Step 1: Setting Up Environment

```
!pip install pandas numpy scikit-learn matplotlib seaborn xgboost nltk spacy
```

```
Requirement already satisfied: xgboost in /usr/local/lib/python3.10/dist-packages (2.1.1)
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages (3.8.1)
Requirement already satisfied: spacy in /usr/local/lib/python3.10/dist-packages (3.7.5)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas) (2024.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.10/dist-packages (from pandas) (2024.2)
Requirement already satisfied: scipy>=1.6.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.13.1)
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (3.5.0)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.3.0)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (4.54.1)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.4.7)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (24.1)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (10.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (3.2.0)
Requirement already satisfied: nvidia-nccl-cu12 in /usr/local/lib/python3.10/dist-packages (from xgboost) (2.23.4)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from nltk) (8.1.7)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.10/dist-packages (from nltk) (2024.9.11)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from nltk) (4.66.5)
Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.11 in /usr/local/lib/python3.10/dist-packages (from spacy) (3.0.12)
Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from spacy) (1.0.5)
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in /usr/local/lib/python3.10/dist-packages (from spacy) (1.0.10)
Requirement already satisfied: cymem<2.1.0,>=2.0.2 in /usr/local/lib/python3.10/dist-packages (from spacy) (2.0.8)
Requirement already satisfied: preshed<3.1.0,>=3.0.2 in /usr/local/lib/python3.10/dist-packages (from spacy) (3.0.9)
Requirement already satisfied: thinc<8.3.0,>=8.2.2 in /usr/local/lib/python3.10/dist-packages (from spacy) (8.2.5)
Requirement already satisfied: wasabi<1.2.0,>=0.9.1 in /usr/local/lib/python3.10/dist-packages (from spacy) (1.1.3)
Requirement already satisfied: srsly<3.0.0,>=2.4.3 in /usr/local/lib/python3.10/dist-packages (from spacy) (2.4.8)
Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in /usr/local/lib/python3.10/dist-packages (from spacy) (2.0.10)
Requirement already satisfied: weasel<0.5.0,>=0.1.0 in /usr/local/lib/python3.10/dist-packages (from spacy) (0.4.1)
Requirement already satisfied: typer<1.0.0,>=0.3.0 in /usr/local/lib/python3.10/dist-packages (from spacy) (0.12.5)
Requirement already satisfied: requests<3.0.0,>=2.13.0 in /usr/local/lib/python3.10/dist-packages (from spacy) (2.32.3)
Requirement already satisfied: pydantic!=1.8,!=1.8.1,<3.0.0,>=1.7.4 in /usr/local/lib/python3.10/dist-packages (from spacy) (2.9
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from spacy) (3.1.4)
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from spacy) (75.1.0)
Requirement already satisfied: langcodes<4.0.0,>=3.2.0 in /usr/local/lib/python3.10/dist-packages (from spacy) (3.4.1)
Requirement already satisfied: language-data>=1.2 in /usr/local/lib/python3.10/dist-packages (from langcodes<4.0.0,>=3.2.0->spac
Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python3.10/dist-packages (from pydantic!=1.8,!=1.8.1,<3.
Requirement already satisfied: pydantic-core==2.23.4 in /usr/local/lib/python3.10/dist-packages (from pydantic!=1.8,!=1.8.1,<3.0
Requirement already satisfied: typing-extensions>=4.6.1 in /usr/local/lib/python3.10/dist-packages (from pydantic!=1.8,!=1.8.1,<
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.2->pandas) (1.16.0
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0,>=2.13.0
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0,>=2.13.0->spacy) (3.
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0,>=2.13.0->spac
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0,>=2.13.0->spac
Requirement already satisfied: blis<0.8.0,>=0.7.8 in /usr/local/lib/python3.10/dist-packages (from thinc<8.3.0,>=8.2.2->spacy) (
Requirement already satisfied: confection<1.0.0,>=0.0.1 in /usr/local/lib/python3.10/dist-packages (from thinc<8.3.0,>=8.2.2->sp
Requirement already satisfied: shellingham>=1.3.0 in /usr/local/lib/python3.10/dist-packages (from typer<1.0.0,>=0.3.0->spacy) (
Requirement already satisfied: rich>=10.11.0 in /usr/local/lib/python3.10/dist-packages (from typer<1.0.0,>=0.3.0->spacy) (13.9.
Requirement already satisfied: cloudpathlib<1.0.0,>=0.7.0 in /usr/local/lib/python3.10/dist-packages (from weasel<0.5.0,>=0.1.0-
Requirement already satisfied: smart-open<8.0.0,>=5.2.1 in /usr/local/lib/python3.10/dist-packages (from weasel<0.5.0,>=0.1.0->s
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->spacy) (3.0.2)
Requirement already satisfied: marisa-trie>=0.7.7 in /usr/local/lib/python3.10/dist-packages (from language-data>=1.2->langcodes
Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python3.10/dist-packages (from rich>=10.11.0->typer<1.0.0
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/python3.10/dist-packages (from rich>=10.11.0->typer<1.0
Requirement already satisfied: wrapt in /usr/local/lib/python3.10/dist-packages (from smart-open<8.0.0,>=5.2.1->weasel<0.5.0,>=0
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.10/dist-packages (from markdown-it-py>=2.2.0->rich>=10.11.0-
```

Double-click (or enter) to edit

Step 2: Load and Prepare the Data

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
```

```
df = pd.read_csv('/content/swiggy - swiggy.csv - swiggy - swiggy.csv.csv')
df.head()
```

	ID	Area	City	Restaurant	Price	Avg ratings	Total ratings	Food type	Address	Delivery time
0	211	Koramangala	Bangalore	Tandoor Hut	300	4.4	100	Biryani,Chinese,North Indian,South Indian	5Th Block	59
1	221	Koramangala	Bangalore	Tunday Kababi	300	4.1	100	Mughlai,Lucknowi	5Th Block	56
2	246	Jogupalya	Bangalore	Kim Lee	650	4.4	100	Chinese	Double	50

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
df.isnull().sum()
```

	0
ID	0
Area	0
City	0
Restaurant	0
Price	0
Avg ratings	0
Total ratings	0
Food type	0
Address	0
Delivery time	0

```
le = LabelEncoder()    #Convert categorical variables like 'City', 'Food Type', and 'Area' into numerical values
df['City'] = le.fit_transform(df['City'])
df['Food Type'] = le.fit_transform(df['Food type'])
df['Area'] = le.fit_transform(df['Area'])
```

Step 3: Model 1 - Restaurant Rating Prediction (Regression Model)

```
X = df[['Price', 'City', 'Area', 'Delivery time']]
y = df['Avg ratings']


#Split the dataset for training and testing
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)


#Standardize the features to bring them to a similar scale
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)


from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(X_train, y_train)
```

```
LinearRegression()
```

```
#Predict and calculate the R-squared score
y_pred = model.predict(X_test)
from sklearn.metrics import r2_score, mean_squared_error
print("R-squared:", r2_score(y_test, y_pred))
print("MSE:", mean_squared_error(y_test, y_pred))
```

```
R-squared: 0.04750157327840199
MSE: 0.3945204225446974
```

```
print(df.columns)
```

```
Index(['ID', 'Area', 'City', 'Restaurant', 'Price', 'Avg ratings',
       'Total ratings', 'Food type', 'Address', 'Delivery time', 'Food Type'],
      dtype='object')
```

Step 4: Model 2 - Customer Churn Prediction (Classification Model)

```
import numpy as np

df['Churn'] = np.where((df['Avg ratings'] < 2.5) & (df['Delivery time'] > 60), 1, 0)
```

```
X = df[['Price', 'City', 'Area', 'Avg ratings', 'Delivery time']]
y = df['Churn']
```

```
#Same as before, split and scale the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

```
#Fit a Random Forest model
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(n_estimators=100, random_state=42)
classifier.fit(X_train, y_train)
```

```
RandomForestClassifier
RandomForestClassifier(random_state=42)
```

```
#Predict and evaluate the accuracy
y_pred = classifier.predict(X_test)
from sklearn.metrics import accuracy_score, classification_report
print("Accuracy:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
Accuracy: 0.9994239631336406
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      1735
           1       0.00      0.00      0.00         1

    accuracy                           1.00      1736
   macro avg       0.50      0.50      0.50      1736
weighted avg       1.00      1.00      1.00      1736

/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1531: UndefinedMetricWarning: Precision is ill-defined an
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1531: UndefinedMetricWarning: Precision is ill-defined an
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1531: UndefinedMetricWarning: Precision is ill-defined an
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

Step 5: Model 3 - Cuisine Preference Prediction (Multiclass Classification)

```
#Select Features and Target
X = df[['Price', 'City', 'Area']]
y = df['Food Type']
```

```
#Use the XGBoost classifier
from xgboost import XGBClassifier
xgb_model = XGBClassifier()
```

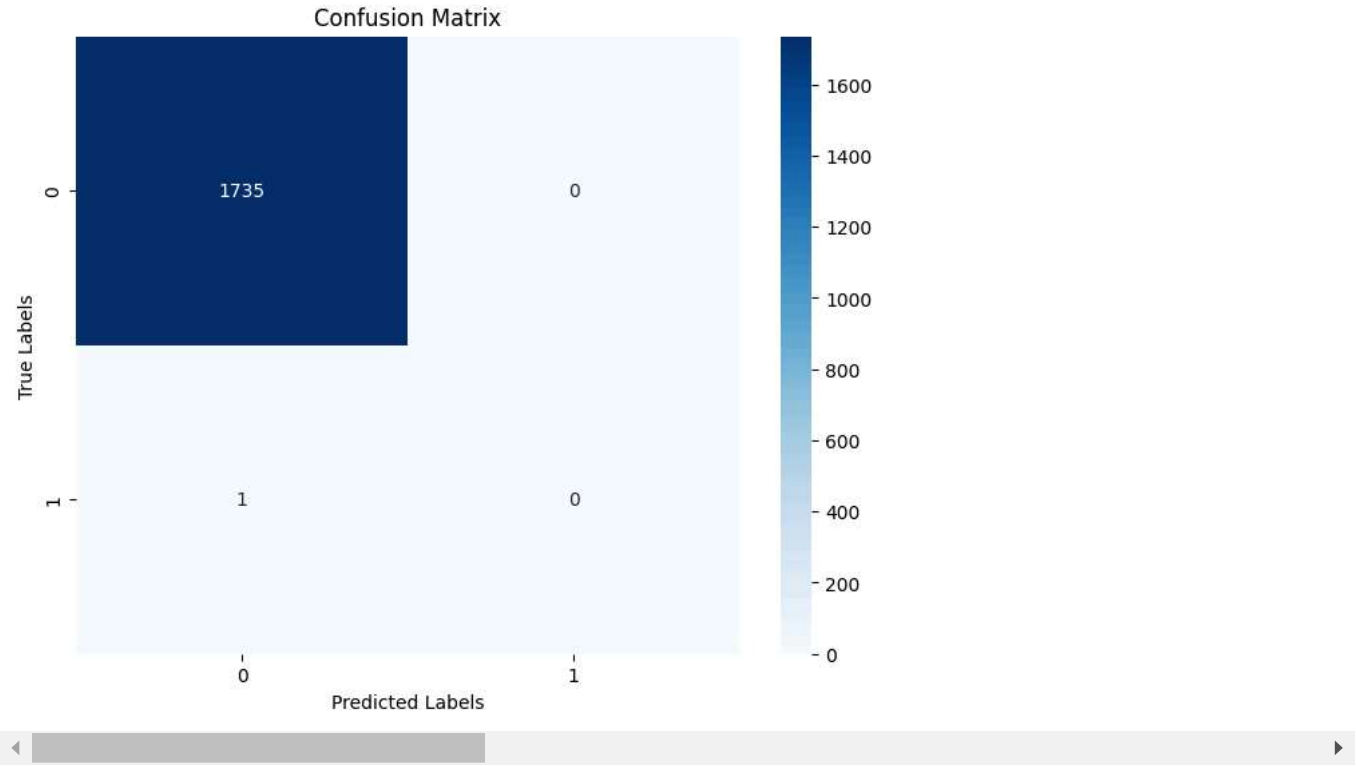
```
xgb_model.fit(X_train, y_train)
```

```
XGBClassifier(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=None, max_bin=None,
              max_cat_threshold=None, max_cat_to_onehot=None,
              max_delta_step=None, max_depth=None, max_leaves=None,
              min_child_weight=None, missing=nan, monotone_constraints=None,
              multi_strategy=None, n_estimators=None, n_jobs=None,
              num_parallel_tree=None, random_state=None, ...)
```

Full Code for Evaluation in XGBoost

```
from xgboost import XGBClassifier
from sklearn.metrics import accuracy_score, f1_score, classification_report, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns

# Instantiate and train the model
xgb_model = XGBClassifier(use_label_encoder=False, eval_metric='mlogloss')
xgb_model.fit(X_train, y_train)

# Make predictions
y_pred = xgb_model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred, average='weighted')

print("Accuracy:", accuracy)
print("F1 Score:", f1)
print("\nClassification Report:\n", classification_report(y_test, y_pred))

# Plot the confusion matrix
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.title('Confusion Matrix')
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/xgboost/core.py:158: UserWarning: [09:36:08] WARNING: /workspace/src/learner.cc:740:
Parameters: { "use_label_encoder" } are not used.

  warnings.warn(smsg, UserWarning)
Accuracy: 0.9994239631336406
F1 Score: 0.9991360276789781

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      1735
           1       0.00      0.00      0.00         1

    accuracy                           1.00      1736
   macro avg       0.50      0.50      0.50      1736
weighted avg       1.00      1.00      1.00      1736

/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1531: UndefinedMetricWarning: Precision is ill-defined an
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1531: UndefinedMetricWarning: Precision is ill-defined an
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1531: UndefinedMetricWarning: Precision is ill-defi
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

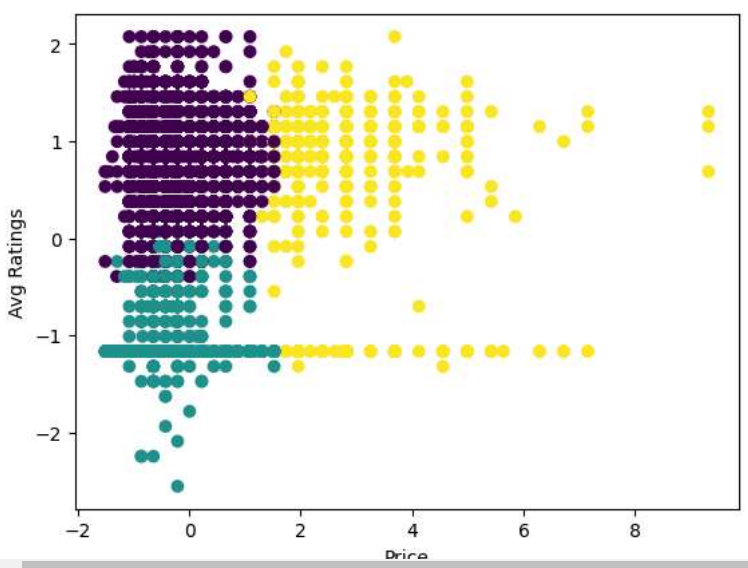


Step 6: Clustering - Restaurant Segmentation

```
X = df[['Price', 'Avg ratings', 'Delivery time']]
X_scaled = scaler.fit_transform(X)
```

```
#Fit a K-means model to create clusters
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=3, random_state=42)
kmeans.fit(X_scaled)
df['Cluster'] = kmeans.labels_
```

```
# Use a scatter plot to visualize clusters
plt.scatter(X_scaled[:, 0], X_scaled[:, 1], c=df['Cluster'])
plt.xlabel('Price')
plt.ylabel('Avg Ratings')
plt.show()
```



Step 7: Model Evaluation and Hyperparameter Tuning

```
from sklearn.model_selection import RandomizedSearchCV, GridSearchCV
from xgboost import XGBClassifier


#We'll define a grid of hyperparameters that we want to search over
param_grid = {
    'n_estimators': [100, 200, 300, 500],           # Number of trees
    'max_depth': [3, 5, 7, 10],                     # Maximum depth of the trees
    'learning_rate': [0.01, 0.05, 0.1, 0.2],        # Step size shrinkage
    'subsample': [0.6, 0.8, 1.0],                   # Percentage of samples used per tree
    'colsample_bytree': [0.6, 0.8, 1.0],            # Percentage of features used per tree
    'gamma': [0, 0.1, 0.2, 0.3],                    # Minimum loss reduction
    'min_child_weight': [1, 3, 5]                   # Minimum sum of instance weight
}


#Set Up RandomizedSearchCV
xgb = XGBClassifier(use_label_encoder=False, eval_metric='mlogloss')

random_search = RandomizedSearchCV(
    estimator=xgb,
    param_distributions=param_grid,
    n_iter=50,                  # Number of different combinations to try
    scoring='f1_weighted',      # Evaluation metric (e.g., 'accuracy', 'f1_weighted')
    cv=3,                       # Cross-validation splits
    verbose=1,                  # Print progress
    random_state=42,
    n_jobs=-1                   # Use all available cores
)


#Fit the Randomized Search
random_search.fit(X_train, y_train)
```

```
Fitting 3 folds for each of 50 candidates, totalling 150 fits
/usr/local/lib/python3.10/dist-packages/sklearn/model_selection/_split.py:776: UserWarning: The least populated class in y has only
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/xgboost/core.py:158: UserWarning: [09:42:25] WARNING: /workspace/src/learner.cc:740:
Parameters: { "use_label_encoder" } are not used.

  warnings.warn(smsg, UserWarning)
▸ RandomizedSearchCV
▸ best_estimator_: XGBClassifier
    ▸ XGBClassifier
```

```
#Retrieve the Best Parameters
print("Best Hyperparameters:", random_search.best_params_)
best_model = random_search.best_estimator_
```

```
Best Hyperparameters: {'subsample': 1.0, 'n_estimators': 300, 'min_child_weight': 5, 'max_depth': 10, 'learning_rate': 0.05, 'gamma
```

```
#Evaluate the Best Model
y_pred = best_model.predict(X_test)
```

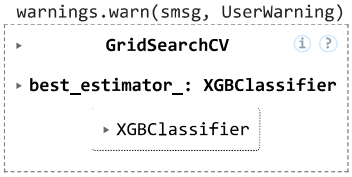
```
#Calculate Accuracy and F1 Score
from sklearn.metrics import accuracy_score, f1_score, classification_report

accuracy = accuracy_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred, average='weighted')

print("Tuned Model Accuracy:", accuracy)
print("Tuned Model F1 Score:", f1)
print("\nClassification Report:\n", classification_report(y_test, y_pred))
```

```
Tuned Model Accuracy: 0.9994239631336406
Tuned Model F1 Score: 0.9991360276789781

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      1735
           1       0.00      0.00      0.00         1

    accuracy                           1.00      1736
   macro avg       0.50      0.50      0.50      1736
weighted avg       1.00      1.00      1.00      1736

/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1531: UndefinedMetricWarning: Precision is ill-defined an
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1531: UndefinedMetricWarning: Precision is ill-defined an
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1531: UndefinedMetricWarning: Precision is ill-defined an
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

```
#Fine-Tuning with GridSearchCV
fine_tuned_param_grid = {
    'n_estimators': [200, 300],
    'max_depth': [6, 7, 8],
    'learning_rate': [0.05, 0.1],
    'subsample': [0.8, 1.0],
    'colsample_bytree': [0.8, 1.0],
    'gamma': [0, 0.1],
    'min_child_weight': [3, 4]
}
```

```
#Set Up and Fit GridSearchCV
grid_search = GridSearchCV(
    estimator=xgb,
    param_grid=fine_tuned_param_grid,
    scoring='f1_weighted',
    cv=3,
    verbose=1,
    n_jobs=-1
)

grid_search.fit(X_train, y_train)
```

```
Fitting 3 folds for each of 192 candidates, totalling 576 fits
/usr/local/lib/python3.10/dist-packages/sklearn/model_selection/_split.py:776: UserWarning: The least populated class in y has only
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/xgboost/core.py:158: UserWarning: [09:47:16] WARNING: /workspace/src/learner.cc:740:
Parameters: { "use_label_encoder" } are not used.

  warnings.warn(smsg, UserWarning)
```




```
#Retrieve the Best Parameters and Evaluate
print("Best Parameters after Grid Search:", grid_search.best_params_)
best_grid_model = grid_search.best_estimator_

# Evaluate the model
```