# Welcome to Colab!

## Access Popular LLMs via Google-Colab-AI Without an API Key

Users with Colab's paid plans have free access to most popular LLMs via google-colab-ai Python library. For more details, refer to the getting started with google colab ai.

```
from google.colab import ai
response = ai.generate_text("What is the capital of France?")
print(response)
```

## 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**

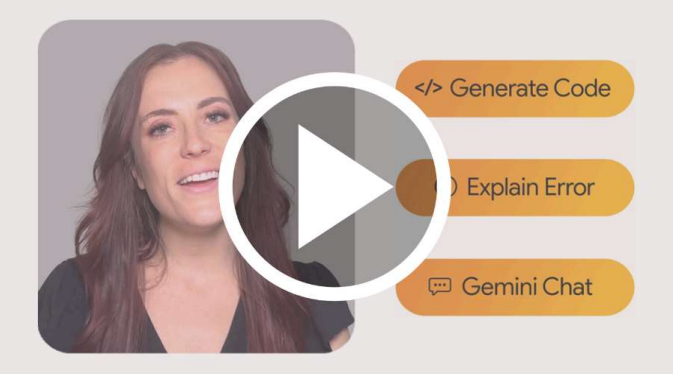
- Play with Gemini multimodal outputs, mixing text and images in an iterative way.
- Discover the multimodal Live API (demo here).
- Learn how to analyze images and detect items in your pictures using Gemini (bonus, there's a 3D version as well!).
- Unlock the power of Gemini thinking model, capable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

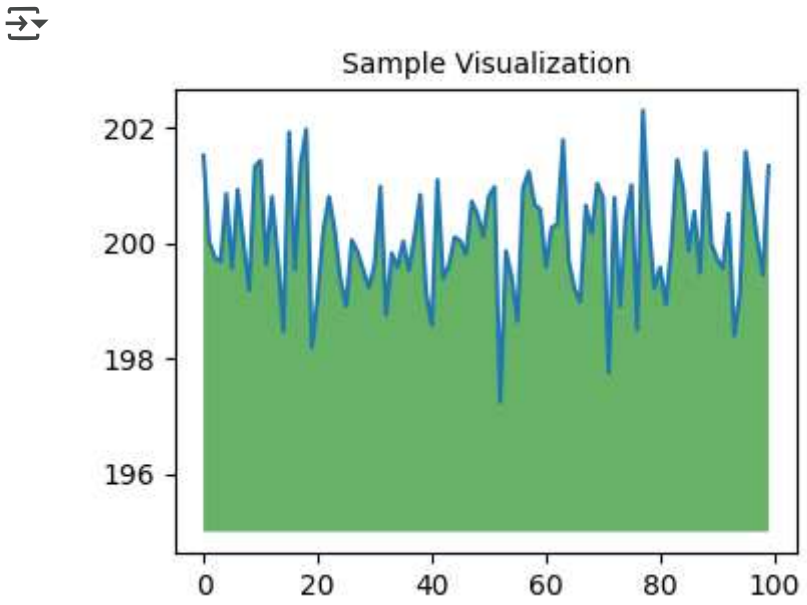
```
import numpy as np
import IPython.display as display
from matplotlib import pyplot as plt
import io
import base64

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"{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.

```
import pandas as pd
df = pd.read_csv('/content/StudentsPerformance.csv')
df.head()
```

| | gender | race/ethnicity | parental level of education | lunch | test preparation course | math score | reading score | writing score |
|---|---|---|---|---|---|---|---|---|
| **0** | female | group B | bachelor's degree | standard | none | 72 | 72 | 74 |
| **1** | female | group C | some college | standard | completed | 69 | 90 | 88 |
| **2** | female | group B | master's degree | standard | none | 90 | 95 | 93 |

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
print(df.shape)
df.info()
df.describe()
```

```
(1000, 8)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 8 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   gender                       1000 non-null   object
 1   race/ethnicity               1000 non-null   object
 2   parental level of education  1000 non-null   object
 3   lunch                        1000 non-null   object
 4   test preparation course      1000 non-null   object
 5   math score                   1000 non-null   int64
 6   reading score                1000 non-null   int64
 7   writing score                1000 non-null   int64
dtypes: int64(3), object(5)
memory usage: 62.6+ KB
```

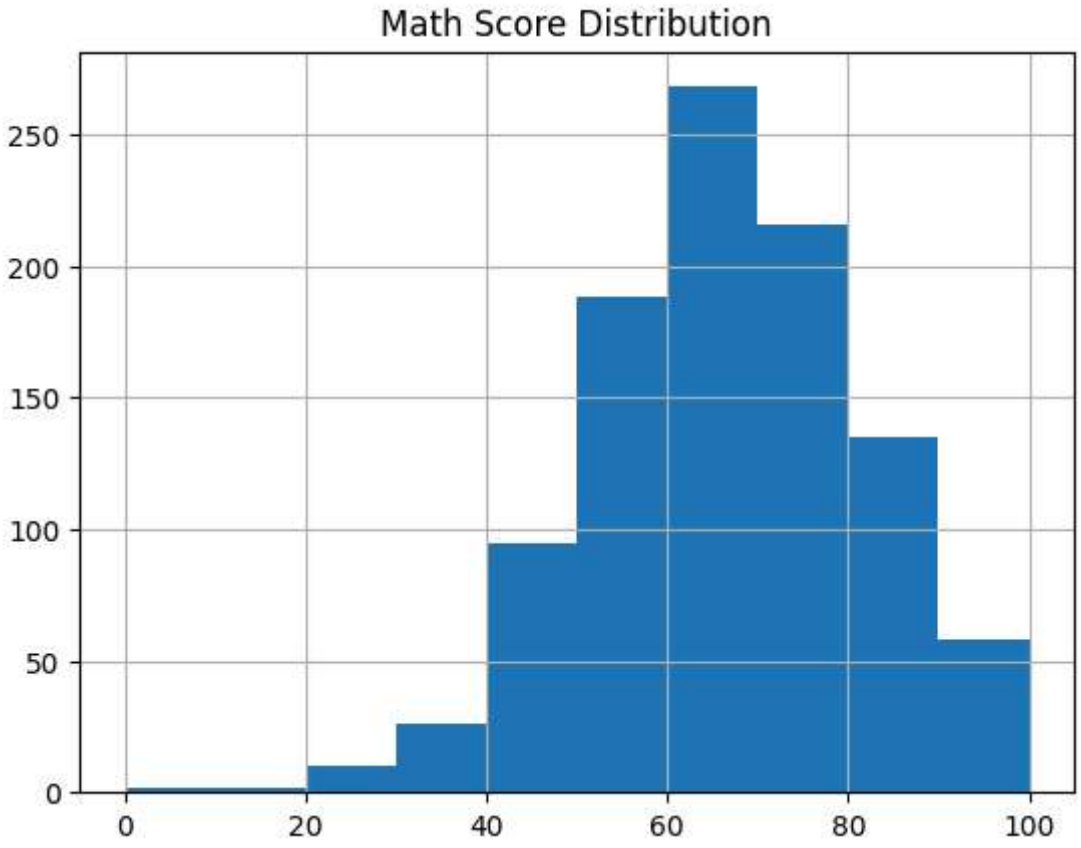
| | math score | reading score | writing score |
|---|---|---|---|
| **count** | 1000.00000 | 1000.000000 | 1000.000000 |
| **mean** | 66.08900 | 69.169000 | 68.054000 |
| **std** | 15.16308 | 14.600192 | 15.195657 |
| **min** | 0.00000 | 17.000000 | 10.000000 |
| **25%** | 57.00000 | 59.000000 | 57.750000 |
| **50%** | 66.00000 | 70.000000 | 69.000000 |
| **75%** | 77.00000 | 79.000000 | 79.000000 |
| **max** | 100.00000 | 100.000000 | 100.000000 |

```
df.columns = [col.strip().lower().replace(' ', '_') for col in df.columns]
df.head()
```

| | gender | race/ethnicity | parental_level_of_education | lunch | test_preparation_co |
|---|---|---|---|---|---|
| **0** | female | group B | bachelor's degree | standard | |
| **1** | female | group C | some college | standard | comp |
| **2** | female | group B | master's degree | standard | |
| **3** | male | group A | associate's degree | free/reduced | |
| **4** | male | group C | some college | standard | |

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
import matplotlib.pyplot as plt
df['math_score'].hist(bins=10)
plt.title('Math Score Distribution')
plt.show()
```

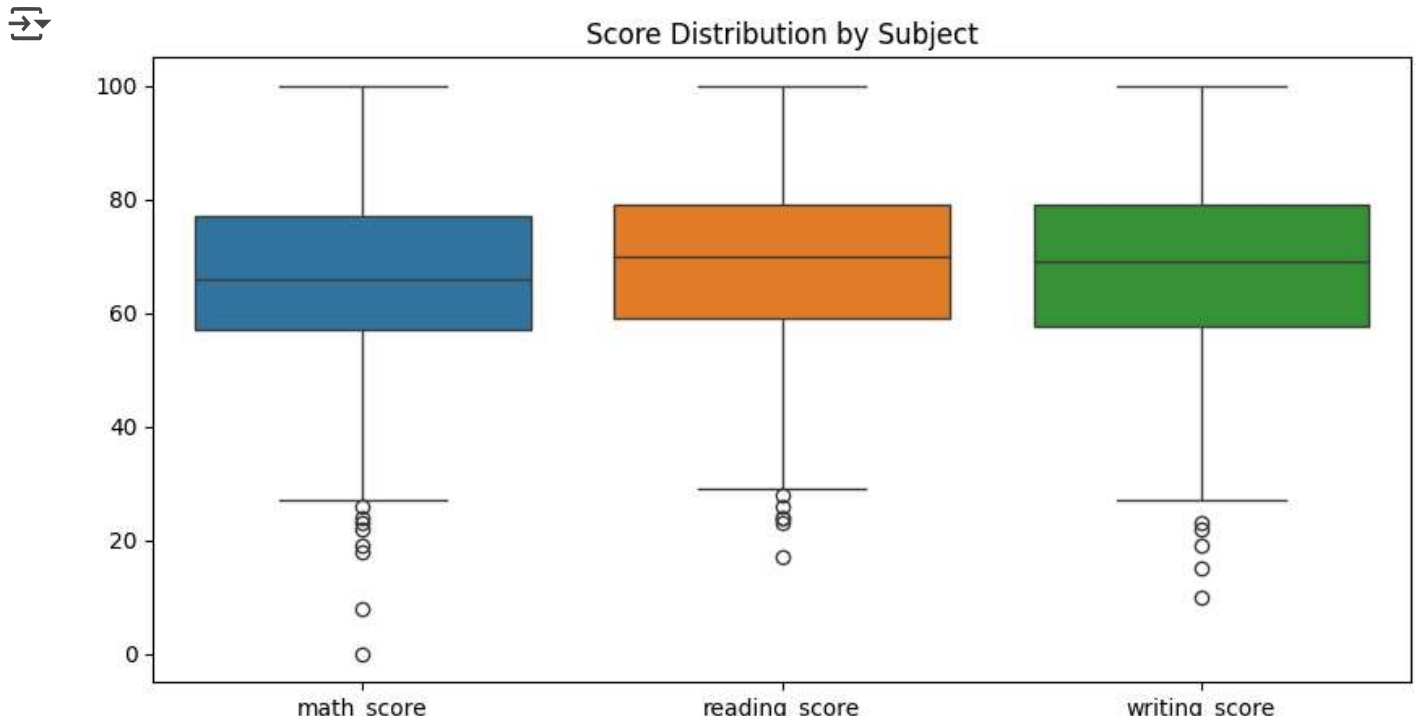



```
df['total_score'] = df['math_score'] + df['reading_score'] + df['writing_score']
top_student = df[df['total_score'] == df['total_score'].max()]
high_achievers = df[(df['math_score'] > 90) & (df['reading_score'] > 90) & (df['writing_scor
```

```
gender_scores = df.groupby('gender')[['math_score', 'reading_score', 'writing_score']].mean(
print("Average Scores by Gender:")
print(gender_scores)
```

```
Average Scores by Gender:
        math_score  reading_score  writing_score
gender
female   63.633205      72.608108      72.467181
male     68.728216      65.473029      63.311203
```

```
import seaborn as sns
plt.figure(figsize=(10, 5))
sns.boxplot(data=df[['math_score', 'reading_score', 'writing_score']])
plt.title("Score Distribution by Subject")
plt.show()
```




```
prep_scores = df.groupby('test_preparation_course')[['math_score', 'reading_score', 'writing_
print("\nAverage Scores based on Test Preparation Course:")
print(prep_scores)
```

```
Average Scores based on Test Preparation Course:
                         math_score  reading_score  writing_score
```