DWDM LAB-5

NAME: MAJJIGA JASWANTH

REGNO:20BCD7171

Question-1

Attribute	Transformation	
	From	To
marital	single	0
	Other	1
housing	no	0
	yes	1
loan	no	0
	yes	1
job	'unknown'	np.nan
	'management'	0
	'technician'	1
	'entrepreneur'	2
	'blue-collar'	3
	'retired'	4
	'admin.'	5
	'services'	6
	'self-employed'	7
	'unemployed'	8
	'housemaid'	9
	'student'	10
education	'unknown'	np.nan
	'tertiary'	0
	'secondary'	1
	'primary'	2
default	no	0
	yes	1
contact	unknown	np.nan
	telephone	0
	cellular	1
month	jan-dec	1-12
poutcome	'unknown'	np.nan
	'failure'	0
	'other'	1
	'success'	2
y	no	0
	yes	1

Code:-

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
df = pd.read_csv("bank.csv")
df.head()
```

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	y
0	30	unemployed	married	primary	no	1787	no	no	cellular	19	oct	79	1	-1	0	unknown	no
1	33	services	married	secondary	no	4789	yes	yes	cellular	11	may	220	1	339	4	failure	no
2	35	management	single	tertiary	no	1350	yes	no	cellular	16	apr	185	1	330	1	failure	no
3	30	management	married	tertiary	no	1476	yes	yes	unknown	3	jun	199	4	-1	0	unknown	no
4	59	blue-collar	married	secondary	no	0	yes	no	unknown	5	may	226	1	-1	0	unknown	no

```
df['marital'] = df['marital'].map({'single': 0, 'married': 1, 'divorced ': 1})
df['housing'] = df['housing'].map({'no': 0, 'yes': 1})
df['loan'] = df['loan'].map({'no': 0, 'yes': 1})
df['job'] = df['job'].map({'unknown': np.nan, 'management': 0, 'technic ian': 1, 'ent
repreneur':2, 'bluecollar':3,'retired':4, 'admin.':5, 'services':6, 'self-
 employed':7, 'unemployed':8, 'housemaid':9,'student':10})
df['education'] = df['education'].map({'unknown': np.nan, 'tertiary': 0, 'secondary':
 1, 'primary':2})
df['default'] = df['default'].map({'no': 0, 'yes': 1})
df['contact'] = df['contact'].map({'unknown': np.nan, 'telephone': 0, ' cellular': 1})
df['month'] = df['month'].map({'jan': 0, 'feb': 1, 'mar':2, 'apr':3,'may':4, 'jun':5, 'jul':
6, 'aug':7, 'sep':8, 'oct':9, 'nov':10,'dec':11})
df['poutcome'] = df['poutcome'].map({'unknown': np.nan, 'failure': 0, ' other': 1, 's
uccess':2})
df['y'] = df['y'].map({'no': 0, 'yes': 1})
df.head()
```

Output:

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	y
0	30	8.0	1	2.0	0	1787	0	0	1.0	19	9	79	1	-1	0	NaN	0
1	33	6.0	1	1.0	0	4789	1	1	1.0	11	4	220	1	339	4	0.0	0
2	35	0.0	0	0.0	0	1350	1	0	1.0	16	3	185	1	330	1	0.0	0
3	30	0.0	1	0.0	0	1476	1	1	NaN	3	5	199	4	-1	0	NaN	0
4	59	3.0	1	1.0	0	0	1	0	NaN	5	4	226	1	-1	0	NaN	0

1. b. Develop user defined functions for min-max normalization to [0,1] and Z-Score normalization for the attributes, duration, pdays and balance.

During min-max normalization, observe the data distribution before and after with a plot, and in Z-Score normalization identify data points which are far from three standard deviations.

After data transformation, save the data into preprocessed_bank.csv file.

Code:-

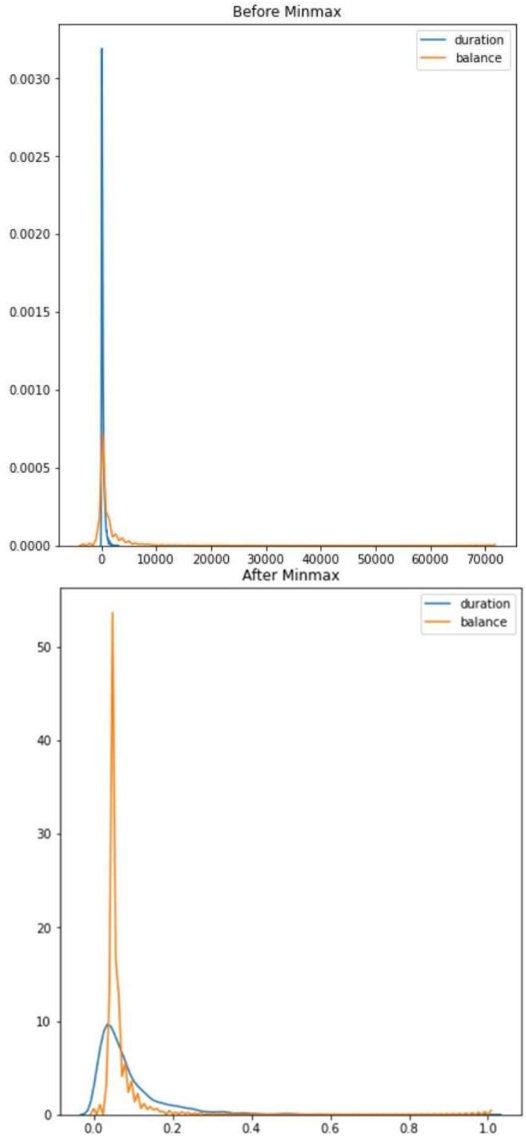
```
df_mm = df.copy()
c = ['duration','pdays','balance']
for column in c:
  df_mm[column] = (df_mm[column] -
 df_mm[column].min())/( df_mm[column].max() - df_mm[column].min())
  df_mm.head()
```

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	y
0	30	8.0	1	2.0	0	0.068455	0	0	1.0	19	9	0.024826	1	0.000000	0	NaN	0
1	33	6.0	1	1.0	0	0.108750	1	1	1.0	11	4	0.071500	1	0.389908	4	0.0	0
2	35	0.0	0	0.0	0	0.062590	1	0	1.0	16	3	0.059914	1	0.379587	1	0.0	0
3	30	0.0	1	0.0	0	0.064281	1	1	NaN	3	5	0.064548	4	0.000000	0	NaN	0
4	59	3.0	1	1.0	0	0.044469	1	0	NaN	5	4	0.073486	1	0.000000	0	NaN	0

```
df_z = df.copy()
c = ['duration','pdays','balance']
for column in c:
df_z[column] = (df_z[column] -
df_z[column].mean()) / df_z[column].std()
df_z.head()
```

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	y
0	30	8.0	1	2.0	0	0.121058	0	0	1.0	19	9	-0.711782	1	-0.407173	0	NaN	0
1	33	6.0	1	1.0	0	1.118521	1	1	1.0	11	4	-0.169175	1	2.988713	4	0.0	0
2	35	0.0	0	0.0	0	-0.024142	1	0	1.0	16	3	-0.303865	1	2.898822	1	0.0	0
3	30	0.0	1	0.0	0	0.017724	1	1	NaN	3	5	-0.249989	4	-0.407173	0	NaN	0
4	59	3.0	1	1.0	0	-0.472701	1	0	NaN	5	4	-0.146086	1	-0.407173	0	NaN	0

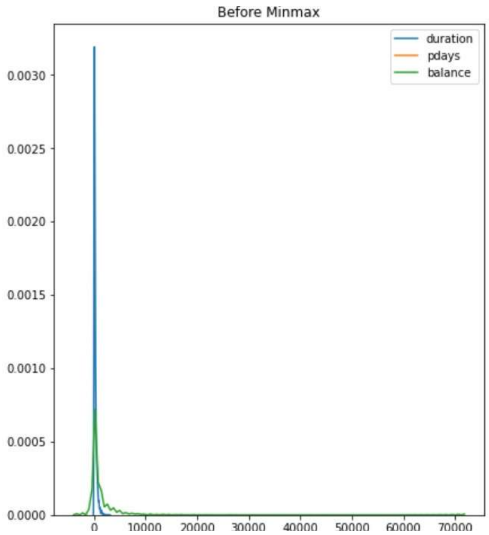
```
fig, (ob1,ob2) = plt.subplots(ncols = 2, figsize= (15,8))
ob1.set_title("Before Minmax") sns.kdeplot(df['duration'], ax=ob1)
sns.kdeplot(df['balance'], ax=ob1)
ob2.set_title("After Minmax") sns.kdeplot(df_mm['duration'], ax=ob2)
sns.kdeplot(df_mm['balance'], ax=ob2)
```

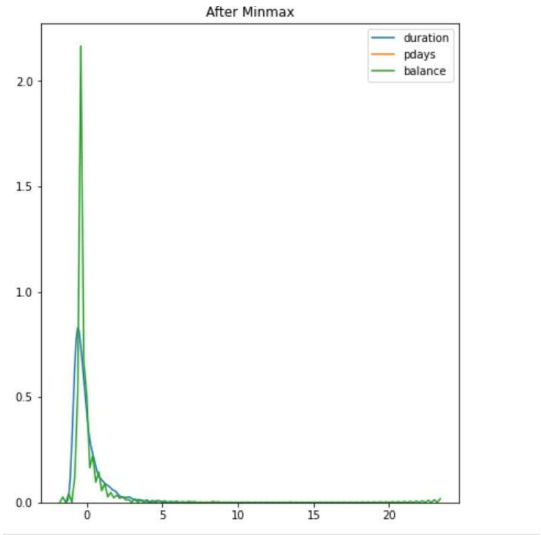


Code:-

```
sns.kdeplot(df['pdays'], ax=ob1)
sns.kdeplot(df_mm['pdays'], ax=ob2)
fig, (ob1,ob2) = plt.subplots(ncols = 2, figsize= (15,8))
ob1.set_title("Before Minmax")
sns.kdeplot(df['duration'], ax=ob1) sns.kdeplot(df['pdays'], ax=ob1)
sns.kdeplot(df['balance'], ax=ob1)
ob2.set_title("After Minmax") sns.kdeplot(df_z['duration'], ax=ob2)
sns.kdeplot(df_z['pdays'], ax=ob2)
sns.kdeplot(df_z['balance'], ax=ob2)
```

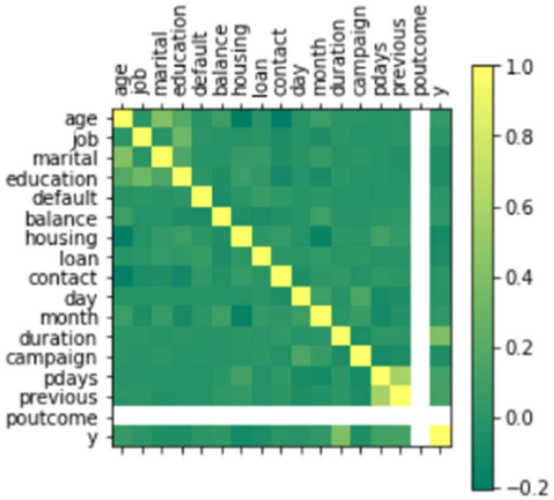
Output:-





Question 2:

2. Reload the preprocessed_bank.csv file and calculate covariance and correlation matrices. Display the correlation matrix with a diagram as:

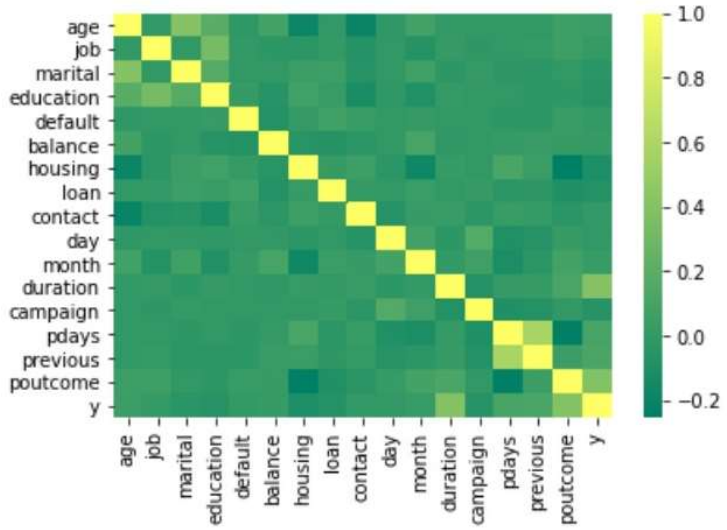


Sort Correlation matrix values of attribute "y" in descending order.

Code:

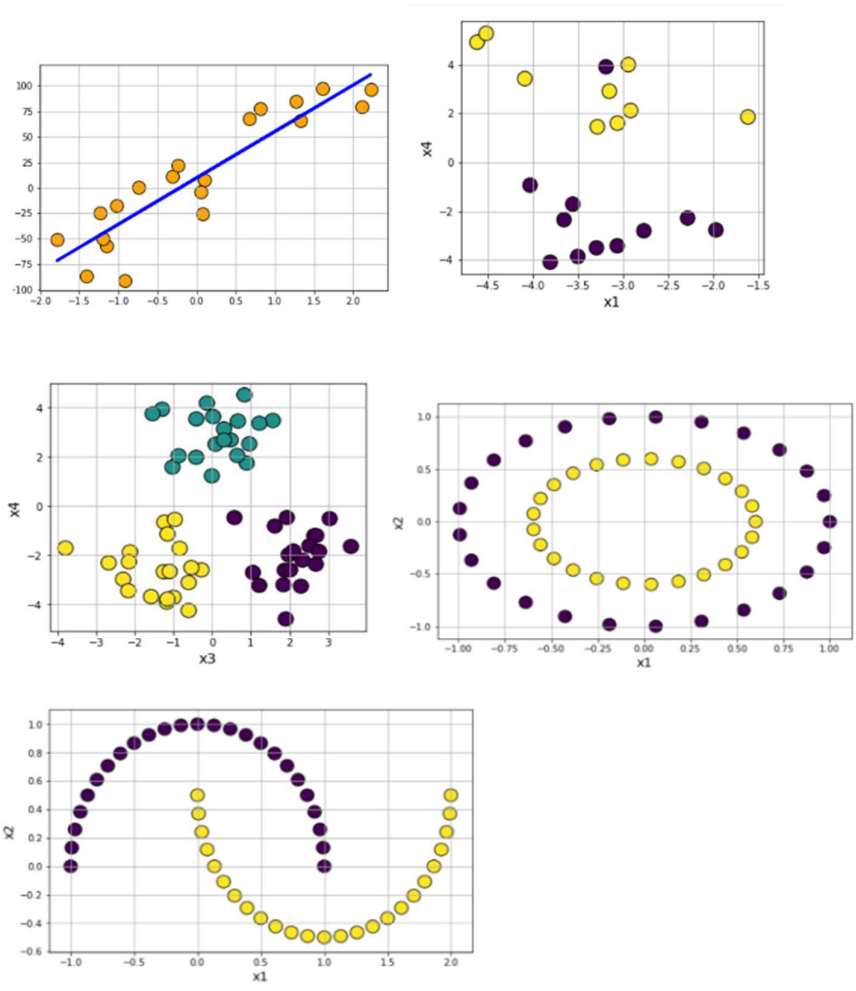
```
dataplot = sns.heatmap(df1.corr(),cmap="summer")
```

output:



Question-3:

3. Generate synthetic data to visualize following kind of images. (Use sklean package)



CODE AND OUTPUT:

```
d1 = make_regression(n_samples = 20, n_features = 4, n_informative = 2, n_targets = 1, bias = 0.0, effective_rank = None,
                     tail_strength = 0.5, noise = 0.0, shuffle = True, coef = False, random_state = None)

df1 = pd.DataFrame(d1[0], columns = ['x'+str(i) for i in range(1,5)])
df1['y'] = d1[1]
df1.head()
```

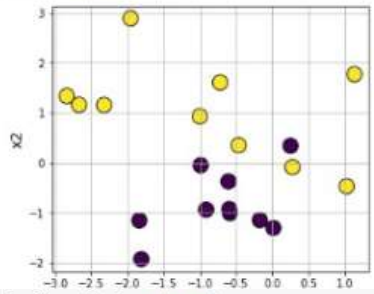
	x1	x2	x3	x4	y
0	-0.219941	-2.067641	-1.373962	1.985504	-67.005156
1	-1.861846	-0.389227	0.302358	-1.179046	-94.668307
2	-1.064474	0.634286	0.278482	-0.046779	-30.466316
3	-1.314073	0.443650	0.770119	0.824777	-46.980608
4	0.497826	-1.414528	1.173223	-0.444175	-16.618808

```
from sklearn.datasets import make_classification
```

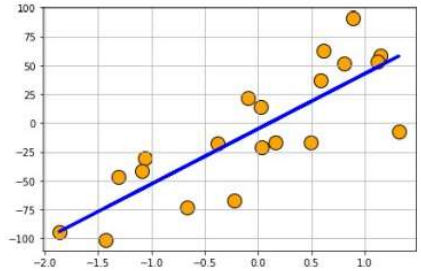
```
d2 = make_classification(n_samples=20, n_features=4, n_informative=4, n_redundant=0, n_repeated=0, n_classes=2,
                         n_clusters_per_class=1, weights=None, flip_y=0.01, class_sep=1.0, hypercube=True,
                         shift=0.0, scale=1.0, shuffle=True, random_state=None)

df2 = pd.DataFrame(d2[0], columns=['x'+str(i) for i in range(1,5)])
df2['y'] = d2[1]
df2.head()
```

	x1	x2	x3	x4	y
0	-2.841544	1.334339	-0.442517	2.545980	1
1	-0.607188	-0.378452	1.535562	2.449090	0
2	-0.486436	0.349992	1.473063	2.805494	1
3	0.251321	0.337464	2.101803	2.771056	0
4	0.277327	-0.090631	1.751864	1.240031	1

```
from itertools import combinations
from math import ceil
lst_var = list(combinations(df2.columns[:-1],2))
len_var = len(lst_var)
plt.figure(figsize = (18,10))
plt.subplot(2, ceil(len_var/2),1)
var1 = lst_var[1-1][0]
var2 = lst_var[1-1][1]
plt.scatter(df2[var1], df2[var2], s=200, c=df2['y'], edgecolor = 'k')
plt.xlabel(var1. fontsize = 14)
plt.ylabel(var2, fontsize = 14)
plt.grid(True)
```




```
plt.figure(figsize = (15,10))
fit = np.polyfit(df1[df1.columns[1-1]], df1['y'], 1)
fit_fn = np.poly1d(fit)
plt.subplot(2,2,1)
plt.scatter(df1[df1.columns[1-1]], df1['y'], s=200, c='orange',edgecolor = 'k')
plt.plot(df1[df1.columns[1-1]], fit_fn(df1[df1.columns[1-1]]),'b-', lw = 3)
plt.grid(True)
```

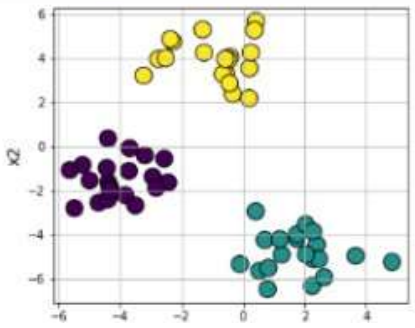


```
from sklearn.datasets import make_blobs
```

```
d3 = make_blobs(n_samples=60, n_features=4, centers=3, cluster_std=1.0, center_box=(-5.0, 5.0),
           shuffle=True, random_state=None)
df3 = pd.DataFrame(d3[0], columns = ['x'+str(i) for i in range(1,5)])
df3['y'] = d3[1]
df3.head()
```

	x1	x2	x3	x4	y
0	-4.712138	-2.549029	0.779563	0.075924	0
1	0.201801	3.539015	2.431481	5.169500	2
2	2.393452	-4.444970	-2.810233	-2.723678	1
3	-4.397631	-1.621647	1.705371	-1.683557	0
4	-0.567559	3.278270	2.854863	5.435253	2

```
from itertools import combinations
from math import ceil
lst_var = list(combinations(df2.columns[:-1],2))
len_var = len(lst_var)
plt.figure(figsize = (18,10))
plt.subplot(2, ceil(len_var/2),1)
var1 = lst_var[1-1][0]
var2 = lst_var[1-1][1]
plt.scatter(df3[var1], df3[var2], s=200, c=df3['y'], edgecolor = 'k')
#plt.xlabel(var1. fontsize = 14)
plt.ylabel(var2, fontsize = 14)
plt.grid(True)
```

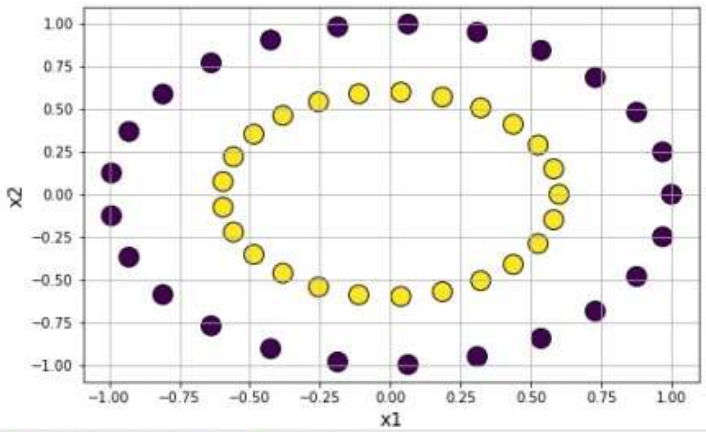



```
from sklearn.datasets import make_circles
```

```
d4 = make_circles(n_samples=50, shuffle=True, noise=None, random_state=None, factor=0.6)
df4 = pd.DataFrame(d4[0], columns = ['x'+str(i) for i in range(1,3)])
df4['y'] = d4[1]
df4.head()
```

	x1	x2	y
0	0.535827	-0.844328	0
1	-0.637424	0.770513	0
2	0.728969	0.684547	0
3	-0.112429	0.589372	1
4	-0.929776	-0.368125	0

```
plt.figure(figsize = (8,5))
plt.scatter(df4['x1'], df4['x2'], c = df4['y'], s = 200, edgecolors = 'k')
plt.xlabel('x1', fontsize = 14)
plt.ylabel('x2', fontsize =14)
plt.grid(True)
plt.show()
```

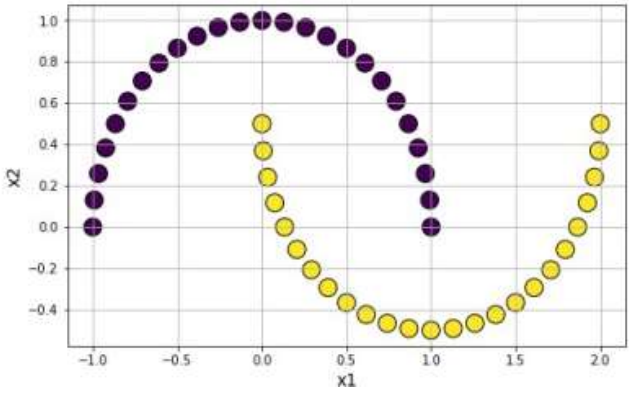



```
from sklearn.datasets import make_moons
```

```
d5 = make_moons(n_samples=50, shuffle=True, noise=None, random_state=None)
df5 = pd.DataFrame(d5[0], columns = ['x'+str(i) for i in range(1,3)])
df5['y'] = d5[1]
df5.head()
```

	x1	x2	y
0	1.608761e+00	-0.293353	1
1	6.123234e-17	1.000000	0
2	1.382683e+00	-0.423880	1
3	3.407417e-02	0.241181	1
4	1.707107e+00	-0.207107	1

```
plt.figure(figsize = (8,5))
plt.scatter(df5['x1'], df5['x2'], c = df5['y'], s = 200, edgecolors = 'k')
plt.xlabel('x1', fontsize = 14)
plt.ylabel('x2', fontsize =14)
plt.grid(True)
plt.show()
```



Question-4

4. Create Fake Customer data of super market with the following fields and store it in a CSV file. (Using Faker package and random package)

- Customer id, Customer name, Customer address, date of birth, gender, blood group, email, phone no.
- Customer id, Transaction ID, Purchase amount
- Transaction ID, Product purchased, # of items for each product
- Product Id, Product Name, cost of single product.

Code

```
import numpy as np
import pandas as pd
import csv
from faker import Faker
import datetime
def datagenerate (records, headers):

    fake = Faker('en_US')
    fake1 = Faker('en_GB')

    with open("Customer_data.csv", "wt") as csvFile:
        writer = csv.DictWriter(csvFile, fieldnames=headers)
        writer.writeheader()
        for i in range(records):
            full_name = fake.name()
            FLname = full_name.split(" ")
            Fname = FLname[0]
            Lname = FLname[1]
            gender = np.random.choice(["Male", "Female"], p=[0.5, 0.5])
            first_name = fake.first_name_male() if gender=="Male" else fake.first_name_female()
            last_name = fake.last_name()
            blood = np.random.choice(["O+", "O-", "A+", "A-", "B+", "B-", "AB+", "AB-"])

            writer.writerow({ "Customer ID": fake.random_int (10000, 9999999),
                              "Fname": first_name,
                              "Lname": last_name,
                              "Email Id": f"{last_name}.{first_name}@vitap.ac.in",
                              "Gender": gender,
                               "Blood Group":blood,
                              "Birth Date": fake.date(pattern="%d-%m-%Y", end_datetime=datetime.date(2000, 1,1)),
                              "Phone Number": fake1.phone_number(),
                              "Address": fake.address(),
                              "Purchase Amount": fake.random_int (1000, 10000)})

if __name__ == '__main__':

    records = 10

    headers = [ "Customer ID", "Fname", "Lname", "Email Id", "Gender", "Blood Group", "Birth Date",
              "Phone Number", "Address", "Purchase Amount"]
    datagenerate(records, headers)
    print("CSV generation Complete")
df = pd.read_csv("Customer_data.csv")
df
```

Output:

	Customer ID	Fname	Lname	Email Id	Gender	Blood Group	Birth Date	Phone Number	Address	Purchase Amount
0	4651411	Matthew	Stone	Stone.Matthew@vitap.ac.in	Male	AB-	18-06-1984	(0118) 4960457	8200 Daniels Club Apt. 092\r\nEast Caleb, NC 4...	3096
1	2819598	Seth	Rasmussen	Rasmussen.Seth@vitap.ac.in	Male	A-	11-04-1983	+44808 157 0274	USCGC Nichols\r\nFPO AP 46271	6149
2	2478531	Kayla	Good	Good.Kayla@vitap.ac.in	Female	A-	04-08-1994	+44131 4960514	5575 Green Coves Suite 394\r\nPort Jonathanbur...	2685
3	8900672	Howard	Walker	Walker.Howard@vitap.ac.in	Male	A+	13-06-1974	(0113) 4960632	USNS Alexander\r\nFPO AP 28380	4982
4	5523021	Patrick	Ponce	Ponce.Patrick@vitap.ac.in	Male	AB+	21-01-1980	+44(0)115 496 0133	621 Brandt Union Suite 696\r\nPort Brianland, ...	9695
5	7815333	Christopher	Mata	Mata.Christopher@vitap.ac.in	Male	AB+	22-12-1991	+44(0)292018216	943 Heather Meadows Suite 654\r\nNew Travisbur...	2003
6	2834744	Nathaniel	Harrison	Harrison.Nathaniel@vitap.ac.in	Male	O-	06-01-1982	+44(0)808 1570298	2354 Esparza Ways Apt. 869\r\nWigginsmouth, PA...	2820
7	2746460	Jacob	Gibson	Gibson.Jacob@vitap.ac.in	Male	AB+	25-10-1992	+44(0)20 7496 0632	197 Robert Run Apt. 748\r\nNorth Rachelfurt, M...	9224
8	6463572	Jessica	Willis	Willis.Jessica@vitap.ac.in	Female	A+	18-09-1994	(01632)960810	69396 Johnson Corner\r\nEast Angela, ND 83924	3435
9	6819845	Elizabeth	Cruz	Cruz.Elizabeth@vitap.ac.in	Female	B+	10-12-1987	+44(0)114 4960860	56359 Fowler Forest\r\nCollinsville, NH 08761	7369