Assignment 5: Human Activity Recognition (HAR) with Smartphones

Syabab Alhaqqi bin Jaafar   2211117

A) Based on this data set select which parameters (independent variables) will you feel will be suitable/useful to form your matrix of features.

Parameters that can be taken are all signal-related features (e.g., BodyAcc-mean()-X, BodyAcc-std()-Y, BodyGyro-bandsEnergy()-X,8,24) which are highly suitable as input features. These features represent time and frequency domain characteristics of body acceleration and gyroscope signals, which are directly relevant for activity recognition.

B – I)

```
Logistic Regression
Accuracy: 0.9850
                    precision    recall  f1-score   support

            LAYING       1.00      1.00      1.00       282
           SITTING       0.96      0.96      0.96       257
          STANDING       0.96      0.96      0.96       275
           WALKING       1.00      1.00      1.00       245
WALKING_DOWNSTAIRS       1.00      1.00      1.00       197
  WALKING_UPSTAIRS       1.00      1.00      1.00       215

          accuracy                           0.99      1471
         macro avg       0.99      0.99      0.99      1471
      weighted avg       0.99      0.99      0.99      1471
```

Performs well due to structured and separable features from sensors but may be outperformed by models that can capture more complex relationships.

```
KNN (k=5)
Accuracy: 0.9606
                    precision    recall  f1-score   support

            LAYING       1.00      1.00      1.00       282
           SITTING       0.93      0.86      0.89       257
          STANDING       0.88      0.94      0.91       275
           WALKING       0.98      1.00      0.99       245
WALKING_DOWNSTAIRS       1.00      0.97      0.99       197
  WALKING_UPSTAIRS       1.00      1.00      1.00       215

          accuracy                           0.96      1471
         macro avg       0.96      0.96      0.96      1471
      weighted avg       0.96      0.96      0.96      1471
```

Can work well if the classes are locally clustered but slows down and becomes less accurate in high dimensions unless features are well-scaled.

```
Linear SVM
Accuracy: 0.9884
                    precision    recall  f1-score   support

            LAYING       1.00      1.00      1.00       282
           SITTING       0.96      0.97      0.97       257
          STANDING       0.97      0.96      0.97       275
           WALKING       1.00      1.00      1.00       245
WALKING_DOWNSTAIRS       1.00      1.00      1.00       197
  WALKING_UPSTAIRS       1.00      1.00      1.00       215

          accuracy                           0.99      1471
         macro avg       0.99      0.99      0.99      1471
      weighted avg       0.99      0.99      0.99      1471
```

Often gives top accuracy since the data is high-dimensional and relatively well-separated by linear boundaries.

```
Non-Linear SVM (RBF)
Accuracy: 0.9721
                    precision    recall  f1-score   support

            LAYING       1.00      1.00      1.00       282
           SITTING       0.93      0.91      0.92       257
          STANDING       0.92      0.94      0.93       275
           WALKING       1.00      1.00      1.00       245
WALKING_DOWNSTAIRS       1.00      0.99      1.00       197
  WALKING_UPSTAIRS       1.00      1.00      1.00       215

          accuracy                           0.97      1471
         macro avg       0.97      0.97      0.97      1471
      weighted avg       0.97      0.97      0.97      1471
```

May slightly underperform linear SVM if the actual decision boundary is close to linear. Otherwise, useful when activity data has more subtle patterns.

```
Naive Bayes
Accuracy: 0.7634
                    precision    recall  f1-score   support

            LAYING       0.97      0.60      0.74       282
           SITTING       0.56      0.60      0.58       257
          STANDING       0.72      0.92      0.81       275
           WALKING       0.96      0.76      0.84       245
WALKING_DOWNSTAIRS       0.80      0.82      0.81       197
  WALKING_UPSTAIRS       0.73      0.93      0.82       215

          accuracy                           0.76      1471
         macro avg       0.79      0.77      0.77      1471
      weighted avg       0.79      0.76      0.76      1471
```

Simple and fast baseline, but generally less accurate due to correlated inertial features (e.g., accelerometer x, y, z are not independent).

```
Decision Tree
Accuracy: 0.9334
                    precision    recall  f1-score   support

            LAYING       1.00      1.00      1.00       282
           SITTING       0.89      0.89      0.89       257
          STANDING       0.89      0.90      0.90       275
           WALKING       0.93      0.96      0.94       245
WALKING_DOWNSTAIRS       0.92      0.95      0.94       197
  WALKING_UPSTAIRS       0.97      0.90      0.93       215

          accuracy                           0.93      1471
         macro avg       0.93      0.93      0.93      1471
      weighted avg       0.93      0.93      0.93      1471
```

May overfit due to many features, leading to lower generalization performance unless pruned.

```
Random Forest
Accuracy: 0.9871
                    precision    recall  f1-score   support

            LAYING       1.00      1.00      1.00       282
           SITTING       0.98      0.96      0.97       257
          STANDING       0.96      0.99      0.97       275
           WALKING       1.00      0.99      0.99       245
WALKING_DOWNSTAIRS       0.98      0.99      0.99       197
  WALKING_UPSTAIRS       1.00      1.00      1.00       215

          accuracy                           0.99      1471
         macro avg       0.99      0.99      0.99      1471
      weighted avg       0.99      0.99      0.99      1471
```

Performs very well and often second to Linear SVM especially when activity boundaries are not perfectly linear.

```
Summary of Model Accuracies:
                      Accuracy
Linear SVM            0.988443
Random Forest         0.987084
Logistic Regression   0.985044
Non-Linear SVM (RBF)  0.972128
KNN (k=5)             0.960571
Decision Tree         0.933379
Naive Bayes           0.763426
```

We can see the most accurate model is Linear SVM.

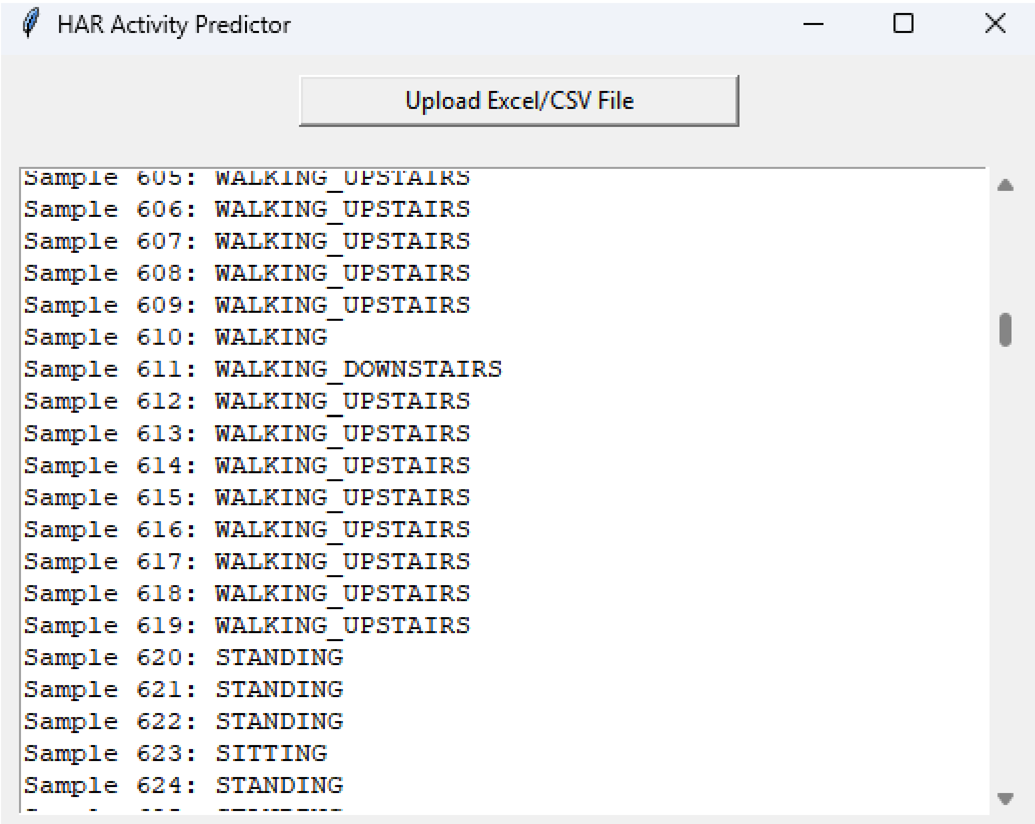
J) Based on your results, determine the best model for this data and justify your answer.

The best model for this is Linear SVM which has the highest accuracy. This is because the HAR dataset has many features which makes Linear SVM works exceptionally well due to the high-dimensional spaces when the data is linearly separable.

H) Develop a simple GUI using tkinter or any other Python tool where users can upload an excel file consisting of human activity data and the GUI will display whether the activity corresponds to walking, running, etc.

Model and LabelEncoder saved as 'har_model.pkl' and 'label_encoder.pkl' using the Linear SVM model.

The uploaded file is test.csv



APPENDIX

1) GUI code

```
import tkinter as tk
from tkinter import filedialog, messagebox, scrolledtext
import pandas as pd
import joblib

# Load pre-trained model and encoder
model = joblib.load("har_model.pkl")
le = joblib.load("label_encoder.pkl")

# Prediction function
def predict_activity():
    file_path = filedialog.askopenfilename(filetypes=[("Excel or CSV files", "*.xlsx *.csv")])
    if not file_path:
        return

    try:
        # Read file
        if file_path.endswith(".csv"):
            data = pd.read_csv(file_path)
        else:
            data = pd.read_excel(file_path)

        # Drop non-feature columns if present
        features = data.drop(columns=['subject', 'Activity'], errors='ignore')

        # Predict
        predictions = model.predict(features)
        activities = le.inverse_transform(predictions)
```

```
        # Display in textbox
        text_box.delete(1.0, tk.END)
        for i, activity in enumerate(activities):
            text_box.insert(tk.END, f"Sample {i+1}: {activity}\n")

    except Exception as e:
        messagebox.showerror("Error", f"Failed to process file:\n{str(e)}")

# Setup GUI
root = tk.Tk()
root.title("HAR Activity Predictor")

upload_button = tk.Button(root, text="Upload Excel/CSV File",
command=predict_activity, width=30)
upload_button.pack(pady=10)

text_box = scrolledtext.ScrolledText(root, width=60, height=20)
text_box.pack(padx=10, pady=10)

root.mainloop()
```

2) Model for GUI using Linear SVM code

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report
import joblib
import warnings

warnings.filterwarnings("ignore")
```

```
# 1. Load datasets
train_df = pd.read_csv("train.csv")
test_df = pd.read_csv("test.csv")

# 2. Prepare features and labels
X_train = train_df.drop(columns=['subject', 'Activity'])
y_train = train_df['Activity']

X_test = test_df.drop(columns=['subject', 'Activity'], errors='ignore')
y_test = test_df['Activity'] if 'Activity' in test_df.columns else None

# 3. Encode target labels
le = LabelEncoder()
y_train_encoded = le.fit_transform(y_train)

if y_test is not None:
    y_test_encoded = le.transform(y_test)

# 4. Train Linear SVM
model = SVC(kernel='linear')
model.fit(X_train, y_train_encoded)

# 5. Predict on test data
y_pred_encoded = model.predict(X_test)
y_pred_labels = le.inverse_transform(y_pred_encoded)

# 6. Output Results
print("✅ Prediction complete.")
print("First 10 predicted activities:")
print(y_pred_labels[:10])
```

```
# Evaluate if y_test is available
if y_test is not None:
   print("\n📊 Evaluation Metrics:")
   print(f"Accuracy: {accuracy_score(y_test_encoded, y_pred_encoded):.4f}")
   print(classification_report(y_test_encoded, y_pred_encoded, target_names=le.classes_))

# 7. Save the model and label encoder
joblib.dump(model, 'har_model.pkl')
joblib.dump(le, 'label_encoder.pkl')
print("\n💾 Model and LabelEncoder saved as 'har_model.pkl' and 'label_encoder.pkl'")
```