Model Development Phase Template

Date	5th July 2024
Team ID	SWTID1720673861
Project Title	Garment Worker Efficiency Calculator
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.

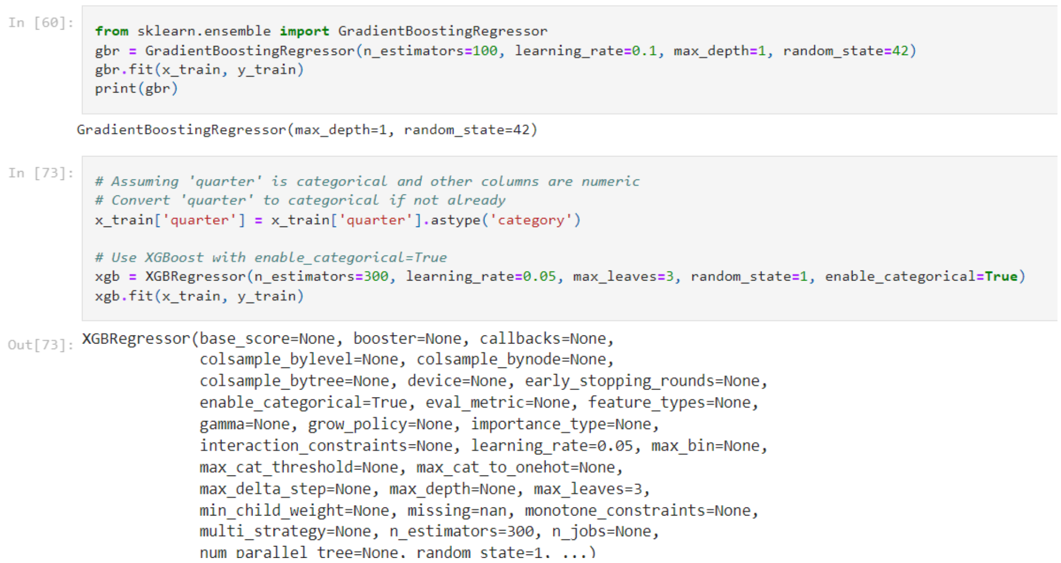
1) Gradient Boosting Regressor Model

The code is fitting a Gradient Boosting Regressor model to the training data X_train and y_train using n_estimators equal to 100 (the number of trees in the forest), learning_rate equal to 0.1 (the step size shrinkage used to prevent overfitting), max_depth equal to 1 (the maximum depth of the individual regression estimators), and random_state equal to 42 (to ensure reproducibility of results).

The fitted model can then be used to make predictions on new data using the predict() method.

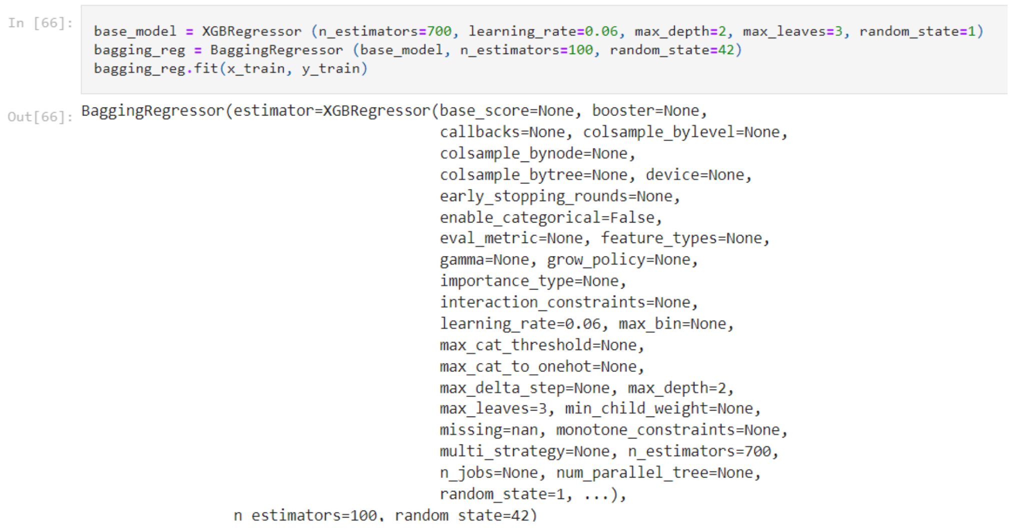
2) Extreme Gradient Boost Regressor Model

This code is training an XGBoost regression model. The XGBRegressor function is being used to create the model. The parameters passed to the function include the number of estimators, learning rate, maximum number of leaves, and random state. The model is trained on the training set (X_train and y_train) using the fit() method. Once trained, the model can be used to make predictions on new data.

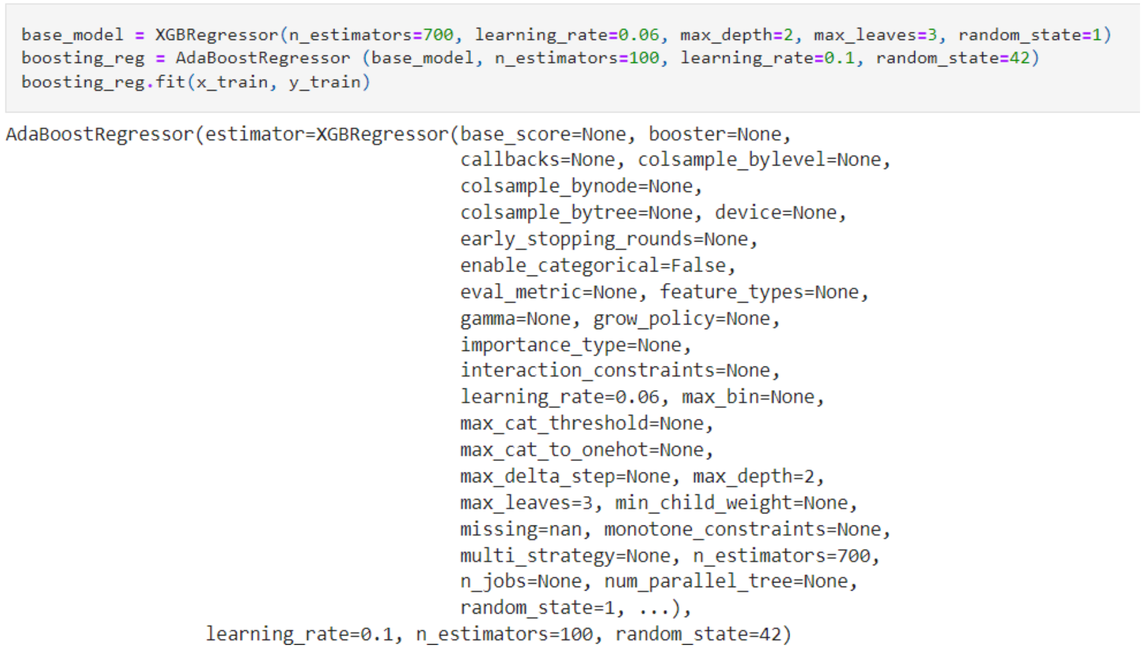


```
In [60]:  from sklearn.ensemble import GradientBoostingRegressor
          gbr = GradientBoostingRegressor(n_estimators=100, learning_rate=0.1, max_depth=1, random_state=42)
          gbr.fit(x_train, y_train)
          print(gbr)

          GradientBoostingRegressor(max_depth=1, random_state=42)
```

```
In [73]:  # Assuming 'quarter' is categorical and other columns are numeric
          # Convert 'quarter' to categorical if not already
          x_train['quarter'] = x_train['quarter'].astype('category')

          # Use XGBoost with enable_categorical=True
          xgb = XGBRegressor(n_estimators=300, learning_rate=0.05, max_leaves=3, random_state=1, enable_categorical=True)
          xgb.fit(x_train, y_train)

Out[73]:  XGBRegressor(base_score=None, booster=None, callbacks=None,
                       colsample_bylevel=None, colsample_bynode=None,
                       colsample_bytree=None, device=None, early_stopping_rounds=None,
                       enable_categorical=True, eval_metric=None, feature_types=None,
                       gamma=None, grow_policy=None, importance_type=None,
                       interaction_constraints=None, learning_rate=0.05, max_bin=None,
                       max_cat_threshold=None, max_cat_to_onehot=None,
                       max_delta_step=None, max_depth=None, max_leaves=3,
                       min_child_weight=None, missing=nan, monotone_constraints=None,
                       multi_strategy=None, n_estimators=300, n_jobs=None,
                       num parallel tree=None. random state=1. ...)
```



```
In [66]:  base_model = XGBRegressor (n_estimators=700, learning_rate=0.06, max_depth=2, max_leaves=3, random_state=1)
          bagging_reg = BaggingRegressor (base_model, n_estimators=100, random_state=42)
          bagging_reg.fit(x_train, y_train)

Out[66]:  BaggingRegressor(estimator=XGBRegressor(base_score=None, booster=None,
                                                  callbacks=None, colsample_bylevel=None,
                                                  colsample_bynode=None,
                                                  colsample_bytree=None, device=None,
                                                  early_stopping_rounds=None,
                                                  enable_categorical=False,
                                                  eval_metric=None, feature_types=None,
                                                  gamma=None, grow_policy=None,
                                                  importance_type=None,
                                                  interaction_constraints=None,
                                                  learning_rate=0.06, max_bin=None,
                                                  max_cat_threshold=None,
                                                  max_cat_to_onehot=None,
                                                  max_delta_step=None, max_depth=2,
                                                  max_leaves=3, min_child_weight=None,
                                                  missing=nan, monotone_constraints=None,
                                                  multi_strategy=None, n_estimators=700,
                                                  n_jobs=None, num_parallel_tree=None,
                                                  random_state=1, ...),
                            n estimators=100. random state=42)
```



```
base_model = XGBRegressor(n_estimators=700, learning_rate=0.06, max_depth=2, max_leaves=3, random_state=1)
boosting_reg = AdaBoostRegressor (base_model, n_estimators=100, learning_rate=0.1, random_state=42)
boosting_reg.fit(x_train, y_train)

AdaBoostRegressor(estimator=XGBRegressor(base_score=None, booster=None,
                                          callbacks=None, colsample_bylevel=None,
                                          colsample_bynode=None,
                                          colsample_bytree=None, device=None,
                                          early_stopping_rounds=None,
                                          enable_categorical=False,
                                          eval_metric=None, feature_types=None,
                                          gamma=None, grow_policy=None,
                                          importance_type=None,
                                          interaction_constraints=None,
                                          learning_rate=0.06, max_bin=None,
                                          max_cat_threshold=None,
                                          max_cat_to_onehot=None,
                                          max_delta_step=None, max_depth=2,
                                          max_leaves=3, min_child_weight=None,
                                          missing=nan, monotone_constraints=None,
                                          multi_strategy=None, n_estimators=700,
                                          n_jobs=None, num_parallel_tree=None,
                                          random_state=1, ...),
                  learning_rate=0.1, n_estimators=100, random_state=42)
```