

✓ Welcome to Colab!

Explore the Gemini API

The Gemini API gives you access to Gemini models created by Google DeepMind. Gemini models are built from the ground up to be multimodal, so you can reason seamlessly across text, images, code, and audio.

How to get started?

- Go to [Google AI Studio](#) and log in with your Google account.
- [Create an API key](#).
- Use a quickstart for [Python](#), or call the REST API using [curl](#).

Discover Gemini's advanced capabilities

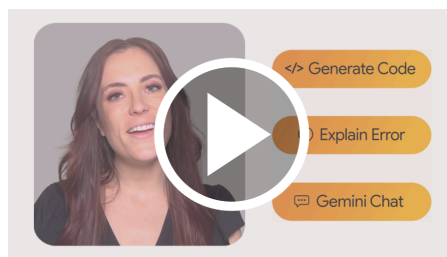
- Play with Gemini [multimodal outputs](#), mixing text and images in an iterative way.
- Discover the [multimodal Live API](#) (demo [here](#)).
- Learn how to [analyze images and detect items in your pictures](#) using Gemini (bonus, there's a [3D version](#) as well!).
- Unlock the power of [Gemini thinking model](#), capable of solving complex task with its inner thoughts.

Explore complex use cases

- Use [Gemini grounding capabilities](#) to create a report on a company based on what the model can find on internet.
- Extract [invoices and form data from PDF](#) in a structured way.
- Create [illustrations based on a whole book](#) using Gemini large context window and Imagen.

To learn more, check out the [Gemini cookbook](#) or visit the [Gemini API documentation](#).

Colab now has AI features powered by [Gemini](#). The video below provides information on how to use these features, whether you're new to Python, or a seasoned veteran.



What is Colab?

Colab, or "Colaboratory", allows you to write and execute Python in your browser, with

- Zero configuration required
- Access to GPUs free of charge
- Easy sharing

Whether you're a **student**, a **data scientist** or an **AI researcher**, Colab can make your work easier. Watch [Introduction to Colab](#) or [Colab Features You May Have Missed](#) to learn more, or just get started below!

✓ Getting started

The document you are reading is not a static web page, but an interactive environment called a **Colab notebook** that lets you write and execute code.

For example, here is a **code cell** with a short Python script that computes a value, stores it in a variable, and prints the result:

```
seconds_in_a_day = 24 * 60 * 60
seconds_in_a_day
```

 86400

To execute the code in the above cell, select it with a click and then either press the play button to the left of the code, or use the keyboard shortcut "Command/Ctrl+Enter". To edit the code, just click the cell and start editing.

Variables that you define in one cell can later be used in other cells:

```
seconds_in_a_week = 7 * seconds_in_a_day
seconds_in_a_week
```

↔ 604800

Colab notebooks allow you to combine **executable code** and **rich text** in a single document, along with **images**, **HTML**, **LaTeX** and more. When you create your own Colab notebooks, they are stored in your Google Drive account. You can easily share your Colab notebooks with co-workers or friends, allowing them to comment on your notebooks or even edit them. To learn more, see [Overview of Colab](#). To create a new Colab notebook you can use the File menu above, or use the following link: [create a new Colab notebook](#).

Colab notebooks are Jupyter notebooks that are hosted by Colab. To learn more about the Jupyter project, see [jupyter.org](#).

✓ Data science

With Colab you can harness the full power of popular Python libraries to analyze and visualize data. The code cell below uses **numpy** to generate some random data, and uses **matplotlib** to visualize it. To edit the code, just click the cell and start editing.

You can import your own data into Colab notebooks from your Google Drive account, including from spreadsheets, as well as from Github and many other sources. To learn more about importing data, and how Colab can be used for data science, see the links below under [Working with Data](#).

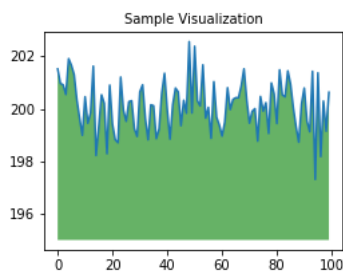
```
import numpy as np
import IPython.display as display
from matplotlib import pyplot as plt
import io
import base64

ys = 200 + np.random.randn(100)
x = [x for x in range(len(ys))]

fig = plt.figure(figsize=(4, 3), facecolor='w')
plt.plot(x, ys, '-')
plt.fill_between(x, ys, 195, where=(ys > 195), facecolor='g', alpha=0.6)
plt.title("Sample Visualization", fontsize=10)

data = io.BytesIO()
plt.savefig(data)
image = F"data:image/png;base64,{base64.b64encode(data.getvalue()).decode()}"
alt = "Sample Visualization"
display.display(display.Markdown(F"!!![{alt}]({image})"))
plt.close(fig)
```

↔



Colab notebooks execute code on Google's cloud servers, meaning you can leverage the power of Google hardware, including [GPUs and TPUs](#), regardless of the power of your machine. All you need is a browser.

For example, if you find yourself waiting for **pandas** code to finish running and want to go faster, you can switch to a GPU Runtime and use libraries like [RAPIDS cuDF](#) that provide zero-code-change acceleration.

To learn more about accelerating pandas on Colab, see the [10 minute guide](#) or [US stock market data analysis demo](#).

✓ Machine learning

With Colab you can import an image dataset, train an image classifier on it, and evaluate the model, all in just [a few lines of code](#).

Colab is used extensively in the machine learning community with applications including:

- Getting started with TensorFlow
- Developing and training neural networks

- Experimenting with TPUs
- Disseminating AI research
- Creating tutorials

To see sample Colab notebooks that demonstrate machine learning applications, see the [machine learning examples](#) below.

✓ More Resources

Working with Notebooks in Colab

- [Overview of Colab](#)
- [Guide to Markdown](#)
- [Importing libraries and installing dependencies](#)
- [Saving and loading notebooks in GitHub](#)
- [Interactive forms](#)
- [Interactive widgets](#)

Working with Data

- [Loading data: Drive, Sheets, and Google Cloud Storage](#)
- [Charts: visualizing data](#)
- [Getting started with BigQuery](#)

Machine Learning

These are a few of the notebooks related to Machine Learning, including Google's online Machine Learning course. See the [full course website](#) for more.

- [Intro to Pandas DataFrame](#)
- [Intro to RAPIDS cuDF to accelerate pandas](#)
- [Getting Started with cuML's accelerator mode](#)
- [Linear regression with tf.keras using synthetic data](#)

Using Accelerated Hardware

- [TensorFlow with GPUs](#)
- [TPUs in Colab](#)

✓ Featured examples

- [Retraining an Image Classifier](#): Build a Keras model on top of a pre-trained image classifier to distinguish flowers.
- [Text Classification](#): Classify IMDB movie reviews as either *positive* or *negative*.
- [Style Transfer](#): Use deep learning to transfer style between images.
- [Multilingual Universal Sentence Encoder Q&A](#): Use a machine learning model to answer questions from the SQuAD dataset.
- [Video Interpolation](#): Predict what happened in a video between the first and the last frame.

Below are the code which import the necessary libraries in Python and also the data has been uploaded and by doing `df.head` we are actually showing the top 5 rows of the data

Yulu is India's leading micro-mobility service provider, which offers unique vehicles for the daily commute. Starting off as a mission to eliminate traffic congestion in India, Yulu provides the safest commute solution through a user-friendly mobile app to enable shared, solo and sustainable commuting.

Yulu zones are located at all the appropriate locations (including metro stations, bus stands, office spaces, residential areas, corporate offices, etc) to make those first and last miles smooth, affordable, and convenient!


Yulu has recently suffered considerable dips in its revenues. They have contracted a consulting company to understand the factors on which the demand for these shared electric cycles depends. Specifically, they want to understand the factors affecting the demand for these shared electric cycles in the Indian market.

The company wants to know:



1.Which variables are significant in predicting the demand for shared electric cycles in the Indian market? 2.How well those variables describe the electric cycle demands? 3.Are the different parameters related with each other? 4.How can the customer experience can be improved?

By looking at the data we can say that the except the datetime all the columns are either integer or float that means they are basically the numbers but the columns 'season', 'holiday', 'working day', 'weather' these are actually the categorical columns because these are having very few numbers. But still they are having very few numbers so there basically categorical column as already mention in the data. For season: when the number is 1 that means it is a spring season. when the number is 2 that means it's summer when the number is 3 it's fall when the number is 4 its winter. Similarly weather column is also like a categorical column: when the number is 1 weather is clear few cloud party cloudy when the number is 2 the weather is mist plus cloudy when the number is 3 it's light snow, light rain some thunderstorm and when the number is 4 it's heavy rain plus eyes palette and thunderstorm, mist snow and fog

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
df=pd.read_csv('/content/yulu_bike_sharing.csv')
df.head()
```



	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count
0	2011-01-01 00:00:00	1	0	0	1	9.84	14.395	81	0.0	3	13	16
1	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	0.0	8	32	40
2	2011-01-01 02:00:00	1	0	0	1	9.02	13.635	80	0.0	5	27	32
3	2011-01-01 03:00:00	1	0	0	1	9.84	14.395	75	0.0	3	10	13
4	2011-01-01 04:00:00	1	0	0	1	9.84	14.395	75	0.0	0	1	1

Next steps:

[Generate code with df](#)

[View recommended plots](#)


[New interactive sheet](#)

To avoid any confusion we are changing the name of the 'count' column as 'total' because count is actually the total number of vehicles maybe casual or registered so that is why count column is being changed to total


```
df.rename(columns={'count':'total'},inplace=True)
```

By describe function we can have all the numerical columns with their count, mean, standard deviation, minimum ,lower quartile, upper quartile median and maximum. There are total 10886 rows. But looking at this data we cannot have any conclusion so we will analyse each of this columns separately or maybe two three columns together to have a better conclusion and insights

```
df.describe()
```



	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	re
count	10886.000000	10886.000000	10886.000000	10886.000000	10886.000000	10886.000000	10886.000000	10886.000000	10886.000000	10886.000000
mean	2.506614	0.028569	0.680875	1.418427	20.23086	23.655084	61.886460	12.799395	36.021955	16.021955
std	1.116174	0.166599	0.466159	0.633839	7.79159	8.474601	19.245033	8.164537	49.960477	16.021955
min	1.000000	0.000000	0.000000	1.000000	0.82000	0.760000	0.000000	0.000000	0.000000	0.000000
25%	2.000000	0.000000	0.000000	1.000000	13.94000	16.665000	47.000000	7.001500	4.000000	3.000000
50%	3.000000	0.000000	1.000000	1.000000	20.50000	24.240000	62.000000	12.998000	17.000000	11.000000
75%	4.000000	0.000000	1.000000	2.000000	26.24000	31.060000	77.000000	16.997900	49.000000	22.000000
max	4.000000	1.000000	1.000000	4.000000	41.00000	45.455000	100.000000	56.996900	367.000000	88.000000



So we can see that none of the columns is having a single null value in the data set so it's a good point that the data set is complete and accurate

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

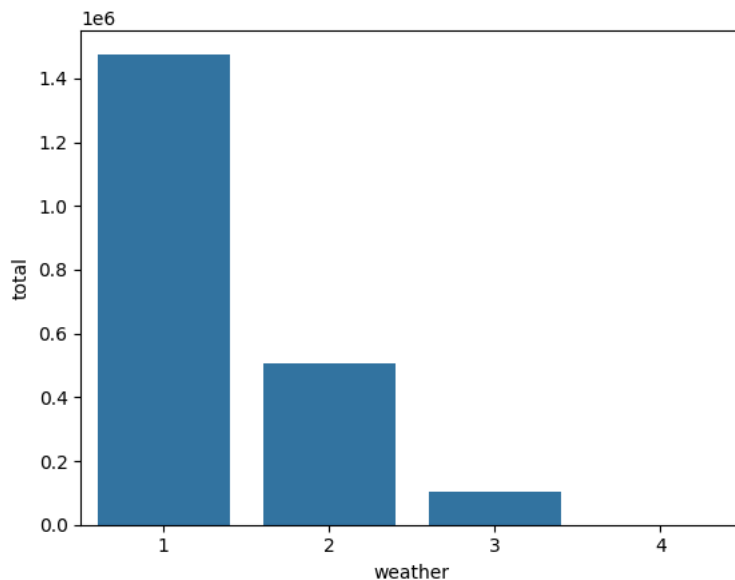
	0
datetime	0
season	0
holiday	0
workingday	0
weather	0
temp	0
atemp	0
humidity	0
windspeed	0
casual	0
registered	0
total	0

dtype: int64

weather column is also like a categorical column: when the number is 1 weather is clear few cloud or partly cloudy when the number is 2 the weather is mist plus cloudy when the number is 3 it's light snow, light rain some thunderstorm and when the number is 4 it's heavy rain plus eyes palette and thunderstorm,mist snow and fog **So we can see that most of the booking has been done when the weather is one which means that when the weather is clear cloud or party partly cloud.** ACTIONABLE SUGGESTIONS: So we can see weather forecasting and if the weather is clear cloud or partly cloudy then that time YULU should have more bikes to take booking**

```
b=df.groupby('weather')['total'].sum()
print(b)
sns.barplot(b)
```

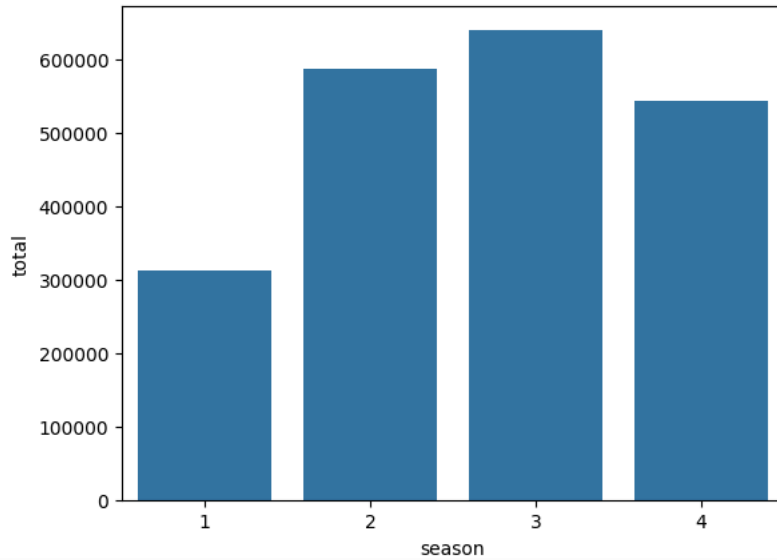
```
weather
1    1476063
2     507160
3     102089
4         164
Name: total, dtype: int64
<Axes: xlabel='weather', ylabel='total'>
```



For season: when the number is 1 that means it is a spring season. when the number is 2 that means it's summer when the number is 3 it's fall when the number is 4 its winter. **observations:**So looking at the bar plot of the season we can see that the maximum number of booking is coming when the season is fall and almost similar kind of booking is coming when the season is summer and winter but we can see that very less number of booking is coming when the season is spring so we should give some offers and discounts during the spring season so that the number of bookings goes up.

```
a=df.groupby('season')['total'].sum()
sns.barplot(a)
```

<Axes: xlabel='season', ylabel='total'>



so when the weather is 4 which means there is heavy rainfall and ice pallette, there is one row of data in the whole dataset. This suggests that this kind of weather is very rare and the bookings at this time are also very high

```
df[df['weather']==4]
```

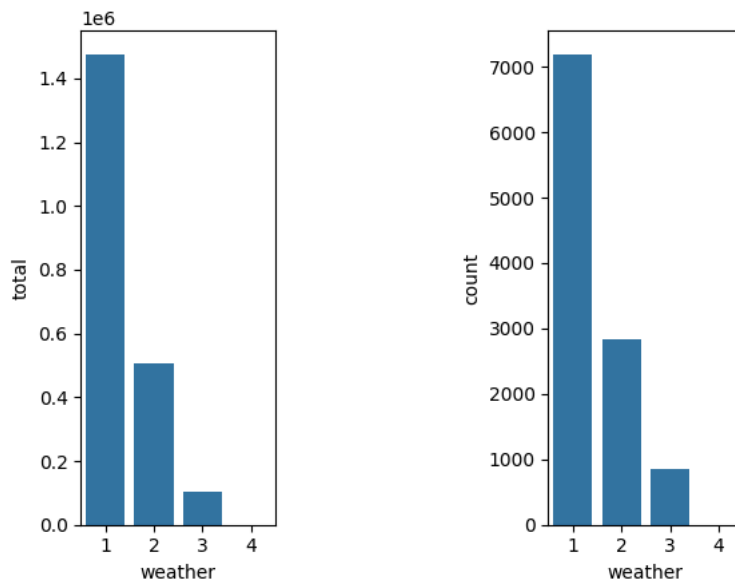
<Axes: xlabel='weather', ylabel='count'>

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	total
5631	2012-01-09 18:00:00	1	0	1	4	8.2	11.365	86	6.0032	6	158	164

Below are the two bar plots. the second bar plot is basically the weather and the number of days for which the data has been recorded so we can see from the graph that the maximum number of days in the data, weather was like clean or party cloudy and due to which the total number of bookings is also maximum for partly cloudy or clean weather whcih can be seen from the first subplot


```
plt.subplot(1,3,1)
sns.barplot(df.groupby('weather')['total'].sum())
plt.subplot(1,3,3)
sns.barplot(df['weather'].value_counts())
```

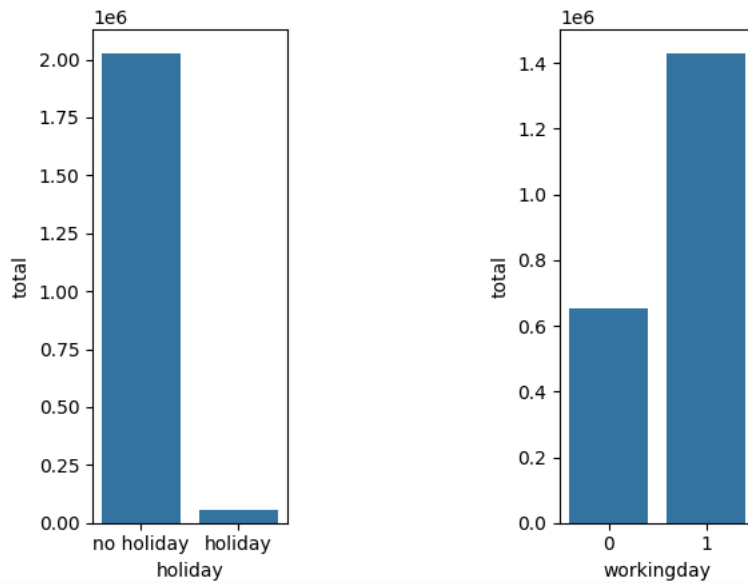
<Axes: xlabel='weather', ylabel='count'>



So according to the below graphs the number of booking is very high when there is no holiday and during the holiday the number of bookings are very low. Also looking at the working day graph it can be said that during the working days the number of booking is very high so it is suggested that **during the non holiday and the working days the number of available bikes should be very high**


```
plt.subplot(1,3,1)
a=sns.barplot(df.groupby('holiday')['total'].sum())
a.set_xticklabels(['no holiday','holiday'])
plt.subplot(1,3,3)
sns.barplot(df.groupby('workingday')['total'].sum())
```

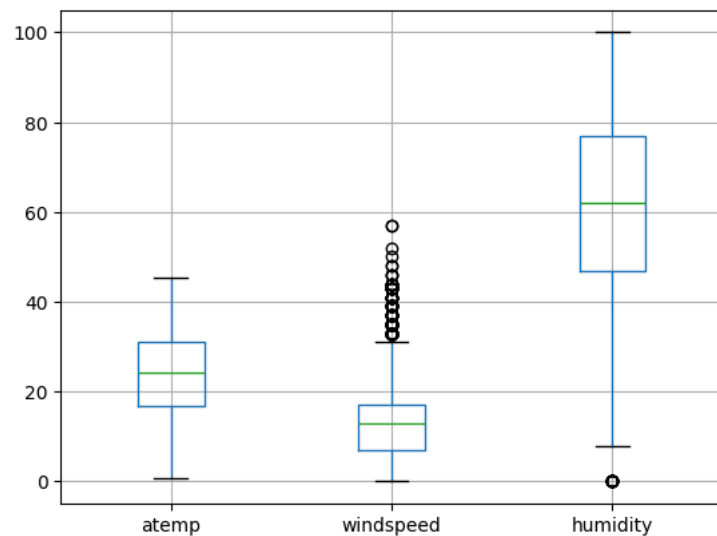
 <ipython-input-16-fba085e239f8>:3: UserWarning: set_ticklabels() should only be used with a fixed number of ticks, i.e. after set_ticks()
a.set_xticklabels(['no holiday','holiday'])
<Axes: xlabel='workingday', ylabel='total'>



By looking below boxplots we can conclude that atemp and humidity have no outliers but windspeed has some outliers

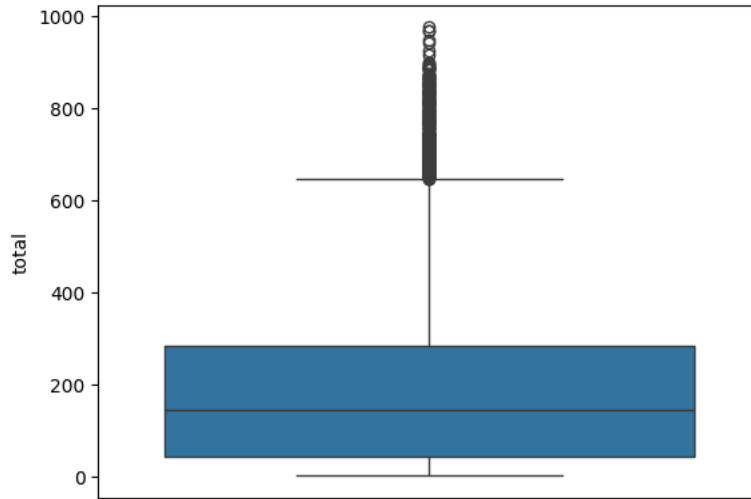
```
list=['atemp','windspeed','humidity']
df[list].boxplot()
```

 <Axes: >



```
sns.boxplot(df['total'])
```

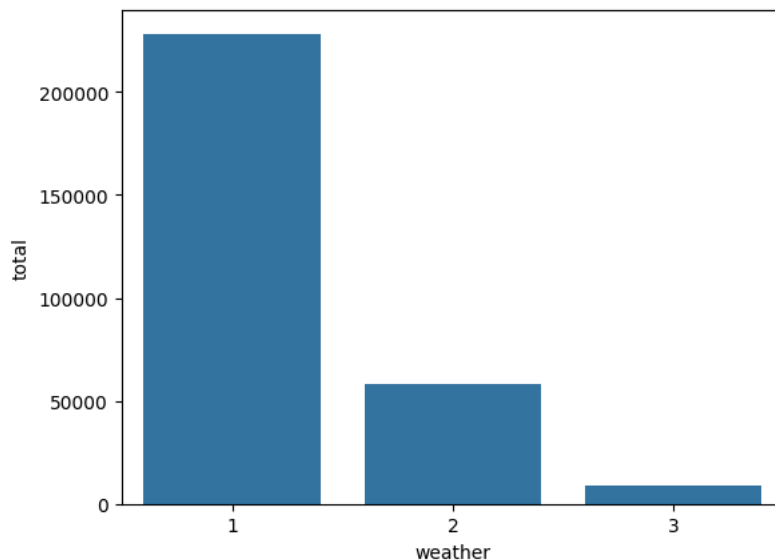
↩ <Axes: ylabel='total'>



Below is the graph of weather and total when the number of booking is more than 600 in a day. we can see here also the booking is very high when there is clean cloud or partly cloud weather

```
a=df[df['total']>600]
a.groupby('weather')['total'].sum()
sns.barplot(a.groupby('weather')['total'].sum())
```

↩ <Axes: xlabel='weather', ylabel='total'>



2- Sample T-Test to check if Working Day has an effect on the number of electric cycles rented

What are the null and alternate hypothesis?

Null hypothesis(H_0): There is no effect of working day on number of bookings ie the mean booking in a working day is same as the mean bookings in a non working day

Alternate hypothesis(H_a): There is effect on number of bookings and the working day ie the mean number of bookings in a working day is not equal to mean number of bookings in non working day

For instance let us take significance level 0.05

```
from scipy.stats import ttest_ind
booking1=df[df['workingday']==0]['total']
booking2=df[df['workingday']==1]['total']
t_stat, pvalue = ttest_ind(booking1, booking2)
if pvalue < 0.05:
    print("Reject the null hypothesis,which means the mean booking in working and non working day are not same")
else:
    print("Fail to reject the null hypothesis,which means the mean booking in working and non working day are same")
```

↩ Fail to reject the null hypothesis,which means the mean booking in working and non working day are same

so we can see the pvalue is greater than alpha . **so working day has no impact on number of bookings**

ANNOVA to check if No. of cycles rented is similar or different in different weather Let significance level alpha be 0.05

Null hypothesis(H0)=There is no effect of weather on booking Alternate hypothesis(Ha):There is effect of weather on booking

```
from scipy.stats import f_oneway
booking1=df[df['weather']==1]['total']
booking2=df[df['weather']==2]['total']
booking3=df[df['weather']==3]['total']
booking4=df[df['weather']==4]['total']
stat,pvalue=f_oneway(booking1,booking2,booking3,booking4)
if pvalue<0.05:
    print('reject null hypothesis,There is effect of weather on booking')
else:
    print('fail to reject null hypothesis,There is no effect of weather on booking')
```

→ reject null hypothesis,There is effect of weather on booking

ANNOVA to check if No. of cycles rented is similar or different in different season. Let significance level alpha be 0.05

Null hypothesis(H0)=There is no effect of season on booking Alternate hypothesis(Ha):There is effect of season on booking

```
from scipy.stats import f_oneway
booking1=df[df['season']==1]['total']
booking2=df[df['season']==2]['total']
booking3=df[df['season']==3]['total']
booking4=df[df['season']==4]['total']
stat,pvalue=f_oneway(booking1,booking2,booking3,booking4)
if pvalue<0.05:
    print('reject null hypothesis,There is effect of season on booking')
else:
    print('fail to reject null hypothesis,There is no effect of season on booking')
```

→ reject null hypothesis,There is effect of season on booking

Chi-square test to check if Weather is dependent on the season

Null hypothesis(H0):There is no effect of season on weather Alternate hypothesis:There is some effect of season on weather

Let us assume that significance level be 0.05

```
from scipy.stats import chi2_contingency
contingency_table=pd.crosstab(df['season'],df['weather'])
#perform the Chi-square test
chi2_stat, pvalue, dof, expected = chi2_contingency(contingency_table)
if pvalue<0.05:
    print('reject null hypothesis,There is effect of season on weather')
else:
    print('fail to reject null hypothesis,There is no effect of season on weather')
```

→ reject null hypothesis,There is effect of season on weather

Looking at the graph below we can see that the bookings are high during 7:00 a.m. to 9:00 a.m. in the morning and then it's moderate from 12 to 3:00 p.m. and then again from 4:00 p.m. the bookings are very high.In the evening 5:00 p.m. to 6:00 p.m. the bookings are very high so we can keep the supply in such a way that when the demand is maximum the supply is fulfilled

```
df['hour'] = pd.to_datetime(df['datetime']).dt.hour
booking_by_hour=df.groupby('hour')['total'].sum()
sns.barplot(booking_by_hour)
```

Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.