In [ ]:

```

```

A Comparative Study of Ridge and Lasso Regularization Based on MSE Performance on Real Estate Data

Simple example to understand how L1 (Lasso) and L2 (Ridge) regularization work in practice.

Scenario: Predicting House Prices Imagine you have a dataset with the following features: Size (sqft) Bedrooms Age (years) Distance to city (km) Noise Level Price ($)

Let’s say you’re using linear regression to predict house prices based on these 4 features.

Simple example to understand how L1 (Lasso) and L2 (Ridge) regularization work in practice. Scenario: Predicting House Prices Imagine you have a dataset with the following features: Size (sqft) Bedrooms Age (years) Distance to city (km) Noise Level Price ($)

Let’s say you’re using linear regression to predict house prices based on these 4 features.

🔧 Step 1: The Problem Without Regularization A linear model would look like this:

Price

$w\,1 \cdot \text{Size} + w\,2 \cdot \text{Bedrooms} + w\,3 \cdot \text{Age} + w\,4 \cdot \text{Distance} + w\,5 \cdot \text{Noise} + b$ Price=w 1·Size+w 2·Bedrooms+w 3·Age+w 4·Distance+w 5·Noise+b When trained without regularization, the model might overfit — it memorizes the training data, including noise, and generalizes poorly to new data.

🧮 Step 2: L2 Regularization (Ridge) Let’s say Ridge regression learns: • w1 = 250 • w2 = 10000 • w3 = -500 • w4 = -3000 • w5 = -50 Here, none of the weights are zero — Ridge shrinks them but keeps all features. Even less important features like “Noise Level” are kept with small weights. ✅ Good when all features have some relevance.

✂️ Step 3: L1 Regularization (Lasso) Lasso regression might learn: • w1 = 300 • w2 = 0 • w3 = -600 • w4 = -3500 • w5 = 0 Here, some weights are exactly zero. Lasso eliminated "Bedrooms" and "Noise Level" — deciding they’re not useful for predicting price. ✅ Good when you want automatic feature selection.

💡 Intuition Recap • Ridge: "Keep all features, just shrink their impact." • Lasso: "Keep only the most important features; ignore the rest."

🧮 Step 2: L2 Regularization (Ridge) Let’s say Ridge regression learns: • w1 = 250 • w2 = 10000 • w3 = -500 • w4 = -3000 • w5 = -50 Here, none of the weights are zero — Ridge shrinks them but keeps all features. Even less important features like “Noise Level” are kept with small weights. ✅ Good when all features have some relevance.

✂️ Step 3: L1 Regularization (Lasso) Lasso regression might learn: • w1 = 300 • w2 = 0 • w3 = -600 • w4 = -3500 • w5 = 0 Here, some weights are exactly zero. Lasso eliminated "Bedrooms" and "Noise Level" — deciding they’re not useful for predicting price. ✅ Good when you want automatic feature selection.

💡 Intuition Recap • Ridge: "Keep all features, just shrink their impact." • Lasso: "Keep only the most important features; ignore the rest."

In [1]:
```python
import pandas as pd
```

In [6]:
```python
# Load the dataset
data = pd.read_csv("customer_satisfacation_vs_repeat_purchases.csv")
```

In [24]:
```python
data
```

Out[24]:

	Product_Quality	Customer_Service	Delivery_Speed	Ease_of_Use	Value_fo
0	7	4	8	5	
1	4	8	5	7	
2	8	5	8	7	
3	7	3	10	9	
4	3	7	3	9	
5	3	7	2	4	
6	1	5	1	1	
7	9	8	4	7	
8	10	10	4	2	
9	7	3	9	6	
10	8	9	7	10	
11	8	8	8	7	
12	10	9	10	10	
13	6	1	7	9	
14	2	8	8	5	
15	4	9	7	9	
16	9	9	10	10	
17	6	6	6	6	
18	7	10	8	8	
19	4	9	2	10	
20	5	6	2	4	
21	1	4	2	3	
22	9	10	10	10	
23	9	10	10	10	
24	6	8	6	9	
25	5	7	4	5	
26	8	6	8	9	
27	10	4	7	8	
28	9	2	9	3	
29	6	2	10	5	
30	6	7	8	10	
31	4	3	2	2	
32	9	10	7	10	

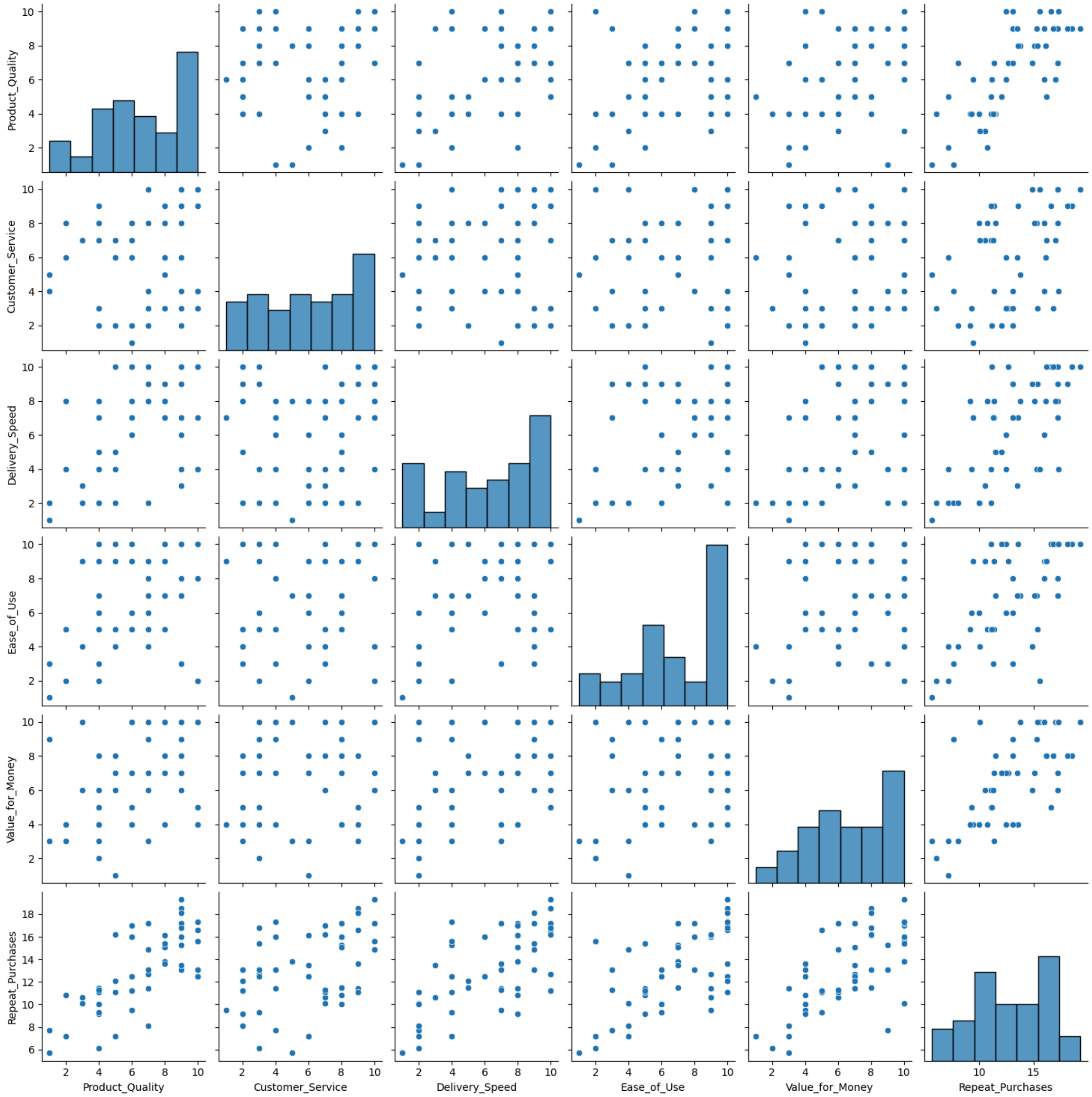
	Product_Quality	Customer_Service	Delivery_Speed	Ease_of_Use	Value_fo
33	9	9	9	10	
34	4	2	8	5	
35	9	4	6	8	
36	2	6	4	2	
37	9	6	3	7	
38	5	7	10	9	
39	4	8	2	6	
40	8	3	9	5	
41	10	3	4	10	
42	7	8	9	7	
43	7	2	2	4	
44	4	3	4	6	
45	9	3	10	10	
46	10	4	4	10	
47	4	7	7	3	
48	7	10	9	4	
49	5	2	5	10	

In [8]:
```
data.columns
```

Out[8]:
```
Index(['Product_Quality', 'Customer_Service', 'Delivery_Speed', 'Ease_of_Us
e',
       'Value_for_Money', 'Repeat_Purchases'],
      dtype='object')
```

In [9]:
```
import seaborn as sn
sn.pairplot(data)
```

Out[9]:
```
<seaborn.axisgrid.PairGrid at 0x786f64af1b50>
```



In [11]:
```python
# Features and target
X = data[['Customer_Service', 'Delivery_Speed', 'Ease_of_Use',
       'Value_for_Money', 'Repeat_Purchases']]
y = data["Product_Quality"]
```

In [12]:
```python
# Split the dataset
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, rar
```

In [13]:
```python
# Linear Regression (no regularization)
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(X_train, y_train)
```

Out[13]:

```
▾ LinearRegression
LinearRegression()
```

In [14]:

```
print("Linear Regression Coefficients:", lr.coef_)
```

```
Linear Regression Coefficients: [-0.51480574 -0.32930405 -0.20035858 -0.5734
8732  1.29952673]
```

In [15]:

```
from sklearn.metrics import mean_squared_error
y_pred = lr.predict(X_test)
mse_No_regulation = mean_squared_error(y_test, y_pred)
mse_No_regulation
```

Out[15]: 1.2194994717157444

In [16]:

```
# Linear Regression (L1 regularization)
from sklearn.linear_model import Lasso
lasso = Lasso(alpha=10000)

lasso.fit(X_train, y_train)
```

Out[16]:

```
▾ Lasso
Lasso(alpha=10000)
```

In [17]:

```
print("Lasso Coefficients:", lasso.coef_)
```

```
Lasso Coefficients: [0. 0. 0. 0. 0.]
```

In [18]:

```
from sklearn.metrics import mean_squared_error
y_pred = lasso.predict(X_test)
mse_L1_regulation = mean_squared_error(y_test, y_pred)
mse_L1_regulation
```

Out[18]: 5.119727891156462

In [19]:

```
# Linear Regression (L2 regularization)
from sklearn.linear_model import Ridge
ridge = Ridge(alpha=1.0)
ridge.fit(X_train, y_train)
```

Out[19]:

```
▾ Ridge
Ridge()
```

In [20]:

```
print("Lasso Coefficients:", ridge.coef_)
```

```
Lasso Coefficients: [-0.50487409 -0.3112752  -0.17493312 -0.54869774  1.2563
7873]
```

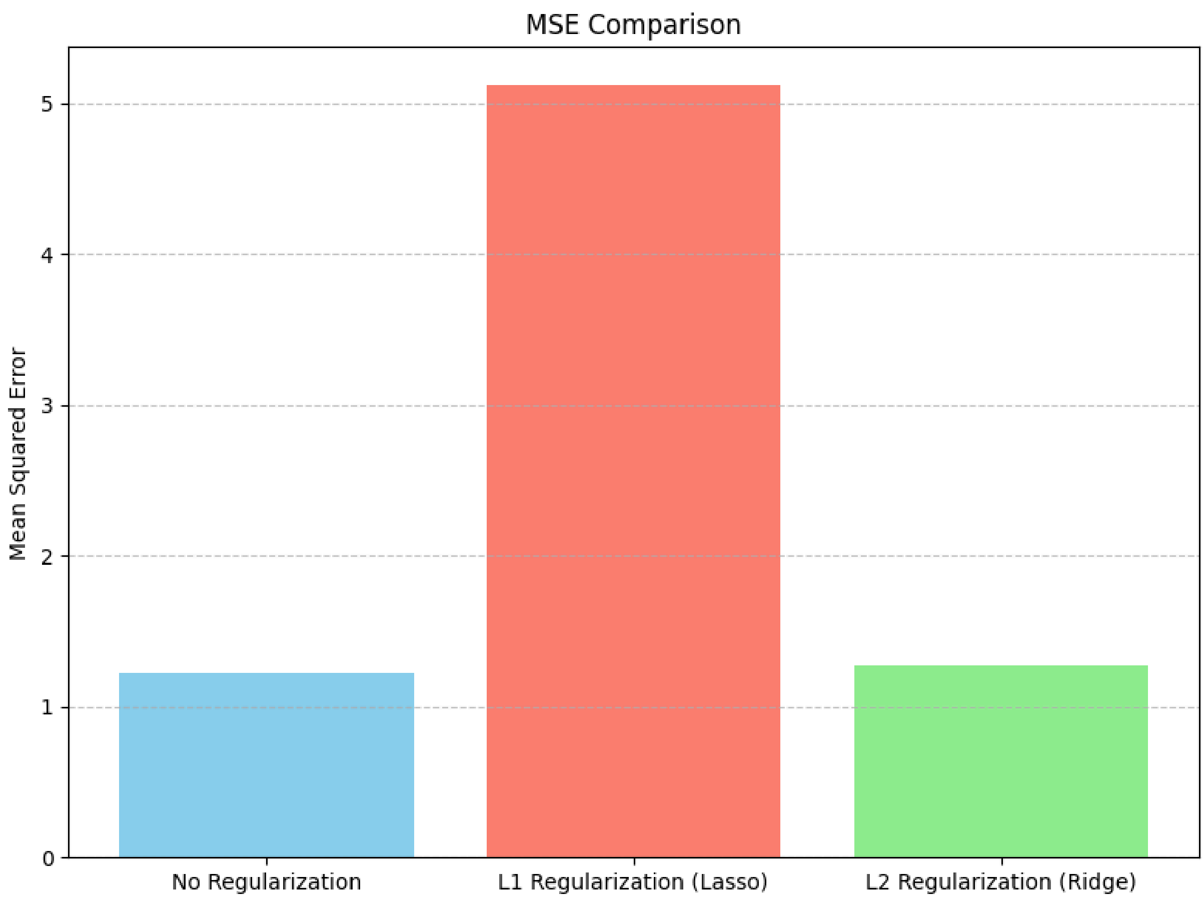
In [21]:
```python
from sklearn.metrics import mean_squared_error
y_pred = ridge.predict(X_test)
mse_L2_regulation = mean_squared_error(y_test, y_pred)
mse_L2_regulation
```

Out[21]: 1.2773266643700885

In [22]:
```python
import matplotlib.pyplot as plt

mse_values = {
    "No Regularization": mse_No_regulation,
    "L1 Regularization (Lasso)": mse_L1_regulation,
    "L2 Regularization (Ridge)": mse_L2_regulation
}

plt.figure(figsize=(8, 6))
plt.bar(mse_values.keys(), mse_values.values(), color=["skyblue", "salmon",
plt.title("MSE Comparison")
plt.ylabel("Mean Squared Error")
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```



In [23]:
```python
# no regulation and l2 is same because the data is cleaned
```