Model Optimization and Tuning Phase Template

Date	15 july 2024
Team ID	739849
Project Title	Doctor’s Salary prediction
Maximum Marks	10 Marks

Model Optimization and Tuning Phase

The Model Optimization and Tuning Phase involves refining neural network 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 (8 Marks):

Model	**Tuned Hyperparameters**
Linear Regression	The chosen hyperparameters (regularization strength and solver) were fine-tuned to improve the model's ability to generalize from the data. Regularization helps in preventing overfitting by penalizing large coefficients, while the solver 'liblinear' was selected for its efficiency with smaller datasets. • **Regularization Strength (alpha)**: Set to 0.1 to prevent overfitting. • **Solver**: 'liblinear' chosen for its efficiency with small datasets. • The model uses Ridge Regression, which is a type of linear regression with L2 regularization. • **Evaluation Metrics**: Mean Squared Error (MSE) and R-squared (R^2) are used to assess the model's performance.

Random Forest regression	Random Forest Regression is a supervised machine learning algorithm that uses an ensemble of decision trees to predict continuous target variables. • **Ensemble Learning**: Combines multiple decision trees to improve predictive performance. • **Bagging Technique**: Each tree is trained on a random subset of the data, which helps in reducing variance. • **Parallel Processing**: Trees are built independently, allowing for efficient computation.
Decision tree regressor	A Decision Tree Regressor is a supervised machine learning algorithm used for predicting continuous target variables. It works by splitting the data into subsets based on the value of input features, creating a tree-like model of decisions. Each internal node represents a feature, each branch represents a decision rule, and each leaf node represents a predicted outcome. • **intuitive and Easy to Interpret**: The tree structure makes it easy to understand and visualize. • **Handles Both Numerical and Categorical Data**: Can work with different types of data without much preprocessing. • **Non-Linear Relationships**: Capable of capturing complex relationships between features and the target variable.
XGBoost	The `XGBRegressor` is a powerful machine learning algorithm from the XGBoost library, designed for regression tasks. It implements the gradient boosting framework, which builds an ensemble of decision trees to improve predictive accuracy and control over-fitting. • **Efficiency**: XGBoost is known for its speed and performance, making it suitable for large datasets. • **Regularization**: Includes L1 and L2 regularization to prevent overfitting. • **Handling Missing Values**: Automatically handles missing data during training. • **Parallel Processing**: Utilizes multiple CPU cores for faster computation.

Final Model Selection Justification (2 Marks):

Final Model	**Reasoning**
Random Forest regression	From the above metrics, compare the Validation R-squared and Validation MSE to determine the best model. Ideally, you want the model with the highest R-squared and the lowest MSE on the validation data, while avoiding overfitting (i.e., training and validation metrics should be relatively close). Based on your code and typical performance, it seems like the Random Forest or XGBRegressor might offer the best balance between training and validation performance. Metrics are vere closely and best validation performance **Training MSE**: 458713655.5555556 **Validation MSE**: 3631440587.5 **R-squared**: 0.289059314547212