Setting Up:

We'll begin by bringing in (importing) the tools (libraries) we'll need from Python and loading the data set.

In [ ]:
```
import numpy as np
import pandas as pd
import numpy as np
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())

import plotly.io as pio
pio.renderers.default = 'notebook'
```

```
   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  
```

Exploring the Data:

This data set includes details like a customer's age, marital status, job, monthly income, education level, family size, location (latitude and longitude), home postal code, whether they ordered again, and feedback from their last order (positive or negative). Let's take a closer look at the information for each category (column) in the data set.

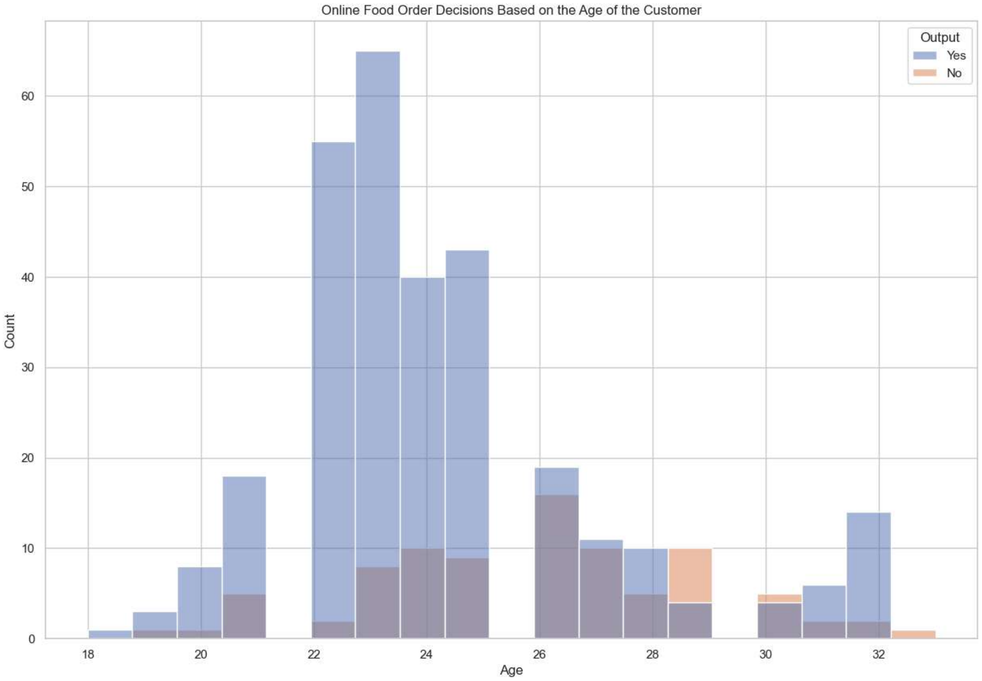
In [ ]:
```
print(data.info())
```

```
<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
```

Analyzing Online Food Ordering Trends:

Now, we'll dive into analyzing this data. First, we'll examine online food ordering decisions based on a customer's age.

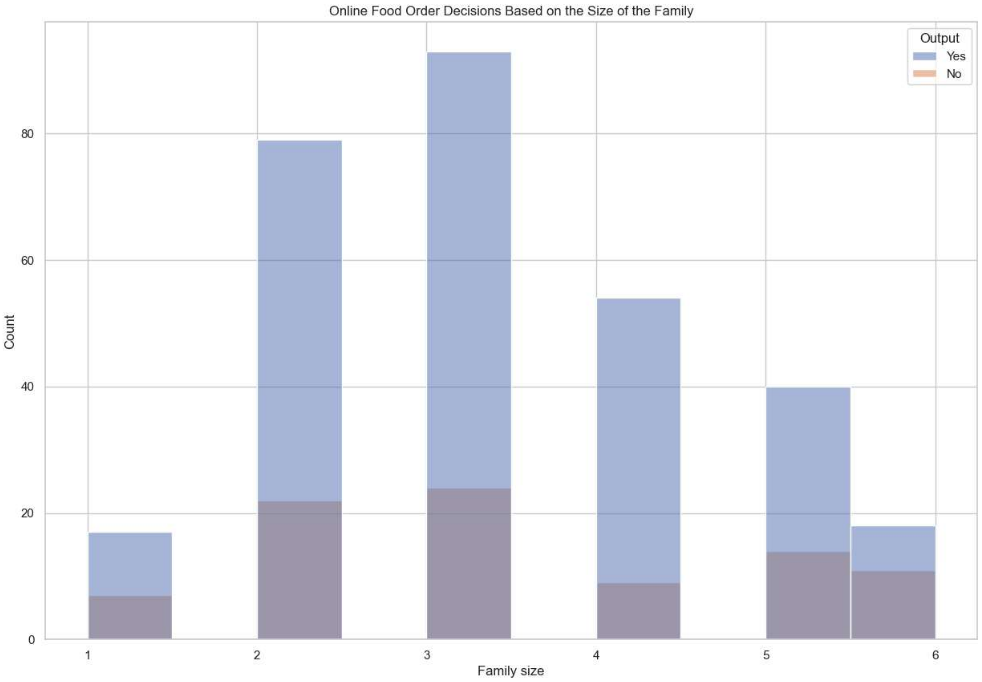
In [ ]:
```
plt.figure(figsize=(15, 10))
plt.title("Online Food Order Decisions Based on the Age of the Customer")
sns.histplot(x="Age", hue="Output", data=data)
plt.show()
```



The data shows that the 22-25 age group orders the most frequently. This suggests they're a target market for online food delivery companies.

Next, let's look at online food ordering decisions based on family size.

In [ ]:
```
plt.figure(figsize=(15,10))
plt.title("Online Food Order Decisions Based on the Size of the Family")
sns.histplot(x='Family size', hue='Output', data=data)
plt.show()
```



Families of 2 and 3 members order food the most often. These could be roommates, couples, or small families.

Focusing on Repeat Customers: Let's create a new data set that only includes customers who ordered again

In [ ]:
```
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
```

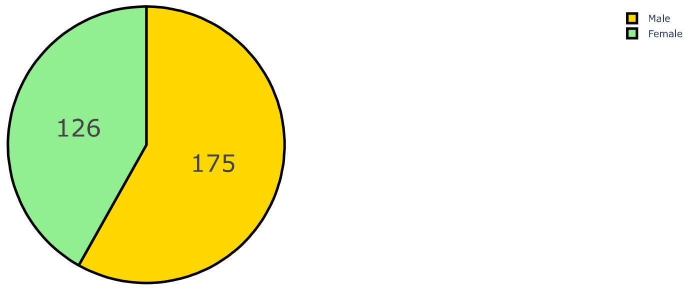
Gender and Repeat Orders:

Now, let's look at the gender data. We want to find out who orders food online more often.

In [ ]:
```
gender = buying_again_data["Gender"].value_counts()
label = gender.index
counts = gender.values
colors = ['gold','lightgreen']

fig = go.Figure(data=[go.Pie(labels=label,values=counts)])
fig.update_layout(title_text='Who Orders Food Online More: Male Vs. Female')
fig.update_traces(hoverinfo='label+percent', textinfo = 'value', textfont_size = 30,marker = dict(colors=colors, line = dict(color='black', width=3)))
fig.show()
```

Who Orders Food Online More: Male Vs. Female



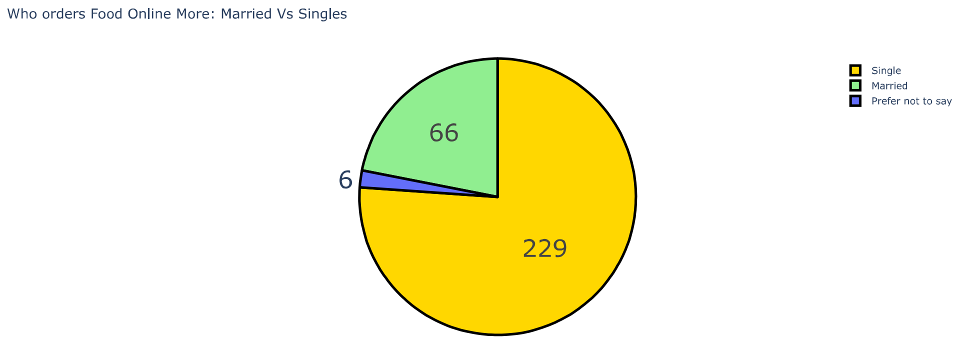
Based on the data, male customers seem to order more compared to females.

Marital Status and Repeat Orders:

Now, let's examine the marital status of customers who ordered again.

In [ ]:
```
marital = buying_again_data["Marital Status"].value_counts()
label = marital.index
counts = marital.values
colors = ['gold', 'lightgreen']
```

```
fig = go.Figure(data=[go.Pie(labels=label, values=counts)])
fig.update_layout(title_text='Who orders Food Online More: Married Vs Singles')
fig.update_traces(hoverinfo='label+percent', textinfo = 'value',textfont_size =30, marker = dict(colors=colors, line=dict(color ='black',width=3 )))
fig.show()
```



The graph shows that 76.1% of frequent customers are single.

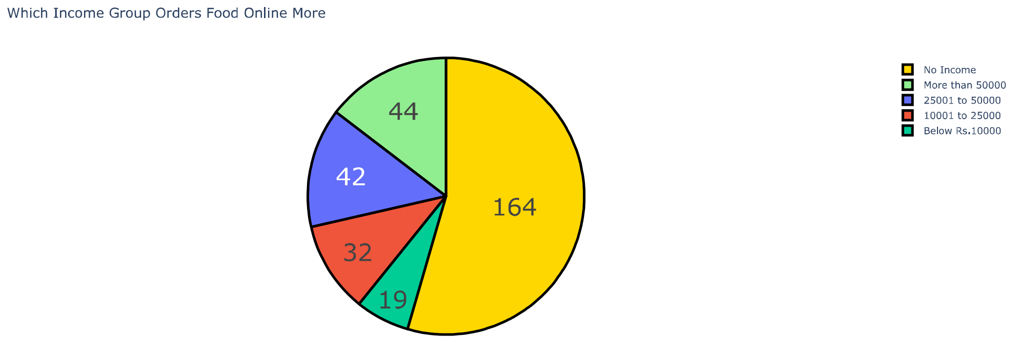
Income and Repeat Orders:

Let's see what the income bracket is for customers who ordered again.

In [ ]:

```
income = buying_again_data["Monthly Income"].value_counts()
label = income.index
counts = income.values
colors = ['gold','lightgreen']

fig = go.Figure(data=[go.Pie(labels=label, values = counts)])
fig.update_layout(title_text="Which Income Group Orders Food Online More")
fig.update_traces(hoverinfo='label+percent', textinfo="value", textfont_size=30,marker=dict(colors=colors, line=dict(color='black', width=3)))
fig.show()
```



According to the graph, 54% of these customers don't fall under any income group. They could be stay-at-home parents or students.

Preparing for Machine Learning:

Now, we'll get the data ready to train a machine learning model. This involves changing all the descriptive categories (categorical features) into numerical values.

In [ ]:

```
data['Gender'] = data["Gender"].map({"Male":1, "Female":0})
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       0               1         1.0             0.0   
1   24       0               1         1.0             NaN   
2   22       1               1         1.0             NaN   
3   22       0               1         1.0             0.0   
4   22       1               1         1.0             NaN   

   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       1.0         Yes  
1    Yes       1.0         Yes  
2    Yes       NaN         Yes  
3    Yes       1.0         Yes  
4    Yes       1.0         Yes  
```

Building a Prediction Model:

Online Food Order Prediction Model:

- We'll now train a machine learning model to forecast whether a customer will order again.
- First, we'll need to split the data into two sets: training data and test data.

In [ ]:

```
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"]])


from sklearn.ensemble import RandomForestClassifier
xtrain,xtest,ytrain,ytest =train_test_split(x,y,test_size=0.10,random_state=42)
model = RandomForestClassifier()
model.fit(xtrain,ytrain)
print(model.score(xtest,ytest))
```

```
0.9230769230769231
```

```
C:\Users\okoro\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\sklearn\base.py:1474: DataConversionWarning:

A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
```

Now, let's create a form where users can enter customer data, and the model will predict whether the customer will order food again.

In [ ]:

```
print("Enter Customer Detail to Predict If Customer Will Order Again")
a = int(input("Enter the Age of the Customer: "))
b = int(input("Enter the Gender of the Customer(1 = Male, 0 = Female): "))
c = int(input("Marital Status of the Customer(1 = Sinagle, 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 = Negetive: "))
features = np.array([[a,b,c,d,e,f,g,h,i]])
print("Finding if the customer will order again: ", model.predict(features))
```

```
Enter Customer Detail to Predict If Customer Will Order Again
Finding if the customer will order again:  ['Yes']
```

Conclusion:

- This is how you can train a machine learning model to predict online food orders.
- Food order prediction systems are valuable tools that food delivery companies can use to streamline their delivery processes.

I hope you enjoyed this aexploration on Online Food Delivery Prediction with Machine Learning. Feel free to ask any questions you may have in the comments below.