project2

July 27, 2024

1 Predictive Modeling and Analysis of Online Food Delivery Services

2 Author: Anuj Prabhu

3 Date: 27 April 2024

3.1 Background

- In the modern era of digital convenience, online food ordering platforms have become an integral part of many people's lives, offering a convenient way to access a wide range of culinary options. Understanding the dynamics of customer behavior and satisfaction within this domain is crucial for platform operators to enhance service quality and cater to evolving consumer preferences.
- The dataset under analysis contains comprehensive information collected from an online food ordering platform over a long period of time. It encompasses demographic attributes such as age, gender, marital status, occupation, and educational qualifications of customers, as well as location-specific details like their latitude, longitude, and pin code data. Additionally, it includes crucial feedback from customers regarding their satisfaction with the service, alongside the outcome of their orders.
- With the aim of delving into the intricate relationship between demographic/location factors and online food ordering behavior, the project embarks on an exploratory journey. Through rigorous analysis and modeling techniques, it seeks to uncover valuable insights that can guide decision-making processes and improve service quality within the online food-ordering landscape.

3.2 Problem Statement

- The objective of this project is to **analyze the impact of demographic attributes on customer feedback regarding their orders and the resultant order output**. By examining factors such as age, gender, location, and any other relevant demographic information, we aim to discern patterns in customer satisfaction and identify potential correlations between demographics and feedback sentiment. Thus, this analysis will provide **valuable** insights for **improving customer experience** and optimizing order fulfillment processes.

[82]:
```python
# library imports
import pandas as pd
import numpy as np
```

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score,␣
 ↪recall_score
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
#from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import roc_curve, roc_auc_score
```

3.3 Loading the Dataset

[84]:
```
# Loading the Dataset
online_foods = pd.read_csv("./onlinefoods.csv")
print(online_foods.shape)
```

```
(388, 13)
```

- We load the "onlinefoods" dataset from our local directory and specify the parameters for loading the data into our project. To facilitate data manipulation, we convert strings to factors, leveraging the ease of handling factors. The dataset is then stored in the variable named online_foods.

3.4 General Summary

[86]:
```
# General summary
print(online_foods.head())
print()
print("- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -␣
 ↪- - -")
print()
print(online_foods.info())
```

```
   Age  Gender Marital Status Occupation  Monthly Income  \
0   20  Female         Single    Student       No Income
1   24  Female         Single    Student  Below Rs.10000
2   22    Male         Single    Student  Below Rs.10000
3   22  Female         Single    Student       No Income
4   22    Male         Single    Student  Below Rs.10000

  Educational Qualifications  Family size  latitude  longitude  Pin code  \
0              Post Graduate            4   12.9766    77.5993    560001
1                   Graduate            3   12.9770    77.5773    560009
2              Post Graduate            3   12.9551    77.6593    560017
3                   Graduate            6   12.9473    77.5616    560019
4              Post Graduate            4   12.9850    77.5533    560010

  Output   Feedback Unnamed: 12
```

```
0    Yes   Positive         Yes
1    Yes   Positive         Yes
2    Yes  Negative          Yes
3    Yes   Positive         Yes
4    Yes   Positive         Yes

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 388 entries, 0 to 387
Data columns (total 13 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   Age                         388 non-null    int64
 1   Gender                      388 non-null    object
 2   Marital Status              388 non-null    object
 3   Occupation                  388 non-null    object
 4   Monthly Income              388 non-null    object
 5   Educational Qualifications  388 non-null    object
 6   Family size                 388 non-null    int64
 7   latitude                    388 non-null    float64
 8   longitude                   388 non-null    float64
 9   Pin code                    388 non-null    int64
 10  Output                      388 non-null    object
 11  Feedback                    388 non-null    object
 12  Unnamed: 12                 388 non-null    object
dtypes: float64(2), int64(3), object(8)
memory usage: 39.5+ KB
None
```

- The dataset comprises of 388 observations across 13 variables. Notably, variables such as Monthly.Income and Educational.Qualifications exhibit numerous factors that are very similar in nature. Additionally, the dataset contains a variable labeled "Unnamed: 12" for which the dataset owner has not provided clarification regarding its significance or definition. Therefore, further exploration of variable "Unnamed: 12" is warranted to ascertain its purpose and relevance within the dataset.

[88]:
```python
# Understand "X" variable better
print(online_foods["Unnamed: 12"].unique())
```

```
['Yes' 'No']
```

- Upon examination, the variable "Unnamed: 12" is observed to have values labeled as "Yes" and "No," which lack clarity and fail to provide explanatory context. In addition to an absence of information about this variable on the dataset's source website, its ambiguity renders it irrelevant for our analysis. Consequently, we proceed to the data cleaning phase of this project, where the variable "Unnamed: 12" will be removed from consideration.

3.5 Data Cleaning and Munging

[90]:
```
online_foods = online_foods.drop("Unnamed: 12", axis=1) # Removing "X" variable␣
↪from online_foods because no background has been provided about this␣
↪attribute and it is not self-explanatory.
```

- After dropping the variable "Unnamed: 12" from online_foods, we'll proceed to inspect the dataframe for any missing values.

[92]:
```
# Check for NA values in dataset
has_na = online_foods.isna().any().any()
print(has_na) # No NA values in dataset
```

```
False
```

- After confirming the absence of NA (missing) values in the dataset, our next step involves addressing the issue of numerous similar factors by amalgamating them where appropriate.

[94]:
```
# Excluding "Prefer not to say" Marital Status values from analysis (handling␣
↪outlying data points)
online_foods["Marital Status"] = online_foods["Marital Status"].astype("str")
online_foods = online_foods[online_foods["Marital Status"] != "Prefer not to␣
↪say"]
#online_foods["Marital Status"] = online_foods["Marital Status"].
↪astype("category")

# Combining like factors
online_foods["Occupation"] = online_foods["Occupation"].astype("str")
online_foods.loc[online_foods["Occupation"].isin(["Employee", "Self␣
↪Employeed"]), "Occupation"] = "Employed"
#online_foods["Occupation"] = online_foods["Occupation"].astype("category")

online_foods["Monthly Income"] = online_foods["Monthly Income"].astype("str")
online_foods.loc[online_foods["Monthly Income"].isin(["10001 to 25000", "25001␣
↪to 50000"]), "Monthly Income"] = "Average Monthly Income"
online_foods.loc[online_foods["Monthly Income"] == "More than 50000", "Monthly␣
↪Income"] = "High Monthly Income"
online_foods.loc[online_foods["Monthly Income"].isin(["No Income", "Below Rs.
↪10000"]), "Monthly Income"] = "Low/No Monthly Income"
#online_foods["Monthly Income"] = online_foods["Monthly Income"].
↪astype("category")

online_foods["Educational Qualifications"] = online_foods["Educational␣
↪Qualifications"].astype("str")
online_foods.loc[online_foods["Educational Qualifications"].isin(["Post␣
↪Graduate", "Ph.D"]), "Educational Qualifications"] = "Higher Education"
online_foods.loc[online_foods["Educational Qualifications"].isin(["School",␣
↪"Uneducated"]), "Educational Qualifications"] = "Lower/No Education"
```

```
#online_foods["Educational Qualifications"] = online_foods["Educational␣
↪Qualifications"].astype("category")

online_foods["Output"] = online_foods["Output"].astype("str")
online_foods.loc[online_foods["Output"] == "No", "Output"] = "Unsuccessful"
online_foods.loc[online_foods["Output"] == "Yes", "Output"] = "Successful"
#online_foods["Output"] = online_foods["Output"].astype("category")

#online_foods["Gender"] = online_foods["Gender"].astype("category")
#online_foods["Feedback"] = online_foods["Feedback"].astype("category")
```

- After consolidating similar factors across multiple variables to alleviate ambiguity, we have enhanced the clarity of our data for exploratory analysis. For instance, the "Monthly.Income" variable originally encompassed values such as "Below Rs.10000", "More than 50000", and "No Income". As it can be observed, one label had a rupee symbol while others did not. These labels presented an inconsistency in the representation of income levels. To rectify this, we have transformed these labels into **three simplified categories**: **"Low/No Monthly Income", "Average Monthly Income", and "High Monthly Income"**. This streamlined data enables a clearer understanding of the relationship between a customer's monthly income and their online food ordering behavior. Similar refinement procedures have been applied to other variables, enhancing their interpretability and facilitating further analysis.

[96]:
```
# Reordering factors according to customs levels
#online_foods["Marital Status"] = pd.Categorical(online_foods["Marital␣
↪Status"], categories=["Single", "Married"], ordered=True)
#online_foods["Occupation"] = pd.Categorical(online_foods["Occupation"],␣
↪categories=["Student", "House wife", "Employed"], ordered=True)
#online_foods["Monthly Income"] = pd.Categorical(online_foods["Monthly␣
↪Income"], categories=["Low/No Monthly Income", "Average Monthly Income",␣
↪"High Monthly Income"], ordered=True)
#online_foods["Educational Qualifications"] = pd.
↪Categorical(online_foods["Educational Qualifications"], categories=["Lower/
↪No Education", "Graduate", "Higher Education"], ordered=True)
#online_foods["Output"] = pd.Categorical(online_foods["Output"],␣
↪categories=["Unsuccessful", "Successful"], ordered=True)

online_foods_reg = online_foods.copy()

online_foods_reg['Feedback'] = online_foods_reg['Feedback'].replace({'Negative␣
↪': 0, 'Positive': 1})
online_foods_reg['Output'] = online_foods_reg['Output'].replace({'Unsuccessful':
↪ 0, 'Successful': 1})
online_foods_reg['Gender'] = online_foods_reg['Gender'].replace({'Female': 0,␣
↪'Male': 1})
online_foods_reg['Marital Status'] = online_foods_reg['Marital Status'].
↪replace({'Single': 0, 'Married': 1})
```

```
online_foods_reg['Occupation'] = online_foods_reg['Occupation'].
 ↪replace({'Student': 0, 'House wife': 1, 'Employed': 2})
online_foods_reg['Monthly Income'] = online_foods_reg['Monthly Income'].
 ↪replace({'Low/No Monthly Income': 0, 'Average Monthly Income': 1, 'High␣
 ↪Monthly Income': 2})
online_foods_reg['Educational Qualifications'] = online_foods_reg['Educational␣
 ↪Qualifications'].replace({'Lower/No Education': 0, 'Graduate': 1, 'Higher␣
 ↪Education': 2})
```

- Following the consolidation of similar factors and the standardization of variable representations, we further preprocess the data by assigning different integers to each unique category in each variable. This is conducted to aid in regression analysis that follows next. With our data now cleaned and standardized, we are well-equipped to conduct exploratory analysis and derive meaningful insights from the dataset.

3.6 Exploratory Data Analysis

- Firstly, we build Logistic Regression Models to predict the **Feedback** on orders and the **Output Status** of orders by consumers of the online food ordering app. We build Logistic Regression Models specifically because we are mainly dealing with categorical data.

3.6.1 Building Logistic Regression Models

Feedback Model

[98]:
```
# GLM for Feedback
X_feedback = online_foods_reg.drop(['Feedback'], axis=1)  # Predictors
y_feedback = online_foods_reg['Feedback']  # Dependent variable

# Add constant term to predictors
X_feedback = sm.add_constant(X_feedback)

# Fit GLM for Feedback
feedback_model = sm.GLM(y_feedback, X_feedback, family=sm.families.Binomial())
feedback_results = feedback_model.fit()

# Summary of Feedback model
print(feedback_results.summary())

# Predictions on the entire dataset for Feedback
feedback_predictions = feedback_results.predict(X_feedback)
feedback_predictions_class = np.where(feedback_predictions > 0.5, 1, 0)

# Calculate accuracy
accuracy = accuracy_score(y_feedback, feedback_predictions_class)
print("Accuracy:", accuracy)

# Calculate precision
precision = precision_score(y_feedback, feedback_predictions_class)
```

```python
print("Precision:", precision)

# Calculate recall (sensitivity)
recall = recall_score(y_feedback, feedback_predictions_class)
print("Recall (Sensitivity):", recall)
```

```
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:               Feedback   No. Observations:                  376
Model:                            GLM   Df Residuals:                      364
Model Family:                Binomial   Df Model:                           11
Link Function:                  Logit   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -112.61
Date:                Sat, 27 Jul 2024   Deviance:                       225.23
Time:                        22:11:01   Pearson chi2:                     379.
No. Iterations:                     6   Pseudo R-squ. (CS):             0.2811
Covariance Type:            nonrobust
================================================================================
==============
                                 coef    std err          z      P>|z|
[0.025      0.975]
--------------------------------------------------------------------------------
--------------
const                       -909.1655   3307.218     -0.275      0.783
-7391.193    5572.862
Age                            0.0164      0.084      0.195      0.845
-0.148       0.181
Gender                        -0.7210      0.381     -1.890      0.059
-1.469       0.027
Marital Status                -0.0604      0.544     -0.111      0.912
-1.127       1.007
Occupation                    -0.7857      0.343     -2.292      0.022
-1.458      -0.114
Monthly Income                 0.7118      0.412      1.726      0.084
-0.096       1.520
Educational Qualifications     0.2350      0.331      0.709      0.478
-0.414       0.884
Family size                    0.0303      0.132      0.229      0.819
-0.229       0.289
latitude                      -4.1684      4.093     -1.019      0.308
-12.190       3.853
longitude                      4.7574      3.589      1.325      0.185
-2.277      11.792
Pin code                       0.0011      0.006      0.177      0.859
-0.011       0.013
Output                         3.0783      0.371      8.304      0.000
2.352       3.805
================================================================================
```

```
==============
Accuracy: 0.8882978723404256
Precision: 0.932258064516129
Recall (Sensitivity): 0.932258064516129
```

- we model the dependent variable **Feedback** based on every other predictor present in the dataset. To model the Feedback variable, we build a logistic regression model which predicts binary outcomes. The binary outcomes are "Positive" feedback or "Negative" feedback in this case. This is also why the family parameter of the model is set to be "binomial".
- We observe that the model is **88.83%** accurate and **93.23%** precise. This is already a significant improvement from the model we trained in project 1, which was **84%** accurate and **88%** precise.
- We are going to try to improve this model. To do so, we check the p-values of each predictor and remove the predictor with the greatest p-value. As long as the p-value of a predictor is > 0.05, we keep that predictor in. We remove one bad predictor at a time / a predictor that is not contributing linearly to the model.

[100]:
```
X_feedback = online_foods_reg.drop(['Feedback', 'Marital Status'], axis=1)  #␣
↪Predictors
y_feedback = online_foods_reg['Feedback']  # Dependent variable

# Add constant term to predictors
X_feedback = sm.add_constant(X_feedback)

# Fit GLM for Feedback
feedback_model = sm.GLM(y_feedback, X_feedback, family=sm.families.Binomial())
feedback_results = feedback_model.fit()

# Summary of Feedback model
print(feedback_results.summary())

# Predictions on the entire dataset for Feedback
feedback_predictions = feedback_results.predict(X_feedback)
feedback_predictions_class = np.where(feedback_predictions > 0.5, 1, 0)

# Calculate accuracy
accuracy = accuracy_score(y_feedback, feedback_predictions_class)
print("Accuracy:", accuracy)

# Calculate precision
precision = precision_score(y_feedback, feedback_predictions_class)
print("Precision:", precision)

# Calculate recall (sensitivity)
recall = recall_score(y_feedback, feedback_predictions_class)
print("Recall (Sensitivity):", recall)
```

```
                 Generalized Linear Model Regression Results
```

```
==============================================================================
Dep. Variable:               Feedback   No. Observations:                  376
Model:                            GLM   Df Residuals:                      365
Model Family:                Binomial   Df Model:                           10
Link Function:                  Logit   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -112.62
Date:                Sat, 27 Jul 2024   Deviance:                       225.24
Time:                        22:11:01   Pearson chi2:                     379.
No. Iterations:                     6   Pseudo R-squ. (CS):             0.2811
Covariance Type:            nonrobust
================================================================================
==============
                                coef    std err          z      P>|z|
[0.025      0.975]
--------------------------------------------------------------------------------
--------------
const                      -929.3680   3303.803     -0.281      0.778
-7404.703    5545.967
Age                           0.0127      0.077      0.165      0.869
-0.139       0.164
Gender                       -0.7125      0.374     -1.908      0.056
-1.445       0.020
Occupation                   -0.7907      0.339     -2.330      0.020
-1.456      -0.126
Monthly Income                0.7056      0.408      1.729      0.084
-0.094       1.505
Educational Qualifications    0.2454      0.318      0.772      0.440
-0.378       0.869
Family size                   0.0273      0.129      0.211      0.833
-0.226       0.281
latitude                     -4.1112      4.062     -1.012      0.311
-12.072       3.850
longitude                     4.7749      3.586      1.331      0.183
-2.254      11.804
Pin code                      0.0011      0.006      0.183      0.855
-0.011       0.013
Output                        3.0828      0.369      8.363      0.000
2.360       3.805
================================================================================
==============
Accuracy: 0.8882978723404256
Precision: 0.932258064516129
Recall (Sensitivity): 0.932258064516129
```

[101]:
```python
X_feedback = online_foods_reg.drop(['Feedback', 'Marital Status', 'Age'],␣
 ↪axis=1)  # Predictors
y_feedback = online_foods_reg['Feedback']  # Dependent variable
```

```python
# Add constant term to predictors
X_feedback = sm.add_constant(X_feedback)

# Fit GLM for Feedback
feedback_model = sm.GLM(y_feedback, X_feedback, family=sm.families.Binomial())
feedback_results = feedback_model.fit()

# Summary of Feedback model
print(feedback_results.summary())

# Predictions on the entire dataset for Feedback
feedback_predictions = feedback_results.predict(X_feedback)
feedback_predictions_class = np.where(feedback_predictions > 0.5, 1, 0)

# Calculate accuracy
accuracy = accuracy_score(y_feedback, feedback_predictions_class)
print("Accuracy:", accuracy)

# Calculate precision
precision = precision_score(y_feedback, feedback_predictions_class)
print("Precision:", precision)

# Calculate recall (sensitivity)
recall = recall_score(y_feedback, feedback_predictions_class)
print("Recall (Sensitivity):", recall)
```

```
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:               Feedback   No. Observations:                  376
Model:                            GLM   Df Residuals:                      366
Model Family:                Binomial   Df Model:                            9
Link Function:                  Logit   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -112.63
Date:                Sat, 27 Jul 2024   Deviance:                       225.27
Time:                        22:11:01   Pearson chi2:                     378.
No. Iterations:                     6   Pseudo R-squ. (CS):             0.2811
Covariance Type:            nonrobust
================================================================================
==============
                                 coef    std err          z      P>|z|
[0.025      0.975]
--------------------------------------------------------------------------------
--------------
const                       -950.2607   3305.294     -0.287      0.774
-7428.517    5527.996
Gender                        -0.7134      0.374     -1.910      0.056
-1.446       0.019
```

```
Occupation                        -0.7741      0.324     -2.389      0.017
-1.409      -0.139
Monthly Income                     0.7153      0.403      1.773      0.076
-0.076       1.506
Educational Qualifications         0.2384      0.315      0.756      0.449
-0.379       0.856
Family size                        0.0318      0.126      0.252      0.801
-0.216       0.280
latitude                          -4.0644      4.055     -1.002      0.316
-12.012       3.883
longitude                          4.7604      3.588      1.327      0.185
-2.271      11.792
Pin code                           0.0011      0.006      0.189      0.850
-0.011       0.013
Output                             3.0777      0.367      8.386      0.000
2.358       3.797
================================================================================
==============
Accuracy: 0.8856382978723404
Precision: 0.9292604501607717
Recall (Sensitivity): 0.932258064516129
```

- The model improves a little more here as the accuracy goes from **88.2%** to **88.5%**.

[103]:
```
X_feedback = online_foods_reg.drop(['Feedback', 'Marital Status', 'Age', 'Pin␣
↪code'], axis=1)  # Predictors
y_feedback = online_foods_reg['Feedback']  # Dependent variable

# Add constant term to predictors
X_feedback = sm.add_constant(X_feedback)

# Fit GLM for Feedback
feedback_model = sm.GLM(y_feedback, X_feedback, family=sm.families.Binomial())
feedback_results = feedback_model.fit()

# Summary of Feedback model
print(feedback_results.summary())

# Predictions on the entire dataset for Feedback
feedback_predictions = feedback_results.predict(X_feedback)
feedback_predictions_class = np.where(feedback_predictions > 0.5, 1, 0)

# Calculate accuracy
accuracy = accuracy_score(y_feedback, feedback_predictions_class)
print("Accuracy:", accuracy)

# Calculate precision
precision = precision_score(y_feedback, feedback_predictions_class)
```

```python
print("Precision:", precision)

# Calculate recall (sensitivity)
recall = recall_score(y_feedback, feedback_predictions_class)
print("Recall (Sensitivity):", recall)
```

```
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:               Feedback   No. Observations:                  376
Model:                            GLM   Df Residuals:                      367
Model Family:                Binomial   Df Model:                            8
Link Function:                  Logit   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -112.65
Date:                Sat, 27 Jul 2024   Deviance:                       225.30
Time:                        22:11:01   Pearson chi2:                     380.
No. Iterations:                     6   Pseudo R-squ. (CS):             0.2810
Covariance Type:            nonrobust
================================================================================
==============
                                 coef    std err          z      P>|z|
[0.025      0.975]
--------------------------------------------------------------------------------
--------------
const                       -326.9017    283.451     -1.153      0.249
-882.455     228.651
Gender                        -0.7172      0.373     -1.922      0.055
-1.448       0.014
Occupation                    -0.7700      0.324     -2.379      0.017
-1.404      -0.136
Monthly Income                 0.7113      0.403      1.764      0.078
-0.079       1.502
Educational Qualifications     0.2371      0.315      0.752      0.452
-0.381       0.855
Family size                    0.0312      0.126      0.247      0.805
-0.217       0.279
latitude                      -4.1351      4.035     -1.025      0.305
-12.043       3.773
longitude                      4.9041      3.500      1.401      0.161
-1.955      11.763
Output                         3.0759      0.367      8.388      0.000
2.357       3.795
================================================================================
==============
Accuracy: 0.8856382978723404
Precision: 0.9292604501607717
Recall (Sensitivity): 0.932258064516129
```

[104]:
```python
X_feedback = online_foods_reg.drop(['Feedback', 'Marital Status', 'Pin code',␣
 ↪'Age', 'Family size'], axis=1)  # Predictors
y_feedback = online_foods_reg['Feedback']  # Dependent variable

# Add constant term to predictors
X_feedback = sm.add_constant(X_feedback)

# Fit GLM for Feedback
feedback_model = sm.GLM(y_feedback, X_feedback, family=sm.families.Binomial())
feedback_results = feedback_model.fit()

# Summary of Feedback model
print(feedback_results.summary())

# Predictions on the entire dataset for Feedback
feedback_predictions = feedback_results.predict(X_feedback)
feedback_predictions_class = np.where(feedback_predictions > 0.5, 1, 0)

# Calculate accuracy
accuracy = accuracy_score(y_feedback, feedback_predictions_class)
print("Accuracy:", accuracy)

# Calculate precision
precision = precision_score(y_feedback, feedback_predictions_class)
print("Precision:", precision)

# Calculate recall (sensitivity)
recall = recall_score(y_feedback, feedback_predictions_class)
print("Recall (Sensitivity):", recall)
```

```
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:               Feedback   No. Observations:                  376
Model:                            GLM   Df Residuals:                      368
Model Family:                Binomial   Df Model:                            7
Link Function:                  Logit   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -112.68
Date:                Sat, 27 Jul 2024   Deviance:                       225.36
Time:                        22:11:01   Pearson chi2:                     381.
No. Iterations:                     6   Pseudo R-squ. (CS):             0.2809
Covariance Type:            nonrobust
================================================================================
==============
                                coef    std err          z      P>|z|
[0.025      0.975]
--------------------------------------------------------------------------------
--------------
const                      -327.1433    283.563     -1.154      0.249
```

```
-882.917     228.631
Gender                          -0.7308      0.369     -1.978      0.048
-1.455      -0.007
Occupation                      -0.7691      0.324     -2.372      0.018
-1.405      -0.134
Monthly Income                   0.7163      0.403      1.777      0.076
-0.074       1.506
Educational Qualifications       0.2319      0.314      0.738      0.461
-0.384       0.848
latitude                        -4.1903      4.025     -1.041      0.298
-12.079       3.698
longitude                        4.9180      3.501      1.405      0.160
-1.943      11.779
Output                           3.0702      0.366      8.391      0.000
2.353       3.787
================================================================================
==============
Accuracy: 0.8856382978723404
Precision: 0.9292604501607717
Recall (Sensitivity): 0.932258064516129
```

[105]:

```
# FINAL MODEL: FEEDBACK
X_feedback = online_foods_reg.drop(['Feedback', 'Marital Status', 'Pin code',␣
↪'Age', 'Family size', 'Educational Qualifications'], axis=1)  # Predictors
y_feedback = online_foods_reg['Feedback']  # Dependent variable

# Add constant term to predictors
X_feedback = sm.add_constant(X_feedback)

# Fit GLM for Feedback
feedback_model = sm.GLM(y_feedback, X_feedback, family=sm.families.Binomial())
feedback_results = feedback_model.fit()

# Summary of Feedback model
print(feedback_results.summary())

# Predictions on the entire dataset for Feedback
feedback_predictions = feedback_results.predict(X_feedback)
feedback_predictions_class = np.where(feedback_predictions > 0.5, 1, 0)

# Calculate accuracy
accuracy = accuracy_score(y_feedback, feedback_predictions_class)
print("Accuracy:", accuracy)

# Calculate precision
precision = precision_score(y_feedback, feedback_predictions_class)
print("Precision:", precision)
```

```python
# Calculate recall (sensitivity)
recall = recall_score(y_feedback, feedback_predictions_class)
print("Recall (Sensitivity):", recall)
```

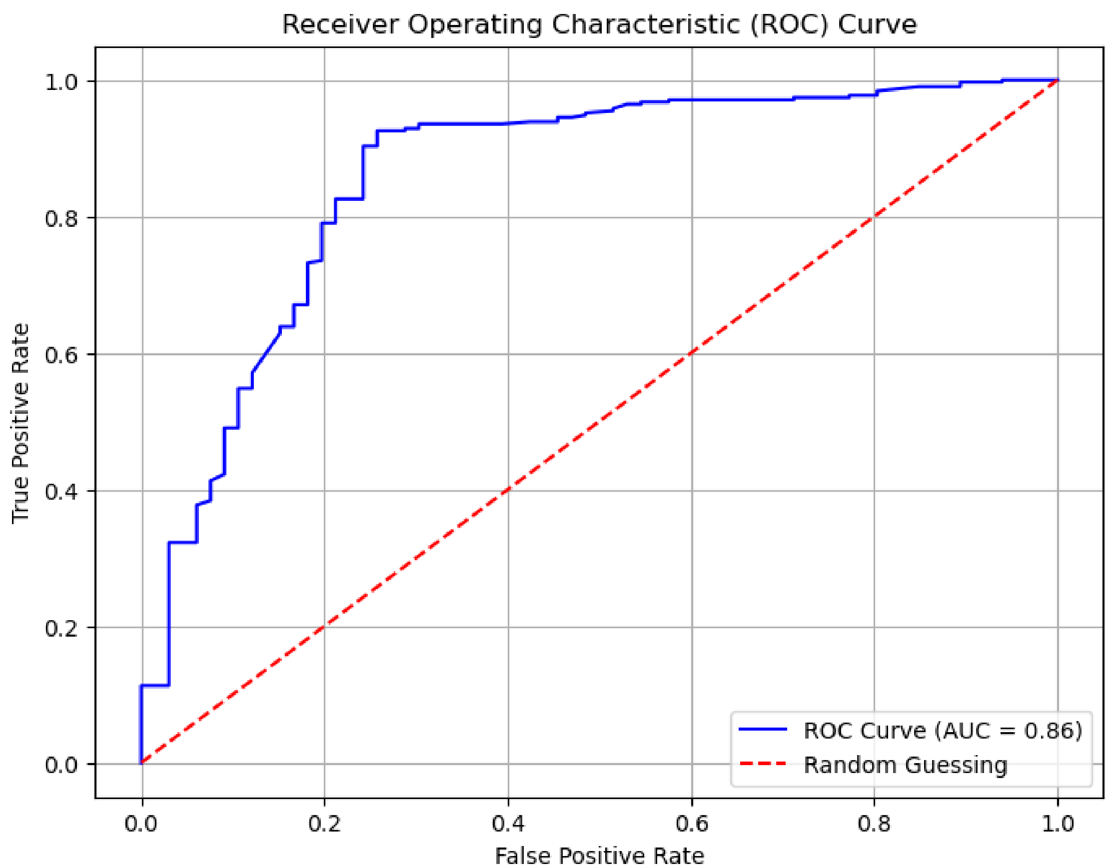
```
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:               Feedback   No. Observations:                  376
Model:                            GLM   Df Residuals:                      369
Model Family:                Binomial   Df Model:                            6
Link Function:                  Logit   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -112.95
Date:                Sat, 27 Jul 2024   Deviance:                       225.91
Time:                        22:11:01   Pearson chi2:                     393.
No. Iterations:                     6   Pseudo R-squ. (CS):             0.2798
Covariance Type:            nonrobust
================================================================================
==
                     coef    std err          z      P>|z|      [0.025
0.975]
--------------------------------------------------------------------------------
--
const           -334.2694    282.499     -1.183      0.237    -887.958
219.419
Gender            -0.7486      0.368     -2.037      0.042      -1.469
-0.028
Occupation        -0.8345      0.310     -2.690      0.007      -1.443
-0.226
Monthly Income     0.7759      0.396      1.961      0.050       0.000
1.551
latitude          -4.2221      4.021     -1.050      0.294     -12.103
3.659
longitude          5.0198      3.487      1.440      0.150      -1.815
11.854
Output             3.0755      0.366      8.414      0.000       2.359
3.792
================================================================================
==
Accuracy: 0.8909574468085106
Precision: 0.9324758842443 73
Recall (Sensitivity): 0.9354838709677419
```

- The GLM Logistic Regression Model reaches its peak with:
  - Accuracy: **89%**
  - Precision: **93.25%**
  - Recall: **93.55%**

[107]:
```
# Calculate ROC curve
fpr, tpr, thresholds = roc_curve(y_feedback, feedback_results.
 ↪predict(X_feedback))

# Calculate AUC score
auc = roc_auc_score(y_feedback, feedback_results.predict(X_feedback))

# Plot ROC curve
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, label='ROC Curve (AUC = {:.2f})'.format(auc), color='blue')
plt.plot([0, 1], [0, 1], color='red', linestyle='--', label='Random Guessing')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.legend(loc='lower right')
plt.grid(True)
plt.show()
```



- Finally, we plot the ROC curve of the model against the AUC curve.

Output Model

[110]:
```
# GLM for Output
X_feedback = online_foods_reg.drop(['Output'], axis=1)  # Predictors
y_feedback = online_foods_reg['Output']  # Dependent variable

# Add constant term to predictors
X_feedback = sm.add_constant(X_feedback)

# Fit GLM for Feedback
feedback_model = sm.GLM(y_feedback, X_feedback, family=sm.families.Binomial())
feedback_results = feedback_model.fit()

# Summary of Feedback model
print(feedback_results.summary())

# Predictions on the entire dataset for Feedback
feedback_predictions = feedback_results.predict(X_feedback)
feedback_predictions_class = np.where(feedback_predictions > 0.5, 1, 0)

# Calculate accuracy
accuracy = accuracy_score(y_feedback, feedback_predictions_class)
print("Accuracy:", accuracy)

# Calculate precision
precision = precision_score(y_feedback, feedback_predictions_class)
print("Precision:", precision)

# Calculate recall (sensitivity)
recall = recall_score(y_feedback, feedback_predictions_class)
print("Recall (Sensitivity):", recall)
```

```
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:                 Output   No. Observations:                  376
Model:                            GLM   Df Residuals:                      364
Model Family:                Binomial   Df Model:                           11
Link Function:                  Logit   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -128.33
Date:                Sat, 27 Jul 2024   Deviance:                       256.65
Time:                        22:11:01   Pearson chi2:                     402.
No. Iterations:                     5   Pseudo R-squ. (CS):             0.3020
Covariance Type:            nonrobust
================================================================================
==============
                                coef    std err          z      P>|z|
[0.025      0.975]
--------------------------------------------------------------------------------
--------------
```

```
const                          1283.1483   3042.222      0.422      0.673
-4679.498    7245.794
Age                              -0.0360      0.075     -0.480      0.631
-0.183       0.111
Gender                            0.4470      0.350      1.278      0.201
-0.239       1.133
Marital Status                   -0.6243      0.491     -1.273      0.203
-1.586       0.337
Occupation                       -0.4527      0.327     -1.383      0.167
-1.095       0.189
Monthly Income                    0.0688      0.390      0.176      0.860
-0.696       0.834
Educational Qualifications       -0.0940      0.293     -0.321      0.748
-0.668       0.480
Family size                      -0.0211      0.130     -0.162      0.871
-0.275       0.233
latitude                         -8.1516      3.756     -2.171      0.030
-15.512      -0.791
longitude                         0.0793      3.301      0.024      0.981
-6.390       6.548
Pin code                         -0.0021      0.005     -0.387      0.699
-0.013       0.009
Feedback                          3.0876      0.375      8.225      0.000
2.352       3.823
================================================================================
==============
Accuracy: 0.875
Precision: 0.8924050632911392
Recall (Sensitivity): 0.9559322033898305
```

- Now, we model the dependent variable **Output**. We model it in the same way as we did for **Feedback**. We get an initial accuracy score of **87.5%** with a precision score of **89.2%**. This model is already significantly better than the model we built in project 1, which had an accuracy of **16%** and a precision of **40%**.
- We continue to improve this model.

[112]:
```
X_feedback = online_foods_reg.drop(['Output', 'longitude'], axis=1)  #␣
↪Predictors
y_feedback = online_foods_reg['Output']  # Dependent variable

# Add constant term to predictors
X_feedback = sm.add_constant(X_feedback)

# Fit GLM for Feedback
feedback_model = sm.GLM(y_feedback, X_feedback, family=sm.families.Binomial())
feedback_results = feedback_model.fit()

# Summary of Feedback model
```

```python
print(feedback_results.summary())

# Predictions on the entire dataset for Feedback
feedback_predictions = feedback_results.predict(X_feedback)
feedback_predictions_class = np.where(feedback_predictions > 0.5, 1, 0)

# Calculate accuracy
accuracy = accuracy_score(y_feedback, feedback_predictions_class)
print("Accuracy:", accuracy)

# Calculate precision
precision = precision_score(y_feedback, feedback_predictions_class)
print("Precision:", precision)

# Calculate recall (sensitivity)
recall = recall_score(y_feedback, feedback_predictions_class)
print("Recall (Sensitivity):", recall)
```

```
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:                 Output   No. Observations:                  376
Model:                            GLM   Df Residuals:                      365
Model Family:                Binomial   Df Model:                           10
Link Function:                  Logit   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -128.33
Date:                Sat, 27 Jul 2024   Deviance:                       256.65
Time:                        22:11:01   Pearson chi2:                     402.
No. Iterations:                     5   Pseudo R-squ. (CS):             0.3020
Covariance Type:            nonrobust
================================================================================
==============
                                 coef    std err          z      P>|z|
[0.025      0.975]
--------------------------------------------------------------------------------
--------------
const                       1278.5582   3036.920      0.421      0.674
-4673.696    7230.812
Age                           -0.0360      0.075     -0.480      0.631
-0.183       0.111
Gender                         0.4467      0.350      1.278      0.201
-0.239       1.132
Marital Status                -0.6257      0.487     -1.284      0.199
-1.581       0.330
Occupation                    -0.4513      0.322     -1.401      0.161
-1.083       0.180
Monthly Income                 0.0684      0.390      0.175      0.861
-0.696       0.833
Educational Qualifications    -0.0942      0.293     -0.321      0.748
```

```
-0.668       0.480
Family size                  -0.0209      0.130     -0.161      0.872
-0.275       0.233
latitude                     -8.1627      3.728     -2.190      0.029
-15.469      -0.856
Pin code                     -0.0021      0.005     -0.387      0.699
-0.013       0.009
Feedback                      3.0885      0.373      8.270      0.000
2.357       3.820
================================================================================
==============
Accuracy: 0.875
Precision: 0.8924050632911392
Recall (Sensitivity): 0.9559322033898305
```

[113]:

```python
X_feedback = online_foods_reg.drop(['Output', 'longitude', 'Family size'],␣
 ↪axis=1)  # Predictors
y_feedback = online_foods_reg['Output']  # Dependent variable

# Add constant term to predictors
X_feedback = sm.add_constant(X_feedback)

# Fit GLM for Feedback
feedback_model = sm.GLM(y_feedback, X_feedback, family=sm.families.Binomial())
feedback_results = feedback_model.fit()

# Summary of Feedback model
print(feedback_results.summary())

# Predictions on the entire dataset for Feedback
feedback_predictions = feedback_results.predict(X_feedback)
feedback_predictions_class = np.where(feedback_predictions > 0.5, 1, 0)

# Calculate accuracy
accuracy = accuracy_score(y_feedback, feedback_predictions_class)
print("Accuracy:", accuracy)

# Calculate precision
precision = precision_score(y_feedback, feedback_predictions_class)
print("Precision:", precision)

# Calculate recall (sensitivity)
recall = recall_score(y_feedback, feedback_predictions_class)
print("Recall (Sensitivity):", recall)
```

```
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:                 Output   No. Observations:                  376
```

```
Model:                            GLM   Df Residuals:                      366
Model Family:                Binomial   Df Model:                            9
Link Function:                  Logit   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -128.34
Date:                Sat, 27 Jul 2024   Deviance:                       256.68
Time:                        22:11:01   Pearson chi2:                     401.
No. Iterations:                     5   Pseudo R-squ. (CS):             0.3019
Covariance Type:            nonrobust
================================================================================
==============
                                coef    std err          z      P>|z|
[0.025      0.975]
--------------------------------------------------------------------------------
--------------
const                      1266.8478   3032.376      0.418      0.676
-4676.499    7210.195
Age                          -0.0380      0.074     -0.514      0.607
-0.183       0.107
Gender                        0.4510      0.348      1.294      0.196
-0.232       1.134
Marital Status               -0.6356      0.483     -1.316      0.188
-1.583       0.311
Occupation                   -0.4442      0.319     -1.393      0.164
-1.069       0.181
Monthly Income                0.0631      0.388      0.163      0.871
-0.697       0.823
Educational Qualifications   -0.0905      0.292     -0.310      0.757
-0.663       0.482
latitude                     -8.1510      3.731     -2.185      0.029
-15.463      -0.839
Pin code                     -0.0021      0.005     -0.384      0.701
-0.013       0.009
Feedback                      3.0869      0.373      8.279      0.000
2.356       3.818
================================================================================
==============
Accuracy: 0.875
Precision: 0.8924050632911392
Recall (Sensitivity): 0.9559322033898305
```

[114]:
```
X_feedback = online_foods_reg.drop(['Output', 'longitude', 'Family size',␣
↪'Monthly Income'], axis=1)  # Predictors
y_feedback = online_foods_reg['Output']  # Dependent variable

# Add constant term to predictors
X_feedback = sm.add_constant(X_feedback)
```

```python
# Fit GLM for Feedback
feedback_model = sm.GLM(y_feedback, X_feedback, family=sm.families.Binomial())
feedback_results = feedback_model.fit()

# Summary of Feedback model
print(feedback_results.summary())

# Predictions on the entire dataset for Feedback
feedback_predictions = feedback_results.predict(X_feedback)
feedback_predictions_class = np.where(feedback_predictions > 0.5, 1, 0)

# Calculate accuracy
accuracy = accuracy_score(y_feedback, feedback_predictions_class)
print("Accuracy:", accuracy)

# Calculate precision
precision = precision_score(y_feedback, feedback_predictions_class)
print("Precision:", precision)

# Calculate recall (sensitivity)
recall = recall_score(y_feedback, feedback_predictions_class)
print("Recall (Sensitivity):", recall)
```

```
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:                 Output   No. Observations:                  376
Model:                            GLM   Df Residuals:                      367
Model Family:                Binomial   Df Model:                            8
Link Function:                  Logit   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -128.35
Date:                Sat, 27 Jul 2024   Deviance:                       256.71
Time:                        22:11:01   Pearson chi2:                     401.
No. Iterations:                     5   Pseudo R-squ. (CS):             0.3019
Covariance Type:            nonrobust
================================================================================
==============
                                 coef    std err          z      P>|z|
[0.025      0.975]
--------------------------------------------------------------------------------
--------------
const                       1255.7882   3029.851      0.414      0.679
-4682.610    7194.187
Age                           -0.0380      0.074     -0.514      0.607
-0.183       0.107
Gender                         0.4637      0.340      1.365      0.172
-0.202       1.130
Marital Status                -0.6217      0.475     -1.309      0.191
-1.553       0.309
```

```
Occupation                        -0.4071      0.223     -1.829      0.067
-0.843       0.029
Educational Qualifications        -0.0791      0.284     -0.279      0.780
-0.635       0.477
latitude                          -8.1323      3.726     -2.183      0.029
-15.435      -0.829
Pin code                          -0.0021      0.005     -0.381      0.703
-0.013       0.009
Feedback                           3.0949      0.370      8.367      0.000
2.370       3.820
================================================================================
==============
Accuracy: 0.875
Precision: 0.8924050632911392
Recall (Sensitivity): 0.9559322033898305
```

[115]:
```
X_feedback = online_foods_reg.drop(['Output', 'longitude', 'Family size',␣
↪'Monthly Income', 'Educational Qualifications'], axis=1)  # Predictors
y_feedback = online_foods_reg['Output']  # Dependent variable

# Add constant term to predictors
X_feedback = sm.add_constant(X_feedback)

# Fit GLM for Feedback
feedback_model = sm.GLM(y_feedback, X_feedback, family=sm.families.Binomial())
feedback_results = feedback_model.fit()

# Summary of Feedback model
print(feedback_results.summary())

# Predictions on the entire dataset for Feedback
feedback_predictions = feedback_results.predict(X_feedback)
feedback_predictions_class = np.where(feedback_predictions > 0.5, 1, 0)

# Calculate accuracy
accuracy = accuracy_score(y_feedback, feedback_predictions_class)
print("Accuracy:", accuracy)

# Calculate precision
precision = precision_score(y_feedback, feedback_predictions_class)
print("Precision:", precision)

# Calculate recall (sensitivity)
recall = recall_score(y_feedback, feedback_predictions_class)
print("Recall (Sensitivity):", recall)
```

```
                 Generalized Linear Model Regression Results
==============================================================================
```

```
Dep. Variable:                 Output   No. Observations:                  376
Model:                            GLM   Df Residuals:                      368
Model Family:                Binomial   Df Model:                            7
Link Function:                  Logit   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -128.39
Date:                Sat, 27 Jul 2024   Deviance:                       256.78
Time:                        22:11:01   Pearson chi2:                     400.
No. Iterations:                     5   Pseudo R-squ. (CS):             0.3017
Covariance Type:            nonrobust
================================================================================
==
                     coef    std err          z      P>|z|      [0.025
0.975]
--------------------------------------------------------------------------------
--
const           1246.4997   3032.205      0.411      0.681   -4696.512
7189.512
Age               -0.0372      0.074     -0.504      0.614      -0.182
0.107
Gender             0.4692      0.339      1.384      0.166      -0.195
1.134
Marital Status    -0.6031      0.470     -1.282      0.200      -1.525
0.319
Occupation        -0.4041      0.222     -1.822      0.068      -0.839
0.031
latitude          -8.0697      3.720     -2.169      0.030     -15.361
-0.779
Pin code          -0.0020      0.005     -0.378      0.706      -0.013
0.009
Feedback           3.0854      0.368      8.392      0.000       2.365
3.806
================================================================================
==
Accuracy: 0.8803191489361702
Precision: 0.8930817610062893
Recall (Sensitivity): 0.9627118644067797
```

[116]:

```
# FINAL MODEL: OUTPUT
X_feedback = online_foods_reg.drop(['Output', 'longitude', 'Family size',␣
↪'Monthly Income', 'Educational Qualifications',
                                   'Pin code', 'Age', 'Gender'], axis=1)  #␣
↪Predictors
y_feedback = online_foods_reg['Output']  # Dependent variable

# Add constant term to predictors
X_feedback = sm.add_constant(X_feedback)
```

```python
# Fit GLM for Feedback
feedback_model = sm.GLM(y_feedback, X_feedback, family=sm.families.Binomial())
feedback_results = feedback_model.fit()

# Summary of Feedback model
print(feedback_results.summary())

# Predictions on the entire dataset for Feedback
feedback_predictions = feedback_results.predict(X_feedback)
feedback_predictions_class = np.where(feedback_predictions > 0.5, 1, 0)

# Calculate accuracy
accuracy = accuracy_score(y_feedback, feedback_predictions_class)
print("Accuracy:", accuracy)

# Calculate precision
precision = precision_score(y_feedback, feedback_predictions_class)
print("Precision:", precision)

# Calculate recall (sensitivity)
recall = recall_score(y_feedback, feedback_predictions_class)
print("Recall (Sensitivity):", recall)
```

```
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:                 Output   No. Observations:                  376
Model:                            GLM   Df Residuals:                      371
Model Family:                Binomial   Df Model:                            4
Link Function:                  Logit   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -129.52
Date:                Sat, 27 Jul 2024   Deviance:                       259.04
Time:                        22:11:01   Pearson chi2:                     416.
No. Iterations:                     5   Pseudo R-squ. (CS):             0.2976
Covariance Type:            nonrobust
================================================================================
==
                     coef    std err          z      P>|z|      [0.025
0.975]
--------------------------------------------------------------------------------
--
const            110.2904     46.997      2.347      0.019      18.178
202.403
Marital Status    -0.7795      0.403     -1.933      0.053      -1.570
0.011
Occupation        -0.4206      0.207     -2.032      0.042      -0.826
-0.015
latitude          -8.5128      3.618     -2.353      0.019     -15.603
-1.423
```

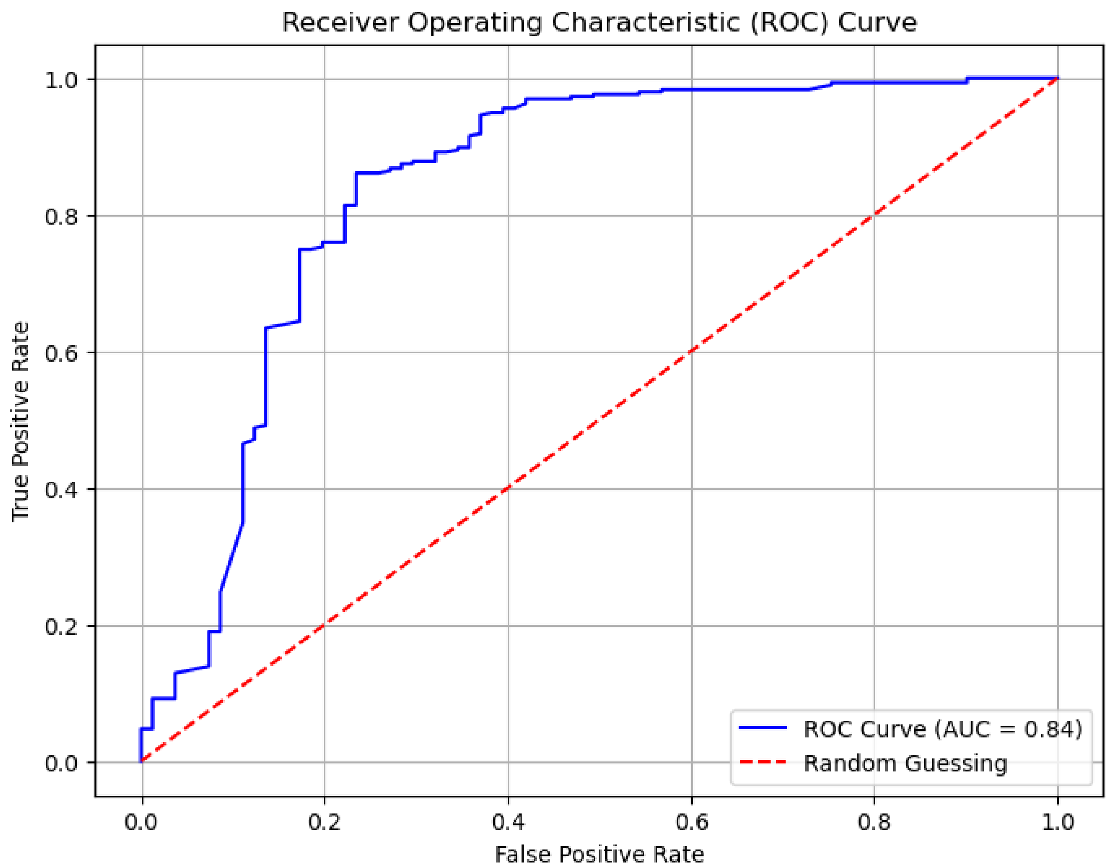
```
Feedback              3.0010      0.356      8.440      0.000       2.304
3.698
================================================================================
==
Accuracy: 0.8803191489361702
Precision: 0.8980891719745223
Recall (Sensitivity): 0.9559322033898305
```

- We achieve a peak model accuracy of **88%** with a precision of **90%** and a recall score of **95.6%**.
- Even though the model above this has a little higher accuracy and recall score, the model's y-intercept has a p-value of **0.681**. This is above the target p-value we set of **0.05**. Even though we cannot directly conduct a hypothesis test on the intercept and remove it, we can remove other predictors with high p-values.
- In doing so, we get a lower criitcal value for the intercept of the model, thus making the model more linear.

[118]:
```
# Calculate ROC curve
fpr, tpr, thresholds = roc_curve(y_feedback, feedback_results.
 ↪predict(X_feedback))

# Calculate AUC score
auc = roc_auc_score(y_feedback, feedback_results.predict(X_feedback))

# Plot ROC curve
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, label='ROC Curve (AUC = {:.2f})'.format(auc), color='blue')
plt.plot([0, 1], [0, 1], color='red', linestyle='--', label='Random Guessing')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.legend(loc='lower right')
plt.grid(True)
plt.show()
```



3.7 Data Visualization

3.7.1 Customer Base By Occupation

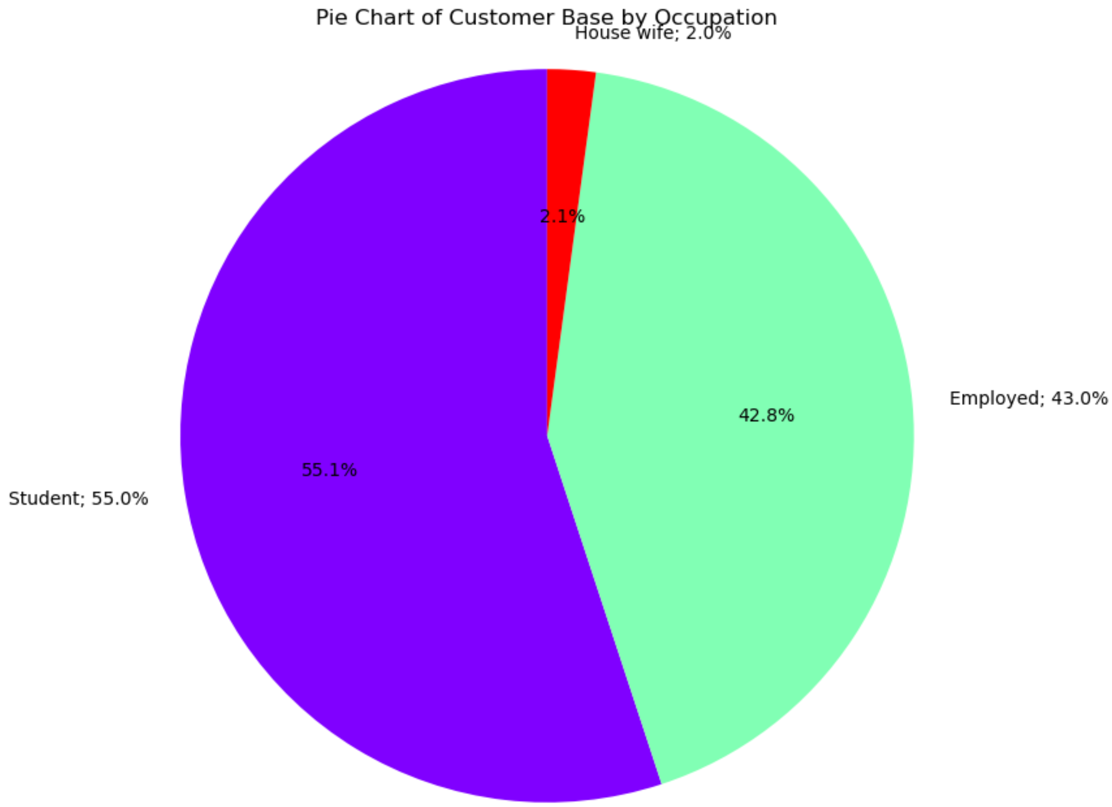
[120]:
```
# Calculate frequencies of each occupation
occupation_freq = online_foods['Occupation'].value_counts()

# Convert frequencies to a data frame
occupation_df = pd.DataFrame({'Occupation': occupation_freq.index, 'Frequency':␣
↪occupation_freq.values})

# Calculate percentages
pct = (occupation_df['Frequency'] / occupation_df['Frequency'].sum() * 100).
↪round()
lbls = [f"{occ}; {p}%" for occ, p in zip(occupation_df['Occupation'], pct)]

plt.figure(figsize=(8, 8))
plt.pie(occupation_df['Frequency'], labels=lbls, autopct='%1.1f%%',␣
↪startangle=90, colors=plt.cm.rainbow(np.linspace(0, 1, len(lbls))))
```

```
plt.title("Pie Chart of Customer Base by Occupation")
plt.axis('equal')
plt.show()
```



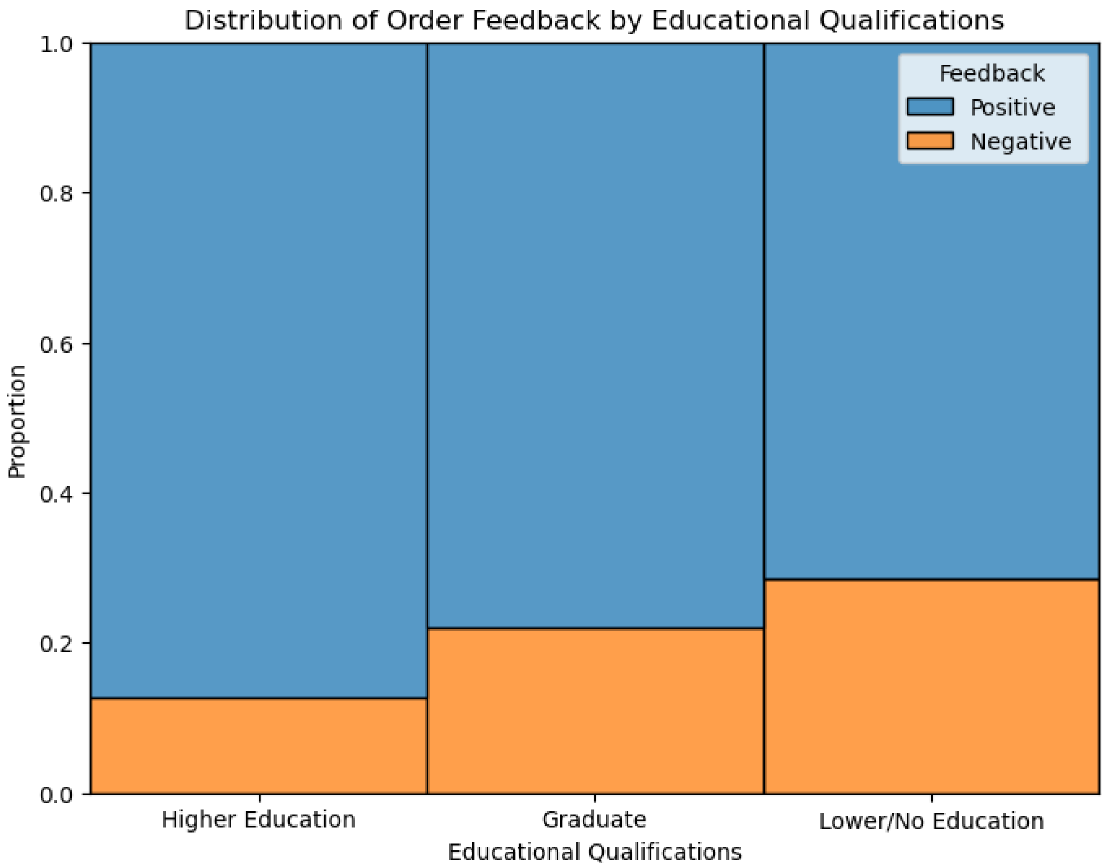
- In this analysis, we utilize a three-dimensional pie chart to visually represent the distribution of the sample according to their occupations.
- The findings reveal that a significant portion (**55%**) of the consumer base of the online food ordering app comprises students, while housewives constitute a much smaller proportion (**2%**).
- Notably, housewives are distinctly categorized as an occupation group within this dataset, possibly indicating regional biases specific to Bengaluru, India.

3.7.2 Distribution of Order Feedback Sentiment by Educational Qualification

[123]:
```
plt.figure(figsize=(8, 6))
sns.histplot(x="Educational Qualifications", hue="Feedback", data=online_foods,␣
↪stat="probability", multiple="fill")
plt.title("Distribution of Order Feedback by Educational Qualifications")
plt.xlabel("Educational Qualifications")
plt.ylabel("Proportion")
```

```
plt.show()
```

```
/opt/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

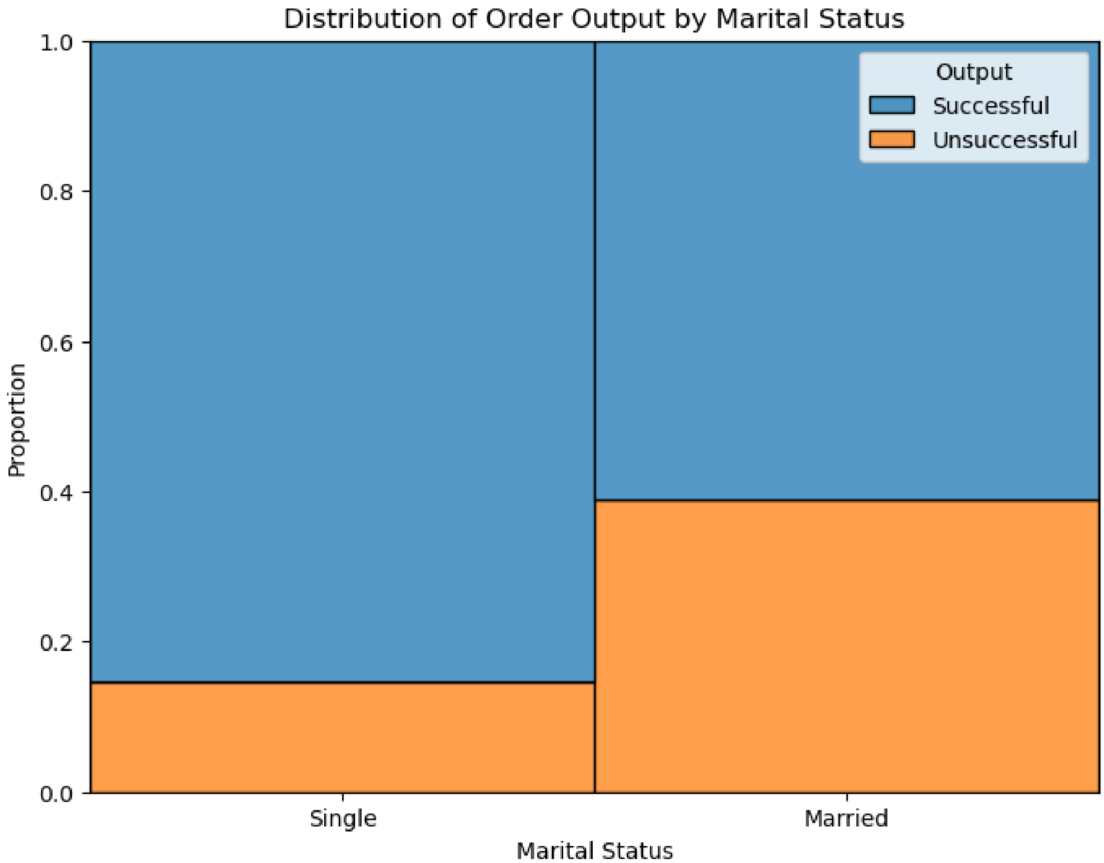


- Individuals with higher educational qualifications demonstrate a propensity to provide positive feedback on their orders significantly more often compared to those with lower levels of education.

3.7.3 Distribution of Order Output by Marital Status

[126]:
```
plt.figure(figsize=(8, 6))
sns.histplot(x="Marital Status", hue="Output", data=online_foods,␣
 ↪stat="probability", multiple="fill")
plt.title("Distribution of Order Output by Marital Status")
plt.xlabel("Marital Status")
plt.ylabel("Proportion")
plt.show()
```

```
/opt/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



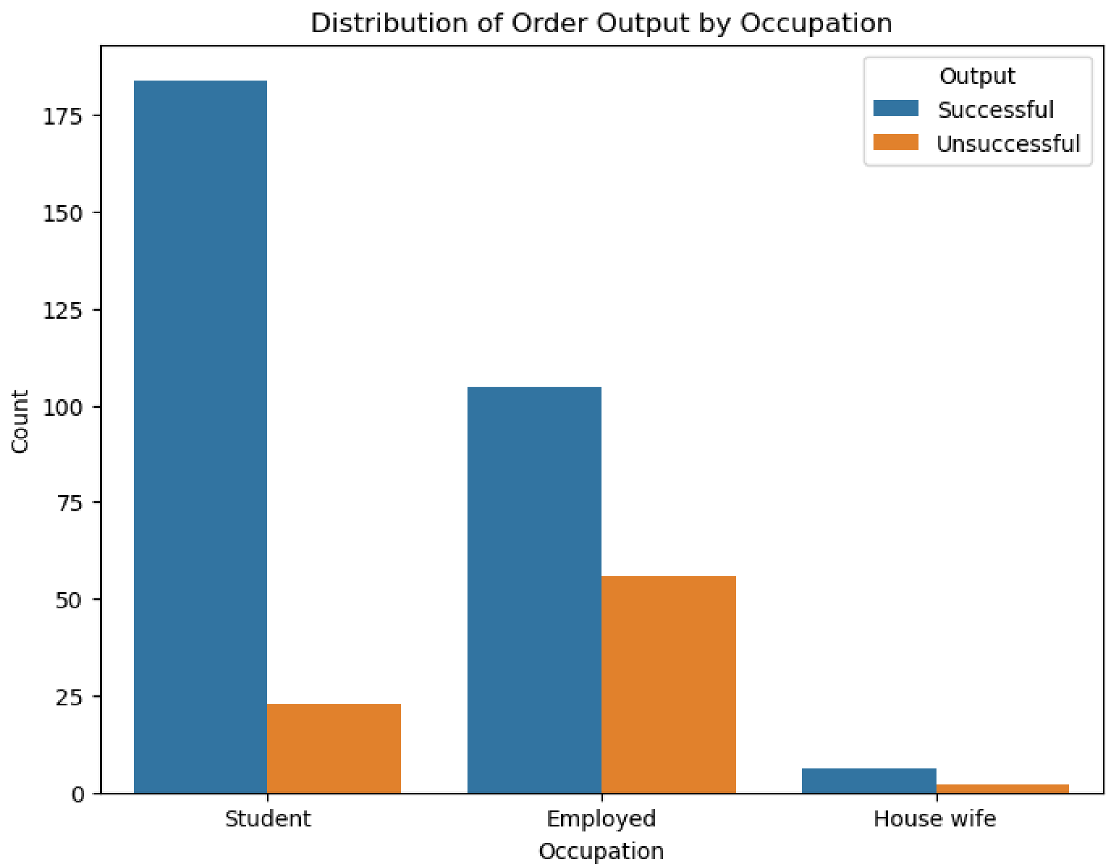
- This barplot suggests that married individuals tend to encounter a notably higher proportion of unsuccessful orders compared to single individuals. This trend may correlate with the observation that students in Bengaluru tend to leave more positive reviews on orders. This connection likely stems from the fact that students, who are typically single, experience a higher proportion of successful order transactions. (**based on regional biases of India, assuming students are generally single**)

3.7.4 Distribution of Order Output by Occupation

[129]:
```
occupation_order = ['Student', 'Employed', 'House wife']

plt.figure(figsize=(8, 6))
sns.countplot(x="Occupation", hue="Output", data=online_foods,␣
  ↪order=occupation_order)
plt.title("Distribution of Order Output by Occupation")
```

```
plt.xlabel("Occupation")
plt.ylabel("Count")
plt.legend(title="Output")
plt.show()
```



- The bar plot indicates a significant disparity in successful order experiences among different demographic groups. Specifically, students receive a substantially higher number of successful orders from the online food ordering platform compared to housewives and employed individuals. Conversely, employed individuals exhibit the lowest proportion of successful order experiences relative to other groups.

3.8 Conclusion

- Throughout this project, we've effectively performed data cleaning, analysis, and visualization to glean valuable insights into the ordering behaviors of Bengaluru, India residents on an online food ordering platform. By systematically refining the dataset, exploring its nuances through analytical techniques, and crafting insightful visualizations, we've uncovered significant patterns and trends shaping consumer preferences and habits in the local online food delivery landscape.

3.9 Dataset link

- Kaggle link to dataset source: dataset