ONLINE FOOD ORDER PREDICTION WITH MACHINE LEARNING

Online food order prediction leverages machine learning to anticipate customer ordering behavior based on historical data, user preferences, and browsing patterns. By applying classification or recommendation algorithms, platforms can predict whether a customer will place an order and recommend personalized options. Key techniques include data preprocessing, feature engineering, and the use of models such as decision trees, logistic regression, and collaborative filtering. This prediction system enhances user engagement, marketing efficiency, and overall operational effectiveness for food delivery services.

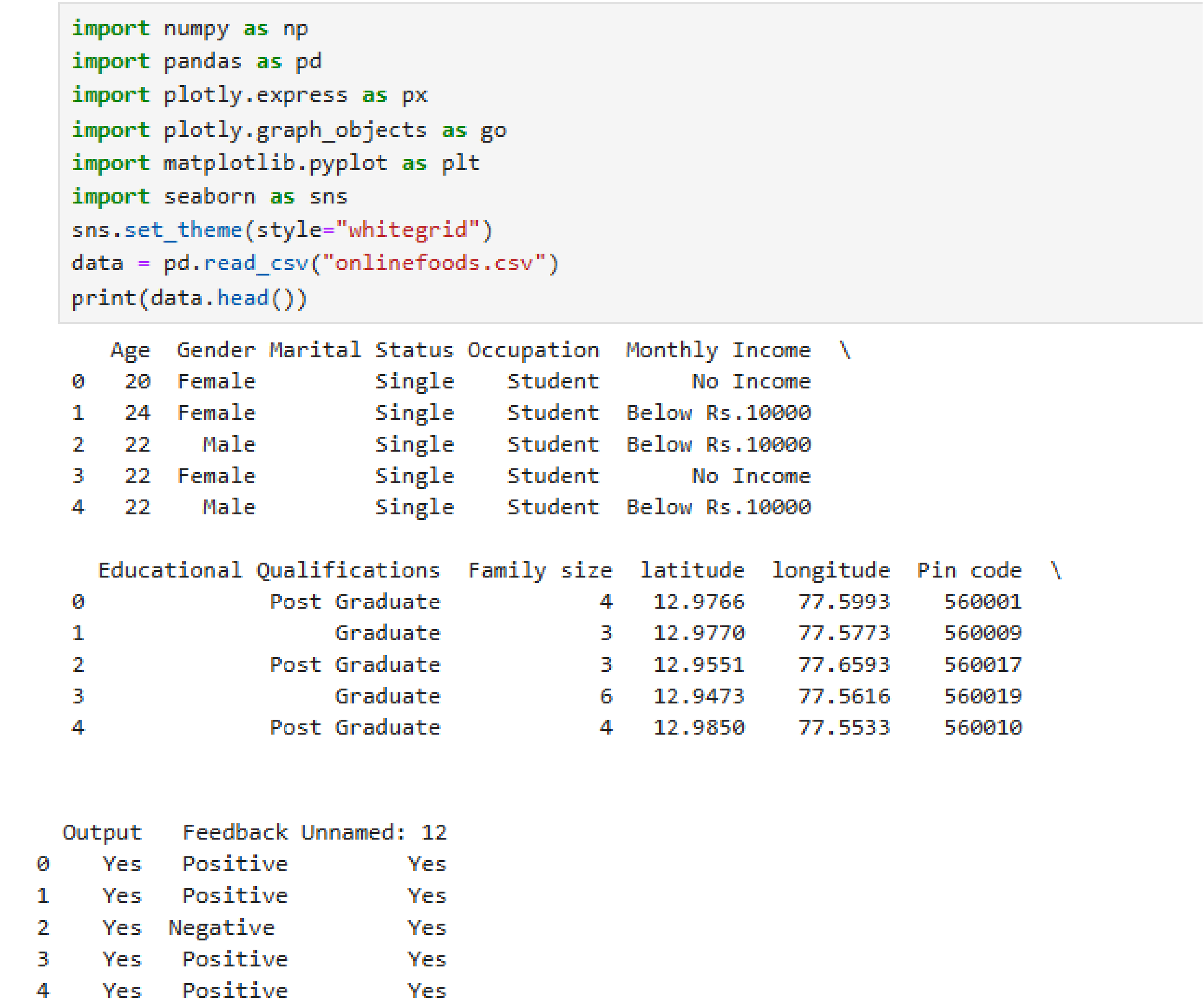
ONLINE FOOD ORDER PREDICTION :USE CASE

The surge in online food delivery services, driven by platforms like Swiggy and Zomato, has highlighted the need for efficient predictive models to streamline operations. This is a machine learning-based approach to predict online food orders, aiming to optimize delivery logistics and enhance customer satisfaction. The model leverages historical data on customer behavior to forecast demand in specific areas, enabling companies to allocate resources more effectively and reduce delivery times. Additionally, it predicts the likelihood of repeat orders, aiding in targeted marketing efforts.

The implementation involves several key steps:

1. Data Collection: Gather customer order history, restaurant details, user demographics, and external data (e.g., time of day, location).
2. Data Preprocessing: Clean data, handle missing values, encode categorical variables, and normalize data.
3. Feature Engineering: Extract relevant features such as order frequency, cuisine preferences, and delivery times.

4. Model Selection: Use algorithms like logistic regression, decision trees, or collaborative filtering for predictions.
5. Training and Evaluation: Split data into training/testing sets and evaluate the model.
6. Deployment: Implement the model in a real-time system.



```python
import numpy as np
import pandas as pd
import plotly.express as px
import plotly.graph_objects as go
import matplotlib.pyplot as plt
import seaborn as sns
sns.set_theme(style="whitegrid")
data = pd.read_csv("onlinefoods.csv")
print(data.head())
```

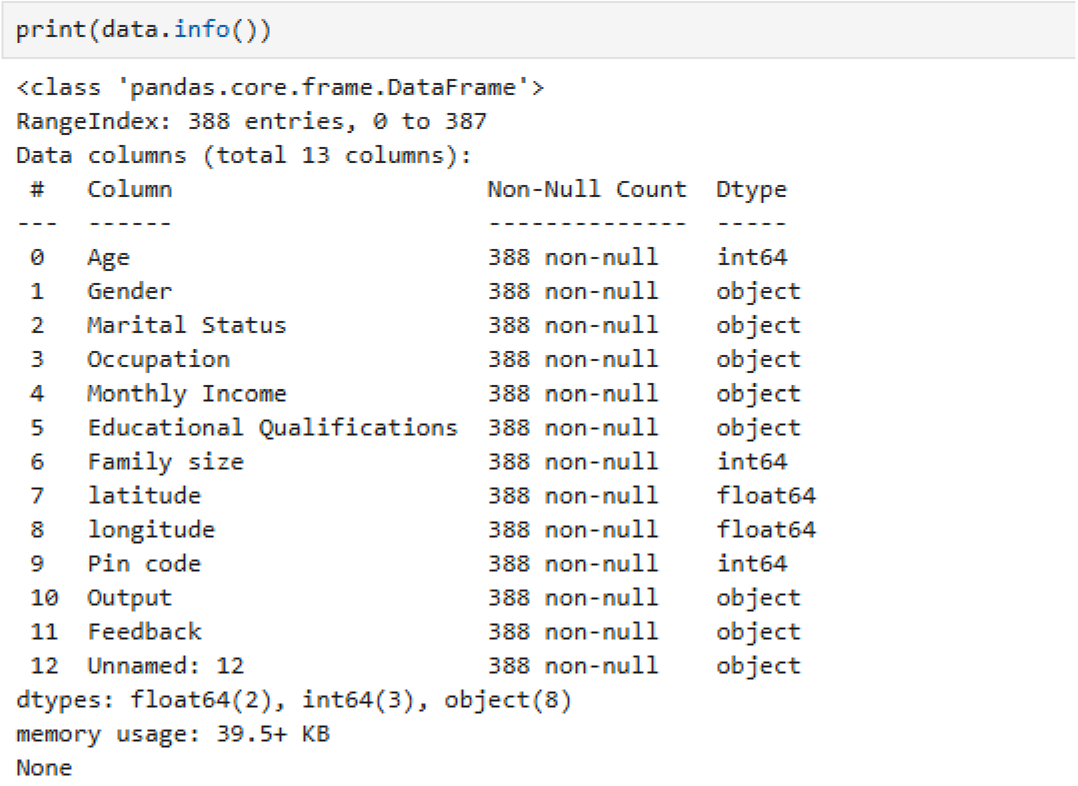
```
   Age  Gender Marital Status Occupation  Monthly Income  \
0   20  Female         Single    Student       No Income
1   24  Female         Single    Student  Below Rs.10000
2   22    Male         Single    Student  Below Rs.10000
3   22  Female         Single    Student       No Income
4   22  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

So the dataset contains information like:

1. the age of the customer
2. marital status of the customer
3. occupation of the customer
4. monthly income of the customer

5. educational qualification of the customer
6. family size of the customer
7. latitude and longitude of the location of the customer
8. pin code of the residence of the customer
9. did the customer order again (Output)
10. Feedback of the last order (Positive or Negative)

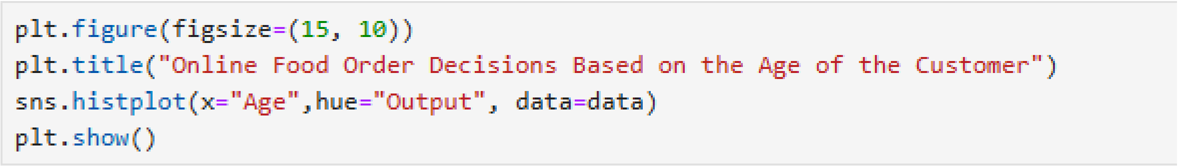
Let’s have a look at the information about all the columns in the dataset:



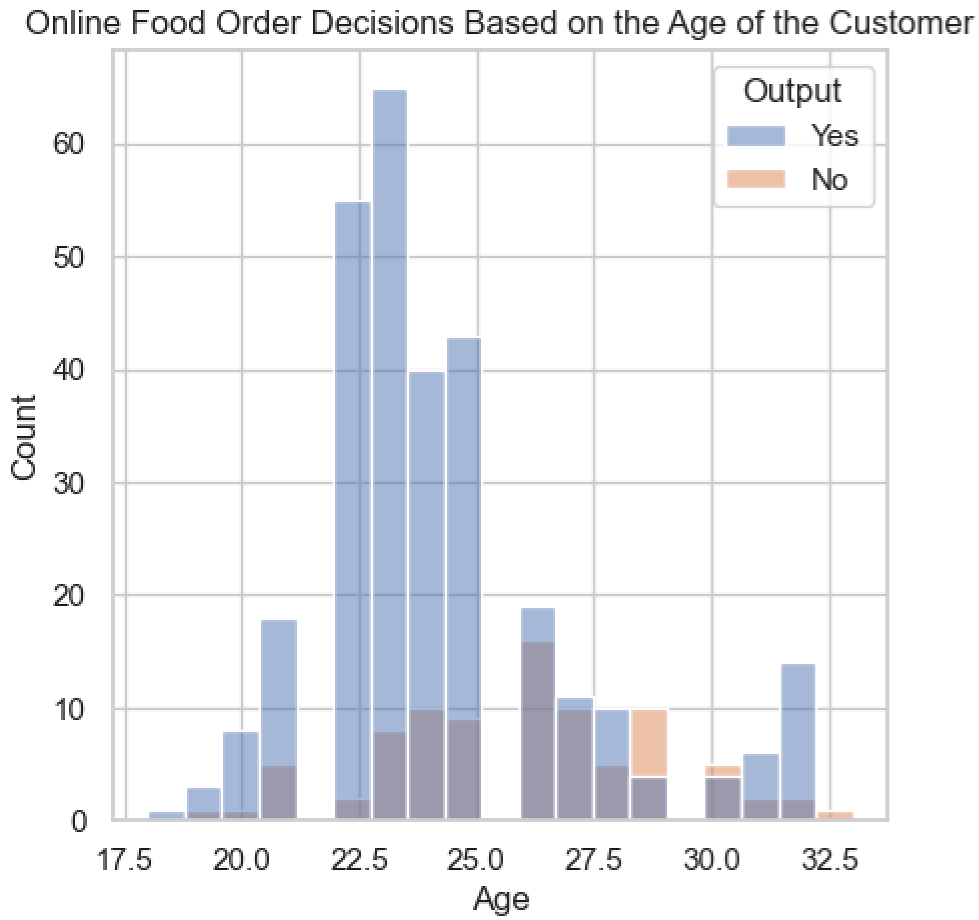
```
print(data.info())
```

```
<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
```

Now let’s move to the analysis of this data. Here I am starting by looking at the online food order decisions based on the age of the customer:

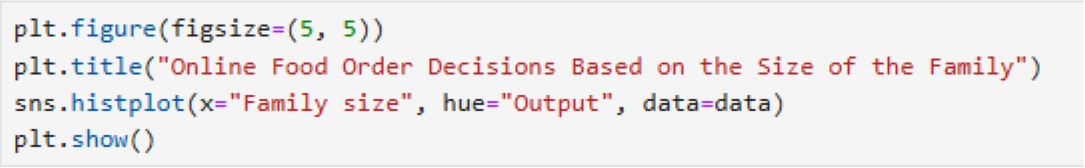


```
plt.figure(figsize=(15, 10))
plt.title("Online Food Order Decisions Based on the Age of the Customer")
sns.histplot(x="Age",hue="Output", data=data)
plt.show()
```

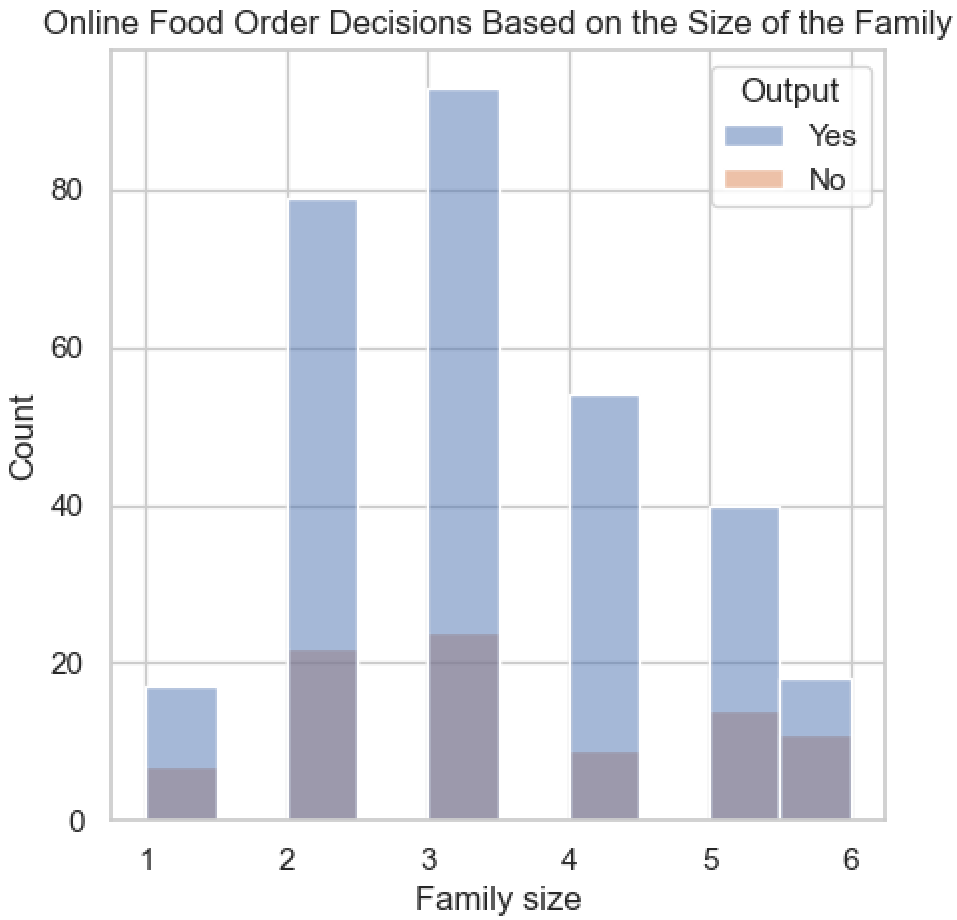


From the above plotting let’s determine that the age group of 22-25 ordered the food often again. It also means this age group is the target of online food delivery companies

Now let’s have a look at the online food order decisions based on the size of the family of the customer:

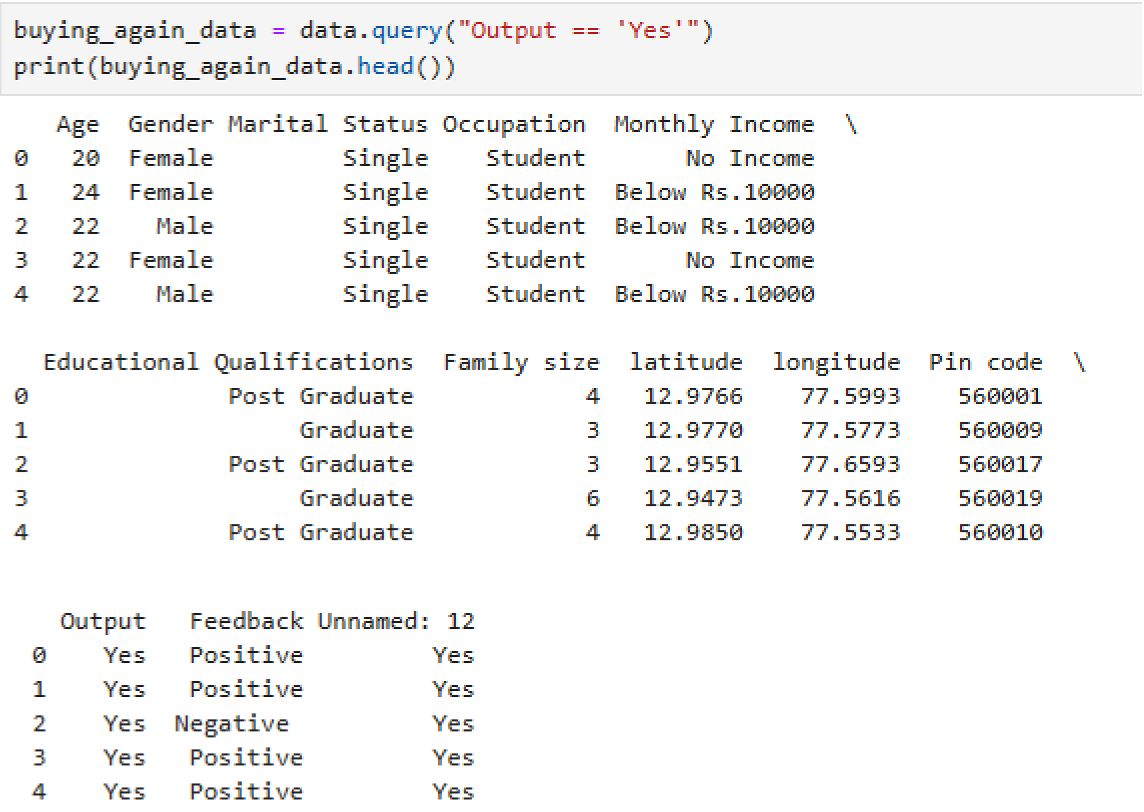


```
plt.figure(figsize=(5, 5))
plt.title("Online Food Order Decisions Based on the Size of the Family")
sns.histplot(x="Family size", hue="Output", data=data)
plt.show()
```



The above plot shows families with 2 and 3 members are ordering food often. These can be roommates, couples, or a family of three.

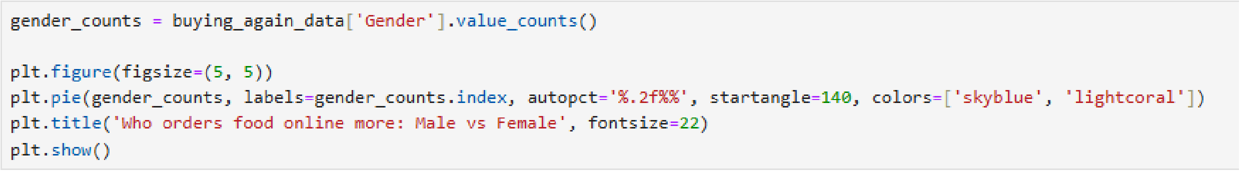
Let's create a dataset of all the customers who ordered the food again:



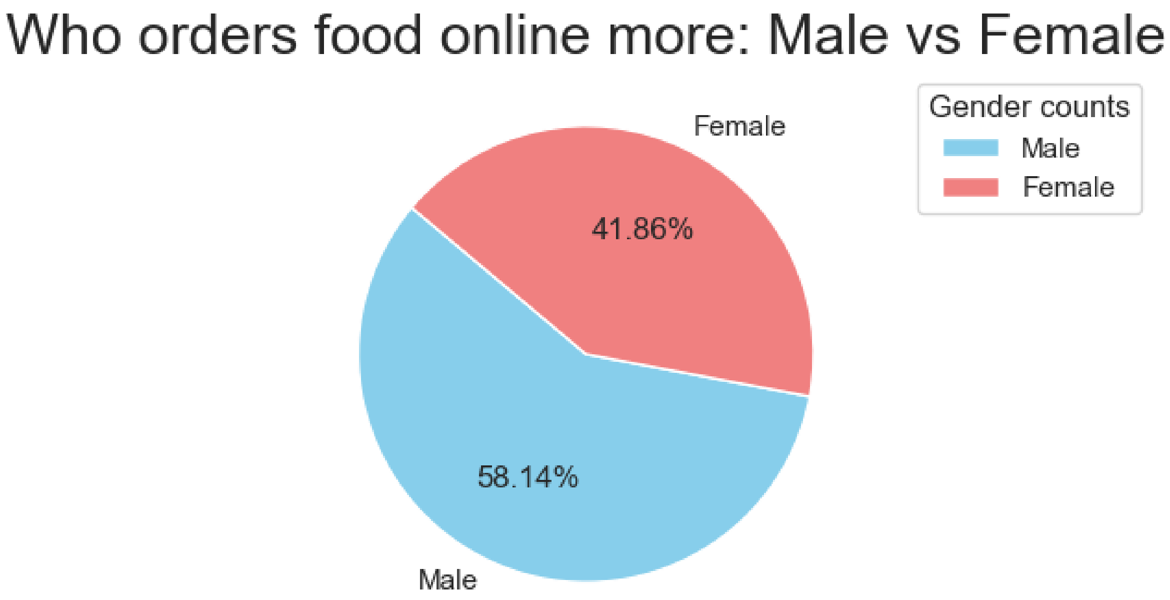
```
buying_again_data = data.query("Output == 'Yes'")
print(buying_again_data.head())
```

```
   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
```

Now let’s go through the gender column and let’s find who orders food more online:

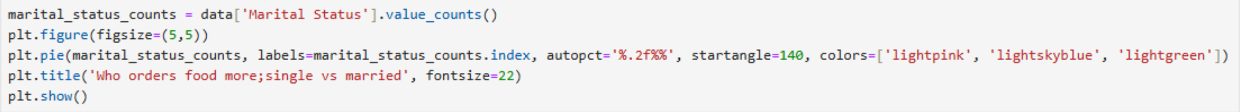


```
gender_counts = buying_again_data['Gender'].value_counts()

plt.figure(figsize=(5, 5))
plt.pie(gender_counts, labels=gender_counts.index, autopct='%.2f%%', startangle=140, colors=['skyblue', 'lightcoral'])
plt.title('Who orders food online more: Male vs Female', fontsize=22)
plt.show()
```

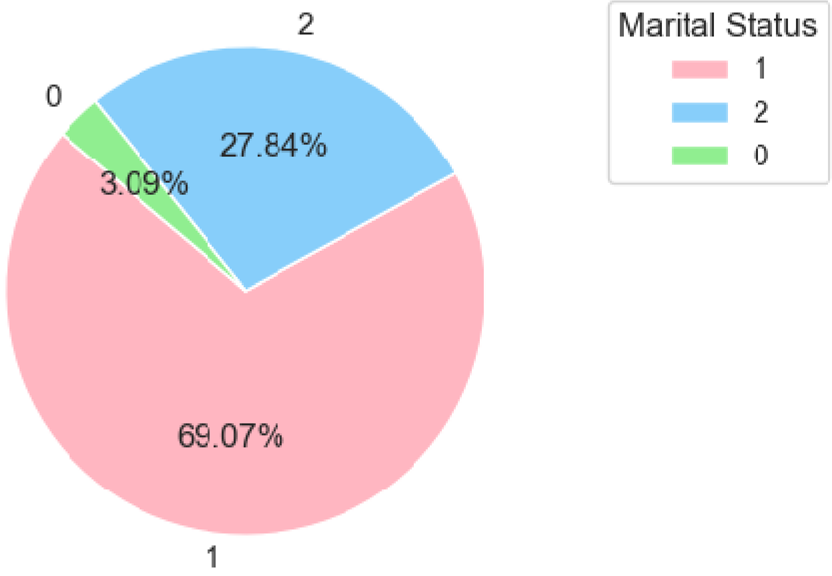


According to the above pie plot male customers are ordering more compared the females.

Now let’s go through the marital status of the customers who ordered again:

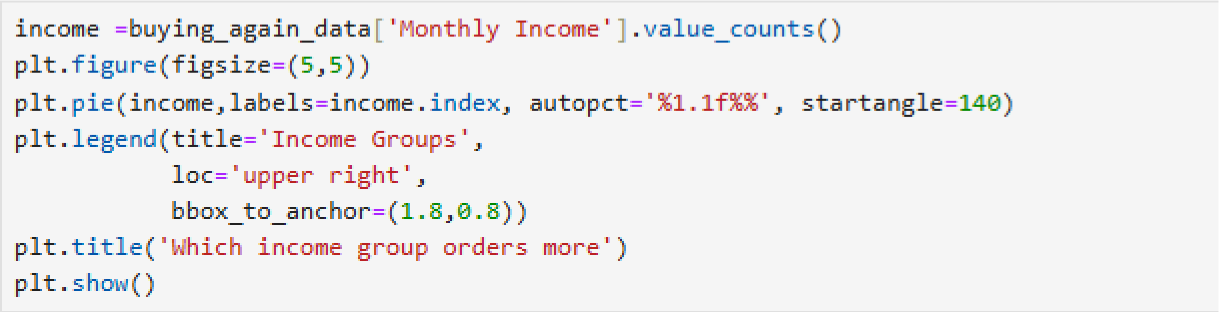


```
marital_status_counts = data['Marital Status'].value_counts()
plt.figure(figsize=(5,5))
plt.pie(marital_status_counts, labels=marital_status_counts.index, autopct='%.2f%%', startangle=140, colors=['lightpink', 'lightskyblue', 'lightgreen'])
plt.title('Who orders food more;single vs married', fontsize=22)
plt.show()
```

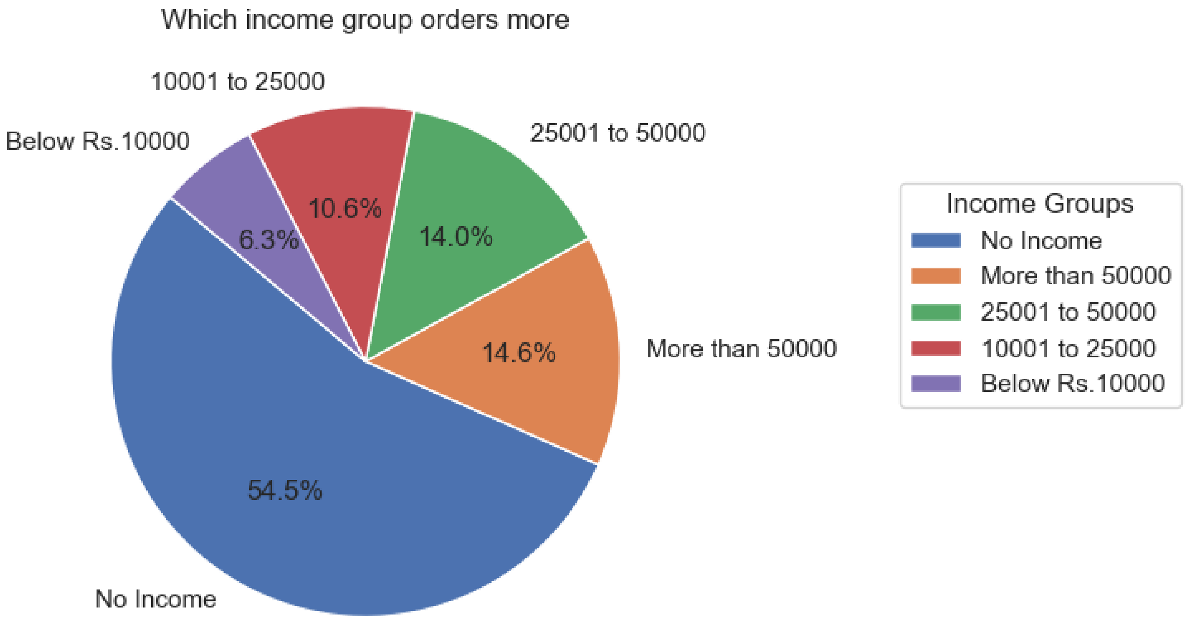


According to the above figure, most of the frequent customers are singles.

Now let’s have a look at what’s the income group of the customers who ordered the food again:

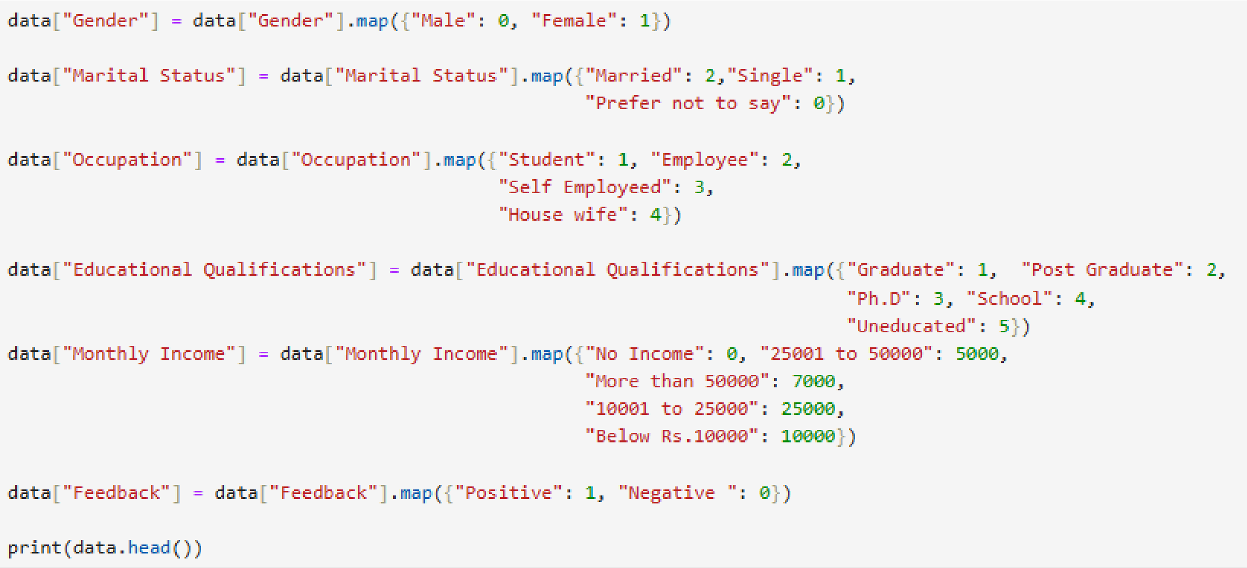


```
income =buying_again_data['Monthly Income'].value_counts()
plt.figure(figsize=(5,5))
plt.pie(income,labels=income.index, autopct='%1.1f%%', startangle=140)
plt.legend(title='Income Groups',
           loc='upper right',
           bbox_to_anchor=(1.8,0.8))
plt.title('Which income group orders more')
plt.show()
```

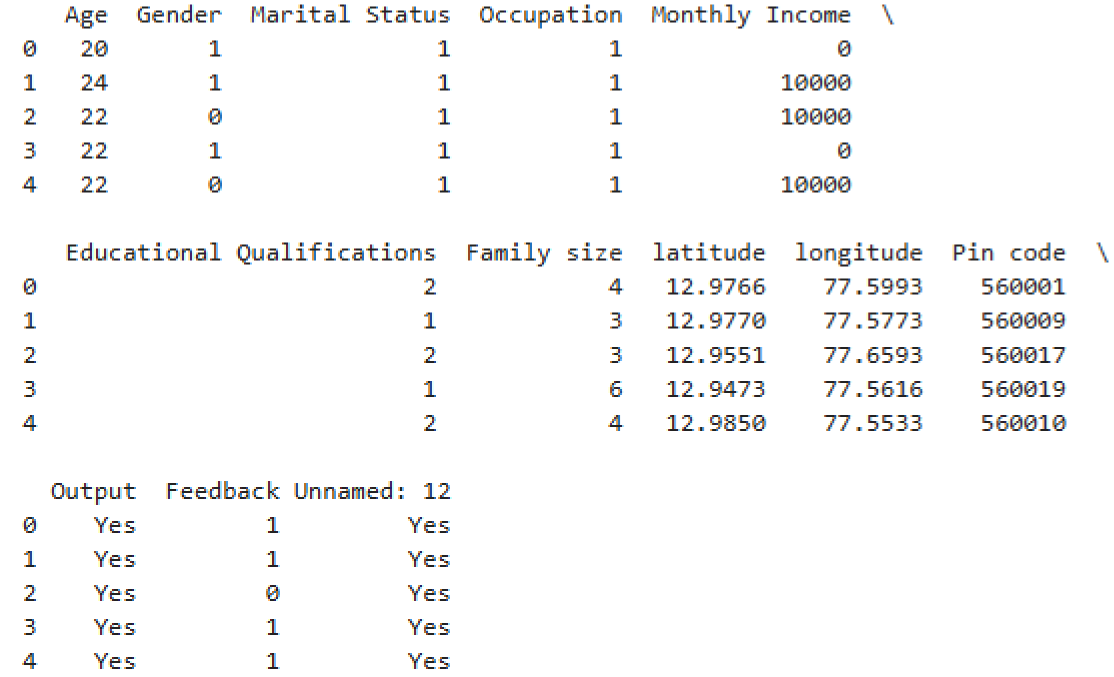


According to the above pie plot, 54% of the customers don’t fall under any income group which includes housewives or students.

Now let’s prepare the data for the task of training a machine learning model. Here I will convert all the categorical features into numerical values:



```
data["Gender"] = data["Gender"].map({"Male": 0, "Female": 1})

data["Marital Status"] = data["Marital Status"].map({"Married": 2,"Single": 1,
                                                     "Prefer not to say": 0})

data["Occupation"] = data["Occupation"].map({"Student": 1, "Employee": 2,
                                             "Self Employeed": 3,
                                             "House wife": 4})

data["Educational Qualifications"] = data["Educational Qualifications"].map({"Graduate": 1,  "Post Graduate": 2,
                                                                             "Ph.D": 3, "School": 4,
                                                                             "Uneducated": 5})
data["Monthly Income"] = data["Monthly Income"].map({"No Income": 0, "25001 to 50000": 5000,
                                                     "More than 50000": 7000,
                                                     "10001 to 25000": 25000,
                                                     "Below Rs.10000": 10000})

data["Feedback"] = data["Feedback"].map({"Positive": 1, "Negative ": 0})

print(data.head())
```

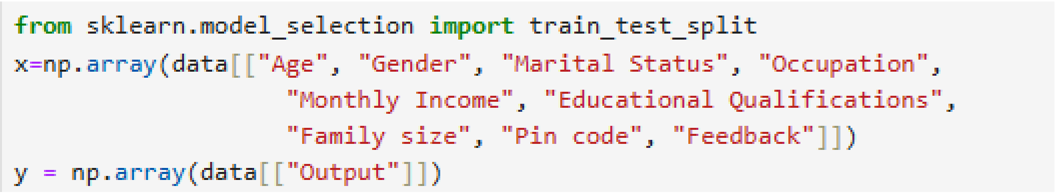


```
   Age  Gender  Marital Status  Occupation  Monthly Income  \
0   20       1               1           1               0
1   24       1               1           1           10000
2   22       0               1           1           10000
3   22       1               1           1               0
4   22       0               1           1           10000

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

  Output  Feedback Unnamed: 12
0    Yes         1         Yes
1    Yes         1         Yes
2    Yes         0         Yes
3    Yes         1         Yes
4    Yes         1         Yes
```

Online Food Order Prediction Model

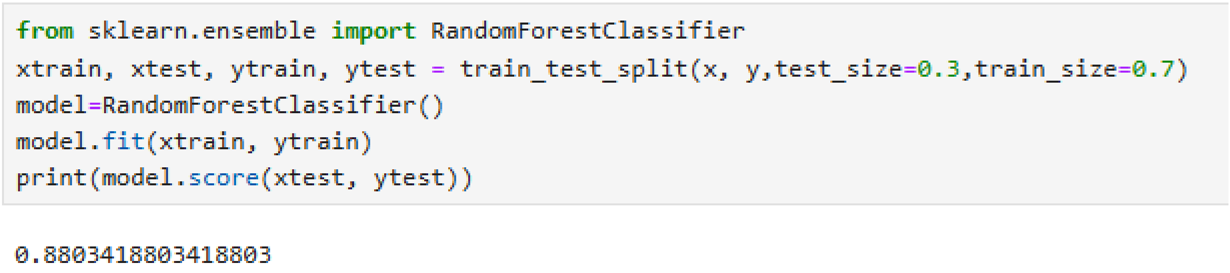
Now let’s train a machine learning model to predict whether a customer will order again or not. Here I’m starting by splitting the data into training and test sets:

#splitting data



```python
from sklearn.model_selection import train_test_split
x=np.array(data[["Age", "Gender", "Marital Status", "Occupation",
                   "Monthly Income", "Educational Qualifications",
                   "Family size", "Pin code", "Feedback"]])
y = np.array(data[["Output"]])
```

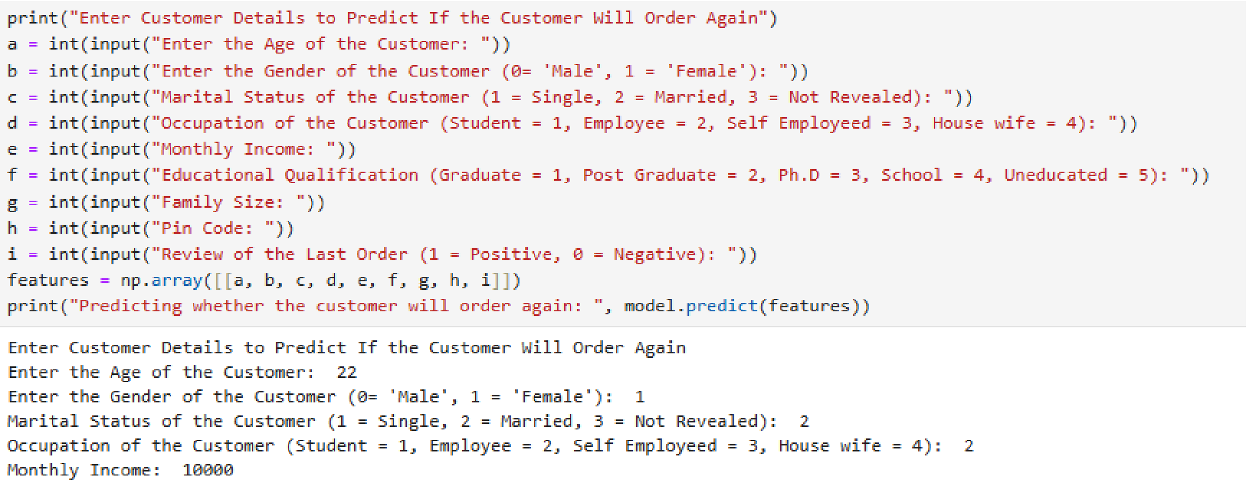
Let’s going through the **training of Machine Learning model**.



```python
from sklearn.ensemble import RandomForestClassifier
xtrain, xtest, ytrain, ytest = train_test_split(x, y,test_size=0.3,train_size=0.7)
model=RandomForestClassifier()
model.fit(xtrain, ytrain)
print(model.score(xtest, ytest))
```

```
0.8803418803418803
```

Let's build a form aimed at collecting customer data to evaluate whether they are likely to place another food order.



```python
print("Enter Customer Details to Predict If the Customer Will Order Again")
a = int(input("Enter the Age of the Customer: "))
b = int(input("Enter the Gender of the Customer (0= 'Male', 1 = 'Female'): "))
c = int(input("Marital Status of the Customer (1 = Single, 2 = Married, 3 = Not Revealed): "))
d = int(input("Occupation of the Customer (Student = 1, Employee = 2, Self Employeed = 3, House wife = 4): "))
e = int(input("Monthly Income: "))
f = int(input("Educational Qualification (Graduate = 1, Post Graduate = 2, Ph.D = 3, School = 4, Uneducated = 5): "))
g = int(input("Family Size: "))
h = int(input("Pin Code: "))
i = int(input("Review of the Last Order (1 = Positive, 0 = Negative): "))
features = np.array([[a, b, c, d, e, f, g, h, i]])
print("Predicting whether the customer will order again: ", model.predict(features))
```

```
Enter Customer Details to Predict If the Customer Will Order Again
Enter the Age of the Customer:  22
Enter the Gender of the Customer (0= 'Male', 1 = 'Female'):  1
Marital Status of the Customer (1 = Single, 2 = Married, 3 = Not Revealed):  2
Occupation of the Customer (Student = 1, Employee = 2, Self Employeed = 3, House wife = 4):  2
Monthly Income:  10000
```

These are the steps to describes how to create a machine learning model capable of predicting online food orders.

Summary

In conclusion this is how you can determine whether a customer will reorder food online. The food order prediction system is a practical technique that food delivery companies can adopt to improve the speed of their delivery operations.