{

"cells": [

{

"cell\_type": "markdown",

"id": "c0ec21ca",

"metadata": {},

"source": [

"# Assignment -2\n",

"# Data Visualization and Pre-processing"

]

},

{

"cell\_type": "markdown",

"id": "6959bc02",

"metadata": {},

"source": [

"# A - Load the dataset"

]

},

{

"cell\_type": "code",

"execution\_count": 41,

"id": "a3f44fb4",

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

" RowNumber CustomerId Surname CreditScore Geography Gender Age \\\n",

"0 1 15634602 Hargrave 619 France Female 42 \n",

"1 2 15647311 Hill 608 Spain Female 41 \n",

"2 3 15619304 Onio 502 France Female 42 \n",

"3 4 15701354 Boni 699 France Female 39 \n",

"4 5 15737888 Mitchell 850 Spain Female 43 \n",

"... ... ... ... ... ... ... ... \n",

"9995 9996 15606229 Obijiaku 771 France Male 39 \n",

"9996 9997 15569892 Johnstone 516 France Male 35 \n",

"9997 9998 15584532 Liu 709 France Female 36 \n",

"9998 9999 15682355 Sabbatini 772 Germany Male 42 \n",

"9999 10000 15628319 Walker 792 France Female 28 \n",

"\n",

" Tenure Balance NumOfProducts HasCrCard IsActiveMember \\\n",

"0 2 0.00 1 1 1 \n",

"1 1 83807.86 1 0 1 \n",

"2 8 159660.80 3 1 0 \n",

"3 1 0.00 2 0 0 \n",

"4 2 125510.82 1 1 1 \n",

"... ... ... ... ... ... \n",

"9995 5 0.00 2 1 0 \n",

"9996 10 57369.61 1 1 1 \n",

"9997 7 0.00 1 0 1 \n",

"9998 3 75075.31 2 1 0 \n",

"9999 4 130142.79 1 1 0 \n",

"\n",

" EstimatedSalary Exited \n",

"0 101348.88 1 \n",

"1 112542.58 0 \n",

"2 113931.57 1 \n",

"3 93826.63 0 \n",

"4 79084.10 0 \n",

"... ... ... \n",

"9995 96270.64 0 \n",

"9996 101699.77 0 \n",

"9997 42085.58 1 \n",

"9998 92888.52 1 \n",

"9999 38190.78 0 \n",

"\n",

"[10000 rows x 14 columns]\n"

]

}

],

"source": [

"import pandas as pd\n",

"df=pd.read\_csv(\"Churn\_Modelling.csv\") # import dataset\n",

"print(df)"

]

},

{

"cell\_type": "markdown",

"id": "956f95aa",

"metadata": {},

"source": [

"# B - Perform Below Visualizations.\n",

"# 1. Univarient Analysis \n",

"## There are three ways to perform univarient analysis \n",

"## i) Summary statistics\n"

]

},

{

"cell\_type": "code",

"execution\_count": 42,

"id": "04dca97e",

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"mean value of CreditScore is 650.5288\n",

"median value of CreditScore is 652.0\n",

"Standard deviation of CreditScore is 96.65329873613061\n"

]

}

],

"source": [

"# Summary statistics\n",

"\n",

"import pandas as pd\n",

"df=pd.read\_csv(\"Churn\_Modelling.csv\")\n",

"\n",

"#mean of CreditScore\n",

"M=df['CreditScore'].mean()\n",

"\n",

"#median of CreditScore\n",

"Me=df['CreditScore'].median()\n",

"\n",

"# standard deviation of CreditScore\n",

"std = df['CreditScore'].std()\n",

"\n",

"print(\"mean value of CreditScore is {}\".format(M))\n",

"print(\"median value of CreditScore is {}\".format(Me))\n",

"print(\"Standard deviation of CreditScore is {}\".format(std))"

]

},

{

"cell\_type": "markdown",

"id": "9850c66e",

"metadata": {},

"source": [

"## ii) Frequency table"

]

},

{

"cell\_type": "code",

"execution\_count": 43,

"id": "9f00e0ae",

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"Frequency table for Age is given below\n",

"37 478\n",

"38 477\n",

"35 474\n",

"36 456\n",

"34 447\n",

" ... \n",

"92 2\n",

"82 1\n",

"88 1\n",

"85 1\n",

"83 1\n",

"Name: Age, Length: 70, dtype: int64\n"

]

}

],

"source": [

"#Frequency table\n",

"import pandas as pd\n",

"df=pd.read\_csv(\"Churn\_Modelling.csv\")\n",

"\n",

"#frequency table for age\n",

"ft=df['Age'].value\_counts()\n",

"\n",

"print(\"Frequency table for Age is given below\")\n",

"print(\"{}\".format(ft))"

]

},

{

"cell\_type": "markdown",

"id": "542a20bf",

"metadata": {},

"source": [

"\n",

"## iii) Charts"

]

},

{

"cell\_type": "code",

"execution\_count": 44,

"id": "a11f220b",

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

" RowNumber CustomerId Surname CreditScore Geography Gender Age \\\n",

"0 1 15634602 Hargrave 619 France Female 42 \n",

"1 2 15647311 Hill 608 Spain Female 41 \n",

"2 3 15619304 Onio 502 France Female 42 \n",

"3 4 15701354 Boni 699 France Female 39 \n",

"4 5 15737888 Mitchell 850 Spain Female 43 \n",

"\n",

" Tenure Balance NumOfProducts HasCrCard IsActiveMember \\\n",

"0 2 0.00 1 1 1 \n",

"1 1 83807.86 1 0 1 \n",

"2 8 159660.80 3 1 0 \n",

"3 1 0.00 2 0 0 \n",

"4 2 125510.82 1 1 1 \n",

"\n",

" EstimatedSalary Exited \n",

"0 101348.88 1 \n",

"1 112542.58 0 \n",

"2 113931.57 1 \n",

"3 93826.63 0 \n",

"4 79084.10 0 \n"

]

},

{

"data": {

"text/plain": [

"Text(0.5, 1.0, 'Box plot')"

]

},

"execution\_count": 44,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"#Chart\n",

"\n",

"import matplotlib.pyplot as plt\n",

"dfs = df.head() # print first five table from top\n",

"print(dfs) \n",

"\n",

"#box plot for Balance column\n",

"\n",

"dfs.boxplot(column=\"Balance\",grid=False,color=\"red\")\n",

"plt.title('Box plot')"

]

},

{

"cell\_type": "code",

"execution\_count": 45,

"id": "180502f9",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"Text(0.5, 1.0, 'Histogram')"

]

},

"execution\_count": 45,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"# Histogram for Credit Score\n",

"\n",

"df.hist(column=\"CreditScore\" ,grid=True, edgecolor ='black', color ='red')\n",

"plt.title('Histogram')"

]

},

{

"cell\_type": "code",

"execution\_count": 46,

"id": "74e4bc8a",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"Text(0.5, 1.0, 'Density Curve')"

]

},

"execution\_count": 46,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"# Density curve\n",

"\n",

"import seaborn as sns #statistical data visualization\n",

"\n",

"sns.kdeplot(df['CreditScore'])\n",

"plt.title('Density Curve')"

]

},

{

"cell\_type": "markdown",

"id": "939f93af",

"metadata": {},

"source": [

"# 2. Bi - Variate Analysis\n",

"## There are three common ways to perform bivariate analysis:\n",

"## i. Scatterplots"

]

},

{

"cell\_type": "code",

"execution\_count": 47,

"id": "f6bf4bc1",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"Text(0, 0.5, 'Balance')"

]

},

"execution\_count": 47,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"import matplotlib.pyplot as plt # library for charts\n",

"\n",

"dfs1 = df.head(20)\n",

"plt.scatter(dfs1.CreditScore,dfs1.Balance)\n",

"plt.title('Scatterplots-- Banking')\n",

"plt.xlabel(\"CreditScore\")\n",

"plt.ylabel(\"Balance\")\n",

"\n"

]

},

{

"cell\_type": "markdown",

"id": "0f1d6a61",

"metadata": {},

"source": [

"## ii.Correlation Coefficient"

]

},

{

"cell\_type": "code",

"execution\_count": 48,

"id": "9c2ca9a8",

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>RowNumber</th>\n",

" <th>CustomerId</th>\n",

" <th>CreditScore</th>\n",

" <th>Age</th>\n",

" <th>Tenure</th>\n",

" <th>Balance</th>\n",

" <th>NumOfProducts</th>\n",

" <th>HasCrCard</th>\n",

" <th>IsActiveMember</th>\n",

" <th>EstimatedSalary</th>\n",

" <th>Exited</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>RowNumber</th>\n",

" <td>1.000000</td>\n",

" <td>0.004202</td>\n",

" <td>0.005840</td>\n",

" <td>0.000783</td>\n",

" <td>-0.006495</td>\n",

" <td>-0.009067</td>\n",

" <td>0.007246</td>\n",

" <td>0.000599</td>\n",

" <td>0.012044</td>\n",

" <td>-0.005988</td>\n",

" <td>-0.016571</td>\n",

" </tr>\n",

" <tr>\n",

" <th>CustomerId</th>\n",

" <td>0.004202</td>\n",

" <td>1.000000</td>\n",

" <td>0.005308</td>\n",

" <td>0.009497</td>\n",

" <td>-0.014883</td>\n",

" <td>-0.012419</td>\n",

" <td>0.016972</td>\n",

" <td>-0.014025</td>\n",

" <td>0.001665</td>\n",

" <td>0.015271</td>\n",

" <td>-0.006248</td>\n",

" </tr>\n",

" <tr>\n",

" <th>CreditScore</th>\n",

" <td>0.005840</td>\n",

" <td>0.005308</td>\n",

" <td>1.000000</td>\n",

" <td>-0.003965</td>\n",

" <td>0.000842</td>\n",

" <td>0.006268</td>\n",

" <td>0.012238</td>\n",

" <td>-0.005458</td>\n",

" <td>0.025651</td>\n",

" <td>-0.001384</td>\n",

" <td>-0.027094</td>\n",

" </tr>\n",

" <tr>\n",

" <th>Age</th>\n",

" <td>0.000783</td>\n",

" <td>0.009497</td>\n",

" <td>-0.003965</td>\n",

" <td>1.000000</td>\n",

" <td>-0.009997</td>\n",

" <td>0.028308</td>\n",

" <td>-0.030680</td>\n",

" <td>-0.011721</td>\n",

" <td>0.085472</td>\n",

" <td>-0.007201</td>\n",

" <td>0.285323</td>\n",

" </tr>\n",

" <tr>\n",

" <th>Tenure</th>\n",

" <td>-0.006495</td>\n",

" <td>-0.014883</td>\n",

" <td>0.000842</td>\n",

" <td>-0.009997</td>\n",

" <td>1.000000</td>\n",

" <td>-0.012254</td>\n",

" <td>0.013444</td>\n",

" <td>0.022583</td>\n",

" <td>-0.028362</td>\n",

" <td>0.007784</td>\n",

" <td>-0.014001</td>\n",

" </tr>\n",

" <tr>\n",

" <th>Balance</th>\n",

" <td>-0.009067</td>\n",

" <td>-0.012419</td>\n",

" <td>0.006268</td>\n",

" <td>0.028308</td>\n",

" <td>-0.012254</td>\n",

" <td>1.000000</td>\n",

" <td>-0.304180</td>\n",

" <td>-0.014858</td>\n",

" <td>-0.010084</td>\n",

" <td>0.012797</td>\n",

" <td>0.118533</td>\n",

" </tr>\n",

" <tr>\n",

" <th>NumOfProducts</th>\n",

" <td>0.007246</td>\n",

" <td>0.016972</td>\n",

" <td>0.012238</td>\n",

" <td>-0.030680</td>\n",

" <td>0.013444</td>\n",

" <td>-0.304180</td>\n",

" <td>1.000000</td>\n",

" <td>0.003183</td>\n",

" <td>0.009612</td>\n",

" <td>0.014204</td>\n",

" <td>-0.047820</td>\n",

" </tr>\n",

" <tr>\n",

" <th>HasCrCard</th>\n",

" <td>0.000599</td>\n",

" <td>-0.014025</td>\n",

" <td>-0.005458</td>\n",

" <td>-0.011721</td>\n",

" <td>0.022583</td>\n",

" <td>-0.014858</td>\n",

" <td>0.003183</td>\n",

" <td>1.000000</td>\n",

" <td>-0.011866</td>\n",

" <td>-0.009933</td>\n",

" <td>-0.007138</td>\n",

" </tr>\n",

" <tr>\n",

" <th>IsActiveMember</th>\n",

" <td>0.012044</td>\n",

" <td>0.001665</td>\n",

" <td>0.025651</td>\n",

" <td>0.085472</td>\n",

" <td>-0.028362</td>\n",

" <td>-0.010084</td>\n",

" <td>0.009612</td>\n",

" <td>-0.011866</td>\n",

" <td>1.000000</td>\n",

" <td>-0.011421</td>\n",

" <td>-0.156128</td>\n",

" </tr>\n",

" <tr>\n",

" <th>EstimatedSalary</th>\n",

" <td>-0.005988</td>\n",

" <td>0.015271</td>\n",

" <td>-0.001384</td>\n",

" <td>-0.007201</td>\n",

" <td>0.007784</td>\n",

" <td>0.012797</td>\n",

" <td>0.014204</td>\n",

" <td>-0.009933</td>\n",

" <td>-0.011421</td>\n",

" <td>1.000000</td>\n",

" <td>0.012097</td>\n",

" </tr>\n",

" <tr>\n",

" <th>Exited</th>\n",

" <td>-0.016571</td>\n",

" <td>-0.006248</td>\n",

" <td>-0.027094</td>\n",

" <td>0.285323</td>\n",

" <td>-0.014001</td>\n",

" <td>0.118533</td>\n",

" <td>-0.047820</td>\n",

" <td>-0.007138</td>\n",

" <td>-0.156128</td>\n",

" <td>0.012097</td>\n",

" <td>1.000000</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"</div>"

],

"text/plain": [

" RowNumber CustomerId CreditScore Age Tenure \\\n",

"RowNumber 1.000000 0.004202 0.005840 0.000783 -0.006495 \n",

"CustomerId 0.004202 1.000000 0.005308 0.009497 -0.014883 \n",

"CreditScore 0.005840 0.005308 1.000000 -0.003965 0.000842 \n",

"Age 0.000783 0.009497 -0.003965 1.000000 -0.009997 \n",

"Tenure -0.006495 -0.014883 0.000842 -0.009997 1.000000 \n",

"Balance -0.009067 -0.012419 0.006268 0.028308 -0.012254 \n",

"NumOfProducts 0.007246 0.016972 0.012238 -0.030680 0.013444 \n",

"HasCrCard 0.000599 -0.014025 -0.005458 -0.011721 0.022583 \n",

"IsActiveMember 0.012044 0.001665 0.025651 0.085472 -0.028362 \n",

"EstimatedSalary -0.005988 0.015271 -0.001384 -0.007201 0.007784 \n",

"Exited -0.016571 -0.006248 -0.027094 0.285323 -0.014001 \n",

"\n",

" Balance NumOfProducts HasCrCard IsActiveMember \\\n",

"RowNumber -0.009067 0.007246 0.000599 0.012044 \n",

"CustomerId -0.012419 0.016972 -0.014025 0.001665 \n",

"CreditScore 0.006268 0.012238 -0.005458 0.025651 \n",

"Age 0.028308 -0.030680 -0.011721 0.085472 \n",

"Tenure -0.012254 0.013444 0.022583 -0.028362 \n",

"Balance 1.000000 -0.304180 -0.014858 -0.010084 \n",

"NumOfProducts -0.304180 1.000000 0.003183 0.009612 \n",

"HasCrCard -0.014858 0.003183 1.000000 -0.011866 \n",

"IsActiveMember -0.010084 0.009612 -0.011866 1.000000 \n",

"EstimatedSalary 0.012797 0.014204 -0.009933 -0.011421 \n",

"Exited 0.118533 -0.047820 -0.007138 -0.156128 \n",

"\n",

" EstimatedSalary Exited \n",

"RowNumber -0.005988 -0.016571 \n",

"CustomerId 0.015271 -0.006248 \n",

"CreditScore -0.001384 -0.027094 \n",

"Age -0.007201 0.285323 \n",

"Tenure 0.007784 -0.014001 \n",

"Balance 0.012797 0.118533 \n",

"NumOfProducts 0.014204 -0.047820 \n",

"HasCrCard -0.009933 -0.007138 \n",

"IsActiveMember -0.011421 -0.156128 \n",

"EstimatedSalary 1.000000 0.012097 \n",

"Exited 0.012097 1.000000 "

]

},

"execution\_count": 48,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"df.corr()"

]

},

{

"cell\_type": "markdown",

"id": "9cffe164",

"metadata": {},

"source": [

"## iii. Simple Linear Regression\n"

]

},

{

"cell\_type": "code",

"execution\_count": 49,

"id": "86693fe4",

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

" OLS Regression Results \n",

"==============================================================================\n",

"Dep. Variable: CreditScore R-squared: 0.000\n",

"Model: OLS Adj. R-squared: -0.000\n",

"Method: Least Squares F-statistic: 0.3929\n",

"Date: Sun, 25 Sep 2022 Prob (F-statistic): 0.531\n",

"Time: 13:06:05 Log-Likelihood: -59900.\n",

"No. Observations: 10000 AIC: 1.198e+05\n",

"Df Residuals: 9998 BIC: 1.198e+05\n",

"Df Model: 1 \n",

"Covariance Type: nonrobust \n",

"==============================================================================\n",

" coef std err t P>|t| [0.025 0.975]\n",

"------------------------------------------------------------------------------\n",

"const 649.7861 1.529 424.948 0.000 646.789 652.783\n",

"Balance 9.71e-06 1.55e-05 0.627 0.531 -2.07e-05 4.01e-05\n",

"==============================================================================\n",

"Omnibus: 132.594 Durbin-Watson: 2.014\n",

"Prob(Omnibus): 0.000 Jarque-Bera (JB): 84.114\n",

"Skew: -0.072 Prob(JB): 5.43e-19\n",

"Kurtosis: 2.574 Cond. No. 1.56e+05\n",

"==============================================================================\n",

"\n",

"Notes:\n",

"[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.\n",

"[2] The condition number is large, 1.56e+05. This might indicate that there are\n",

"strong multicollinearity or other numerical problems.\n"

]

}

],

"source": [

"import statsmodels.api as sm\n",

"# response variable\n",

"y = df['CreditScore']\n",

"\n",

"# explanatory variable\n",

"x = df[['Balance']]\n",

"\n",

"#add constant to predictor variables\n",

"x = sm.add\_constant(x)\n",

"\n",

"#fit linear regression model\n",

"model = sm.OLS(y, x).fit()\n",

"\n",

"#view model summary\n",

"print(model.summary())"

]

},

{

"cell\_type": "markdown",

"id": "8141a889",

"metadata": {},

"source": [

"# 3. Multi - Variate Analysis\n",

"\n"

]

},

{

"cell\_type": "code",

"execution\_count": 126,

"id": "a87fc7cf",

"metadata": {},

"outputs": [],

"source": [

"import pandas as pd\n",

"import numpy as np\n",

"import matplotlib.pyplot as plt\n",

"import seaborn as sns\n"

]

},

{

"cell\_type": "markdown",

"id": "4c3405d8",

"metadata": {},

"source": [

"### i. A Matrix Scatterplot\n",

"### ii. A Scatterplot with the Data Points Labelled by their Group\n",

"### iii. A Profile Plot\n",

"### iv. Calculating Summary Statistics for Multivariate Data\n",

"### v. Means and Variances Per Group\n",

"### vi. Between-groups Variance and Within-groups Variance for a Variable\n",

"### vii. Between-groups Covariance and Within-groups Covariance for Two Variables\n",

"### viii. Calculating Correlations for Multivariate Data\n",

"### ix. Standardising Variables\n"

]

},

{

"cell\_type": "code",

"execution\_count": 127,

"id": "235af018",

"metadata": {

"scrolled": false

},

"outputs": [

{

"name": "stderr",

"output\_type": "stream",

"text": [

"/home/lokesh/anaconda3/lib/python3.9/site-packages/seaborn/categorical.py:1296: UserWarning: 80.8% of the points cannot be placed; you may want to decrease the size of the markers or use stripplot.\n",

" warnings.warn(msg, UserWarning)\n",

"/home/lokesh/anaconda3/lib/python3.9/site-packages/seaborn/categorical.py:1296: UserWarning: 62.1% of the points cannot be placed; you may want to decrease the size of the markers or use stripplot.\n",

" warnings.warn(msg, UserWarning)\n",

"/home/lokesh/anaconda3/lib/python3.9/site-packages/seaborn/categorical.py:1296: UserWarning: 62.6% of the points cannot be placed; you may want to decrease the size of the markers or use stripplot.\n",

" warnings.warn(msg, UserWarning)\n"

]

},

{

"name": "stdout",

"output\_type": "stream",

"text": [

"<seaborn.axisgrid.FacetGrid object at 0x7ffb0fd0b1c0>\n"

]

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432.125x360 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"df=sns.catplot(x=\"Geography\",y=\"EstimatedSalary\",hue=\"Gender\",kind=\"swarm\",data=df)\n",

"print(df)"

]

},

{

"cell\_type": "markdown",

"id": "f62a3cc9",

"metadata": {},

"source": [

"## 4. Perform descriptive statistics on the dataset."

]

},

{

"cell\_type": "code",

"execution\_count": 52,

"id": "37b8de90",

"metadata": {},

"outputs": [],

"source": [

"#load data set into ld\n",

"ld= pd.read\_csv(\"Churn\_Modelling.csv\")\n",

"five = ld.head() #for print first five rows"

]

},

{

"cell\_type": "code",

"execution\_count": 53,

"id": "ef2e96d2",

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"<class 'pandas.core.frame.DataFrame'>\n",

"RangeIndex: 10000 entries, 0 to 9999\n",

"Data columns (total 14 columns):\n",

" # Column Non-Null Count Dtype \n",

"--- ------ -------------- ----- \n",

" 0 RowNumber 10000 non-null int64 \n",

" 1 CustomerId 10000 non-null int64 \n",

" 2 Surname 10000 non-null object \n",

" 3 CreditScore 10000 non-null int64 \n",

" 4 Geography 10000 non-null object \n",

" 5 Gender 10000 non-null object \n",

" 6 Age 10000 non-null int64 \n",

" 7 Tenure 10000 non-null int64 \n",

" 8 Balance 10000 non-null float64\n",

" 9 NumOfProducts 10000 non-null int64 \n",

" 10 HasCrCard 10000 non-null int64 \n",

" 11 IsActiveMember 10000 non-null int64 \n",

" 12 EstimatedSalary 10000 non-null float64\n",

" 13 Exited 10000 non-null int64 \n",

"dtypes: float64(2), int64(9), object(3)\n",

"memory usage: 1.1+ MB\n"

]

}

],

"source": [

"# information about used data set\n",

"ld.info()"

]

},

{

"cell\_type": "code",

"execution\_count": 54,

"id": "2b785af5",

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>RowNumber</th>\n",

" <th>CustomerId</th>\n",

" <th>CreditScore</th>\n",

" <th>Age</th>\n",

" <th>Tenure</th>\n",

" <th>Balance</th>\n",

" <th>NumOfProducts</th>\n",

" <th>HasCrCard</th>\n",

" <th>IsActiveMember</th>\n",

" <th>EstimatedSalary</th>\n",

" <th>Exited</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>count</th>\n",

" <td>10000.00000</td>\n",

" <td>1.000000e+04</td>\n",

" <td>10000.000000</td>\n",

" <td>10000.000000</td>\n",

" <td>10000.000000</td>\n",

" <td>10000.000000</td>\n",

" <td>10000.000000</td>\n",

" <td>10000.00000</td>\n",

" <td>10000.000000</td>\n",

" <td>10000.000000</td>\n",

" <td>10000.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>mean</th>\n",

" <td>5000.50000</td>\n",

" <td>1.569094e+07</td>\n",

" <td>650.528800</td>\n",

" <td>38.921800</td>\n",

" <td>5.012800</td>\n",

" <td>76485.889288</td>\n",

" <td>1.530200</td>\n",

" <td>0.70550</td>\n",

" <td>0.515100</td>\n",

" <td>100090.239881</td>\n",

" <td>0.203700</td>\n",

" </tr>\n",

" <tr>\n",

" <th>std</th>\n",

" <td>2886.89568</td>\n",

" <td>7.193619e+04</td>\n",

" <td>96.653299</td>\n",

" <td>10.487806</td>\n",

" <td>2.892174</td>\n",

" <td>62397.405202</td>\n",

" <td>0.581654</td>\n",

" <td>0.45584</td>\n",

" <td>0.499797</td>\n",

" <td>57510.492818</td>\n",

" <td>0.402769</td>\n",

" </tr>\n",

" <tr>\n",

" <th>min</th>\n",

" <td>1.00000</td>\n",

" <td>1.556570e+07</td>\n",

" <td>350.000000</td>\n",

" <td>18.000000</td>\n",

" <td>0.000000</td>\n",

" <td>0.000000</td>\n",

" <td>1.000000</td>\n",

" <td>0.00000</td>\n",

" <td>0.000000</td>\n",

" <td>11.580000</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>25%</th>\n",

" <td>2500.75000</td>\n",

" <td>1.562853e+07</td>\n",

" <td>584.000000</td>\n",

" <td>32.000000</td>\n",

" <td>3.000000</td>\n",

" <td>0.000000</td>\n",

" <td>1.000000</td>\n",

" <td>0.00000</td>\n",

" <td>0.000000</td>\n",

" <td>51002.110000</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>50%</th>\n",

" <td>5000.50000</td>\n",

" <td>1.569074e+07</td>\n",

" <td>652.000000</td>\n",

" <td>37.000000</td>\n",

" <td>5.000000</td>\n",

" <td>97198.540000</td>\n",

" <td>1.000000</td>\n",

" <td>1.00000</td>\n",

" <td>1.000000</td>\n",

" <td>100193.915000</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>75%</th>\n",

" <td>7500.25000</td>\n",

" <td>1.575323e+07</td>\n",

" <td>718.000000</td>\n",

" <td>44.000000</td>\n",

" <td>7.000000</td>\n",

" <td>127644.240000</td>\n",

" <td>2.000000</td>\n",

" <td>1.00000</td>\n",

" <td>1.000000</td>\n",

" <td>149388.247500</td>\n",

" <td>0.000000</td>\n",

" </tr>\n",

" <tr>\n",

" <th>max</th>\n",

" <td>10000.00000</td>\n",

" <td>1.581569e+07</td>\n",

" <td>850.000000</td>\n",

" <td>92.000000</td>\n",

" <td>10.000000</td>\n",

" <td>250898.090000</td>\n",

" <td>4.000000</td>\n",

" <td>1.00000</td>\n",

" <td>1.000000</td>\n",

" <td>199992.480000</td>\n",

" <td>1.000000</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"</div>"

],

"text/plain": [

" RowNumber CustomerId CreditScore Age Tenure \\\n",

"count 10000.00000 1.000000e+04 10000.000000 10000.000000 10000.000000 \n",

"mean 5000.50000 1.569094e+07 650.528800 38.921800 5.012800 \n",

"std 2886.89568 7.193619e+04 96.653299 10.487806 2.892174 \n",

"min 1.00000 1.556570e+07 350.000000 18.000000 0.000000 \n",

"25% 2500.75000 1.562853e+07 584.000000 32.000000 3.000000 \n",

"50% 5000.50000 1.569074e+07 652.000000 37.000000 5.000000 \n",

"75% 7500.25000 1.575323e+07 718.000000 44.000000 7.000000 \n",

"max 10000.00000 1.581569e+07 850.000000 92.000000 10.000000 \n",

"\n",

" Balance NumOfProducts HasCrCard IsActiveMember \\\n",

"count 10000.000000 10000.000000 10000.00000 10000.000000 \n",

"mean 76485.889288 1.530200 0.70550 0.515100 \n",

"std 62397.405202 0.581654 0.45584 0.499797 \n",

"min 0.000000 1.000000 0.00000 0.000000 \n",

"25% 0.000000 1.000000 0.00000 0.000000 \n",

"50% 97198.540000 1.000000 1.00000 1.000000 \n",

"75% 127644.240000 2.000000 1.00000 1.000000 \n",

"max 250898.090000 4.000000 1.00000 1.000000 \n",

"\n",

" EstimatedSalary Exited \n",

"count 10000.000000 10000.000000 \n",

"mean 100090.239881 0.203700 \n",

"std 57510.492818 0.402769 \n",

"min 11.580000 0.000000 \n",

"25% 51002.110000 0.000000 \n",

"50% 100193.915000 0.000000 \n",

"75% 149388.247500 0.000000 \n",

"max 199992.480000 1.000000 "

]

},

"execution\_count": 54,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"ld.describe() #description of the data in the Dataset"

]

},

{

"cell\_type": "markdown",

"id": "a5859780",

"metadata": {},

"source": [

"# 5. Handle the Missing values."

]

},

{

"cell\_type": "code",

"execution\_count": 56,

"id": "65710509",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"RowNumber False\n",

"CustomerId False\n",

"Surname False\n",

"CreditScore False\n",

"Geography False\n",

"Gender False\n",

"Age False\n",

"Tenure False\n",

"Balance False\n",

"NumOfProducts False\n",

"HasCrCard False\n",

"IsActiveMember False\n",

"EstimatedSalary False\n",

"Exited False\n",

"dtype: bool"

]

},

"execution\_count": 56,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"ld.isnull().any()"

]

},

{

"cell\_type": "code",

"execution\_count": 58,

"id": "37fc9046",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"RowNumber 0\n",

"CustomerId 0\n",

"Surname 0\n",

"CreditScore 0\n",

"Geography 0\n",

"Gender 0\n",

"Age 0\n",

"Tenure 0\n",

"Balance 0\n",

"NumOfProducts 0\n",

"HasCrCard 0\n",

"IsActiveMember 0\n",

"EstimatedSalary 0\n",

"Exited 0\n",

"dtype: int64"

]

},

"execution\_count": 58,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"ld.isnull().sum()"

]

},

{

"cell\_type": "code",

"execution\_count": 60,

"id": "41dec879",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"<AxesSubplot:>"

]

},

"execution\_count": 60,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 2 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"sns.heatmap(ld.corr(),annot=True) # heatmap -a plot of rectangular data as a color-encoded matrix"

]

},

{

"cell\_type": "markdown",

"id": "603d606e",

"metadata": {},

"source": [

"# 6. Find the outliers and replace the outliers"

]

},

{

"cell\_type": "code",

"execution\_count": 73,

"id": "cd48d5b8",

"metadata": {},

"outputs": [

{

"name": "stderr",

"output\_type": "stream",

"text": [

"/home/lokesh/anaconda3/lib/python3.9/site-packages/seaborn/\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.\n",

" warnings.warn(\n"

]

},

{

"data": {

"text/plain": [

"<AxesSubplot:xlabel='CreditScore'>"

]

},

"execution\_count": 73,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"#occurence of outliers\n",

"ld1= pd.read\_csv(\"Churn\_Modelling.csv\")\n",

"sns.boxplot(ld1.CreditScore)"

]

},

{

"cell\_type": "code",

"execution\_count": 70,

"id": "db08968f",

"metadata": {},

"outputs": [

{

"name": "stderr",

"output\_type": "stream",

"text": [

"/home/lokesh/anaconda3/lib/python3.9/site-packages/seaborn/\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.\n",

" warnings.warn(\n"

]

},

{

"data": {

"text/plain": [

"<AxesSubplot:xlabel='CreditScore'>"

]

},

"execution\_count": 70,

"metadata": {},

"output\_type": "execute\_result"

},

{

"data": {

"image/png": "\n",

"text/plain": [

"<Figure size 432x288 with 1 Axes>"

]

},

"metadata": {

"needs\_background": "light"

},

"output\_type": "display\_data"

}

],

"source": [

"#Use Mean Detection and Nearest Fill Methods - Outliers\n",

"\n",

"Q1= ld1.CreditScore.quantile(0.25)\n",

"Q3=ld1.CreditScore.quantile(0.75)\n",

"IQR=Q3-Q1\n",

"upper\_limit =Q3 + 1.5\*IQR\n",

"lower\_limit =Q1 - 1.5\*IQR\n",

"ld1['CreditScore'] = np.where(ld1['CreditScore']>upper\_limit,30,ld1['CreditScore'])\n",

"sns.boxplot(ld1.CreditScore)"

]

},

{

"cell\_type": "markdown",

"id": "54066528",

"metadata": {},

"source": [

"# 7. Check for Categorical columns and perform encoding."

]

},

{

"cell\_type": "code",

"execution\_count": 77,

"id": "ce18977c",

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>RowNumber</th>\n",

" <th>CustomerId</th>\n",

" <th>Surname</th>\n",

" <th>CreditScore</th>\n",

" <th>Geography</th>\n",

" <th>Gender</th>\n",

" <th>Age</th>\n",

" <th>Tenure</th>\n",

" <th>Balance</th>\n",

" <th>NumOfProducts</th>\n",

" <th>HasCrCard</th>\n",

" <th>IsActiveMember</th>\n",

" <th>EstimatedSalary</th>\n",

" <th>Exited</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>0</th>\n",

" <td>1</td>\n",

" <td>15634602</td>\n",

" <td>Hargrave</td>\n",

" <td>619</td>\n",

" <td>France</td>\n",

" <td>0</td>\n",

" <td>42</td>\n",

" <td>2</td>\n",

" <td>0.00</td>\n",

" <td>1</td>\n",

" <td>1</td>\n",

" <td>1</td>\n",

" <td>101348.88</td>\n",

" <td>1</td>\n",

" </tr>\n",

" <tr>\n",

" <th>1</th>\n",

" <td>2</td>\n",

" <td>15647311</td>\n",

" <td>Hill</td>\n",

" <td>608</td>\n",

" <td>Spain</td>\n",

" <td>0</td>\n",

" <td>41</td>\n",

" <td>1</td>\n",

" <td>83807.86</td>\n",

" <td>1</td>\n",

" <td>0</td>\n",

" <td>1</td>\n",

" <td>112542.58</td>\n",

" <td>0</td>\n",

" </tr>\n",

" <tr>\n",

" <th>2</th>\n",

" <td>3</td>\n",

" <td>15619304</td>\n",

" <td>Onio</td>\n",

" <td>502</td>\n",

" <td>France</td>\n",

" <td>0</td>\n",

" <td>42</td>\n",

" <td>8</td>\n",

" <td>159660.80</td>\n",

" <td>3</td>\n",

" <td>1</td>\n",

" <td>0</td>\n",

" <td>113931.57</td>\n",

" <td>1</td>\n",

" </tr>\n",

" <tr>\n",

" <th>3</th>\n",

" <td>4</td>\n",

" <td>15701354</td>\n",

" <td>Boni</td>\n",

" <td>699</td>\n",

" <td>France</td>\n",

" <td>0</td>\n",

" <td>39</td>\n",

" <td>1</td>\n",

" <td>0.00</td>\n",

" <td>2</td>\n",

" <td>0</td>\n",

" <td>0</td>\n",

" <td>93826.63</td>\n",

" <td>0</td>\n",

" </tr>\n",

" <tr>\n",

" <th>4</th>\n",

" <td>5</td>\n",

" <td>15737888</td>\n",

" <td>Mitchell</td>\n",

" <td>850</td>\n",

" <td>Spain</td>\n",

" <td>0</td>\n",

" <td>43</td>\n",

" <td>2</td>\n",

" <td>125510.82</td>\n",

" <td>1</td>\n",

" <td>1</td>\n",

" <td>1</td>\n",

" <td>79084.10</td>\n",

" <td>0</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"</div>"

],

"text/plain": [

" RowNumber CustomerId Surname CreditScore Geography Gender Age \\\n",

"0 1 15634602 Hargrave 619 France 0 42 \n",

"1 2 15647311 Hill 608 Spain 0 41 \n",

"2 3 15619304 Onio 502 France 0 42 \n",

"3 4 15701354 Boni 699 France 0 39 \n",

"4 5 15737888 Mitchell 850 Spain 0 43 \n",

"\n",

" Tenure Balance NumOfProducts HasCrCard IsActiveMember \\\n",

"0 2 0.00 1 1 1 \n",

"1 1 83807.86 1 0 1 \n",

"2 8 159660.80 3 1 0 \n",

"3 1 0.00 2 0 0 \n",

"4 2 125510.82 1 1 1 \n",

"\n",

" EstimatedSalary Exited \n",

"0 101348.88 1 \n",

"1 112542.58 0 \n",

"2 113931.57 1 \n",

"3 93826.63 0 \n",

"4 79084.10 0 "

]

},

"execution\_count": 77,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"ld1.head(5)"

]

},

{

"cell\_type": "code",

"execution\_count": 75,

"id": "8a8ac99c",

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>RowNumber</th>\n",

" <th>CustomerId</th>\n",

" <th>Surname</th>\n",

" <th>CreditScore</th>\n",

" <th>Geography</th>\n",

" <th>Gender</th>\n",

" <th>Age</th>\n",

" <th>Tenure</th>\n",

" <th>Balance</th>\n",

" <th>NumOfProducts</th>\n",

" <th>HasCrCard</th>\n",

" <th>IsActiveMember</th>\n",

" <th>EstimatedSalary</th>\n",

" <th>Exited</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>0</th>\n",

" <td>1</td>\n",

" <td>15634602</td>\n",

" <td>Hargrave</td>\n",

" <td>619</td>\n",

" <td>France</td>\n",

" <td>0</td>\n",

" <td>42</td>\n",

" <td>2</td>\n",

" <td>0.00</td>\n",

" <td>1</td>\n",

" <td>1</td>\n",

" <td>1</td>\n",

" <td>101348.88</td>\n",

" <td>1</td>\n",

" </tr>\n",

" <tr>\n",

" <th>1</th>\n",

" <td>2</td>\n",

" <td>15647311</td>\n",

" <td>Hill</td>\n",

" <td>608</td>\n",

" <td>Spain</td>\n",

" <td>0</td>\n",

" <td>41</td>\n",

" <td>1</td>\n",

" <td>83807.86</td>\n",

" <td>1</td>\n",

" <td>0</td>\n",

" <td>1</td>\n",

" <td>112542.58</td>\n",

" <td>0</td>\n",

" </tr>\n",

" <tr>\n",

" <th>2</th>\n",

" <td>3</td>\n",

" <td>15619304</td>\n",

" <td>Onio</td>\n",

" <td>502</td>\n",

" <td>France</td>\n",

" <td>0</td>\n",

" <td>42</td>\n",

" <td>8</td>\n",

" <td>159660.80</td>\n",

" <td>3</td>\n",

" <td>1</td>\n",

" <td>0</td>\n",

" <td>113931.57</td>\n",

" <td>1</td>\n",

" </tr>\n",

" <tr>\n",

" <th>3</th>\n",

" <td>4</td>\n",

" <td>15701354</td>\n",

" <td>Boni</td>\n",

" <td>699</td>\n",

" <td>France</td>\n",

" <td>0</td>\n",

" <td>39</td>\n",

" <td>1</td>\n",

" <td>0.00</td>\n",

" <td>2</td>\n",

" <td>0</td>\n",

" <td>0</td>\n",

" <td>93826.63</td>\n",

" <td>0</td>\n",

" </tr>\n",

" <tr>\n",

" <th>4</th>\n",

" <td>5</td>\n",

" <td>15737888</td>\n",

" <td>Mitchell</td>\n",

" <td>850</td>\n",

" <td>Spain</td>\n",

" <td>0</td>\n",

" <td>43</td>\n",

" <td>2</td>\n",

" <td>125510.82</td>\n",

" <td>1</td>\n",

" <td>1</td>\n",

" <td>1</td>\n",

" <td>79084.10</td>\n",

" <td>0</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"</div>"

],

"text/plain": [

" RowNumber CustomerId Surname CreditScore Geography Gender Age \\\n",

"0 1 15634602 Hargrave 619 France 0 42 \n",

"1 2 15647311 Hill 608 Spain 0 41 \n",

"2 3 15619304 Onio 502 France 0 42 \n",

"3 4 15701354 Boni 699 France 0 39 \n",

"4 5 15737888 Mitchell 850 Spain 0 43 \n",

"\n",

" Tenure Balance NumOfProducts HasCrCard IsActiveMember \\\n",

"0 2 0.00 1 1 1 \n",

"1 1 83807.86 1 0 1 \n",

"2 8 159660.80 3 1 0 \n",

"3 1 0.00 2 0 0 \n",

"4 2 125510.82 1 1 1 \n",

"\n",

" EstimatedSalary Exited \n",

"0 101348.88 1 \n",

"1 112542.58 0 \n",

"2 113931.57 1 \n",

"3 93826.63 0 \n",

"4 79084.10 0 "

]

},

"execution\_count": 75,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"#label encoder\n",

"from sklearn.preprocessing import LabelEncoder\n",

"le=LabelEncoder()\n",

"ld1.Gender= le.fit\_transform(ld1.Gender)\n",

"ld1.head(5)"

]

},

{

"cell\_type": "code",

"execution\_count": 76,

"id": "17e12142",

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>RowNumber</th>\n",

" <th>CustomerId</th>\n",

" <th>Surname</th>\n",

" <th>CreditScore</th>\n",

" <th>Gender</th>\n",

" <th>Age</th>\n",

" <th>Tenure</th>\n",

" <th>Balance</th>\n",

" <th>NumOfProducts</th>\n",

" <th>HasCrCard</th>\n",

" <th>IsActiveMember</th>\n",

" <th>EstimatedSalary</th>\n",

" <th>Exited</th>\n",

" <th>Geography\_France</th>\n",

" <th>Geography\_Germany</th>\n",

" <th>Geography\_Spain</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>0</th>\n",

" <td>1</td>\n",

" <td>15634602</td>\n",

" <td>Hargrave</td>\n",

" <td>619</td>\n",

" <td>0</td>\n",

" <td>42</td>\n",

" <td>2</td>\n",

" <td>0.00</td>\n",

" <td>1</td>\n",

" <td>1</td>\n",

" <td>1</td>\n",

" <td>101348.88</td>\n",

" <td>1</td>\n",

" <td>1</td>\n",

" <td>0</td>\n",

" <td>0</td>\n",

" </tr>\n",

" <tr>\n",

" <th>1</th>\n",

" <td>2</td>\n",

" <td>15647311</td>\n",

" <td>Hill</td>\n",

" <td>608</td>\n",

" <td>0</td>\n",

" <td>41</td>\n",

" <td>1</td>\n",

" <td>83807.86</td>\n",

" <td>1</td>\n",

" <td>0</td>\n",

" <td>1</td>\n",

" <td>112542.58</td>\n",

" <td>0</td>\n",

" <td>0</td>\n",

" <td>0</td>\n",

" <td>1</td>\n",

" </tr>\n",

" <tr>\n",

" <th>2</th>\n",

" <td>3</td>\n",

" <td>15619304</td>\n",

" <td>Onio</td>\n",

" <td>502</td>\n",

" <td>0</td>\n",

" <td>42</td>\n",

" <td>8</td>\n",

" <td>159660.80</td>\n",

" <td>3</td>\n",

" <td>1</td>\n",

" <td>0</td>\n",

" <td>113931.57</td>\n",

" <td>1</td>\n",

" <td>1</td>\n",

" <td>0</td>\n",

" <td>0</td>\n",

" </tr>\n",

" <tr>\n",

" <th>3</th>\n",

" <td>4</td>\n",

" <td>15701354</td>\n",

" <td>Boni</td>\n",

" <td>699</td>\n",

" <td>0</td>\n",

" <td>39</td>\n",

" <td>1</td>\n",

" <td>0.00</td>\n",

" <td>2</td>\n",

" <td>0</td>\n",

" <td>0</td>\n",

" <td>93826.63</td>\n",

" <td>0</td>\n",

" <td>1</td>\n",

" <td>0</td>\n",

" <td>0</td>\n",

" </tr>\n",

" <tr>\n",

" <th>4</th>\n",

" <td>5</td>\n",

" <td>15737888</td>\n",

" <td>Mitchell</td>\n",

" <td>850</td>\n",

" <td>0</td>\n",

" <td>43</td>\n",

" <td>2</td>\n",

" <td>125510.82</td>\n",

" <td>1</td>\n",

" <td>1</td>\n",

" <td>1</td>\n",

" <td>79084.10</td>\n",

" <td>0</td>\n",

" <td>0</td>\n",

" <td>0</td>\n",

" <td>1</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"</div>"

],

"text/plain": [

" RowNumber CustomerId Surname CreditScore Gender Age Tenure \\\n",

"0 1 15634602 Hargrave 619 0 42 2 \n",

"1 2 15647311 Hill 608 0 41 1 \n",

"2 3 15619304 Onio 502 0 42 8 \n",

"3 4 15701354 Boni 699 0 39 1 \n",

"4 5 15737888 Mitchell 850 0 43 2 \n",

"\n",

" Balance NumOfProducts HasCrCard IsActiveMember EstimatedSalary \\\n",

"0 0.00 1 1 1 101348.88 \n",

"1 83807.86 1 0 1 112542.58 \n",

"2 159660.80 3 1 0 113931.57 \n",

"3 0.00 2 0 0 93826.63 \n",

"4 125510.82 1 1 1 79084.10 \n",

"\n",

" Exited Geography\_France Geography\_Germany Geography\_Spain \n",

"0 1 1 0 0 \n",

"1 0 0 0 1 \n",

"2 1 1 0 0 \n",

"3 0 1 0 0 \n",

"4 0 0 0 1 "

]

},

"execution\_count": 76,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"#one hot encoding\n",

"ld1\_main=pd.get\_dummies(ld1,columns=['Geography'])\n",

"ld1\_main.head()"

]

},

{

"cell\_type": "markdown",

"id": "3b9f0319",

"metadata": {},

"source": [

"# 8. Split the data into dependent and independent variables."

]

},

{

"cell\_type": "code",

"execution\_count": 85,

"id": "a87f66a3",

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"[[1 15634602 'Hargrave' ... 1 1 101348.88]\n",

" [2 15647311 'Hill' ... 0 1 112542.58]\n",

" [3 15619304 'Onio' ... 1 0 113931.57]\n",

" ...\n",

" [9998 15584532 'Liu' ... 0 1 42085.58]\n",

" [9999 15682355 'Sabbatini' ... 1 0 92888.52]\n",

" [10000 15628319 'Walker' ... 1 0 38190.78]]\n"

]

}

],

"source": [

"#Splitting the Dataset into the Independent Feature Matrix\n",

"df=pd.read\_csv(\"Churn\_Modelling.csv\")\n",

"X = df.iloc[:, :-1].values\n",

"print(X)"

]

},

{

"cell\_type": "code",

"execution\_count": 86,

"id": "0301a95c",

"metadata": {},

"outputs": [

{

"name": "stdout",

"output\_type": "stream",

"text": [

"[1 0 1 ... 1 1 0]\n"

]

}

],

"source": [

"#Extracting the Dataset to Get the Dependent Vector\n",

"Y = df.iloc[:, -1].values\n",

"print(Y)"

]

},

{

"cell\_type": "markdown",

"id": "701a4bee",

"metadata": {},

"source": [

"# 9. Scale the independent variables"

]

},

{

"cell\_type": "code",

"execution\_count": 96,

"id": "a6d76b29",

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>Age</th>\n",

" <th>Balance</th>\n",

" <th>EstimatedSalary</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>0</th>\n",

" <td>42</td>\n",

" <td>0.00</td>\n",

" <td>101348.88</td>\n",

" </tr>\n",

" <tr>\n",

" <th>1</th>\n",

" <td>41</td>\n",

" <td>83807.86</td>\n",

" <td>112542.58</td>\n",

" </tr>\n",

" <tr>\n",

" <th>2</th>\n",

" <td>42</td>\n",

" <td>159660.80</td>\n",

" <td>113931.57</td>\n",

" </tr>\n",

" <tr>\n",

" <th>3</th>\n",

" <td>39</td>\n",

" <td>0.00</td>\n",

" <td>93826.63</td>\n",

" </tr>\n",

" <tr>\n",

" <th>4</th>\n",

" <td>43</td>\n",

" <td>125510.82</td>\n",

" <td>79084.10</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"</div>"

],

"text/plain": [

" Age Balance EstimatedSalary\n",

"0 42 0.00 101348.88\n",

"1 41 83807.86 112542.58\n",

"2 42 159660.80 113931.57\n",

"3 39 0.00 93826.63\n",

"4 43 125510.82 79084.10"

]

},

"execution\_count": 96,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"w = df.head()\n",

"q = w[['Age','Balance','EstimatedSalary']] #spliting the dataset into measureable values\n",

"q"

]

},

{

"cell\_type": "code",

"execution\_count": 101,

"id": "d9287479",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"array([[0.75 , 0. , 0.63892099],\n",

" [0.5 , 0.52491194, 0.96014087],\n",

" [0.75 , 1. , 1. ],\n",

" [0. , 0. , 0.42305883],\n",

" [1. , 0.78610918, 0. ]])"

]

},

"execution\_count": 101,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"from sklearn.preprocessing import scale # library for scallling\n",

"from sklearn.preprocessing import MinMaxScaler\n",

"mm = MinMaxScaler()\n",

"\n",

"x\_scaled = mm.fit\_transform(q)\n",

"x\_scaled"

]

},

{

"cell\_type": "code",

"execution\_count": 103,

"id": "450a4893",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"array([[ 0.44232587, -1.13763618, 0.09337626],\n",

" [-0.29488391, 0.15434425, 0.96285595],\n",

" [ 0.44232587, 1.32369179, 1.07074687],\n",

" [-1.76930347, -1.13763618, -0.49092058],\n",

" [ 1.17953565, 0.79723632, -1.6360585 ]])"

]

},

"execution\_count": 103,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"from sklearn.preprocessing import StandardScaler\n",

"sc = StandardScaler()\n",

"x\_ss = sc.fit\_transform(q)\n",

"x\_ss"

]

},

{

"cell\_type": "code",

"execution\_count": 110,

"id": "0ae750af",

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>Age</th>\n",

" <th>Balance</th>\n",

" <th>EstimatedSalary</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>0</th>\n",

" <td>0.442326</td>\n",

" <td>-1.137636</td>\n",

" <td>0.093376</td>\n",

" </tr>\n",

" <tr>\n",

" <th>1</th>\n",

" <td>-0.294884</td>\n",

" <td>0.154344</td>\n",

" <td>0.962856</td>\n",

" </tr>\n",

" <tr>\n",

" <th>2</th>\n",

" <td>0.442326</td>\n",

" <td>1.323692</td>\n",

" <td>1.070747</td>\n",

" </tr>\n",

" <tr>\n",

" <th>3</th>\n",

" <td>-1.769303</td>\n",

" <td>-1.137636</td>\n",

" <td>-0.490921</td>\n",

" </tr>\n",

" <tr>\n",

" <th>4</th>\n",

" <td>1.179536</td>\n",

" <td>0.797236</td>\n",

" <td>-1.636059</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"</div>"

],

"text/plain": [

" Age Balance EstimatedSalary\n",

"0 0.442326 -1.137636 0.093376\n",

"1 -0.294884 0.154344 0.962856\n",

"2 0.442326 1.323692 1.070747\n",

"3 -1.769303 -1.137636 -0.490921\n",

"4 1.179536 0.797236 -1.636059"

]

},

"execution\_count": 110,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"from sklearn.preprocessing import scale\n",

"X\_scaled=pd.DataFrame(scale(q),columns=q.columns)\n",

"X\_scale=X\_scaled.head()\n",

"X\_scale"

]

},

{

"cell\_type": "markdown",

"id": "318a7a0d",

"metadata": {},

"source": [

"# 10. Split the data into training and testing"

]

},

{

"cell\_type": "code",

"execution\_count": 114,

"id": "54920cfb",

"metadata": {},

"outputs": [

{

"data": {

"text/html": [

"<div>\n",

"<style scoped>\n",

" .dataframe tbody tr th:only-of-type {\n",

" vertical-align: middle;\n",

" }\n",

"\n",

" .dataframe tbody tr th {\n",

" vertical-align: top;\n",

" }\n",

"\n",

" .dataframe thead th {\n",

" text-align: right;\n",

" }\n",

"</style>\n",

"<table border=\"1\" class=\"dataframe\">\n",

" <thead>\n",

" <tr style=\"text-align: right;\">\n",

" <th></th>\n",

" <th>Age</th>\n",

" <th>Balance</th>\n",

" <th>EstimatedSalary</th>\n",

" </tr>\n",

" </thead>\n",

" <tbody>\n",

" <tr>\n",

" <th>0</th>\n",

" <td>42</td>\n",

" <td>0.00</td>\n",

" <td>101348.88</td>\n",

" </tr>\n",

" <tr>\n",

" <th>1</th>\n",

" <td>41</td>\n",

" <td>83807.86</td>\n",

" <td>112542.58</td>\n",

" </tr>\n",

" <tr>\n",

" <th>2</th>\n",

" <td>42</td>\n",

" <td>159660.80</td>\n",

" <td>113931.57</td>\n",

" </tr>\n",

" <tr>\n",

" <th>3</th>\n",

" <td>39</td>\n",

" <td>0.00</td>\n",

" <td>93826.63</td>\n",

" </tr>\n",

" <tr>\n",

" <th>4</th>\n",

" <td>43</td>\n",

" <td>125510.82</td>\n",

" <td>79084.10</td>\n",

" </tr>\n",

" <tr>\n",

" <th>...</th>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" <td>...</td>\n",

" </tr>\n",

" <tr>\n",

" <th>9995</th>\n",

" <td>39</td>\n",

" <td>0.00</td>\n",

" <td>96270.64</td>\n",

" </tr>\n",

" <tr>\n",

" <th>9996</th>\n",

" <td>35</td>\n",

" <td>57369.61</td>\n",

" <td>101699.77</td>\n",

" </tr>\n",

" <tr>\n",

" <th>9997</th>\n",

" <td>36</td>\n",

" <td>0.00</td>\n",

" <td>42085.58</td>\n",

" </tr>\n",

" <tr>\n",

" <th>9998</th>\n",

" <td>42</td>\n",

" <td>75075.31</td>\n",

" <td>92888.52</td>\n",

" </tr>\n",

" <tr>\n",

" <th>9999</th>\n",

" <td>28</td>\n",

" <td>130142.79</td>\n",

" <td>38190.78</td>\n",

" </tr>\n",

" </tbody>\n",

"</table>\n",

"<p>10000 rows × 3 columns</p>\n",

"</div>"

],

"text/plain": [

" Age Balance EstimatedSalary\n",

"0 42 0.00 101348.88\n",

"1 41 83807.86 112542.58\n",

"2 42 159660.80 113931.57\n",

"3 39 0.00 93826.63\n",

"4 43 125510.82 79084.10\n",

"... ... ... ...\n",

"9995 39 0.00 96270.64\n",

"9996 35 57369.61 101699.77\n",

"9997 36 0.00 42085.58\n",

"9998 42 75075.31 92888.52\n",

"9999 28 130142.79 38190.78\n",

"\n",

"[10000 rows x 3 columns]"

]

},

"execution\_count": 114,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"x= df[['Age','Balance','EstimatedSalary']]\n",

"x"

]

},

{

"cell\_type": "code",

"execution\_count": 116,

"id": "43b8f672",

"metadata": {

"scrolled": true

},

"outputs": [

{

"data": {

"text/plain": [

"0 0.00\n",

"1 83807.86\n",

"2 159660.80\n",

"3 0.00\n",

"4 125510.82\n",

" ... \n",

"9995 0.00\n",

"9996 57369.61\n",

"9997 0.00\n",

"9998 75075.31\n",

"9999 130142.79\n",

"Name: Balance, Length: 10000, dtype: float64"

]

},

"execution\_count": 116,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"y = df['Balance']\n",

"y"

]

},

{

"cell\_type": "code",

"execution\_count": 118,

"id": "878e064d",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"array([[ 0.29351742, -1.22584767, 0.02188649],\n",

" [ 0.19816383, 0.11735002, 0.21653375],\n",

" [ 0.29351742, 1.33305335, 0.2406869 ],\n",

" ...,\n",

" [-0.27860412, -1.22584767, -1.00864308],\n",

" [ 0.29351742, -0.02260751, -0.12523071],\n",

" [-1.04143285, 0.85996499, -1.07636976]])"

]

},

"execution\_count": 118,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"#scaling\n",

"from sklearn.preprocessing import StandardScaler, MinMaxScaler\n",

"sc = StandardScaler()\n",

"x\_scaled1 = sc.fit\_transform(x)\n",

"x\_scaled1"

]

},

{

"cell\_type": "code",

"execution\_count": 119,

"id": "75315a1e",

"metadata": {},

"outputs": [],

"source": [

"#train and test data\n",

"from sklearn.model\_selection import train\_test\_split\n",

"x\_train, x\_test, y\_train, y\_test = train\_test\_split(x\_scaled1, y, test\_size = 0.3, random\_state = 0)"

]

},

{

"cell\_type": "code",

"execution\_count": 120,

"id": "2a41693d",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"array([[-0.56466489, 1.11721307, -0.77021814],\n",

" [ 0.00745665, -1.22584767, -1.39576675],\n",

" [ 3.53553951, 1.35419118, -1.49965629],\n",

" ...,\n",

" [-0.37395771, 1.35890908, 1.41441489],\n",

" [-0.08789694, -1.22584767, 0.84614739],\n",

" [ 0.86563897, 0.50630343, 0.32630495]])"

]

},

"execution\_count": 120,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"x\_train"

]

},

{

"cell\_type": "code",

"execution\_count": 121,

"id": "7cbb5036",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"(7000, 3)"

]

},

"execution\_count": 121,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"x\_train.shape"

]

},

{

"cell\_type": "code",

"execution\_count": 122,

"id": "38c6ee3a",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"array([[-0.37395771, 0.87532296, 1.61304597],\n",

" [ 0.10281024, 0.42442221, 0.49753166],\n",

" [ 0.29351742, 0.30292727, -0.4235611 ],\n",

" ...,\n",

" [ 0.10281024, 1.46672809, 1.17045451],\n",

" [ 2.86806437, 1.25761599, -0.50846777],\n",

" [ 0.96099256, 0.19777742, -1.15342685]])"

]

},

"execution\_count": 122,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"x\_test"

]

},

{

"cell\_type": "code",

"execution\_count": 123,

"id": "655a9540",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"(3000, 3)"

]

},

"execution\_count": 123,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"x\_test.shape"

]

},

{

"cell\_type": "code",

"execution\_count": 124,

"id": "3e0be5d3",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"7681 146193.60\n",

"9031 0.00\n",

"3691 160979.68\n",

"202 0.00\n",

"5625 143262.04\n",

" ... \n",

"9225 120074.97\n",

"4859 114440.24\n",

"3264 161274.05\n",

"9845 0.00\n",

"2732 108076.33\n",

"Name: Balance, Length: 7000, dtype: float64"

]

},

"execution\_count": 124,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"y\_train"

]

},

{

"cell\_type": "code",

"execution\_count": 125,

"id": "9f976094",

"metadata": {},

"outputs": [

{

"data": {

"text/plain": [

"9394 131101.04\n",

"898 102967.41\n",

"2398 95386.82\n",

"5906 112079.58\n",

"2343 163034.82\n",

" ... \n",

"4004 0.00\n",

"7375 80926.02\n",

"9307 168001.34\n",

"8394 154953.94\n",

"5233 88826.07\n",

"Name: Balance, Length: 3000, dtype: float64"

]

},

"execution\_count": 125,

"metadata": {},

"output\_type": "execute\_result"

}

],

"source": [

"y\_test"

]

},

{

"cell\_type": "code",

"execution\_count": null,

"id": "876fdc2f",

"metadata": {},

"outputs": [],

"source": []

}

],

"metadata": {

"kernelspec": {

"display\_name": "Python 3 (ipykernel)",

"language": "python",

"name": "python3"

},

"language\_info": {

"codemirror\_mode": {

"name": "ipython",

"version": 3

},

"file\_extension": ".py",

"mimetype": "text/x-python",

"name": "python",

"nbconvert\_exporter": "python",

"pygments\_lexer": "ipython3",

"version": "3.9.12"

}

},

"nbformat": 4,

"nbformat\_minor": 5

}