Model Development Phase Template

Date	15 March 2024
Team ID	738214
Project Title	Predicting Mental Health Illness Of WorkingProfessionals Using Machine Learning.
Maximum Marks	6 Marks

Model Selection Report

In the forthcoming Model Selection Report, various models will be outlined, detailing their descriptions, hyperparameters, and performance metrics, including Accuracy or F1 Score. This comprehensive report will provide insights into the chosen models and their effectiveness.

Model Selection Report:

Model	Description	Hyperparameters	Performance Metric (e.g., Accuracy, F1 Score)
Logistic Regression	Logistic Regression is a linear classification algorithm used for binary classification problems. It models the probability that a given input belongs to a particular class using the logistic function.	Best Hyper parameters : Regularization parameter 'C': 10 maximum number of iterations 'max_iter': 100 Penalty Type 'penalty': 'l1' Algorithm for optimization 'solver': 'liblinear'	Accuracy : 74.93 % After Hypertuning : 75.2 %

Kneighbors Classifier	K-Nearest Neighbors (KNN) is a non-parametric algorithm used for both classification and regression tasks. In classification, it assigns the class label by a majority vote of its k-nearest neighbors. KNN is instance-based and lazy, meaning it does not explicitly learn a model during training. Instead, it memorizes the training instances and uses them for prediction.	Best Hyper Parameters : Distance metric 'metric': 'manhattan' Number of neighbours 'n_neighbors': 9 Weight function used in prediction 'weights': 'distance'	Accuracy : 65.07 % After Hypertuning : 70.13%
Decision Tree Classifier	Decision Trees are versatile algorithms capable of performing both classification and regression tasks. They partition the feature space into regions, where each region corresponds to a specific class label. At each node of the tree, a decision is made based on a feature value to split	Best Hyper parameters : Maximum depth of the tree 'max_depth': 5 Minimum number of samples required to split an internal node 'min_samples_split': 15 Minimum number of samples required to be at a leaf node 'min_samples_leaf': 1 Number of features to consider at each split	Accuracy : 69.87 % After Hypertuning : 73.6 %

	the data into two or more child nodes.	'max_features': None Split criterion 'criterion': 'entropy'	
Random Forest Classifier	Random Forest is an ensemble learning method that builds multiple decision trees during training and combines their predictions for better accuracy and generalization. Each tree in the forest is trained on a random subset of the training data and a random subset of the features. Random Forest reduces overfitting and provides more robust predictions compared to individual decision trees.	Best Hyper paramters : Number of trees in the forest 'n_estimators': 400 Maximum depth of the trees 'max_depth': None Minimum number of samples required to split an internal node 'min_samples_split': 10 Minimum number of samples required to be at a leaf node 'min_samples_leaf': 8 Number of features to consider at each split 'max_features': 'sqrt' Split criterion 'criterion': 'entropy'}	Accuracy : 76.8 % After Hypertuning : 78.93%
AdaBoost Classifier	AdaBoost (Adaptive Boosting) is an ensemble learning method that combines multiple weak learners (typically decision trees) to build a strong classifier. It iteratively trains weak learners on the	Best Hyper paramters : Base Estimator 'base_estimator': DecisionTreeClassifier(max_depth=2) Learning Rate 'learning_rate': 0.01 Number of estimators (base models) 'n_estimators': 200	Accuracy : 78.67 % After Hypertuning : 78.93%

	data, focusing more on instances that are misclassified in previous iterations.		
Gradient Boosting Classifier	Gradient Boosting is another ensemble learning method that builds a strong learner by sequentially adding weak learners (typically decision trees) to correct the errors made by the previous models. It optimizes a loss function by using gradient descent to minimize the errors. Gradient Boosting typically uses shallow trees as weak learners and can handle both regression and classification tasks.	Best Hyper Parameters : Number of boosting stages 'n_estimators': 100 Learning rate 'learning_rate': 0.01 Maximum depth of the individual estimators 'max_depth': 5 Minimum number of samples required to split a node 'min_samples_leaf': 2 Minimum number of samples required to be at a leaf node 'min_samples_split': 2 Fraction of samples used for fitting the individual base learners 'subsample': 0.5	Accuracy : 78.4 % After Hypertuning : 78.93%
XGB Classifier	XGBoost (Extreme Gradient Boosting) is an optimized implementation of gradient boosting. It offers several	Best Hyper parameters Number of boosting rounds 'n_estimators': 50 Learning rate 'learning_rate': 0.01	Accuracy : 72.53 % After Hypertuning : 79.47%

	enhancements over traditional gradient boosting, including handling missing values, regularization, and parallel processing. XGBoost is widely used in machine learning competitions and is known for its efficiency, speed, and high performance.	Maximum tree depth 'max_depth': 5 Minimum sum of instance weight (hessian) needed in a child 'min_child_weight': 1 Subsample ratio of the training instances 'subsample': 0.6 Subsample ratio of columns when constructing each tree 'colsample_bytree': 1.0	