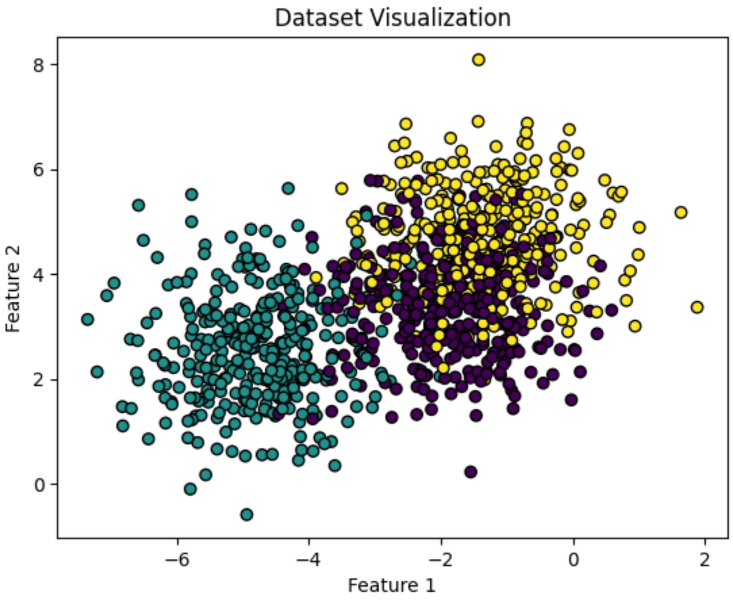
KNN and SVM

January 20, 2025

[18]:
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import make_blobs
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, confusion_matrix,␣
 ↪classification_report
```

[19]:
```
X, y = make_blobs(n_samples=1000, centers=3, n_features=2, random_state=22051662)
```

[20]:
```
plt.scatter(X[:, 0], X[:, 1], c=y, cmap='viridis', edgecolor='k')
plt.title("Dataset Visualization")
plt.xlabel("Feature 1")
plt.ylabel("Feature 2")
plt.show()
```



[17]:

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

0.0.1 KNN

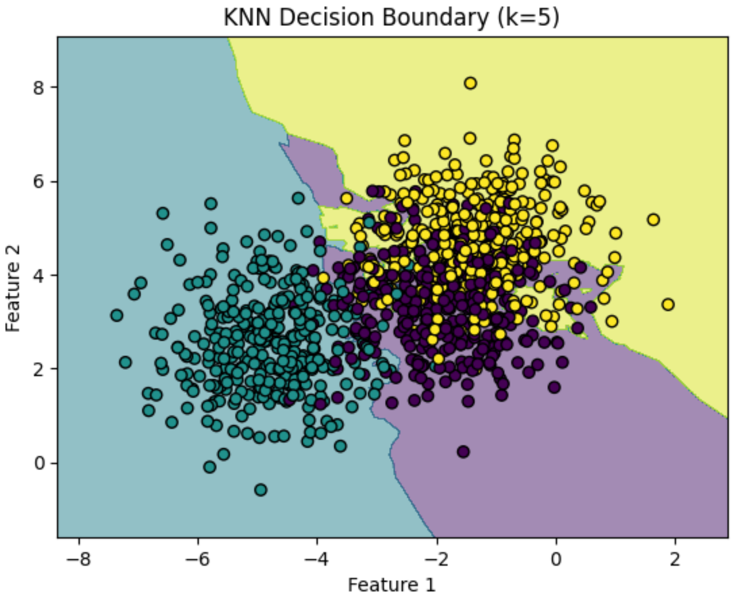
[21]:

```
k = 5
knn = KNeighborsClassifier(n_neighbors=k)
knn.fit(X_train, y_train)

y_pred = knn.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)

print(f"Accuracy: {accuracy:.2f}")
print("\nConfusion Matrix:")
print(conf_matrix)
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```

```
Accuracy: 0.77

Confusion Matrix:
[[ 60   8  28]
 [  4 103   1]
 [ 26   1  69]]

Classification Report:
              precision    recall  f1-score   support

           0       0.67      0.62      0.65        96
           1       0.92      0.95      0.94       108
           2       0.70      0.72      0.71        96

    accuracy                           0.77       300
   macro avg       0.76      0.77      0.76       300
weighted avg       0.77      0.77      0.77       300
```

[12]:

```
x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, 0.01), np.arange(y_min, y_max, 0.
 ↪01))
Z = knn.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)

plt.contourf(xx, yy, Z, alpha=0.5, cmap='viridis')
plt.scatter(X[:, 0], X[:, 1], c=y, edgecolor='k', cmap='viridis')
plt.title(f"KNN Decision Boundary (k={k})")
plt.xlabel("Feature 1")
plt.ylabel("Feature 2")
plt.show()
```



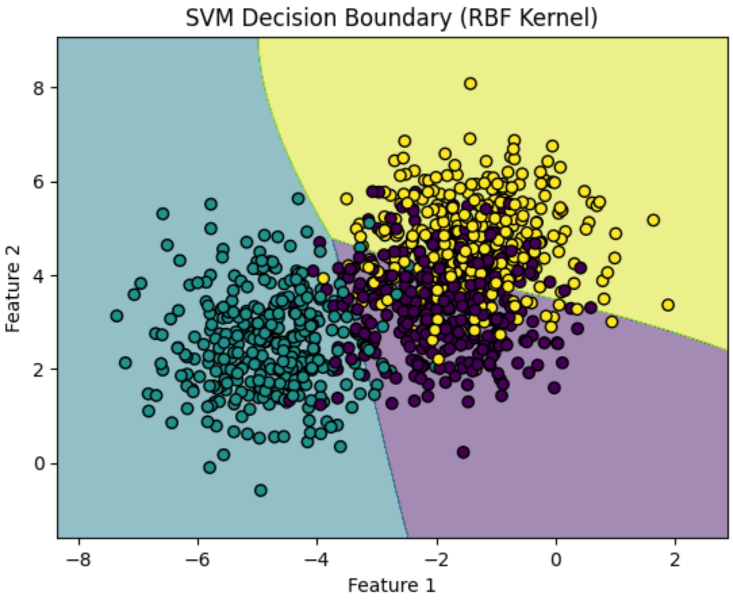
0.0.2 SVM

[23]:
```
svm = SVC(kernel='rbf', C=1, gamma='scale')  # RBF kernel, adjustable parameters␣
↪C and gamma
svm.fit(X_train, y_train)

y_pred = svm.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)

print(f"Accuracy: {accuracy:.2f}")
print("\nConfusion Matrix:")
print(conf_matrix)
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```

```
Accuracy: 0.84

Confusion Matrix:
[[ 65   9  22]
 [  5 102   1]
 [ 11   1  84]]

Classification Report:
              precision    recall  f1-score   support

           0       0.80      0.68      0.73        96
           1       0.91      0.94      0.93       108
           2       0.79      0.88      0.83        96

    accuracy                           0.84       300
   macro avg       0.83      0.83      0.83       300
weighted avg       0.84      0.84      0.83       300
```

[24]:

```python
x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, 0.01), np.arange(y_min, y_max, 0.
 ↪01))
Z = svm.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)

plt.contourf(xx, yy, Z, alpha=0.5, cmap='viridis')
plt.scatter(X[:, 0], X[:, 1], c=y, edgecolor='k', cmap='viridis')
plt.title("SVM Decision Boundary (RBF Kernel)")
plt.xlabel("Feature 1")
plt.ylabel("Feature 2")
plt.show()
```


SVM Decision Boundary (RBF Kernel)
Feature 2
Feature 1