


## Project Development Phase Model Performance Test

Date	20 November 2023
Team ID	591765
Project Name	Project - xxx
Maximum Marks	10 Marks

### Model Performance Testing:

Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Values	Screenshot
1.	Metrics	<div>Regression Model:</div> <div><div>#ModelAccuracyF1 ScoreRecallPrecision</div><div>+-----+-----+-----+-----+</div><div>-----+</div><div>0Logistic Regression64.2769.2267.7970.72</div><div>1Logistic Regression CV63.8269.1068.2569.97</div><div>2XGBoost64.0068.5566.1871.09</div><div>3Ridge Classifier65.2370.4569.9470.97</div><div>4KNN62.8668.1767.1069.28</div><div>5Random Forest65.5968.6563.5774.62</div><div>6Support Vector Machine67.0967.1256.6782.29</div></div>	

2.	Tune the Model	<div><div><div>Hyperparameter Tuning –</div><div>Logistic Regression- 0.64</div><div>Random forest - 0.68</div><div>Xgboost - 0.67</div><div>SVM - 0.61</div><div>Validation Method -</div><div>0.86</div></div><div><div>Hyperparameter optimization for XGBoost</div><pre>from sklearn.metrics import log_loss from sklearn.model_selection import GridSearchCV from sklearn.datasets import load_ionosphere from sklearn.preprocessing import StandardScaler from xgboost import XGBClassifier  X, y = load_ionosphere(return_X_y=True) scaler = StandardScaler() X = scaler.fit_transform(X)  param_grid = {     'max_depth': [3, 4, 5],     'min_child_weight': [1, 2, 3],     'learning_rate': [0.1, 0.05, 0.01] }  xgb_clf = XGBClassifier() grid_search = GridSearchCV(xgb_clf, param_grid, scoring='neg_log_loss', cv=5) grid_search.fit(X, y)  print('Best score: %f' % grid_search.best_score_) print('Best parameters: %s' % grid_search.best_params_)  Best score: -0.67000000 Best parameters: {'learning_rate': 0.01, 'max_depth': 5, 'min_child_weight': 1}</pre><div>Hyperparameter optimization for Logistic Regression</div><pre>from sklearn.metrics import log_loss from sklearn.model_selection import GridSearchCV from sklearn.datasets import load_ionosphere from sklearn.preprocessing import StandardScaler from sklearn.linear_model import LogisticRegression  X, y = load_ionosphere(return_X_y=True) scaler = StandardScaler() X = scaler.fit_transform(X)  param_grid = {     'C': [0.1, 1, 10],     'max_iter': [100, 200, 500] }  log_clf = LogisticRegression() grid_search = GridSearchCV(log_clf, param_grid, scoring='neg_log_loss', cv=5) grid_search.fit(X, y)  print('Best score: %f' % grid_search.best_score_) print('Best parameters: %s' % grid_search.best_params_)  Best score: -0.68000000 Best parameters: {'C': 1, 'max_iter': 500}</pre><div>Hyperparameter optimization for Random Forest</div><pre>from sklearn.metrics import log_loss from sklearn.model_selection import GridSearchCV from sklearn.datasets import load_ionosphere from sklearn.preprocessing import StandardScaler from sklearn.ensemble import RandomForestClassifier  X, y = load_ionosphere(return_X_y=True) scaler = StandardScaler() X = scaler.fit_transform(X)  param_grid = {     'n_estimators': [50, 100, 200],     'max_depth': [None, 10, 20, 30],     'min_samples_split': [2, 5, 10] }  rf_clf = RandomForestClassifier() grid_search = GridSearchCV(rf_clf, param_grid, scoring='neg_log_loss', cv=5) grid_search.fit(X, y)  print('Best score: %f' % grid_search.best_score_) print('Best parameters: %s' % grid_search.best_params_)  Best score: -0.68000000 Best parameters: {'max_depth': 10, 'min_samples_split': 2, 'n_estimators': 100}</pre><div>Hyperparameter optimization for SVM</div><pre>from sklearn.metrics import log_loss from sklearn.model_selection import GridSearchCV from sklearn.datasets import load_ionosphere from sklearn.preprocessing import StandardScaler from sklearn.svm import SVC  X, y = load_ionosphere(return_X_y=True) scaler = StandardScaler() X = scaler.fit_transform(X)  param_grid = {     'C': [0.1, 1, 10],     'gamma': [0.001, 0.01, 0.1] }  svm_clf = SVC() grid_search = GridSearchCV(svm_clf, param_grid, scoring='neg_log_loss', cv=5) grid_search.fit(X, y)  print('Best score: %f' % grid_search.best_score_) print('Best parameters: %s' % grid_search.best_params_)  Best score: -0.61000000 Best parameters: {'C': 1, 'gamma': 0.001}</pre></div></div>
----	----------------	---