SUPERVISED MACHINE LEARNING: REGRESSION

MODULE1:

INTRODUCTION TO SUPERVISED MACHINE LEARNING AND LINEAR REGRESSION

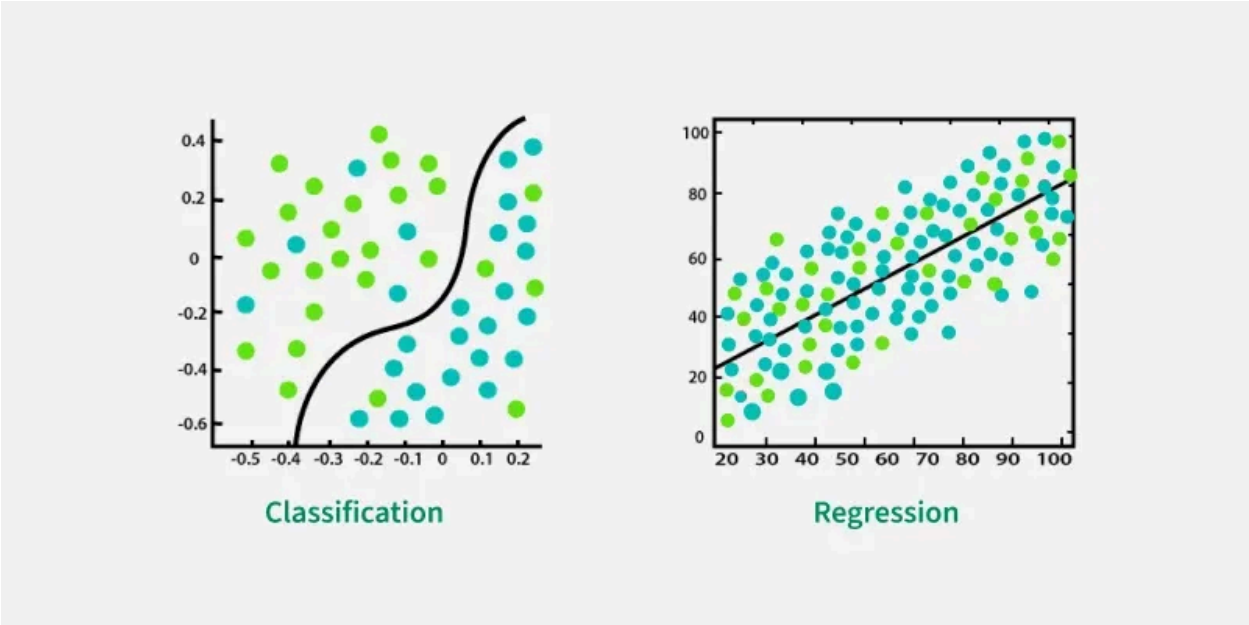
TABLE OF CONTENTS

Introduction to Supervised Machine Learning.....2
Models and Parameters.....2
Training a Model.....3
Trade-off: Interpretation vs Prediction..... 4
Linear Regression..... 4
Implementation in Python (scikit-learn)..... 5
Data Transformations..... 6
Key Takeaways.....8

Introduction to Supervised Machine Learning

- **Machine Learning (ML)** enables computers to learn from data instead of being explicitly programmed.
- Unlike traditional statistical models (where we often know the data-generating process), ML deals with complex or unknown processes.
- ML is a subset of **Artificial Intelligence (AI)**, focusing on **learning ability**.

Two main types of Supervised Learning:



- **Regression**: predicts a **continuous variable** (e.g., house prices, movie revenue).
- **Classification**: predicts a **categorical variable** (e.g., churn, spam detection, face recognition).

Models and Parameters

- **Model**: a simplified representation of reality that captures important relationships.

Concept	Definition	Example
Variable	A symbol or placeholder representing data values.	x_1, x_2, y
Coefficient	A numeric value that represents the weight/strength of a feature's impact on the target.	β1,β2
Argument	A concrete value passed into a function or model.	`f(3)` → 3 is argument
Parameter	Quantities the model learns from data, define how the model maps inputs to outputs.	coefficients β
Hyperparameter	Settings chosen before training, which control the model's structure or learning process.	`learning_rate=0.01, epochs=50`

Training a Model

1. Collect labeled data (X,y)(X, y)(X,y).
2. Split into **training** and **testing** sets to prevent overfitting.
3. Train the model by finding parameters that minimize the **loss function**

General formula:

$$y_p = f(\Omega, x)$$

where Ω are learned parameters.

- **Loss function**: ***J(y, ŷ)*** measures the error between predictions and actual values.
- Example (linear regression):

$$J(\beta_0, \beta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(\left(\beta_0 + \beta_1 x_{obs}^{(i)}\right) - y_{obs}^{(i)}\right)^2$$

Trade-off: Interpretation vs Prediction

- **Interpretation**: explain model + feature impact (sales drivers, safety, marketing).
- **Prediction**: maximize accuracy, accept black-box models (churn, defaults, purchases).
- **Trade-off**: balance depends on business goals; both can complement each other.
- **Examples**:
    - Housing prices → interpret feature importance vs predict prices.
    - Customer churn → understand reasons vs predict who leaves.

Linear Regression

- Model:

$$y_\beta(x) = \beta_0 + \beta_1 x$$

- Sum of Square Error (SSE):

$$\sum_{i=1}^{m} \left(y_\beta(x^{(i)}) - y_{obs}^{(i)} \right)^2$$

- Total Sum of Squares (TSS):

$$\sum_{i=1}^{m} \left(\overline{y_{obs}} - y_{obs}^{(i)} \right)^2$$

- Coefficient of Determination (R2):

$$1 - \frac{SSE}{TSS}$$

Implementation in Python (scikit-learn)

- Import libraries

```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score
```

- Split train and test set

```
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=42)
```

- Train model

```
lr = LinearRegression()
lr.fit(X_train, y_train)
```

- Predict

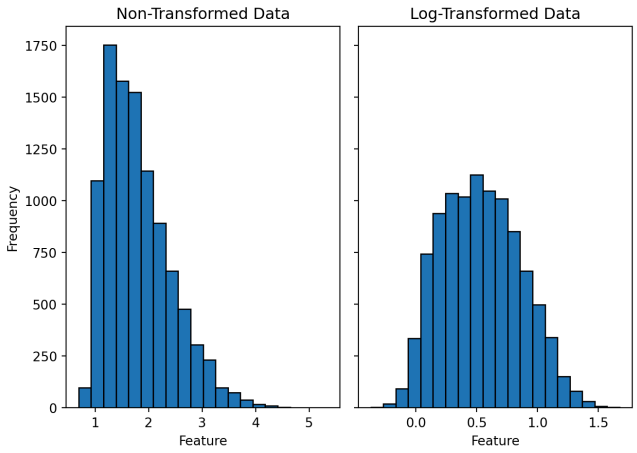
```
y_pred = lr.predict(X_test)
```

- Evaluate

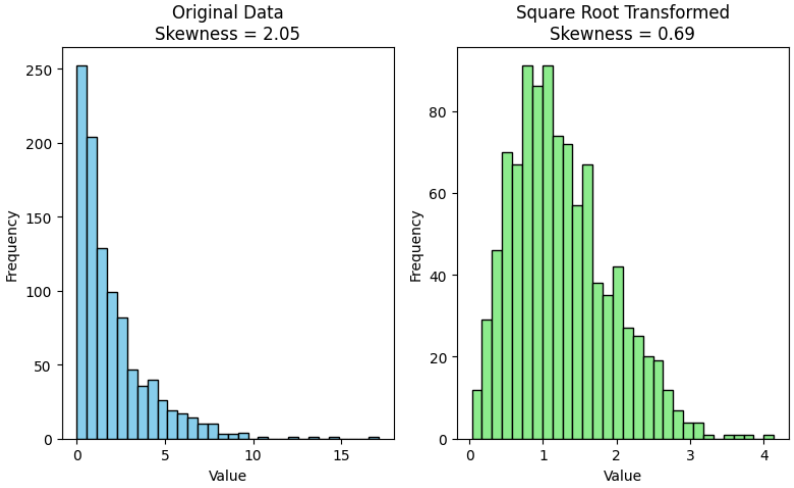
```
print("R2:", r2_score(y_test, y_pred))
```

Data Transformations

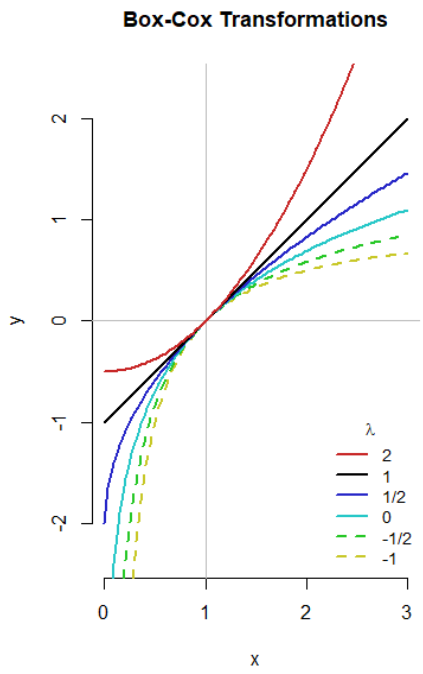
- To improve target distribution, common transformations include:
  - **Log transform: $y' = \log(y+1)$**



  - **Square root transform: $y' = \sqrt{(y+c)}$**



- **Box-Cox transform** (chooses optimal λ automatically)



- Example with Box-Cox:
  - Import libraries

```
from scipy.stats import boxcox
from scipy.special import inv_boxcox
```

  - Transform target variable

```
y_train_bc, lam = boxcox(y_train)
```

  - Inverse transform predictions

```
y_pred = inv_boxcox(y_pred_bc, lam)
```

Key Takeaways

- **Machine Learning**: a subset of AI that learns from data.
- **Supervised Learning**:
  - **Regression** → predicts numbers (continuous).
  - **Classification** → predicts categories.
- **Concept:**
  - Variable
  - Coefficient
  - Argument
  - Parameter
  - Hyperparameter
- **Train/Test Split**: needed to avoid overfitting.
- **Interpretation vs Prediction**: trade-off between explainability and accuracy.
- **Linear Regression**: basic supervised model, minimizes MSE.
- **R^2 Score**: measures how much variance the model explains.
- **Transformations**: log, sqrt, Box-Cox.
- **Python (scikit-learn)**: simple tools for training, predicting, evaluating.