```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

import warnings
warnings.filterwarnings("ignore")


from google.colab import files
uploaded = files.upload()
```

Choose Files Churn_Modelling.csv

- **Churn_Modelling.csv**(text/csv) - 684858 bytes, last modified: 10/12/2022 - 100% done

```
Saving Churn_Modelling.csv to Churn_Modelling (1).csv
```

```
df=pd.read_csv("/content/Churn_Modelling.csv")


df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 14 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   RowNumber        10000 non-null  int64  
 1   CustomerId       10000 non-null  int64  
 2   Surname          10000 non-null  object 
 3   CreditScore      10000 non-null  int64  
 4   Geography        10000 non-null  object 
 5   Gender           10000 non-null  object 
 6   Age              10000 non-null  int64  
 7   Tenure           10000 non-null  int64  
 8   Balance          10000 non-null  float64
 9   NumOfProducts    10000 non-null  int64  
 10  HasCrCard        10000 non-null  int64  
 11  IsActiveMember   10000 non-null  int64  
 12  EstimatedSalary  10000 non-null  float64
 13  Exited           10000 non-null  int64  
dtypes: float64(2), int64(9), object(3)
memory usage: 1.1+ MB
```

```
df.describe()
```

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balar
count	10000.00000	1.000000e+04	10000.000000	10000.000000	10000.000000	10000.000(
mean	5000.50000	1.569094e+07	650.528800	38.921800	5.012800	76485.889:
std	2886.89568	7.193619e+04	96.653299	10.487806	2.892174	62397.405:
min	1.00000	1.556570e+07	350.000000	18.000000	0.000000	0.000(

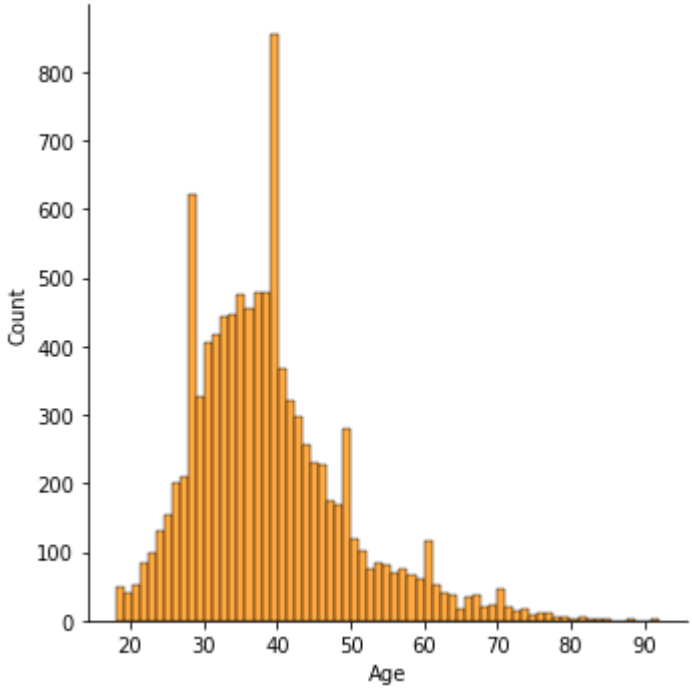
```
**Univariate·analysis**
```

Univariate analysis

Histogram

```
sns.displot(df["Age"], color='darkorange')
```

```
<seaborn.axisgrid.FacetGrid at 0x7f338c48abd0>
```

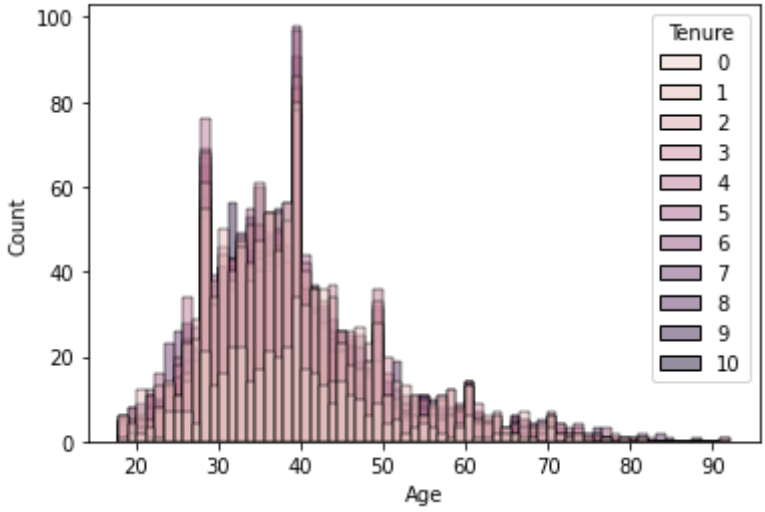



```
sns.histplot(y="Age",data=df,color='darkorange')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f3389a104d0>
```


```
sns.histplot(x='Age',data=df,hue=df['Tenure'])
```

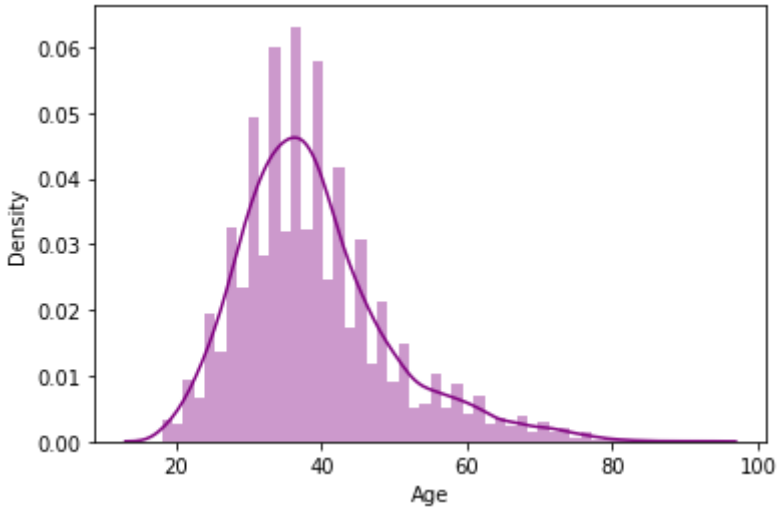
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f338940c090>
```



Displot

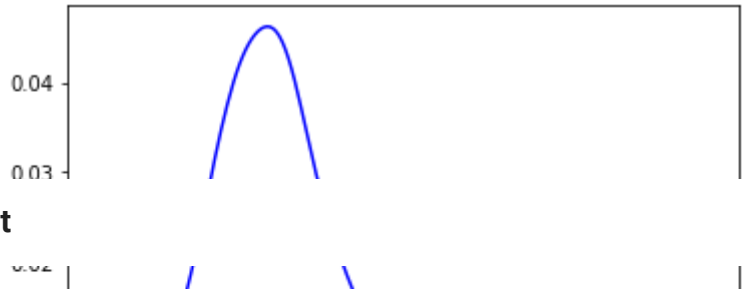
```
sns.distplot(df["Age"],color='purple')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f3389421690>
```




```
sns.distplot(df["Age"],hist=False,color='blue')
```

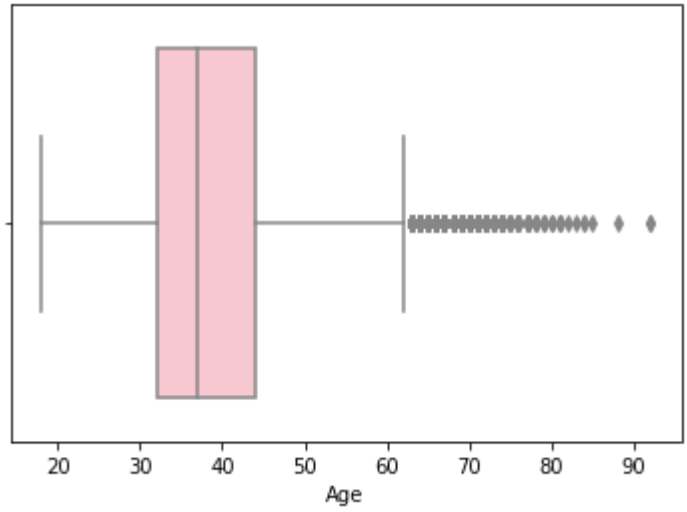
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f3389531cd0>
```



Box plot

```
sns.boxplot(df["Age"],color='pink')
```

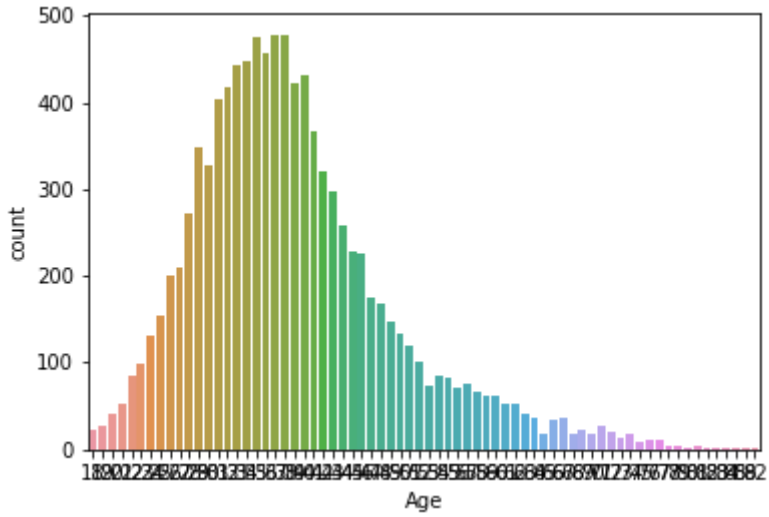
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f33889f03d0>
```



Count plot

```
sns.countplot(df['Age'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f3388902410>
```

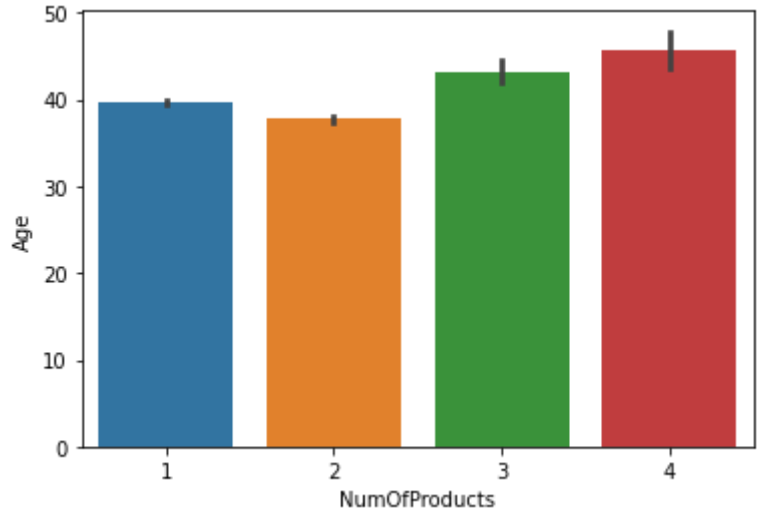


Bivariate analysis

Bar plot

```
sns.barplot(df["NumOfProducts"],df["Age"])
```

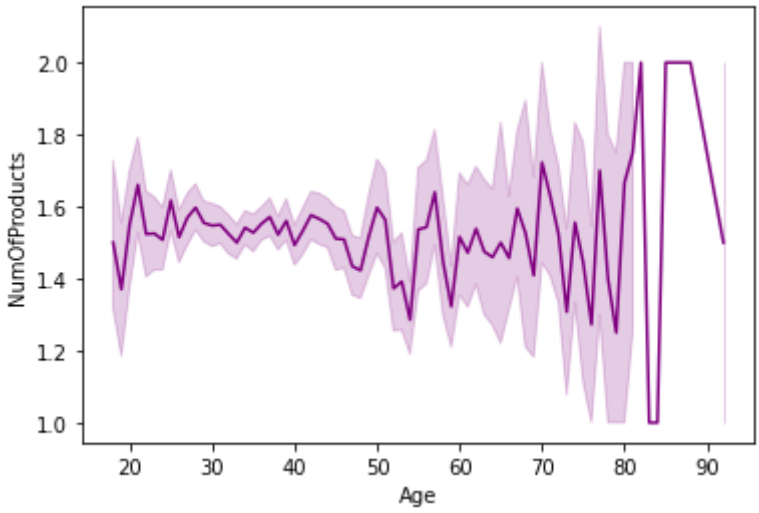
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f3388717f10>
```



Lineplot

```
sns.lineplot(df["Age"],df["NumOfProducts"], color='purple')
```

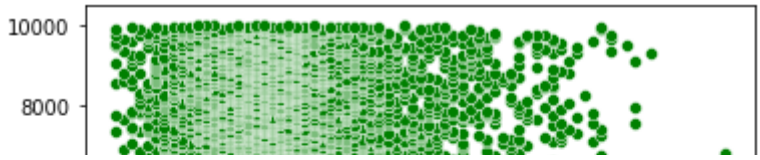
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f338874a1d0>
```



Scatterplot

```
sns.scatterplot(x=df.Age,y=df.RowNumber,color='green')
```

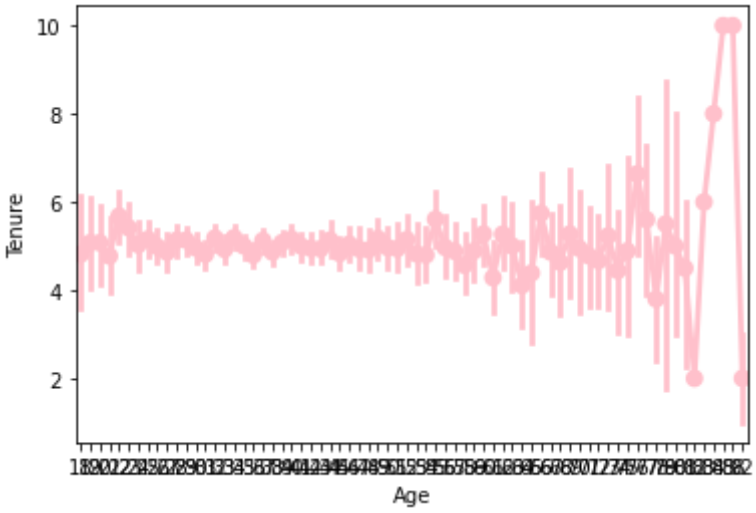
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f2c1dccea10>
```



Pointplot

```
sns.pointplot(x='Age',y='Tenure',data=df,color='pink')
```

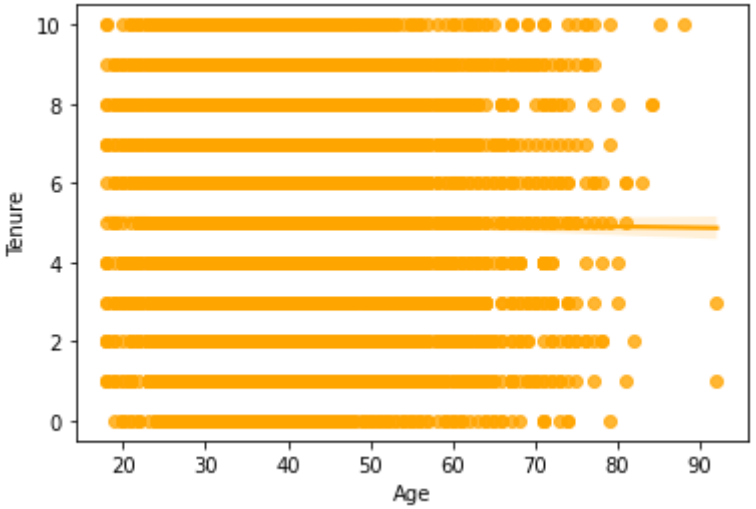
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f33885e8b10>
```



Regplot

```
sns.regplot(df['Age'],df['Tenure'],color='orange')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f33885c7f90>
```

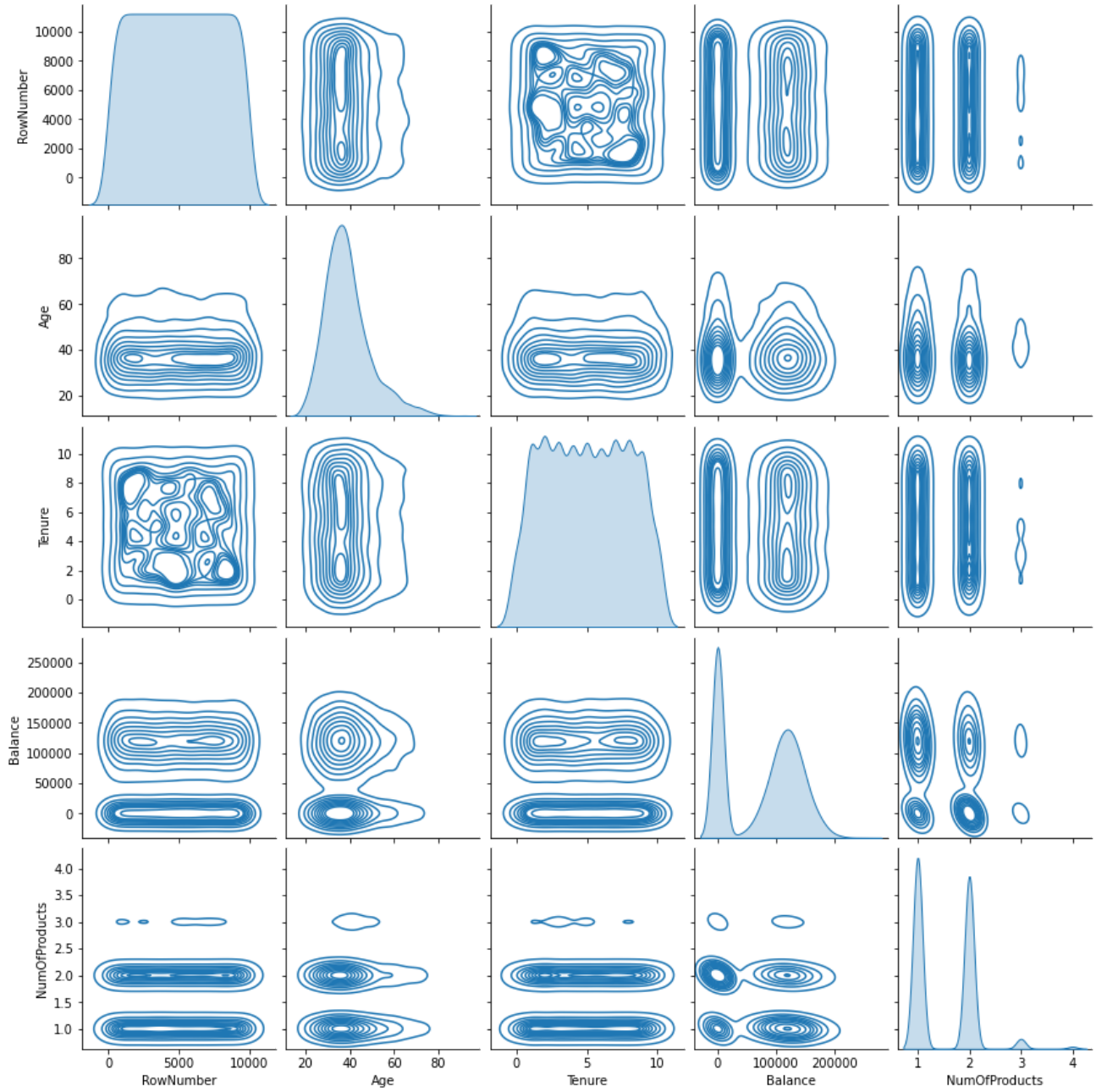


Multi-variate analysis

Pairplot

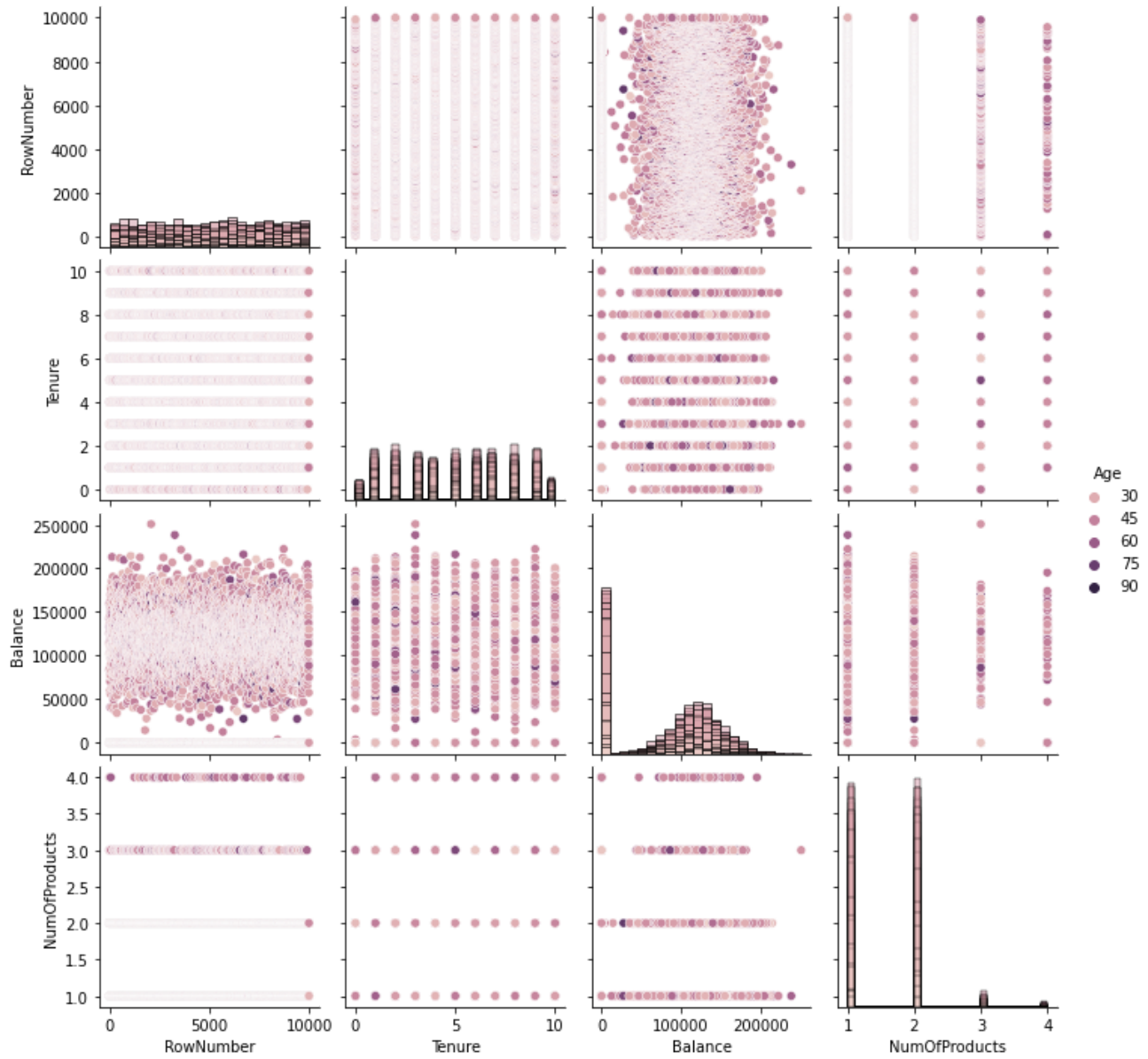
```
sns.pairplot(data=df[["RowNumber","Age","Tenure","Balance","NumOfProducts"]],kind="kde")
```

```
<seaborn.axisgrid.PairGrid at 0x7f33883acc10>
```



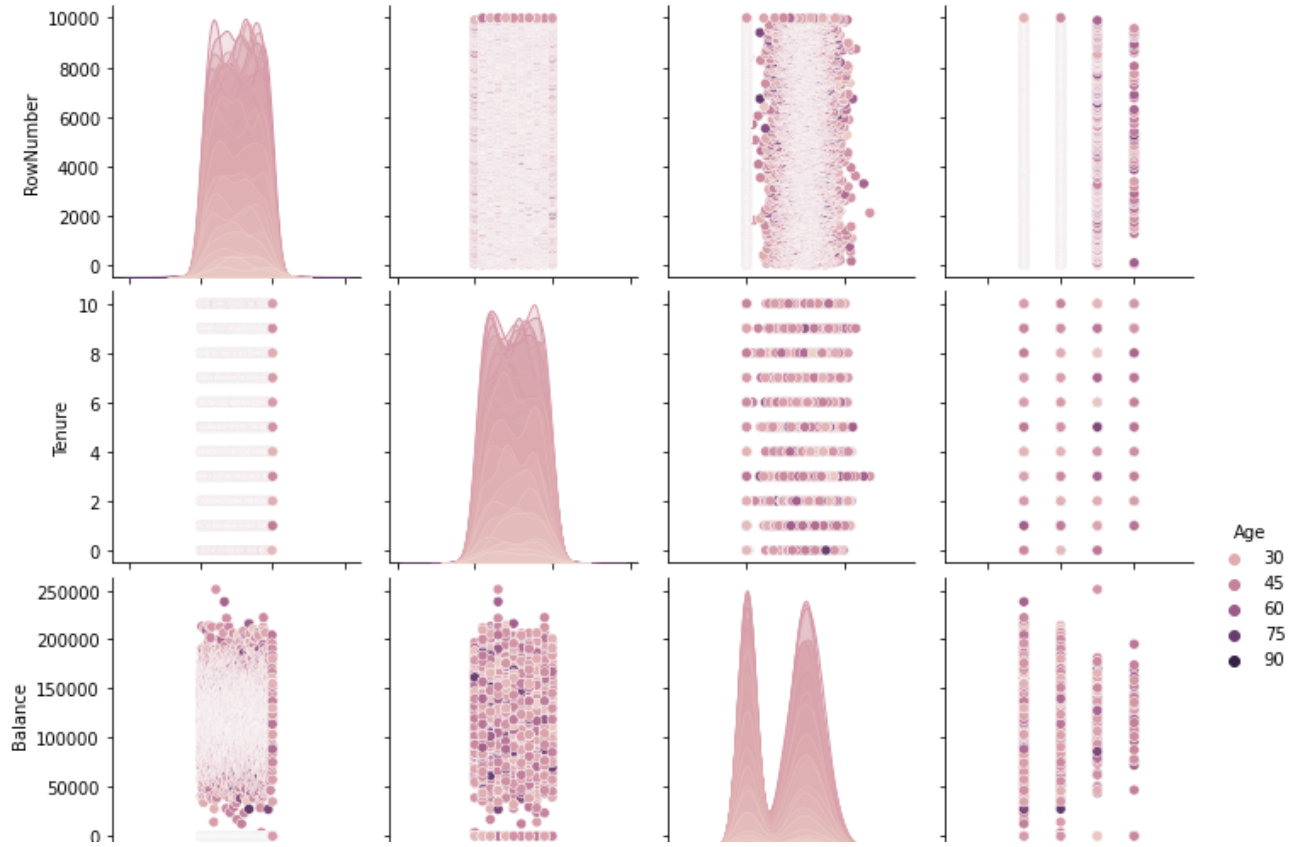

```
sns.pairplot(data=df[["RowNumber","Age","Tenure","Balance","NumOfProducts"]], hue="Age", d
```

```
<seaborn.axisgrid.PairGrid at 0x7f33874d58d0>
```




```
sns.pairplot(data=df[["RowNumber","Age","Tenure","Balance","NumOfProducts"]], hue="Age")
```

```
<seaborn.axisgrid.PairGrid at 0x7f33881f5b50>
```



Descriptive statistics

```
df.describe()
```

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balar
count	10000.00000	1.000000e+04	10000.000000	10000.000000	10000.000000	10000.000(
mean	5000.50000	1.569094e+07	650.528800	38.921800	5.012800	76485.889
std	2886.89568	7.193619e+04	96.653299	10.487806	2.892174	62397.405
min	1.00000	1.556570e+07	350.000000	18.000000	0.000000	0.000(
25%	2500.75000	1.562853e+07	584.000000	32.000000	3.000000	0.000(
50%	5000.50000	1.569074e+07	652.000000	37.000000	5.000000	97198.540(
75%	7500.25000	1.575323e+07	718.000000	44.000000	7.000000	127644.240(
max	10000.00000	1.581569e+07	850.000000	92.000000	10.000000	250898.090(

Handling the missing values

```
data=pd.DataFrame({"a":[1,2,np.nan],"b":[1,np.nan,np.nan],"c":[1,2,4]})
data
```

	a	b	c
0	1.0	1.0	1
1	2.0	NaN	2
2	NaN	NaN	4

```
data.isnull().any()
```

```
a     True
b     True
c    False
dtype: bool
```

```
data.isnull().sum()
```

```
a    1
b    2
c    0
dtype: int64
```

```
data.fillna(value = "S")
```

	a	b	c
0	1.0	1.0	1
1	2.0	S	2
2	S	S	4

```
data["a"].mean()
```

```
1.5
```

```
data["a"].median()
```

```
1.5
```

Finding and replacing outliers

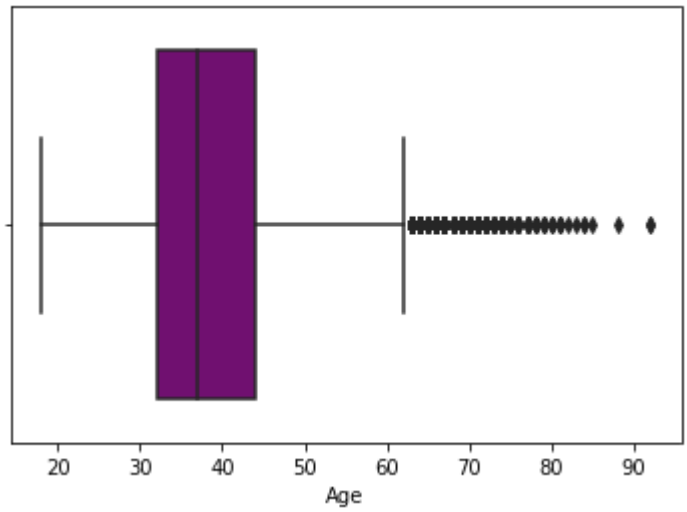
```
outlierss=df.quantile(q=(0.25,0.75))
```

```
outlierss
```

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	Has
0.25	2500.75	15628528.25	584.0	32.0	3.0	0.00	1.0	
0.75	7500.25	15753233.75	718.0	44.0	7.0	127644.24	2.0	

```
sns.boxplot(df["Age"],color='purple')
```

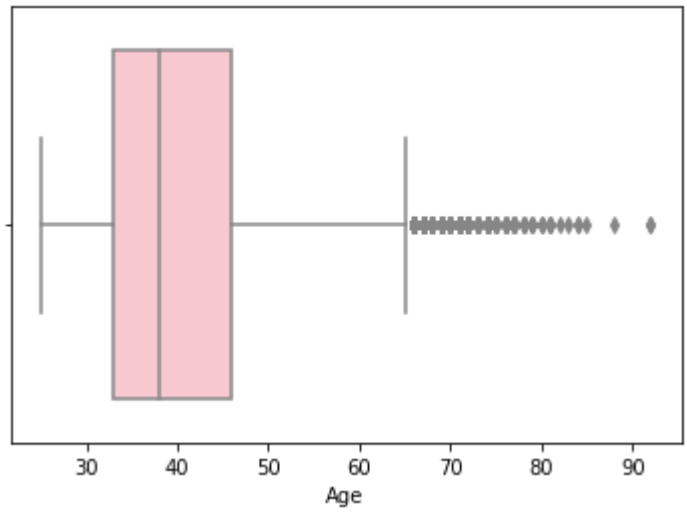
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f337f497650>
```




```
df["Age"]=np.where(df["Age"]<25,50,df["Age"])
```

```
sns.boxplot(df["Age"],color='pink')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f338c44bad0>
```



Check for Categorical columns and perform encoding

```
df.head(4)
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Ba
0	1	15634602	Hargrave	619	France	Female	42	2	
1	2	15647311	Hill	608	Spain	Female	41	1	838
2	3	15619304	Onio	502	France	Female	42	8	1596
3	4	15701354	Boni	699	France	Female	39	1	

```
df["Gender"].replace({"Female":0,"Male":1},inplace = True)
df["Geography"].replace({"France":1,"Spain":2,"Germany":3},inplace = True)
df["Gender"].replace({"Female":0,"Male":1},inplace = True)
df["Geography"].replace({"France":1,"Spain":2,"Germany":3},inplace = True)
```

```
df.head(4)
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Ba
0	1	15634602	Hargrave	619	1	0	42	2	
1	2	15647311	Hill	608	2	0	41	1	838
2	3	15619304	Onio	502	1	0	42	8	1596
3	4	15701354	Boni	699	1	0	39	1	

Split the data into dependent and independent variables

```
y = df["Surname"]
```

```
x=df.drop(columns=["Surname"],axis=1)
```

```
x.head()
```

	RowNumber	CustomerId	CreditScore	Geography	Gender	Age	Tenure	Balance	Num
0	1	15634602	619	1	0	42	2	0.00	
1	2	15647311	608	2	0	41	1	83807.86	
2	3	15619304	502	1	0	42	8	159660.80	
3	4	15701354	699	1	0	39	1	0.00	
4	5	15737888	850	2	0	43	2	125510.82	

Scale the independent variables

```
names=x.columns
names
```

```
Index(['RowNumber', 'CustomerId', 'CreditScore', 'Geography', 'Gender', 'Age',
       'Tenure', 'Balance', 'NumOfProducts', 'HasCrCard', 'IsActiveMember',
       'EstimatedSalary', 'Exited'],
      dtype='object')
```

```
from sklearn.preprocessing import scale
```

```
X=scale(x)
```

```
X
```

```
array([[-1.73187761, -0.78321342, -0.32622142, ...,  0.97024255,
         0.02188649,  1.97716468],
       [-1.7315312 , -0.60653412, -0.44003595, ...,  0.97024255,
         0.21653375, -0.50577476],
       [-1.73118479, -0.99588476, -1.53679418, ..., -1.03067011,
         0.2406869 ,  1.97716468],
       ...,
       [ 1.73118479, -1.47928179,  0.60498839, ...,  0.97024255,
        -1.00864308,  1.97716468],
       [ 1.7315312 , -0.11935577,  1.25683526, ..., -1.03067011,
        -0.12523071,  1.97716468],
       [ 1.73187761, -0.87055909,  1.46377078, ..., -1.03067011,
        -1.07636976, -0.50577476]])
```

```
x = pd.DataFrame(X,columns = names )
x
```

	RowNumber	CustomerId	CreditScore	Geography	Gender	Age	Tenure	
0	-1.731878	-0.783213	-0.326221	-0.902587	-1.095988	0.179622	-1.041760	-
1	-1.731531	-0.606534	-0.440036	0.301665	-1.095988	0.080092	-1.387538	
2	-1.731185	-0.995885	-1.536794	-0.902587	-1.095988	0.179622	1.032908	
3	-1.730838	0.144767	0.501521	-0.902587	-1.095988	-0.118968	-1.387538	-
4	-1.730492	0.652659	2.063884	0.301665	-1.095988	0.279152	-1.041760	
...	...	...	...	...	...	...	...	
9995	1.730492	-1.177652	1.246488	-0.902587	0.912419	-0.118968	-0.004426	-
9996	1.730838	-1.682806	-1.391939	-0.902587	0.912419	-0.517088	1.724464	-
9997	1.731185	-1.479282	0.604988	-0.902587	-1.095988	-0.417558	0.687130	-
9998	1.731531	-0.119356	1.256835	1.505917	0.912419	0.179622	-0.695982	-
9999	1.731878	-0.870559	1.463771	-0.902587	-1.095988	-1.213798	-0.350204	

10000 rows × 13 columns

Split the data into training and testing

```
from sklearn.model_selection import train_test_split
```

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

```
x_train.head()
```

	RowNumber	CustomerId	CreditScore	Geography	Gender	Age	Tenure	
7389	0.827747	-0.195066	0.170424	0.301665	-1.095988	-0.616618	-0.004426	-
9275	1.481077	0.810821	-2.312802	1.505917	0.912419	0.179622	-1.387538	-
2995	-0.694379	-1.507642	-1.195351	-0.902587	-1.095988	-1.114268	-1.041760	
5316	0.109639	1.243462	0.035916	0.301665	0.912419	-0.019438	-0.004426	
356	-1.608556	-1.100775	2.063884	0.301665	-1.095988	1.672571	1.032908	

```
x_train.shape,y_train.shape,x_test.shape,y_test.shape
```

```
((8000, 13), (8000,), (2000, 13), (2000,))
```