CS4830 BIG DATA LAB Project

TEAM BIG DATA LABOURERS

Burhanuddin Sabuwala - BE17B011

Sriram Ragunathan - CH17B072

Saarthak Marathe - ME17B162

Indian Institute of Technology Madras

JAN-MAY 2021

TABLE OF CONTENTS

INTRODUCTION

Big Data Laboratory Project: The team was given a dataset on which we had to implement and utilize the learnings, on Google Cloud Platform, during the course in the most effective manner and complete the required tasks.

PROBLEM STATEMENT

Natural Language Processing-based classification problem on Yelp review dataset (of size 5.5GB). Here, we had to classify the star ratings of the reviews based on few input variables.

The team had to pre-process the given dataset, train multiple models with each using a different method. Utilization of the Kafka streaming platform for testing the model. All these steps have been broken down as follows:

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

EXPLORATORY DATA ANALYSIS

The given **Yelp dataset** had a total of **78,63,924** data points (reviews). The different labels and their descriptions are given below:

1. Star - Output label. It's the rating of a particular business given by an user
2. Useful - Input label. The number of people who found the review useful
3. Funny - Input label. The number of people who found the review funny
4. User_id - Input label. Unique ID provided by Yelp to a particular user
5. Business_id - Input label. Unique ID provided by Yelp to a particular business
6. Date - Input label. Date on which the review was uploaded
7. Review_id - Input label. Unique ID provided by Yelp to a particular review
8. Text - Input label. Text feedback provided by the user as part of the review
9. Cool - Input label. The number of people who found the review cool

Numerical tabular analysis:

Query complete (13.9 sec elapsed, 5.2 GB processed)

Job information | Results | JSON | Execution details

Row	useful	stars	funny	user_id	business_id	date	review_id	text	cool
1	265	5	207	1948393	209393	7702369	7863924	7843119	199

Number of unique data points for each columns

Row	f0_	useful	stars	funny	cool
1	0.75	1.0	5.0	0.0	0.0
2	max	1122.0	5.0	976.0	502.0
3	stddev	3.5536407033698256	1.4904425235439291	2.1888664668518056	2.4790479420631084
4	nulls	0.0	0.0	0.0	0.0
5	0.25	0.0	3.0	0.0	0.0
6	mean	1.3231136262252803	3.7035633864213264	0.4596598339454943	0.5746910829758787
7	min	-1.0	1.0	0.0	-1.0
8	median	0.0	4.0	0.0	0.0

Statistical Analysis of the numerical data points

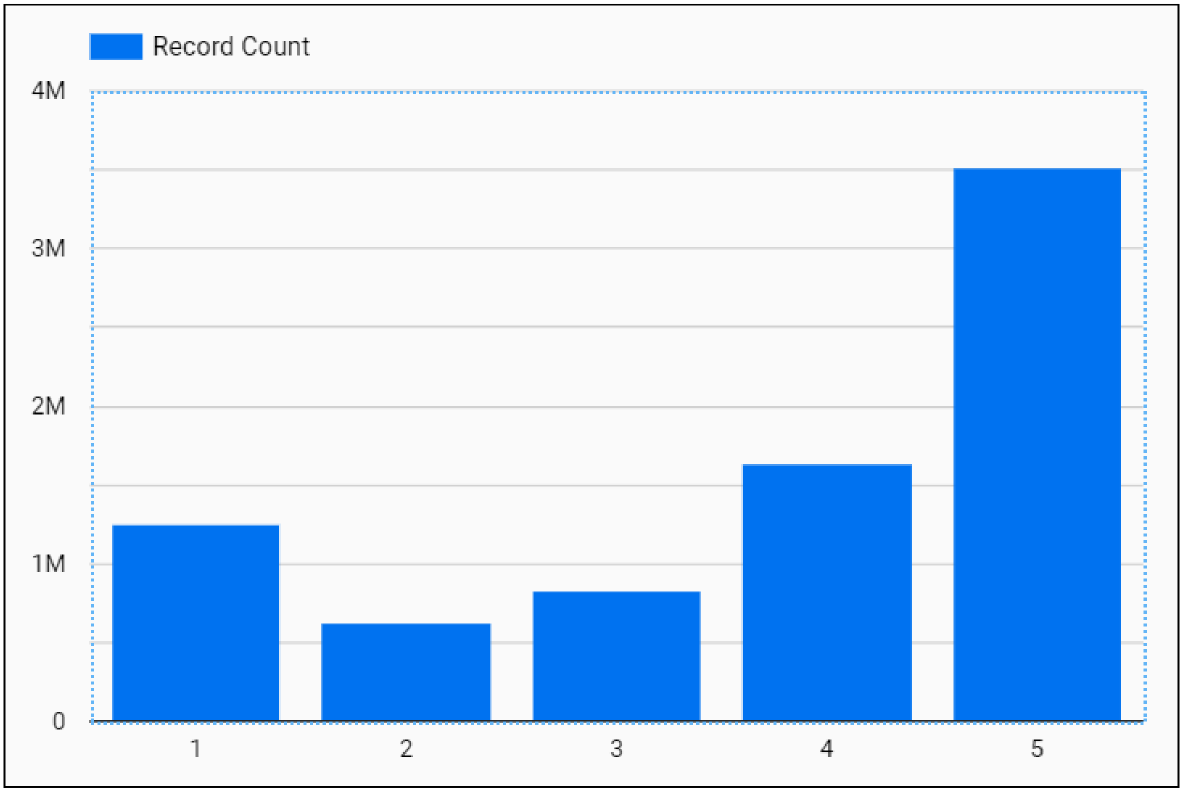
(0.75 quartile, max, min, standard deviation, null values, 0.25 quartile, mean, median)

From these two tables, we can see that:

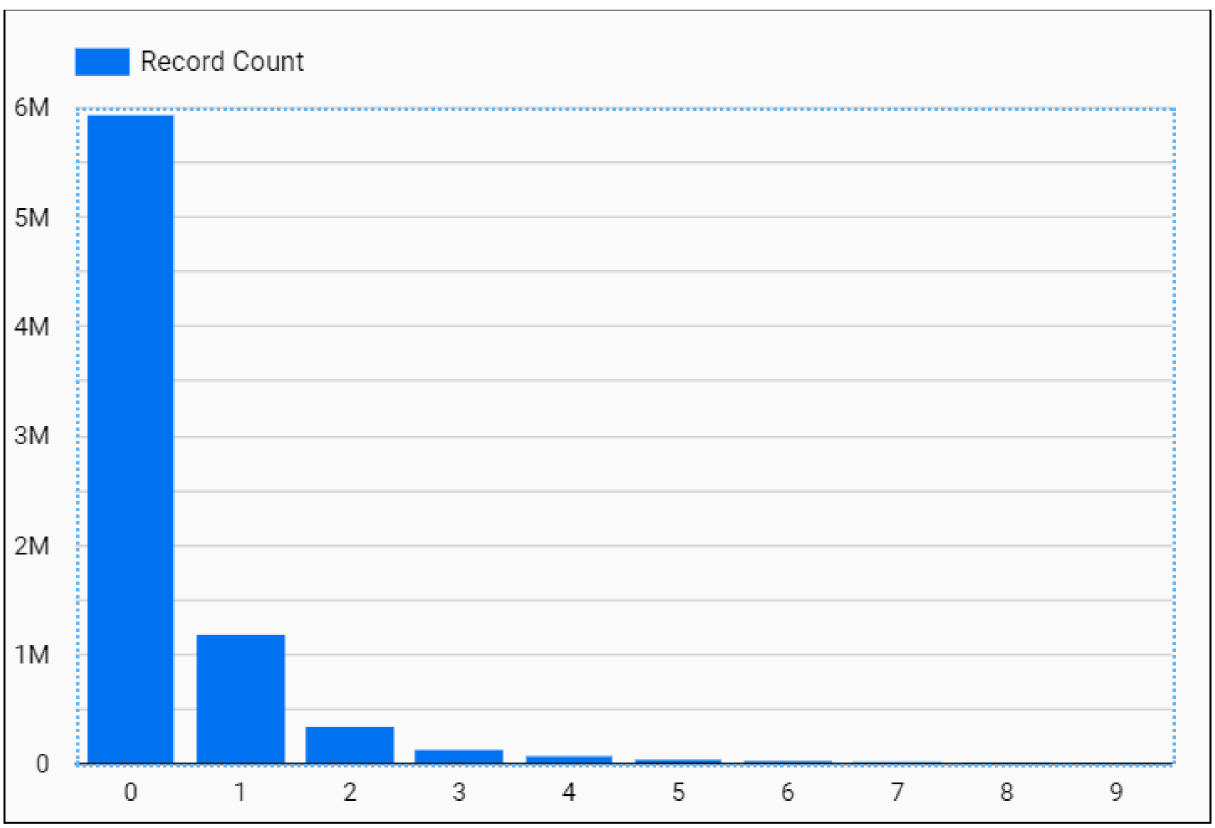
1. Same users have given reviews for multiple businesses
2. Star values range from 1.0 to 5.0
3. There is no particular limited range as such for useful, funny, cool variables because the value completely depends on other users on the website and their response to the reviews
4. No null values were found in the dataset
5. 0.75 quartile data shows that more than 75% of star data points are at max 5.0 and that of funny, cool are at the lowest 0. This suggests high skewness of the data
6. Similar skewness can be found for the 'useful' data variable as well

This skewness can be seen through the following graphs

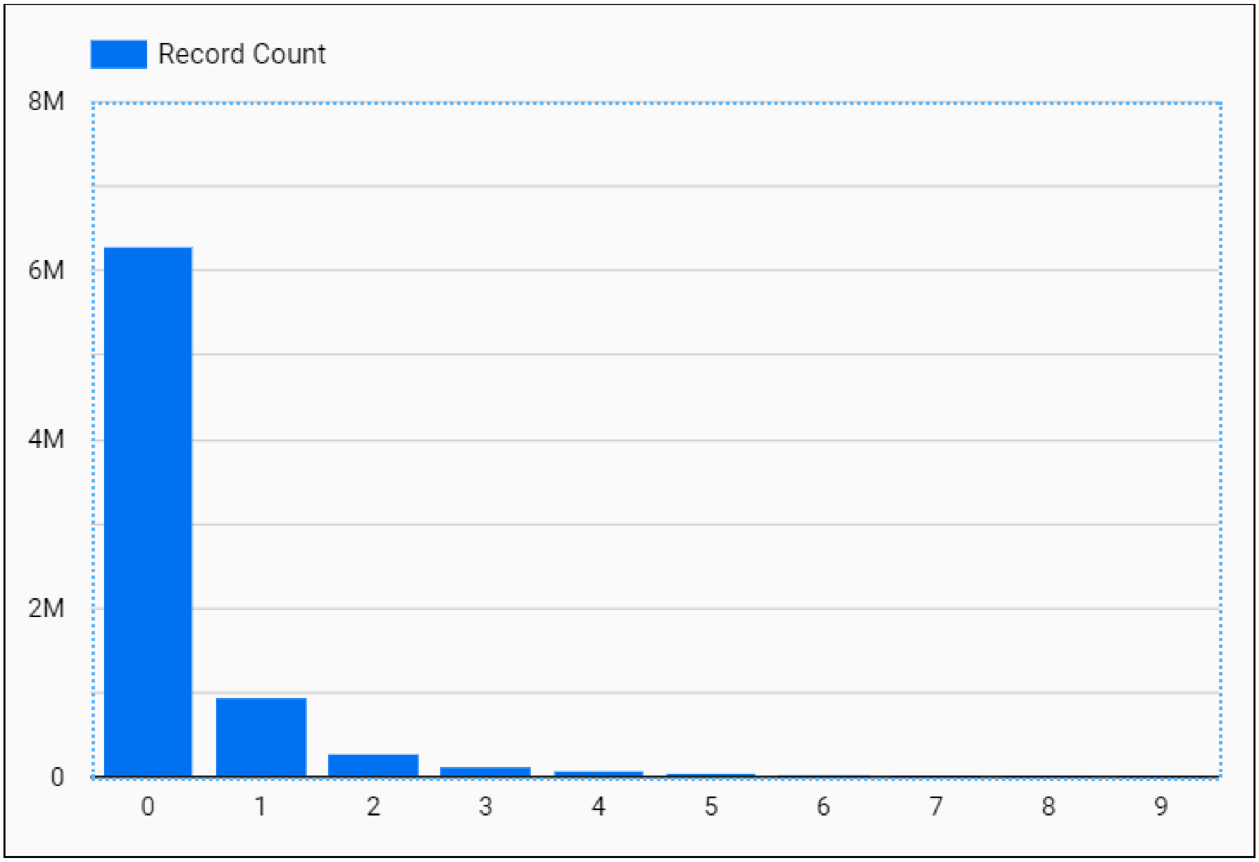
Graphical representations:



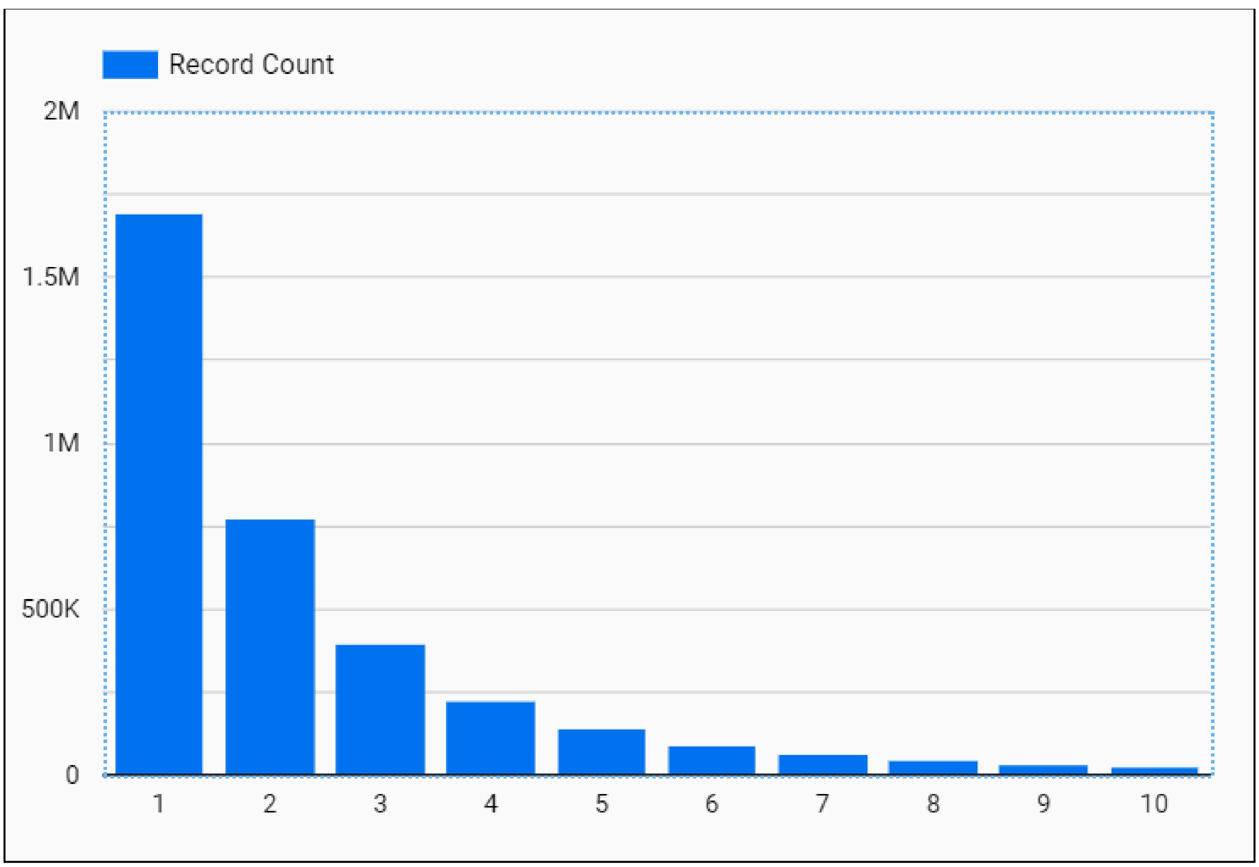
Distribution of Stars



Distribution of cool



Distribution of funny



Distribution of useful

Row	f0_	f1_	useful	stars	funny	cool
1	1	useful	1.0	-0.0882	0.6696	0.7786
2	2	stars	-0.0882	1.0	-0.0415	0.0506
3	3	funny	0.6696	-0.0415	1.0	0.7451
4	4	cool	0.7786	0.0506	0.7451	1.0

Correlation matrix

From this correlation matrix, we can see that the given numerical input variables have very low correlation with ‘stars’ variables. We had predicted from the skewness of the data provided.

Thus, from this we can sense that there will be high dependency on the ‘text’ data variable for classification modelling.

PRE-PROCESSING AND FEATURE ENGINEERING:

Pre-processing was done on the text variable mostly. We used the following methods:

1. **Document assembler -**
   Transforms the given data into the required format for the Annotator in NLP Spark to work on
2. **Tokenizer -**
   It breaks the raw text into small chunks. Tokenization breaks the raw text into words, sentences 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 analyzing the sequence of the words. For example, the text “It is raining” can be tokenized into ‘It’, ‘is’, ‘raining’
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 punctuations.
4. **Stemmer -**
   Stemming usually refers to a crude heuristic process that chops off the ends of words in the hope of achieving this goal correctly most of the time, and often includes the removal of derivational affixes. Stemmer tries to achieve this goal.
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,an,not,are,etc’ in the given text. This cleaner dataset helps in feature engineering

After this pre-processing, we move on to feature engineering. For the given text dataset, we tried TF-IDF and Word2Vec feature engineering. Word2Vec was very time consuming as compared to TF-IDF. Therefore, for all the model training, we stuck with TF-IDF feature engineering.

TRAINING MODELS

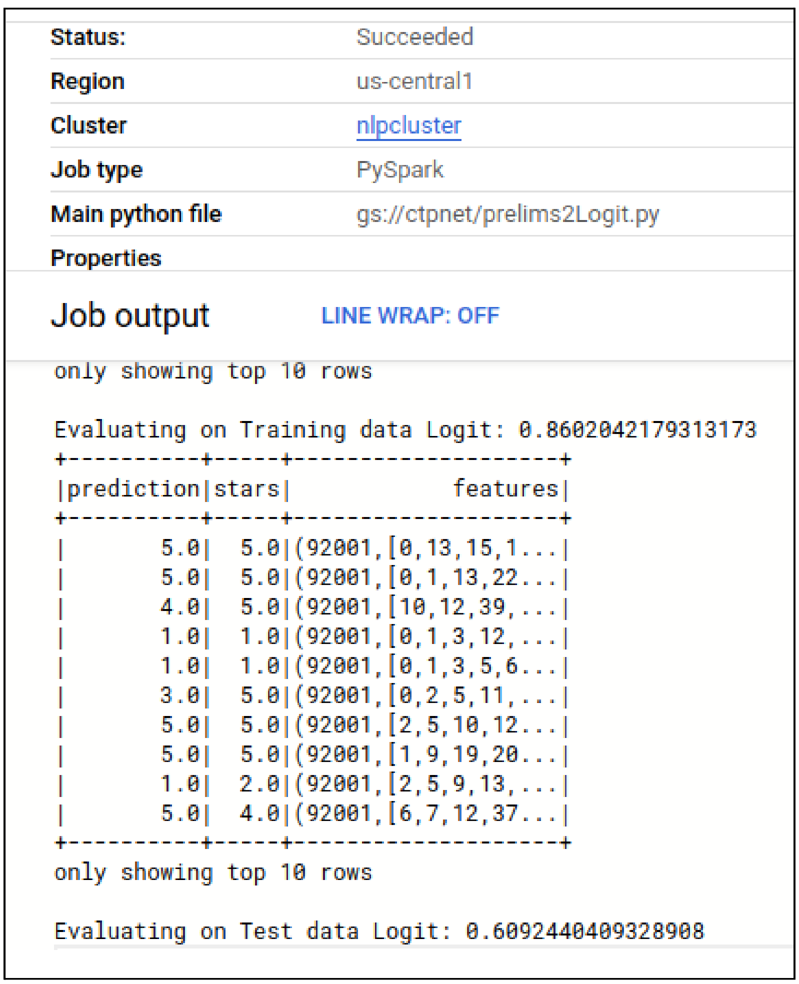
After the pre-processing and feature engineering, we use the modified dataset to train logistic regression, Naive Bayes, decision tree models. As shown before, the ‘useful’, ‘cool’, ‘funny’ variables show very low correlation with ‘stars’, therefore we did not use these as features.

Since the dataset is huge and the TF-IDF vectors obtained after preprocessing are large, choosing a complex model (such as SVM or Random Forest) would potentially result in very high training time. Therefore, we used simple models such as Naive Bayes Classifier, Logistic Regression and Decision Tree Classifier. The training is a small part of the dataset.

For training, we used a Dataproc Cluster with one master node and two worker nodes (n1-standard-8). Along with spark-nlp 3.0.3 installed.

Codes can be found along with the submission (Refer to the Appendix for the nomenclature used for the code files)

Logistic Regression:



Status:	Succeeded
Region	us-central1
Cluster	nlpcluster
Job type	PySpark
Main python file	gs://ctpnet/prelims2Logit.py
Properties	

Job output LINE WRAP: OFF

```
only showing top 10 rows

Evaluating on Training data Logit: 0.8602042179313173
+----------+-----+--------------------+
|prediction|stars|            features|
+----------+-----+--------------------+
|       5.0|  5.0|(92001,[0,13,15,1...|
|       5.0|  5.0|(92001,[0,1,13,22...|
|       4.0|  5.0|(92001,[10,12,39,...|
|       1.0|  1.0|(92001,[0,1,3,12,...|
|       1.0|  1.0|(92001,[0,1,3,5,6...|
|       3.0|  5.0|(92001,[0,2,5,11,...|
|       5.0|  5.0|(92001,[2,5,10,12...|
|       5.0|  5.0|(92001,[1,9,19,20...|
|       1.0|  2.0|(92001,[2,5,9,13,...|
|       5.0|  4.0|(92001,[6,7,12,37...|
+----------+-----+--------------------+
only showing top 10 rows

Evaluating on Test data Logit: 0.6092440409328908
```

Decision Tree:

Status:	Succeeded
Region	us-central1
Cluster	nlpcluster
Job type	PySpark
Main python file	gs://ctpnet/prelimsDTC.py
Properties	
spark.jars.packages	com.johnsnowlabs.nlp:spark-nlp_2.11:2.7.2
Arguments	ctpnet

Job output LINE WRAP: OFF

```
Evaluating on Training data Decision Tree Classifier: 0.40997157680093677
+----------+-----+--------------------+
|prediction|stars|            features|
+----------+-----+--------------------+
|       5.0|  5.0|(92001,[0,13,15,1...|
|       5.0|  5.0|(92001,[0,1,13,22...|
|       5.0|  5.0|(92001,[10,12,39,...|
|       1.0|  1.0|(92001,[0,1,3,12,...|
|       4.0|  1.0|(92001,[0,1,3,5,6...|
|       5.0|  5.0|(92001,[0,2,5,11,...|
|       5.0|  5.0|(92001,[2,5,10,12...|
|       5.0|  5.0|(92001,[1,9,19,20...|
|       1.0|  2.0|(92001,[2,5,9,13,...|
|       4.0|  4.0|(92001,[6,7,12,37...|
+----------+-----+--------------------+
only showing top 10 rows

Evaluating on Test data Decision Tree Classifier: 0.4086002188672149
```

Job output is complete

Naive Bayes:

Status:	Succeeded
Region	us-central1
Cluster	nlpcluster
Job type	PySpark
Main python file	gs://ctpnet/prelims2NB.py
Properties	
spark.jars.packages	com.johnsnowlabs.nlp:spark-nlp_2.11:2.7.2

Job output LINE WRAP: OFF

```
Evaluating on Training data NB: 0.06433109085889098
+----------+-----+--------------------+
|prediction|stars|            features|
+----------+-----+--------------------+
|       4.0|  5.0|(92001,[0,13,15,1...|
|       4.0|  5.0|(92001,[0,1,13,22...|
|       3.0|  5.0|(92001,[10,12,39,...|
|       0.0|  1.0|(92001,[0,1,3,12,...|
|       0.0|  1.0|(92001,[0,1,3,5,6...|
|       0.0|  5.0|(92001,[0,2,5,11,...|
|       4.0|  5.0|(92001,[2,5,10,12...|
|       0.0|  5.0|(92001,[1,9,19,20...|
|       0.0|  2.0|(92001,[2,5,9,13,...|
|       4.0|  4.0|(92001,[6,7,12,37...|
+----------+-----+--------------------+
only showing top 10 rows

Evaluating on Test data NB: 0.11574009966136894
```

Job output is complete

Models	F1 scores	
	Training dataset	**Testing dataset**
Logistic Regression	0.8602	0.609244
Decision Tree	0.4099	0.4086
Naive Bayes	0.064	0.1157

Models	Accuracy	
	Training dataset	**Testing dataset**
Logistic Regression	0.8607	0.60745
Decision Tree	0.36391	0.35817
Naive Bayes	0.06606	0.11436

From this, we can see that the Logistic Regression model gives the best combination of train and test dataset. We would be using this model going forward.

The Logistic Regression model is trained on the complete dataset. The code for training logistic regression on the complete dataset and the codes for Decision Tree Classifier, Logistic Regression and Naive Bayes Classifier can be found in the submission.

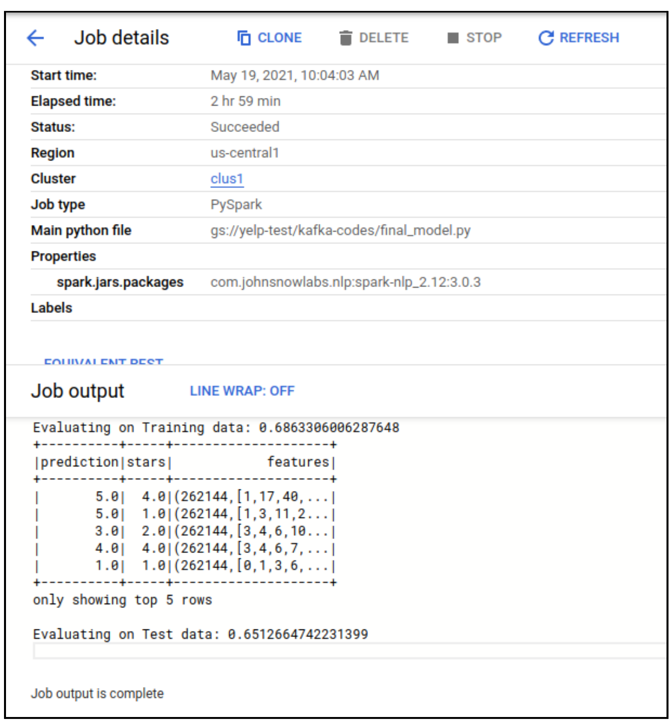


Fig: Logistic Regression model over the complete dataset

KAFKA STREAMING

For the Kafka streaming task, we have created the publisher and subscriber (code can be found along this submission). Refer to appendix as well

A kafka VM was created and the publisher code was run on the virtual machine SSH window. It is shown as follows:


```
be17b011@instance-1:~$ python3 publisher.py
{"business_id":"--cjBEbXMI2obtaRHNSFrA","cool":0,"date":"2015-04-09 23:27:13","funny":0,"review_id":"u834UGhcazCssc7l7RbJWQ","stars":4.0,"text":"Opening night, newest bar in d
owntown Pittsburgh, plenty of staff, awesome drink list...  I'll finish this later but this WILL be our 'Go-to' bar when we come into town for before &/or after a show...  Sta
y tuned...!","useful":1,"user_id":"vD2HML4pseH8-0HH3CrWzA"}
{"business_id":"--cjBEbXMI2obtaRHNSFrA","cool":0,"date":"2015-11-15 02:25:24","funny":4,"review_id":"yyryUfOlGbeWN7fU5z1C1w","stars":1.0,"text":"If you're looking for a fun pl
ace to hear shitty versions of smash mouth and/or hootie and the blowfish then look no further than howl at the moon!\n\nThey'll play all your favorites from the likes of cree
d fuel and the goo goo dolls!\n\nTrust me save your $5 at the door and take it to Charlie Murdochs or Sing Sing cause this place sucks ass!","useful":2,"user_id":"Ta8pkIeY35mt
WEbLosKdog"}
{"business_id":"--cjBEbXMI2obtaRHNSFrA","cool":0,"date":"2016-08-16 05:21:50","funny":0,"review_id":"EwtR4ua5fF05NRUAr7QwxQ","stars":2.0,"text":"My least favorite bar in Pitts
burgh.  The live music is not good and it's so loud it's painful. Never really gets too crowded which is nice and it's a really big place so you can move around. Drinks are ok
ay, nothing special. \n\nI wouldn't recommend but if you absolutely love live music, I would say check it out because you might like it!","useful":0,"user_id":"HqjI842IEQGpOdr
YGEy6EA"}
{"business_id":"--cjBEbXMI2obtaRHNSFrA","cool":0,"date":"2018-02-14 18:35:54","funny":0,"review_id":"Ay0CuPZgZcOu0QuTC-Fnzw","stars":4.0,"text":"Howl at the Moon is on our lis
t of go-to places when we want to dance and have fun drinks! We have gone at various times on weekends and have had various experiences. Each have been enjoyable and tons of f
un!\n\nDepending on the time you get there, you may be able to snag a table. Table top space is limited and once it gets busy, you are unlikely to find a place to sit. The ven
ue itself is relatively large and has two bars. Drinks can be expensive, but the happy hour specials are relatively good. Typically we order a bucket to split between a few pe
ople. The music is always interesting and varies depending on the crowd and the requests. The band is very talented and are excellent at engaging with the audience. We have ex
perienced incredibly crowded evenings where there is little room to dance, let alone move, and we have also seen it pretty quite and intimate. We celebrated a bachelorette par
ty here and had a BLAST, so that goes to show what kind of crowd you can expect at times! We don't mind the occasional packed house, but if you are looking for a place to sit
down and enjoy the music, you may want to go elsewhere.","useful":1,"user_id":"x1oGkd_OmPwX2x08Aarliw"}
{"business_id":"-03HVYxkeYWaafEpNJo1SA","cool":0,"date":"2018-03-27 15:48:05","funny":0,"review_id":"erq3I3oyr9Q-1Yz6-MrHzg","stars":1.0,"text":"This place is supposed to open
 at 11am\nBut it was not. I've waiting for 30 minutes but no appeared.  Not good for business..","useful":0,"user_id":"AadM66mhXlRDY1IWseipKw"}
{"business_id":"-050d_XIor1NpCuWkbIVaQ","cool":0,"date":"2008-10-14 22:22:31","funny":0,"review_id":"-ktkDm44jLRptoRj_sGqJw","stars":5.0,"text":"Saw this place on Diners, Driv
e-In's and Dives on the Food Network and I knew I had to go if ever in the Phoenix area.\nAs luck would have it I was out there this weekend for a wedding and this place was o
n my must visit list. Even at 10:30am on a Friday there is a line out the door for this place. We happily waited the 30 minutes and it was well worth the wait. The husband had
 the pork chop and eggs served with hash browns and thick hand cut toast. I opted for the breakfast special of the day which was a hot link scramble with fresh jalapenos and c
heddar served with a warm tortilla and hash browns. I will go on record as saying this is the best breakfast I have ever had. The eggs were cooked perfectly, the tortilla warm
 and fresh, the hash browns were crispy (not greasy) on the outside and perfect on the inside. We couldn't resist trying the pancake made from scratch as well. Fluffy and ligh
t served with warm maple syrup. YUM!  We didn't have room for all this food but we wanted to make sure we sampled all Matt's had to offer. WOW great food , excellent prices an
d warm friendly service. This is a must visit local breakfast joint in Phoenix.","useful":0,"user_id":"HVXF6OwCPI-6zpG-0M5MHg"}
{"business_id":"-050d_XIor1NpCuWkbIVaQ","cool":0,"date":"2009-08-29 17:42:13","funny":0,"review_id":"vp0adLbIlf3RBESXzomZrg","stars":5.0,"text":"One of the best breakfast plac
es in Phoenix. Arrived for a late Thursday morning breakfast (11am) and only had to wait 5-10 minutes. Server was friendly and I had the chorizo scramble special. It was great
! Very tasty and flavorful. I am one of the pickiest breakfast eaters (i.e. I don't really consider US Egg & Denny's breakfast places). I expect my eggs and all the ingredient
s to be fresh, and that was certainly the case at Matt's!! This is one of my two favorite breakfast places in the valley (for different reasons, I consider Matt's and the Oran
ge Table to be complimentary).","useful":0,"user_id":"LBweMNqEoQi9_-7HhNcWQg"}
```


Fig: Kafka VM SSH which runs the publisher code

This shows that as we run the command, the output on the window are each data point as outputs in the 'json' format.

We establish a subscriber on a separate Dataproc cluster simultaneously. As the subscriber code runs, the specifics of the job can be found below:

Start time:	May 19, 2021, 10:17:47 PM
Elapsed time:	1 min 17 sec
Status:	Running
Region	us-central1
Cluster	clus1
Job type	PySpark
Main python file	gs://yelp-test/kafka-codes/subscriber3.py
Jar files	gs://str_stream_jar_files/commons-pool2-2.6.2.jar gs://str_stream_jar_files/kafka-clients-2.6.0.jar gs://str_stream_jar_files/lz4-java-1.7.1.jar gs://str_stream_jar_files/scala-library-2.12.10.jar gs://str_stream_jar_files/slf4j-api-1.7.30.jar gs://str_stream_jar_files/snappy-java-1.1.7.3.jar gs://str_stream_jar_files/spark-sql-kafka-0-10_2.12-3.1.1.jar gs://str_stream_jar_files/spark-tags_2.12-3.1.1.jar gs://str_stream_jar_files/spark-token-provider-kafka-0-10_2.12-3.1.1.jar gs://str_stream_jar_files/unused-1.0.0.jar gs://str_stream_jar_files/zstd-jni-1.4.4-7.jar
Properties	
spark.jars.packages	com.johnsnowlabs.nlp:spark-nlp_2.12:3.0.3

Fig: DataProc job that runs the subscriber code

The publisher and subscriber run simultaneously. The subscriber takes the data that is provided by the publisher. Upon receiving the data, the subscriber runs the model and predicts the output which can be seen in the log of the subscriber job.

The outputs are seen in batches. So as we update the publisher (or when it does while going over the dataset), each review is taken in one batch and the accuracy is calculated cumulatively over the tested reviews.

Fig: Output shown by the subscriber job

To test the real-time nature of this environment, we input our own review in the required 'json' format in the SSH window of the publisher. This input is then taken by the subscriber to run the model over and predict the stars rating. The output is then shown as part of a separate batch (batch 73 below)

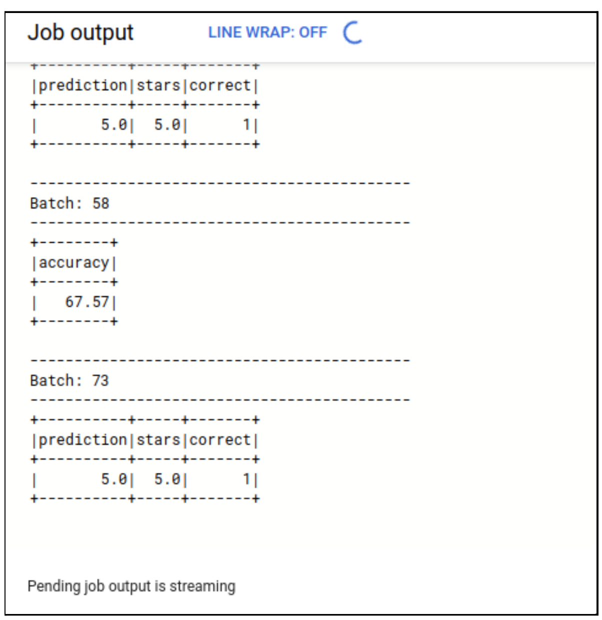


Fig: Real-time output of the subscriber when review entered in the publisher SSH window

MAJOR CHALLENGES

- Few problems were faced in utilizing the spark NLP module in the codes. This eventually led a lot of time spent in identifying and utilizing the correct jar files and correct versions of pyspark, spark-nlp and kafka to be used
- For the kafka streaming part, some issues were faced in the message parsing part due to the 'json' format.

CONCLUSION

- The given dataset of Yelp is highly skewed. The numerical variables have very less correlation with the 'stars' (output) variable.
- For Exploratory Data Analysis, we used BigQuery and SQL functions.
- Because of this, it doesn't make any difference if we include them in our model or not. The models show the same evaluation scores (f1 score, accuracy) if we use the numerical variables or not, along with the text data
- The logistic regression model works the best for the given dataset
- The skewness of the model plays a major role, because of which the accuracy is low for the given dataset
- Kafka streaming was executed properly as shown above and we were able to obtain accuracies on the test data in real time.

APPENDIX - CODES

A zipped file with all the codes has been submitted with this report.

- prelims2Logit.py - the code used to train the Logistic Regression model
- prelimesDTC.py - the code used to train the Decision Tree model
- prelims2NB.py - the code used to train the Naive Bayes model
- final_model.py - the code used to train the best model on 80% of the complete dataset
- publisher.py - the publisher code file used for the Kafka streaming part
- subscriber.py - the subscriber code file used for the Kafka streaming part
- command.txt - all the commands that were used in the cloud shell, VM instance and the jar files used for the Dataproc jobs

---

THE END