

# CS4830 - Big Data Lab

## Final Project

### Classification on the Yelp dataset using NLP

Shashwat Patel(MM19B053), Pragalbh (MM19B012), Hrishabh (MM19B033)

#### **Contents**

1. Introduction
  2. Problem Statement
  3. EDA & Data pre-processing
  4. Feature Engineering
  5. Model Training
  6. Final Model
  7. Kafka Streaming
  8. Conclusion
- 

#### **1. Introduction**

In this project, we were given a real-world dataset upon which we performed analysis using tools and services learnt during the course on the Google Cloud Platform. The dataset that we are provided with is a subset of data from the Yelp website which contains many user-submitted reviews of various businesses.

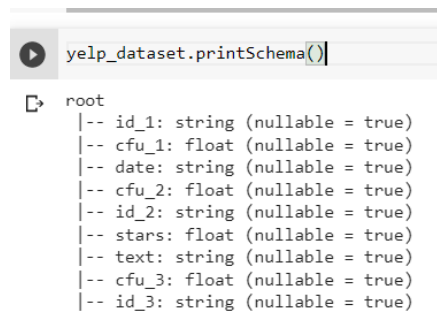
#### **2. Problem Statement**

The dataset provided was around 3.5 GB large. The aim was to classify a review into a given star-rating category by performing sentiment analysis using NLP on the review text. The approach, we considered was to pre-process the provided data and train different machine learning models on a smaller subset of training data, and take the best model from there to train on the full dataset. We use Kafka streaming model for testing the model. The steps that have been performed in this project are as follows:

1. Exploratory Data Analysis & Pre-processing
2. Feature Engineering
3. Training of different models
4. Accuracy/F1-score
5. Kafka streaming

### 3. EDA & Data pre-processing

The original yelp dataset had **6600446** rows.



```
yelp_dataset.printSchema()

root
 |-- id_1: string (nullable = true)
 |-- cfu_1: float (nullable = true)
 |-- date: string (nullable = true)
 |-- cfu_2: float (nullable = true)
 |-- id_2: string (nullable = true)
 |-- stars: float (nullable = true)
 |-- text: string (nullable = true)
 |-- cfu_3: float (nullable = true)
 |-- id_3: string (nullable = true)
```

*Figure 1: Schema of the dataset*

Three labels are given as id\_1, id\_2 and id\_3 which represent three unique IDs:

1. User ID: Unique ID provided by Yelp to a particular user.
2. Business ID: Unique ID provided by Yelp to a particular business
3. Review ID: Unique ID provided by Yelp to a particular review

It was difficult to determine which ID was which from the dataset description, so the naming was done as id\_1, id\_2 and id\_3.

cfu\_1, cfu\_2, cfu\_3 labels are numerical ratings:

1. Useful - The number of people who found the review useful
2. Funny - The number of people who found the review funny
3. Cool - The number of people who found the review cool

It was unclear with the provided dataset as to which column corresponds to which type of rating.

The other columns we had were:

1. Date: date on which the review was uploaded
2. Text - Text feedback provided by the user as part of the review
3. stars(**Output Label**): The rating of a particular business given by an user.

The **'text'** column was used as feature for training the model and predict the rating of a business given by the user.

	id_1	cfu_1	date	cfu_2	id_2	stars	text	cfu_3	id_3
0	--3Q_8lhuyMHbSocNWd6DQ	0.0	2018-04-30 16:56:24	0.0	DBJXMz1Rir91eFUd0gKjrA	4.0	My child has been attending since he was 4. H...	1.0	LH2QmIXtq8CLj65iNs76Q
1	--7PUIdqRWpRSpXeblyxTg	0.0	2019-11-02 23:30:02	0.0	0O4RqD91ZHdCPw4rUzBIYA	1.0	Lunchtime while visiting a family member at th...	0.0	IzusZJGVrtdxda-sKXwh0Q
2	--8IbOsAAxjKRoYsBFL-PA	0.0	2015-06-16 02:10:33	0.0	UYyEUH8qldJg4C4XqVX3Gg	2.0	Eh.. not that great don't waste your money on ...	6.0	gdcRiubKDmslUYFPHUp1Cg
3	--OS_l7dnABrXvRCCuWOGQ	0.0	2019-03-14 15:52:59	0.0	gOCygeYyXUHL7D0Kw_nw	5.0	We recently had our 93 Cadillac repaired at Le...	0.0	DzWfToz-VRSOGB6Cje57NoA
4	--eBbs3HpZyIym5pEw8Qdw	0.0	2019-03-29 22:31:53	0.0	NjIC7n0vIRFqx9EToHvNug	1.0	Very poor experience. We had 5 queen size room...	0.0	Um1ayg0vDsEzWRDVwqM8zA

Figure 2: Dataset before any pre-processing

For pre-processing, **null values** present in 'text' & 'stars' column were filtered out. The 'stars' column also had some garbage values such as -90, and some other **non-sensical values** as a 'rating' label.

Processed dataset summary

summary	cfu_1	cfu_2	cfu_3	stars
count	5191529	5191529	5056831	5191534
mean	0.4768387116781973	0.29676613575692246	1.082545981860933	3.8076795798698417
stddev	2.116213233201615	1.5919634896144714	6.4183602806375255	1.4521008304718614
min	-1.0	-1.0	-1.0	0.0
max	404.0	378.0	2021.0	5.0

Figure 3: Processed dataset summary

Some other insights:

1. There is no limited range as for useful, funny, cool variables because the value completely depends on other users on the website and their response to the reviews.

## Plots

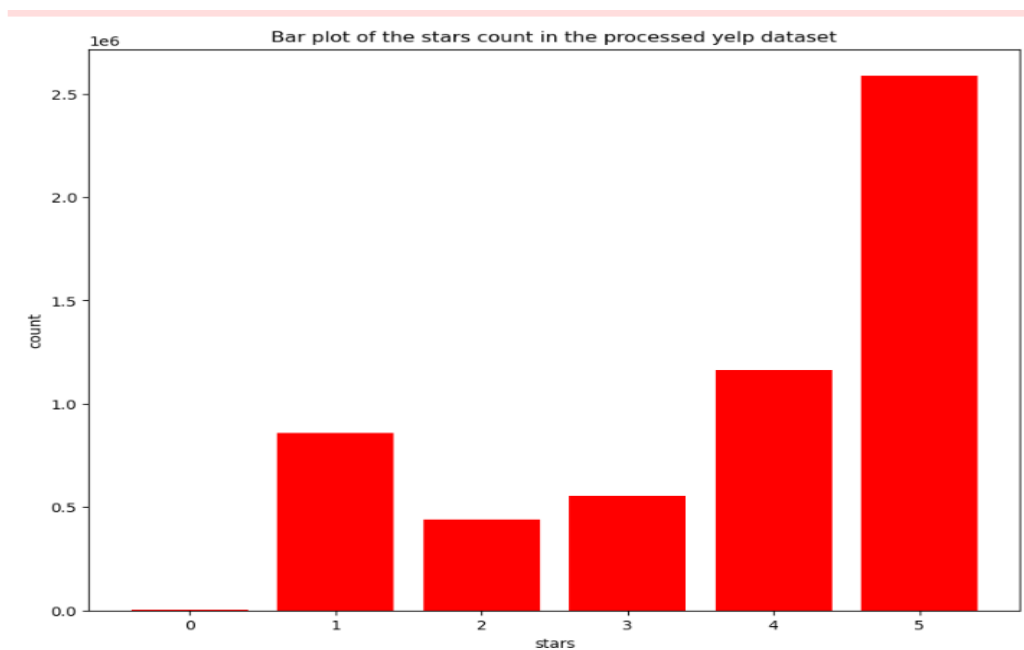


Figure 4: Distribution of 'star' (output column)

From the above plot, we can see that majority of the reviews have 5-star ratings, only some of them have a very low rating.

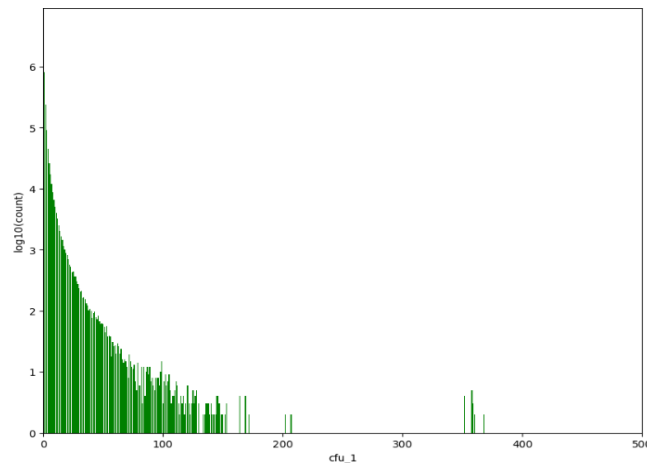


Figure 5: Distribution of cfu\_1

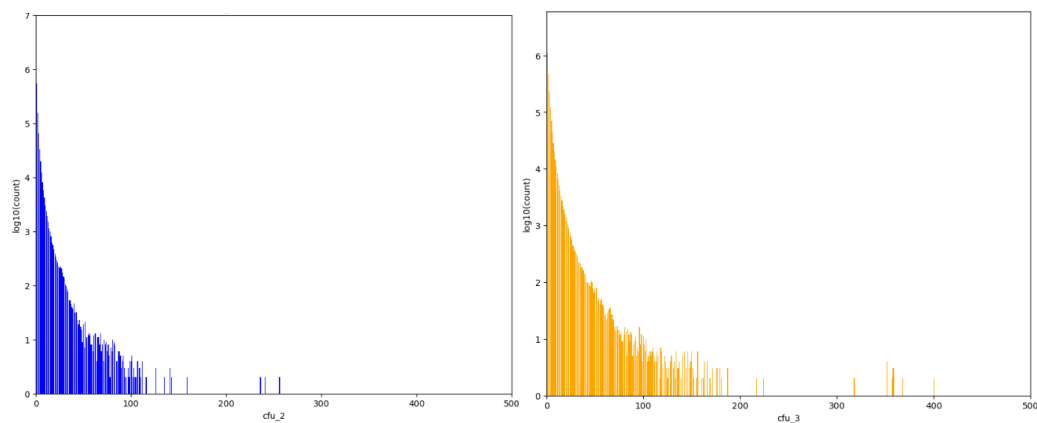


Figure 6: Distribution of cfu\_2 & cfu\_3

For, all three cfu\_1, cfu\_2 & cfu\_3 data seems to be highly skewed.

## 4. Feature Engineering

We used spark-nlp for some additional processing.

The following series of operations were performed on the 'text' column to obtain a tf-idf vector representation.

1. Document assembler - Transforms the given data into the required format for the Annotator in spark-nlp to work.
2. Tokenizer - Tokenization breaks the raw text into words called tokens. These tokens help in understanding the context or developing the model for the NLP. The tokenization helps in interpreting the meaning of the text by analysing the sequence of words.
3. Normalizer - Removes all dirty characters from text following a regex pattern and transforms words based on a provided dictionary. It brings all text to lowercase and removes punctuation.

4. Stemmer - Works by cutting off the end of the word, taking into account a list of common suffixes that can be found in an inflected word.

5. Finisher - This is used to close the annotation and transformation done for the NLP spark module's usage

6. Stopword Remover - This removes all the stopwords like 'is', 'a', 'the', 'are', etc in the text.

After the processing the text using the above processes, feature engineering was done on the text using TF-IDF.

## 5. Model Training

After feature engineering, the processed dataset was used for training models.

Here, our approach was that first we took 2 csv files out of 40 csv files (a subset of training data) given for training smaller models. The models we tested were Logistic Regression, Naïve Bayes & Decision Tree Classifier. We didn't choose models like Random Forest & SVM because the dataset is huge and the TF-IDF vectors obtained after preprocessing were large, choosing a complex model will result in very high training time.

After training on the smaller subset of data using these models, we choose that model which gave us the best F1-score for training on full training data.

### Results of Models:

#### 1. Logistic Regression

```
+-----+-----+-----+
|prediction|stars|          features|
+-----+-----+-----+
|         5.0|  5.0|(108793,[0,1,2,4,...|
|         5.0|  5.0|(108793,[0,2,4,6,...|
|         5.0|  5.0|(108793,[7,16,232...|
|         5.0|  5.0|(108793,[5,20,30,...|
|         5.0|  5.0|(108793,[1,7,8,10...|
+-----+-----+-----+
only showing top 5 rows

Evaluating on Training data(F1): 0.842851931679268
Evaluating on Training data(Accuracy): 0.8447065173911158
```

**Figure 7: Training Data**

```
+-----+-----+-----+
|prediction|stars|          features|
+-----+-----+-----+
|         3.0|  3.0|(108793,[1,2,4,6,...|
|         3.0|  4.0|(108793,[0,1,43,4...|
|         3.0|  5.0|(108793,[0,1,4,19...|
|         2.0|  4.0|(108793,[0,1,2,4,...|
|         5.0|  5.0|(108793,[0,7,8,9,...|
+-----+-----+-----+
only showing top 5 rows

F1: 0.5900026711490889
Accuracy: 0.5952188162714479
```

**Figure 8: Test Data**

## 2. Naïve Bayes

```

+-----+-----+-----+
|prediction|stars|          features|
+-----+-----+-----+
|         4.0| 5.0|(108793,[0,1,2,4,...|
|         0.0| 5.0|(108793,[0,2,4,6,...|
|         4.0| 5.0|(108793,[7,16,232...|
|         3.0| 5.0|(108793,[5,20,30,...|
|         4.0| 5.0|(108793,[1,7,8,10...|
+-----+-----+-----+
only showing top 5 rows

Evaluating on Training data: 0.06481034888055147

```

Figure 9: Training Data

```

[ ] prediction_test = nb_model.transform(test_data)
   print("F1:", nb_evaluator.evaluate(prediction_test))

F1: 0.10948483438148156

```

Figure 10: Test Data

## 3. Decision Tree

```

23/04/27 00:58:43 WARN org.apache.spark.scheduler.DAGScheduler: Broadcasting large task binary with size 4.2 MiB

+-----+-----+-----+
|prediction|stars|          features|
+-----+-----+-----+
|         5.0| 5.0|(262144,[6122,626...|
|         5.0| 5.0|(262144,[8538,420...|
|         5.0| 5.0|(262144,[75018,90...|
|         4.0| 5.0|(262144,[21823,57...|
|         5.0| 5.0|(262144,[8287,124...|
+-----+-----+-----+
only showing top 5 rows

23/04/27 00:58:46 WARN org.apache.spark.scheduler.DAGScheduler: Broadcasting large task binary with size 4.2 MiB
23/04/27 00:59:04 WARN org.apache.spark.scheduler.DAGScheduler: Broadcasting large task binary with size 4.2 MiB

Evaluating on Training data(F1-score)): 0.4493276195192957

[Stage 80:=====>                                     (1 + 1) / 2]

Evaluating on Training data(Accuracy)): 0.5138054180663257

```

Figure 11: Training Data

```

23/04/27 00:59:23 WARN org.apache.spark.scheduler.DAGScheduler: Broadcasting large task binary with size 4.2
23/04/27 00:59:29 WARN org.apache.spark.scheduler.DAGScheduler: Broadcasting large task binary with size 4.2

F1: 0.4364123943867634

[Stage 84:=====>                                     (1 + 1) / 2]

Accuracy: 0.5020821283979179

```

Figure 12: Test Data

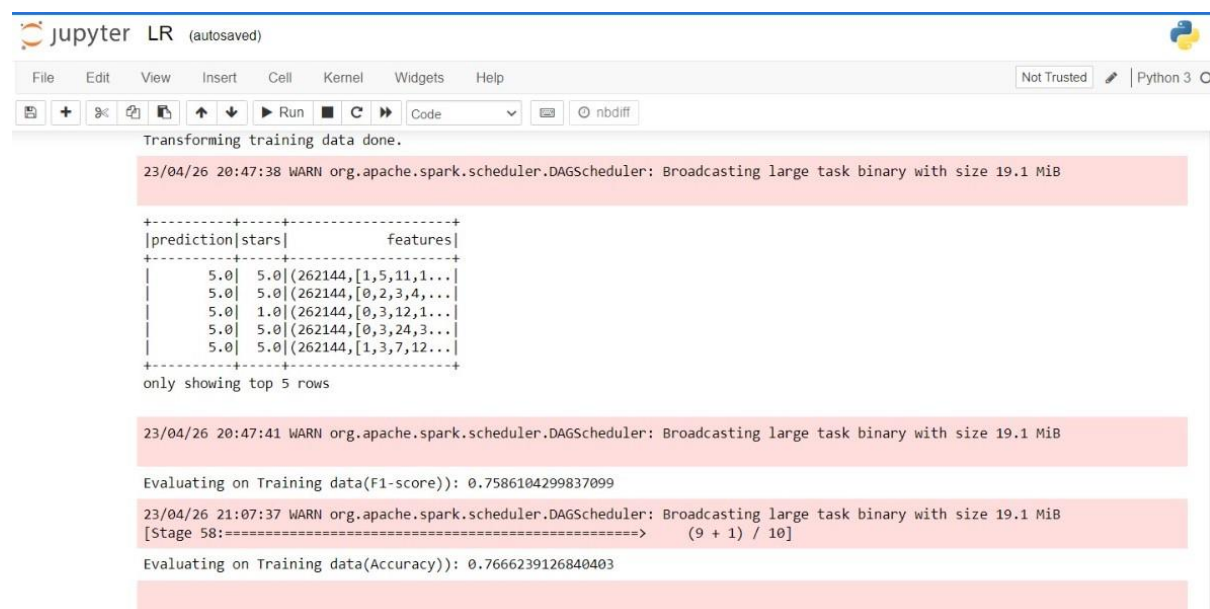
## Results Summary:

A summary of the results from the above models is shown below:

Model	F1-score(Train)	F1-score(Test)
Logistic Regression	0.843	0.59
Naïve Bayes	0.065	0.11
Decision Tree	0.45	0.436

From the above results, it is easily seen that the Logistic Regression model performs best on the train and test dataset. We use this model to train on full dataset.

## Final Model



```
jupyter LR (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Not Trusted Python 3
+ + + + +
Transforming training data done.
23/04/26 20:47:38 WARN org.apache.spark.scheduler.DAGScheduler: Broadcasting large task binary with size 19.1 MiB

+-----+-----+
|prediction|stars|      features|
+-----+-----+
|      5.0|   5.0|(262144,[1,5,11,1...|
|      5.0|   5.0|(262144,[0,2,3,4,...|
|      5.0|   1.0|(262144,[0,3,12,1...|
|      5.0|   5.0|(262144,[0,3,24,3...|
|      5.0|   5.0|(262144,[1,3,7,12...|
+-----+-----+
only showing top 5 rows

23/04/26 20:47:41 WARN org.apache.spark.scheduler.DAGScheduler: Broadcasting large task binary with size 19.1 MiB

Evaluating on Training data(F1-score)): 0.7586104299837099

23/04/26 21:07:37 WARN org.apache.spark.scheduler.DAGScheduler: Broadcasting large task binary with size 19.1 MiB
[Stage 58:=====> (9 + 1) / 10]

Evaluating on Training data(Accuracy)): 0.7666239126840403
```

Figure 13: Final Model (Training Data)

```
: prediction_test = lr_model.transform(test_data)
print("F1:", lr_evaluator_f1.evaluate(prediction_test))

23/04/26 21:27:29 WARN org.apache.spark.scheduler.DAGScheduler: Broadcasting large task binary with size 19.1 MiB
[Stage 60:=====> (9 + 1) / 10]

F1: 0.6470477040834679
```

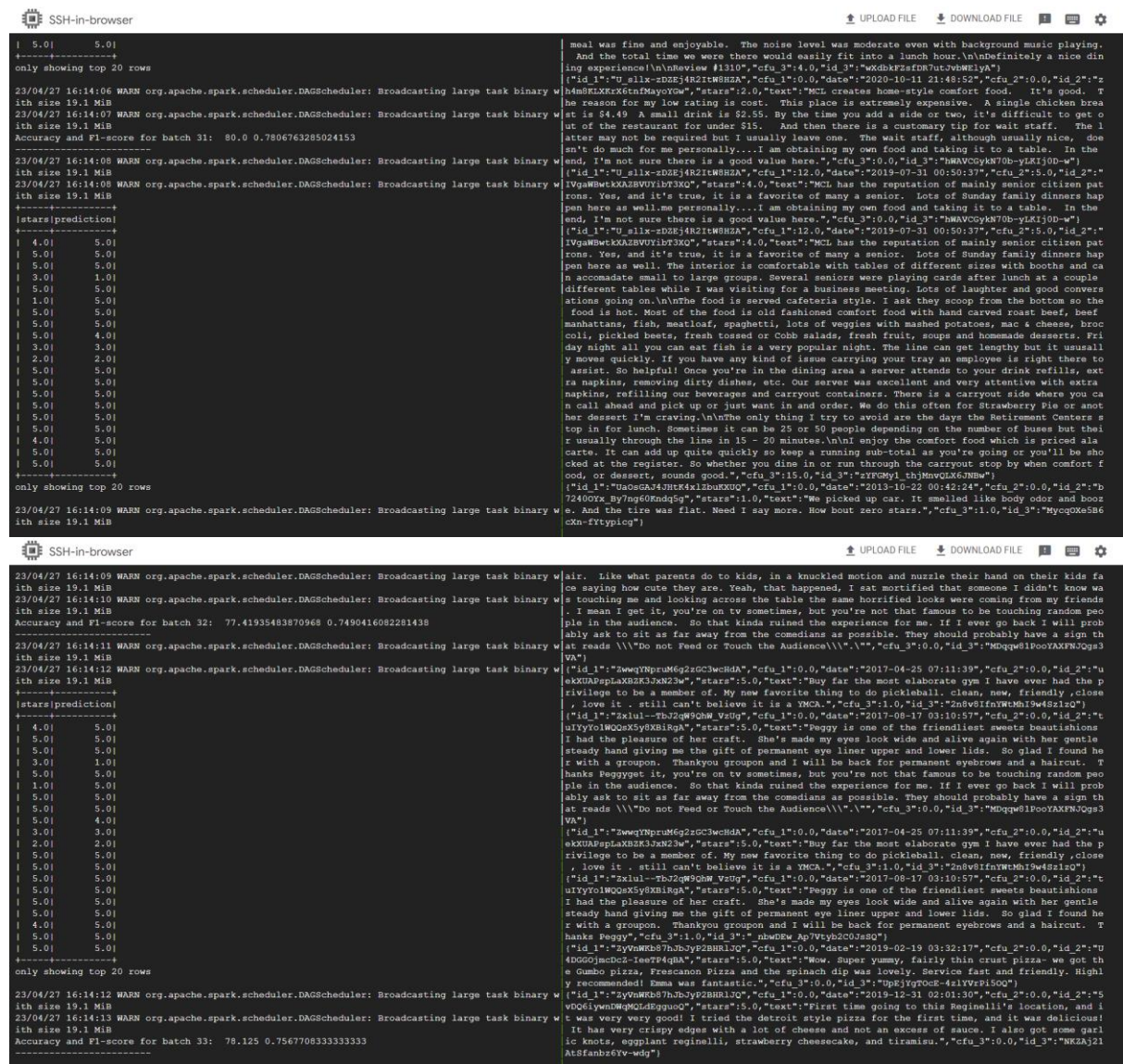
Figure 14: Final Model (Test Data)

From the above two figures, we can see that the training set F1 score is 0.758 and 0.647 for the test set.

## Kafka Streaming



The below figures show the subscriber and producer:





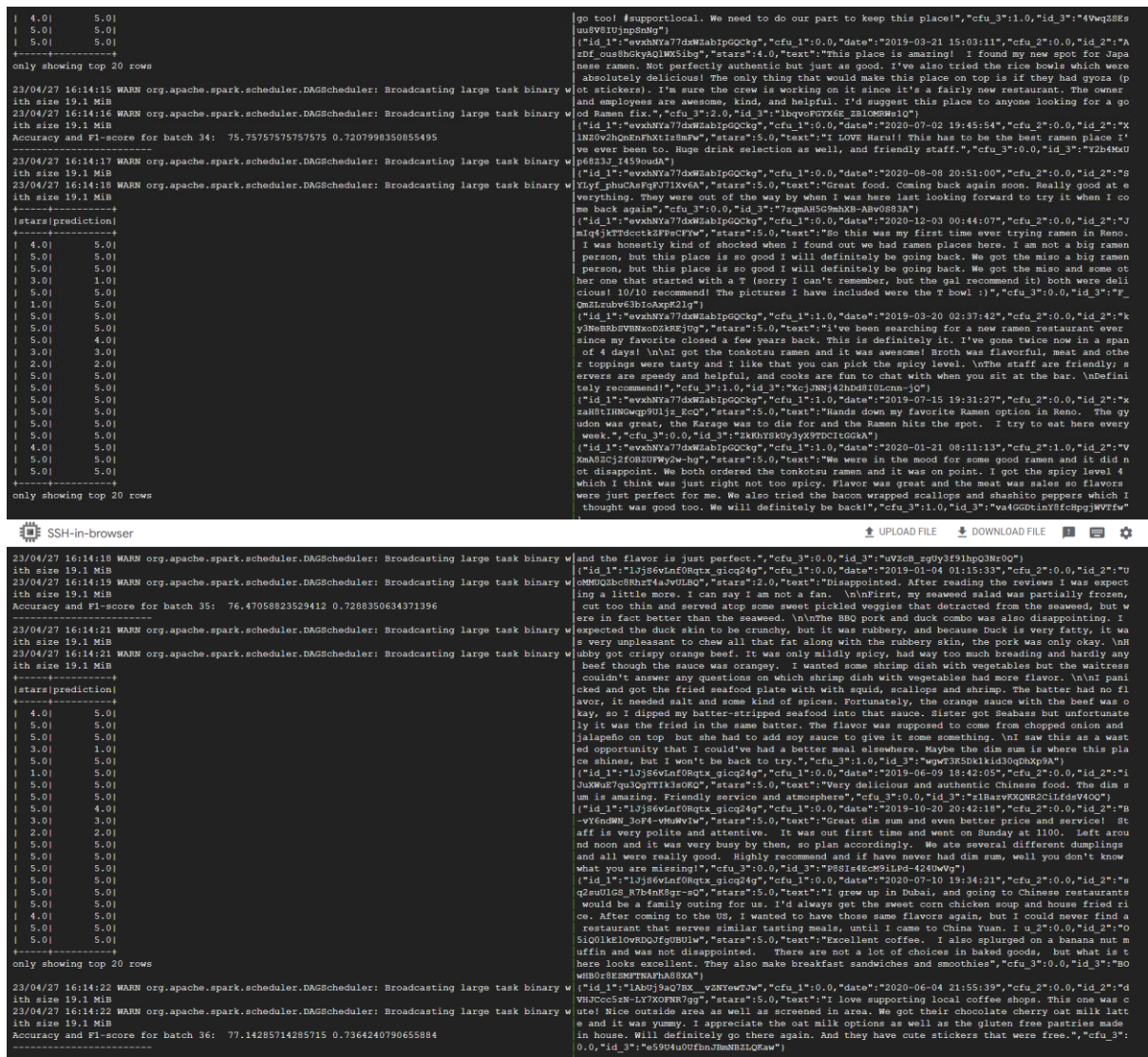


Figure 15: Kafka Results

As seen from the figure, the subscriber gives out the F1-score and accuracy for each-batch of data.

## Conclusion

1. The given dataset of Yelp is highly skewed and the dataset is highly imbalanced.
2. The logistic regression model works the best for the given dataset. Decision Tree Classifier works better than Naïve Bayes model.
3. The training set F-1 score is 0.758 and for the test it is 0.647.
4. Kafka streaming was executed properly as shown above and we were able to obtain accuracies on the test data in real time.