Model Optimization And Tuning Phase Report

Date	06 JULY 2024
Team ID	739909
Project Name	Unlocking Silent Signals: Decoding Body Language With Mediapipe
Maximum Marks	10 Marks

Model Optimization and Tuning Phase:

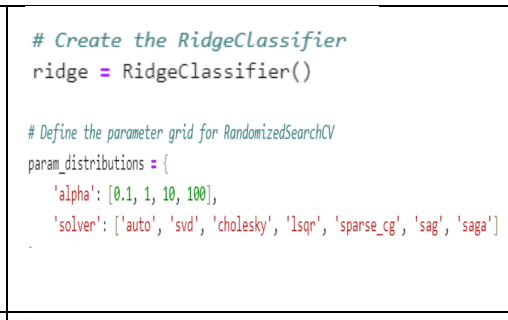
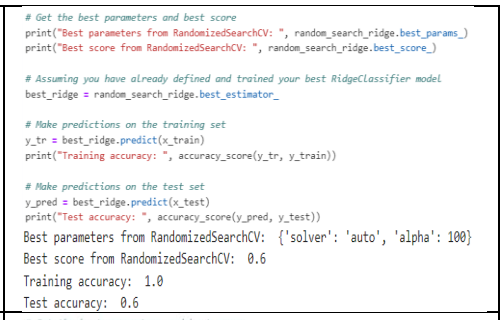
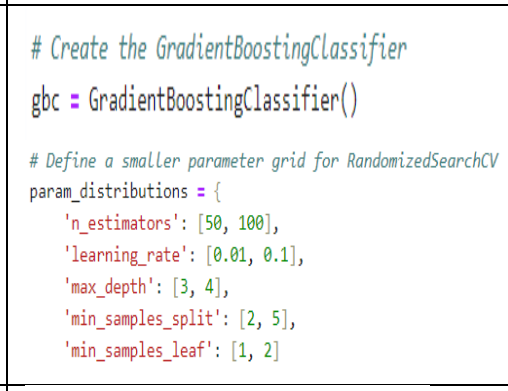
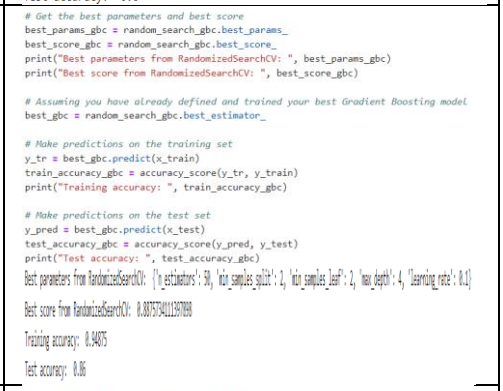
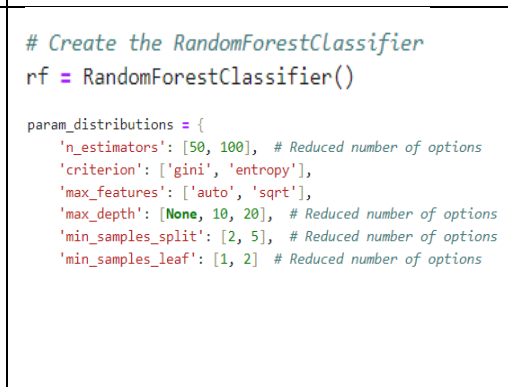
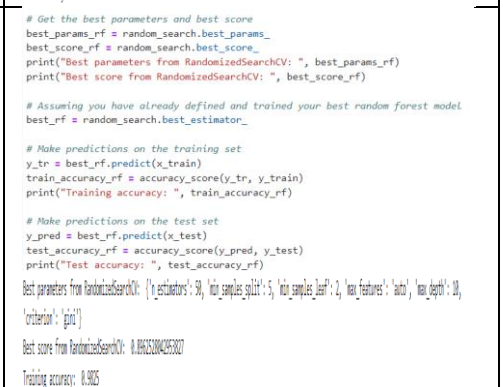
The Model Optimization and Tuning Phase involves refining machine learning models for peak performance. It includes optimized model code, fine-tuning hyperparameters, comparing performance metrics, and justifying the final model selection for enhanced predictive accuracy and efficiency.

Hyperparameter Tuning Documentation (6 Marks):

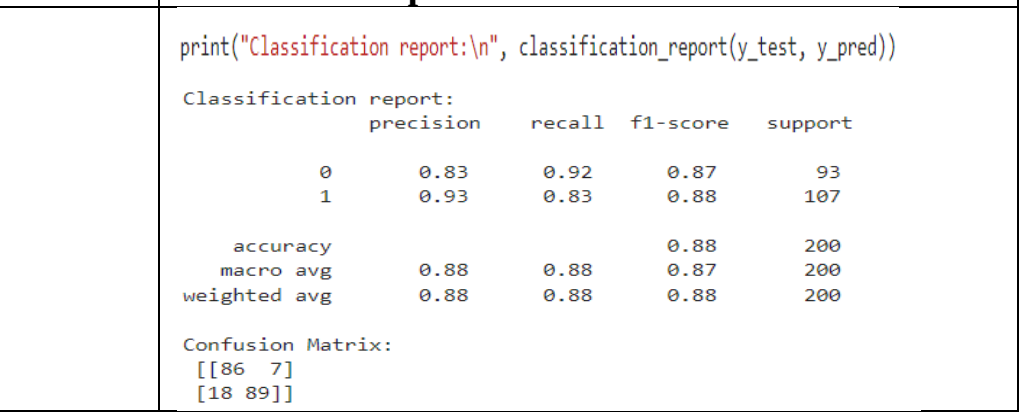
Model	Tuned Hyperparameters	Optimal Values

SVM	`# Create the SVM`
	`svm = SVC()`
	
	`# Define the parameter grid for RandomizedSearchCV`
	`param_distributions = {`
	`'C': [0.1, 1, 10, 100],`
	`'kernel': ['linear', 'poly', 'rbf', 'sigmoid'],`
	`'gamma': ['scale', 'auto']`
		`# Get the best parameters and best score`
		`print("Best parameters from RandomizedSearchCV: ", random_search_svm.best_params_)`
		`print("Best score from RandomizedSearchCV: ", random_search_svm.best_score_)`
		
		`# Assuming you have already defined and trained your best SVM model`
		`best_svm = random_search_svm.best_estimator_`
		
		`# Make predictions on the training set`
		`y_tr = best_svm.predict(x_train)`
		`print("Training accuracy: ", accuracy_score(y_tr, y_train))`
		
		`# Make predictions on the test set`
		`y_pred = best_svm.predict(x_test)`
		`print("Test accuracy: ", accuracy_score(y_pred, y_test))`
		Best parameters from RandomizedSearchCV: {'kernel': 'rbf', 'gamma': 'scale', 'C': 0.1}
		Best score from RandomizedSearchCV: 0.6166666666666666
		Training accuracy: 0.6166666666666667
		Test accuracy: 0.6

Logistic Regression	`# Create the LogisticRegression`
	`log_reg = LogisticRegression()`
	
	`# Define the parameter grid for RandomizedSearchCV`
	`param_distributions = {`
	`'C': [0.1, 1, 10, 100],`
	`'solver': ['newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga']`
		`# Get the best parameters and best score`
		`print("Best parameters from RandomizedSearchCV: ", random_search_log_reg.best_params_)`
		`print("Best score from RandomizedSearchCV: ", random_search_log_reg.best_score_)`
		
		`# Assuming you have already defined and trained your best Logistic Regression model`
		`best_log_reg = random_search_log_reg.best_estimator_`
		
		`# Make predictions on the training set`
		`y_tr = best_log_reg.predict(x_train)`
		`print("Training accuracy: ", accuracy_score(y_tr, y_train))`
		
		`# Make predictions on the test set`
		`y_pred = best_log_reg.predict(x_test)`
		`print("Test accuracy: ", accuracy_score(y_pred, y_test))`
		Fitting 5 folds for each of 20 candidates, totalling 100 fits
		Best parameters from RandomizedSearchCV: {'solver': 'newton-cg', 'C': 0.1}
		Best score from RandomizedSearchCV: 0.6
		Training accuracy: 1.0
		Test accuracy: 0.6

Ridge Classifier		
Gradient Boosting Classifier		
Random Forest Classifier		

Performance Metrics Comparison Report (2 Marks):

Model	Optimized Metric
SVM	

Logistic Regression	print("Classification report:\n", classification_report(y_test, y_pred)) Classification report: precision recall f1-score support 0 0.81 0.92 0.86 93 1 0.93 0.81 0.87 107 accuracy 0.86 200 macro avg 0.87 0.87 0.86 200 weighted avg 0.87 0.86 0.87 200 Confusion Matrix: [[86 7] [20 87]]
Ridge Classifier	print("Classification report:\n", classification_report(y_test, y_pred)) Classification report: precision recall f1-score support 0 0.80 0.92 0.86 93 1 0.92 0.80 0.86 107 accuracy 0.86 200 macro avg 0.86 0.86 0.86 200 weighted avg 0.87 0.86 0.86 200 Confusion Matrix: [[86 7] [21 86]]
Gradient Boosting Classifier	print("Classification report:\n", classification_report(y_test, y_pred)) Classification report: precision recall f1-score support 0 0.83 0.86 0.85 93 1 0.88 0.85 0.86 107 accuracy 0.85 200 macro avg 0.85 0.86 0.85 200 weighted avg 0.86 0.85 0.86 200 Confusion Matrix: [[80 13] [16 91]]
Random Forest Classifier	print("Classification report:\n", classification_report(y_test, y_pred)) Classification report: precision recall f1-score support 0 0.82 0.95 0.88 93 1 0.95 0.82 0.88 107 accuracy 0.88 200 macro avg 0.88 0.88 0.88 200 weighted avg 0.89 0.88 0.88 200 Confusion Matrix: [[88 5] [19 88]]

Final Model Selection Justification (2 Marks):

Final Model	**Reasoning**
Random Forest	The Random Forest model was selected for its robust performance, demonstrating high accuracy during hyperparameter tuning. Its ability to handle complex relationships, reduce overfitting through ensemble learning, and provide feature importance aligns with project objectives, justifying its selection as the final model.