**Assignment -2**

**Data visualization and Preprocessing**

|  |  |
| --- | --- |
| Assignment Date | 21 September 2022 |
| Student Name | Jothi Priya G |
| Student Roll Number | 610519104045 |
| Maximum Marks | 2 Marks |

# 1.DOWNLOAD THE DATASET

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **2.LOAD THE DATASET**  **import** pandas **as** pd **import** numpy **as** np **import** matplotlib.pyplot **as** plt  **import** seaborn **as** sns  df**=**pd**.**read\_csv('/content/Churn\_Modelling.csv')  df |  |  |  | In [1]:        In [2]:    In [6]:    Out[6]: |
| **Row Cust Sur Cred Geo Ge A Te Bal**  **Num omer na itSco grap nd g nu anc ber Id me re hy er e re e** | **NumO fProdu cts** | **Has CrC ard** | **IsActiv eMem ber** | **Estima Ex tedSal ite ary d** |

1. 1 46021563  Harvegra 619 France lemaFe 42 2 0.00 1 1 1 101348.88 1

Fe 838

1. 2 15647311 Hill 608 Spain male 41 1 07.86 1 0 1 112542.58 0

1561 Oni Fran Fe 4 159 113931

1. 3 9304 o 502 ce ma 2 8 660. 3 1 0 .57 1 le 80

1. 4 15701354 Boni 699 France maFe 39 1 0.00 2 0 0 93826.63 0

le

Mit Fe 125

1. 5 15737888 chell 850 Spain male 43 2 510.82 1 1 1 79084.10 0

**...** ... ... ... ... ... ... ... ... ... ... ... ... ... ...

**99**  1560 Obi Fran Ma 3 5 0.00 2 1 0 96270. 0

9996 jiak 771

**95** 6229 u ce le 9 64

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Row Cust** | | **Sur na me** | **Cred itSco re** | **Geo grap hy** | **Ge nd er** | **A g e** | **Te nu re** | **Bal anc e** | **NumO fProdu cts** | **Has CrC ard** | **IsActiv eMem ber** | **Estima tedSal ary** | **Ex ite d** |
| **Num omer ber Id** |  |
| **9**  **9**  **9**  **6** | 9997 1556  9892 |  | Joh nsto ne | 516 | Fran ce | Ma le | 3  5 | 10 | 573  69.6  1 | 1 | 1 | 1 | 101699  .77 | 0 |
| **9**  **9**  **9**  **7** | 9998 1558  4532 |  | Liu | 709 | Fran ce | Fe ma  le | 3  6 | 7 | 0.00 | 1 | 0 | 1 | 42085.  58 | 1 |
| **9**  **9**  **9**  **8** | 9999 1568  2355 |  | Sab  bati ni | 772 | Ger man y | Ma le | 4  2 | 3 | 750  75.3  1 | 2 | 1 | 0 | 92888.  52 | 1 |
| **9**  **9**  **9**  **9** | 1000 1562  0 8319 |  |  | 792 | Fran ce |  |  | 4 | 130  142.  79 | 1 | 1 | 0 | 38190.  78 | 0 |
| Wal ker | | Fe  ma 2 le 8 | |
| 10000 rows × 14 columns      df**.**head() | | | |  |  |  | |  |  |  |  |  |  |  |
| In [3]:    Out[3]: | |
| **Row Cust Sur Cred**  **Num omer na itSco ber Id me re** | | | | | **Geo grap hy** | **Ge A nd g er e** | | **Te nu re** | **Bal anc e** | **NumO fProdu cts** | **Has CrC ard** | **IsActiv eMemb**  **er** | **Estima Ex tedSala ite ry d** | |

1. 1 46021563 Harvegra  619 France maleFe 24 2 0.00 1 1 1 101348.88 1

Fe 838

1. 2 15647311 Hill 608 Spain male 41 1 07.86 1 0 1 112542.58 0

1. 15619304 Onio 502 France maleFe 24 8 660.15980 3 1 0 113931.57 1

3

1. 15701354 Boni France lemaFe  39 1 0.00 2 0 0 93826.63 0
2. 699

**Row Cust Sur Cred Geo Ge**

**Num omer na itSco grap nd ber Id me re hy er**

**A Te Bal NumO Has IsActiv Estima Ex g nu anc fProdu CrC eMemb tedSala ite e re e cts ard er ry d**

**4**  5 78881573 chelMitl 850 Spain leFe 3 510.12582 79084.10 0

ma 4 2 1 1 1

In [4]: df**.**shape

(10000, 14) Out[4]:

**3.Univariate,Bivariate & MultiVariate**

# Analysis

## Univariate Analysis

df\_france**=**df**.**loc[df['Geography']**==**'France'] df\_spain**=**df**.**loc[df['Geography']**==**'Spain'] In [9]: df\_germany**=**df**.**loc[df['Geography']**==**'Germany']

In [17]:

plt**.**plot(df\_france['Balance'],np**.**zeros\_like(df\_france['Balance']),'o') plt**.**plot(df\_spain['Balance'],np**.**zeros\_like(df\_spain['Balance']),'o') plt**.**plot(df\_germany['Balance'],np**.**zeros\_like(df\_germany['Balance']),'o') plt**.**xlabel('Age') plt**.**show()

## Bivariate Analysis

In [18]:

sns**.**FacetGrid(df,hue**=**"Geography",size**=**5)**.**map(plt**.**scatter,"Age","Balance")**.**a dd\_legend(); plt**.**show()

/usr/local/lib/python3.7/dist-packages/seaborn/axisgrid.py:337: UserWarning : The `size` parameter has been renamed to `height`; please update your cod e.

warnings.warn(msg, UserWarning)

## Multivariate Analysis

In [24]: sns**.**pairplot(df,hue**=**"Gender",size**=**3) /usr/local/lib/python3.7/dist-packages/seaborn/axisgrid.py:2076: UserWarnin g: The `size` parameter has been renamed to `height`; please update your co de. warnings.warn(msg, UserWarning)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| <seaborn.axisgrid.PairGrid at 0x7f9a9f3029d0> | |  |  |  | Out[24]:          In [29]:    Out[29]: |
| **4.Descriptive Statistics** df**.**head() |  |
| **Row Cust Sur Cred Geo Ge A Te**  **Num omer na itSco grap nd g nu ber Id me re hy er e re** | **Bal anc e** | **NumO fProdu cts** | **Has CrC ard** | **IsActiv eMemb er** | **Estima Ex tedSala ite ry d** |

1. 1 15634602 Hargrave 619 France maleFe 42 2 0.00 1 1 1 101348.88 1

1564 Spai Fe 4 838 112542

1. 2 7311 Hill 608 n male 1 1 07.86 1 0 1 .58 0

1. 3 93041561 Onio 502 France maleFe 42 8 660.15980 3 1 0 113931.57 1

1. 4 15701354 Boni 699 France maleFe 39 1 0.00 2 0 0 93826.63 0

1. 5 15737888 chelMitl 850 Spain maFele 43 2 510.12582  1 1 1 79084.10 0

In [30]: df**.**mean() *# Get the mean of each column*

/usr/local/lib/python3.7/dist-packages/ipykernel\_launcher.py:1: FutureWarni ng: Dropping of nuisance columns in DataFrame reductions (with 'numeric\_onl y=None') is deprecated; in a future version this will raise TypeError. Sel ect only valid columns before calling the reduction.

"""Entry point for launching an IPython kernel.

Out[30]:

|  |  |  |
| --- | --- | --- |
| RowNumber | 5.000500e+03 | |
| CustomerId | 1.569094e+07 | |
| CreditScore | 6.505288e+02 | |
| Age | 3.892180e+01 | |
| Tenure | 5.012800e+00 | |
| Balance | | 7.648589e+04 | |  |
| NumOfProducts | | 1.530200e+00 | |  |
| HasCrCard | | 7.055000e-01 | |  |
| IsActiveMember | | 5.151000e-01 | |  |
| EstimatedSalary | | 1.000902e+05 | |  |
| Exited dtype: float64 | | 2.037000e-01 | | In [31]: |
| df**.**mean(axis**=**1) | | *# Get the mean of each row* | |  |

/usr/local/lib/python3.7/dist-packages/ipykernel\_launcher.py:1: FutureWarni ng: Dropping of nuisance columns in DataFrame reductions (with 'numeric\_onl y=None') is deprecated; in a future version this will raise TypeError. Sel ect only valid columns before calling the reduction.

"""Entry point for launching an IPython kernel.

Out[31]:

1. 1.430602e+06
2. 1.440392e+06
3. 1.444860e+06

1. 1.435993e+06
2. 1.449399e+06

...

1. 1.428483e+06
2. 1.430866e+06
3. 1.421579e+06
4. 1.441922e+06
5. 1.437044e+06

Length: 10000, dtype: float64

df**.**median() *# Get the median of each column* In [32]:

/usr/local/lib/python3.7/dist-packages/ipykernel\_launcher.py:1: FutureWarni ng: Dropping of nuisance columns in DataFrame reductions (with 'numeric\_onl y=None') is deprecated; in a future version this will raise TypeError. Sel ect only valid columns before calling the reduction. """Entry point for launching an IPython kernel.

|  |  |  |
| --- | --- | --- |
| RowNumber  CustomerId  CreditScore  Age  Tenure  Balance  NumOfProducts  HasCrCard  IsActiveMember  EstimatedSalary Exited dtype: float64 | 5.000500e+03 1.569074e+07 6.520000e+02 3.700000e+01  5.000000e+00  9.719854e+04  1.000000e+00 1.000000e+00 1.000000e+00  1.001939e+05  0.000000e+00 | Out[32]:                  In [39]: |
| norm\_data **=** pd**.**DataFrame(np**.**random**.**normal(size**=**100000)) | |

norm\_data**.**plot(kind**=**"density",

figsize**=**(10,10)); plt**.**vlines(norm\_data**.**mean(), *# Plot black line at*

*mean*

ymin**=**0, ymax**=**0.4, linewidth**=**5.0);

plt**.**vlines(norm\_data**.**median(), *# Plot red line at median* ymin**=**0, ymax**=**0.4, linewidth**=**2.0, color**=**"red");

In [36]: skewed\_data **=** pd**.**DataFrame(np**.**random**.**exponential(size**=**100000))

skewed\_data**.**plot(kind**=**"density",

figsize**=**(10,10), xlim**=**(**-**1,5));

plt**.**vlines(skewed\_data**.**mean(), *# Plot black line at mean*

ymin**=**0, ymax**=**0.8, linewidth**=**5.0);

plt**.**vlines(skewed\_data**.**median(), *# Plot red line at median* ymin**=**0, ymax**=**0.8, linewidth**=**2.0, color**=**"red");

In [40]: norm\_data **=** np**.**random**.**normal(size**=**50) outliers **=** np**.**random**.**normal(15, size**=**3)

combined\_data **=** pd**.**DataFrame(np**.**concatenate((norm\_data, outliers), axis**=**0))

combined\_data**.**plot(kind**=**"density",

figsize**=**(10,10), xlim**=**(**-**5,20));

|  |  |  |
| --- | --- | --- |
| plt**.**vlines(combined\_data**.**mean(),  ymin**=**0, ymax**=**0.2, linewidth**=**5.0); | *# Plot black line at mean* |  |
| plt**.**vlines(combined\_data**.**median(),  ymin**=**0, ymax**=**0.2, linewidth**=**2.0, color**=**"red"); | *# Plot red line at median* |  |
| df**.**mode() |  | In [42]: |

Out[42]:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Row Num** | | **Cust Sur omer na**  **Id me** | | **Cred Geo Ge itSco grap nd** | | | **A Te Bal g nu anc e re e** | | | **NumO fProdu cts** | **Has CrC ard** | **IsActiv eMem ber** | **Estima Ex tedSala ite ry d** | |
|  | **ber** | **re** | **hy er** | |
| **0** |  | 1 | 1556 Smi | | 850.0 | Fran Ma ce le | | 3  7. 2.0 0.0 | | | 1.0 | 1.0 | 1.0 | 24924. 0.  92 0 | |
| 5701 | th |
| 0 |  |  |
| **1** |  | 2 | 1556  5706 | Na  N | NaN | Na  NaN  N | | N a  N | Na  N | Na  N | NaN | NaN | NaN | NaN | Na  N |
| **2** |  | 3 | 1556  5714 | Na  N | NaN | Na  NaN N | | N a  N | Na  N | Na  N | NaN | NaN | NaN | NaN | Na  N |
| **3** |  | 4 | 1556  5779 | Na  N | NaN | NaN Na | | N a  N | Na  N | Na  N | NaN | NaN | NaN | NaN | Na  N |
|  | N |
| **4** |  | 5 | 1556  5796 | Na  N | NaN | NaN | Na N | N a  N | Na  N | Na  N | NaN | NaN | NaN | NaN | Na  N |
|  |  | ... | ... | ... |  |  | ... | ... | ... | ... | ... | ... | ... |  |  |
| **...** | | ... ... | | ... ... | |
| **9** | |  | 1581  5628 | Na N | NaN NaN | | Na N | N a  N | Na  N | Na  N | NaN | NaN | NaN | NaN | Na  N |
| **9** 9996  **9**  **5** | | |
| **9**  **9** 9997  **9**  **6** | | | 1581  5645 | Na  N | NaN NaN | | Na N | N a  N | Na  N | Na  N | NaN | NaN | NaN | NaN | Na  N |
| **9**  **9** 9998  **9**  **7** | | | 1581  5656 | Na  N | NaN NaN | | Na N | N a  N | Na  N | Na  N | NaN | NaN | NaN | NaN | Na  N |
| **9**  **9** 9999  **9** | | | 1581  5660 | Na N | NaN NaN | | Na  N | N a  N | Na  N | Na  N | NaN | NaN | NaN | NaN | Na  N |

**8**

**9**

**9** 1000 1581 Na

**9** 0 5690 N

**9**

NaN NaN Na Na NaN NaN

N

N

NaN NaN NaN NaN Na N

|  |  |
| --- | --- |
| 10000 rows × 14 columns    **Measures of Spread**  max(df["Age"]) **-** min(df["Age"])  74    five\_num **=** [df["Age"]**.**quantile(0), df["Age"]**.**quantile(0.25), df["Age"]**.**quantile(0.50), df["Age"]**.**quantile(0.75), df["Age"]**.**quantile(1)]  five\_num  [18.0, 32.0, 37.0, 44.0, 92.0]    df["Age"]**.**describe()    count 10000.000000 mean 38.921800 std 10.487806 min 18.000000 25% 32.000000  50% 37.000000 75% 44.000000 max 92.000000  Name: Age, dtype: float64    df["Age"]**.**quantile(0.75) **-** df["Age"]**.**quantile(0.25)  12.0    df**.**boxplot(column**=**"Age", return\_type**=**'axes', figsize**=**(8,8))  plt**.**text(x**=**0.74, y**=**22.25, s**=**"3rd Quartile") plt**.**text(x**=**0.8, y**=**18.75, s**=**"Median") plt**.**text(x**=**0.75, y**=**15.5, s**=**"1st Quartile") plt**.**text(x**=**0.9, y**=**10, s**=**"Min") plt**.**text(x**=**0.9, y**=**33.5, s**=**"Max")  plt**.**text(x**=**0.7, y**=**19.5, s**=**"IQR", rotation**=**90, size**=**25); | In [43]:    Out[43]:  In [45]:            Out[45]:  In [46]:    Out[46]:              In [47]:    Out[47]:  In [49]: |
| df["Age"]**.**var()  109.99408416841683    df["Age"]**.**std() | In [50]:    Out[50]:  In [51]: |

|  |  |
| --- | --- |
| 10.487806451704609    abs\_median\_devs **=** abs(df["Age"] **-** df["Age"]**.**median()) abs\_median\_devs**.**median() **\*** 1.4826 | Out[51]:  In [52]:      Out[52]:      In [53]:    Out[53]:  In [54]:    Out[54]:  In [55]: |
| 8.8956    **Skewness and Kurtosis**    df["Age"]**.**skew() *# Check skewness*    1.0113202630234552    df["Age"]**.**kurt() *# Check kurtosis*    1.3953470615086956 |
| norm\_data **=** np**.**random**.**normal(size**=**100000) |
| skewed\_data **=** np**.**concatenate((np**.**random**.**normal(size**=**35000)**+**2, np**.**random**.**exponential(size**=**65000)), axis**=**0)  uniform\_data **=** np**.**random**.**uniform(0,2, size**=**100000)  peaked\_data **=** np**.**concatenate((np**.**random**.**exponential(size**=**50000), |

np**.**random**.**exponential(size**=**50000)**\***(**-**1)), axis**=**0)

|  |  |
| --- | --- |
| data\_df **=** pd**.**DataFrame({"norm":norm\_data,  "skewed":skewed\_data,  "uniform":uniform\_data,  "peaked":peaked\_data})    data\_df**.**plot(kind**=**"density",  figsize**=**(10,10), xlim**=**(**-**5,5)); | In [56]: |
| data\_df**.**skew()    norm -0.007037 skewed 1.002549 uniform -0.004434 peaked 0.018058 dtype: float64 data\_df**.**kurt()  norm -0.009914 skewed 1.314497 | In [57]:    Out[57]:        In [58]:    Out[58]: |

uniform -1.201740

peaked 2.971592

dtype: float64

**5.Handle the Missing**

**values**

In [83]:

df**=**pd**.**read\_csv('/content/Churn\_Modelling.csv') In [84]:

df**.**head()

Out[84]:

**Row Cust Sur Cred Geo Ge A Te Bal NumO Has IsActiv Estima Ex**

**Num omer na itSco grap nd g nu anc fProdu CrC eMemb tedSala ite**

**ber Id me re hy er e re e cts ard er ry d**

Har

1. 1 15634602 grave 619 France maleFe 42 2 0.00 1 1 1 101348.88 1

1564 Spai Fe 4 838 112542

1. 2 7311 Hill 608 n male 1 1 07.86 1 0 1 .58 0

1561 Onio 502 Fran maFe 4 8 660.159 3 1 0 113931 1

1. 3 9304 ce le 2 80 .57

699 France Fe 3 1 0.00 2 0 0 93826. 0

1. 4 15701354 Boni male 9 63

1573 chelMit 850 Spai maFe 4 2 510.125 1 1 1 79084. 0

1. 5 7888 l n le 3 82 10

In [86]:

df**.**isnull()

Out[86]:

**Row Cust Sur Cred Geo Ge A Te Bal NumO Has IsActiv Estima Ex**

**Num omer na itSco grap nd ge nu anc fProdu CrC eMem tedSala ite**

**ber Id me re hy er re e cts ard ber ry d**

Fals Fal F Fal Fal Fa

1. False False e False False se al se se False False False False lse

se

**Row Cust Sur Cred Geo Ge A**

**Num omer na itSco grap nd ge ber Id me re hy er**

**Te Bal NumO Has IsActiv Estima Ex nu anc fProdu CrC eMem tedSala ite**

**re e cts ard ber ry d**

1. False False False False False False sealF False  False False False False False lseFa

Fals False False Fal alF Fal Fal False False False False Fa

1. False False e se se se se lse

Fals Fal F Fal Fal Fa

1. False False e False False se seal se se False False False False lse

1. False False False False False False sealF False False False False False False lseFa

**...** ... ... ... ... ... ... ... ... ... ... ... ... ... ...

**99**  False False Fals  False False F Fal Fal False False False False Fa Fal al

**9** e se se se se lse

**5**

**99**  False False Fals  False False False alF False False False False False False lseFa

**9** e se

**6**

**99**  False Fals Fal F Fal Fal False False False False Fa False False False al

**9** e se se se se lse

**7**

**99** False False  Fal F Fal Fal Fa

**9** False False False se seal se se False False False False lse

**8**

**99**  False False Fals  False False Fal F Fal Fal False False False False Fa al

**99** e se se se se lse

10000 rows × 14 columns

In [89]: sns**.**heatmap(df**.**isnull(),yticklabels**=False**,cbar**=False**,cmap**=**'viridis')

|  |  |
| --- | --- |
| <matplotlib.axes.\_subplots.AxesSubplot at 0x7f9a987d8290>    sns**.**set\_style('whitegrid') sns**.**countplot(x**=**'Geography',data**=**df)  <matplotlib.axes.\_subplots.AxesSubplot at 0x7f9a92a88850>    sns**.**set\_style('whitegrid')  sns**.**countplot(x**=**'Geography',hue**=**'Gender',data**=**df,palette**=**'RdBu\_r')    <matplotlib.axes.\_subplots.AxesSubplot at 0x7f9a92ec10d0>    sns**.**set\_style('whitegrid')  sns**.**countplot(x**=**'Geography',hue**=**'Gender',data**=**df,palette**=**'rainbow')    <matplotlib.axes.\_subplots.AxesSubplot at 0x7f9a92afac50>    sns**.**distplot(df['Age']**.**dropna(),kde**=False**,color**=**'darkred',bins**=**40) | Out[89]:  In [93]:    Out[93]:    In [94]:    Out[94]:    In [96]:    Out[96]: |

In [97]:

/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: Futur eWarning: `distplot` is a deprecated function and will be removed in a futu re version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function fo r histograms).

warnings.warn(msg, FutureWarning)

|  |  |
| --- | --- |
| <matplotlib.axes.\_subplots.AxesSubplot at 0x7f9a98787590>    df['Age']**.**hist(bins**=**30,color**=**'darkred',alpha**=**0.3)  <matplotlib.axes.\_subplots.AxesSubplot at 0x7f9a92d64c10>    sns**.**countplot(x**=**'NumOfProducts',data**=**df)  <matplotlib.axes.\_subplots.AxesSubplot at 0x7f9a9306f790>    df['Age']**.**hist(color**=**'green',bins**=**40,figsize**=**(8,4))  <matplotlib.axes.\_subplots.AxesSubplot at 0x7f9a90f52d90>  **Cufflinks for plots** | Out[97]:  In [98]:    Out[98]:    In [100]:    Out[100]:    In [101]:    Out[101]: |

|  |  |
| --- | --- |
| **import** cufflinks **as** cf cf**.**go\_offline()    df['Age']**.**iplot(kind**=**'hist',bins**=**30,color**=**'green')    **Data Cleaning** | In [102]:    In [ ]: |
| plt**.**figure(figsize**=**(12, 7))  sns**.**boxplot(x**=**'Gender',y**=**'Age',data**=**df,palette**=**'winter')  <matplotlib.axes.\_subplots.AxesSubplot at 0x7f9a90f59450>    **def** impute\_age(cols):  Age **=** cols[0]  Pclass **=** cols[1]    **if** pd**.**isnull(Age):    **if** Pclass **==** 1:  **return** 37    **elif** Pclass **==** 2: **return** 29    **else**:  **return** 24  **else**:  **return** Age sns**.**heatmap(df**.**isnull(),yticklabels**=False**,cbar**=False**,cmap**=**'viridis')  <matplotlib.axes.\_subplots.AxesSubplot at 0x7f9a8aa699d0>    df**.**drop('Gender',axis**=**1,inplace**=True**)  df**.**head() | In [107]:    Out[107]:  In [307]:                        In [122]:    Out[122]:  In [112]:    In [114]:    Out[114]: |
| **RowN Custo Sur Credi Geog Age Tenure Balance ProductNumOfs HasCrCard IsActiveMember umbe merI nam tScor raph r d e e y** | **Estimat Ex edSalar ite y d** |

Har

1. 1 15634602 grave 619 France 42 2 0.00 1 1 1 101348.88 1

**RowN Custo Sur Credi Geog Age Tenure Balance ProductNumOfs HasCrCard IsActiveMember EstimatedSalary Exited umbe merI nam tScor raph r d e e y**

1. 2 15647311 Hill 608 Spain 41 1 83807.86  1 0 1 112542.58 0

15619 Oni Franc 4 1596 113931. **2** 3 304 o 502 e 2 8 60.80 3 1 0 57 1

1. 4 15701354 Boni 699 France 39 1 0.00 2 0 0 93826.63 0

1. 5 15737888 Mitchell 850 Spain 43 2 125510.82 1 1 1 79084.10 0

## Converting Categorical Features

In [116]: df**.**info()

<class 'pandas.core.frame.DataFrame'> RangeIndex: 10000 entries, 0 to 9999 Data columns (total 13 columns):

# Column Non-Null Count Dtype

1. RowNumber 10000 non-null int64
2. CustomerId 10000 non-null int64
3. Surname 10000 non-null object 3 CreditScore 10000 non-null int64 4 Geography 10000 non-null object
4. Age 10000 non-null int64
5. Tenure 10000 non-null int64
6. Balance 10000 non-null float64 8 NumOfProducts 10000 non-null int64 9 HasCrCard 10000 non-null int64
7. IsActiveMember 10000 non-null int64
8. EstimatedSalary 10000 non-null float64
9. Exited 10000 non-null int64 dtypes: float64(2), int64(9), object(2) memory usage: 1015.8+ KB

In [118]: pd**.**get\_dummies(df['Geography'],drop\_first**=True**)**.**head()

Out[118]:

**Germany Spain**

1. 0 0

**Germany Spain**

1. 0 1

1. 0 0

1. 0 0

1. 0 1

In [124]: df**.**info

Out[124]:

<bound method DataFrame.info of RowNumber CustomerId Surname Cre ditScore Geography Age Tenure \

1. 1 15634602 Hargrave 619 France 42 2
2. 2 15647311 Hill 608 Spain 41 1 2 3 15619304 Onio 502 France 42 8

3 4 15701354 Boni 699 France 39 1 4 5 15737888 Mitchell 850 Spain 43 2

... ... ... ... ... ... ... ...

1. 9996 15606229 Obijiaku 771 France 39 5
2. 9997 15569892 Johnstone 516 France 35 10
3. 9998 15584532 Liu 709 France 36 7
4. 9999 15682355 Sabbatini 772 Germany 42 3
5. 10000 15628319 Walker 792 France 28 4

Balance NumOfProducts HasCrCard IsActiveMember EstimatedSalary

\

1. 0.00 1 1 1 101348.88
2. 83807.86 1 0 1 112542.58
3. 159660.80 3 1 0 113931.57 3 0.00 2 0 0 93826.63 4 125510.82 1 1 1 79084.10 ... ... ... ... ... ... 9995 0.00 2 1 0 96270.64
4. 57369.61 1 1 1 101699.77
5. 0.00 1 0 1 42085.58
6. 75075.31 2 1 0 92888.52
7. 130142.79 1 1 0 38190.78

Exited

1. 1
2. 0
3. 1
4. 0
5. 0

... ...

1. 0
2. 0
3. 1
4. 1
5. 0

|  |  |
| --- | --- |
| [10000 rows x 13 columns]>    sex **=** pd**.**get\_dummies(df['Age'],drop\_first**=True**) embark **=** pd**.**get\_dummies(df['Balance'],drop\_first**=True**) | In [125]: |

n

8

.58

.

In [127]:

df**.**drop(['Age','HasCrCard','Surname','CustomerId'],axis**=**1,inplace**=True**)

In [129]:

df**.**head()

Out[129]:

**RowNum CreditSc Geogra Tenu Balanc NumOfProd IsActiveMe EstimatedSa Exit**

**ber ore phy re e ucts mber lary ed**

1. 1 619 France 2 0.00 1 1 101348.88 1

1. 2 608 Spain 1 83807.86 1 1 112542.58 0

159660.

1. 3 502 France 8 80 3 0 113931.57 1

1. 4 699 France 1 0.00 2 0 93826.63 0

1. 5 850 Spain 2 125510.82 1 1 79084.10 0

In [130]:

train **=** pd**.**concat([df,sex,embark],axis**=**1)

train**.**head() In [131]:

Out[131]:

**2 2 2 2 2 2 2 2 2 2**

**Ro**

**w Cr G T N B Nu IsA Est E 1 1 1 1 1 1 2 22 38 50**

**ed eo n m e u itS co alna mOodufPr ctiveMem imaSalted itxe 1 . 26 26 72 13 34 61 1 7.26 7.83 8.89**

**gr**

**ap u ce cts ber ary d 9 . 9 9 7 4 4 0 5 6 5 0**

**be re hy r**

**.**

**r e 2. 6. 8. 6. 6. 9. 3**

**9 3 2 2 9 8 2.**

**7 2**  **6 8**  **8 3 6 9**

Fr 0. 101 .

an 2 0 1 1 348 1 0 . 0 0 0 0 0 0 0 0 0 0

61 ce 0 .88 .

1. 1 9

Sp 8 112 .

ai 1 3 1 1 542 0 0 . 0 0 0 0 0 0 0 0 0 0

1. 620

8

**Ro**  **1 1 12 12 41 16 22 222 238 250 2 2 2 2**

**w Cr G T B Nu IsA Est E**

**N ed eo al mOfPr ctiveM imated itx . 629 692 277 134 6.43 9.10 15 7.62 7.83 8.89 u itS gr e a**

**m co ap n odu em Sal e 19 . 2. 6. 8. 6. 9 8 6 5 0 be re hy n ce cts ber ary d 9 3 2 2 3 r**  **u .**

**r 2. e**

**7 2**  **6 8**  **8 3 6 9**

7.

8

6

113 .

3 0 931 1 0 . 0 0 0 0 0 0 0 0 0 0

1 .57 .

5

Fr 9

50 6

1. 3 an 8 938 .

2 ce 6 2 0 26. 0 0 . 0 0 0 0 0 0 0 0 0 0

63 .

0.

8

0

790 .

1 1 84.10 0 0 .. 0 0 0 0 0 0 0 0 0 0 69 Fr 0.

1. 4 9 an 1 0 ce 0

1

2

Sp 5

1. 5 85 ai 2 5

0 n 1

0.

8

2

1. rows × 6459 columns

# 6.Find the outliers and replace the outliers

In [147]:

dataset**=** [11,10,12,14,12,15,14,13,15,102,12,14,17,19,107,

10,13,12,14,12,108,12,11,14,13,15,10,15,12,10,14,13,15,10]

## Detecting outlier using Z score Using Z score

In [148]: outliers**=**[] **def** detect\_outliers(data):

threshold**=**3 mean **=** np**.**mean(data)

std **=**np**.**std(data)

**for** i **in** data:

z\_score**=** (i **-** mean)**/**std

**if** np**.**abs(z\_score) **>** threshold:

outliers**.**append(y)

**return** outliers

In [151]:

outlier\_pt**=**detect\_outliers(dataset)

In [152]:

outlier\_pt

Out[152]:

[0 101348.88

1. 112542.58
2. 113931.57 3 93826.63

4 79084.10

...

9995 96270.64  9996 101699.77  9997 42085.58

1. 92888.52
2. 38190.78

Name: EstimatedSalary, Length: 10000, dtype: float64, 0 101348.88

1. 112542.58
2. 113931.57 3 93826.63 4 79084.10

...

9995 96270.64  9996 101699.77  9997 42085.58

1. 92888.52
2. 38190.78

Name: EstimatedSalary, Length: 10000, dtype: float64, 0 101348.88

1 112542.58 2 113931.57 3 93826.63

4 79084.10

...

9995 96270.64  9996 101699.77  9997 42085.58

1. 92888.52
2. 38190.78

Name: EstimatedSalary, Length: 10000, dtype: float64]

In [153]:

*## Perform all the steps of IQR*

sorted(dataset)

Out[153]:

[10,

10,

10,

|  |  |
| --- | --- |
| 10,  10,  11,  11,  12,  12,  12,  12,  12,  12,  12,  13,  13,  13,  13,  14,  14,  14,  14,  14,  14,  15,  15,  15,  15,  15,  17,  19,  102,  107,  108] | In [155]:    In [156]:    In [157]:        In [159]:        In [160]: |
| quantile1, quantile3**=** np**.**percentile(dataset,[25,75])  print(quantile1,quantile3)  12.0 15.0  *## Find the IQR*  iqr\_value**=**quantile3**-**quantile1 print(iqr\_value)  3.0  *## Find the lower bound value and the higher bound value*  lower\_bound\_val **=** quantile1 **-**(1.5 **\*** iqr\_value) upper\_bound\_val **=** quantile3 **+**(1.5 **\*** iqr\_value)  print(lower\_bound\_val,upper\_bound\_val) |

7.5 19.5

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **7.Check for Categorical columns** | | |  | In [161]:    In [162]:    Out[162]: |
| **and perform encoding** df**=**pd**.**read\_csv('/content/Churn\_Modelling.csv')  df**.**head() |  |  |
| **Row Cust Sur Cred Geo Ge A Te Bal**  **Num omer na itSco grap nd g nu anc ber Id me re hy er e re e** | **NumO fProdu cts** | **Has CrC ard** | **IsActiv eMemb er** | **Estima Ex tedSala ite ry d** |

1. 1 15634602 Hargra 619 France maFe 42  2 0.00 1 1 1 101348.88 1

ve le

1564 Spai Fe 4 838 112542

1. 2 7311 Hill 608 n male 1 1 07.86 1 0 1 .58 0

1561 Oni Fran Fe 4 159 113931

1. 3 9304 o 502 ce male 2 8 660.80 3 1 0 .57 1

1. 4 13541570 Boni 699 France leFe 93 1 0.00 2 0 0 93826.63 0 ma
2. 5 78881573 Mitl Spain leFe 43 12582 79084.10  0

chel850ma2510.111

In [163]:

df\_numeric **=** df[['RowNumber', 'CustomerId', 'CreditScore', 'Age', 'Tenure',

'Balance',

'NumOfProducts','HasCrCard','IsActiveMember','EstimatedSalary','Exited']] df\_categorical **=** df[['Surname', 'Geography', 'Gender']]

In [164]: df\_numeric**.**head()

Out[164]:

**RowNu Custo Credit A Ten Balan NumOfPr HasCr IsActiveM Estimated Exi mber merId Score ge ure ce oducts Card ember Salary ted**

156346

1. 1 02 619 42 2 0.00 1 1 1 101348.88 1 **RowNu Custo Credit A Ten Balan NumOfPr HasCr IsActiveM Estimated Exi mber merId Score ge ure ce oducts Card ember Salary ted**

156473 83807

1. 2 11 608 41 1 .86 1 0 1 112542.58 0

1. 3 15619304 502 42 8 159660.80 3 1 0 113931.57 1

157013

1. 4 54 699 39 1 0.00 2 0 0 93826.63 0

1. 5 15737888 850 43 2 125510.82 1 1 1 79084.10 0

|  |  |
| --- | --- |
| df\_categorical**.**head()      **Surname Geography Gender**     1. Hargrave France Female      1. Hill Spain Female      1. Onio France Female      1. Boni France Female      1. Mitchell Spain Female | In [165]:    Out[165]:                      In [166]:          In [167]:      In [168]: |
| print(df['Surname']**.**unique()) print(df['Geography']**.**unique()) print(df['Gender']**.**unique()) |
| ['Hargrave' 'Hill' 'Onio' ... 'Kashiwagi' 'Aldridge' 'Burbidge']  ['France' 'Spain' 'Germany']  ['Female' 'Male']    **from** sklearn.preprocessing **import** LabelEncoder marry\_encoder **=** LabelEncoder()  marry\_encoder**.**fit(df\_categorical['Gender'])  LabelEncoder() |

Out[168]:

In [169]:

marry\_values **=** marry\_encoder**.**transform(df\_categorical['Gender'])

In [170]: print("Before Encoding:", list(df\_categorical['Gender'][**-**10:])) print("After Encoding:", marry\_values[**-**10:]) print("The inverse from the encoding result:", marry\_encoder**.**inverse\_transform(marry\_values[**-**10:]))

Before Encoding: ['Male', 'Female', 'Male', 'Male', 'Female', 'Male', 'Male

', 'Female', 'Male', 'Female']

After Encoding: [1 0 1 1 0 1 1 0 1 0]

The inverse from the encoding result: ['Male' 'Female' 'Male' 'Male' 'Femal e' 'Male' 'Male' 'Female' 'Male'

'Female']

In [171]:

residence\_encoder **=** LabelEncoder() residence\_values **=** residence\_encoder**.**fit\_transform(df\_categorical['Geography'])

print("Before Encoding:", list(df\_categorical['Geography'][:5])) print("After Encoding:", residence\_values[:5]) print("The inverse from the encoding result:", residence\_encoder**.**inverse\_transform(residence\_values[:5]))

Before Encoding: ['France', 'Spain', 'France', 'France', 'Spain']

After Encoding: [0 2 0 0 2]

The inverse from the encoding result: ['France' 'Spain' 'France' 'France' ' Spain']

In [172]: **from** sklearn.preprocessing **import** OneHotEncoder

gender\_encoder **=** OneHotEncoder()

**from** sklearn.preprocessing **import** OneHotEncoder In [174]:

**import** numpy **as** np

gender\_encoder **=** OneHotEncoder() gender\_reshaped **=** np**.**array(df\_categorical['Gender'])**.**reshape(**-**1, 1) gender\_values **=** gender\_encoder**.**fit\_transform(gender\_reshaped)

print(df\_categorical['Gender'][:5]) print() print(gender\_values**.**toarray()[:5]) print()

print(gender\_encoder**.**inverse\_transform(gender\_values)[:5])

1. Female
2. Female
3. Female
4. Female
5. Female

Name: Gender, dtype: object

[[1. 0.]

[1. 0.]

[1. 0.]

[1. 0.]

[1. 0.]]

|  |  |
| --- | --- |
| [['Female']  ['Female']  ['Female']  ['Female']  ['Female']]    smoke\_encoder **=** OneHotEncoder() | In [175]: |

smoke\_reshaped **=** np**.**array(df\_categorical['Surname'])**.**reshape(**-**1, 1)

smoke\_values **=** smoke\_encoder**.**fit\_transform(smoke\_reshaped)

print(df\_categorical['Surname'][:5])

print()

print(smoke\_values**.**toarray()[:5]) print()

print(smoke\_encoder**.**inverse\_transform(smoke\_values)[:5])

1. Hargrave
2. Hill
3. Onio
4. Boni
5. Mitchell

Name: Surname, dtype: object

[[0. 0. 0. ... 0. 0. 0.]

[0. 0. 0. ... 0. 0. 0.]

[0. 0. 0. ... 0. 0. 0.]

[0. 0. 0. ... 0. 0. 0.]

[0. 0. 0. ... 0. 0. 0.]]

[['Hargrave']

['Hill']

['Onio']

['Boni']

['Mitchell']]

work\_encoder **=** OneHotEncoder() In [176]:

work\_reshaped **=** np**.**array(df\_categorical['Geography'])**.**reshape(**-**1, 1) work\_values **=** work\_encoder**.**fit\_transform(work\_reshaped)

print(df\_categorical['Geography'][:5]) print() print(work\_values**.**toarray()[:5]) print() print(work\_encoder**.**inverse\_transform(work\_values)[:5])

0 France 1 Spain

1. France
2. France
3. Spain

Name: Geography, dtype: object

[[1. 0. 0.]

1. 0. 1.]
2. 0. 0.]

[1. 0. 0.]

[0. 0. 1.]]

[['France'] ['Spain']

['France']

['France']

['Spain']]

In [178]:

df\_categorical\_encoded **=** pd**.**get\_dummies(df\_categorical, drop\_first**=True**) df\_categorical\_encoded**.**head()

Out[178]:

**S S S S S**

**S S Su u**

**u u S S S Su S Su Su rnaSu ur Sur naure\_m rne\_ma mnar ur ur Geog eoG Ge r r ur ur u rn ur rn rn**

**n n na na r a na a a me n n Z Z e n n ra gr n a u ub \_**

**m m e\_ e\_ a e\_ e\_ e\_ \_A m**

1. **a m m n m m m me owabr\_Aitz  ... e\_mZ mae barev areva Zue ma\_e e\_mZa Gephy\_r y\_pha d\_re**

**e e\_ A A m A A A br**

**\_ A b b e be br br a ot \_ Z u m S M**

**A b d d \_ rn a a m Z**

1. **b ul ul A at m m ov vao ox**   **uy yev any aip lea**

**bi ot la ov b hy ov ov ic**

**e t h el a h v ve  a n**

.

**0** 0 0 0 0 0 0 0 0 0 0 . 0 0 0 0 0 0 0 0 0 0

.

.

0 0 0 0 0 0 0 0 1 0 **1** 0 0 0 0 0 0 0 0 0 0 .

.

.

0 0 0 0 0 0 0 0 0 0 **2** 0 0 0 0 0 0 0 0 0 0 .

.

.

0 0 0 0 0 0 0 0 0 0 **3** 0 0 0 0 0 0 0 0 0 0 .

.

.

0 0 0 0 0 0 0 0 1 0 **4** 0 0 0 0 0 0 0 0 0 0 .

.

5 rows × 2934 columns

In [179]: df\_new **=** pd**.**concat([df\_numeric, df\_categorical\_encoded], axis**=**1) df\_new**.**head()

Out[179]:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **0**            **1**      **2** | **R o w N u m b**  **er**      1      2  3 | **C u st o m e rI d**    1  5  6  3  4  6  0  2      1  5    6 | | **r e**  **di t**  **S c o r e**  6  1  9  6  0 8  5  0  2 |
| 4  7  3 1  1  1  5  6  1  9  3  0  4  1  5 |  |

**C**

**Es**  **S** **G**

**H Is Su u Su Su S Su Su Ge Ge**

**T B N a Activ ti rna nr rna amrn naur rna rna ograp ogra ne a**

**u s Me m . e\_m ma me\_Z Zue\_ e\_m e\_m e\_m \_Ghy phy\_ d A e l g n a m C eat . Zoto e\_Z ubar revba ueZ uyZ Zuye maer Spai er e u n Of r m ed .** **va ox** **ev** **a** **v** **ev** **va** **ny** **n \_**

**Pr C ber Sa**

**r c M od a la**

**e e**

**uc r ry al ts d e**

10

0

4 . 1 1 1 13 . 0 0 0 0 0 0 0 0 0 0

2 2 0 48 .

.8 .

0

8

8

### 3 11

4 80 1 0 1 2542 .. 0 0 0 0 0 0 0 0 1 0

1

1 7. .58 .

8 6

1

* 1. 9 11
  2. 39 .

8 6 3 1 0 31 . 0 0 0 0 0 0 0 0 0 0

0 .5 . . 7

8

0

4

2

70 6 3 0. 8293 .

**3** 4 1 9 9 1 0 2 0 0 6. . 0 0 0 0 0 0 0 0 0 0

3 9 0 63 .

5

4

1

1

5 25 79 .

7 8 5 1 1 1 08 . 0 0 0 0 0 0 0 0 1 0

* 1. 4 2 1 4. .

5

* 1. 5 7 0 3 0 10

8

.

8 8

8 2

5 rows × 2945 columns

|  |  |  |  |
| --- | --- | --- | --- |
| **8.Split the data into dependent and independent variables.**  df**=**pd**.**read\_csv('/content/Churn\_Modelling.csv')  print(df["Balance"]**.**min()) print(df["Balance"]**.**max()) print(df["Balance"]**.**mean())  0.0  250898.09 76485.889288 print(df**.**count(0)) | | | In [180]:    In [182]:          In [183]:                        In [184]:    In [185]:    In [187]:              In [271]: |
| RowNumber  CustomerId  Surname  CreditScore  Geography  Gender Age  Tenure  Balance  NumOfProducts  HasCrCard  IsActiveMember  EstimatedSalary Exited dtype: int64    print(df**.**shape) (10000, 14)  print(df**.**size)  140000 | 10000  10000  10000  10000  10000  10000  10000  10000  10000  10000  10000  10000  10000  10000 | |
| X **=** df**.**iloc[:, :**-**1]**.**values print(X)  [[1 15634602 'Hargrave' ... 1 1 101348.88]   1. 15647311 'Hill' ... 0 1 112542.58] 2. 15619304 'Onio' ... 1 0 113931.57] ... 3. 15584532 'Liu' ... 0 1 42085.58] 4. 15682355 'Sabbatini' ... 1 0 92888.52] 5. 15628319 'Walker' ... 1 0 38190.78]]   Y **=** df**.**iloc[:, **-**1]**.**values print(Y)  [1 0 1 ... 1 1 0] | | |
| **9.Scale the independent variables**  df **=** pd**.**read\_csv('/content/Churn\_Modelling.csv')  x **=** df[['Age', 'Tenure']]**.**values y **=** df['Gender']**.**values fig, ax **=** plt**.**subplots(ncols**=**2, figsize**=**(12, 4))  ax[0]**.**scatter(x[:,0], y) ax[1]**.**scatter(x[:,1], y)  plt**.**show() | | In [215]: | |
| fig, ax **=** plt**.**subplots(figsize**=**(12, 4))  ax**.**scatter(x[:,0], y) ax**.**scatter(x[:,1], y)  <matplotlib.collections.PathCollection at 0x7f9a8a854ad0>    fig, ax **=** plt**.**subplots(figsize**=**(12, 4))  ax**.**hist(x[:,0]) ax**.**hist(x[:,1]) | | In [216]:        Out[216]:    In [217]: | |

Out[217]:

(array([ 413., 1035., 1048., 1009., 989., 1012., 967., 1028., 1025.,

1474.]), array([ 0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10.]),

<a list of 10 Patch objects>)

|  |  |  |
| --- | --- | --- |
| **from** sklearn.preprocessing **import** StandardScaler **from** sklearn.preprocessing **import** MinMaxScaler fig, ax **=** plt**.**subplots(figsize**=**(12, 4))  scaler **=** StandardScaler() x\_std **=** scaler**.**fit\_transform(x)  ax**.**hist(x\_std[:,0]) ax**.**hist(x\_std[:,1]) |  | In [220]: |
| Out[220]: | |

(array([ 413., 1035., 1048., 1009., 2001., 0., 1995., 0., 1025.,

1474.]), array([-1.73331549, -1.38753759, -1.04175968, -0.69598177, -0.35020386,

-0.00442596, 0.34135195, 0.68712986, 1.03290776, 1.37868567, 1.72446358]),

<a list of 10 Patch objects>)

In [219]:

|  |  |
| --- | --- |
| fig, ax **=** plt**.**subplots(figsize**=**(12, 4)) |  |
| scaler **=** StandardScaler() x\_std **=** scaler**.**fit\_transform(x)  ax**.**scatter(x\_std[:,0], y) ax**.**scatter(x\_std[:,1], y)  <matplotlib.collections.PathCollection at 0x7f9a8a2fde50>    fig, ax **=** plt**.**subplots(figsize**=**(12, 4))  scaler **=** MinMaxScaler()  x\_minmax **=** scaler**.**fit\_transform(x)  ax**.**hist(x\_minmax [:,0]) ax**.**hist(x\_minmax [:,1]) | Out[219]:    In [221]: |

Out[221]:

(array([ 413., 1035., 1048., 1009., 989., 1012., 967., 1028., 1025.,

1474.]), array([0. , 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1. ]),

<a list of 10 Patch objects>)

In [222]:

|  |  |
| --- | --- |
| fig, ax **=** plt**.**subplots(figsize**=**(12, 4)) |  |
| scaler **=** MinMaxScaler()  x\_minmax **=** scaler**.**fit\_transform(x)  ax**.**scatter(x\_minmax [:,0], y) ax**.**scatter(x\_minmax [:,1], y)  <matplotlib.collections.PathCollection at 0x7f9a8a0cae10>    fig, ax **=** plt**.**subplots(figsize**=**(12, 4))  scaler **=** MinMaxScaler() x\_minmax **=** scaler**.**fit\_transform(x) ax**.**scatter(x\_minmax [:,0], y)  <matplotlib.collections.PathCollection at 0x7f9a8a0caf10>    fig, ax **=** plt**.**subplots(figsize**=**(12, 4))  scaler **=** MinMaxScaler() x\_minmax **=** scaler**.**fit\_transform(x) | Out[222]:    In [223]:          Out[223]:    In [224]: |

ax**.**hist(x\_minmax [:,0])

Out[224]:

(array([ 611., 2179., 3629., 1871., 910., 441., 208., 127., 20.,

4.]),

array([0. , 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1. ]),

<a list of 10 Patch objects>)

In [227]:

**from** sklearn.model\_selection **import** train\_test\_split **from** sklearn.pipeline **import** Pipeline

**from** sklearn.linear\_model **import** SGDRegressor **from** sklearn.preprocessing **import** StandardScaler **from**

sklearn.preprocessing **import** MinMaxScaler **from** sklearn.metrics **import** mean\_absolute\_error **import** sklearn.metrics **as** metrics

**import** pandas **as** pd **import** numpy **as** np **import** matplotlib.pyplot **as** plt

*# Import Data*

df **=** pd**.**read\_csv('/content/Churn\_Modelling.csv') x **=** df[['Age', 'Tenure']]**.**values y **=** df['Balance']**.**values

*# Split into a training and testing set*

X\_train, X\_test, Y\_train, Y\_test **=** train\_test\_split(x, y)

*# Define the pipeline for scaling and model fitting*

pipeline **=** Pipeline([

("MinMax Scaling", MinMaxScaler()),

("SGD Regression", SGDRegressor())

])

*# Scale the data and fit the model*

pipeline**.**fit(X\_train, Y\_train)

*# Evaluate the model*

Y\_pred **=** pipeline**.**predict(X\_test)

print('Mean Absolute Error: ', mean\_absolute\_error(Y\_pred, Y\_test))

print('Score', pipeline**.**score(X\_test, Y\_test))

Mean Absolute Error: 57120.533393590835

Score 0.0004207814312172653

|  |  |
| --- | --- |
| **10.Split the data into training and testing**  dataset **=** pd**.**read\_csv('/content/Churn\_Modelling.csv') print(dataset) | In [267]: |
| RowNumber CustomerId Surname CreditScore Geography Gender Age  \   1. 1 15634602 Hargrave 619 France Female 42 2. 2 15647311 Hill 608 Spain Female 41 3. 3 15619304 Onio 502 France Female 42 4. 4 15701354 Boni 699 France Female 39 | |

4 5 15737888 Mitchell 850 Spain Female 43

... ... ... ... ... ... ... ...

1. 9996 15606229 Obijiaku 771 France Male 39
2. 9997 15569892 Johnstone 516 France Male 35
3. 9998 15584532 Liu 709 France Female 36
4. 9999 15682355 Sabbatini 772 Germany Male 42
5. 10000 15628319 Walker 792 France Female 28

Tenure Balance NumOfProducts HasCrCard IsActiveMember \

1. 2 0.00 1 1 1
2. 1 83807.86 1 0 1 2 8 159660.80 3 1 0
3. 1 0.00 2 0 0
4. 2 125510.82 1 1 1 ... ... ... ... ... ...
5. 5 0.00 2 1 0
6. 10 57369.61 1 1 1
7. 7 0.00 1 0 1
8. 3 75075.31 2 1 0
9. 4 130142.79 1 1 0

EstimatedSalary Exited

1. 101348.88 1
2. 112542.58 0
3. 113931.57 1
4. 93826.63 0
5. 79084.10 0 ... ... ...

9995 96270.64 0 9996 101699.77 0

1. 42085.58 1
2. 92888.52 1
3. 38190.78 0

[10000 rows x 14 columns]

In [287]:

dataset**.**drop(["HasCrCard"],axis**=**1,inplace**=True**)

print(dataset**.**shape)*#no. of rows and colume* In [288]:

print(dataset**.**head(10))

(10000, 7)

CustomerId CreditScore Age Tenure Balance IsActiveMember \

1. 15634602 619 42 2 0.00 1
2. 15647311 608 41 1 83807.86 1
3. 15619304 502 42 8 159660.80 0 3 15701354 699 39 1 0.00 0 4 15737888 850 43 2 125510.82 1

5 15574012 645 44 8 113755.78 0 6 15592531 822 50 7 0.00 1

7 15656148 376 29 4 115046.74 0 8 15792365 501 44 4 142051.07 1

9 15592389 684 27 2 134603.88 1

EstimatedSalary

0 101348.88 1 112542.58 2 113931.57

1. 93826.63
2. 79084.10
3. 149756.71
4. 10062.80
5. 119346.88
6. 74940.50
7. 71725.73

In [289]:

X**=**dataset**.**iloc[:,:**-**1]**.**values

X

Out[289]: array([[1.5634602e+07, 6.1900000e+02, 4.2000000e+01, 2.0000000e+00,

0.0000000e+00, 1.0000000e+00],

[1.5647311e+07, 6.0800000e+02, 4.1000000e+01, 1.0000000e+00,

8.3807860e+04, 1.0000000e+00],

[1.5619304e+07, 5.0200000e+02, 4.2000000e+01, 8.0000000e+00, 1.5966080e+05, 0.0000000e+00],

...,

[1.5584532e+07, 7.0900000e+02, 3.6000000e+01, 7.0000000e+00, 0.0000000e+00, 1.0000000e+00],

[1.5682355e+07, 7.7200000e+02, 4.2000000e+01, 3.0000000e+00,

7.5075310e+04, 0.0000000e+00],

[1.5628319e+07, 7.9200000e+02, 2.8000000e+01, 4.0000000e+00,

1.3014279e+05, 0.0000000e+00]])

Y**=**dataset**.**iloc[:,**-**1]**.**values In [290]:

Y

array([101348.88, 112542.58, 113931.57, ..., 42085.58, 92888.52,

Out[290]: 38190.78])

**from** sklearn.model\_selection **import** train\_test\_split

In [291]:

X\_train,X\_test,Y\_train,Y\_test **=** train\_test\_split( X, Y, test\_size **=** 0.25, random\_state **=** 0 )

In [306]:

**from** sklearn.preprocessing **import** StandardScaler

sc**=**StandardScaler()

X\_train **=** sc**.**fit\_transform(X\_train) X\_test **=** sc**.**transform(X\_test) print(X\_train)

[[-1.34333028 -0.73550706 0.01526571 0.00886037 0.67316003 -1.03446007]

[ 1.55832963 1.02442719 -0.65260917 0.00886037 -1.20772417 -1.03446007]

[-0.65515619 0.80829492 -0.46178778 1.39329338 -0.35693706 0.96668786]

...

[-1.63542994 0.90092304 -0.36637708 0.00886037 1.36657199 -1.03446007]

[-0.38540456 -0.62229491 -0.08014499 1.39329338 -1.20772417 0.96668786] [-1.37829524 -0.28265848 0.87396199 -1.37557264 0.51741687 -1.03446007]]

In [305]:

print(X\_test)

[[-1.05852196 -0.55025082 -0.36637708 1.04718513 0.88494297 0.96668786]

[-0.51554728 -1.31185979 0.11067641 -1.02946438 0.43586703 -1.03446007]

[-0.8058485 0.57157862 0.3014978 1.04718513 0.31486378 0.96668786]

...

[ 0.25326371 1.95070838 0.01526571 -1.37557264 0.30819395 -1.03446007]

[-0.17836122 0.29369426 -0.08014499 0.70107688 0.55698791 -1.03446007]

[ 0.40190663 0.870047 -0.74801987 -0.68335613 0.7006957 -1.03446007]]