Machine Learning Methods

Yagya Raj Pandeya, PhD

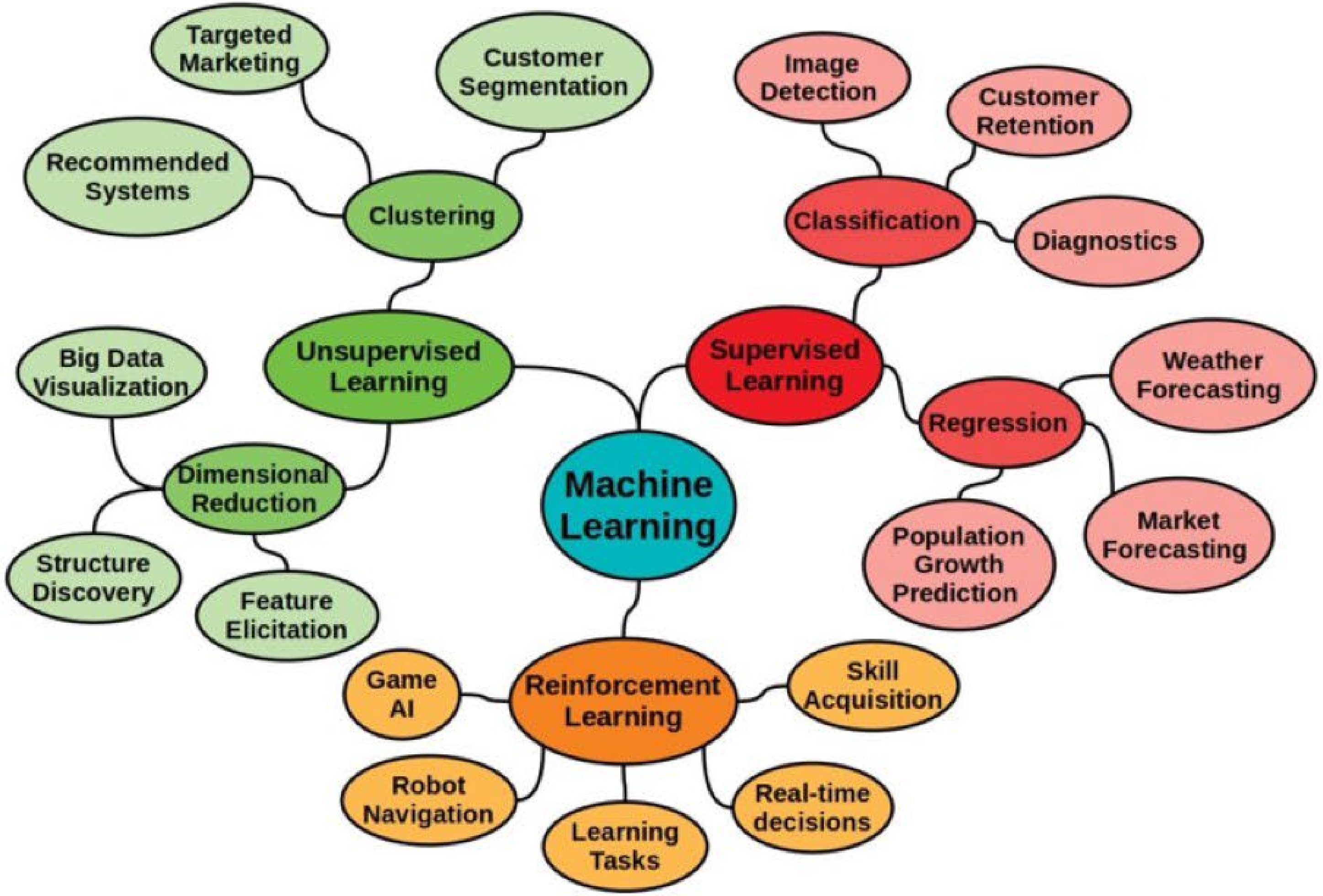
Asst. Professor and AI Program Coordinator
DoSCE, SOE, Kathmandu University, Kavre

Founder and CEO, Guru Technology Pvt. Ltd., Kathmandu

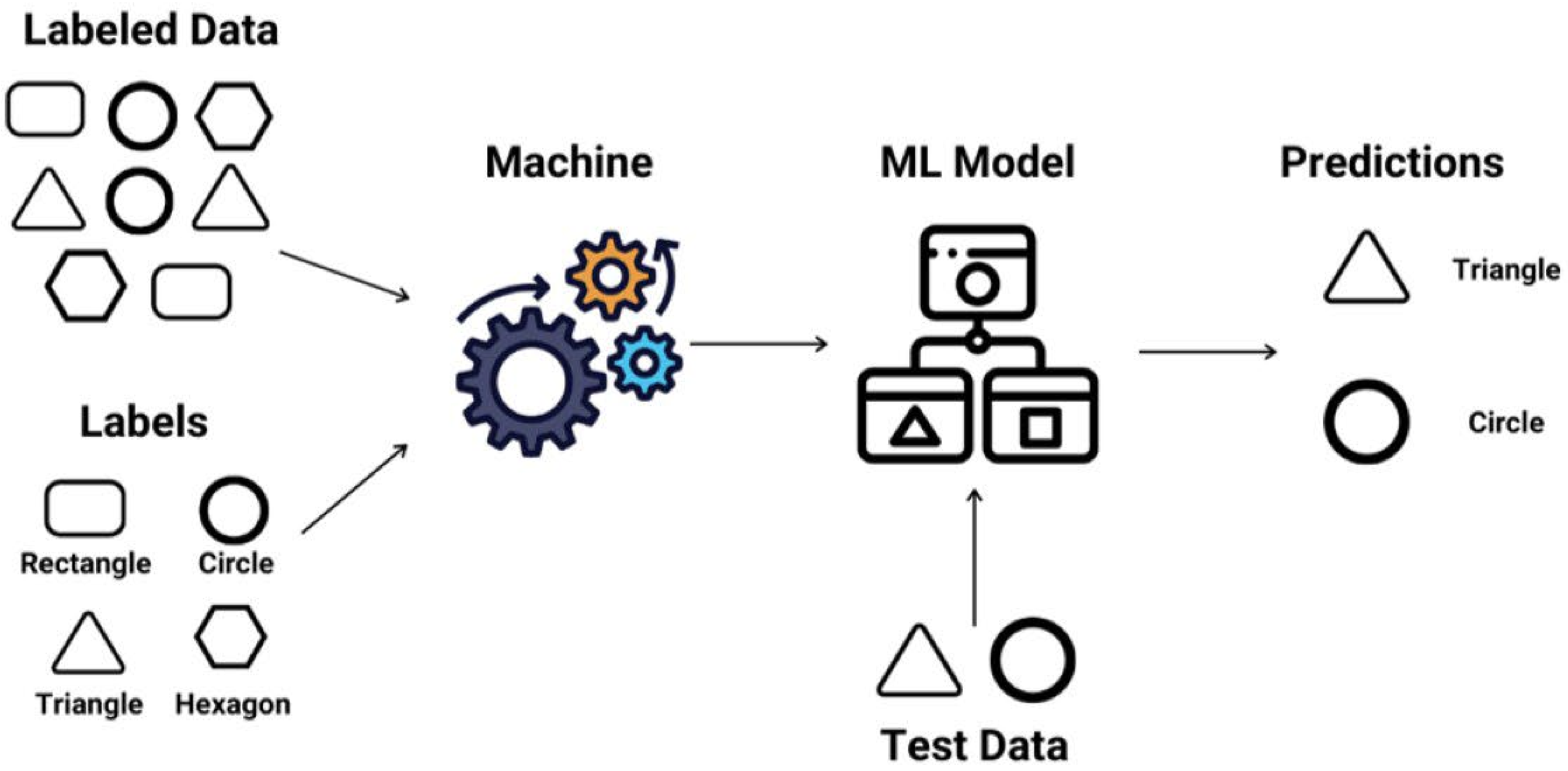
2024-06-11

ML Approaches

- Supervised Learning
- Unsupervised Learning
- Semi-Supervised Learning
- Weakly-supervised Learning
- Reinforcement Learning
- Transfer Learning
- Online Learning

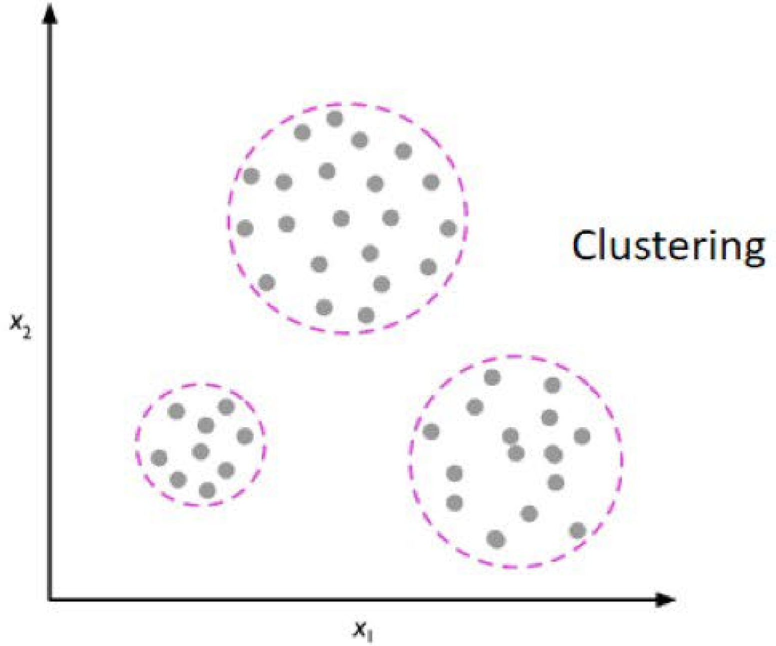


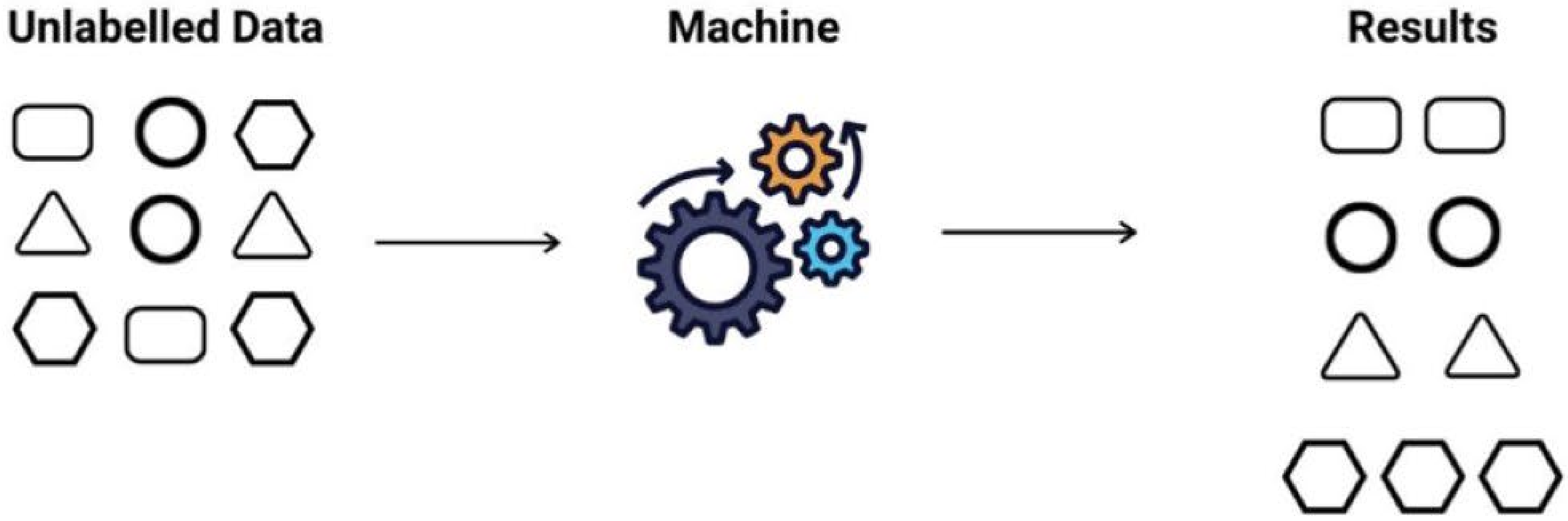
Supervised Learning



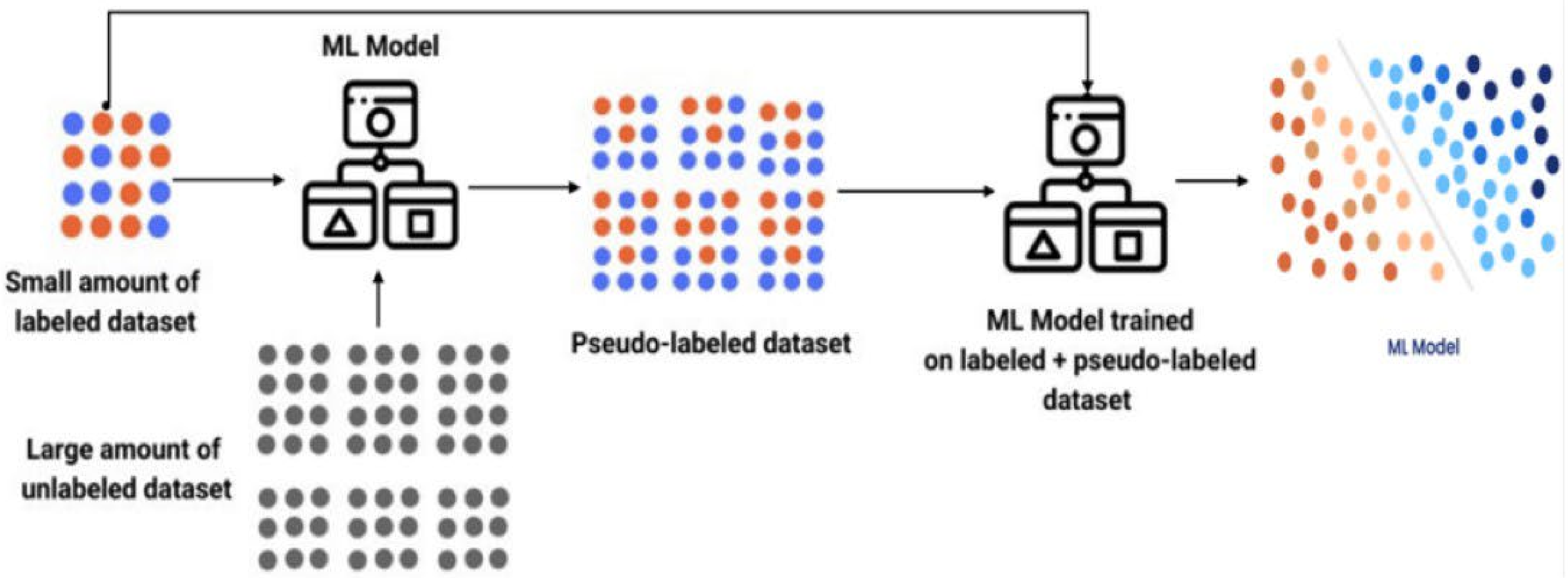
https://www.enjoyalgorithms.com/blogs/supervised-unsupervised-and-semisupervised-learning

Unsupervised Learning

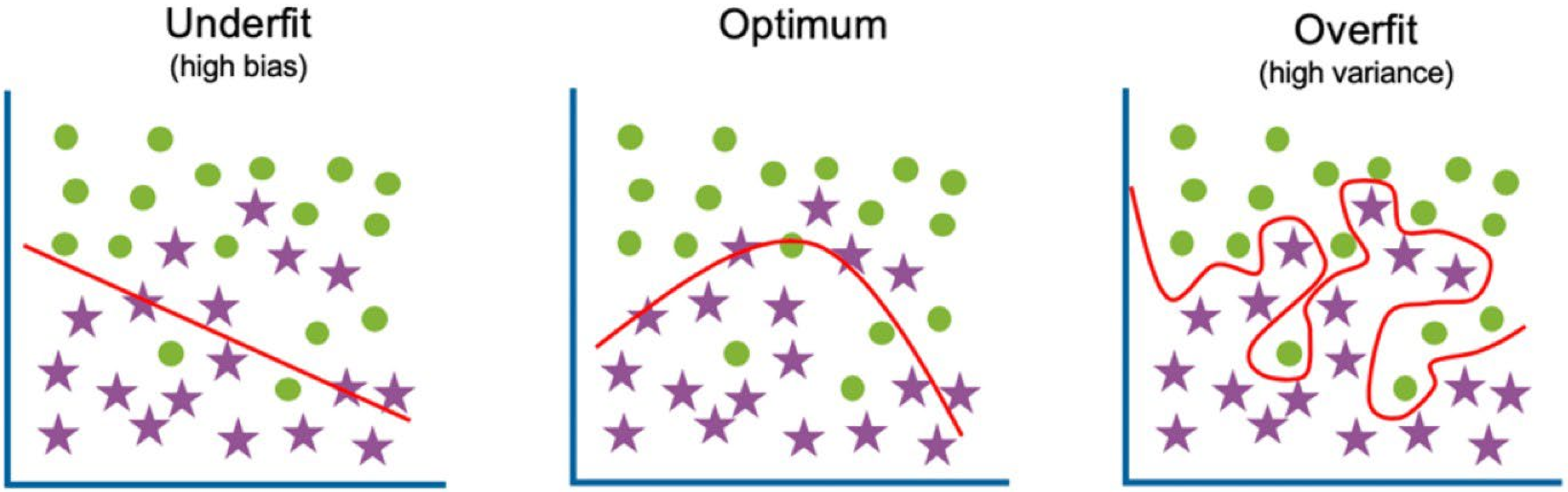




Semi-supervised learning



Underfit, optimal and overfit model

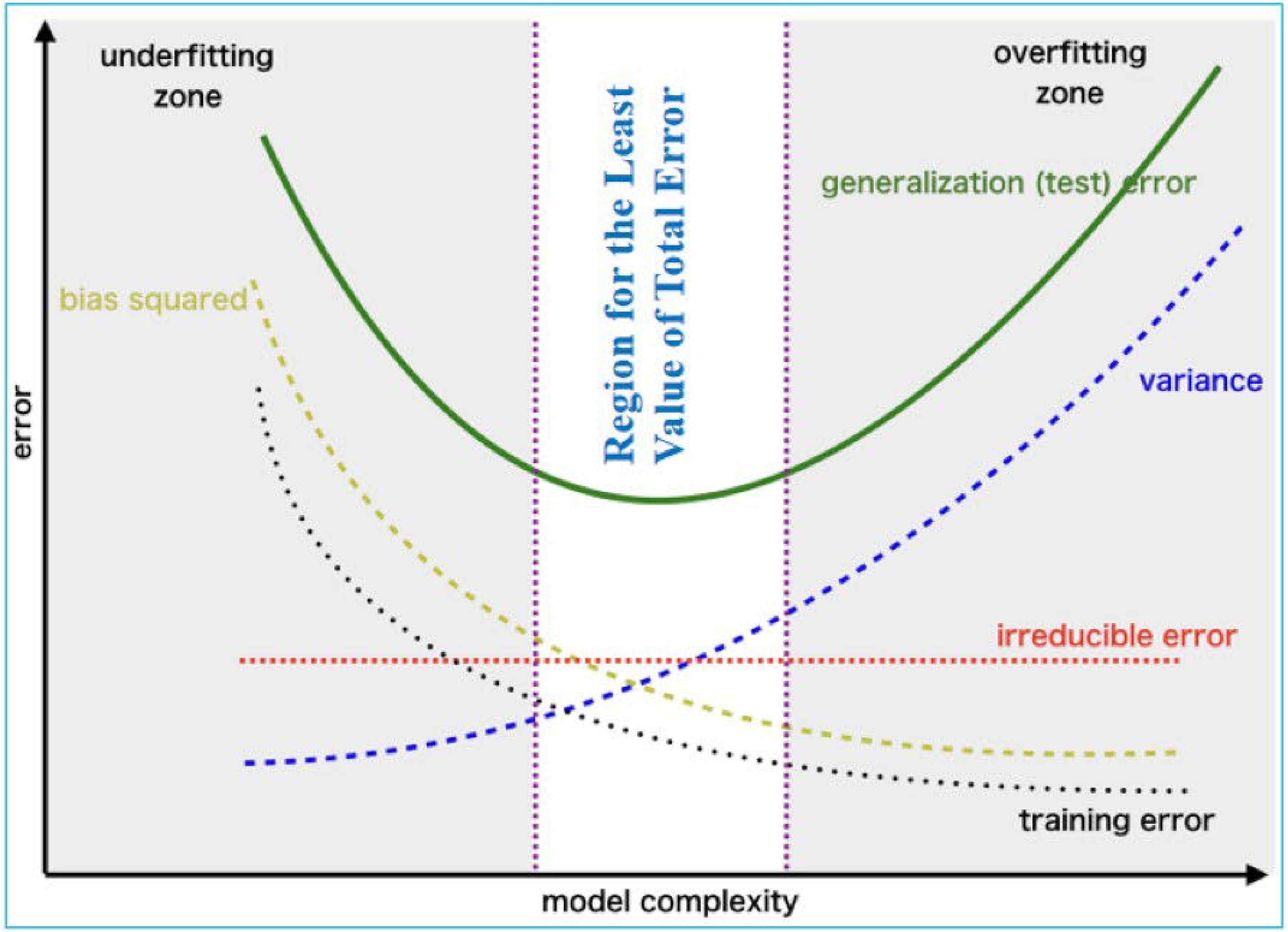


High training error
High test error

Low training error
Low test error

Low training error
High test error

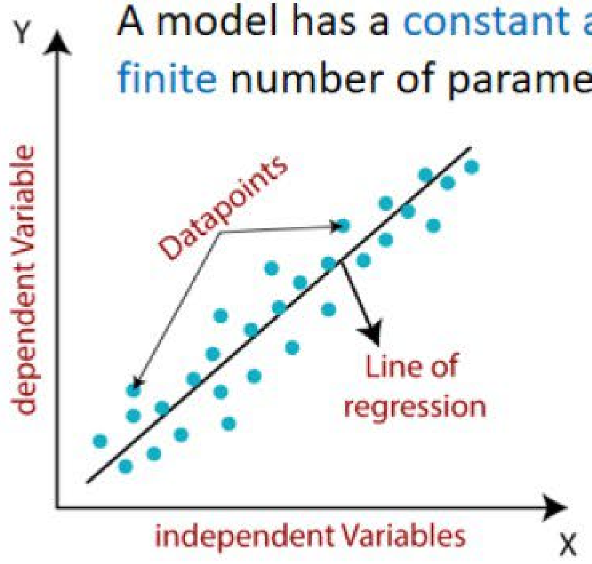
Balancing model in bias-variance trade-off



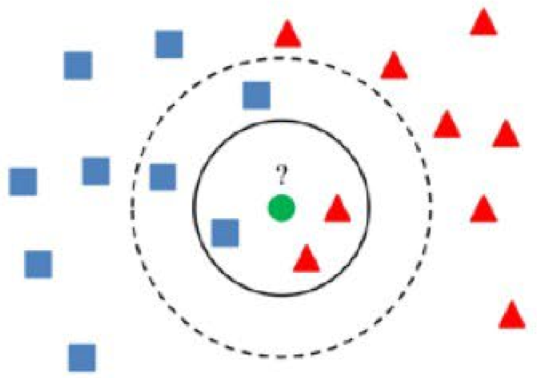
ML Algorithms

Parametric ML algorithms	Non-parametric ML Algorithms
• Linear Regression • Logistic Regression • Linear Discriminant Analysis • Ridge Regression and Lasso Regression • Gaussian Mixture Models (GMM) • Bayesian Networks • Linear Support Vector Machine • Perceptron	• Naive Bayes • Decision Trees • Random Forest • K-Nearest Neighbor (k-NN) • Support Vector Machines with Gaussian Kernels • Gradient Boosting Machines (GBM) • Gaussian Processes • Kernel Density Estimation (KDE) • Artificial Neural Networks

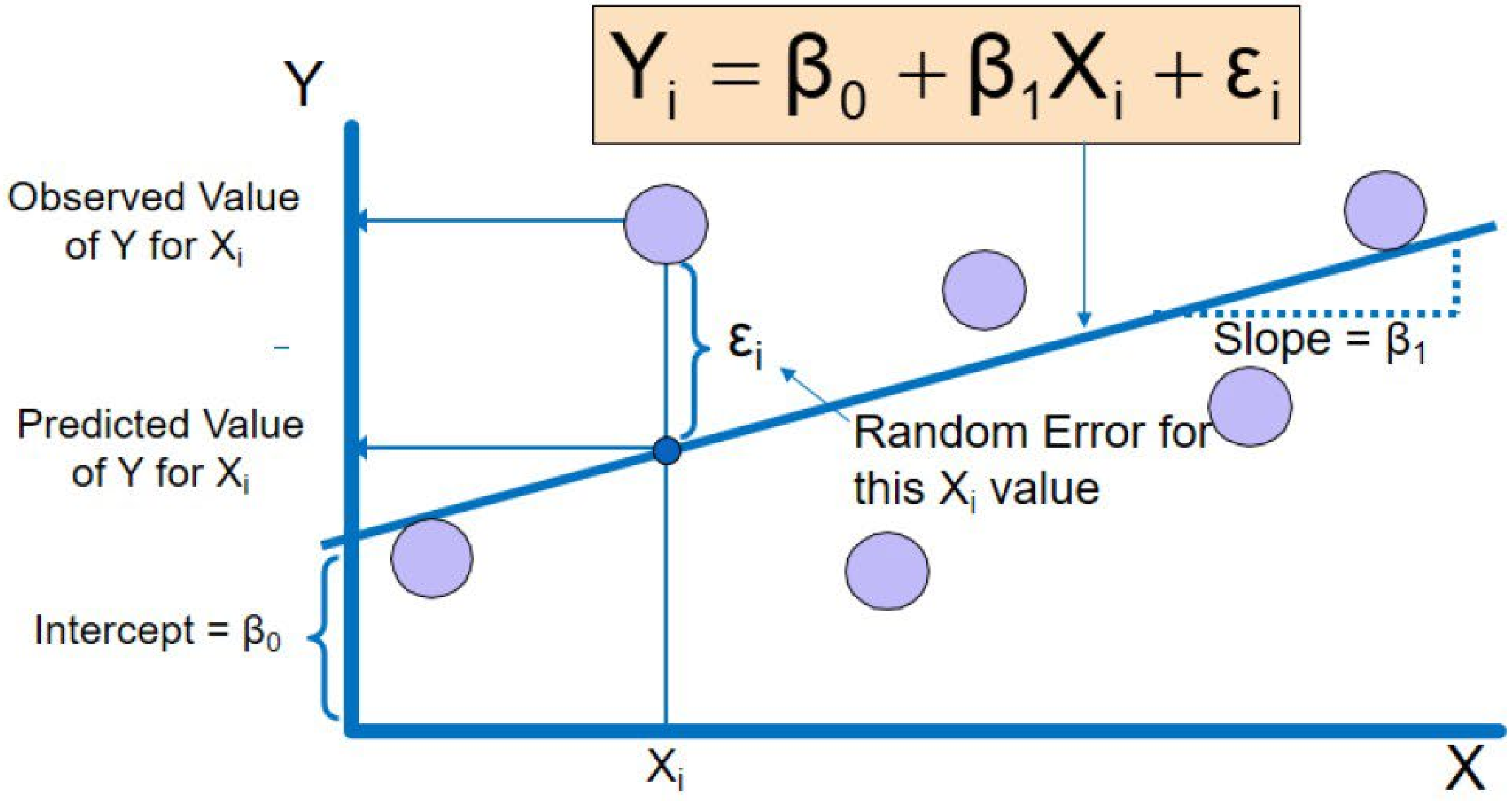
A model has a constant and finite number of parameters.



Use the data itself to determine the complexity and number of parameters required for modeling.



Simple Linear Regression



Evaluations

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$$

$$R^2 = 1 - \frac{SSR}{SST}$$

R^2 measure the impact of independent variable to the dependent variable.

SSR (Sum of Squared Residuals)

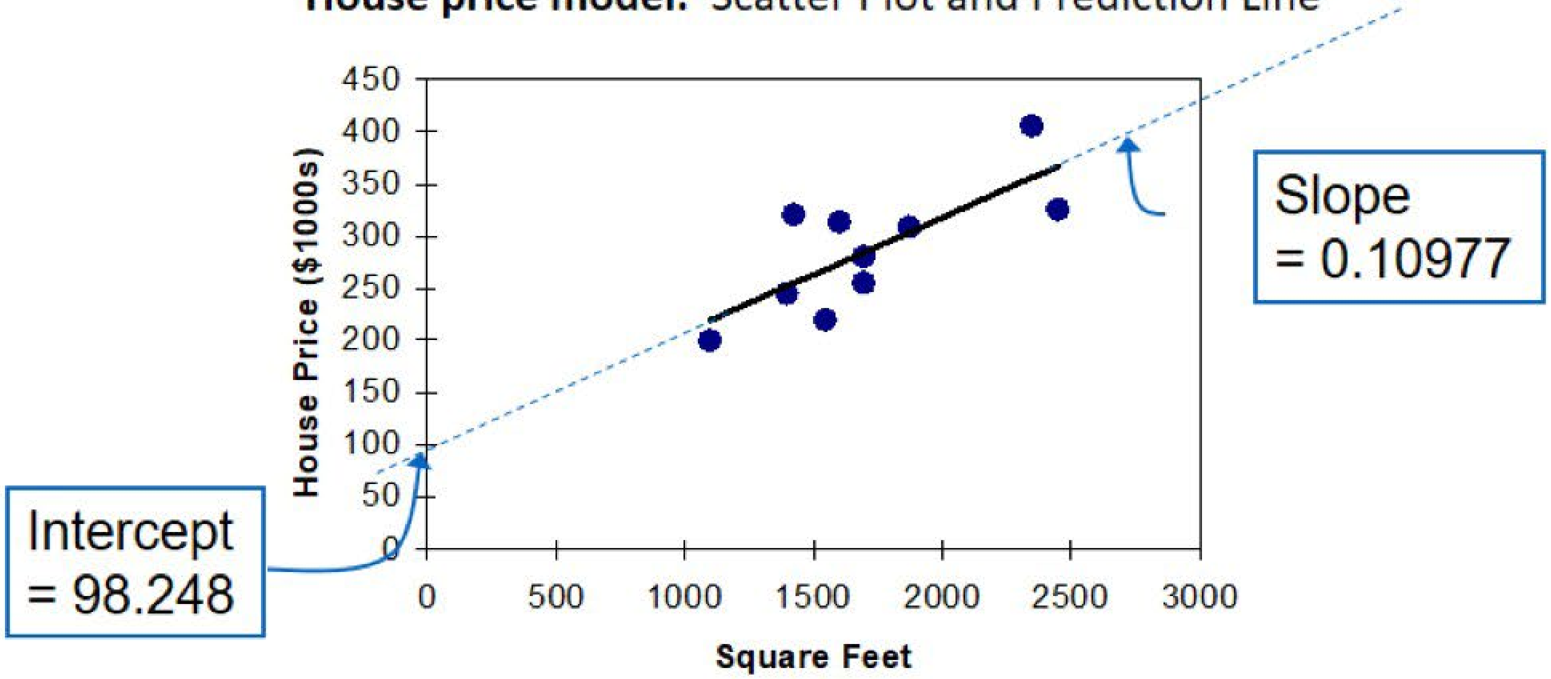
SST (Total Sum of Squares)

R^2=1 indicates that the model perfectly explains all the variability in the dependent variable, and it is an excellent fit to the data.

Simple Linear Regression Example

House Price in $1000s (Y)	Square Feet (X)
245	1400
312	1600
279	1700
308	1875
199	1100
219	1550
405	2350
324	2450
319	1425
255	1700

House price model: Scatter Plot and Prediction Line



$$\widehat{\text{house price}} = 98.24833 + 0.10977 \text{ (square feet)}$$

$$\text{house price} = 98.25 + 0.1098 \text{ (sq.ft.)}$$

$$= 98.25 + 0.1098(2000)$$

New area

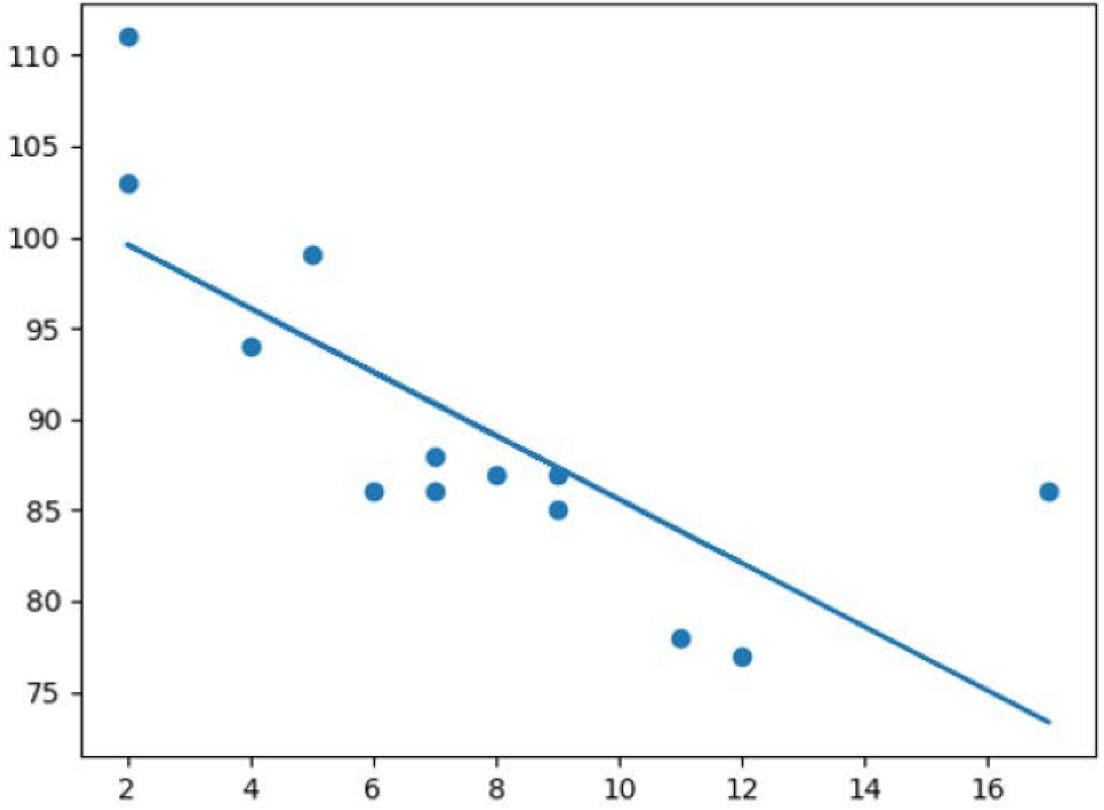
$$= 317.85$$

Simple Linear Regression Example (Python)

```
import matplotlib.pyplot as plt
from scipy import stats
 x = [5,7,8,7,2,17,2,9,4,11,12,9,6]
y = [99,86,87,88,111,86,103,87,94,78,77,85,86]
 slope, intercept, r, p, std_err = stats.linregress(x, y)


def myfunc(x):
   return slope * x + intercept


mymodel = list(map(myfunc, x))
 plt.scatter(x, y)
plt.plot(x, mymodel)
plt.show()
```



Polynomial Regression

The relationship between the dependent variable (Y) and only one independent variable (X) is represented by a polynomial equation:

- $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 * X^3 + \varepsilon$

where:

- Y is the dependent variable (target),
- X is the independent variable (feature),
- $\beta_0, \beta_1, \beta_2, \ldots, \beta_n$ are the coefficients to be determined, and
- n is the degree of the polynomial.

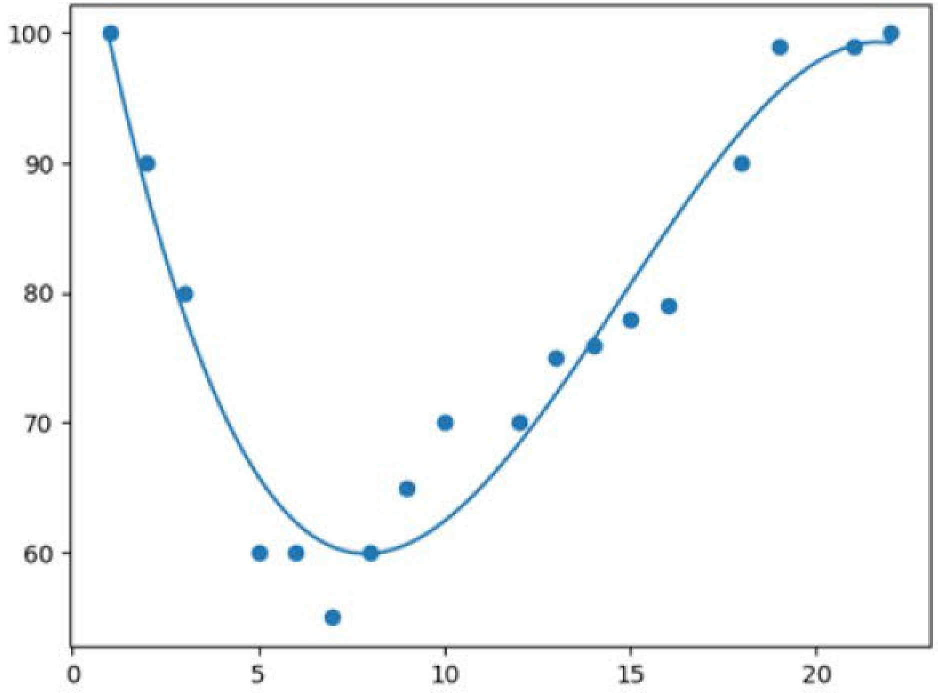
```
import numpy
import matplotlib.pyplot as plt

x = [1,2,3,5,6,7,8,9,10,12,13,14,15,16,18,19,21,22]
y = [100,90,80,60,60,55,60,65,70,70,75,76,78,79,90,99,99,100]

mymodel = numpy.poly1d(numpy.polyfit(x, y, 3))

myline = numpy.linspace(1, 22, 100)
 plt.scatter(x, y)
plt.plot(myline, mymodel(myline))
plt.show()
```

Degree of polynomial



Multiple Linear Regression

- **Simple linear regression:** models the relationship between a dependent variable and *a single independent variable*.
- **Multiple linear regression:** model the relationship between a dependent variable and two or more independent variables.

Dependent Variable (Response Variable)

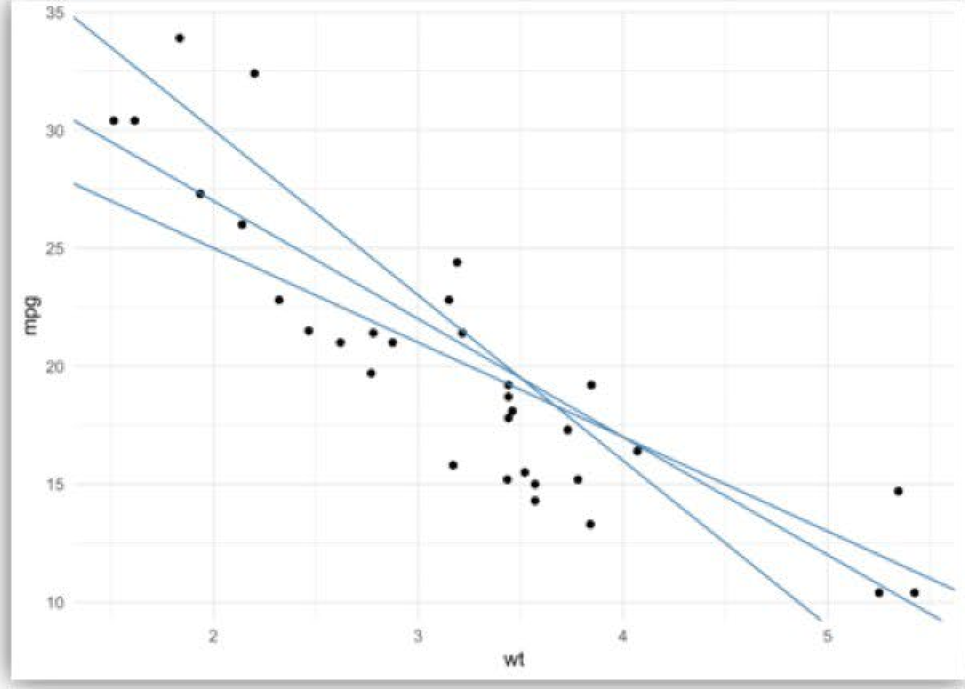
Independent Variables (Predictors)

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \varepsilon$$

Y intercept

Slope Coefficient

Error Term



Multiple Linear Regression: Example (Python)

```python
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# Generate sample data (replace with your dataset)
data = {
    'SquareFootage': [1500, 2000, 1200, 1800, 1350, 1650, 1550],
    'Bedrooms': [3, 4, 2, 3, 2, 3, 3],
    'AgeOfHouse': [5, 10, 8, 3, 15, 6, 2],
    'Price': [250000, 320000, 200000, 290000, 210000, 260000, 255000]
}

# Create a DataFrame from the data
df = pd.DataFrame(data)

# Select independent variables (features) and dependent variable (target)
X = df[['SquareFootage', 'Bedrooms', 'AgeOfHouse']]
y = df['Price']

# Split the 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)
```

```python
# Create and fit the linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Make predictions on the test data
y_pred = model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error: {mse}")
print(f"R-squared (R2) Score: {r2}")

# Coefficients and intercept of the linear
regression model
coefficients = model.coef_
intercept = model.intercept_

print("Coefficients:", coefficients)
print("Intercept:", intercept)
```

THANK YOU

CONTACT INFO

yagyapandeya@gmail.com
yagy.apandeya@ku.edu.np

+977-9848-542617

https://yagyapandeya.github.io/