Data_Acquisition_Case_Study

January 1, 2025

#Data Acquisition Case Study

[69]:
```
import os
from pathlib import Path
```

0.1 Q1. Write Python code to create a new file named “sample_data.txt” in your documents folder and write the following content to it

ICTAK

Thejaswini,

Technopark Rd,

Technopark Campus,

Thiruvananthapuram,

Kerala 695581

[70]:
```
with open("sample_data.txt", "w") as file:
    file.write("""ICTAK

Thejaswini,

Technopark Rd,

Technopark Campus,

Thiruvananthapuram,

Kerala 695581""")
```

[71]:
```
os.listdir()
```

[71]:
```
['Practice_Reading_And Writing_Data.ipynb',
 'array.txt',
 '.DS_Store',
 'corr.csv',
 'requirements.txt',
 'example.db',
```

```
 'arrays.npz',
 'Data_Acquisition_Case_Study.ipynb',
 '__pycache__',
 'data.xlsx',
 'data.csv',
 'Day_18_Pandas_Reading_data_PySpark.pdf',
 'sample_write.txt',
 '.venv',
 'helper.py',
 'sample.jpg',
 'bike+sharing+dataset',
 'download_list.csv',
 'Classwork_Reading_and_Writing_Data.ipynb',
 '.ipynb_checkpoints',
 'sample_data.txt',
 'main.py',
 'array.npy',
 'random.txt',
 'Data',
 'random_array.npy',
 'sample.txt']
```

0.2 Q2. Write Python code to read and print the contents in "sample_data.txt"

[72]:
```
with open("sample_data.txt","r") as file:
    data = file.read()
    print(data)
```

```
ICTAK

Thejaswini,

Technopark Rd,

Technopark Campus,

Thiruvananthapuram,

Kerala 695581
```

0.3 Q3. Write Python code to check if "sample_data.txt" exists in documents folder

[73]:
```
print(os.path.exists("sample_data.txt"))
```

```
True
```

0.4 Q4: Save the following dataframe content to a CSV file (data.csv) in your downloads folder

[74]:
```
import pandas as pd

data = {"Id": [1, 2, 3],
        "Name": ["Alice", "Bob", "Charlie"],
        "Subject": ["Science", "Maths", "History"]}
df = pd.DataFrame(data)
```

[75]:
```
df.to_csv("data.csv", index=False)
```

[76]:
```
os.listdir()
```

[76]:
```
['Practice_Reading_And Writing_Data.ipynb',
 'array.txt',
 '.DS_Store',
 'corr.csv',
 'requirements.txt',
 'example.db',
 'arrays.npz',
 'Data_Acquisition_Case_Study.ipynb',
 '__pycache__',
 'data.xlsx',
 'data.csv',
 'Day_18_Pandas_Reading_data_PySpark.pdf',
 'sample_write.txt',
 '.venv',
 'helper.py',
 'sample.jpg',
 'bike+sharing+dataset',
 'download_list.csv',
 'Classwork_Reading_and_Writing_Data.ipynb',
 '.ipynb_checkpoints',
 'sample_data.txt',
 'main.py',
 'array.npy',
 'random.txt',
 'Data',
 'random_array.npy',
 'sample.txt']
```

0.5 Q5: Save the above dataframe content to an Excel (data.xlsx, sheet name: Sheet1) file in your documents folder

[77]:
```
df.to_excel("data.xlsx", sheet_name="Sheet1", index=False)
os.listdir()
```

[77]:
```
['Practice_Reading_And Writing_Data.ipynb',
 'array.txt',
 '.DS_Store',
 'corr.csv',
 'requirements.txt',
 'example.db',
 'arrays.npz',
 'Data_Acquisition_Case_Study.ipynb',
 '__pycache__',
 'data.xlsx',
 'data.csv',
 'Day_18_Pandas_Reading_data_PySpark.pdf',
 'sample_write.txt',
 '.venv',
 'helper.py',
 'sample.jpg',
 'bike+sharing+dataset',
 'download_list.csv',
 'Classwork_Reading_and_Writing_Data.ipynb',
 '.ipynb_checkpoints',
 'sample_data.txt',
 'main.py',
 'array.npy',
 'random.txt',
 'Data',
 'random_array.npy',
 'sample.txt']
```

0.6 Q6. Write code to get the list of files in your Downloads folder and save it to a CSV file name “download_list.csv”

[78]:
```
files = os.listdir(os.path.join(Path.home(), "Downloads"))
df = pd.DataFrame(files, columns=["File Name"])
df.to_csv("download_list.csv", index=False)
os.listdir()
```

[78]:
```
['Practice_Reading_And Writing_Data.ipynb',
 'array.txt',
 '.DS_Store',
 'corr.csv',
 'requirements.txt',
```

```
 'example.db',
 'arrays.npz',
 'Data_Acquisition_Case_Study.ipynb',
 '__pycache__',
 'data.xlsx',
 'data.csv',
 'Day_18_Pandas_Reading_data_PySpark.pdf',
 'sample_write.txt',
 '.venv',
 'helper.py',
 'sample.jpg',
 'bike+sharing+dataset',
 'download_list.csv',
 'Classwork_Reading_and_Writing_Data.ipynb',
 '.ipynb_checkpoints',
 'sample_data.txt',
 'main.py',
 'array.npy',
 'random.txt',
 'Data',
 'random_array.npy',
 'sample.txt']
```

0.7 Q7. Write Python code to save the contents of the given random_array variable as a numpy file

[79]:
```
import numpy as np
random_array = np.random.rand(10, 10)
```

[80]:
```
np.save("random_array.npy", random_array)
os.listdir()
```

[80]:
```
['Practice_Reading_And Writing_Data.ipynb',
 'array.txt',
 '.DS_Store',
 'corr.csv',
 'requirements.txt',
 'example.db',
 'arrays.npz',
 'Data_Acquisition_Case_Study.ipynb',
 '__pycache__',
 'data.xlsx',
 'data.csv',
 'Day_18_Pandas_Reading_data_PySpark.pdf',
 'sample_write.txt',
 '.venv',
 'helper.py',
```

```
 'sample.jpg',
 'bike+sharing+dataset',
 'download_list.csv',
 'Classwork_Reading_and_Writing_Data.ipynb',
 '.ipynb_checkpoints',
 'sample_data.txt',
 'main.py',
 'array.npy',
 'random.txt',
 'Data',
 'random_array.npy',
 'sample.txt']
```

0.8 Q8. Write python code to save the contents of the above numpy file as text file named "random.txt" with a delimitter of ";" to Documents folder

[81]:
```
np.savetxt("random.txt", random_array, delimiter=";")
os.listdir()
```

[81]:
```
['Practice_Reading_And Writing_Data.ipynb',
 'array.txt',
 '.DS_Store',
 'corr.csv',
 'requirements.txt',
 'example.db',
 'arrays.npz',
 'Data_Acquisition_Case_Study.ipynb',
 '__pycache__',
 'data.xlsx',
 'data.csv',
 'Day_18_Pandas_Reading_data_PySpark.pdf',
 'sample_write.txt',
 '.venv',
 'helper.py',
 'sample.jpg',
 'bike+sharing+dataset',
 'download_list.csv',
 'Classwork_Reading_and_Writing_Data.ipynb',
 '.ipynb_checkpoints',
 'sample_data.txt',
 'main.py',
 'array.npy',
 'random.txt',
 'Data',
 'random_array.npy',
 'sample.txt']
```

0.9 Download and analyze Bike Sharing Dataset (hour.csv) for UCI Irvin Repository (https://archive.ics.uci.edu/dataset/275/bike+sharing+dataset) and answer the following questions

0.10 Q9. What is the size of the dataset? (Number of rows and columns)

[82]:
```
# df_day = pd.read_csv(os.path.join("bike+sharing+dataset", "day.csv"))
df_hour = pd.read_csv(os.path.join("bike+sharing+dataset", "hour.csv"))
# print("day.csv : ", df_day.shape)
print("hour.csv :", "\nNo. of rows = ", df_hour.shape[0] ,"\nNo. of columns =␣
  ↪", df_hour.shape[1])
```

```
hour.csv :
No. of rows =  17379
No. of columns =  17
```

0.11 Q10. What are the data types of each column?

[83]:
```
df_hour.dtypes
```

[83]:
```
instant         int64
dteday         object
season          int64
yr              int64
mnth            int64
hr              int64
holiday         int64
weekday         int64
workingday      int64
weathersit      int64
temp          float64
atemp         float64
hum           float64
windspeed     float64
casual          int64
registered      int64
cnt             int64
dtype: object
```

0.12 Q11. Are there any missing values in the dataset? If so, which columns have missing values and how many?

[84]:
```
df_hour.isna().sum()
```

[84]:
```
instant       0
dteday        0
season        0
yr            0
```

```
mnth          0
hr            0
holiday       0
weekday       0
workingday    0
weathersit    0
temp          0
atemp         0
hum           0
windspeed     0
casual        0
registered    0
cnt           0
dtype: int64
```

0.13 Q.12. For the windspeed column, calculate the mean, median, and standard deviation.

[85]:
```
print("Mean: ", df_hour["windspeed"].mean())
print("Median: ", df_hour["windspeed"].median())
print("Standrad Deviation: ", df_hour["windspeed"].std())
```

```
Mean:  0.1900976063064618
Median:  0.194
Standrad Deviation:  0.12234022857279413
```

0.14 Q13. Identify any potential outliers in a numerical column of your choice. Explain your approach.

[86]:
```
col = df_hour["windspeed"]
Q1 = col.quantile(0.25)
Q3 = col.quantile(0.75)
IQR = Q3 - Q1
outliers = col[(col < Q1 - 1.5 * IQR) | (col > Q3 + 1.5 * IQR)]
print(outliers)


# Identified outliers using IQR method.
# Everything below lower bound and above upper bound are considered to be␣
 ↪outliers.
```

```
175      0.4925
178      0.5522
194      0.5224
196      0.5224
265      0.5821
         ...
17327    0.5522
```

```
17328    0.4925
17341    0.5821
17343    0.5821
17344    0.6567
Name: windspeed, Length: 342, dtype: float64
```

0.15 Q.14 Find the correlation between numerical columns and discuss any interesting relationships.

[87]:
```
df_num = df_hour.drop("dteday", axis=1)
df_num.corr()
```

[87]:

	instant	season	yr	mnth	hr	holiday
instant	1.000000	0.404046	0.866014	0.489164	-0.004775	0.014723
season	0.404046	1.000000	-0.010742	0.830386	-0.006117	-0.009585
yr	0.866014	-0.010742	1.000000	-0.010473	-0.003867	0.006692
mnth	0.489164	0.830386	-0.010473	1.000000	-0.005772	0.018430
hr	-0.004775	-0.006117	-0.003867	-0.005772	1.000000	0.000479
holiday	0.014723	-0.009585	0.006692	0.018430	0.000479	1.000000
weekday	0.001357	-0.002335	-0.004485	0.010400	-0.003498	-0.102088
workingday	-0.003416	0.013743	-0.002196	-0.003477	0.002285	-0.252471
weathersit	-0.014198	-0.014524	-0.019157	0.005400	-0.020203	-0.017036
temp	0.136178	0.312025	0.040913	0.201691	0.137603	-0.027340
atemp	0.137615	0.319380	0.039222	0.208096	0.133750	-0.030973
hum	0.009577	0.150625	-0.083546	0.164411	-0.276498	-0.010588
windspeed	-0.074505	-0.149773	-0.008740	-0.135386	0.137252	0.003988
casual	0.158295	0.120206	0.142779	0.068457	0.301202	0.031564
registered	0.282046	0.174226	0.253684	0.122273	0.374141	-0.047345
cnt	0.278379	0.178056	0.250495	0.120638	0.394071	-0.030927

	weekday	workingday	weathersit	temp	atemp	hum
instant	0.001357	-0.003416	-0.014198	0.136178	0.137615	0.009577
season	-0.002335	0.013743	-0.014524	0.312025	0.319380	0.150625
yr	-0.004485	-0.002196	-0.019157	0.040913	0.039222	-0.083546
mnth	0.010400	-0.003477	0.005400	0.201691	0.208096	0.164411
hr	-0.003498	0.002285	-0.020203	0.137603	0.133750	-0.276498
holiday	-0.102088	-0.252471	-0.017036	-0.027340	-0.030973	-0.010588
weekday	1.000000	0.035955	0.003311	-0.001795	-0.008821	-0.037158
workingday	0.035955	1.000000	0.044672	0.055390	0.054667	0.015688
weathersit	0.003311	0.044672	1.000000	-0.102640	-0.105563	0.418130
temp	-0.001795	0.055390	-0.102640	1.000000	0.987672	-0.069881
atemp	-0.008821	0.054667	-0.105563	0.987672	1.000000	-0.051918
hum	-0.037158	0.015688	0.418130	-0.069881	-0.051918	1.000000
windspeed	0.011502	-0.011830	0.026226	-0.023125	-0.062336	-0.290105
casual	0.032721	-0.300942	-0.152628	0.459616	0.454080	-0.347028
registered	0.021578	0.134326	-0.120966	0.335361	0.332559	-0.273933
cnt	0.026900	0.030284	-0.142426	0.404772	0.400929	-0.322911

	windspeed	casual	registered	cnt
instant	-0.074505	0.158295	0.282046	0.278379
season	-0.149773	0.120206	0.174226	0.178056
yr	-0.008740	0.142779	0.253684	0.250495
mnth	-0.135386	0.068457	0.122273	0.120638
hr	0.137252	0.301202	0.374141	0.394071
holiday	0.003988	0.031564	-0.047345	-0.030927
weekday	0.011502	0.032721	0.021578	0.026900
workingday	-0.011830	-0.300942	0.134326	0.030284
weathersit	0.026226	-0.152628	-0.120966	-0.142426
temp	-0.023125	0.459616	0.335361	0.404772
atemp	-0.062336	0.454080	0.332559	0.400929
hum	-0.290105	-0.347028	-0.273933	-0.322911
windspeed	1.000000	0.090287	0.082321	0.093234
casual	0.090287	1.000000	0.506618	0.694564
registered	0.082321	0.506618	1.000000	0.972151
cnt	0.093234	0.694564	0.972151	1.000000

0.16 Q.15 Based on your analysis, provide a brief summary of any insights or patterns you discovered in the dataset.

1. Apparent temperature and actual temperature are almost identical in this dataset (Correlation = 0.999). 2. High humidity correlates with lower bike rentals (Correlation = -0.733). 3. Total bike rentals are heavily influenced by both casual (Correlation = 0.876) and registered (Correlation = 0.987) users, with a stronger dependency on registered users. 4. Negative Impact of Weather Conditions: • Weather Situation (weathersit) has a negative correlation with total bike rentals (cnt): -0.605. • Bad weather (like heavy rain or snow) reduces bike usage significantly. 5. Seasonal Trends: • Moderate positive correlation between season and temperature (temp): 0.340. • This indicates warmer seasons (like spring/summer) lead to increased bike rentals.

Summary of Insights: 1. Peak Rental Hours: Higher correlations between hr and cnt suggest specific peak hours for bike usage, likely during commuting times. 2. Weather Sensitivity: Bad weather and high humidity negatively impact rentals, implying a preference for fair weather biking. 3. User Types: Registered users contribute more consistently to total rentals compared to casual users. 4. Temperature Influence: Warmer weather correlates with increased rentals, particularly during summer and spring seasons.

0.17 Q.16 In which season (Spring, Summer, Fall, Winter) people rented bikes the most?

[88]:
```
# 1:winter, 2:spring, 3:summer, 4:fall
season_rentals = df_hour.groupby("season")["cnt"].sum()
print(season_rentals.idxmax())
```

```
3
```

0.18 Q.17 What is the peak hour in which bike rents the most?

[89]:
```
hour_rentals = df_hour.groupby("hr")["cnt"].sum()
print(hour_rentals.idxmax())
```

```
17
```

0.19 Q.18 In which day of a week bikes rents out most?

[90]:
```
day_rentals = df_hour.groupby("weekday")["cnt"].sum()
print(day_rentals.idxmax())
```

```
5
```

0.20 Q.19 In which hour Casual users rents bikes the most?

[91]:
```
hour_rentals = df_hour.groupby("hr")["casual"].sum()
print(hour_rentals.idxmax())
```

```
14
```

0.21 Q.20 What is the maximum temperature observed in each of the seasons?

[92]:
```
max_temp = df_hour.groupby("season")["temp"].max()
print(max_temp)
```

```
season
1    0.72
2    0.94
3    1.00
4    0.76
Name: temp, dtype: float64
```