

```
In [33]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import svm
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, classification_report
```

```
In [34]: pip install seaborn
```

Requirement already satisfied: seaborn in c:\users\admin\.conda\envs\myenv\lib\site-packages (0.13.2)
Requirement already satisfied: numpy!=1.24.0,>=1.20 in c:\users\admin\.conda\envs\myenv\lib\site-packages (from seaborn) (2.3.1)
Requirement already satisfied: pandas>=1.2 in c:\users\admin\.conda\envs\myenv\lib\site-packages (from seaborn) (2.3.1)
Requirement already satisfied: matplotlib!=3.6.1,>=3.4 in c:\users\admin\.conda\envs\myenv\lib\site-packages (from seaborn) (3.10.0)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\admin\.conda\envs\myenv\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (1.3.1)
Requirement already satisfied: cyclery>=0.10 in c:\users\admin\.conda\envs\myenv\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\admin\.conda\envs\myenv\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (4.55.3)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\admin\.conda\envs\myenv\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (1.4.8)
Requirement already satisfied: packaging>=20.0 in c:\users\admin\.conda\envs\myenv\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (24.2)
Requirement already satisfied: pillow>=8 in c:\users\admin\.conda\envs\myenv\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (11.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\admin\.conda\envs\myenv\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (3.2.0)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\admin\.conda\envs\myenv\lib\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\admin\.conda\envs\myenv\lib\site-packages (from pandas>=1.2->seaborn) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in c:\users\admin\.conda\envs\myenv\lib\site-packages (from pandas>=1.2->seaborn) (2025.2)
Requirement already satisfied: six>=1.5 in c:\users\admin\.conda\envs\myenv\lib\site-packages (from python-dateutil>=2.7->matplotlib!=3.6.1,>=3.4->seaborn) (1.17.0)
Note: you may need to restart the kernel to use updated packages.

```
In [35]: # Visualization settings
sns.set(font_scale=1.2)
# Load the dataset
recipes = pd.read_csv(r"C:\Users\Admin\Downloads\cupcakes - cupcakes.csv")
recipes.head()
```

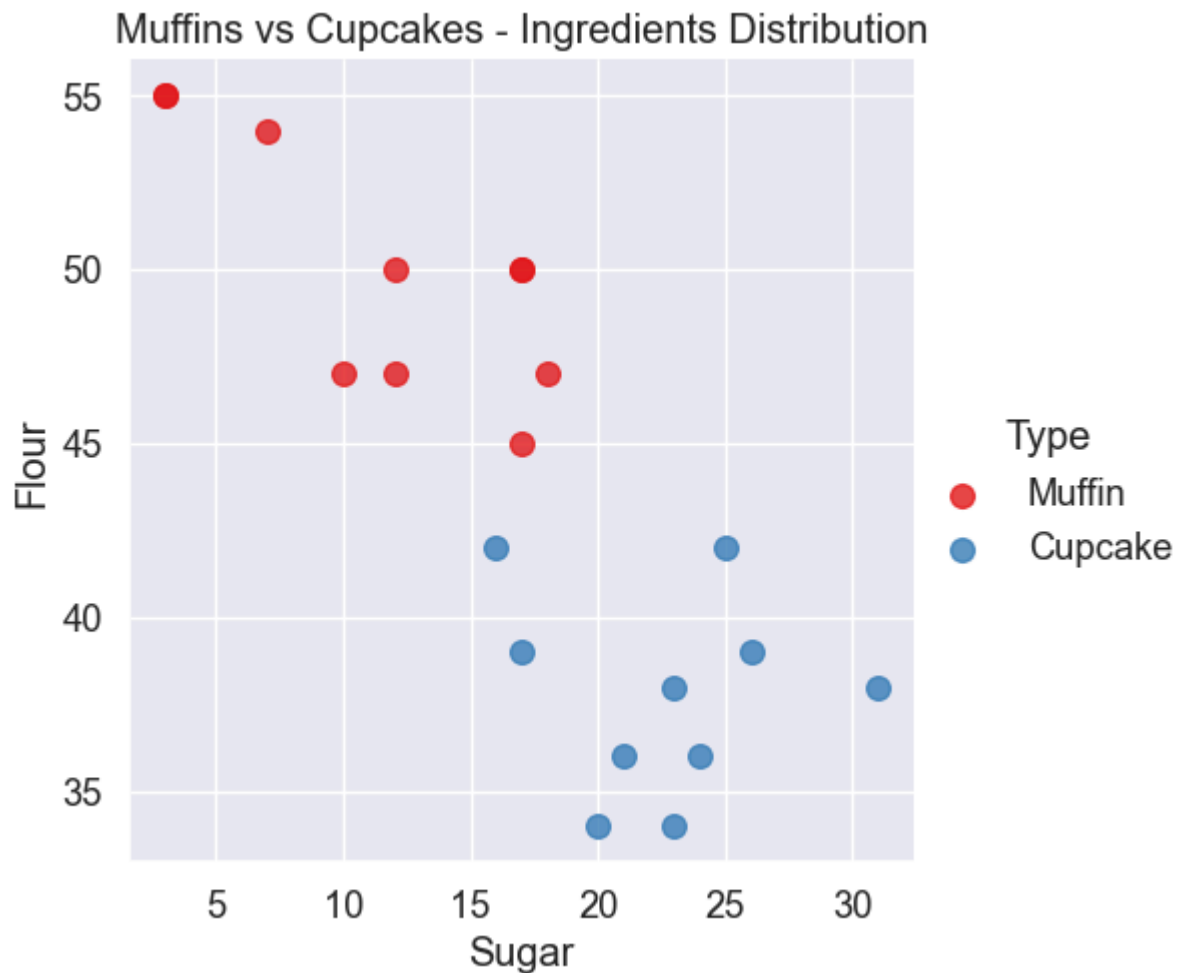
```
Out[35]:
```

	Type	Flour	Milk	Sugar	Butter	Egg	Baking Powder	Vanilla	Salt
0	Muffin	55	28	3	7	5	2	0	0
1	Muffin	47	24	12	6	9	1	0	0
2	Muffin	47	23	18	6	4	1	0	0
3	Muffin	45	11	17	17	8	1	0	0
4	Muffin	50	25	12	6	5	2	1	0

```
In [36]: recipes.shape
```

Out[36]: (20, 9)

```
In [37]: # Plot the data
sns.lmplot(x='Sugar', y='Flour', data=recipes, hue='Type',
           palette='Set1', fit_reg=False, scatter_kws={"s": 70})
plt.title("Muffins vs Cupcakes - Ingredients Distribution")
plt.show()
```



```
In [38]: # Prepare the data
X = recipes[['Sugar', 'Flour']]
y = recipes['Type']
```

```
In [39]: # Train the SVM model
model = svm.SVC(kernel='linear')
model.fit(X, y)
```

Out[39]:

▼ SVC ⓘ ?

► Parameters

```
In [40]: # Get model coefficients  $w_0 * x + w_1 * y + b = 0$ 
w0 = model.coef_[0][0]
w1 = model.coef_[0][1]
b = model.intercept_[0]
```

```
In [41]: # Calculate slope and intercept for the main decision boundary  $y = -(w_0/w_1)x - b/w_1$ 
a = -w0 / w1
xx = np.linspace(5, 30) # Range for x-axis
yy = a * xx - (b / w1) # Decision boundary line
```

In [42]: *# Calculate upper and lower margin lines using support vectors*

```
b_down = model.support_vectors_[0]
yy_down = a * xx + (b_down[1] - a * b_down[0])
b_up = model.support_vectors_[-1]
yy_up = a * xx + (b_up[1] - a * b_up[0])
```

In [43]: *# Plot the data and the SVM boundaries*

```
sns.lmplot(x='Sugar', y='Flour', data=recipes, hue='Type',
           palette='Set1', fit_reg=False, scatter_kws={"s": 70})

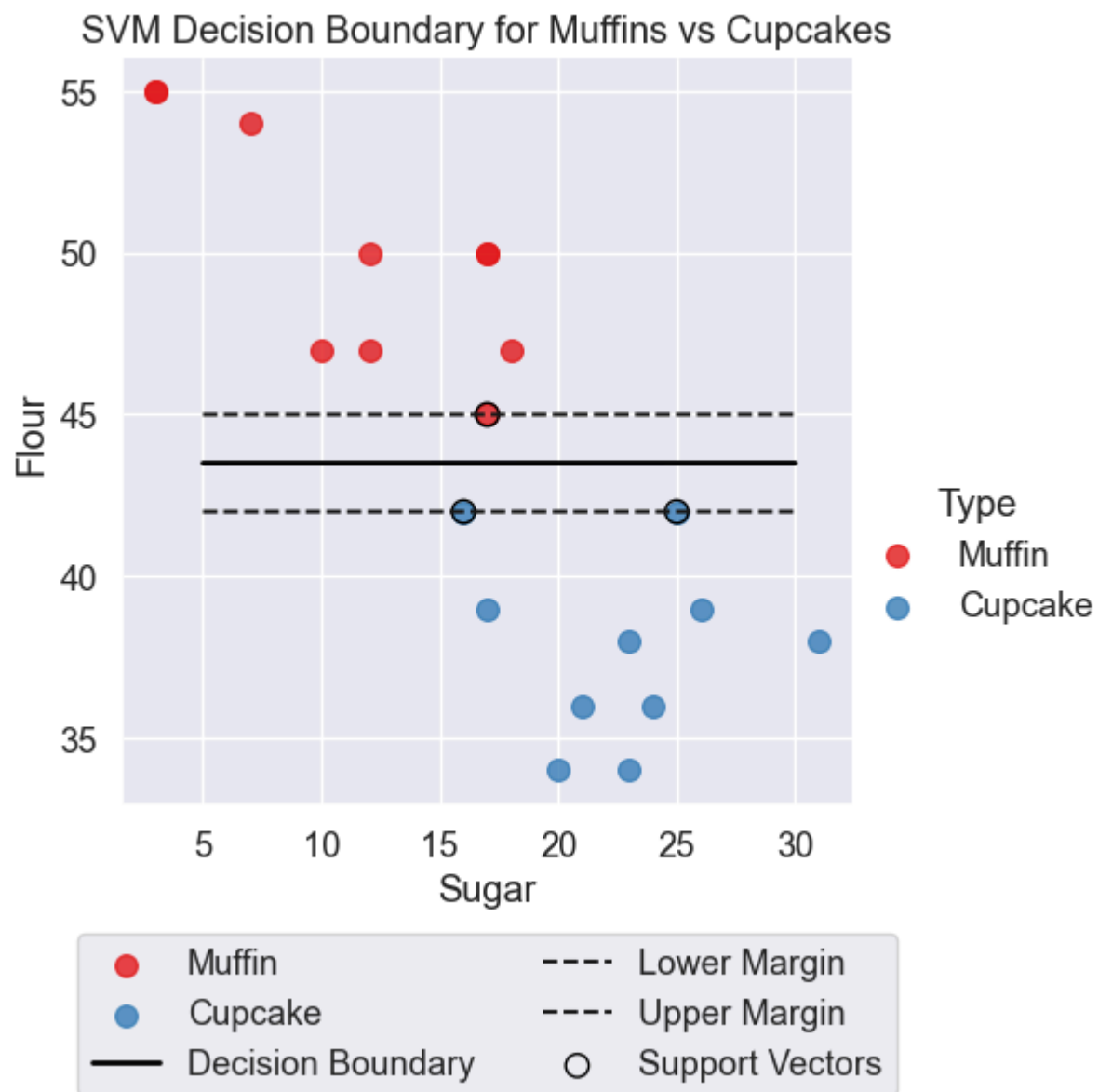
# Plot the decision boundary and margins
plt.plot(xx, yy, linewidth=2, color='black', label='Decision Boundary')
plt.plot(xx, yy_down, 'k--', label='Lower Margin')
plt.plot(xx, yy_up, 'k--', label='Upper Margin')

# Highlight the support vectors
plt.scatter(model.support_vectors_[0],
            model.support_vectors_[-1],
            s=80, facecolors='none', edgecolors='black', label='Support Vectors')

# Final plot formatting
plt.title("SVM Decision Boundary for Muffins vs Cupcakes")
plt.xlabel("Sugar")
plt.ylabel("Flour")

# Legend at the bottom
plt.legend(loc='upper center', bbox_to_anchor=(0.5, -0.15), ncol=2)

plt.show()
```



```
In [44]: # Train-test split
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train model again on training data
model1 = svm.SVC(kernel='linear')
model1.fit(x_train, y_train)
```

Out[44]:

▼ SVC ⓘ ?

► Parameters

```
In [45]: # Make predictions
pred = model1.predict(x_test)
```

```
In [46]: # Evaluate the model
print("\nConfusion Matrix:")
print(confusion_matrix(y_test, pred))
print("\nClassification Report:")
print(classification_report(y_test, pred, target_names=['Muffin', 'Cupcake']))
```

Confusion Matrix:

```
[[1 1]
 [0 2]]
```

Classification Report:

	precision	recall	f1-score	support
Muffin	1.00	0.50	0.67	2
Cupcake	0.67	1.00	0.80	2
accuracy			0.75	4
macro avg	0.83	0.75	0.73	4
weighted avg	0.83	0.75	0.73	4

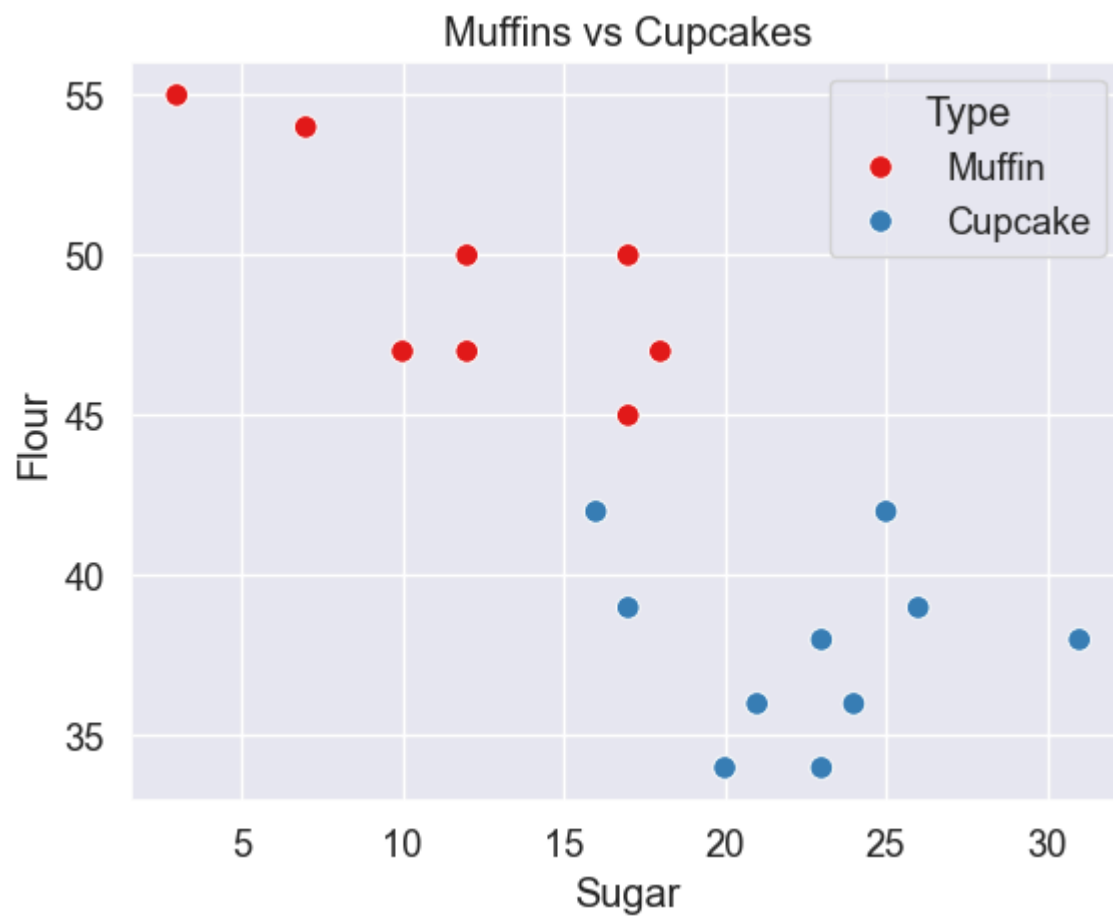
```
In [47]: # Import Libraries
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix, classification_report
# Load the dataset
recipes = pd.read_csv(r"C:\Users\Admin\Downloads\cupcakes - cupcakes.csv")
# Features and target
X = recipes[['Sugar', 'Flour']]
y = recipes['Type']
# Split into training and testing
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
# Train SVM model
model = SVC(kernel='linear')
model.fit(X_train, y_train)
# Make predictions
y_pred = model.predict(X_test)
# Evaluate
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
# Plot data with predictions
sns.scatterplot(data=recipes, x='Sugar', y='Flour', hue='Type', palette='Set1', s=70)
plt.title("Muffins vs Cupcakes")
plt.xlabel("Sugar")
plt.ylabel("Flour")
plt.show()
```

Confusion Matrix:

```
[[1 1]
 [0 2]]
```

Classification Report:

	precision	recall	f1-score	support
Cupcake	1.00	0.50	0.67	2
Muffin	0.67	1.00	0.80	2
accuracy			0.75	4
macro avg	0.83	0.75	0.73	4
weighted avg	0.83	0.75	0.73	4



In []: