V2_full_osemn

November 7, 2025

1 Activity: Full OSEMN

1.1 Introduction

In this assignment, you will work on a data analysis project. This project will let you practice the skills you have learned in this course and write real code in Python.

You will perform the following steps of the OSEMN framework:
- Section 1.2 - Section 1.3 - Section 1.5

[1]:
```
# We'll import the libraries you'll likely use for this activity
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Data
df = pd.read_csv('transactions-pet_store.csv')
df_orig = df.copy()
```

1.2 Scrub

You will scrub the data. It's important that you follow the directions as stated. Doing more or less than what is asked might lead to not getting full points for the question.

If while you're working on the scrubbing phase you need to reset the DataFrame, you can restart the kernel (in the toolbar: "Kernel" > "Restart").

Question 1 Remove all rows that have are missing either the `Product_Name` or the `Product_Category`. Assign the cleaned DataFrame to the variable `df` (overwriting the original DataFrame.).

[11]:
```
# Your code here
df=df.dropna(subset=['Product_Name','Product_Category'])
df.describe()
# your code here
```

[11]:

	Price	Quantity
count	2758.000000	2758.000000
mean	25.621632	1.988397
std	535.327290	0.819961
min	-19873.000000	1.000000
25%	17.160000	1.000000
50%	25.480000	2.000000
75%	34.380000	3.000000
max	19873.000000	3.000000

[5]:
```
# Question 1 Grading Checks

assert df.shape[0] <= 2874, 'Did you remove all the rows with missing values␣
↪for the columns Product_Name & Product_Category?'
assert df.shape[0] >= 2700, 'Did you remove too many the rows with missing␣
↪values?'
assert len(df.columns) == 10, 'Make sure you do not drop any columns.'
```

Question 2 Find any clearly "incorrect" values in the `Price` column and "clean" the DataFrame to address those values.

Ensure you make the changes to the DataFrame assigned to the variable `df`.

[20]:
```
df_test = df[~(df['Product_Name'] == 'MissingNo Plush')]
df= df_test



# your code here
```

[21]:
```
# Question 2 Grading Checks

assert (df.Price < df.Price.quantile(0.0001)).sum() == 0, 'Check for very small␣
↪values'
assert (df.Price > df.Price.quantile(0.999)).sum() == 0, 'Check for very large␣
↪values'
```

Question 3 After you've done the cleaning above, remove any column that has more than `500` missing values.

Ensure you make the changes to the DataFrame assigned to the variable `df`.

[24]:
```
# Your code here
df=df.drop(columns='Size')
# your code here
```

[25]:
```
# Question 3 Grading Checks

assert len(df.columns) < 10, 'You should have dropped 1 or more columns (with␣
 ↪more than 500 missing values)'
```

Question 4 Address the other missing values. You can replace the values or remvove them, but whatever method you decide to clean the DataFrame, you should no longer have any missing values.

Ensure you make the changes to the DataFrame assigned to the variable `df`.

[32]:
```
# Your code here
df=df.dropna(subset=['Customer_ID'])
df.info()

# your code here
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2714 entries, 0 to 2902
Data columns (total 9 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Date              2714 non-null   object
 1   Order_Number      2714 non-null   object
 2   Customer_ID       2714 non-null   object
 3   Product_Name      2714 non-null   object
 4   SKU               2714 non-null   object
 5   Price             2714 non-null   float64
 6   Quantity          2714 non-null   int64
 7   Product_Category  2714 non-null   object
 8   Product_Line      2714 non-null   object
dtypes: float64(1), int64(1), object(7)
memory usage: 212.0+ KB
```

[33]:
```
# Question 4 Grading Checks

assert df.Customer_ID.isna().sum() == 0, 'Did you address all the missing␣
 ↪values?'
```

1.3 Explore

You will explore the data. It's important that you follow the directions as stated. Doing more or less than what is asked might lead to not getting full points for the question.

You may use either exploratory statistics or exploratory visualizations to help answer these questions.

Note that the DataFrame loaded for this section (in the below cell) is different from the data you used in the Section 1.2 section.

If while you're working on the scrubbing phase you need to reset the DataFrame, you can restart the kernel (in the toolbar: "Kernel" > "Restart").

[34]:
```
df = pd.read_csv('transactions-pet_store-clean.csv')
```

Question 5 Create a `Subtotal` column by multiplying the `Price` and `Quantity` values. This represents how much was spent for a given transaction (row).

[36]:
```
# Your code here
df['Subtotal']= df.Price * df.Quantity
df
# your code here
```

[36]:

	Date	Order_Number	Customer_ID	\
0	5/22/2021	SXF-7309-1727-1334	476582ea-1bba-4289-8775-3fcd8074821c	
1	5/22/2021	SXF-7309-1727-1334	476582ea-1bba-4289-8775-3fcd8074821c	
2	5/22/2021	SXF-7309-1727-1334	476582ea-1bba-4289-8775-3fcd8074821c	
3	3/23/2020	DG7-5410-5845-1340	5929a0e9-95a7-4dbf-896e-c11d1988615e	
4	3/23/2020	DG7-5410-5845-1340	5929a0e9-95a7-4dbf-896e-c11d1988615e	
…	…	…	…	
2709	10/16/2020	P8K-8079-0264-6569	468f40b3-59ca-47fd-9739-c7f19cf48f32	
2710	10/16/2020	P8K-8079-0264-6569	468f40b3-59ca-47fd-9739-c7f19cf48f32	
2711	10/16/2020	P8K-8079-0264-6569	468f40b3-59ca-47fd-9739-c7f19cf48f32	
2712	10/16/2020	P8K-8079-0264-6569	468f40b3-59ca-47fd-9739-c7f19cf48f32	
2713	12/10/2019	6ZD-7972-0320-6653	f2a090b3-ec77-4018-939e-1a18d2b4f4ef	

	Product_Name	SKU	Price	Quantity	Product_Category	\
0	Feline Fix Mix	RKAPY3I1TP	39.55	1	treat	
1	Scratchy Post	MPH6SCD7UT	26.95	3	toy	
2	Reddy Beddy	DJWE1V9LZK	23.07	3	bedding	
3	Snoozer Essentails	GABWVMEL2R	28.04	3	bedding	
4	Reddy Beddy	KDTMPSBZKZ	13.84	1	bedding	
…	…	…	…	…	…	
2709	Cat Cave	058G0P7V60	29.66	1	bedding	
2710	Kitty Climber	W86BRJ9SSG	39.32	1	toy	
2711	Fetch Blaster	M291KHJ4LW	29.47	1	toy	
2712	Snoozer Essentails	GABWVMEL2R	28.04	1	bedding	
2713	Snoozer Essentails	GABWVMEL2R	28.04	1	bedding	

	Product_Line	Subtotal
0	cat	39.55
1	cat	80.85
2	dog	69.21
3	dog	84.12

```
4              dog     13.84
…              …       …
2709           cat     29.66
2710           cat     39.32
2711           dog     29.47
2712           dog     28.04
2713           dog     28.04

[2714 rows x 10 columns]
```

[37]:
```
# Question 5 Grading Checks

assert 'Subtotal' in df.columns, ''
```

Question 6 Determine most common category (`Product_Category`) purchases (number of total items) for both `Product_Line` categories. Assign the (string) name of these categories to their respective variables `common_category_cat` & `common_category_dog`.

[88]:
```
# Your code here
df_dog = df[df['Product_Line'] == 'dog']
df_dog.groupby('Product_Category').count()
df_cat = df[df['Product_Line'] == 'cat']
df_cat.groupby('Product_Category').count()
common_category_cat = 'treat'
common_category_dog = 'bedding'

# your code here
```

[56]:
```
# Question 6 Grading Checks

assert isinstance(common_category_dog, str), 'Ensure you assign the name of the␣
↪category (string) to the variable common_category_dog'
assert isinstance(common_category_cat, str), 'Ensure you assign the name of the␣
↪category (string) to the variable common_category_cat'
```

Question 7 Determine which categories (`Product_Category`), by `Product_Line` have the ***median*** highest `Price`. Assign the (string) name of these categories to their respective variables `priciest_category_cat` & `priciest_category_dog`.

[60]:
```
# Your code here
df_cat.groupby('Product_Category')['Price'].median()
priciest_category_dog= 'toy'
priciest_category_cat = 'bedding'
# your code here
```

[61]:
```
# Question 7 Grading Checks

assert isinstance(priciest_category_dog, str), 'Ensure you assign the name of␣
 ↪the category (string) to the variable priciest_category_dog'
assert isinstance(priciest_category_cat, str), 'Ensure you assign the name of␣
 ↪the category (string) to the variable priciest_category_cat'
```

1.4 Modeling

This is the point of the framework where we'd work on modeling with our data. However, in this activity, we're going to move straight to interpretting.

1.5 Interpret

You will interpret the data based on what you found so far. It's important that you follow the directions as stated. Doing more or less than what is asked might lead to not getting full points for the question.

Note that the DataFrame loaded for this section (in the below cell) is the same as the data you used in the Section 1.3 section.

If while you're working on the scrubbing phase you need to reset the DataFrame, you can restart the kernel (in the toolbar: "Kernel" > "Restart").

Question 8 You want to emphasize to your stakeholders that the total number of product categories sold differ between the two `Product_Line` categories (`'cat'` & `'dog'`).

Create a ***horizontal* bar plot** that has `Product_Category` on the y-axis and the total number of that category sold (using the `Quantity`) by each `Product_Line` category. Also **change the axis labels** to something meaningful and add a title.

You will likely want to use Seaborn. Make sure you set the result to the variable `ax` like the following:

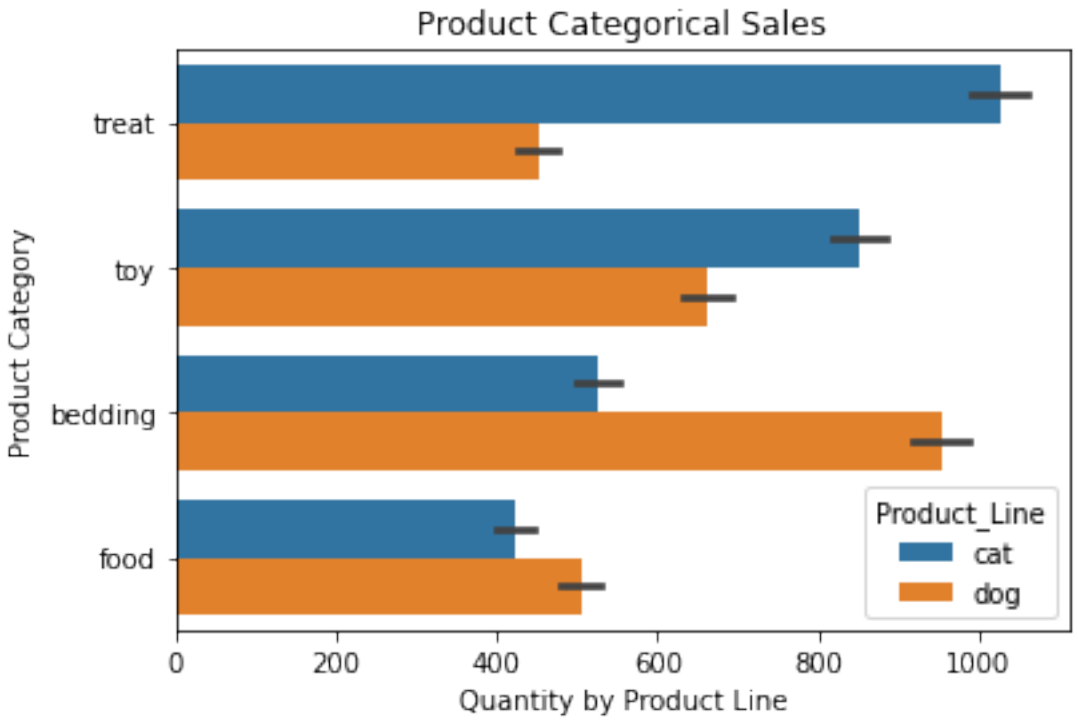
`ax = # code to create a bar plot`

[86]:
```
# Your code here
ax=sns.barplot(x=df['Quantity'], y=df['Product_Category'], estimator=sum, hue=␣
 ↪df['Product_Line'])

# Set custom axis labels and title
plt.xlabel('Quantity by Product Line')
plt.ylabel('Product Category')
plt.title('Product Categorical Sales')

# your code here
```

[86]:
```
Text(0.5, 1.0, 'Product Categorical Sales')
```



[87]:
```
# Question 8 Grading Checks

assert isinstance(ax, plt.Axes), 'Did you assign the plot result to the␣
↪variable ax?'
```

Question 9 Based on the plot from Section 1.5, what would you conclude for your stakeholders about what products they should sell? What would be the considerations and/or caveats you'd communicate to your stakeholders?

Write at least a couple sentences of your thoughts in a string assigned to the variable `answer_to_9`.

The cell below should look something like this:

```
answer_to_9 = '''
I think that based on the visualization that ****.
Therefore I would communicate with the stakeholders that ****
'''
```

[89]:
```
# Your code here
answer_to_9 = '''
I think based on the visualisation the company has strong market for treats and␣
↪bedding and should really emphasis on those categories to improve sales.
```

```
There is a strong potential in the treats market for dogs as well as in bedding␣
 ↪market for cats.
'''
print(len(answer_to_9))
# your code here
```

252

[90]:
```
# Question 9 Grading Checks

assert isinstance(answer_to_9, str), 'Make sure you create a string for your␣
 ↪answer.'
```

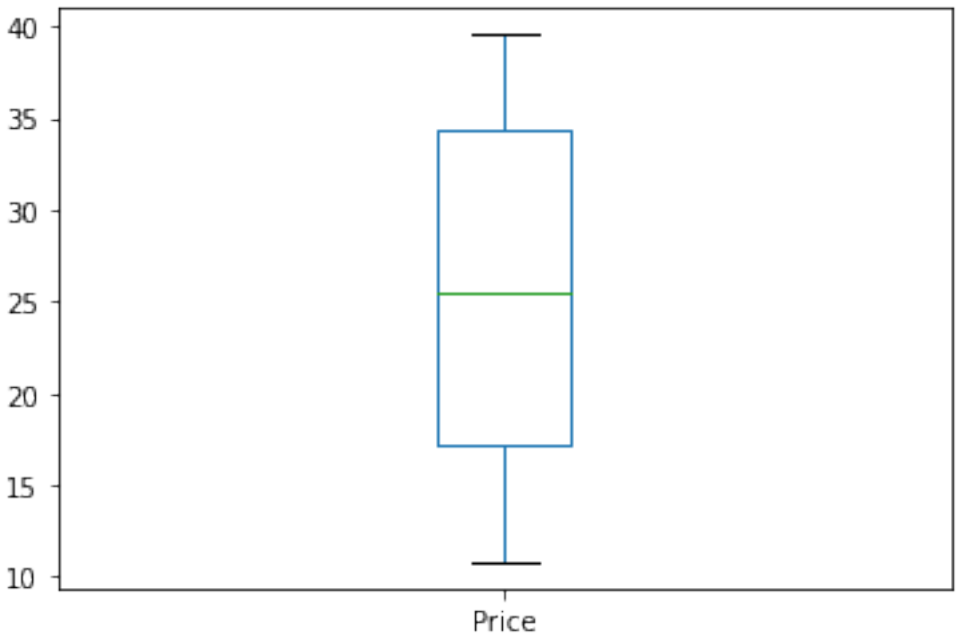
Question 10 The plot you created for Section 1.5 is good but could be modified to emphasize which products are important for the business.

Create an explanatory visualization that emphasizes the insight you about the product category. This would be a visualization you'd share with the business stakeholders.

Make sure you set the result to the variable `ax` like the following:

```
ax = # code to create explanatory visualization
```

[94]:
```
# Your code here
ax= df['Price'].plot.box()
# your code here
```



[95]:
```
# Question 10 Grading Checks

assert isinstance(ax, plt.Axes), 'Did you assign the plot result to the␣
↪variable ax?'
```