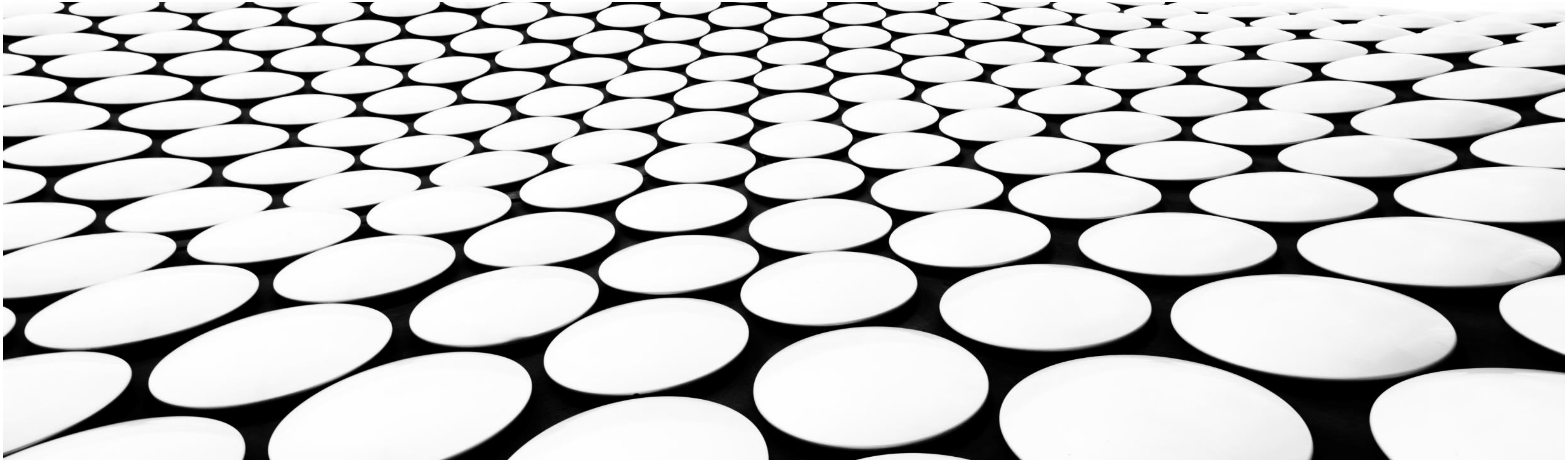
PROJECT 1 - BIKE SHARING

CREATED BY LEKSHMI, AURELIAN AND AMBER



HYPOTHESIS

Our core hypothesis for this Bike Sharing project is to discover whether Bike Sharing usage and rental behaviors can be impacted by environmental and seasonal settings, weather conditions, day of the week and by the hour of the day.

" If the environmental and seasonal settings are favorable, then there will be a significant increase in the number of rental bikes being used."

QUESTIONS:

- Do Weather Patterns Affect the amount of Bike Rentals there are? What Weather Type has the highest/lowest amount of bike rentals?
- Does a particular season have higher or lower rentals? Why?
- Are there any trends in rental amounts when it is a weekend vs a weekday? How about rental usage during holidays?

MOTIVATIONS:

- We were curious about this dataset because we wanted to test to see if an increase in rentals would be due to clearer days, weekends or possibly summertime when more people are out!
- We also wanted to see if these people who rented were people using this method of bike sharing as their primary mode of transportation on a daily basis.

SUMMARY OF EXPLORING DATASETS

We found our Dataset on Kaggle.com (Bike Sharing in Washington D.C. Dataset | Kaggle).

We used this dataset because we were able to extract 2 years of Bike Rental usage for Washington D.C. right down to the hour! This dataset also includes the daily weather as well as rental usage per season, weekends, weekdays & holidays.

We still had to clean up the dataset by renaming certain numeric codes to what the actual description of it was as well as extracting only the files we needed to explore our questions. We then created two new CSV files to work from and analyzed the data.

For example:

Seasons were listed as (1) (2) (3) (4) & we had to go in and rename those to correlate to the correct season like:

Winter, Summer, Spring and Fall. (See following Slides)

JUPYTER NOTEBOOK: DATA EXPLORATION AND CLEANUP PROCESS

n [4]:

```
#Dependencies
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import requests
import csv
import time
import scipy.stats as st
from scipy.stats import linregress
```

n [5]:

```
## Output File (CSV)
data_file_hour="archive/hour.csv"
data_file_day="archive/day.csv"
#Creating DataFrame
Bike_Share_Day_df=pd.read_csv(data_file_day)
Bike_Share_Hour=pd.read_csv(data_file_hour)
Bike_Share_Hour.head()
```

ut[5]:

	instant	dteday	season	yr	mnth	hr	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual	register
0	1	2011-01-01	1	0	1	0	0	6	0	1	0.24	0.2879	0.81	0.0	3	13
1	2	2011-01-01	1	0	1	1	0	6	0	1	0.22	0.2727	0.80	0.0	8	32
2	3	2011-01-01	1	0	1	2	0	6	0	1	0.22	0.2727	0.80	0.0	5	27
3	4	2011-01-01	1	0	1	3	0	6	0	1	0.24	0.2879	0.75	0.0	3	10
4	5	2011-01-01	1	0	1	4	0	6	0	1	0.24	0.2879	0.75	0.0	0	1

n [6]:

```
Bike_Share=Bike_Share_Hour[['instant','dteday','season','yr','mnth','hr',
                                'holiday','weekday','workingday','weathersit','cnt' ]].copy()
Bike_Share.head()
```

ut[6]:

	instant	dteday	season	yr	mnth	hr	holiday	weekday	workingday	weathersit	cnt
0	1	2011-01-01	1	0	1	0	0	6	0	1	16
1	2	2011-01-01	1	0	1	1	0	6	0	1	40
2	3	2011-01-01	1	0	1	2	0	6	0	1	32
3	4	2011-01-01	1	0	1	3	0	6	0	1	13
4	5	2011-01-01	1	0	1	4	0	6	0	1	1

JUPYTER NOTEBOOK:
DATA EXPLORATION AND CLEANUP PROCESS

SEASONS WERE LISTED AS (1) (2) (3) (4) & WE HAD TO GO IN AND RENAME THOSE TO CORRELATE TO THE CORRECT SEASON LIKE: WINTER, SUMMER, FALL AND SPRING

Adding Season labels

In [7]:
```
Bike_Share_Season=Bike_Share.copy()
bins=[0,1,2,3,4]
group=['Spring','Summer','Fall','Winter']
Bike_Share_Season['Season Name']=pd.cut(Bike_Share_Season['season'],bins,labels=group,include_lowest=True)
Bike_Share_Season.head()
```

Out[7]:

	instant	dteday	season	yr	mnth	hr	holiday	weekday	workingday	weathersit	cnt	Season Name
0	1	2011-01-01	1	0	1	0	0	6	0	1	16	Spring
1	2	2011-01-01	1	0	1	1	0	6	0	1	40	Spring
2	3	2011-01-01	1	0	1	2	0	6	0	1	32	Spring
3	4	2011-01-01	1	0	1	3	0	6	0	1	13	Spring
4	5	2011-01-01	1	0	1	4	0	6	0	1	1	Spring

JUPYTER NOTEBOOK: DATA EXPLORATION AND CLEANUP PROCESS

Adding Weather labels

In [5]:
```
#weathersit :
#                - 1: Clear, Few clouds, Partly cloudy, Partly cloudy
#                - 2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist
#                - 3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds
#                - 4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog
```

In [8]:
```
Bike_Share_Day=Bike_Share_Day_df[['instant','dteday','season','yr','mnth',
                                'holiday','weekday','workingday','weathersit','cnt' ]].copy()
bins=[0,1,2,3]
group=['Clear','Cloudy','Rainy']
Bike_Share_Day['Weather']=pd.cut(Bike_Share_Day['weathersit'],bins,labels=group,include_lowest=True)
Bike_Share_Day.head()
```

Out[8]:

	instant	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit	cnt	Weather
0	1	2011-01-01	1	0	1	0	6	0	2	985	Cloudy
1	2	2011-01-02	1	0	1	0	0	0	2	801	Cloudy
2	3	2011-01-03	1	0	1	0	1	1	1	1349	Clear
3	4	2011-01-04	1	0	1	0	2	1	1	1562	Clear
4	5	2011-01-05	1	0	1	0	3	1	1	1600	Clear

TIME TO ANALYZE OUR CODE!

JUPYTER NOTEBOOK: ANALYSIS PROCESS

(TIME OF DAY)

In [16]:
```
## Output File (CSV)
data_file_day = "Resources/Bike_Share_Season.csv"
#Creating DataFrame
Bike_Share_Season_df=pd.read_csv(data_file_day)
Bike_Share_Season_df.head()
```

Out[16]:

	instant	dteday	season	yr	mnth	hr	holiday	weekday	workingday	weathersit	cnt	Season Name
0	1	2011-01-01	1	0	1	0	0	6	0	1	16	Spring
1	2	2011-01-01	1	0	1	1	0	6	0	1	40	Spring
2	3	2011-01-01	1	0	1	2	0	6	0	1	32	Spring
3	4	2011-01-01	1	0	1	3	0	6	0	1	13	Spring
4	5	2011-01-01	1	0	1	4	0	6	0	1	1	Spring

Time of day vs Bike Rentals

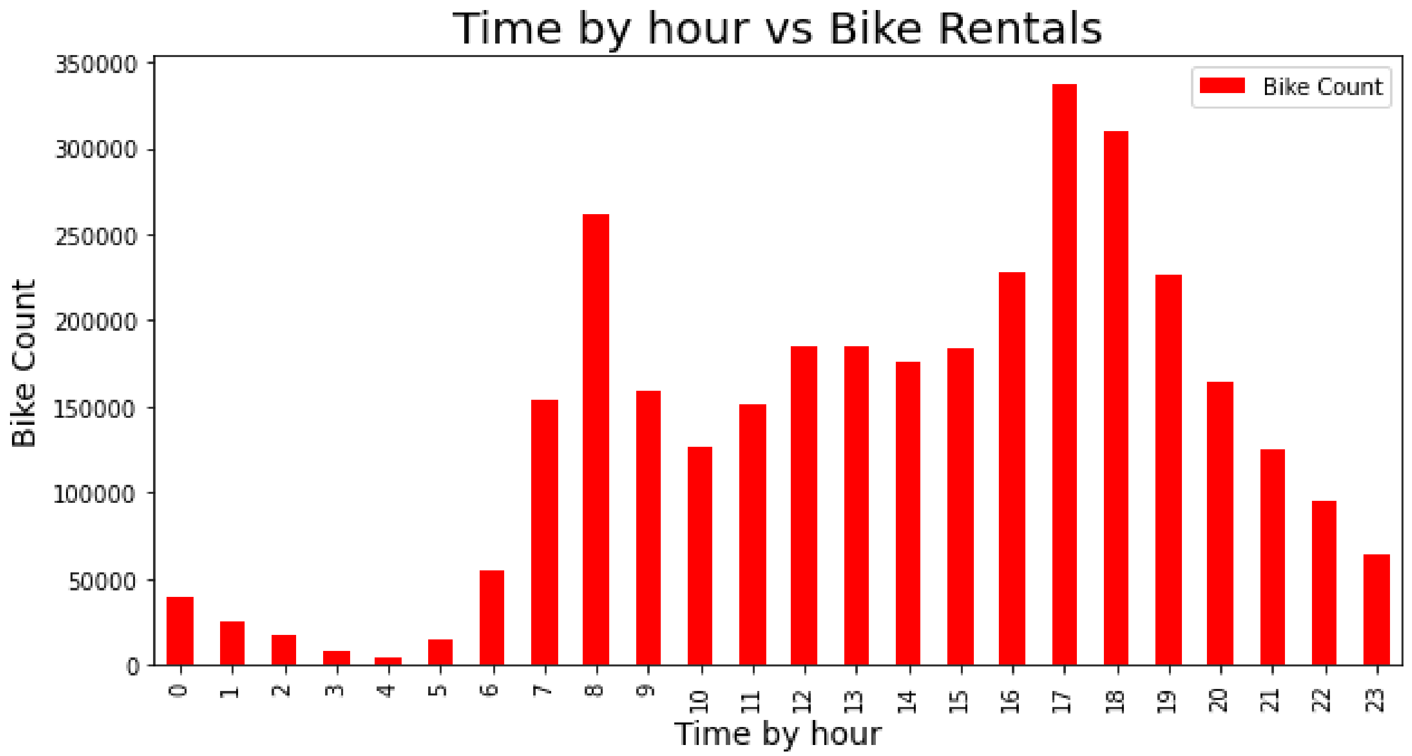
In [17]:
```
#Checking if the Time of the day affetcs Bike Rentals
Bike_Share_Hour=Bike_Share_Season_df.copy()
Bike_Hour_Count=Bike_Share_Hour.groupby(Bike_Share_Hour['hr']).sum()['cnt']
Bike_Hour_Count_df=pd.DataFrame({'Bike Count':Bike_Hour_Count})
Bike_Hour_Count_df
```

Out[17]:

hr	Bike Count
0	39130
1	24164
2	16352
3	8174
4	4428
5	14261
6	55132
7	154171
8	261001
9	159438
10	126257
11	151320
12	184414
13	184919
14	175652

JUPYTER NOTEBOOK: ANALYSIS PROCESS

(TIME OF DAY)



JUPYTER NOTEBOOK: ANALYSIS PROCESS

(TIME OF DAY)

By analyzing the data, we have found that during the early time of 0400 am, there are the least number of rentals.

At the peak time of 1700 (5pm) we found that at that time there are the greatest number of rentals.

In [18]:
```
#Checking for the hour of day with minimum Bike Rentals
Bike_Hour_Count_df.loc[Bike_Hour_Count_df['Bike Count']==Bike_Hour_Count_df['Bike Count'].min()]
```

Out[18]:

	Bike Count
hr	
4	4428

In [19]:
```
#Checking for the hour of day with maximum Bike Rentals
Bike_Hour_Count_df.loc[Bike_Hour_Count_df['Bike Count']==Bike_Hour_Count_df['Bike Count'].max()]
```

Out[19]:

	Bike Count
hr	
17	336860

JUPYTER NOTEBOOK: ANALYSIS PROCESS

(SEASONS)

Seasons vs Bike Rentals

In [21]:
```
#Adding Season labels
Bike_Share_Season=Bike_Share_Hour.copy()

Bike_Share_Season.head()
```

Out[21]:

	instant	dteday	season	yr	mnth	hr	holiday	weekday	workingday	weathersit	cnt	Season Name
0	1	2011-01-01	1	0	1	0	0	6	0	1	16	Spring
1	2	2011-01-01	1	0	1	1	0	6	0	1	40	Spring
2	3	2011-01-01	1	0	1	2	0	6	0	1	32	Spring
3	4	2011-01-01	1	0	1	3	0	6	0	1	13	Spring
4	5	2011-01-01	1	0	1	4	0	6	0	1	1	Spring

Calculating number of bike rentals per season

In [41]:
```
Bike_Season_Count=Bike_Share_Season.groupby(['season','Season Name']).sum()['cnt']
Bike_Season_Count_df=pd.DataFrame({'Bike Count':Bike_Season_Count})
Bike_Season_Count_df
```

Out[41]:

		Bike Count
season	Season Name	
1	Spring	471348
2	Summer	918589
3	Fall	1061129
4	Winter	841613

JUPYTER NOTEBOOK: ANALYSIS PROCESS

(SEASONS)

In [26]:
```
#For TTest
Bike_df=Bike_Share_Season.groupby(Bike_Share_Season['season']).sum()['cnt']
Bike_df=pd.DataFrame({'Bike Count':Bike_df})
Bike_df=Bike_df.reset_index()
Bike_df
```

Out[26]:

	season	Bike Count
0	1	471348
1	2	918589
2	3	1061129
3	4	841613

In [27]:
```
#Checking for the season with minimum Bike Rentals
Bike_Season_Count_df.loc[Bike_Season_Count_df['Bike Count']==Bike_Season_Count_df['Bike Count'].min()]
```

Out[27]:

	Bike Count
Season Name	
Spring	471348

In [28]:
```
#Checking for the season with maximum Bike Rentals
Bike_Season_Count_df.loc[Bike_Season_Count_df['Bike Count']==Bike_Season_Count_df['Bike Count'].max()]
```

Out[28]:

	Bike Count
Season Name	
Fall	1061129

Fall has the highest number of people renting bikes, while Spring has the lowest amount.

JUPYTER NOTEBOOK: ANALYSIS PROCESS

(SEASONS)

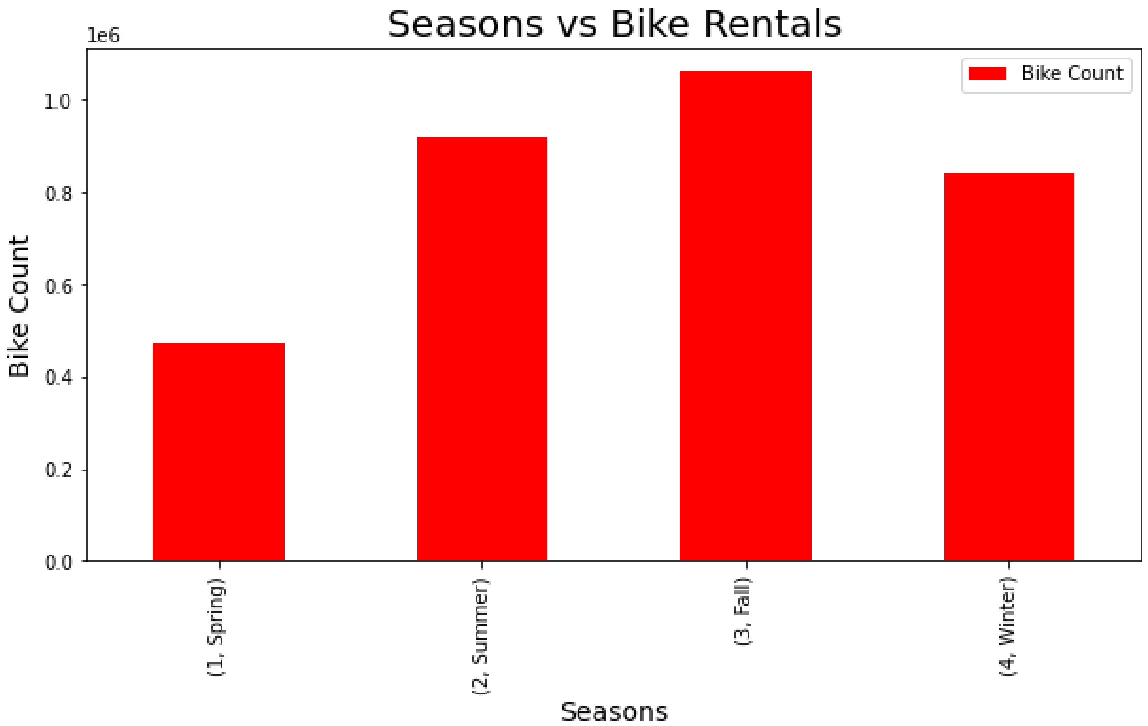
Bar chart

In [42]:
```
#Plotting the bike rental count per season
Bike_Season_Count_df
```

Out[42]:

		Bike Count
season	**Season Name**	
1	**Spring**	471348
2	**Summer**	918589
3	**Fall**	1061129
4	**Winter**	841613

In [43]:
```
Bike_Season_Count_df.plot.bar(figsize=(10,5), color='r',fontsize = 10)
plt.xlabel("Seasons",fontsize = 14)
plt.ylabel("Bike Count",fontsize = 14)
plt.title("Seasons vs Bike Rentals",fontsize = 20)
plt.savefig('Output/barplot_Seasons.png', bbox_inches = "tight")
plt.show()
```

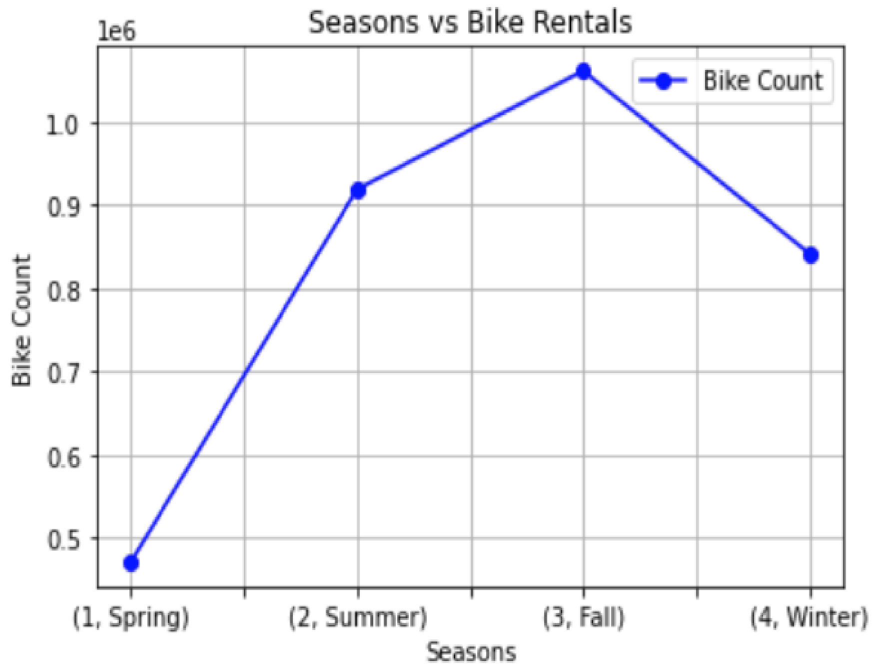


JUPYTER NOTEBOOK:

ANALYSIS PROCESS

(SEASONS)

In [62]:
```
Bike_Season_Count_df.plot(marker="o",color="blue")
plt.title('Seasons vs Bike Rentals')
plt.xlabel('Seasons')
plt.ylabel('Bike Count')
plt.grid()
plt.savefig('Output/Lineplot_Seasons.png', bbox_inches = "tight")
plt.show()
```



JUPYTER NOTEBOOK: ANALYSIS PROCESS

(SEASONS)

- CHI SQUARED TEST

In [44]:
```
#Chisquared test
#st.ttest_ind(Bike_df['season'],Bike_df['Bike Count'],equal_var=False)
observed = pd.Series([471348, 918589, 1061129, 841613], index=["1", "2", "3", "4"])

# Create a data frame
df = pd.DataFrame([observed]).T

# Add a column whose default values are the expected values
df[3] = 823169

# Rename the columns
df.columns = ["observed", "expected"]

# View the data frame
df
```

Out[44]:

	observed	expected
1	471348	823169
2	918589	823169
3	1061129	823169
4	841613	823169

In [45]:
```
criticalvalue=st.chi2.ppf(q=.95,df=3)
criticalvalue
```

Out[45]: 7.814727903251179

In [46]:
```
st.chisquare(df['observed'],df['expected'])
```

Out[46]: Power_divergenceResult(statistic=230630.81235687932, pvalue=0.0)

We can conclude this data is significant because the Statistic number is greater than the critical value.

Weather patterns vs Bike Rentals

In [39]:
```
#Amber
Bike_Day_Count=Bike_Share_Day.groupby(Bike_Share_Day['Weather']).sum()['cnt']

Bike_Day_Count_df=pd.DataFrame({'Rental Count via weather type':Bike_Day_Count})
Bike_Day_Count_df
```

Out[39]:

	Rental Count via weather type
Weather	
Clear	2257952
Cloudy	996858
Rainy	37869

JUPYTER NOTEBOOK: ANALYSIS PROCESS

WEATHER TYPE

JUPYTER NOTEBOOK:

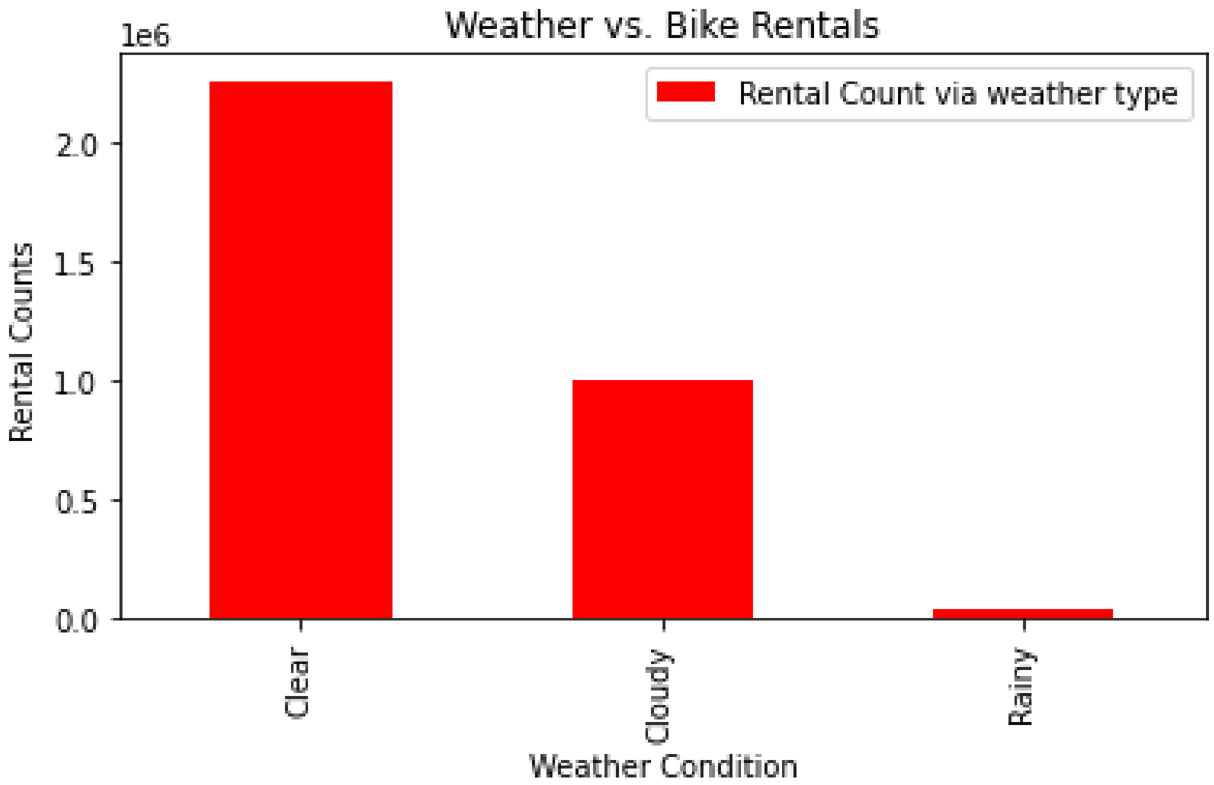
ANALYSIS PROCESS

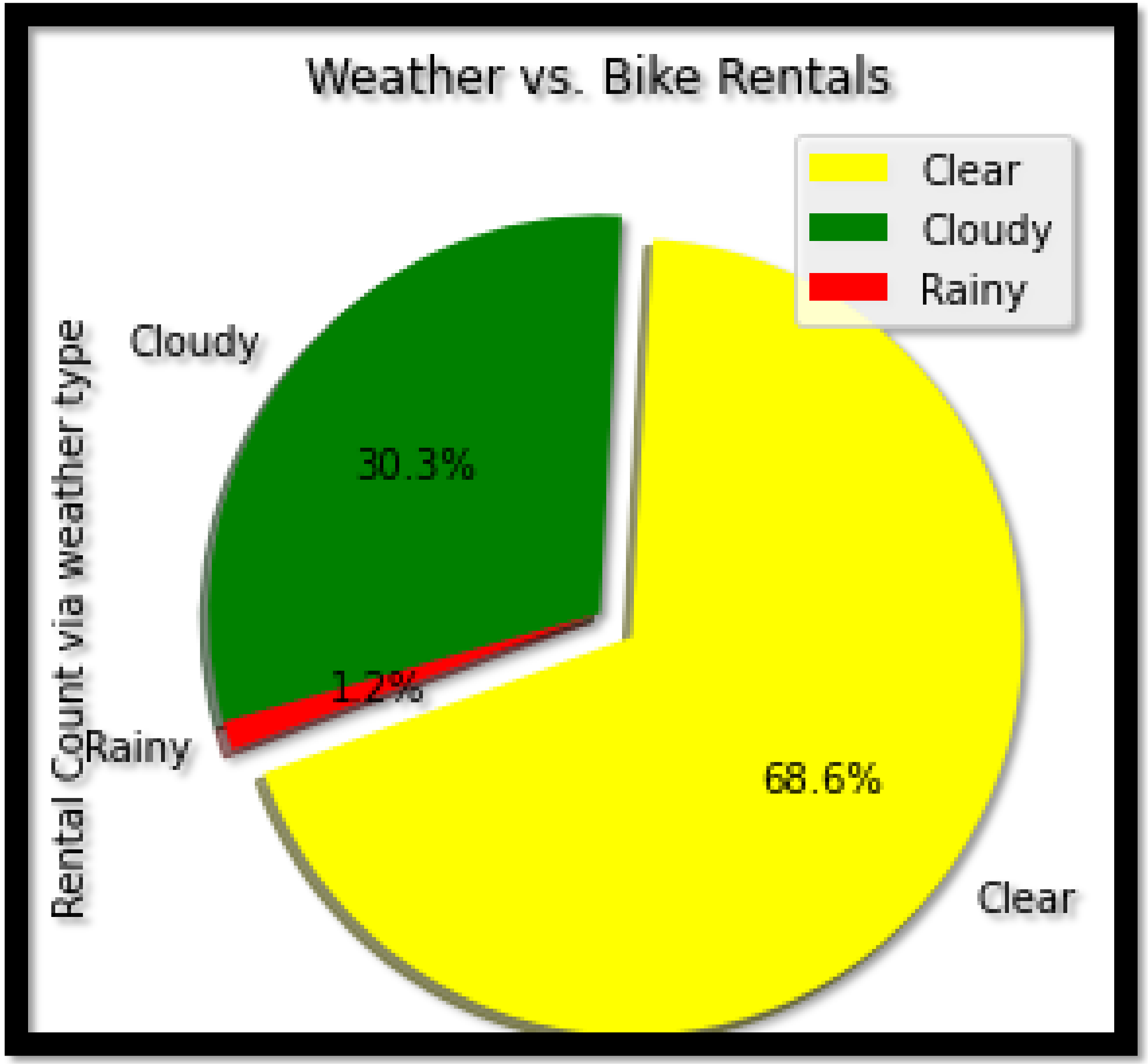
WEATHER TYPE

Plotting the Data

In [40]:
```
Bike_Day_Count_df.plot(kind="bar", facecolor="red")

plt.title ("Weather vs. Bike Rentals")
plt.ylabel("Rental Counts")
plt.xlabel("Weather Condition")
plt.tight_layout()
plt.savefig('Output/barplot_Weather.png', bbox_inches = "tight")
plt.show()
```





JUPYTER NOTEBOOK: ANALYSIS PROCESS

WEATHER TYPE

- From this pie chart generated by the data, we can see that 68.6% of renters used bike sharing on a clear day.

JUPYTER NOTEBOOK: ANALYSIS PROCESS

(WEATHER)

- CHI SQUARED TEST

In [42]:
```
observed = pd.Series([2257952,996858,37869], index=["Clear", "Cloudy", "Rainy"])

df = pd.DataFrame([observed]).T

df[2] = 1097560

df.columns = ["observed", "expected"]

df
```

Out[42]:

	observed	expected
Clear	2257952	1097560
Cloudy	996858	1097560
Rainy	37869	1097560

In [43]:
```
# The degree of freedom is 3-1 = 2
# With a p-value of 0.05, the confidence level is 1.00-0.05 = 0.95.
critical_value = st.chi2.ppf(q = 0.95, df = 2)
```

In [44]:
```
# The critical value
critical_value
```

Out[44]: 5.991464547107979

In [45]:
```
# Run the chi square test with stats.chisquare()
st.chisquare(df['observed'], df['expected'])
```

Out[45]: Power_divergenceResult(statistic=2259189.0210548854, pvalue=0.0)

We can conclude this data is significant because the Statistic number is greater than the critical value.

JUPYTER NOTEBOOK: ANALYSIS PROCESS

TYPE OF DAY

Weekday versus Weekend versus Holidays Bike Rentals

In [59]:
```
# Clean up the df to only have types of days and counts
Bike_Share_Day_Drop = Bike_Share_Day[['weekday', 'workingday', 'holiday', 'cnt']]
Bike_Share_Day_Drop['cnt'].sum()
# Number of rentals over the workingdays

# Identifying the day types (could alternatively use groupby method)
WorkingDay_Bike_Share = Bike_Share_Day_Drop.loc[Bike_Share_Day_Drop['workingday'] == 1]
Weekends_Bike_Share = Bike_Share_Day_Drop.loc[(Bike_Share_Day_Drop['weekday'] == 0) | (Bike_Share_Day_Drop['weekday'] == 6)]
Holiday_Bike_Share = Bike_Share_Day_Drop.loc[Bike_Share_Day_Drop['holiday'] == 1]

# Summing number of rentals per day type
WorkingDay_Counts = WorkingDay_Bike_Share['cnt'].sum()
WeekendsDay_Counts = Weekends_Bike_Share['cnt'].sum()
HolidayDays_Counts = Holiday_Bike_Share['cnt'].sum()

# printing results to see what we're working with
print(f'Working Days Count: {WorkingDay_Counts}, Weekend Days Count: {WeekendsDay_Counts} and Holidays Count: {HolidayDays_Cou
nts}')

# Verifying counts for sanity check
totalbike_counts = Bike_Share_Day_Drop['cnt'].sum()
totalbike_counts_2 = WorkingDay_Counts + WeekendsDay_Counts + HolidayDays_Counts
print(f' OG Counts {totalbike_counts}, OG verification {totalbike_counts_2}')

# getting average counts per diem
Number_of_WorkingDays = WorkingDay_Bike_Share['cnt'].count()
WorkingDay_Counts_perdiem = WorkingDay_Counts/Number_of_WorkingDays

Number_of_Weekend_Days = Weekends_Bike_Share['cnt'].count()
WeekendDay_Counts_perdiem = WeekendsDay_Counts/Number_of_Weekend_Days

Number_of_Holidays = Holiday_Bike_Share['cnt'].count()
HoliDay_Counts_perdiem = HolidayDays_Counts/Number_of_Holidays
```

JUPYTER NOTEBOOK: ANALYSIS PROCESS

TYPE OF DAY

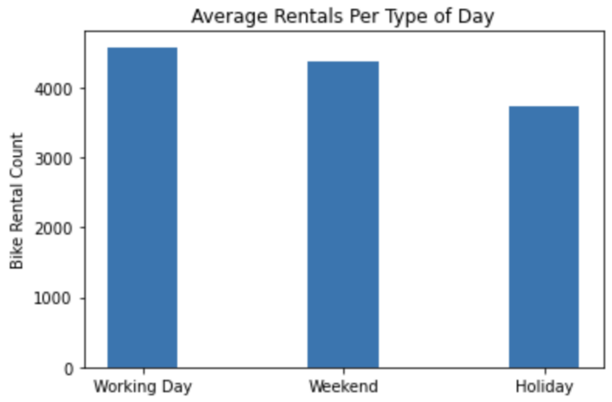
```
# printing numbers to see what we're working with
print(f' Workingdays {Number_of_WorkingDays}, Weekends {Number_of_Weekend_Days}, Holidays {Number_of_Holidays}')
print(f' Total Days {Number_of_WorkingDays + Number_of_Weekend_Days + Number_of_Holidays}')
print(f" Working Days Average Ride Count: {WorkingDay_Counts_perdiem}, Weekend Days Average Ride Count: {WeekendDay_Counts_per
diem}, Holidays Average Ride Count: {HoliDay_Counts_perdiem}")
```

```
Working Days Count: 2292410, Weekend Days Count: 921834 and Holidays Count: 78435
 OG Counts 3292679, OG verification 3292679
 Workingdays 500, Weekends 210, Holidays 21
 Total Days 731
 Working Days Average Ride Count: 4584.82, Weekend Days Average Ride Count: 4389.685714285714, Holidays Average Ride Count: 37
35.0
```

In [61]:

```
# creating the bar chart to visualize the data
x = np.arange(len(height))
height = [WorkingDay_Counts_perdiem, WeekendDay_Counts_perdiem, HoliDay_Counts_perdiem]
tick_locations = [value for value in x]
plt.xticks(tick_locations, ["Working Day", "Weekend", "Holiday"])
plt.title("Average Rentals Per Type of Day")
plt.ylabel('Bike Rental Count')
plt.bar(x, height, width=0.35, bottom=None, align='center', data=None)
```

Out[61]: <BarContainer object of 3 artists>



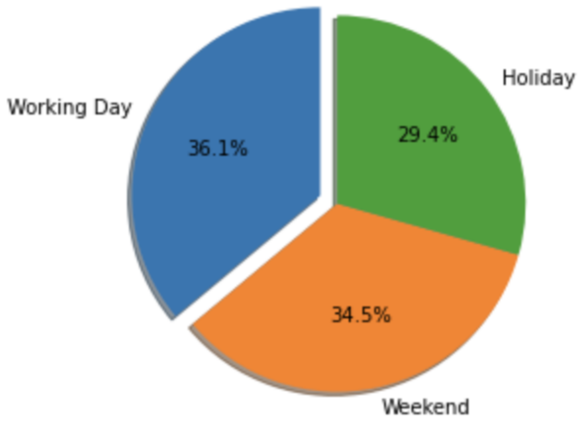
JUPYTER NOTEBOOK: ANALYSIS PROCESS

TYPE OF DAY

In [67]:
```
labels = 'Working Day', 'Weekend', 'Holiday'
sizes = [WorkingDay_Counts_perdiem, WeekendDay_Counts_perdiem, HoliDay_Counts_perdiem]
explode = (0.1, 0, 0)

fig1, ax1 = plt.subplots()
ax1.pie(sizes, explode=explode, labels=labels, autopct='%1.1f%%',
        shadow=True, startangle=90)
ax1.axis('equal')

plt.show()
```



In [65]:
```
# Performing ANOVA test on the number of rentals for each type of day.
group1 = WorkingDay_Bike_Share['cnt']
group2 = Weekends_Bike_Share['cnt']
group3 = Holiday_Bike_Share['cnt']
stats.f_oneway(group1, group2, group3)
```

Out[65]:
```
F_onewayResult(statistic=2.465184866770529, pvalue=0.08570238266758495)
```

SUMMARY OF CONCLUSIONS:

The Bike-sharing rental process is highly correlated to environmental and seasonal settings.

WHAT DO THESE FINDINGS MEAN?

These findings conclude that while there is an even number on bikes used on any given day, there are far greater spikes in usage during the Fall season and when the weather has clear skies.

Not many people like to rent these bikes when those Spring showers hit!

REFERENCES:

The core data set is related to the two-year historical log corresponding to years 2011 and 2012 from Capital Bikeshare system, Washington D.C., USA which is publicly available in http://capitalbikeshare.com/system-data.

Weather information are extracted from http://www.freemeteo.com.

QUESTIONS ?