navyas-credit-card

May 14, 2023

1 CREDIT CARD -CASE STUDY

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import datetime as dt
%matplotlib inline
import os
customer=pd.read_csv(r"C:\Users\lenovo\Downloads\Python Foundation Case Study 2␣
↪- Credit Card Case Study\Customer Acqusition.csv")

customer
```

[1]:

	No	Customer	Age	City	Product	Limit	Company	Segment
0	1	A1	76	BANGALORE	Gold	500000.0	C1	Self Employed
1	2	A2	71	CALCUTTA	Silver	100000.0	C2	Salaried_MNC
2	3	A3	34	COCHIN	Platimum	10000.0	C3	Salaried_Pvt
3	4	A4	47	BOMBAY	Platimum	10001.0	C4	Govt
4	5	A5	56	BANGALORE	Platimum	10002.0	C5	Normal Salary
..	…	…	…	…	…	…	…	…
95	96	A96	54	CHENNAI	Silver	100000.0	C19	Salaried_Pvt
96	97	A97	58	TRIVANDRUM	Platimum	10000.0	C20	Govt
97	98	A98	51	CALCUTTA	Platimum	10001.0	C21	Normal Salary
98	99	A99	35	CALCUTTA	Platimum	10002.0	C22	Self Employed
99	100	A100	36	COCHIN	Silver	100000.0	C5	Salaried_MNC

[100 rows x 8 columns]

2 1.a. In case age is less than 18, replace it with mean of age values.

[2]:
```
customer.loc[customer["Age"] < 18,"Age"] = customer["Age"].mean()

customer
```

[2]:

	No	Customer	Age	City	Product	Limit	Company	Segment
0	1	A1	76.0	BANGALORE	Gold	500000.0	C1	Self Employed
1	2	A2	71.0	CALCUTTA	Silver	100000.0	C2	Salaried_MNC
2	3	A3	34.0	COCHIN	Platimum	10000.0	C3	Salaried_Pvt
3	4	A4	47.0	BOMBAY	Platimum	10001.0	C4	Govt
4	5	A5	56.0	BANGALORE	Platimum	10002.0	C5	Normal Salary
..	…	…	…	…	…	…	…	
95	96	A96	54.0	CHENNAI	Silver	100000.0	C19	Salaried_Pvt
96	97	A97	58.0	TRIVANDRUM	Platimum	10000.0	C20	Govt
97	98	A98	51.0	CALCUTTA	Platimum	10001.0	C21	Normal Salary
98	99	A99	35.0	CALCUTTA	Platimum	10002.0	C22	Self Employed
99	100	A100	36.0	COCHIN	Silver	100000.0	C5	Salaried_MNC

[100 rows x 8 columns]

3 b. In case spend amount is more than the limit, replace it with 50% of that customer's limit.

(customer's limit provided in acquisition table is the per transaction limit on his card)

[5]:
```
spend=pd.read_csv(r"C:\Users\lenovo\Downloads\Python Foundation Case Study 2 -␣
↪Credit Card Case Study\spend.csv")
spend
```

[5]:

	Sl No:	Customer	Month	Type	Amount
0	1	A1	12-Jan-04	JEWELLERY	485470.80
1	2	A1	3-Jan-04	PETRO	410556.13
2	3	A1	15-Jan-04	CLOTHES	23740.46
3	4	A1	25-Jan-04	FOOD	484342.47
4	5	A1	17-Jan-05	CAMERA	369694.07
…	…	…	…	…	…
1495	1496	A67	4-Feb-06	BUS TICKET	356872.73
1496	1497	A68	25-Mar-06	BUS TICKET	204971.10
1497	1498	A69	31-Mar-06	BUS TICKET	50449.44
1498	1499	A70	23-Mar-06	BUS TICKET	80593.94
1499	1500	A71	24-Mar-06	BUS TICKET	194447.62

[1500 rows x 5 columns]

[6]:
```
customer_spend=pd.merge(left=customer,right=spend,on='Customer',how='inner')
customer_spend
```

[6]:

	No	Customer	Age	City	Product	Limit	Company	\
0	1	A1	76.0	BANGALORE	Gold	500000.0	C1	
1	1	A1	76.0	BANGALORE	Gold	500000.0	C1	
2	1	A1	76.0	BANGALORE	Gold	500000.0	C1	

```
3       1       A1  76.0   BANGALORE      Gold  500000.0      C1
4       1       A1  76.0   BANGALORE      Gold  500000.0      C1
…   …       …   …         …       …       …     …
1495   96      A96  54.0     CHENNAI    Silver  100000.0     C19
1496   97      A97  58.0  TRIVANDRUM  Platimum   10000.0     C20
1497   98      A98  51.0    CALCUTTA  Platimum   10001.0     C21
1498   99      A99  35.0    CALCUTTA  Platimum   10002.0     C22
1499  100     A100  36.0      COCHIN    Silver  100000.0      C5

             Segment  Sl No:      Month        Type     Amount
0     Self Employed       1  12-Jan-04   JEWELLERY  485470.80
1     Self Employed       2   3-Jan-04       PETRO  410556.13
2     Self Employed       3  15-Jan-04     CLOTHES   23740.46
3     Self Employed       4  25-Jan-04        FOOD  484342.47
4     Self Employed       5  17-Jan-05      CAMERA  369694.07
…              …     …         …          …         …
1495   Salaried_Pvt      98  25-Jan-04        BIKE   54729.66
1496           Govt      99  12-Jan-04        AUTO  139018.26
1497  Normal Salary     100   3-Jan-04    SHOPPING  284521.06
1498  Self Employed     101  15-Jan-04  AIR TICKET   90184.22
1499   Salaried_MNC     102  25-Jan-04  BUS TICKET  224786.88

[1500 rows x 12 columns]
```

[7]:
```
customer_spend.
 ↪loc[customer_spend['Amount']>customer_spend['Limit'],'Amount']=(50*customer_spend['Limit'])
 ↪div(100)
customer_spend
```

[7]:
```
        No Customer   Age        City   Product     Limit Company  \
0       1       A1  76.0   BANGALORE      Gold  500000.0      C1
1       1       A1  76.0   BANGALORE      Gold  500000.0      C1
2       1       A1  76.0   BANGALORE      Gold  500000.0      C1
3       1       A1  76.0   BANGALORE      Gold  500000.0      C1
4       1       A1  76.0   BANGALORE      Gold  500000.0      C1
…   …       …   …         …       …       …     …
1495   96      A96  54.0     CHENNAI    Silver  100000.0     C19
1496   97      A97  58.0  TRIVANDRUM  Platimum   10000.0     C20
1497   98      A98  51.0    CALCUTTA  Platimum   10001.0     C21
1498   99      A99  35.0    CALCUTTA  Platimum   10002.0     C22
1499  100     A100  36.0      COCHIN    Silver  100000.0      C5

             Segment  Sl No:      Month        Type     Amount
0     Self Employed       1  12-Jan-04   JEWELLERY  485470.80
1     Self Employed       2   3-Jan-04       PETRO  410556.13
2     Self Employed       3  15-Jan-04     CLOTHES   23740.46
3     Self Employed       4  25-Jan-04        FOOD  484342.47
```

```
4     Self Employed       5  17-Jan-05      CAMERA  369694.07
…               …     …         …          …          …
1495   Salaried_Pvt      98  25-Jan-04        BIKE   54729.66
1496           Govt      99  12-Jan-04        AUTO    5000.00
1497  Normal Salary     100   3-Jan-04    SHOPPING    5000.50
1498  Self Employed     101  15-Jan-04  AIR TICKET    5001.00
1499   Salaried_MNC     102  25-Jan-04  BUS TICKET   50000.00

[1500 rows x 12 columns]
```

4 c. Incase the repayment amount is more than the limit, replace the repayment with the limit.

[9]:
```
repayment=pd.read_csv(r"C:\Users\lenovo\Downloads\Python Foundation Case Study 2 - Credit Card Case Study\Repayment.csv")
repayment
```

[9]:
```
      SL No: Customer      Month     Amount  Unnamed: 4
0        NaN       A1  12-Jan-04  495414.75         NaN
1        2.0       A1   3-Jan-04  245899.02         NaN
2        3.0       A1  15-Jan-04  259490.06         NaN
3        4.0       A1  25-Jan-04  437555.12         NaN
4        5.0       A1  17-Jan-05  165972.88         NaN
…        …         …         …          …          …
1518     NaN      NaN        NaN        NaN        NaN
1519     NaN      NaN        NaN        NaN        NaN
1520     NaN      NaN        NaN        NaN        NaN
1521     NaN      NaN        NaN        NaN        NaN
1522     NaN      NaN        NaN        NaN        NaN

[1523 rows x 5 columns]
```

[10]:
```
customer_repay=pd.merge(left=customer,right=repayment,on='Customer',how='inner')
customer_repay
```

[10]:
```
      No Customer   Age        City   Product     Limit Company  \
0      1       A1  76.0   BANGALORE      Gold  500000.0      C1
1      1       A1  76.0   BANGALORE      Gold  500000.0      C1
2      1       A1  76.0   BANGALORE      Gold  500000.0      C1
3      1       A1  76.0   BANGALORE      Gold  500000.0      C1
4      1       A1  76.0   BANGALORE      Gold  500000.0      C1
…   …      …   …         …       …        …     …
1495  96      A96  54.0     CHENNAI    Silver  100000.0     C19
1496  97      A97  58.0  TRIVANDRUM  Platimum   10000.0     C20
1497  98      A98  51.0    CALCUTTA  Platimum   10001.0     C21
1498  99      A99  35.0    CALCUTTA  Platimum   10002.0     C22
```

```
1499  100     A100  36.0      COCHIN    Silver  100000.0      C5

             Segment  SL No:      Month     Amount  Unnamed: 4
0     Self Employed     NaN  12-Jan-04  495414.75         NaN
1     Self Employed     2.0   3-Jan-04  245899.02         NaN
2     Self Employed     3.0  15-Jan-04  259490.06         NaN
3     Self Employed     4.0  25-Jan-04  437555.12         NaN
4     Self Employed     5.0  17-Jan-05  165972.88         NaN
…               …     …         …         …          …
1495   Salaried_Pvt    98.0  25-Jan-04  310992.30         NaN
1496           Govt    99.0  12-Jan-04  121874.90         NaN
1497  Normal Salary   100.0   3-Jan-04  337815.57         NaN
1498  Self Employed   101.0  15-Jan-04   25682.73         NaN
1499   Salaried_MNC   102.0  25-Jan-04   69551.19         NaN

[1500 rows x 12 columns]
```

[11]:
```
customer_repay.
 ↪loc[customer_repay['Amount']>customer_repay['Limit'],'Amount']=customer_repay['Limit']
customer_repay
```

[11]:
```
        No Customer   Age        City   Product     Limit Company  \
0       1       A1  76.0   BANGALORE      Gold  500000.0      C1
1       1       A1  76.0   BANGALORE      Gold  500000.0      C1
2       1       A1  76.0   BANGALORE      Gold  500000.0      C1
3       1       A1  76.0   BANGALORE      Gold  500000.0      C1
4       1       A1  76.0   BANGALORE      Gold  500000.0      C1
…   …       …   …          …       …        …      …
1495   96      A96  54.0     CHENNAI    Silver  100000.0     C19
1496   97      A97  58.0  TRIVANDRUM  Platimum   10000.0     C20
1497   98      A98  51.0    CALCUTTA  Platimum   10001.0     C21
1498   99      A99  35.0    CALCUTTA  Platimum   10002.0     C22
1499  100     A100  36.0      COCHIN    Silver  100000.0      C5

             Segment  SL No:      Month     Amount  Unnamed: 4
0     Self Employed     NaN  12-Jan-04  495414.75         NaN
1     Self Employed     2.0   3-Jan-04  245899.02         NaN
2     Self Employed     3.0  15-Jan-04  259490.06         NaN
3     Self Employed     4.0  25-Jan-04  437555.12         NaN
4     Self Employed     5.0  17-Jan-05  165972.88         NaN
…               …     …         …         …          …
1495   Salaried_Pvt    98.0  25-Jan-04  100000.00         NaN
1496           Govt    99.0  12-Jan-04   10000.00         NaN
1497  Normal Salary   100.0   3-Jan-04   10001.00         NaN
1498  Self Employed   101.0  15-Jan-04   10002.00         NaN
1499   Salaried_MNC   102.0  25-Jan-04   69551.19         NaN
```

```
[1500 rows x 12 columns]
```

2. From the above dataset create the following summaries:

a. How many distinct customers exist?

[13]:
```
unique_customers=customer['Customer'].nunique()

unique_customers
```

[13]:
```
100
```

b. How many distinct categories exist?

[14]:
```
customer['Segment'].value_counts()
```

[14]:
```
Govt             29
Self Employed    23
Normal Salary    22
Salaried_MNC     13
Salaried_Pvt     13
Name: Segment, dtype: int64
```

c. What is the average monthly spend by customers?

[15]:
```
spend['Month'] = pd.to_datetime(spend['Month'])
spend['Monthly'] = spend['Month'].apply(lambda x:pd.Timestamp.
 ↪strftime(x,format="%B"))
spend['Yearly'] = spend['Month'].apply(lambda x:pd.Timestamp.
 ↪strftime(x,format="%Y"))
customer_spend_group= round(spend.groupby(['Yearly','Monthly']).mean())
customer_spend_group
```

[15]:

		Sl No:	Amount
Yearly	Monthly		
2004	April	730.0	235272.0
	February	750.0	221215.0
	January	752.0	251712.0
	March	726.0	232146.0
	May	734.0	217539.0
	November	746.0	270486.0
	September	742.0	310923.0
2005	April	730.0	252181.0
	August	750.0	233735.0

```
          December    762.0  147503.0
          February    751.0  229802.0
          January     746.0  292741.0
          July        758.0  229117.0
          June        756.0  262688.0
          May         747.0  245697.0
          November    750.0  286143.0
          October     760.0  194569.0
          September   760.0  188666.0
2006      April       744.0  232469.0
          August      770.0  240700.0
          December    774.0  270471.0
          February    758.0  253858.0
          January     764.0  282058.0
          July        754.0  287505.0
          June        768.0  177252.0
          March       751.0  247166.0
          May         744.0  236163.0
          November    758.0  182841.0
          October     772.0  236136.0
          September   772.0  158520.0
```

8.1 d. What is the average monthly repayment by customers?

[16]:
```
customer_repay["Month"] = pd.to_datetime(customer_repay["Month"])
customer_repay['Monthly'] = customer_repay['Month'].apply(lambda x:pd.Timestamp.
 ↪strftime(x,format="%B"))
customer_repay['Yearly'] = customer_repay['Month'].apply(lambda x:pd.Timestamp.
 ↪strftime(x,format="%Y"))
monthly_repay=customer_repay.groupby(['Yearly','Monthly'])['Amount'].mean()

monthly_repay
```

[16]:
```
Yearly  Monthly
2004    April        175632.658000
        February     125612.505556
        January      159971.502571
        March        177767.948000
        May          151310.396400
        November     119226.575333
        September    118926.025625
2005    April        121163.452000
        August       149984.104865
        December     179119.050833
        February     157356.791358
        January      181260.750000
```

```
         July          149944.928333
         June           97052.375833
         May           200121.848636
         November      169225.884048
         October       136268.268333
         September      73559.961667
2006     April         180529.321556
         August        161553.497500
         December      201158.939167
         February      188198.167436
         January       204422.038333
         July          170152.780811
         June          165429.070000
         March         154861.950196
         May           171270.320230
         November      145565.170370
         October       203969.589167
         September     199024.565833
Name: Amount, dtype: float64
```

9 e. If the monthly rate of interest is 2.9%, what is the profit for the bank for each month?

(Profit is defined as interest earned on Monthly Profit. Monthly Profit = Monthly repayment – Monthly spend. Interest is earned only on positive profits and not on negative amounts

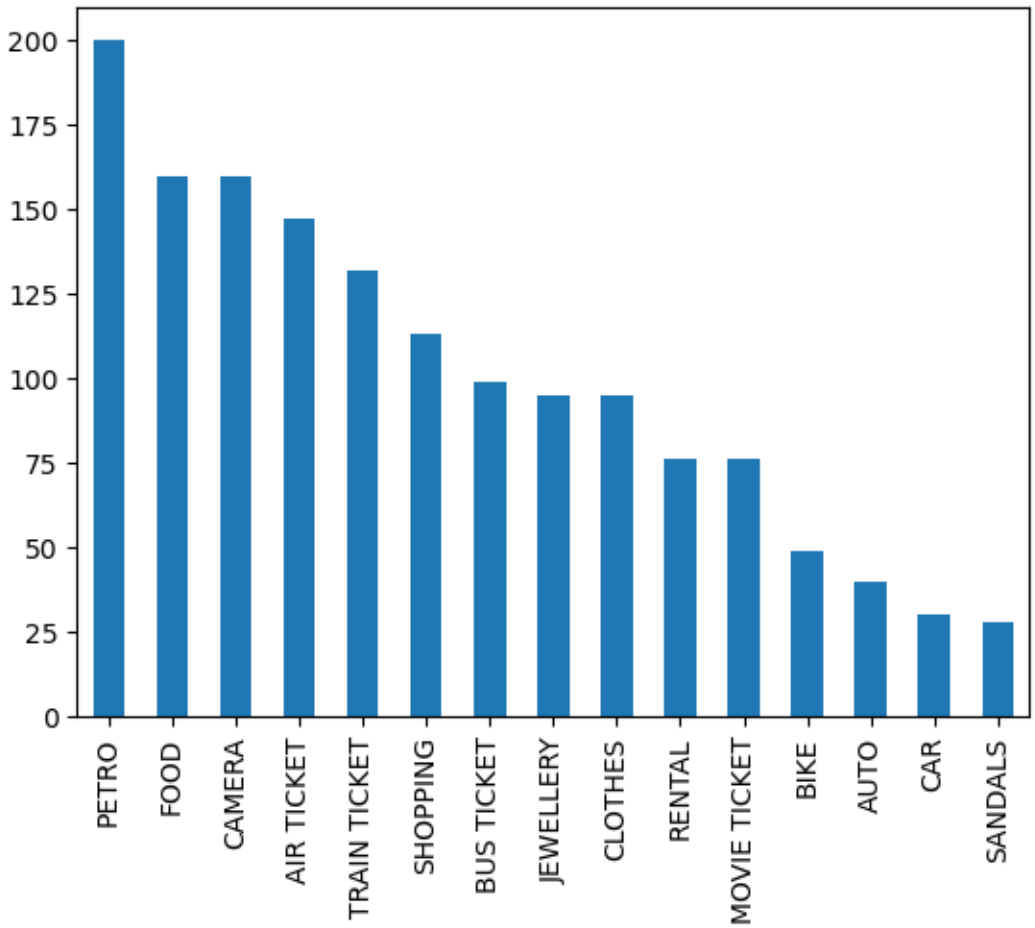
[17]:
```
customer_spend_repay = pd.
 ↪merge(left=customer_spend,right=repayment,on="Customer",how="inner")
customer_spend_repay

customer_spend_repay.rename(columns=["Amount_x":"Spend_Amount","Amount_y":
 ↪"Repay_Amount"],inplace=True)

intrst = customer_spend_repay.
 ↪groupby(["Yearly","Monthly"])['Spend_Amount','Repay_Amount'].sum()

intrst['Monthly Profit'] = intrst['Repay_Amount'] - intrst['Spend_Amount']

intrst['Interest Earned'] = (2.9* intrst['Monthly Profit'])/100
intrst
```

```
  File "C:\Users\lenovo\AppData\Local\Temp\ipykernel_7664\2538134583.py", line 
    customer_spend_repay.rename(columns=["Amount_x":"Spend_Amount","Amount_y":
 ↪"Repay_Amount"],inplace=True)
                                                  ^
SyntaxError: invalid syntax
```

10 f. What are the top 5 product types?

[18]:
```
spend['Type'].value_counts(ascending=False).plot(kind='bar')
```

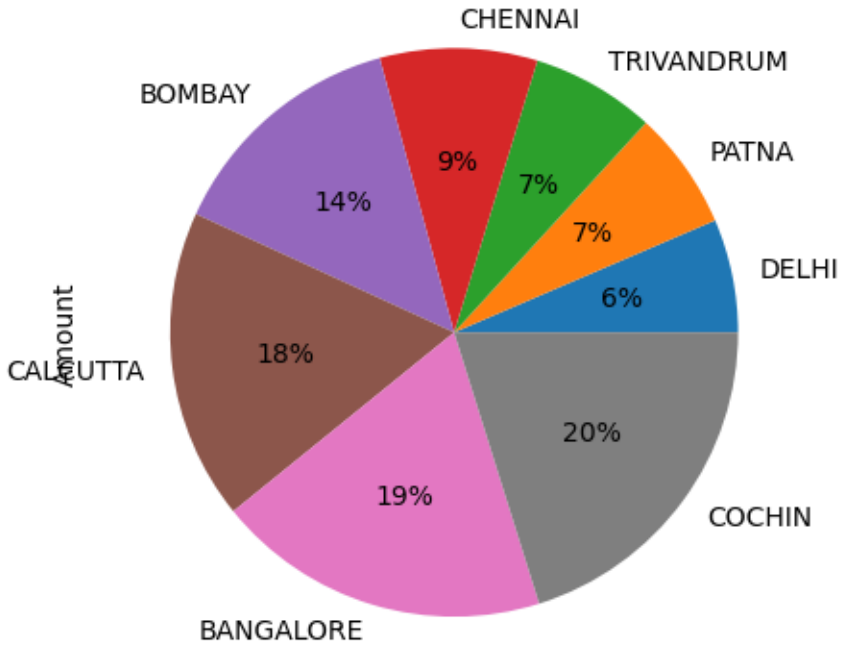
[18]: <AxesSubplot:>



11 g. Which city is having maximum spend?

[27]:
```
customer_spend.groupby("City")["Amount"].sum().sort_values().
↪plot(kind='pie',autopct="%1.0f%%")
```

[27]: <AxesSubplot:ylabel='Amount'>



[28]:
```
print('COCHIN is having maximum spend')
```

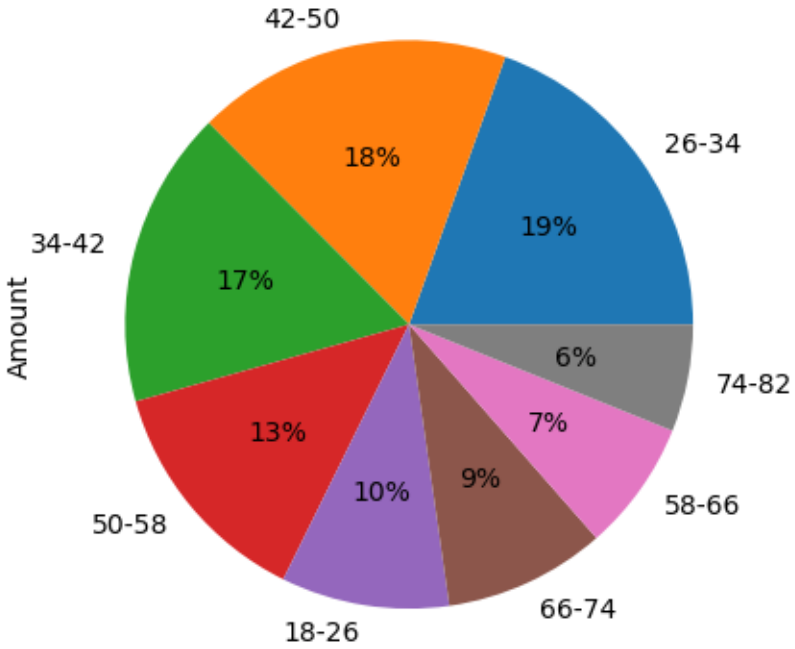
```
COCHIN is having maximum spend
```

12 h. Which age group is spending more money?

[29]:
```
customer_spend['Age Group'] = pd.cut(customer_spend["Age"],bins=np.
↪arange(18,88,8),labels=["18-26","26-34", "34-42" ,"42-50"␣
↪,"50-58","58-66","66-74","74-82"],include_lowest=True)
```

[30]:
```
customer_spend.groupby('Age Group')['Amount'].sum().
↪sort_values(ascending=False).plot(kind='pie',autopct="%1.0f%%")
```

[30]:
```
<AxesSubplot:ylabel='Amount'>
```



[31]:
```
print('Age group of 42-50 is spending more money')
```

```
Age group of 42-50 is spending more money
```

13 i. Who are the top 10 customers in terms of repayment

[32]:
```
customer_repay.groupby('Customer')['Amount'].sum().sort_values(ascending=False).
↪head(10)
```

[32]:
```
Customer
A61    10539142.91
A60     9876290.74
A13     9572000.66
A43     8489871.46
A45     8448334.87
A12     8334760.16
A14     7943268.63
A44     7744730.12
A39     7622483.30
A42     7615460.86
Name: Amount, dtype: float64
```

14 3. Calculate the city wise spend on each product on yearly basis. Also include a graphical representation for the same.

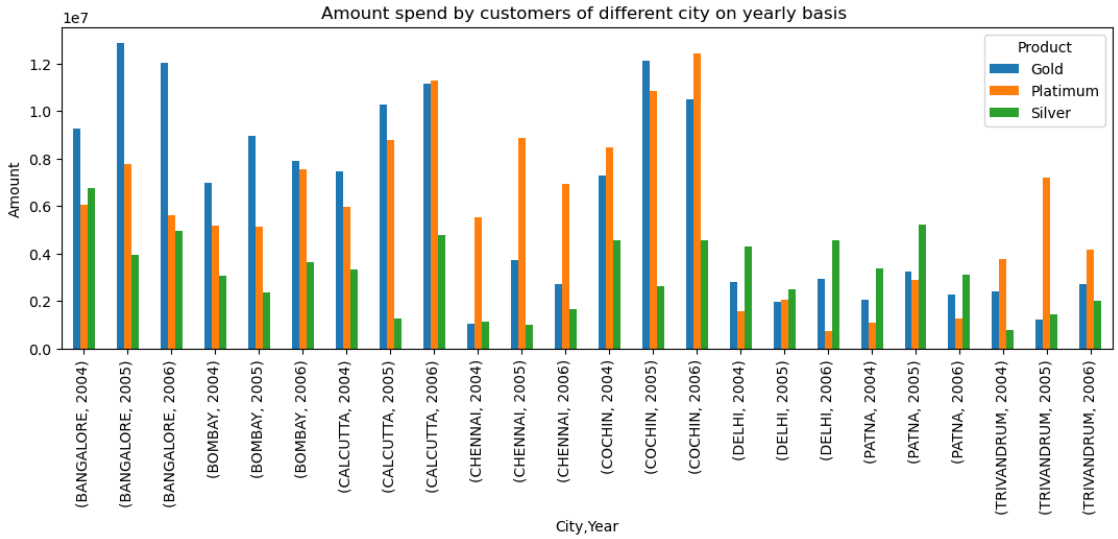
[33]:
```
customer_spend["Month"] = pd.to_datetime(customer_spend["Month"])

customer_spend['Year'] = customer_spend['Month'].apply(lambda x:pd.Timestamp.
 ↪strftime(x,format="%Y"))

customer_spend_pivot = pd.pivot_table(data =␣
 ↪customer_spend,index=["City","Year"],columns='Product',aggfunc="sum",values="Amount")
customer_spend_pivot
```

[33]:

Product		Gold	Platimum	Silver
City	Year			
BANGALORE	2004	9289878.54	6046763.93	6773901.65
	2005	12892362.99	7784194.68	3936068.22
	2006	12030611.09	5620904.86	4967945.66
BOMBAY	2004	6987853.53	5199581.00	3061805.08
	2005	8983018.28	5134074.48	2368375.97
	2006	7917144.31	7547549.72	3628864.37
CALCUTTA	2004	7477140.98	5961165.93	3321963.10
	2005	10303355.80	8810284.03	1285609.11
	2006	11167532.77	11305526.03	4783182.26
CHENNAI	2004	1059618.50	5558572.68	1156129.37
	2005	3740945.58	8868435.43	1000540.48
	2006	2704288.62	6921130.79	1665326.62
COCHIN	2004	7315850.15	8472832.23	4587738.70
	2005	12110613.03	10856722.82	2619231.25
	2006	10499142.38	12453968.83	4579249.87
DELHI	2004	2806495.00	1565199.90	4293224.28
	2005	1964845.27	2068490.16	2515127.59
	2006	2957103.32	764247.73	4581831.29
PATNA	2004	2072567.90	1113069.60	3398795.65
	2005	3252615.77	2883231.07	5244763.90
	2006	2276181.69	1256137.65	3111911.31
TRIVANDRUM	2004	2415102.84	3761433.87	795897.19
	2005	1240375.85	7186762.35	1445540.63
	2006	2735710.87	4174473.45	2005942.36

[34]:
```
customer_spend_pivot.plot(kind='bar',figsize=(13,4))
plt.ylabel('Amount')
plt.title('Amount spend by customers of different city on yearly basis ')
```

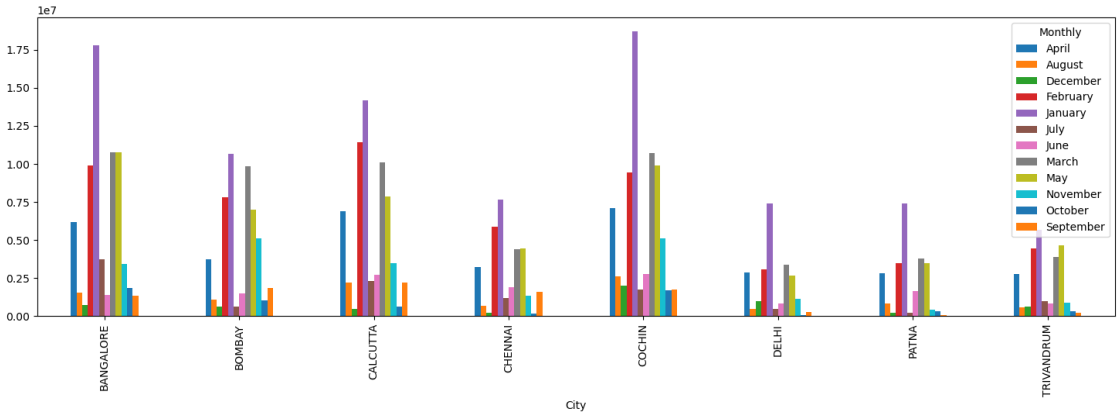
[34]:
```
Text(0.5, 1.0, 'Amount spend by customers of different city on yearly basis ')
```



15 4. Create graphs for

16 a. Monthly comparison of total spends, city wise

[35]:
```
customer_spend['Monthly'] = customer_spend['Month'].apply(lambda x:pd.Timestamp.
↪strftime(x,format="%B"))

month_city = customer_spend.groupby(["Monthly","City"])[["Amount"]].sum().
↪sort_index().reset_index()

month_city =pd.
↪pivot_table(data=customer_spend,values='Amount',index='City',columns='Monthly',aggfunc='sum

month_city.plot(kind="bar",figsize=(18,5))
plt.show()
```



17 b. Comparison of yearly spend on air tickets

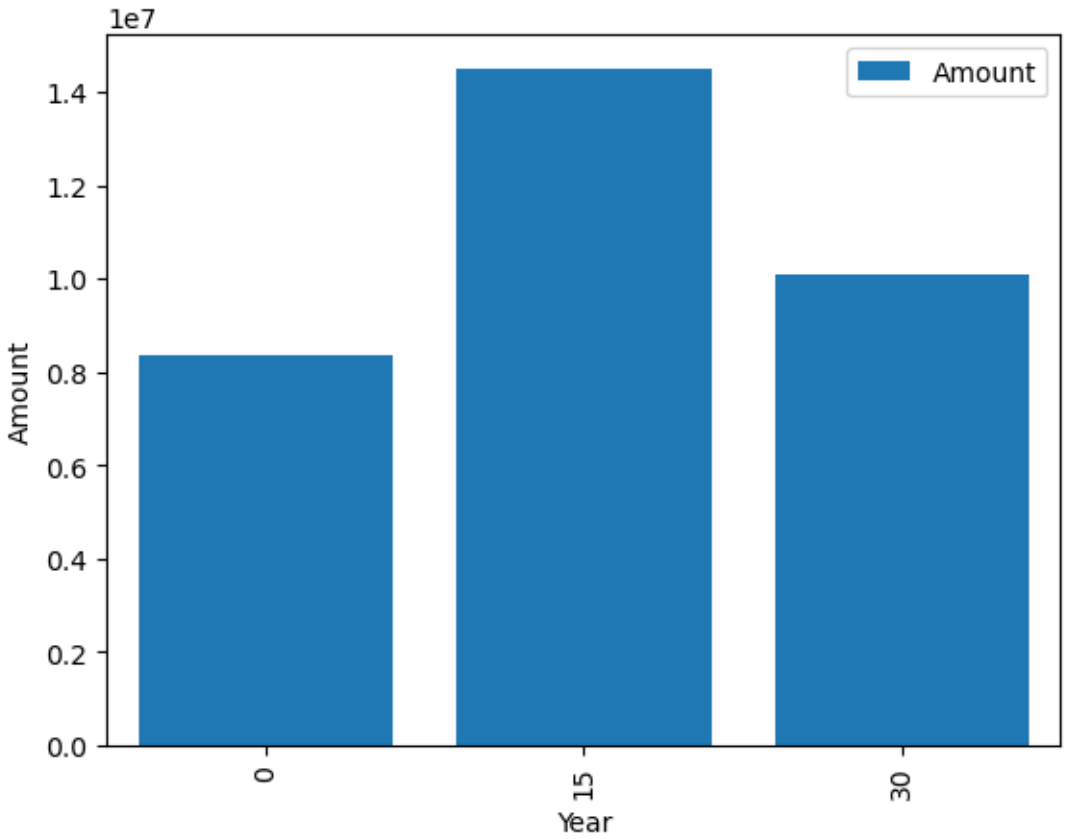
[36]:
```
customer_spend
air_tickets = customer_spend.groupby(["Year","Type"])[["Amount"]].sum().
 ↪reset_index()
filtered = air_tickets.loc[air_tickets["Type"]=="AIR TICKET"]
filtered
```

[36]:

	Year	Type	Amount
0	2004	AIR TICKET	8370914.59
15	2005	AIR TICKET	14495718.73
30	2006	AIR TICKET	10088812.10

[37]:
```
filtered.plot(kind='bar')
plt.bar(filtered["Year"],height=filtered["Amount"])
plt.xlabel('Year')
plt.ylabel('Amount')
```

[37]: Text(0, 0.5, 'Amount')

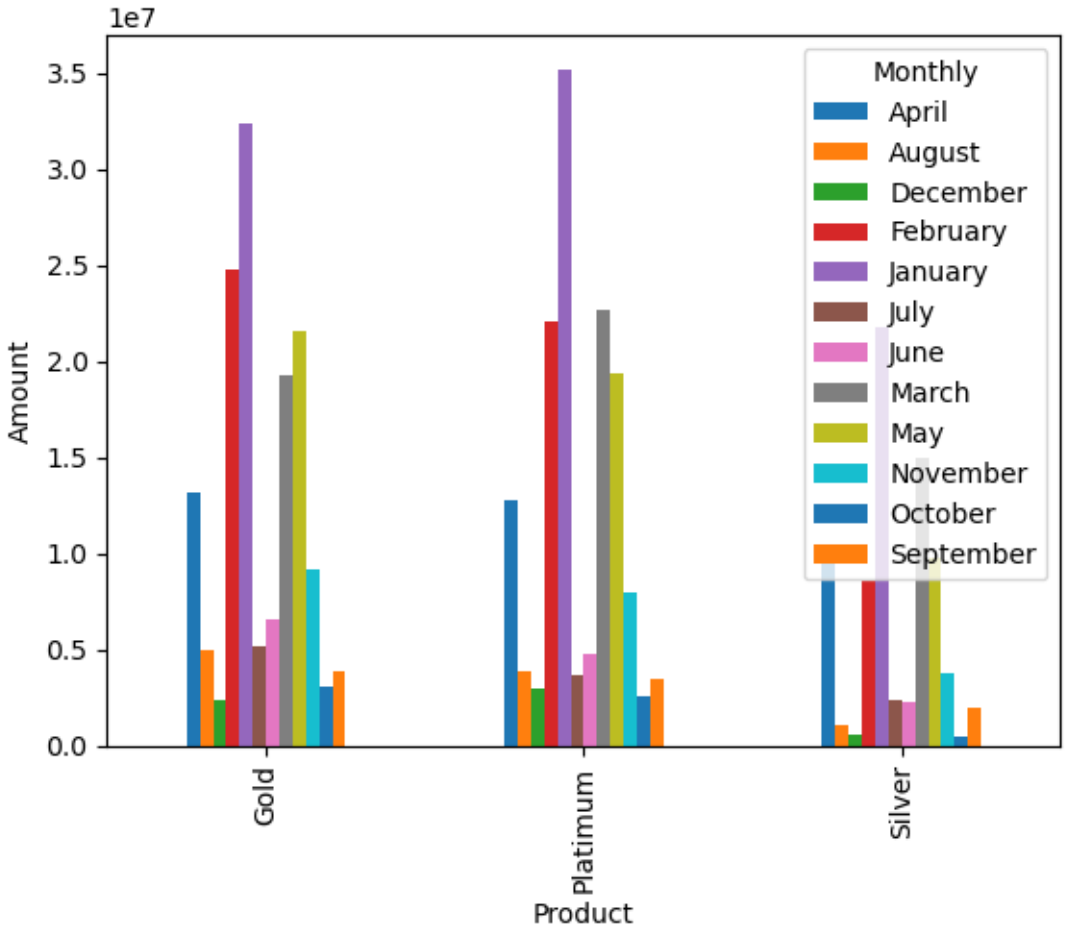


18 c. Comparison of monthly spend for each product

(look for any seasonality that exists in terms of spend)

[38]:
```
product = pd.
  ↪pivot_table(data=customer_spend,index='Product',columns='Monthly',values='Amount',aggfunc='
product.plot(kind='bar')
plt.ylabel('Amount')
```

[38]:
```
Text(0, 0.5, 'Amount')
```



18.1 5. Write user defined PYTHON function to perform the following analysis:

You need to find top 10 customers for each city in terms of their repayment amount by different products and by different time periods i.e. year or month.

The user should be able to specify the product (Gold/Silver/Platinum) and time period (yearly or monthly) and the function should automatically take these inputs while identifying the top 10 customer

[39]:
```
customer_repay['Month'] = pd.to_datetime(customer_repay['Month'])

#creating new column "Monthly" and "Yearly" using already existing 'Month' column

customer_repay['Monthly'] = customer_repay['Month'].apply(lambda x:pd.Timestamp.strftime(x,format="%B"))
customer_repay['Yearly'] = customer_repay['Month'].apply(lambda x:pd.Timestamp.strftime(x,format="%Y"))
```

[40]:
```
def summary_report(product,timeperiod):
    print('Give the product name and timeperiod for which you want the data')
    if product.lower()=='gold' and timeperiod.lower()=='monthly':
        pivot = customer_repay.pivot_table(index=['Product','City','Customer'],columns='Monthly',aggfunc='sum',values='Amo
        result = pivot.loc[('Gold',['BANGALORE','COCHIN','CALCUTTA','BOMBAY','CHENNAI','TRIVANDRUM','PATNA','DELHI
    elif product.lower()=='gold' and timeperiod.lower()=='yearly':
        pivot = customer_repay.pivot_table(index=['Product','City','Customer'],columns='Yearly',aggfunc='sum',values='Amou
        result = pivot.loc[('Gold',['BANGALORE','COCHIN','CALCUTTA','BOMBAY','CHENNAI','TRIVANDRUM','PATNA','DELHI
    elif product.lower()=='silver' and timeperiod.lower()=='monthly':
        pivot = customer_repay.pivot_table(index=['Product','City','Customer'],columns='Monthly',aggfunc='sum',values='Amo
        result = pivot.loc[('Silver',['BANGALORE','COCHIN','CALCUTTA','BOMBAY','CHENNAI','TRIVANDRUM','PATNA','DEL
    elif product.lower()=='silver' and timeperiod.lower()=='yearly':
        pivot = customer_repay.pivot_table(index=['Product','City','Customer'],columns='Yearly',aggfunc='sum',values='Amou
        result = pivot.loc[('Silver',['BANGALORE','COCHIN','CALCUTTA','BOMBAY','CHENNAI','TRIVANDRUM','PATNA','DEL
    if product.lower()=='platinum' and timeperiod.lower()=='monthly':
        pivot = customer_repay.pivot_table(index=['Product','City','Customer'],columns='Monthly',aggfunc='sum',values='Amo
        result = pivot.loc[('Platinum',['BANGALORE','COCHIN','CALCUTTA','BOMBAY','CHENNAI','TRIVANDRUM','PATNA','D
    elif product.lower()=='platinum' and timeperiod.lower()=='yearly':
```

```
        pivot = customer_repay.
↪pivot_table(index=['Product','City','Customer'],columns='Yearly',aggfunc='sum',values='Amou
        result = pivot.
↪loc[('Platinum',['BANGALORE','COCHIN','CALCUTTA','BOMBAY','CHENNAI','TRIVANDRUM','PATNA','D
↪]
    return result
```

[41]:
```
summary_report('gold','monthly')
```

```
Give the product name and timeperiod for which you want the data
```

[41]:

Monthly			April	August	December	February
Product	City	Customer				
Gold	BANGALORE	A1	508949.16	NaN	NaN	781873.80
		A13	494392.40	873304.51	NaN	2049808.15
		A14	812582.20	198623.13	388821.96	1482923.34
		A30	425694.16	NaN	NaN	1321469.80
		A43	612541.80	NaN	NaN	763846.93
		A63	NaN	NaN	NaN	NaN
		A81	NaN	NaN	NaN	NaN
		A88	NaN	NaN	NaN	NaN
		A92	459105.69	NaN	NaN	NaN
	COCHIN	A11	958466.08	332677.61	NaN	1069789.26
		A16	338710.86	NaN	NaN	331143.85
		A32	333042.82	99528.43	NaN	851127.91
		A45	1435443.21	431769.44	410935.52	1241580.90
		A61	3412860.60	197804.13	NaN	1345911.54
		A65	NaN	NaN	NaN	NaN
		A90	109931.32	NaN	NaN	NaN
		A94	NaN	NaN	NaN	NaN
	CALCUTTA	A10	478747.41	NaN	NaN	652033.51
		A15	NaN	17981.25	NaN	516544.51
		A29	NaN	534103.41	NaN	627147.67
		A31	13054.67	NaN	NaN	962656.24
		A60	2397565.91	735660.40	NaN	1403731.08
		A89	497237.97	NaN	NaN	NaN
	BOMBAY	A12	601326.07	120471.17	NaN	1978117.59
		A17	391463.29	98341.04	479227.30	374865.01
		A42	173199.87	NaN	NaN	476213.23
		A62	1115760.01	NaN	NaN	478764.07
		A91	247781.00	NaN	NaN	NaN
	CHENNAI	A27	379529.81	472123.57	136860.63	841114.02
		A67	NaN	NaN	NaN	NaN
	TRIVANDRUM	A28	115326.71	437823.07	451630.26	968709.79
	PATNA	A26	558432.60	398164.37	417177.67	1298000.32
		A66	NaN	NaN	NaN	NaN
		A95	NaN	NaN	NaN	NaN

	DELHI	A44	346650.76	539494.72	79696.21	745752.19
		A64	NaN	NaN	212614.48	NaN
		A82	NaN	NaN	NaN	NaN
		A93	NaN	NaN	NaN	NaN

Monthly			January	July	June	March	\
Product	City	Customer					
Gold	BANGALORE	A1	2407806.29	297176.74	NaN	NaN	
		A13	2014186.98	424603.55	837101.91	1183569.05	
		A14	993114.87	41962.19	41128.98	1682886.51	
		A30	608066.49	NaN	72609.24	599331.38	
		A43	1978038.15	803935.53	23525.91	1378774.72	
		A63	372179.26	NaN	NaN	NaN	
		A81	NaN	NaN	NaN	207780.32	
		A88	NaN	NaN	NaN	146821.30	
		A92	NaN	NaN	NaN	NaN	
	COCHIN	A11	958556.39	128484.37	NaN	508638.02	
		A16	1071852.72	367100.15	92055.06	886861.02	
		A32	NaN	137150.37	284410.26	540858.16	
		A45	2033076.43	NaN	303151.38	869686.35	
		A61	1755251.93	441593.08	NaN	382632.66	
		A65	468922.60	NaN	NaN	NaN	
		A90	NaN	NaN	NaN	NaN	
		A94	153465.89	NaN	NaN	NaN	
	CALCUTTA	A10	1091589.54	358194.65	NaN	720391.63	
		A15	1051502.61	NaN	49334.38	1738696.08	
		A29	1474900.52	NaN	NaN	1269579.93	
		A31	NaN	130109.14	NaN	845269.25	
		A60	2465584.81	273877.36	657931.74	109367.55	
		A89	NaN	NaN	NaN	NaN	
	BOMBAY	A12	1220387.18	1201928.94	526225.82	1001134.17	
		A17	761645.74	118112.45	257937.78	2007441.50	
		A42	2516249.24	NaN	NaN	579721.80	
		A62	996722.42	NaN	NaN	546251.18	
		A91	NaN	NaN	NaN	NaN	
	CHENNAI	A27	1215350.93	241912.50	484419.68	976107.81	
		A67	67845.60	NaN	NaN	NaN	
	TRIVANDRUM	A28	1034461.46	380460.40	NaN	1547067.86	
	PATNA	A26	1129827.37	213185.89	NaN	416810.29	
		A66	477039.78	NaN	NaN	NaN	
		A95	369405.96	NaN	NaN	NaN	
	DELHI	A44	2415712.91	501962.99	NaN	696222.43	
		A64	NaN	NaN	NaN	NaN	
		A82	NaN	NaN	NaN	342200.74	
		A93	342326.14	NaN	NaN	NaN	

Monthly	May	November	October	September

Product	City	Customer				
Gold	BANGALORE	A1	NaN	571458.18	NaN	186427.50
		A13	632600.75	622592.45	NaN	439840.91
		A14	1418286.63	420638.02	NaN	462300.80
		A30	3086549.14	NaN	NaN	19761.75
		A43	2058983.99	454364.16	415860.27	NaN
		A63	416676.34	211117.12	NaN	NaN
		A81	NaN	NaN	NaN	NaN
		A88	NaN	NaN	NaN	NaN
		A92	NaN	NaN	NaN	NaN
	COCHIN	A11	1354322.30	NaN	NaN	NaN
		A16	572216.09	260514.95	NaN	339373.25
		A32	312176.83	NaN	NaN	NaN
		A45	1657916.98	NaN	NaN	64774.66
		A61	1819382.29	423828.84	289863.84	470014.00
		A65	230667.34	NaN	NaN	NaN
		A90	NaN	NaN	NaN	NaN
		A94	NaN	NaN	NaN	NaN
	CALCUTTA	A10	394690.78	NaN	NaN	NaN
		A15	1953768.97	NaN	227585.03	NaN
		A29	2612139.63	NaN	NaN	NaN
		A31	1432949.95	NaN	NaN	NaN
		A60	1189900.27	294357.01	NaN	348314.61
		A89	NaN	NaN	NaN	NaN
	BOMBAY	A12	1048233.62	NaN	347528.59	289407.01
		A17	382881.69	NaN	412459.19	NaN
		A42	3305341.39	362899.70	197897.82	3937.81
		A62	758067.14	798121.60	276130.73	NaN
		A91	NaN	NaN	NaN	NaN
	CHENNAI	A27	1855518.11	409963.50	NaN	NaN
		A67	55638.77	NaN	NaN	NaN
	TRIVANDRUM	A28	2233253.41	NaN	NaN	NaN
	PATNA	A26	2215839.66	255915.07	351333.85	NaN
		A66	113094.58	NaN	NaN	NaN
		A95	NaN	NaN	NaN	NaN
	DELHI	A44	1988158.24	431079.67	NaN	NaN
		A64	110614.61	NaN	NaN	NaN
		A82	NaN	NaN	NaN	NaN
		A93	NaN	NaN	NaN	NaN

[ ]:

[ ]: