Intelligence System:

Supervised Learning in Machine Learning

សាកលវិទ្យាល័យន័រតុន
NORTON UNIVERSITY

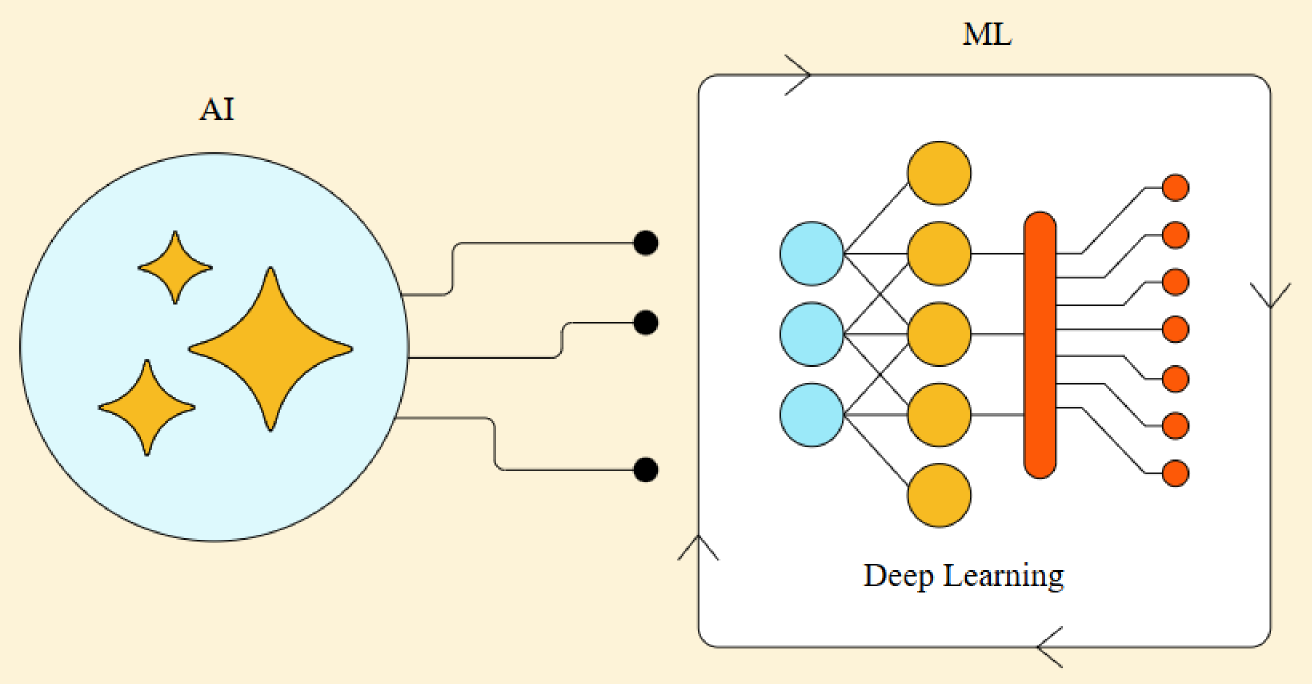
Intelligence System Development

2024 – 2025
Y4E1 – DCS – NU

By: SEK SOCHEAT
Advisor to DCS and Lecturer
Mobile: 017 879 967
Email: socheat.sek@gmail.com

Table of Contents

Introduction to Machine Learning



https://brave.com/ai/images/Brave-Blog-02.svg

This is an introduction to supervised, unsupervised, Semin-supervised and reinforcement learning, followed by key data preparation steps, including data cleaning, scaling, splitting, and encoding, to enhance model performance.

What is Supervised Learning?

Supervised Learning is a type of machine learning where a model is trained using labeled data. In supervised learning, each training example is a pair consisting of an input (features) and the correct output (label or target). The model learns to map inputs to outputs by finding patterns in the labeled data, allowing it to make predictions on new, unseen data.

Key Concepts of Supervised Learning:

- **Labeled Data:** The dataset contains inputs paired with their correct outputs, providing "supervision" to guide learning.
- **Training:** The model learns from labeled data to minimize errors in predicting the output.

Types:

- **Classification:** Predicts a discrete label (e.g., identifying emails as "spam" or "not spam").
- **Regression:** Predicts a continuous value (e.g., forecasting house prices based on features).

សាកលវិទ្យាល័យន័រតុន
NORTON UNIVERSITY

Supervised Learning:

Classification

Supervised Learning

Training a model on labeled data where each data point has an associated output.

What is Classification?

Classification: Used for categorical outputs. It is a supervised machine learning task aimed at predicting discrete labels or categories for new data points, by learning patterns from labeled training data. The model assigns data into predefined classes based on its features.

Supervised Learning

Key Points about Classification:

Input and Output: The model takes feature inputs and predicts a categorical output (e.g., class labels like "cat" or "dog").

Example Algorithms: ***Common classification algorithms include:***

- **Decision Trees:** Splits data into subsets based on feature values.
- **Logistic Regression:** Estimates probabilities for binary classes.
- **Support Vector Machines (SVM):** Finds a boundary that best separates classes.
- **K-Nearest Neighbors (KNN):** Classifies based on the closest labeled data points.
- **Naive Bayes:** Uses probability based on feature likelihoods.
- **Neural Networks:** Learns complex patterns in data through multiple layers.

Supervised Learning

Key Points about Classification:

Decision Trees: Splits data into subsets based on feature values.

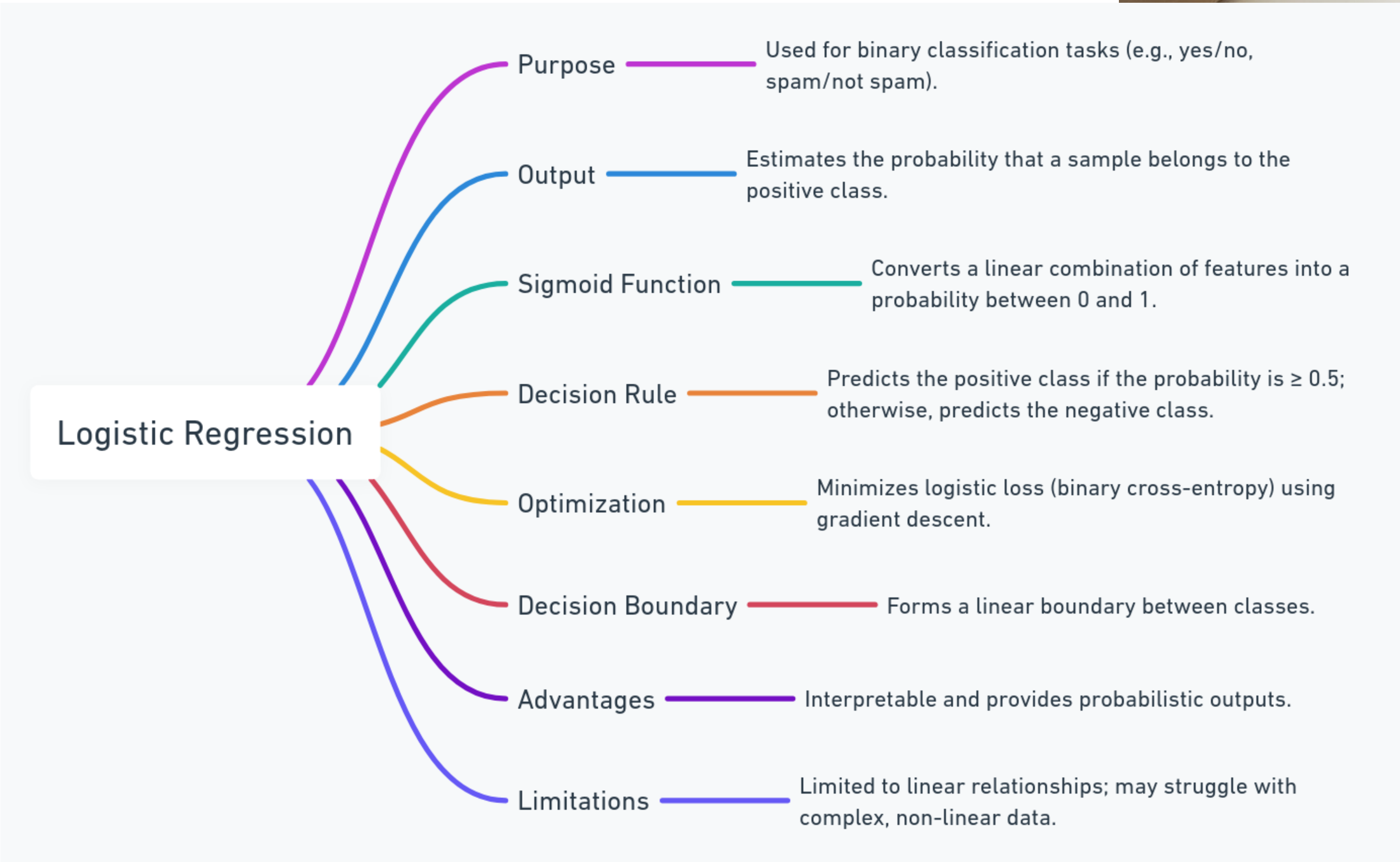
Supervised Learning

Key Points about Classification:

Logistic Regression:

Estimates probabilities

for binary classes.

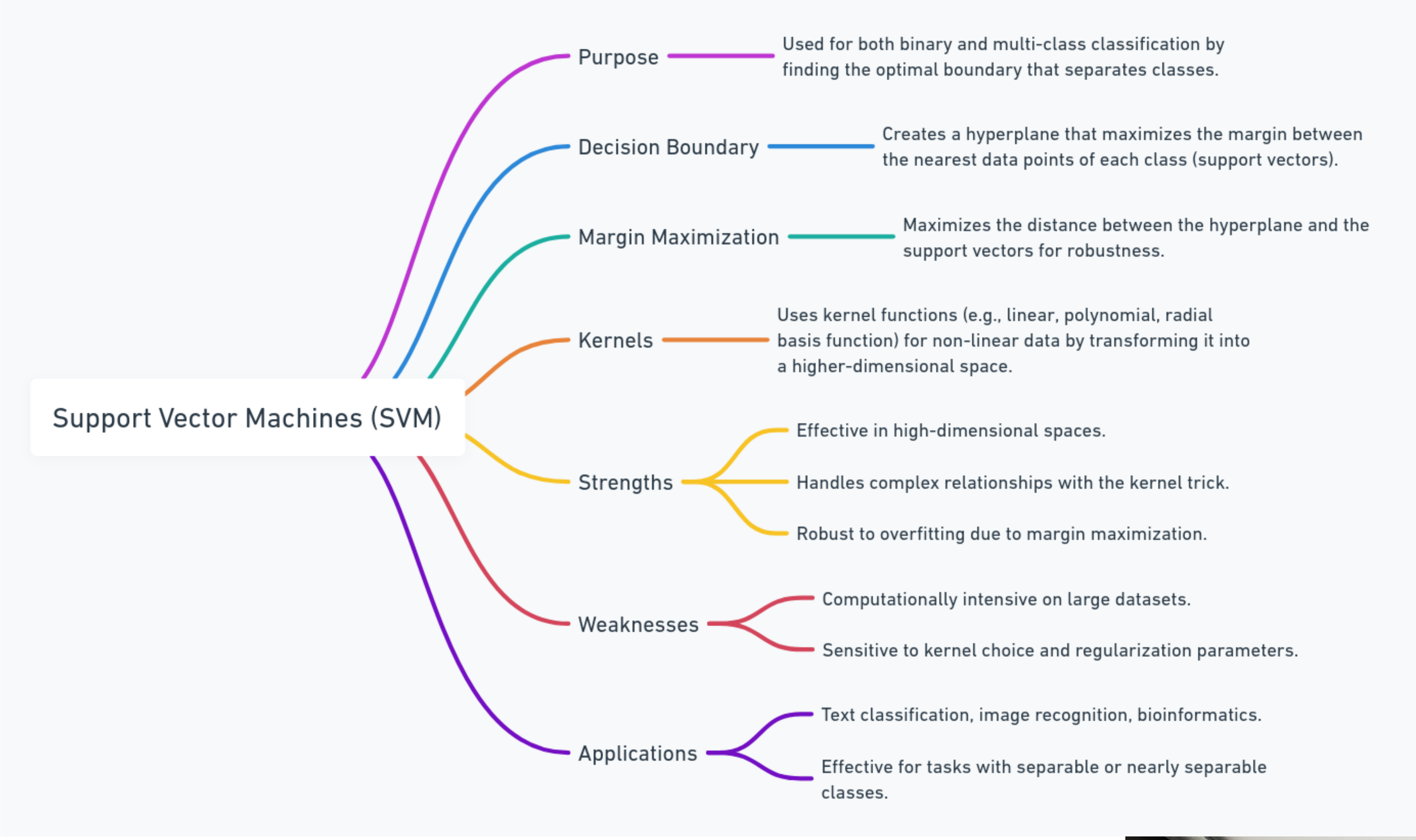


Supervised Learning

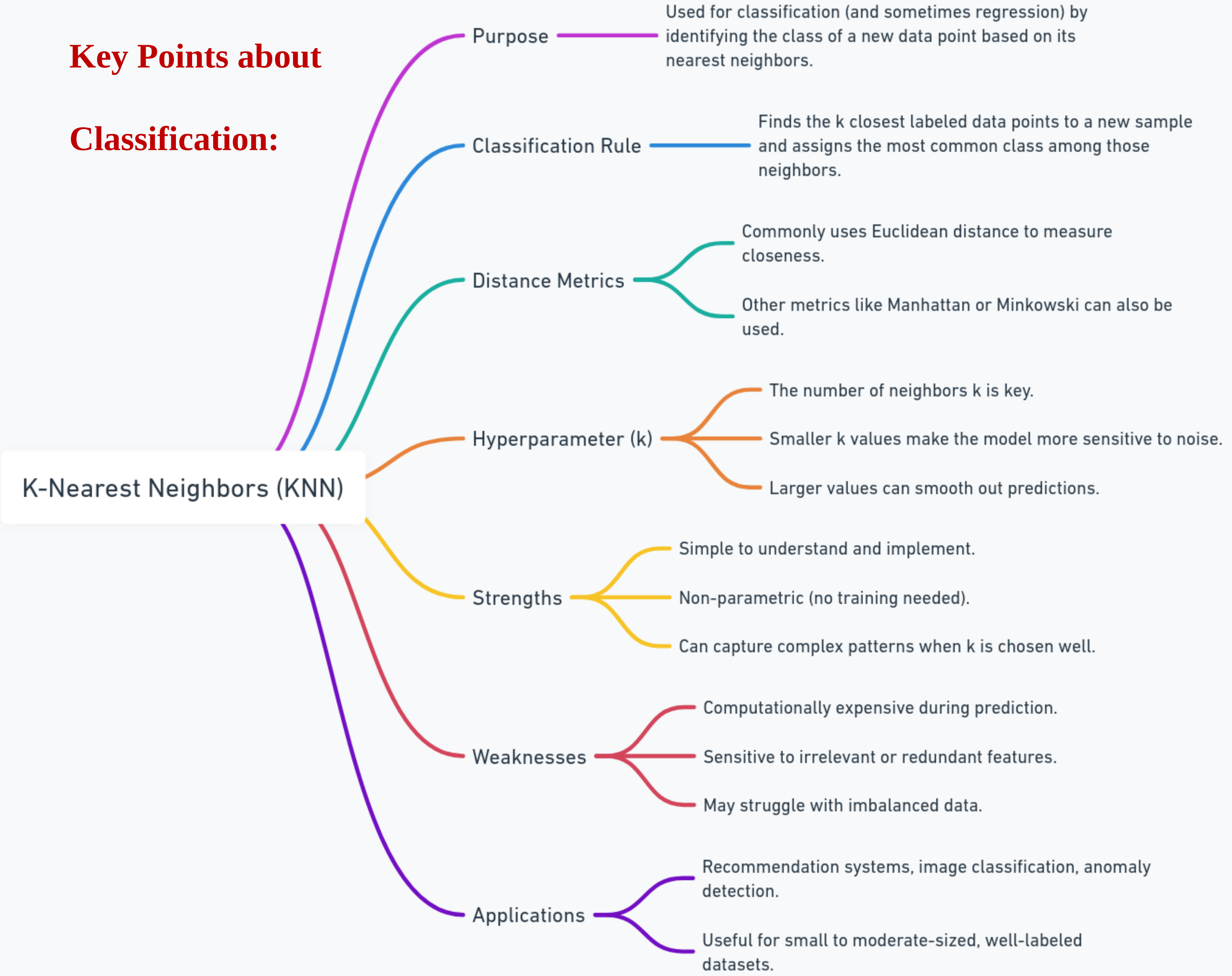
Key Points about Classification:

Support Vector Machines (SVM):

Finds a boundary that best separates classes.



Key Points about Classification:



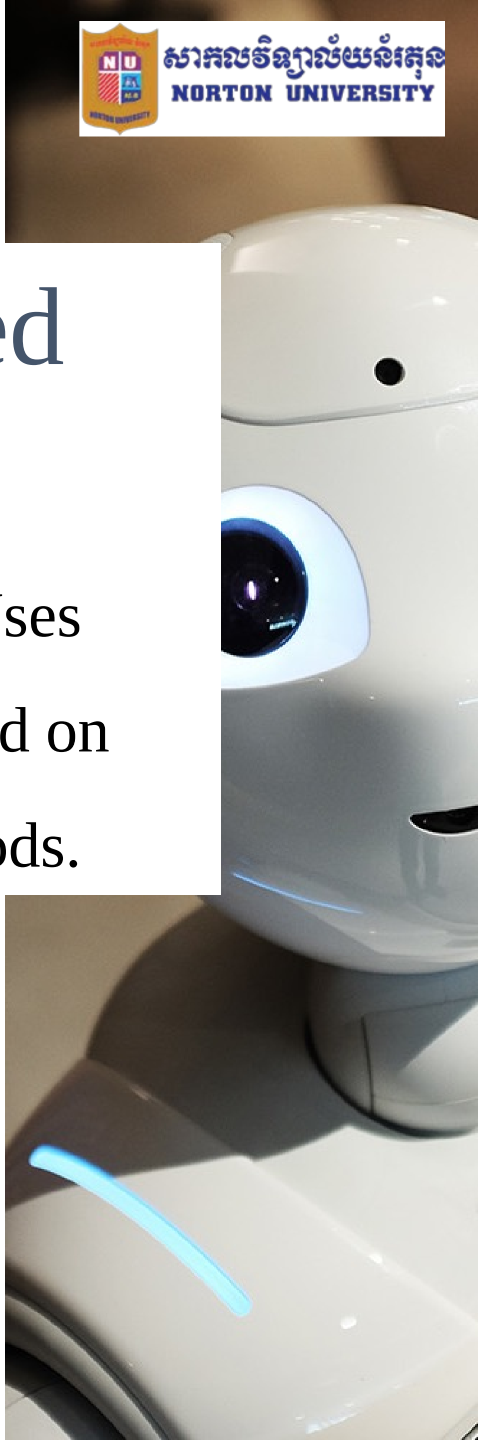
Supervised Learning

K-Nearest Neighbors (KNN): Classifies based on the closest labeled data points.

Supervised Learning

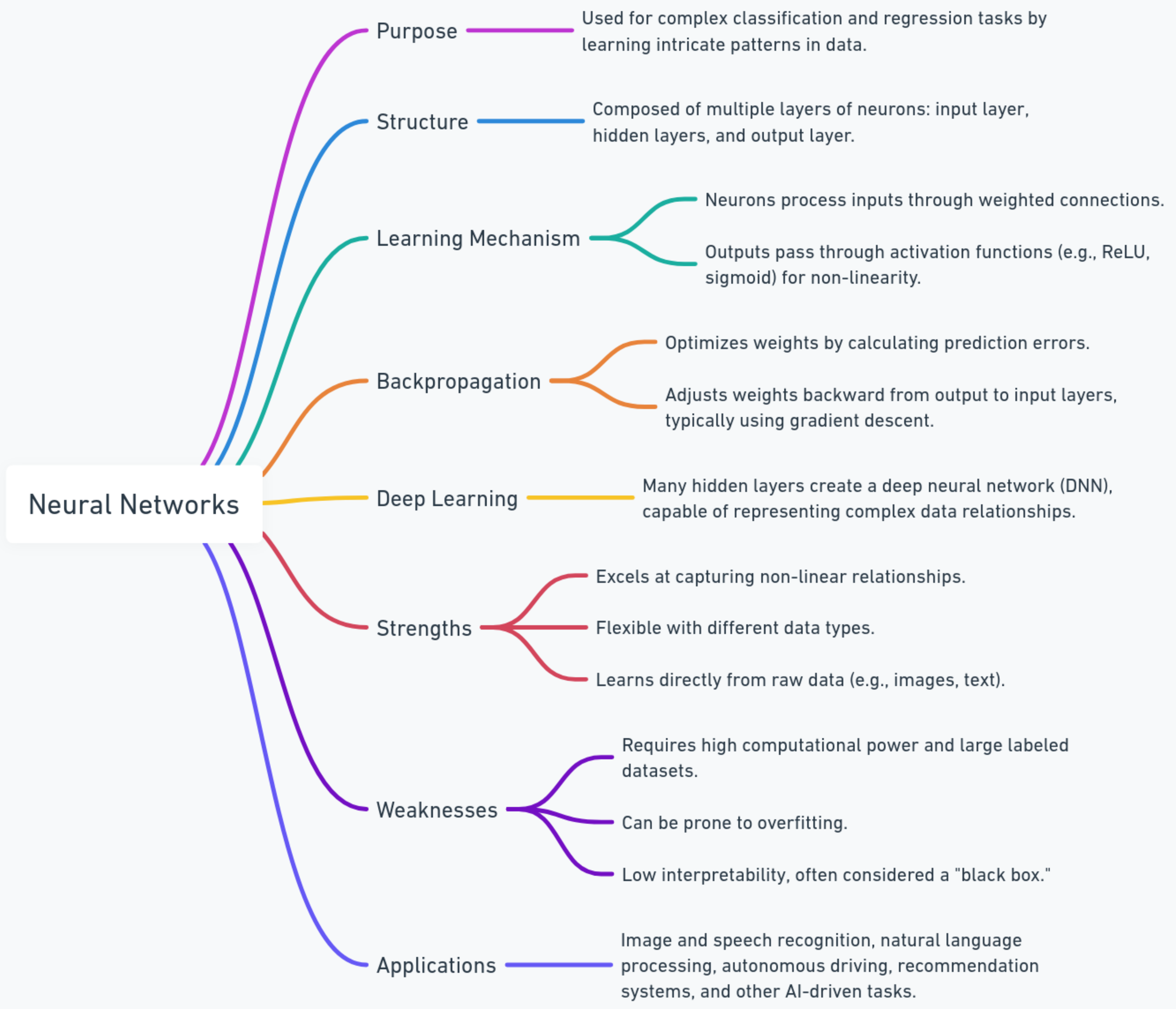
Naive Bayes: Uses probability based on feature likelihoods.

Key Points about Classification:

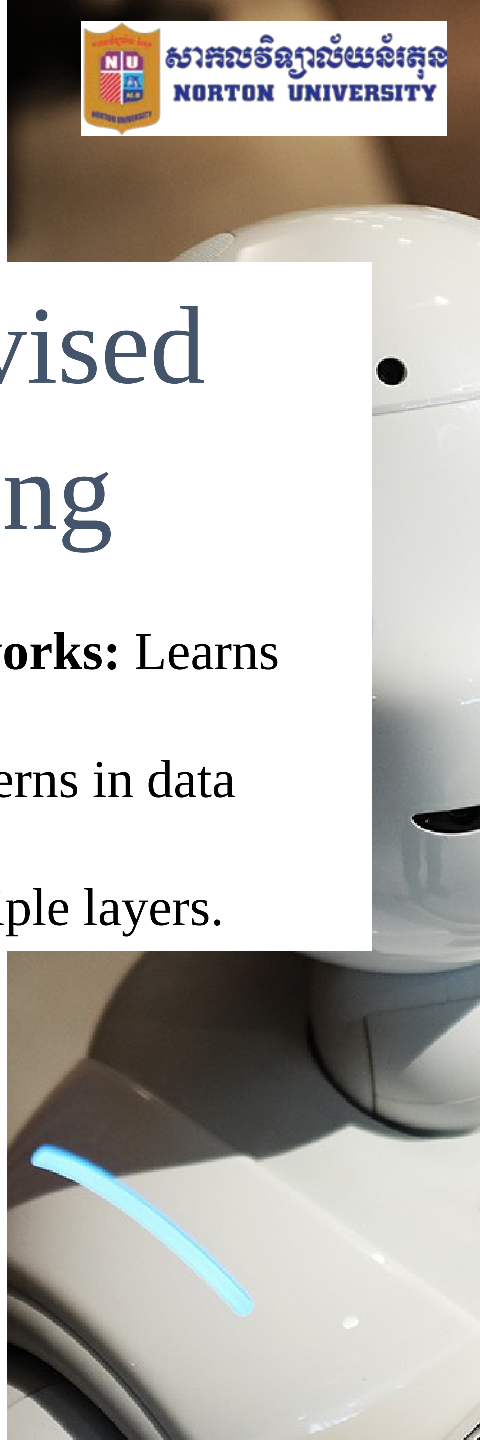


Supervised Learning

Neural Networks: Learns complex patterns in data through multiple layers.



Key Points about Classification:



Supervised Learning

Real-World Examples:

- **Email Spam Detection:** Classifying emails as "spam" or "not spam."
- **Image Recognition:** Classifying images as "cat," "dog," or "bird."
- **Medical Diagnosis:** Predicting diseases based on symptoms.

Supervised Learning

Classification Types:

- **Binary Classification:** Only two possible classes (e.g., "yes" or "no").
- **Multiclass Classification:** More than two classes (e.g., types of animals).
- **Multilabel Classification:** Each data point can belong to multiple classes (e.g., an image with both "cat" and "outdoor" labels).

Supervised Learning

Process:

- **Training:** The model learns from a labeled dataset.
- **Prediction:** The model predicts the class of new, unseen data.
- **Evaluation:** Performance is assessed using metrics like accuracy, precision, and recall.

Classification is essential in various applications, enabling systems to make informed decisions based on learned categories.

Supervised Learning

Project Problem: Dog vs. Cat Classification

Build a machine learning model to classify images as either dogs or cats. Train the model on labeled images, preprocess the data, and evaluate its prediction accuracy on new images.

Objective:

- **Goal:** Build a classifier that accurately distinguishes between dog and cat images.
- **Skills Practiced:** Data processing, model training, evaluation, and basic image classification.

This is a hands-on introduction to image classification using supervised learning techniques.

Supervised Learning

Steps to Create and Set Up a Virtual Environment in VS Code

Step 1: python -m venv VirtualEnv_Name

Step 2: To activate virtual environment

1. **Window:** VirtualEnv_Name\Scripts\activate
2. **MacOS or Linux:** source VirtualEnv_Name/bin/activate

Step 3: To ensure you have the latest version of pip, type:

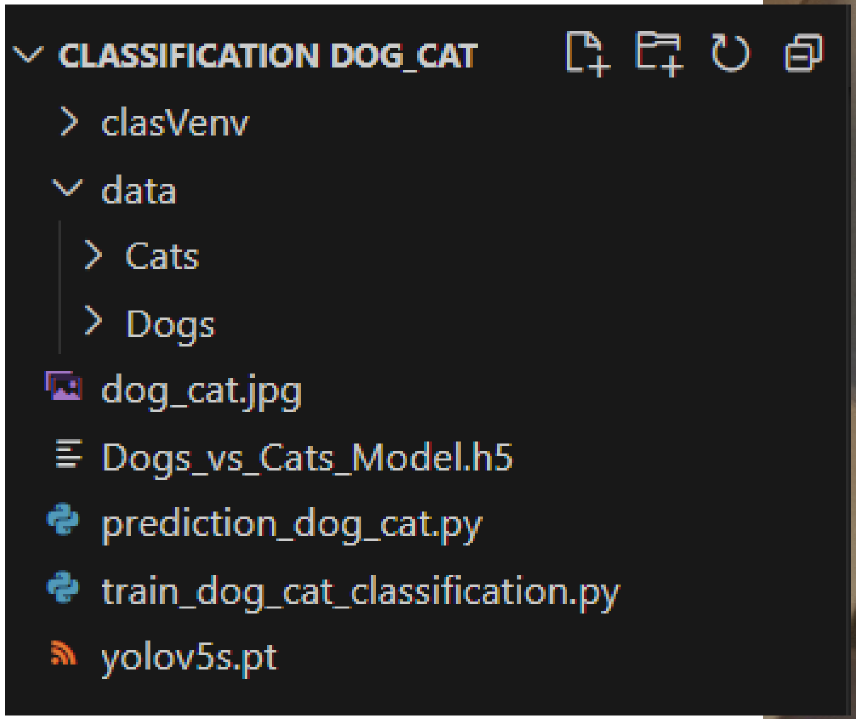
```
python -m pip install --upgrade pip
```

Step 4: Installing all libraries:

1. pip install tensorflow
2. pip install pillow
3. pip install scipy
4. pip install torch torchvision torchaudio
5. pip install opencv-python
6. pip install pandas

Step 5. Confirming Installed Libraries:

```
pip list
```



Supervised Learning

Example: train_dog_cat_by_category.py

```python
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout
from tensorflow.keras.optimizers import Adam

# Paths for training data
train_dir = 'data'  # Contains 'Cats' and 'Dogs' subdirectories

# Data Augmentation and Normalization
train_datagen = ImageDataGenerator(
    rescale=1.0/255,
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    validation_split=0.2  # 20% of the data used for validation
)

# Train and validation generators
train_generator = train_datagen.flow_from_directory(
    train_dir,
    target_size=(150, 150),
    batch_size=32,
    class_mode='binary',  # Binary classification: cat vs. dog
    subset='training'
)

validation_generator = train_datagen.flow_from_directory(
    train_dir,
    target_size=(150, 150),
    batch_size=32,
    class_mode='binary',
    subset='validation'
)
```

```python
# Model Architecture
model = Sequential([
    Conv2D(32, (3, 3), activation='relu', input_shape=(150, 150, 3)),
    MaxPooling2D(2, 2),
    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D(2, 2),
    Conv2D(128, (3, 3), activation='relu'),
    MaxPooling2D(2, 2),
    Flatten(),
    Dense(512, activation='relu'),
    Dropout(0.5),
    Dense(1, activation='sigmoid')  # Single neuron for binary classification
])

# Compile the Model
model.compile(
    optimizer=Adam(learning_rate=0.001),
    loss='binary_crossentropy',
    metrics=['accuracy']
)

# Train the Model
epochs = 15
history = model.fit(
    train_generator,
    steps_per_epoch=train_generator.samples // train_generator.batch_size,
    epochs=epochs,
    validation_data=validation_generator,
    validation_steps=validation_generator.samples // validation_generator.batch_size
)

# Evaluate the Model
loss, accuracy = model.evaluate(validation_generator)
print(f'Validation accuracy: {accuracy:.2f}')

# Save the Model
model.save("Dogs_vs_Cats_Model.h5")
```

Supervised Learning

Example: train_dog_cat_by_category.py

Supervised Learning

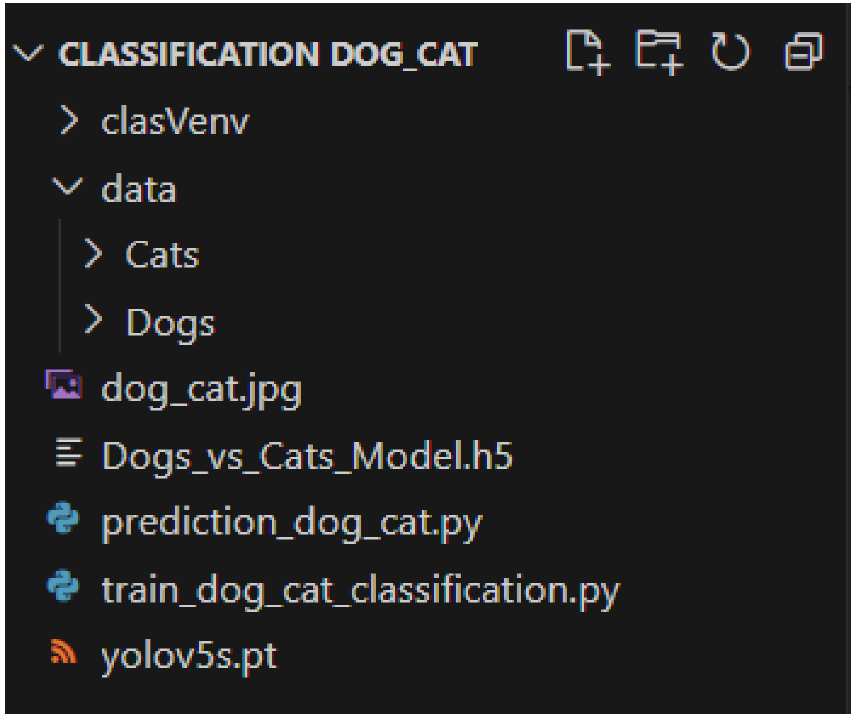
Example: **prediction_dog_cat.py**

```python
import torch
from PIL import Image

# Load the YOLOv5 model (pre-trained on the COCO dataset, which includes 'cat' and 'dog' classes)
model = torch.hub.load('ultralytics/yolov5', 'yolov5s', pretrained=True)

# Define the class labels
# 'cat' is class 15 and 'dog' is class 16 in the COCO dataset
class_labels = {15: 'Cat', 16: 'Dog'}

def detect_cat_and_dog(image_path):
    """Detect if both a cat and a dog are present in the image."""
    # Load the image
    img = Image.open(image_path)

    # Run inference on the image using YOLOv5
    results = model(img)

    # Extract detection results
    detections = results.pandas().xyxy[0]  # Get detection results in pandas DataFrame format

    # Check if a cat and a dog are present in the detections
    cat_present = any(detections['class'] == 15)  # Check if class 15 ('cat') is in the detections
    dog_present = any(detections['class'] == 16)  # Check if class 16 ('dog') is in the detections

    # Display the results
    if cat_present and dog_present:
        print("Both a cat and a dog are present in the image.")
    elif cat_present:
        print("Only a cat is present in the image.")
    elif dog_present:
        print("Only a dog is present in the image.")
    else:
        print("Neither a cat nor a dog is present in the image.")
```

Supervised Learning

Example: **prediction_dog_cat.py**

```python

# Main function to interact with the user
def main():
    print("Welcome to the Cat and Dog Recognizer!")
    image_path = input("Please enter the path to the image file: ")
    detect_cat_and_dog(image_path)

if __name__ == "__main__":
    main()

```



Supervised Learning:

Regression

Supervised Learning

What is Regression?

Regression is a type of supervised machine learning task focused on predicting continuous numerical values for new data points based on patterns learned from labeled training data. Unlike classification, which predicts discrete categories, regression models output a continuous value.

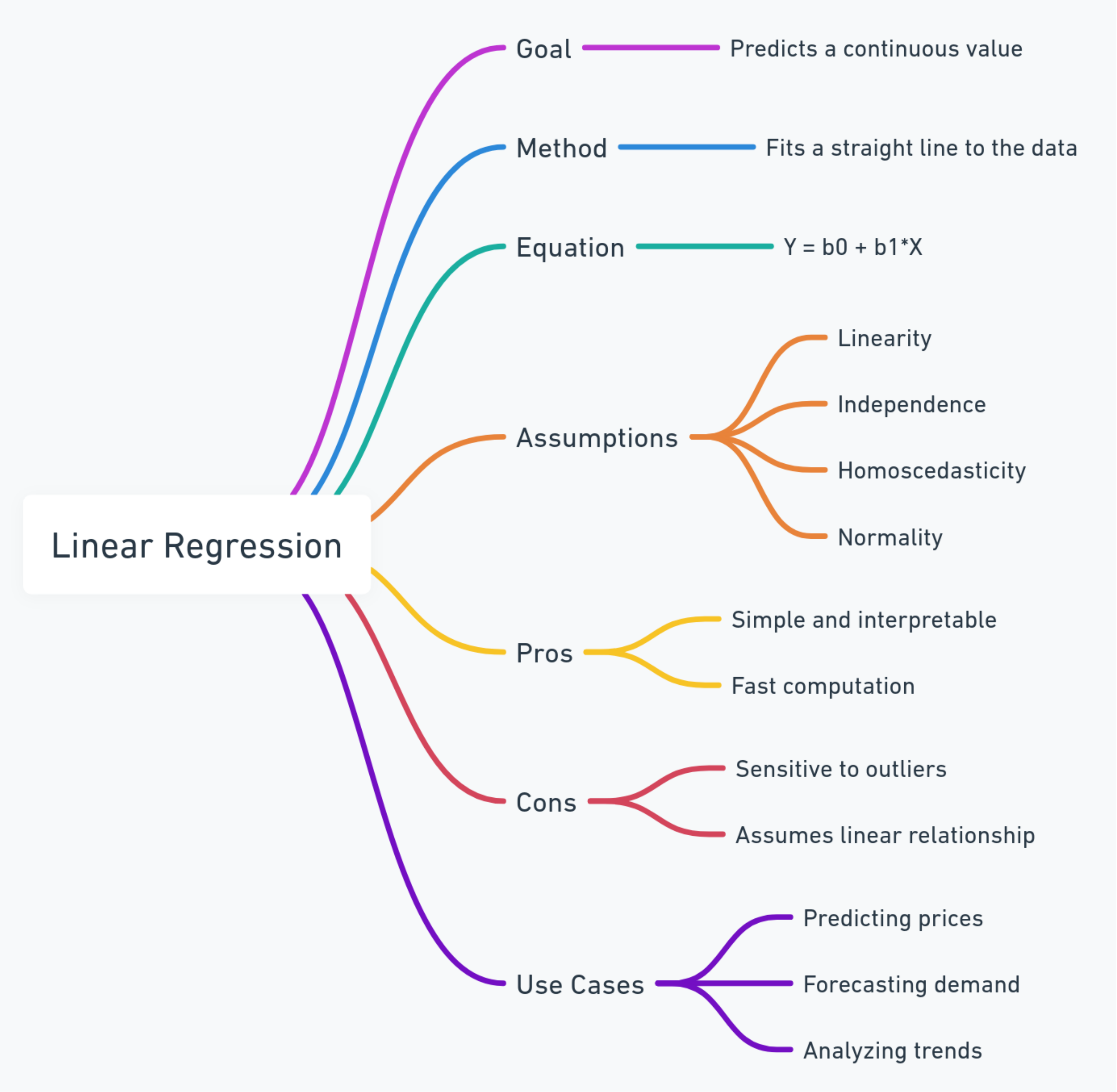
Supervised Learning

Key Points about Regression:

To predict a continuous output (e.g., price, temperature, or probability).

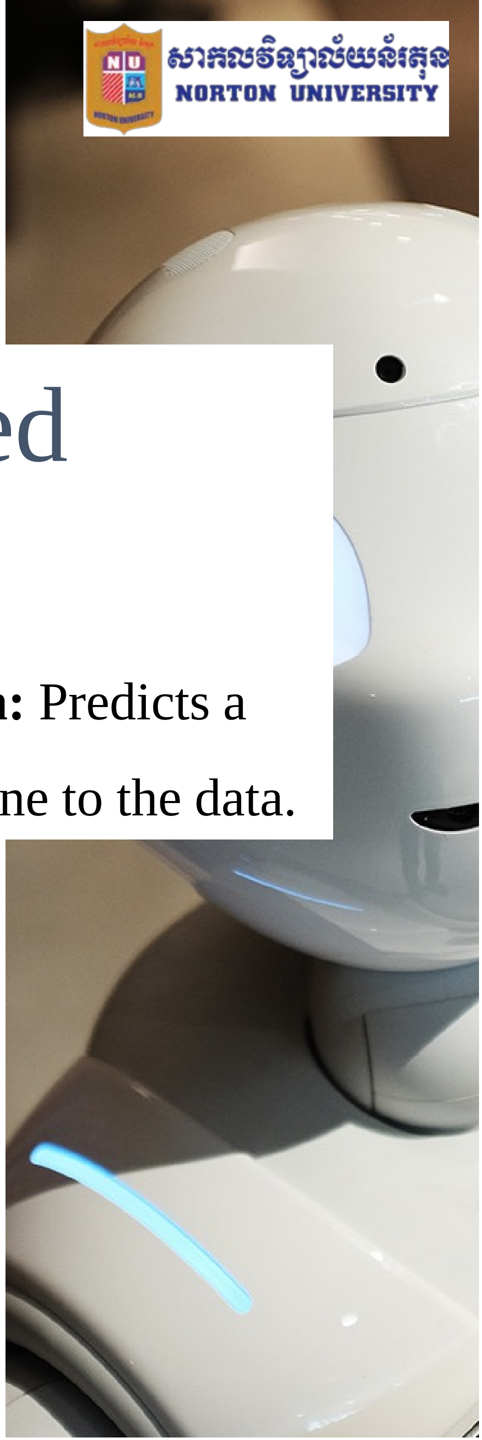
Example Algorithms: Common regression algorithms include:

- **Linear Regression:** Predicts a value by fitting a line to the data.
- **Polynomial Regression:** Fits a curve to capture nonlinear relationships.
- **Support Vector Regression (SVR):** Uses support vectors to predict continuous values.
- **Decision Trees/Random Forests:** Regression variants predict continuous outcomes.
- **Neural Networks:** Complex models that can capture intricate patterns for regression tasks.



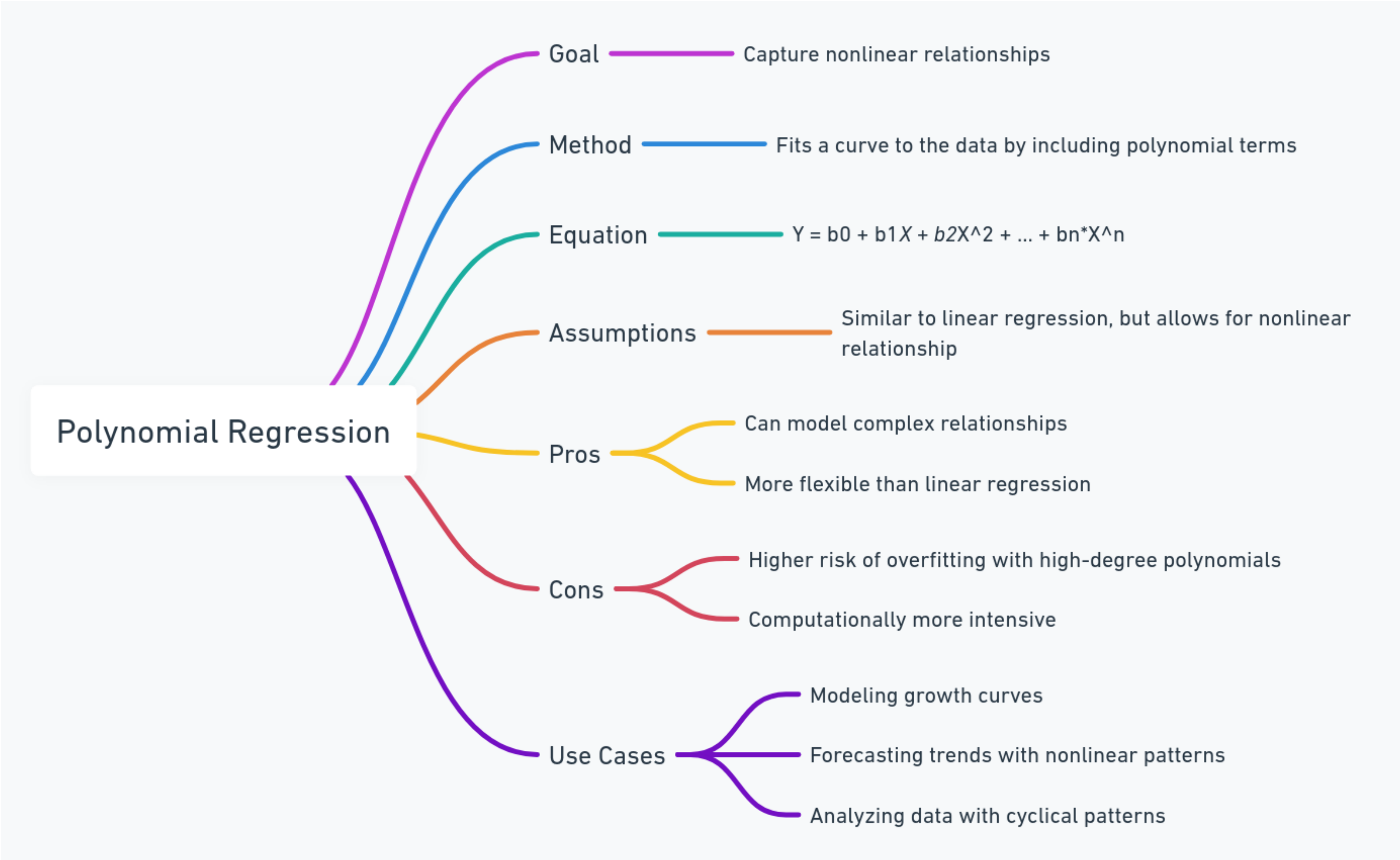
Supervised Learning

Linear Regression: Predicts a value by fitting a line to the data.



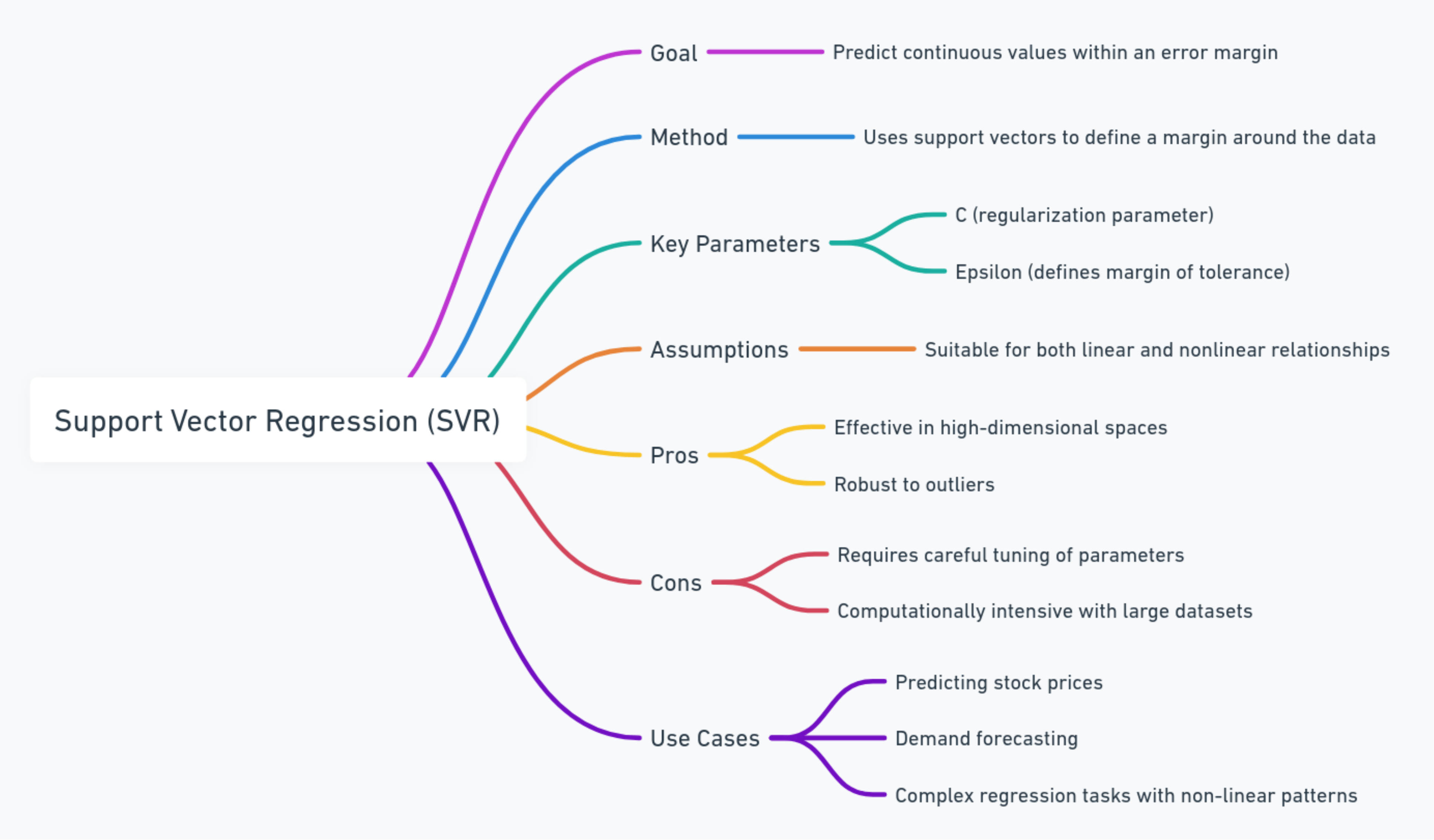
Supervised Learning

Polynomial Regression: Fits a curve to capture nonlinear relationships.



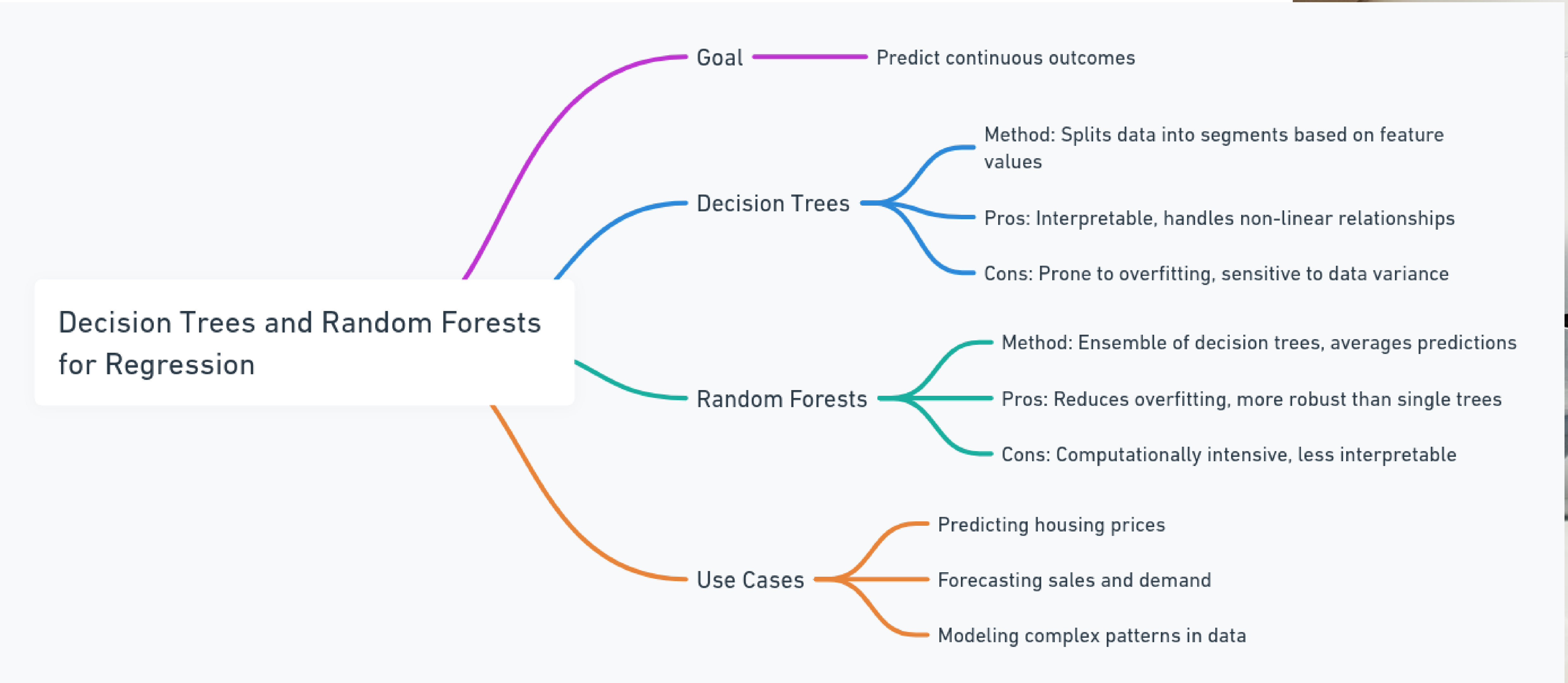
Supervised Learning

Support Vector Regression (SVR): Uses support vectors to predict continuous values.



Supervised Learning

Decision Trees/Random Forests: Regression variants predict continuous outcomes.



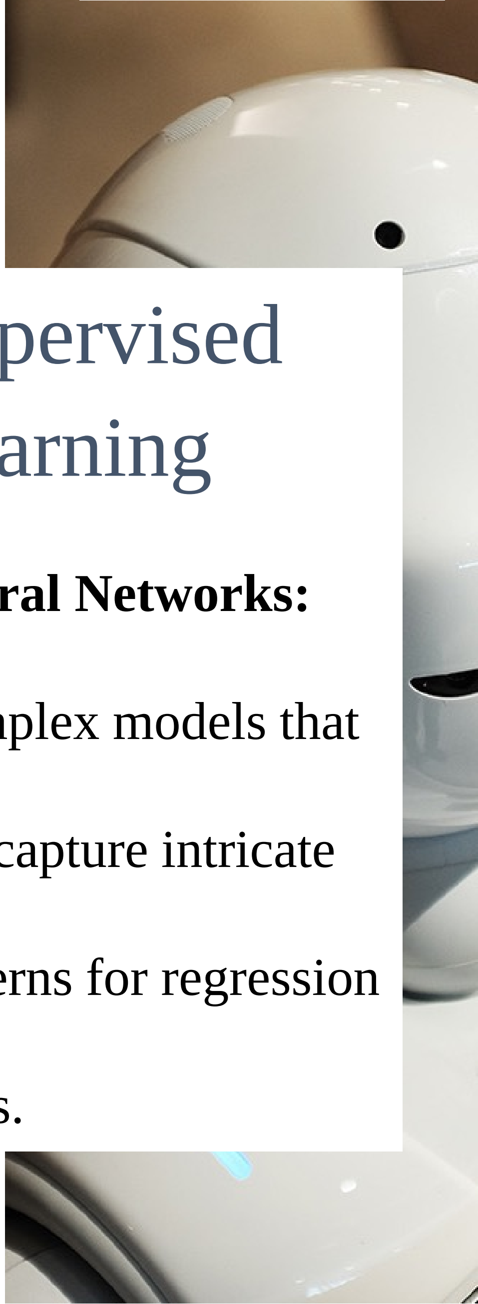
Supervised Learning

Neural Networks:

Complex models that can capture intricate patterns for regression tasks.

Neural Networks for Regression

- Goal
  - Capture intricate patterns and non-linear relationships in data
- Method
  - Layers of interconnected neurons process inputs through weights
  - Can use multiple layers (Deep Neural Networks) for complex tasks
- Key Components
  - Input layer, hidden layers, output layer
  - Activation functions (e.g., ReLU, Sigmoid)
- Pros
  - Highly flexible, can model complex and non-linear relationships
  - Works well with large datasets
- Cons
  - Requires significant computational resources
  - Prone to overfitting without proper regularization
  - Less interpretable compared to simpler models
- Use Cases
  - Predicting complex outcomes like financial trends
  - Forecasting in time series with complex seasonality
  - Image and speech-based regression tasks



Supervised Learning

Real-World Examples:

- **House Price Prediction:** Estimating the price of a house based on features like size, location, and number of rooms.
- **Weather Forecasting:** Predicting temperature or rainfall for the coming days.
- **Stock Price Prediction:** Forecasting the future price of stocks based on historical data.

Supervised Learning

Process:

- **Training:** The model learns relationships from input features and continuous output values.
- **Prediction:** The model uses learned relationships to predict continuous values for new data.
- **Evaluation:** Performance is measured using metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE).

Regression is crucial for tasks that require precise value predictions, enabling data-driven decision-making in finance, engineering, healthcare, and beyond.

Supervised Learning

Process:

- **Training:** The model learns relationships from input features and continuous output values.
- **Prediction:** The model uses learned relationships to predict continuous values for new data.
- **Evaluation:** Performance is measured using metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE).

Regression is crucial for tasks that require precise value predictions, enabling data-driven decision-making in finance, engineering, healthcare, and beyond.

Supervised Learning: Regression

Project Problem: Predict house prices based on features like size, location, rooms, and age.

Objective:

- **Goal:** Build a regression model to estimate house prices.
- **Skills Practiced:** Data preprocessing, feature engineering, model training, and evaluation.

Instructions:

1. **Collect Data:** Use a house price dataset with relevant features.
2. **Preprocess:** Clean data, handle missing values, scale features, and encode categorical variables.
3. **Explore Data:** Analyze relationships between features and price.
4. **Split Data:** Separate data into training and test sets.
5. **Train Model:** Use regression models like Linear Regression or Random Forest.
6. **Evaluate Model:** Use metrics like MAE and R^2 to assess accuracy.
7. **Save and Deploy:** Save the model and create a simple interface for predictions.

Supervised Learning: Regression

Steps to Create and Set Up a Virtual Environment in VS Code

Step 1: python -m venv VirtualEnv_Name

Step 2: To activate virtual environment

1. **Window:** VirtualEnv_Name\Scripts\activate
2. **MacOS or Linux:** source VirtualEnv_Name/bin/activate

Step 3: To ensure you have the latest version of pip, type:

python -m pip install --upgrade pip

Step 4: Installing all libraries:

1. pip install pandas
2. pip install scikit-learn
3. pip install joblib
4. pip install matplotlib
5. pip install seaborn

Step 5. Confirming Installed Libraries:

pip list

Supervised Learning: Regression

Example: train_model.py

```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error
import joblib
import os

# Load data or create a new file if not present
def load_or_create_data():
    if os.path.exists('house_data.csv'):
        print("Loading existing data from 'house_data.csv'...")
        data = pd.read_csv('house_data.csv')
    else:
        print("No data found. Creating 'house_data.csv'...")
        data = pd.DataFrame(columns=['size', 'location', 'rooms', 'age', 'price'])
    return data

```

Supervised Learning

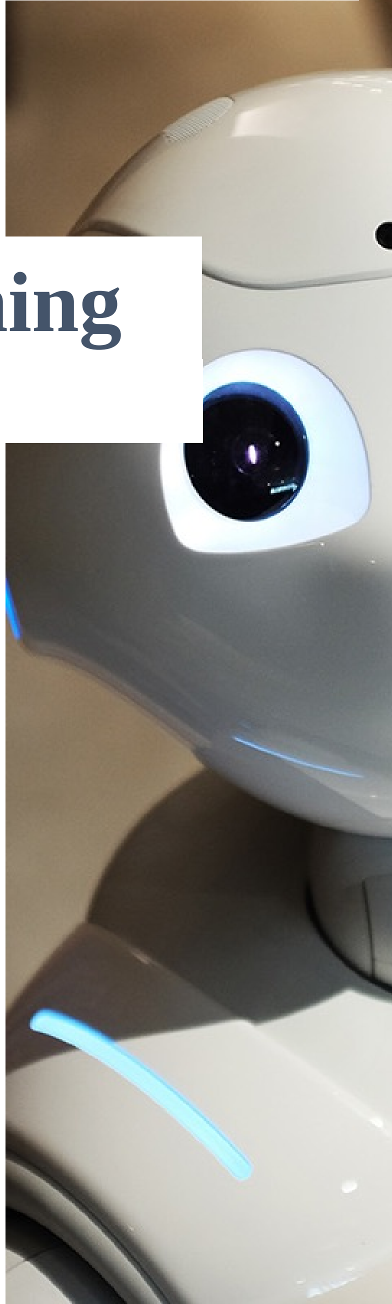
Example: train_model.py

```python
# Collect new data from the admin
def collect_data(data):
    while True:
        print("\nEnter new house data:")
        size = float(input("House size (sq ft): "))
        location = int(input("Location score (1-10): "))
        rooms = int(input("Number of rooms: "))
        age = int(input("House age (years): "))
        price = float(input("House price ($): "))

        # Create a new DataFrame for the single row of data
        new_data = pd.DataFrame({
            'size': [size],
            'location': [location],
            'rooms': [rooms],
            'age': [age],
            'price': [price]
        })

        # Concatenate the new data row with the existing DataFrame
        data = pd.concat([data, new_data], ignore_index=True)

        # Ask if the admin wants to add more data
        add_more = input("Add more data? (yes/no): ").strip().lower()
        if add_more != 'yes':
            break

    # Save the updated data to CSV
    data.to_csv('house_data.csv', index=False)
    print("'house_data.csv' updated with new data.")
    return data
```

Supervised Learning

Example: train_model.py

```python
# Load or create a model function
def load_or_create_model():
    if os.path.exists('house_price_model.joblib'):
        print("Loading existing model...")
        model = joblib.load('house_price_model.joblib')
    else:
        print("No existing model found. Creating a new model...")
        model = RandomForestRegressor()
    return model

def train_model(data):
    # Define features and target variable
    X = data[['size', 'location', 'rooms', 'age']]
    y = data['price']

    # Split data into training and testing sets
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

    # Load or initialize the model
    model = load_or_create_model()

    # Train the model
    print("\nTraining the model...")
    model.fit(X_train, y_train)

    # Evaluate model on test set
    y_pred = model.predict(X_test)
    mae = mean_absolute_error(y_test, y_pred)
    print(f"Training complete. Mean Absolute Error on Test Set: {mae:.2f}")

    # Save the model
    joblib.dump(model, 'house_price_model.joblib')
    print("Model saved as 'house_price_model.joblib'.")
```



Supervised Learning

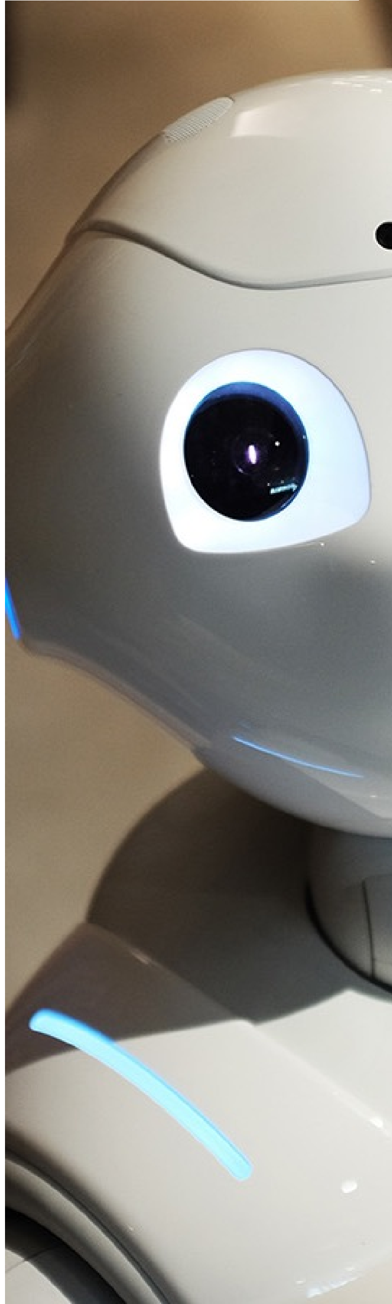
Example: train_model.py

```python

def main():
    # Ask admin if they want to input new data or train existing data
    choice = input("Do you want to input new data before training? (yes/no): ").strip().lower(

    # Load or create data
    data = load_or_create_data()

    if choice == 'yes':
        # Collect new data from admin and update 'house_data.csv'
        data = collect_data(data)

    # Train or retrain the model with all data
    train_model(data)

if __name__ == "__main__":
    main()

```

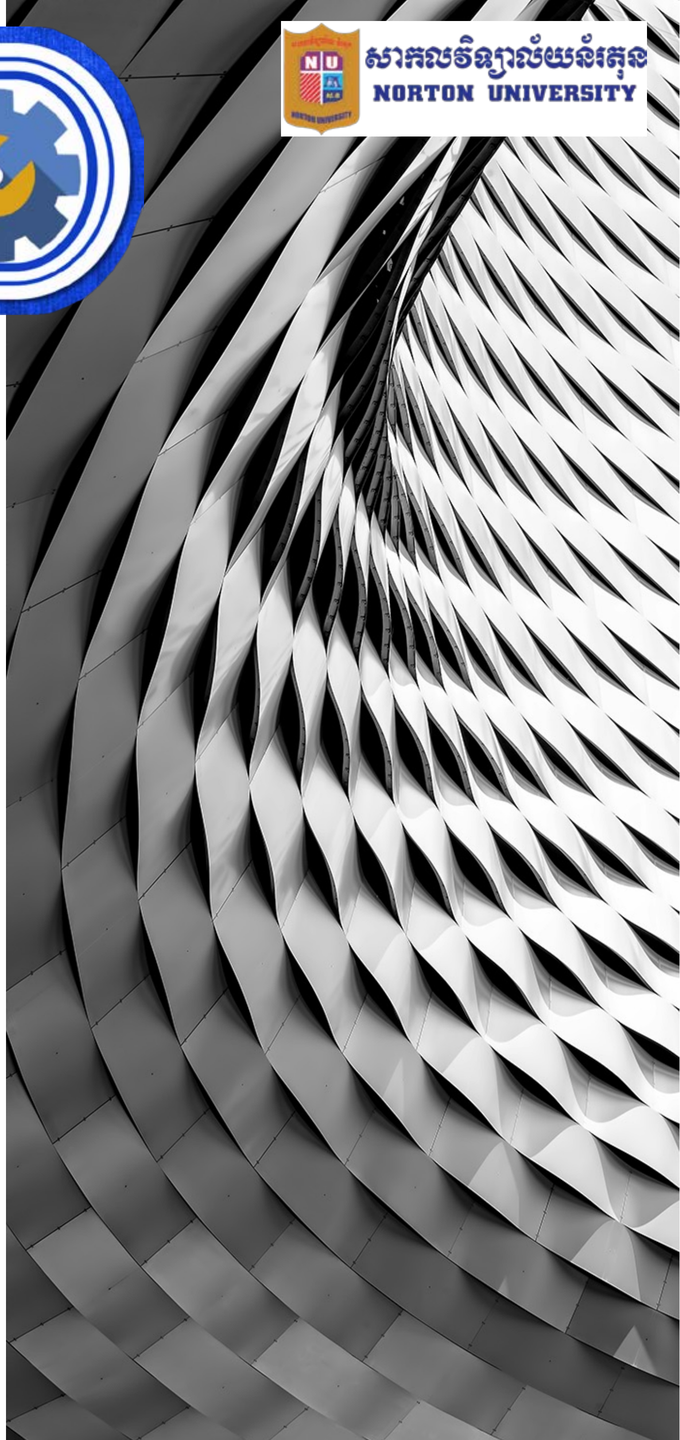
Supervised Learning

Example: predict_price.py

```python
import joblib

# Load the trained model
model = joblib.load('house_price_model.joblib')

def get_user_input():
    """Collect user input for house features."""
    print("Please enter the following details about the house:")
    size = float(input("Enter the house size (sq ft): "))
    # Example: 1-10 scale for location desirability
    location = int(input("Enter location score (1-10): "))
    rooms = int(input("Enter number of rooms: "))
    age = int(input("Enter the age of the house (years): "))
    return [[size, location, rooms, age]]

def main():
    print("House Price Prediction System")
    user_input = get_user_input()
    predicted_price = model.predict(user_input)
    print(f"Estimated House Price: ${predicted_price[0]:,.2f}")

if __name__ == "__main__":
    main()
```



Homework:



Answer Questions below:

1. What is the difference between classification and regression in supervised learning, and can you give an example of each?

2. Explain how the train-test split method helps in evaluating a supervised learning model. Why is it important to test a model on data it hasn't seen during training?

សាកលវិទ្យាល័យន័រតុន
NORTON UNIVERSITY

Thank you