

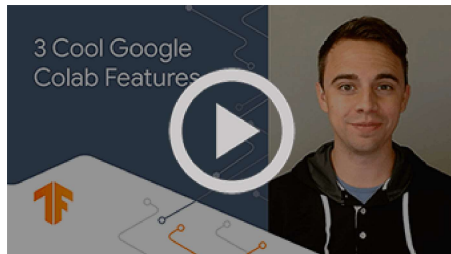


Welcome to Colab!

(New) Try the Gemini API

- [Generate a Gemini API key](#)
- [Talk to Gemini with the Speech-to-Text API](#)
- [Gemini API: Quickstart with Python](#)
- [Gemini API code sample](#)
- [Compare Gemini with ChatGPT](#)
- [More notebooks](#)

If you're already familiar with Colab, check out this video to learn about interactive tables, the executed code history view and the command palette.



Start coding or [generate](#) with AI.

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](#) to find out more, or just get started below!

✓ Getting started

The document that 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 find out 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 find out more about the Jupyter project, see jupyter.org.

✓ Data science

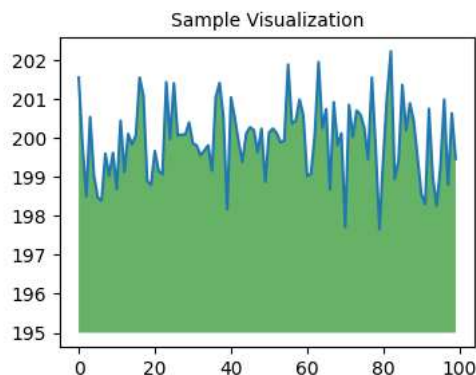
With Colab you can harness the full power of popular Python libraries to analyse and visualise data. The code cell below uses **numpy** to generate some random data, and uses **matplotlib** to visualise it. To edit the code, just click the cell and start editing.

```
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)
```



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 find out more about importing data, and how Colab can be used for data science, see the links below under [Working with data](#).

✓ 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 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.

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 Colaboratory](#)
- [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: visualising data](#)
- [Getting started with BigQuery](#)

Machine learning crash course

These are a few of the notebooks from Google's online machine learning course. See the [full course website](#) for more.

- [Intro to Pandas DataFrame](#)
- [Linear regression with tf.keras using synthetic data](#)

Using accelerated hardware


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

✓ Featured examples


- [NeMo voice swap](#): Use Nvidia NeMo conversational AI toolkit to swap a voice in an audio fragment with a computer-generated one.
- [Retraining an Image Classifier](#): Build a Keras model on top of a pre-trained image classifier to distinguish flowers.
- [Text Classification](#): Classify IMDB film 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.

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

```
df=pd.read_csv("/content/sample_data/Student_Performance (1).csv")
df.head()
```



	Hours Studied	Previous Scores	Extracurricular Activities	Sleep Hours	Sample Question Papers Practiced	Performance Index
0	7	99	Yes	9	1	91
1	4	82	No	4	2	65
2	8	51	Yes	7	2	45




Next steps:

[Generate code with df](#)

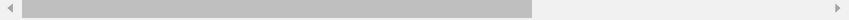
[View recommended plots](#)

[New interactive sheet](#)

```
df.describe()
```



	Hours Studied	Previous Scores	Sleep Hours	Sample Question Papers Practiced	Performance Index
count	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000
mean	4.992900	69.445700	6.530600	4.583300	55.224800
std	2.589309	17.343152	1.695863	2.867348	19.212558
min	1.000000	40.000000	4.000000	0.000000	10.000000
25%	3.000000	54.000000	5.000000	2.000000	40.000000
50%	5.000000	69.000000	7.000000	5.000000	55.000000



```
#plotting box plot for all numericals columns
plt.figure(figsize=(12,10))

plt.subplot(3,3,1)
sns.boxplot(df['Hours Studied'],color='gold',fill=True,orient='h',linecolor='black')
plt.title(' Box Plot for Hours Studied')

plt.subplot(3,3,2)
sns.boxplot(df['Previous Scores'],color='gold',fill=True,orient='h',linecolor='black')
plt.title(' Box Plot for Previous Scores      ')

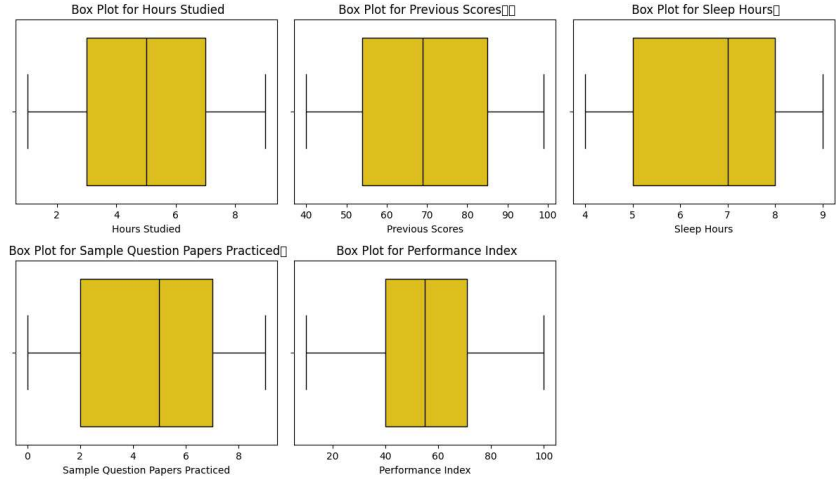
plt.subplot(3,3,3)
sns.boxplot(df['Sleep Hours'],color='gold',fill=True,orient='h',linecolor='black')
plt.title(' Box Plot for Sleep Hours      ')

plt.subplot(3,3,4)
sns.boxplot(df['Sample Question Papers Practiced'],color='gold',fill=True,orient='h',linecolor='black')
plt.title(' Box Plot for Sample Question Papers Practiced      ')

plt.subplot(3,3,5)
sns.boxplot(df['Performance Index'],color='gold',fill=True,orient='h',linecolor='black')
plt.title(' Box Plot for Performance Index')

plt.tight_layout()
```

```
<ipython-input-11-63ca90caa464>:24: UserWarning: Glyph 9 ( ) missing from font
plt.tight_layout()
/usr/local/lib/python3.11/dist-packages/IPython/core/events.py:89: UserWarning: Gl
func(*args, **kwargs)
/usr/local/lib/python3.11/dist-packages/IPython/core/pylabtools.py:151: UserWarnin
fig.canvas.print_figure(bytes_io, **kw)
```



```
#displaying total counts for categorical column
df['Extracurricular Activities'].value_counts()
```

count	
Extracurricular Activities	
No	5052
Yes	4948



```
#encoding categorical column with label encoding
from sklearn import preprocessing
le= preprocessing.LabelEncoder()
df['Extracurricular Activities']=le.fit_transform(df['Extracurricular Activities'])
df.head()
```

	Hours Studied	Previous Scores	Extracurricular Activities	Sleep Hours	Sample Question Papers Practiced	Performance Index
0	7	99	1	9	1	91
1	4	82	0	4	2	65
2	8	51	1	7	2	45

```
#scaling of columns
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
scaler.fit(df)
```

StandardScaler

StandardScaler()

```
#scaled dataframe
cols=df.columns
scaled_df=scaler.transform(df)
new_df=pd.DataFrame(scaled_df,columns=cols)#converting into new dataframe
new_df.head()
```

	Hours Studied	Previous Scores	Extracurricular Activities	Sleep Hours	Sample Question Papers Practiced	Performance Index
0	0.775188	1.704176	1.010455	1.456205	-1.249754	1.862167
1	-0.383481	0.723913	-0.989654	-1.492294	-0.900982	0.508818
2	1.161410	-1.063626	1.010455	0.276805	-0.900982	-0.532220

Next steps:

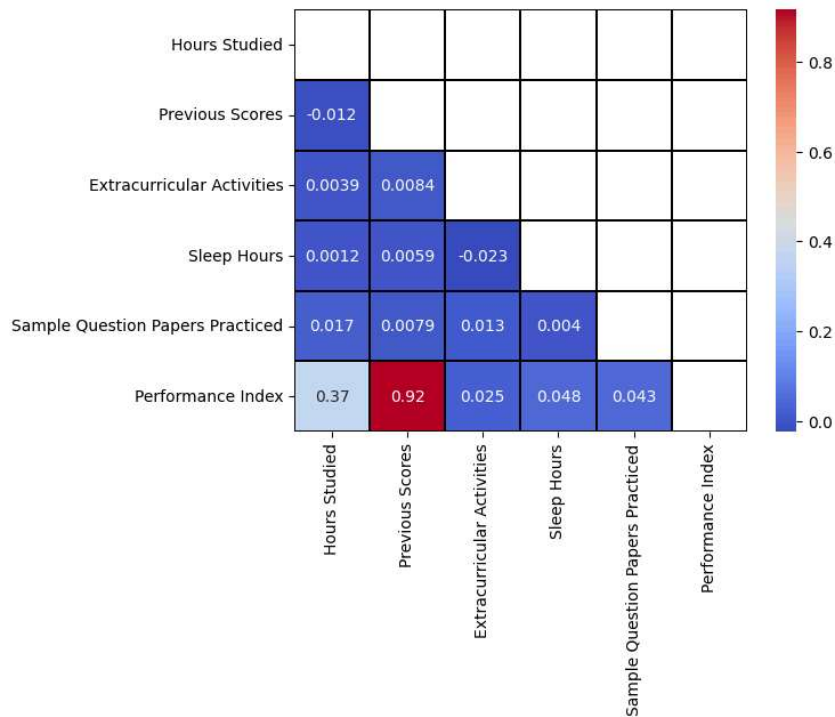
[Generate code with new_df](#)
[View recommended plots](#)
[New interactive sheet](#)

```
#checking for correlation between performance index and other
correlation=new_df.corr()['Performance Index'].sort_values(ascending=True)
correlation
```

	Performance Index
Extracurricular Activities	0.024525
Sample Question Papers Practiced	0.043268
Sleep Hours	0.048106
Hours Studied	0.373730
Previous Scores	0.915189
Performance Index	1.000000

dtype: float64

```
#plotting heatmap for correlation
mask=np.triu(np.ones_like(correlation, dtype=bool))
sns.heatmap(new_df.corr(),linecolor='black',annot=True,mask=mask,cmap='coolwarm',linewidth=0.5)
plt.show()
```



```
#separating inputs and outputs for multiple linear regression
```

```
x = new_df.drop(columns='Performance Index')
```

```
y = new_df ['Performance Index']
```

```
x.head()
```



	Hours Studied	Previous Scores	Extracurricular Activities	Sleep Hours	Sample Question Papers Practiced
0	0.775188	1.704176	1.010455	1.456205	-1.249754
1	-0.383481	0.723913	-0.989654	-1.492294	-0.900982
2	1.161410	-1.063626	1.010455	0.276805	-0.900982
3	0.002742	-1.005963	1.010455	-0.902594	-0.900982

Next steps:

[Generate code with x](#)
[View recommended plots](#)
[New interactive sheet](#)

```
y.head()
```



	Performance Index
0	1.862167
1	0.508818
2	-0.532220
3	-1.000687
4	0.560870

```
#Train Test split
```

```
from sklearn.model_selection import train_test_split
```

```
x_train,x_test, y_train,y_test = train_test_split(x,y,test_size = 0.2,random_state = 4
```

```
x_train.shape
x_test.shape
y_train.shape
```

```
#linear regression model buileding
from sklearn.linear_model import LinearRegression
```

```
model = LinearRegression()
model.fit(x_train,y_train) #TRAIN MODEL
```

```
coefficients = model.coef_
print(coefficients)
```

```
intercept = model.intercept_
print(intercept)
```

```
#prededction on y
y_predicted = model.predict(x_test)
y_predicted
```

```
y_predicted.shape
```

```
↗ [0.38479631 0.91976489 0.01544393 0.04337244 0.02917903]
0.0012344405393052804
(2000,)
```

```
#creating result table data frame for actual y and predicted y
table= pd.DataFrame({'Actual y': y_test, 'Predicted y': y_predicted, 'Residual(Error)'}
table.head()
```

```
↗
```

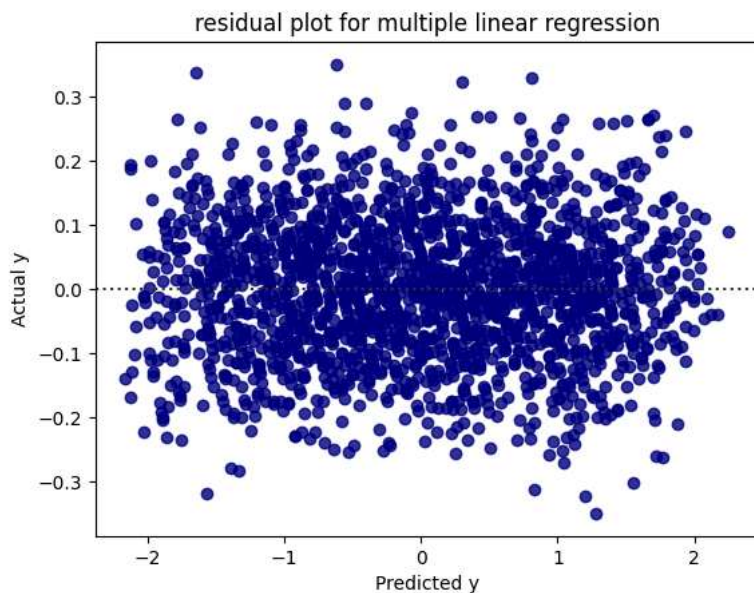
	Actual y	Predicted y	Residual(Error)
1603	-0.271961	-0.214596	-0.057365
8713	1.706011	1.768736	-0.062725
4561	-2.041725	-2.019591	-0.022134
6600	-1.104791	-1.091910	-0.012881
2558	-0.011701	-0.109780	0.098079

Next steps:

[Generate code with table](#)
[View recommended plots](#)
[New interactive sheet](#)

```
#plotting residuals as scatter p[lot
sns.residplot(x=table['Predicted y'],y=table['Actual y'],robust=True,color='navy')
plt.title('residual plot for multiple linear regression')
plt.show()
```

```
↗
```



```
from sklearn import metrics
```



```
from sklearn import metrics
```

```
meanAbsErr=metrics.mean_absolute_error(y_test,y_predicted)  
meanSqErr=metrics.mean_squared_error(y_test,y_predicted)  
RMSErr=np.sqrt(meanSqErr)
```

```
print('Mean absolute error in regrssion',meanAbsErr)
```

```
print('Mean square error in regrssion',meanSqErr)
```

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

Disk

78.68 GB available