

## ▼ MIE 223 Assignment 03

### ▼ Winter 2025

```
# Do NOT modify this block of code
```

```
import numpy as np
import numpy.typing as npt
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
```

### ▼ Q1

```
# DO NOT MODIFY
# loading in the data
accident = pd.read_csv("https://raw.githubusercontent.com/1
print(accident.shape)
accident.head()
```

(12422, 8)

	ID	Severity	Time	City	Traffic_Signal	Humidity(%)	Crossin
0	3515243	A-	2016-06-24 09:33:07	Tempe	False	19.0	Fal
1	1713347	A-	2019-12-13 08:19:47	Buffalo	False	63.0	Fal
2	1882938	A-	2019-09-27 08:30:25	Cincinnati	True	84.0	Tr
3	904576	A-	2021-09-10 08:03:42	Syracuse	False	86.0	Fal
4	3386624	A-	2017-08-18 18:12:05	Ladson	False	66.0	Fal

Next steps:

[Generate code with accident](#)[New interactive sheet](#)

## ▼ Q1(a)

```
# your code starts here #
# Find the time that is 13th of the month
accident['Time'] = pd.to_datetime(accident['Time'])
accident['Time']
# if the row has a date of 13th, get dummy
accident['is_13th'] = (accident['Time'].dt.day == 13).astype(int)
accident['is_13th']
# end #
```

**is\_13th**

<b>0</b>	0
<b>1</b>	1
<b>2</b>	0
<b>3</b>	0
<b>4</b>	0
...	...
<b>12417</b>	0
<b>12418</b>	0
<b>12419</b>	0
<b>12420</b>	0
<b>12421</b>	0

12422 rows × 1 columns

**dtype:** int64

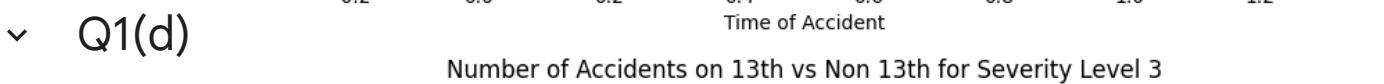
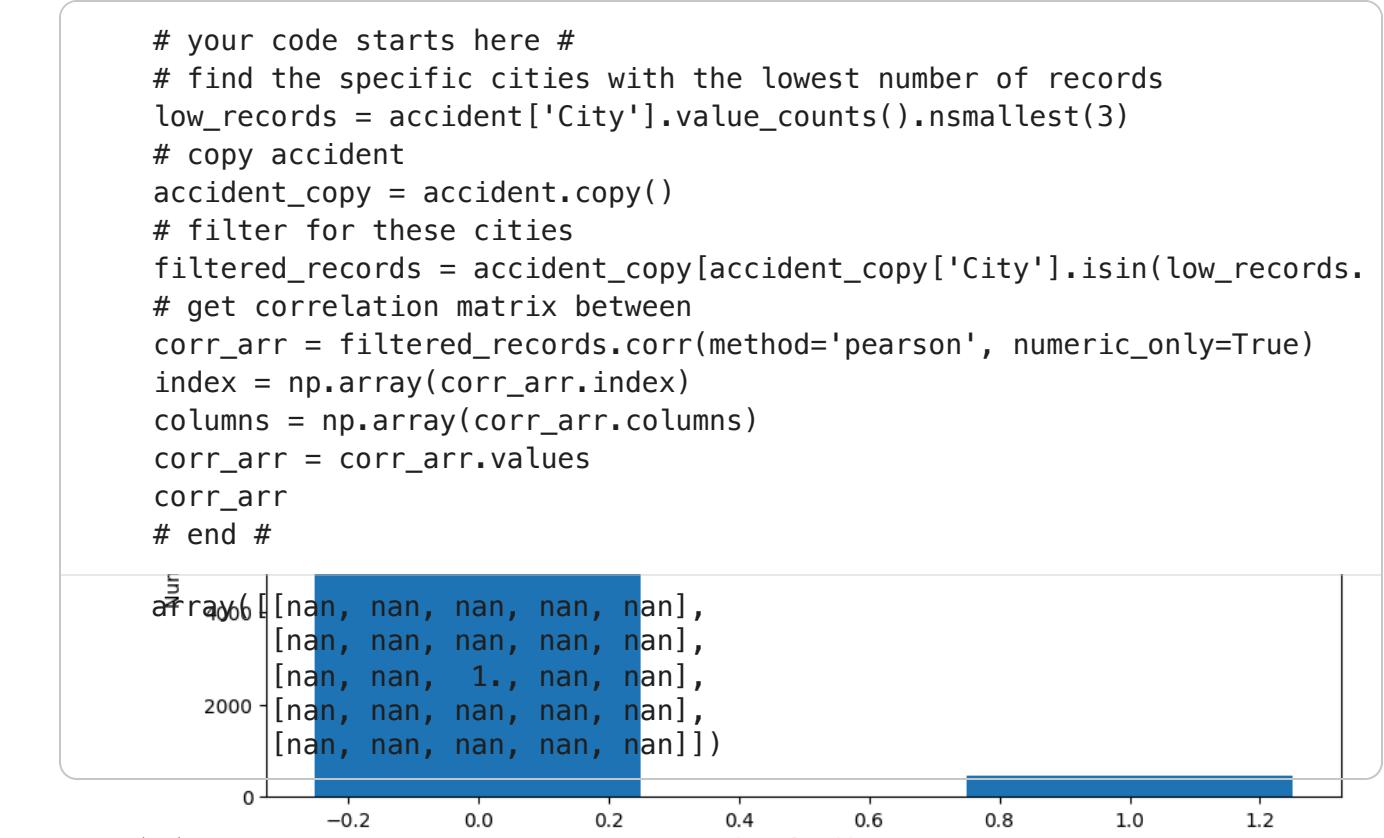
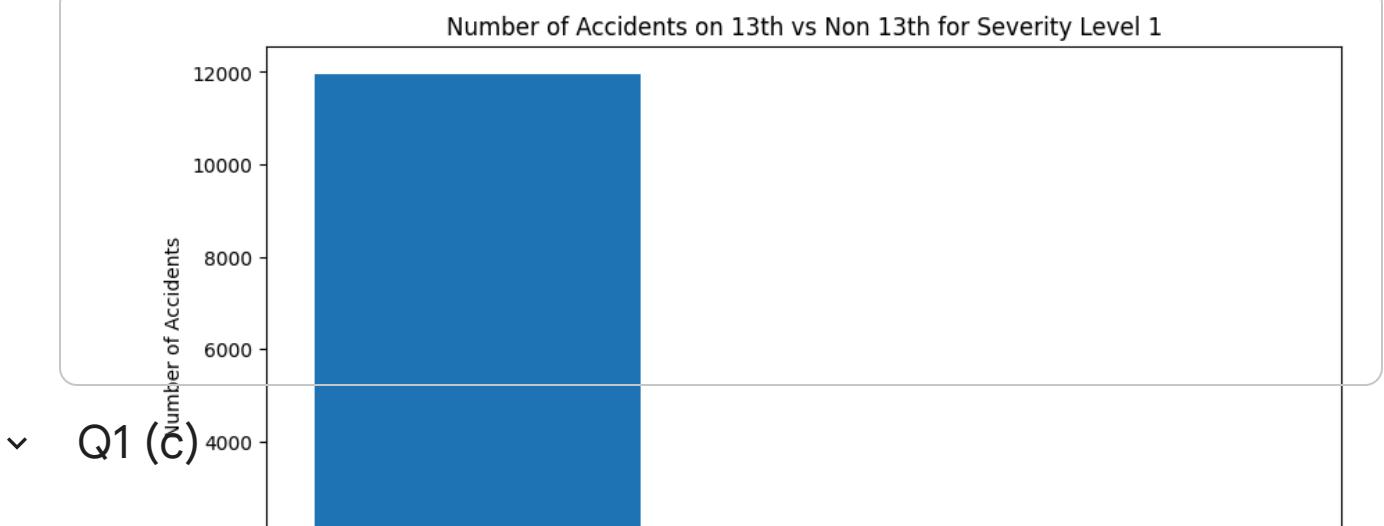
```
# Do not uncomment the code until you finish Q1a
# Do not change the code
accident['is_13th'].value_counts()
```

```
count  
is_13th  
0    11963  
1     459  
  
dtype: int64
```

## Q1(b)

```
# your code starts here #  
# 4 bars with 4 levels severity levels  
# bar represents the number of accidents on the 13th and non 13th  
fig = plt.figure(figsize=(10, 25))  
  
for i in range(1,5):  
    plt.subplot(4, 1, i)  
  
    severity_data = accident[accident['Severity'] == i]  
    number_accidents = accident['is_13th'].value_counts()  
  
    plt.bar(number_accidents.index, number_accidents.values, width=0.5)  
    plt.ylabel("Number of Accidents")  
    plt.xlabel("Time of Accident")  
    plt.title(f"Number of Accidents on 13th vs Non 13th for Severity Level {i}")  
plt.show()  
# end #
```





```
# your code starts here #
# convert the severity column to serious levels
conditions = [
    accident['Severity'] == 1,
    accident['Severity'] == 2,
    accident['Severity'] == 3
]
choices = ['most serious', 'very serious', 'serious']
accident['Severity'] = np.select(conditions, choices, default='light')

# crosstab
crosstab = pd.crosstab(accident['Severity'], accident['Crossing'], norma
```

```
crosstab
# end #
```

Crossing	False	True	Time of Accident
Severity	Number of Accidents		13th vs Non 13th for Severity Level 4
light	0.938462	0.061538	
most serious	0.650000	0.350000	
serious	0.963087	0.036913	
very serious	0.861770	0.138230	

Next steps.

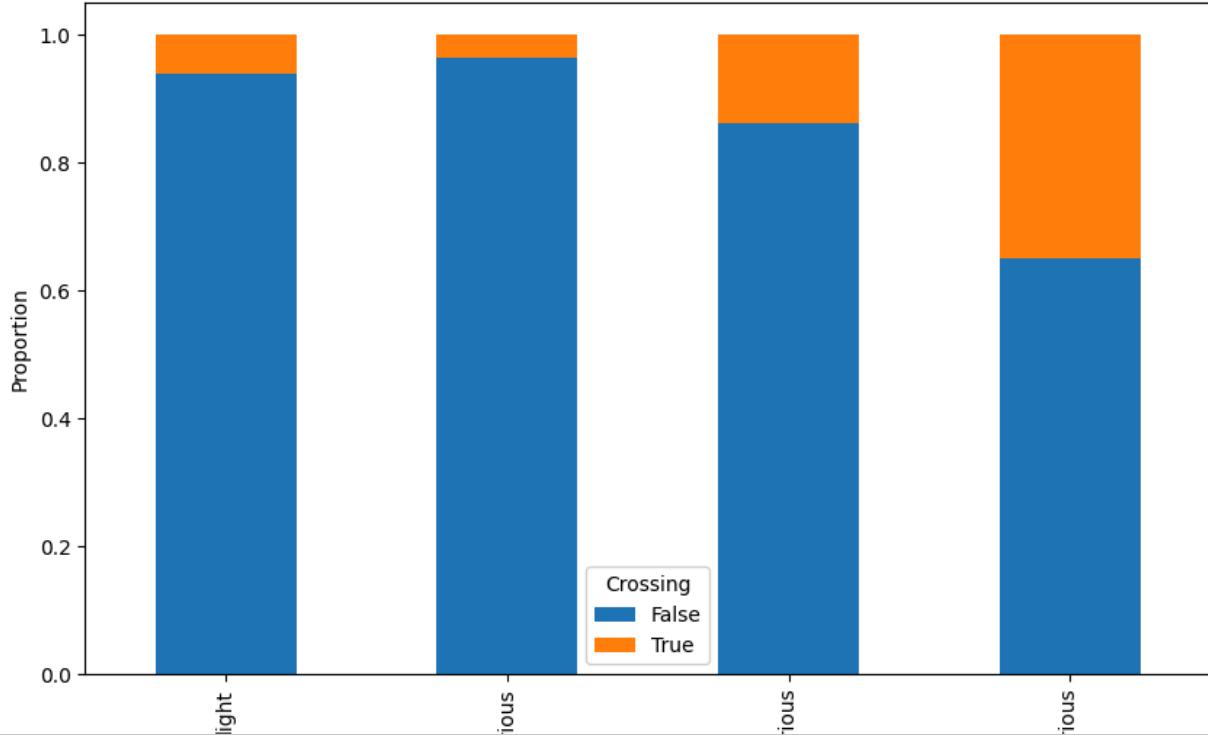
[Generate code with crosstab](#)[New interactive sheet](#)

```
# DO NOT modify the code,
# UNCOMMEND it until you finish Q1d
```

```
order = ["light", "serious", "very serious", "most serious"]
crosstab = crosstab.reindex(order)
crosstab.plot(kind='bar', stacked=True, figsize=(10, 6))
plt.title("Accident Severity by Crossing Presence")
plt.xlabel("Severity")
plt.ylabel("Proportion")
```

```
Text(0, 0.5, 'Proportion')
```

Accident Severity by Crossing Presence



## Q2

```
# Do NOT modify this block of code

# loading in the data
google_playstore_df = pd.read_csv("https://raw.githubusercontent.com/MIE

print(google_playstore_df.shape)
google_playstore_df.head()
```

(10841, 13)

	App	Category	Rating	Reviews	Size	Installs	Type	Price
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND DESIGN	4.1	159	19M	10,000+	Free	(
1	Coloring book moana	ART_AND DESIGN	3.9	967	14M	500,000+	Free	(
2	FREE Live Cool Themes, Hide ...	ART_AND DESIGN	4.7	87510	8.7M	5,000,000+	Free	(
3	Sketch - Draw & Paint	ART_AND DESIGN	4.5	215644	25M	50,000,000+	Free	(
4	Pixel Draw - Number Art Coloring Book	ART_AND DESIGN	4.3	967	2.8M	100,000+	Free	(

Next steps:

[Generate code with google\\_playstore\\_df](#)

[New interactive sheet](#)

### Cleaning the Data

```
# Do NOT modify this block of code
google_playstore_df[google_playstore_df['Category'] == '1.9'] # corrupte
```

```
# Dropping the corrupted row
google_playstore_df = google_playstore_df.drop(10472, axis=0)
google_playstore_df.shape
```

```
(10840, 13)
```

```
# Do NOT modify this block of code
google_playstore_df.isnull().sum() / google_playstore_df.shape[0] # % of
```

	0
<b>App</b>	0.000000
<b>Category</b>	0.000000
<b>Rating</b>	0.135978
<b>Reviews</b>	0.000000
<b>Size</b>	0.000000
<b>Installs</b>	0.000000
<b>Type</b>	0.000092
<b>Price</b>	0.000000
<b>Content Rating</b>	0.000000
<b>Genres</b>	0.000000
<b>Last Updated</b>	0.000000
<b>Current Ver</b>	0.000738
<b>Android Ver</b>	0.000185

```
dtype: float64
```

```
# Do NOT modify this block of code

# Dropping rows with null values
google_playstore_df = google_playstore_df.dropna(axis=0)

print(f"Number of missing values is: {google_playstore_df.isnull().sum()}
google_playstore_df.head()
```

Number of missing values is: 0, shape of data is: (9360, 13)

	App	Category	Rating	Reviews	Size	Installs	Type	Price
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND DESIGN	4.1	159	19M	10,000+	Free	(
1	Coloring book moana	ART_AND DESIGN	3.9	967	14M	500,000+	Free	)
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND DESIGN	4.7	87510	8.7M	5,000,000+	Free	(
3	Sketch - Draw & Paint	ART_AND DESIGN	4.5	215644	25M	50,000,000+	Free	)
4	Pixel Draw - Number Art Coloring Book	ART_AND DESIGN	4.3	967	2.8M	100,000+	Free	)

Next steps:

[Generate code with google\\_playstore\\_df](#)

[New interactive sheet](#)

```
# Do NOT modify this block of code
google_playstore_df.dtypes # checking the data types
```

0	
<b>App</b>	object
<b>Category</b>	object
<b>Rating</b>	float64
<b>Reviews</b>	object
<b>Size</b>	object
<b>Installs</b>	object
<b>Type</b>	object
<b>Price</b>	object
<b>Content Rating</b>	object
<b>Genres</b>	object
<b>Last Updated</b>	object
<b>Current Ver</b>	object
<b>Android Ver</b>	object

**dtype:** object

```
# Do NOT modify this block of code
google_playstore_df.unique() # checking the number of unique values per
```

	0
<b>App</b>	8190
<b>Category</b>	33
<b>Rating</b>	39
<b>Reviews</b>	5990
<b>Size</b>	413
<b>Installs</b>	19
<b>Type</b>	2
<b>Price</b>	73
<b>Content Rating</b>	6
<b>Genres</b>	115
<b>Last Updated</b>	1299
<b>Current Ver</b>	2638
<b>Android Ver</b>	31

**dtype:** int64

```
# Do NOT modify this block of code
def format_number(num: str) -> float:
    """Function to format a number by converting the place value from str
    """
    num = num.lower() # convert to lowercase

    # if str number can be converted to float without further cleanup, c
    try:
        return float(num.strip())
    except ValueError:
        pass

    # if after replacing the place value with number, num is still not co
    try:
        float(num[:-1].strip())
    except ValueError:
        return np.nan

    # else, replace the str place value by multiplying by the appropriat
    suffix_mapper = {'k': 1E3, 'm': 1E6, 'g': 1E9}

    return float(num[:-1]) * suffix_mapper[num[-1]]
```

```
# Do NOT modify this block of code
def format_place_value(num: str) -> str:
    """Function to format a number by converting it to its abbreviated p
    """
    num = int(num.strip("+").strip().replace(",",""))

    if num >= 1_000_000_000:
        return f"{num // 1_000_000_000}G+"

    if num >= 1_000_000:
        return f"{num // 1_000_000}M+"

    if num >= 1000:
        return f"{num // 1000}k+"

    return f"{num}+"
```

```
# Do NOT modify this block of code
google_playstore_df['Log-Reviews'] = np.log(google_playstore_df.loc[:, 'Reviews'])

google_playstore_df['Price'] = google_playstore_df['Price'].apply(lambda x: float(x))

google_playstore_df['Last Updated'] = pd.to_datetime(google_playstore_df['Last Updated'])

google_playstore_df['Size'] = google_playstore_df['Size'].apply(format_size)

google_playstore_df['Installs'] = google_playstore_df['Installs'].apply(lambda x: float(x.replace(',', '')))

google_playstore_df['is_good_rating'] = google_playstore_df.loc[:, 'Rating'].apply(lambda x: 1 if x >= 4 else 0)

google_playstore_df["Install dummy"] = google_playstore_df["Installs"].apply(lambda x: 1 if x >= 1000000000 else 0)
```

```
# Do NOT modify this block of code
google_playstore_df.head() # checking the head of the cleaned data
```

	App	Category	Rating	Reviews	Size	Installs	Type	P
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND DESIGN	4.1	159	19000000.0		10k+	Free
1	Coloring book moana	ART_AND DESIGN	3.9	967	14000000.0		500k+	Free
2	U Launcher Lite – FREE Live Cool Themes, Hide ...	ART_AND DESIGN	4.7	87510	8700000.0		5M+	Free
3	Sketch - Draw & Paint	ART_AND DESIGN	4.5	215644	25000000.0		50M+	Free
4	Pixel Draw - Number Art Coloring Book	ART_AND DESIGN	4.3	967	2800000.0		100k+	Free

Next steps: [Generate code with google\\_playstore\\_df](#) [New interactive sheet](#)

## ▼ Q2(a)

```
# Do NOT modify this block of code
continuous_cols = ['Rating', 'Log-Reviews', 'Size', 'Price']
```

```
# Your code starts here ##  
# create subset containing only rows where the app is not free  
non_free_apps = google_playstore_df[google_playstore_df['Price'] != 0]  
# take continuous cols and remove Rating  
plot_vars = [col for col in continuous_cols if col != 'Rating']  
# pairplot
```

