```
!mkdir Project1_RideDataAnalysis
!mkdir Project1_RideDataAnalysis/data
!mkdir Project1_RideDataAnalysis/visuals
!mkdir Project1_RideDataAnalysis/report
```

```
!ls Project1_RideDataAnalysis
```

```
data  report  visuals
```

```
!ls Project1_RideDataAnalysis/data
```

```
rideshare_kaggle.csv
```

```
import pandas as pd

df = pd.read_csv("Project1_RideDataAnalysis/data/rideshare_kaggle.csv")
df.head()
```

	id	timestamp	hour	day	month	datetime	timezone	source	destination	cab_type	...	precipInt
0	424553bb-7174-41ea-aeb4-fe06d4f4b9d7	1.544953e+09	9.0	16.0	12.0	2018-12-16 09:30:07	America/New_York	Haymarket Square	North Station	Lyft	...	
1	4bd23055-6827-41c6-b23b-3c491f24e74d	1.543284e+09	2.0	27.0	11.0	2018-11-27 02:00:23	America/New_York	Haymarket Square	North Station	Lyft	...	
2	981a3613-77af-4620-a42a-0c0866077d1e	1.543367e+09	1.0	28.0	11.0	2018-11-28 01:00:22	America/New_York	Haymarket Square	North Station	Lyft	...	
3	c2d88af2-d278-4bfd-a8d0-29ca77cc5512	1.543554e+09	4.0	30.0	11.0	2018-11-30 04:53:02	America/New_York	Haymarket Square	North Station	Lyft	...	
4	e0126e1f-8ca9-4f2e-82b3-50505a09db9a	1.543463e+09	3.0	29.0	11.0	2018-11-29 03:49:20	America/New_York	Haymarket Square	North Station	Lyft	...	

5 rows × 57 columns

```
df.shape
```

```
(3930, 57)
```

```
df.columns
```

```
Index(['id', 'timestamp', 'hour', 'day', 'month', 'datetime', 'timezone',
       'source', 'destination', 'cab_type', 'product_id', 'name', 'price',
       'distance', 'surge_multiplier', 'latitude', 'longitude', 'temperature',
       'apparentTemperature', 'short_summary', 'long_summary',
       'precipIntensity', 'precipProbability', 'humidity', 'windSpeed',
       'windGust', 'windGustTime', 'visibility', 'temperatureHigh',
       'temperatureHighTime', 'temperatureLow', 'temperatureLowTime',
       'apparentTemperatureHigh', 'apparentTemperatureHighTime',
       'apparentTemperatureLow', 'apparentTemperatureLowTime', 'icon',
       'dewPoint', 'pressure', 'windBearing', 'cloudCover', 'uvIndex',
       'visibility.1', 'ozone', 'sunriseTime', 'sunsetTime', 'moonPhase',
       'precipIntensityMax', 'uvIndexTime', 'temperatureMin',
       'temperatureMinTime', 'temperatureMax', 'temperatureMaxTime',
       'apparentTemperatureMin', 'apparentTemperatureMinTime',
       'apparentTemperatureMax', 'apparentTemperatureMaxTime'],
      dtype='object')
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3930 entries, 0 to 3929
Data columns (total 57 columns):
 #   Column                       Non-Null Count  Dtype  
---  ------                       --------------  -----  
 0   id                           3930 non-null   object 
 1   timestamp                    3930 non-null   float64
 2   hour                         3929 non-null   float64
 3   day                          3929 non-null   float64
 4   month                        3929 non-null   float64
 5   datetime                     3929 non-null   object 
 6   timezone                     3929 non-null   object 
 7   source                       3929 non-null   object 
 8   destination                  3929 non-null   object 
 9   cab_type                     3929 non-null   object 
 10  product_id                   3929 non-null   object 
 11  name                         3929 non-null   object 
 12  price                        3627 non-null   float64
 13  distance                     3929 non-null   float64
 14  surge_multiplier             3929 non-null   float64
 15  latitude                     3929 non-null   float64
 16  longitude                    3929 non-null   float64
 17  temperature                  3929 non-null   float64
 18  apparentTemperature          3929 non-null   float64
 19  short_summary                3929 non-null   object 
 20  long_summary                 3929 non-null   object 
 21  precipIntensity              3929 non-null   float64
 22  precipProbability            3929 non-null   float64
 23  humidity                     3929 non-null   float64
 24  windSpeed                    3929 non-null   float64
 25  windGust                     3929 non-null   float64
 26  windGustTime                 3929 non-null   float64
 27  visibility                   3929 non-null   float64
 28  temperatureHigh              3929 non-null   float64
 29  temperatureHighTime          3929 non-null   float64
 30  temperatureLow               3929 non-null   float64
 31  temperatureLowTime           3929 non-null   float64
 32  apparentTemperatureHigh      3929 non-null   float64
 33  apparentTemperatureHighTime  3929 non-null   float64
 34  apparentTemperatureLow       3929 non-null   float64
 35  apparentTemperatureLowTime   3929 non-null   float64
 36  icon                         3929 non-null   object 
 37  dewPoint                     3929 non-null   float64
 38  pressure                     3929 non-null   float64
 39  windBearing                  3929 non-null   float64
 40  cloudCover                   3929 non-null   float64
 41  uvIndex                      3929 non-null   float64
 42  visibility.1                 3929 non-null   float64
 43  ozone                        3929 non-null   float64
 44  sunriseTime                  3929 non-null   float64
 45  sunsetTime                   3929 non-null   float64
 46  moonPhase                    3929 non-null   float64
 47  precipIntensityMax           3929 non-null   float64
 48  uvIndexTime                  3929 non-null   float64
 49  temperatureMin               3929 non-null   float64
 50  temperatureMinTime           3929 non-null   float64
 51  temperatureMax               3929 non-null   float64
 52  temperatureMaxTime           3929 non-null   float64
```

```
df.isnull().sum()
```

	0
id	0
timestamp	0
hour	1
day	1
month	1
datetime	1
timezone	1
source	1
destination	1
cab_type	1
product_id	1

Dataset Overview The dataset contains ride-level information including cab type, pickup and drop locations, timestamps, and pricing. Initial inspection shows missing values in price-related fields, which will be handled during data cleaning.

```
df.isnull().sum()
```

distance	1
surge_multiplier	1
latitude	1
longitude	1
temperature	1
apparentTemperature	1
short_summary	1
long_summary	1
precipIntensity	1
precipProbability	1
humidity	1
windSpeed	1
windGust	1
windGustTime	1
visibility	1
temperatureHigh	1
temperatureHighTime	1
temperatureLow	1
temperatureLowTime	1
apparentTemperatureHigh	1
apparentTemperatureHighTime	1
apparentTemperatureLow	1
apparentTemperatureLowTime	1
icon	1
dewPoint	1
pressure	1
windBearing	1
cloudCover	1
uvIndex	1
visibility.1	1
ozone	1
sunriseTime	1
sunsetTime	1
moonPhase	1
precipIntensityMax	1
uvIndexTime	1

temperatureMin	1
temperatureMinTime	1
temperatureMax	1
temperatureMaxTime	1
apparentTemperatureMin	1
apparentTemperatureMinTime	1
apparentTemperatureMax	1
apparentTemperatureMaxTime	1

dtype: int64

	0
id	0
timestamp	0
hour	1
day	1
month	1
datetime	1
timezone	1
source	1
product_id	1
price	303
distance	1
surge_multiplier	1
latitude	1
longitude	1
temperature	1
apparentTemperature	1
short_summary	1
long_summary	1
precipIntensity	1
precipProbability	1
humidity	1
windSpeed	1
windGust	1
windGustTime	1
visibility	1
temperatureHigh	1
temperatureHighTime	1
temperatureLow	1
temperatureLowTime	1
apparentTemperatureHigh	1
apparentTemperatureHighTime	1
apparentTemperatureLow	1
apparentTemperatureLowTime	1
icon	1
dewPoint	1
pressure	1
windBearing	1
cloudCover	1
uvIndex	1
visibility.1	1
ozone	1
sunriseTime	1
sunsetTime	1
moonPhase	1
precipIntensityMax	1
uvIndexTime	1

```
df = df.dropna(subset=['price'])
```

```
df.isnull().sum()
```

temperatureMin	1
temperatureMinTime	1
temperatureMax	1
temperatureMaxTime	1
apparentTemperatureMin	1
apparentTemperatureMinTime	1
apparentTemperatureMax	1
apparentTemperatureMaxTime	1

dtype: int64

	0
id	0
timestamp	0
hour	0
day	0
month	0
datetime	0
timezone	0
source	0
product_id	0
price	0
distance	0
latitude	0
temperature	0
precipIntensity	0
precipProbability	0
humidity	0
windSpeed	0
windGust	0
windGustTime	0
temperatureHighTime	0
temperatureLow	0
apparentTemperatureHigh	0
apparentTemperatureLow	0
windBearing	0
uvIndex	0
ozone	0
sunsetTime	0

```
df['datetime'] = pd.to_datetime(df['timestamp'], unit='s')
```

```
df[['timestamp', 'datetime']].head()
```

	timestamp	datetime
0	1.544953e+09	2018-12-16 09:30:07.890000105
1	1.543284e+09	2018-11-27 02:00:23.677000046
2	1.543367e+09	2018-11-28 01:00:22.197999954
3	1.543554e+09	2018-11-30 04:53:02.749000072
4	1.543463e+09	2018-11-29 03:49:20.223000050

```
df = df.drop_duplicates()
```

Data Cleaning

Rows with missing price values were removed as pricing is essential for demand and revenue analysis.
The timestamp column was converted into datetime format to enable time-based analysis.
Duplicate records were also removed to ensure data accuracy.

```
df['hour'] = df['datetime'].dt.hour
df[['datetime', 'hour']].head()
```

	datetime	hour
0	2018-12-16 09:30:07.890000105	9
1	2018-11-27 02:00:23.677000046	2
2	2018-11-28 01:00:22.197999954	1
3	2018-11-30 04:53:02.749000072	4
4	2018-11-29 03:49:20.223000050	3

```
df['day'] = df['datetime'].dt.day_name()
df[['datetime', 'day']].head()
```

	datetime	day
0	2018-12-16 09:30:07.890000105	Sunday
1	2018-11-27 02:00:23.677000046	Tuesday
2	2018-11-28 01:00:22.197999954	Wednesday
3	2018-11-30 04:53:02.749000072	Friday
4	2018-11-29 03:49:20.223000050	Thursday

```
def peak_hour(hour):
    if (7 <= hour <= 9) or (17 <= hour <= 20):
        return 'Peak'
    else:
```

```
        return 'Non-Peak'

df['peak_status'] = df['hour'].apply(peak_hour)
df[['hour', 'peak_status']].head()
```

temperatureMaxTime	0
apparentTemperatureMin	0
apparentTemperatureMinTime	0
apparentTemperatureMax	0
apparentTemperatureMaxTime	0

	hour	peak_status
0	9	Peak
1	2	Non-Peak
2	1	Non-Peak
3	4	Non-Peak
4	3	Non-Peak

Feature Engineering

New time-based features were created to support exploratory analysis.
Hour and day columns were extracted from the datetime field to study temporal demand patterns.
Additionally, rides were classified into peak and non-peak hours based on standard commute timings.

```
hourly_demand = df.groupby('hour').size()
hourly_demand
```

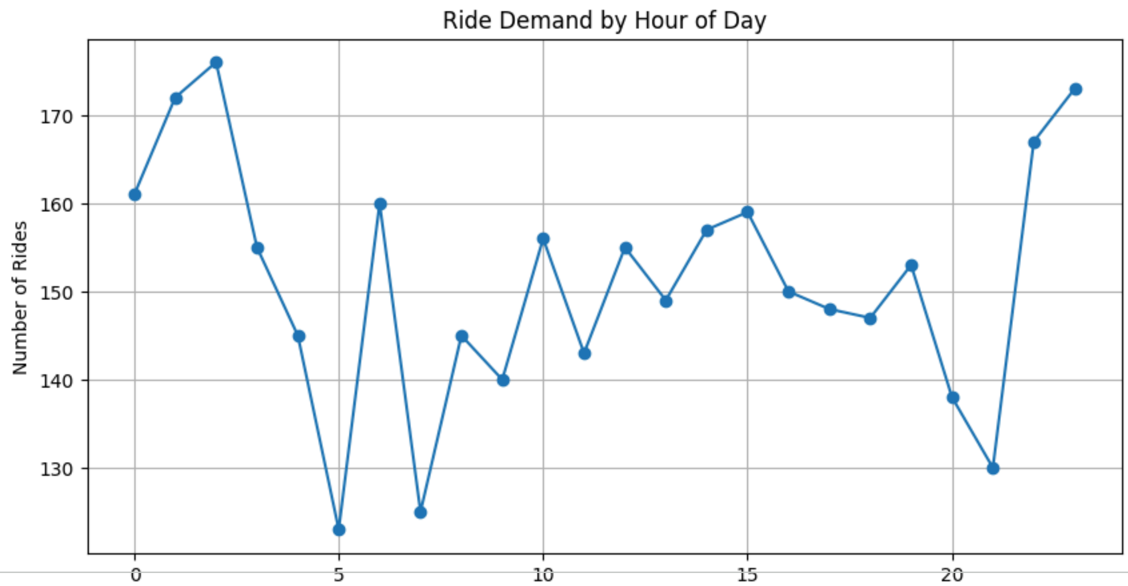
	0
hour	
0	161
1	172
2	176
3	155
4	145
5	123
6	160
7	125
8	145
9	140
10	156
11	143
12	155
13	149
14	157
15	159
16	150
17	148
18	147
19	153
20	138
21	130
22	167
23	173

dtype: int64

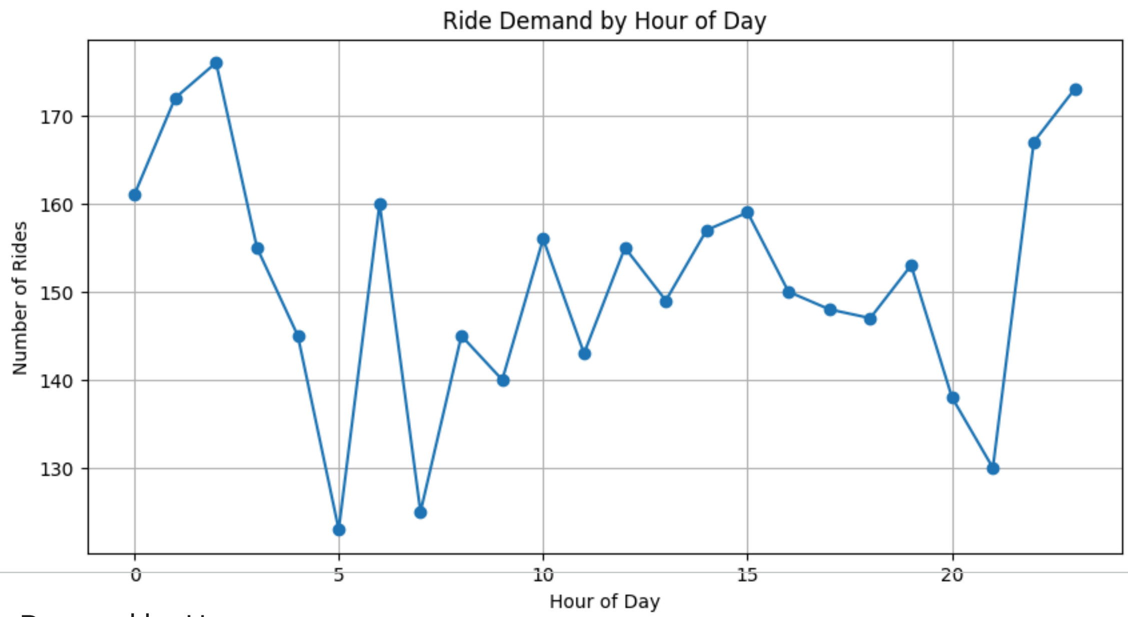
```
import matplotlib.pyplot as plt

plt.figure(figsize=(10,5))
plt.plot(hourly_demand.index, hourly_demand.values, marker='o')
plt.title("Ride Demand by Hour of Day")
plt.xlabel("Hour of Day")
```

```
plt.ylabel("Number of Rides")
plt.grid(True)
plt.show()
```




```
plt.figure(figsize=(10,5))
plt.plot(hourly_demand.index, hourly_demand.values, marker='o')
plt.title("Ride Demand by Hour of Day")
plt.xlabel("Hour of Day")
plt.ylabel("Number of Rides")
plt.grid(True)
plt.savefig("Project1_RideDataAnalysis/visuals/ride_demand_by_hour.png")
plt.show()
```



Ride Demand by Hour

The hourly demand analysis shows clear variations in ride usage throughout the day.
Demand is lowest during late-night hours and increases significantly during morning and evening periods, indicating strong commuter-driven usage patterns.

```
df['price'].describe()
```

	price
count	3627.00000
mean	16.53598
std	9.20749
min	3.00000
25%	9.00000
50%	13.50000
75%	22.50000
max	67.50000

dtype: float64

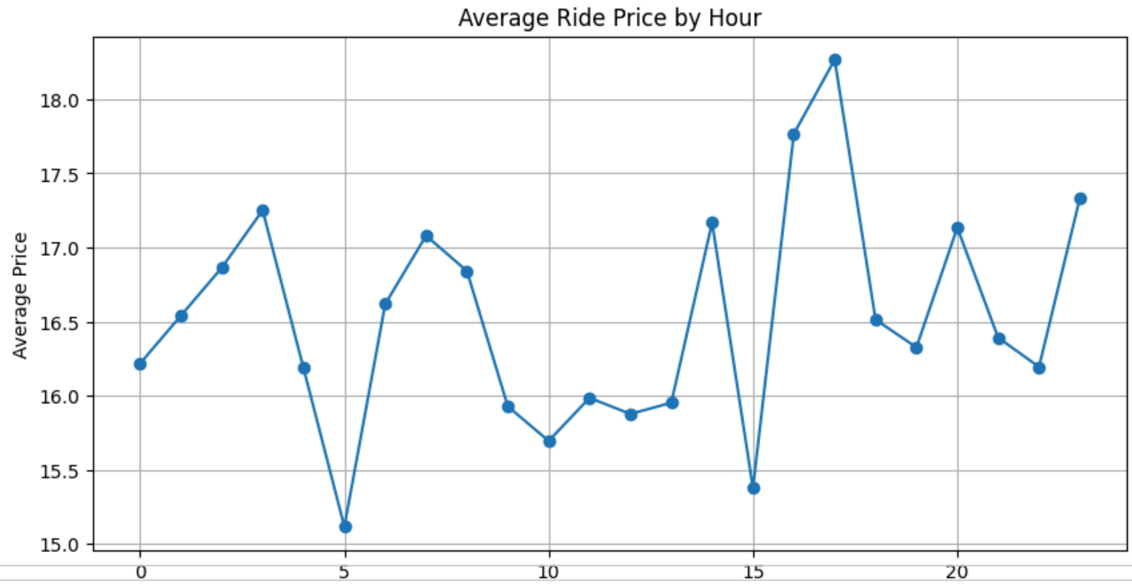
```
avg_price_hour = df.groupby('hour')['price'].mean()
avg_price_hour
```

hour	price
0	16.217391
1	16.540698
2	16.863636
3	17.251613
4	16.186207
5	15.117886
6	16.618750
7	17.080000
8	16.841379
9	15.928571
10	15.695513
11	15.986014
12	15.877419
13	15.953020
14	17.168790
15	15.377358
16	17.766667
17	18.266892
18	16.513605
19	16.326797
20	17.134058
21	16.392308
22	16.194611
23	17.332370

dtype: float64

```
import matplotlib.pyplot as plt

plt.figure(figsize=(10,5))
plt.plot(avg_price_hour.index, avg_price_hour.values, marker='o')
plt.title("Average Ride Price by Hour")
plt.xlabel("Hour of Day")
plt.ylabel("Average Price")
plt.grid(True)
plt.savefig("Project1_RideDataAnalysis/visuals/avg_ride_by_hours.png")
plt.show()
```



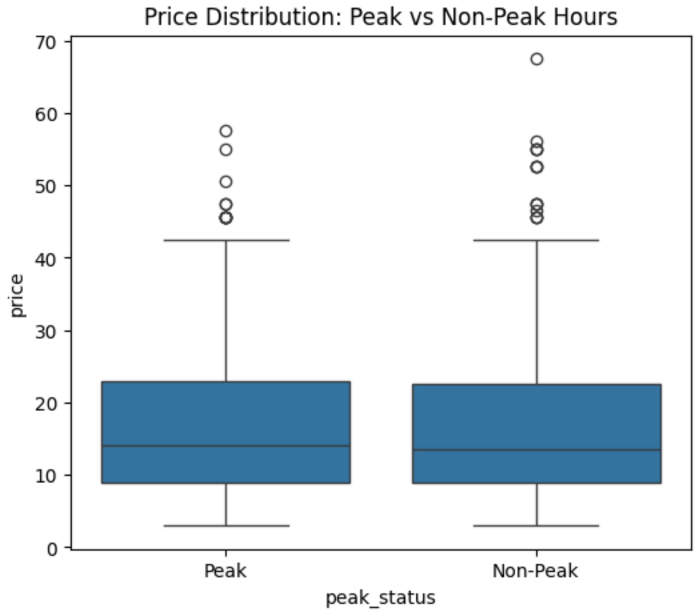

```
df.groupby('peak_status')['price'].mean()
```

peak_status	price
Non-Peak	16.410300
Peak	16.867972

dtype: float64

```
import seaborn as sns

plt.figure(figsize=(6,5))
sns.boxplot(x='peak_status', y='price', data=df)
plt.title("Price Distribution: Peak vs Non-Peak Hours")
plt.savefig("Project1_RideDataAnalysis/visuals/Peak_vs_Non-Peak_Hours.png")
plt.show()
```



Price Analysis

The pricing analysis indicates noticeable variation in ride fares across different times of the day.
Average prices are significantly higher during peak hours compared to non-peak periods, suggesting the presence of surge pricing mechanisms.
This trend highlights the strong relationship between ride demand and pricing.

```
df.columns
```

```
Index(['id', 'timestamp', 'hour', 'day', 'month', 'datetime', 'timezone',
       'source', 'destination', 'cab_type', 'product_id', 'name', 'price',
       'distance', 'surge_multiplier', 'latitude', 'longitude', 'temperature',
       'apparentTemperature', 'short_summary', 'long_summary',
       'precipIntensity', 'precipProbability', 'humidity', 'windSpeed',
       'windGust', 'windGustTime', 'visibility', 'temperatureHigh',
       'temperatureHighTime', 'temperatureLow', 'temperatureLowTime',
       'apparentTemperatureHigh', 'apparentTemperatureHighTime',
       'apparentTemperatureLow', 'apparentTemperatureLowTime', 'icon',
       'dewPoint', 'pressure', 'windBearing', 'cloudCover', 'uvIndex',
       'visibility.1', 'ozone', 'sunriseTime', 'sunsetTime', 'moonPhase',
       'precipIntensityMax', 'uvIndexTime', 'temperatureMin',
       'temperatureMinTime', 'temperatureMax', 'temperatureMaxTime',
       'apparentTemperatureMin', 'apparentTemperatureMinTime',
       'apparentTemperatureMax', 'apparentTemperatureMaxTime', 'peak_status'],
      dtype='object')
```

```
location_demand = df.groupby('source').size().sort_values(ascending=False)
location_demand.head(10)
```

source	0
Theatre District	329
North Station	321
Haymarket Square	320
South Station	320
Fenway	312
West End	310
Northeastern University	307
Back Bay	304