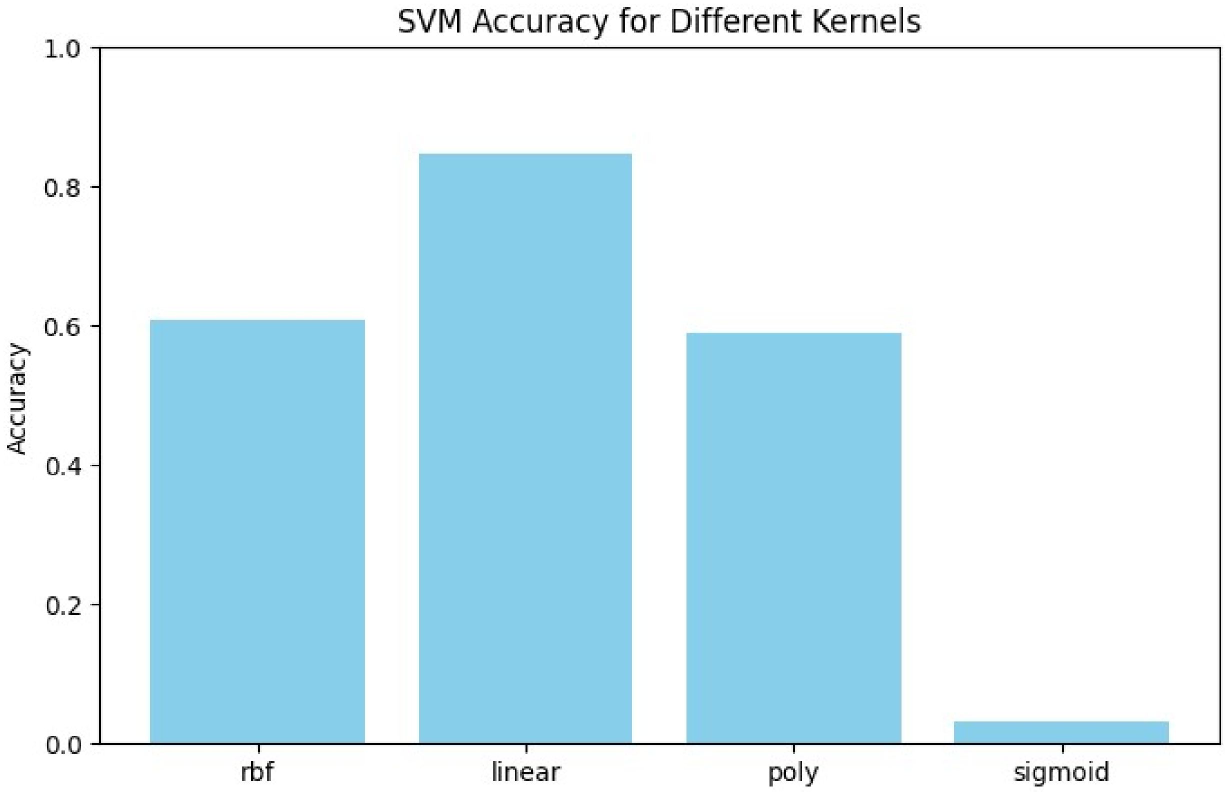
```
import zipfile
import os
import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split, cross_validate,
StratifiedKFold
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix, accuracy_score,
precision_score, recall_score, f1_score
import matplotlib.pyplot as plt

zip_path = 'obesity.dataset.zip'
extract_path = './obesity_dataset'

with zipfile.ZipFile(zip_path, 'r') as zip_ref:
    zip_ref.extractall(extract_path)

csv_file = os.path.join(extract_path,
'ObesityDataSet_raw_and_data_sinthetic.csv')

df = pd.read_csv(csv_file)
print("Dataset loaded. Shape:", df.shape)
print(df.head())

X = df.drop('NObeyesdad', axis=1)
y = df['NObeyesdad']

le_target = LabelEncoder()
y_encoded = le_target.fit_transform(y)

X_encoded = X.copy()
for col in X_encoded.select_dtypes(include='object').columns:
    X_encoded[col] = LabelEncoder().fit_transform(X_encoded[col])

X_train, X_test, y_train, y_test = train_test_split(
    X_encoded, y_encoded, test_size=0.3, stratify=y_encoded,
random_state=42)

kernels = {
    'rbf': {'C': 1.0, 'gamma': 'scale'},
    'linear': {'C': 1.0},
    'poly': {'degree': 3, 'C': 1.0, 'gamma': 'scale'},
    'sigmoid': {'C': 1.0, 'gamma': 'scale'}
}

def train_evaluate_svm(kernel, params, X_train, X_test, y_train,
y_test):
    print(f"\nTraining SVM with kernel = '{kernel}'")
    model = SVC(kernel=kernel, **params, random_state=42)
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)

    cm = confusion_matrix(y_test, y_pred)
    print("Confusion Matrix:\n", cm)

    acc = accuracy_score(y_test, y_pred)
    precision = precision_score(y_test, y_pred, average=None,
zero_division=0)
    recall = recall_score(y_test, y_pred, average=None,
zero_division=0)
    f1 = f1_score(y_test, y_pred, average=None, zero_division=0)

    print(f"Accuracy: {acc:.4f}")
    for i, label in enumerate(le_target.classes_):
        print(f"Class '{label}': Precision={precision[i]:.4f},
Recall={recall[i]:.4f}, F1-score={f1[i]:.4f}")

    return model, acc, precision.mean(), recall.mean(), f1.mean()

results = {}

for kernel, params in kernels.items():
    model, acc, prec, rec, f1 = train_evaluate_svm(kernel, params,
X_train, X_test, y_train, y_test)
    results[kernel] = {'model': model, 'accuracy': acc, 'precision':
prec, 'recall': rec, 'f1': f1}

plt.figure(figsize=(8,5))
plt.bar(results.keys(), [v['accuracy'] for v in results.values()],
color='skyblue')
plt.ylabel('Accuracy')
plt.title('SVM Accuracy for Different Kernels')
plt.ylim(0,1)
plt.show()

cv_results = {'kernel': [], 'accuracy': [], 'precision': [], 'recall':
[], 'f1': []}
skf = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)

for kernel, params in kernels.items():
    svm = SVC(kernel=kernel, **params, random_state=42)
    scoring = ['accuracy', 'precision_macro', 'recall_macro',
'f1_macro']
    scores = cross_validate(svm, X_encoded, y_encoded, cv=skf,
scoring=scoring)

    cv_results['kernel'].append(kernel)
    cv_results['accuracy'].append(np.mean(scores['test_accuracy']))

cv_results['precision'].append(np.mean(scores['test_precision_macro'])
)
    cv_results['recall'].append(np.mean(scores['test_recall_macro']))
    cv_results['f1'].append(np.mean(scores['test_f1_macro']))

plt.figure(figsize=(10,6))
bar_width = 0.2
x = np.arange(len(kernels))

metrics = ['accuracy', 'precision', 'recall', 'f1']
for i, metric in enumerate(metrics):
    plt.bar(x + i*bar_width, cv_results[metric], width=bar_width,
label=metric.capitalize())

plt.xticks(x + bar_width*1.5, cv_results['kernel'])
plt.ylim(0,1)
plt.ylabel('Score')
plt.title('SVM Performance Metrics (5-Fold CV) for Different Kernels')
plt.legend()
plt.show()

rbf_model = results['rbf']['model']
print("\nFirst 5 Support Vectors (RBF kernel):")
print(rbf_model.support_vectors_[:5])
```

```
Dataset loaded. Shape: (2111, 17)
   Gender   Age  Height  Weight family_history_with_overweight FAVC
FCVC  \
0  Female  21.0    1.62    64.0                            yes   no
2.0
1  Female  21.0    1.52    56.0                            yes   no
3.0
2    Male  23.0    1.80    77.0                            yes   no
2.0
3    Male  27.0    1.80    87.0                             no   no
3.0
4    Male  22.0    1.78    89.8                             no   no
2.0

   NCP       CAEC SMOKE  CH2O  SCC  FAF  TUE        CALC  \
0  3.0  Sometimes    no   2.0   no  0.0  1.0          no
1  3.0  Sometimes   yes   3.0  yes  3.0  0.0   Sometimes
2  3.0  Sometimes    no   2.0   no  2.0  1.0  Frequently
3  3.0  Sometimes    no   2.0   no  2.0  0.0  Frequently
4  1.0  Sometimes    no   2.0   no  0.0  0.0   Sometimes

                  MTRANS           NObeyesdad
0  Public_Transportation        Normal_Weight
```

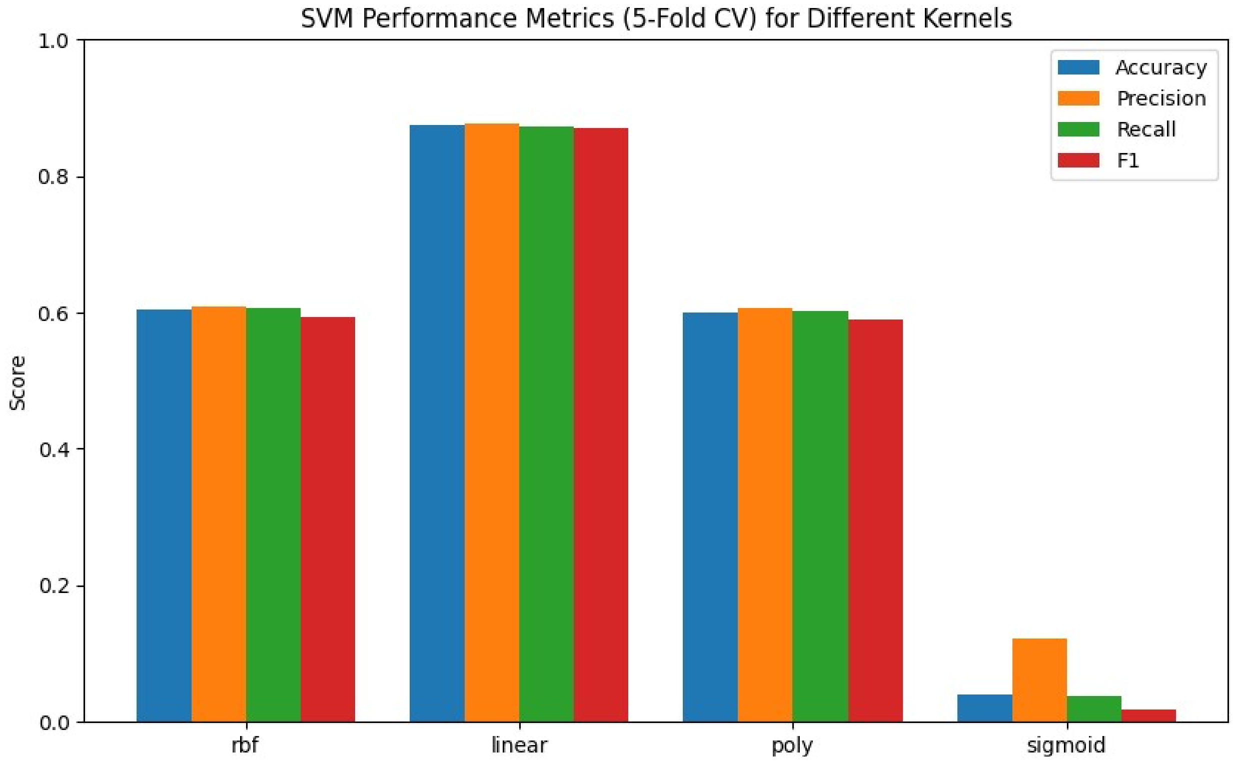
```
1  Public_Transportation        Normal_Weight
2  Public_Transportation        Normal_Weight
3                Walking   Overweight_Level_I
4  Public_Transportation  Overweight_Level_II

Training SVM with kernel = 'rbf'
Confusion Matrix:
 [[68 14  0  0  0  0  0]
 [24 41  0  0  0 19  2]
 [ 0  0 37  3 17  1 48]
 [ 0  0 10 42 37  0  0]
 [ 0  0  0  0 97  0  0]
 [ 2 15  3  0  0 56 11]
 [ 0  3 20  0  0 19 45]]
Accuracy: 0.6088
Class 'Insufficient_Weight': Precision=0.7234, Recall=0.8293, F1-
score=0.7727
Class 'Normal_Weight': Precision=0.5616, Recall=0.4767, F1-
score=0.5157
Class 'Obesity_Type_I': Precision=0.5286, Recall=0.3491, F1-
score=0.4205
Class 'Obesity_Type_II': Precision=0.9333, Recall=0.4719, F1-
score=0.6269
Class 'Obesity_Type_III': Precision=0.6424, Recall=1.0000, F1-
score=0.7823
Class 'Overweight_Level_I': Precision=0.5895, Recall=0.6437, F1-
score=0.6154
Class 'Overweight_Level_II': Precision=0.4245, Recall=0.5172, F1-
score=0.4663

Training SVM with kernel = 'linear'
Confusion Matrix:
 [[76  6  0  0  0  0  0]
 [11 61  0  0  0 13  1]
 [ 0  0 92  5  0  2  7]
 [ 0  0  2 87  0  0  0]
 [ 0  0  0  1 96  0  0]
 [ 0  9  1  0  0 69  8]
 [ 0  1 14  0  0 15 57]]
Accuracy: 0.8486
Class 'Insufficient_Weight': Precision=0.8736, Recall=0.9268, F1-
score=0.8994
Class 'Normal_Weight': Precision=0.7922, Recall=0.7093, F1-
score=0.7485
Class 'Obesity_Type_I': Precision=0.8440, Recall=0.8679, F1-
score=0.8558
Class 'Obesity_Type_II': Precision=0.9355, Recall=0.9775, F1-
score=0.9560
Class 'Obesity_Type_III': Precision=1.0000, Recall=0.9897, F1-
```

```
score=0.9948
Class 'Overweight_Level_I': Precision=0.6970, Recall=0.7931, F1-
score=0.7419
Class 'Overweight_Level_II': Precision=0.7808, Recall=0.6552, F1-
score=0.7125

Training SVM with kernel = 'poly'
Confusion Matrix:
 [[74  8  0  0  0  0  0]
 [26 46  0  0  0 12  2]
 [ 0  0 43  3  6  3 51]
 [ 0  0 11 38 40  0  0]
 [ 0  0  3  0 94  0  0]
 [ 3 29  1  0  0 42 12]
 [ 0  5 18  0  0 26 38]]
Accuracy: 0.5915
Class 'Insufficient_Weight': Precision=0.7184, Recall=0.9024, F1-
score=0.8000
Class 'Normal_Weight': Precision=0.5227, Recall=0.5349, F1-
score=0.5287
Class 'Obesity_Type_I': Precision=0.5658, Recall=0.4057, F1-
score=0.4725
Class 'Obesity_Type_II': Precision=0.9268, Recall=0.4270, F1-
score=0.5846
Class 'Obesity_Type_III': Precision=0.6714, Recall=0.9691, F1-
score=0.7932
Class 'Overweight_Level_I': Precision=0.5060, Recall=0.4828, F1-
score=0.4941
Class 'Overweight_Level_II': Precision=0.3689, Recall=0.4368, F1-
score=0.4000

Training SVM with kernel = 'sigmoid'
Confusion Matrix:
 [[  2   0   0   0  80   0   0]
 [  0   0   0   0  86   0   0]
 [  0   3   0   0 100   3   0]
 [  0  74   0   0  14   1   0]
 [  2  66   0   0  19  10   0]
 [  0   0   0   0  87   0   0]
 [  0   0   0   0  87   0   0]]
Accuracy: 0.0331
Class 'Insufficient_Weight': Precision=0.5000, Recall=0.0244, F1-
score=0.0465
Class 'Normal_Weight': Precision=0.0000, Recall=0.0000, F1-
score=0.0000
Class 'Obesity_Type_I': Precision=0.0000, Recall=0.0000, F1-
score=0.0000
Class 'Obesity_Type_II': Precision=0.0000, Recall=0.0000, F1-
score=0.0000
```

```
Class 'Obesity_Type_III': Precision=0.0402, Recall=0.1959, F1-
score=0.0667
Class 'Overweight_Level_I': Precision=0.0000, Recall=0.0000, F1-
score=0.0000
Class 'Overweight_Level_II': Precision=0.0000, Recall=0.0000, F1-
score=0.0000
```



```
/usr/local/lib/python3.12/dist-packages/sklearn/metrics/
_classification.py:1565: UndefinedMetricWarning: Precision is ill-
defined and being set to 0.0 in labels with no predicted samples. Use
`zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/usr/local/lib/python3.12/dist-packages/sklearn/metrics/_classificatio
n.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/usr/local/lib/python3.12/dist-packages/sklearn/metrics/_classificatio
n.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/usr/local/lib/python3.12/dist-packages/sklearn/metrics/_classificatio
n.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
/usr/local/lib/python3.12/dist-packages/sklearn/metrics/_classificatio
n.py:1565: UndefinedMetricWarning: Precision is ill-defined and being
set to 0.0 in labels with no predicted samples. Use `zero_division`
parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is",
len(result))
```



```

First 5 Support Vectors (RBF kernel):
[[ 0.        21.798856  1.672007 49.980968  1.         1.
2.979383
   3.        1.        0.        2.975887  0.        0.945093
1.241755
   3.        3.       ]
 [ 0.        19.272573  1.71367  50.        0.        1.
1.005578
   4.        1.        0.        1.        0.        1.683957
0.704978
```

```
   2.        3.      ]
 [ 1.       19.        1.7      50.        0.         1.         1.
   4.        1.        0.        1.        0.         2.         1.
   2.        3.      ]
 [ 0.       19.665881  1.676346 49.105025  0.         0.
2.720701
   3.        2.        0.        2.        0.         1.862235  1.
   2.        3.      ]
 [ 1.       17.038222  1.710564 51.588874  0.         1.         2.
   2.057935  2.        0.        2.371015  0.         0.288032
0.714627
   2.        3.      ]]
```