Model Development Phase Template

Date	24 June 2025
Team ID	SWTID1749708868
Project Title	Revolutionizing Liver Care : Predicting Liver Cirrhosis Using Advanced Machine Learning Techniques
Maximum Marks	4 Marks

Initial Model Training Code, Model Validation and Evaluation Report

Multiple classification models (Logistic Regression, SVM, KNN, Decision Tree, Random Forest) were trained to predict liver cirrhosis outcomes. Feature scaling was applied where necessary using StandardScaler. The target variable was encoded for compatibility. Models were evaluated using accuracy, weighted F1-score, confusion matrix, and classification report. This approach allowed performance comparison across algorithms and ensured robust validation of predictive capabilities.

Initial Model Training Code:

Train-Test Split

```
[ ] from sklearn.model_selection import train_test_split

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, stratify=y)
```

Feature-Scaling

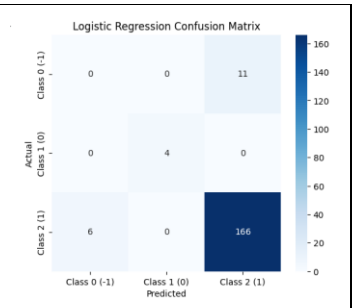
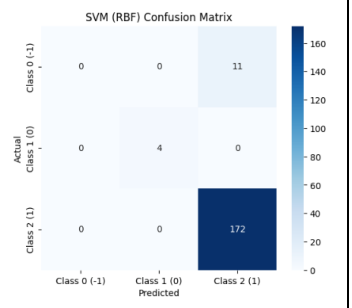
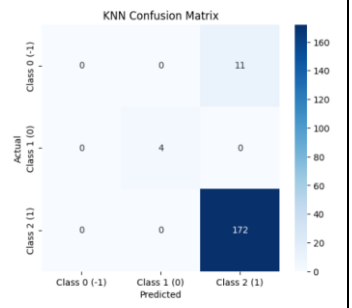
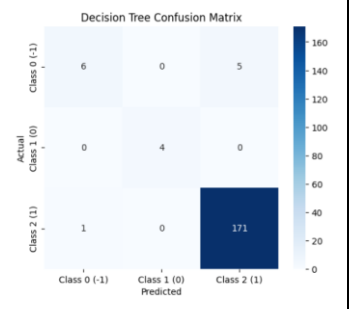
```
[ ] from sklearn.preprocessing import StandardScaler

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

Model Defining and Evaluation

```
from sklearn.metrics import accuracy_score, f1_score, confusion_matrix, classification_report
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
# Define the models
models = {
    "Logistic Regression": LogisticRegression(max_iter=1000),
    "SVM (RBF)": SVC(gamma='auto'),
    "KNN": KNeighborsClassifier(),
    "Decision Tree": DecisionTreeClassifier(),
    "Random Forest": RandomForestClassifier(),

}

# Map the target variable values from [-1, 0, 1] to [0, 1, 2]
y_train_mapped = y_train.map({-1: 0, 0: 1, 1: 2})
y_test_mapped = y_test.map({-1: 0, 0: 1, 1: 2})


for name, model in models.items():
    if name in ["Logistic Regression", "SVM (RBF)", "KNN"]:
        model.fit(X_train_scaled, y_train_mapped)
        y_pred = model.predict(X_test_scaled)
    else:
        model.fit(X_train, y_train_mapped)
        y_pred = model.predict(X_test)

    print(f" {name}")

    # For consistency with model training and prediction, using the mapped y_test is appropriate here
    print("Accuracy:", accuracy_score(y_test_mapped, y_pred))
    print("F1 Score:", f1_score(y_test_mapped, y_pred, average='weighted'))
    print("Confusion Matrix:\n", confusion_matrix(y_test_mapped, y_pred))
    print("Classification Report:\n", classification_report(y_test_mapped, y_pred))
    print("-" * 60)
```

Model Validation and Evaluation Report:

Model	Classification Report	Accuracy	Confusion Matrix

Logistic Regression	Classification Report: precision recall f1-score support 0 0.00 0.00 0.00 11 1 1.00 1.00 1.00 4 2 0.94 0.97 0.95 172 accuracy 0.91 187 macro avg 0.65 0.66 0.65 187 weighted avg 0.88 0.91 0.90 187	0.9090909090909091	
SVM (RBF)	Classification Report: precision recall f1-score support 0 0.00 0.00 0.00 11 1 1.00 1.00 1.00 4 2 0.94 1.00 0.97 172 accuracy 0.94 187 macro avg 0.65 0.67 0.66 187 weighted avg 0.89 0.94 0.91 187	0.9411764705882353	
KNN	Classification Report: precision recall f1-score support 0 0.00 0.00 0.00 11 1 1.00 1.00 1.00 4 2 0.94 1.00 0.97 172 accuracy 0.94 187 macro avg 0.65 0.67 0.66 187 weighted avg 0.89 0.94 0.91 187	0.9411764705882353	
Decision Tree	Classification Report: precision recall f1-score support 0 0.60 0.55 0.57 11 1 1.00 1.00 1.00 4 2 0.97 0.98 0.97 172 accuracy 0.95 187 macro avg 0.86 0.84 0.85 187 weighted avg 0.95 0.95 0.95 187	0.9518716577540107	
Random Forest	Classification Report: precision recall f1-score support 0 0.00 0.00 0.00 11 1 1.00 1.00 1.00 4 2 0.94 0.98 0.96 172 accuracy 0.93 187 macro avg 0.65 0.66 0.65 187 weighted avg 0.88 0.93 0.90 187	0.9251336898395722