```
pip install pymongo

     Requirement already satisfied: pymongo in /usr/local/lib/python3.10/dist-packages (4.4.1)
     Requirement already satisfied: dnspython<3.0.0,>=1.16.0 in /usr/local/lib/python3.10/dist-packages (from pymongo) (2.4.2)
```

```
import pandas as pd
import numpy as np
import pymongo
import json
```

```
from google.colab import drive
drive.mount('/content/drive/')

!ls '/content/drive/My Drive/'
filepath = '/content/drive/My Drive/'

     Drive already mounted at /content/drive/; to attempt to forcibly remount, call drive.mount("/content/drive/", force_remount=True).
     '01a. UrbanSEC.gdoc'
     '01b. RURAL SEC GRID.gdoc'
     '01c. New SEC Qns and Grid.gdoc'
     '01. SAMPLING DESIGN.gslides'
     '02. Solution - Sample Size determination.gsheet'
     '03a. Seat Computation (1).gsheet'
     '03a. Seat Computation.gsheet'
     '03. Opinion Polls and its complexities.gslides'
      4.gsheet
     'AbhishekLal_C23039_Data Exploration in Spark.ipynb'
     'Abhishek Lal_C23039.xlsx'
     'AbhishekLal_Resume (1).pdf'
      AbhishekLal_Resume.pdf
     'Adventure Works Sales (Multiple Sheets).gsheet'
      advertising.csv
     'AIF 01 nifty cos data.gsheet'
     'AIF 01 two asset portfolio (1).gsheet'
     'AIF 01 two asset portfolio.gsheet'
     'AIF 02 BANK DATA.gsheet'
      bangladesh.gsheet
     'car data.csv'
      cardekho.gsheet
     'Case - Data Driven Decision Making.gslides'
      Classroom
     'Class Work Dataset(Abhishek Lal) (1).gsheet'
     'Class Work Dataset(Abhishek Lal).gsheet'
     'class work.gsheet'
     'Colab Notebooks'
      communication_infrastructure.gsheet
     'data for class 17th.gsheet'
     'Data variable.txt'
     'FSA DuPont-Apollo tyres.gsheet'
      GapminderHealth.gsheet
      GTA5_review.txt
      gta_data_batch_2.csv
      kddcup.data_10_percent_corrected
      mean_years_school.gsheet
     "Participants' Worksheet.gsheet"
      PCA.ipynb
     'PGPDS Student Abhishek Lal.jpg'
      pima-indians-diabetes.gsheet
      PX03_Churn.gsheet
     'PX22DS-Introduction_to_ML-05_Classification_KNN (1).gdoc'
      PX22DS-Introduction_to_ML-05_Classification_KNN.gdoc
      Python_Programs_Set_1
      Python_Programs_Set_2
      san_francisco.gsheet
     'Supermarket aggregate data.csv'
      Train_BigMart.gsheet
     'Uber Cab Fare.csv'
      UNHealth.gsheet
      weatherAUS.csv
      weatherAUS.gsheet
```

```
xy = pd.read_csv(filepath + 'Uber Cab Fare.csv')
```

Reading the Data and Converting the Same to Dictionary

```
data=xy.to_dict(orient="records")
data
```

```
       'num_of_passengers': 1.0,
       'fare': 116.25,
       'tip': 0,
       'miscellaneous_fees': 6.0,
       'total_fare': 122.25,
       'surge_applied': 0},
      {'trip_duration': 955.0,
       'distance_traveled': 17.24,
       'num_of_passengers': 1.0,
       'fare': 221.25,
       'tip': 57,
       'miscellaneous_fees': 5.850000000000023,
       'total_fare': 284.1,
       'surge_applied': 0},
      {'trip_duration': 2687.0,
       'distance_traveled': 23.64,
       'num_of_passengers': 1.0,
       'fare': 345.0,
       'tip': 94,
       'miscellaneous_fees': 30.200000000000102,
       'total_fare': 469.2000000000001,
       'surge_applied': 1},
      {'trip_duration': 415.0,
       'distance_traveled': 1.87,
       'num_of_passengers': 1.0,
       'fare': 48.75,
       'tip': 0,
       'miscellaneous_fees': 26.625,
       'total_fare': 75.375,
       'surge_applied': 1},
      {'trip_duration': 654.0,
       'distance_traveled': 2.59,
       'num_of_passengers': 1.0,
       'fare': 67.5,
       'tip': 15,
       'miscellaneous_fees': 5.700000000000003,
       'total_fare': 88.2,
       'surge_applied': 0},
      {'trip_duration': 2251.0,
       'distance_traveled': 10.65,
       'num_of_passengers': 1.0,
       'fare': 198.75,
       'tip': 42,
       'miscellaneous_fees': 13.949999999999989,
       'total_fare': 254.7,
       'surge_applied': 0},
      {'trip_duration': 2540.0,
       'distance_traveled': 12.23,
       'num_of_passengers': 1.0,
       'fare': 243.75,
       'tip': 0,
       'miscellaneous_fees': 13.5,
       'total_fare': 257.25,
       'surge_applied': 0},
      ...]
```

Installation of Spark

```
!pip3 -q install pyspark
from pyspark.sql import SparkSession
spark = SparkSession.builder.appName('Praxis').getOrCreate()
```

```
from pyspark.sql import SparkSession
from pyspark.conf import SparkConf
from pyspark.sql.types import *
import pyspark.sql.functions as F
from pyspark.sql.functions import col, asc,desc
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
from pyspark.sql import SQLContext
from pyspark.mllib.stat import Statistics
import pandas as pd
from pyspark.sql.functions import udf
from pyspark.ml.feature import OneHotEncoder, StringIndexer, VectorAssembler,StandardScaler
from pyspark.ml import Pipeline
from sklearn.metrics import confusion_matrix
```

Converting Mongodb data to spark dataframe

```
ls
```

```
    drive/  sample_data/
```

SCHEMA FOR SPARK TABLE

```
# schema='{"fields":[{"metadata":{},"name":"selling_price","nullable":true,"type":"long"},{"metadata":{},"name":"km_driven","nullable":tr
```

```
# new_schema = StructType.fromJson(json.loads(schema))
```

Load Data

```
# Load and Read the dataset
data=spark.read.csv('/content/drive/My Drive/Uber Cab Fare.csv',inferSchema=True,header=True)
```

```
# Chech the datatypes of the inputs
data.printSchema()
```

```
    root
     |-- trip_duration: double (nullable = true)
     |-- distance_traveled: double (nullable = true)
     |-- num_of_passengers: double (nullable = true)
     |-- fare: double (nullable = true)
     |-- tip: integer (nullable = true)
     |-- miscellaneous_fees: double (nullable = true)
     |-- total_fare: double (nullable = true)
     |-- surge_applied: integer (nullable = true)
```

Data Exploration

```
data.count(), len(data.columns)
```

```
    (209673, 8)
```

```
data.dtypes
```

```
    [('trip_duration', 'double'),
     ('distance_traveled', 'double'),
     ('num_of_passengers', 'double'),
     ('fare', 'double'),
     ('tip', 'int'),
     ('miscellaneous_fees', 'double'),
     ('total_fare', 'double'),
     ('surge_applied', 'int')]
```

```
data.printSchema()
```

```
    root
     |-- trip_duration: double (nullable = true)
     |-- distance_traveled: double (nullable = true)
     |-- num_of_passengers: double (nullable = true)
     |-- fare: double (nullable = true)
     |-- tip: integer (nullable = true)
     |-- miscellaneous_fees: double (nullable = true)
     |-- total_fare: double (nullable = true)
     |-- surge_applied: integer (nullable = true)
```

```
data.count()
```

```
    209673
```

```
data.columns
```

```
    ['trip_duration',
     'distance_traveled',
     'num_of_passengers',
     'fare',
     'tip',
     'miscellaneous_fees',
     'total_fare',
     'surge_applied']
```

```
data.describe().show()
```

```
+-------+-----------------+------------------+------------------+-----------------+------------------+------------------+----------
|summary|    trip_duration| distance_traveled| num_of_passengers|             fare|               tip|miscellaneous_fees|        to
+-------+-----------------+------------------+------------------+-----------------+------------------+------------------+----------
|  count|           209673|            209673|            209673|           209673|            209673|            209673|
|   mean|1173.181477824994| 5.054431185703426|1.2928083253447034|99.62343112847154|13.030824188140581| 15.13682865700149|127.791083
| stddev|4775.653620869363|125.21741931669423| 0.930753593324805|85.60270241033903| 20.36776412100705|12.553435595551546| 98.797374
|    min|              0.0|              0.02|               0.0|              0.0|                 0|              -0.5|
|    max|          86387.0|          57283.91|               9.0|          4466.25|              2500|             435.0|
+-------+-----------------+------------------+------------------+-----------------+------------------+------------------+----------
```

```
for col in data.columns:
  print(col, "has", data.filter(data[col].isNull()).count(), "Null values.")
```

```
trip_duration has 0 Null values.
distance_traveled has 0 Null values.
num_of_passengers has 0 Null values.
fare has 0 Null values.
tip has 0 Null values.
miscellaneous_fees has 0 Null values.
total_fare has 0 Null values.
surge_applied has 0 Null values.
```

```
data.where(data.surge_applied == 0).show()
```

```
+-------------+-----------------+-----------------+------+---+------------------+------------------+-------------+
|trip_duration|distance_traveled|num_of_passengers|  fare|tip|miscellaneous_fees|        total_fare|surge_applied|
+-------------+-----------------+-----------------+------+---+------------------+------------------+-------------+
|        748.0|             2.75|              1.0|  75.0| 24| 6.299999999999997|             105.3|            0|
|       1187.0|             3.43|              1.0| 105.0| 24|13.200000000000017|142.20000000000002|            0|
|        671.0|             5.63|              3.0|  90.0|  0|              9.75|             99.75|            0|
|        329.0|             2.09|              1.0|  45.0| 12|13.199999999999989| 70.19999999999999|            0|
|        453.0|             2.22|              1.0|  52.5|  0|               6.0|              58.5|            0|
|        134.0|             1.48|              1.0| 33.75|  0|               6.0|             39.75|            0|
|        980.0|             3.48|              1.0|  90.0|  0|               6.0|              96.0|            0|
|        305.0|             2.04|              1.0|  45.0|  0|              9.75|             54.75|            0|
|       4059.0|             30.0|              1.0| 390.0|  0|            55.125|           445.125|            0|
|        530.0|             2.43|              1.0| 56.25|  0|              9.75|              66.0|            0|
|        773.0|             3.41|              2.0|  75.0|  9|              13.5|              97.5|            0|
|        357.0|             1.43|              1.0| 112.5|  0|              2.25|            114.75|            0|
|       1163.0|             5.62|              1.0|108.75|  0|              13.5|            122.25|            0|
|       1059.0|             5.39|              1.0| 112.5|  0|               6.0|             118.5|            0|
|       1467.0|             5.21|              1.0| 127.5|  0|               6.0|             133.5|            0|
|        929.0|             4.31|              1.0| 93.75| 25| 5.900000000000006|            124.65|            0|
|       1023.0|             2.41|              2.0|  90.0| 19|             6.125|           115.125|            0|
|        255.0|             2.48|              1.0|  45.0|  0|               6.0|              51.0|            0|
|        668.0|             3.11|              2.0|  75.0|  0|              9.75|             84.75|            0|
|      11323.0|             0.14|              1.0| 337.5| 68|2.1999999999999886|             407.7|            0|
+-------------+-----------------+-----------------+------+---+------------------+------------------+-------------+
only showing top 20 rows
```

```
new_data = data.filter((data.surge_applied != 0))
```

```
new_data.show()
```

```
+-------------+-----------------+-----------------+------+---+------------------+------------------+-------------+
|trip_duration|distance_traveled|num_of_passengers|  fare|tip|miscellaneous_fees|        total_fare|surge_applied|
+-------------+-----------------+-----------------+------+---+------------------+------------------+-------------+
|        730.0|             3.12|              1.0| 71.25|  0|            26.625|            97.875|            1|
|        355.0|             1.74|              1.0|  45.0|  0|            26.625|            71.625|            1|
|       1288.0|             5.21|              1.0|116.25| 36| 26.32499999999999|           178.575|            1|
|      84305.0|             4.94|              2.0|116.25| 29| 26.19999999999999|            171.45|            1|
|        328.0|             2.16|              1.0| 48.75| 15|26.700000000000003|             90.45|            1|
|       1848.0|             4.83|              1.0| 142.5| 17| 26.42500000000001|           185.925|            1|
|        234.0|             1.61|              0.0|  37.5|  0|            26.625|            64.125|            1|
|       1537.0|             7.64|              1.0|146.25| 14| 26.42500000000001|           186.675|            1|
|        319.0|             1.67|              1.0| 41.25|  0|            26.625|            67.875|            1|
|        375.0|             1.45|              1.0|  45.0| 24|            33.825|           102.825|            1|
|        586.0|             2.57|              1.0| 63.75|  0|            26.625|            90.375|            1|
|       1617.0|             3.23|              6.0|123.75|  8|            26.125|           157.875|            1|
|       1271.0|             4.84|              1.0|116.25|  0|            26.625|           142.875|            1|
|        915.0|             4.46|              1.0| 93.75| 26| 33.70000000000002|153.45000000000002|            1|
|        742.0|             4.89|              3.0|  90.0| 11|            30.625|           131.625|            1|
|       1149.0|             9.35|              1.0|153.75|  0|            26.625|           180.375|            1|
|       1146.0|             3.88|              1.0|  97.5| 37|26.825000000000017|161.32500000000002|            1|
|       3722.0|            19.17|              1.0| 322.5| 89| 34.30000000000001|             445.8|            1|
|       1911.0|             6.76|              1.0|161.25| 47| 26.57499999999999|           234.825|            1|
|        622.0|             2.03|              1.0|  60.0|  0|            26.625|            86.625|            1|
+-------------+-----------------+-----------------+------+---+------------------+------------------+-------------+
only showing top 20 rows
```

```
from pyspark.sql.functions import *
```

Showing group wise average selling_price

```
data.groupBy('surge_applied').mean('total_fare').show()
```

```
+-------------+------------------+
|surge_applied|   avg(total_fare)|
+-------------+------------------+
|            1| 170.0919412425104|
|            0|111.29808193296972|
+-------------+------------------+
```

```
import pandas as pd
import numpy as np
```

```
DF = data.toPandas()
```

```
DF.head()
```

	trip_duration	distance_traveled	num_of_passengers	fare	tip	miscellaneous_fees	total_fare	surge_applied
0	748.0	2.75	1.0	75.00	24	6.300	105.300	0
1	1187.0	3.43	1.0	105.00	24	13.200	142.200	0
2	730.0	3.12	1.0	71.25	0	26.625	97.875	1
3	671.0	5.63	3.0	90.00	0	9.750	99.750	0
4	329.0	2.09	1.0	45.00	12	13.200	70.200	0

```
cor = DF.corr()
```

```
cor.total_fare.sort_values(ascending=False)
```

```
total_fare            1.000000
fare                  0.966748
tip                   0.508639
miscellaneous_fees    0.452568
surge_applied         0.267350
trip_duration         0.142159
distance_traveled     0.036677
num_of_passengers     0.014234
Name: total_fare, dtype: float64
```

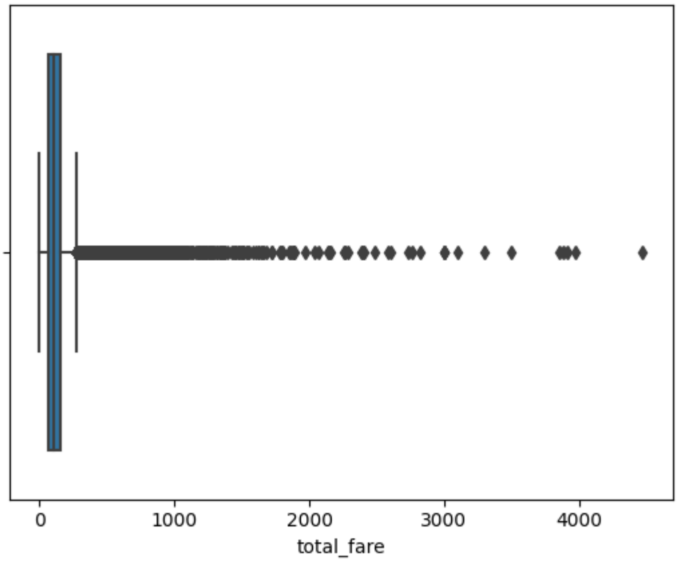
```
import matplotlib.pyplot as plt
import seaborn as sns
```

```
sns.distplot(x=DF['total_fare'], bins=20, kde=True)
```

```
<ipython-input-44-b42c81b330c0>:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751
```

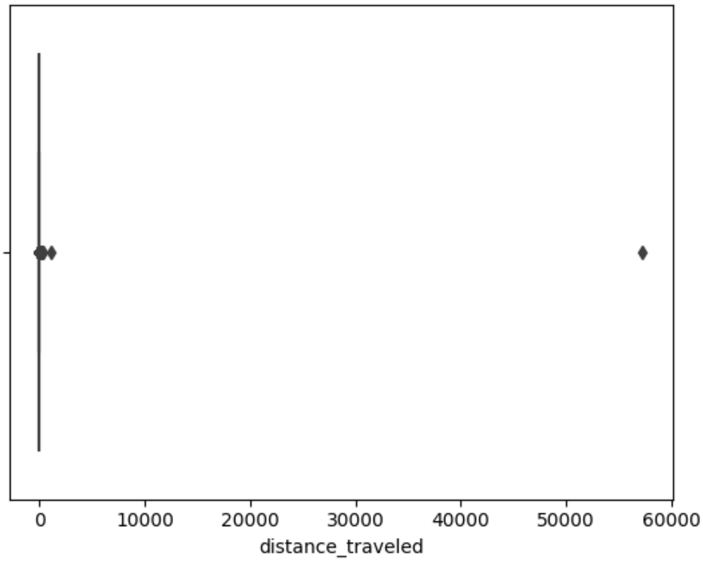
```
sns.boxplot(x='total_fare',data=DF)
```

```
<Axes: xlabel='total_fare'>
```



```
sns.boxplot(x='distance_traveled', data=DF)
```

```
<Axes: xlabel='distance_traveled'>
```

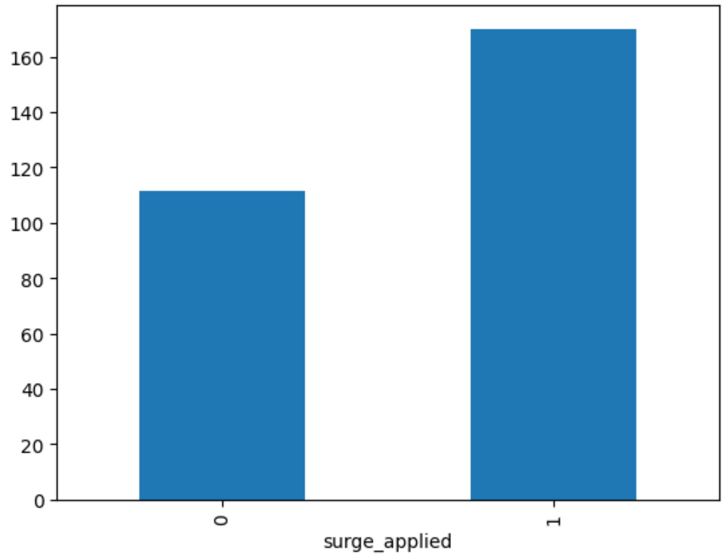


```
DF['total_fare'].plot(kind='hist', bins=25)
```

```
<Axes: ylabel='Frequency'>
```

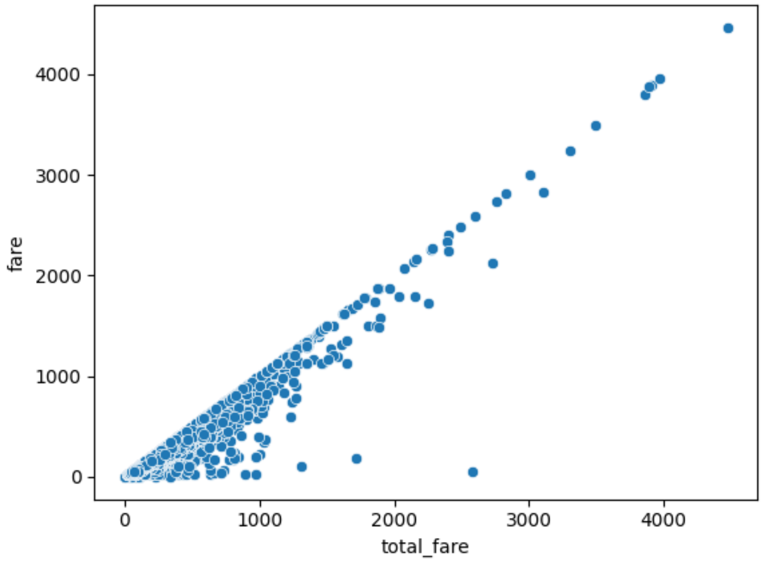
```
plt.figure()
DF.groupby('surge_applied')['total_fare'].mean().plot(kind='bar')
```

```
<Axes: xlabel='surge_applied'>
```



```
sns.scatterplot(x='total_fare', y='fare',data=DF )
```

```
<Axes: xlabel='total_fare', ylabel='fare'>
```



```
from datetime import date
```

```
data.show(5)
```

```
+-------------+-----------------+-----------------+-----+---+------------------+------------------+-------------+
|trip_duration|distance_traveled|num_of_passengers| fare|tip|miscellaneous_fees|        total_fare|surge_applied|
+-------------+-----------------+-----------------+-----+---+------------------+------------------+-------------+
|        748.0|             2.75|              1.0| 75.0| 24| 6.299999999999997|             105.3|            0|
|       1187.0|             3.43|              1.0|105.0| 24|13.200000000000017|142.20000000000002|            0|
|        730.0|             3.12|              1.0|71.25|  0|            26.625|            97.875|            1|
|        671.0|             5.63|              3.0| 90.0|  0|              9.75|             99.75|            0|
|        329.0|             2.09|              1.0| 45.0| 12|13.199999999999989| 70.19999999999999|            0|
+-------------+-----------------+-----------------+-----+---+------------------+------------------+-------------+
only showing top 5 rows
```

Checking For Correlation

```
from pyspark.sql.functions import corr
```

```
type(data)
```

```
pyspark.sql.dataframe.DataFrame
```

Data Preparation using One-Hot Encoder and Vector Assembler

```
from pyspark.ml.feature import StringIndexer
```

```
from pyspark.ml.feature import OneHotEncoder
```

```
from pyspark.ml.feature import VectorAssembler
```

```
data.show(7)
```

```
+-------------+-----------------+-----------------+-----+---+------------------+------------------+-------------+
|trip_duration|distance_traveled|num_of_passengers| fare|tip|miscellaneous_fees|        total_fare|surge_applied|
+-------------+-----------------+-----------------+-----+---+------------------+------------------+-------------+
|        748.0|             2.75|              1.0| 75.0| 24| 6.299999999999997|             105.3|            0|
|       1187.0|             3.43|              1.0|105.0| 24|13.200000000000017|142.20000000000002|            0|
|        730.0|             3.12|              1.0|71.25|  0|            26.625|            97.875|            1|
|        671.0|             5.63|              3.0| 90.0|  0|              9.75|             99.75|            0|
|        329.0|             2.09|              1.0| 45.0| 12|13.199999999999989| 70.19999999999999|            0|
|        355.0|             1.74|              1.0| 45.0|  0|            26.625|            71.625|            1|
|        453.0|             2.22|              1.0| 52.5|  0|               6.0|              58.5|            0|
+-------------+-----------------+-----------------+-----+---+------------------+------------------+-------------+
only showing top 7 rows
```

```
from pyspark.ml.linalg import DenseVector
```

```
from pyspark.ml.feature import StandardScaler
```

```
input_data = new_data.rdd.map(lambda x: (x[0], DenseVector(x[1:])))
```

```
df_assembler = VectorAssembler(inputCols=['trip_duration', 'distance_traveled', 'num_of_passengers', 'tip',
                                          'surge_applied'], outputCol="features")
#df = df_assembler.transform(df)
```

```
df = df_assembler.transform(data)
```

```
df.show(5)
```

```
+-------------+-----------------+-----------------+-----+---+------------------+------------------+-------------+------------------
|trip_duration|distance_traveled|num_of_passengers| fare|tip|miscellaneous_fees|        total_fare|surge_applied|           featur
+-------------+-----------------+-----------------+-----+---+------------------+------------------+-------------+------------------
|        748.0|             2.75|              1.0| 75.0| 24| 6.299999999999997|             105.3|            0|[748.0,2.75,1.0,2.
|       1187.0|             3.43|              1.0|105.0| 24|13.200000000000017|142.20000000000002|            0|[1187.0,3.43,1.0,.
|        730.0|             3.12|              1.0|71.25|  0|            26.625|            97.875|            1|[730.0,3.12,1.0,0.
|        671.0|             5.63|              3.0| 90.0|  0|              9.75|             99.75|            0|[671.0,5.63,3.0,0.
|        329.0|             2.09|              1.0| 45.0| 12|13.199999999999989| 70.19999999999999|            0|[329.0,2.09,1.0,1.
+-------------+-----------------+-----------------+-----+---+------------------+------------------+-------------+------------------
only showing top 5 rows
```

```
df.select(['features','total_fare']).show(5)
```

```
+--------------------+------------------+
|            features|        total_fare|
+--------------------+------------------+
|[748.0,2.75,1.0,2...|             105.3|
|[1187.0,3.43,1.0,...|142.20000000000002|
|[730.0,3.12,1.0,0...|            97.875|
|[671.0,5.63,3.0,0...|             99.75|
|[329.0,2.09,1.0,1...| 70.19999999999999|
+--------------------+------------------+
only showing top 5 rows
```

Building and Comparing ML Models

Creation and application of 2 Transformers / Estimators in a pipeline

```
# pipeline_stages=Pipeline()\
#                 .setStages([type_indexer,type_encoder])
# pipeline_model=pipeline_stages.fit(new_data)
```

```
from pyspark.ml.feature import StandardScaler
```

```
# Initialize the `standardScaler`
standardScaler = StandardScaler(inputCol="features", outputCol="features_scaled")
```

```
# Fit the DataFrame to the scaler
scaler = standardScaler.fit(df)
```

```
# Transform the data in `df` with the scaler
scaled_df = scaler.transform(df)
```

```
scaled_df.take(2)
```

```
[Row(trip_duration=748.0, distance_traveled=2.75, num_of_passengers=1.0, fare=75.0, tip=24, miscellaneous_fees=6.299999999999997,
total_fare=105.3, surge_applied=0, features=DenseVector([748.0, 2.75, 1.0, 24.0, 0.0]), features_scaled=DenseVector([0.1566,
0.022, 1.0744, 1.1783, 0.0])),
 Row(trip_duration=1187.0, distance_traveled=3.43, num_of_passengers=1.0, fare=105.0, tip=24,
miscellaneous_fees=13.200000000000017, total_fare=142.20000000000002, surge_applied=0, features=DenseVector([1187.0, 3.43, 1.0,
24.0, 0.0]), features_scaled=DenseVector([0.2486, 0.0274, 1.0744, 1.1783, 0.0]))]
```

Train-Test Split

```
# Split the data into train and test sets
train_data, test_data = scaled_df.randomSplit([.7,.3],seed=1234)
print("Training Dataset Count: " + str(train_data.count()))
print("Test Dataset Count: " + str(test_data.count()))
```

```
Training Dataset Count: 146904
Test Dataset Count: 62769
```

```
train_data.show()
```

```
+-------------+-----------------+-----------------+------+---+------------------+-----------------+-------------+------------------
|trip_duration|distance_traveled|num_of_passengers|  fare|tip|miscellaneous_fees|       total_fare|surge_applied|            featur
+-------------+-----------------+-----------------+------+---+------------------+-----------------+-------------+------------------
|          0.0|             0.02|              1.0| 150.0|  0|              2.25|           152.25|            0|(5,[1,2],[0.02,1.0
|          0.0|             0.02|              1.0| 187.5|  0|              2.25|           189.75|            0|(5,[1,2],[0.02,1.0
|          0.0|             0.06|              2.0| 225.0| 45|2.6999999999999886|            272.7|            0|[0.0,0.06,2.0,45..
|          0.0|             0.06|              2.0| 352.5| 71|2.1999999999999886|            425.7|            0|[0.0,0.06,2.0,71..
|          0.0|              0.1|              1.0|  60.0|  8|              1.75|            69.75|            0|[0.0,0.1,1.0,8.0,.
|          0.0|              0.1|              1.0|  75.0| 15| 2.700000000000003|             92.7|            0|[0.0,0.1,1.0,15.0.
|          0.0|             0.11|              1.0| 18.75| 15|               6.0|            39.75|            0|[0.0,0.11,1.0,15..
|          0.0|             0.11|              1.0| 18.75| 38|              9.25|             66.0|            0|[0.0,0.11,1.0,38..
|          0.0|             0.13|              1.0| 262.5|  0|              2.25|           264.75|            0|(5,[1,2],[0.13,1.0
|          0.0|             0.16|              1.0| 165.0|  1|1.8499999999999943|           167.85|            0|[0.0,0.16,1.0,1.0.
|          0.0|             0.16|              2.0| 600.0|  0|              2.25|           602.25|            0|(5,[1,2],[0.16,2.0
|          0.0|             0.18|              1.0| 18.75| 38|              9.25|             66.0|            0|[0.0,0.18,1.0,38..
|          0.0|             0.18|              1.0| 187.5| 38|2.1999999999999886|            227.7|            0|[0.0,0.18,1.0,38..
|          0.0|             0.18|              1.0| 487.5|  8|              1.75|           497.25|            0|[0.0,0.18,1.0,8.0.
|          0.0|             0.21|              2.0| 187.5| 38|2.1999999999999886|            227.7|            0|[0.0,0.21,2.0,38..
|          0.0|             0.23|              1.0| 487.5| 98|2.2000000000000455|            587.7|            0|[0.0,0.23,1.0,98..
|          0.0|             0.24|              1.0|  75.0|  0|              2.25|            77.25|            0|(5,[1,2],[0.24,1.0
|          0.0|              0.4|              1.0|  60.0| 16|1.8500000000000085|77.85000000000001|            0|[0.0,0.4,1.0,16.0.
|          0.0|             7.19|              1.0|138.75|  0|              9.75|            148.5|            0|(5,[1,2],[7.19,1.0
|          1.0|             0.02|              1.0| 18.75|375|               6.0|           399.75|            0|[1.0,0.02,1.0,375.
+-------------+-----------------+-----------------+------+---+------------------+-----------------+-------------+------------------
only showing top 20 rows
```

Training the Model

```
from pyspark.ml.regression import LinearRegression,DecisionTreeRegressor
from pyspark.ml.evaluation import RegressionEvaluator


# Initialize `lr`
lr = LinearRegression(labelCol="total_fare", maxIter=10, regParam=0.3, elasticNetParam=0.8)


# Fit the data to the model
linearModel = lr.fit(train_data)


# Generate predictions
predicted = linearModel.transform(test_data)
predicted.show()
```

```
+-------------+-----------------+-----------------+-----+---+------------------+----------+-------------+--------------------+-----
|trip_duration|distance_traveled|num_of_passengers| fare|tip|miscellaneous_fees|total_fare|surge_applied|            features|
+-------------+-----------------+-----------------+-----+---+------------------+----------+-------------+--------------------+-----
|          0.0|             0.02|              1.0|18.75|  0|              13.5|     32.25|            0|(5,[1,2],[0.02,1.0])|(5,[1
|          0.0|             0.02|              1.0|135.0|  0|              2.25|    137.25|            0|(5,[1,2],[0.02,1.0])|(5,[1
|          0.0|             0.05|              1.0| 45.0|  0|              2.25|     47.25|            0|(5,[1,2],[0.05,1.0])|(5,[1
|          0.0|             0.43|              1.0|300.0| 60|2.6999999999999886|     362.7|            0|[0.0,0.43,1.0,60....|[0.0,
|          1.0|             0.02|              1.0| 75.0| 19|2.5999999999999943|      96.6|            0|[1.0,0.02,1.0,19....|[2.09
|          1.0|             0.03|              1.0|18.75| 98|               5.5|    122.25|            0|[1.0,0.03,1.0,98....|[2.09
|          1.0|             0.03|              1.0|112.5| 29|1.9000000000000057|     143.4|            0|[1.0,0.03,1.0,29....|[2.09
|          1.0|             0.05|              1.0|150.0| 30|2.6999999999999886|     182.7|            0|[1.0,0.05,1.0,30....|[2.09
|          1.0|             0.05|              2.0| 75.0| 15| 2.700000000000003|      92.7|            0|[1.0,0.05,2.0,15....|[2.09
|          1.0|             0.11|              1.0|18.75|  0|               6.0|     24.75|            0|[1.0,0.11,1.0,0.0...|[2.09
|          1.0|             0.11|              1.0| 90.0| 18| 2.700000000000003|     110.7|            0|[1.0,0.11,1.0,18....|[2.09
|          1.0|             0.37|              1.0| 75.0| 15| 2.700000000000003|      92.7|            0|[1.0,0.37,1.0,15....|[2.09
|          2.0|             0.02|              1.0| 60.0| 19|1.9249999999999972|    80.925|            0|[2.0,0.02,1.0,19....|[4.18
|          2.0|             0.02|              1.0| 75.0| 15| 2.700000000000003|      92.7|            0|[2.0,0.02,1.0,15....|[4.18
|          2.0|             0.02|              1.0|300.0|  0|              2.25|    302.25|            0|[2.0,0.02,1.0,0.0...|[4.18
|          2.0|             0.03|              1.0|18.75|  0|               6.0|     24.75|            0|[2.0,0.03,1.0,0.0...|[4.18
|          2.0|             0.03|              1.0| 90.0|  4|               2.0|      96.0|            0|[2.0,0.03,1.0,4.0...|[4.18
|          2.0|             0.03|              1.0|150.0|  0|              2.25|    152.25|            0|[2.0,0.03,1.0,0.0...|[4.18
|          2.0|             0.05|              1.0| 75.0| 15| 2.700000000000003|      92.7|            0|[2.0,0.05,1.0,15....|[4.18
|          2.0|             0.05|              1.0| 75.0| 15| 2.700000000000003|      92.7|            0|[2.0,0.05,1.0,15....|[4.18
+-------------+-----------------+-----------------+-----+---+------------------+----------+-------------+--------------------+-----
only showing top 20 rows
```

Getting the outcome

```
# Coefficients for the model
linearModel.coefficients
```

```
DenseVector([0.0026, 0.0152, 0.8405, 2.3339, 32.2252])
```

```
# Intercept for the model
linearModel.intercept
```

```
84.20828665780299
```

```
# Get the RMSE
linearModel.summary.rootMeanSquaredError
```

```
83.66711413638198
```

```
# Get the R2
linearModel.summary.r2
```

```
0.2939909414691463
```

```
dtr = DecisionTreeRegressor(labelCol="total_fare",featuresCol='features')


dtrModel = dtr.fit(train_data)


predicted = dtrModel.transform(test_data)
predicted.show()
```

```
+-------------+-----------------+-----------------+-----+---+------------------+----------+-------------+--------------------+-----
|trip_duration|distance_traveled|num_of_passengers| fare|tip|miscellaneous_fees|total_fare|surge_applied|            features|
+-------------+-----------------+-----------------+-----+---+------------------+----------+-------------+--------------------+-----
|          0.0|             0.02|              1.0|18.75|  0|              13.5|     32.25|            0|(5,[1,2],[0.02,1.0])|(5,[1
|          0.0|             0.02|              1.0|135.0|  0|              2.25|    137.25|            0|(5,[1,2],[0.02,1.0])|(5,[1
|          0.0|             0.05|              1.0| 45.0|  0|              2.25|     47.25|            0|(5,[1,2],[0.05,1.0])|(5,[1
|          0.0|            0.43|             1.0|300.0| 60|2.6999999999999886|     362.7|           0|[0.0,0.43,1.0,60....|[0.0,
|          1.0|            0.02|             1.0| 75.0| 19|2.5999999999999943|      96.6|           0|[1.0,0.02,1.0,19....|[2.09
|          1.0|            0.03|             1.0|18.75| 98|               5.5|    122.25|           0|[1.0,0.03,1.0,98....|[2.09
|          1.0|            0.03|             1.0|112.5| 29|1.9000000000000057|     143.4|           0|[1.0,0.03,1.0,29....|[2.09
|          1.0|            0.05|             1.0|150.0| 30|2.6999999999999886|     182.7|           0|[1.0,0.05,1.0,30....|[2.09
|          1.0|            0.05|             2.0| 75.0| 15| 2.700000000000003|      92.7|           0|[1.0,0.05,2.0,15....|[2.09
|          1.0|            0.11|             1.0|18.75|  0|               6.0|     24.75|           0|[1.0,0.11,1.0,0.0...|[2.09
|          1.0|            0.11|             1.0| 90.0| 18| 2.700000000000003|     110.7|           0|[1.0,0.11,1.0,18....|[2.09
|          1.0|            0.37|             1.0| 75.0| 15| 2.700000000000003|      92.7|           0|[1.0,0.37,1.0,15....|[2.09
|          2.0|            0.02|             1.0| 60.0| 19|1.9249999999999972|    80.925|           0|[2.0,0.02,1.0,19....|[4.18
|          2.0|            0.02|             1.0| 75.0| 15| 2.700000000000003|      92.7|           0|[2.0,0.02,1.0,15....|[4.18
|          2.0|            0.02|             1.0|300.0|  0|              2.25|    302.25|           0|[2.0,0.02,1.0,0.0...|[4.18
|          2.0|            0.03|             1.0|18.75|  0|               6.0|     24.75|           0|[2.0,0.03,1.0,0.0...|[4.18
|          2.0|            0.03|             1.0| 90.0|  4|               2.0|      96.0|           0|[2.0,0.03,1.0,4.0...|[4.18
|          2.0|            0.03|             1.0|150.0|  0|              2.25|    152.25|           0|[2.0,0.03,1.0,0.0...|[4.18
|          2.0|            0.05|             1.0| 75.0| 15| 2.700000000000003|      92.7|           0|[2.0,0.05,1.0,15....|[4.18
|          2.0|            0.05|             1.0| 75.0| 15| 2.700000000000003|      92.7|           0|[2.0,0.05,1.0,15....|[4.18
+-------------+----------------+----------------+-----+---+------------------+----------+------------+--------------------+-----
only showing top 20 rows
```

```
evaluate_r2 = RegressionEvaluator(predictionCol="prediction",labelCol="total_fare",metricName="r2")
```

```
evaluate_r2.evaluate(predicted)
```

```
0.7179675648501878
```

```
evaluate = RegressionEvaluator(labelCol="total_fare",metricName="rmse")
```

```
rmse = evaluate.evaluate(predicted)
print("RMSE: ",rmse)
```

```
RMSE:  51.488353691131145
```

✓ 0s completed at 11:06 PM