HEART DISEASE PREDICTION

Moe Htet Min

Table of Content

Project Report: Heart Disease Prediction

GitHub: https://github.com/Moehtetmin28/Heart-Disease-Prediction

1. Objective

This project aims to develop a predictive machine learning model which determines whether heart disease exists in patients from their clinical characteristic data points that include age, sex, cholesterol levels, chest pain type, and other variables. The predicted results enable early heart disease diagnosis followed by treatment which reduces health complications from heart disease.

2. Dataset

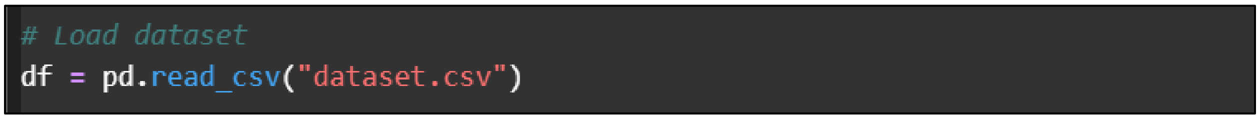
The heart disease data collection serves as the basis for this project since the target variable shows heart disease status through 0 for no disease and 1 for disease presence. The prepared dataset includes features which contain medical information about patients. The features include:

- Age: Age of the patient.
- Sex: Gender of the patient (male or female).
- Chest Pain Type: Type of chest pain experienced by the patient.
- Resting Blood Pressure: Resting blood pressure in mm Hg.
- Cholesterol: Serum cholesterol in mg/dl.
- Fasting Blood Sugar: Blood sugar level after fasting.
- Resting Electrocardiographic Results: Resting electrocardiographic results.
- Max Heart Rate: Maximum heart rate achieved during exercise.
- Exercise Induced Angina: Whether the patient experiences chest pain induced by exercise.
- Oldpeak: Depression induced by exercise relative to rest.
- ST Slope: The slope of the peak exercise ST segment.

3. Data Preprocessing

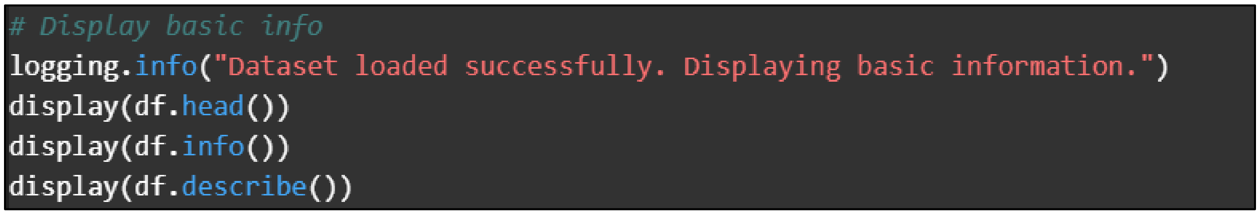
Loading the Dataset

The first step involves loading my dataset into a DataFrame through **pandas**:



```python
# Load dataset
df = pd.read_csv("dataset.csv")
```

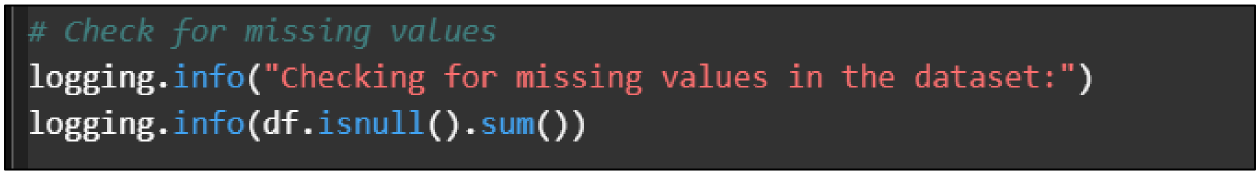
The first step involves checking the data structure through an inspection procedure:



```python
# Display basic info
logging.info("Dataset loaded successfully. Displaying basic information.")
display(df.head())
display(df.info())
display(df.describe())
```

Checking for Missing Values

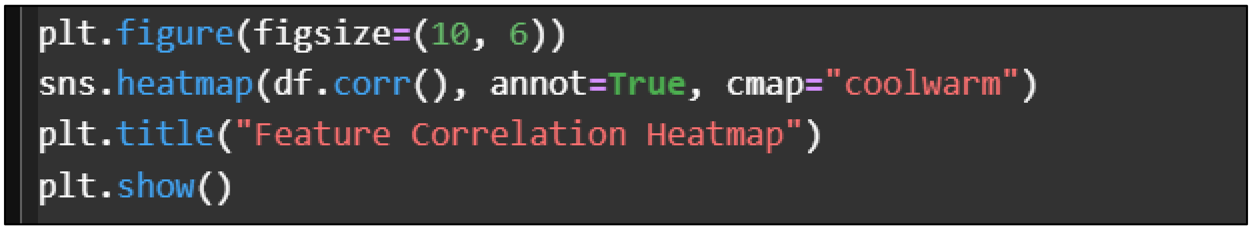
I conduct an examination of the dataset for absent values next:



```python
# Check for missing values
logging.info("Checking for missing values in the dataset:")
logging.info(df.isnull().sum())
```

Visualizing Feature Correlation

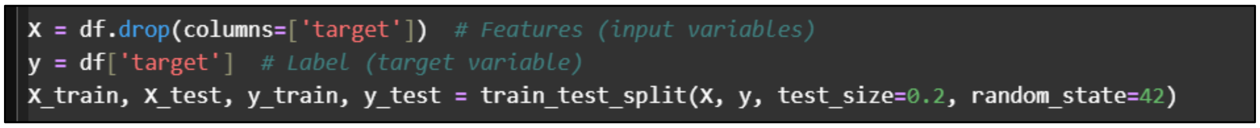
The dataset shows its feature relations through a heatmap analysis:



```python
plt.figure(figsize=(10, 6))
sns.heatmap(df.corr(), annot=True, cmap="coolwarm")
plt.title("Feature Correlation Heatmap")
plt.show()
```

4. Feature Selection and Data Splitting

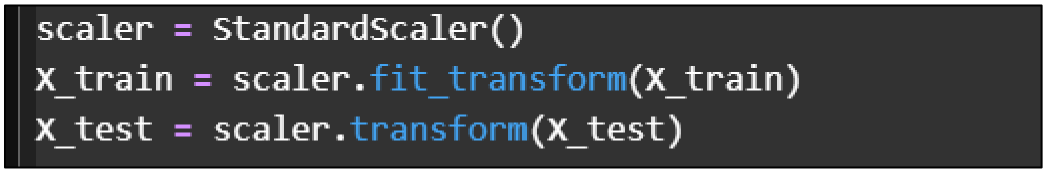
The database divides its content into two sections which consist of the input features (**X**) alongside the target variable (**y**). The column **'target'** contains the heart disease presence value as the target variable.



```
X = df.drop(columns=['target'])  # Features (input variables)
y = df['target']  # Label (target variable)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

5. Feature Scaling

The implementation of **StandardScaler** performs feature standardization to ensure features are comparable between each other:



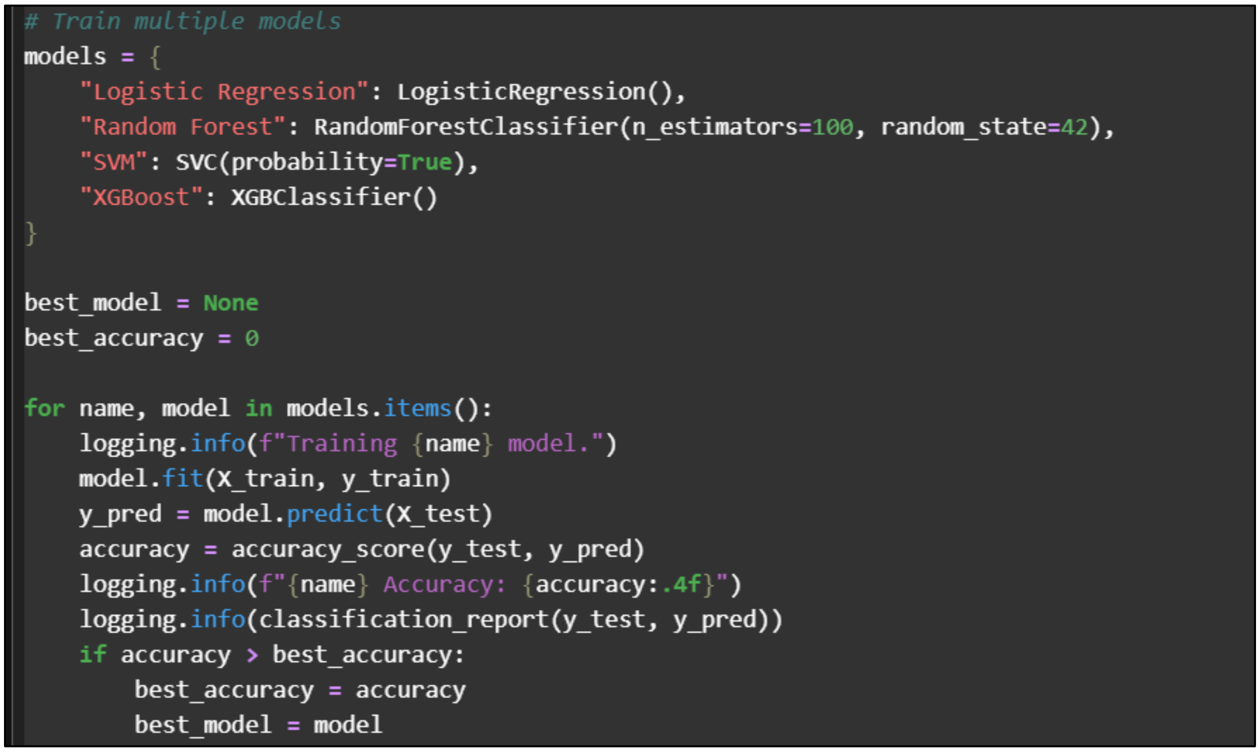
```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

6. **Model Training**

Defining and Training Multiple Models

I examine the test set accuracy of various models I have trained. The models used are:

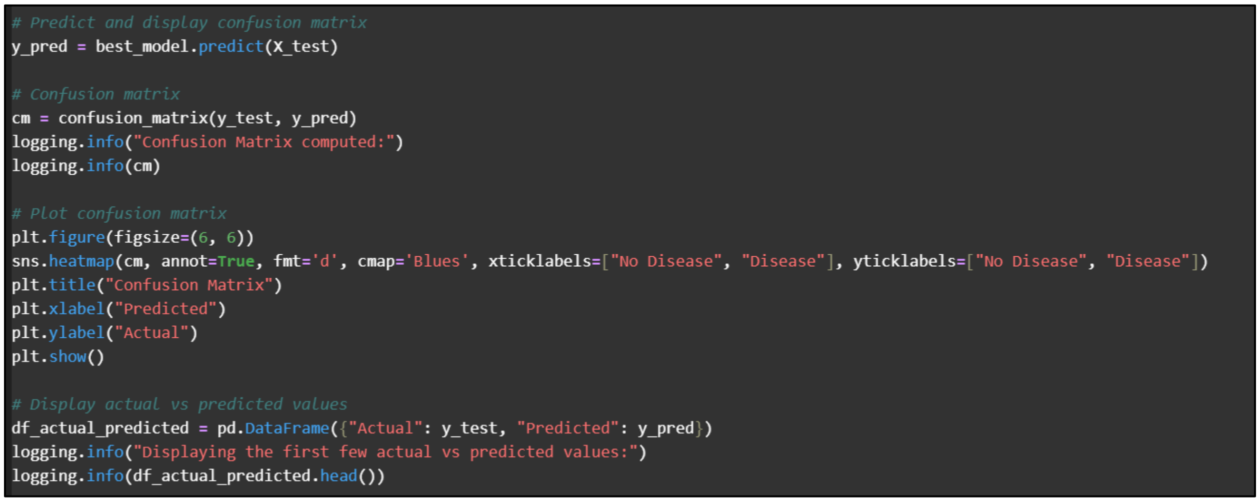
- Logistic Regression
- Random Forest Classifier
- Support Vector Machine (SVM)
- XGBoost



```python
# Train multiple models
models = {
    "Logistic Regression": LogisticRegression(),
    "Random Forest": RandomForestClassifier(n_estimators=100, random_state=42),
    "SVM": SVC(probability=True),
    "XGBoost": XGBClassifier()
}

best_model = None
best_accuracy = 0

for name, model in models.items():
    logging.info(f"Training {name} model.")
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    logging.info(f"{name} Accuracy: {accuracy:.4f}")
    logging.info(classification_report(y_test, y_pred))
    if accuracy > best_accuracy:
        best_accuracy = accuracy
        best_model = model
```

7. Model Evaluation

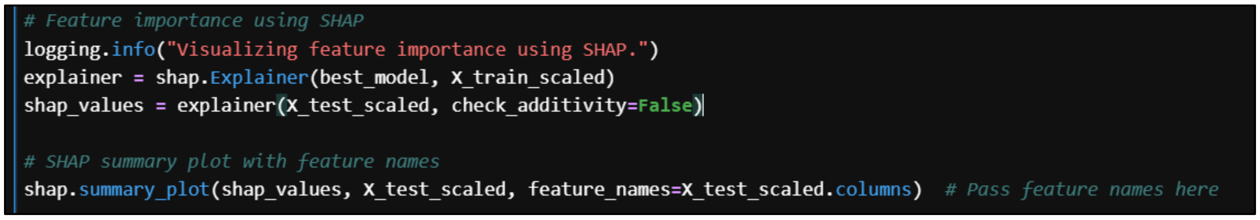
I measure the best model using accuracy and create a classification report for assessment purposes.



```python
# Predict and display confusion matrix
y_pred = best_model.predict(X_test)

# Confusion matrix
cm = confusion_matrix(y_test, y_pred)
logging.info("Confusion Matrix computed:")
logging.info(cm)

# Plot confusion matrix
plt.figure(figsize=(6, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=["No Disease", "Disease"], yticklabels=["No Disease", "Disease"])
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()

# Display actual vs predicted values
df_actual_predicted = pd.DataFrame({"Actual": y_test, "Predicted": y_pred})
logging.info("Displaying the first few actual vs predicted values:")
logging.info(df_actual_predicted.head())
```

8. Feature Importance using SHAP

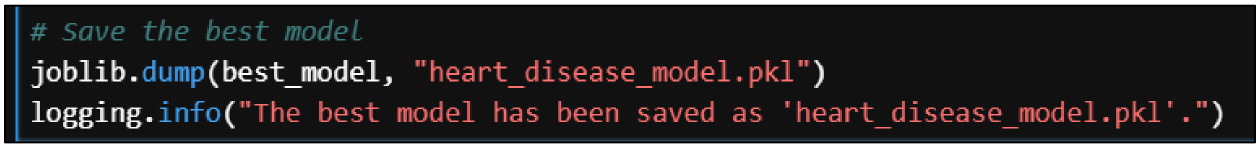
The SHAP (SHapley Additive exPlanations) technique enables visualization of important features for the best model selected.



```
# Feature importance using SHAP
logging.info("Visualizing feature importance using SHAP.")
explainer = shap.Explainer(best_model, X_train_scaled)
shap_values = explainer(X_test_scaled, check_additivity=False)

# SHAP summary plot with feature names
shap.summary_plot(shap_values, X_test_scaled, feature_names=X_test_scaled.columns)  # Pass feature names here
```

9. Saving the Best Model

The best model acquirement occurred through **joblib** for potential future utilization.



```
# Save the best model
joblib.dump(best_model, "heart_disease_model.pkl")
logging.info("The best model has been saved as 'heart_disease_model.pkl'.")
```

10. Conclusion

- The machine learning system delivered acceptable performance results since it offered decent accuracy during heart disease predictions.
- This operational model enables medical professionals to use datasets for diagnosing heart disease in their clinical practice.
- The SHAP visualization tools helped me determine which age alongside sex and cholesterol levels were the most significant features driving model prediction results.
- The selected model is ready for implementation after its successful preservation.

11. Future Work

- **Hyperparameter Tuning:** A performance boost could occur through hyperparameter optimization in models such as Random Forest and XGBoost as well as SVM.
- **Model Comparison:** An analysis must be conducted to determine if deep learning approaches perform better than predictive models in terms of accuracy.
- **Additional Data:** The accuracy enhancements will benefit from adding new features to datasets or expanding their sizes.

==