Abstract: This project involved development of a supervised classification model to predict on the test dataset, which contains 20k comments from different communities on Reddit platform.
The goal was to provide the most accurate predictions of the communities for each test comment. There should be text pre-processing to clean the comments. There are many characters and symbols in the text of comments (especially social media comments).
The classifiers accept numbers and matrices, therefore the cleaned text must be processed and transformed into shapes that classifiers accept.
Bernoulli Naive bayes is the classifiers mentioned by instruction to be used. Also two more classifiers to be added in order to compare the results. As a competition, the best achieved result should be submitted in Kaggle platform to be compared with other students.

Introduction: The training data contained 2 main rows (comments, subreddit) and 60k columns. each column provided a sample comment of the related subreddit.
There was no null value and the considered rows have object data types. The comments row of the train data set has passed to the cleaning function which process the text by different functions.
After converting the text to lower case, we provide a pattern to combine a regular expression pattern into pattern objects. Then we replace the passed pattern with space (" ") in the text. In the next step we replace some of the most misspelled words with the correct in order to provide better vocabulary.
After the cleaning process we pass the text to the tokenizer to prepare a table for the stripping the vocabulary. The strip function eleminates some alphabets (up to 3) from the end of each word in order to bring the vocabulary close to their root. Then by using the stop-words function we eliminate all words which are possible not in English dictionary.
Finally we append all the cleaned words in the same order in the empty list which we prepared before, and the train data-set has been cleaned and prepared.
As a comparison with the strip function, we use the lemmatizer and we look at the most frequent used words in our text.
There are some types of vectorizers available, DicVectorizer, CountVectorizer, but for the case of this project the TfidfVectorizer provided the best result after trying the other possible vectorizers. This vectorizer convert the

words to feature vectors in a matrix. For the next step we apply the values of the parameters on the actual data and gives the normalized value by transforming the text, and here the data is ready for prediction.
By using the train-test-split function from sklearn, we split the data into train and test data-sets. Many runs attempt by adjusting the hyper parameters, and the best performed has been submitted to Kaggle.
Bernoulli Naive Bayes classifiers is mentioned to be used in this project. This classifier has been implemented, and by using the k-fold validation the range of possible accuracy has been calculated.
Out of curiosity the MultinomialNB is also has been used. Interesting fact is that the results of this classifier is always higher than the BernoulliNB classifier.
As mentioned by (C.D. Manning, P. Raghavan and H. Schuetze (2008)), "BernoulliNB classifier is suitable for discrete data. The difference is that while MultinomialNB works with occurrence counts, BernoulliNB is designed for binary/boolean features."
We move to the test data-set after all these processes.
By looking at the data we can see that there is ni null value and the comment row data-type is object. We clean the text by passing the comment column of the test data-set to the cleaning function that we placed earlier. Then by using the defined vectorizer we transform the text to matrix. We predict the y-pred by passing the vectorized input to the classifier.
By using the submission function we create a csv file containing the ID of the comments and the most reliable predict(subreddit) for each column. This submission file has been used for Kaggle competition.
The first option for selective classifiers is the Random Forest classifier. The training and test, data-sets have been passed to this classifier. Its noticeable that by incleasing the max-depth the accuracy score improved to a point where it started decreasing by increasing the max-depth.
A cross-validation score has been placed in order to display the range of the possible accuracy. The results of the croos-validation are somehow close to the main achived result from Random Forest.
The Second option for selective classifiers is the Logistic-Regression. As before the data-sts have been passed to the classifier. The run time required for this classifier is noticeable by comparing to other two classifiers.
The accuracy increased by higher tuning the maximum iteration, to a point that by max-iter=20 the accuracy result has a higher difference with the average of its cross-validation results. The average of the cross-validation of

other two classifiers were close to the single result achieved, but this was not the case in Logistic regression.
In order to have a stable result of single run for each classifier, the random state 42 has been placed.

Related work: Sentiment Analysis is the most common text classification tool that analyses an incoming message and tells whether the underlying sentiment is positive, negative our neutral. This method is really useful for today's businesses to understand the social sentiment of their brand, product or service while monitoring online conversations(Gupta, 2020).

The traditional way of conducting the above task is based on sentiment lexicons . Sentiment lexicons can serve as a word-level basis to help analyze the sentiment of unlabeled documents for the discrete information such as polarities and strengths they contain(Catal and Nangir, 2017). Lexicon-based methods mainly exploited features such as the counts, the total strengths, and the maximum strengths of positive and negative words.
Although such methods have been shown simple and efficient, they suffer from the imitation of existing sentiment lexicons(Chen et al., 2019).

The study on 200k online product reviews demonstrates that, there is a benefit to use higher order n-grams beyond uni and bi-grams(Cui, H., Mittal, V. and Datar, M., 2006). But for the case of this project, there was no improvement in the results by experimenting higher n-grams. As mentioned by the authors this method is useful for large-scale data-sets.

Dataset and setup: The data-set which as a 20-class classification problem with a balanced dataset has been downloaded from Kaggle platform. The data was provided in csv format.
As mentioned in the instruction the data is balanced and comments are enclosed with quotes. Each column of the training data-set (60k columns) contain an ID, comment and the related community to the comment.
There are two rows for the test data-set. The ID and the comment that should be predicted belongs to which community. By looking at the data after importing it, we can see that there is no null value and the data types of the comments and subreddits are objects.

We pass the values if the comments row (train) as a list to lower case function, in order to make all the alphabets lower case.
We provide a pattern in order to differentiate between the useful and not useful values and we add the spacing (" "). As a practice we try to eliminate any possible shapes and emojis from the comment. It is a common practice between the social media users to uses these in their comments. We replace all of them with a space as well.
then we try to replace the most frequent misspelled vocabularies with correct format of spelling. by using the re.sub function we replace the symbols with space.
By using the word tokenizer to split the comments into tokens of individual words for further processing. Then by using the strip function, we try to bring the tokenized vocabularies to their root as much as possible.
In the next step we eleminate all the unnecessary values which are not in English dictionary by using the stop word function.
Then by appending the processed vocabularies to the defined list before, we prepared the data for further processes.

Proposed approach: Bernoulli Naive Bayes wa specified by the project to be used, and additional two more classifiers to be added to the project.
The BernoulliNB was used through the project, but as a comparison, the MultinomialNB was added as well. But no matter how I adjust the hyper parameters for the BernoulliNB, I was not able to achive higher accuracy than the MultinomialNB. Cross validation (k-fold) was implemented for both of the models, and the average of the results indicate the above observation as well. There must be a consideration that we do not over fit the model by simple increasing the hyper parameter.
one of the selective models is Random Forest. Observation is that by increasing the maximum depth (hyper parameter) of the model the accuracy increased as well, which was not correct. But the overall accuracy result was lower then the other two methods.
Logistic Regression is the second selective model chosen for this project. The first impression is that the run time required for this model is much higher then others. By increasing the maximum iterations from 5 to 20 there was a large improvement (0.19) in accuracy, and by looking at the cross validation results we can notice that it has the same issue as Random Forest. But it

seems that this method is not sutable for large scale data-sets. It runs, but it gives a notification that either re scale the data or increase the maximum iteration. Also other observation is that the results of the cross validation is always much lower then the single run, But this is not the case for other models.

Results: Provide results on the different models you implemented (e.g., accuracy on the validation set, runtimes). You should report your leaderboard test set accuracy in this section, but most of your results should be on your validation set (or from cross validation).

Discussion and Conclusion: As mentioned by (Brash, 2015) to overcome the over-fitting, we should optimize a tuning parameter that governs the number of features that are randomly chosen to grow each tree from the bootstrapped data. we do this via k-fold cross-validation, therefore the results of the cross validation (Random Forest, Logistic Regression) indicate the fact that no matter how single run accuracy is high, the cross validation provide the most accurate results.

ummarize the key takeaways from the project and possibly directions for future investigation.

Statement of Contributions: This project has been carried out individually, by the instructions provided by Prof. Brahim Chaib draa, and under the supervision of Dr. Amar Ali Bey.

References: C.D. Manning, P. Raghavan and H. Schuetze (2008). Introduction to Information Retrieval. Cambridge University Press, pp. 234-265. https://nlp.stanford.edu/IR-book/html/htmledition/the-bernoulli-model-1.html

A. McCallum and K. Nigam (1998). A comparison of event models for naive Bayes text classification. Proc. AAAI/ICML-98 Workshop on Learning for Text Categorization, pp. 41-48.

V. Metsis, I. Androutsopoulos and G. Paliouras (2006). Spam filtering with naive Bayes – Which naive Bayes? 3rd Conf. on Email and Anti-Spam (CEAS).

Cui, H., Mittal, V. and Datar, M., 2006, July. Comparative experiments on sentiment classification for online product reviews. In AAAI (Vol. 6, No. 1265-1270, p. 30).

Gupta, S., 2020. Sentiment Analysis: Concept, Analysis And Applications. [online] Medium. Available at: ¡https://towardsdatascience.com/sentiment-analysis-concept-analysis-and-applications-6c94d6f58c17¿ [Accessed 13 July 2020].

Catal, C. and Nangir, M., 2017. A sentiment classification model based on multiple classifiers. Applied Soft Computing, 50, pp.135-141.

Chen, X., Rao, Y., Xie, H., Wang, F., Zhao, Y. and Yin, J., 2019. Sentiment Classification Using Negative and Intensive Sentiment Supplement Information. Data Science and Engineering, 4(2), pp.109-118.

overfitting, R., Equilibrium, B., Ooi, H. and Grigorev, A., 2020. Random Forest - How To Handle Overfitting. [online] Cross Validated. Available at: ¡https://stats.stackexchange.com/questions/111968/random-forest-how-to-handle-overfitting¿ [Accessed 13 July 2020].