Brian Mallari - UK National Rail Exploratory Analysis - Python (pandas) (Jan 2024 - April 2024; MOCK DATA)

November 18, 2024

1 Preliminary Work

[23]:
```
# Import pandas
import pandas as pd
```

[24]:
```
# Import the railway.csv file into a dataframe
df_copy_0 = pd.read_csv('railway.csv')
```

[25]:
```
# Look at the first few rows of this dataframe in order to confirm that things␣
 ↪are going well so far
df_copy_0.head()
```

[25]:

	Transaction ID	Date of Purchase	Time of Purchase	Purchase Type
0	da8a6ba8-b3dc-4677-b176	2023-12-08	12:41:11	Online
1	b0cdd1b0-f214-4197-be53	2023-12-16	11:23:01	Station
2	f3ba7a96-f713-40d9-9629	2023-12-19	19:51:27	Online
3	b2471f11-4fe7-4c87-8ab4	2023-12-20	23:00:36	Station
4	2be00b45-0762-485e-a7a3	2023-12-27	18:22:56	Online

	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

	Departure Station	Arrival Destination	Date of Journey
0	London Paddington	Liverpool Lime Street	2024-01-01
1	London Kings Cross	York	2024-01-01
2	Liverpool Lime Street	Manchester Piccadilly	2024-01-02
3	London Paddington	Reading	2024-01-01
4	Liverpool Lime Street	London Euston	2024-01-01

	Departure Time	Arrival Time	Actual Arrival Time	Journey Status
0	11:00:00	13:30:00	13:30:00	On Time

```
1       09:45:00     11:35:00            11:40:00        Delayed
2       18:15:00     18:45:00            18:45:00        On Time
3       21:30:00     22:30:00            22:30:00        On Time
4       16:45:00     19:00:00            19:00:00        On Time

  Reason for Delay Refund Request
0              NaN              No
1   Signal Failure              No
2              NaN              No
3              NaN              No
4              NaN              No
```

2 Top 10 Most Popular Train Routes in the Dataset

[27]:
```
# Acquire counts of transaction IDs, group those counts by routes, and convert␣
 ↪the series to a dataframe
df_copy_1a = df_copy_0.groupby(['Departure Station', 'Arrival␣
 ↪Destination'])['Transaction ID'].agg('count').to_frame()

# Sort rows in the previous dataframe by counts in descending order, and then␣
 ↪look at only the top ten routes
df_copy_1b = df_copy_1a.sort_values(by = 'Transaction ID', ascending = False).
 ↪head(10)

# Rename the column with the counts of transaction IDs for clarity
df_copy_1final = df_copy_1b.rename(columns = {'Transaction ID' : 'Count of␣
 ↪Tickets Sold'})

# Return the final dataframe
df_copy_1final
```

[27]:

Departure Station	Arrival Destination	Count of Tickets Sold
Manchester Piccadilly	Liverpool Lime Street	4628
London Euston	Birmingham New Street	4209
London Kings Cross	York	3922
London Paddington	Reading	3873
London St Pancras	Birmingham New Street	3471
Liverpool Lime Street	Manchester Piccadilly	3002
	London Euston	1097
London Euston	Manchester Piccadilly	712
Birmingham New Street	London St Pancras	702
London Paddington	Oxford	485

3 Top 10 Most Popular Departure Times in the Dataset

[29]:
```
# Define a custom function for this analysis
def avg_count_of_train_tickets_purchased(df_group):
    return round(
        (
            (df_group['Transaction ID'].count()) / (df_group['Date of Journey'].
 ↪nunique())
        )
        , 2
    )

# Group by departure times, apply the custom function to each group
s_copy_2a = df_copy_0.groupby(['Departure Time']).
 ↪apply(avg_count_of_train_tickets_purchased, include_groups = False)
s_copy_2a

# Create a dataframe with the departure times as one column and the average␣
 ↪counts as another column
df_copy_2a = pd.DataFrame(
    {'Departure Time' : s_copy_2a.index, 'Avg Count of Train Tickets Purchased'␣
 ↪: s_copy_2a.values}
)



# Sort rows of the pervious dataframe by average counts in descending order,␣
 ↪and then look at only the top ten departure times
df_copy_2b = df_copy_2a.sort_values(by = 'Avg Count of Train Tickets␣
 ↪Purchased', ascending = False).head(10)

# Set clean up the final dataframe by making the index empty
blank_index = [''] * len(df_copy_2b)
df_copy_2final = df_copy_2b
df_copy_2final.index = blank_index
df_copy_2final
```

[29]:

Departure Time	Avg Count of Train Tickets Purchased
18:45:00	21.47
17:45:00	14.07
06:30:00	11.98
08:00:00	11.49
16:00:00	11.01
07:45:00	10.68
07:30:00	9.26
06:15:00	8.55
16:15:00	6.14
17:15:00	4.68

4 Revenue in British Pounds for Different Ticket Types and Classes

[31]:
```
# Group rows by ticket class and ticket type, and take the sum of transaction␣
 ↪prices for each group
df_copy_3a = df_copy_0.groupby(['Ticket Class', 'Ticket Type'])['Price'].
 ↪agg('sum').to_frame()

# Sort the rows first by ticket class in descending order and the by ticket␣
 ↪type in descending order
df_copy_3b = df_copy_3a.sort_values(['Ticket Class', 'Ticket Type'], ascending␣
 ↪= False)

# Rename the column with the sums of prices for clarity
df_copy_3final = df_copy_3b.rename(columns = {'Price' : 'Total Revenue (GBP)'})
df_copy_3final
```

[31]:

Ticket Class	Ticket Type	Total Revenue (GBP)
Standard	Off-Peak	178666
	Anytime	171468
	Advance	242388
First Class	Off-Peak	44672
	Anytime	37841
	Advance	66886

5 Contributing Factors to Delays and Cancelations

[33]:
```
# Replace 'Signal failure' with 'Signal Failure' in order for the final counts␣
 ↪to make sense
df_copy_4a = df_copy_0.replace('Signal failure', 'Signal Failure')

# Explicitly remove rows with null values for reasons for delay
# (By default, pandas excludes rows with null values, but those rows excluded␣
 ↪explicitly
# as a safeguard against bad luck)
df_copy_4b = df_copy_4a.dropna(subset = ['Reason for Delay'])

# Group rows by reason for delay and journey status, and then acquire counts of␣
 ↪transaction IDs for each group
df_copy_4c = df_copy_4b.groupby(['Reason for Delay', 'Journey␣
 ↪Status'])['Transaction ID'].agg('count').to_frame()

# Sort the rows first by reason for delay in ascending order and then by␣
 ↪journey status in ascending order
```

```
df_copy_4d = df_copy_4c.sort_values(['Reason for Delay', 'Journey Status'],␣
 ↪ascending = True)

# Rename the column with the sums of transaction IDs for clarity
df_copy_4final = df_copy_4d.rename(columns = {'Transaction ID' : 'Number of␣
 ↪Instances'})
df_copy_4final
```

[33]:

		Number of Instances
Reason for Delay	Journey Status	
Signal Failure	Cancelled	519
	Delayed	451
Staff Shortage	Cancelled	216
	Delayed	183
Staffing	Cancelled	238
	Delayed	172
Technical Issue	Cancelled	235
	Delayed	472
Traffic	Cancelled	227
	Delayed	87
Weather	Cancelled	237
	Delayed	758
Weather Conditions	Cancelled	208
	Delayed	169