Model Development Phase Template

Date	6th July 2024
Team ID	739719
Project Title	Garment Workers Productivity Predictions
Maximum Marks	10 Marks

Initial Model Training Code, Model Validation and Evaluation Report

The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include a summary and training and validation performance metrics for multiple models, presented through respective screenshots.

Initial Model Training Code (5 marks):

```
df = pd.read_csv(r'C:\Users\srira\Downloads\miniProject\garments_worker_productivity.csv')
df.head()
```

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.30, random_state=42)
```

```
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(823, 12)
(353, 12)
(823,)
(353,)
```

```
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
from math import sqrt
from sklearn.metrics import mean_absolute_percentage_error
```

Model Validation and Evaluation Report (5 marks)

	Summary	Training and Validation Performance Metrics

Model 1

Random Forest Regressor

Random Forest Regressor Summary

Model Parameters

- **Number of Trees**: Optimal number of trees determined through hyperparameter tuning.
- **Max Depth**: Maximum depth of the trees, optimized to prevent overfitting.
- **Min Samples Split**: Minimum number of samples required to split an internal node.
- **Min Samples Leaf**: Minimum number of samples required to be at a leaf node.

Training Process:

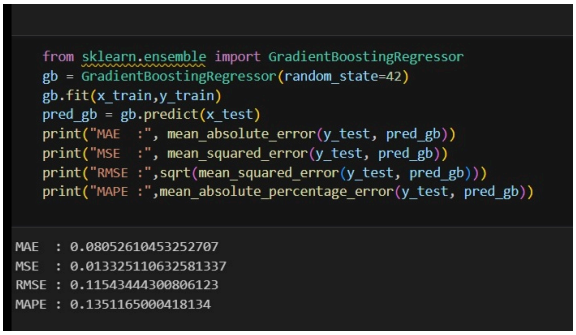
- **Data Preprocessing**: Standardized or normalized input features.
- **Bootstrapping**: Random sampling with replacement to create multiple training sets for the trees.
- **Feature Selection**: Random selection of features at each split to ensure diverse trees.

Evaluation Metrics:

- **Mean Absolute Error (MAE)**: Measures the average magnitude of the errors in the predictions.
- **Mean Squared Error (MSE)**: Measures the average of the squares of the errors, penalizing larger errors.
- **R^2 Score**: Indicates the proportion of the variance in the dependent variable that is predictable from the independent variables.

```
from sklearn.ensemble import RandomForestRegressor
randf = RandomForestRegressor(random_state=42)
randf.fit(x_train,y_train)
pred_randf = randf.predict(x_test)
print("MAE  :", mean_absolute_error(y_test, pred_randf))
print("MSE  :", mean_squared_error(y_test, pred_randf))
print("RMSE :",sqrt(mean_squared_error(y_test, pred_randf)))
print("MAPE :",mean_absolute_percentage_error(y_test, pred_randf))
```

```
MAE  : 0.08366785595438364
MSE  : 0.015441874867015823
RMSE : 0.12426534057015183
MAPE : 0.14067390864389964
```

Model 2
 	
Gradient Boosting Regressor
	Gradient Boosting Regressor Summary
	
	Model Parameters:
	
	• **Number of Estimators**: Total number of boosting stages (trees).
	• **Learning Rate**: Shrinks the contribution of each tree.
	• **Max Depth**: Maximum depth of the individual regression estimators (trees).
	• **Min Samples Split**: Minimum number of samples required to split an internal node.
	• **Min Samples Leaf**: Minimum number of samples required to be at a leaf node.
	• **Subsample**: Fraction of samples used for fitting the individual base learners.
	Training Process:
	
	• **Data Preprocessing**: Standardized or normalized input features.
	• **Initialization**: Starts with an initial prediction, often the mean of the target values.
	• **Sequential Training**: Each tree is trained on the residuals of the previous trees' predictions.
	• **Loss Function**: Mean Squared Error (MSE) to minimize the difference between predicted and actual values.
	Evaluation Metrics:
	
	• **Mean Absolute Error (MAE)**: Average magnitude of the errors in the predictions.
	• **Mean Squared Error (MSE)**: Average of the squares of the errors, penalizing larger errors.
	• **R^2 Score**: Proportion of the variance in the dependent variable that is predictable from the independent variables.
		
		`from sklearn.ensemble import GradientBoostingRegressor`
		`gb = GradientBoostingRegressor(random_state=42)`
		`gb.fit(x_train,y_train)`
		`pred_gb = gb.predict(x_test)`
		`print("MAE :", mean_absolute_error(y_test, pred_gb))`
		`print("MSE :", mean_squared_error(y_test, pred_gb))`
		`print("RMSE :",sqrt(mean_squared_error(y_test, pred_gb)))`
		`print("MAPE :",mean_absolute_percentage_error(y_test, pred_gb))`
		
		`MAE : 0.08052610453252707`
		`MSE : 0.013325110632581337`
		`RMSE : 0.11543444300806123`
		`MAPE : 0.1351165000418134`
