03-Ecommerce Purchases Exercise

October 25, 2019

___ # Ecommerce Purchases Exercise

In this Exercise you will be given some Fake Data about some purchases done through Amazon! Just go ahead and follow the directions and try your best to answer the questions and complete the tasks. Feel free to reference the solutions. Most of the tasks can be solved in different ways. For the most part, the questions get progressively harder.

Please excuse anything that doesn't make "Real-World" sense in the dataframe, all the data is fake and made-up.

Also note that all of these questions can be answered with one line of code. ____ ** Import pandas and read in the Ecommerce Purchases csv file and set it to a DataFrame called ecom. **

```
In [1]: import pandas as pd
```

```
In [2]: ecom = pd.read_csv('Ecommerce Purchases')
```

Check the head of the DataFrame.

```
In [3]: ecom.head()
```

```
Out[3]:                                              Address    Lot AM or PM  \
        0  16629 Pace Camp Apt. 448\nAlexisborough, NE 77...  46 in       PM
        1  9374 Jasmine Spurs Suite 508\nSouth John, TN 8...  28 rn       PM
        2                    Unit 0065 Box 5052\nDPO AP 27450  94 vE       PM
        3               7780 Julia Fords\nNew Stacy, WA 45798  36 vm       PM
        4  23012 Munoz Drive Suite 337\nNew Cynthia, TX 5...  20 IE       AM

                                             Browser Info  \
        0  Opera/9.56.(X11; Linux x86_64; sl-SI) Presto/2...
        1  Opera/8.93.(Windows 98; Win 9x 4.90; en-US) Pr...
        2  Mozilla/5.0 (compatible; MSIE 9.0; Windows NT ...
        3  Mozilla/5.0 (Macintosh; Intel Mac OS X 10_8_0 ...
        4  Opera/9.58.(X11; Linux x86_64; it-IT) Presto/2...

                                Company       Credit Card CC Exp Date  \
        0                 Martinez-Herman  6011929061123406       02/20
        1  Fletcher, Richards and Whitaker  3337758169645356       11/18
        2       Simpson, Williams and Pham      675957666125       08/19
```

```
3  Williams, Marshall and Buchanan  6011578504430710       02/24
4        Brown, Watson and Andrews  6011456623207998       10/25

   CC Security Code                  CC Provider  \
0               900                 JCB 16 digit
1               561                   Mastercard
2               699                 JCB 16 digit
3               384                     Discover
4               678  Diners Club / Carte Blanche

                             Email                                       Job  \
0               pdunlap@yahoo.com  Scientist, product/process development
1              anthony41@reed.com                        Drilling engineer
2  amymiller@morales-harrison.com                Customer service manager
3     brent16@olson-robinson.info                        Drilling engineer
4     christopherwright@gmail.com                              Fine artist

        IP Address Language  Purchase Price
0  149.146.147.205       el           98.14
1     15.160.41.51       fr           70.73
2   132.207.160.22       de            0.95
3     30.250.74.19       es           78.04
4     24.140.33.94       es           77.82
```

** How many rows and columns are there? **

```
In [4]: ecom.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 14 columns):
Address             10000 non-null object
Lot                 10000 non-null object
AM or PM            10000 non-null object
Browser Info        10000 non-null object
Company             10000 non-null object
Credit Card         10000 non-null int64
CC Exp Date         10000 non-null object
CC Security Code    10000 non-null int64
CC Provider         10000 non-null object
Email               10000 non-null object
Job                 10000 non-null object
IP Address          10000 non-null object
Language            10000 non-null object
Purchase Price      10000 non-null float64
dtypes: float64(1), int64(2), object(11)
memory usage: 1.1+ MB
```

** What is the average Purchase Price? **

```
In [5]: ecom['Purchase Price'].mean()
```

```
Out[5]: 50.34730200000025
```

** What were the highest and lowest purchase prices? **

```
In [6]: ecom['Purchase Price'].max()
```

```
Out[6]: 99.99
```

```
In [7]: ecom['Purchase Price'].min()
```

```
Out[7]: 0.0
```

** How many people have English 'en' as their Language of choice on the website? **

```
In [30]: len(ecom[ecom['Language']== 'en'])
```

```
Out[30]: 1098
```

** How many people have the job title of "Lawyer" ? **

```
In [31]: len(ecom[ecom['Job']=='Lawyer'])
```

```
Out[31]: 30
```

** How many people made the purchase during the AM and how many people made the purchase during PM ? **
(Hint: Check out value_counts())

```
In [21]: ecom['AM or PM'].value_counts()
```

```
Out[21]: PM    5068
         AM    4932
         Name: AM or PM, dtype: int64
```

** What are the 5 most common Job Titles? **

```
In [22]: ecom['Job'].value_counts().head(5)
```

```
Out[22]: Interior and spatial designer        31
         Lawyer                               30
         Social researcher                    28
         Research officer, political party    27
         Purchasing manager                   27
         Name: Job, dtype: int64
```

** Someone made a purchase that came from Lot: "90 WT" , what was the Purchase Price for this transaction? **

```
In [34]: ecom[ecom['Lot']=='90 WT']['Purchase Price']

Out[34]: 513    75.1
         Name: Purchase Price, dtype: float64
```

** What is the email of the person with the following Credit Card Number: 4926535242672853 **

```
In [61]: ecom[ecom['Credit Card']==4926535242672853]['Email']

Out[61]: 1234    bondellen@williams-garza.com
         Name: Email, dtype: object
```

** How many people have American Express as their Credit Card Provider *and* made a purchase above $95 ?**

```
In [48]: len(ecom[(ecom['CC Provider']=='American Express')&(ecom['Purchase Price']>95)])

Out[48]: 39
```

** Hard: How many people have a credit card that expires in 2025? **

```
In [53]: len(ecom[ecom['CC Exp Date'].apply(lambda exp: exp[3:]=='25')])

Out[53]: 1033
```

** Hard: What are the top 5 most popular email providers/hosts (e.g. gmail.com, yahoo.com, etc...) **

```
In [60]: ecom['Email'].apply(lambda email: email.split('@')[1]).value_counts().head(5)

Out[60]: hotmail.com     1638
         yahoo.com       1616
         gmail.com       1605
         smith.com         42
         williams.com      37
         Name: Email, dtype: int64
```

1 Great Job!