Model Development Phase Template

Date	8 July 2024
Team ID	739928
Project Title	Rhythmic Revenue: Unveiling The Future Of Music Sales With Machine Learning
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.preprocessing import StandardScaler
    scaler=StandardScaler()
    X=scaler.fit_transform(X)
```

```
[ ] X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=5)
```

```
[ ] from sklearn.tree import DecisionTreeRegressor
    dt_model=DecisionTreeRegressor(random_state=5)
    dt_model.fit(X_train,y_train)
```

```
▾ DecisionTreeRegressor
DecisionTreeRegressor(random_state=5)
```

```
y_pred_train=dt_model.predict(X_train)
train_score=r2_score(y_train,y_pred_train)
print("Training Accuracy:", train_score * 100,"%")

y_pred_test=dt_model.predict(X_test)
test_score_dtr=r2_score(y_test,y_pred_test)
print("Test Accuracy:", test_score_dtr * 100,"%")
```

```
Training Accuracy: 99.99999999958447 %
Test Accuracy: 94.89294925162281 %
```

```
[ ] from sklearn.ensemble import RandomForestRegressor
    rf_model=RandomForestRegressor( n_estimators=100,random_state=5)
    rf_model.fit(X_train,y_train)
```

```
▾ RandomForestRegressor
RandomForestRegressor(random_state=5)
```

```
[ ] y_pred_train=rf_model.predict(X_train)
    train_score=r2_score(y_train,y_pred_train)
    print("Training Accuracy:", train_score * 100,"%")

    y_pred_test=rf_model.predict(X_test)
    test_score_rf=r2_score(y_test,y_pred_test)
    print("Test Accuracy:", test_score_rf * 100,"%")
```

```
Training Accuracy: 99.26854202343576 %
Test Accuracy: 94.65594791365056 %
```

```
[ ] from sklearn.linear_model import LinearRegression
    model = LinearRegression()
    model.fit(X_train, y_train)
```

```
LinearRegression()
```

```
[ ] y_pred_train=model.predict(X_train) # Use 'model' instead of 'lr_model'
    train_score=r2_score(y_train,y_pred_train)
    print("Training Accuracy:", train_score * 100,"%")

    y_pred_test=model.predict(X_test)  # Use 'model' instead of 'lr_model'
    test_score_lr=r2_score(y_test,y_pred_test)
    print("Test Accuracy:", test_score_lr * 100,"%") # Print 'test_score_lr' instead of 'test_score_rf'
```

```
Training Accuracy: 18.33641886352857 %
Test Accuracy: 20.02269680287848 %
```

```
[ ] from sklearn.neighbors import KNeighborsRegressor
    knn_model=KNeighborsRegressor(n_neighbors=5)
    knn_model.fit(X_train,y_train)
```

```
KNeighborsRegressor()
```

```
y_pred_train=knn_model.predict(X_train)
train_score_knn=r2_score(y_train,y_pred_train)
print("Training Accuracy(KNN):", train_score_knn * 100,"%")

y_pred_test=knn_model.predict(X_test)
test_score_knn=r2_score(y_test,y_pred_test)
print("Test Accuracy(KNN):", test_score_knn * 100,"%")
```

```
Training Accuracy(KNN): 74.98212654710072 %
Test Accuracy(KNN): 60.03212174293324 %
```

```
[ ] import xgboost as xgb
    from xgboost import XGBRegressor
    xgboost_model = XGBRegressor()
    xgboost_model.fit(X_train, y_train)
```

```
XGBRegressor(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, device=None, early_stopping_rounds=None,
             enable_categorical=False, eval_metric=None, feature_types=None,
             gamma=None, grow_policy=None, importance_type=None,
             interaction_constraints=None, learning_rate=None, max_bin=None,
             max_cat_threshold=None, max_cat_to_onehot=None,
             max_delta_step=None, max_depth=None, max_leaves=None,
             min_child_weight=None, missing=nan, monotone_constraints=None,
             multi_strategy=None, n_estimators=None, n_jobs=None,
             num_parallel_tree=None, random_state=None, ...)
```

```
[ ] y_pred_train_xgboost = xgboost_model.predict(X_train)
    train_score_xgboost = r2_score(y_train,y_pred_train_xgboost)
    print("Training Accuracy(XGBoost):", train_score_xgboost * 100,"%")

    y_pred_test_xgboost = xgboost_model.predict(X_test)
    test_score_xgboost = r2_score(y_test,y_pred_test_xgboost)
    print("Test Accuracy(XGBoost):", test_score_xgboost * 100,"%")
```

```
Training Accuracy(XGBoost): 99.66071820984286 %
Test Accuracy(XGBoost): 94.60949318282277 %
```

Model Validation and Evaluation Report:

Model	Classification Report	Accuracy	Confusion Matrix
DecisionTreeRegression	Training Accuracy: 99.99999999958447 % Test Accuracy: 94.89294925162281 %	99% and 94%	-
Random forest regression	Training Accuracy: 99.26854202343576 % Test Accuracy: 94.65594791365056 %	99% and 94%	-
Linearregression	Training Accuracy: 18.33641886352857 % Test Accuracy: 20.02269680287848 %	18%and 20%	-
KNN	Training Accuracy(KNN): 74.98212654710072 % Test Accuracy(KNN): 60.03212174293324 %	74% and 60%	-
Xgbregression	Training Accuracy(XGBoost): 99.66071820984286 % Test Accuracy(XGBoost): 94.60949318282277 %	99% and 94%	-