Model Optimization and Tuning Phase Template

Date	24 April 2024
Team ID	739877
Project Title	Crystal Ball Analysis: Projecting Share Prices Of The Leading Gpu Titans
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	Imports necessary libraries and tools for data handling, model training, and evaluation.Defines `param_grid` with hyperparameters for `LinearRegression`: `fit_intercept`, `positive`, `copy_X`.Uses GridSearchCV (`grid_search`) to find the best model configuration based on 5-fold cross-validation.Evaluates the best model (`best_lr`) on test data, computing Mean Squared Error and R-squared metrics.
	
	Hyperparameter Tuning
	
	``python
	# Import necessary libraries
	import pandas as pd
	from sklearn.model_selection import train_test_split, GridSearchCV
	from sklearn.linear_model import LinearRegression # Import LinearRegression
	from sklearn.metrics import mean_squared_error, r2_score # Import appropriate metrics
	
	param_grid = {
	    'fit_intercept': [True, False],
	    'positive': [True, False],
	    'copy_X': [True, False]
	} # Adjusted hyperparameters for LinearRegression
	
	grid_search = GridSearchCV(estimator=lr, param_grid=param_grid, cv=5, n_jobs=-1, verbose=2)
	grid_search.fit(x_train, y_train)
	
	# Best parameters and best estimator
	best_params = grid_search.best_params_
	best_lr = grid_search.best_estimator_
	
	# Predictions and evaluation
	y_pred = best_lr.predict(x_test)
	
	# Evaluation metrics for regression
	print("Best Parameters:", best_params)
	print("Mean Squared Error:", mean_squared_error(y_test, y_pred))
	print("R-squared:", r2_score(y_test, y_pred))
	``
	
	``
	Fitting 5 folds for each of 8 candidates, totalling 40 fits
	Best Parameters: {'copy_X': True, 'fit_intercept': False, 'positive': True}
	Mean Squared Error: 2.3123559886068095
	R-squared: 0.9997722332806338
	``

Decision Tree	The parameters (params) define a grid for hyperparameter tuning of the Decision Tree Classifier (DecisionTreeClassifier), including max_depth, min_samples_leaf, and criterion ('gini' or 'entropy'). GridSearchCV (dt_model) is used with 5-fold cross-validation (cv=5), evaluating model performance based on accuracy (scoring="accuracy")
	
	``python
	# Import necessary libraries
	import pandas as pd
	from sklearn.model_selection import train_test_split, GridSearchCV
	from sklearn.tree import DecisionTreeRegressor # Import DecisionTreeRegressor
	from sklearn.metrics import mean_squared_error, r2_score # Import appropriate metrics
	
	# Assuming x_train, x_test, y_train, y_test are already defined from train_test_split
	
	# Define the parameter grid for GridSearchCV with DecisionTreeRegressor
	param_grid = {
	    'max_depth': [None, 10, 20, 30],
	    'min_samples_split': [2, 5, 10],
	    'min_samples_leaf': [1, 2, 4]
	}
	
	# Create a DecisionTreeRegressor instance
	
	# Instantiate GridSearchCV with the decision tree regressor and parameter grid
	grid_search = GridSearchCV(estimator=det, param_grid=param_grid, cv=5, n_jobs=-1, verbose=2)
	
	# Fit GridSearchCV to find the best model parameters
	grid_search.fit(x_train, y_train)
	
	# Get the best parameters and best estimator found by GridSearchCV
	best_params = grid_search.best_params_
	best_dt = grid_search.best_estimator_
	
	# Make predictions using the best model
	y_pred = best_dt.predict(x_test)
	
	# Evaluate the model performance using metrics for regression
	print("Best Parameters:", best_params)
	print("Mean Squared Error:", mean_squared_error(y_test, y_pred))
	print("R-squared:", r2_score(y_test, y_pred))
	``
	
	``
	Fitting 5 folds for each of 36 candidates, totalling 180 fits
	Best Parameters: {'max_depth': 10, 'min_samples_leaf': 2, 'min_samples_split': 5}
	Mean Squared Error: 22.242800736487503
	R-squared: 0.9978090874509689
	``
	
	Fitting 5 folds for each of 36 candidates, totalling 180 fits
	Best Parameters: {'max_depth': 10, 'min_samples_leaf': 2, 'min_samples_split': 5}
	Mean Squared Error: 22.242800736487503
	R-squared: 0.9978090874509689

Final Model Selection Justification (2 Marks):

Final Model	**Reasoning**
Linear Regression	Linear regression model is chosen for its robustness in handling complex datasets and its ability to mitigate overfitting while providing high predictive accuracy. ``` Fitting 5 folds for each of 8 candidates, totalling 40 fits Best Parameters: {'copy_X': True, 'fit_intercept': False, 'positive': True} Mean Squared Error: 2.3123559886068095 R-squared: 0.9997722332806338 ``` Above two models Linear regression model have the highest accuracy among the models.