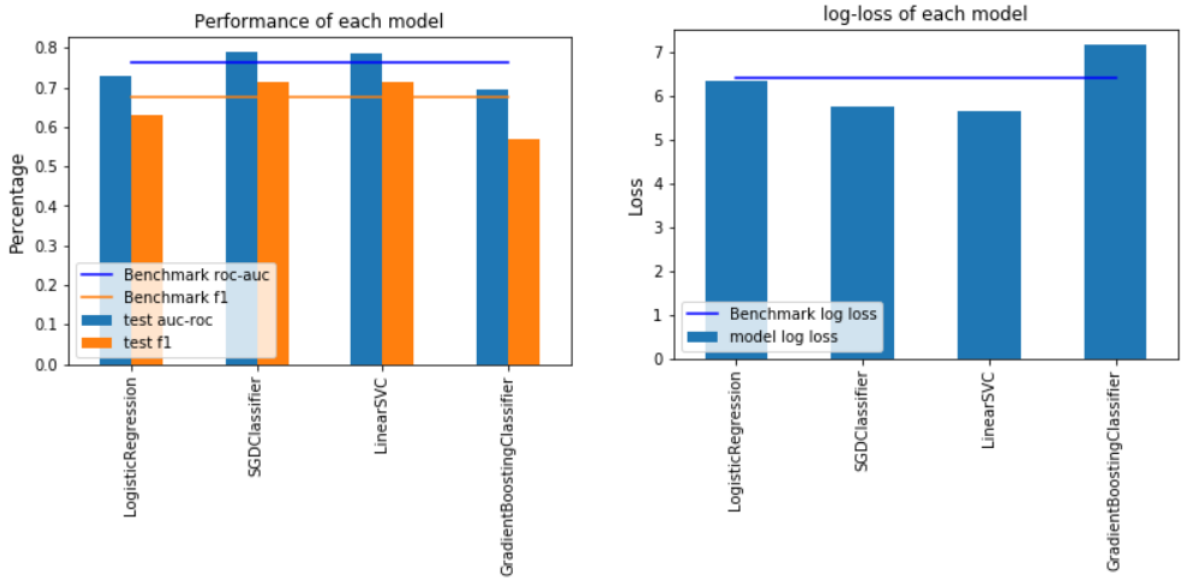
First I will clean up the strings by removing double quotations, dots, colons, brackets and other similar characters, exclamation and question marks, numbers, slashes and string commands like newline commands \n and tab command \t, mentions and URLs, HTML code, these funny characters like \xa0 and \xc2, and remove the stop words of English language . Next, I will replace all words contain apostrophes with their original words like n't to not, short words like u and em to you and them, also by using bad words text downloaded from this link, I will replace all bad words derivatives into the original word, Then I will use stemmer.

Implementation

I have tried 4 supervised learning models excluding the benchmark model; I normalized all feature values before doing model fit. The following are the implementation of each model and by optimizing the vectorizer for best parameters with the best F1-score

```
best parameters for classifiers LogisticRegression and TfidfVectorizer is {'vect__ngram_range': (1, 4), 'vect__stop_words': 'en
glish', 'vect__sublinear_tf': True, 'vect__min_df': 3, 'vect__analyzer': 'char'}
best parameters for classifiers SGDClassifier and TfidfVectorizer is {'vect__ngram_range': (1, 5), 'vect__stop_words': None, 'v
ect__sublinear_tf': True, 'vect__min_df': 1, 'vect__analyzer': 'word'}
best parameters for classifiers LinearSVC and TfidfVectorizer is {'vect__ngram_range': (1, 3), 'vect__stop_words': None, 'vect_
_sublinear_tf': True, 'vect__min_df': 1, 'vect__analyzer': 'word'}
best parameters for classifiers GradientBoostingClassifier and TfidfVectorizer is {'vect__ngram_range': (1, 4), 'vect__stop_wor
ds': 'english', 'vect__sublinear_tf': True, 'vect__min_df': 1, 'vect__analyzer': 'char'}
```

	test auc-roc	test f1	test log loss	train auc-roc	train f1	train log loss
LogisticRegression	0.730289	0.630137	6.336695	0.782055	0.717385	5.159070
SGDClassifier	0.787848	0.713870	5.769552	0.995558	0.994046	0.132034
LinearSVC	0.787341	0.714709	5.652201	0.996648	0.995371	0.102693
GradientBoostingClassifier	0.695648	0.568396	7.158118	0.755326	0.674944	5.633404

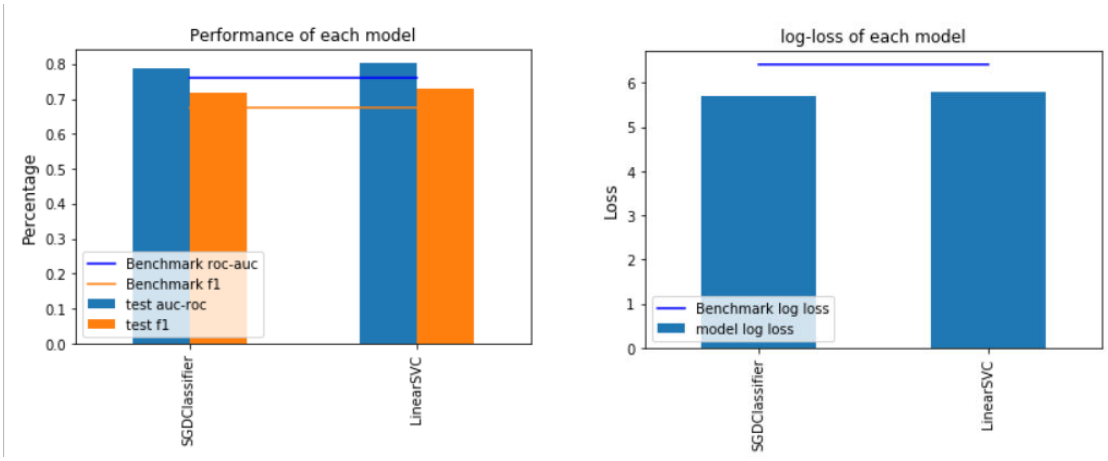


From These results, I will take SGDClasifier and LinearSVC to final optimization step to find the best parameters for them and the vectorizer they will use (I couldn't do that with the four classifiers as that will take much time for my machine)

For the final optimization step, I had very close results for the two models as expected

```
best parameters for classifiers SGDClassifier and TfidfVectorizer is {'vect__sublinear_tf': True, 'clf__alpha': 0.0001, 'clf__c
lass_weight': None, 'vect__min_df': 1, 'vect__analyzer': 'word', 'clf__penalty': 'l2', 'vect__ngram_range': (1, 3), 'vect__stop
_words': None, 'clf__n_jobs': -1, 'clf__fit_intercept': True}
best parameters for classifiers LinearSVC and TfidfVectorizer is {'clf__C': 0.5, 'vect__stop_words': None, 'clf__class_weight':
'balanced', 'vect__min_df': 1, 'vect__analyzer': 'word', 'vect__ngram_range': (1, 4), 'vect__sublinear_tf': True, 'clf__fit_int
ercept': True}
```

	test auc-roc	test f1	test log loss	train auc-roc	train f1	train log loss
SGDClassifier	0.788502	0.715543	5.691319	0.994909	0.993164	0.151595
LinearSVC	0.802894	0.729262	5.808683	0.996407	0.993418	0.146706



For these close results, I will use sgd model because of its low log-loss results

Refinement

For the final model (sgd model) I've used few parameters to tune the model and vectorizer, by defining the ngram_range from 1 to 3 words the auc-roc increased from 76.2 to 78.8 and log-loss decreased from 6.06 to 5.71

IV. Results

Model Evaluation and Validation

The final model I used is SGDClassifier (linear classifiers with stochastic gradient descent), I have used K-fold cross-validation on the training data to obtain best features with the highest f1-score and to be sure there is no overfitting and to have the best parameters within reasonable training time. Also, the Kaggle competition score is based on unseen data that are not accessible to anyone.

If I can get similar auc-roc value in my training and the "unseen" testing data in the competition, I could make it to the top twenty

Justification

The final result is better than the benchmark model.

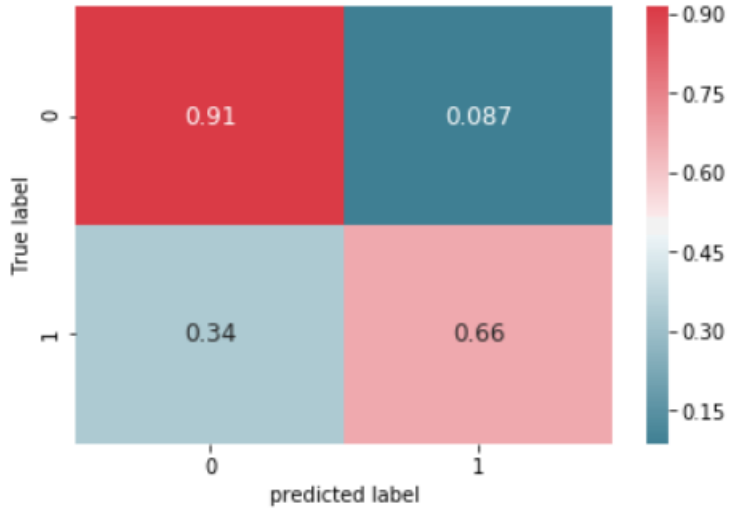
I think SGD model has done a good job with the help of TfidfVectorizer.

In my project, benchmark model has log-loss 6.4 and my model is 5.69 on the test data while the auc-roc of the benchmark is 76.1% and my model is 78.9%, I believe it's somehow good results for solving this problem and could help in detecting insults

V. Conclusion

Free-Form Visualization

Here is normalized confusion matrix on heatmap that describes how the model dealt with the test set



There are a lot of False Negatives; about the third of the insults were classified as neutral, maybe due to the small number of insult comments compared to the number of neutral comments

Reflection

This problem was very challenging especially with totally uncleaned data.

The tricky part was to clean the data and choose appropriate model to handle tfidf vectors well, I tried many different solutions and many different models but they were too bad to be included in the final solution like the SVM, Bernoulli naïve Bayes, K neighbors classifier and Random forest, I tried character analysis but it decreased the score of the most models, I have tried to choose wider range of parameters to optimize, but the training took very long time and didn't yield effective change in scores

What really interested me in the project was:

- Preprocessing not increased only the model score but also the vectorizer speed
- Data Analysis, especially by visualization helped a lot in preprocessing by defining the frequency of the most common words and realizing what is really important and what is not
- The hybrid ML algorithms like LinearSVC and SGDClassifier give better scores than the pure ML algorithms like SVM and LogisticRegression

Improvement

To improve my model in the future, I will do more text analysis for all comments, I will use some techniques to determine the best features only like chi^2 or PCA technique.

I don't know if that possible but If I could do feature stacking and apply more than one n-gram type and choose the best features from each one .

If computing power allowed, I can also try doing deep learning neural network on this problem, since we can extract more and more features and try to use CNN or RNN to see what happens.