SDSU | HealthLINK Center

# Data Science Training
# Fall 2025 Workshop: Introduction to Machine Learning

## Machine Learning using Google Cloud AutoML and MATLAB

*SDSU HealthLINK Center for Transdisciplinary Health Disparities Research*
*Funder: National Institute on Minority Health and Health Disparities (U54MD012397 & S21MD010690)*
*MPIs: Guadalupe X. Ayala, PhD, MPH, MA and Kristen J. Wells, PhD, MPH*

# Session 1: Monday, September 8, 2025
# Introduction to ML & Google Cloud AutoML

Machine Learning Foundations

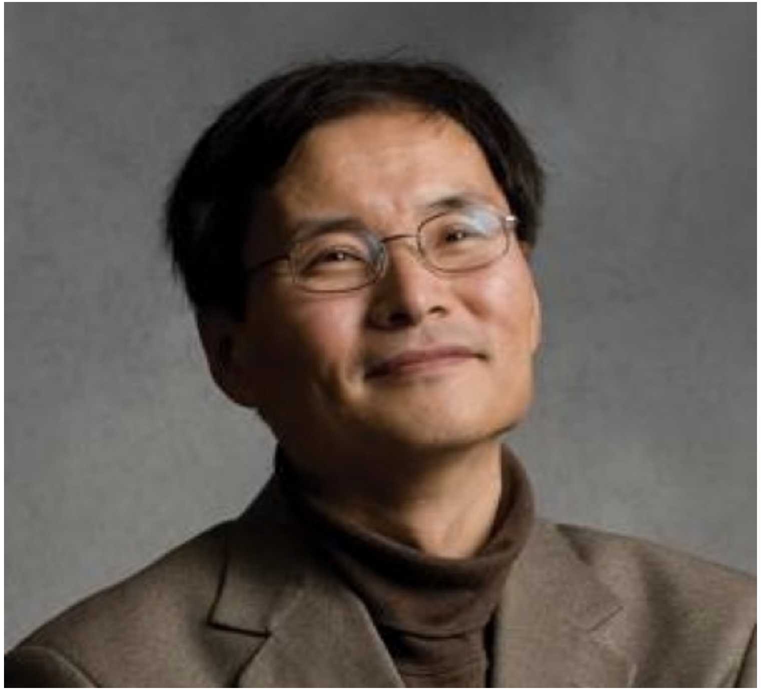
Google Cloud AutoML (Vertex AI)

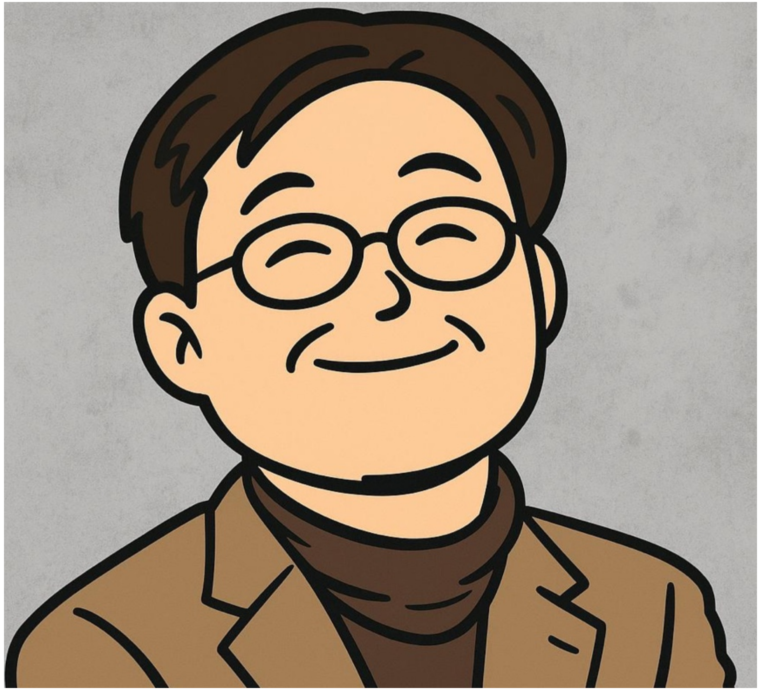
Hands-On Exercise 1: Penguin Species Classification

# Machine Learning Foundations

- What is Machine Learning?
- ML Types: Supervised, Unsupervised, Reinforcement
- Real-world ML applications
- Organizing datasets

# What is Machine Learning (ML)?





# Human can observe and learn.

Human can learn from past experience and make decision of its own.

What about a Machine ?

We want a machine to act like a human.

# What is machine learning?

- We need to provide **experience** to the machine to take decision of its own.
- Then, execute the **generated program** on the NEW DATA.

Dataset

+

Then, using the programs,

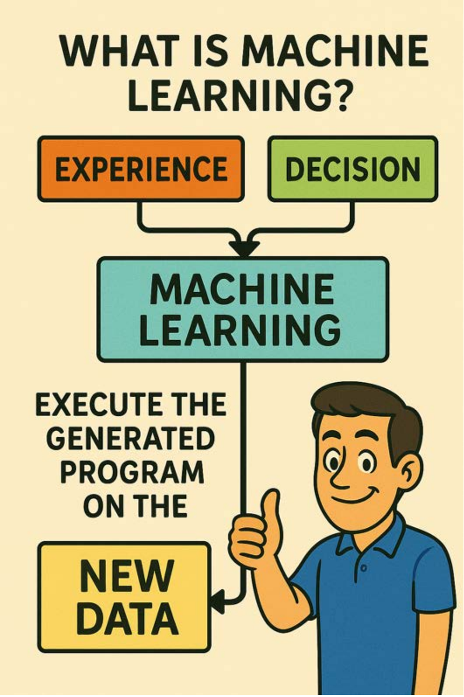
Identify required rules

Extract required patterns

Identify relations

# F25 SDSU HealthLINK Center Data Science Training Program

- Two-day data science training program to train SDSU and community partner researchers to perform data science methodologies (e.g., machine learning using the GoogleCloud AutoML and MatLab) .
- This two-class (6-hour) topic covers elementary machine learning techniques in GoogleCloud AutoML and MATLAB® utilizing Machine Learning Toolbox™. The training course explains how to use unsupervised learning to uncover features in large data sets and supervised learning to develop prediction models. Participants with some programming knowledge will benefit from simple but professional examples and activities.



WHAT IS MACHINE LEARNING?

EXPERIENCE

DECISION

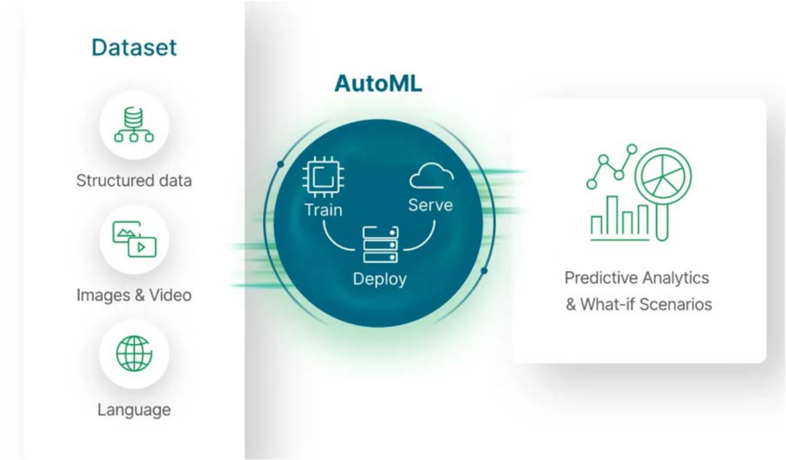
MACHINE LEARNING

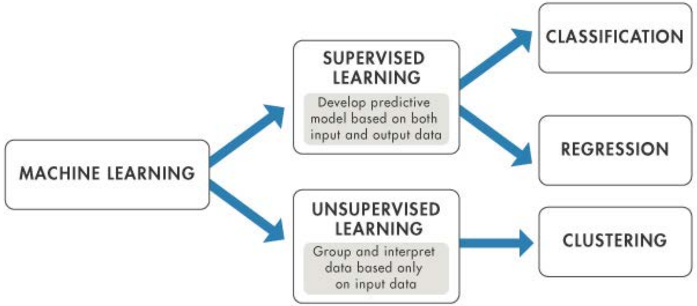
EXECUTE THE GENERATED PROGRAM ON THE

NEW DATA

# Topics include:

- Introduction to Machine Learning
- Machine Learning paradigms
  - Supervised
  - Unsupervised
  - Reinforcement
- Organizing and preprocessing data
  - Generating training and checking data
  - Building Machine Learning Models
- Machine Learning practice
  - Creating classification and (regression) models: AUTOML & MATLAB
    - Interpreting and evaluating models
  - Clustering data : MATLAB
    - Interpreting and evaluating models




MACHINE LEARNING
SUPERVISED LEARNING
Develop predictive model based on both input and output data
UNSUPERVISED LEARNING
Group and interpret data based only on input data
CLASSIFICATION
REGRESSION
CLUSTERING

# ML(machine learning) using the GoogleCloud AutoML and MatLab

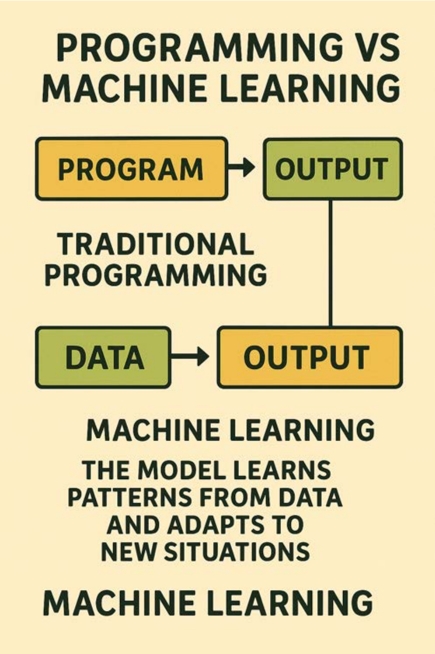
## AutoML

- Starting with AutoML (Automated Machine Learning)
- Start Google AutoML API (Vertex AI)
- Create a Cloud Storage Bucket
- Google Cloud AutoML Supervised Classification (Tabular)

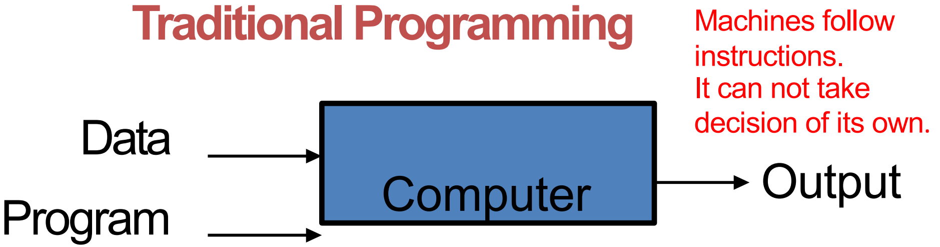
## MatLab ML

- Machine Learning analysis using Matlab ML APP
- Creating classification and (regression) models
- Clustering data

# What is machine learning?

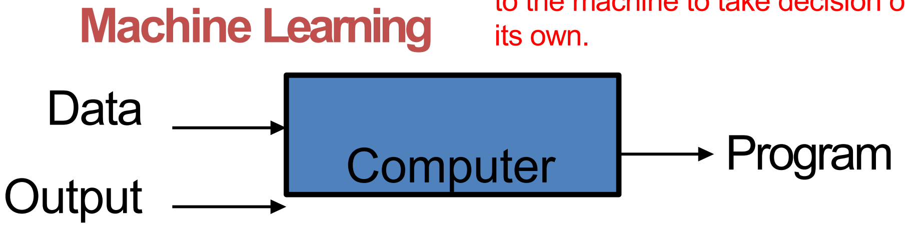

PROGRAMMING VS MACHINE LEARNING
PROGRAM
OUTPUT
TRADITIONAL PROGRAMMING
DATA
OUTPUT
MACHINE LEARNING
THE MODEL LEARNS PATTERNS FROM DATA AND ADAPTS TO NEW SITUATIONS
MACHINE LEARNING

In traditional programming, the programmer defines the rules, while in machine learning, the model learns patterns from data and adapts to new situations.



Machines follow instructions.
It can not take decision of its own.

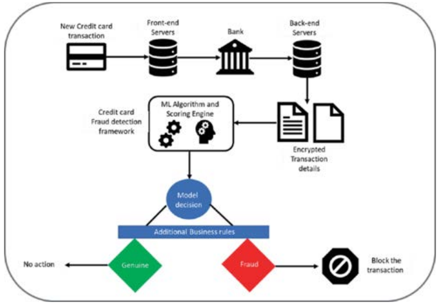
Just like, what we did to human, we need to provide experience to the machine to take decision of its own.

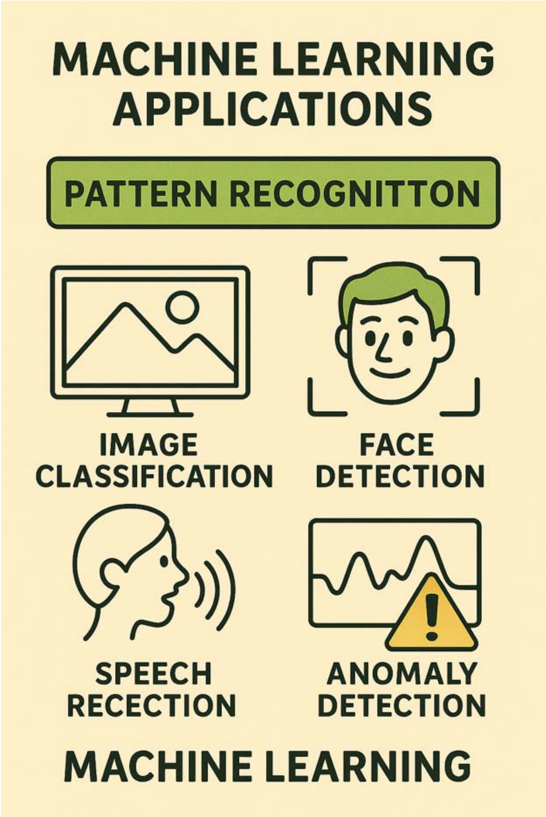


# ML applications include:

- Some more examples of tasks that are best solved by using a machine learning algorithm
  - Recognizing patterns:
    - Facial identities or facial expressions
    - Handwritten or spoken words
    - Medical images
  - Recognizing anomalies:
    - Unusual credit card transactions
    - Unusual patterns of sensor readings in a nuclear power plant

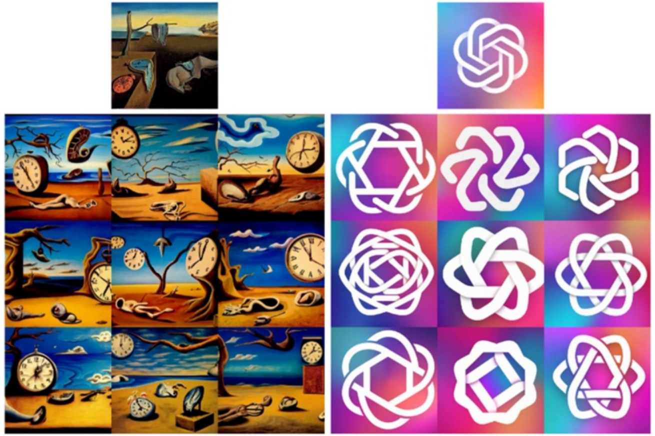
fraud.com
Types of
Credit Card Fraud
Card Theft
Account Takeover
Cloned Cards
Card-not-present Theft
Other Types of Credit Card Fraud


New Credit card transaction
Front-end Servers
Bank
Back-end Servers
Credit card Fraud detection framework
ML Algorithm and Scoring Engine
Encrypted Transaction details
Model decision
Additional Business rules
No action
Genuine
Fraud
Block the transaction



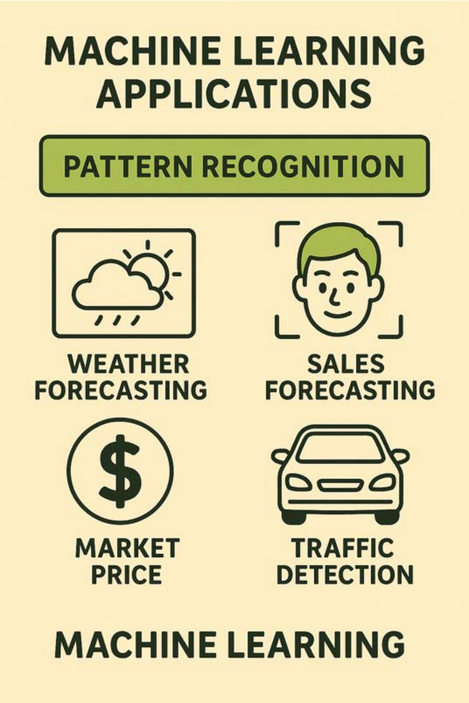
# ML applications include:

- Generating patterns:
  - Generating images or motion sequences
- Prediction:
  - Future stock prices or currency exchange rates

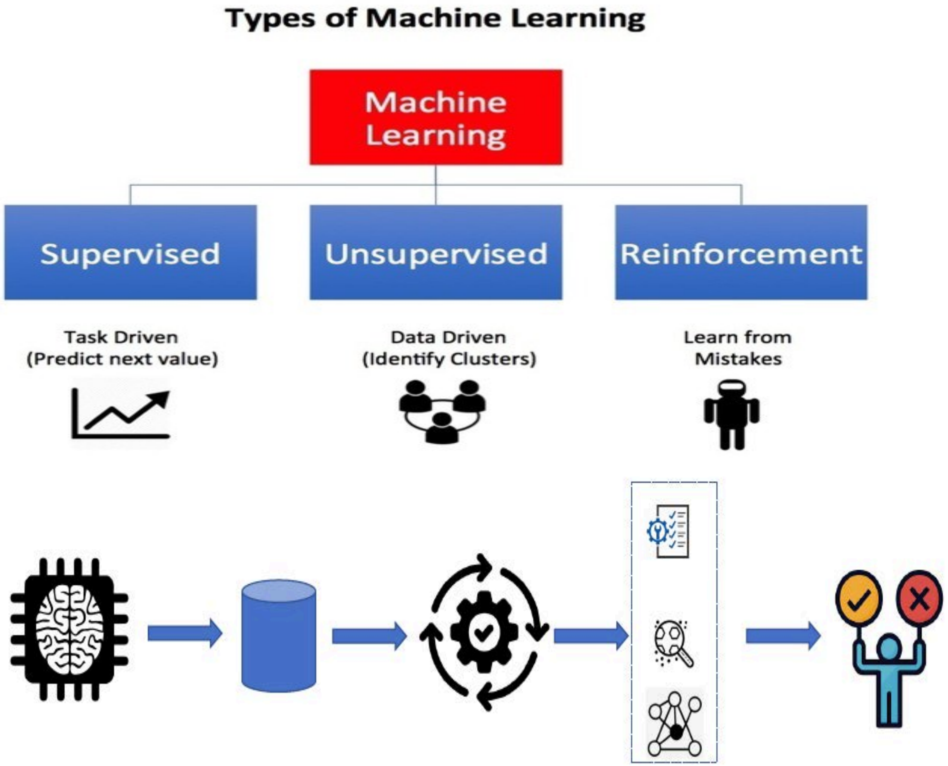


Variations of an input image by encoding with CLIP and then decoding with a diffusion model. The variations preserve both semantic information like presence of a clock in the painting and the overlapping strokes in the logo, as well as stylistic elements like the surrealism in the painting and the color gradients in the logo, while varying the non-essential details.

https://www.v7labs.com/blog/ai-generated-art



# Machine Learning Types

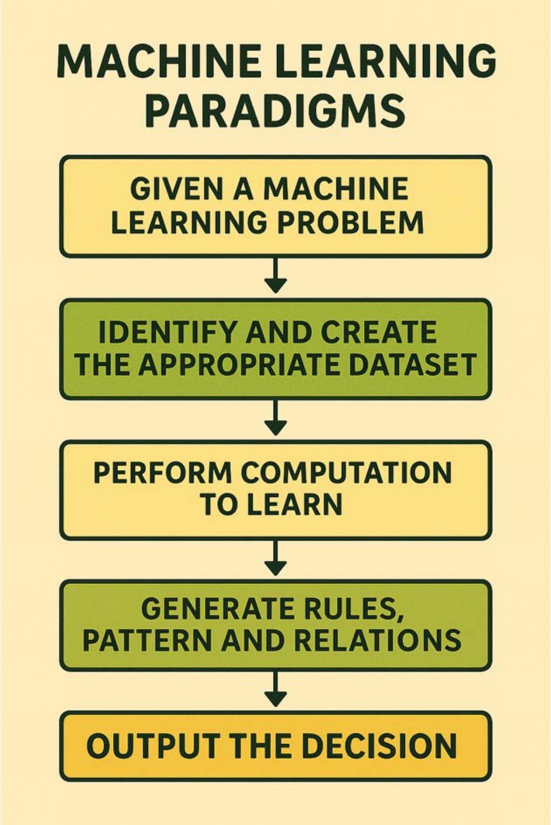


Depending on the nature of the problem, machine learning tasks  can be broadly divided in

- **Supervised (inductive) learning**
  – Given: training data + desired outputs (labels)
- **Unsupervised learning**
  – Given: training data (without desired outputs)
- **Semi-supervised learning**
  – Given: training data + a few desired outputs
- **Reinforcement learning**
  – Rewards from sequence of actions

## Machine Learning Paradigm Flowchart

1. **Given a Machine Learning Problem**
   - Identify the specific problem or task to be solved.
2. **Identify and Create the Appropriate Dataset**
   - Collect and preprocess data relevant to the problem.
3. **Perform Computation to Learn**
   - Utilize algorithms to process the dataset and begin learning.
4. **Generate Rules, Patterns, and Relations**
   - Extract meaningful insights, patterns, and relationships from the learned data.
5. **Output the Decision**
   - Use the generated rules and patterns to make informed decisions or predictions.



MACHINE LEARNING PARADIGMS

GIVEN A MACHINE LEARNING PROBLEM

IDENTIFY AND CREATE THE APPROPRIATE DATASET

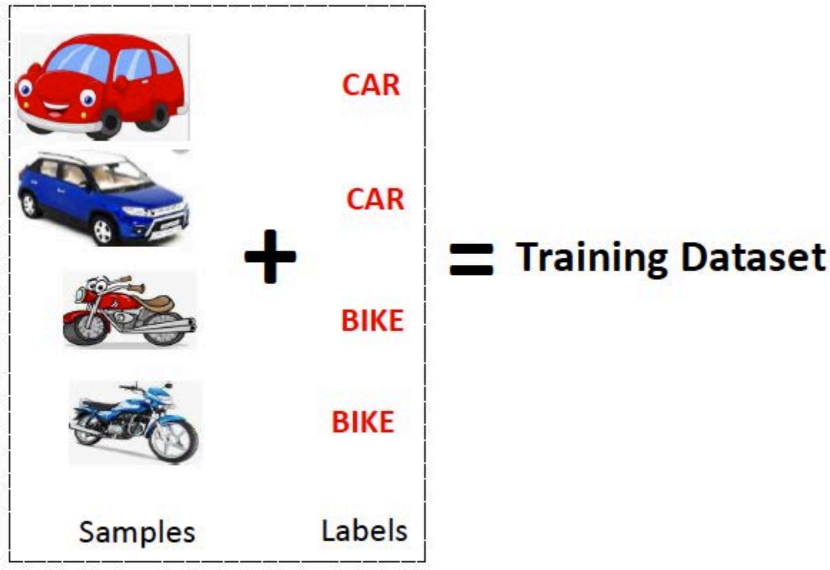
PERFORM COMPUTATION TO LEARN

GENERATE RULES, PATTERN AND RELATIONS

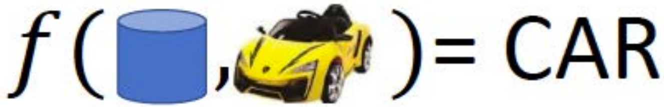
OUTPUT THE DECISION

# Supervised Learning

In supervised learning, we need some thing called a Labelled Training Dataset.

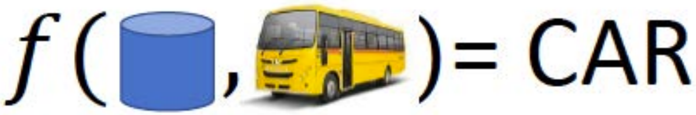


Given a labelled dataset, the task is to devise a function which takes the dataset, and a new sample, and produces an output value.



it will produce output only from the labels defined in the dataset. For example, even if we input  a bus, it will produce either CAR or BIKE.

Classification

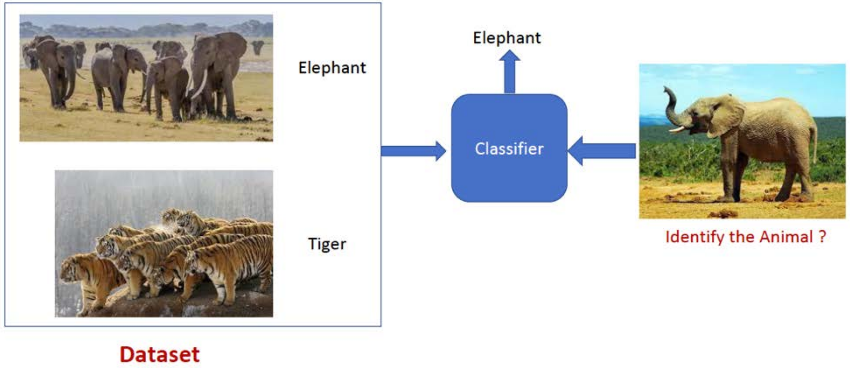


# Supervised Learning

In machine learning, a classifier is an algorithm that categorizes data into predefined classes or categories.  Think of it like a smart sorting machine.  You feed it data (which could be anything from images to text to sensor readings), and it assigns each piece of data to a specific label or category.

If the possible output values of the function are **continuous real values**, then it is called **Regression**.

## Classifier



## Regression

Dataset

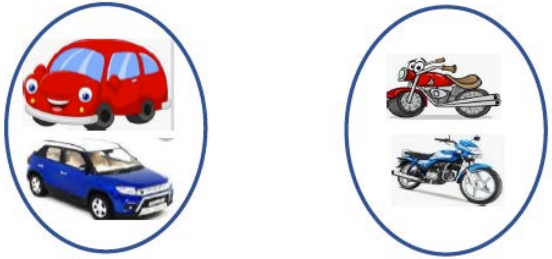
Regression
f ( , )= 20500.50

# Unsupervised Learning

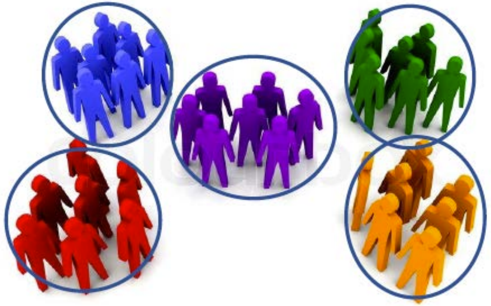
- In the unsupervised learning, we do not need to know the labels or Ground truth values.
- The task is to identify the patterns like group the similar objects together.

Dataset

Dataset



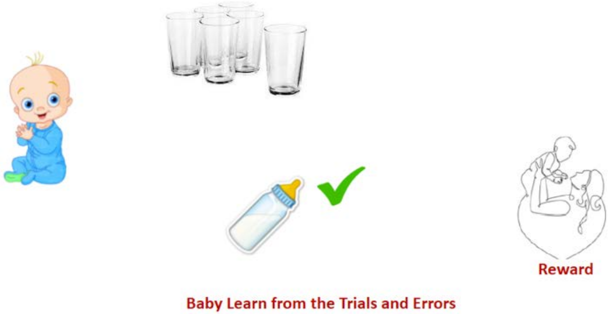
Clustering

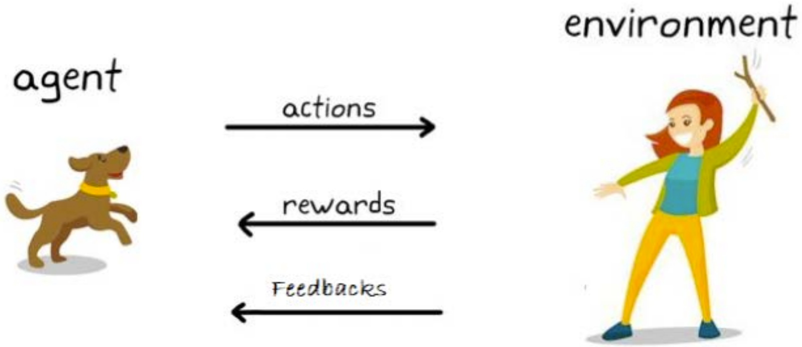


# Reinforcement Learning

- Reinforcement learning is a type of machine learning where an agent learns to interact with an environment by taking actions and receiving rewards or penalties.

- It's like training a pet – you reward good behavior and discourage bad behavior until the pet learns to perform the desired actions.





- Agent: The learner that interacts with the environment and makes decisions.
- Environment: The world in which the agent operates. It can be a simulated environment (like a game) or a real-world environment (like a robot navigating a room).
- Action: What the agent can do in the environment.
- Reward: A signal from the environment that tells the agent whether its action was good or bad. Positive rewards encourage the agent to repeat the action, while negative rewards (penalties) discourage it.

# Classification: Organizing and preprocessing data

Represent the sample

Identify the features which can represent the objects

$$F = \{f_1 f_2 f_3 \dots f_k\}$$

Feature set={ #Wheel Height Weight Color }

- Identify the features
- Represent the vehicles by the features
- Remove non informative features
- Build the classification model from the data
- Perform the classification task

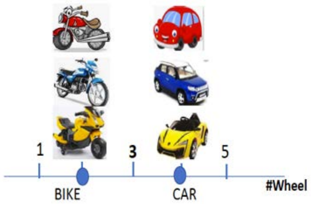
| #Wheel | Height | Weight | Color |
|---|---|---|---|
| 4 | 6 | 500 | Red |
| 4 | 5.5 | 600 | Blue |
| 4 | 5 | 550 | Yellow |
| 2 | 3 | 200 | Red |
| 2 | 3.5 | 150 | blue |
| 2 | 4 | 250 | Yellow |

# Decision Tree (Rule Based Approach)

- Build the classification model from the data

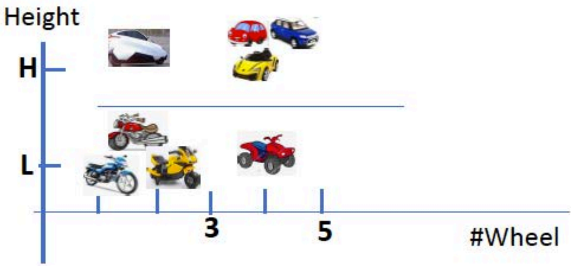
- Perform the classification task

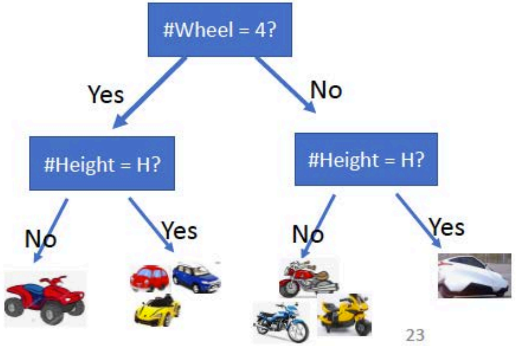
| #Wheel | Class Label |
|---|---|
| 4 | CAR |
| 4 | CAR |
| 4 | CAR |
| 2 | BIKE |
| 2 | BIKE |
| 2 | BIKE |



2 → **Classifier**
If #Wheel < 3, then it is BIKE → BIKE

| #Wheel | Height | Class Label |
|---|---|---|
| 4 | H | CAR |
| 4 | H | CAR |
| 4 | H | CAR |
| 2 | L | BIKE |
| 2 | L | BIKE |
| 2 | L | BIKE |
| 4 | L | BIKE |
| 2 | H | CAR |





# Decision Tree (Rule Based Approach) Example

| outlook | temperature | humidity | windy | play |
|---|---|---|---|---|
| sunny | hot | high | false | no |
| sunny | hot | high | true | no |
| overcast | hot | high | false | yes |
| rainy | mild | high | false | yes |
| rainy | cool | normal | false | yes |
| rainy | cool | normal | true | no |
| overcast | cool | normal | true | yes |
| sunny | mild | high | false | no |
| sunny | cool | normal | false | yes |
| rainy | mild | normal | false | yes |
| sunny | mild | normal | true | yes |
| overcast | mild | high | true | yes |
| overcast | hot | normal | false | yes |
| rainy | mild | high | true | no |

**Given :** <sunny, cool, high, true>

**Predict, if there will be a match?**

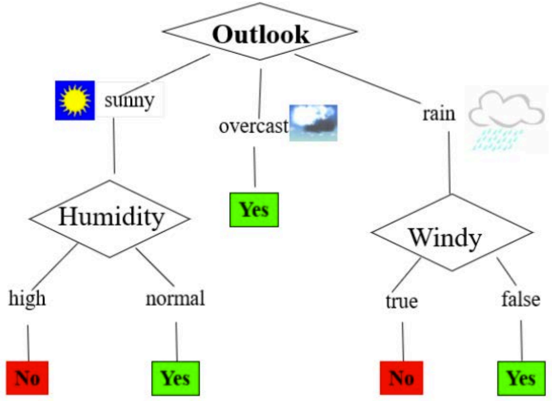
Assume that I have a set of rules:

- If *((lookout=sunny)* **and** *( humudity=high)* **and** *(windy=false)) then (yes) else (no)*
- If *(lookout=overcast) then (yes)*
- If *((lookout=sunny)* **and** *( humudity=high)) then (yes) else (no)*
- *so on…..*

Rule 1: If *((lookout=sunny)* and *( humudity=high)) then (yes) else (no)*

Rule 2: If *(lookout=overcast) then (yes)*

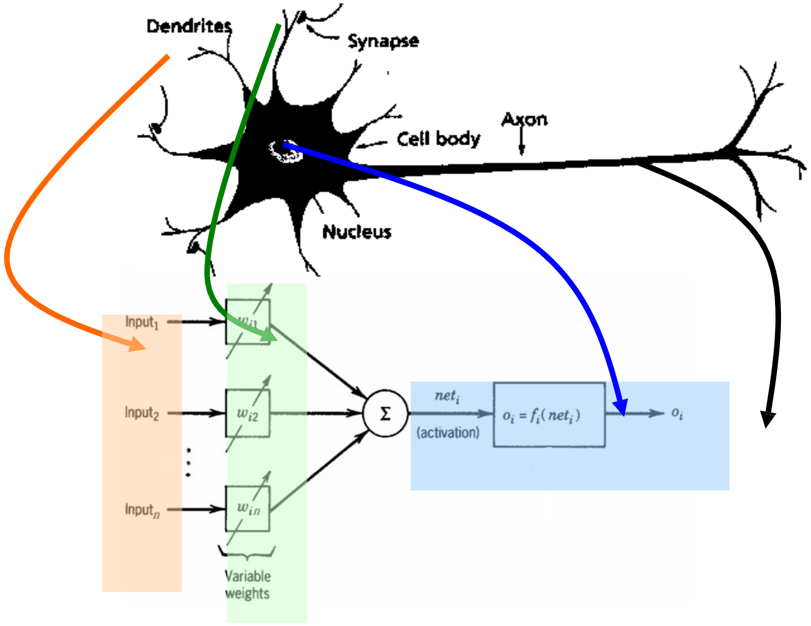
Rule 3: If *((lookout=rain)* and *( windy=true)) then (no) else (yes)*



# Classification: Artificial neural networks (ANNs)

- Often just called neural networks, are a core component of machine learning, particularly in the subfield of deep learning.
- They are computational models inspired by the structure and function of the human brain.

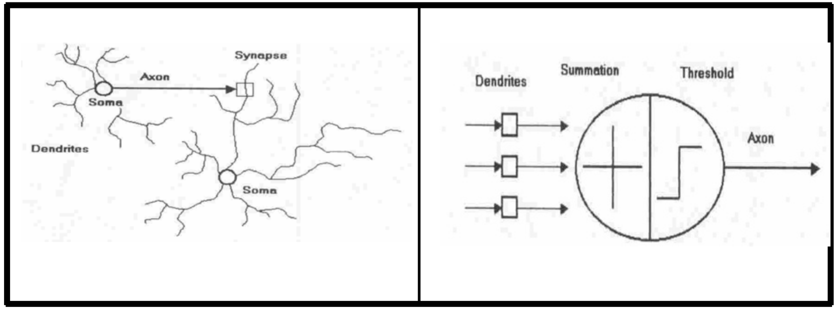
- Neurons (Nodes): The basic building blocks of a neural network. They receive input, process it, and produce an output.
- Layers: Neurons are organized into layers:
- Input Layer: Receives the initial data.
- Hidden Layers: Perform the complex processing of the data (can be multiple layers).
- Output Layer: Produces the final result.
- Connections (Edges): Connect neurons between layers. Each connection has a weight associated with it, which determines the strength of the connection.
- Activation Function: A function applied to the output of a neuron to introduce non-linearity, allowing the network to learn complex patterns.

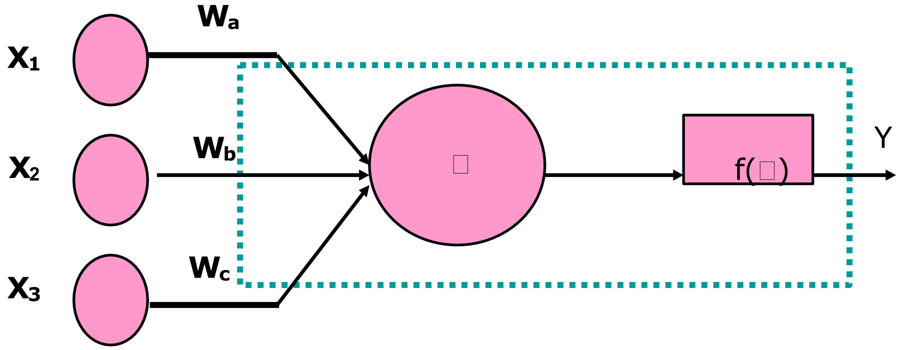


Structure and Components:

# How Neural Networks Work

- Data Flow: Data flows through the network from the input layer to the output layer.
- Weighted Sum: Each neuron receives inputs from the neurons connected to it. These inputs are multiplied by the weights of the connections, and the results are summed.
- Activation: The weighted sum is passed through the activation function, producing the neuron's output.
- Learning: The network learns by adjusting the weights of the connections. This is done through a process called backpropagation, where the network compares its output to the desired output and adjusts the weights to reduce the error.




$X_1$ $W_a$

$X_2$ $W_b$

$X_3$ $W_c$

□

f(□)

Y

Input units

Connect in weights

Summing function

Computation

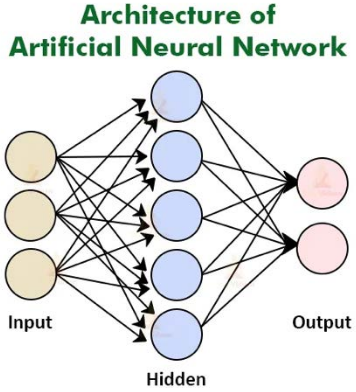
**Types of Neural Networks:**

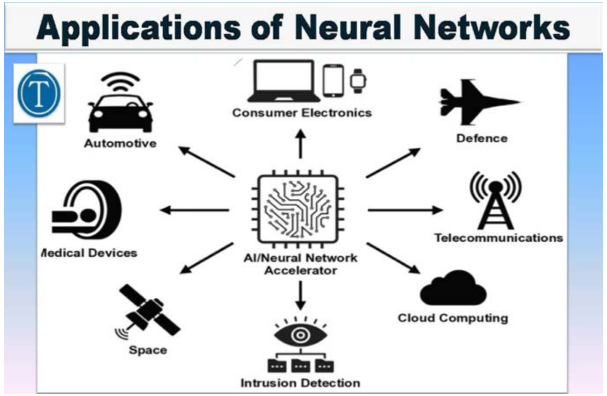
- **Feedforward Neural Networks:** The most basic type, where data flows in one direction from input to output.
- **Recurrent Neural Networks (RNNs):** Designed to handle sequential data, like text or time series. They have feedback loops that allow them to "remember" previous inputs.
- **Convolutional Neural Networks (CNNs):** Specialized for processing images and videos. They use convolutional layers to extract features from the input data.

**Applications of Neural Networks:**

Neural networks are used in a wide range of applications, including:

- **Image Recognition:** Identifying objects in images (e.g., facial recognition, object detection).
- **Natural Language Processing:** Understanding and generating human language (e.g., machine translation, sentiment analysis).
- **Speech Recognition:** Converting spoken words into text.
- **Recommendation Systems:** Suggesting products or content to users.
- **Medical Diagnosis:** Assisting in the diagnosis of diseases



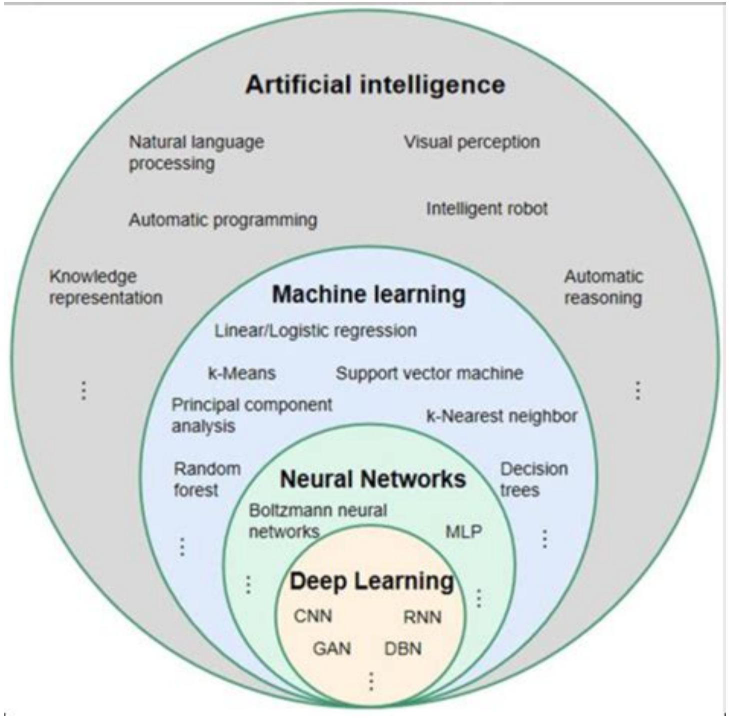


**Challenges of Neural Networks:**

- **Computational Cost:** Training large neural networks can be computationally expensive.
- **Data Requirements:** Neural networks typically require large amounts of training data.
- **Black Box Nature:** It can be difficult to understand how a neural network arrives at its decisions.

- Artificial neural networks are powerful tools in machine learning that can learn complex patterns and solve a wide range of problems.
- They are inspired by the structure of the human brain and are a key component of deep learning.
- Sources and related content

**Deep learning** is a subfield of machine learning that focuses on artificial neural networks with multiple layers (hence "deep"). These deep networks are capable of learning complex patterns and representations from vast amounts of data, often surpassing the performance of traditional machine learning algorithms on challenging tasks.



# Google Cloud AutoML (Vertex AI)

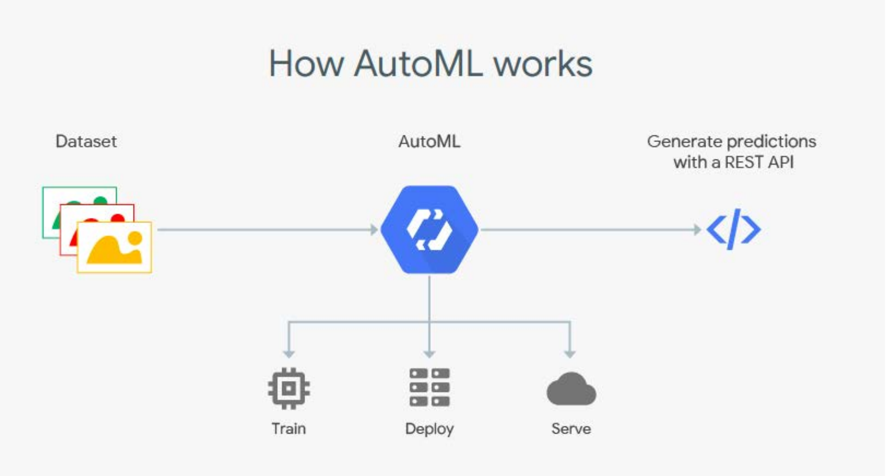
- Overview of AutoML and Vertex AI
- Creating a project
- Enabling required APIs
- Creating a Cloud Storage bucket

# Step-by-Step to start Google AutoML API (Vertex AI)

**1. Set Up Google Cloud Project**
•Go to: https://console.cloud.google.com/
•Create or select a project
•Enable **Billing**

**2. Enable Required APIs**
Go to: API Library
Enable the following:
•**Vertex AI API**
•**Cloud Storage API** (for dataset access)
• **IAM API** (for permissions)

**3. Create a Cloud Storage Bucket**
Used to store training data and model artifacts.
•Go to Cloud Storage
•Create a new bucket (e.g., my-automl-bucket)
•Upload your training dataset (CSV for tabular, images for vision, etc.)



# Create a Project for Vertex AI API

**1. Create a Google Cloud Project**

- Go to the Google Cloud Console
- Click on the **Project dropdown** (top navigation bar)
- Click **"New Project"**
- Enter:
    1. **Project name**
    2. **Billing account**
    3. (Optional) Organization
- Click **"Create"**
- Once it's created, switch to the project.

**2. Enable Vertex AI API**

- With your project selected, go to the API Library
- Search for **Vertex AI API**
- Click **Enable**
- Also enable:
- **Cloud Storage API** (to access datasets)
- **AM API** (for access control)

**3. Set Up Billing**

If you haven't already:
Go to Billing settings
Link your project to an existing billing account, or set one up

**4. Enable Cloud Storage**

Vertex AI requires data stored in **Google Cloud Storage**, so:
Go to Cloud Storage
Click **"Create Bucket"**
Upload your data (e.g., CSVs for tabular models.
Hands-On Exercise 1: Penguin Species Classification

# To set up billing for a Google Cloud project

**Open the Billing section**

- In the left-hand navigation menu, go to **Billing**.
- If it’s your first time, it may prompt you to set up a billing account.

**Create or link a billing account**

- If you don’t have a billing account: click **Create Account**, then enter your payment information (credit card, bank account, or other supported methods).
- If you already have one: select **Link a billing account** and choose the account you want.

**Attach billing to your project**

- Once you have a billing account, select your project in the **Billing** page. (Use Search.)
- Click **Link billing account** and choose the account you want to connect.
- Confirm.

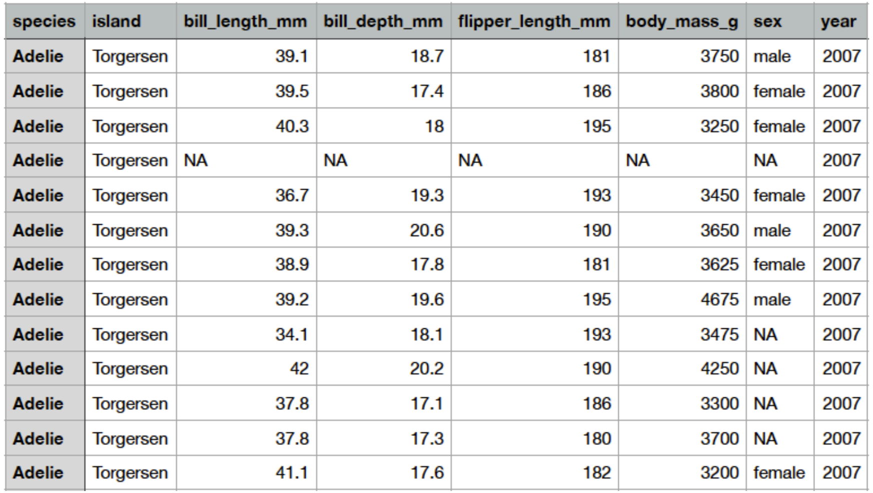
**Verify billing is active**

- Go to **Billing** → **Account Management**.
- You should see your billing account listed and linked to your project.

# Exercise 1: Penguin Species Classification

**Raw data for three different species of penguins: Adélie, Chinstrap, and Gentoo.**
In-session activity dataset (penguins.csv)

penguins

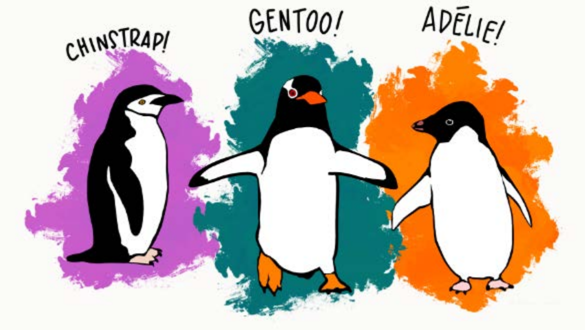


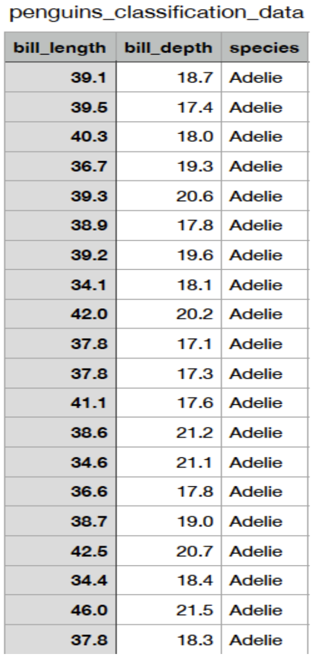
| species | island | bill_length_mm | bill_depth_mm | flipper_length_mm | body_mass_g | sex | year |
|---|---|---|---|---|---|---|---|
| Adelie | Torgersen | 39.1 | 18.7 | 181 | 3750 | male | 2007 |
| Adelie | Torgersen | 39.5 | 17.4 | 186 | 3800 | female | 2007 |
| Adelie | Torgersen | 40.3 | 18 | 195 | 3250 | female | 2007 |
| Adelie | Torgersen | NA | NA | NA | NA | NA | 2007 |
| Adelie | Torgersen | 36.7 | 19.3 | 193 | 3450 | female | 2007 |
| Adelie | Torgersen | 39.3 | 20.6 | 190 | 3650 | male | 2007 |
| Adelie | Torgersen | 38.9 | 17.8 | 181 | 3625 | female | 2007 |
| Adelie | Torgersen | 39.2 | 19.6 | 195 | 4675 | male | 2007 |
| Adelie | Torgersen | 34.1 | 18.1 | 193 | 3475 | NA | 2007 |
| Adelie | Torgersen | 42 | 20.2 | 190 | 4250 | NA | 2007 |
| Adelie | Torgersen | 37.8 | 17.1 | 186 | 3300 | NA | 2007 |
| Adelie | Torgersen | 37.8 | 17.3 | 180 | 3700 | NA | 2007 |
| Adelie | Torgersen | 41.1 | 17.6 | 182 | 3200 | female | 2007 |

Adélie Penguin
Gentoo Penguin
Chinstrap Penguin

# Exercise 1: Data for AutoML Supervised Classification (Tabular)

**Training data for three different species of penguins: Adélie, Chinstrap, and Gentoo.**
In-session activity dataset
(penguins_classification_data.csv)




penguins_classification_data

| bill_length | bill_depth | species |
|---|---|---|
| 39.1 | 18.7 | Adelie |
| 39.5 | 17.4 | Adelie |
| 40.3 | 18.0 | Adelie |
| 36.7 | 19.3 | Adelie |
| 39.3 | 20.6 | Adelie |
| 38.9 | 17.8 | Adelie |
| 39.2 | 19.6 | Adelie |
| 34.1 | 18.1 | Adelie |
| 42.0 | 20.2 | Adelie |
| 37.8 | 17.1 | Adelie |
| 37.8 | 17.3 | Adelie |
| 41.1 | 17.6 | Adelie |
| 38.6 | 21.2 | Adelie |
| 34.6 | 21.1 | Adelie |
| 36.6 | 17.8 | Adelie |
| 38.7 | 19.0 | Adelie |
| 42.5 | 20.7 | Adelie |
| 34.4 | 18.4 | Adelie |
| 46.0 | 21.5 | Adelie |
| 37.8 | 18.3 | Adelie |

# AutoML Supervised Classification (Tabular)

**Goal**: Train a model that predicts a **categorical label** (e.g., Yes/No, class A/B/C) from structured/tabular data using Google's AutoML.

**1. Prepare Your Data**

**Format:**

- Use a **CSV** file
- One **header row** with feature names
- One column must be your **target label** (e.g., "Category")
- The **target column** is "Subscribed" (binary classification)

**2. Upload to Cloud Storage:**

- Go to Cloud Storage > Create a bucket
- Upload your dataset

**3. Create Dataset in Vertex AI**

- Go to Vertex AI > Datasets
- Click **+ Create**
- Choose **Tabular** and **Classification**
- Import your data from:

  **Cloud Storage** (CSV file)
- Select the **target column.**

**4. Train Your Model**

- After the dataset is imported, click **Train New Model**
- Set:
  1. Model name
  2. Target column
  3. Model type: **AutoML**
  4. Training budget (e.g., 1 hour or $50 — you can limit this)
- Click **Start Training**

  Google will automatically:
  1. Select algorithms
  2. Do feature engineering
  3. Evaluate performance (e.g., accuracy, precision, recall)

# Preparing CSV for AutoML Classification

**Structure Your CSV File**
Your dataset should be **tabular** and follow this structure:

| feature_1 | feature_2 | ... | feature_n | label |
|---|---|---|---|---|
| value | value | ... | value | class_A |
| value | value | ... | value | class_B |
| value | value | ... | value | class_C |

- The **last column** is the **target label** (a **categorical** value like A, B, C).
- Include a **header row** with column names.
- All rows should have **complete data** (fill or remove missing values).

**Rules for Format Compatibility**

**File type**: CSV only (comma-separated, UTF-8 encoding)
**Header row**: Required
**Target column**:

  - Must be **categorical** (strings, not numbers like 1/2/3)
  - Should not contain missing values

**No special characters or formulas**
**No merged cells**

**Save the CSV File**

- Save as: your_data.csv
- Make sure encoding is **UTF-8**
- You can test open in a plain text editor to confirm format

## Upload to Cloud Storage

Go to Google Cloud Storage
Create a **bucket** (if you don't have one)
Upload your your_data.csv file
Copy the **full path**, e.g.:
gs://my-bucket-name/your_data.csv

## Import Into Vertex AI (AutoML)

Go to Vertex AI > Datasets
Click **Create Dataset**
Select **Tabular**, and **Classification**
Choose **"Import data from Cloud Storage"**
Paste the path: gs://.../your_data.csv
Select the **label column**

## Tips for Best Results

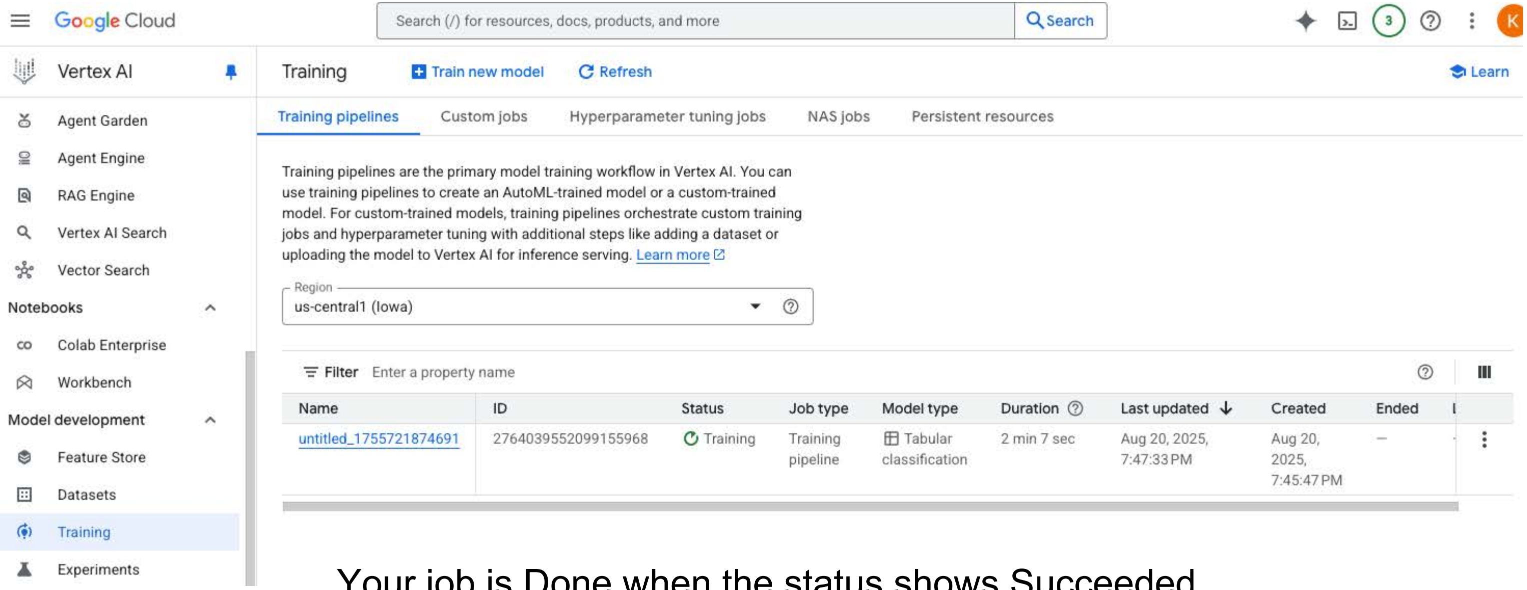
Ensure classes (labels) are balanced—AutoML handles imbalance, but balanced data improves performance.

Avoid high-cardinality categorical features (>1000 unique values).

Pre-clean missing values or standardize them (e.g., fill or remove).

Use clear, consistent naming (avoid spaces in column headers).

# Train Your Model

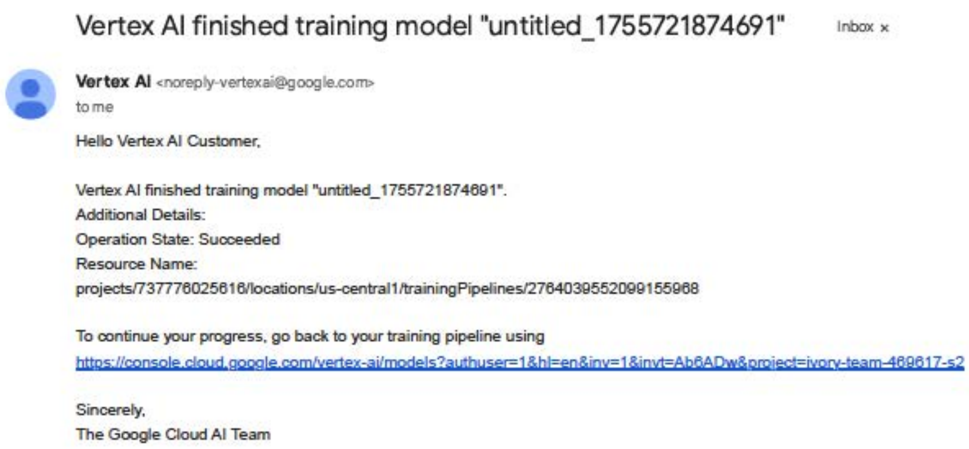


Training pipelines are the primary model training workflow in Vertex AI. You can use training pipelines to create an AutoML-trained model or a custom-trained model. For custom-trained models, training pipelines orchestrate custom training jobs and hyperparameter tuning with additional steps like adding a dataset or uploading the model to Vertex AI for inference serving. Learn more

Region: us-central1 (Iowa)

| Name | ID | Status | Job type | Model type | Duration | Last updated | Created | Ended |
|---|---|---|---|---|---|---|---|---|
| untitled_1755721874691 | 2764039552099155968 | Training | Training pipeline | Tabular classification | 2 min 7 sec | Aug 20, 2025, 7:47:33 PM | Aug 20, 2025, 7:45:47 PM | – |

Your job is Done when the status shows Succeeded (green check). From the run page you can open the Graph and Logs to confirm each step completed.



Vertex AI finished training model "untitled_1755721874691"

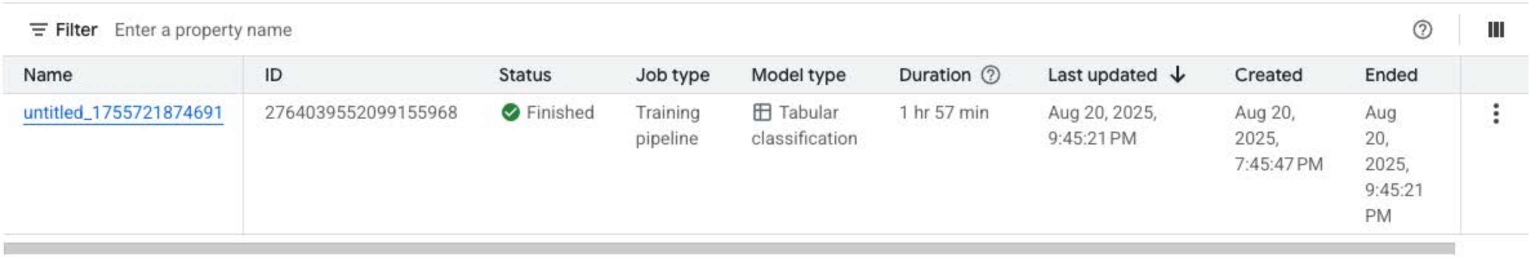
**Vertex AI** <noreply-vertexai@google.com>
to me

Hello Vertex AI Customer,

Vertex AI finished training model "untitled_1755721874691".
Additional Details:
Operation State: Succeeded
Resource Name:
projects/737776025616/locations/us-central1/trainingPipelines/2764039552099155968

To continue your progress, go back to your training pipeline using
https://console.cloud.google.com/vertex-ai/models?authuser=1&hl=en&inv=1&invt=Ab6ADw&project=ivory-team-469617-s2

Sincerely,
The Google Cloud AI Team



| Name | ID | Status | Job type | Model type | Duration | Last updated ↓ | Created | Ended |
|---|---|---|---|---|---|---|---|---|
| untitled_1755721874691 | 2764039552099155968 | ✓ Finished | Training pipeline | Tabular classification | 1 hr 57 min | Aug 20, 2025, 9:45:21 PM | Aug 20, 2025, 7:45:47 PM | Aug 20, 2025, 9:45:21 PM |

# Evaluate and deploy the AutoML Model

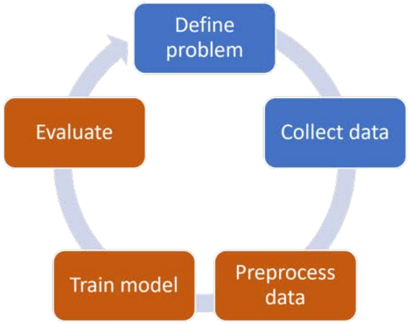
**Evaluate the Model**
After training:

- Go to the **Model Registry** tab
- Review metrics: accuracy, confusion matrix, AUC, etc.
- Download predictions if needed

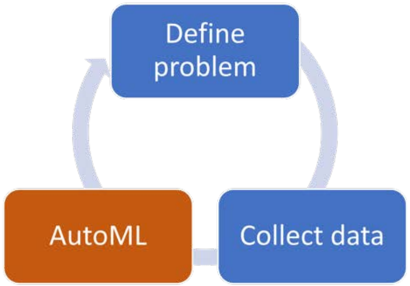
**Deploy the Model**
To make **online predictions**:

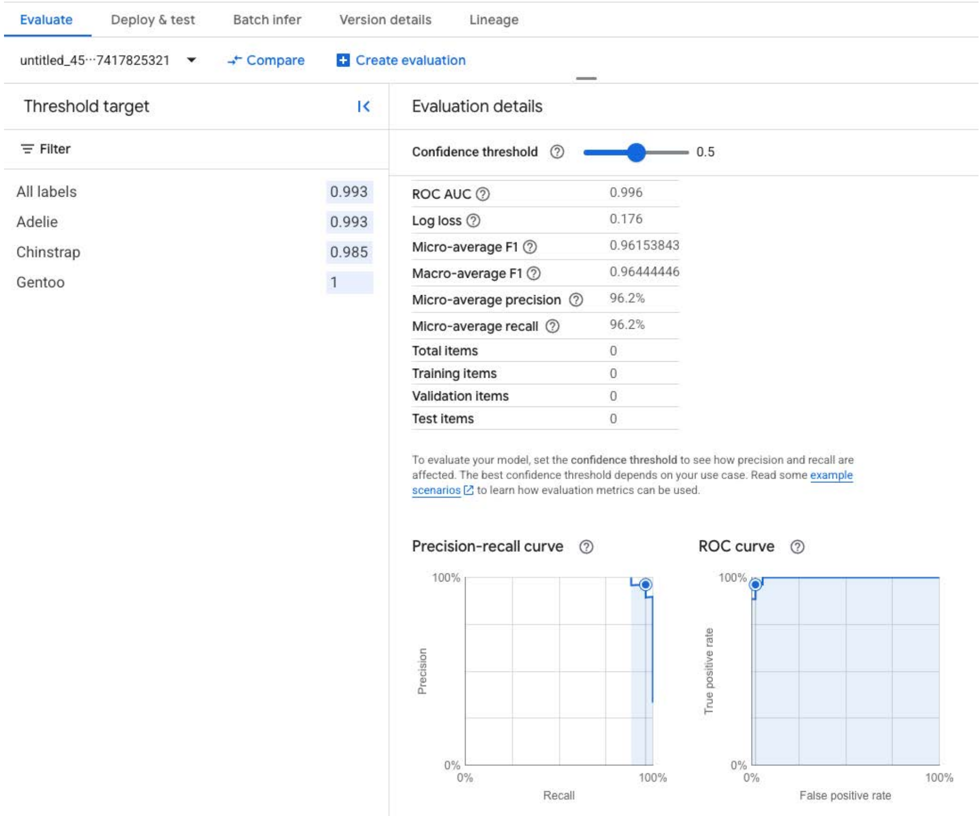
- Click **Deploy to Endpoint**
- Once deployed, use the REST API or Python SDK to send prediction requests

**Traditional ML training workflow**



**AutoML workflow**





Threshold target

| | |
|---|---|
| All labels | 0.993 |
| Adelie | 0.993 |
| Chinstrap | 0.985 |
| Gentoo | 1 |

Evaluation details

Confidence threshold 0.5

| | |
|---|---|
| ROC AUC | 0.996 |
| Log loss | 0.176 |
| Micro-average F1 | 0.96153843 |
| Macro-average F1 | 0.96444446 |
| Micro-average precision | 96.2% |
| Micro-average recall | 96.2% |
| Total items | 0 |
| Training items | 0 |
| Validation items | 0 |
| Test items | 0 |

To evaluate your model, set the **confidence threshold** to see how precision and recall are affected. The best confidence threshold depends on your use case. Read some example scenarios to learn how evaluation metrics can be used.

# Thank you, All