Comparative Study of Classification Algorithms to Analyze and Predict a Twitter Sentiment in Apache Spark

1 Wafaa S. Albaldawi, 2 Rafah M. Almuttairi

Rafah@uobabylon.edu.iq ; stud.wafaa.shakir@uobabylon.edu.iq

1Ministry of Higher Education and Scientific Research, Department of Computer Science, College of Science, University of Kerbala, Karbala, Iraq

1,2Ministry of Higher Education and Scientific Research, Department of Software, College of Information Technology, University of Babylon, Hillah, Iraq

Abstract

Nowadays, the major sources of information exchange are Twitter, Facebook, WordPress, etc. The tweets can be considered as the source of the public opinion on an event, a product, or a topic. Consequently, it contains large volumes of natural data. Enormous dataset contains a huge volume and variety of information. Therefore, it cannot be prepared utilizing normal conventional tools. It can be processed by building up distributed environment or by contracting cloud based isolated infrastructure. Therefore, better approaches and instruments are required to bring the respect of the information. Apache spark is actually appropriate for performing machine learning on large-scale information. To discover out how rapidly Spark processes of huge information, we make an approach that utilize Machine Learning library (MLlib) classification algorithms in Apache Spark. We implement Logistic Regression, Multilayer Perceptron (MLP), Random Forest and Support Vector Machine (SVM) and compare among them. The models are for analysis and predicting the sentiment based on a corona tweets. A sentiment analysis based on tweets is a challenging issue. The classification algorithm is assessed by precision, recall, f-measure, accuracy and time consumed. The results show that logistic regression algorithm has higher speed in doing enormous information processing than other chosen algorithms. Moreover, according to the obtained results, apache spark has exceptionally great speed in handling big data. The classification algorithm is assessed by precision, recall, f-measure, accuracy and time consumed.

Keywords: Machine Learning, Multilayer Perceptron, Random Forest, Support Vector Machines, Logistic Regression, Apache Spark, Big Data, Hadoop, Classification, Sentiment Analysis.
Introduction

Technology is one of the important factors that support human improvement, and provide Information. Therefore, there are various data sources such as reviews of product, transaction records of purchase, posting in social media and others. The data can be structured data, semi structured or unstructured format. This becomes an issue in data analysis processes. Big data is the term that used for the huge data. There are three characteristics of it are namely Volume, Velocity, and Variety (3Vs). The characteristics can be considered as a challenge of the system when implementing the machine learning framework. Therefore, a good machine learning model, strategy, and environment are needed to process big data.

Hadoop is one of the big data processing technologies. It is open source software that designed for analytics purposes. It allows distributed processing of big data across multiple machines with a degree of fault tolerance. Hadoop architecture is known as MapReduce framework. It divides, processes the data and run it in parallel. However, there are some defects in Hadoop. MapReduce has a high overhead while running a job. Therefore, Hadoop relatively ill-suited while is used for cases of an iterative or low-latency environment. So, there is a need to develop the framework and solve some issues in hadoop. One of the developed software is Apache Spark that is built on top of Hadoop architecture.

Apache Spark is an open source distributed computing framework. It is designed to perform the low-latency jobs, and store in-between data and results in memory. Therefore, it is called Memory Computing that improves the efficiency of data computing. As a result, Spark is better than Hadoop for data mining and machine learning applications. Moreover, to perform scalable machine learning, graph analysis, streaming and structured data processing, there are upper-level libraries in it. Spark is a cluster computing framework with Scala, Java, Python and R language-integrated APIs. In addition, it extends the MapReduce model to efficiently use more types of computations which includes Interactive Queries and Stream Processing. The superior feature of Apache Spark is RDD (Resilient Distributed Dataset) which offers fault tolerant objects that can be deployed across a distributed cluster.

Machine Learning Library (MLlib) is one of the Apache Spark components. Four MLlib classification algorithms are implemented for building sentiment analysis models. The algorithms are Logistic Regression, Multilayer Perceptron (MLP), Random Forest and Support Vector Machine (SVM).

Based on previous researches, the SVM algorithm is suitable for Text Mining, Opinion Mining, and Sentiment Analysis. In 2017, Etaiwi et al. use in their research “Evaluation of classification algorithms for banking customer behavior under Apache Spark data processing system” Naive Bayes and the SVM algorithm in Apache Spark. The results of comparing them show that Naive Bayes approach is more efficient than SVM. The evaluation metrics are precision, recall and f-measure.

Sentiment analysis has performed in this study. There are several steps of Sentiment analysis. They are data collection, preprocessing, feature extraction, building the classifiers, evaluation of models, choosing the best model to predict the sentiment. Sentiment analysis commonly implicates classifying the text as positive or negative. It is carried out on terms, sentences level and extended
to other parts because of a rule gets the opinion of a single sentence. The goal of sentiment analysis is discovering how individuals feel about a specific topic. 

In 2016, Kharde and Sonawane have compared many of the classification algorithms in Twitter data to perform sentiment analysis and opinion Mining. Performance of Naïve Bayes and Support Vector Machine have been evaluated in the specific domain of tweets. According to the results, SVM model proved to have the best performance.

Srivastava and Bhambhu have used four datasets that are diabetes data, heart data, satellite data, and shuttle data with different number of features. They have classified all datasets by using support vector machine and RSES. In spite of SVM took longer time, SVM was higher accuracy compared to RSES. It could be concluded that the greater the amount of data classified, the greater the value of accuracy predictions.

Authors have discussed a Decision Tree, K-Nearest Neighbor, Apriori and Support Vector Machine and compared on the basis of their performance.

In the study, researchers have demonstrated a result based on feature selection methods before performing classifiers. They have used the MapReduce technique on evolutionary computation fundamental and Apache Spark technology. SVM, Logistic regression classifiers and Naïve Bayes for the evaluation of the feature selection method have been chosen.

In terms of the classification algorithm with the framework, Apache Spark has better results than some others. Gilheany has used Frequency-Inverse Document Frequency (TF-IDF) process to compare processing times in Apache Spark and Hadoop. He has found that Spark has a more stable processing time than Hadoop. In contrast, when he has performed the process by using the Naïve Bayes algorithm, the results have viewed that Spark has a slower time than Hadoop. The results of three iterations when performing the process, Spark has an improved processing time from previous iterations.

Authors has performed a sample analysis by using Machine Learning methodologies alongside with NLP techniques and utilizing Apache Spark’s Machine learning library, MLlib, on a labeled (positive/negative) with simulated stream of data from Kafka database.

Garcia-Gil et al directed a study titled “A comparison on scalability for large batch data processing on Apache Spark and Apache Flink”. They have compared the scalability of Apache Spark and Flink by using SVM, Linear Regression and DITFS algorithms on the same dataset. In terms of learning time, experimental results show that all algorithms using Apache Spark faster than Flink.

In 2017, Assefi et al have focused on a SVM, Decision Tree, Naïve Baye, and Random Forest classification algorithms using Apache Spark MLlib 2.0. Then, they have compared the models to the same algorithms in Weka. The outcome shows that Apache Spark MLlib has a robust tool for big data processes.
The authors of\textsuperscript{17} have developed a novel method for sentiment learning in the Spark framework. Their system exploits the hashtags and emoticons inside a tweet to get sentiment labels. Moreover, they have processed a diverse sentiment type’s classification procedure in a parallel distributed environment. In addition, they have used Bloom filters to compact the storage size of intermediate data.

The purpose of this research is to obtain a sentiment analysis system and compare results among Logistic Regression, MLP, and Random Forest and SVM classifiers under the Apache Spark framework. Precision, Recall, F-Measure, and confusion matrix are used for evaluation. This study can be considered as a reference for future studies in defining the suitable classification algorithms. Moreover, we want to proof how excessive and fast Apache Spark in big data processing.

Proposed Methodology

This study is performed several steps. Figure 1 shows the steps of the study. Figure 1 is drawn by using Dia Diagram Editor. It is installed in Ubuntu 16.04. It is an open source graphic design and editing application and a nice open source Visio alternative for Linux.

The fifth stage is to perform an evaluation based on the machine learning classification algorithm used.

The five stages are:

1. Dataset Collection of Tweets
2. Data Preprocessing
3. Feature Extraction
4. Building Machine Learning Model
5. Models evaluation

Data Collection

We download COVID-19 Tweets Dataset (corona_tweets_18.csv and corona_tweets_19.csv) from ieee-dataport. The dataset is available in the link https://ieee-dataport.org/open-access/corona-virus-covid-19-tweets-dataset. This dataset includes CSV files. Schema of the CSV files: tweet ID, and Sentiment score for the particular tweet. We removed the records that have the negative sentiment score from the downloaded CSV files. Also, we rounded the real values to integer values because we perform the binary classification. Therefore, we need positive integer number as 0 or 1.

To get the authorization to view and print the tweets’ text, we have to get the authorization of the website. In our research, the website is the twitter. OAuth is an open protocol to permit secure authorization in a straightforward and standard strategy from web, mobile and desktop applications. This system allows users to allow their consent to act on their sake without sharing the account password. After the client has given permission, OAuth will return a token, and this token itself awards get form demands on sake of the client\textsuperscript{(18,19)}. 
We have used the code below to view 12 tweets text using their tweets' IDs. This code is part from the whole code. Then, we save the output to CSV file and used it as training dataset. After that, we delete all the ‘;’ or ‘,’ from the tweets' texts because the ‘;’ and ‘,’ used to make new column in CSV file.

![Diagram showing the steps of the Sentiment analysis of tweets data by using four algorithms (Random Forest, SVM, MLP, and Logistic Regression).](image-url)

Figure 1 the Diagram shows the steps of the Sentiment analysis of tweets data by using four algorithms (Random Forest, SVM, MLP, and Logistic Regression).
import tweepy as tw
import pandas as pd
import json

# App Auth
CONSUMER_KEY = 'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx'
CONSUMER_SECRET = 'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx'
OAUTH_TOKEN = 'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx'
OAUTH_TOKEN_SECRET = 'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx'
auth = tw.OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
auth.set_access_token(OAUTH_TOKEN, OAUTH_TOKEN_SECRET)
api = tw.API(auth)
tweet1 = api.get_status(1244539115154046976)
tweet2 = api.get_status(1244539146162462720)
tweet3 = api.get_status(124453921455763460)
tweet4 = api.get_status(1244539476673626112)
tweet5 = api.get_status(1244539751803375623)
tweet6 = api.get_status(1244539781683359746)
tweet7 = api.get_status(12445399795377922048)
tweet8 = api.get_status(12445399948885086208)
tweet9 = api.get_status(124453948906229763)
tweet10 = api.get_status(1244539948906229763)
tweet11 = api.get_status(1244540044011945984)
tweet12 = api.get_status(1244540261449060354)
print(1244539115154046976, ',', 1, ',', tweet1.text)
print(1244539146162462720, ',', 1, ',', tweet2.text)
print(124453921455763460, ',', 1, ',', tweet3.text)
print(1244539476673626112, ',', 1, ',', tweet4.text)
print(1244539751803375623, ',', 1, ',', tweet5.text)
print(1244539781683359746, ',', 1, ',', tweet6.text)
print(12445399795377922048, ',', 0, ',', tweet7.text)
print(12445399948885086208, ',', 0, ',', tweet8.text)
print(1244539948906229763, ',', 0, ',', tweet9.text)
print(1244540044011945984, ',', 1, ',', tweet10.text)
print(1244540261449060354, ',', 1, ',', tweet12.text)

Tweets are successfully obtained from twitter as much as 8000 records. The tweets will be processed with Apache Spark. Because of the data is unbalanced data, we need to "under-sample" the negative class. The logistic loss objective function should treat the positive class (label == 1) with higher weight. We add a new column to the data frame for each record in the dataset for generating the weight in Scala code.

If number of records is 1000, the down sampling is done as below code:

val fractions = Map("0" ->0.122334456, "1" -> 1.0)
val df = train_df.stat.sampleBy("Sentiment", fractions, 12345L)
df.groupBy("Sentiment").count.show()

In addition, when the data has 2000, 4000, 8000 records, the code have to be changed as following respectively:

val fractions = Map("0" ->0.123595506, "1" -> 1.0)
val fractions = Map("0" ->0.100110011, "1" -> 1.0)
val fractions = Map("0" ->0.102383905, "1" -> 1.0)
Data Preprocessing

The preprocessing of the data can be considered as the most important stage. The aim of its steps to make the data more understandable by machine. Therefore, uncertainty is decreased in feature extraction. Moreover, we change the streaming input to data frame to perform preprocessing pipeline that contains preprocessing and feature extraction.

Not all tweets texts are consistent in the use of capital letters. Case folding is used to change all letters in the tweets into lowercase. ‘a’ to ‘z’ are accepted only.  

Data Cleaning

At this stage, we check text based on existing tweets. This step used to remove punctuation and corrects words that can damage the actual expression.

We utilize it in pipeline. It replaces all the substrings coordinating the Regular Expressions (@[a-zA-Z0-9_]+, "&(lt)?(gt)? (amp)?(quot)?;") to delete the nickname and html labels from the tweet. We are doing it to maintain a strategic distance from handling of words that are not related to the sentence that the algorithm will analyze.

Tokenizing

The results from Tokenizer function are the number of tokens in each sentence.

Stop Word Removal

Within the filtering stage, we utilized stop word algorithm to delete the word less imperative or wordless. Stop words are non-descriptive words that can be displaced within the bag-of-words approach. It clears articles, prepositions, conjunctions, pronouns, as they are not semantically vital to characterize the opinion. Stop word removal can reduce index size and processing time. In addition, it can also reduce the noise level.

Tagging Positive or Negative

The process of classifying the sentence as positive or negative can be done by calculating the number of positive words or negative words in each sentence divided by the number of words in that sentence. When the result shows that the negative count is more prominent than negative value, the sentence is negative and vice versa. The value of negative is 0 and the value of the positive that is 1.

Feature Extraction

At this stage, the system converts the set of sentences within the existing tweets into a collection of vectors.  

Word2Vec vectors the words that are semantically critical. In addition, normalizer brings the values to the same scale.

Classification Methods
We utilized four models for classification: Logistic Regression, MLP, SVM and Random Forest. The output is that several CSV files with data in DateTime, Sentence and Class columns.

For training, train.csv and test.csv dataset were utilized. We used train.csv file with train validation split for testing. This dataset comprises of tweets and their class either positive (1) or negative (0).

Hyper parameter tuning is used to find the best parameters to get the highest accuracy value for a specific model. We have utilized GridSearch with 5-fold Cross Validation on train.csv. Then, the output is the best accuracy of the model and its parameters. This output of all models is stored in HDFS as modelname.model within the catalog (/user/spuser) and will be stacked within the class of opinion analyzer for expectation.

Evaluation Method

To evaluate the performance of implemented classifiers, entropy, purity, confusion matrix, accuracy, precision, recall, F measure, and computation time can be used as the evaluation metrics.

In 2017, authors have used accuracy and training time as evaluation metrics to compare the performance among few applied classifiers on massive datasets that have various types and sizes.23

According to Saito, et al18, evaluation metrics are precision and recall for binary classifiers. The values of both precision and recall are used to calculate the F-measure.

We evaluate the performance of the classification algorithms by several evaluation metrics: accuracy, F1 score, precision, recall, Confusion Matrix and computation time.

Results and Discussion

This section discusses how to perform the data processing and test the Logistic regression, Random Forest, MLP and the SVM algorithms, so that various evaluation metrics on all the models are resulted.

We create the data frame from the dataset. The data frame contains a row and column. It likes a variable. This process takes milliseconds of time.

The obtained results from Tokenizer function are the number of tokens in each sentence. Then, Stop words Remover is useful for stop word deletion process. Then, we classify the sentences as have positive or negative meaning. In this step, a positive English tweet and a negative tweet in English language are used. In feature extraction, spark convert the words in to a vector list by applying machine learning process.

We tested four models namely Logistic Regression, Random Forest classifier, MLP classifier and the SVM. To find out how fast Apache Spark doing data processing using the algorithms, the experiment is executed in 4 times 1000, 2000, 4000 and 8000 tweets.

| Algorithm            | Time taken 1000 records | Time taken 2000 records | Time taken 4000 records | Time taken 8000 records |
|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
| Logistic Regression  | 15s                      | 15s                      | 15s                      | 16s                      |
| Random Forest        | 16s                      | 16s                      | 16s                      | 19s                      |
Table 1 shows the results obtained based on processing time in the Logistic Regression, Random Forest, MLP and SVM algorithms. The results obtained in Table 1 shows that Logistic Regression has a best speed compared to other models in classifying data. The time taken for each algorithm are captured.

There are a positive relation between the increasing the amount of data and the processing time. In contrast, it does not have an impressive time.

Table 2 shows results obtained based on the evaluation metric (Accuracy) in the 4 algorithms.

| Algorithm   | Accuracy 1000 records | Accuracy 2000 records | Accuracy 4000 records | Accuracy 8000 records | Average  |
|-------------|-----------------------|-----------------------|-----------------------|-----------------------|---------|
| Logistic Regression | 0.6388888888888888   | 0.6027397260273972   | 0.7096774935483687  | 0.7091633466135459  | 0.665117345 |
| Random Forest    | 0.5555555555555556   | 0.6438356164383562   | 0.6612903225806451  | 0.730677290836654   | 0.648437306 |
| MLP           | 0.5833333333333334   | 0.5616438356164384   | 0.677413548387096   | 0.6972111553784861  | 0.62990192 |
| SVM           | 0.69444444444444444  | 0.6164385616438563   | 0.7338706774193555  | 0.6733067729083665  | 0.679515135 |

Table 3 shows results obtained based on the evaluation metric (F1-score) in the 4 algorithms.

| Algorithm   | F1-score 1000 records | F1-score 2000 records | F1-score 4000 records | F1-score 8000 records | Average  |
|-------------|-----------------------|-----------------------|-----------------------|-----------------------|---------|
| Logistic Regression | 0.6666666666666667   | 0.6576141791047755   | 0.6785714285714286   | 0.693884773662552   | 0.652997688 |
| Random Forest    | 0.5555555555555555   | 0.6578947368421052   | 0.6818181818181818   | 0.714936170212766   | 0.636631432 |
| MLP           | 0.6511627906976745   | 0.6610169491525424   | 0.685904132321345    | 0.624532538          | 0.624532538 |
| SVM           | 0.7317073170731707   | 0.611111111111111111| 0.5079646017699116   | 0.6611570247933884   | 0.675985014 |

Table 4 shows results gotten based on the evaluation metric (Confusion Matrix) in the 4 algorithms.

| Algorithm   | Confusion Matrix 1000 records | Confusion Matrix 2000 records | Confusion Matrix 4000 records | Confusion Matrix 8000 records | Average  |
|-------------|-------------------------------|-------------------------------|-------------------------------|-------------------------------|---------|
| Logistic Regression | 13.0 10.0 6.0 7.0 10.0 19.0 25.0 | 19.0 13.0 20.0 38.0 40.0 16.0 50.0 | 38.0 20.0 50.0 85.0 40.0 10.0 | 85.0 32.0 41.0 93.0 | 0.785 19.0 44.5 |
| Random Forest    | 10.0 9.0 7.0 10.0 7.0 22.0 20.0 | 12.0 10.0 40.0 16.0 18.0 48.0 | 34.0 24.0 48.0 84.0 34.0 | 94.0 34.0 10.0 | 0.785 19.5 45 |
| MLP           | 14.0 5.0 10.0 7.0 10.0 20.0 21.0 | 11.0 25.0 45.0 10.0 50.0 | 39.0 19.0 20.0 43.0 12.0 | 83.0 34.0 42.0 92.0 | 0.785 19.5 45 |
| SVM           | 15.0 4.0 7.0 10.0 15.0 13.0 20.0 | 15.0 23.0 15.0 51.0 | 40.0 18.0 90.0 37.0 45.0 | 39.25 18.5 20 42.25 | 0.785 19.5 45 |

Table 5 shows results gotten based on the evaluation metric (Recall) in the 4 algorithms.

| Algorithm   | Recall 1000 records | Recall 2000 records | Recall 4000 records | Recall 8000 records | Average  |
|-------------|---------------------|---------------------|---------------------|---------------------|---------|
| Logistic Regression | 13.0 10.0 6.0 7.0 10.0 19.0 25.0 | 19.0 13.0 20.0 38.0 40.0 16.0 50.0 | 38.0 20.0 50.0 85.0 40.0 10.0 | 85.0 32.0 41.0 93.0 | 0.785 19.0 44.5 |
| Random Forest    | 10.0 9.0 7.0 10.0 7.0 22.0 20.0 | 12.0 10.0 40.0 16.0 18.0 48.0 | 34.0 24.0 48.0 84.0 34.0 | 94.0 34.0 10.0 | 0.785 19.5 45 |
| MLP           | 14.0 5.0 10.0 7.0 10.0 20.0 21.0 | 11.0 25.0 45.0 10.0 50.0 | 39.0 19.0 20.0 43.0 12.0 | 83.0 34.0 42.0 92.0 | 0.785 19.5 45 |
| SVM           | 15.0 4.0 7.0 10.0 15.0 13.0 20.0 | 15.0 23.0 15.0 51.0 | 40.0 18.0 90.0 37.0 45.0 | 39.25 18.5 20 42.25 | 0.785 19.5 45 |
Table 6 shows results gotten based on the evaluation metric (Precision) in the 4 algorithms.

| Algorithm    | Precision 1000 records | Precision 2000 records | Precision 4000 records | Precision 8000 records | Average    |
|--------------|------------------------|------------------------|------------------------|------------------------|------------|
| Logistic Regression | 0.6842105263157895 5 | 0.5135135135135135  | 0.65517241379310 34 | 0.72649572649572 65 | 0.644848045 |
| Random Forest | 0.5263157894736842 2 | 0.6756756756756757  | 0.58620689655172 41 | 0.71794871794871 8 | 0.62653677  |
| MLP          | 0.7368421052631579 9 | 0.45234232432432432426 83 | 0.67241379310344 94 | 0.70940170940170 94 | 0.63777251 |
| SVM          | 0.7894736842105263 3 | 0.5945945945945945945946 31 | 0.68965517241379 38 | 0.68376068376068 38 | 0.689371034 |

Table 6 shows results gotten based on the evaluation metric (Precision) in the 4 algorithms.

Table 2, Table 3, Table 4, Table 5 and Table 6 show the results gotten based on the Accuracy, F1-score, Confusion Matrix, Recall and Precision for the four models respectively. The experiment is performed in 4 times with various amount of data, 1000, 2000, 4000 and 8000 tweets.

The four algorithms have good results of accuracy, f-measure, confusion matrix, recall, and precision. Also, in the test process it can be seen that the fourth time has the higher value of each evaluation metric (accuracy) in the three algorithms logistic regression, random forest, and MLP as shown in Table 2.

Table 3 shows the F1-score results and the average of it of the four times of each algorithm. The results show that SVM is better with average 68% compared to logistic regression with average of recall 65%, random forest with mean value of 64%, and MLP with average value 62%.

Table 4 shows the results of the confusion matrix in the four algorithms. The results show that the True Positive value is higher than False Positive and False Negative, but less than True Negative in all models.

Table 5 shows the recall results and the average of recall of the four times of each algorithm. The results show that SVM is better with average 69% compared to logistic regression with average of recall 64%, random forest with mean value of 63%, and MLP with average value 64%.

Table 6 shows the precision value of evaluation metrics of all four algorithms. It can be seen that SVM is the best classifier with average of 67% compared to Logistic regression with average of 67%, random forest with mean value of 65%, and MLP with average value 62%.

The best model from our experiment is SVM model as it shows the best performance on validation and stream data, and F1-score show that these accuracies are achieved without over fitting. The average of accuracy of SVM algorithm in streaming data is 60%.

The stream tweets and the outputs from all the four models as following.

The input stream tweets as below:
While the label is equal to 1,1,1,0,0,0,0,0, the predicted column is equal to 1,1,0,0,0,1,1,0 respectively. Therefore, 5 of 8 predicted sentiments are correct. This output is found when the data has 8000 tweets.

In MLP, the cross validation is expensive computationally. Therefore, there is no need for it. We have done the cross validation in logistic regression and find the best parameters which we should change them in model manually. Therefore, we have got the accuracy equals to 0.7171833141982396 when the number of tweets is 8000. The best values of parameters as below:

```
logreg_a465ac2e7523-aggregationDepth: 2,
logreg_a465ac2e7523-elasticNetParam: 0.1,
logreg_a465ac2e7523-family: auto,
logreg_a465ac2e7523-featuresCol: normedW2V,
logreg_a465ac2e7523-fitIntercept: true,
logreg_a465ac2e7523-labelCol: label,
logreg_a465ac2e7523-maxIter: 10,
logreg_a465ac2e7523-predictionCol: prediction,
logreg_a465ac2e7523-probabilityCol: probability,
logreg_a465ac2e7523-rampredictionCol: rawPrediction,
logreg_a465ac2e7523-regParam: 0.01,
logreg_a465ac2e7523-standardization: true,
logreg_a465ac2e7523-threshold: 0.5,
logreg_a465ac2e7523-tol: 1.0E-6
```
Also, we have done the cross validation in random forest and find the best parameters which we should change them in model manually. Therefore, we have got the accuracy equals to 0.7156206148743463 when the number of records of dataset is 8000. The best values of parameters as below:

```
rfc_b31f04f019de-cacheNodeIds: false,
rfc_b31f04f019de-checkpointInterval: 10,
rfc_b31f04f019de-featureSubsetStrategy: auto,
rfc_b31f04f019de-featuresCol: normedW2V,
rfc_b31f04f019de-impurity: gini,
rfc_b31f04f019de-labelCol: label,
rfc_b31f04f019de-maxBins: 32,
rfc_b31f04f019de-maxDepth: 12,
rfc_b31f04f019de-maxMemoryInMB: 256,
rfc_b31f04f019de-minInfoGain: 0.0,
rfc_b31f04f019de-minInstancesPerNode: 8,
rfc_b31f04f019de-numTrees: 70,
rfc_b31f04f019de-predictionCol: prediction,
rfc_b31f04f019de-probabilityCol: probability,
rfc_b31f04f019de-rawPredictionCol: rawPrediction,
rfc_b31f04f019de-seed: 207336481,
rfc_b31f04f019de-subsamplingRate: 1.0
```

Conclusion

According to the results obtained, Spark has a good speed in doing data processing. The SVM algorithm takes the largest time consumed.

The results obtained by logistic regression, random forest, MLP and SVM based on the average value of accuracy: 67%, 65%, 63%, and 68%, precision value 67%, 65%, 62%, and 67%, recall 64%, 63%, 64%, and 69%, and f-score value 65%, 64%, 62%, and 68%. Finally, the results show that there are more negative tweets than the positive tweets about corona virus.

Therefore, the SVM algorithm has good results compared to other models. As future work, further experiments can be conducted to collect more tweets and implement more classification models. In addition, we can perform that experiment on cluster of many computer nodes to reduce the process time for each stage. Also, we hope the collected tweets have better sentence structure and words to get a valid accuracy of data values.
References

1. Etaiwi W, Biltawi M, Naymat G. Evaluation of classification algorithms for banking customer’s behavior under Apache Spark Data Processing System. Procedia Computer Science. 2017;113:559–64.

2. Bhosale H, Gadekar D. A Review Paper on Big Data and Hadoop’, International Journal of Scientific and Research Publications. 2014; 4(10), pp. 1–7.

3. Fu J, Sun J, Wang K. SPARK – A Big Data Processing Platform for Machine Learning. 2016 International Conference on Industrial Informatics - Computing Technology, Intelligent Technology, Industrial Information Integration (ICIICII). 2016;

4. Salloum S, Dautov R, Chen X, Peng P, Huang J. Big data analytics on Apache Spark. International Journal of Data Science and Analytics. 2016;1(3-4):145-164.

5. Jonnalagadda V, Srikanth P, Thumati K, Nallamala S. A Review Study of Apache Spark in Big Data Processing. International Journal of Computer Science Trends and Technology (IJCST). 2016; 4(3), pp. 93–98.

6. Zaharia M, Xin R, Wendell P, Das T, Armbrust M, Dave A et al. Apache Spark. Communications of the ACM. 2016;59(11):56-65. doi: 10.1145/2934664.

7. Ghag K, Shah K. Conceptual Sentiment Analysis Model. International Journal of Electrical and Computer Engineering (IJECE). 2018;8(4):2358.

8. A. V, Sonawane S. Sentiment Analysis of Twitter Data: A Survey of Techniques. International Journal of Computer Applications. 2016;139(11):5-15.

9. Srivastava, D. K. and Bhambhu, L. Data classification using support vector machine. Journal of Theoretical and Applied Information Technology. 2010;12(1), pp. 1–7.

10. Nikam S. S. A Comparative Study of Classification Techniques in Data Mining Algorithms. Orient. J. Comp. Sci. and Technol;8(1).2015;ISSN: 0974-6471, Vol. 8, No. (1): Pgs. 13-19. Available from: http://www.computerseijournal.org/?p=1592

11. Peralta D, del Rio S, Ramirez-Gallego S, Triguero I, Benitez J, Herrera F. Evolutionary Feature Selection for Big Data Classification: A MapReduce Approach. Mathematical Problems in Engineering. 2015;2015:1-11.

12. Gilheany, E. Processing time of TFIDF and Naive Bayes on Spark 2.0, Hadoop 2.6 and Hadoop 2.7: Which Tool Is More Efficient?.Masters thesis,Dublin. National College of Ireland, 2016.

13. Szandala T. Comparing Results of Sentiment Analysis Using Naive Bayes and Support Vector Machine in Distributed Apache Spark Environment. Computer Science & Information Technology (CS & IT). 2018;
14. Nodarakis, N. et al. Large scale sentiment analysis on twitter with spark. CEUR Workshop Proceedings. 2016; 1558.

15. García-Gil D, Ramírez-Gallego S, García S, Herrera F. A comparison on scalability for batch big data processing on Apache Spark and Apache Flink. Big Data Analytics. 2017;2(1).

16. Assefi M, Behravesh E, Liu G, Tafti A. Big data machine learning using apache spark MLLib. 2017 IEEE International Conference on Big Data (Big Data). 2017;.

17. Jose, K., Bhatia, N., and Krishna, S. Twitter Sentiment Analysis. National Institute of Technology Calicut. 2010.

18. Saito T, Rehmsmeier M. The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets. PLOS ONE. 2015;10(3):e0118432.

19. Farisi A, Sibaroni Y, Faraby S. Sentiment analysis on hotel reviews using Multinomial Naïve Bayes classifier. Journal of Physics: Conference Series. 2019;1192:012024.

20. Tala, F. A study of stemming effects on information retrieval in Bahasa Indonesia’, M.S. thesis, University of Amsterdam.2003.

21. Machine C. Comparing Naive Bayes and Linear Support Vector Machine - Free Summaries Examples [Internet]. Palawan Board. 2020 [cited 16 June 2020]. Available from: https://palawanboard.com/comparing-naive-bayes-and-linear-support-vector-machine/

22. Kolhe M, Tiwari S, Trivedi M, Mishra K. Advances in data and information sciences. Singapore: Springer; 2020.

23. Hafez MM, Shehab ME, Fakharany EE, Abd El Ftah Abdel Ghfar Hegazy. Effective Selection of Machine Learning Algorithms for Big Data Analytics Using Apache Spark. Advances in Intelligent Systems and Computing Proceedings of the International Conference on Advanced Intelligent Systems and Informatics 2016. 2016;:692–704.