# Classifying Reddit posts as NSFW or SFW

## Final Report

## Abstract

The goal of this project was to build classifiers that could classify a given text as NSFW or SFW. The dataset used here is taken from Redditt. Reddit caters to a large and diverse audience, and its posts may or may not include ones that contain profanity, violence, gore, nudity, intense sexuality, or other potentially disturbing subject matter. A good classifier should be able to classify posts as NSFW or SFW. Several approaches were used to build the classifier. This repost lists them all and makes a detailed comparison between them.

## a. How was the data collected?

The data was collected directly from Reddit using their developer API. The reddit posts usually contain a title, and either a picture or text. The posts with a picture are ignored and only the ones with text are taken. This was done using a python script that captured live data. The script checks if the data contains a text or not and accepts the same. Reddit categorizes the posts using an entity called subreddits. Subreddits are like discussion forums that are used filter out posts by category. The dataset does not associate with a particular subreddit, in fact, it is collection of posts from several subreddits.

Since the objective of the project was to classify posts as NSFW or not, it was important the support of the labels was sufficient to build a good classifier. The Reddit developer API requires configuration to show posts that are also NSFW.

The python script which was using the Reddit developer API captured the live posts. The script listens for reddit posts in real time, each time a reddit user posts it is captured by the script. The script then filters out the posts that do not contain pictures. Since a large proportion of reddit posts are picture based, it takes some time collect the data. The script was run in several intervals of time and the data was collected in various chunks. These chunks were later merged into a single file.

Avoiding duplicates is also important for the data to be of good quality. Following were the steps taken to avoid duplication of the data.

i. Filtering out the data marked as **stickied**. A sticky data is the one that is flagged to always appear at the top of the results. This would mean that each time the data is collected certain posts may appear again and again. The script filtered out these posts using a property, provided by the developer API, of the post itself.
ii. The Subreddit was set to ‘all’. This ensured that the API itself did not cluster the posts. The script listened for live posts and accepted any valid post that passed the checks.

Following were the references used while collecting data:

[1] https://www.reddit.com/dev/api/

[2] https://github.com/reddit-archive/reddit/wiki/API

[3] https://praw.readthedocs.io/en/latest/

## b. how was the data labeled?

Reddit posts that contain text are an average about 100 words long. However, some posts can be as long as 500 words. A human annotator can easily identify the label using the following methods

i. Looking at the title of the posts. A majority of the time the title of the post is sufficient to classify the data. The title may contain words that are inappropriate. Sometimes the title itself has the word “NSFW”. Also, the title may contain names of subreddits (or a short form of it) that post only NSFW content.

ii. The text itself can be used to classify the data point as NSFW or not.

iii. The subreddit of the posts. Certain subreddits always appear in the NSFW section. However, this is not used to classify the data as the above two methods are sufficient.

Following is a sample of how the data looked like before being annotated.

```
{

  "text": "hi may i know if i could withdraw btc from bitstamp to an electrum bech32 segwit wallet? it seems that there's no btc bech32 segwit support on bitstamp. can't find any info from your faq.  hope to hear from you guys soon. thanks.",

  "title": "btc withdrawal to electrum bech 32 segwit wallet"

}

{

  "text": "i'm an experienced dom, several years, looking for an almost no limits slut/slave/fuck toy to come to me and serve. due to weird work schedule i'm free during the weekdays only. list of kinks and fetishes is long but a quick summary (raceplay, ageplay, slavery, free use, body worship) most links are flexible and negotiable. white, 39, 6ft, blonde, blue eyes. in san jose and can host. please be fit or well endowed up top, and able to travel. non white slaves to the front but will consider anyone if chemistry is right. feel free to pm me with questions or interest.",

  "title": "39 [m4f] ca bay area dom seeks slut slave for daytime play"

}
```

The python script that was used to collect the data also contained the title of the post. Although the title itself was not used in classification. The data was collected over a period of two months and annotated manually by the project team. The python script simplified the data collection part which help in speeding up the annotation part significantly.

## c. Description of the classifier approach

There were several classifiers that were built to classify the data. The naive classifier, which simply checks a dictionary of inappropriate words for every data point, is used as a baseline approach. The classifiers that improve upon this are

i. Linear SVM
ii. Logistic regression
iii. AdaBoost
iv. Random Forest
v. Ensemble Voting classifier

## 1. Naive Classifier

In this baseline approach, we have collected 608 inappropriate words (NSFW) in total and saved it in "badwords.txt". We are comparing every unique word from "badwords.txt" with each word in every subreddit text. If the word is present in both the text and "badwords.txt" then, we are classifying it as NSFW (Not Safe For Work) as 1 and if there is no intersection between them, it is classified as SFW (Safe for Work) as 0. The NaïveClassifier Class in classifier.py creates a list of 1s (NSFW) and 0s (SFW) for comparing it with the actual labels and calculate different metrics of the classifier.

Below is the precision, recall, and F1-score of Naive Classifier on train and test datasets for the NSFW Labels:

```
Train
-------------
              precision    recall  f1-score   support

         SFW       0.89      0.93      0.91      7037
        NSFW       0.55      0.43      0.48      1463

   micro avg       0.84      0.84      0.84      8500
   macro avg       0.72      0.68      0.69      8500
weighted avg       0.83      0.84      0.83      8500

Test
-------------
              precision    recall  f1-score   support

         SFW       0.95      0.93      0.94      2271
        NSFW       0.33      0.42      0.37       201

   micro avg       0.89      0.89      0.89      2472
   macro avg       0.64      0.67      0.65      2472
weighted avg       0.90      0.89      0.89      2472
```

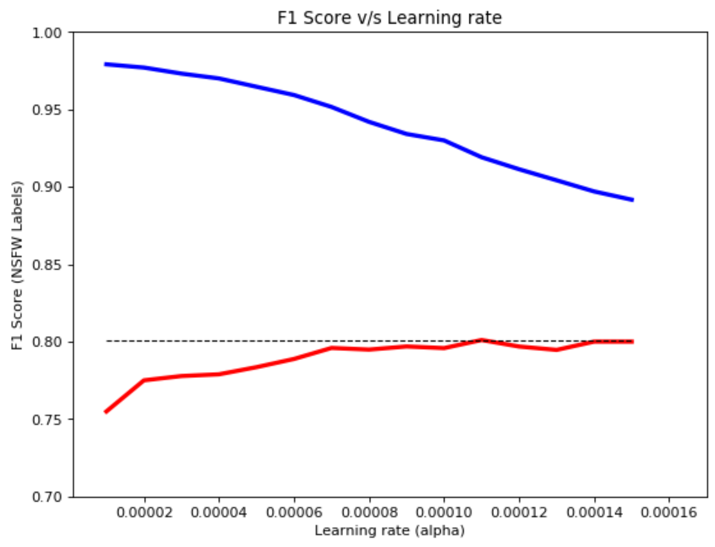
The model seems to classify most of the data as SFW. It can be clearly seen from the F1-score of the SFW labels. However, the model performs poorly on the NSFW labels.

## 2. Linear SVM classifier

Linear SVM (Support Vector Machines) classifier is built using the SGDClassifier from python's sklearn library. A pipeline that sequentially applies the transformation and a final estimator to the dataset, is built using the sklearn Pipeline class.

To begin with the data points are passed as a list of strings. Then the features are extracted from these data points using the TfidfVectorizer class of sklearn. This converts a collection of raw texts into a matrix of TF-IDF vectors. The classifier is then trained using the SGDClassfier, which implements regularized linear models with stochastic gradient descent (SGD) learning. The loss function used in our analysis is hinge loss. The hinge loss is a loss function used for training "maximum-margin" classifiers like Support Vector Machines (SVMs)

The classifier was trained for several values of learning rate in order to obtain the best F1-score for NSFW labels. Following is plot that depicts the results of using different learning rates



Below is the precision, recall and F1-Score of Linear SVM classifier on train and test datasets –

```
Train
-------------
              precision    recall  f1-score   support

         SFW       0.97      1.00      0.98      7037
        NSFW       0.98      0.87      0.92      1463

   micro avg       0.97      0.97      0.97      8500
   macro avg       0.98      0.93      0.95      8500
weighted avg       0.97      0.97      0.97      8500

Test
-------------
              precision    recall  f1-score   support

         SFW       0.98      0.99      0.98      2271
        NSFW       0.85      0.76      0.80       201

   micro avg       0.97      0.97      0.97      2472
   macro avg       0.91      0.87      0.89      2472
weighted avg       0.97      0.97      0.97      2472
```

This approach has the best results compared to other approached defined. The F1-Score for NSFW labels was 0.80. There seems to be a little bit of overfitting as the F1 score for training data is 0.92.

## 3. Logistic Regression Classifier

The Logistic Regression Classifier uses the LogisticRegressionCV from python’s sklearn library. As in the case of earlier classifier, a pipeline that sequentially applies the transformation and a final estimator to the dataset is built using the sklearn Pipeline class. The data points are passed as a list of strings, and the features are extracted from these data points using the TfidfVectorizer class of sklearn. This converts a collection of raw texts into a matrix of TF-IDF vectors.

The LogisticRegressionCV class is an estimator that has built-in cross-validation capabilities to automatically select the best hyper-parameters. The dataset that was vectorized earlier is used to train this classifier.

Below is the precision, recall and F1-Score of Linear SVM classifier on train and test datasets,

```
Train
-------------
              precision    recall  f1-score   support

         SFW       0.99      1.00      1.00      7037
        NSFW       1.00      0.96      0.98      1463

   micro avg       0.99      0.99      0.99      8500
   macro avg       0.99      0.98      0.99      8500
weighted avg       0.99      0.99      0.99      8500

Test
-------------
              precision    recall  f1-score   support

         SFW       0.98      0.98      0.98      2271
        NSFW       0.80      0.75      0.78       201

   micro avg       0.96      0.96      0.96      2472
   macro avg       0.89      0.87      0.88      2472
weighted avg       0.96      0.96      0.96      2472
```

The classifier achieves a F1-Score of 0.78 for the NSFW labels in test set. However, it clearly overfits the data because the F1-score of 0.98 in the train set.

## 4. AdaBoost Classifier

AdaBoost Classifier uses the AdaBoostClassifier class from python’s sklearn library. As in the case of earlier classifier, a pipeline that sequentially applies the transformation and a final estimator to the dataset is built using the sklearn Pipeline class. The data points are passed as a list of strings, and the features are extracted from these data points using the TfidfVectorizer class of sklearn. This converts a collection of raw texts into a matrix of TF-IDF vectors.

An AdaBoost classifier is a meta-estimator that begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases. The dataset that was vectorized earlier is used to train this classifier.

Below is the precision, recall and F1-Score of Linear SVM classifier on train and test datasets,

```
Train
-------------
              precision    recall  f1-score   support

         SFW       0.93      0.99      0.96      7037
        NSFW       0.93      0.63      0.75      1463

   micro avg       0.93      0.93      0.93      8500
   macro avg       0.93      0.81      0.85      8500
weighted avg       0.93      0.93      0.92      8500

Test
-------------
              precision    recall  f1-score   support

         SFW       0.97      0.99      0.98      2271
        NSFW       0.79      0.62      0.69       201

   micro avg       0.96      0.96      0.96      2472
   macro avg       0.88      0.80      0.84      2472
weighted avg       0.95      0.96      0.95      2472
```

The classifier achieves a F1-Score of 0.69 for the NSFW labels in test set. Unlike the other classifiers it doe not seem to overfit the data.

## 4. Random Forest Classifier

The Random Forest classifier uses the RandomForestClassifier from python's sklearn library. As in the case of earlier classifier, a pipeline that sequentially applies the transformation and a final estimator to the dataset is built using the sklearn Pipeline class. The data points are passed as a list of strings, and the features are extracted from these data points using the TfidfVectorizer class of sklearn. This converts a collection of raw texts into a matrix of TF-IDF vectors.

A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. The sub-sample size is always the same as the original input sample size but the samples are drawn with replacement if bootstrap=True (default). The dataset that was vectorized earlier is used to train this classifier.

```
Train
-------------
              precision    recall  f1-score   support

         SFW       0.99      1.00      0.99      7037
        NSFW       0.99      0.96      0.97      1463

   micro avg       0.99      0.99      0.99      8500
   macro avg       0.99      0.98      0.98      8500
weighted avg       0.99      0.99      0.99      8500
```

```
Test
-------------
              precision    recall  f1-score   support

         SFW       0.97      0.98      0.97      2271
        NSFW       0.76      0.61      0.68       201

   micro avg       0.95      0.95      0.95      2472
   macro avg       0.86      0.80      0.83      2472
weighted avg       0.95      0.95      0.95      2472
```

The classifier achieves a F1-Score of 0.68 for the NSFW labels in test set. However, it clearly overfits the data because the F1-score of 0.97 in the train set.

## 5. Ensemble Voting Classifier

The Ensemble Voting Classifier uses the VotingClassifier from python's sklearn library. As in the case of earlier classifier, a pipeline that sequentially applies the transformation and a final estimator to the dataset is built using the sklearn Pipeline class. The data points are passed as a list of strings, and the features are extracted from these data points using the TfidfVectorizer class of sklearn. This converts a collection of raw texts into a matrix of TF-IDF vectors.

The classifier is built using the classifiers that were described earlier, i.e. Logistic Regression, Adaboost, and Random Forest. The goal is to compare the outputs of the earlier classifiers and use a voting to predict the best results.

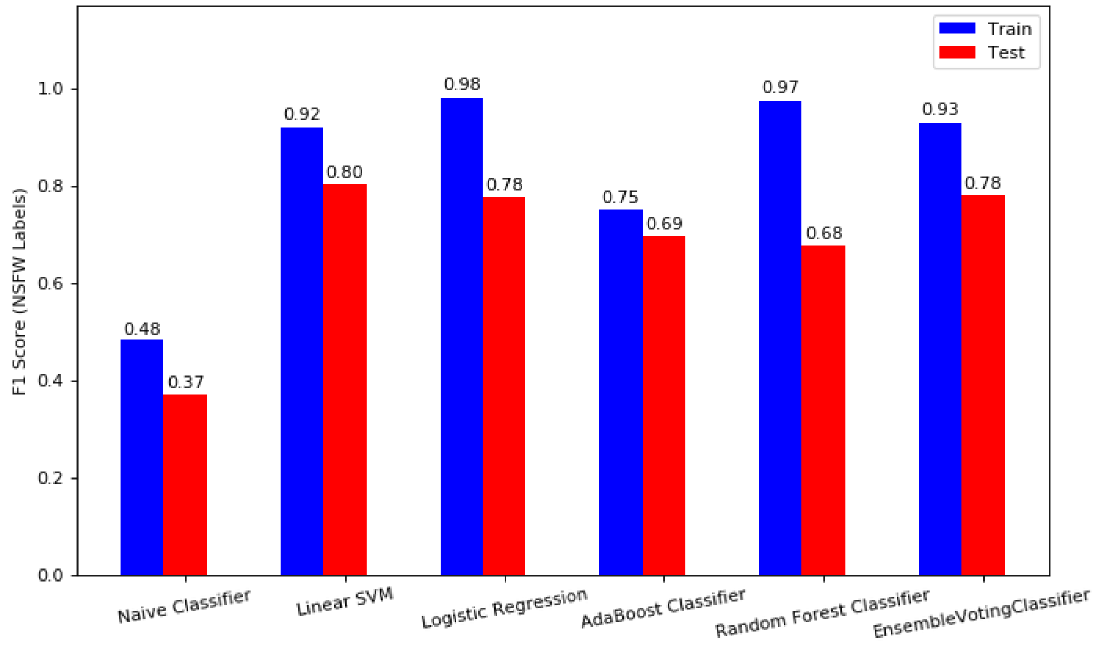
Below is the precision, recall and F1-score of Final Ensemble on train and test datasets -

```
Train
-------------
              precision    recall  f1-score   support

         SFW       0.98      1.00      0.99      7037
        NSFW       0.99      0.88      0.93      1463

   micro avg       0.98      0.98      0.98      8500
   macro avg       0.98      0.94      0.96      8500
weighted avg       0.98      0.98      0.98      8500

Test
-------------
              precision    recall  f1-score   support

         SFW       0.97      0.99      0.98      2271
        NSFW       0.86      0.71      0.78       201

   micro avg       0.97      0.97      0.97      2472
   macro avg       0.92      0.85      0.88      2472
weighted avg       0.97      0.97      0.97      2472
```

This approach does not seem to perform better than the logistic regression, because the other two classifiers are not performing as good as the former.

## d. An analysis of the results (preferably with charts and figures) showing comparison to at least one reasonable baseline approach.

The base line approach, as discussed earlier, naïve classifier which simply checks a dictionary of inappropriate words for every data point. A total of five classifiers were used to improve upon this. The parameter for comparison used is the F1-Score for NSFW labels. Since the dataset contains a large number of SFW labels, every classifier seems to do a good job in labelling them, including the naïve classifier. Hence, the differentiating factor among these classifiers is how they perfom on the NSFW labels. Following is a comparison of the approaches described above,



Looking at the comparison chart following conclusions can be made,

i. Linear SVM achieves the best results with an F1-Score of 0.80 on the test set
ii. Logistic regression also does well but overfits the data
iii. Random Forest classifier has the highest overfitting and AdaBoost the least.
iv. Ensemble voting classifier has very similar performance compared to Linear SVM