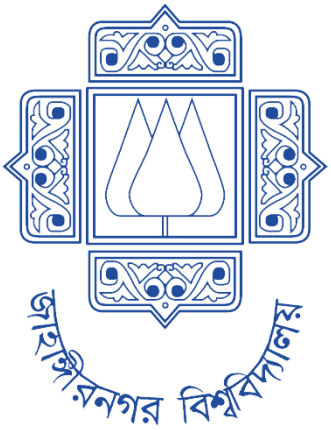
Institute of Information Technology (IIT)
Jahangirnagar University



Course Code: MICT 5402
Course Title: Advanced Machine Learning

Assignment - 02

Submitted to:
Jesmin Akhter
Professor
IIT, JU

Submitted by:
Name: Md. Shakil Hossain
Roll No: 1061
MSc 2nd Semester
IIT, JU

Submission Date: 10/04/2025

Code:

```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import GradientBoostingClassifier, AdaBoostClassifier
from xgboost import XGBClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import accuracy_score


df = pd.read_csv('ML_ass1054.csv')


le = LabelEncoder()
df['Chest Pain'] = le.fit_transform(df['Chest Pain'])
df['Blocked Arteries'] = le.fit_transform(df['Blocked Arteries'])
df['Heart Disease'] = le.fit_transform(df['Heart Disease'])


X = df[['Chest Pain', 'Blocked Arteries', 'Patient Weight']]
y = df['Heart Disease']


X_train, X_test = X[:8], X[8:]
y_train, y_test = y[:8], y[8:]


weights_train = [0.125] * len(X_train)  # 8 weights for 8 training samples
weights_test = [0.125] * len(X_test)   # 4 weights for 4 testing samples

```

Using Gradient Boosting:

```python
# 1. Gradient Boosting
gb_model = GradientBoostingClassifier()
gb_model.fit(X_train, y_train, sample_weight=weights_train)
gb_pred = gb_model.predict(X_test)

print(f"Gradient Boosting- Performance Metrics:")

accuracy = accuracy_score(y_test, gb_pred)
print(f"Accuracy: {accuracy}")
```

Output:

```
Gradient Boosting- Performance Metrics:
Accuracy: 0.25
Confusion Matrix:
[[0 2]
 [1 1]]
Classification Report:
              precision    recall  f1-score   support

   No Disease       0.00      0.00      0.00         2
Heart Disease       0.33      0.50      0.40         2

     accuracy                           0.25         4
    macro avg       0.17      0.25      0.20         4
 weighted avg       0.17      0.25      0.20         4
```

Using AdaBoost:

```python
# 2. AdaBoost
ada_model = AdaBoostClassifier()
ada_model.fit(X_train, y_train, sample_weight=weights_train)
ada_pred = ada_model.predict(X_test)

print(f"AdaBoost Performance Metrics:")

accuracy = accuracy_score(y_test, ada_pred)
print(f"Accuracy: {accuracy}")


cm = confusion_matrix(y_test, ada_pred)
print("Confusion Matrix:")
print(cm)

report = classification_report(y_test, ada_pred, target_names=['No Disease', 'Heart Disease'])
print("Classification Report:")
print(report)
```

Output:

```
AdaBoost Performance Metrics:
Accuracy: 0.5
Confusion Matrix:
[[1 1]
 [1 1]]
Classification Report:
              precision    recall  f1-score   support

   No Disease       0.50      0.50      0.50         2
Heart Disease       0.50      0.50      0.50         2

     accuracy                           0.50         4
    macro avg       0.50      0.50      0.50         4
 weighted avg       0.50      0.50      0.50         4
```

Using XGBoost:

```python
# 3. XGBoost
xgb_model = XGBClassifier(use_label_encoder=False, eval_metric='logloss')
xgb_model.fit(X_train, y_train, sample_weight=weights_train)
xgb_pred = xgb_model.predict(X_test)


print(f"XGBoost - Performance Metrics:")

accuracy = accuracy_score(y_test, xgb_pred)
print(f"Accuracy: {accuracy}")


cm = confusion_matrix(y_test, xgb_pred)
print("Confusion Matrix:")
print(cm)

report = classification_report(y_test, xgb_pred, target_names=['No Disease', 'Heart Disease'])
print("Classification Report:")
print(report)
```

Output:

```
XGBoost - Performance Metrics:
Accuracy: 0.5
Confusion Matrix:
[[2 0]
 [2 0]]
Classification Report:
              precision    recall  f1-score   support

   No Disease       0.50      1.00      0.67         2
Heart Disease       0.00      0.00      0.00         2

     accuracy                           0.50         4
    macro avg       0.25      0.50      0.33         4
 weighted avg       0.25      0.50      0.33         4
```