prompt-SRN.06.27.2022

June 29, 2022

0.0.1 Module 5 Project Report (Shaji R. Nathan)

Will a Customer Accept the Coupon? Context

Imagine driving through town and a coupon is delivered to your cell phone for a restaraunt near where you are driving. Would you accept that coupon and take a short detour to the restaraunt? Would you accept the coupon but use it on a sunbsequent trip? Would you ignore the coupon entirely? What if the coupon was for a bar instead of a restaraunt? What about a coffee house? Would you accept a bar coupon with a minor passenger in the car? What about if it was just you and your partner in the car? Would weather impact the rate of acceptance? What about the time of day?

Obviously, proximity to the business is a factor on whether the coupon is delivered to the driver or not, but what are the factors that determine whether a driver accepts the coupon once it is delivered to them? How would you determine whether a driver is likely to accept a coupon?

Overview

The goal of this project is to use what you know about visualizations and probability distributions to distinguish between customers who accepted a driving coupon versus those that did not.

Data

This data comes to us from the UCI Machine Learning repository and was collected via a survey on Amazon Mechanical Turk. The survey describes different driving scenarios including the destination, current time, weather, passenger, etc., and then ask the person whether he will accept the coupon if he is the driver. Answers that the user will drive there 'right away' or 'later before the coupon expires' are labeled as 'Y = 1' and answers 'no, I do not want the coupon' are labeled as 'Y = 0'. There are five different types of coupons -- less expensive restaurants (under \$20), coffee houses, carry out & take away, bar, and more expensive restaurants (\$20 - \$50).

Deliverables

Your final product should be a brief report that highlights the differences between customers who did and did not accept the coupons. To explore the data you will utilize your knowledge of plotting, statistical summaries, and visualization using Python. You will publish your findings in a public facing github repository as your first portfolio piece.

0.0.2 Data Description

The attributes of this data set include: 1. User attributes - Gender: male, female - Age: below 21, 21 to 25, 26 to 30, etc. - Marital Status: single, married partner, unmarried partner, or widowed - Number of children: 0, 1, or more than 1 - Education: high school, bachelors degree, associates degree, or graduate degree - Occupation: architecture & engineering, business & financial, etc. -

Annual income: less than \$12500, \$12500 - \$24999, \$25000 - \$37499, etc. - Number of times that he/she goes to a bar: 0, less than 1, 1 to 3, 4 to 8 or greater than 8 - Number of times that he/she buys takeaway food: 0, less than 1, 1 to 3, 4 to 8 or greater than 8 - Number of times that he/she goes to a coffee house: 0, less than 1, 1 to 3, 4 to 8 or greater than 8 - Number of times that he/she eats at a restaurant with average expense less than \$20 per person: 0, less than 1, 1 to 3, 4 to 8 or greater than 8 - Number of times that he/she goes to a bar: 0, less than 1, 1 to 3, 4 to 8 or greater than 8

2. Contextual attributes
    - Driving destination: home, work, or no urgent destination
    - Location of user, coupon and destination: we provide a map to show the geographical location of the user, destination, and the venue, and we mark the distance between each two places with time of driving. The user can see whether the venue is in the same direction as the destination.
    - Weather: sunny, rainy, or snowy
    - Temperature: 30F, 55F, or 80F
    - Time: 10AM, 2PM, or 6PM
    - Passenger: alone, partner, kid(s), or friend(s)
3. Coupon attributes
    - time before it expires: 2 hours or one day

```
In [1]: import matplotlib.pyplot as plt
        import seaborn as sns
        import pandas as pd
        import numpy as np
```

```
In [2]: # importing the sys module
        import sys

        # the setrecursionlimit function is
        # to modify the default recursion limit set by python. Using this,
        # Increased to make notebook load properly
        sys.setrecursionlimit(10**6)
```

0.0.3 Problems

Use the prompts below to get started with your data analysis.

1. Read in the `coupons.csv` file.

```
In [3]: data = pd.read_csv('data/coupons.csv')
```

```
In [4]: data.head()
```

```
Out[4]:        destination  passanger weather  temperature  time  \
        0  No Urgent Place      Alone   Sunny           55   2PM
        1  No Urgent Place  Friend(s)   Sunny           80  10AM
        2  No Urgent Place  Friend(s)   Sunny           80  10AM
```

```
3  No Urgent Place  Friend(s)   Sunny           80   2PM
4  No Urgent Place  Friend(s)   Sunny           80   2PM

                  coupon expiration  gender age      maritalStatus  ...  \
0        Restaurant(<20)         1d  Female  21  Unmarried partner  ...
1           Coffee House         2h  Female  21  Unmarried partner  ...
2  Carry out & Take away         2h  Female  21  Unmarried partner  ...
3           Coffee House         2h  Female  21  Unmarried partner  ...
4           Coffee House         1d  Female  21  Unmarried partner  ...

   CoffeeHouse CarryAway RestaurantLessThan20 Restaurant20To50  \
0        never       NaN                 4~8              1~3
1        never       NaN                 4~8              1~3
2        never       NaN                 4~8              1~3
3        never       NaN                 4~8              1~3
4        never       NaN                 4~8              1~3

  toCoupon_GEQ5min toCoupon_GEQ15min toCoupon_GEQ25min direction_same  \
0                1                 0                 0              0
1                1                 0                 0              0
2                1                 1                 0              0
3                1                 1                 0              0
4                1                 1                 0              0

  direction_opp  Y
0             1  1
1             1  0
2             1  1
3             1  0
4             1  0

[5 rows x 26 columns]
```

2. Investigate the dataset for missing or problematic data.

```
In [5]: #dimensions of the datframe

        data.shape

Out[5]: (12684, 26)

In [6]: #start with some libraries needed
        import plotly.express as px

In [7]: #Check the series data for unique values

        for col in data.columns:
            print('')
            print("{} column has {} unique values".format(col,data[col].nunique()))
```

```
destination column has 3 unique values
passanger column has 4 unique values
weather column has 3 unique values
temperature column has 3 unique values
time column has 5 unique values
coupon column has 5 unique values
expiration column has 2 unique values
gender column has 2 unique values
age column has 8 unique values
maritalStatus column has 5 unique values
has_children column has 2 unique values
education column has 6 unique values
occupation column has 25 unique values
income column has 9 unique values
car column has 5 unique values
Bar column has 5 unique values
CoffeeHouse column has 5 unique values
CarryAway column has 5 unique values
RestaurantLessThan20 column has 5 unique values
Restaurant20To50 column has 5 unique values
toCoupon_GEQ5min column has 1 unique values
toCoupon_GEQ15min column has 2 unique values
toCoupon_GEQ25min column has 2 unique values
direction_same column has 2 unique values
```

```
direction_opp column has 2 unique values

Y column has 2 unique values
```

```
In [8]: data['Y'].unique()
```

```
Out[8]: array([1, 0], dtype=int64)
```

```
In [9]: analyze=data.rename(columns={"Y":"coupon_redeem_status"})
        analyze.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12684 entries, 0 to 12683
Data columns (total 26 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   destination           12684 non-null  object
 1   passanger             12684 non-null  object
 2   weather               12684 non-null  object
 3   temperature           12684 non-null  int64
 4   time                  12684 non-null  object
 5   coupon                12684 non-null  object
 6   expiration            12684 non-null  object
 7   gender                12684 non-null  object
 8   age                   12684 non-null  object
 9   maritalStatus         12684 non-null  object
 10  has_children          12684 non-null  int64
 11  education             12684 non-null  object
 12  occupation            12684 non-null  object
 13  income                12684 non-null  object
 14  car                   108 non-null    object
 15  Bar                   12577 non-null  object
 16  CoffeeHouse           12467 non-null  object
 17  CarryAway             12533 non-null  object
 18  RestaurantLessThan20  12554 non-null  object
 19  Restaurant20To50      12495 non-null  object
 20  toCoupon_GEQ5min      12684 non-null  int64
 21  toCoupon_GEQ15min     12684 non-null  int64
 22  toCoupon_GEQ25min     12684 non-null  int64
 23  direction_same        12684 non-null  int64
 24  direction_opp         12684 non-null  int64
 25  coupon_redeem_status  12684 non-null  int64
dtypes: int64(8), object(18)
memory usage: 2.5+ MB
```

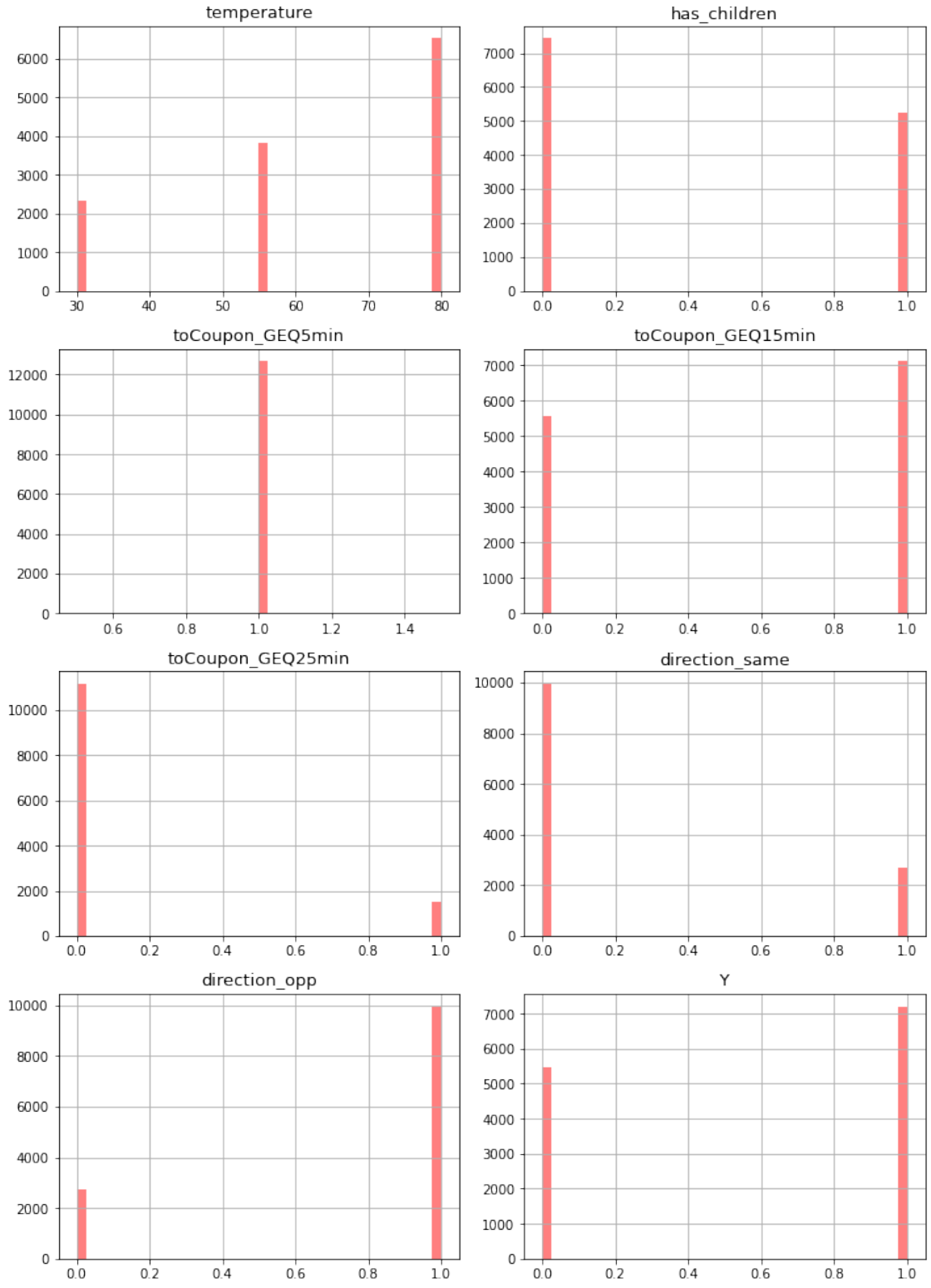
```
In [10]: #Examine the  numerical data in the data frame
```

```
numerical_values = data.select_dtypes(include = ['int64'])

numerical_values.head(3).T
```

Out[10]:

	0	1	2
temperature	55	80	80
has_children	1	1	1
toCoupon_GEQ5min	1	1	1
toCoupon_GEQ15min	0	0	1
toCoupon_GEQ25min	0	0	0
direction_same	0	0	0
direction_opp	1	1	1
Y	1	0	1

In [11]:
```
# Function to  visualize the numerical data

def display_hist(df, variables, n_rows, n_columns):
    fig=plt.figure()
    for i, var_name in enumerate(variables):
        ax=fig.add_subplot(n_rows,n_columns,i+1)
        df[var_name].hist(bins=40,ax=ax,color = 'red',alpha=0.5, figsize = (10, 15))
        ax.set_title(var_name, fontsize = 13)
        ax.tick_params(axis = 'both', which = 'major', labelsize = 10)
        ax.tick_params(axis = 'both', which = 'minor', labelsize = 10)
        ax.set_xlabel('')
    fig.tight_layout(rect = [0, 0.03, 1, 0.95])
    plt.show()

display_hist(numerical_values, numerical_values.columns, 4, 2)
```

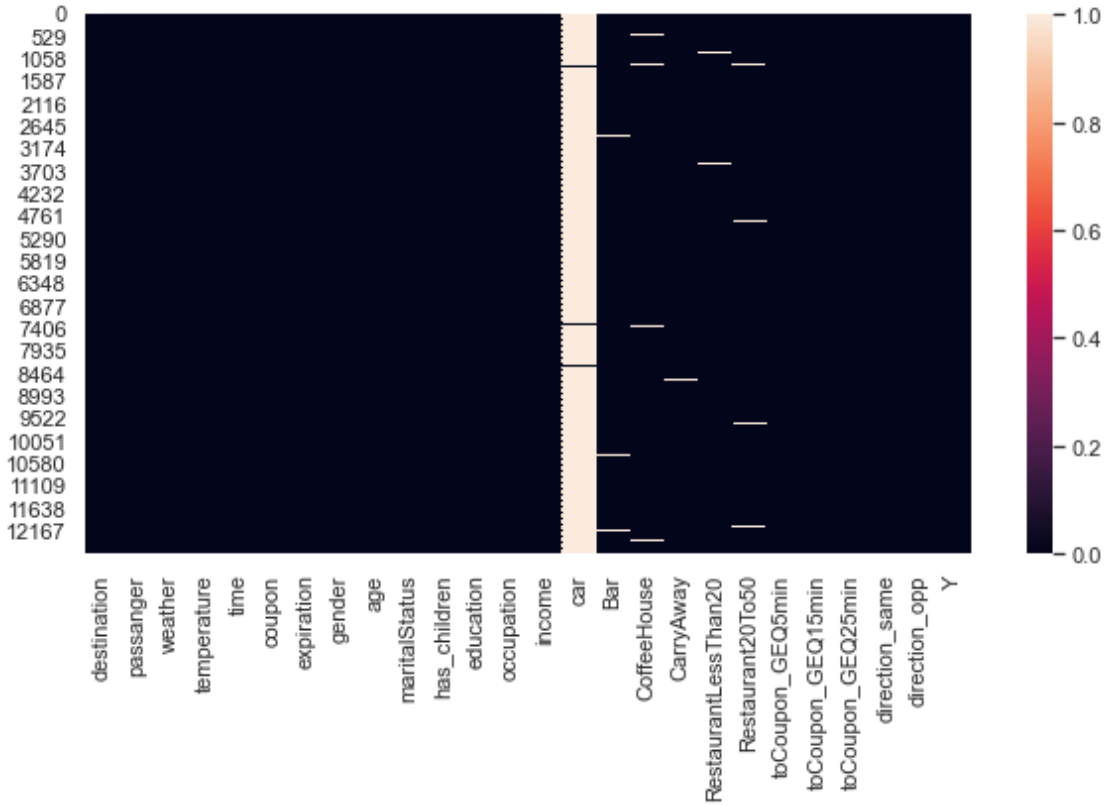


In [12]: *#Check for columns with missing data*

```
missing_columns = data.columns[data.isnull().any()].values
total_missing_columns = np.count_nonzero(data.isnull().sum())
print('Number of Columns with missing values: ', total_missing_columns, ' names of fea

#Visually inspect to verify whether there is missing data using a heatmap


sns.set(rc = {'figure.figsize':(10,5)})
sns.heatmap(data.isnull())
```

```
Number of Columns with missing values:  6  names of features:  ['car' 'Bar' 'CoffeeHouse' 'Car
 'Restaurant20To50']
```

Out[12]: <AxesSubplot:>



3. Decide what to do about your missing data -- drop, replace, other...

There are some missing values in several columns as can be seen from 11. 'car' column has 108 non-null values, which means more than 99% of the values are marked as "NaN". So this column can be dropped. The data given is insufficient for any kind of analysis, so it is best to remove this column.

```
In [13]: #Counts of unique values in the Car series

         print(analyze["car"].value_counts())

         #dropping the car series from the dataframe
         analyze.drop('car', inplace=True, axis=1)

         analyze.info()
```

```
do not drive                                22
Scooter and motorcycle                      22
Mazda5                                      22
Car that is too old to install Onstar :D    21
crossover                                   21
Name: car, dtype: int64
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12684 entries, 0 to 12683
Data columns (total 25 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   destination           12684 non-null  object
 1   passanger             12684 non-null  object
 2   weather               12684 non-null  object
 3   temperature           12684 non-null  int64
 4   time                  12684 non-null  object
 5   coupon                12684 non-null  object
 6   expiration            12684 non-null  object
 7   gender                12684 non-null  object
 8   age                   12684 non-null  object
 9   maritalStatus         12684 non-null  object
 10  has_children          12684 non-null  int64
 11  education             12684 non-null  object
 12  occupation            12684 non-null  object
 13  income                12684 non-null  object
 14  Bar                   12577 non-null  object
 15  CoffeeHouse           12467 non-null  object
 16  CarryAway             12533 non-null  object
 17  RestaurantLessThan20  12554 non-null  object
 18  Restaurant20To50      12495 non-null  object
 19  toCoupon_GEQ5min      12684 non-null  int64
 20  toCoupon_GEQ15min     12684 non-null  int64
 21  toCoupon_GEQ25min     12684 non-null  int64
 22  direction_same        12684 non-null  int64
 23  direction_opp         12684 non-null  int64
 24  coupon_redeem_status  12684 non-null  int64
dtypes: int64(8), object(17)
memory usage: 2.4+ MB
```

Next we find columns with empty or NaN values. For each column find the largest variable count and fill the empty values with a corresponding variable with maximum count.

```
In [14]: for x in analyze.columns[analyze.isna().any()]:
             analyze = analyze.fillna({x: analyze[x].value_counts().idxmax()})
```

Basic descriptive statistics and visualization of categories in the dataframe

```
In [15]: #Basic Statistics
         analyze.describe(include='all')
```

Out[15]:

	destination	passanger	weather	temperature	time	coupon	\
count	12684	12684	12684	12684.000000	12684	12684	
unique	3	4	3	NaN	5	5	
top	No Urgent Place	Alone	Sunny	NaN	6PM	Coffee House	
freq	6283	7305	10069	NaN	3230	3996	
mean	NaN	NaN	NaN	63.301798	NaN	NaN	
std	NaN	NaN	NaN	19.154486	NaN	NaN	
min	NaN	NaN	NaN	30.000000	NaN	NaN	
25%	NaN	NaN	NaN	55.000000	NaN	NaN	
50%	NaN	NaN	NaN	80.000000	NaN	NaN	
75%	NaN	NaN	NaN	80.000000	NaN	NaN	
max	NaN	NaN	NaN	80.000000	NaN	NaN	

	expiration	gender	age	maritalStatus	...	CoffeeHouse	CarryAway	\
count	12684	12684	12684	12684	...	12684	12684	
unique	2	2	8	5	...	5	5	
top	1d	Female	21	Married partner	...	less1	1~3	
freq	7091	6511	2653	5100	...	3602	4823	
mean	NaN	NaN	NaN	NaN	...	NaN	NaN	
std	NaN	NaN	NaN	NaN	...	NaN	NaN	
min	NaN	NaN	NaN	NaN	...	NaN	NaN	
25%	NaN	NaN	NaN	NaN	...	NaN	NaN	
50%	NaN	NaN	NaN	NaN	...	NaN	NaN	
75%	NaN	NaN	NaN	NaN	...	NaN	NaN	
max	NaN	NaN	NaN	NaN	...	NaN	NaN	

	RestaurantLessThan20	Restaurant20To50	toCoupon_GEQ5min	\
count	12684	12684	12684.0	
unique	5	5	NaN	
top	1~3	less1	NaN	
freq	5506	6266	NaN	
mean	NaN	NaN	1.0	
std	NaN	NaN	0.0	
min	NaN	NaN	1.0	
25%	NaN	NaN	1.0	
50%	NaN	NaN	1.0	
75%	NaN	NaN	1.0	

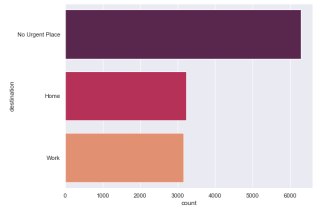
```
max                      NaN              NaN              1.0

       toCoupon_GEQ15min toCoupon_GEQ25min direction_same direction_opp  \
count       12684.000000      12684.000000   12684.000000  12684.000000
unique               NaN               NaN            NaN           NaN
top                  NaN               NaN            NaN           NaN
freq                 NaN               NaN            NaN           NaN
mean            0.561495          0.119126       0.214759      0.785241
std             0.496224          0.323950       0.410671      0.410671
min             0.000000          0.000000       0.000000      0.000000
25%             0.000000          0.000000       0.000000      1.000000
50%             1.000000          0.000000       0.000000      1.000000
75%             1.000000          0.000000       0.000000      1.000000
max             1.000000          1.000000       1.000000      1.000000

        coupon_redeem_status
count           12684.000000
unique                   NaN
top                      NaN
freq                     NaN
mean                0.568433
std                 0.495314
min                 0.000000
25%                 0.000000
50%                 1.000000
75%                 1.000000
max                 1.000000

[11 rows x 25 columns]
```

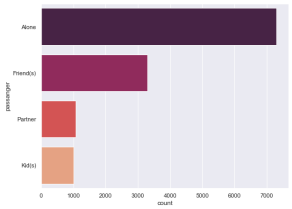
In [16]:
```
# Data types of the columnar data
analyze.dtypes
```

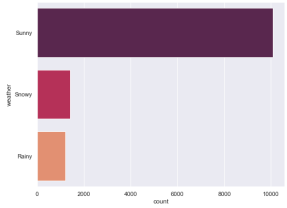
Out[16]:
```
destination             object
passanger               object
weather                 object
temperature              int64
time                    object
coupon                  object
expiration              object
gender                  object
age                     object
maritalStatus           object
has_children             int64
education               object
occupation              object
income                  object
Bar                     object
```

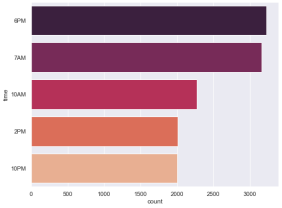
```
CoffeeHouse             object
CarryAway               object
RestaurantLessThan20    object
Restaurant20To50        object
toCoupon_GEQ5min         int64
toCoupon_GEQ15min        int64
toCoupon_GEQ25min        int64
direction_same           int64
direction_opp            int64
coupon_redeem_status     int64
dtype: object
```

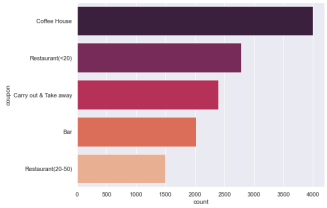
In [17]:
```
# Create distribution charts of all the object  datatypes that seem like categories.

fig, axes = plt.subplots(9, 2, figsize=(20,50))
axes = axes.flatten()

for ax, col in zip(axes, analyze.select_dtypes('object').columns):
    sns.countplot(y=col, data=analyze, ax=ax,
                  palette="rocket", order=analyze[col].value_counts().index);

plt.tight_layout()
plt.show()
```

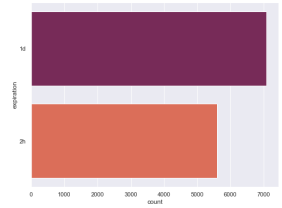

destination
No Urgent Place
Home
Work
0
1000
2000
3000
4000
5000
6000
count

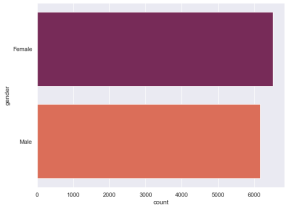

passanger
Alone
Friend(s)
Partner
Kid(s)
0
1000
2000
3000
4000
5000
6000
7000
count

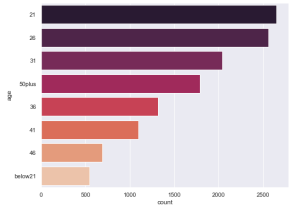

weather
Sunny
Snowy
Rainy
0
2000
4000
6000
8000
10000
count

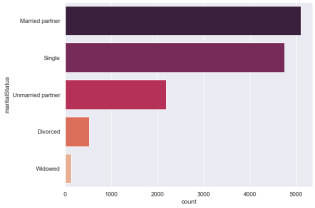

time
6PM
7AM
10AM
2PM
10PM
0
500
1000
1500
2000
2500
3000
count

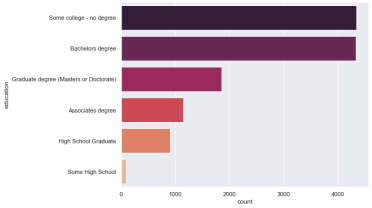

coupon
Coffee House
Restaurant(<20)
Carry out & Take away
Bar
Restaurant(20-50)
0
500
1000
1500
2000
2500
3000
3500
4000
count

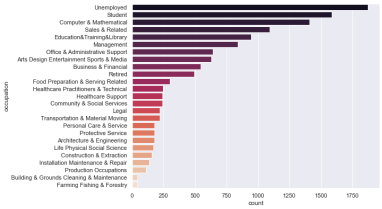

expiration
1d
2h
0
1000
2000
3000
4000
5000
6000
7000
count

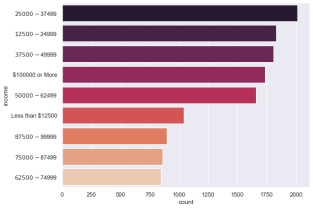

gender
Female
Male
0
1000
2000
3000
4000
5000
6000
count

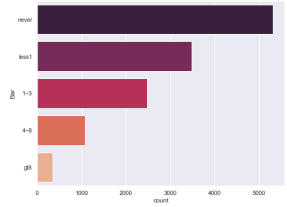

age
21
26
31
50plus
36
41
46
below21
0
500
1000
1500
2000
2500
count

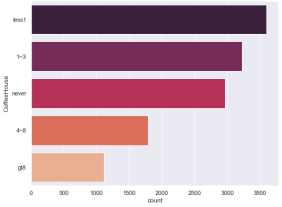

maritalStatus
Married partner
Single
Unmarried partner
Divorced
Widowed
0
1000
2000
3000
4000
5000
count

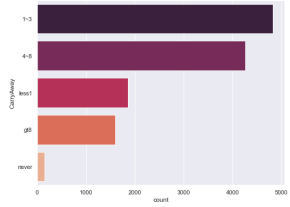

education
Some college - no degree
Bachelors degree
Graduate degree (Masters or Doctorate)
Associates degree
High School Graduate
Some High School
0
1000
2000
3000
4000
count

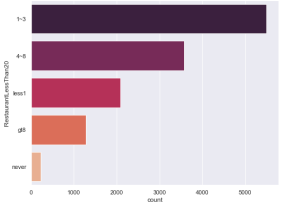

occupation
Unemployed
Student
Computer & Mathematical
Sales & Related
Education&Training&Library
Management
Office & Administrative Support
Arts Design Entertainment Sports & Media
Business & Financial
Retired
Food Preparation & Serving Related
Healthcare Practitioners & Technical
Healthcare Support
Community & Social Services
Legal
Transportation & Material Moving
Personal Care & Service
Protective Service
Architecture & Engineering
Life Physical Social Science
Construction & Extraction
Installation Maintenance & Repair
Production Occupations
Building & Grounds Cleaning & Maintenance
Farming Fishing & Forestry
0
250
500
750
1000
1250
1500
1750
count

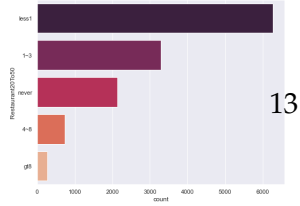

income
25000 – 37499
12500 – 24999
37500 – 49999
$100000 or More
50000 – 62499
Less than $12500
87500 – 99999
75000 – 87499
62500 – 74999
0
250
500
750
1000
1250
1500
1750
2000
count

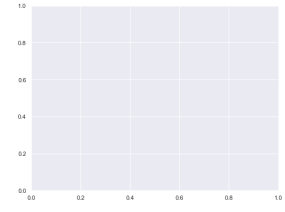

Bar
never
less1
1~3
4~8
gt8
0
1000
2000
3000
4000
5000
count


CoffeeHouse
less1
1~3
never
4~8
gt8
0
500
1000
1500
2000
2500
3000
3500
count


CarryAway
1~3
4~8
less1
gt8
never
0
1000
2000
3000
4000
5000
count


RestaurantLessThan20
1~3
4~8
less1
gt8
never
0
1000
2000
3000
4000
5000
count


Restaurant20To50
less1
1~3
never
4~8
gt8
0
1000
2000
3000
4000
5000
6000
count


1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0

4. What proportion of the total observations chose to accept the coupon?

```
In [18]: couponUsage=analyze['coupon_redeem_status'].value_counts()
         couponUsage

Out[18]: 1    7210
         0    5474
         Name: coupon_redeem_status, dtype: int64

In [19]: # Share of Coupon Accepted or rejected in the data
         df = analyze
         fig = px.pie(df, values=couponUsage, names=['Coupon Accepted','Coupon Rejected'], titl
         fig.update_traces(textinfo='value')

         fig.show()
```

Answer 1. A total of 12684 Coupons were issued. 7210 out of 12684 coupons issued were accepted by customers. 5474 coupons were rejected or not used.

5. Use a bar plot to visualize the coupon column.

```
In [20]: fig = px.histogram(analyze, y='coupon', color='age',title="Coupon Usage among differe
         fig.show()
```

6. Use a histogram to visualize the temperature column.

```
In [21]: fig = px.histogram(analyze, y='temperature', color='coupon', title="Coupon Usage acro
         fig.show()

In [22]: fig2 = px.histogram(analyze, x="age",color="coupon", title="Coupon Usage across age g
         fig2.show()
```

Investigating the Bar Coupons
Now, we will lead you through an exploration of just the bar related coupons.

1. Create a new DataFrame that contains just the bar coupons.

Answer Creating a New Dataframe containing just the bar coupons

```
In [23]: #Create a new Dataframe containing just the bar coupons

         BarCouponDf=analyze.query('coupon=="Bar"')
         BarCouponDf.head(10)
```

Out[23]:

	destination	passanger	weather	temperature	time	coupon	expiration	\
9	No Urgent Place	Kid(s)	Sunny	80	10AM	Bar	1d	
13	Home	Alone	Sunny	55	6PM	Bar	1d	
17	Work	Alone	Sunny	55	7AM	Bar	1d	
24	No Urgent Place	Friend(s)	Sunny	80	10AM	Bar	1d	
35	Home	Alone	Sunny	55	6PM	Bar	1d	
39	Work	Alone	Sunny	55	7AM	Bar	1d	
46	No Urgent Place	Friend(s)	Sunny	80	10AM	Bar	1d	
57	Home	Alone	Sunny	55	6PM	Bar	1d	
61	Work	Alone	Sunny	55	7AM	Bar	1d	
75	No Urgent Place	Kid(s)	Sunny	80	10AM	Bar	1d	

	gender	age	maritalStatus	...	CoffeeHouse	CarryAway	\
9	Female	21	Unmarried partner	...	never	1~3	
13	Female	21	Unmarried partner	...	never	1~3	
17	Female	21	Unmarried partner	...	never	1~3	
24	Male	21	Single	...	less1	4~8	
35	Male	21	Single	...	less1	4~8	
39	Male	21	Single	...	less1	4~8	
46	Male	46	Single	...	4~8	1~3	
57	Male	46	Single	...	4~8	1~3	
61	Male	46	Single	...	4~8	1~3	
75	Male	46	Married partner	...	1~3	1~3	

	RestaurantLessThan20	Restaurant20To50	toCoupon_GEQ5min	toCoupon_GEQ15min	\
9	4~8	1~3	1	1	
13	4~8	1~3	1	0	
17	4~8	1~3	1	1	
24	4~8	less1	1	0	
35	4~8	less1	1	0	
39	4~8	less1	1	1	
46	1~3	never	1	0	
57	1~3	never	1	0	
61	1~3	never	1	1	
75	1~3	less1	1	1	

	toCoupon_GEQ25min	direction_same	direction_opp	coupon_redeem_status
9	0	0	1	0
13	0	1	0	1
17	1	0	1	0
24	0	0	1	1
35	0	1	0	1
39	1	0	1	1
46	0	0	1	0
57	0	1	0	0
61	1	0	1	0
75	0	0	1	1

```
[10 rows x 25 columns]
```

2. What proportion of bar coupons were accepted?

```
In [24]: barcouponUsage=BarCouponDf['coupon_redeem_status'].value_counts()
         barcouponUsage

Out[24]: 0    1190
         1     827
         Name: coupon_redeem_status, dtype: int64

In [25]: # Share of Coupon Accepted or rejected in the data

         fig = px.pie(BarCouponDf, values=barcouponUsage, names=['Coupon Rejected','Coupon Acc
         fig.update_traces(textinfo='value')

         fig.show()
```

Answer 2. A total of 2017 Bar Coupons were issued. 827 out of 2017 coupons issued were accepted by customers. 1190 coupons were rejected or not used.

3. Compare the acceptance rate between those who went to a bar 3 or fewer times a month to those who went more.

```
In [26]: analyze["Bar"].unique()

Out[26]: array(['never', 'less1', '1~3', 'gt8', '4~8'], dtype=object)

In [27]: barThreefew=BarCouponDf[BarCouponDf.Bar.isin(["never", "less1", "1~3"])]
         barThreefew["Bar"].unique()

Out[27]: array(['never', 'less1', '1~3'], dtype=object)

In [28]: barThreefewUsage=barThreefew['coupon_redeem_status'].value_counts()
         barThreefewUsage

Out[28]: 0    1144
         1     674
         Name: coupon_redeem_status, dtype: int64

In [29]: # Share of Coupon Accepted or rejected by customers who went to the bar 3 or fewer ti

         fig = px.pie(barThreefew, values= barThreefewUsage,names=['Not Accepted','Accepted'],t
         fig.show()

In [30]: barFourMore=BarCouponDf[BarCouponDf.Bar.isin(['4~8','gt8'])]
         barFourMore["Bar"].unique()

Out[30]: array(['gt8', '4~8'], dtype=object)
```

```
In [31]: barFourMoreUsage=barFourMore['coupon_redeem_status'].value_counts()
         barFourMoreUsage

Out[31]: 1    153
         0     46
         Name: coupon_redeem_status, dtype: int64

In [32]: # Share of Coupon Accepted or rejected by customers who went to the 4 or more times a

         fig = px.pie(barFourMore, values= barFourMoreUsage,names=['Not Accepted','Accepted'],t
         fig.show()
```

Answer 3. 23.1% percent of the customers who went to the bar four or more times used Bar Coupons compared to 37.1% of customers who went to the bar 3 times or less. Bar coupons are less likely to be used by customers who frequently go to the bar.

4. Compare the acceptance rate between drivers who go to a bar more than once a month and are over the age of 25 to the all others. Is there a difference?

```
In [33]: barOnceOverTwentyFive=BarCouponDf.query('Bar.isin(["1~3","4~8","gt8"]) & age>"25"')
         barOnceOverTwentyFive.head()
```

Out[33]:

	destination	passanger	weather	temperature	time	coupon	expiration	\
112	No Urgent Place	Friend(s)	Sunny	80	10AM	Bar	1d	
123	Home	Alone	Sunny	55	6PM	Bar	1d	
127	Work	Alone	Sunny	55	7AM	Bar	1d	
156	No Urgent Place	Friend(s)	Sunny	80	10AM	Bar	1d	
167	Home	Alone	Sunny	55	6PM	Bar	1d	

	gender	age	maritalStatus	...	CoffeeHouse	CarryAway	\
112	Male	26	Unmarried partner	...	gt8	4~8	
123	Male	26	Unmarried partner	...	gt8	4~8	
127	Male	26	Unmarried partner	...	gt8	4~8	
156	Male	26	Single	...	gt8	gt8	
167	Male	26	Single	...	gt8	gt8	

	RestaurantLessThan20	Restaurant20To50	toCoupon_GEQ5min	toCoupon_GEQ15min	\
112	1~3	less1	1	0	
123	1~3	less1	1	0	
127	1~3	less1	1	1	
156	gt8	gt8	1	0	
167	gt8	gt8	1	0	

	toCoupon_GEQ25min	direction_same	direction_opp	coupon_redeem_status
112	0	0	1	1
123	0	1	0	1
127	1	0	1	1
156	0	0	1	1
167	0	1	0	1

```
[5 rows x 25 columns]
```

In [34]:
```
# Share of Coupon Accepted or rejected by customers who went to the bar at least onc

fig = px.pie(barOnceOverTwentyFive, values= barOnceOverTwentyFive['coupon_redeem_statu
fig.show()
```

In [35]:
```
#all others

barallOthers=BarCouponDf.query('Bar.isin(["never","less1"]) & age<="25"')
barallOthers.head()
```

Out[35]:

	destination	passanger	weather	temperature	time	coupon	expiration	...	gender	age	maritalStatus	...	CoffeeHouse	CarryAway	RestaurantLessThan20	Restaurant20To50	toCoupon_GEQ5min	toCoupon_GEQ15min	toCoupon_GEQ25min	direction_same	direction_opp	coupon_redeem_status
9	No Urgent Place	Kid(s)	Sunny	80	10AM	Bar	1d	...	Female	21	Unmarried partner	...	never	1~3	4~8	1~3	1	1	0	0	1	0
13	Home	Alone	Sunny	55	6PM	Bar	1d	...	Female	21	Unmarried partner	...	never	1~3	4~8	1~3	1	0	0	1	0	1
17	Work	Alone	Sunny	55	7AM	Bar	1d	...	Female	21	Unmarried partner	...	never	1~3	4~8	1~3	1	1	1	0	1	0
24	No Urgent Place	Friend(s)	Sunny	80	10AM	Bar	1d	...	Male	21	Single	...	less1	4~8	4~8	less1	1	0	0	0	1	1
35	Home	Alone	Sunny	55	6PM	Bar	1d	...	Male	21	Single	...	less1	4~8	4~8	less1	1	0	0	1	0	1

```
[5 rows x 25 columns]
```

In [36]:
```
# Share of Coupon Accepted or rejected by all other customers < 25 years

fig = px.pie(barallOthers, values= barallOthers['coupon_redeem_status'].value_counts()
fig.show()
```

Answer 4. 31.2% percent of the customers abover the age of 24 years who went to the bar more than one time used Bar Coupons compared to 38.8% of customers who were under the age of 25 went to the bar 3 times or less. Bar coupons are less likely to be used by customers who frequently go to the bar and are above 24 years.

5. Use the same process to compare the acceptance rate between drivers who go to bars more than once a month and had passengers that were not a kid and had occupations other than farming, fishing, or forestry.

```
In [37]: BarCouponDf['passanger'].unique()
```

```
Out[37]: array(['Kid(s)', 'Alone', 'Friend(s)', 'Partner'], dtype=object)
```

```
In [38]: BarCouponDf['occupation'].unique()
```

```
Out[38]: array(['Unemployed', 'Architecture & Engineering', 'Student',
                'Education&Training&Library', 'Healthcare Support',
                'Healthcare Practitioners & Technical', 'Sales & Related',
                'Management', 'Arts Design Entertainment Sports & Media',
                'Computer & Mathematical', 'Life Physical Social Science',
                'Personal Care & Service', 'Community & Social Services',
                'Office & Administrative Support', 'Construction & Extraction',
                'Legal', 'Retired', 'Installation Maintenance & Repair',
                'Transportation & Material Moving', 'Business & Financial',
                'Protective Service', 'Food Preparation & Serving Related',
                'Production Occupations',
                'Building & Grounds Cleaning & Maintenance',
                'Farming Fishing & Forestry'], dtype=object)
```

```
In [39]: filter_ = ~BarCouponDf['occupation'].isin(['Farming Fishing & Forestry'])
         print(filter_)
```

```
9         True
13        True
17        True
24        True
35        True
          ...
12663     True
12664     True
12667     True
12670     True
12682     True
Name: occupation, Length: 2017, dtype: bool
```

```
In [40]: BarCouponDfiltered= BarCouponDf[filter_]

         barOnceNoKids=BarCouponDfiltered.query('Bar.isin(["1~3","4~8","gt8"]) & passanger.isi

         barOnceNoKids.head()
```

Out[40]:

	destination	passanger	weather	temperature	time	coupon	expiration	gender	age	maritalStatus	...	CoffeeHouse	CarryAway	RestaurantLessThan20	Restaurant20To50	toCoupon_GEQ5min	toCoupon_GEQ15min	toCoupon_GEQ25min	direction_same	direction_opp	coupon_redeem_status
90	No Urgent Place	Friend(s)	Sunny	80	10AM	Bar	1d	Male	21	Single	...	less1	1~3	less1	1~3	1	0	0	0	1	1
101	Home	Alone	Sunny	55	6PM	Bar	1d	Male	21	Single	...	less1	1~3	less1	1~3	1	0	0	1	0	1
105	Work	Alone	Sunny	55	7AM	Bar	1d	Male	21	Single	...	less1	1~3	less1	1~3	1	1	1	0	1	0
112	No Urgent Place	Friend(s)	Sunny	80	10AM	Bar	1d	Male	26	Unmarried partner	...	gt8	4~8	1~3	less1	1	0	0	0	1	1
123	Home	Alone	Sunny	55	6PM	Bar	1d	Male	26	Unmarried partner	...	gt8	4~8	1~3	less1	1	0	0	1	0	1

[5 rows x 25 columns]

```
In [41]: # Share of Coupon Accepted or rejected by acceptance rate between drivers who go to b

         fig = px.pie(barOnceNoKids, values= barOnceNoKids['coupon_redeem_status'].value_counts
         fig.show()
```

6. Compare the acceptance rates between those drivers who:

- go to bars more than once a month, had passengers that were not a kid, and were not widowed *OR*
- go to bars more than once a month and are under the age of 30 *OR*
- go to cheap restaurants more than 4 times a month and income is less than 50K.

```
In [42]: #go to bars more than once a month
         filter2_ = BarCouponDf['Bar'].isin(["1~3","4~8","gt8"])
         print(filter2_)

         #go to bars more than once a month and had passengers that were not a kid
         filter3_ =~BarCouponDf['passanger'].isin(["kid"])
         print(filter3_)
```

```
        #go to bars more than once a month, had passengers that were not a kid, and were not
        filter4_=~BarCouponDf['maritalStatus'].isin(["widowed"])
        print(filter4_)
```

```
9        False
13       False
17       False
24       False
35       False
         ...
12663    False
12664    False
12667    False
12670    False
12682    False
Name: Bar, Length: 2017, dtype: bool
9        True
13       True
17       True
24       True
35       True
         ...
12663    True
12664    True
12667    True
12670    True
12682    True
Name: passanger, Length: 2017, dtype: bool
9        True
13       True
17       True
24       True
35       True
         ...
12663    True
12664    True
12667    True
12670    True
12682    True
Name: maritalStatus, Length: 2017, dtype: bool
```

```
In [43]: testdf=BarCouponDf[filter2_]
         testdf=BarCouponDf[filter3_]
         WidowedOnceNoKids=BarCouponDf[filter4_]
         WidowedOnceNoKids.head()
```

```
Out[43]:         destination  passanger weather  temperature  time coupon expiration  \
```

```
9   No Urgent Place     Kid(s)   Sunny          80  10AM    Bar         1d
13            Home      Alone   Sunny          55   6PM    Bar         1d
17            Work      Alone   Sunny          55   7AM    Bar         1d
24  No Urgent Place  Friend(s)   Sunny          80  10AM    Bar         1d
35            Home      Alone   Sunny          55   6PM    Bar         1d

    gender age      maritalStatus  ...  CoffeeHouse CarryAway  \
9   Female  21  Unmarried partner  ...        never       1~3
13  Female  21  Unmarried partner  ...        never       1~3
17  Female  21  Unmarried partner  ...        never       1~3
24    Male  21             Single  ...        less1       4~8
35    Male  21             Single  ...        less1       4~8

   RestaurantLessThan20 Restaurant20To50 toCoupon_GEQ5min toCoupon_GEQ15min  \
9                   4~8             1~3                1                 1
13                  4~8             1~3                1                 0
17                  4~8             1~3                1                 1
24                  4~8           less1                1                 0
35                  4~8           less1                1                 0

   toCoupon_GEQ25min direction_same direction_opp  coupon_redeem_status
9                  0              0             1                     0
13                 0              1             0                     1
17                 1              0             1                     0
24                 0              0             1                     1
35                 0              1             0                     1

[5 rows x 25 columns]
```

In [44]:
```
fig = px.pie(WidowedOnceNoKids, values= WidowedOnceNoKids['coupon_redeem_status'].valu
fig.show()
```

In [45]:
```
#go to bars more than once a month and are under the age of 30

barOnceUnder30=barOnceNoKids=BarCouponDf.query('Bar.isin(["1~3","4~8","gt8"]) & age <'

fig = px.pie(barOnceUnder30, values= barOnceUnder30['coupon_redeem_status'].value_cou
fig.show()
```

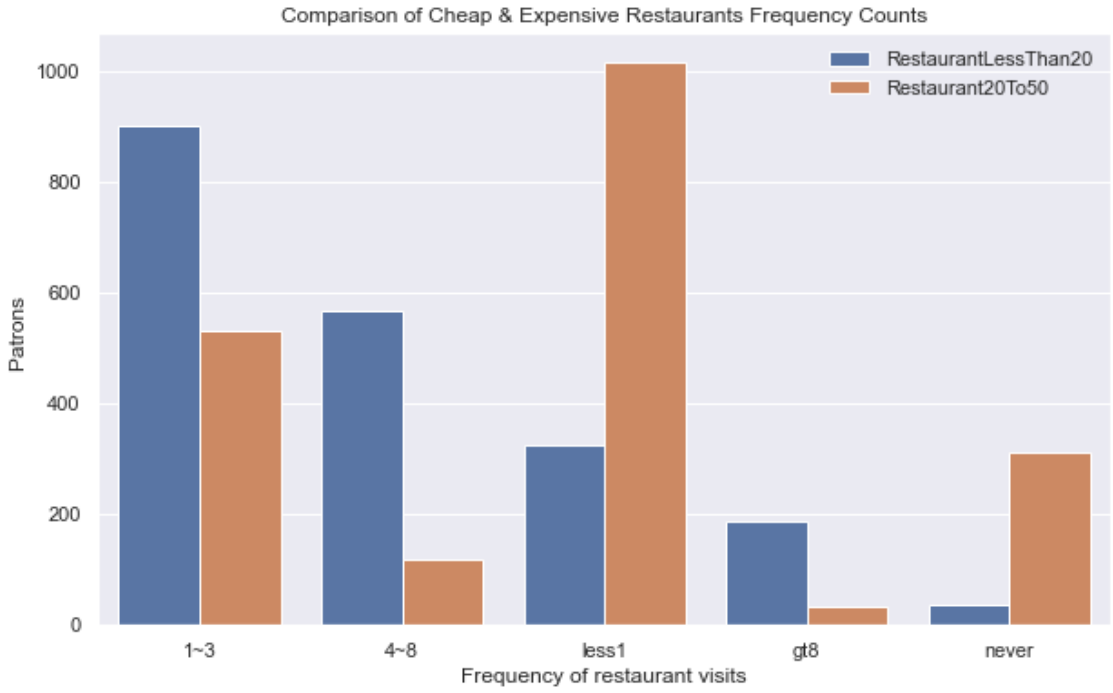
In [46]:
```
print(BarCouponDf["RestaurantLessThan20"].unique())
print(BarCouponDf["income"].unique())
```

```
['4~8' '1~3' 'less1' 'gt8' 'never']
['$37500 - $49999' '$62500 - $74999' '$12500 - $24999' '$75000 - $87499'
 '$50000 - $62499' '$25000 - $37499' '$100000 or More' '$87500 - $99999'
 'Less than $12500']
```

In [47]:
```
#Drivers who go to cheap restaurants more than 4 times a month and income is less tha
```

```
CheapRestFourMore=BarCouponDf.query('RestaurantLessThan20.isin(["1~3","less1"]) & inc

fig = px.pie(CheapRestFourMore, values= CheapRestFourMore['coupon_redeem_status'].valu
fig.show()
```

In [48]:
```
# Comparison of customer behavior for coupon responses for cheap and expensive cheap
cheaprest = BarCouponDf['RestaurantLessThan20'].value_counts()
exp = BarCouponDf['Restaurant20To50'].value_counts()

# combining frequencies of grps for cheap and expensive restaurants

combined = pd.merge(cheaprest,exp,left_index=True,right_index=True).reset_index()

# Rearrange for plotting

unpivotdf = pd.melt(combined,id_vars=['index'], value_vars=['RestaurantLessThan20','R

# plotting the data
plt.figure(figsize=(10,6))

sns.barplot(data=unpivotdf,x='index',y='value',hue='variable')
plt.xlabel("Frequency of restaurant visits")
plt.ylabel("Patrons ")
plt.title("Comparison of Cheap & Expensive Restaurants Frequency Counts")
plt.legend(frameon=False,loc='upper right')
plt.show()
```



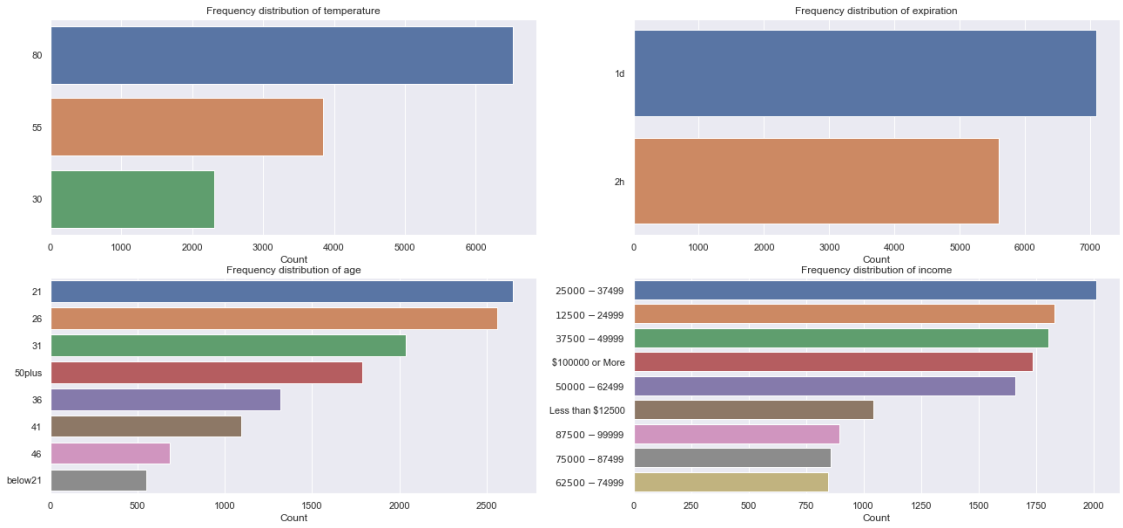
7. Based on these observations, what do you hypothesize about drivers who accepted the bar coupons?

Answer Most patronage is for cheaper restaurants. The visits range from 1 to 8 times. Bars and expensive restaurants, mostly reject coupons. It is not worthwile to issue coupons for patrons who frequent bars.

0.0.4 Independent Investigation

Using the bar coupon example as motivation, you are to explore one of the other coupon groups and try to determine the characteristics of passengers who accept the coupons.

In [49]:
```
#Reference Code Used:   https://stackoverflow.com/questions/31726643/how-to-plot-in-m

freqcolumns = ['temperature','expiration','age','income']

def calculate_frequency(nrows, ncols,figsize,cols_to_plot):
    '''
    Helper function to print countplots / frequency-distribution of each attribute
    '''
    fig,ax = plt.subplots(nrows=nrows, ncols=ncols,figsize=figsize)
    ax = ax.flatten()
    i = 0
    for col in analyze.columns:
        if col in cols_to_plot and col!='Y':  # we don't want to see distribution of
            if analyze[col].dtype==np.int64:   # Any numeric column is converted to c
                analyze[col] = analyze[col].astype(str)
            temp = df[col].value_counts()
            sns.barplot(x=temp.values,y=temp.index,ax=ax[i])
            ax[i].set_xlabel('Count')
            ax[i].set_title('Frequency distribution of {}'.format(col))

            i += 1

calculate_frequency(nrows=2, ncols=2,figsize=(22,10),cols_to_plot=freqcolumns)
```



Frequency Distribution of Categorical Data From the chart above we can see the distribution of various categorical values in the Bar Coupon Dataset. We can make the following statements based on the data:

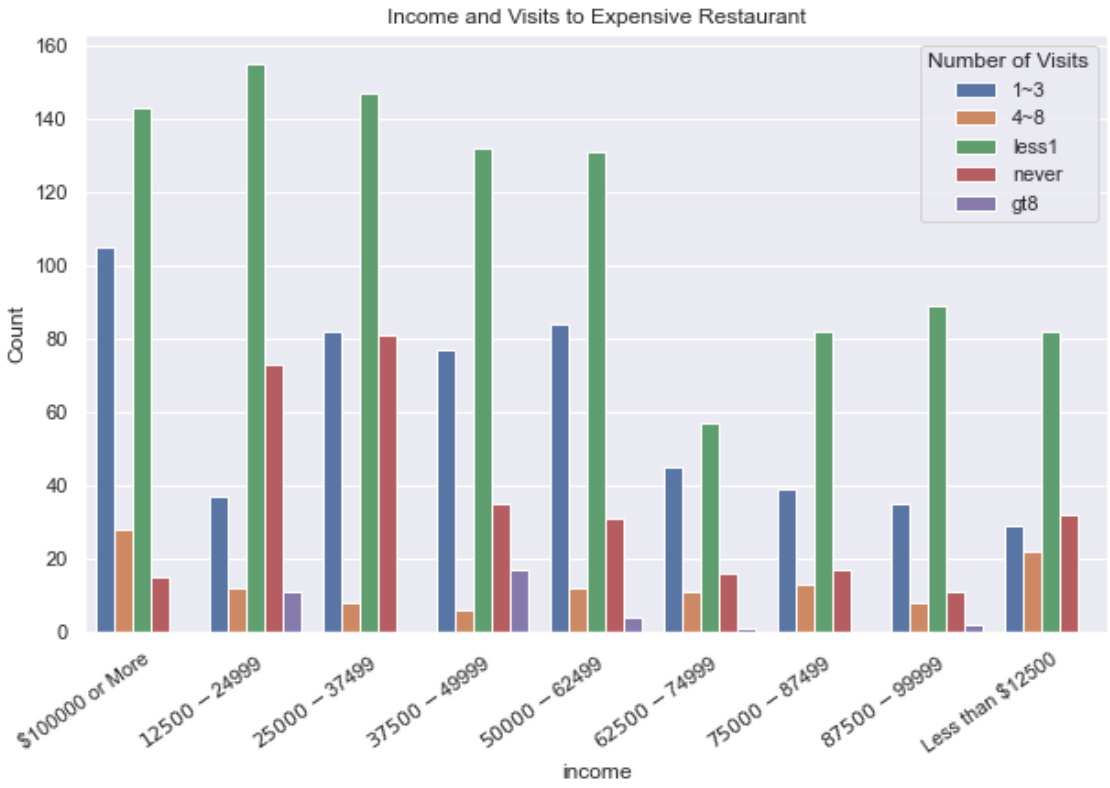
1. Temperature value recorded is mostly on the higher side, so most survey responses came from

2. Most of the responses came from drivers in the age group of 21-31 year olds and also tended

3. Coupon acceptance rate in regions with high temperature is more.

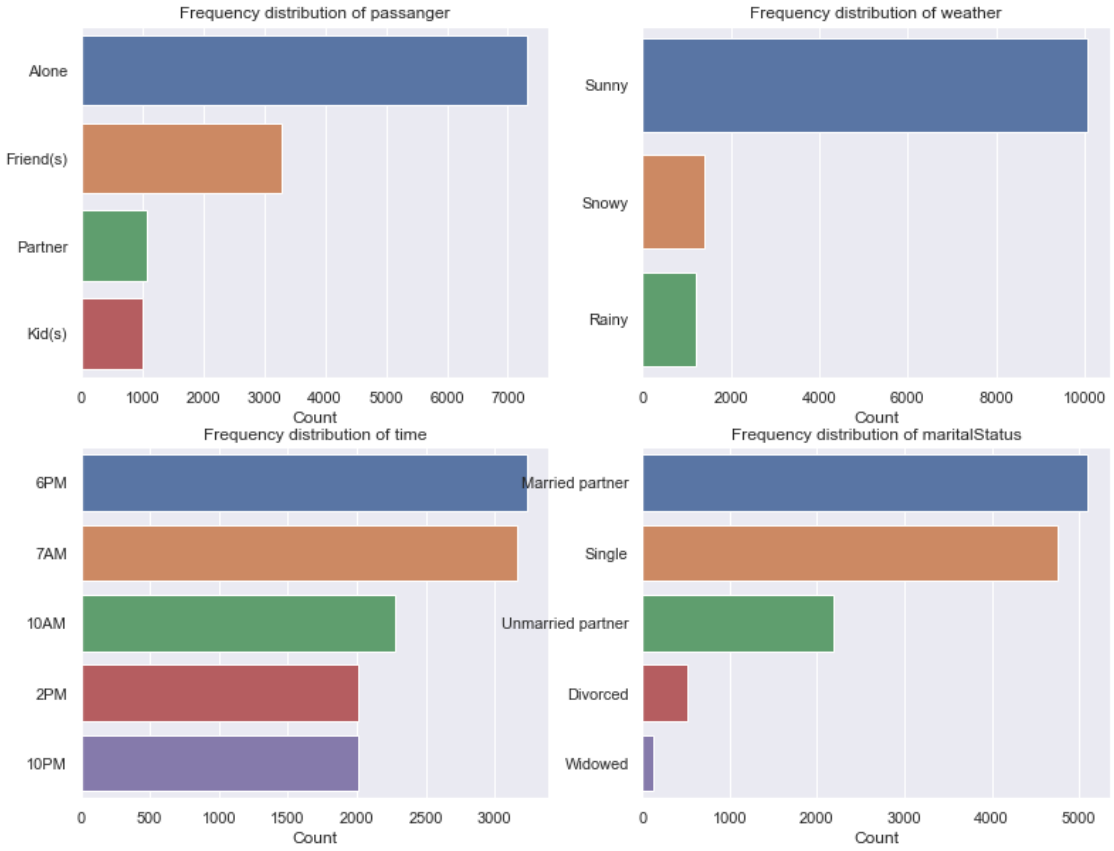
4. Coupon usage is prevalent regardless of age or income.

In [50]:
```
# Check for effect of income and visits to expensive restaurants

analyzetemp = pd.DataFrame(BarCouponDf.groupby(['income','Restaurant20To50']).size())
analyzetemp.columns = ['income','Restaurant20To50','values']

plt.figure(figsize=(10,6))
sns.barplot(x='income',y='values',hue='Restaurant20To50',data=analyzetemp)
plt.xticks(rotation = 35,rotation_mode = "anchor",ha = "right")
plt.ylabel("Count")
plt.title("Income and Visits to Expensive Restaurant")
plt.legend(title="Number of Visits",loc='upper right')
plt.show()
```



Expensive Restaurants are frequented by patrons with an income of Hundred thousand dollars or more and the visits range from 1-3

```
In [51]: calculate_frequency(nrows=2, ncols=2,figsize=(13,10),cols_to_plot=['passanger', 'weatl
```



Single drivers are more likely to use coupons in the mornings around 7, when the weather is fine followed by married people in the evenings around 6 PM Conclusion:

1. A total of 12684 Coupons were issued. 7210 out of 12684 coupons issued were accepted by customers. 5474 coupons were rejected or not used.
2. 23.1% percent of the customers who went to the bar four or more times used Bar Coupons compared to 37.1% of customers who went to the bar 3 times or less. Bar coupons are less likely to be used by customers who frequently go to the bar.
3. 31.2% percent of the customers abover the age of 24 years who went to the bar more than one time used Bar Coupons compared to 38.8% of customers who were under the age of 25 went to the bar 3 times or less. Bar coupons are less likely to be used by customers who frequently go to the bar and are above 24 years.
4. Most patronage is for cheaper restaurants. The visits range from 1 to 8 times
5. Single drivers are more likely to use coupons in the mornings around 7, when the weather is fine followed by married people in the evenings around 6 PM
6. Coupons are accepted by patrons of cheaper restaurants and takeaway. On the whole patrons of bars and expensive restaurants, mostly reject coupons. It is not worthwile to issue coupons for patrons who frequent bars.

Future Studies:

1. Current analysis is mostly based on exploratory data analysis. It would be interesting to see how different machine learning techniques can be applied to get more granular insights.