COURSE WORK TITLE:

DATASET ANALYSIS AND VISUALISATION USING BIG DATA PROGRAM

COURSE WORK TOPIC:

PREDICTING AIRLINE CUSTOMER'S RECOMMENDATION THROUGH DATA INSIGHT

COLLEGE OF COMPUTING, ENVIRONMENT AND SCIENCE

COVENTRY UNIVERSITY

NAME:

OBIANUJU VIVIAN OKAFOR

STUDENT ID:

14281642

COURSE:

BIG DATA ANALYTICS AND DATA VISUALISATION

APRIL 2024

ABSTRACT

Many people travel by air daily, and there are several airline businesses that are prepared to accommodate these consumers' needs. Customers do, however, select airlines based on the services they receive, and if they are happy with the airline and the services rendered, they also recommend the airline to friends and family.

These reviews have the potential to help these airlines provide better service.

The project's objective is to use big data analytics to anticipate, based on various airline service ratings, whether a customer's recommendation is good or negative. The Airline Review Dataset from Kaggle will be used for this investigation.

The dataset will undergo preprocessing and data cleaning, which will prepare the data for further modelling. Three different machine learning models—logistic regression, decision trees, and random forests—will be used in this study to generate predictions.

Keywords: Random forest, decision tree, logistic regression, and airline review dataset.

TABLE OF CONTENT

INTRODUCTION

With so many brands to choose from, consumers have high expectations. From the first contact to the after-sale period and beyond, every stage of the customer experience must involve ensuring their satisfaction.

Customer service is the help and direction a business offers to customers before, during, and after they make a purchase of a good or service.

Research shows that building client loyalty and promoting business growth depends on providing excellent customer service (Savage, 2024).

Customers' service preferences during interactions vary based on factors such as the nature of assistance required, and their level of personal familiarity with the product or service being evaluated for purchase. Many technologically savvy customers explore many information sources on the internet and frequently know far more about the various components of the interesting product or service than friendly representatives in the store (Lee & Lee, 2019).

Excellent customer service can improve a business’s relationships with its customers and also help improve a business as customers tend to recommend a business based on their experience with the business and the services the business rendered.

To predict if a customer will likely recommend an airline based on their services the airline review dataset will be analyzed using big data analytics. The project uses machine learning techniques including Logistic Regression, Random Forest, and Decision Tree to model and predict customer’s recommendation is positive or negative. Accuracy is used to evaluate how well these models perform at making reliable and accurate predictions. This project's software tools include Tableau, Jupyter Environment, and Pyspark. PySpark is our major analytics engine. Tableau helps visualise and explain patterns, resulting in analytically sound and meaningful solutions.

In summary, we will use PySpark and Tableau to conduct the analysis for this project.

BACKGROUND/RELATED WORK

Patel et al. (2023) wrote that customer feedback is important as it is an opportunity for businesses to improve the services given to its customers. The study used sentimental analysis to analyze the airline reviews dataset. The study tested the performance of sentimental analysis using machine learning algorithms like Naïve Bayes, Support Vector Machine and Decision Tree.

Analysing customer experience and emotion creates a sustainable technology for everyone. Sustainable technology enables us to adopt factors such as stronger business output, a smoother user experience, attracting more customers etc.

To thrive in this highly competitive environment, airlines are constantly working to improve service quality, which has become a key survival strategy. Just as product quality transformed manufacturing competitiveness, the success of the aviation sector now depends greatly on creative service quality, which has become the distinguishing factor between an airline's success and failure. Continuous improvement in service quality is recognised as a critical strategy for creating a competitive advantage and ensuring customer satisfaction. However, many airlines continue to struggle to fully understand and exceed their clients' expectations, compromising the overall quality of their services. As a result, client satisfaction remains a critical variable in providing excellent service, which require airlines to precisely identify and address these needs (Arpita et al., 2023).

DATASET DESCRIPTION

This dataset contains reviews of the top 10 rated airlines obtained from the Kaggle website https://www.kaggle.com/datasets/sujalsuthar/airlines-reviews/download?datasetVersionNumber=1. The reviews cover various aspects of the flight experience, including seat comfort, staff service, food and beverages, inflight entertainment, value for money, and overall rating. The dataset can be used for sentiment analysis, customer satisfaction analysis, visualisations, machine learning and other similar tasks.

It consists of 8,365 rows and 17 columns, these columns include:

Column names	Description
Title	Title of the review
Name	The reviewer's name
Review date	Date when the review was made
Verified	Shows if the review is verified or not
Reviews	The context of the review
Type of traveller	Shows the type of traveller the reviewer is. There are 4 types- solo leisure, family leisure, couple leisure or business.
Month Flown	The month the flight was taken
Route	The route of the flight
Class	The class of service, is it business class, economy class, premium class or first class
Airline	The airline used by the reviewer
Seat comfort	The reviewer gives a rating score between 1-5 for this particular service
Staff service	The reviewer gives a rating score between 1-5 for this particular service
Food and beverages	The reviewer gives a rating score between 1-5 for this particular service
Inflight entertainment	The reviewer gives a rating score between 1-5 for this particular service
Value for money	The reviewer gives a rating score between 1-5 to confirm if they had value for their money
Overall ratings	The reviewer gives an overall rating score between 1-10 for each airline based on their services
Recommended	This shows if the customer will recommend the airline or not

Table 1. Dataset Description

The airlines reviewed in this dataset are:

1. Singapore Airlines
2. Qatar Airways

3. All Nippon Airways
4. Emirates
5. Japan Airlines
6. Turkish Airlines
7. Air France
8. Cathay Pacific Airways
9. EVA Air
10. Korean Air

The data types represented in the dataset are integer, string and Boolean.

METHODOLOGY

The dataset includes qualitative and quantitative data that will be used for the analysis to make useful predictions. The design is experimental, and it will be done using machine learning models, the research methods adopted in this project are stated below:

4.1 Data Collection

The dataset is obtained from Kaggle on this link https://www.kaggle.com/datasets/sujalsuthar/airlines-reviews/download?datasetVersionNumber=1.

4.2 Data Preprocessing

The processing stage includes the process of cleaning the raw data by resolving issues such as class imbalance, checking for missing values, standardising your dataset, feature selection (removing columns that may not be required for the analysis), and feature encoding. All of these will be implemented before the application of the machine learning algorithm.

4.3 Application of Machine Learning Algorithms

The dataset will be split into two, a test set and a training set. The machine model will be trained using the training dataset. The following methods will be used for this project:

4.3.1 Logistics Regression:

Kanade (2022) defined logistic regression as a supervised machine learning technique that performs binary classification tasks by predicting the likelihood of an outcome, occurrence, or observation. The model produces a binary outcome with only two potential values: yes/no, 0/1, or true/false.

Logical regression examines the relationship between one or more independent variables and assigns data to discrete classes. It is often used in predictive modelling, where the model calculates the mathematical probability of whether an instance falls into a particular group or another (Kanade, 2022).

4.3.2 Random Forest:

Random Forest is a widely used machine learning method that combines the outputs of different decision trees to get a conclusion. Because of its ease of use and flexibility, it has boosted its recognition, as it can handle classification and regression problems.

The random forest method is an extension of the bagging approach by joining bagging and feature randomness to create an uncorrelated forest of decision trees. Feature randomization, which can be called feature bagging or "the random subspace method," creates a random selection of features to ensure low correlation among decision trees. This is one important difference between decision trees and random forests. Random forests select only a subset of the available feature splits, whereas decision trees consider all of them (*What Is Random Forest? | IBM*, n.d.).

4.3.3 Decision Tree:

A decision tree is a non-parametric supervised learning method that is used for classification and regression tasks. It has a hierarchical tree structure that consists of a root node, branches, internal nodes, and leaf nodes.

A decision tree begins with a root node that has no incoming branches. The root node's outgoing branches feed into internal nodes, known as decision nodes. Based on the available attributes, both node types are evaluated to generate homogeneous subsets and they are denoted by leaf nodes or terminal nodes. The leaf nodes reflect every possible outcome in the dataset (*What Is a Decision Tree? | IBM*, n.d.-b).

4.4 Software Installation

The required version of Hadoop and Spark was installed on Ubuntu. The proof of the installations can be seen in Fig 1, Fig 2 and Fig 3. Tableau software was also installed for visualization (Fig 4).

EXPERIMENTAL SECTION

To begin this process PySpark and findspark and imported were installed on the Jupyter environment. The installation process can be seen in Fig 15. With the software being installed the next stage was to import all the necessary libraries including the machine learning libraries from PySpark (Fig 16). After this, the next step was to import that airline review dataset. The dataset was loaded into Spark DataFrame and then started the preprocessing stage (Fig 17).

5.1 Preprocessing of the dataset

In this stage, steps to prepare the data for the models are being carried out. This process involves cleaning the data to be able to fit into the language the machine understands and also removing some information like columns that may not be needed for the analysis.

The preprocessing stage begins with checking for missing or null values and dropping these columns from the DataFrame as seen in Fig 18. In the case of the airline review dataset, there were no missing values hence this step wasn't required. Fig 18 also shows the number of rows and columns in the dataset. The next step was to drop columns that were not required for the analysis, the experiment uses the ratings from these customers to predict if they will recommend that airline or not. Fig 19 shows the columns that were not needed for the experiment and the datatype of the rest of the columns. Notice that Pyspark is reading all of these columns as a string. To change it to its correct dtype we imported the datatype library from spark and used the cast() function to make these changes (Fig 20 and Fig 21). After this, we converted the target variable column from string yes and no to integers 1 and 0 and the Boolean column-true and false to integers 1 and 0. After converting the datatypes and changing the target and Boolean column we noticed there were missing values and we dropped them, this reduced the number of rows (Fig 22, Fig 23).

The next step is to check for class imbalance, fig 22 shows that our target variable may not have been perfectly balanced however the degree of distribution between the two classes can be negligible.

Class imbalance is when some classes have more instances than others. However, in most cases, if the degree of distribution between the classes is minute it is usually neglected and no class balance technique is applied.

The next step was encoding categorical columns by converting them to binary to enable the machine to work with them as the machine understands binary or numeric columns. First, we pick out the categorical columns that require modification. Every unique string value of these columns is assigned to a unique numerical index to enable them to be useable for machine learning. Then encoding takes place to convert these numeric indices to binary vectors, next a pipeline is constructed to effectively apply the indexing and encoding processes. The pipeline transforms the columns of the original dataset to encode, this process can be seen in Fig 24. After the encoding process, we drop the original columns and use the encoded columns.

The next step is to separate the target variable and then combine all the numerical columns into one output column as in Fig 25, after this is the data scaling or normalization, this is done to have all the values in the data in one scale which is important for machine learning models as it can help enhance model accuracy (Fig 26).

The next step is to split our data into a testing set and a training set. 70% of the data is assigned to the training set and 30% of the data is assigned to the testing set. Setting seed is done to ensure that the same data is been used anytime the data is run again, it helps with consistency every time the data is run (Fig 27).

Once the preprocessing stage is done the data is now ready for training.

5.2 Model Training and Performance evaluation using Accuracy score.

In this stage, three models are implemented and used for this experiment. After training, the models are used to predict the outcomes of the testing data. The accuracy score will be used to check the model that predicts best.

Fig 28 shows the implementation of the logistic regression method.

Fig 29 shows the implementation of the random forest method.

Fig 30 shows the implementation of the decision method.

5.3 Tableau Visualization

The dataset was imported into the Tableau environment then I dropped a row with a value zero as the ratings for each service were between 1 and 5 and the overall rating was between 1 and 10, next I removed some columns that won't be used for the visualization process. This column includes title, name, review date, verified, and reviews. In the visualization process, we want to be able to check the airlines that had more positive reviews, the type of travel customers would prefer to make, the airline with the best services and the airline with the highest recommendation.

Fig 5 shows the airline with the highest overall rating is Qatar Airways, this shows that Qatar Airways had a lot of ratings and the airline with the least highest overall rating is Korean Air. The airline with the overall lowest rating is Turkish Airlines.

From Fig 6 we can conclude that the customers who went for solo leisure gave the highest overall ratings and the customers who went for Business gave the lowest ratings.

Fig 7 shows that customers in economy class gave the highest overall ratings however majority gave the lowest ratings. The majority were not okay with the services they received from these airlines.

We will also see ratings based on each service they received from these airlines:

Fig 8 shows that Qatar Airways had comfortable seats as they had the highest positive review while Korean Air had the lowest positive review however Turkish Airlines had the highest negative review.

In Fig 9 we can see that Qatar Airways has the best staff service while Turkish Airways doesn't have excellent staff service.

Fig 10 shows that Customers don't really like the food and beverages served by Turkish Airlines however there is an equal number that likes it. While a lot of customers like the food and beverages served by Qatar Airways. From the plot, we can also see that Japan had the worst food and beverage ratings.

Fig 11 shows that customers who used Qatar Airways were very entertained while customers who used Korean Air were very entertained however a lot of customers were not pleased with the services of Turkish Airlines.

Fig 12 shows that a lot of customers who used Qatar Airways agreed that they had value for their money.

From the charts, we can see that Qatar Airways has the highest chance of being positively recommended by customers because they have shown excellent services based on the ratings from the airline review dataset, this is proven in Fig 13. Fig 14 shows that over the years Qatar Airways has always provided good services to their customers.

RESULT DISCUSSION

The models were trained using 14 features out of 17 features. These 14 features were of utmost importance in the prediction of airline customers' recommendations. We use the accuracy scores to determine the performance of the machine learning algorithms applied. Below is the accuracy scores and other metrics evaluation for each of the models:

Models	Accuracy Score	Precision	Recall	F1 Score	AUC-ROC
Logistic Regression	71.44%	71.42%	71.44%	71.43	80.1%
Random Forest	91.83%	92.1%	91.8%	91.9%	97.72%
Decision Tree	91.6%	91.9%	91.6%	91.7%	94.5%

Table 2. Accuracy score for each machine learning model applied.

CONCLUSION AND FUTURE WORK

From the visualization of the dataset, we were able to know the airline that will likely have the highest positive recommendations based on the reviews by airline customers, we can also see that many airlines need to improve their services. I will recommend they are more in touch with their customers, take their reviews very seriously and possibly ask these customers what they would do differently, customers might have ideas that can help the business. Qatar Airways may have had the highest positive ratings however they have some customers that disagree with their services, this airline can go through their reviews to see where they may have lacked and work on those areas.

Based on the result of our experiment Random Forest classifier and Decision Tree classifier passed the threshold of 80% however Random Forest is the most suitable models for predicting the outcome of airline customer recommendation choice however other models can also be applied to this experiment.

For future works, I recommend applying other machine learning models like deep learning or neural networks and using advanced techniques like hyperparameter tuning for optimization which can enhance prediction accuracy.

SOCIAL IMPACT OF THIS PROJECT

Being able to predict if a customer will come back to a business or recommend a business can help a business improve its services and using data insights and machine learning approaches can help businesses act early and faster as technology has made it easier to get insights faster.

These insights can provide an in-depth look at customers' attitudes and feelings towards various areas of the airline's services, allowing these airlines to better understand what drives their client satisfaction or disappointment and inform better decisions.

REFERENCE

Arpita, H. D., Ryan, A. A., Hossen, M. S., & Rahman, M. S. (2023). Airline Sentiments Unplugged: Leveraging Deep Learning for Customer Insights. *2023 5th International Conference on Sustainable Technologies for Industry 5.0 (STI)*. https://doi.org/10.1109/sti59863.2023.10464925

Kanade, V. (2022, April 18). *Logistic Regression: equation, assumptions, types, and best practices - Spiceworks*. Spiceworks. https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-logistic-regression/#:~:text=Logistic%20regression%20is%20a%20supervised%20machine%20learning%20algorithm%20that%20accomplishes,1%2C%20or%20true%2Ffalse.

Lee, S. M., & Lee, D. (2019). "Untact": a new customer service strategy in the digital age. *Service Business*, *14*(1), 1–22. https://doi.org/10.1007/s11628-019-00408-2

Patel, A., Oza, P., & Agrawal, S. (2023). Sentiment Analysis of Customer Feedback and Reviews for Airline Services using Language Representation Model. *Procedia Computer Science*, *218*, 2459–2467. https://doi.org/10.1016/j.procs.2023.01.221

Savage, M. (2024, April 2). What is customer service in 2024: Definition, types, benefits, stats. *The Future of Commerce*. https://www.the-future-of-commerce.com/2021/08/02/what-is-customer-service-definition-examples/

What is Random Forest? | IBM. (n.d.). https://www.ibm.com/topics/random-forest

What is a Decision Tree? | IBM. (n.d.-b). https://www.ibm.com/topics/decision-trees

APPENDIX I

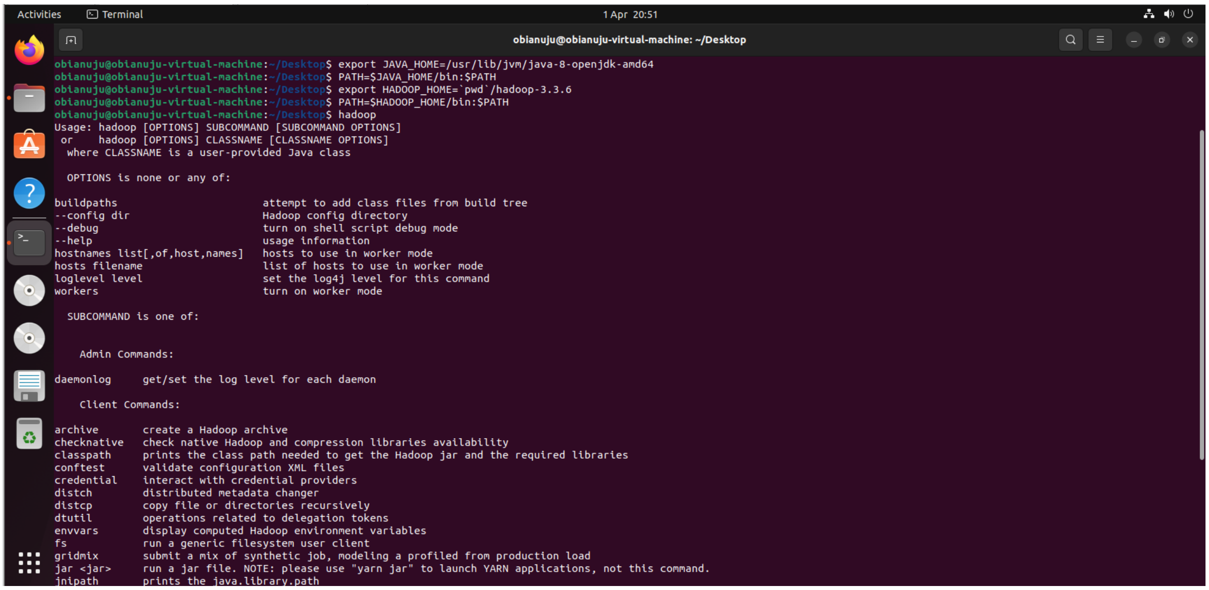


Fig 1. Proof of Hadoop being installed.

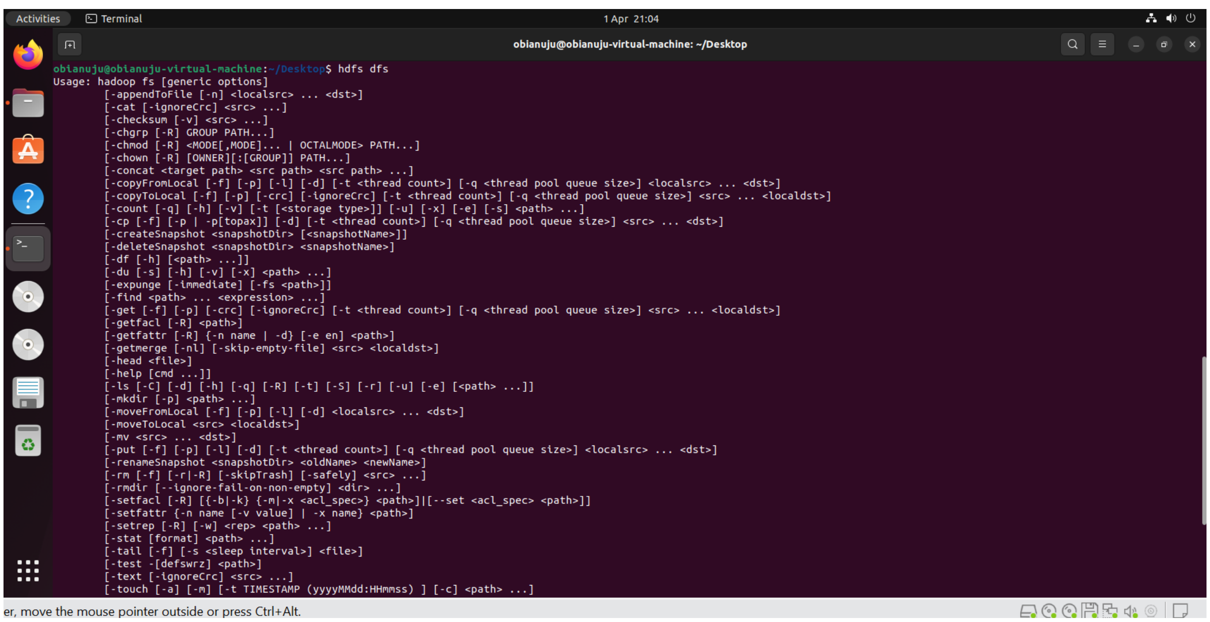


Fig 2. Calling Hadoop File System (HDFS).

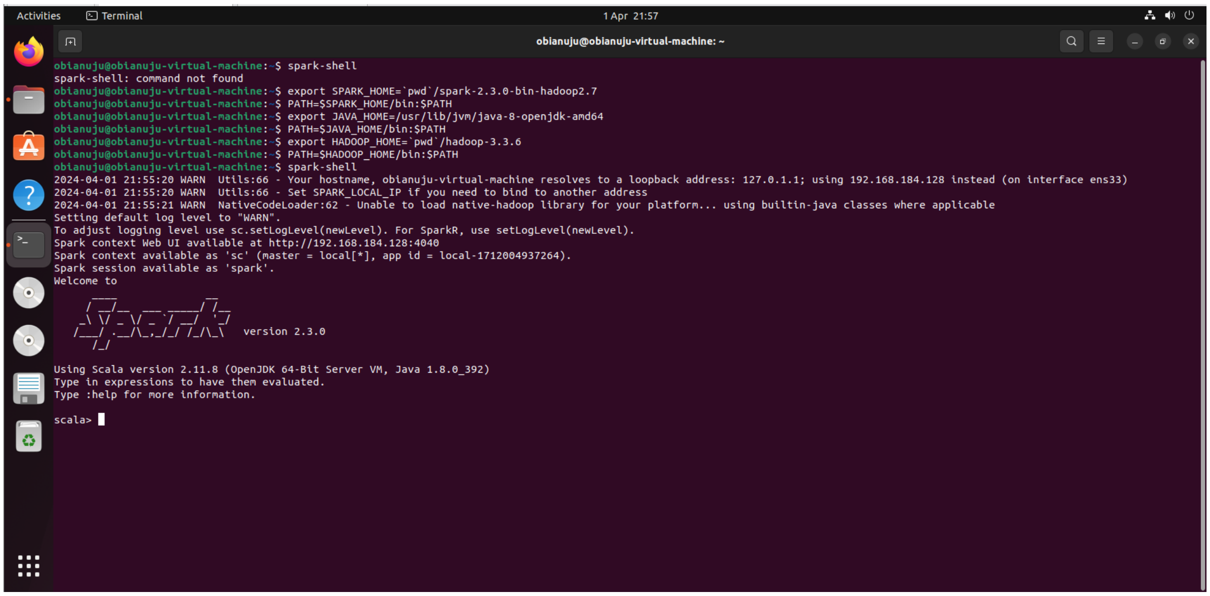


Fig 3. Spark Installation.

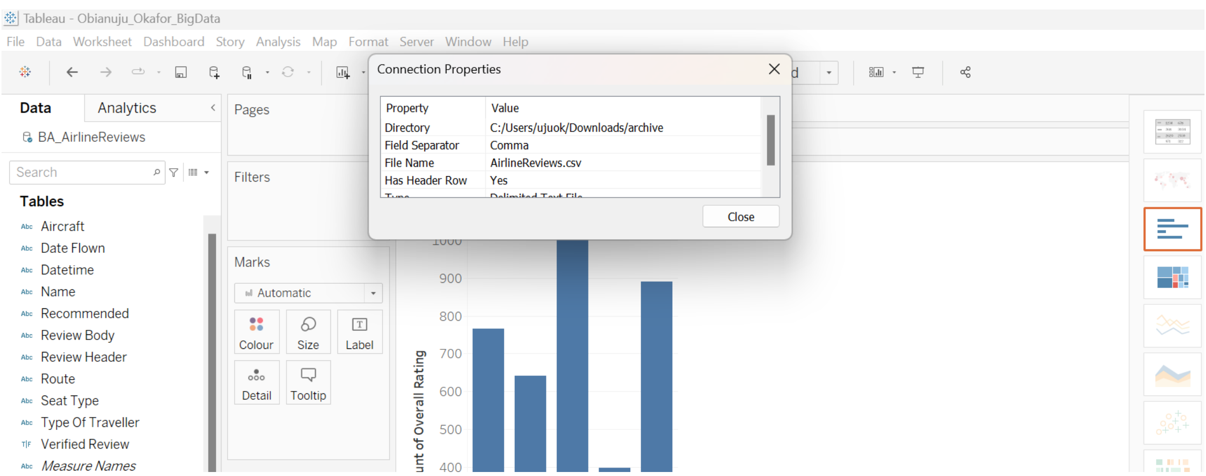


Fig 5. Tableau Proof

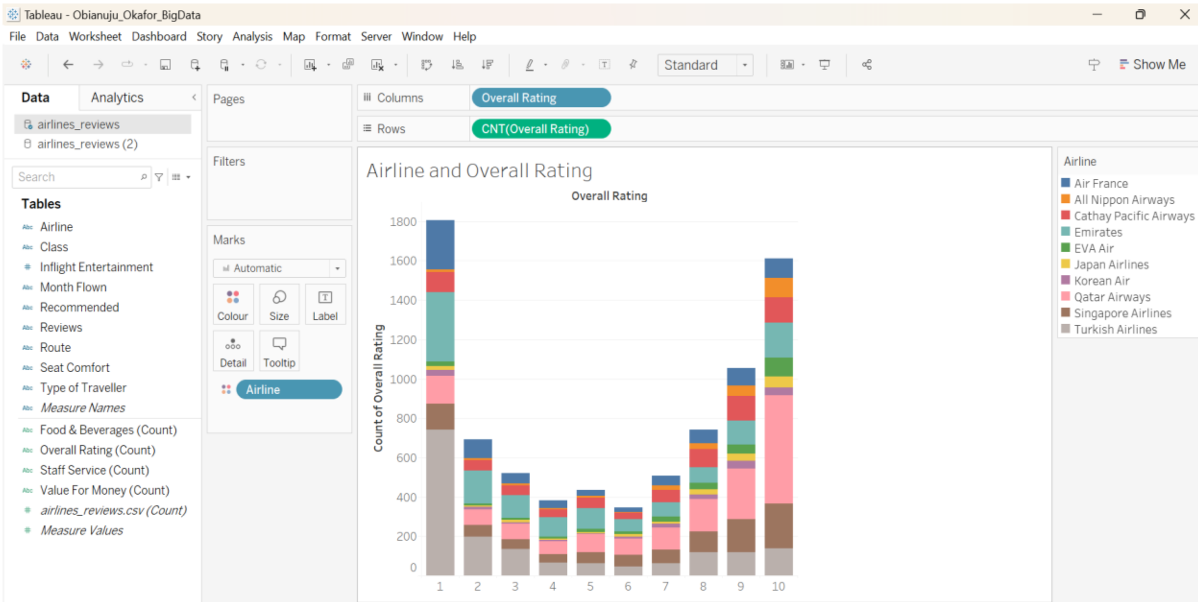


Fig 5. Overall ratings of each airline

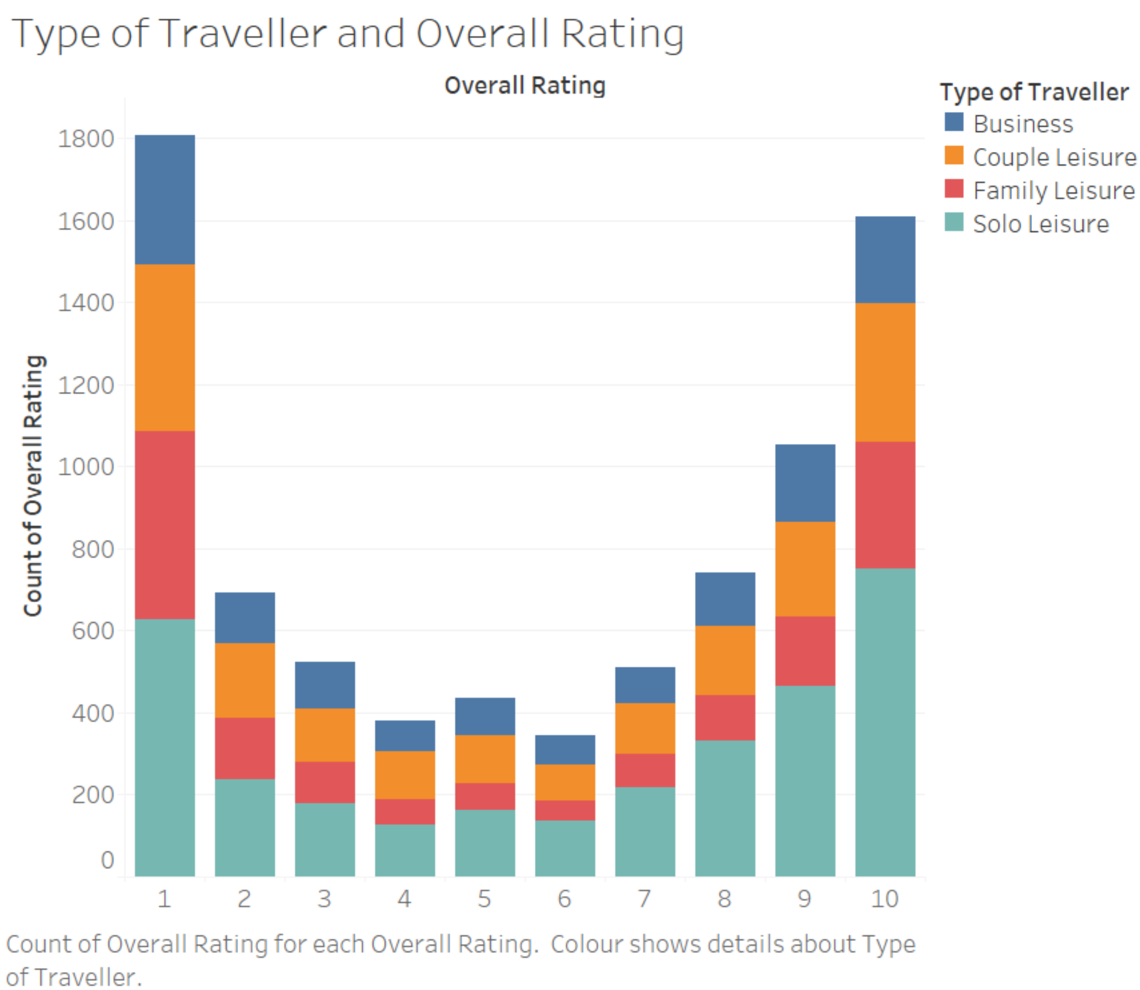


Fig 6. Overall ratings of type of traveller.

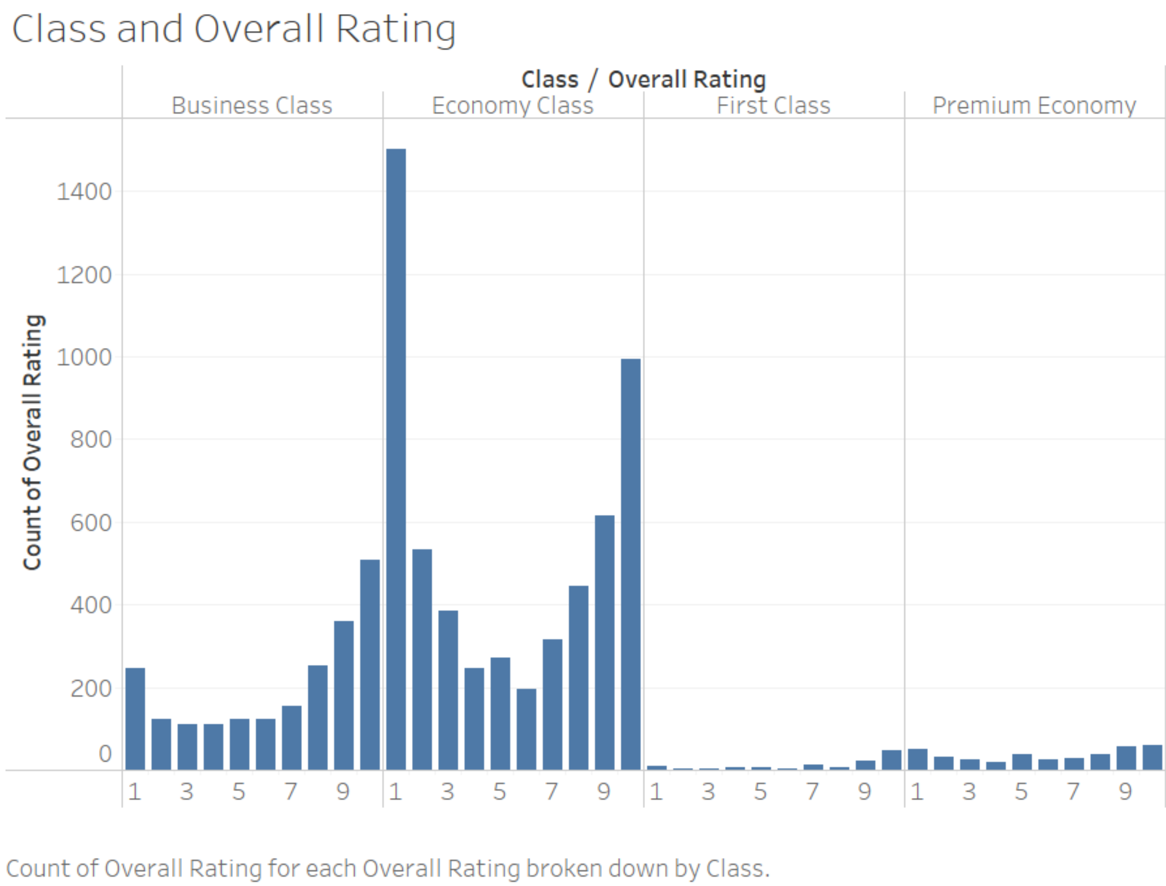


Fig 7 Overall ratings of Class

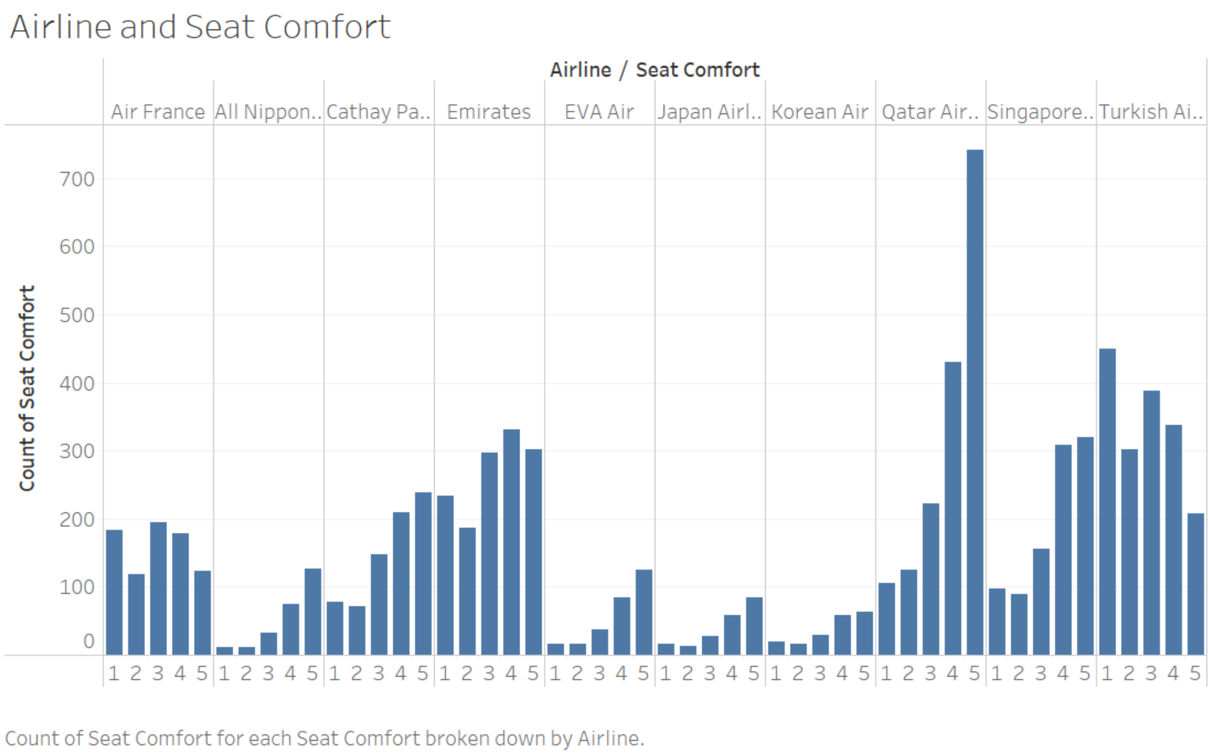


Fig 8. Ratings of airlines and their seat comfort service

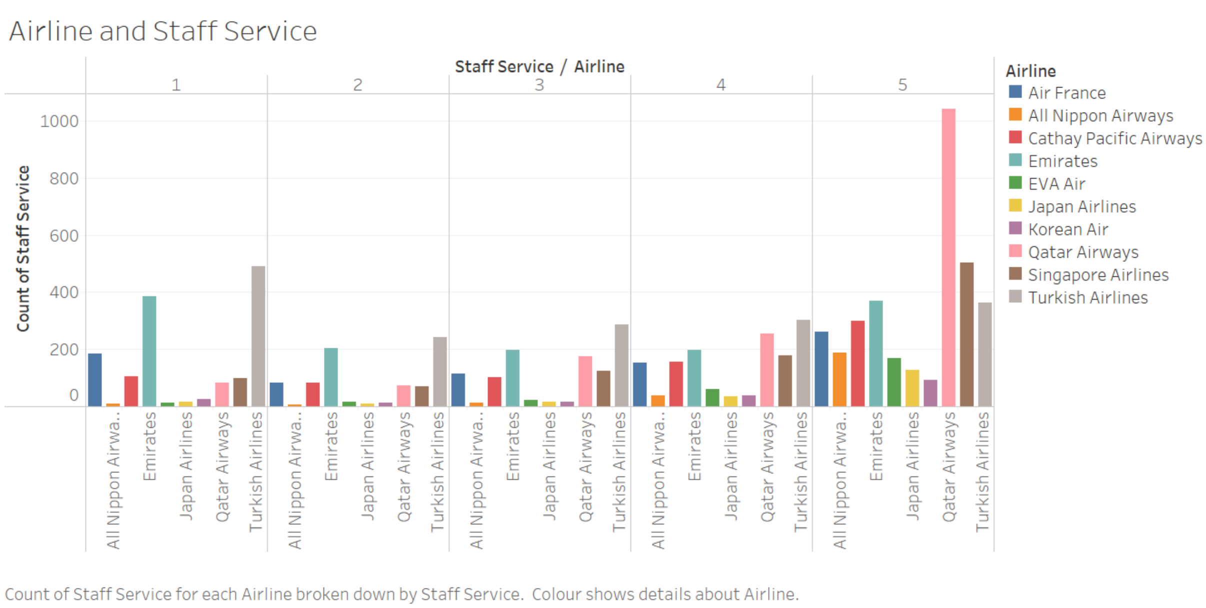


Fig 9. Ratings of airlines and their staff service

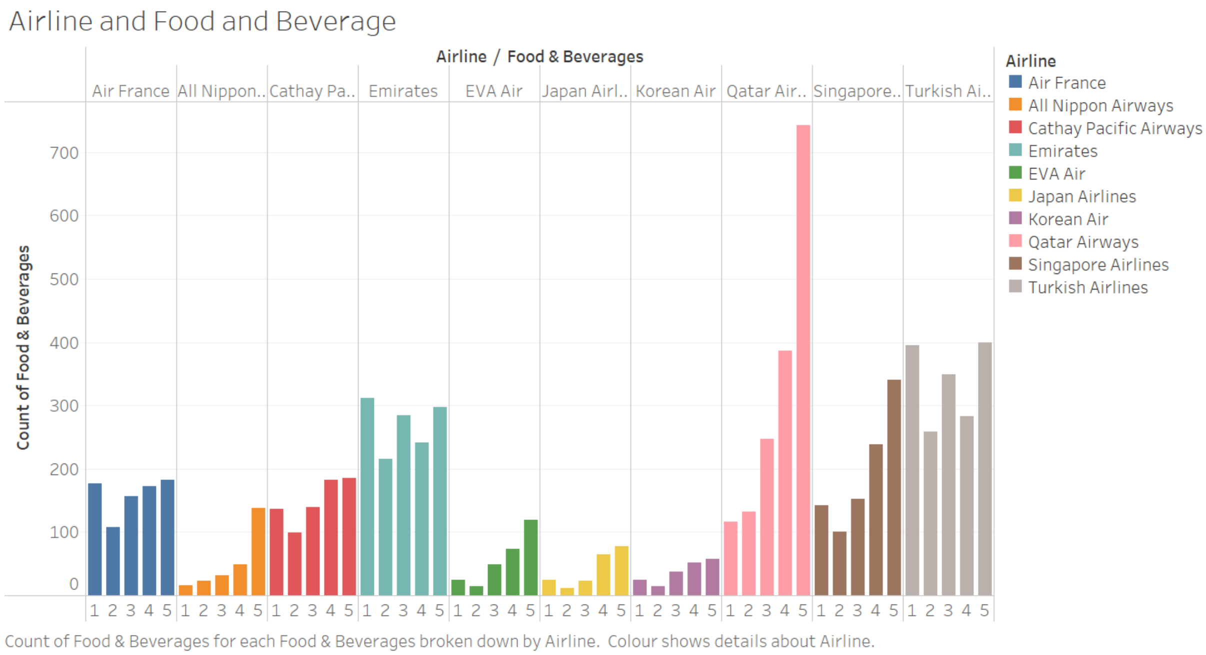


Fig 10. Ratings of airlines food and beverages

Airline and Inflight Entertainment

Airline

Inflight..	Air Fr..	All Ni..	Cath..	Emir..	EVA ..	Japa..	Kore..	Qata..	Singa..	Turki..
1	146	7	52	144	20	17	22	48	63	344
2	110	18	47	128	16	17	23	78	66	238
3	187	57	143	222	55	55	45	239	142	394
4	199	80	246	320	90	60	60	526	348	355
5	155	96	256	536	100	52	37	733	353	354

Count of Inflig..
7 733

Count of Inflight Entertainment broken down by Airline vs. Inflight Entertainment. Colour shows count of Inflight Entertainment. The marks are labelled by count of Inflight Entertainment.

Fig 11. Airlines' inflight entertainment ratings

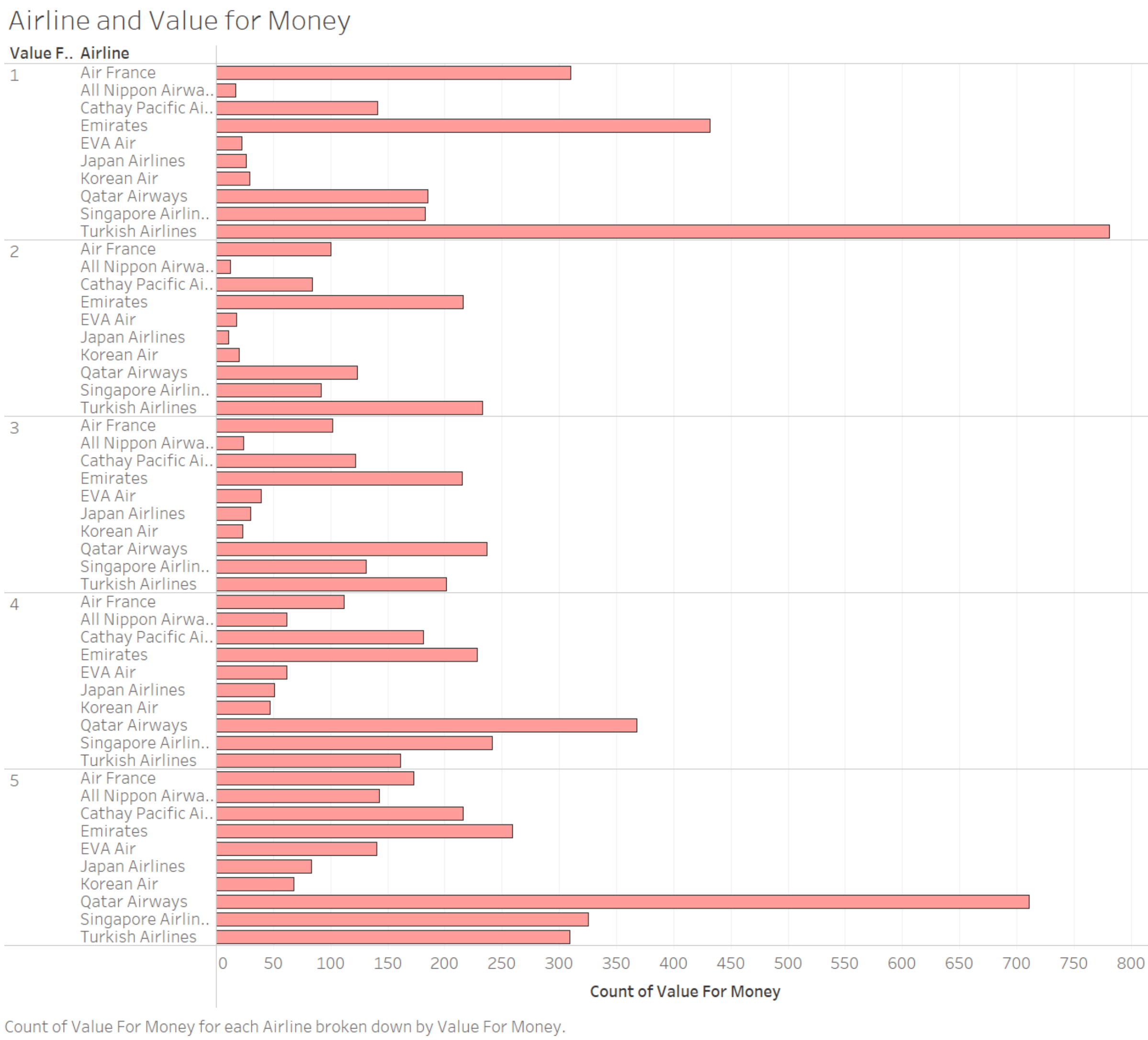
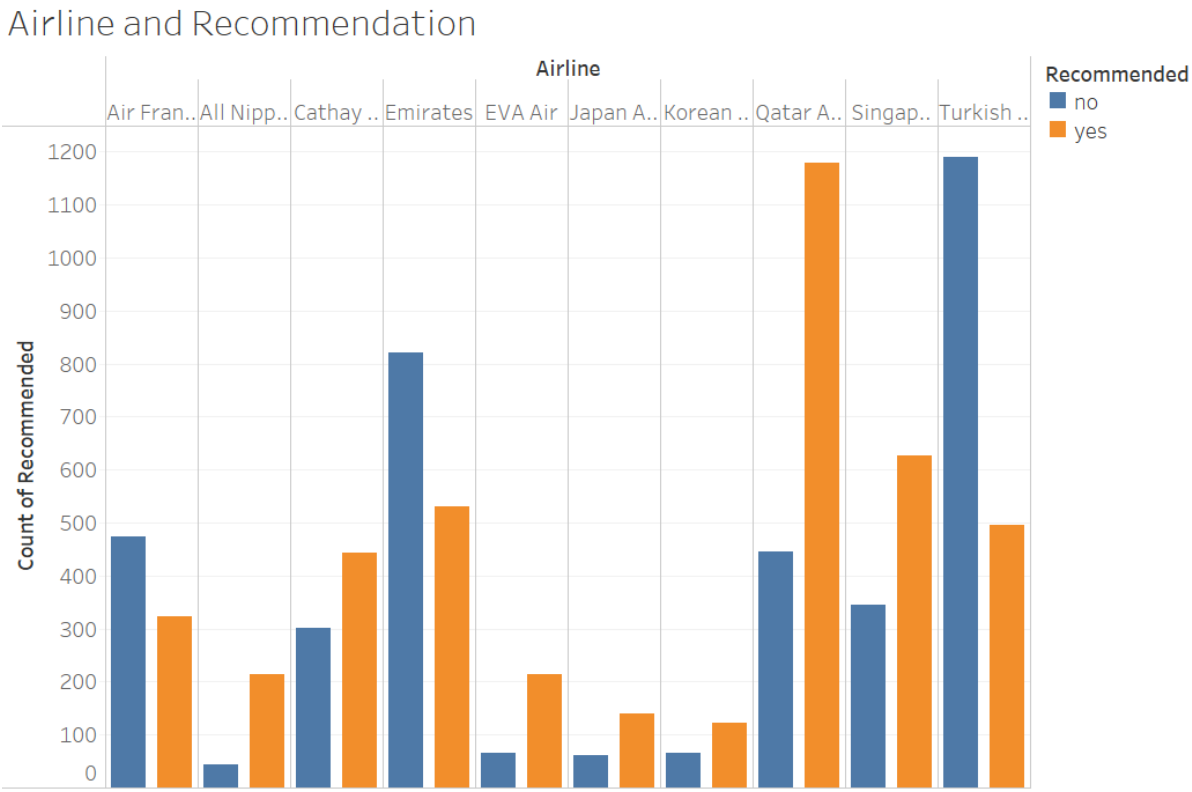


Fig 12. Airlines that gave value for the money been paid by their customers.



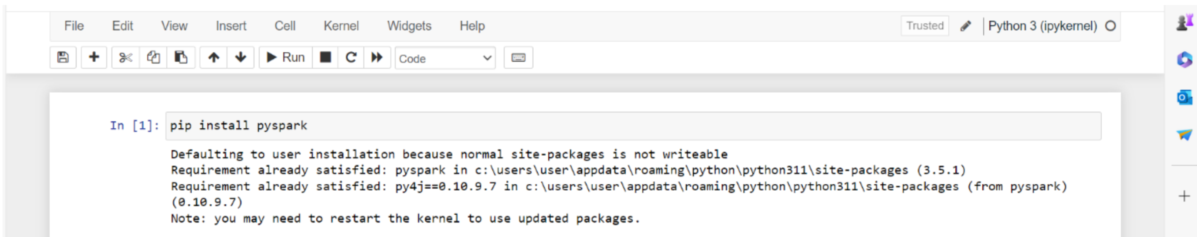
Count of Recommended for each Recommended broken down by Airline. Colour shows details about Recommended.

Fig 13. Airlines that customers will or will not recommend.

Month Flown, Airline and Recommendation

Airline (colour) broken down by Recommended vs. Month Flown.

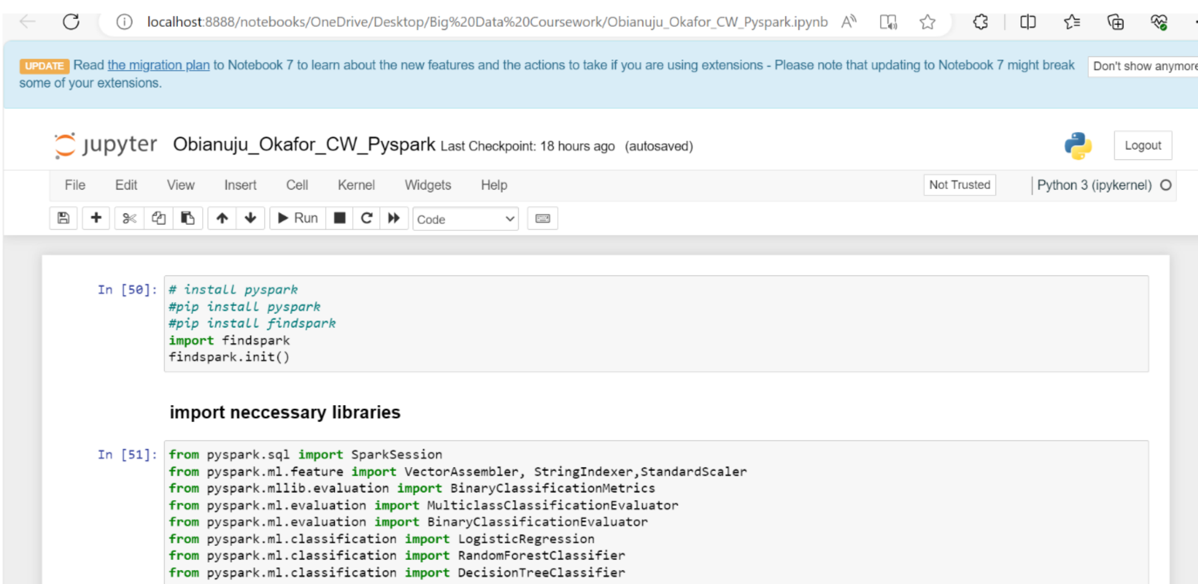
Fig 14. Airlines' recommendations from 2013 to 2024



```
In [1]: pip install pyspark

Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: pyspark in c:\users\user\appdata\roaming\python\python311\site-packages (3.5.1)
Requirement already satisfied: py4j==0.10.9.7 in c:\users\user\appdata\roaming\python\python311\site-packages (from pyspark)
(0.10.9.7)
Note: you may need to restart the kernel to use updated packages.
```

Fig 15. Pyspark Installation.




```
In [50]: # install pyspark
         #pip install pyspark
         #pip install findspark
         import findspark
         findspark.init()
```

import neccessary libraries

```
In [51]: from pyspark.sql import SparkSession
         from pyspark.ml.feature import VectorAssembler, StringIndexer,StandardScaler
         from pyspark.mllib.evaluation import BinaryClassificationMetrics
         from pyspark.ml.evaluation import MulticlassClassificationEvaluator
         from pyspark.ml.evaluation import BinaryClassificationEvaluator
         from pyspark.ml.classification import LogisticRegression
         from pyspark.ml.classification import RandomForestClassifier
         from pyspark.ml.classification import DecisionTreeClassifier
```

Fig 16. Importing findspark and all the libraries needed.

```
In [98]: # Create a SparkSession
         spark = SparkSession.builder.appName("AirlineRecommendationPredictor").master("local").getOrCreate()

In [99]: # Load the dataset
         data = spark.read.csv("C:/Users/ujuok/OneDrive/Desktop/Big Data Coursework/airlines_reviews.csv", header=True, inferSchema=True)
```

Fig 17. Load dataset in Pyspark.

```
In [100]: import pyspark.sql.functions as fc
          print((data.count(), len(data.columns)))
          #data.describe().show()


          # finding null values in each column
          data_null = data.agg(*[fc.count(fc.when(fc.isnull(c), c)).alias(c) for c in data.columns])
          #data_null.show()
          # no null values

          (8365, 17)
```

Fig 18. Number of rows and columns of the dataset and checking for missing values.

```
In [101]: # Drop columns not neccessary for classification
          airline_data = data.drop('Name', 'Review Date', 'Month Flown')

In [102]: airline_data.dtypes
          #airline_data.summary
          #airline_data.describe().show()

Out[102]: [('Title', 'string'),
           ('Airline', 'string'),
           ('Verified', 'string'),
           ('Reviews', 'string'),
           ('Type of Traveller', 'string'),
           ('Route', 'string'),
           ('Class', 'string'),
           ('Seat Comfort', 'string'),
           ('Staff Service', 'string'),
           ('Food & Beverages', 'string'),
           ('Inflight Entertainment', 'string'),
           ('Value For Money', 'string'),
           ('Overall Rating', 'string'),
           ('Recommended', 'string')]
```

Fig 19. Column drop and column dtype.

```
In [103]: # converting the interger columns to intergers

          from pyspark.sql.types import IntegerType
          from pyspark.sql.types import BooleanType

          airline_data = airline_data.withColumn("Seat Comfort", airline_data["Seat Comfort"].cast(IntegerType()))
          airline_data = airline_data.withColumn("Staff Service", airline_data["Staff Service"].cast(IntegerType()))
          airline_data = airline_data.withColumn("Food & Beverages", airline_data["Food & Beverages"].cast(IntegerType()))
          airline_data = airline_data.withColumn("Inflight Entertainment", airline_data["Inflight Entertainment"].cast(IntegerType()))
          airline_data = airline_data.withColumn("Value For Money", airline_data["Value For Money"].cast(IntegerType()))
          airline_data = airline_data.withColumn("Overall Rating", airline_data["Overall Rating"].cast(IntegerType()))
          airline_data = airline_data.withColumn("Verified", airline_data["Verified"].cast(BooleanType()))
```

Fig 20. Converting the columns to the correct datatype.

```
In [104]: airline_data.dtypes

Out[104]: [('Title', 'string'),
           ('Airline', 'string'),
           ('Verified', 'boolean'),
           ('Reviews', 'string'),
           ('Type of Traveller', 'string'),
           ('Route', 'string'),
           ('Class', 'string'),
           ('Seat Comfort', 'int'),
           ('Staff Service', 'int'),
           ('Food & Beverages', 'int'),
           ('Inflight Entertainment', 'int'),
           ('Value For Money', 'int'),
           ('Overall Rating', 'int'),
           ('Recommended', 'string')]
```

Fig 21. Converted columns.

jupyter Obianuju_Okafor_CW_Pyspark Last Checkpoint: Yesterday at 02:02 (autosaved)

```
In [105]: # Converting the target variable column from string no and yes to integer 0 and 1
          from pyspark.sql.functions import when

          airline_data = airline_data.withColumn('Recommended_new', when(airline_data.Recommended=='yes', 1).otherwise(0))
          airline_data = airline_data.drop("Recommended")

In [106]: # Converting the boolean variable column from bool False and True to integer 0 and 1
          from pyspark.sql.functions import when

          airline_data = airline_data.withColumn('Verified_new', when(airline_data.Verified=='True', 1).otherwise(0))
          airline_data = airline_data.drop("Verified")

In [107]: airline_data=airline_data.dropna()

In [108]: airline_data.groupBy('Recommended_new').count().orderBy('count').show()

          +---------------+-----+
          |Recommended_new|count|
          +---------------+-----+
          |              0| 3408|
          |              1| 3948|
          +---------------+-----+
```

Fig 22.

```
In [109]: print((airline_data.count(), len(airline_data.columns)))

          (7356, 14)
```

Fig 23. The number of rows and columns after converting and dropping

```
In [110]: from pyspark.ml import Pipeline
          from pyspark.ml.feature import StringIndexer

          #label encoding of categorical columns
          categorical_cols = ['Title','Airline','Reviews','Type of Traveller','Route','Class','Seat Comfort','Staff Service','Food & Bevera
          label_encoders = [StringIndexer(inputCol=col, outputCol=col + "_encoded").fit(airline_data) for col in categorical_cols]
          pipeline = Pipeline(stages=label_encoders)
          airline_data = pipeline.fit(airline_data).transform(airline_data)
```

Fig 24. Label encoding of categorical variables.

```
In [116]: # seperating target variable from other features
          features = airline_data.drop("Recommended_new")

          # putting all the other features as one
          features_col = features.columns
          print(features_col)
          assembler = VectorAssembler(inputCols=features_col, outputCol="Vfeatures")
          airline_data = assembler.transform(airline_data)
          #airline_data.show(2)
          airline_data = airline_data.select("Vfeatures", "Recommended_new")

          ['Title_encoded', 'Airline_encoded', 'Reviews_encoded', 'Type of Traveller_encoded', 'Route_encoded', 'Class_encoded', 'Seat Co
          mfort_encoded', 'Staff Service_encoded', 'Food & Beverages_encoded', 'Value For Money_encoded', 'Overall Rating_encoded', 'Infl
          ight Entertainment_encoded', 'Verified_new_encoded']
```

Fig 25. Separating the target features from the others

```
In [118]: scaled_data = StandardScaler(inputCol="Vfeatures", outputCol="features")
          airline_data = scaled_data.fit(airline_data).transform(airline_data)
```

Fig 26. Scaling the data

```
In [124]: #splitting into test and train data
          train_data, test_data = airline_data.randomSplit([0.8, 0.2], seed=42)
```

Fig 27. Splitting the data

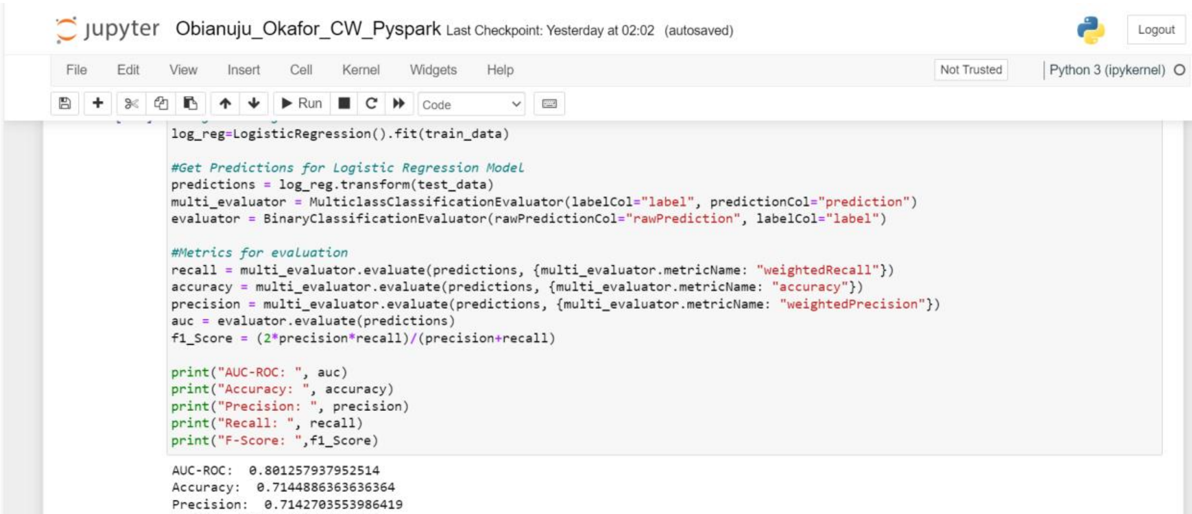


Fig 28. Logistic regression classification

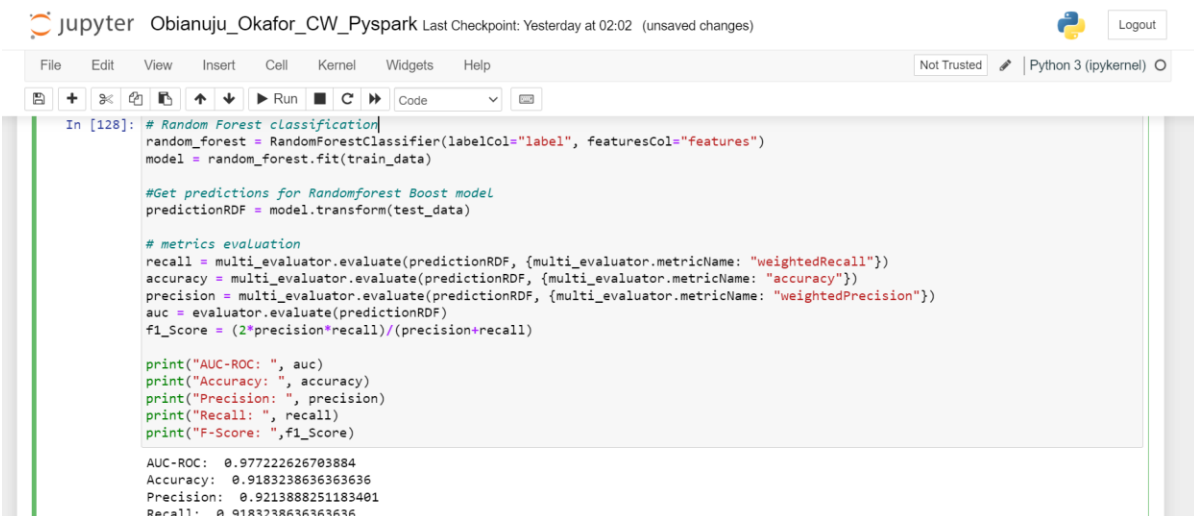


Fig 29. Random forest classification

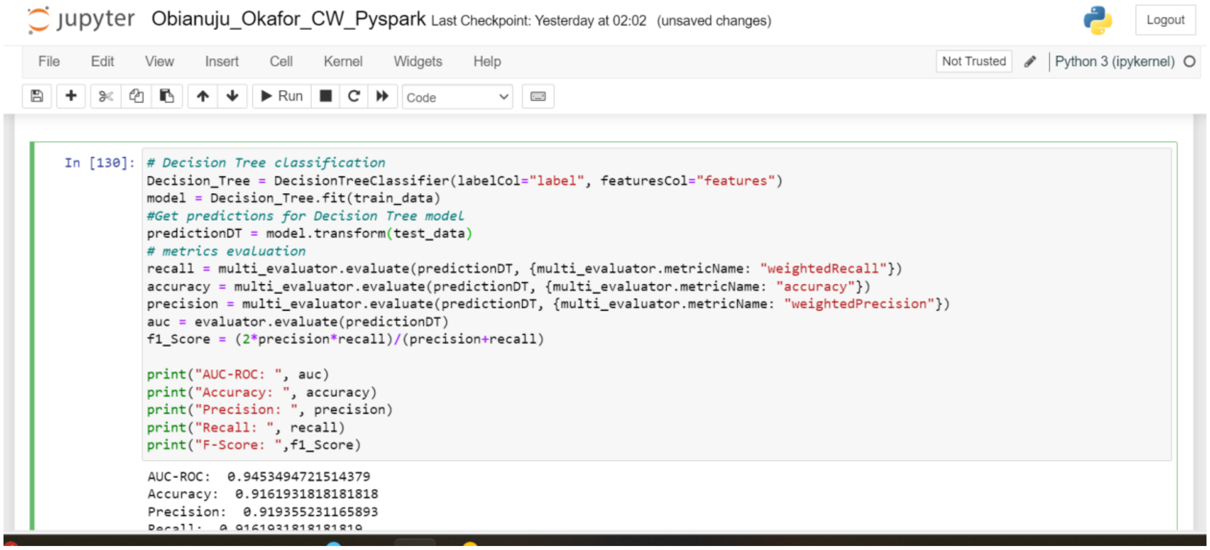


Fig 30. Decision Tree.

APPENDIX II

Code used for the implementation of the experiment:

```
from pyspark.sql import SparkSession
from pyspark.ml.feature import VectorAssembler, StringIndexer,StandardScaler
from pyspark.mllib.evaluation import BinaryClassificationMetrics
from pyspark.ml.evaluation import MulticlassClassificationEvaluator
from pyspark.ml.evaluation import BinaryClassificationEvaluator
from pyspark.ml.classification import LogisticRegression
from pyspark.ml.classification import RandomForestClassifier
from pyspark.ml.classification import DecisionTreeClassifier

# Create a SparkSession
spark =
SparkSession.builder.appName("AirlineRecommendationPredictor").master("local").getOrCreate()

# Load the dataset
data = spark.read.csv("C:/Users/ujuok/OneDrive/Desktop/Big Data Coursework/airlines_reviews.csv",
header=True, inferSchema=True)
```

```
import pyspark.sql.functions as fc
print((data.count(), len(data.columns)))
#data.describe().show()



# finding null values in each column
data_null = data.agg(*[fc.count(fc.when(fc.isnull(c), c)).alias(c) for c in data.columns])
#data_null.show()
# no null values

# Drop columns not neccessary for classification
airline_data = data.drop('Name', 'Review Date', 'Month Flown')

airline_data.dtypes
#airline_data.summary
#airline_data.describe().show()

# converting the interger columns from string to intergers

from pyspark.sql.types import IntegerType
from pyspark.sql.types import BooleanType

airline_data = airline_data.withColumn("Seat Comfort", airline_data["Seat
Comfort"].cast(IntegerType()))
airline_data = airline_data.withColumn("Staff Service", airline_data["Staff Service"].cast(IntegerType()))
airline_data = airline_data.withColumn("Food & Beverages", airline_data["Food &
Beverages"].cast(IntegerType()))
airline_data = airline_data.withColumn("Inflight Entertainment", airline_data["Inflight
Entertainment"].cast(IntegerType()))
airline_data = airline_data.withColumn("Value For Money", airline_data["Value For
Money"].cast(IntegerType()))
airline_data = airline_data.withColumn("Overall Rating", airline_data["Overall
Rating"].cast(IntegerType()))
```

```
airline_data = airline_data.withColumn("Verified", airline_data["Verified"].cast(BooleanType()))

airline_data.dtypes

# Converting the target variable column from string no and yes to integer 0 and 1
from pyspark.sql.functions import when

airline_data = airline_data.withColumn('Recommended_new', when(airline_data.Recommended=='yes',
1).otherwise(0))
airline_data = airline_data.drop("Recommended")


# Converting the boolean variable column from bool False and True to integer 0 and 1
from pyspark.sql.functions import when

airline_data = airline_data.withColumn('Verified_new', when(airline_data.Verified=='True',
1).otherwise(0))
airline_data = airline_data.drop("Verified")
airline_data.groupBy('Recommended_new').count().orderBy('count').show()
print((airline_data.count(), len(airline_data.columns)))

from pyspark.ml import Pipeline
from pyspark.ml.feature import StringIndexer

#label encoding of categorical columns
categorical_cols = ['Title','Airline','Reviews','Type of Traveller','Route','Class','Seat Comfort','Staff
Service','Food & Beverages','Value For Money','Overall Rating','Inflight Entertainment','Verified_new']
label_encoders = [StringIndexer(inputCol=col, outputCol=col + "_encoded").fit(airline_data) for col in
categorical_cols]
pipeline = Pipeline(stages=label_encoders)
airline_data = pipeline.fit(airline_data).transform(airline_data)
airline_data.dtypes
airline_data = airline_data.drop('Title','Airline','Reviews','Verified_new','Type of
Traveller','Route','Class','Seat Comfort','Staff Service','Food & Beverages','Value For Money','Overall
Rating','Inflight Entertainment','Recommended')
```

```
airline_data.dtypes

# seperating target variable from other features
features = airline_data.drop("Recommended_new")

# putting all the other features as one
features_col = features.columns
print(features_col)
assembler = VectorAssembler(inputCols=features_col, outputCol="Vfeatures")
airline_data = assembler.transform(airline_data)
#airline_data.show(2)
airline_data = airline_data.select("Vfeatures", "Recommended_new")

airline_data.show(5)

scaled_data = StandardScaler(inputCol="Vfeatures", outputCol="features")
airline_data = scaled_data.fit(airline_data).transform(airline_data)
#renaming the target column to label
airline_data = airline_data.select("features", "Recommended_new")
airline_data = airline_data.withColumnRenamed("Recommended_new","label")
#splitting into test and train data
train_data, test_data = airline_data.randomSplit([0.8, 0.2], seed=42)


# Logistic Regression
log_reg=LogisticRegression().fit(train_data)

#Get Predictions for Logistic Regression Model
predictions = log_reg.transform(test_data)
multi_evaluator = MulticlassClassificationEvaluator(labelCol="label", predictionCol="prediction")
evaluator = BinaryClassificationEvaluator(rawPredictionCol="rawPrediction", labelCol="label")

#Metrics for evaluation
recall = multi_evaluator.evaluate(predictions, {multi_evaluator.metricName: "weightedRecall"})
accuracy = multi_evaluator.evaluate(predictions, {multi_evaluator.metricName: "accuracy"})
```

```
precision = multi_evaluator.evaluate(predictions, {multi_evaluator.metricName: "weightedPrecision"})
auc = evaluator.evaluate(predictions)
f1_Score = (2*precision*recall)/(precision+recall)

print("AUC-ROC: ", auc)
print("Accuracy: ", accuracy)
print("Precision: ", precision)
print("Recall: ", recall)
print("F-Score: ",f1_Score)

#Display the Logistic Regresssion predictions
predictions.show()

# Random Forest classification
random_forest = RandomForestClassifier(labelCol="label", featuresCol="features")
model = random_forest.fit(train_data)

#Get predictions for Randomforest Boost model
predictionRDF = model.transform(test_data)

# metrics evaluation
recall = multi_evaluator.evaluate(predictionRDF, {multi_evaluator.metricName: "weightedRecall"})
accuracy = multi_evaluator.evaluate(predictionRDF, {multi_evaluator.metricName: "accuracy"})
precision = multi_evaluator.evaluate(predictionRDF, {multi_evaluator.metricName:
"weightedPrecision"})
auc = evaluator.evaluate(predictionRDF)
f1_Score = (2*precision*recall)/(precision+recall)

print("AUC-ROC: ", auc)
print("Accuracy: ", accuracy)
print("Precision: ", precision)
print("Recall: ", recall)
print("F-Score: ",f1_Score)
```

```
predictionRDF.show()

# Decision Tree classification
Decision_Tree = DecisionTreeClassifier(labelCol="label", featuresCol="features")
model = Decision_Tree.fit(train_data)
#Get predictions for Decision Tree model
predictionDT = model.transform(test_data)
# metrics evaluation
recall = multi_evaluator.evaluate(predictionDT, {multi_evaluator.metricName: "weightedRecall"})
accuracy = multi_evaluator.evaluate(predictionDT, {multi_evaluator.metricName: "accuracy"})
precision = multi_evaluator.evaluate(predictionDT, {multi_evaluator.metricName: "weightedPrecision"})
auc = evaluator.evaluate(predictionDT)
f1_Score = (2*precision*recall)/(precision+recall)

print("AUC-ROC: ", auc)
print("Accuracy: ", accuracy)
print("Precision: ", precision)
print("Recall: ", recall)
print("F-Score: ",f1_Score)

predictionDT.show()
```