Sales_data_analysis

May 16, 2024

[1]:
```
#Importing the required libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

[2]:
```
pd.options.display.max_rows = 500
pd.options.display.max_columns = 500
```

[3]:
```
#Reading the data file
products = pd.read_csv('/content/drive/MyDrive/Computers/Python/Datasets for␣
 ↪data analysis/CRM + Sales data analysis/products.csv')
```

[4]:
```
#Checking the data in the file
print(products.head(8))
```

```
          product series  sales_price
0       GTX Basic    GTX          550
1         GTX Pro    GTX         4821
2      MG Special     MG           55
3     MG Advanced     MG         3393
4    GTX Plus Pro    GTX         5482
5  GTX Plus Basic    GTX         1096
6         GTK 500    GTK        26768
```

[5]:
```
#Getting the decription of the dataset
print(products.describe())
```

```
        sales_price
count      7.000000
mean    6023.571429
std     9388.428070
min       55.000000
25%      823.000000
50%     3393.000000
75%     5151.500000
max    26768.000000
```

[6]:
```
#Checking for duplicates
print(products.duplicated().sum())
#There are no duplicates
```

```
0
```

[7]:
```
#Getting the count of values
print(len(products))
```

```
7
```

[8]:
```
#Getting the number of rows
products.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7 entries, 0 to 6
Data columns (total 3 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   product      7 non-null      object
 1   series       7 non-null      object
 2   sales_price  7 non-null      int64
dtypes: int64(1), object(2)
memory usage: 296.0+ bytes
```

[9]:
```
#Getting the number of null values in the data
products.isnull().sum()
#There are no empty values in the dataset.
```

[9]:
```
product        0
series         0
sales_price    0
dtype: int64
```

[10]:
```
#Getting the number of rows and columns
print(products.shape)
#There are 7 rows and 3 columns in the dataset
```

```
(7, 3)
```

[11]:
```
#Importing another dataset
accounts = pd.read_csv('/content/drive/MyDrive/Computers/Python/Datasets for␣
 ↪data analysis/CRM + Sales data analysis/accounts.csv')
```

[12]:
```
#Getting the head
accounts.head()
```

[12]:
```
             account     sector  year_established  revenue  employees  \
0  Acme Corporation  technolgy              1996  1100.04       2822
1        Betasoloin    medical              1999   251.41        495
2          Betatech    medical              1986   647.18       1185
3        Bioholding    medical              2012   587.34       1356
4           Bioplex    medical              1991   326.82       1016

  office_location subsidiary_of
0   United States           NaN
1   United States           NaN
2           Kenya           NaN
3      Philipines           NaN
4   United States           NaN
```

[13]:
```
#Getting the info about the data
accounts.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 85 entries, 0 to 84
Data columns (total 7 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   account           85 non-null     object
 1   sector            85 non-null     object
 2   year_established  85 non-null     int64
 3   revenue           85 non-null     float64
 4   employees         85 non-null     int64
 5   office_location   85 non-null     object
 6   subsidiary_of     15 non-null     object
dtypes: float64(1), int64(2), object(4)
memory usage: 4.8+ KB
```

[14]:
```
#Checking if there are duplicates
print(accounts.duplicated().sum())
#There are no duplicates in the dataset
```

```
0
```

[15]:
```
#Getting the number of rows and columns in the dataset
print(accounts.shape)
#There are 85 rows and 6 columns in the dataset
```

```
(85, 7)
```

[16]:
```
#Getting the description of the data
accounts.describe()
```

[16]:

	year_established	revenue	employees
count	85.000000	85.000000	85.000000
mean	1996.105882	1994.632941	4660.823529
std	8.865427	2169.491436	5715.601198
min	1979.000000	4.540000	9.000000
25%	1989.000000	497.110000	1179.000000
50%	1996.000000	1223.720000	2769.000000
75%	2002.000000	2741.370000	5595.000000
max	2017.000000	11698.030000	34288.000000

[17]:
```
#Getting the number of null values in the data
accounts.isnull().sum()
#We can see that there are null values
```

[17]:
```
account              0
sector               0
year_established     0
revenue              0
employees            0
office_location      0
subsidiary_of       70
dtype: int64
```

[18]:
```
#Getting the percentage of null values in the dataset
print((accounts.isnull().sum()/len(accounts))*100)
#We can see that 82% of the data in subsidiary of is null values.
```

```
account              0.000000
sector               0.000000
year_established     0.000000
revenue              0.000000
employees            0.000000
office_location      0.000000
subsidiary_of       82.352941
dtype: float64
```

[19]:
```
#Removing the column with more errors
accounts.drop(columns = ['subsidiary_of'],inplace = True)
```

[20]:
```
#Getting the head of the data now
accounts.head()
```

[20]:

	account	sector	year_established	revenue	employees	\
0	Acme Corporation	technolgy	1996	1100.04	2822	
1	Betasoloin	medical	1999	251.41	495	
2	Betatech	medical	1986	647.18	1185	
3	Bioholding	medical	2012	587.34	1356	

```
4          Bioplex    medical              1991   326.82       1016

  office_location
0   United States
1   United States
2           Kenya
3      Philipines
4   United States
```

[21]:
```
#Reading a new dataset
sales_pipeline = pd.read_csv('/content/drive/MyDrive/Computers/Python/Datasets␣
↪for data analysis/CRM + Sales data analysis/sales_pipeline.csv')
```

[22]:
```
#Checking the dataset for duplicates
print(sales_pipeline.duplicated().sum())
#There are no duplicates in the dataset
```

```
0
```

[23]:
```
#Getting the number of rows and columns in the dataset
print(sales_pipeline.shape)
#There are 8800 rows and 8 columns
```

```
(8800, 8)
```

[24]:
```
#Checking the head of the dataset
sales_pipeline.head()
```

[24]:
```
  opportunity_id      sales_agent         product  account deal_stage  \
0       1C1I7A6R      Moses Frase  GTX Plus Basic  Cancity        Won
1       Z063OYWO  Darcel Schlecht          GTXPro    Isdom        Won
2       EC4QE1BX  Darcel Schlecht      MG Special  Cancity        Won
3       MV1LWRNH      Moses Frase       GTX Basic  Codehow        Won
4       PE84CX4O        Zane Levy       GTX Basic   Hatfan        Won

  engage_date  close_date  close_value
0  2016-10-20  2017-03-01       1054.0
1  2016-10-25  2017-03-11       4514.0
2  2016-10-25  2017-03-07         50.0
3  2016-10-25  2017-03-09        588.0
4  2016-10-25  2017-03-02        517.0
```

[25]:
```
#Getting the info of the data
sales_pipeline.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8800 entries, 0 to 8799
Data columns (total 8 columns):
```

```
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   opportunity_id  8800 non-null   object
 1   sales_agent     8800 non-null   object
 2   product         8800 non-null   object
 3   account         7375 non-null   object
 4   deal_stage      8800 non-null   object
 5   engage_date     8300 non-null   object
 6   close_date      6711 non-null   object
 7   close_value     6711 non-null   float64
dtypes: float64(1), object(7)
memory usage: 550.1+ KB
```

[26]:
```
#Getting the data description
sales_pipeline.describe()
```

[26]:
```
        close_value
count   6711.000000
mean    1490.915512
std     2320.670773
min        0.000000
25%        0.000000
50%      472.000000
75%     3225.000000
max    30288.000000
```

[27]:
```
#Getting the number of null values
sales_pipeline.isnull().sum()
#We can see that there are null values in the dataset
```

[27]:
```
opportunity_id       0
sales_agent          0
product              0
account           1425
deal_stage           0
engage_date        500
close_date        2089
close_value       2089
dtype: int64
```

[28]:
```
#Getting the percentage of null values
print((sales_pipeline.isnull().sum()/len(sales_pipeline))*100)
#The null values are less than 25%. So we need to replace them.
```

```
opportunity_id     0.000000
sales_agent        0.000000
product            0.000000
account           16.193182
```

```
deal_stage          0.000000
engage_date         5.681818
close_date         23.738636
close_value        23.738636
dtype: float64
```

[29]:
```
#Looking the columns
sales_pipeline['close_value'].head(10)

#Replacing the null values with 0
sales_pipeline['close_value'].fillna(0,inplace= True)

#Getting the data
sales_pipeline.head(10)
```

[29]:

	opportunity_id	sales_agent	product	account	deal_stage
0	1C1I7A6R	Moses Frase	GTX Plus Basic	Cancity	Won
1	Z063OYW0	Darcel Schlecht	GTXPro	Isdom	Won
2	EC4QE1BX	Darcel Schlecht	MG Special	Cancity	Won
3	MV1LWRNH	Moses Frase	GTX Basic	Codehow	Won
4	PE84CX4O	Zane Levy	GTX Basic	Hatfan	Won
5	ZNBS69V1	Anna Snelling	MG Special	Ron-tech	Won
6	9ME3374G	Vicki Laflamme	MG Special	J-Texon	Won
7	7GN8Q4LL	Markita Hansen	GTX Basic	Cheers	Won
8	OLK9LKZB	Niesha Huffines	GTX Plus Basic	Zumgoity	Won
9	HAXMC4IX	James Ascencio	MG Advanced	NaN	Engaging

	engage_date	close_date	close_value
0	2016-10-20	2017-03-01	1054.0
1	2016-10-25	2017-03-11	4514.0
2	2016-10-25	2017-03-07	50.0
3	2016-10-25	2017-03-09	588.0
4	2016-10-25	2017-03-02	517.0
5	2016-10-29	2017-03-01	49.0
6	2016-10-30	2017-03-02	57.0
7	2016-11-01	2017-03-07	601.0
8	2016-11-01	2017-03-03	1026.0
9	2016-11-03	NaN	0.0

[30]:
```
#Checking the close date
sales_pipeline['close_date'].head()

#Replacing missing close date values with 3 days after engage date
sales_pipeline['engage_date'] = pd.to_datetime(sales_pipeline['engage_date'])
sales_pipeline['close_date'] = sales_pipeline.apply(lambda x: x['engage_date']␣
 ↪+ pd.DateOffset(days=3) if pd.isnull(x['close_date']) else x['close_date'],␣
 ↪axis=1)
```

[31]:
```
#Checking the engage date
sales_pipeline['engage_date'].head()

#Replacing missing values with 3 days before close date
sales_pipeline['engage_date'] = sales_pipeline.apply(lambda y:y['close_date'] -␣
 ↪pd.DateOffset(days=3) if pd.isnull(y['engage_date']) else␣
 ↪y['engage_date'],axis = 1)
```

[32]:
```
#Recalculating the null value percentage
print((sales_pipeline.isnull().sum()/len(sales_pipeline))*100)
#We have fixed all the null values in the data
```

```
opportunity_id     0.000000
sales_agent        0.000000
product            0.000000
account           16.193182
deal_stage         0.000000
engage_date        5.681818
close_date         5.681818
close_value        0.000000
dtype: float64
```

[33]:
```
#Getting the head of the data
sales_pipeline.head()
```

[33]:

	opportunity_id	sales_agent	product	account	deal_stage	engage_date	close_date	close_value
0	1C1I7A6R	Moses Frase	GTX Plus Basic	Cancity	Won	2016-10-20	2017-03-01	1054.0
1	Z063OYW0	Darcel Schlecht	GTXPro	Isdom	Won	2016-10-25	2017-03-11	4514.0
2	EC4QE1BX	Darcel Schlecht	MG Special	Cancity	Won	2016-10-25	2017-03-07	50.0
3	MV1LWRNH	Moses Frase	GTX Basic	Codehow	Won	2016-10-25	2017-03-09	588.0
4	PE84CX4O	Zane Levy	GTX Basic	Hatfan	Won	2016-10-25	2017-03-02	517.0

[34]:
```
#Getting the tail of the data
sales_pipeline.tail()
```

[34]:

	opportunity_id	sales_agent	product	account	deal_stage	engage_date	close_date	close_value
8795	9MIWFW5J	Versie Hillebrand	MG Advanced	NaN	Prospecting	2016-10-25	2017-03-07	50.0
8796	6SLKZ8FI	Versie Hillebrand	MG Advanced	NaN	Prospecting	2016-10-25	2017-03-09	588.0
8797	LIB4KUZJ	Versie Hillebrand	MG Advanced	NaN	Prospecting	2016-10-25	2017-03-02	517.0
8798	18IUIUK0	Versie Hillebrand	MG Advanced	NaN	Prospecting	2016-10-25	2017-03-07	50.0

```
8799       8I5ONXJX  Versie Hillebrand  MG Advanced     NaN  Prospecting

     engage_date close_date  close_value
8795         NaT        NaT          0.0
8796         NaT        NaT          0.0
8797         NaT        NaT          0.0
8798         NaT        NaT          0.0
8799         NaT        NaT          0.0
```

[35]:
```
#Importing another dataset
sales_team = pd.read_csv('/content/drive/MyDrive/Computers/Python/Datasets for data analysis/CRM + Sales data analysis/sales_teams.csv')
```

[36]:
```
#Checking if there are any duplicates in the dataset
print(sales_team.duplicated().sum())
#There are no duplicates in the dataset
```

```
0
```

[37]:
```
#Checking the number of rows and columns
print(sales_team.shape)
#There are 35 rows and 3 columns in the sales_team
```

```
(35, 3)
```

[38]:
```
#Checking the data
print(sales_team.head())
```

```
         sales_agent           manager regional_office
0      Anna Snelling  Dustin Brinkmann         Central
1     Cecily Lampkin  Dustin Brinkmann         Central
2  Versie Hillebrand  Dustin Brinkmann         Central
3    Lajuana Vencill  Dustin Brinkmann         Central
4        Moses Frase  Dustin Brinkmann         Central
```

[39]:
```
#Getting info about the data
print(sales_team.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 35 entries, 0 to 34
Data columns (total 3 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   sales_agent      35 non-null     object
 1   manager          35 non-null     object
 2   regional_office  35 non-null     object
dtypes: object(3)
```

```
memory usage: 968.0+ bytes
None
```

[40]:
```
#Getting the number of rows in the dataset
print(len(sales_team))
```

```
35
```

[41]:
```
#Getting not nulls
print(sales_team.isnull().sum())
#There are no null values in the sheet
```

```
sales_agent        0
manager            0
regional_office    0
dtype: int64
```

1 Analysis

[42]:
```
#Get top 5 sales person by sales
sales_plus_pipeline = pd.merge(sales_team,sales_pipeline, on = 'sales_agent',␣
 ↪how = 'outer')
#Grouping the data using the sales agent and closing value and then getting the␣
 ↪sum
grouped_sales_data = sales_plus_pipeline.groupby('sales_agent')['close_value'].
 ↪sum().sort_values(ascending = False).head()
#printing the values
print(grouped_sales_data)
```

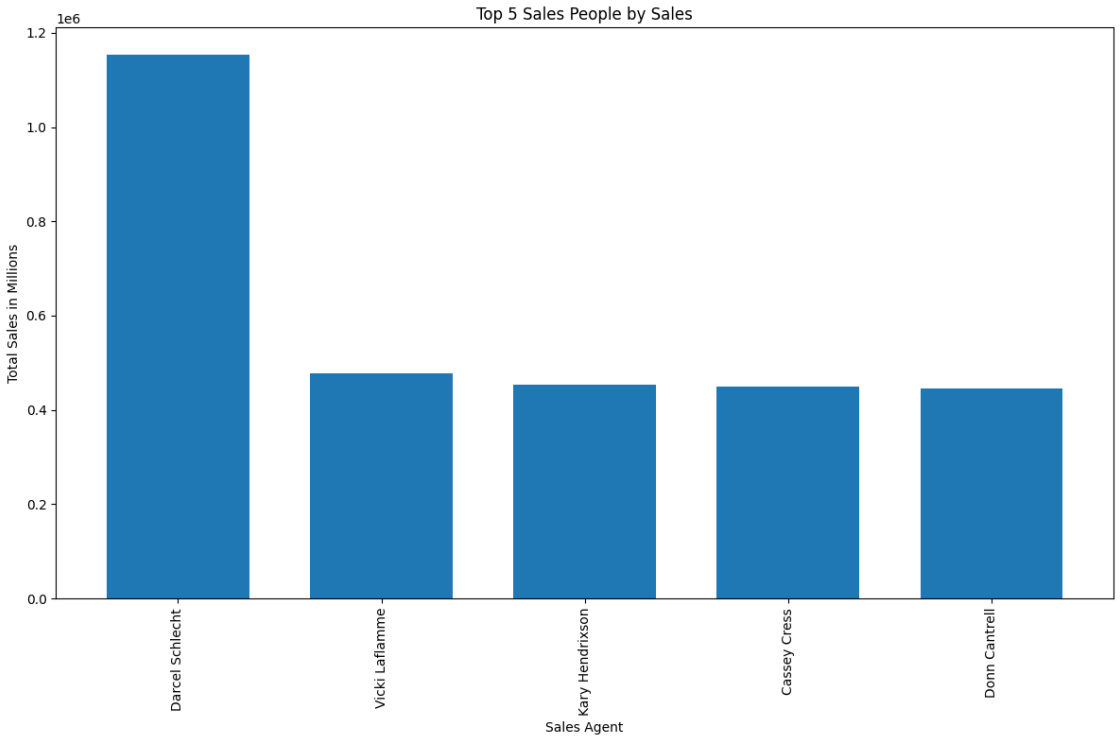
```
sales_agent
Darcel Schlecht    1153214.0
Vicki Laflamme      478396.0
Kary Hendrixson     454298.0
Cassey Cress        450489.0
Donn Cantrell       445860.0
Name: close_value, dtype: float64
```

[43]:
```
# Create a bar chart directly from the grouped data
grouped_sales_data.plot(kind='bar',figsize=(12,8),width = 0.7)

# Add labels and title
plt.xlabel("Sales Agent")
plt.ylabel("Total Sales in Millions")
plt.title("Top 5 Sales People by Sales")

# Improve layout
plt.tight_layout()
```

```
# Show the plot
plt.show()
```



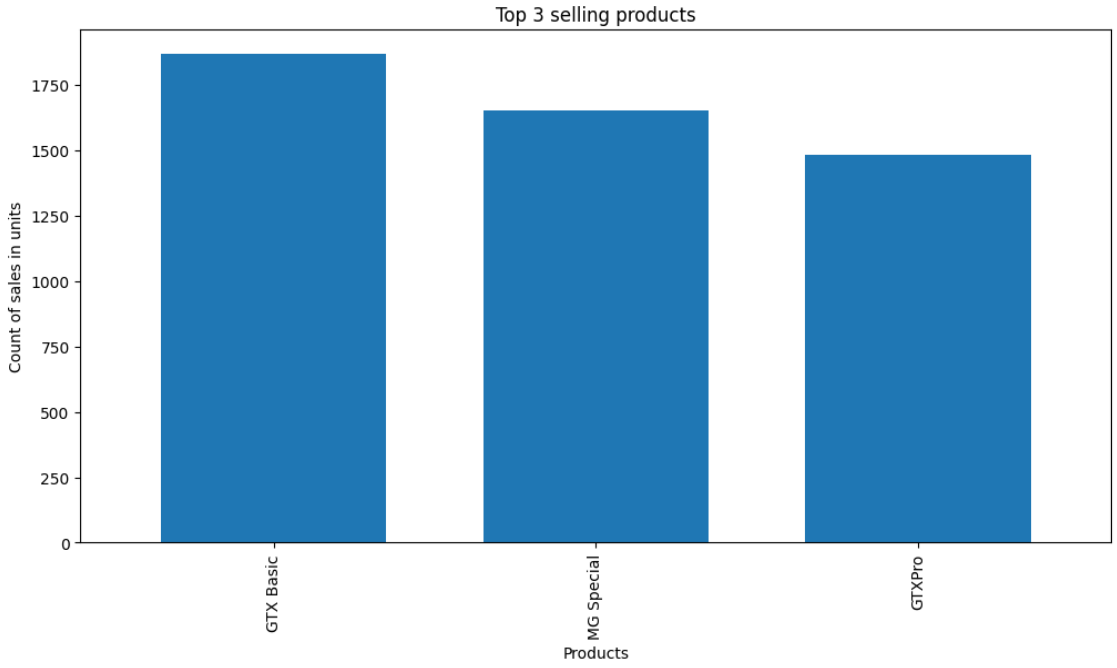
[44]:
```
#Top 3 selling prodcuts
top_3_products = sales_plus_pipeline['product'].value_counts().
 ↪sort_values(ascending = False).head(3)
#printing the top 3 products
print(top_3_products)
```

```
product
GTX Basic     1866
MG Special    1651
GTXPro        1480
Name: count, dtype: int64
```

[45]:
```
#Plotting the data for top 3 selling products
top_3_products.plot(kind='bar',figsize=(12,6),width = 0.7)

#Adding labels
plt.xlabel('Products')
plt.ylabel('Count of sales in units')
plt.title('Top 3 selling products')
```

```
#Show the graph
plt.show()
```



[46]:
```
#Top 5 sales persons by the number of products sold.
grouped_salesof_products = sales_plus_pipeline.
  ↪groupby('sales_agent')['product'].count().sort_values(ascending = False).
  ↪head()
print(grouped_salesof_products)
```

```
sales_agent
Darcel Schlecht    747
Vicki Laflamme     451
Anna Snelling      448
Kary Hendrixson    438
Kami Bicknell      362
Name: product, dtype: int64
```

[47]:
```
#Top 5 sales person and count of thier top selling productss
group_sales_products = sales_plus_pipeline.groupby(['sales_agent','product']).
  ↪size().sort_values(ascending = False).head()
print(group_sales_products)
```

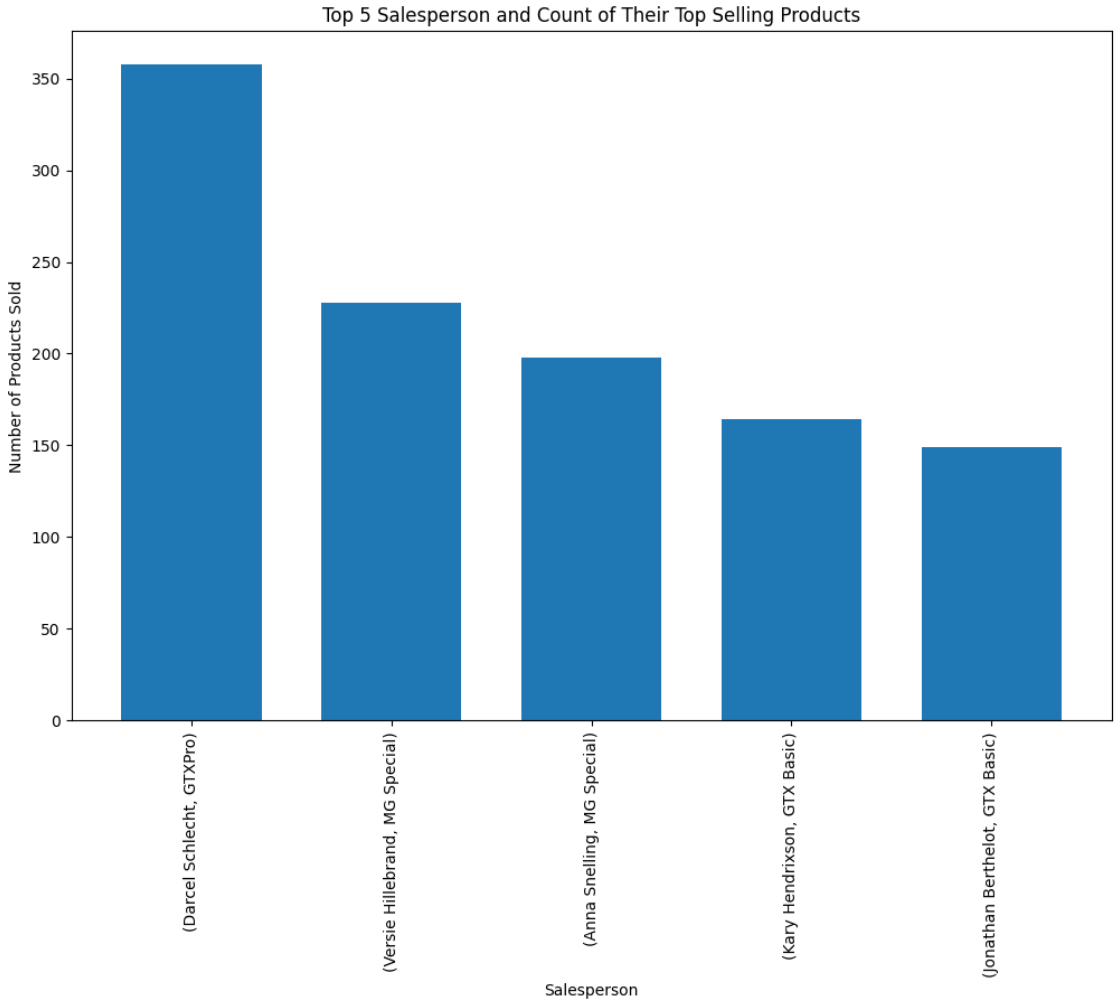
```
sales_agent         product
Darcel Schlecht     GTXPro        358
Versie Hillebrand   MG Special    228
```

```
Anna Snelling       MG Special    198
Kary Hendrixson     GTX Basic     164
Jonathan Berthelot  GTX Basic     149
dtype: int64
```

[48]:
```
# Plot the bar chart.
group_sales_products.plot(kind='bar', figsize=(12, 8), width=0.7)

# Add title and labels.
plt.title('Top 5 Salesperson and Count of Their Top Selling Products')
plt.xlabel('Salesperson')
plt.ylabel('Number of Products Sold')

# Show the plot.
plt.show()
```



[49]:
```
#Total deals closed, lost and enganging per sales agent.
sales_plus_pipeline.columns
deal_stage_per_salesper = sales_plus_pipeline.
  ↪groupby(['sales_agent','deal_stage']).size()
print(deal_stage_per_salesper)
```

```
sales_agent         deal_stage
Anna Snelling       Engaging        57
                    Lost           128
                    Prospecting     55
                    Won            208
Boris Faz           Engaging        57
                    Lost            52
                    Won            101
Cassey Cress        Engaging        85
                    Lost            98
                    Won            163
Cecily Lampkin      Engaging        19
                    Lost            53
                    Prospecting     24
                    Won            107
Corliss Cosme       Engaging        81
                    Lost            79
                    Won            150
Daniell Hammack     Engaging        72
                    Lost            73
                    Won            114
Darcel Schlecht     Engaging        83
                    Lost           204
                    Prospecting    111
                    Won            349
Donn Cantrell       Lost           117
                    Won            158
Elease Gluck        Engaging        51
                    Lost            46
                    Won             80
Garret Kinder       Lost            48
                    Won             75
Gladys Colclough    Engaging        36
                    Lost            97
                    Prospecting     49
                    Won            135
Hayden Neloms       Engaging        50
                    Lost            45
                    Won            107
James Ascencio      Engaging        61
                    Lost            71
```

	Won	135
Jonathan Berthelot	Engaging	33
	Lost	93
	Prospecting	48
	Won	171
Kami Bicknell	Engaging	90
	Lost	98
	Won	174
Kary Hendrixson	Engaging	103
	Lost	126
	Won	209
Lajuana Vencill	Engaging	40
	Lost	104
	Prospecting	40
	Won	127
Markita Hansen	Engaging	79
	Lost	97
	Won	130
Marty Freudenburg	Engaging	33
	Lost	72
	Prospecting	54
	Won	122
Maureen Marcano	Engaging	72
	Lost	64
	Won	149
Moses Frase	Engaging	34
	Lost	66
	Prospecting	31
	Won	129
Niesha Huffines	Engaging	30
	Lost	70
	Prospecting	34
	Won	105
Reed Clapper	Lost	82
	Won	155
Rosalina Dieter	Engaging	50
	Lost	38
	Won	72
Rosie Papadopoulos	Engaging	39
	Lost	43
	Won	78
Versie Hillebrand	Engaging	43
	Lost	88
	Prospecting	54
	Won	176
Vicki Laflamme	Engaging	104
	Lost	126
	Won	221

```
Violet Mclelland    Engaging        68
                    Lost            71
                    Won            122
Wilburn Farren      Engaging        31
                    Lost            24
                    Won             55
Zane Levy           Engaging        88
                    Lost           100
                    Won            161
dtype: int64
```

[50]:
```
#Won vs Lost percentage per sales person
filtered_won_lost = sales_plus_pipeline[sales_plus_pipeline['deal_stage'].
 ↪isin(['Won','Lost'])]

won_count = sales_plus_pipeline[sales_plus_pipeline['deal_stage'] == 'Won'].
 ↪groupby('sales_agent')['deal_stage'].value_counts()
lost_count = sales_plus_pipeline[sales_plus_pipeline['deal_stage'] == 'Lost'].
 ↪groupby('sales_agent')['deal_stage'].value_counts()

total_deals = filtered_won_lost.groupby('sales_agent')['deal_stage'].count()

#calculating won and loss percentage
won_percent = (won_count/total_deals)*100
print(won_percent)
```

```
sales_agent         deal_stage
Anna Snelling       Won           61.904762
Boris Faz           Won           66.013072
Cassey Cress        Won           62.452107
Cecily Lampkin      Won           66.875000
Corliss Cosme       Won           65.502183
Daniell Hammack     Won           60.962567
Darcel Schlecht     Won           63.110307
Donn Cantrell       Won           57.454545
Elease Gluck        Won           63.492063
Garret Kinder       Won           60.975610
Gladys Colclough    Won           58.189655
Hayden Neloms       Won           70.394737
James Ascencio      Won           65.533981
Jonathan Berthelot  Won           64.772727
Kami Bicknell       Won           63.970588
Kary Hendrixson     Won           62.388060
Lajuana Vencill     Won           54.978355
Markita Hansen      Won           57.268722
Marty Freudenburg   Won           62.886598
Maureen Marcano     Won           69.953052
```

```
Moses Frase          Won          66.153846
Niesha Huffines      Won          60.000000
Reed Clapper         Won          65.400844
Rosalina Dieter      Won          65.454545
Rosie Papadopoulos   Won          64.462810
Versie Hillebrand    Won          66.666667
Vicki Laflamme       Won          63.688761
Violet Mclelland     Won          63.212435
Wilburn Farren       Won          69.620253
Zane Levy            Won          61.685824
dtype: float64
```

[51]:
```
#loss percentage
lost_percent = (lost_count/total_deals)*100
print(lost_percent)
```

```
sales_agent          deal_stage
Anna Snelling        Lost         38.095238
Boris Faz            Lost         33.986928
Cassey Cress         Lost         37.547893
Cecily Lampkin       Lost         33.125000
Corliss Cosme        Lost         34.497817
Daniell Hammack      Lost         39.037433
Darcel Schlecht      Lost         36.889693
Donn Cantrell        Lost         42.545455
Elease Gluck         Lost         36.507937
Garret Kinder        Lost         39.024390
Gladys Colclough     Lost         41.810345
Hayden Neloms        Lost         29.605263
James Ascencio       Lost         34.466019
Jonathan Berthelot   Lost         35.227273
Kami Bicknell        Lost         36.029412
Kary Hendrixson      Lost         37.611940
Lajuana Vencill      Lost         45.021645
Markita Hansen       Lost         42.731278
Marty Freudenburg    Lost         37.113402
Maureen Marcano      Lost         30.046948
Moses Frase          Lost         33.846154
Niesha Huffines      Lost         40.000000
Reed Clapper         Lost         34.599156
Rosalina Dieter      Lost         34.545455
Rosie Papadopoulos   Lost         35.537190
Versie Hillebrand    Lost         33.333333
Vicki Laflamme       Lost         36.311239
Violet Mclelland     Lost         36.787565
Wilburn Farren       Lost         30.379747
Zane Levy            Lost         38.314176
dtype: float64
```