

# Regression Model Notes (Python)

## 1. Introduction

Regression is a supervised learning technique used to predict a continuous target variable.

Common types:

- Linear Regression (simple & multiple)
- Polynomial Regression
- Regularized Regression (Ridge, Lasso, ElasticNet)

## 2. Workflow of a Regression Model

### 1. Import Libraries

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

### 2. Load Dataset

```
data = pd.read_csv("dataset.csv")  
  
X = data[['feature1','feature2']]  
  
y = data['target']
```

### 3. Split Data

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,  
random_state=42)
```

### 4. Train Model

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

### 5. Make Predictions

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

## 6. Evaluate Model

```
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print("MSE:", mse)
print("R2:", r2)
```

## 3. Key Concepts

- Coefficients: Show the impact of each feature on the target.
- Intercept: Baseline prediction when all features are zero.
- MSE (Mean Squared Error): Measures average squared difference between predicted and actual values.
- R2 Score: Proportion of variance explained by the model.

## 4. Extensions

- Polynomial Regression: Captures non-linear relationships.
- Regularization:
  - Ridge (L2 penalty) reduces coefficient magnitude.
  - Lasso (L1 penalty) performs feature selection.
- Cross-validation: Improves generalization.

## 5. Best Practices

- Always check assumptions (linearity, independence, homoscedasticity).
- Perform feature scaling if using regularization.
- Use visualizations (scatter plots, residual plots) to validate model fit.