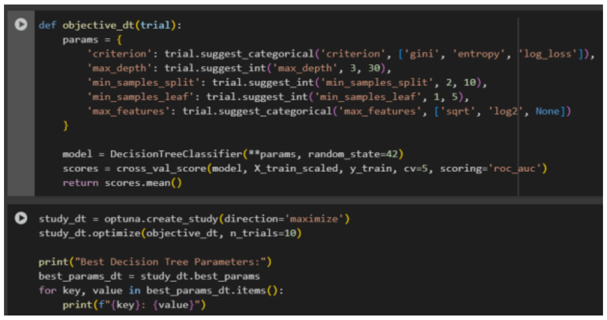
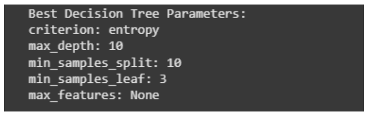
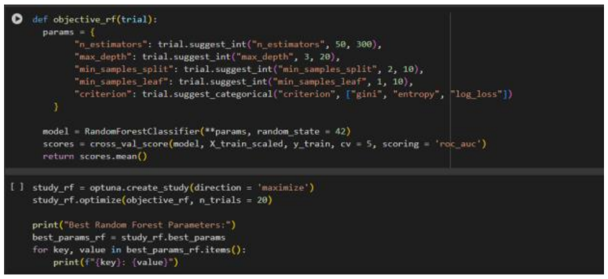
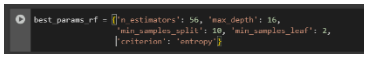
Model Optimization and Tuning Phase

Date	20 June 2025
Team ID	xxxxxx
Project Title	sloan digital sky survey (sdss) galaxy classification using machine learning
Maximum Marks	10 Marks

Model Optimization and Tuning Phase

The Model Optimization and Tuning Phase involves refining machine learning models for peak performance. It includes optimized model code, fine-tuning hyperparameters, comparing performance metrics, and justifying the final model selection for enhanced predictive accuracy and efficiency.

Hyperparameter Tuning Documentation (6 Marks):

Model	Tuned Hyperparameters	Optimal Values
Decision Tree Classifier		
Random Forest Classifier		

Decision Tree Classifier – Tuned Hyperparameters:

```
def objective_dt(trial):
    params = {
        'criterion': trial.suggest_categorical('criterion', ['gini', 'entropy', 'log_loss']),
        'max_depth': trial.suggest_int('max_depth', 3, 30),
        'min_samples_split': trial.suggest_int('min_samples_split', 2, 10),
        'min_samples_leaf': trial.suggest_int('min_samples_leaf', 1, 5),
        'max_features': trial.suggest_categorical('max_features', ['sqrt', 'log2', None])
    }

    model = DecisionTreeClassifier(**params, random_state=42)
    scores = cross_val_score(model, X_train_scaled, y_train, cv=5, scoring='roc_auc')
    return scores.mean()
```

```
study_dt = optuna.create_study(direction='maximize')
study_dt.optimize(objective_dt, n_trials=10)

print("Best Decision Tree Parameters:")
best_params_dt = study_dt.best_params
for key, value in best_params_dt.items():
    print(f"{key}: {value}")
```

Decision Tree Classifier – Optimal Values:

```
Best Decision Tree Parameters:
criterion: entropy
max_depth: 10
min_samples_split: 10
min_samples_leaf: 3
max_features: None
```

Random Forest Classifier – Tuned Hyperparameters:

```
def objective_rf(trial):
  params = {
        "n_estimators": trial.suggest_int("n_estimators", 50, 300),
        "max_depth": trial.suggest_int("max_depth", 3, 20),
        "min_samples_split": trial.suggest_int("min_samples_split", 2, 10),
        "min_samples_leaf": trial.suggest_int("min_samples_leaf", 1, 10),
        "criterion": trial.suggest_categorical("criterion", ["gini", "entropy", "log_loss"])
    }

  model = RandomForestClassifier(**params, random_state = 42)
  scores = cross_val_score(model, X_train_scaled, y_train, cv = 5, scoring = 'roc_auc')
  return scores.mean()
```

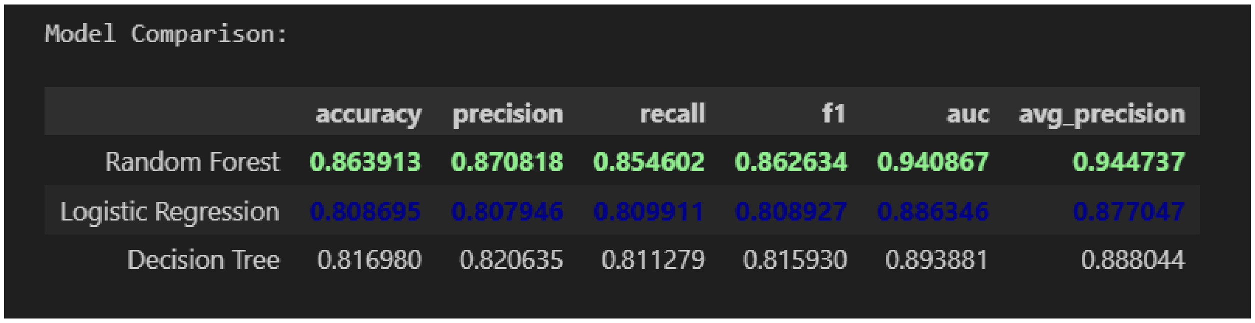
```
[ ] study_rf = optuna.create_study(direction = 'maximize')
    study_rf.optimize(objective_rf, n_trials = 20)

    print("Best Random Forest Parameters:")
    best_params_rf = study_rf.best_params
    for key, value in best_params_rf.items():
        print(f"{key}: {value}")
```

Random Forest Classifier – Optimal Values:

```
best_params_rf = {'n_estimators': 56, 'max_depth': 16,
                  'min_samples_split': 10, 'min_samples_leaf': 2,
                  'criterion': 'entropy'}
```

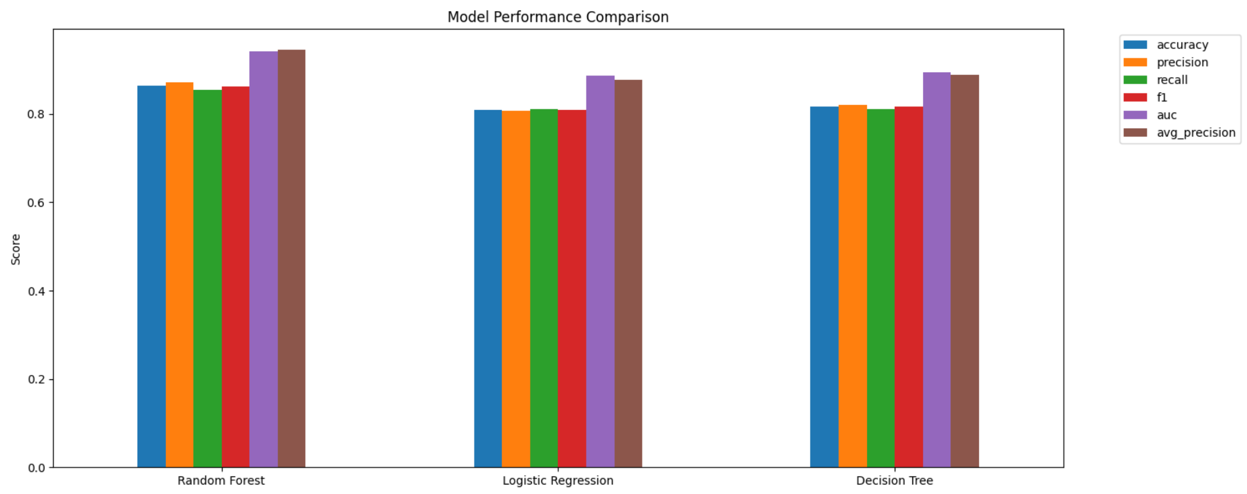
Performance Metrics Comparison Report (2 Marks):

Scenario 1:

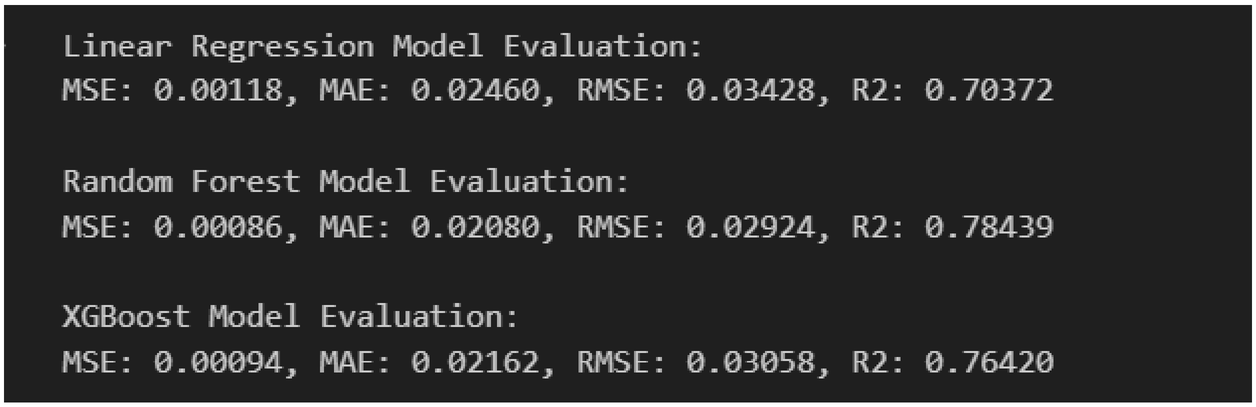


Model Comparison:

	accuracy	precision	recall	f1	auc	avg_precision
Random Forest	0.863913	0.870818	0.854602	0.862634	0.940867	0.944737
Logistic Regression	0.808695	0.807946	0.809911	0.808927	0.886346	0.877047
Decision Tree	0.816980	0.820635	0.811279	0.815930	0.893881	0.888044



Scenario 2:



```
Linear Regression Model Evaluation:
MSE: 0.00118, MAE: 0.02460, RMSE: 0.03428, R2: 0.70372

Random Forest Model Evaluation:
MSE: 0.00086, MAE: 0.02080, RMSE: 0.02924, R2: 0.78439

XGBoost Model Evaluation:
MSE: 0.00094, MAE: 0.02162, RMSE: 0.03058, R2: 0.76420
```

Final Model Selection Justification (2 Marks):

Final Model	**Reasoning**
Random Forest Classifier (**Scenario 1**)	I picked Random Forest because it consistently outperformed other models on all key metrics. It had the highest accuracy (0.864), precision (0.8708), recall (0.8546), F1 score (0.8626), AUC (0.9409), and average precision (0.9447). This means it balanced true positives and false positives better and provided more reliable predictions overall.
Random Forest Regressor (**Scenario 2**)	I chose Random Forest because it delivered the best overall error metrics. It had the lowest MSE (0.00086), MAE (0.02080), RMSE (0.02924) and the highest R^2 (0.78439), indicating it fit the data better than Linear Regression and XGBoost. The lower error and higher R^2 show it captured patterns in the data more effectively.