FLIGHT TICKET PRICE PREDICTION

Submitted by:

KONATALA MOHIT

ACKNOWLEDGMENT

I am grateful to the writers, developers and authority of the website pages machinelearningmastry, geekforgeeks, wikipedia and stackoverflow which have helped me to refer for whenever there was need for guidance. Economic times helped me to get a better understanding of the data to analyse and explore data. I am most grateful to DataTrained who have guided me in learning and enhancing my data science skills required to analyse and solve this project.

INTRODUCTION:

In this report I will be discussing about flight ticket price prediction using few of the machine learning models via python and its libraries. Transportation is one greatest invention of man-kind which truly helped connecting the world resources around the world. The invention of airplane was done by Wright Brothers on December 17th 1903. The invention of airplane was an indeed a revolution as it abled man-kind to travel different continent in a jiffy. It has been over a century since the invention and Flight travelling is much more convenient and comfortable and if one has hefty money even luxurious too. For today's generation air travel has become one of the many common travelling commodities.

The ticket price of the airbus varies drastically every day. There are various factors that influence the ticket price. To predict the pricing of airbus travelling one must have a good understanding about the air travelling company reservation booking system and air travel traffic also basic air travel seating reservation class.

Review of Literature:

For the better understanding of the data refer to various article. I would recommend economic times from India times as it only not elaborate about the factors but also government regulations on air travel.

Undertaken Problem:

Objective of the project is to predict the flight ticket price by analysing various factors that affect the ticket price. Flight ticket price prediction is one the many problems in Data Science as it ticks all the basic fundamentals required to analyse data and creating a machine learning model to predict the necessary outcome.

Mathematical/ Analytical Modelling of the Problem:

Price is the target data in the dataset and the variables present and the target data are in continuous form and categorical form, therefore Regression Machine Learning Models are used in this project. Regression is for estimating continuous form of data by using established relation between feature and target variables. Descriptive Analysis is used to study and observe the data.

Data Sources and their formats:

The source of the data is collected from makemytrip website using selenium webdriver. The data is provided in the excel format. Data contains 5823 entries having 10 variables in the dataset. The data contains flight from Delhi to three other major Indian cities i.e., Mumbai, Kolkata and Chennai.

Data Pre-processing:

Data pre-processing has two main steps i.e., Data Cleaning and Data Transforming.

Data Cleaning:

Data Cleaning is one of the most important steps creating a machine model. If an uncleaned data is fed to a machine learning model, then the model will perform very poorly.

The first and foremost step in data cleaning is removing and replacing null values in the dataset though our dataset doesn't contain null values. The dataset contains few null values in one row. Dropping those null values as it is only one row. Date column contains weekday and date of journey in it. Splitting them into two columns and dropping the Date columns, i.e., separating weekday of journey and date of journey. The date is still a bit company it has to much variation so splitting into three more columns i.e., day, month and year and dropping date of journey columns as it has been divided into three new columns. Month and year have only one unique value each so dropping these two columns as they won't affect the outcome price.

Duration time is in string format so making it numerical by splitting its first into two columns i.e., hours and minutes respectively and removing time units from it (hour and minute indications). After splitting multiplying hours column with 60 to convert it into minutes, after converting the hours into minutes adding the minutes column to make the complete duration of flight travel. Removing the old Duration columns and its split columns as we have new duration column in numerical minutes format.

Departure time and Arrival time are bit complexed to reduced the complexity of the columns each of the column into two columns each i.e., hour and minutes of departure and arrival. This simplifies the complexity of the column and convert into numerical form.

Number of stops columns should have only three variations though due mentioning of city stop in them has made them a bit complex so splitting them and dropping the stoppage cities as they aren't useful in determining our target variable. Flights with only two stoppage data is less than 1% of the whole so they are outliers therefore dropping flights with two stoppage points.

Data Transforming:

After Data Cleaning the data must be transformed into numerical form as one can't feed ordinal data to machine learning model and also the data must be normalized as normalizing the data the machine learning model gives equal importance to all the data.

In this project I am using label encoder to transform the ordinal data present in the data set into numerical form. Using Standard Scalar function to normalize the data.

Note: Before normalize the data separated the target variable from feature variables.

Using VIF to check that value is under 5 as having higher number indicates that the dataset the multi collinearity between the independent variables which must be arrested else we cannot achieve optimal machine learning model.

After standardizing the data, we must split the data into train and test sets. Train Test Split method is to split the data into train and test data

Data Inputs- Logic- Output Relationships:

After Separating the Input (feature) and output (target) data we split them into train and test division one part of the data is used to train the ML model and other part of the data to predict the output. When the train data is fed to machine learning model it generates an algorithm or simply put an equation that is applicable to all the data and when test data is fed to it implements the trained data equation to the current input values to predict the outcome. If the input has no outliers and is clean that there is no over or underfitting in the outcome.

Hardware and Software Requirements and Tools Used:

Hardware Required for Jupyter Notebook Software is as follows:

Memory and disk space required per user: 1GB RAM + 1GB of disk + .5 CPU core.

Server overhead: 2-4GB or 10% system overhead (whatever is larger), .5 CPU cores, 10GB disk space.

Port requirements: Port 8000 plus 5 unique, random ports per notebook

Libraries Used:

Pandas library: To frame raw data, visualize and perform task on it via other libraries.

Numpy library: To perform mathematical functions on the framed data numpy is used. In this project used it to location nan values and replace them with desired value also to find mean and standard values.

Matplotlib library: This library is used to visualize the data. Used to visualize univariate and bivariate analysis (pie plot, count plot and scatter plot) also to visualize outliers via box plot.

Seaborn library: heatmap to see co-relation between feature variable to arrest high collinearity.

Sklearn library: Imported this library to normalize the data, split the data into train and test data, various machine learning model and cross validation techniques.

Warnings library: To ignore filter warning shown while compiling block of codes

Pickle: To save the trained machine learning model.

Algorithm Used:

1. XGB Regression
2. Random Forest Regressor
3. K-Neighbors Regressor
4. Decision Tree Regressor
5. Linear Regression

XGB Regressor:

XGBoost stands for "Extreme Gradient Boosting" and it is an implementation of gradient boosting trees algorithm. The XGBoost is a popular supervised machine learning model with characteristics like computation speed, parallelization, and performance

```
model1=XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
             colsample_bynode=1, colsample_bytree=1, enable_categorical=False,
             gamma=0.0, gpu_id=-1, importance_type=None,
             interaction_constraints='', learning_rate=0.1, max_delta_step=0,
             max_depth=6, min_child_weight=1,
             monotone_constraints='()', n_estimators=250, n_jobs=200,
             num_parallel_tree=1, predictor='auto', random_state=0, reg_alpha=0,
             reg_lambda=1, scale_pos_weight=1, subsample=1, tree_method='exact',
             validate_parameters=1, verbosity=None)
model1.fit(X_train,Y_train)
```

```
XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
             colsample_bynode=1, colsample_bytree=1, enable_categorical=False,
             gamma=0.0, gpu_id=-1, importance_type=None,
             interaction_constraints='', learning_rate=0.1, max_delta_step=0,
             max_depth=6, min_child_weight=1, missing=nan,
             monotone_constraints='()', n_estimators=250, n_jobs=200,
             num_parallel_tree=1, predictor='auto', random_state=0, reg_alpha=0,
             reg_lambda=1, scale_pos_weight=1, subsample=1, tree_method='exact',
             validate_parameters=1, verbosity=None)
```

```
scores1 = cross_val_score(model1, X_test, Y_test, scoring='r2', cv=10)
print('Mean R2 Score for XGB Regression :',mean(scores1),'\nStandard Deviation is :
      ,std(scores1))
```

```
Mean R2 Score for XGB Regression : 0.9325845199958562
Standard Deviation is :  0.031117270448561957
```

Random Forest Regressor:

A random forest is a meta estimator that fits a number of classifying decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting

```
model2=RandomForestRegressor(max_depth=9, n_estimators=250)
model2.fit(X_train,Y_train)
```

```
RandomForestRegressor(max_depth=9, n_estimators=250)
```

```
scores2 = cross_val_score(model2, X_test, Y_test, scoring='r2', cv=10)
print('Mean R2 Score for Random Forest Regressor :',mean(scores2),
      '\nStandard Deviation is : ',std(scores2))
```

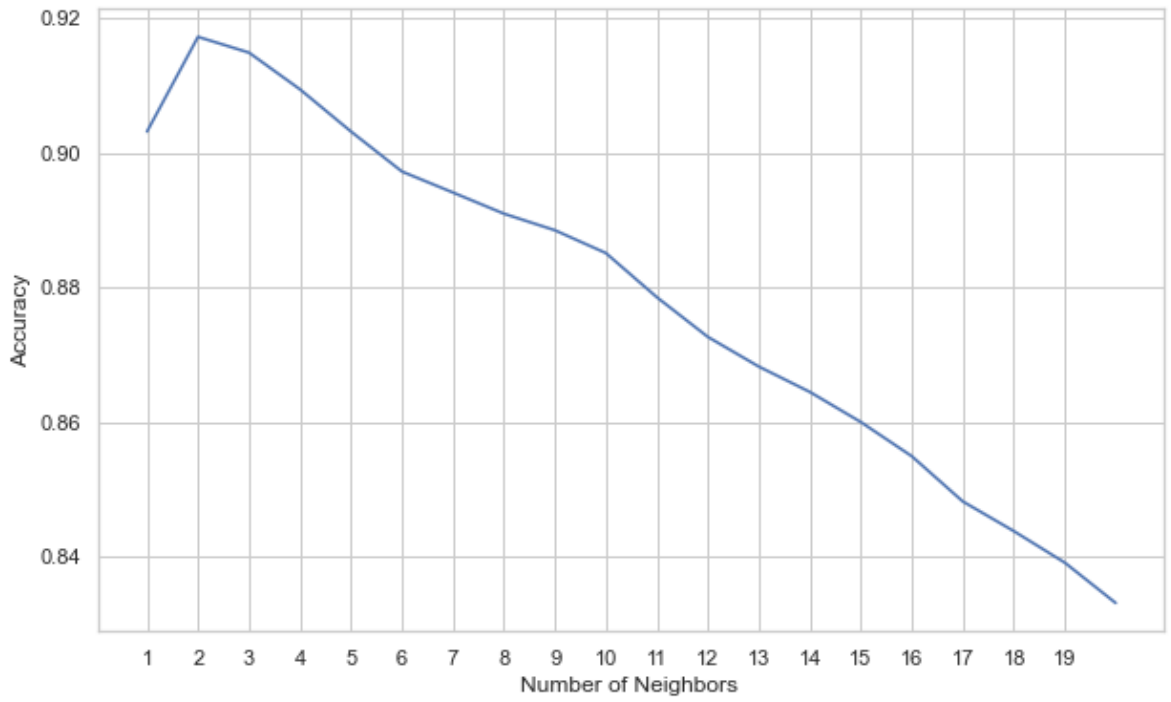
```
Mean R2 Score for Random Forest Regressor : 0.9038202795174526
Standard Deviation is :  0.04932362156980524
```

K-Neighbors Regressor:

KNN regression is a non-parametric method that, in an intuitive manner, approximates the association between independent variables and the continuous outcome by averaging the observations in the same neighbourhood.

```
knn_p=KNeighborsRegressor()
mean_acc = np.zeros(20)
for i in range(1,21):
    #Train Model and Predict
    knn = KNeighborsRegressor(n_neighbors = i).fit(X_train,Y_train)
    yhat= knn.predict(X_test)
    mean_acc[i-1] = metrics.r2_score(Y_test, yhat)

loc = np.arange(1,20,step=1.0)
plt.figure(figsize = (10, 6))
plt.plot(range(1,21), mean_acc)
plt.xticks(loc)
plt.xlabel('Number of Neighbors ')
plt.ylabel('Accuracy')
plt.show()
```




```
model3=KNeighborsRegressor(leaf_size=40, n_neighbors=2, weights='distance')
model3.fit(X_train,Y_train)
```

```
KNeighborsRegressor(leaf_size=40, n_neighbors=2, weights='distance')
```

```
scores3 = cross_val_score(model3, X_test, Y_test, scoring='r2', cv=10)
print('Mean R2 Score for KNN Regressor :',mean(scores3),
      '\nStandard Deviation is : ',std(scores3))
```

```
Mean R2 Score for KNN Regressor : 0.7703099893121697
Standard Deviation is :  0.10631823370086936
```

Decision Tree Regressor:

A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute, each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes).

```
model4=DecisionTreeRegressor(max_depth=7, max_features='auto', max_leaf_nodes=30,
                      min_samples_leaf=5, min_weight_fraction_leaf=0.1)
model4.fit(X_train,Y_train)
```

```
DecisionTreeRegressor(max_depth=7, max_features='auto', max_leaf_nodes=30,
                      min_samples_leaf=5, min_weight_fraction_leaf=0.1)
```

```
scores4 = cross_val_score(model4, X_test, Y_test, scoring='r2', cv=10)
print('Mean R2 Score for Decision Tree Regressor :',mean(scores4),
      '\nStandard Deviation is : ',std(scores4))
```

```
Mean R2 Score for Decision Tree Regressor : 0.8311149705930921
Standard Deviation is :  0.04887422914218663
```

Linear Regression:

linear regression is a linear approach for modelling the relationship between a scalar response and one or more explanatory variables (also known as dependent and independent variables). The case of one explanatory variable is called simple linear regression; for more than one, the process is called multiple linear regression. This term is distinct from multivariate linear regression, where multiple correlated dependent variables are predicted, rather than a single scalar variable

```
model5=LinearRegression()
model5.fit(X_train,Y_train)
```

```
LinearRegression()
```

```
scores5 = cross_val_score(model5, X_test, Y_test, scoring='r2', cv=10)
print('Mean R2 Score for Linear Regression :',mean(scores5),
      '\nStandard Deviation is : ',std(scores5))
```

```
Mean R2 Score for Linear Regression : 0.4577348201076344
Standard Deviation is :  0.04962366372171507
```

Metrics Used in the project:

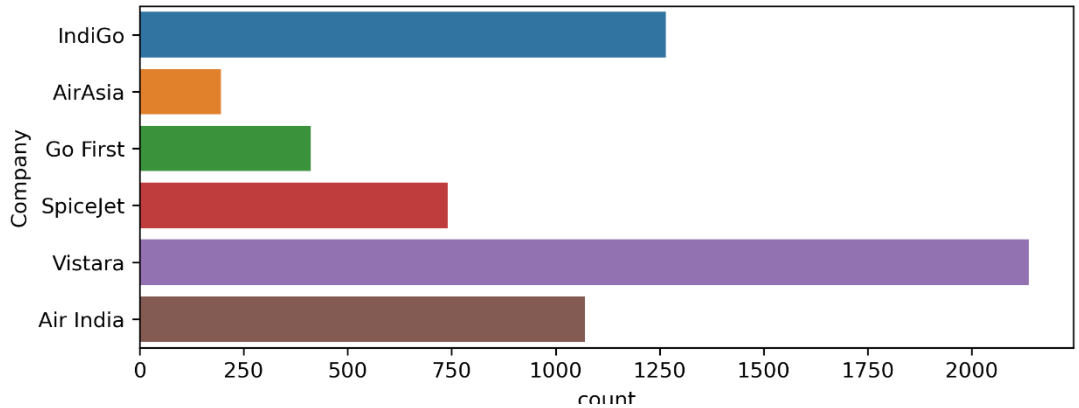
cross_val_score: Used cross_val_score to evaluate the model and observe (r2 value) it perform for particular number of folds to determine which model performs better

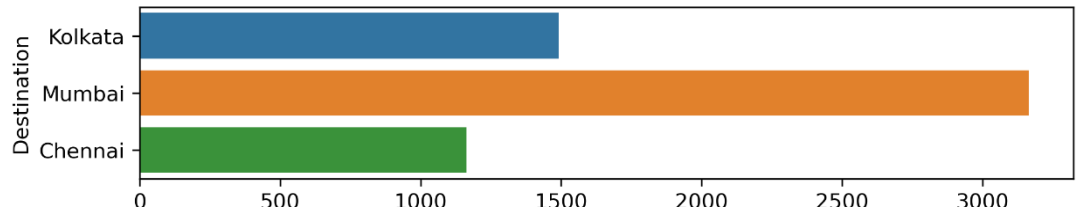
Kfold: to split and randomize the data.

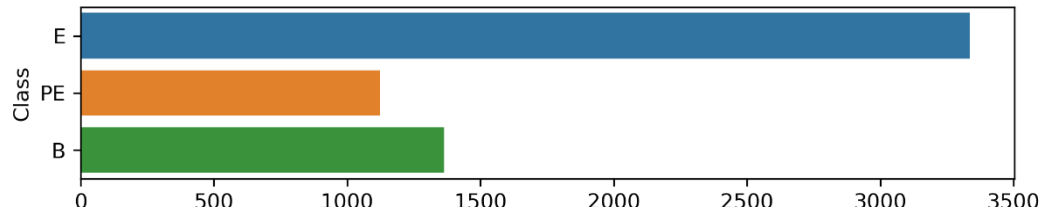
mean: To get the mean value of cross validation score and determine the model accuracy.

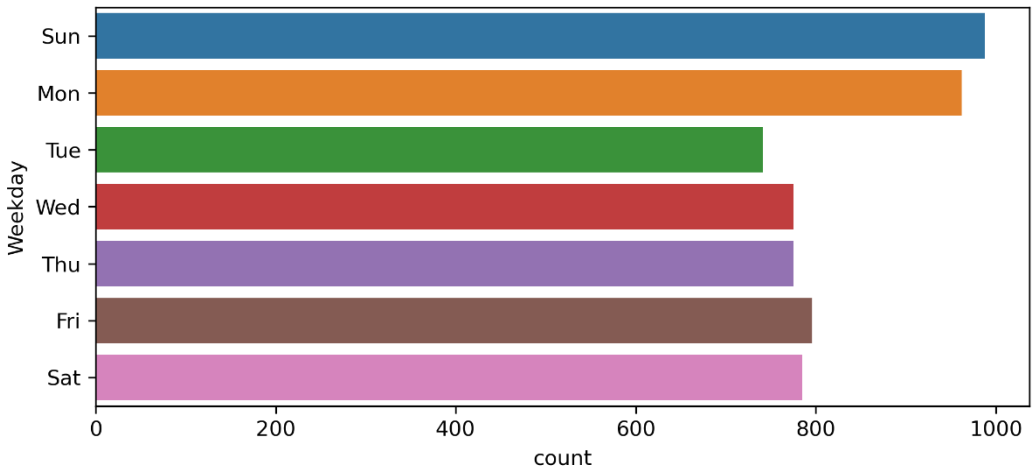
Standard deviation: To determine the average deviation for all the fold observed in cross validation scores

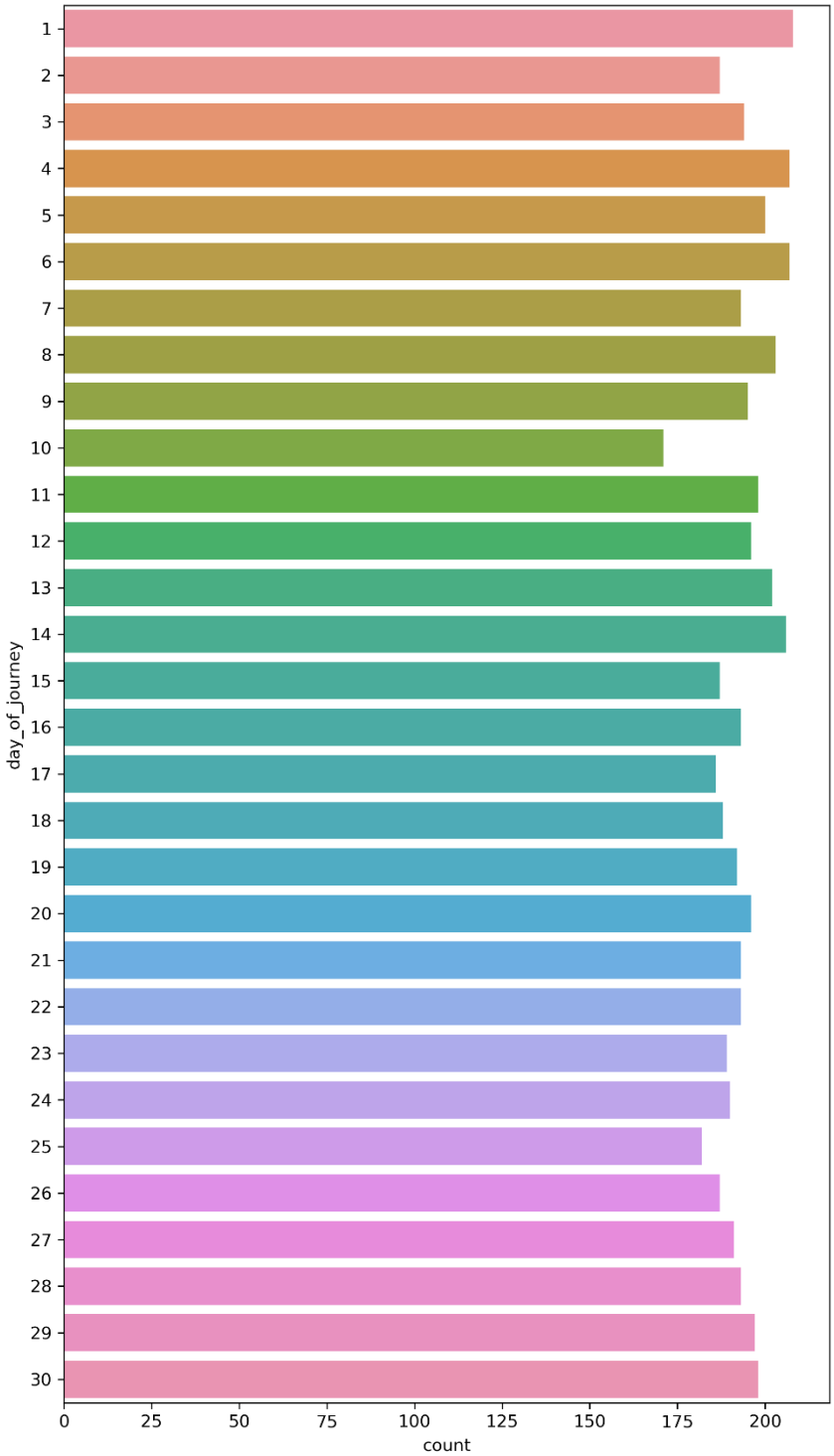
Visualization:

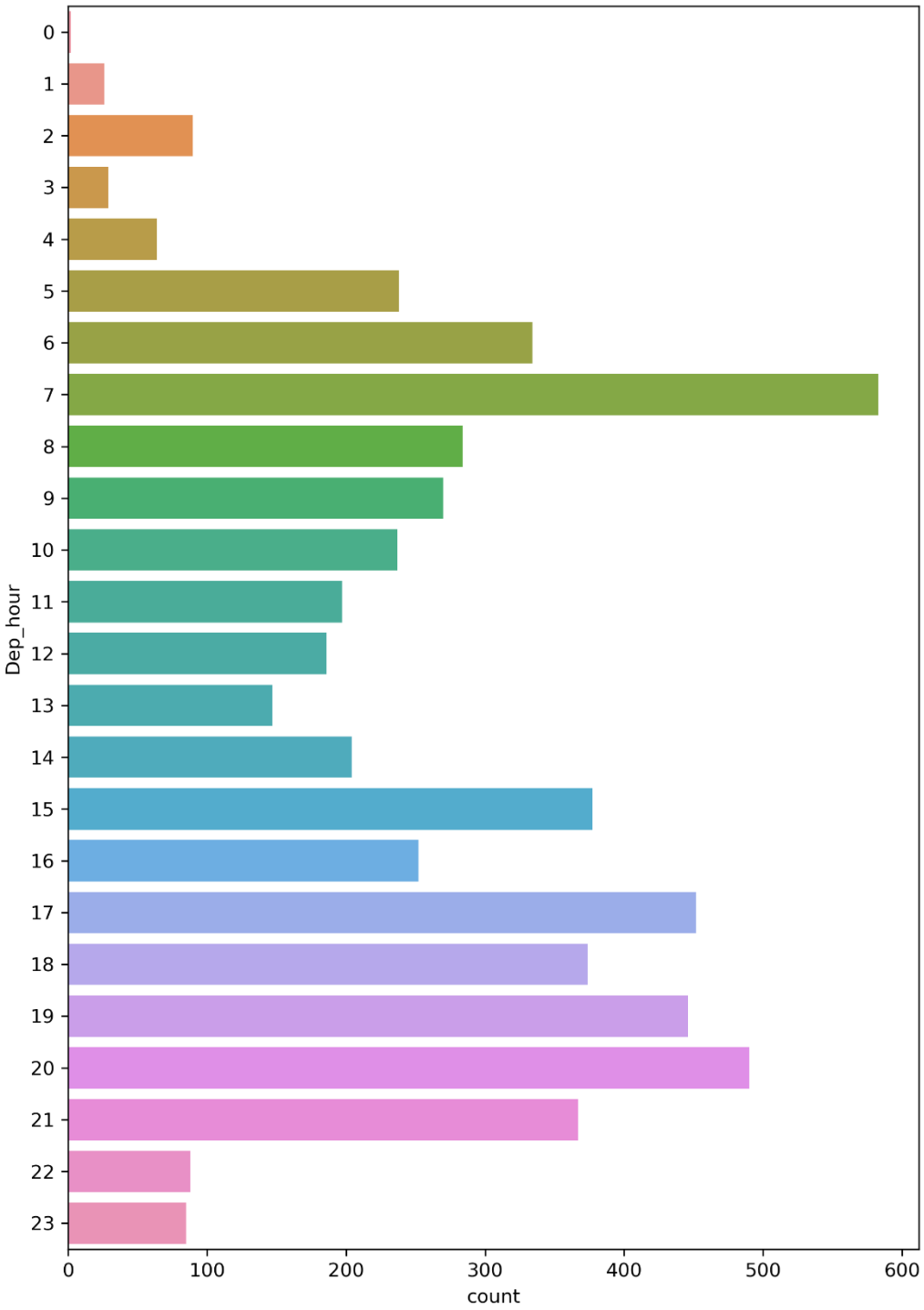


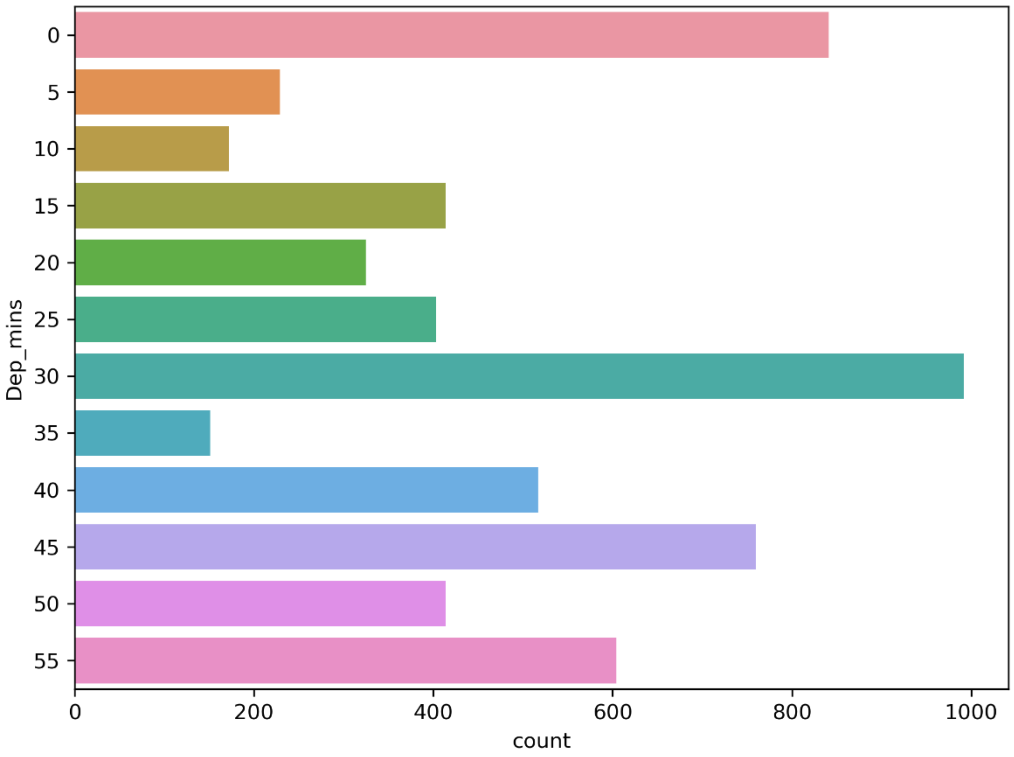


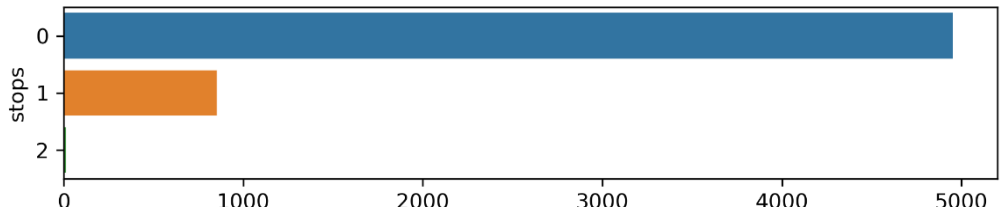


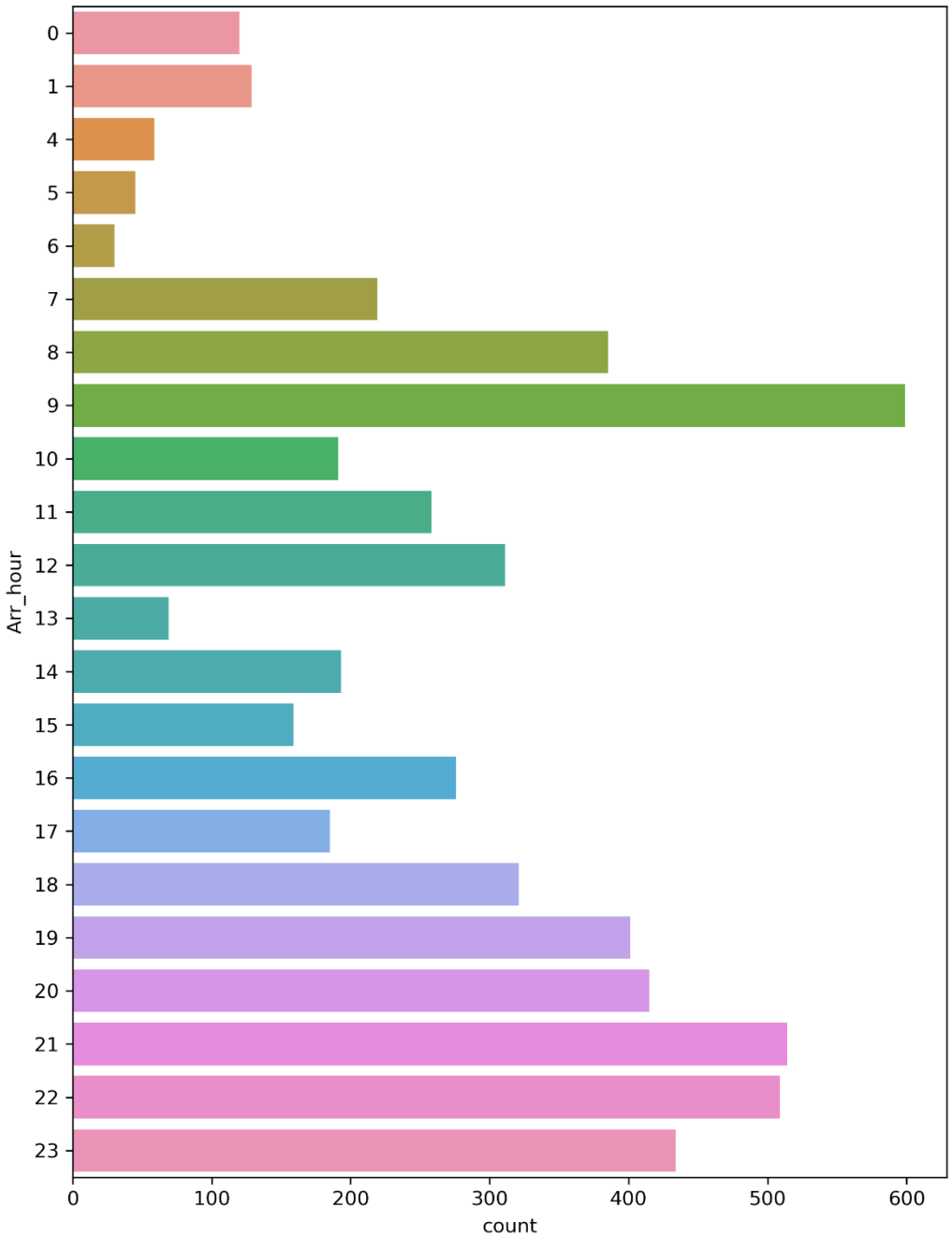


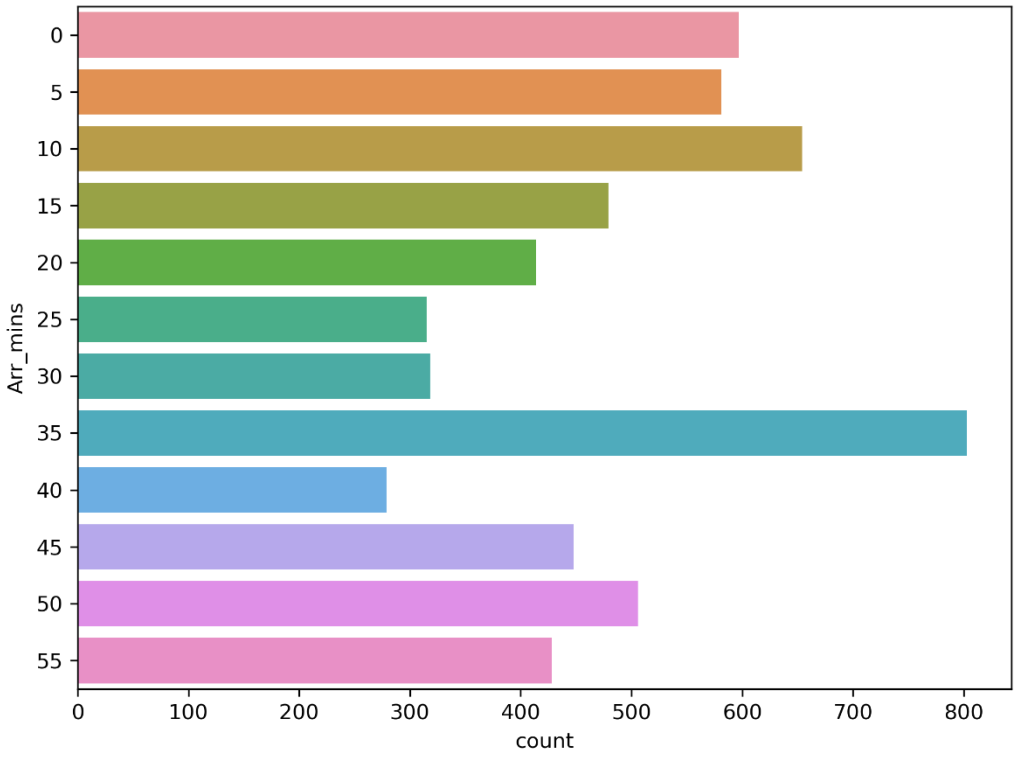

day_of_journey
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
0
25
50
75
100
125
150
175
200
count

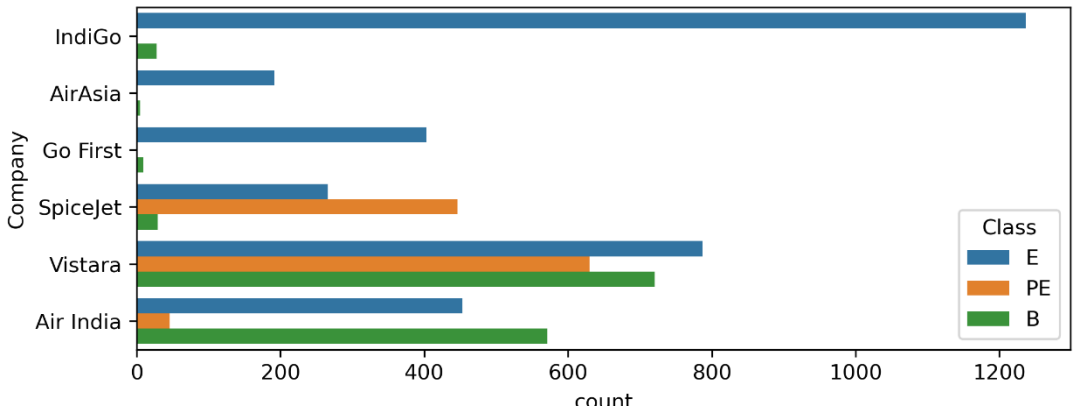

Dep_hour
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
0
100
200
300
400
500
600
count

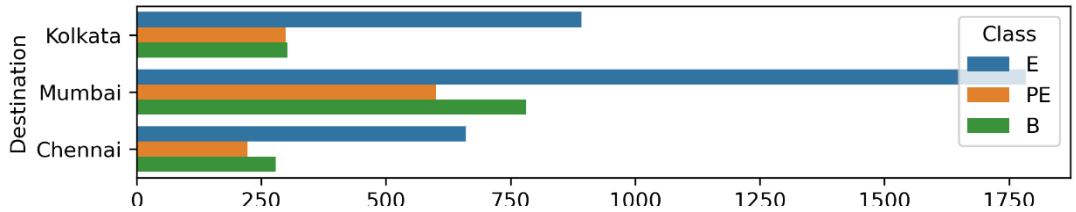

Dep_mins
0
5
10
15
20
25
30
35
40
45
50
55
0
200
400
600
800
1000
count

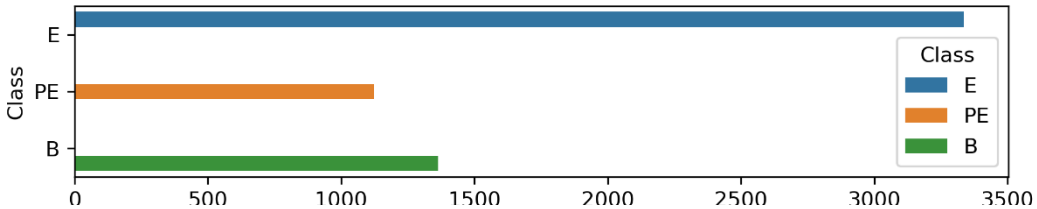

stops
0
1
2
0
1000
2000
3000
4000
5000

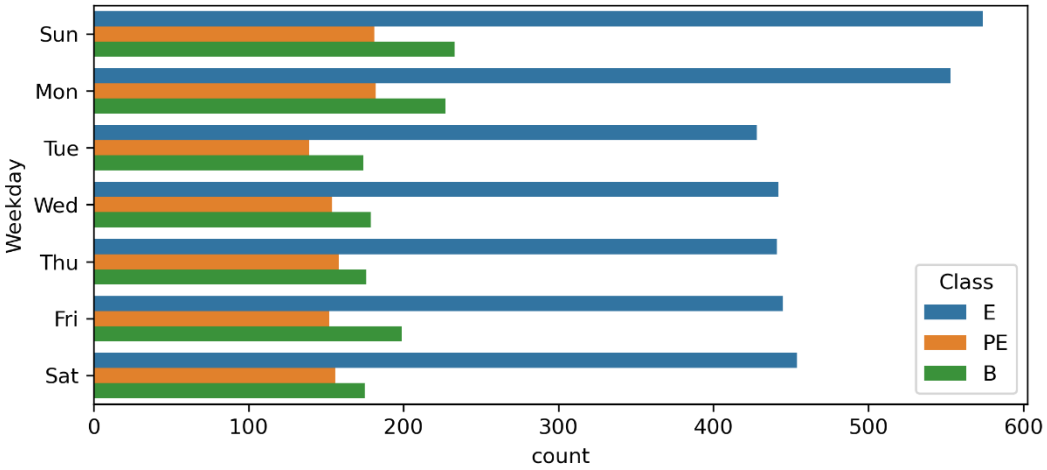

Arr_hour
0
1
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
0
100
200
300
400
500
600
count

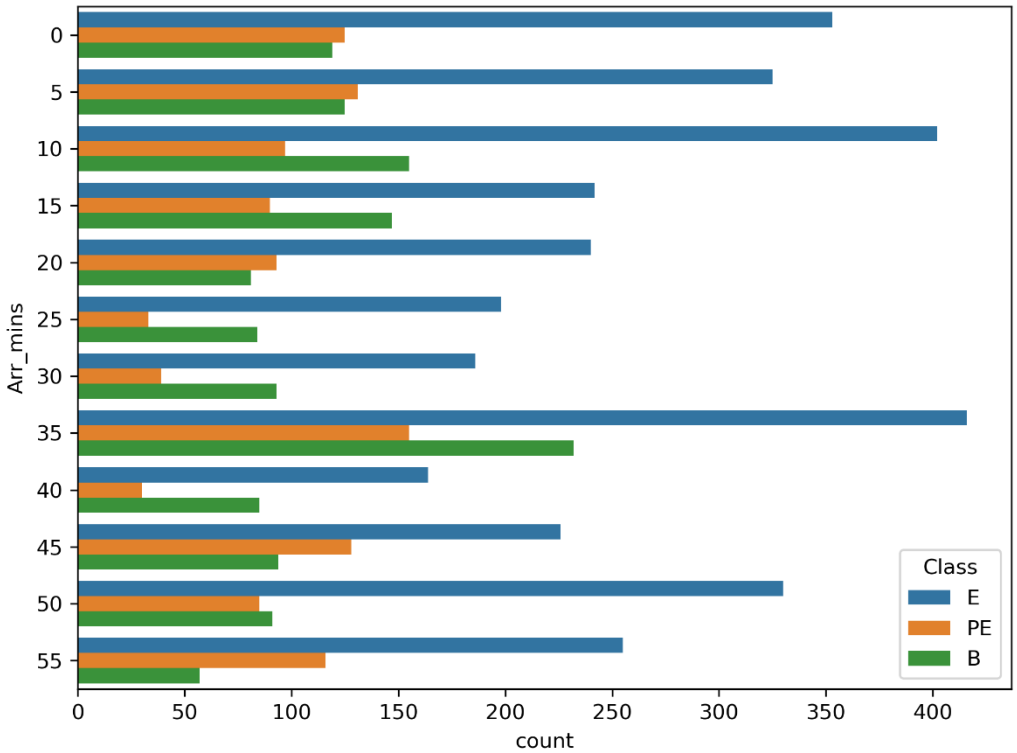

Arr_mins
0
5
10
15
20
25
30
35
40
45
50
55
0
100
200
300
400
500
600
700
800
count

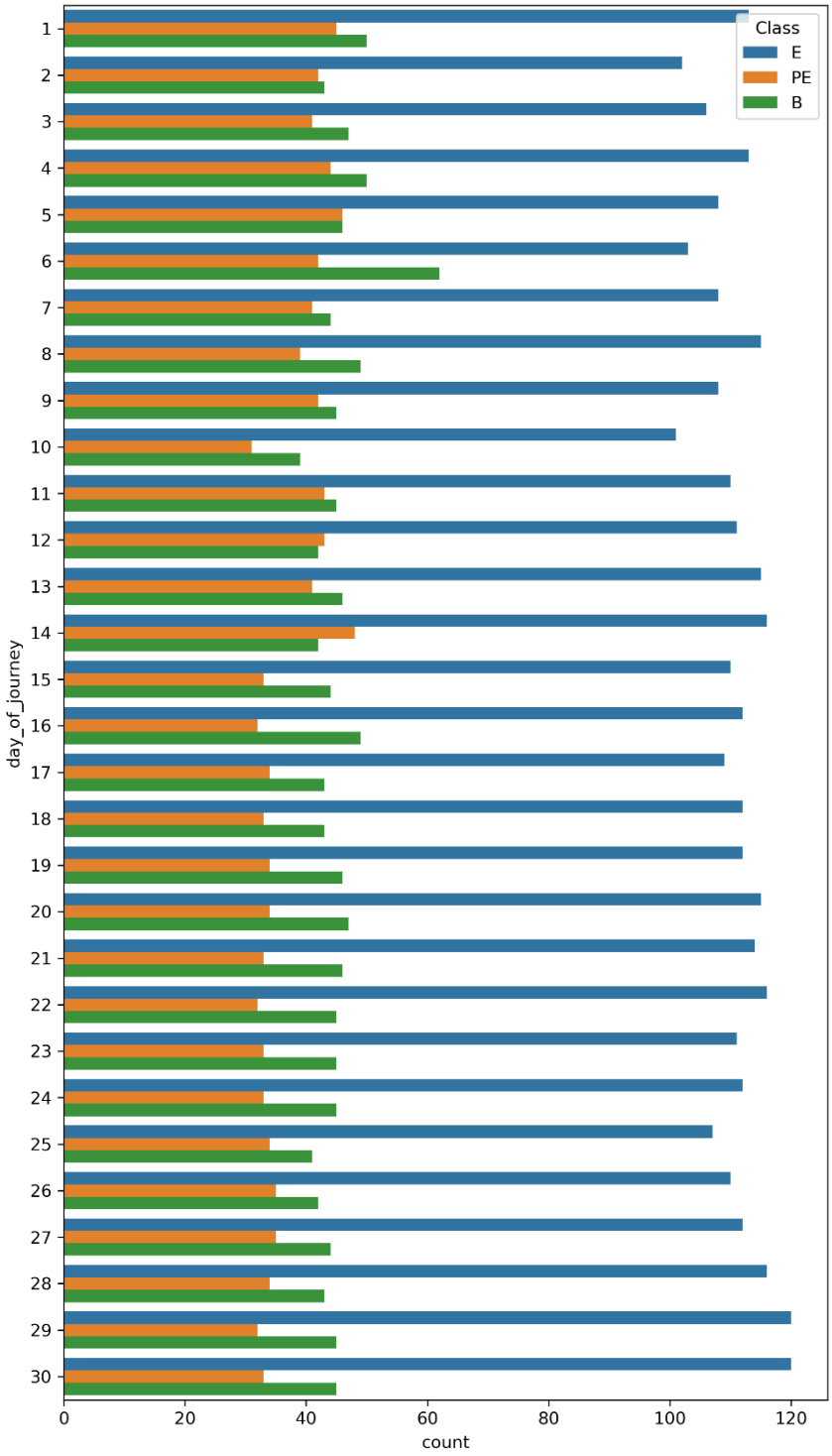

Company
IndiGo
AirAsia
Go First
SpiceJet
Vistara
Air India
0
200
400
600
800
1000
1200
count
Class
E
PE
B

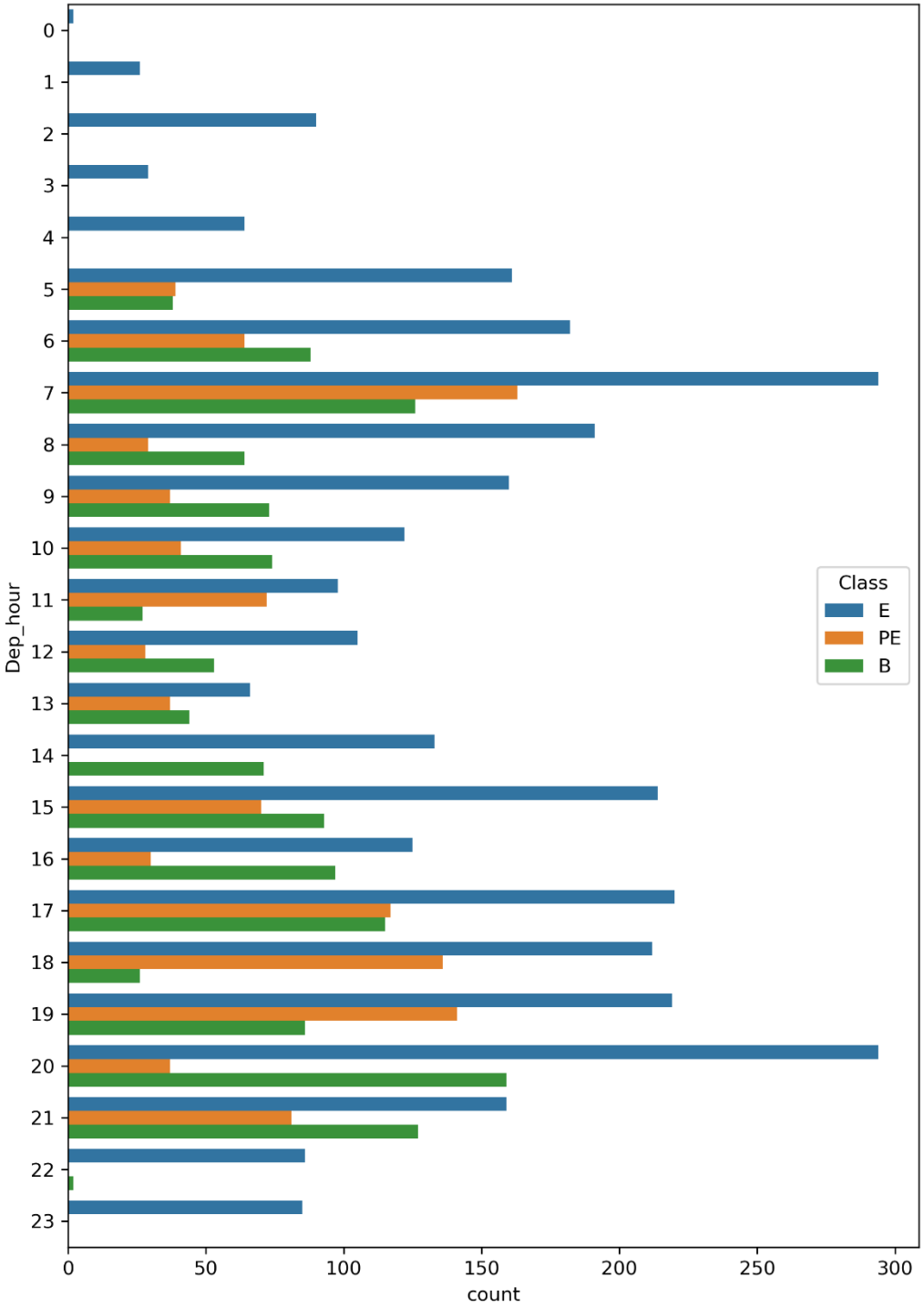

Destination
Kolkata
Mumbai
Chennai
0
250
500
750
1000
1250
1500
1750
Class
E
PE
B

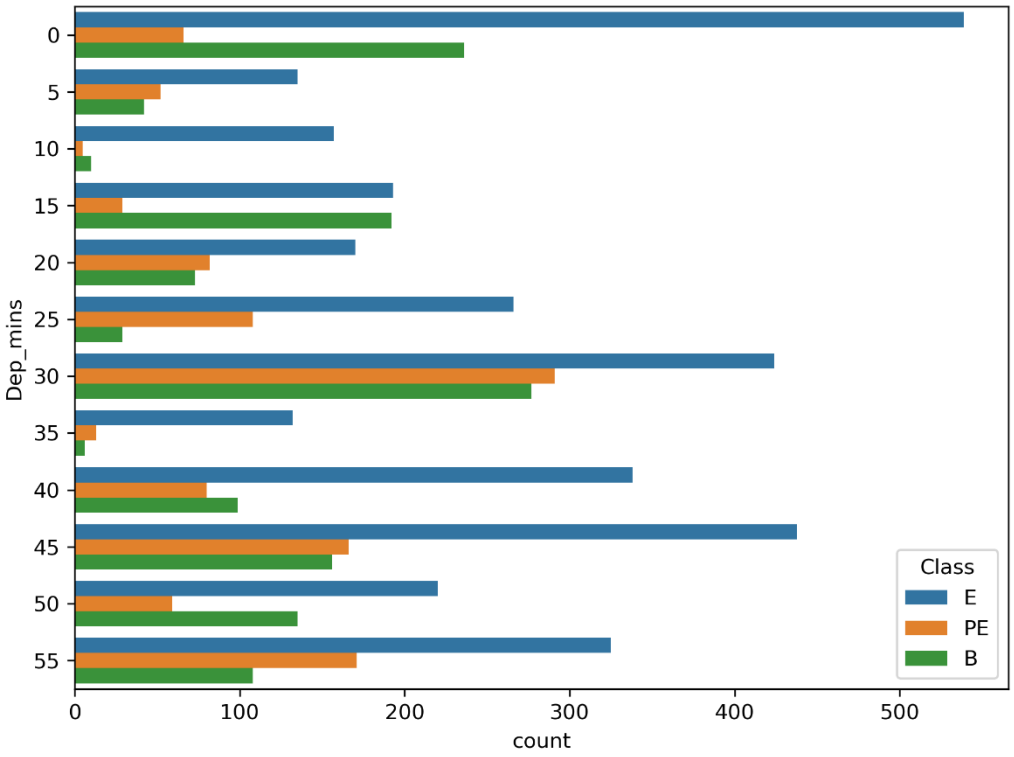

Class
E
PE
B
0
500
1000
1500
2000
2500
3000
3500
Class
E
PE
B

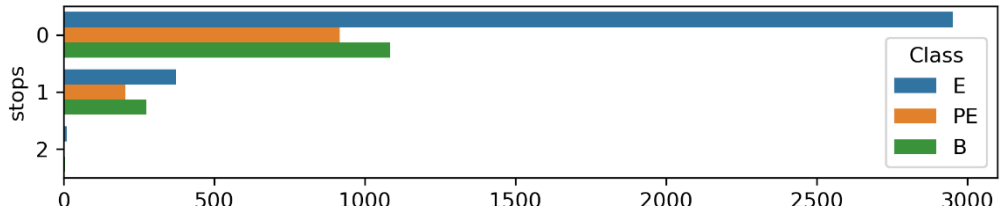

Weekday
Sun
Mon
Tue
Wed
Thu
Fri
Sat
0
100
200
300
400
500
600
count
Class
E
PE
B

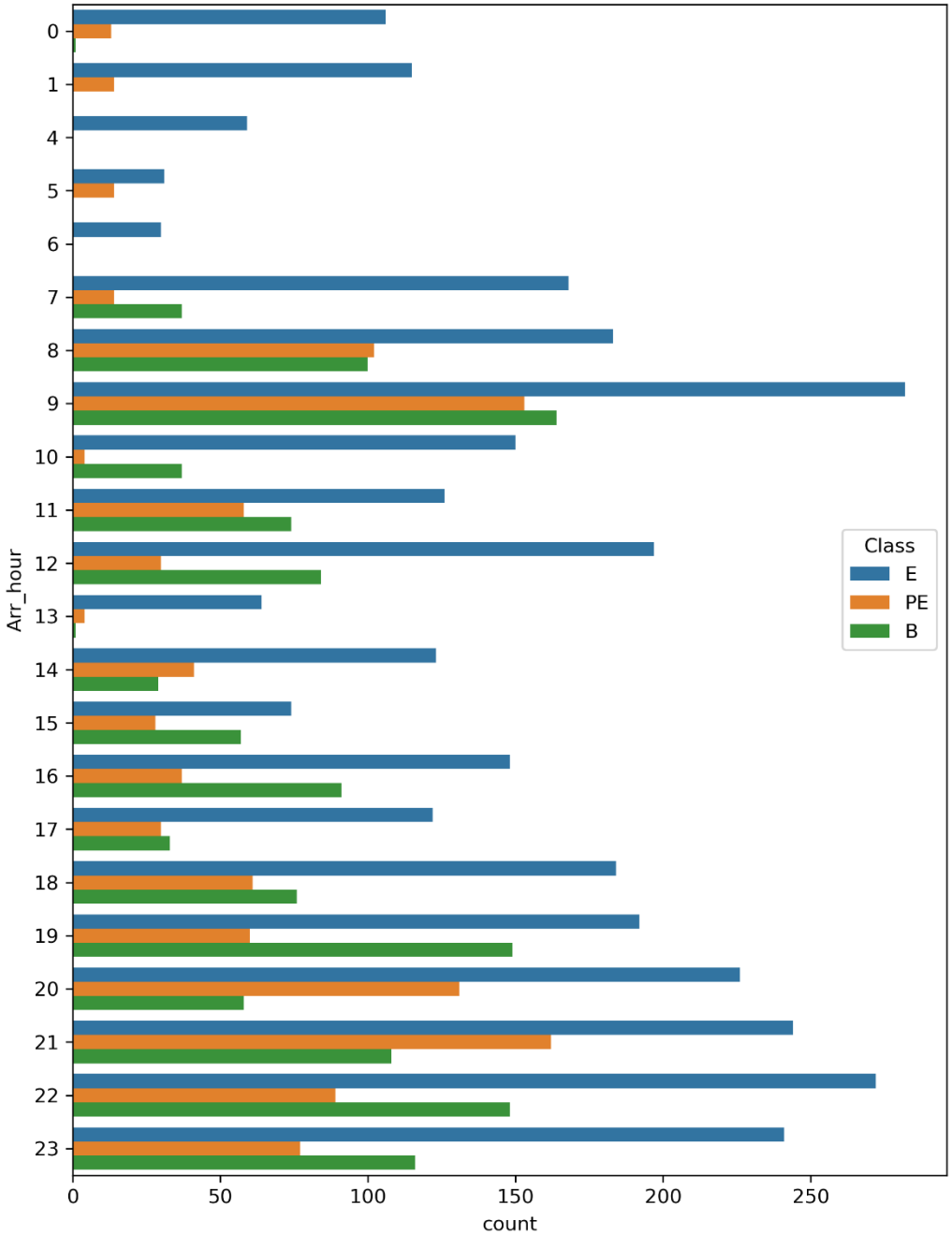

Arr_mins
0
5
10
15
20
25
30
35
40
45
50
55
0
50
100
150
200
250
300
350
400
count
Class
E
PE
B

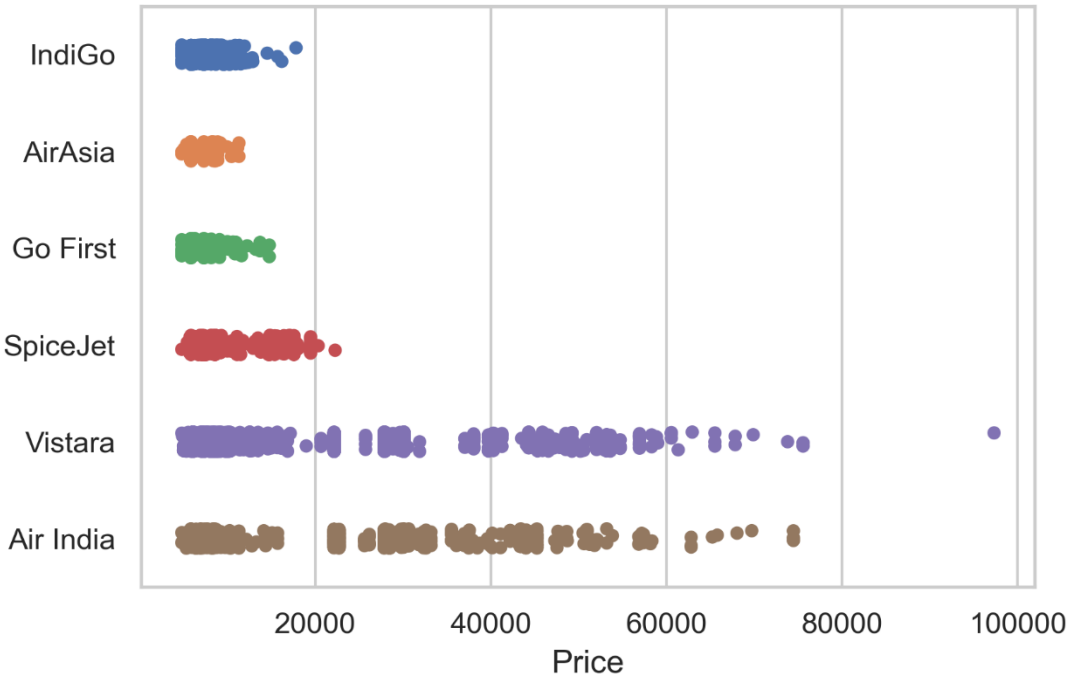

Class
E
PE
B
day_of_journey
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
0
20
40
60
80
100
120
count

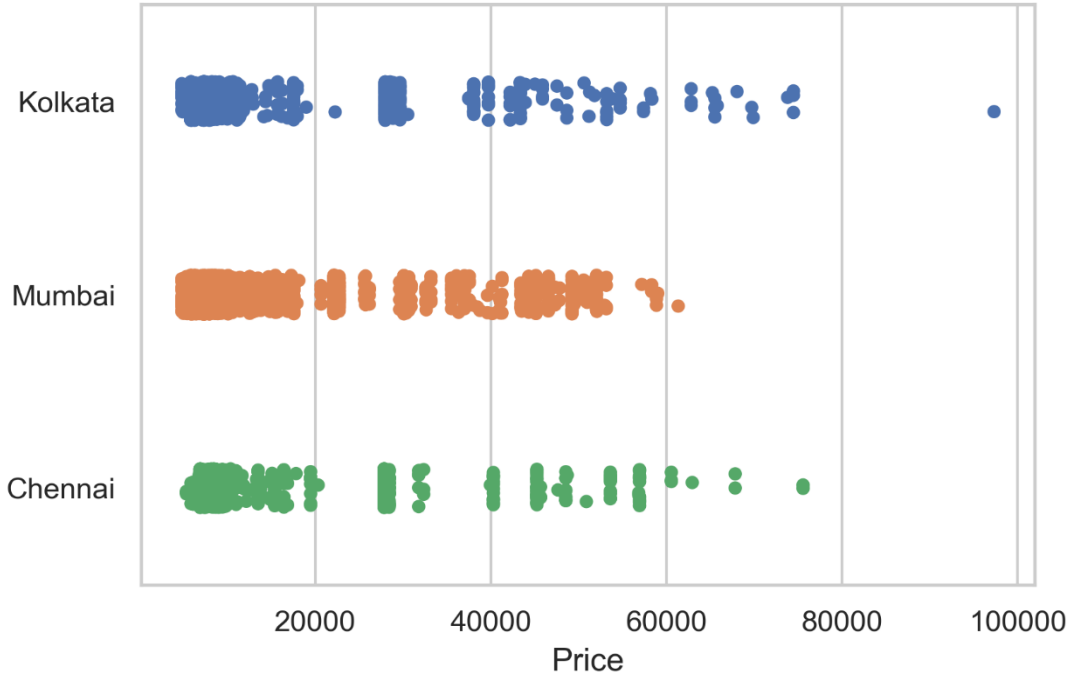

Dep_hour
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
0
50
100
150
200
250
300
count
Class
E
PE
B

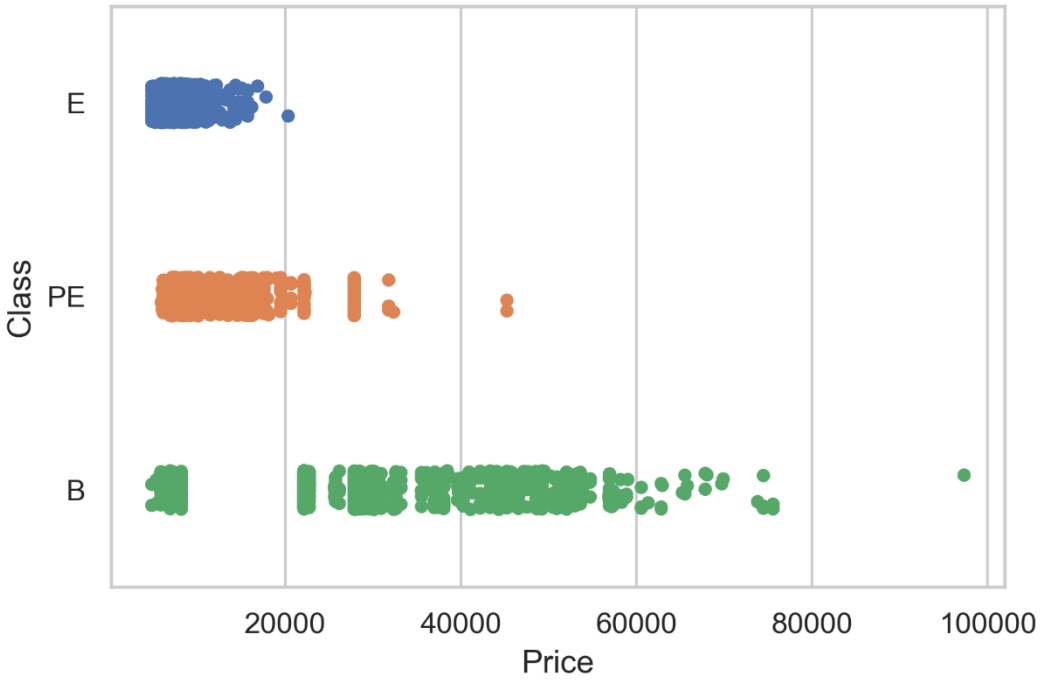

Dep_mins
0
5
10
15
20
25
30
35
40
45
50
55
0
100
200
300
400
500
count
Class
E
PE
B

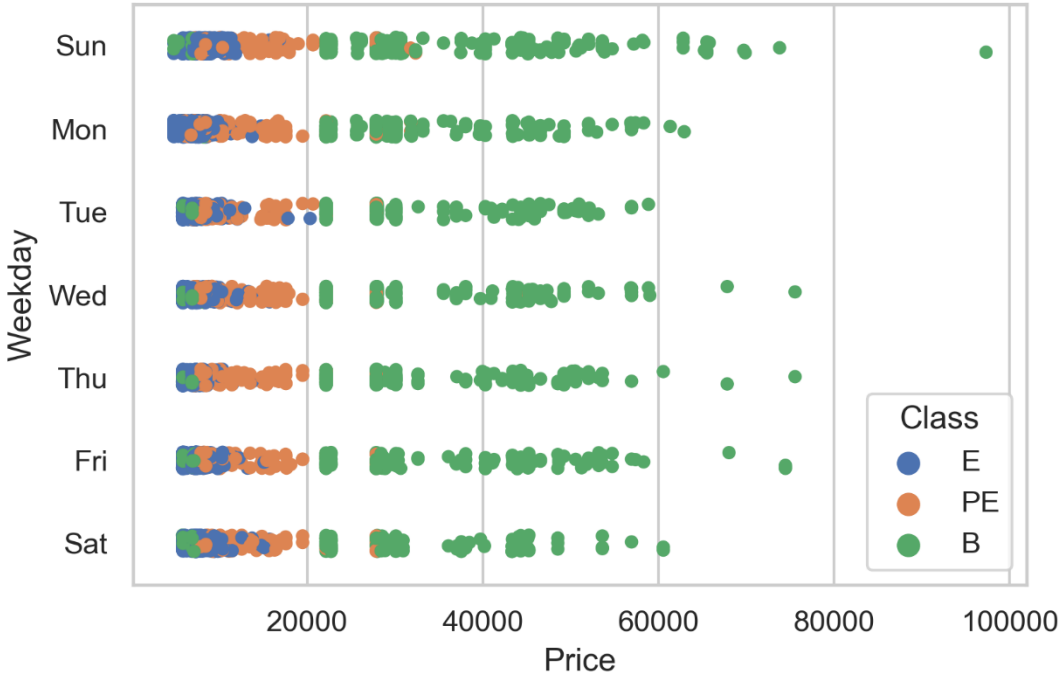

stops
0
1
2
0
500
1000
1500
2000
2500
3000
Class
E
PE
B

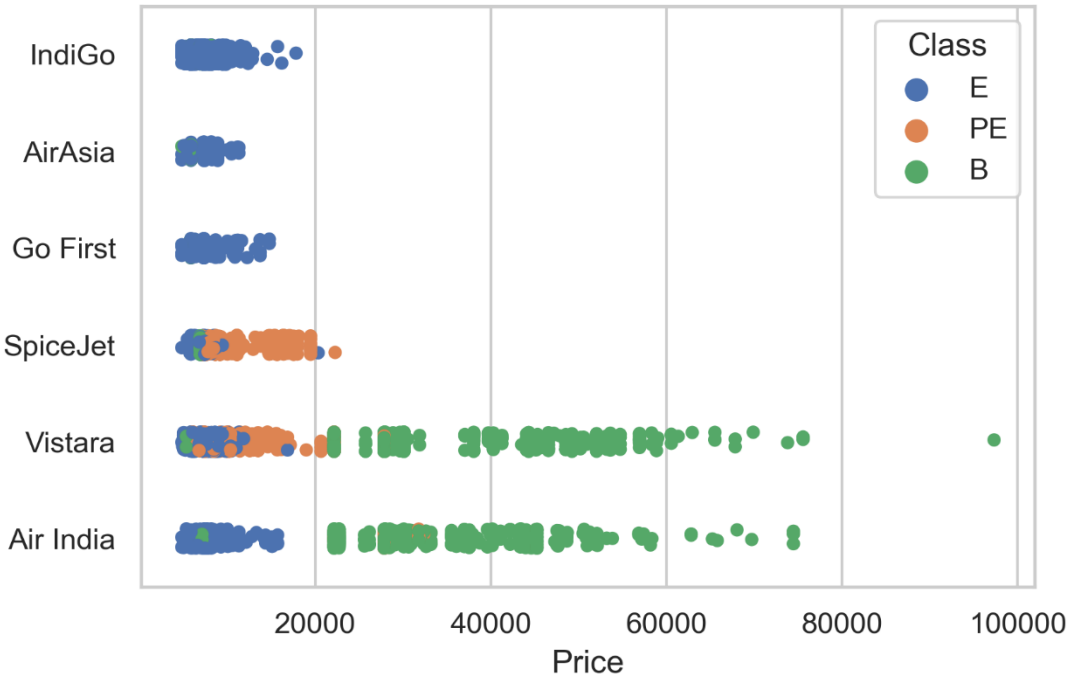

Arr_hour
0
1
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
0
50
100
150
200
250
count
Class
E
PE
B

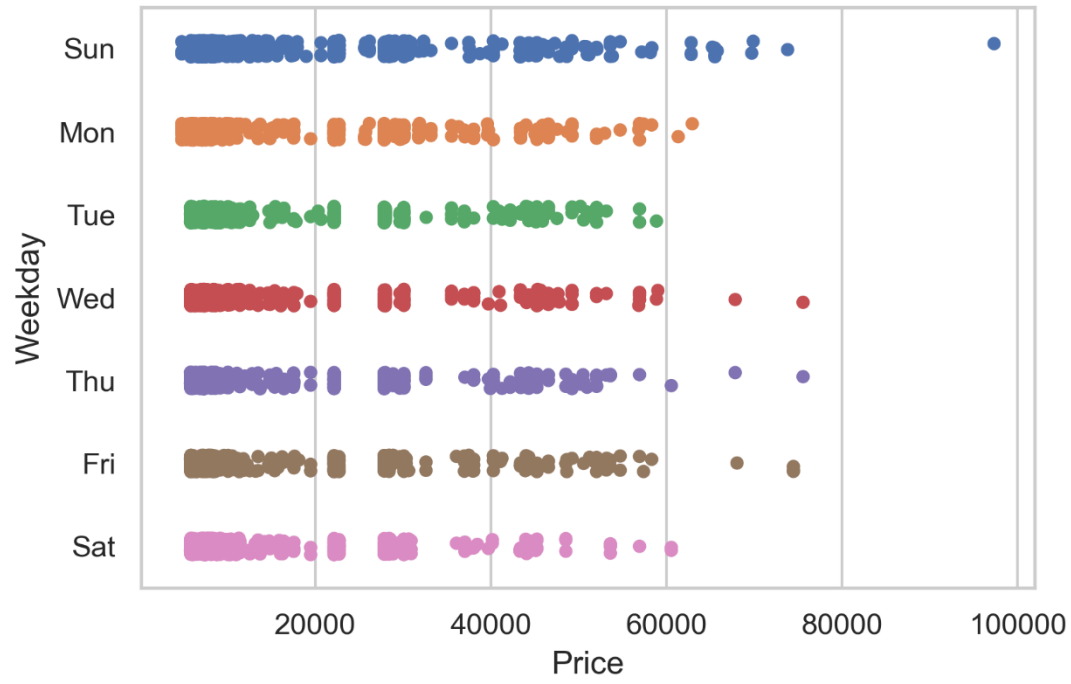

IndiGo
AirAsia
Go First
SpiceJet
Vistara
Air India
20000
40000
60000
80000
100000
Price

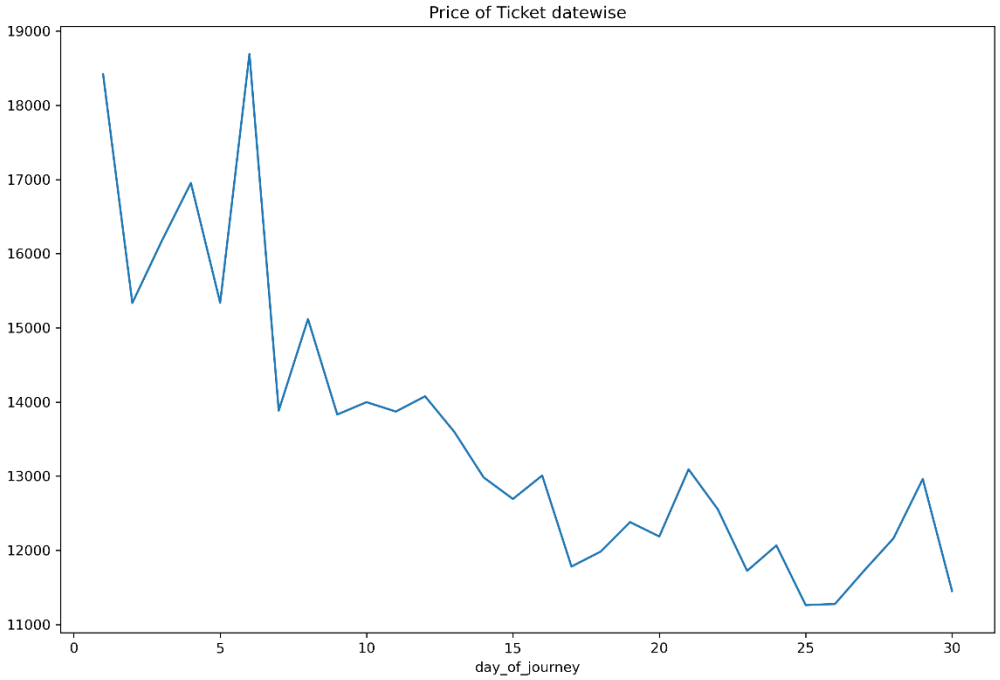

Kolkata
Mumbai
Chennai
20000
40000
60000
80000
100000
Price

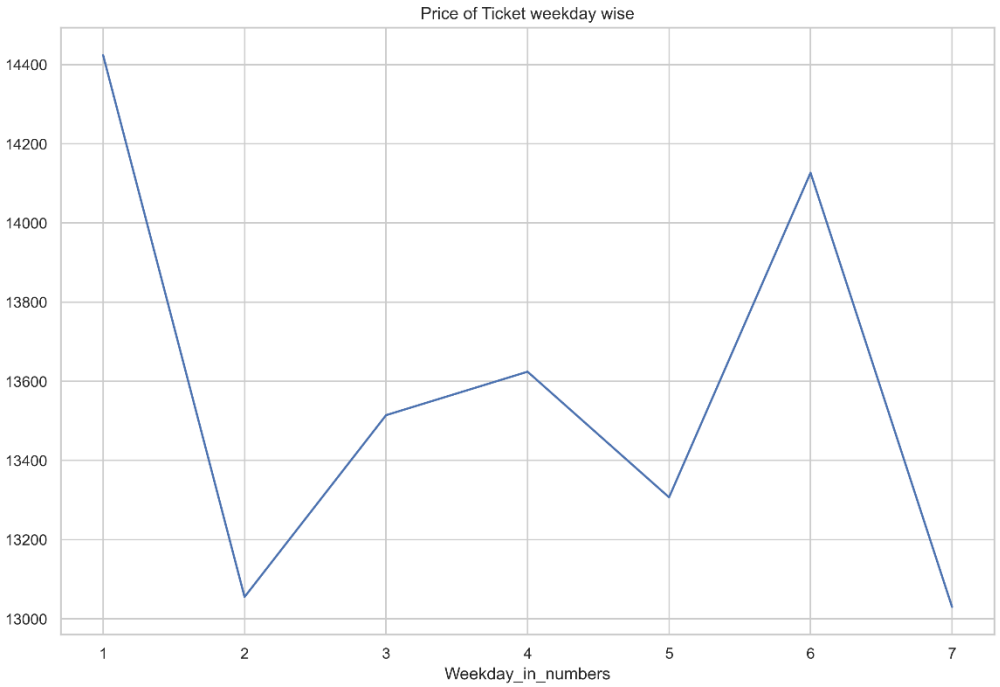

Class
E
PE
B
20000
40000
60000
80000
100000
Price

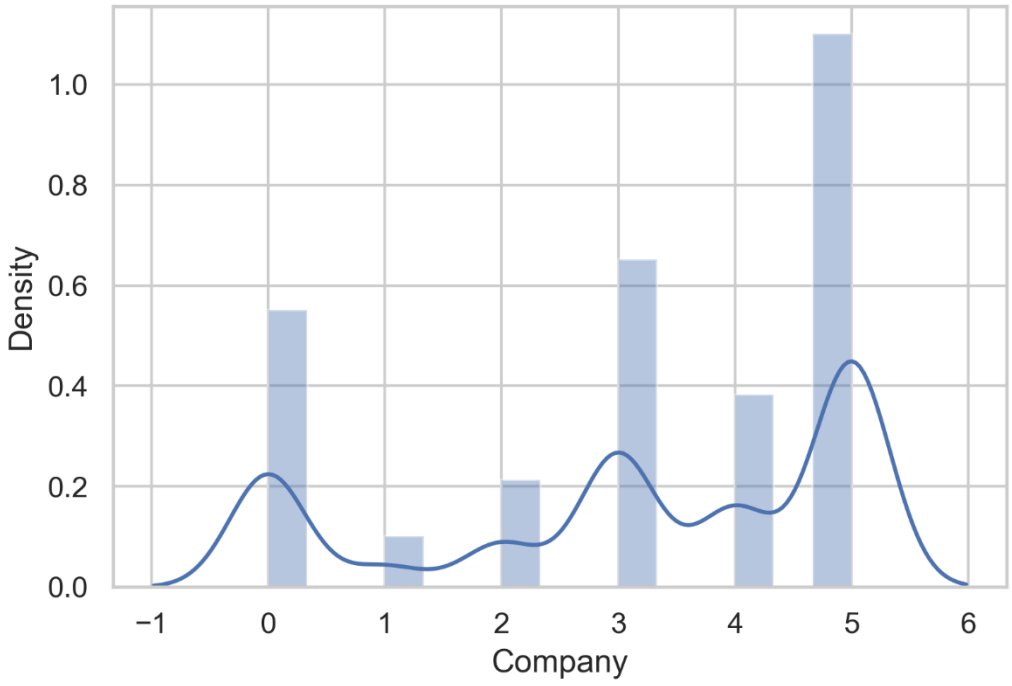

Weekday
Sun
Mon
Tue
Wed
Thu
Fri
Sat
20000
40000
60000
80000
100000
Price
Class
E
PE
B

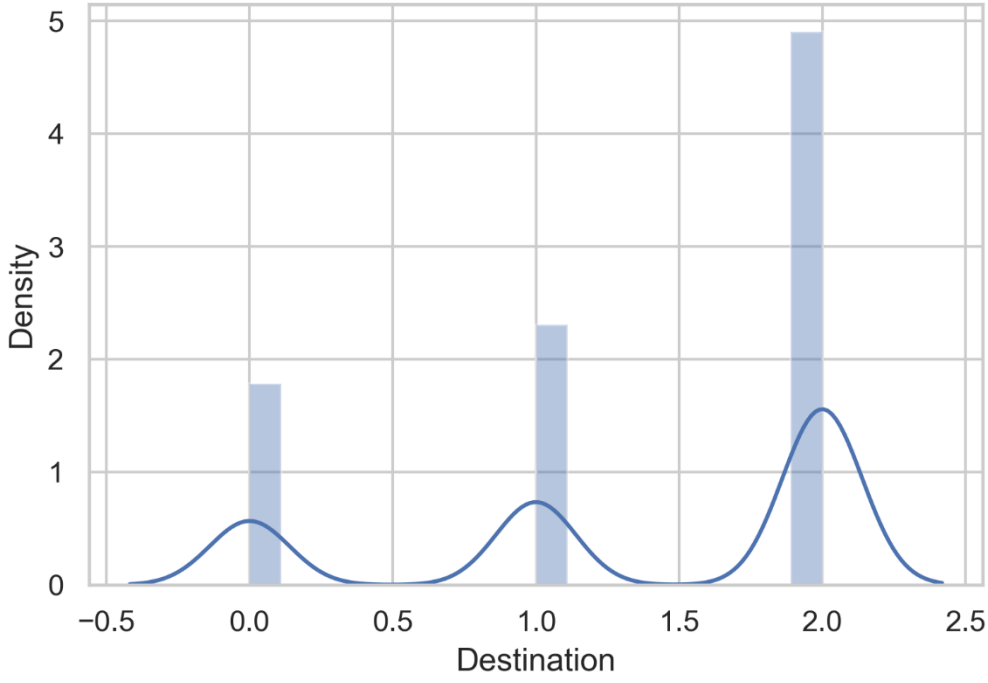

IndiGo
AirAsia
Go First
SpiceJet
Vistara
Air India
Class
E
PE
B
20000
40000
60000
80000
100000
Price

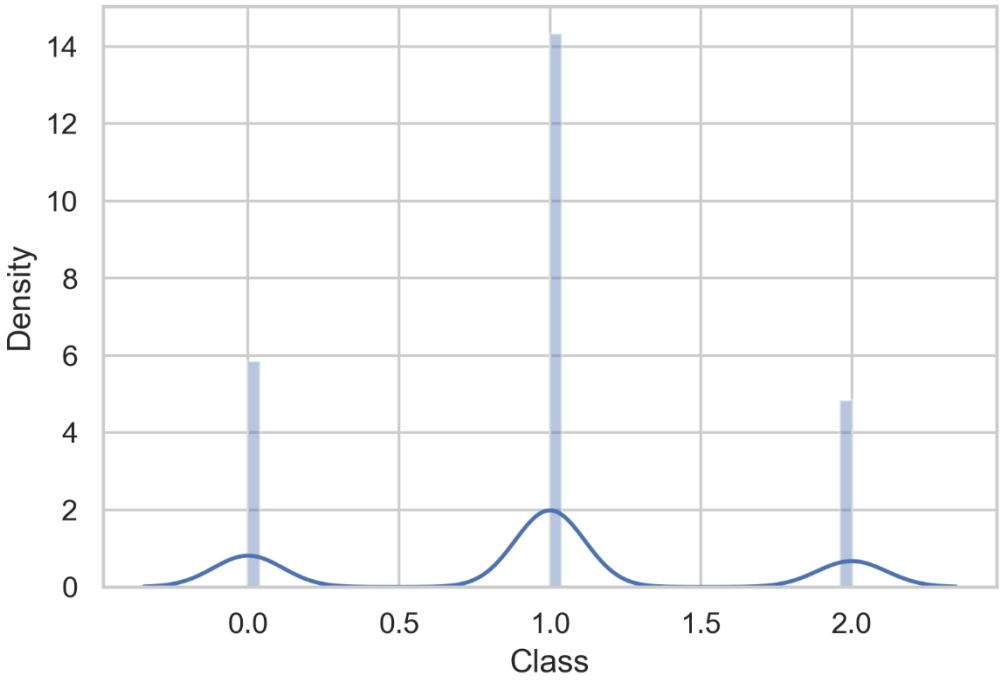

Sun
Mon
Tue
Wed
Thu
Fri
Sat
Weekday
20000
40000
60000
80000
100000
Price

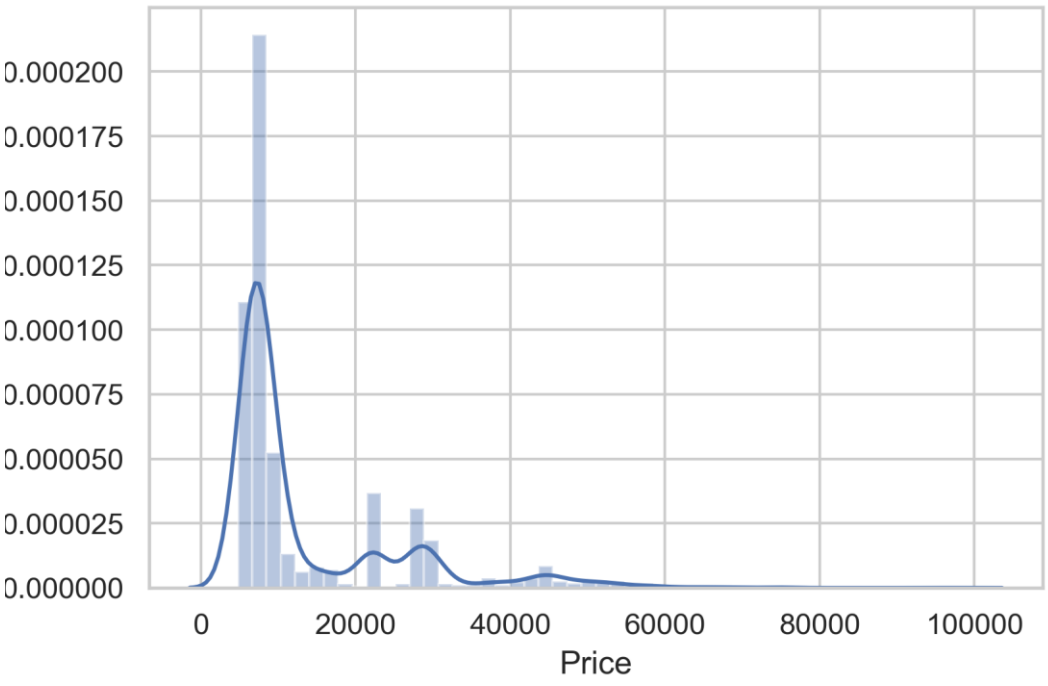

Price of Ticket datewise
19000
18000
17000
16000
15000
14000
13000
12000
11000
0
5
10
15
20
25
30
day_of_journey

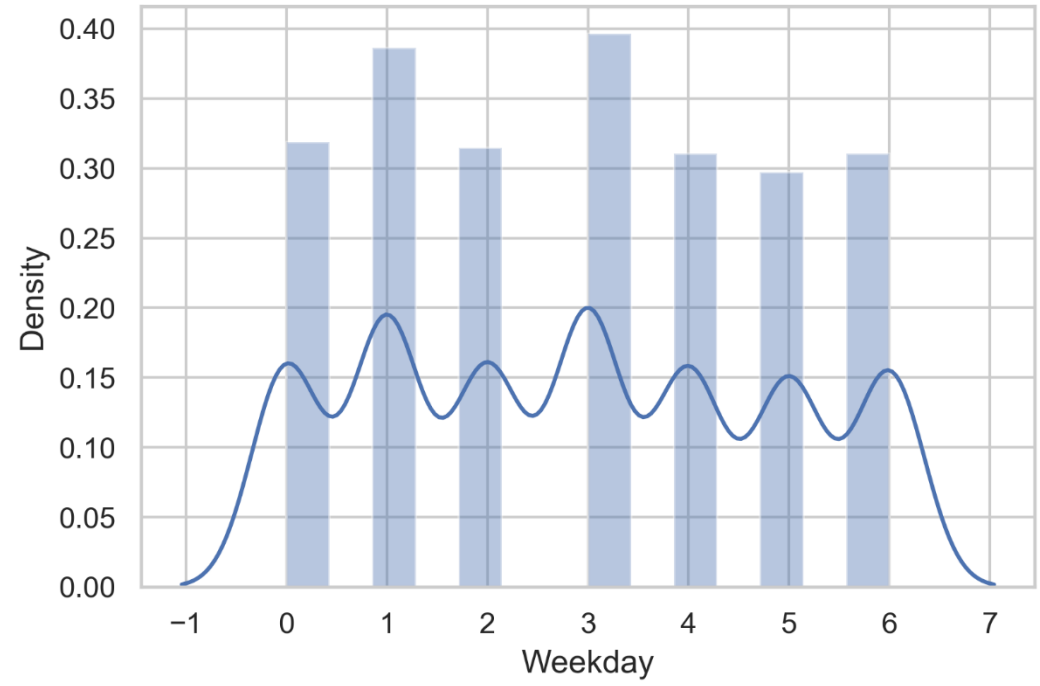

Price of Ticket weekday wise
14400
14200
14000
13800
13600
13400
13200
13000
1
2
3
4
5
6
7
Weekday_in_numbers

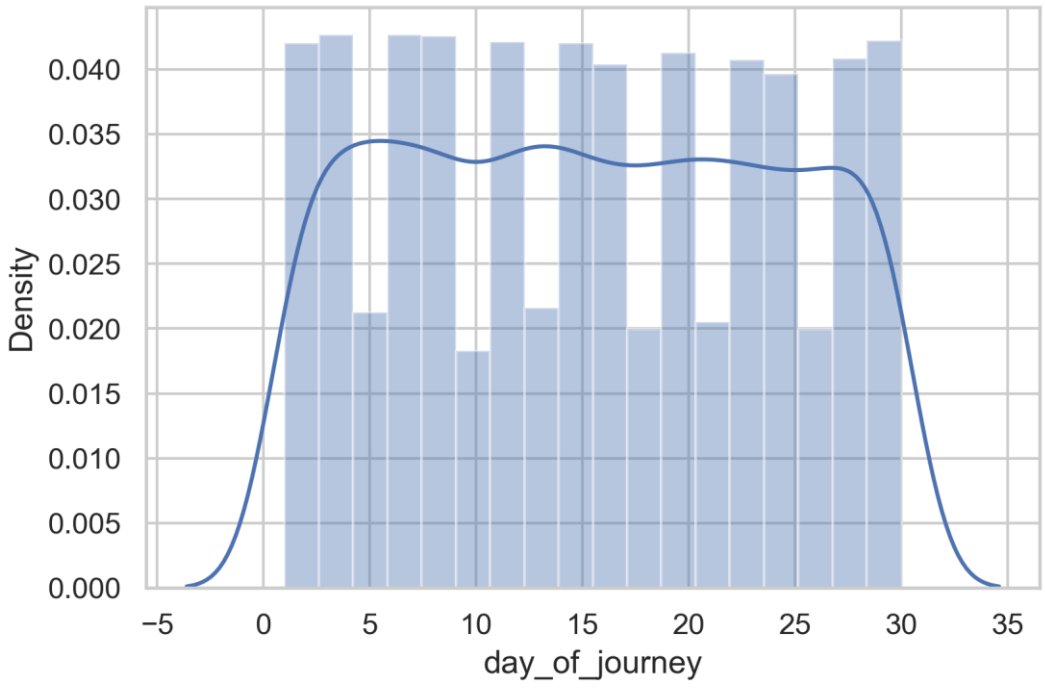

Density
1.0
0.8
0.6
0.4
0.2
0.0
−1
0
1
2
3
4
5
6
Company

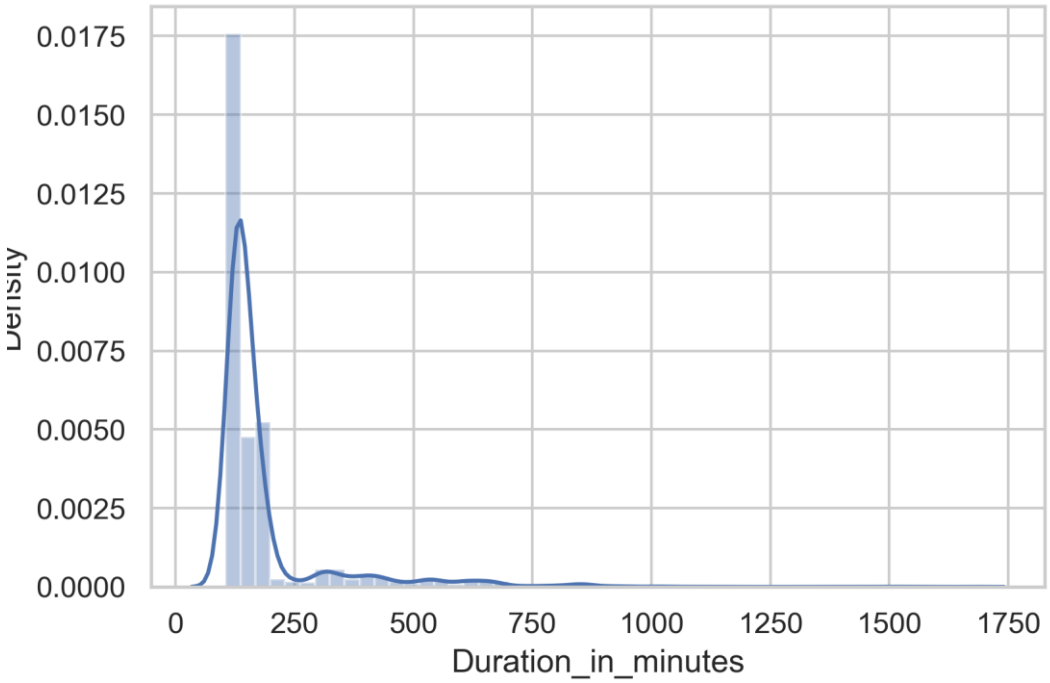

Density
5
4
3
2
1
0
−0.5
0.0
0.5
1.0
1.5
2.0
2.5
Destination

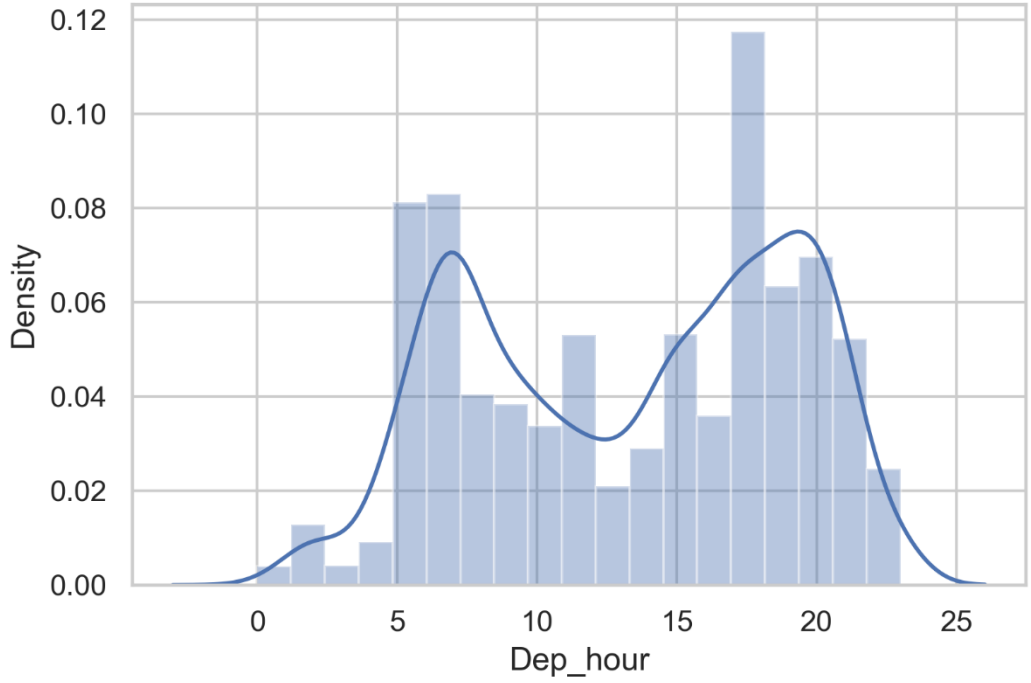

14
12
10
8
6
4
2
0
Density
0.0
0.5
1.0
1.5
2.0
Class

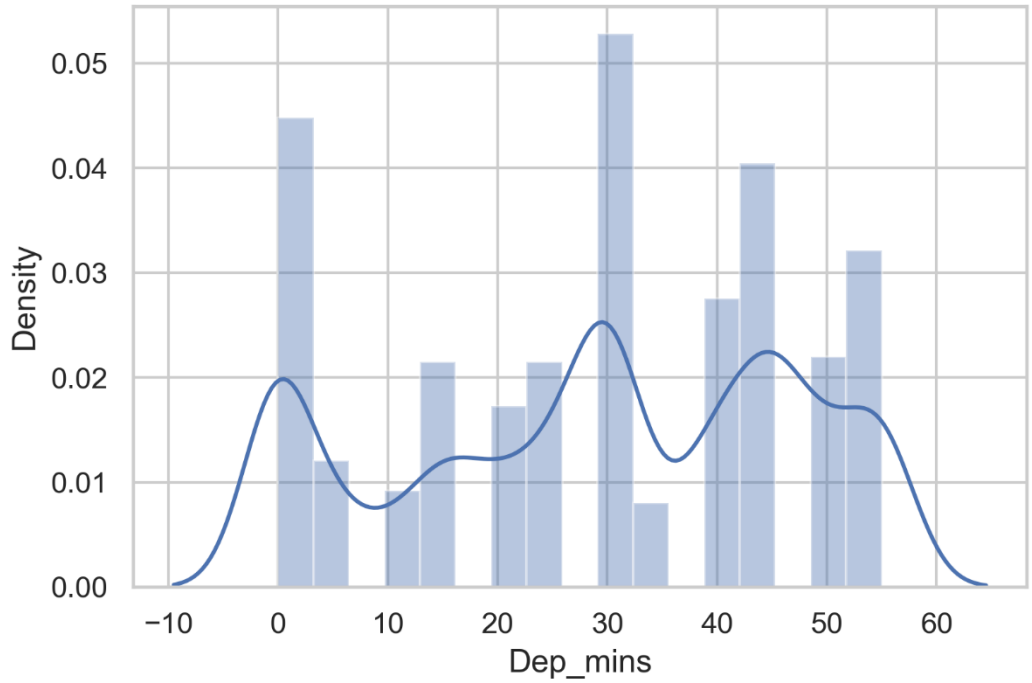

0.000200
0.000175
0.000150
0.000125
0.000100
0.000075
0.000050
0.000025
0.000000
0
20000
40000
60000
80000
100000
Price

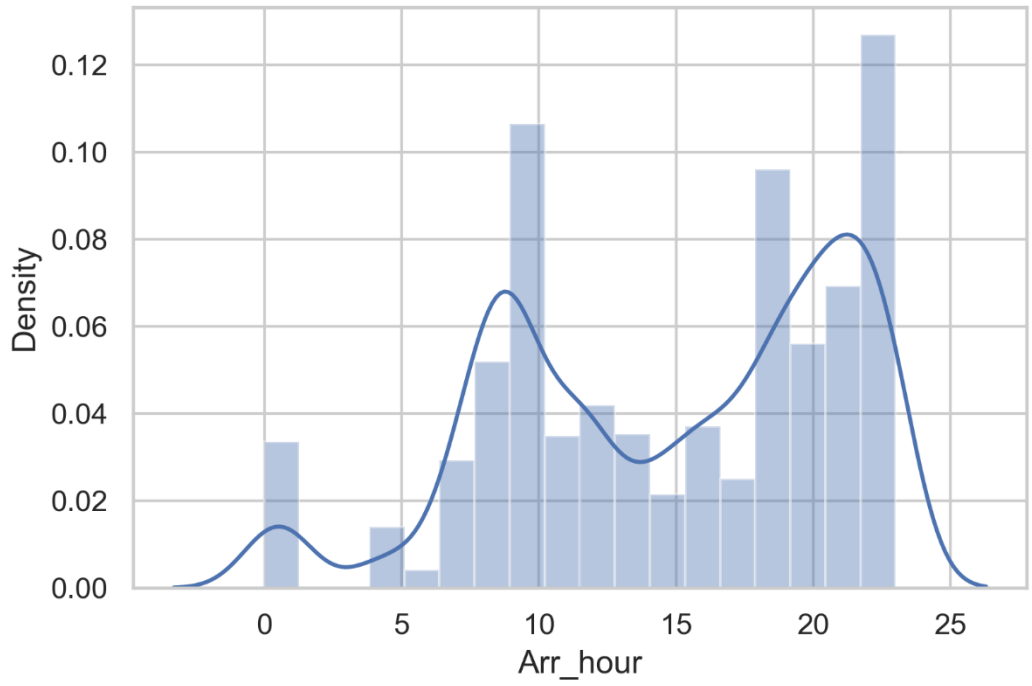

0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
Density
−1
0
1
2
3
4
5
6
7
Weekday

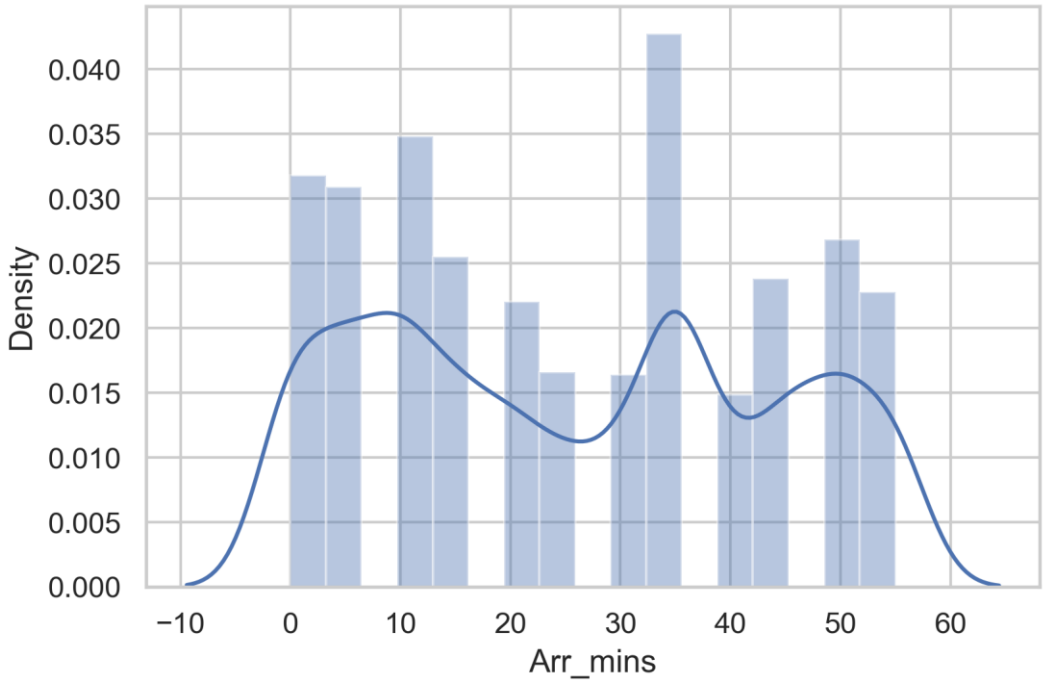

0.040
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
Density
−5
0
5
10
15
20
25
30
35
day_of_journey

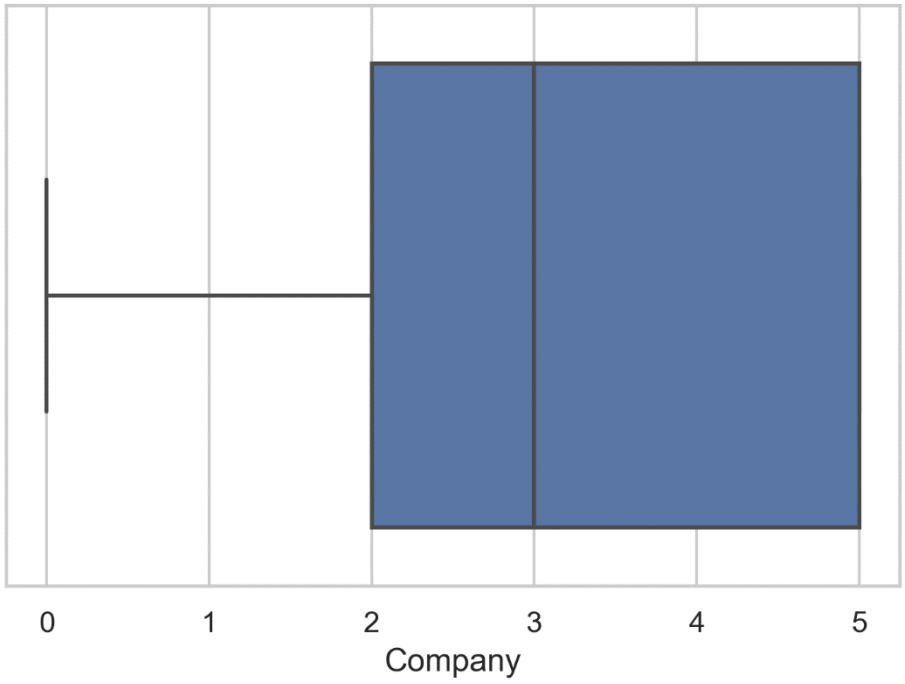

0.0175
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
0.0000
0
250
500
750
1000
1250
1500
1750
Duration_in_minutes

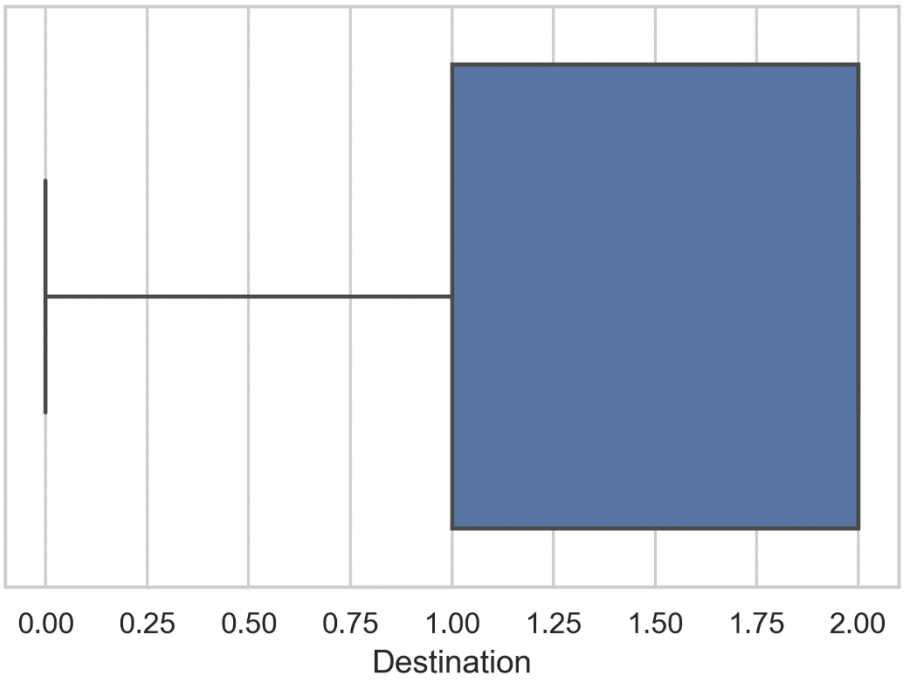

0.12
0.10
0.08
0.06
0.04
0.02
0.00
Density
0
5
10
15
20
25
Dep_hour

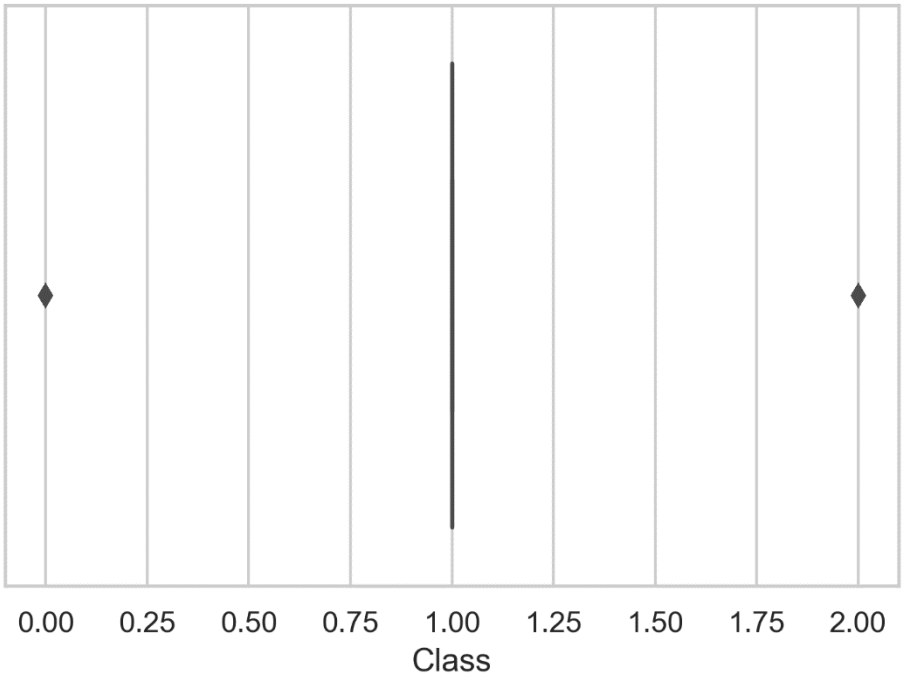

0.05
0.04
0.03
0.02
0.01
0.00
Density
−10
0
10
20
30
40
50
60
Dep_mins

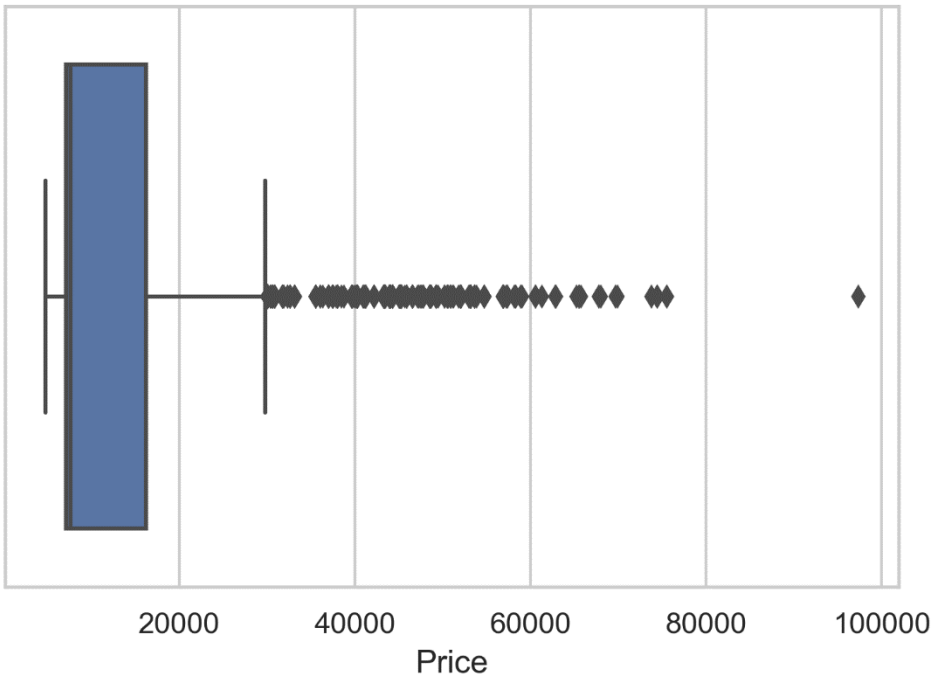

0.12
0.10
0.08
0.06
0.04
0.02
0.00
Density
0
5
10
15
20
25
Arr_hour

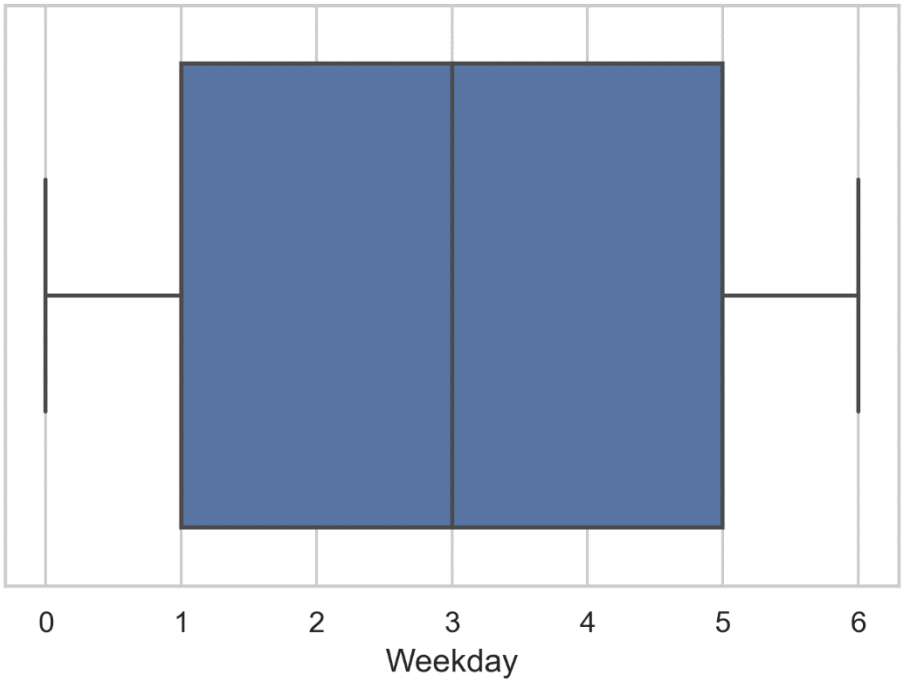

0.040
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
Density
−10
0
10
20
30
40
50
60
Arr_mins

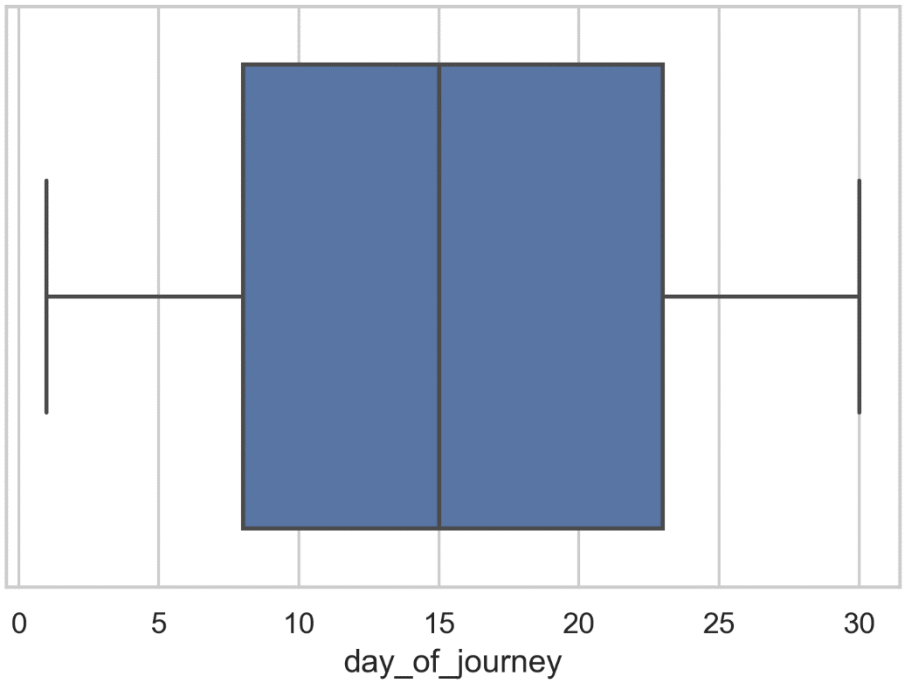

0
1
2
3
4
5
Company

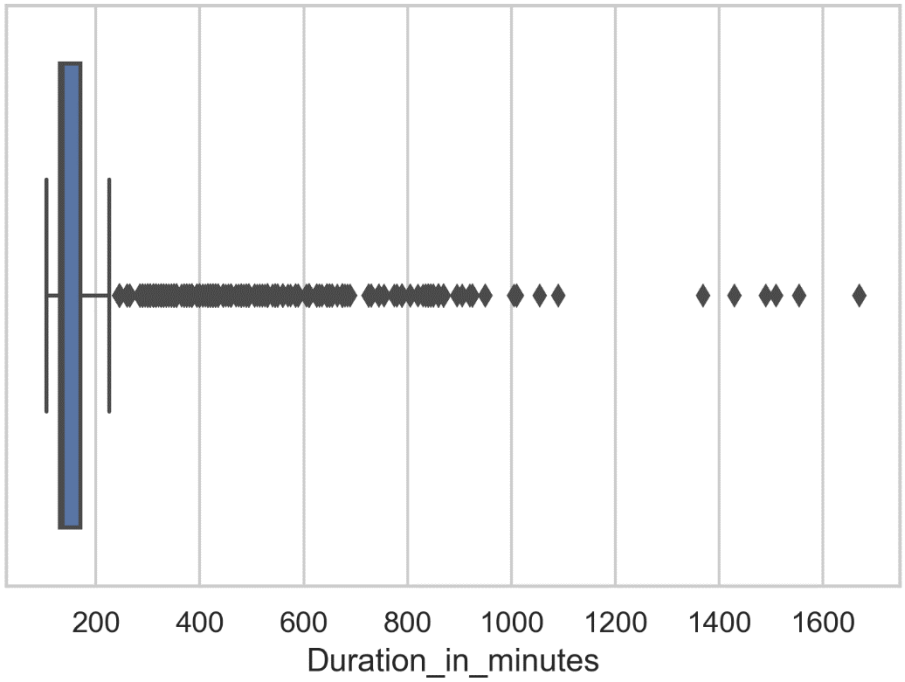

0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Destination

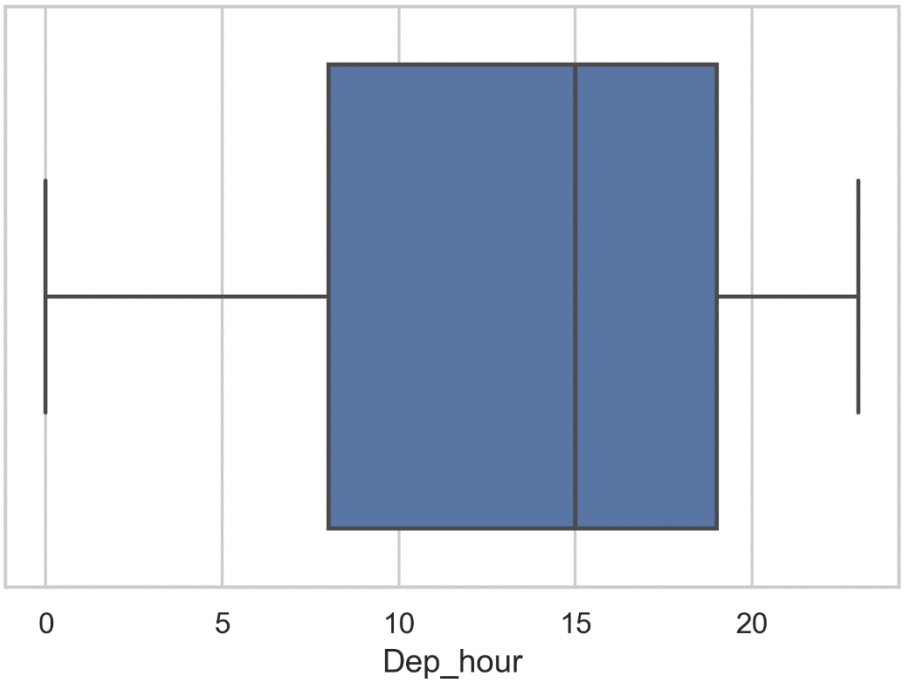

0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Class

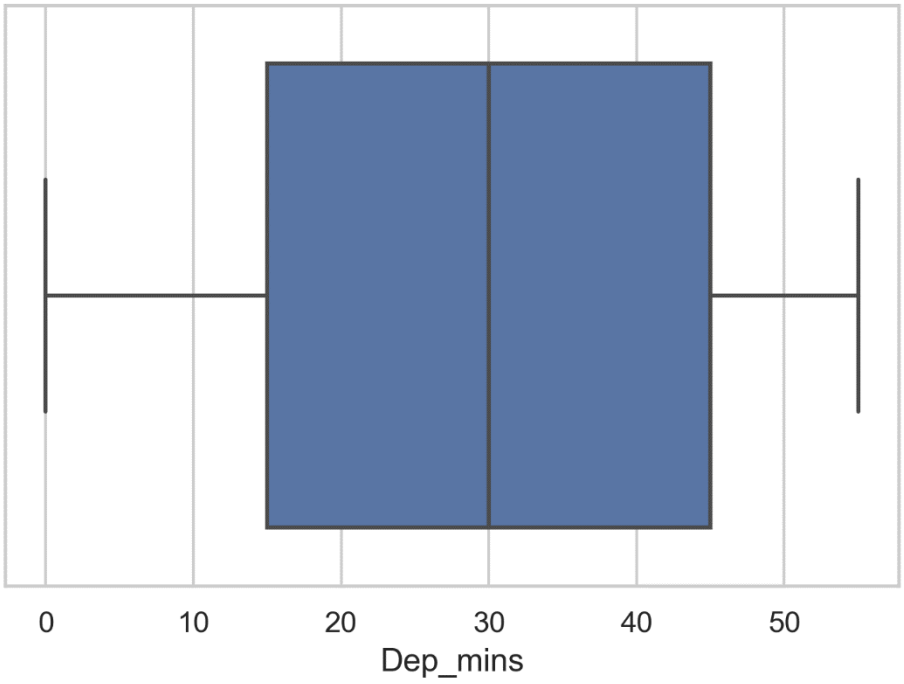

20000
40000
60000
80000
100000
Price

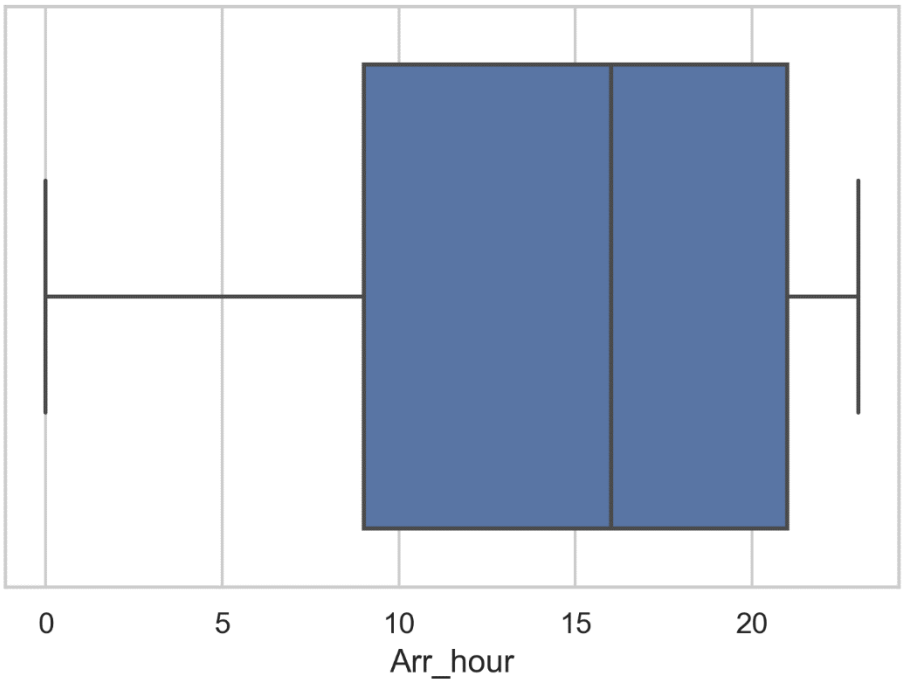

0
1
2
3
4
5
6
Weekday

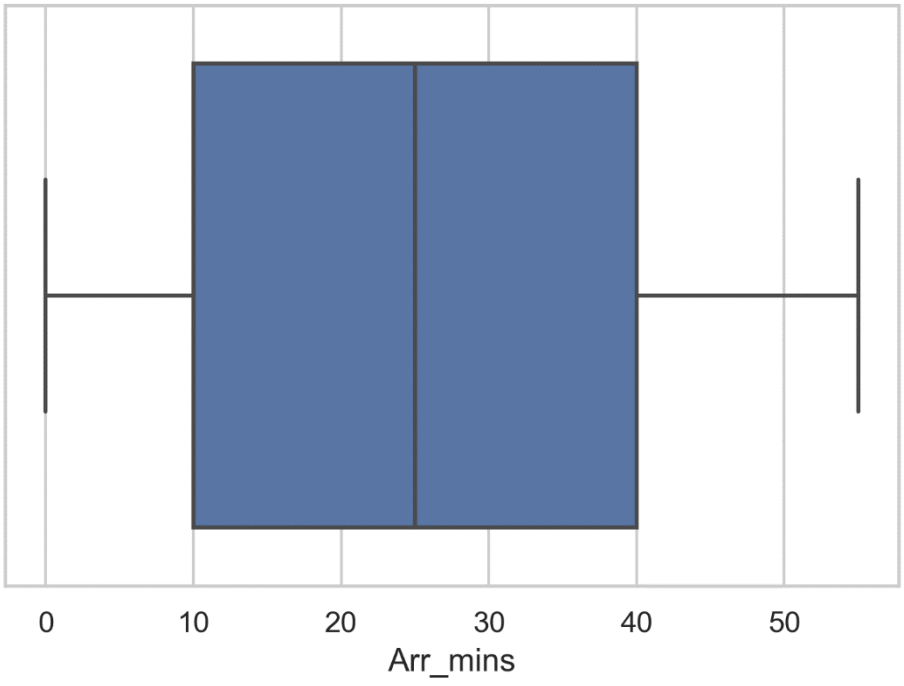

0
5
10
15
20
25
30
day_of_journey

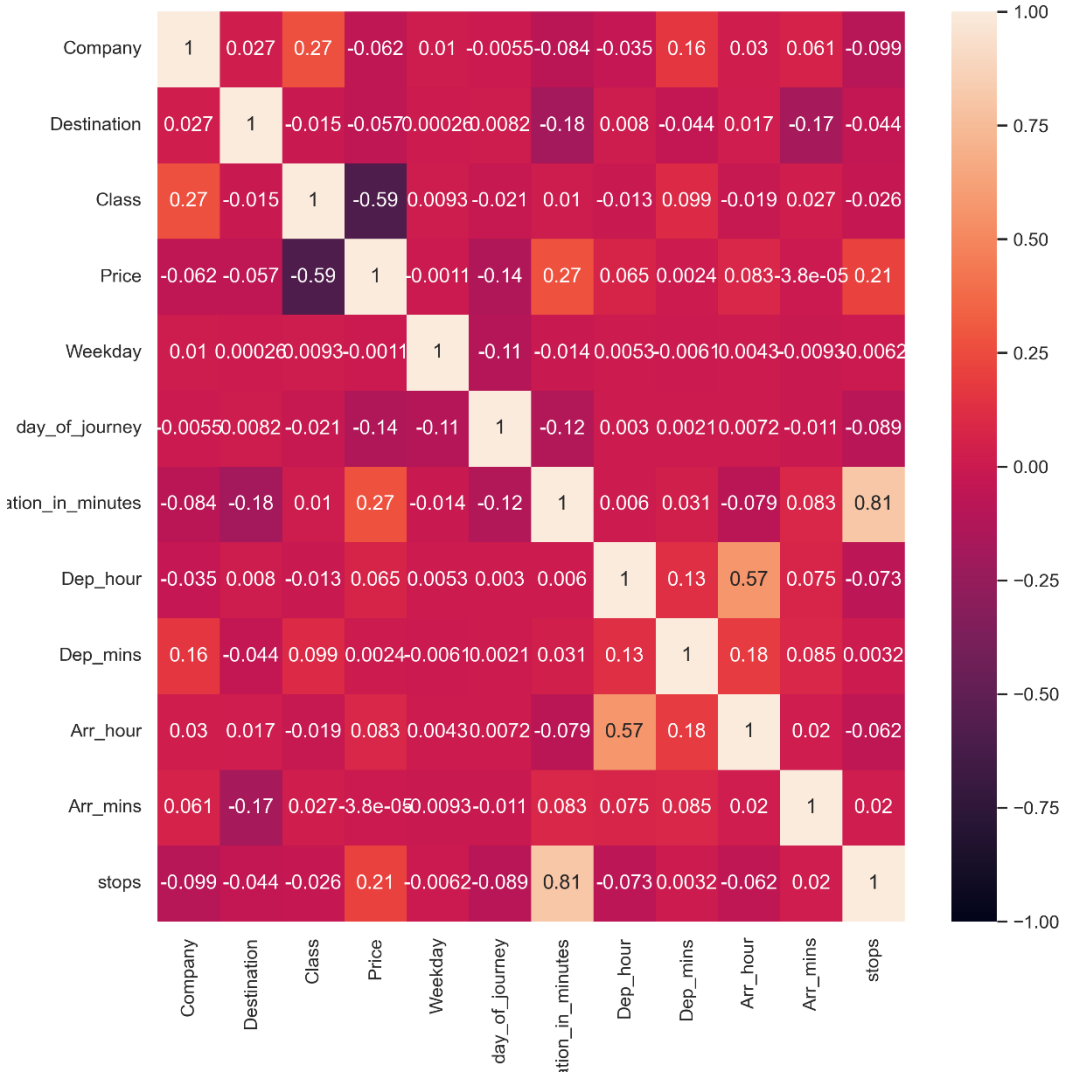

200
400
600
800
1000
1200
1400
1600
Duration_in_minutes


0
5
10
15
20
Dep_hour


0
10
20
30
40
50
Dep_mins

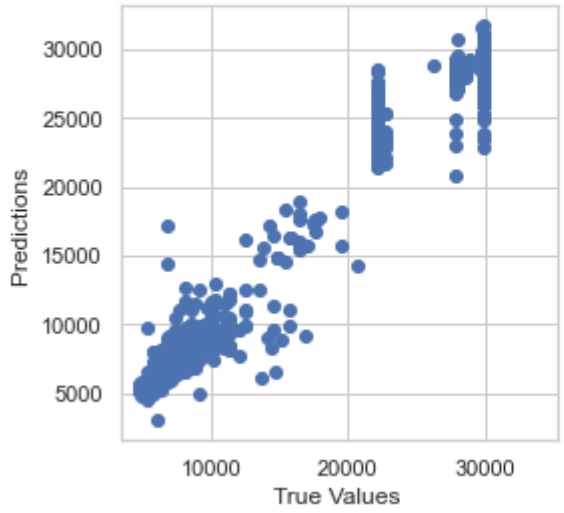

0
5
10
15
20
Arr_hour


0
10
20
30
40
50
Arr_mins


Company 1 0.027 0.27 -0.062 0.01 -0.0055 -0.084 -0.035 0.16 0.03 0.061 -0.099
Destination 0.027 1 -0.015 -0.057 0.00026 0.0082 -0.18 0.008 -0.044 0.017 -0.17 -0.044
Class 0.27 -0.015 1 -0.59 0.0093 -0.021 0.01 -0.013 0.099 -0.019 0.027 -0.026
Price -0.062 -0.057 -0.59 1 -0.0011 -0.14 0.27 0.065 0.0024 0.083 -3.8e-05 0.21
Weekday 0.01 0.00026 0.0093 -0.0011 1 -0.11 -0.014 0.0053 -0.0061 0.0043 -0.0093 0.0062
day_of_journey -0.0055 0.0082 -0.021 -0.14 -0.11 1 -0.12 0.003 0.0021 0.0072 -0.011 -0.089
ation_in_minutes -0.084 -0.18 0.01 0.27 -0.014 -0.12 1 0.006 0.031 -0.079 0.083 0.81
Dep_hour -0.035 0.008 -0.013 0.065 0.0053 0.003 0.006 1 0.13 0.57 0.075 -0.073
Dep_mins 0.16 -0.044 0.099 0.0024 -0.0061 0.0021 0.031 0.13 1 0.18 0.085 0.0032
Arr_hour 0.03 0.017 -0.019 0.083 0.0043 0.0072 -0.079 0.57 0.18 1 0.02 -0.062
Arr_mins 0.061 -0.17 0.027 -3.8e-05 0.0093 -0.011 0.083 0.075 0.085 0.02 1 0.02
stops -0.099 -0.044 -0.026 0.21 -0.0062 -0.089 0.81 -0.073 0.0032 -0.062 0.02 1
Company
Destination
Class
Price
Weekday
day_of_journey
ition_in_minutes
Dep_hour
Dep_mins
Arr_hour
Arr_mins
stops
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00

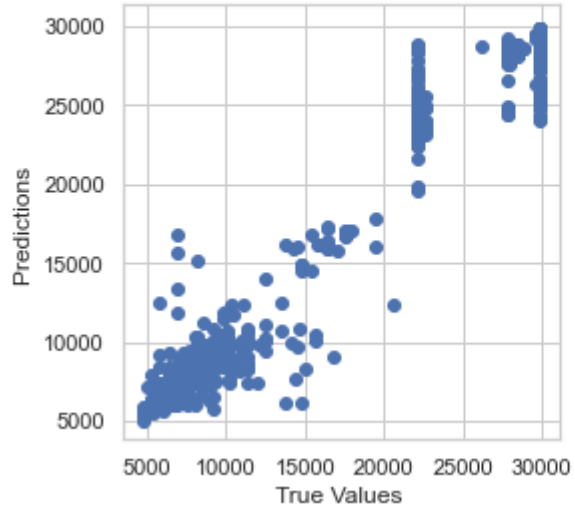
XGB Regression

```
(3545.9375, 35224.60009765625, 1544.376220703125, 33223.038818359375)
```



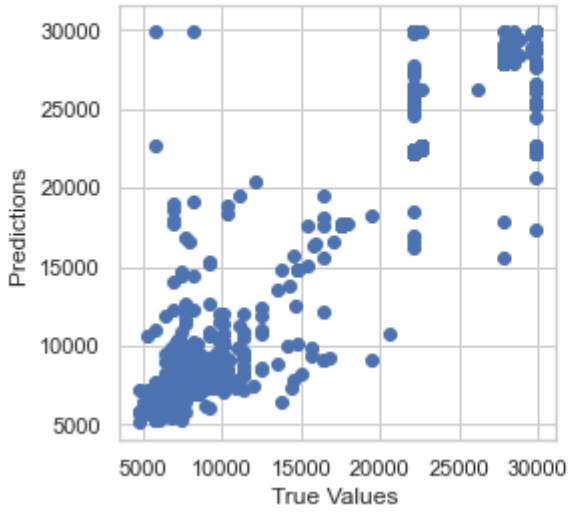
Random Forest Regression

```
(3545.9375, 31113.3125, 3832.7055758512074, 31400.080575851207)
```



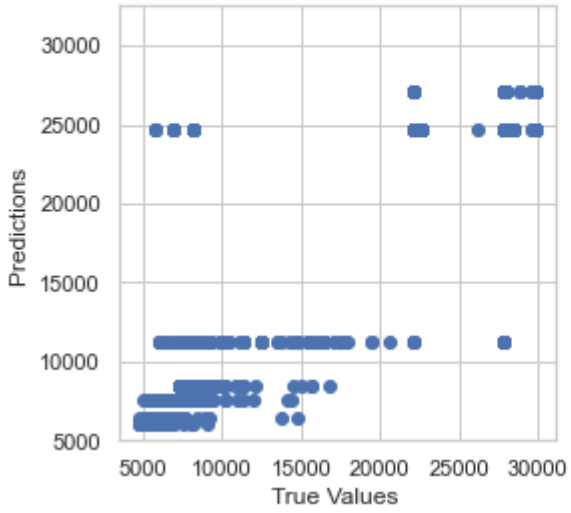
KNN Regression

```
(3545.9375, 31113.3125, 4002.0344314655767, 31569.409431465578)
```



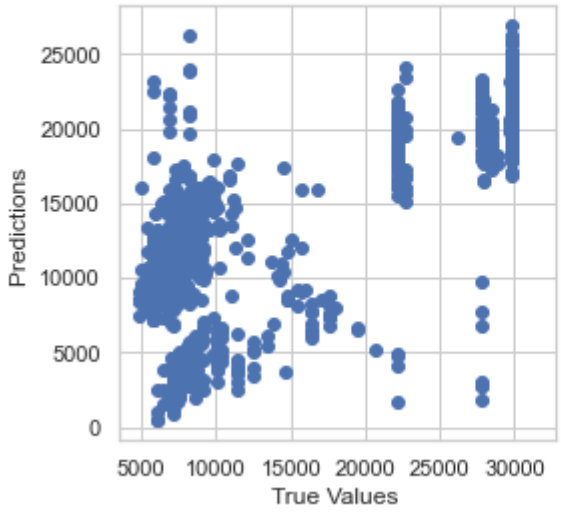
Decision Tree

```
(3545.9375, 31113.3125, 4999.642049088359, 32567.01704908836)
```



Linear Regression

(3545.9375, 32740.17013912629, -904.9289606299224, 28289.30367849637)



Visualization Observation:

1. Vistara has a greater number of flights from Delhi to mentioned cities followed by Indigo, Air Asia has very a smaller number of flights compared to all the other airlines.
2. Delhi to Mumbai has maximum number of flights from Delhi followed by Kolkata and then Chennai
3. Economy Class is the majority class of all followed by Business and Premium Economy.
4. Sunday and Monday have majority reservation and Tuesday has the minimum of all the weekdays.
5. All the days of the month has even number of reservations except for 10^{th} of the month.
6. Early morning and mid night Departure have less traffic, evening and morning has maximum traffic.
7. Morning 9AM and late night from 9PM to 11PM have the maximum arrival of flights.
8. $5/6^{th}$ of the flights are non-stop flights.
9. The price of the ticket is high if the number of days between the date of booking and date of journey is less.
10. Vistara and Air India are the only air companies to have maximum business class seats. Indigo has the majority, Economy Class. Air Asia has the less travelling flights.
11. Flights from Delhi to Mumbai are high in number compared to other two cities.
12. Vistara and Air India ticket price are the highest and Air Asia has the lowest.
13. Business class tickets are comparatively higher than the other two classes.

14. Travelling on Sunday and Friday is way more expensive than other week days.
15. Price and Duration of Flight has outliers.
16. Class and Price have high correlativity followed by flight duration time.
17. XGB Regression has the best fit of all the machine learning models.

Results:

Out of all the Regression models XGB Regressor has performed well, KNN has some over fitting compared all the other model i.e., Random Forest too have performed good. Linear regression is does not have good fit compared to all the other models and is not recommended for this data. Also, the predicted values and true value have good fit in case of XGB.

Conclusion:

In the whole dataset Class column has the most influence on the price of the flight ticket. Flight Travel Duration in minutes is the other good influence on the price prediction.

Outcome of the Study:

Visualizing data helped to negotiate few outliers and biased data. Data Cleaning helps in minimizing the overfitting created during model training and improves the model performance. Random Forest can neglect outliers even when the data is fed with outliers to the machine learning model. XGB Regression is worth the use in this type of problem though it required ample amount of time for the results. Simplifying the data was the most challenge part in this project but it can overcome if one can fluently use python libraries and its functions.

Limitations of this work and Scope for Future Work:

The ticket price predicted are only limited to travel from Delhi to particular given cities. When we try to predict ticket price of different city the machine learning model will fail. Also, for 2 stoppage flights tickets it will also fail as the model is not trained for 2 stoppage flights, as most of the data available are data of the one or non-stop therefore prediction of 2 stop model is always constrained. To improve the model efficiency, we can introduce various other factors such as Airbus model, baggage load and travel meals etc., inclusion of these can improve the model quality.