TASK 3 - 210042112

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_auc_score, classification_report
```

```
def filter_binary_data(X, y):
    binary_filter = (y == 0) | (y == 1)
    return X[binary_filter], y[binary_filter]

# Load MNIST
from tensorflow.keras.datasets import mnist
(X_train_full, y_train_full), (X_test_full, y_test_full) = mnist.load_data()

# Filter only digits 0 and 1
X_train, y_train = filter_binary_data(X_train_full.reshape(-1, 28 * 28), y_train_full)
X_test, y_test = filter_binary_data(X_test_full.reshape(-1, 28 * 28), y_test_full)

# Normalize data
X_train = X_train / 255.0
X_test = X_test / 255.0
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11490434/11490434 ━━━━━━━━━━━━━━━━━━━━ 0s 0us/step
```

```
# Logistic Regression
log_reg = LogisticRegression()
log_reg.fit(X_train, y_train)
y_pred_lr = log_reg.predict(X_test)

# Decision Tree
dt = DecisionTreeClassifier()
dt.fit(X_train, y_train)
y_pred_dt = dt.predict(X_test)

# Naive Bayes
nb = GaussianNB()
nb.fit(X_train, y_train)
y_pred_nb = nb.predict(X_test)

# Metrics
metrics = {
    "Accuracy": accuracy_score,
    "Precision": lambda y_true, y_pred: precision_score(y_true, y_pred, average='binary'),
    "Recall": lambda y_true, y_pred: recall_score(y_true, y_pred, average='binary'),
    "F1 Score": lambda y_true, y_pred: f1_score(y_true, y_pred, average='binary'),
}

for name, model_preds in zip(["Logistic Regression", "Decision Tree", "Naive Bayes"], [y_pred_lr, y_pred_dt, y_pred_nb]):
    print(f"Metrics for {name}:")
    for metric_name, metric_func in metrics.items():
        print(f"  {metric_name}: {metric_func(y_test, model_preds):.4f}")
    print("\n")
```

```
Metrics for Logistic Regression:
  Accuracy: 0.9995
  Precision: 0.9991
  Recall: 1.0000
  F1 Score: 0.9996


Metrics for Decision Tree:
  Accuracy: 0.9939
  Precision: 0.9956
  Recall: 0.9930
  F1 Score: 0.9943


Metrics for Naive Bayes:
```

```
Accuracy: 0.9877
Precision: 0.9964
Recall: 0.9806
F1 Score: 0.9885
```

```
# Multi-Class Decision Tree
dt_multi = DecisionTreeClassifier()
dt_multi.fit(X_train_full.reshape(-1, 28 * 28) / 255.0, y_train_full)

# Predictions for multi-class
y_pred_multi = dt_multi.predict(X_test_full.reshape(-1, 28 * 28) / 255.0)

# Classification Report
print("Multi-Class Decision Tree Performance:")
print(classification_report(y_test_full, y_pred_multi))
```

```
Multi-Class Decision Tree Performance:
              precision    recall  f1-score   support

           0       0.92      0.93      0.92       980
           1       0.95      0.96      0.96      1135
           2       0.87      0.85      0.86      1032
           3       0.83      0.86      0.84      1010
           4       0.87      0.87      0.87       982
           5       0.84      0.83      0.84       892
           6       0.89      0.87      0.88       958
           7       0.91      0.91      0.91      1028
           8       0.82      0.82      0.82       974
           9       0.85      0.85      0.85      1009

    accuracy                           0.88     10000
   macro avg       0.88      0.87      0.87     10000
weighted avg       0.88      0.88      0.88     10000
```