EDA & REGRESSION COURSEWORK 2024

The dataset for this coursework assignment, *MavenRail.csv*, is a synthetic dataset created by Maven Analytics for an online data analytics challenge. Participants were tasked with building an exploratory dashboard based on fictional traveler behavior and operational performance on the UK National Rail network.

According to Spiegelhalter's 'star rating' system, the dataset would earn a "4*" rating—if it were factual—indicating high confidence as "numbers we can believe" (Muldoon, 2024).

The dataset contains 31,645 rows of individual traveler journeys, described across 13 columns:

1	Payment Method	Contactless, Credit Card, Debit Card
2	Railcard Type	Adult, Disabled, Senior, None
3	Ticket Class	First Class, Standard
4	Ticket Type	Advance, Anytime, Off-Peak
5	Price	GBP
6	Departure Station	Various UK National Rail Stations
7	Arrival Station	Various UK National Rail Stations
8	Departure Time	YYYY/mm/dd HH:MM
9	Scheduled Arrival	YYYY/mm/dd HH:MM
10	Actual Arrival	
11	Journey Status	On Time, Delayed, Cancelled
12	Reason for Delay	Signal Failure, Technical Issue, Weather, Staffing, Staff, Traffic
13	Refund Request	Yes/No

These details suggest the dataset was designed to study how journey delays and cancellations impact refund requests, a critical performance metric for service providers like National Rail.

Data Preparation

To begin, I examined the dataset structure. Most values were categorical, including Departure, Scheduled Arrival, and Actual Arrival, with some missing values in key columns: Railcard (67%), Actual Arrival (5.9%), and Reason for Delay (86.8%).

- **Railcard:** Null values were replaced with "None," a valid category alongside "Adult," "Disabled," and "Senior."
- **Reason for Delay:** Null values were similarly replaced with "None."

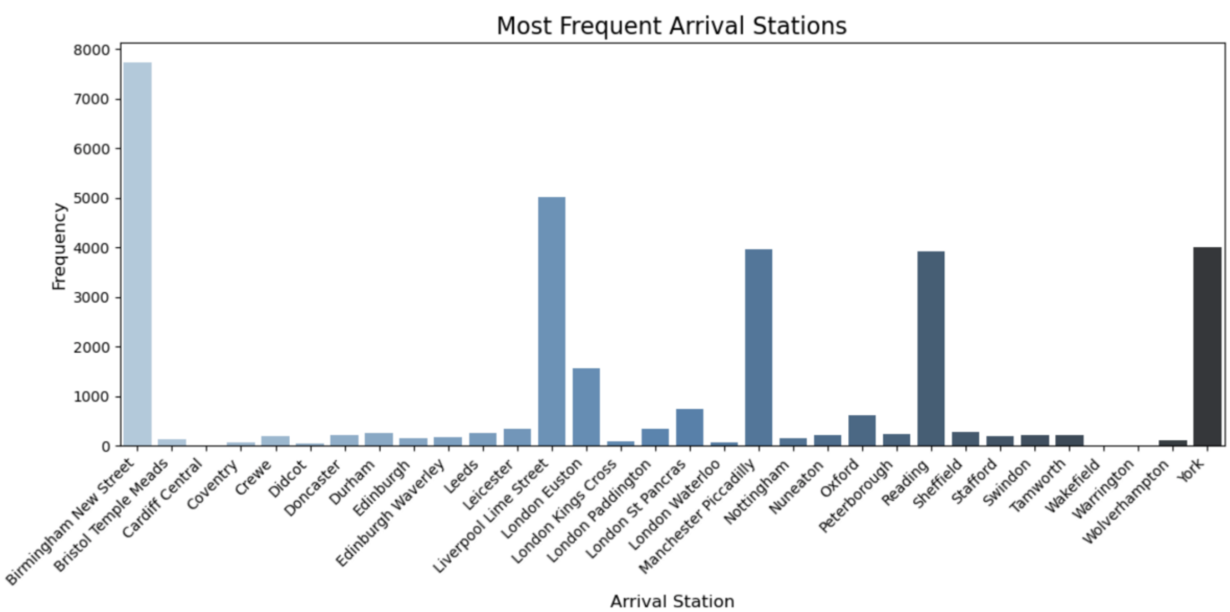
- **Scheduled Arrival and Departure:** Null values were removed due to their low frequency.
- **Actual Arrival:** Null values were replaced with "NA," indicating no delay.

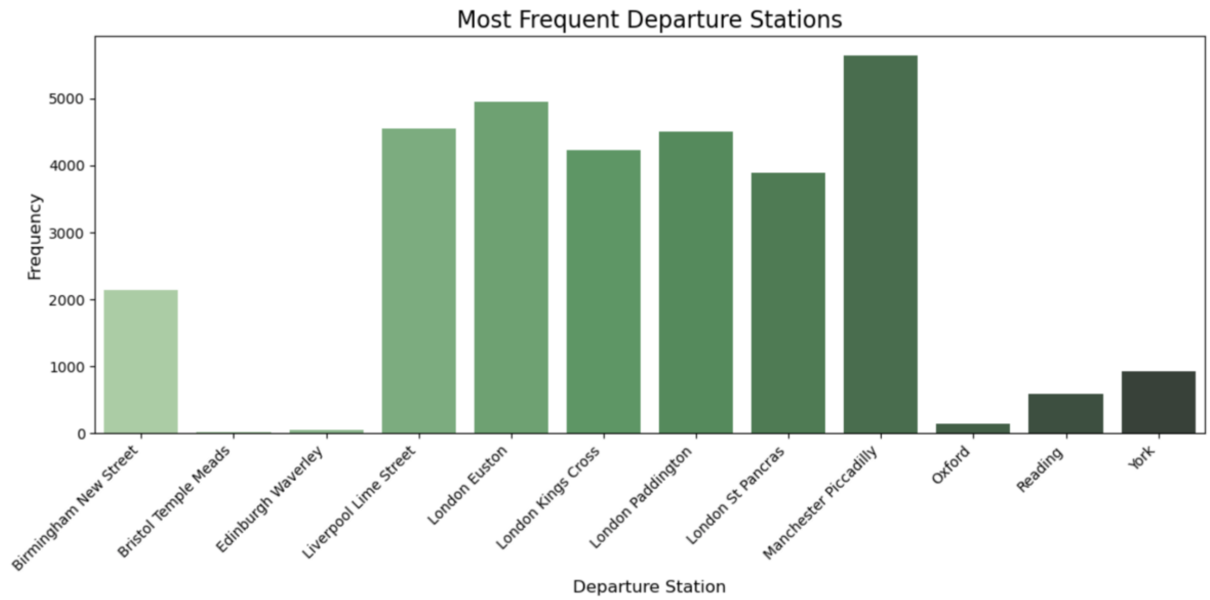
For analysis, departure and arrival times were converted to datetime64. Null values in Actual Arrival were marked as "Not a Time" to avoid exceptions during conversion.

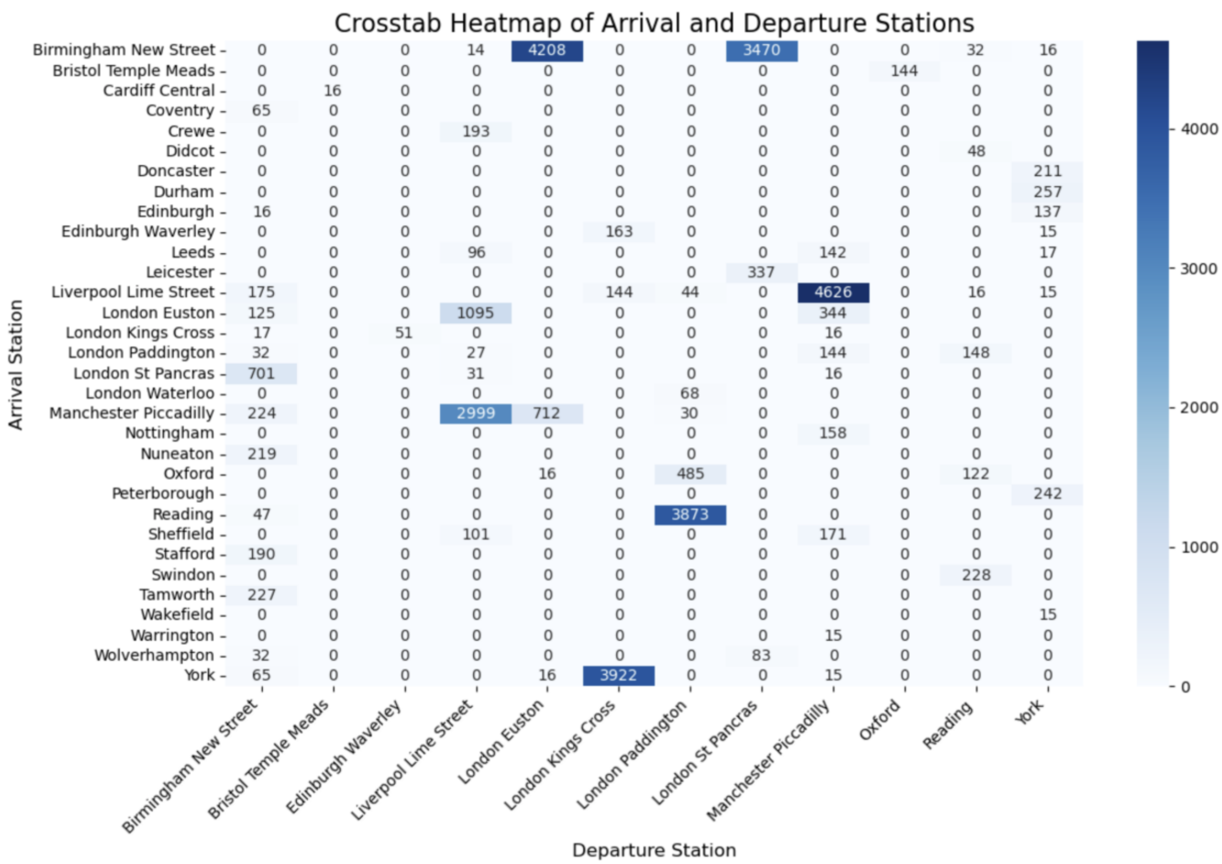
Exploratory Data Analysis

Departure and Arrival Patterns

Using cross-tabulations, I found that Manchester Piccadilly was the most frequent departure station, and Liverpool Lime Street the most common arrival station. The most frequent journey was from Manchester Piccadilly to Liverpool Lime Street.







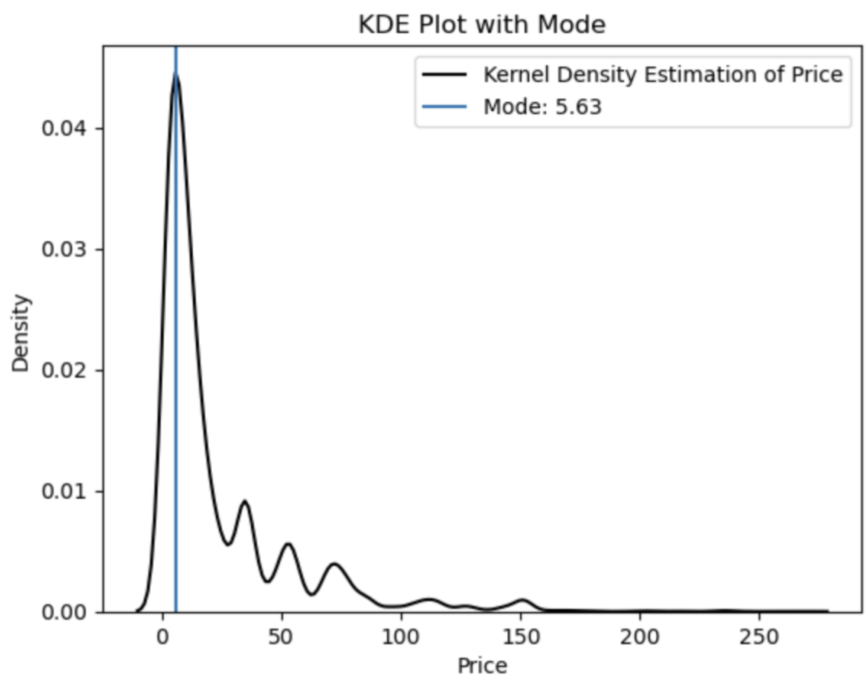
Ticket Prices

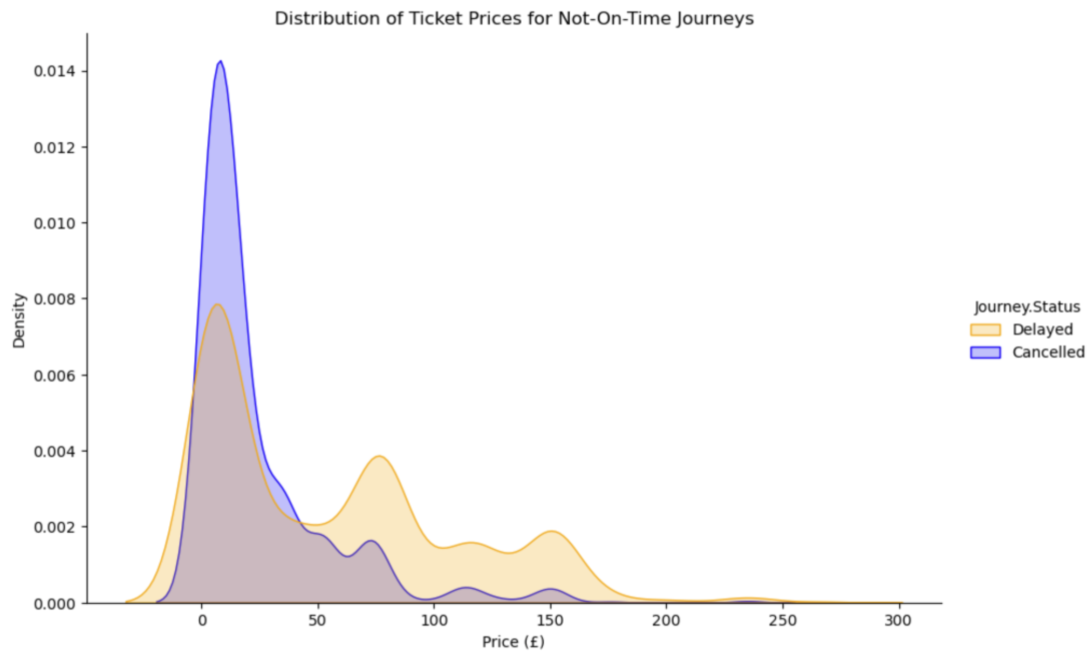
Univariate analysis of ticket prices revealed:

- Mean: £23.43

- Standard deviation: £29.98
- Minimum: £1
- Maximum: £267
- Median: £11
- Mode: £3

A Kernel Density Estimation (KDE) plot showed a KDE mode of £5.63, differing from the raw data mode likely due to the influence of outliers (e.g., tickets above £250). Price analysis by journey status showed that delayed trips were costlier, while cancelled trips had a higher density at lower prices. This might indicate that cheaper, more frequent routes are cancelled more often, while costlier routes are delayed to avoid cancellations.





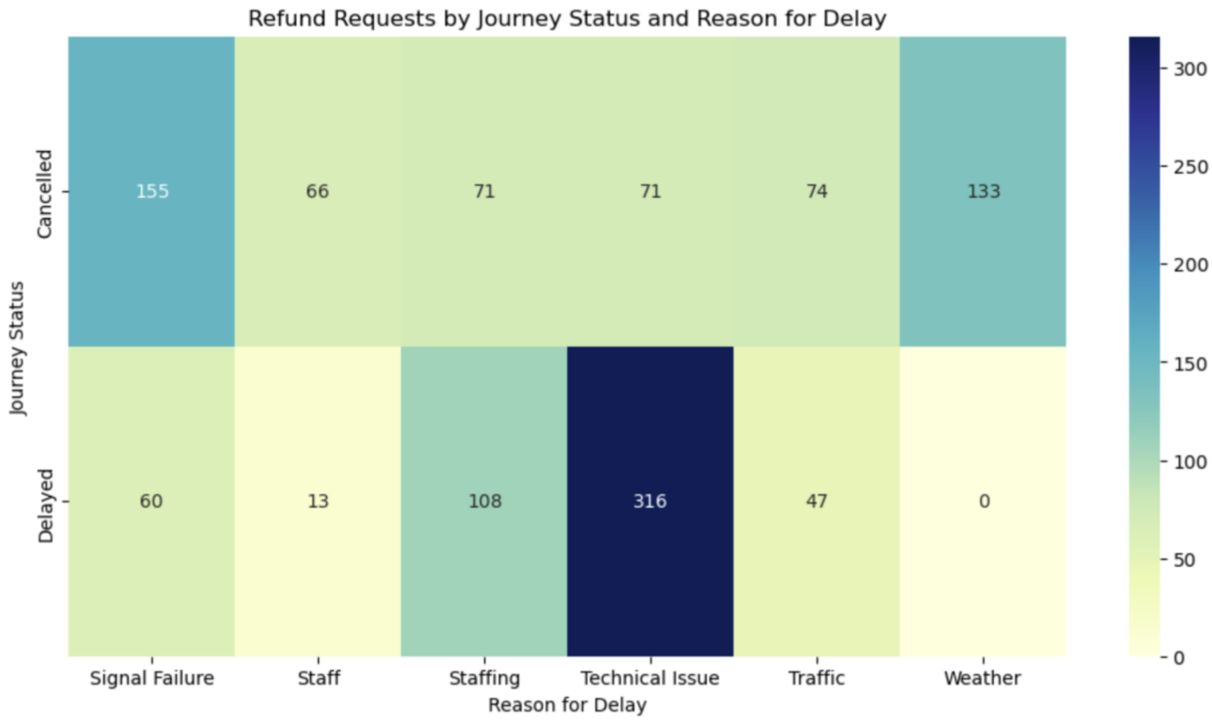
Analysis of Refund Requests

To examine refund requests, I filtered the dataset to include only delayed or cancelled journeys and cross-tabulated them with Reason for Delay. Results showed that:

- **Delayed Journeys:** Refund requests were most frequent when delays were due to technical issues.
- **Cancelled Journeys:** Signal failure was the leading cause, drawing the most refund requests.

When analyzing the relationship between Reason for Delay and Minutes Delayed:

- Travelers on delayed trains waited most often due to weather.
- For cancelled trains, the longest delays were caused by signal failure



Data Transformation

To prepare for modeling, I calculated the delay in minutes by subtracting Scheduled Arrival from Actual Arrival, with no-delay values marked as 0.0 and then replaced with "NA." A new column, *MediumPrice*, was created to indicate whether ticket prices fell between £10 and £30, mapped as binary values ("Yes" or "No").

Outliers in ticket prices were removed using a Z-score threshold, reducing the maximum price to £113 and the mean to £20.43 across 30,881 values. I converted categorical columns to numerical or categorical types as needed, with DelayInMinutes set to numeric, replacing "NA" values with 0.

Price	Departure.Station	Arrival.Station	Departure	Scheduled.Arrival	Actual.Arrival	Journey.Status	Reason.for.Delay	Refund.Request	DelayInMinutes
43	London Paddington	Liverpool Lime Street	2024-01-01 11:00:00	2024-01-01 13:30:00	2024-01-01 13:30:00	On Time	NA	No	NA
23	London Kings Cross	York	2024-01-01 09:45:00	2024-01-01 11:35:00	2024-01-01 11:40:00	Delayed	Signal Failure	No	5.0
3	Liverpool Lime Street	Manchester Piccadilly	2024-01-02 18:15:00	2024-01-02 18:45:00	2024-01-02 18:45:00	On Time	NA	No	NA
13	London Paddington	Reading	2024-01-01 21:30:00	2024-01-01 22:30:00	2024-01-01 22:30:00	On Time	NA	No	NA
76	Liverpool Lime Street	London Euston	2024-01-01 16:45:00	2024-01-01 19:00:00	2024-01-01 19:00:00	On Time	NA	No	NA

Logistic Regression Models

Single-Predictor Model

```
#fit small model with one predictor
X_1 = journey_final['MediumPrice'].values.reshape(-1, 1)
X_1 = sm.add_constant(X_1)
#set response variable
y = journey_final.loc[:, journey_final.columns == 'Refund.Request_Yes']
```

```
logit_model=sm.Logit(y,X_1)
result1=logit_model.fit()
print(result1.summary2())
```

```
Optimization terminated successfully.
         Current function value: 0.578797
         Iterations 5
                          Results: Logit
==================================================================
Model:              Logit            Method:           MLE
Dependent Variable: Refund.Request_Yes Pseudo R-squared: 0.003
Date:               2024-11-21 00:14 AIC:              4825.3797
No. Observations:   4165             BIC:              4838.0487
Df Model:           1                Log-Likelihood:   -2410.7
Df Residuals:       4163             LL-Null:          -2418.7
Converged:          1.0000           LLR p-value:      6.1887e-05
No. Iterations:     5.0000           Scale:            1.0000
--------------------------------------------------------------------
            Coef.   Std.Err.      z      P>|z|    [0.025    0.975]
--------------------------------------------------------------------
const      -1.0753    0.0393  -27.3720  0.0000  -1.1523   -0.9983
x1          0.3540    0.0872    4.0583  0.0000   0.1830    0.5250
==================================================================
```

Using *MediumPrice* as the sole predictor, I developed a logistic regression model to predict refund requests. In this model:

- Travelers paying less than £10 or more than £30 had lower refund probabilities.
- For a £5 ticket, the refund probability was 25%.

$$p = \frac{1}{1+\mathrm{e}^{-\theta}} = \frac{1}{1+e^{-(-1.0753+0.3540\times 0)}} = \frac{1}{1+e^{-(-1.0753)}} = \frac{1}{3.93} = 0.25\times\ 100 = 25\%$$

- For a £25 ticket, the refund probability was 33%.

$$p = \frac{1}{1+\mathrm{e}^{-\theta}} = \frac{1}{1+e^{-(-1.0753+0.3540\times 0)}} = \frac{1}{1+e^{-(-0.7213)}} = \frac{1}{3.057} = 0.33\ \times\ 100 = 33\%$$

However, this model performed poorly, with a Pseudo R-squared value of 0.003, indicating minimal explanatory power.

```
# Predict refund request using test data from to_predict

X_1 = predict_journey_final['MediumPrice'].values.reshape(-1, 1)
X_1 = sm.add_constant(X_1)

yhat = result1.predict(X_1)
prediction = list(map(round, yhat))

print("Predicted probabilities:", yhat)
print("Binary predictions (0 or 1):", prediction)
```

```
Predicted probabilities: [0.2543911 0.2543911 0.2543911 0.2543911 0.2543911 0.3271028]
Binary predictions (0 or 1): [0, 0, 0, 0, 0, 0]
```

Multiple Logistic Regression

To improve predictions, I used the following predictors: *MediumPrice*, *Price*, *DelayInMinutes*, *Journey Status Delayed*, *Reason for Delay (Staffing)*, and *Reason for Delay (Technical Issue)*. This model achieved a better Pseudo R-squared value of 0.179. Key results:

```
Optimization terminated successfully.
         Current function value: 0.476949
         Iterations 7
                                 Results: Logit
================================================================================
Model:                  Logit                  Method:                 MLE
Dependent Variable:     Refund.Request_Yes     Pseudo R-squared:       0.179
Date:                   2024-11-22 00:08       AIC:                    3986.9828
No. Observations:       4165                   BIC:                    4031.3241
Df Model:               6                      Log-Likelihood:         -1986.5
Df Residuals:           4158                   LL-Null:                -2418.7
Converged:              1.0000                 LLR p-value:            1.8257e-183
No. Iterations:         7.0000                 Scale:                  1.0000
--------------------------------------------------------------------------------
                                  Coef.  Std.Err.    z     P>|z|   [0.025  0.975]
--------------------------------------------------------------------------------
const                            -1.0594   0.0696 -15.2188 0.0000 -1.1958 -0.9230
MediumPrice                      -0.0408   0.0985  -0.4141 0.6788 -0.2338  0.1523
Price                            -0.0048   0.0010  -4.5999 0.0000 -0.0068 -0.0028
DelayInMinutes                   -0.0593   0.0038 -15.5963 0.0000 -0.0667 -0.0518
Journey.Status_Delayed            1.4501   0.1293  11.2189 0.0000  1.1968  1.7035
Reason.for.Delay_Staffing         0.9982   0.1172   8.5144 0.0000  0.7684  1.2280
Reason.for.Delay_Technical Issue  1.4509   0.0965  15.0341 0.0000  1.2617  1.6400
================================================================================
```

- **Journey Status Delayed (coefficient: 1.45):** Strongly associated with refund requests.
- **Reason for Delay - Technical Issue (coefficient: 1.45):** Significant contributor.
- **MediumPrice (coefficient: -0.0408):** Not a meaningful predictor compared to other variables.

Predictions on Test Data

Using the multiple logistic model, I predicted refund probabilities for the *To_Predict* dataset:

- Journey 2: 1%
- Journey 3: 14.8%
- Journey 4: 55.7%
- Journey 5: 45.4%
- Journey 6: 0.1%
- Journey 7: 12.3%

These probabilities reflect that journey delays, especially those caused by technical issues, significantly increase refund requests, far more than ticket price or whether the price falls within a specific range.

Conclusion

Delays and cancellations are critical drivers of refund requests, with technical issues and staffing delays being the most problematic causes. While pricing influences refund probabilities to a small extent, operational performance, particularly avoiding delays, has a more substantial impact. National Rail should focus on addressing the root causes of delays, especially technical issues, to improve customer satisfaction and reduce operational costs.

REFERENCES

Agresti, A. (2018). 'Multiple Logistic Regression', *Statistical Methods for the Social Sciences.* Fifth edn: Pearson, pp. 477-479.

Maven Rail Challenge (2024). Maven Analytics. Available at: https://bit.ly/MavenRailChallenge (Accessed: 25/11/2024).

Muldoon, M. (2024). 'Lecture 2A: Univariate Exploratory Data Analysis', *Statistics and Machine Learning, Week 2*, 2024(15/11/2024) Available at: https://online.manchester.ac.uk/ultra/courses/_83866_1/cl/outline.

1. MavenRail Data Description

In [2]:
```python
import pandas as pd
import numpy as np
df = pd.read_csv('MavenRail.csv')
df.head()
```

Out[2]:

	Payment.Method	Railcard	Ticket.Class	Ticket.Type	Price	Departure.Station	Arrival.Station	Departure	Scl
0	Contactless	Adult	Standard	Advance	43	London Paddington	Liverpool Lime Street	2024-01-01 11:00	20
1	Credit Card	Adult	Standard	Advance	23	London Kings Cross	York	2024-01-01 09:45	20
2	Credit Card	NaN	Standard	Advance	3	Liverpool Lime Street	Manchester Piccadilly	2024-01-02 18:15	20
3	Credit Card	NaN	Standard	Advance	13	London Paddington	Reading	2024-01-01 21:30	20
4	Contactless	NaN	Standard	Advance	76	Liverpool Lime Street	London Euston	2024-01-01 16:45	20

In [3]:
```python
df.describe
```

Out[3]:
```
<bound method NDFrame.describe of       Payment.Method Railcard Ticket.Class Ticket.Type  Price  \
0        Contactless    Adult     Standard     Advance     43
1        Credit Card    Adult     Standard     Advance     23
2        Credit Card      NaN     Standard     Advance      3
3        Credit Card      NaN     Standard     Advance     13
4        Contactless      NaN     Standard     Advance     76
...              ...      ...          ...         ...    ...
31640    Credit Card      NaN     Standard    Off-Peak      4
31641    Contactless      NaN     Standard    Off-Peak     10
31642    Credit Card      NaN     Standard    Off-Peak      4
31643    Credit Card      NaN     Standard    Off-Peak     10
31644    Credit Card    Adult     Standard    Off-Peak      3

           Departure.Station        Arrival.Station         Departure  \
0          London Paddington  Liverpool Lime Street  2024-01-01 11:00
1         London Kings Cross                   York  2024-01-01 09:45
2      Liverpool Lime Street  Manchester Piccadilly  2024-01-02 18:15
3          London Paddington                Reading  2024-01-01 21:30
4      Liverpool Lime Street          London Euston  2024-01-01 16:45
...                      ...                    ...               ...
31640  Manchester Piccadilly  Liverpool Lime Street  2024-04-30 20:00
31641          London Euston  Birmingham New Street  2024-04-30 20:15
31642  Manchester Piccadilly  Liverpool Lime Street  2024-04-30 20:15
31643          London Euston  Birmingham New Street  2024-04-30 21:15
31644  Liverpool Lime Street  Manchester Piccadilly  2024-04-30 21:30

        Scheduled.Arrival    Actual.Arrival Journey.Status Reason.for.Delay  \
0        2024-01-01 13:30  2024-01-01 13:30        On Time              NaN
1        2024-01-01 11:35  2024-01-01 11:40        Delayed   Signal Failure
2        2024-01-02 18:45  2024-01-02 18:45        On Time              NaN
3        2024-01-01 22:30  2024-01-01 22:30        On Time              NaN
4        2024-01-01 19:00  2024-01-01 19:00        On Time              NaN
...                   ...               ...            ...              ...
31640  2024-04-30 20:30  2024-04-30 20:30        On Time              NaN
31641  2024-04-30 21:35  2024-04-30 21:35        On Time              NaN
31642  2024-04-30 20:45  2024-04-30 20:45        On Time              NaN
31643  2024-04-30 22:35  2024-04-30 22:35        On Time              NaN
31644  2024-04-30 22:00  2024-04-30 22:00        On Time              NaN

      Refund.Request
0                 No
1                 No
2                 No
3                 No
4                 No
...              ...
31640             No
31641             No
31642             No
31643             No
31644             No

[31645 rows x 13 columns]>
```

In [4]:
```python
df.shape
```

Out[4]:
```
(31645, 13)
```

In [5]:
```python
df.columns
```

Out[5]:
```
Index(['Payment.Method', 'Railcard', 'Ticket.Class', 'Ticket.Type', 'Price',
       'Departure.Station', 'Arrival.Station', 'Departure',
       'Scheduled.Arrival', 'Actual.Arrival', 'Journey.Status',
       'Reason.for.Delay', 'Refund.Request'],
      dtype='object')
```

In [6]:
```python
for column in ['Payment.Method', 'Railcard', 'Ticket.Class', 'Ticket.Type', 'Price',
       'Departure.Station', 'Arrival.Station', 'Departure',
       'Scheduled.Arrival', 'Actual.Arrival', 'Journey.Status',
       'Reason.for.Delay', 'Refund.Request']:
    print(f'{column}: {df[column].dtype}')
```

```
Payment.Method: object
Railcard: object
Ticket.Class: object
Ticket.Type: object
Price: int64
Departure.Station: object
Arrival.Station: object
Departure: object
Scheduled.Arrival: object
Actual.Arrival: object
Journey.Status: object
Reason.for.Delay: object
Refund.Request: object
```

In [7]:
```python
df.isnull().sum()
```

Out[7]:
```
Payment.Method           0
Railcard             20911
Ticket.Class             0
Ticket.Type              0
Price                    0
Departure.Station        0
Arrival.Station          0
Departure                3
Scheduled.Arrival        4
Actual.Arrival        1880
Journey.Status           0
Reason.for.Delay     27479
Refund.Request           0
dtype: int64
```

In [8]:
```python
(df.isnull().sum()/(len(df)))*100
```

Out[8]:
```
Payment.Method        0.000000
Railcard             66.079949
Ticket.Class          0.000000
Ticket.Type           0.000000
Price                 0.000000
Departure.Station     0.000000
Arrival.Station       0.000000
Departure             0.009480
Scheduled.Arrival     0.012640
Actual.Arrival        5.940907
Journey.Status        0.000000
Reason.for.Delay     86.835203
Refund.Request        0.000000
dtype: float64
```

In [9]:
```python
payment_methods = []
for value in df['Payment.Method']:
    if value in payment_methods:
        pass
    else:
        payment_methods.append(value)
print(payment_methods)
```

```
['Contactless', 'Credit Card', 'Debit Card']
```

In [10]:
```python
journey_status = []
for value in df['Journey.Status']:
    if value in journey_status:
        pass
    else:
        journey_status.append(value)
print(journey_status)
```

```
['On Time', 'Delayed', 'Cancelled']
```

In [11]:
```python
reasons_for_delay = []
for value in df['Reason.for.Delay']:
    if value in reasons_for_delay:
        pass
    else:
        reasons_for_delay.append(value)
print(reasons_for_delay)
```

```
[nan, 'Signal Failure', 'Technical Issue', 'Weather', 'Staffing', 'Staff', 'Traffic']
```

In [12]:
```python
railcard_type = []
for value in df['Railcard']:
    if value in railcard_type:
        pass
    else:
        railcard_type.append(value)
print(railcard_type)
```

```
['Adult', nan, 'Disabled', 'Senior']
```

In [13]:
```python
df_clean = df.copy()
```

In [14]:
```python
df_clean.isnull().sum()
(df.isnull().sum()/(len(df)))*100
```

Out[14]:
```
Payment.Method        0.000000
Railcard             66.079949
Ticket.Class          0.000000
Ticket.Type           0.000000
Price                 0.000000
Departure.Station     0.000000
Arrival.Station       0.000000
Departure             0.009480
Scheduled.Arrival     0.012640
Actual.Arrival        5.940907
Journey.Status        0.000000
Reason.for.Delay     86.835203
Refund.Request        0.000000
dtype: float64
```

In [15]:
```python
# Remove null values. Convert 'nan' for 'No reason' entry in 'Reason for delay' to 'none'.
df_clean['Reason.for.Delay'] = df_clean['Reason.for.Delay'].fillna('NA')
```

In [16]:
```python
# Remove null values. Convert 'nan' for 'no railcard' entry in 'Railcards' to 'none'.
df_clean['Railcard'] = df_clean['Railcard'].fillna('None')
```

In [17]:
```python
df_clean.isnull().sum()
```

Out[17]:
```
Payment.Method          0
Railcard                0
Ticket.Class            0
Ticket.Type             0
Price                   0
Departure.Station       0
Arrival.Station         0
Departure               3
Scheduled.Arrival       4
Actual.Arrival       1880
Journey.Status          0
Reason.for.Delay        0
Refund.Request          0
dtype: int64
```

In [18]:
```python
# Remove missing values in 'Scheduled Arrival' and 'Departure' due to low number and lack of altern
df_clean = df_clean[~df_clean['Scheduled.Arrival'].isnull()].copy()
df_clean = df_clean[~df_clean['Departure'].isnull()].copy()
```

In [19]:
```python
# Remove null values. Convert null values in 'Actual Arrival' to duplicate 'Scheduled Arrival' valu
df_clean['Actual.Arrival'] = df_clean['Actual.Arrival'].fillna(df_clean['Scheduled.Arrival'])
```

In [20]:
```python
df_clean['Actual.Arrival']
```

Out[20]:
```
0        2024-01-01 13:30
1        2024-01-01 11:40
2        2024-01-02 18:45
3        2024-01-01 22:30
4        2024-01-01 19:00
               ...
31640    2024-04-30 20:30
31641    2024-04-30 21:35
31642    2024-04-30 20:45
31643    2024-04-30 22:35
31644    2024-04-30 22:00
Name: Actual.Arrival, Length: 31639, dtype: object
```

In [21]:
```python
df_clean.isnull().sum()
```

Out[21]:
```
Payment.Method       0
Railcard             0
Ticket.Class         0
Ticket.Type          0
Price                0
Departure.Station    0
Arrival.Station      0
Departure            0
Scheduled.Arrival    0
Actual.Arrival       0
Journey.Status       0
Reason.for.Delay     0
Refund.Request       0
dtype: int64
```

In [22]:
```python
crosstab_result = pd.crosstab(df_clean['Ticket.Class'], df_clean['Ticket.Type'])
print(crosstab_result)
```

```
Ticket.Type   Advance  Anytime  Off-Peak
Ticket.Class
First Class      1762      490       804
Standard        15793     4850      7940
```

In [23]:
```python
# Convert string values for 'Actual Arrival', 'Departure', and 'Scheduled Arrival' to datetime64
df_clean['Actual.Arrival'] = pd.to_datetime(df_clean['Actual.Arrival'], format='%Y-%m-%d %H:%M' )
df_clean['Departure'] = pd.to_datetime(df_clean['Departure'], format='%Y-%m-%d %H:%M' )
df_clean['Scheduled.Arrival'] = pd.to_datetime(df_clean['Scheduled.Arrival'], format='%Y-%m-%d %H:%
```

In [24]:
```python
df_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 31639 entries, 0 to 31644
Data columns (total 13 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Payment.Method     31639 non-null  object
 1   Railcard           31639 non-null  object
 2   Ticket.Class       31639 non-null  object
 3   Ticket.Type        31639 non-null  object
 4   Price              31639 non-null  int64
 5   Departure.Station  31639 non-null  object
 6   Arrival.Station    31639 non-null  object
 7   Departure          31639 non-null  datetime64[ns]
 8   Scheduled.Arrival  31639 non-null  datetime64[ns]
 9   Actual.Arrival     31639 non-null  datetime64[ns]
 10  Journey.Status     31639 non-null  object
 11  Reason.for.Delay   31639 non-null  object
 12  Refund.Request     31639 non-null  object
dtypes: datetime64[ns](3), int64(1), object(9)
memory usage: 3.4+ MB
```

In [25]:
```python
crosstab_result = pd.crosstab(df_clean['Arrival.Station'], df_clean['Departure.Station'], margins=T
print(crosstab_result)
```

```
Departure.Station      Birmingham New Street  Bristol Temple Meads  \
Arrival.Station
Birmingham New Street                      0                     0
Bristol Temple Meads                       0                     0
Cardiff Central                            0                    16
Coventry                                  65                     0
Crewe                                      0                     0
Didcot                                     0                     0
Doncaster                                  0                     0
Durham                                     0                     0
Edinburgh                                 16                     0
Edinburgh Waverley                         0                     0
Leeds                                      0                     0
Leicester                                  0                     0
Liverpool Lime Street                    175                     0
London Euston                            125                     0
London Kings Cross                        17                     0
London Paddington                         32                     0
London St Pancras                        701                     0
London Waterloo                            0                     0
Manchester Piccadilly                    224                     0
Nottingham                                 0                     0
Nuneaton                                 219                     0
Oxford                                     0                     0
Peterborough                               0                     0
Reading                                   47                     0
Sheffield                                  0                     0
Stafford                                 190                     0
Swindon                                    0                     0
Tamworth                                 227                     0
Wakefield                                  0                     0
Warrington                                 0                     0
Wolverhampton                             32                     0
York                                      65                     0
All                                     2135                    16

Departure.Station      Edinburgh Waverley  Liverpool Lime Street  \
Arrival.Station
Birmingham New Street                   0                     14
Bristol Temple Meads                    0                      0
Cardiff Central                         0                      0
Coventry                                0                      0
Crewe                                   0                    193
Didcot                                  0                      0
Doncaster                               0                      0
Durham                                  0                      0
Edinburgh                               0                      0
Edinburgh Waverley                      0                      0
Leeds                                   0                     96
Leicester                               0                      0
Liverpool Lime Street                   0                      0
London Euston                           0                   1095
London Kings Cross                     51                      0
London Paddington                       0                     27
London St Pancras                       0                     31
London Waterloo                         0                      0
Manchester Piccadilly                   0                   2999
Nottingham                              0                      0
Nuneaton                                0                      0
Oxford                                  0                      0
Peterborough                            0                      0
Reading                                 0                      0
Sheffield                               0                    101
Stafford                                0                      0
Swindon                                 0                      0
Tamworth                                0                      0
Wakefield                               0                      0
Warrington                              0                      0
Wolverhampton                           0                      0
York                                    0                      0
All                                    51                   4556

Departure.Station      London Euston  London Kings Cross  London Paddington  \
Arrival.Station
Birmingham New Street           4208                   0                  0
Bristol Temple Meads               0                   0                  0
Cardiff Central                    0                   0                  0
Coventry                           0                   0                  0
Crewe                              0                   0                  0
Didcot                             0                   0                  0
Doncaster                          0                   0                  0
Durham                             0                   0                  0
Edinburgh                          0                   0                  0
Edinburgh Waverley                 0                 163                  0
Leeds                              0                   0                  0
Leicester                          0                   0                  0
Liverpool Lime Street              0                 144                 44
London Euston                      0                   0                  0
London Kings Cross                 0                   0                  0
London Paddington                  0                   0                  0
London St Pancras                  0                   0                  0
London Waterloo                    0                   0                 68
Manchester Piccadilly            712                   0                 30
Nottingham                         0                   0                  0
Nuneaton                           0                   0                  0
Oxford                            16                   0                485
Peterborough                       0                   0                  0
Reading                            0                   0               3873
Sheffield                          0                   0                  0
Stafford                           0                   0                  0
Swindon                            0                   0                  0
Tamworth                           0                   0                  0
Wakefield                          0                   0                  0
Warrington                         0                   0                  0
Wolverhampton                      0                   0                  0
York                              16                3922                  0
All                             4952                4229               4500

Departure.Station      London St Pancras  Manchester Piccadilly  Oxford  \
Arrival.Station
Birmingham New Street               3470                      0       0
Bristol Temple Meads                   0                      0     144
Cardiff Central                        0                      0       0
Coventry                               0                      0       0
Crewe                                  0                      0       0
Didcot                                 0                      0       0
Doncaster                              0                      0       0
Durham                                 0                      0       0
Edinburgh                              0                      0       0
Edinburgh Waverley                     0                      0       0
Leeds                                  0                    142       0
Leicester                            337                      0       0
Liverpool Lime Street                  0                   4626       0
London Euston                          0                    344       0
London Kings Cross                     0                     16       0
London Paddington                      0                    144       0
London St Pancras                      0                     16       0
London Waterloo                        0                      0       0
Manchester Piccadilly                  0                      0       0
Nottingham                             0                    158       0
Nuneaton                               0                      0       0
Oxford                                 0                      0       0
Peterborough                           0                      0       0
Reading                                0                      0       0
Sheffield                              0                    171       0
Stafford                               0                      0       0
Swindon                                0                      0       0
Tamworth                               0                      0       0
Wakefield                              0                      0       0
Warrington                             0                     15       0
Wolverhampton                         83                      0       0
York                                   0                     15       0
All                                 3890                   5647     144

Departure.Station      Reading  York    All
Arrival.Station
Birmingham New Street       32    16   7740
Bristol Temple Meads         0     0    144
Cardiff Central              0     0     16
Coventry                     0     0     65
Crewe                        0     0    193
Didcot                      48     0     48
Doncaster                    0   211    211
Durham                       0   257    257
Edinburgh                    0   137    153
Edinburgh Waverley           0    15    178
Leeds                        0    17    255
Leicester                    0     0    337
Liverpool Lime Street       16    15   5020
London Euston                0     0   1564
London Kings Cross           0     0     84
London Paddington          148     0    351
London St Pancras            0     0    748
London Waterloo              0     0     68
Manchester Piccadilly        0     0   3965
Nottingham                   0     0    158
Nuneaton                     0     0    219
Oxford                     122     0    623
Peterborough                 0   242    242
Reading                      0     0   3920
Sheffield                    0     0    272
Stafford                     0     0    190
Swindon                    228     0    228
Tamworth                     0     0    227
Wakefield                    0    15     15
Warrington                   0     0     15
Wolverhampton                0     0    115
York                         0     0   4018
All                        594   925  31639
```

2. Exploratory Data Analysis - Univariate

In [53]:
```python
df_clean['Price'] = df_clean['Price'].astype(float)
```

In [55]:
```python
import seaborn as sns
import matplotlib.pyplot as plt
```

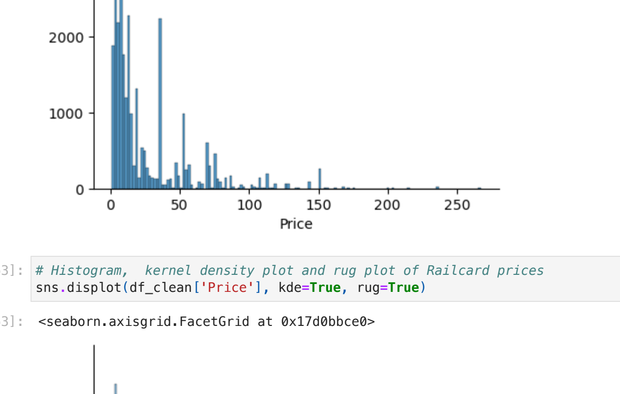
In [57]:
```python
# Univariate analysis of Price

df_clean['Price'].describe
df_clean['Price'].mode()
```

Out[57]:
```
0    3.0
Name: Price, dtype: float64
```

In [59]:
```python
df_clean['Price'].median()
```

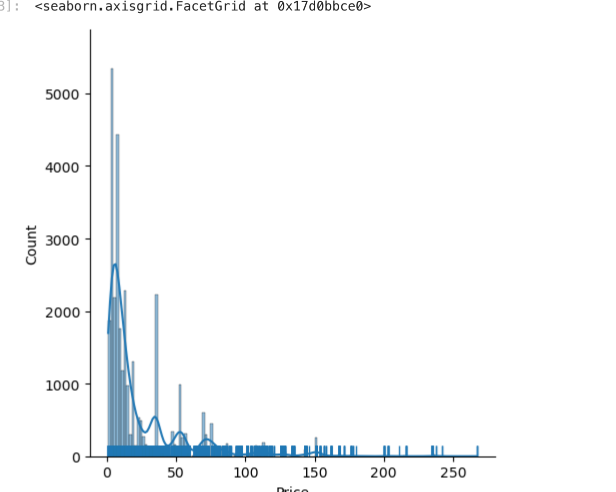
Out[59]:
```
11.0
```

In [61]:
```python
# Histogram of railcard prices
sns.displot(df_clean['Price'], label='Price Distribution')
plt.title("Distribution of Ticket Prices")
plt.xlabel("Price")
plt.ylabel("Count")
plt.legend()
plt.show()
```

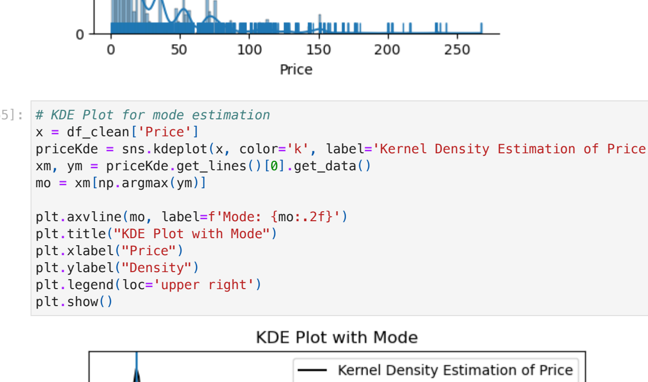


In [63]:
```python
# Histogram,  kernel density plot and rug plot of Railcard prices
sns.displot(df_clean['Price'], kde=True, rug=True)
```

Out[63]:
```
<seaborn.axisgrid.FacetGrid at 0x17d0bbce0>
```

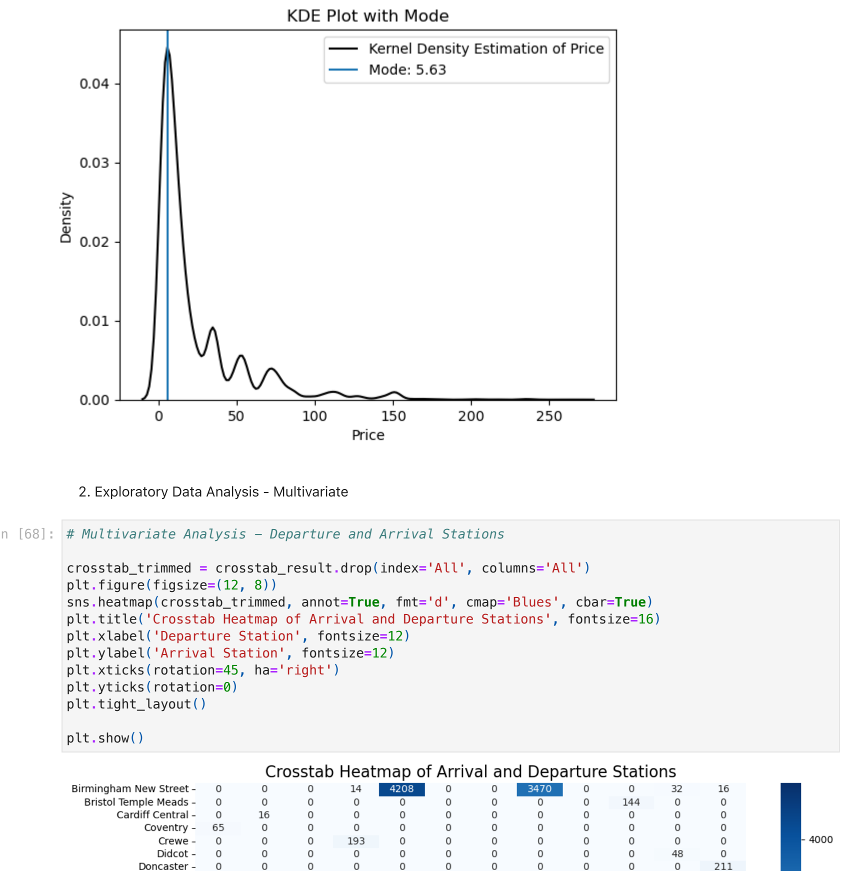


In [65]:
```python
# KDE Plot for mode estimation
x = df_clean['Price']
priceKde = sns.kdeplot(x, color='k', label='Kernel Density Estimation of Price')
xm, ym = priceKde.get_lines()[0].get_data()
mo = xm[np.argmax(ym)]

plt.axvline(mo, label=f'Mode: {mo:.2f}')
plt.title("KDE Plot with Mode")
plt.xlabel("Price")
plt.ylabel("Density")
plt.legend(loc='upper right')
plt.show()
```

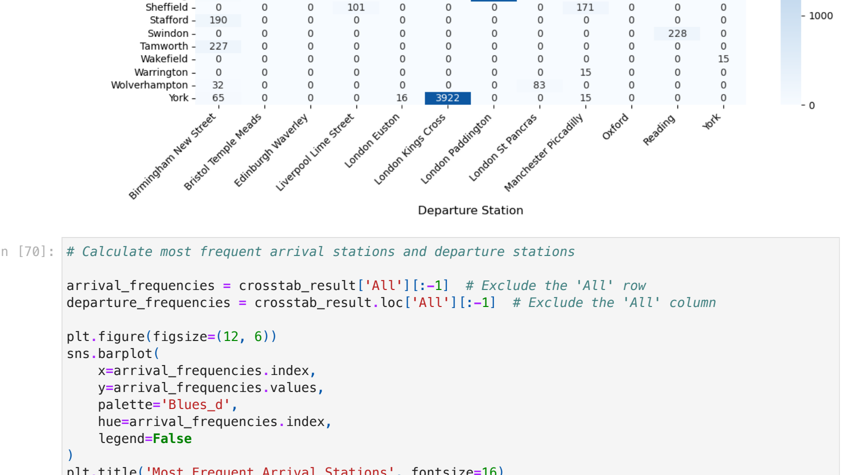


2. Exploratory Data Analysis - Multivariate

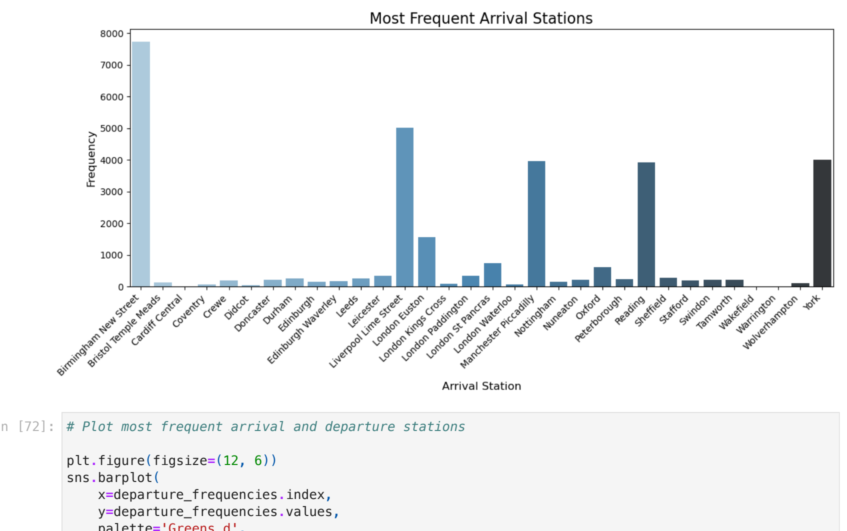
In [68]:
```python
# Multivariate Analysis - Departure and Arrival Stations

crosstab_trimmed = crosstab_result.drop(index='All', columns='All')
plt.figure(figsize=(12, 8))
sns.heatmap(crosstab_trimmed, annot=True, fmt='d', cmap='Blues', cbar=True)
plt.title('Crosstab Heatmap of Arrival and Departure Stations', fontsize=16)
plt.xlabel('Departure Station', fontsize=12)
plt.ylabel('Arrival Station', fontsize=12)
plt.xticks(rotation=45, ha='right')
plt.yticks(rotation=0)
plt.tight_layout()

plt.show()
```



In [70]:
```python
# Calculate most frequent arrival stations and departure stations

arrival_frequencies = crosstab_result['All'][:-1]  # Exclude the 'All' row
departure_frequencies = crosstab_result.loc['All'][:-1]  # Exclude the 'All' column

plt.figure(figsize=(12, 6))
sns.barplot(
    x=arrival_frequencies.index,
    y=arrival_frequencies.values,
    palette='Blues_d',
    hue=arrival_frequencies.index,
    legend=False
)
plt.title('Most Frequent Arrival Stations', fontsize=16)
plt.xlabel('Arrival Station', fontsize=12)
plt.ylabel('Frequency', fontsize=12)
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



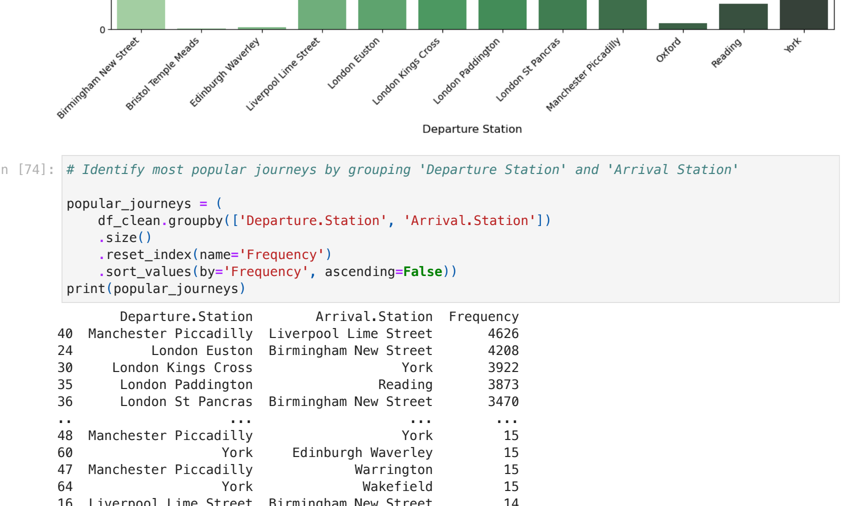
In [72]:
```python
# Plot most frequent arrival and departure stations

plt.figure(figsize=(12, 6))
sns.barplot(
    x=departure_frequencies.index,
    y=departure_frequencies.values,
    palette='Greens_d',
    hue=departure_frequencies.index
)
plt.title('Most Frequent Departure Stations', fontsize=16)
plt.xlabel('Departure Station', fontsize=12)
plt.ylabel('Frequency', fontsize=12)
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



In [74]:
```python
# Identify most popular journeys by grouping 'Departure Station' and 'Arrival Station'

popular_journeys = (
    df_clean.groupby(['Departure.Station', 'Arrival.Station'])
    .size()
    .reset_index(name='Frequency')
    .sort_values(by='Frequency', ascending=False))
print(popular_journeys)
```

```
        Departure.Station        Arrival.Station  Frequency
40  Manchester Piccadilly  Liverpool Lime Street       4626
24          London Euston  Birmingham New Street       4208
30     London Kings Cross                   York       3922
35      London Paddington                Reading       3873
36      London St Pancras  Birmingham New Street       3470
..                    ...                    ...        ...
48  Manchester Piccadilly                   York         15
60                   York     Edinburgh Waverley         15
47  Manchester Piccadilly             Warrington         15
64                   York              Wakefield         15
16  Liverpool Lime Street  Birmingham New Street         14

[65 rows x 3 columns]
```

In [76]:
```python
# Plot most popular journeys

top_journeys = popular_journeys.head()

plt.figure(figsize=(12, 6))
sns.barplot(
    data=top_journeys,
    x='Frequency',
    y=top_journeys.apply(
        lambda row: f"{row['Departure.Station']} → {row['Arrival.Station']}", axis=1
    ),
    palette='viridis',
hue=top_journeys.apply(
        lambda row: f"{row['Departure.Station']} → {row['Arrival.Station']}", axis=1
    )
)
plt.title('Most Popular Journeys', fontsize=16)
plt.xlabel('Frequency', fontsize=12)
plt.ylabel('Journey (Departure → Arrival)', fontsize=12)
plt.tight_layout()
plt.show()
```



In [78]:
```python
# Multivariate analysis: what is the most common cause for a refund request?

# What are the reasons for delay when a train is cancelled?

reasons_for_cancel = []

cancelled_journeys = df_clean[df_clean['Journey.Status'] == 'Cancelled']

for reason in cancelled_journeys['Reason.for.Delay']:
    if reason not in reasons_for_cancel:
        reasons_for_cancel.append(reason)

print(reasons_for_cancel)
```

```
['Technical Issue', 'Staffing', 'Staff', 'Signal Failure', 'Weather', 'Traffic']
```

```
In [80]: # What is the most frequent reason for delay when a train is cancelled?

         cancelled_journeys['Reason.for.Delay'].mode()
```

```
Out[80]: 0    Signal Failure
         Name: Reason.for.Delay, dtype: object
```

```
In [82]: pd.crosstab(df_clean['Journey.Status'], df_clean['Reason.for.Delay'])
```

Out[82]:

Reason.for.Delay	NA	Signal Failure	Staff	Staffing	Technical Issue	Traffic	Weather
Journey.Status							
Cancelled	0	518	216	237	234	227	445
Delayed	0	449	183	170	472	87	927
On Time	27474	0	0	0	0	0	0

```
In [84]: # Filter journey status to only ones that are delayed or cancelled

         cancelled_or_delayed = df_clean[(df_clean['Journey.Status']=='Cancelled') | (df_clean['Journey.Stat
```

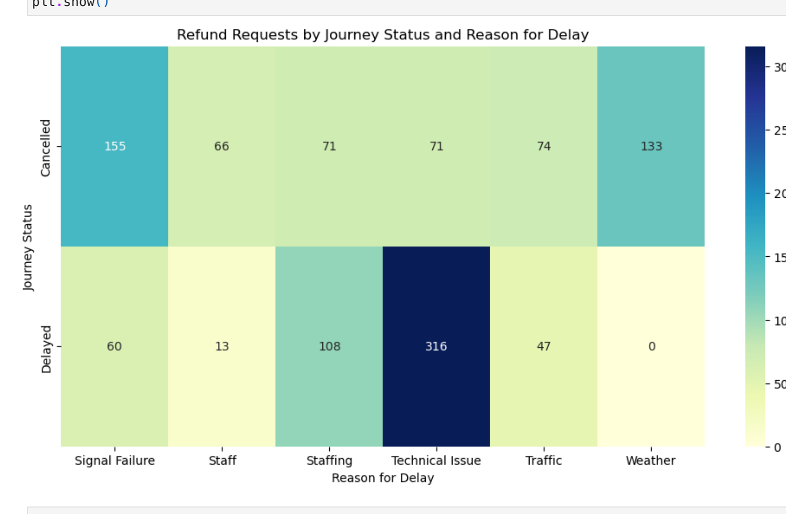
```
In [86]: # Filter for when there is a refund request

         refund_requested = df_clean[df_clean['Refund.Request']=='Yes']
```

```
In [88]: # Cross-tabulate canceled and delayed trains and when a refund is requested according to reason for

         pd.crosstab(cancelled_or_delayed['Journey.Status'], df_clean['Reason.for.Delay'], values=refund_rec
```

Out[88]:

Reason.for.Delay	Signal Failure	Staff	Staffing	Technical Issue	Traffic	Weather	All
Journey.Status							
Cancelled	155	66	71	71	74	133	570
Delayed	60	13	108	316	47	0	544
All	215	79	179	387	121	133	1114

```
In [90]: cross_tab_result_refund = pd.crosstab(cancelled_or_delayed['Journey.Status'], df_clean['Reason.for.

         plt.figure(figsize=(12, 6))
         sns.heatmap(cross_tab_result_refund, annot=True, fmt="d", cmap="YlGnBu")
         plt.title("Refund Requests by Journey Status and Reason for Delay")
         plt.xlabel("Reason for Delay")
         plt.ylabel("Journey Status")
         plt.show()
```




```
In [92]: # Refund request occurs most frequently when a train is delayed due to a technical issue. Why?
```

3. Add Column "DelayInMinutes"

```
In [95]: # Add a new column, DelayInMinutes, and calculate delay times in minutes


         delay_time = df_clean['Actual.Arrival']-df_clean['Scheduled.Arrival']
         print(delay_time)
```

```
0       0 days 00:00:00
1       0 days 00:05:00
2       0 days 00:00:00
3       0 days 00:00:00
4       0 days 00:00:00
              ...
31640   0 days 00:00:00
31641   0 days 00:00:00
31642   0 days 00:00:00
31643   0 days 00:00:00
31644   0 days 00:00:00
Length: 31639, dtype: timedelta64[ns]
```

```
In [97]: secs_per_min = 60
         secs_per_hour= 60*60
         df_clean['DelayInMinutes'] = delay_time.dt.total_seconds() / secs_per_min
```

```
In [99]: unique_values = df_clean['DelayInMinutes'].unique()
         print(unique_values)
```

```
[  0.   5.  17.  31.  24.  11.  18.  30.  59. 174.  54.  37.  32.  29.
  25.  12. 168.  27.  20.  19.  33.  40. 172. 113.  45.  14.  16.  48.
   4.  56.  65.  77.   9.  13.  50.  52.  53.  21.   8.  15.  39.  35.
 138.  34. 155. 167.  62. 143.  41.  58.  36. 180. 105.  55.  26.  38.
 152. 107.  49.  44.  10.   6.  92.  42. 136.  57.  51. 149. 131.   2.
  70.  81. 160.   1. 151.  22.   7. 158.   3.  43.  80. 161.  47. 100.
  60. 114.  89.  28. 175.  91.  66.  46. 134.  69. 163.  23. 178.  99.
  82. 176.  61. 147. 129. 159.  64. 162. 157. 179.  73. 137.  96.  75.
 156. 171. 101. 146. 144. 154. 103.  97.  63. 177. 132. 165.  88.  98.
  67. 173. 170. 125. 141.  85. 169.]
```

```
In [101… # Where the train arrived on time, set delay to NA.
         df_clean['DelayInMinutes'] = df_clean['DelayInMinutes'].replace(0.0, 'NA')
         df_clean.head()
```

Out[101…

	Payment.Method	Railcard	Ticket.Class	Ticket.Type	Price	Departure.Station	Arrival.Station	Departure	Sc
0	Contactless	Adult	Standard	Advance	43.0	London Paddington	Liverpool Lime Street	2024-01-01 11:00:00	
1	Credit Card	Adult	Standard	Advance	23.0	London Kings Cross	York	2024-01-01 09:45:00	
2	Credit Card	None	Standard	Advance	3.0	Liverpool Lime Street	Manchester Piccadilly	2024-01-02 18:15:00	
3	Credit Card	None	Standard	Advance	13.0	London Paddington	Reading	2024-01-01 21:30:00	
4	Contactless	None	Standard	Advance	76.0	Liverpool Lime Street	London Euston	2024-01-01 16:45:00	

```
In [103… df_clean['DelayInMinutes'].info()
```

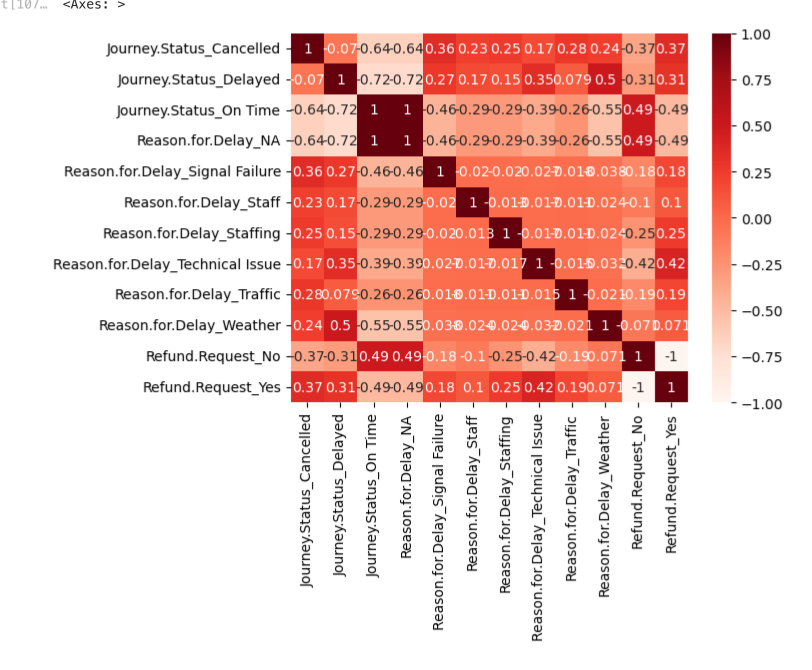
```
<class 'pandas.core.series.Series'>
Index: 31639 entries, 0 to 31644
Series name: DelayInMinutes
Non-Null Count  Dtype
--------------  -----
31639 non-null  object
dtypes: object(1)
memory usage: 1.5+ MB
```

```
In [105… dummies = pd.get_dummies(df_clean[['Journey.Status', 'Reason.for.Delay', 'Refund.Request']])
         correlation_matrix = dummies.corr()
         print(correlation_matrix)
```

```
                                 Journey.Status_Cancelled  \
Journey.Status_Cancelled                         1.000000
Journey.Status_Delayed                          -0.070116
Journey.Status_On Time                          -0.644992
Reason.for.Delay_NA                             -0.644992
Reason.for.Delay_Signal Failure                  0.358038
Reason.for.Delay_Staff                           0.230602
Reason.for.Delay_Staffing                        0.252723
Reason.for.Delay_Technical Issue                 0.174024
Reason.for.Delay_Traffic                         0.281247
Reason.for.Delay_Weather                         0.238852
Refund.Request_No                               -0.365800
Refund.Request_Yes                               0.365800

                                 Journey.Status_Delayed  \
Journey.Status_Cancelled                      -0.070116
Journey.Status_Delayed                         1.000000
Journey.Status_On Time                        -0.717084
Reason.for.Delay_NA                           -0.717084
Reason.for.Delay_Signal Failure                0.268732
Reason.for.Delay_Staff                         0.168567
Reason.for.Delay_Staffing                      0.152220
Reason.for.Delay_Technical Issue               0.347773
Reason.for.Delay_Traffic                       0.079147
Reason.for.Delay_Weather                       0.495950
Refund.Request_No                             -0.306836
Refund.Request_Yes                             0.306836

                                 Journey.Status_On Time  Reason.for.Delay_NA  \
Journey.Status_Cancelled                      -0.644992            -0.644992
Journey.Status_Delayed                        -0.717084            -0.717084
Journey.Status_On Time                         1.000000             1.000000
Reason.for.Delay_NA                            1.000000             1.000000
Reason.for.Delay_Signal Failure               -0.456032            -0.456032
Reason.for.Delay_Staff                        -0.290258            -0.290258
Reason.for.Delay_Staffing                     -0.293191            -0.293191
Reason.for.Delay_Technical Issue              -0.388012            -0.388012
Reason.for.Delay_Traffic                      -0.257142            -0.257142
Reason.for.Delay_Weather                      -0.546822            -0.546822
Refund.Request_No                              0.490646             0.490646
Refund.Request_Yes                            -0.490646            -0.490646

                                 Reason.for.Delay_Signal Failure  \
Journey.Status_Cancelled                                0.358038
Journey.Status_Delayed                                  0.268732
Journey.Status_On Time                                 -0.456032
Reason.for.Delay_NA                                    -0.456032
Reason.for.Delay_Signal Failure                         1.000000
Reason.for.Delay_Staff                                 -0.020067
Reason.for.Delay_Staffing                              -0.020269
Reason.for.Delay_Technical Issue                       -0.026825
Reason.for.Delay_Traffic                               -0.017777
Reason.for.Delay_Weather                               -0.037804
Refund.Request_No                                      -0.180274
Refund.Request_Yes                                      0.180274

                                 Reason.for.Delay_Staff  \
Journey.Status_Cancelled                       0.230602
Journey.Status_Delayed                         0.168567
Journey.Status_On Time                        -0.290258
Reason.for.Delay_NA                           -0.290258
Reason.for.Delay_Signal Failure               -0.020067
Reason.for.Delay_Staff                         1.000000
Reason.for.Delay_Staffing                     -0.012901
Reason.for.Delay_Technical Issue              -0.017073
Reason.for.Delay_Traffic                      -0.011315
Reason.for.Delay_Weather                      -0.024062
Refund.Request_No                             -0.099816
Refund.Request_Yes                             0.099816

                                 Reason.for.Delay_Staffing  \
Journey.Status_Cancelled                          0.252723
Journey.Status_Delayed                            0.152220
Journey.Status_On Time                           -0.293191
Reason.for.Delay_NA                              -0.293191
Reason.for.Delay_Signal Failure                  -0.020269
Reason.for.Delay_Staff                           -0.012901
Reason.for.Delay_Staffing                         1.000000
Reason.for.Delay_Technical Issue                 -0.017246
Reason.for.Delay_Traffic                         -0.011429
Reason.for.Delay_Weather                         -0.024305
Refund.Request_No                                -0.250593
Refund.Request_Yes                                0.250593

                                 Reason.for.Delay_Technical Issue  \
Journey.Status_Cancelled                                 0.174024
Journey.Status_Delayed                                   0.347773
Journey.Status_On Time                                  -0.388012
Reason.for.Delay_NA                                     -0.388012
Reason.for.Delay_Signal Failure                         -0.026825
Reason.for.Delay_Staff                                  -0.017073
Reason.for.Delay_Staffing                               -0.017246
Reason.for.Delay_Technical Issue                         1.000000
Reason.for.Delay_Traffic                                -0.015126
Reason.for.Delay_Weather                                -0.032165
Refund.Request_No                                       -0.420453
Refund.Request_Yes                                       0.420453

                                 Reason.for.Delay_Traffic  \
Journey.Status_Cancelled                         0.281247
Journey.Status_Delayed                           0.079147
Journey.Status_On Time                          -0.257142
Reason.for.Delay_NA                             -0.257142
Reason.for.Delay_Signal Failure                 -0.017777
Reason.for.Delay_Staff                          -0.011315
Reason.for.Delay_Staffing                       -0.011429
Reason.for.Delay_Technical Issue                -0.015126
Reason.for.Delay_Traffic                         1.000000
Reason.for.Delay_Weather                        -0.021316
Refund.Request_No                               -0.190202
Refund.Request_Yes                               0.190202

                                 Reason.for.Delay_Weather  Refund.Request_No  \
Journey.Status_Cancelled                         0.238852          -0.365800
Journey.Status_Delayed                           0.495950          -0.306836
Journey.Status_On Time                          -0.546822           0.490646
Reason.for.Delay_NA                             -0.546822           0.490646
Reason.for.Delay_Signal Failure                 -0.037804          -0.180274
Reason.for.Delay_Staff                          -0.024062          -0.099816
Reason.for.Delay_Staffing                       -0.024305          -0.250593
Reason.for.Delay_Technical Issue                -0.032165          -0.420453
Reason.for.Delay_Traffic                        -0.021316          -0.190202
Reason.for.Delay_Weather                         1.000000          -0.071307
Refund.Request_No                               -0.071307           1.000000
Refund.Request_Yes                               0.071307          -1.000000

                                 Refund.Request_Yes
Journey.Status_Cancelled                   0.365800
Journey.Status_Delayed                     0.306836
Journey.Status_On Time                    -0.490646
Reason.for.Delay_NA                       -0.490646
Reason.for.Delay_Signal Failure            0.180274
Reason.for.Delay_Staff                     0.099816
Reason.for.Delay_Staffing                  0.250593
Reason.for.Delay_Technical Issue           0.420453
Reason.for.Delay_Traffic                   0.190202
Reason.for.Delay_Weather                   0.071307
Refund.Request_No                         -1.000000
Refund.Request_Yes                         1.000000
```

```
In [107… sns.heatmap(correlation_matrix, annot=True, cmap='Reds')
```

```
Out[107… <Axes: >
```



4. Restrict "Journey.Status" to Cancelled and Delayed

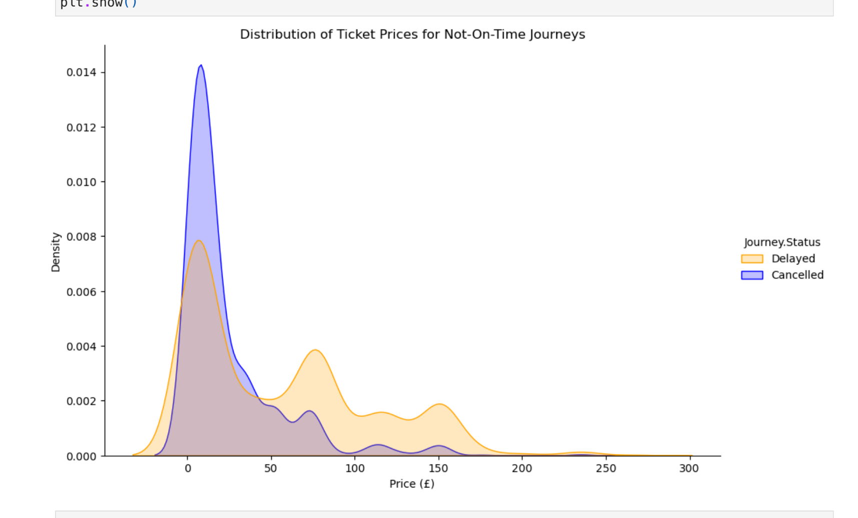
```
In [110… # Restrict to journeys that did not arrive on time and determine whether the ticket price was great
         journey_status_not_ontime = df_clean[(df_clean['Journey.Status']=='Cancelled') | (df_clean['Journey
         journey_status_not_ontime['MediumPrice'] = (journey_status_not_ontime['Price'] > 10) & (journey_sta

         # Map 'True/False' values to 'Yes/No' to replicate values in 'Refund Request'
         journey_status_not_ontime['MediumPrice'] = journey_status_not_ontime['MediumPrice'].map({True: 'Yes
```

```
In [112… journey_status_not_ontime.head()
```

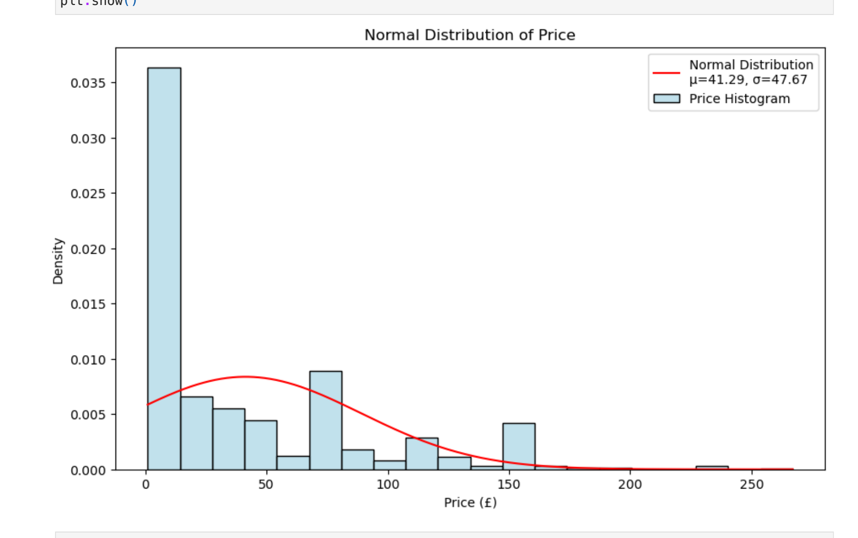
Out[112…

	Payment.Method	Railcard	Ticket.Class	Ticket.Type	Price	Departure.Station	Arrival.Station	Departure	S
1	Credit Card	Adult	Standard	Advance	23.0	London Kings Cross	York	2024-01-01 09:45:00	
8	Credit Card	None	Standard	Advance	37.0	London Euston	York	2024-01-01 00:00:00	
20	Debit Card	Adult	Standard	Advance	7.0	Birmingham New Street	Manchester Piccadilly	2024-01-01 11:15:00	
26	Credit Card	Senior	First Class	Advance	34.0	Oxford	Bristol Temple Meads	2024-01-01 14:15:00	
39	Credit Card	None	Standard	Advance	7.0	London Euston	Birmingham New Street	2024-01-02 02:15:00	

```
In [114… # plt.figure(figsize=(10, 6))

         # # Plot two distributions with explicit labels
         # sns.histplot(journey_status_not_ontime[journey_status_not_ontime['Journey.Status'] == 'Cancelled'
         #              bins=20, kde=True, label='Cancelled', color='blue', alpha=0.6)
         # sns.histplot(journey_status_not_ontime[journey_status_not_ontime['Journey.Status'] == 'Delayed'][
         #              bins=20, kde=True, label='Delayed', color='orange', alpha=0.6)

         # # Add titles and legend
         # plt.title("Distribution of Ticket Prices for Not-On-Time Journeys")
         # plt.xlabel("Price (£)")
         # plt.ylabel("Frequency")
         # plt.legend(title='Journey.Status')  # Explicitly providing labels
         # plt.show()

         sns.displot(
             data=journey_status_not_ontime,
             x='Price',
             hue='Journey.Status',
             kind='kde',  # To include KDE
             fill=True,  # Fill the area under the curve
             height=6,  # Set figure height
             aspect=1.5,  # Adjust width-to-height ratio
             palette={'Cancelled': 'blue', 'Delayed': 'orange'}  # Custom color palette
         )

         # Add titles and labels
         plt.title("Distribution of Ticket Prices for Not-On-Time Journeys")
         plt.xlabel("Price (£)")
         plt.ylabel("Density")
         plt.show()
```




```
In [116… from scipy.stats import norm

         price_data = journey_status_not_ontime['Price']
         # Calculate mean and standard deviation
         mean_price = price_data.mean()
         std_price = price_data.std()

         # Generate x values for the normal distribution
         x_values = np.linspace(price_data.min(), price_data.max(), 1000)

         # Compute the PDF of the normal distribution
         pdf = norm.pdf(x_values, loc=mean_price, scale=std_price)

         # Plot the histogram and the normal distribution
         plt.figure(figsize=(10, 6))
         sns.histplot(price_data, bins=20, kde=False, stat='density', color='lightblue', label='Price Histog
         plt.plot(x_values, pdf, color='red', label=f'Normal Distribution\nμ={mean_price:.2f}, σ={std_price:

         # Add labels and title
         plt.title("Normal Distribution of Price")
         plt.xlabel("Price (£)")
         plt.ylabel("Density")
         plt.legend()
         plt.show()
```




```
In [118… # Restrict to journeys that did not arrive on time and determine whether the ticket price was great
         journey_not_ontime = df_clean[(df_clean['Journey.Status']=='Cancelled') | (df_clean['Journey.Status
         journey_not_ontime['MediumPrice'] = (journey_not_ontime['Price'] > 10) & (journey_not_ontime['Price

         # Map 'True/False' values to 'Yes/No' to replicate values in 'Refund Request'
         journey_not_ontime['MediumPrice'] = journey_not_ontime['MediumPrice'].map({True: 'Yes', False: 'No'
```

```
In [120… journey_not_ontime.head()
```

Out[120…

	Payment.Method	Railcard	Ticket.Class	Ticket.Type	Price	Departure.Station	Arrival.Station	Departure	S
1	Credit Card	Adult	Standard	Advance	23.0	London Kings Cross	York	2024-01-01 09:45:00	
8	Credit Card	None	Standard	Advance	37.0	London Euston	York	2024-01-01 00:00:00	
20	Debit Card	Adult	Standard	Advance	7.0	Birmingham New Street	Manchester Piccadilly	2024-01-01 11:15:00	
26	Credit Card	Senior	First Class	Advance	34.0	Oxford	Bristol Temple Meads	2024-01-01 14:15:00	
39	Credit Card	None	Standard	Advance	7.0	London Euston	Birmingham New Street	2024-01-02 02:15:00	

```
In [122… journey_not_ontime.info()
```

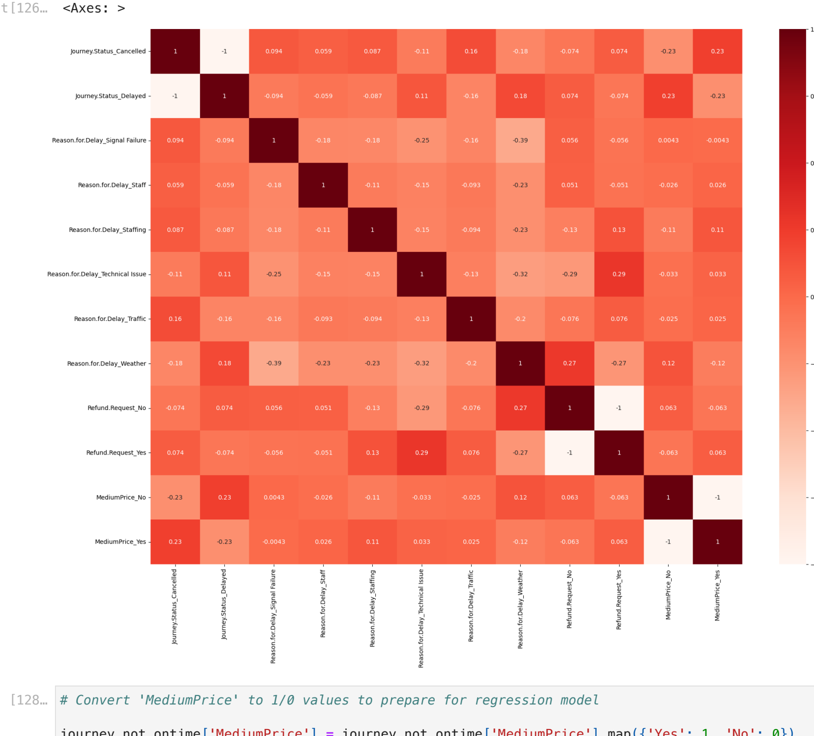
```
<class 'pandas.core.frame.DataFrame'>
Index: 4165 entries, 1 to 31637
Data columns (total 15 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Payment.Method     4165 non-null   object
 1   Railcard           4165 non-null   object
 2   Ticket.Class       4165 non-null   object
 3   Ticket.Type        4165 non-null   object
 4   Price              4165 non-null   float64
 5   Departure.Station  4165 non-null   object
 6   Arrival.Station    4165 non-null   object
 7   Departure          4165 non-null   datetime64[ns]
 8   Scheduled.Arrival  4165 non-null   datetime64[ns]
 9   Actual.Arrival     4165 non-null   datetime64[ns]
 10  Journey.Status     4165 non-null   object
 11  Reason.for.Delay   4165 non-null   object
 12  Refund.Request     4165 non-null   object
 13  DelayInMinutes     4165 non-null   object
 14  MediumPrice        4165 non-null   object
dtypes: datetime64[ns](3), float64(1), object(11)
memory usage: 520.6+ KB
```

```
In [124… # Get dummy values for object types to find correlation

         dummies_journey = pd.get_dummies(journey_not_ontime[['Journey.Status', 'Reason.for.Delay', 'Refund.
         correlation_matrix1 = dummies_journey.corr()
         print(correlation_matrix1)
```

```
                                 Journey.Status_Cancelled  \
Journey.Status_Cancelled                         1.000000
Journey.Status_Delayed                          -1.000000
Reason.for.Delay_Signal Failure                  0.093958
Reason.for.Delay_Staff                           0.059330
Reason.for.Delay_Staffing                        0.087075
Reason.for.Delay_Technical Issue                -0.108247
Reason.for.Delay_Traffic                         0.156254
Reason.for.Delay_Weather                        -0.177933
Refund.Request_No                               -0.074092
Refund.Request_Yes                               0.074092
MediumPrice_No                                  -0.233021
MediumPrice_Yes                                  0.233021

                                 Journey.Status_Delayed  \
Journey.Status_Cancelled                      -1.000000
Journey.Status_Delayed                         1.000000
Reason.for.Delay_Signal Failure               -0.093958
Reason.for.Delay_Staff                        -0.059330
Reason.for.Delay_Staffing                     -0.087075
Reason.for.Delay_Technical Issue               0.108247
Reason.for.Delay_Traffic                      -0.156254
Reason.for.Delay_Weather                       0.177933
Refund.Request_No                              0.074092
Refund.Request_Yes                            -0.074092
MediumPrice_No                                 0.233021
MediumPrice_Yes                               -0.233021

                                 Reason.for.Delay_Signal Failure  \
Journey.Status_Cancelled                                0.093958
Journey.Status_Delayed                                 -0.093958
Reason.for.Delay_Signal Failure                         1.000000
Reason.for.Delay_Staff                                 -0.178987
Reason.for.Delay_Staffing                              -0.180964
Reason.for.Delay_Technical Issue                       -0.248428
Reason.for.Delay_Traffic                               -0.157019
Reason.for.Delay_Weather                               -0.385403
Refund.Request_No                                       0.056065
Refund.Request_Yes                                     -0.056065
MediumPrice_No                                          0.004290
MediumPrice_Yes                                        -0.004290

                                 Reason.for.Delay_Staff  \
Journey.Status_Cancelled                       0.059330
Journey.Status_Delayed                        -0.059330
Reason.for.Delay_Signal Failure               -0.178987
Reason.for.Delay_Staff                         1.000000
Reason.for.Delay_Staffing                     -0.107119
Reason.for.Delay_Technical Issue              -0.147053
Reason.for.Delay_Traffic                      -0.092945
Reason.for.Delay_Weather                      -0.228133
Refund.Request_No                              0.051087
Refund.Request_Yes                            -0.051087
MediumPrice_No                                -0.026015
MediumPrice_Yes                                0.026015

                                 Reason.for.Delay_Staffing  \
Journey.Status_Cancelled                          0.087075
Journey.Status_Delayed                           -0.087075
Reason.for.Delay_Signal Failure                  -0.180964
Reason.for.Delay_Staff                           -0.107119
Reason.for.Delay_Staffing                         1.000000
Reason.for.Delay_Technical Issue                 -0.148678
Reason.for.Delay_Traffic                         -0.093972
Reason.for.Delay_Weather                         -0.230654
Refund.Request_No                                -0.128129
Refund.Request_Yes                                0.128129
MediumPrice_No                                   -0.109079
MediumPrice_Yes                                   0.109079

                                 Reason.for.Delay_Technical Issue  \
Journey.Status_Cancelled                                -0.108247
Journey.Status_Delayed                                   0.108247
Reason.for.Delay_Signal Failure                         -0.248428
Reason.for.Delay_Staff                                  -0.147053
Reason.for.Delay_Staffing                               -0.148678
Reason.for.Delay_Technical Issue                         1.000000
Reason.for.Delay_Traffic                                -0.129005
Reason.for.Delay_Weather                                -0.316642
Refund.Request_No                                       -0.286489
Refund.Request_Yes                                       0.286489
MediumPrice_No                                          -0.033389
MediumPrice_Yes                                          0.033389

                                 Reason.for.Delay_Traffic  \
Journey.Status_Cancelled                         0.156254
Journey.Status_Delayed                          -0.156254
Reason.for.Delay_Signal Failure                 -0.157019
Reason.for.Delay_Staff                          -0.092945
Reason.for.Delay_Staffing                       -0.093972
Reason.for.Delay_Technical Issue                -0.129005
Reason.for.Delay_Traffic                         1.000000
Reason.for.Delay_Weather                        -0.200133
Refund.Request_No                               -0.076047
Refund.Request_Yes                               0.076047
MediumPrice_No                                  -0.024941
MediumPrice_Yes                                  0.024941

                                 Reason.for.Delay_Weather  Refund.Request_No  \
Journey.Status_Cancelled                        -0.177933          -0.074092
Journey.Status_Delayed                           0.177933           0.074092
Reason.for.Delay_Signal Failure                 -0.385403           0.056065
Reason.for.Delay_Staff                          -0.228133           0.051087
Reason.for.Delay_Staffing                       -0.230654          -0.128129
Reason.for.Delay_Technical Issue                -0.316642          -0.286489
Reason.for.Delay_Traffic                        -0.200133          -0.076047
Reason.for.Delay_Weather                         1.000000           0.270016
Refund.Request_No                                0.270016           1.000000
Refund.Request_Yes                              -0.270016          -1.000000
MediumPrice_No                                   0.122014           0.063087
MediumPrice_Yes                                 -0.122014          -0.063087

                                 Refund.Request_Yes  MediumPrice_No  \
Journey.Status_Cancelled                   0.074092       -0.233021
Journey.Status_Delayed                    -0.074092        0.233021
Reason.for.Delay_Signal Failure           -0.056065        0.004290
Reason.for.Delay_Staff                    -0.051087       -0.026015
Reason.for.Delay_Staffing                  0.128129       -0.109079
Reason.for.Delay_Technical Issue           0.286489       -0.033389
Reason.for.Delay_Traffic                   0.076047       -0.024941
Reason.for.Delay_Weather                  -0.270016        0.122014
Refund.Request_No                         -1.000000        0.063087
Refund.Request_Yes                         1.000000       -0.063087
MediumPrice_No                            -0.063087        1.000000
MediumPrice_Yes                            0.063087       -1.000000

                                 MediumPrice_Yes
Journey.Status_Cancelled                0.233021
Journey.Status_Delayed                 -0.233021
Reason.for.Delay_Signal Failure        -0.004290
Reason.for.Delay_Staff                  0.026015
Reason.for.Delay_Staffing               0.109079
Reason.for.Delay_Technical Issue        0.033389
Reason.for.Delay_Traffic                0.024941
Reason.for.Delay_Weather               -0.122014
Refund.Request_No                      -0.063087
Refund.Request_Yes                      0.063087
MediumPrice_No                         -1.000000
MediumPrice_Yes                         1.000000
```

```
In [126… plt.figure(figsize=(22, 16))
         sns.heatmap(correlation_matrix1, annot=True, cmap='Reds')
```

```
Out[126… <Axes: >
```



```
In [128… # Convert 'MediumPrice' to 1/0 values to prepare for regression model

         journey_not_ontime['MediumPrice'] = journey_not_ontime['MediumPrice'].map({'Yes': 1, 'No': 0})

         journey_not_ontime.head()
```

Out[128…

	Payment.Method	Railcard	Ticket.Class	Ticket.Type	Price	Departure.Station	Arrival.Station	Departure	S
1	Credit Card	Adult	Standard	Advance	23.0	London Kings Cross	York	2024-01-01 09:45:00	
8	Credit Card	None	Standard	Advance	37.0	London Euston	York	2024-01-01 00:00:00	
20	Debit Card	Adult	Standard	Advance	7.0	Birmingham New Street	Manchester Piccadilly	2024-01-01 11:15:00	
26	Credit Card	Senior	First Class	Advance	34.0	Oxford	Bristol Temple Meads	2024-01-01 14:15:00	
39	Credit Card	None	Standard	Advance	7.0	London Euston	Birmingham New Street	2024-01-02 02:15:00	

```
In [130… journey_not_ontime['DelayInMinutes'] = pd.to_numeric(journey_not_ontime['DelayInMinutes'], errors='
```

```
In [132… journey_not_ontime.head()
```

Out[132…

	Payment.Method	Railcard	Ticket.Class	Ticket.Type	Price	Departure.Station	Arrival.Station	Departure	S
1	Credit Card	Adult	Standard	Advance	23.0	London Kings Cross	York	2024-01-01 09:45:00	
8	Credit Card	None	Standard	Advance	37.0	London Euston	York	2024-01-01 00:00:00	
20	Debit Card	Adult	Standard	Advance	7.0	Birmingham New Street	Manchester Piccadilly	2024-01-01 11:15:00	
26	Credit Card	Senior	First Class	Advance	34.0	Oxford	Bristol Temple Meads	2024-01-01 14:15:00	
39	Credit Card	None	Standard	Advance	7.0	London Euston	Birmingham New Street	2024-01-02 02:15:00	

```
In [134… journey_not_ontime['DelayInMinutes'] = journey_not_ontime['DelayInMinutes'].fillna(0)
```

```
In [136… journey_not_ontime.head()
```

Out[136…

	Payment.Method	Railcard	Ticket.Class	Ticket.Type	Price	Departure.Station	Arrival.Station	Departure	S
1	Credit Card	Adult	Standard	Advance	23.0	London Kings Cross	York	2024-01-01 09:45:00	
8	Credit Card	None	Standard	Advance	37.0	London Euston	York	2024-01-01 00:00:00	
20	Debit Card	Adult	Standard	Advance	7.0	Birmingham New Street	Manchester Piccadilly	2024-01-01 11:15:00	
26	Credit Card	Senior	First Class	Advance	34.0	Oxford	Bristol Temple Meads	2024-01-01 14:15:00	
39	Credit Card	None	Standard	Advance	7.0	London Euston	Birmingham New Street	2024-01-02 02:15:00	

```
In [138… journey_not_ontime.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 4165 entries, 1 to 31637
Data columns (total 15 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Payment.Method     4165 non-null   object
 1   Railcard           4165 non-null   object
 2   Ticket.Class       4165 non-null   object
 3   Ticket.Type        4165 non-null   object
 4   Price              4165 non-null   float64
 5   Departure.Station  4165 non-null   object
 6   Arrival.Station    4165 non-null   object
 7   Departure          4165 non-null   datetime64[ns]
 8   Scheduled.Arrival  4165 non-null   datetime64[ns]
 9   Actual.Arrival     4165 non-null   datetime64[ns]
 10  Journey.Status     4165 non-null   object
 11  Reason.for.Delay   4165 non-null   object
 12  Refund.Request     4165 non-null   object
 13  DelayInMinutes     4165 non-null   float64
 14  MediumPrice        4165 non-null   int64
dtypes: datetime64[ns](3), float64(2), int64(1), object(9)
memory usage: 520.6+ KB
```

```
In [140… journey_not_ontime.isnull().sum()
```

Out[140…

```
Payment.Method       0
Railcard             0
Ticket.Class         0
Ticket.Type          0
Price                0
Departure.Station    0
Arrival.Station      0
Departure            0
Scheduled.Arrival    0
Actual.Arrival       0
Journey.Status       0
Reason.for.Delay     0
Refund.Request       0
DelayInMinutes       0
MediumPrice          0
dtype: int64
```

In [142…

```python
journey_not_ontime.head()
```

Out[142…

	Payment.Method	Railcard	Ticket.Class	Ticket.Type	Price	Departure.Station	Arrival.Station	Departure
1	Credit Card	Adult	Standard	Advance	23.0	London Kings Cross	York	2024-01-01 09:45:00
8	Credit Card	None	Standard	Advance	37.0	London Euston	York	2024-01-01 00:00:00
20	Debit Card	Adult	Standard	Advance	7.0	Birmingham New Street	Manchester Piccadilly	2024-01-01 11:15:00
26	Credit Card	Senior	First Class	Advance	34.0	Oxford	Bristol Temple Meads	2024-01-01 14:15:00
39	Credit Card	None	Standard	Advance	7.0	London Euston	Birmingham New Street	2024-01-02 02:15:00

In [144…

```python
# # Logistic regression model using sklearn

# from sklearn.model_selection import train_test_split
# from sklearn.linear_model import LogisticRegression
# from sklearn.metrics import classification_report, accuracy_score

# X = journey_not_ontime[['MediumPrice']]
# y = journey_not_ontime['Refund.Request']

# X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)

# # Fit the logistic regression model
# log_model = LogisticRegression()
# log_model.fit(X_train, y_train)

# # Make predictions
# y_pred = log_model.predict(X_test)

# # Evaluate the model
# print("Accuracy:", accuracy_score(y_test, y_pred))
# print(classification_report(y_test, y_pred, zero_division=0))
```

4. Logistic Regression Model with Single Predictor "MediumPrice"

In [147…

```python
# Logistic regression model using statsmodels

import statsmodels.api as sm
```

In [149…

```python
# Convert object types to categorical

journey_not_ontime = journey_not_ontime.apply(lambda col: col.astype('category') if col.dtypes == '
```

In [151…

```python
print(journey_not_ontime.dtypes)
```

```
Payment.Method             category
Railcard                   category
Ticket.Class               category
Ticket.Type                category
Price                       float64
Departure.Station          category
Arrival.Station            category
Departure            datetime64[ns]
Scheduled.Arrival    datetime64[ns]
Actual.Arrival       datetime64[ns]
Journey.Status             category
Reason.for.Delay           category
Refund.Request             category
DelayInMinutes              float64
MediumPrice                   int64
dtype: object
```

In [153…

```python
# Concatenate category values to dataset
cat_vars = journey_not_ontime.select_dtypes(include=['category']).columns

for var in cat_vars:
    cat_list = pd.get_dummies(journey_not_ontime[var], prefix=var, drop_first=True)
    journey_not_ontime = pd.concat([journey_not_ontime, cat_list], axis=1)
```

In [155…

```python
data_vars=journey_not_ontime.columns.values.tolist()
to_keep=[i for i in data_vars if i not in cat_vars]
```

In [157…

```python
# Final dataset for logistic regression model

journey_final=journey_not_ontime[to_keep]
journey_final.columns.values
```

Out[157…

```
array(['Price', 'Departure', 'Scheduled.Arrival', 'Actual.Arrival',
       'DelayInMinutes', 'MediumPrice', 'Payment.Method_Credit Card',
       'Payment.Method_Debit Card', 'Railcard_Disabled', 'Railcard_None',
       'Railcard_Senior', 'Ticket.Class_Standard', 'Ticket.Type_Anytime',
       'Ticket.Type_Off-Peak', 'Departure.Station_Edinburgh Waverley',
       'Departure.Station_Liverpool Lime Street',
       'Departure.Station_London Euston',
       'Departure.Station_London Kings Cross',
       'Departure.Station_London Paddington',
       'Departure.Station_London St Pancras',
       'Departure.Station_Manchester Piccadilly',
       'Departure.Station_Oxford', 'Departure.Station_Reading',
       'Departure.Station_York', 'Arrival.Station_Bristol Temple Meads',
       'Arrival.Station_Coventry', 'Arrival.Station_Crewe',
       'Arrival.Station_Didcot', 'Arrival.Station_Doncaster',
       'Arrival.Station_Durham', 'Arrival.Station_Edinburgh',
       'Arrival.Station_Edinburgh Waverley', 'Arrival.Station_Leeds',
       'Arrival.Station_Leicester',
       'Arrival.Station_Liverpool Lime Street',
       'Arrival.Station_London Euston',
       'Arrival.Station_London Kings Cross',
       'Arrival.Station_London Paddington',
       'Arrival.Station_London St Pancras',
       'Arrival.Station_London Waterloo',
       'Arrival.Station_Manchester Piccadilly',
       'Arrival.Station_Nottingham', 'Arrival.Station_Nuneaton',
       'Arrival.Station_Oxford', 'Arrival.Station_Peterborough',
       'Arrival.Station_Reading', 'Arrival.Station_Sheffield',
       'Arrival.Station_Stafford', 'Arrival.Station_Swindon',
       'Arrival.Station_Tamworth', 'Arrival.Station_Wakefield',
       'Arrival.Station_Wolverhampton', 'Arrival.Station_York',
       'Journey.Status_Delayed', 'Reason.for.Delay_Staff',
       'Reason.for.Delay_Staffing', 'Reason.for.Delay_Technical Issue',
       'Reason.for.Delay_Traffic', 'Reason.for.Delay_Weather',
       'Refund.Request_Yes'], dtype=object)
```

In [159…

```python
journey_final.head()
```

Out[159…

	Price	Departure	Scheduled.Arrival	Actual.Arrival	DelayInMinutes	MediumPrice	Payment.Method_Credit Card
1	23.0	2024-01-01 09:45:00	2024-01-01 11:35:00	2024-01-01 11:40:00	5.0	1	True
8	37.0	2024-01-01 00:00:00	2024-01-01 01:50:00	2024-01-01 02:07:00	17.0	0	True
20	7.0	2024-01-01 11:15:00	2024-01-01 12:35:00	2024-01-01 13:06:00	31.0	0	False
26	34.0	2024-01-01 14:15:00	2024-01-01 15:30:00	2024-01-01 15:54:00	24.0	0	True
39	7.0	2024-01-02 02:15:00	2024-01-02 03:35:00	2024-01-02 03:35:00	0.0	0	True

5 rows × 60 columns

In [161…

```python
#fit small model with one predictor
X_1 = journey_final['MediumPrice'].values.reshape(-1, 1)
X_1 = sm.add_constant(X_1)

#set response variable
y = journey_final.loc[:, journey_final.columns == 'Refund.Request_Yes']
```

In [163…

```python
logit_model=sm.Logit(y,X_1)
result1=logit_model.fit()
print(result1.summary2())
```

```
Optimization terminated successfully.
         Current function value: 0.578797
         Iterations 5
                             Results: Logit
==================================================================
Model:              Logit              Method:            MLE
Dependent Variable: Refund.Request_Yes Pseudo R-squared: 0.003
Date:               2024-11-22 00:08   AIC:               4825.3797
No. Observations:   4165               BIC:               4838.0487
Df Model:           1                  Log-Likelihood:    -2410.7
Df Residuals:       4163               LL-Null:           -2418.7
Converged:          1.0000             LLR p-value:       6.1887e-05
No. Iterations:     5.0000             Scale:             1.0000
--------------------------------------------------------------------
            Coef.    Std.Err.      z       P>|z|     [0.025    0.975]
--------------------------------------------------------------------
const      -1.0753     0.0393   -27.3720   0.0000   -1.1523   -0.9983
x1          0.3540     0.0872     4.0583   0.0000    0.1830    0.5250
==================================================================
```

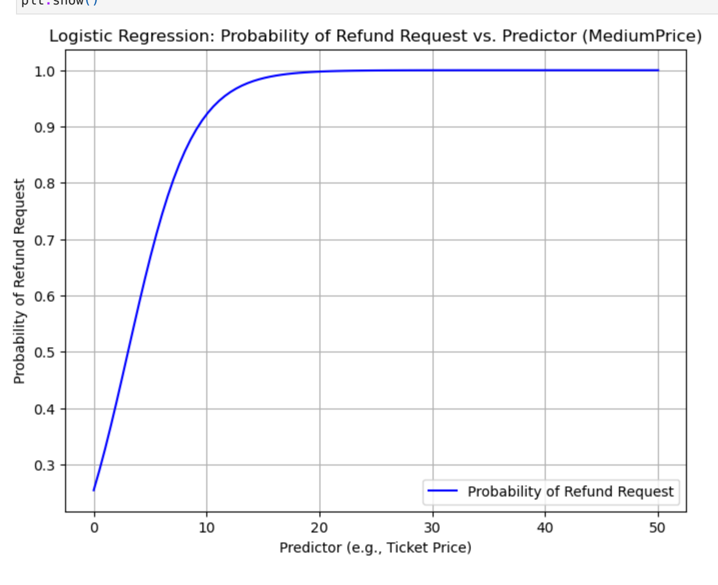
In [165…

```python
print(result1.params)
```

```
const   -1.075328
x1       0.354010
dtype: float64
```

In [167…

```python
# Plot model results

# import numpy as np
# import matplotlib.pyplot as plt

# Model coefficients
intercept = -1.0753
coef_x1 = 0.3540

# Define the logistic function
def logistic_function(x):
    return 1 / (1 + np.exp(-(intercept + coef_x1 * x)))

# Generate a range of values for the predictor (e.g., ticket prices)
x1_values = np.linspace(0, 50, 100)  # You can change the range based on your data

# Calculate the predicted probabilities for these x1 values
predicted_probabilities = logistic_function(x1_values)

# Create the plot
plt.figure(figsize=(8, 6))
plt.plot(x1_values, predicted_probabilities, label='Probability of Refund Request', color='blue')
plt.title('Logistic Regression: Probability of Refund Request vs. Predictor (MediumPrice)')
plt.xlabel('Predictor (e.g., Ticket Price)')
plt.ylabel('Probability of Refund Request')
plt.grid(True)
plt.legend()
plt.show()
```



5. Fit Regression Models to To_Predict using MavenRail Predictors

In [170…

```python
# Clean and format To_Predict data to conform with MavenRail data
```

In [172…

```python
to_predict = pd.read_csv('ToPredict.csv')
to_predict.head()
```

Out[172…

	Payment.Method	Railcard	Ticket.Class	Ticket.Type	Price	Departure.Station	Arrival.Station	Departure	Sc
0	Debit Card	NaN	First Class	Advance	54	London St Pancras	Birmingham New Street	2024-01-04 17:45	20
1	Credit Card	NaN	Standard	Advance	7	London Euston	Birmingham New Street	2024-01-05 08:15	20
2	Debit Card	NaN	Standard	Off-Peak	113	Liverpool Lime Street	London Euston	2024-01-09 15:30	20
3	Contactless	Adult	Standard	Off-Peak	3	Liverpool Lime Street	Manchester Piccadilly	2024-01-31 05:45	20
4	Credit Card	NaN	Standard	Off-Peak	4	Manchester Piccadilly	Liverpool Lime Street	2024-02-10 16:00	20

In [174…

```python
# Remove null values. Convert 'nan' for 'No reason' entry in 'Reason for delay' to 'none'.
to_predict['Reason.for.Delay'] = to_predict['Reason.for.Delay'].fillna('NA')
```

In [176…

```python
# Remove null values. Convert 'nan' for 'no railcard' entry in 'Railcards' to 'none'.
to_predict['Railcard'] = to_predict['Railcard'].fillna('None')
```

In [178…

```python
to_predict['Actual.Arrival'] = to_predict['Actual.Arrival'].fillna(to_predict['Scheduled.Arrival'])
```

In [180…

```python
to_predict.isnull().sum()
```

Out[180…

```
Payment.Method       0
Railcard             0
Ticket.Class         0
Ticket.Type          0
Price                0
Departure.Station    0
Arrival.Station      0
Departure            0
Scheduled.Arrival    0
Actual.Arrival       0
Journey.Status       0
Reason.for.Delay     0
dtype: int64
```

In [182…

```python
# Convert string values for 'Actual Arrival', 'Departure', and 'Scheduled Arrival' to datetime64
to_predict['Actual.Arrival'] = pd.to_datetime(to_predict['Actual.Arrival'], format='%Y-%m-%d %H:%M'
to_predict['Departure'] = pd.to_datetime(to_predict['Departure'], format='%Y-%m-%d %H:%M' )
to_predict['Scheduled.Arrival'] = pd.to_datetime(to_predict['Scheduled.Arrival'], format='%Y-%m-%d
```

In [184…

```python
to_predict.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8 entries, 0 to 7
Data columns (total 12 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Payment.Method     8 non-null      object
 1   Railcard           8 non-null      object
 2   Ticket.Class       8 non-null      object
 3   Ticket.Type        8 non-null      object
 4   Price              8 non-null      int64
 5   Departure.Station  8 non-null      object
 6   Arrival.Station    8 non-null      object
 7   Departure          8 non-null      datetime64[ns]
 8   Scheduled.Arrival  8 non-null      datetime64[ns]
 9   Actual.Arrival     8 non-null      datetime64[ns]
 10  Journey.Status     8 non-null      object
 11  Reason.for.Delay   8 non-null      object
dtypes: datetime64[ns](3), int64(1), object(8)
memory usage: 900.0+ bytes
```

In [186…

```python
to_predict.head()
```

Out[186…

	Payment.Method	Railcard	Ticket.Class	Ticket.Type	Price	Departure.Station	Arrival.Station	Departure
0	Debit Card	None	First Class	Advance	54	London St Pancras	Birmingham New Street	2024-01-04 17:45:00
1	Credit Card	None	Standard	Advance	7	London Euston	Birmingham New Street	2024-01-05 08:15:00
2	Debit Card	None	Standard	Off-Peak	113	Liverpool Lime Street	London Euston	2024-01-09 15:30:00
3	Contactless	Adult	Standard	Off-Peak	3	Liverpool Lime Street	Manchester Piccadilly	2024-01-31 05:45:00
4	Credit Card	None	Standard	Off-Peak	4	Manchester Piccadilly	Liverpool Lime Street	2024-02-10 16:00:00

In [188…

```python
# Add a new column, DelayInMinutes, and calculate delay times in minutes

delay_time = to_predict['Actual.Arrival']-to_predict['Scheduled.Arrival']

secs_per_min = 60
secs_per_hour= 60*60
to_predict['DelayInMinutes'] = delay_time.dt.total_seconds() / secs_per_min
```

In [190…

```python
to_predict.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8 entries, 0 to 7
Data columns (total 13 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Payment.Method     8 non-null      object
 1   Railcard           8 non-null      object
 2   Ticket.Class       8 non-null      object
 3   Ticket.Type        8 non-null      object
 4   Price              8 non-null      int64
 5   Departure.Station  8 non-null      object
 6   Arrival.Station    8 non-null      object
 7   Departure          8 non-null      datetime64[ns]
 8   Scheduled.Arrival  8 non-null      datetime64[ns]
 9   Actual.Arrival     8 non-null      datetime64[ns]
 10  Journey.Status     8 non-null      object
 11  Reason.for.Delay   8 non-null      object
 12  DelayInMinutes     8 non-null      float64
dtypes: datetime64[ns](3), float64(1), int64(1), object(8)
memory usage: 964.0+ bytes
```

In [192…

```python
# Restrict to journeys that did not arrive on time and determine whether the ticket price was great
predict_journey_not_ontime = to_predict[(to_predict['Journey.Status']=='Cancelled') | (to_predict['
predict_journey_not_ontime['MediumPrice'] = (predict_journey_not_ontime['Price'] > 10) & (predict_j

# Map 'True/False' values to 'Yes/No' to replicate values in 'Refund Request'
predict_journey_not_ontime['MediumPrice'] = predict_journey_not_ontime['MediumPrice'].map({True: 'Y
```

In [194…

```python
predict_journey_not_ontime.head()
```

Out[194…

	Payment.Method	Railcard	Ticket.Class	Ticket.Type	Price	Departure.Station	Arrival.Station	Departure
2	Debit Card	None	Standard	Off-Peak	113	Liverpool Lime Street	London Euston	2024-01-09 15:30:00
3	Contactless	Adult	Standard	Off-Peak	3	Liverpool Lime Street	Manchester Piccadilly	2024-01-31 05:45:00
4	Credit Card	None	Standard	Off-Peak	4	Manchester Piccadilly	Liverpool Lime Street	2024-02-10 16:00:00
5	Contactless	None	Standard	Advance	3	Manchester Piccadilly	Liverpool Lime Street	2024-02-25 15:45:00
6	Debit Card	None	Standard	Off-Peak	126	Manchester Piccadilly	London Euston	2024-03-20 15:30:00

In [196…

```python
predict_journey_not_ontime.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 6 entries, 2 to 7
Data columns (total 14 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Payment.Method     6 non-null      object
 1   Railcard           6 non-null      object
 2   Ticket.Class       6 non-null      object
 3   Ticket.Type        6 non-null      object
 4   Price              6 non-null      int64
 5   Departure.Station  6 non-null      object
 6   Arrival.Station    6 non-null      object
 7   Departure          6 non-null      datetime64[ns]
 8   Scheduled.Arrival  6 non-null      datetime64[ns]
 9   Actual.Arrival     6 non-null      datetime64[ns]
 10  Journey.Status     6 non-null      object
 11  Reason.for.Delay   6 non-null      object
 12  DelayInMinutes     6 non-null      float64
 13  MediumPrice        6 non-null      object
dtypes: datetime64[ns](3), float64(1), int64(1), object(9)
memory usage: 720.0+ bytes
```

In [198…

```python
# Convert 'MediumPrice' to 1/0 values to prepare for regression model

predict_journey_not_ontime['MediumPrice'] = predict_journey_not_ontime['MediumPrice'].map({'Yes': 1

predict_journey_not_ontime.head()
```

Out[198…

	Payment.Method	Railcard	Ticket.Class	Ticket.Type	Price	Departure.Station	Arrival.Station	Departure
2	Debit Card	None	Standard	Off-Peak	113	Liverpool Lime Street	London Euston	2024-01-09 15:30:00
3	Contactless	Adult	Standard	Off-Peak	3	Liverpool Lime Street	Manchester Piccadilly	2024-01-31 05:45:00
4	Credit Card	None	Standard	Off-Peak	4	Manchester Piccadilly	Liverpool Lime Street	2024-02-10 16:00:00
5	Contactless	None	Standard	Advance	3	Manchester Piccadilly	Liverpool Lime Street	2024-02-25 15:45:00
6	Debit Card	None	Standard	Off-Peak	126	Manchester Piccadilly	London Euston	2024-03-20 15:30:00

In [200…

```python
# Convert object types to categorical

predict_journey_not_ontime = predict_journey_not_ontime.apply(lambda col: col.astype('category') if
```

In [202…

```python
print(predict_journey_not_ontime.dtypes)
```

```
Payment.Method             category
Railcard                   category
Ticket.Class               category
Ticket.Type                category
Price                         int64
Departure.Station          category
Arrival.Station            category
Departure            datetime64[ns]
Scheduled.Arrival    datetime64[ns]
Actual.Arrival       datetime64[ns]
Journey.Status             category
Reason.for.Delay           category
DelayInMinutes              float64
MediumPrice                   int64
dtype: object
```

In [204…

```python
predict_cat_vars = predict_journey_not_ontime.select_dtypes(include=['category']).columns

for var in predict_cat_vars:
    cat_list = pd.get_dummies(predict_journey_not_ontime[var], prefix=var, drop_first=True)
    predict_journey_not_ontime = pd.concat([predict_journey_not_ontime, cat_list], axis=1)
```

In [206…

```python
predict_data_vars=predict_journey_not_ontime.columns.values.tolist()
predict_to_keep=[i for i in predict_data_vars if i not in predict_cat_vars]
```

In [208…

```python
predict_journey_final=predict_journey_not_ontime[predict_to_keep]
predict_journey_final.columns.values
```

Out[208…

```
array(['Price', 'Departure', 'Scheduled.Arrival', 'Actual.Arrival',
       'DelayInMinutes', 'MediumPrice', 'Payment.Method_Credit Card',
       'Payment.Method_Debit Card', 'Railcard_None',
       'Ticket.Type_Off-Peak', 'Departure.Station_Liverpool Lime Street',
       'Departure.Station_Manchester Piccadilly',
       'Arrival.Station_London Euston',
       'Arrival.Station_London St Pancras',
       'Arrival.Station_Manchester Piccadilly', 'Journey.Status_Delayed',
       'Reason.for.Delay_Staffing', 'Reason.for.Delay_Technical Issue'],
      dtype=object)
```

Predictions for To_Predict using single predictor logit model

In [211…

```python
# Predict refund request using test data from to_predict

X_1 = predict_journey_final['MediumPrice'].values.reshape(-1, 1)
X_1 = sm.add_constant(X_1)

yhat = result1.predict(X_1)
prediction = list(map(round, yhat))

print("Predicted probabilities:", yhat)
print("Binary predictions (0 or 1):", prediction)
```

```
Predicted probabilities: [0.2543911 0.2543911 0.2543911 0.2543911 0.2543911 0.3271028]
Binary predictions (0 or 1): [0, 0, 0, 0, 0, 0]
```

Multiple Predictor Logistic Model Trained on MavenRail

In [214…

```python
# Fit multiple predictors logistic regression model
```

In [216…

```python
# Select numerical predictors
numerical_predictors = journey_final[['MediumPrice', 'Price', 'DelayInMinutes']]

# Select categorical predictors (assuming they are already dummy-encoded)
categorical_predictors = journey_final[['Journey.Status_Delayed',
                                        'Reason.for.Delay_Staffing',
                                        'Reason.for.Delay_Technical Issue']]

# Combine numerical and categorical predictors
X = pd.concat([numerical_predictors, categorical_predictors], axis=1)
# X = journey_final[['Price', 'DelayInMinutes', 'Journey.Status_Delayed','Reason.for.Delay_Staff','
# 'Reason.for.Delay_Technical Issue','Reason.for.Delay_Traffic','Reason.for.Delay_Weather']]

# Add a constant for the intercept
X = sm.add_constant(X)

# Set response variable
y = journey_final['Refund.Request_Yes'].astype(int)

# Build the logistic regression model
logit_model = sm.Logit(y, X.astype(float))
result = logit_model.fit()

# Print the model summary
print(result.summary2())
```

```
Optimization terminated successfully.
         Current function value: 0.476949
         Iterations 7
                             Results: Logit
================================================================================
Model:                   Logit                 Method:              MLE
Dependent Variable:      Refund.Request_Yes    Pseudo R-squared:    0.179
Date:                    2024-11-22 00:08      AIC:                 3986.9828
No. Observations:        4165                  BIC:                 4031.3241
Df Model:                6                     Log-Likelihood:      -1986.5
Df Residuals:            4158                  LL-Null:             -2418.7
Converged:               1.0000                LLR p-value:         1.8257e-183
No. Iterations:          7.0000                Scale:               1.0000
--------------------------------------------------------------------------------
                                  Coef.  Std.Err.    z     P>|z|   [0.025  0.975]
--------------------------------------------------------------------------------
const                            -1.0594   0.0696 -15.2188 0.0000 -1.1958 -0.9230
MediumPrice                      -0.0408   0.0985  -0.4141 0.6788 -0.2338  0.1523
Price                            -0.0048   0.0010  -4.5999 0.0000 -0.0068 -0.0028
DelayInMinutes                   -0.0593   0.0038 -15.5963 0.0000 -0.0667 -0.0518
Journey.Status_Delayed            1.4501   0.1293  11.2189 0.0000  1.1968  1.7035
Reason.for.Delay_Staffing         0.9982   0.1172   8.5144 0.0000  0.7684  1.2280
Reason.for.Delay_Technical Issue  1.4509   0.0965  15.0341 0.0000  1.2617  1.6400
================================================================================
```

Test Multiple Predictor Logistic Model on To_Predict

In [219…

```python
# Predict refund request using test data from to_predict

X_new = predict_journey_final[['Price', 'MediumPrice', 'DelayInMinutes', 'Journey.Status_Delayed',
                               'Reason.for.Delay_Technical Issue']]

X_new = sm.add_constant(X_new)
X_new = X_new.astype(float)

predictions = result.predict(X_new)


print("Predicted probabilities:", predictions)
# print("Binary predictions (0 or 1):", prediction)
```

```
Predicted probabilities: 2    0.010726
3    0.148387
4    0.556840
5    0.454240
6    0.009024
7    0.123304
dtype: float64
```