Model Development Phase Template

Date	11 July 2024
Team ID	SWTID1720455879
Project Title	Human Resource Management: Predicting Employee Promotions Using Machine Learning
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:

```
#Importing the models from sklearn
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from xgboost import XGBClassifier
from sklearn.model_selection import cross_val_score

# Initialize models
dt_model = DecisionTreeClassifier(random_state=42)
rf_model = RandomForestClassifier(random_state=42)
knn_model = KNeighborsClassifier()
xgb_model = XGBClassifier(random_state=42)

# Train the models
dt_model.fit(X_train, y_train)
rf_model.fit(X_train, y_train)
knn_model.fit(X_train, y_train)
xgb_model.fit(X_train, y_train)
```

```python
# Evaluate each model
models = {
    'Decision Tree': dt_model,
    'Random Forest': rf_model,
    'KNN': knn_model,
    'XGBoost': xgb_model
}

for model_name, model in models.items():
    y_pred = model.predict(X_test)
    print(f"Evaluation for {model_name}:")
    print(confusion_matrix(y_test, y_pred))
    print(classification_report(y_test, y_pred))
    print("="*80)
```

Model Validation and Evaluation Report:

Model	Classification Report	Accuracy	Confusion Matrix

Decision Tree	`print(classification_report(y_test, y_pred))`
	precision recall f1-score support
	0 0.93 0.93 0.93 15180
	1 0.93 0.93 0.93 14904
	accuracy 0.93 30084
	macro avg 0.93 0.93 0.93 30084
	weighted avg 0.93 0.93 0.93 30084
		`# Decision Tree Classifier`
		`dt_pred = dt_model.predict(X_test)`
		`dt_accuracy = accuracy_score(y_test, dt_pred)`
		`print(f"Decision Tree Classifier Accuracy: {dt_accuracy:.4f}")`
		Decision Tree Classifier Accuracy: 0.9333
			`print(confusion_matrix(y_test, y_pred))`
			[[14187 993]
			[1064 13840]]

Random Forest	`print(classification_report(y_test, y_pred))`
	precision recall f1-score support
	0 0.94 0.96 0.95 15180
	1 0.96 0.94 0.95 14904
	accuracy 0.95 30084
	macro avg 0.95 0.95 0.95 30084
	weighted avg 0.95 0.95 0.95 30084
		`# Random Forest Classifier`
		`rf_pred = rf_model.predict(X_test)`
		`rf_accuracy = accuracy_score(y_test, rf_pred)`
		`print(f"Random Forest Classifier Accuracy: {rf_accuracy:.4f}")`
		Random Forest Classifier Accuracy: 0.9506
			`print(confusion_matrix(y_test, y_pred))`
			[[14552 628]
			[877 14027]]

KNN	`print(classification_report(y_test, y_pred))`	`# K-Nearest Neighbors Classifier`
		`knn_pred = knn_model.predict(X_test)`
		`knn_accuracy = accuracy_score(y_test, knn_pred)`
		`print(f"K-Nearest Neighbors Classifier Accuracy: {knn_accuracy:.4f}")`
		K-Nearest Neighbors Classifier Accuracy: 0.8924
			`print(confusion_matrix(y_test, y_pred))`
			[[12406 2774]
			[439 14465]]

	precision recall f1-score support 0 0.97 0.82 0.89 15180 1 0.84 0.97 0.90 14904 accuracy 0.89 30084 macro avg 0.90 0.89 0.89 30084 weighted avg 0.90 0.89 0.89 30084		

XGBoost	`print(classification_report(y_test, y_pred))`
	
	precision recall f1-score support
	
	0 0.91 0.98 0.94 15180
	1 0.98 0.90 0.94 14904
	
	accuracy 0.94 30084
	macro avg 0.94 0.94 0.94 30084
	weighted avg 0.94 0.94 0.94 30084
		`# XGBoost Classifier`
		`xgb_pred = xgb_model.predict(X_test)`
		`xgb_accuracy = accuracy_score(y_test, xgb_pred)`
		`print(f"XGBoost Classifier Accuracy: {xgb_accuracy:.4f}")`
		
		XGBoost Classifier Accuracy: 0.9416
			`print(confusion_matrix(y_test, y_pred))`
			
			[[14863 317]
			[1421 13483]]