Model Development Phase Template

Date	15 July 2024
Team ID	SWTID1720108643
Project Title	Garment Worker Predictivity Prediction
Maximum Marks	4 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 classification reports, accuracy, and confusion matrices for multiple models, presented through respective screenshots.

Initial Model Training Code:

Linear regression

```
regressor_lr = LinearRegression()

regressor_lr.fit(x_train, y_train)

y_pred = regressor_lr.predict(x_test)

mse = mean_squared_error(y_test, y_pred)
rmse_lr =np.sqrt(mse)
print(f"Linear Regression - Root Mean Squared Error: {rmse_lr}")

mse_train = mean_squared_error(y_test, y_pred)
r2_lr = r2_score(y_test, y_pred)
print(f"Linear Regression - R^2 Score: {r2_lr}")
```

Decision Tree Regressor

```
dt = DecisionTreeRegressor()

param_grid = {
    'criterion': ['squared_error', 'friedman_mse', 'poisson', 'absolute_error'],
    'splitter': ['best', 'random'],
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
    'max_features': [None, 'sqrt', 'log2', 0.5, 1],
}

grid_search = GridSearchCV(estimator=dt, param_grid=param_grid, cv=5, n_jobs=-1, verbose=2)
grid_search.fit(x_train, y_train)

print(f"Best Parameters: {grid_search.best_params_}")
print(f"Best Score: {grid_search.best_score_}")

dtr = grid_search.best_estimator_
y_pred = dtr.predict(x_test)

mse = mean_squared_error(y_test, y_pred)

rmse_dtr =np.sqrt(mse)
print(f"DecisionTreeRegressor - Root Mean Squared Error: {rmse_dtr}")
```

Ada Boost Regressor

```
ada_boost = AdaBoostRegressor(random_state=42)

param_grid = {
    'n_estimators': [50, 100, 200, 300],
    'learning_rate': [0.001,0.01, 0.1, 1.0]
}

grid_search = GridSearchCV(estimator=ada_boost, param_grid=param_grid, cv=5, scoring='neg_mean_squared_error', n_jobs=-1)
grid_search.fit(x_train, y_train)

abr = grid_search.best_estimator_
y_pred = abr.predict(x_test)

print(f'Best Parameters: {grid_search.best_params_}')

mse = mean_squared_error(y_test, y_pred)
rmse_abr = np.sqrt(mse)
print(f'AdaBoostRegressor - Root Mean Squared Error: {rmse_abr}')
print(f"AdaBoostRegressor - R^2 Score: {r2_abr}")
```

Gradient Boosting Regressor

```
gb = GradientBoostingRegressor()

param_grid = {
    'n_estimators': [100, 200, 300],
    'learning_rate': [0.001, 0.01, 0.05],
    'max_depth': [3, 4, 5, 6],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
    'max_features': [ 'sqrt', 'log2']
}


grid_search = GridSearchCV(estimator=gb, param_grid=param_grid,
                           cv=5, n_jobs=-1, verbose=2)
grid_search.fit(x_train, y_train)

print("Best parameters found: ", grid_search.best_params_)
print("Best cross-validation score: ", grid_search.best_score_)


gbr = grid_search.best_estimator_
y_pred = gbr.predict(x_test)

mse = mean_squared_error(y_test, y_pred)

rmse_gbr =np.sqrt(mse)
print(f"GradientBoostingRegressor - Root Mean Squared Error: {rmse_gbr}")

r2_gbr = r2_score(y_test, y_pred)
print(f"GradientBoostingRegressor - R^2 Score: {r2_gbr}")
```

Random Forest Regressor

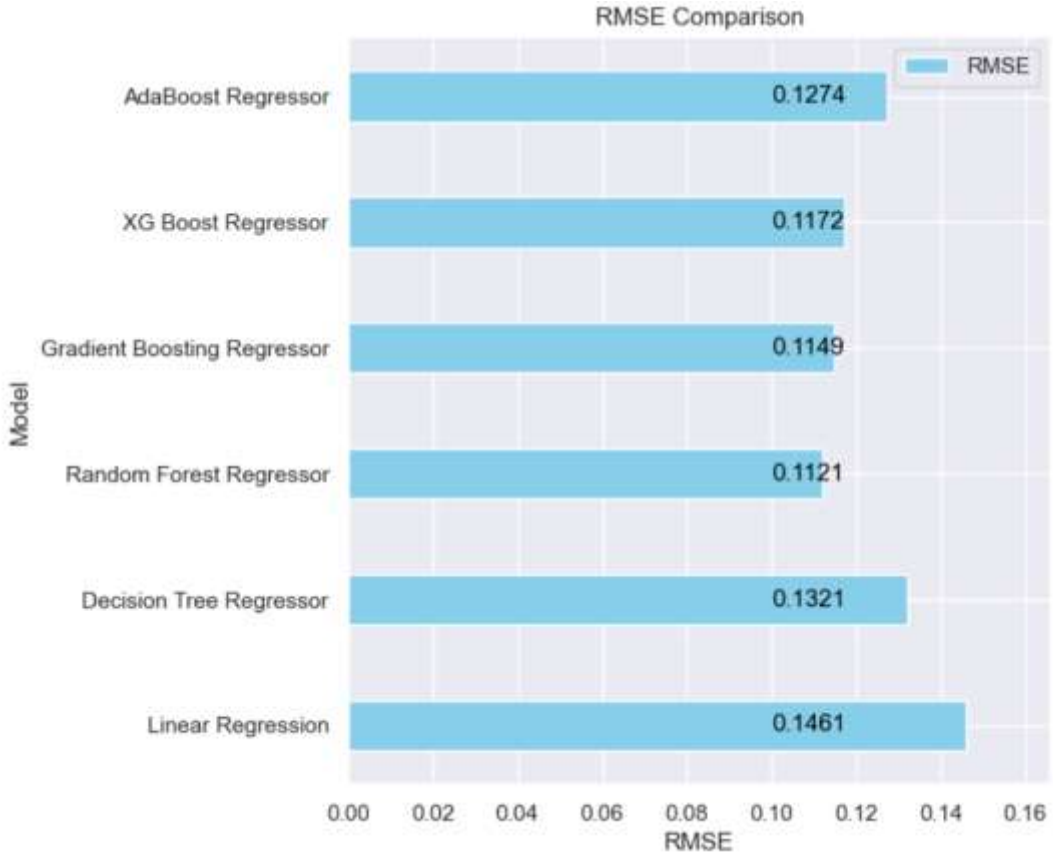
```
rfr = RandomForestRegressor()

param_grid = {
    'n_estimators': [100, 200, 300],
    'max_features': [ 'sqrt', 'log2', None],
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
    'bootstrap': [True, False]
}

grid_search = GridSearchCV(estimator=rfr, param_grid=param_grid,
                           cv=5, n_jobs=-1, verbose=2)

grid_search.fit(x_train, y_train)

print("Best parameters found: ", grid_search.best_params_)
print("Best cross-validation score: ", grid_search.best_score_)

rf = grid_search.best_estimator_
y_pred = rf.predict(x_test)

mse = mean_squared_error(y_test, y_pred)

rmse_rf =np.sqrt(mse)
print(f"RandomForestRegressor - Root Mean Squared Error: {rmse_rf}")
```

XGB Regressor

```
xg = XGBRegressor(objective='reg:squarederror')


param_grid = {
    'n_estimators': [100, 200, 300],
    'learning_rate': [0.001, 0.0, 0.1],
    'max_depth': [3, 4, 5, 6],
    'min_child_weight': [1, 3, 5],
    'subsample': [0.6, 0.8, 1.0],
    'colsample_bytree': [0.6, 0.8, 1.0]
}


grid_search = GridSearchCV(estimator=xg, param_grid=param_grid,
                           cv=5, n_jobs=-1, verbose=2)
grid_search.fit(x_train, y_train)

print("Best parameters found: ", grid_search.best_params_)
print("Best cross-validation score: ", grid_search.best_score_)


xgb = grid_search.best_estimator_
y_pred = xgb.predict(x_test)

mse = mean_squared_error(y_test, y_pred)

rmse_xgb =np.sqrt(mse)
print(f"XGBRegressor - Root Mean Squared Error: {rmse_xgb}")

r2_xgb = r2_score(y_test, y_pred)
print(f"XGBRegressor - R^2 Score: {r2_xgb}")
```

Model Validation and Evaluation Report :

Model	Root Mean Squared Error	R2 Score
Linear Regressor	0.14607	0.19
Decision Tree Regressor	0.13213	0.386
Random Forest Regressor	0.11212	0.52
Gradient Boosting Regressor	0.11485	0.50
XGB Regressor	0.11716	0.48
AdaBoost Regressor	0.12741	0.388


RMSE Comparison
RMSE
AdaBoost Regressor
0.1274
XG Boost Regressor
0.1172
Gradient Boosting Regressor
0.1149
Random Forest Regressor
0.1121
Decision Tree Regressor
0.1321
Linear Regression
0.1461
Model
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
0.16
RMSE