Model Development Phase Template

Date	20 June 2025
Team ID	SWTID1749791625
Project Title	Smart Lender- Applicant Credibility Prediction for Loan Approval
Maximum Marks	4 Marks

Initial Model Training Code, Model Validation and Evaluation Report

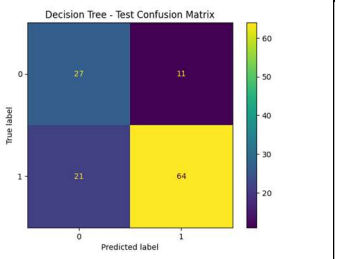
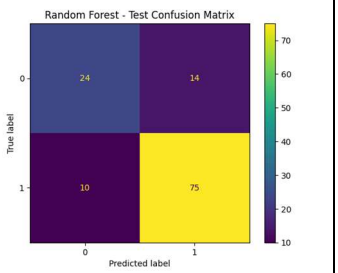
The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include classification reports, accuracy, and confusion matrices for multiple models, presented through respective screenshots.

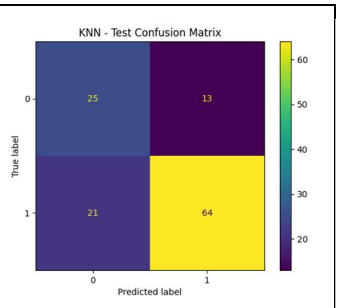
Initial Model Training Code:

```
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, ConfusionMatrixDisplay
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from xgboost import XGBClassifier
import matplotlib.pyplot as plt

# Model dictionary
models = {
    "Decision Tree": DecisionTreeClassifier(random_state=42),
    "Random Forest": RandomForestClassifier(random_state=42),
    "KNN": KNeighborsClassifier(),
    "XGBoost": XGBClassifier(use_label_encoder=False, eval_metric='logloss', random_state=42)
}

# Dictionary to store accuracy results
results = {}

# Loop through each model
for name, model in models.items():
    print(f"\n🔷 Training {name}...")

    # Train on SMOTE-balanced training data
    model.fit(X_train_final, y_train_final)

    # -------- Validation Performance --------
    val_preds = model.predict(X_val)
    print(f"\n📋 {name} - Validation Results:")
    print(classification_report(y_val, val_preds))
    ConfusionMatrixDisplay.from_predictions(y_val, val_preds).plot()
    plt.title(f"{name} - Validation Confusion Matrix")
    plt.show()
```

```
        # -------- Test Performance --------
        test_preds = model.predict(X_test_scaled)
        print(f"\n📄 {name} - Test Results:")
        print(classification_report(y_test, test_preds))
        ConfusionMatrixDisplay.from_predictions(y_test, test_preds).plot()
        plt.title(f"{name} - Test Confusion Matrix")
        plt.show()

        # Accuracy on test set
        acc = accuracy_score(y_test, test_preds)
        results[name] = acc
        print(f"✅ {name} Test Accuracy: {acc:.4f}")

    # -------- Compare All Accuracies --------
    print("\n📊 Final Model Accuracies on Test Set:")
    for name, score in results.items():
        print(f"{name}: {score:.4f}")

    # Bar Chart
    plt.bar(results.keys(), results.values(), color=['skyblue', 'orange', 'green', 'red'])
    plt.title("Model Accuracy Comparison (Test Set)")
    plt.ylabel("Accuracy")
    plt.ylim(0, 1)
    plt.xticks(rotation=15)
    plt.tight_layout()
    plt.show()
```

Model Validation and Evaluation Report:

Model	Classification Report	Accuracy	Confusion Matrix
Decision Tree	📄 Decision Tree - Test Results: precision recall f1-score support 0 0.56 0.71 0.63 38 1 0.85 0.75 0.80 85 accuracy 0.74 123 macro avg 0.71 0.73 0.71 123 weighted avg 0.76 0.74 0.75 123	74%	
Random Forest	📄 Random Forest - Test Results: precision recall f1-score support 0 0.71 0.63 0.67 38 1 0.84 0.88 0.86 85 accuracy 0.80 123 macro avg 0.77 0.76 0.76 123 weighted avg 0.80 0.80 0.80 123	80%	

KNN	KNN - Test Results: precision recall f1-score support 0 0.54 0.66 0.60 38 1 0.83 0.75 0.79 85 accuracy 0.72 123 macro avg 0.69 0.71 0.69 123 weighted avg 0.74 0.72 0.73 123	72%	
XGBoost	XGBoost - Test Results: precision recall f1-score support 0 0.64 0.71 0.68 38 1 0.86 0.82 0.84 85 accuracy 0.79 123 macro avg 0.75 0.77 0.76 123 weighted avg 0.80 0.79 0.79 123	79%