Model Development Phase Template

Date	17 July 2024
Team ID	xxxxxx
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:

Descision Tree Model

```
def decisionTree(X_train, X_test, y_train, y_test):
    # Initialize the DecisionTreeClassifier
    model = DecisionTreeClassifier(random_state=42)

    # Fit the model on the training data
    model.fit(X_train, y_train)

    # Make predictions on the test data
    y_pred = model.predict(X_test)

    # Evaluate the model
    cm = confusion_matrix(y_test, y_pred)
    cr = classification_report(y_test, y_pred)
    accuracy = accuracy_score(y_test, y_pred)

    print("Confusion Matrix:")
    print(cm)
    print("\nClassification Report:")
    print(cr)
    print(f"Accuracy: {accuracy:.2f}")

    return model

# Call the function with training and testing data
decisionTree(x_train, x_test, y_train, y_test)
```

Random Forest Model

```python
def randomForest(X_train, X_test, y_train, y_test):
    # Initialize the RandomForestClassifier
    model = RandomForestClassifier(random_state=42, n_estimators=100)
    # Fit the model on the training data
    model.fit(X_train, y_train)

    # Make predictions on the test data
    y_pred = model.predict(X_test)

    # Evaluate the model
    cm = confusion_matrix(y_test, y_pred)
    cr = classification_report(y_test, y_pred)
    accuracy = accuracy_score(y_test, y_pred)
    print("Confusion Matrix:")
    print(cm)
    print("\nClassification Report:")
    print(cr)
    print(f"Accuracy: {accuracy:.2f}")

    return model
# Call the function with training and testing data
randomForest(x_train, x_test, y_train, y_test)
```

KNN Model

```python
# Function to train and evaluate a KNN model
def KNN(X_train, X_test, y_train, y_test):
    # Initialize the KNeighborsClassifier
    model = KNeighborsClassifier(n_neighbors=5)  # You can adjust the number of neighbors (k) as needed
    # Fit the model on the training data
    model.fit(X_train, y_train)

    # Make predictions on the test data
    y_pred = model.predict(X_test)

    # Evaluate the model
    cm = confusion_matrix(y_test, y_pred)
    cr = classification_report(y_test, y_pred)
    accuracy = accuracy_score(y_test, y_pred)
    print("Confusion Matrix:")
    print(cm)
    print("\nClassification Report:")
    print(cr)
    print(f"Accuracy: {accuracy:.2f}")
    return model
# Call the function with training and testing data
KNN(x_train, x_test, y_train, y_test)
```

XGboost Model

```
def xgboost(X_train, X_test, y_train, y_test):
    # Initialize the GradientBoostingClassifier
    model = GradientBoostingClassifier(random_state=42)
    # Fit the model on the training data
    model.fit(X_train, y_train)

    # Make predictions on the test data
    y_pred = model.predict(X_test)

    # Evaluate the model
    cm = confusion_matrix(y_test, y_pred)
    cr = classification_report(y_test, y_pred)
    accuracy = accuracy_score(y_test, y_pred)
    print("Confusion Matrix:")
    print(cm)
    print("\nClassification Report:")
    print(cr)
    print(f"Accuracy: {accuracy:.2f}")
    return model

# Call the function with training and testing data
xgboost(x_train, x_test, y_train, y_test)
```

Model Validation and Evaluation Report:

Model	Classification Report	Accuracy	Confusion Matrix
Decision Tree	Classification Report: precision recall f1-score support 0 0.94 0.92 0.93 15065 1 0.92 0.94 0.93 15019 accuracy 0.93 30084 macro avg 0.93 0.93 0.93 30084 weighted avg 0.93 0.93 0.93 30084 Accuracy: 0.93	93%	Confusion Matrix: [[13875 1190] [902 14117]]
Random Forest	Classification Report: precision recall f1-score support 0 0.95 0.94 0.95 15065 1 0.94 0.95 0.95 15019 accuracy 0.95 30084 macro avg 0.95 0.95 0.95 30084 weighted avg 0.95 0.95 0.95 30084 Accuracy: 0.95	95%	Confusion Matrix: [[14195 870] [748 14271]]
KNN	Classification Report: precision recall f1-score support 0 0.96 0.81 0.88 15065 1 0.84 0.97 0.90 15019 accuracy 0.89 30084 macro avg 0.90 0.89 0.89 30084 weighted avg 0.90 0.89 0.89 30084 Accuracy: 0.89	89%	Confusion Matrix: [[12266 2799] [476 14543]]
Xgboost	Classification Report: precision recall f1-score support 0 0.88 0.84 0.86 15065 1 0.85 0.89 0.87 15019 accuracy 0.86 30084 macro avg 0.86 0.86 0.86 30084 weighted avg 0.86 0.86 0.86 30084 Accuracy: 0.86	86%	Confusion Matrix: [[12631 2434] [1669 13350]]