Emotion Analysier for Text

(EAFT)

**By:**

**N17366337 Sheng Zhao**

**N18927927 Shuwei Zheng**

**N12614619 Yuanzhe Liu**

Dec 2020

**CONTENTS**

**1) Introduction**

**2) Technical Summary**

**3) Performance Metrics**

**4) Team Member Roles**

**5) Reference**

1. **Introduction**
   * **Introduction**

A movie comments classifier and predictor where criticisms and applause are clearly presented. This classifier has unique features, which can be used to predict the specific score of particular comments from 0 to 10 (which are 'Reject','','Trash(0-2)','','Mediocre(2-4)','','Indifferent(4-6)','','Good(6-8)','','Masterwork(8-10)') and to identify people's feelings and emotions out of 4 (which are 'Fierce','Sadness','Happiness','Reject'). The accuracy is above 70% according to the tfscore.

We crawled 50000 comments from IMDB, and we manually cleaned and relabeled the. With high volume samples and accurate labels, our project reachs a high accuracy.

We also build a GUI to classify the emotion and score of a customized and specific move comment.

1. **Technical Summary**
   * **Overall Project Architecture and Design**
     + For both Emotional Classifier and Score Classifier, we input the most part (90%) of classified dataset as training data to train the classifier, then let the classifier work on the left dataset, to get the output (emotion and score labels) and the accuracy.
   * **Software**
     + Compositive Develop Environment: Anaconda3 (64-bit）
     + Programming Language: Python
     + IDE:Jupyter Notebook and Pycharm
     + GUI: QT designer
   * **Python packages:**
     + NumPy: a library for the Python programming language.
     + Pandas: a open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.
     + The Natural Language Toolkit (NLTK): a suite of libraries and programs for symbolic and statistical natural language processing (NLP) for English written in the Python programming language.
     + Scikit-learn: an open source machine learning module for Python that builds on the NumPy, SciPy, and matplotlib modules.
   * **Algorithms (Classifiers’ models):**
     + TFIDF Transformer: In information retrieval, tf–idf or TFIDF, short for term frequency–inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus.
     + Naive Bayes Classifiers: In machine learning, naive Bayes classifiers are a family of simple “probabilistic classifiers” based on applying Bayes’ theorem with strong (naive) independence assumptions between the features. Naive Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. Maximum-likelihood training can be done by evaluating a closed-form expression, which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers.
     + Decision Tree Classifiers: A decision tree is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.
     + Random Forests Classifiers: Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks, that operate 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. Random decision forests correct for decision trees’ habit of overfitting to their training set.
     + Gradient Boosting Classifiers: Boosting is a machine learning ensemble meta-algorithm for primarily reducing bias, and also variance in supervised learning, and a family of machine learning algorithms that convert weak learners to strong ones.
     + Stochastic Gradient Descent Classifiers: A stochastic approximation of the gradient descent optimization and iterative method for minimizing an objective function that is written as a sum of differentiable functions. In other words, Stochastic Gradient Descent tries to find minimum or maximum value by iteration.
     + Support Vector Machines: In machine learning, support vector machines are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis.
2. **Performance Metrics**
   * **The definition of success metrics**
     + In sklearn package, there is a method ‘ classification\_report ()’ that can return the accuracy matrix of classifier. We use this method to get the accuracy which is ‘ f1-score’. Our success metric is ‘f1-score’ over 70%.
   * **Description of Datasets and Experiments:**
     + 1st Demo:
       - Dataset: We call this dataset ‘imdb50000’. It has50000 imdb comments in .txt format which are classified to:
         * Train:

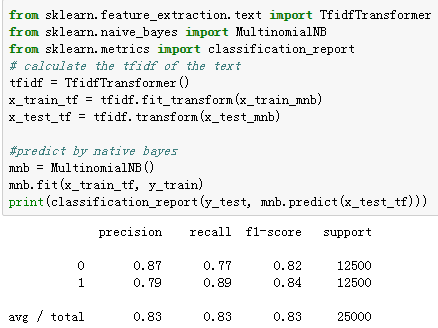
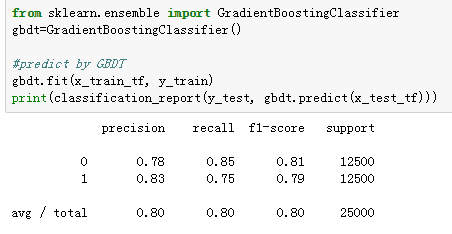
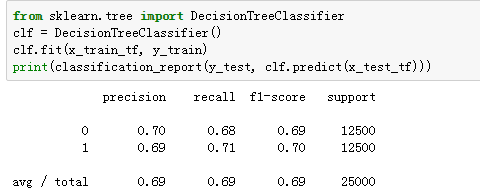
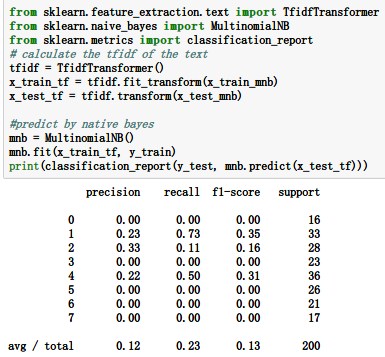
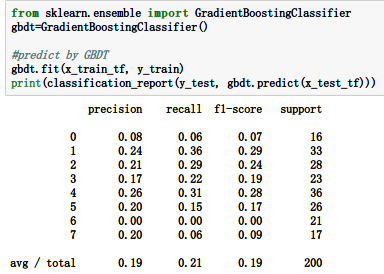
Neg

Pos

* + - * + Test:

Neg

Pos

* + - * Experiment: At first, we produced the first demo of the classifier. Through the learning of 25,000 movie review data, we successfully performed negative or positive classification. The average values of precision, recall, and f1-score all reached 0.83. 
      * Then we improved the previous classifier and added the Model of Support Vector Machines，Decision Tree and Gradient Boosting Classifier. 
      * Until Mid Term, the core functions of Movie Comments Emotions Classifier has finished.
  + 2nd Version:
    - Dataset: Manually classified the ‘imdb50000’ dataset above into the following eight classes:
      * Positive:
        + Good: This movie is good. It achieves the audience's exception.
        + Interested: The movie is funny, interesting. P.S. Always be cartoon, comedy.
        + Amazing: The film is higher than audiences' exception, or it brings new experience to the audience.
        + Moved: The film effects audiences' emotion.
      * Negative:
        + Upset: Like it could be better, or I have higher exception for this film, but it's not good enough.
        + Angry: This is the most fierce emotion. If the audience used some fierce words, like garbage, trash, etc.
        + Disgusted: This is not as fierce as angry, but still means the audience think the movie is foolish.
        + Boring: The movie is easy to predict, same as previous movies, waste time, such like this.
    - Experiment: After that, we improved and rewrote the code to classify them with these eight classes. The accuracy is not good. 
  + 3rd Version:
    - Dataset: To re-category the data, we found a classified Twitter text dataset online, which has 40000 comments and is classified into 13 different classes: anger, hate, fun, happiness, love, relief, surprise, neutral, empty, enthusiasm, worry, sadness and boredom.

Then we re-category the comments into four classes:

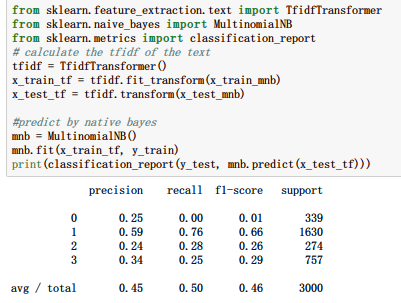
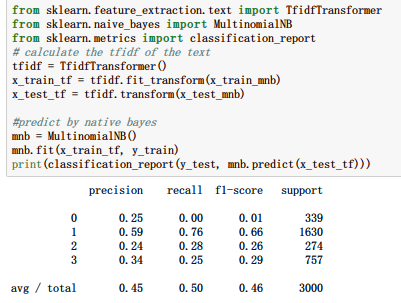
• Fierce: The comment expresses fierce emotion, and the tendency is negative, like anger/ hate/ surprise.

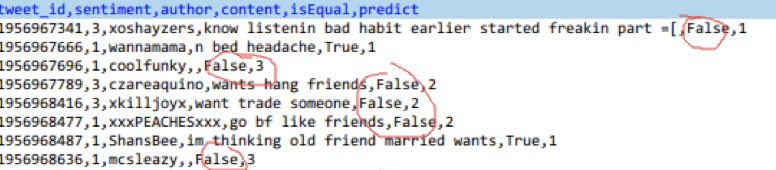
• Happiness: The comment is not fierce, and happy.

• Sadness: The comment is not fierce, but negative.

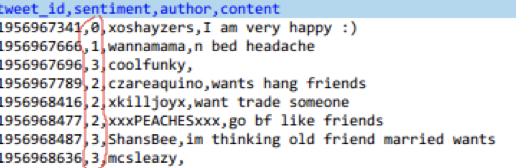
• Reject: The text does not contain any emotion.

And we call this dataset ‘twitter’.

* + - After re-category, we improved and rewrote the code to classify them with these four classes. 
  + 4th Version and successful case:
    - Dataset: We manually reclassify all the 40000 sentiments of the ‘Twitter’ dataset above by the output of the classifier. For example:
      * Before reclassify:



* + - * We manually reclassify the comments whose ‘isEqual’ parameter is False, which means its original sentiment does not match the predicted sentiment by classifier,
      * After reclassify:



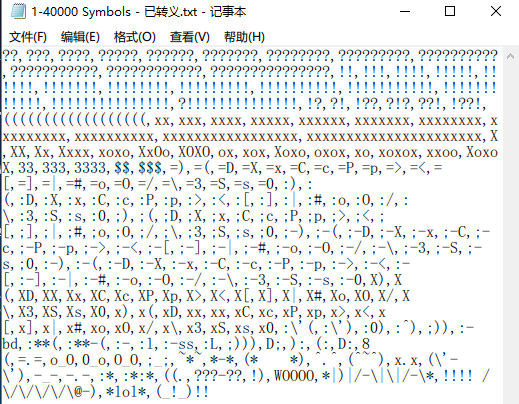
* + - Experiment: What’s more, we noticed that in emotional classification, symbols play an important role in some cases, like:

Why does it snow?”

“Why does it snow???????!!!!!!!!”

The above two sentences express differently: Null and Angry.

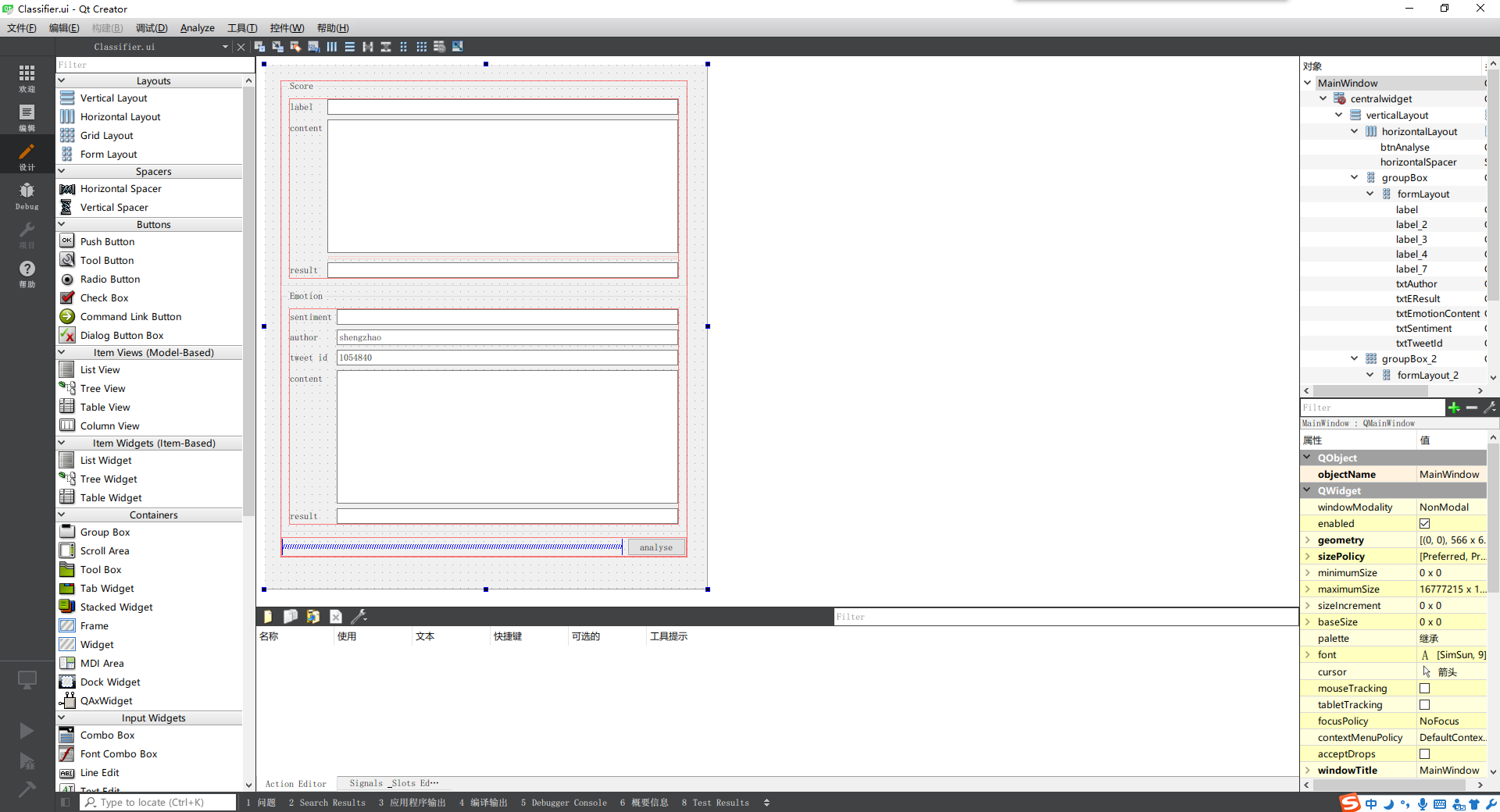
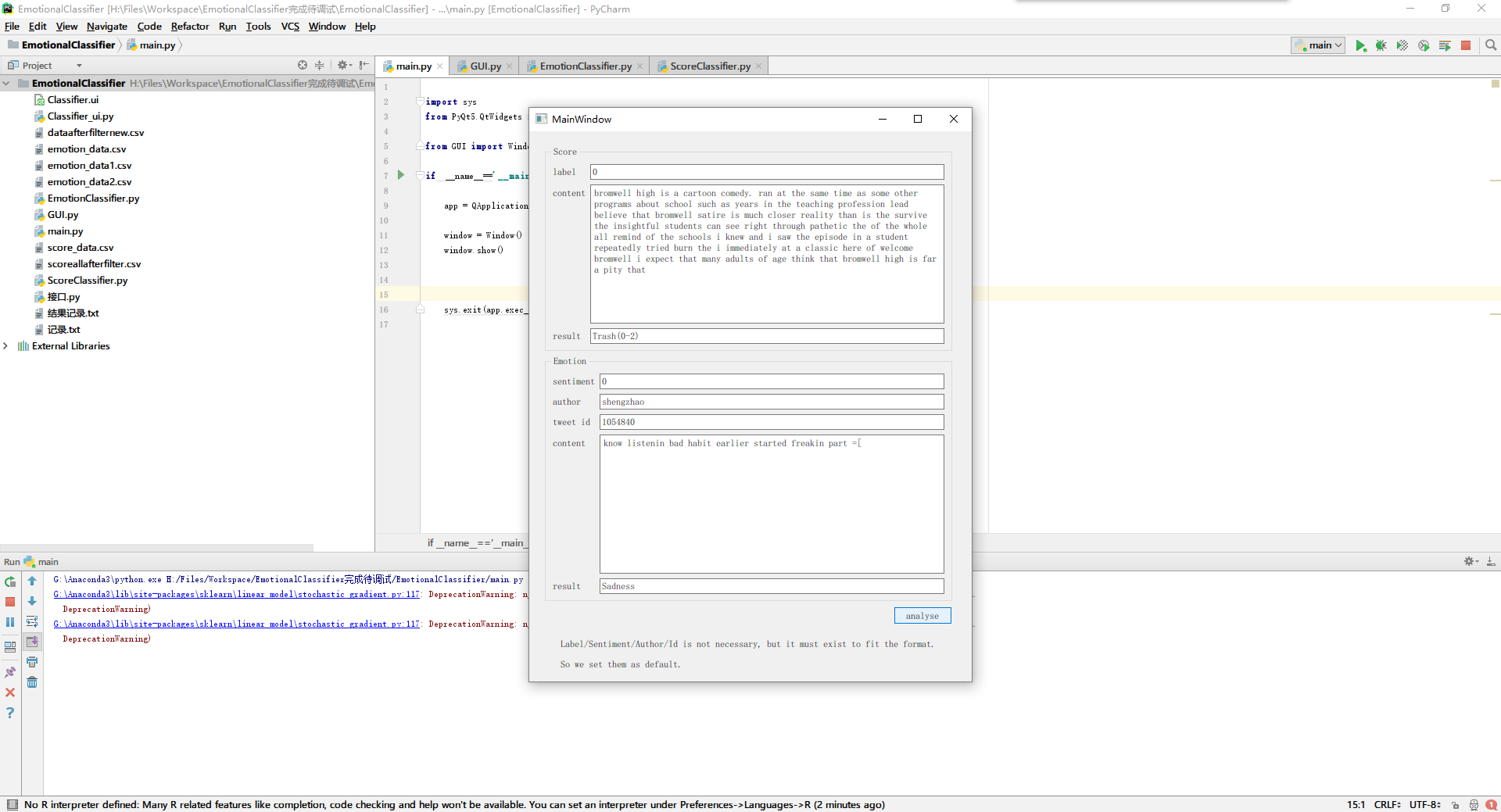
So, we enumerated all the meaningful symbols in Tweet to process them in classifier.



After reclassification, we use the classifier to classify the data and the final accuracy is 72%, which achieves our goal.

图片包含 屏幕截图

已生成极高可信度的说明

* + GUI:
    - We used the QT designer to build the GUI, and abstracted the codes of the classifiers’ function. 
    - After connected the GUI to the classifier and tough debugging, we successfully got the output of our project. 

1. **Team Member Roles**
   1. **Sheng Zhao:** Leader. Project Management, Design the project architecture, Write the final deliverable code, designed make the GUI, Make the dataset’s re-categorizing method, Assign all the work to members and review for deliverable.
   2. **Yuanzhe Liu:** Write the part of sklearn classifiers’ code, Manually classify 20000 datasets of training data. Manually classified all ‘imdb50000’ dataset, Manually classified 20000 for ‘Twitter’ dataset.
   3. **Shuwei Zheng:** Write the part of input code, Manually classify 20000 datasets of training data. Write the part of filter and cooperated in GUI’s designing with Sheng.

1. **Reference**
2. “1.4. Support Vector Machines.” 1.4. Support Vector Machines - Scikit-Learn 0.19.1 Documentation, scikit-learn.org/stable/modules/svm.html.
3. Ray, Sunil, et al. “6 Easy Steps to Learn Naive Bayes Algorithm (with Code in Python).” Analytics Vidhya, 11 Apr. 2018, [www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/](http://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/).
4. Patel, Savan. “Chapter 3 : Decision Tree Classifier - Theory – Machine Learning 101 – Medium.” Medium, Augmenting Humanity, 11 May 2017, medium.com/machine-learning-101/chapter-3-decision-trees-theory-e7398adac567.
5. Patel, Savan. “Chapter 5: Random Forest Classifier – Machine Learning 101 – Medium.” Medium, Augmenting Humanity, 18 May 2017, medium.com/machine-learning-101/chapter-5-random-forest-classifier-56dc7425c3e1.
6. “1.5. Stochastic Gradient Descent.” 1.4. Support Vector Machines - Scikit-Learn 0.19.1 Documentation, scikit-learn.org/stable/modules/sgd.html.
7. “3.2.4.3.5. Sklearn.ensemble.GradientBoostingClassifier.” 1.4. Support Vector Machines - Scikit-Learn 0.19.1 Documentation, scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html.