- 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.

```
!pip install -q -U google-generativeai
```

```
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 155.1/155.1 kB 2.7 MB/s eta 0:00:00
```

```
import os
os.environ['GEMINI_API_KEY']='AIzaSyDlXfU49JVQaRY_gAaKQG2eL73QgmEUxgM'
```

```
import pathlib
import textwrap

import google.generativeai as genai

from IPython.display import display
from IPython.display import Markdown

def to_markdown(text):
  text = text.replace('.', ' *')
  return Markdown(textwrap.indent(text, '>', predicate=lambda _ : True))
```

```
import pathlib
import textwrap
import google.generativeai as genai
```

```
from IPython.display import display
from IPython.display import Markdown
```

```
def to_markdown(text):
  text = text.replace('.', ' *')
  return Markdown(textwrap.indent(text, '>', predicate=lambda _ : True))
```

```
for m in genai.list_models():
  if 'generatContent' in m.supported_generation_methods:
       print(m.name)
```

```
for m in genai.list_models():
  if 'generatContent' in m.supported_generation_methods:
    print(m.name)
```

```
#for m in genai.list_models()
for m in genai.list_models():
  print(m)
```

```
---------------------------------------------------------------------------
DefaultCredentialsError                   Traceback (most recent call last)
/tmp/ipython-input-600428360.py in <cell line: 0>()
      1 #for m in genai.list_models()
----> 2 for m in genai.list_models():
      3   print(m)

 ──────────────────────────────────────────────────── 8 frames ─────────────────────────────────────────────────────
/usr/local/lib/python3.12/dist-packages/google/auth/_default.py in default(scopes, request, quota_project_id,
default_scopes)
    737             return credentials, effective_project_id
    738
--> 739     raise exceptions.DefaultCredentialsError(_CLOUD_SDK_MISSING_CREDENTIALS)

DefaultCredentialsError:
  No API_KEY or ADC found. Please either:
    - Set the `GOOGLE_API_KEY` environment variable.
    - Manually pass the key with `genai.configure(api_key=my_api_key)`.
    - Or set up Application Default Credentials, see https://ai.google.dev/gemini-api/docs/oauth for more information.
```

Next steps: Explain error

```
for m in genai.list_models():
  print(m)
```

```
      version='3.1',
      display_name='Veo 3.1',
      description='Veo 3.1',
      input_token_limit=480,
      output_token_limit=8192,
      supported_generation_methods=['predictLongRunning'],
      temperature=None,
      max_temperature=None,
      top_p=None,
      top_k=None)
Model(name='models/veo-3.1-fast-generate-preview',
      base_model_id='',
      version='3.1',
      display_name='Veo 3.1 fast',
      description='Veo 3.1 fast',
      input_token_limit=480,
      output_token_limit=8192,
      supported_generation_methods=['predictLongRunning'],
      temperature=None,
      max_temperature=None,
      top_p=None,
      top_k=None)
Model(name='models/gemini-2.5-flash-native-audio-latest',
      base_model_id='',
      version='Gemini 2.5 Flash Native Audio Latest',
      display_name='Gemini 2.5 Flash Native Audio Latest',
      description='Latest release of Gemini 2.5 Flash Native Audio',
      input_token_limit=131072,
      output_token_limit=8192,
      supported_generation_methods=['countTokens', 'bidiGenerateContent'],
      temperature=1.0,
      max_temperature=2.0,
      top_p=0.95,
      top_k=64)
Model(name='models/gemini-2.5-flash-native-audio-preview-09-2025',
      base_model_id='',
      version='gemini-2.5-flash-preview-native-audio-dialog-2025-05-19',
      display_name='Gemini 2.5 Flash Native Audio Preview 09-2025',
      description='Gemini 2.5 Flash Native Audio Preview 09-2025',
      input_token_limit=131072,
      output_token_limit=8192,
      supported_generation_methods=['countTokens', 'bidiGenerateContent'],
      temperature=1.0,
      max_temperature=2.0,
      top_p=0.95,
      top_k=64)
Model(name='models/gemini-2.5-flash-native-audio-preview-12-2025',
      base_model_id='',
      version='12-2025',
      display_name='Gemini 2.5 Flash Native Audio Preview 12-2025',
      description='Gemini 2.5 Flash Native Audio Preview 12-2025',
      input_token_limit=131072,
      output_token_limit=8192,
      supported_generation_methods=['countTokens', 'bidiGenerateContent'],
      temperature=1.0,
      max_temperature=2.0,
      top_p=0.95,
      top_k=64)
```

```
model = genai.GenerativeModel('models/gemini-3-flash-preview')
```

```
response = model.generate_content('create table for ai vs gen ai vs agentic ai vs mcp')
```

```
print(response.text)
```

```
To understand the differences between these four terms, it is helpful to look at them as an evolution: **AI** is the field,

### Comparison Table: AI vs. Gen AI vs. Agentic AI vs. MCP

| Feature | Artificial Intelligence (AI) | Generative AI (Gen AI) | Agentic AI (AI Agents) | Model Context Protocol (MCP) |
| :--- | :--- | :--- | :--- | :--- |
| **Core Definition** | The broad field of creating systems that can perform tasks requiring human intelligence. | A subset
| **Primary Goal** | To analyze data and solve problems (prediction/classification). | To generate a response or creative ou
| **Autonomy Level** | Low: Typically follows rigid algorithms or specific models. | Medium: Responds to a human prompt; fol
| **Interaction Style** | Data Input → Result. | Prompt → Generated Content. | Goal → Sequence of Actions → Result. | Model
| **Key Capability** | Logic, pattern recognition, and decision-making. | Synthesis, creativity, and summarization. | Reasor
| **Real-World Example** | A Netflix recommendation or a fraud detection system. | ChatGPT writing a poem or DALL-E creating
| **Analogy** | The concept of a motorized vehicle. | The engine that produces power. | The self-driving car (using the engi

---

### Deep Dive: How they work together

#### 1. Artificial Intelligence (AI) - The Umbrella
This is the "grandfather" term. It includes everything from simple calculators to complex neural networks. Traditional AI (F

#### 2. Generative AI (Gen AI) - The Engine
Gen AI changed the game by moving from *predicting* to *creating*. Using Large Language Models (LLMs) like GPT-4 or Claude,

#### 3. Agentic AI - The Worker
While Gen AI is a "chatbot," Agentic AI is a "coworker." Instead of you saying "Write an email," you say "Organize a meeting
1.  Check your calendar.
2.  Find an open slot.
3.  Draft the invite.
4.  Send it.
It uses **reasoning loops** (like Chain-of-Thought) to decide its own next steps.

#### 4. Model Context Protocol (MCP) - The Connector
Until recently, every time a developer wanted to connect an AI to a tool (like GitHub, Slack, or a SQL database), they had t
*   **MCP (introduced by Anthropic)** is a universal standard.
*   If a database supports MCP, *any* AI model (Agentic or otherwise) can instantly understand how to interact with it.
*   It solves the "integration" problem, making Agentic AI much easier to build.
```

```
response = model.generate_content('create list of holidays in January 2026  in a table')
```

```
print(response.text)
```

```
Here is a list of major international and national holidays for January 2026.

| Date | Day | Holiday Name | Location/Context |
| :--- | :--- | :--- | :--- |
| **January 1** | Thursday | **New Year's Day** | Global |
| **January 5** | Monday | Twelfth Night | Christian Tradition |
| **January 6** | Tuesday | Epiphany / Three Kings Day | International (Various) |
| **January 7** | Wednesday | Orthodox Christmas Day | International (Orthodox Church) |
| **January 14** | Wednesday | Orthodox New Year | International (Julian Calendar) |
| **January 14** | Wednesday | Makar Sankranti / Pongal | India (Hinduism) |
| **January 19** | Monday | **Martin Luther King Jr. Day** | United States |
| **January 19** | Monday | Timkat | Ethiopia |
| **January 26** | Monday | **Republic Day** | India |
| **January 26** | Monday | **Australia Day** | Australia |
| **January 29** | Thursday | **Lunar New Year (Year of the Horse)** | East Asia / Global |
| **January 30** | Friday | Second Day of Lunar New Year | East Asia / Global |

### Key Highlights for January 2026:

*   **Martin Luther King Jr. Day (US):** This falls on the third Monday of January every year. In 2026, it is observed on Ja
*   **Republic Day (India) & Australia Day:** Both of these national holidays coincide on Monday, January 26, 2026, creating
*   **Lunar New Year:** 2026 is the **Year of the Horse**. The Lunar New Year begins on January 29, which is relatively late
*   **Orthodox Holidays:** For those following the Julian calendar, Christmas falls on January 7 and the New Year on January
```

```
Start coding or generate with AI.
```