```
{
 "cells": [
  {
   "cell_type": "markdown",
   "id": "e5c11b1b",
   "metadata": {},
   "source": [
    "# Bank Customer Churn Model"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "daea73c6",
   "metadata": {},
   "source": [
    "# Import Library"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 1,
   "id": "a8bde5ea",
   "metadata": {},
   "outputs": [],
   "source": [
    "import pandas as pd"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 2,
   "id": "50290e77",
   "metadata": {},
   "outputs": [],
   "source": [
    "import numpy as np"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 3,
   "id": "51b12d21",
   "metadata": {},
   "outputs": [],
   "source": [
    "import matplotlib.pyplot as plt"
   ]
  },
```

```
 {
  "cell_type": "code",
  "execution_count": 4,
  "id": "a9568165",
  "metadata": {},
  "outputs": [],
  "source": [
   "import seaborn as sns"
  ]
 },
 {
  "cell_type": "markdown",
  "id": "511b59d0",
  "metadata": {},
  "source": [
   "# Import Data"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 5,
  "id": "18070d76",
  "metadata": {},
  "outputs": [],
  "source": [
   "df =
pd.read_csv(\"https://raw.githubusercontent.com/YBI-Foundation/Dataset/main/Bank%20Ch
urn%20Modelling.csv\")"
  ]
 },
 {
  "cell_type": "markdown",
  "id": "8594be6c",
  "metadata": {},
  "source": [
   "# Analyse Data"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 6,
  "id": "da684868",
  "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>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>Num Of Products</th>\n",
"      <th>Has Credit Card</th>\n",
"      <th>Is Active Member</th>\n",
"      <th>Estimated Salary</th>\n",
"      <th>Churn</th>\n",
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th>0</th>\n",
"      <td>15634602</td>\n",
"      <td>Hargrave</td>\n",
"      <td>619</td>\n",
"      <td>France</td>\n",
"      <td>Female</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>15647311</td>\n",
"      <td>Hill</td>\n",
"      <td>608</td>\n",
"      <td>Spain</td>\n",
"      <td>Female</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>15619304</td>\n",
"      <td>Onio</td>\n",
"      <td>502</td>\n",
"      <td>France</td>\n",
"      <td>Female</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>15701354</td>\n",
"      <td>Boni</td>\n",
"      <td>699</td>\n",
"      <td>France</td>\n",
"      <td>Female</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>15737888</td>\n",
    "      <td>Mitchell</td>\n",
    "      <td>850</td>\n",
    "      <td>Spain</td>\n",
    "      <td>Female</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": [
    "   CustomerId   Surname  CreditScore Geography  Gender  Age  Tenure  \\\n",
    "0    15634602  Hargrave          619    France  Female   42       2   \n",
    "1    15647311      Hill          608     Spain  Female   41       1   \n",
    "2    15619304      Onio          502    France  Female   42       8   \n",
    "3    15701354      Boni          699    France  Female   39       1   \n",
    "4    15737888  Mitchell          850     Spain  Female   43       2   \n",
    "\n",
    "     Balance  Num Of Products  Has Credit Card  Is Active Member  \\\n",
    "0       0.00                1                1                 1   \n",
    "1   83807.86                1                0                 1   \n",
    "2  159660.80                3                1                 0   \n",
    "3       0.00                2                0                 0   \n",
    "4  125510.82                1                1                 1   \n",
    "\n",
    "   Estimated Salary  Churn  \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": 6,
  "metadata": {},
  "output_type": "execute_result"
 }
```

```
 ],
 "source": [
  "df.head()"
 ]
},
{
 "cell_type": "code",
 "execution_count": 7,
 "id": "f9525017",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "<class 'pandas.core.frame.DataFrame'>\n",
    "RangeIndex: 10000 entries, 0 to 9999\n",
    "Data columns (total 13 columns):\n",
    " #   Column            Non-Null Count  Dtype  \n",
    "---  ------            --------------  -----  \n",
    " 0   CustomerId        10000 non-null  int64  \n",
    " 1   Surname           10000 non-null  object \n",
    " 2   CreditScore       10000 non-null  int64  \n",
    " 3   Geography         10000 non-null  object \n",
    " 4   Gender            10000 non-null  object \n",
    " 5   Age               10000 non-null  int64  \n",
    " 6   Tenure            10000 non-null  int64  \n",
    " 7   Balance           10000 non-null  float64\n",
    " 8   Num Of Products   10000 non-null  int64  \n",
    " 9   Has Credit Card   10000 non-null  int64  \n",
    " 10  Is Active Member  10000 non-null  int64  \n",
    " 11  Estimated Salary  10000 non-null  float64\n",
    " 12  Churn             10000 non-null  int64  \n",
    "dtypes: float64(2), int64(8), object(3)\n",
    "memory usage: 1015.8+ KB\n"
   ]
  }
 ],
 "source": [
  "df.info()"
 ]
},
{
 "cell_type": "code",
 "execution_count": 8,
 "id": "c692250c",
 "metadata": {},
 "outputs": [
```

```
  {
   "data": {
    "text/plain": [
     "0"
    ]
   },
   "execution_count": 8,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "df.duplicated('CustomerId').sum()"
 ]
},
{
 "cell_type": "code",
 "execution_count": 9,
 "id": "7b48ed23",
 "metadata": {},
 "outputs": [],
 "source": [
  "df = df.set_index('CustomerId')"
 ]
},
{
 "cell_type": "code",
 "execution_count": 10,
 "id": "b0266691",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "<class 'pandas.core.frame.DataFrame'>\n",
    "Int64Index: 10000 entries, 15634602 to 15628319\n",
    "Data columns (total 12 columns):\n",
    " #   Column            Non-Null Count  Dtype  \n",
    "---  ------            --------------  -----  \n",
    " 0   Surname           10000 non-null  object \n",
    " 1   CreditScore       10000 non-null  int64  \n",
    " 2   Geography         10000 non-null  object \n",
    " 3   Gender            10000 non-null  object \n",
    " 4   Age               10000 non-null  int64  \n",
    " 5   Tenure            10000 non-null  int64  \n",
    " 6   Balance           10000 non-null  float64\n",
    " 7   Num Of Products   10000 non-null  int64  \n",
```

```
    " 8   Has Credit Card   10000 non-null  int64  \n",
    " 9   Is Active Member  10000 non-null  int64  \n",
    " 10  Estimated Salary  10000 non-null  float64\n",
    " 11  Churn             10000 non-null  int64  \n",
    "dtypes: float64(2), int64(7), object(3)\n",
    "memory usage: 1015.6+ KB\n"
   ]
  }
 ],
 "source": [
  "df.info()"
 ]
},
{
 "cell_type": "markdown",
 "id": "1326b1a7",
 "metadata": {},
 "source": [
  "# Encoding"
 ]
},
{
 "cell_type": "code",
 "execution_count": 11,
 "id": "ee1f246e",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "France     5014\n",
     "Germany    2509\n",
     "Spain      2477\n",
     "Name: Geography, dtype: int64"
    ]
   },
   "execution_count": 11,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "df['Geography'].value_counts()"
 ]
},
{
 "cell_type": "code",
 "execution_count": 12,
```

```
 "id": "2c26fc92",
 "metadata": {},
 "outputs": [],
 "source": [
  "df.replace({'Geography' : {'France' : 2, 'Germany' : 1, 'Spain' : 0}}, inplace = True)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 13,
 "id": "0c4057e7",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "Male      5457\n",
     "Female    4543\n",
     "Name: Gender, dtype: int64"
    ]
   },
   "execution_count": 13,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "df['Gender'].value_counts()"
 ]
},
{
 "cell_type": "code",
 "execution_count": 14,
 "id": "aea29223",
 "metadata": {},
 "outputs": [],
 "source": [
  "df.replace({'Gender' : {'Male' : 0, 'Female' : 1}}, inplace = True)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 15,
 "id": "bbd05c50",
 "metadata": {},
 "outputs": [
  {
   "data": {
```

```
    "text/plain": [
     "1    5084\n",
     "2    4590\n",
     "3     266\n",
     "4      60\n",
     "Name: Num Of Products, dtype: int64"
    ]
   },
   "execution_count": 15,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "df['Num Of Products'].value_counts()"
 ]
},
{
 "cell_type": "code",
 "execution_count": 16,
 "id": "cd4d42a1",
 "metadata": {},
 "outputs": [],
 "source": [
  "df.replace({'Num Of Products' : {1 : 0, 2 : 1, 3 : 1, 4 : 1}}, inplace = True)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 17,
 "id": "8a1f2483",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "1    7055\n",
     "0    2945\n",
     "Name: Has Credit Card, dtype: int64"
    ]
   },
   "execution_count": 17,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "df['Has Credit Card'].value_counts()"
```

```
 ]
},
{
 "cell_type": "code",
 "execution_count": 18,
 "id": "6c4527fe",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "1    5151\n",
     "0    4849\n",
     "Name: Is Active Member, dtype: int64"
    ]
   },
   "execution_count": 18,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "df['Is Active Member'].value_counts()"
 ]
},
{
 "cell_type": "code",
 "execution_count": 19,
 "id": "e7a7ecd7",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "0    3117\n",
     "1     500\n",
     "Name: Churn, dtype: int64"
    ]
   },
   "execution_count": 19,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "df.loc[(df['Balance'] == 0), 'Churn'].value_counts()"
 ]
},
```

{
"cell_type": "code",
"execution_count": 20,
"id": "d77620f9",
"metadata": {},
"outputs": [],
"source": [
"df['Zero Balance'] = np.where(df['Balance'] > 0, 1, 0)"
]
},
{
"cell_type": "code",
"execution_count": 21,
"id": "af879d50",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"<AxesSubplot:>"
]
},
"execution_count": 21,
"metadata": {},
"output_type": "execute_result"
},
{
"data": {
"image/png":
"iVBORw0KGgoAAAANSUhEUgAAAX0AAAD4CAYAAAAAczaOAAAAOXRFWHRTb2Z0d2F
yZQBNYXRwbG90bGliIHZlcnNpb24zLjQuMywgaHR0cHM6Ly9tYXRwbG90bGliLm9yZy/Mn
kTPAAAACXBIWXMAAAsTAAALEwEAmpwYAAATUUlEQVR4nO3dcYxI5X3e8e9j1tgucQyY
ZIQW2qXypikJso1GgJUqHZt2WUjlRaqDsJyyoFVXSqnltqjtuv2DFmLJqHJcYzlON2XLYpFgQ
uqyCjR0tWZktepioDhgIC4TDGG34E28sO0Y2em6v/5x3yETvMPc2blzL5P3+5FG95z3vOec9
zezPOfcc889pKqQJPXhLZMegCRpfAx9SeqIoS9JHTH0Jakjhr4kdWTDpAfwRs4666zatGnT
Sa//ve99j9NOO210A3qT661esOZveWPPKProo39SVT9xomVM71gzb2w5pV8vxSy7y8I0kdMfQlqSOGvqS9JHTyNXkiZp0677Jrbv27euzWMnPNOXpI4Y+pLUEUNfkjpi6kjye5MJF29ne+j+T
ZPtaFSVJOrFhz/Q/B/xeVf008F7gaWAXcKCqNgMH2jzA5cDm9rMT+CJAkjOBG4GLgYuAGx
cOFJKk8Vg29JO8C/h54QD/h54DaAqvrTqnoF2Absbd32AIe26W3AHTVwEDg9ydcZ+qjpaVS8D
+4GtI6xFkrSMYZ69cx7wX8B/SPJe4EHgOmB9VR0HqKrn24ZH0ETwAXh+vqoeSfI4/u5QD
qakpZmdnh63lR8zPz69q/fWmt3BmnsxuOD42Pe5YK1qHHib0NwAXAh+tqgeSfI4/u5QD
QFVVkhrFgKpqN7AbYGbHHwp8ZDcGSPq6eGbpmrnxJ11Ssz5YyZ6czMe5K6xjMm7AbK9R/
fWmt3rBmnsxqZqvnfAD19ai5mGu6R8C
DlXVQ23+HgYHge+0gyYHge+0yza01yNt+WHg3EXrn9PalmqXJXJI3JsqFfVS8BLyT5a63pU
uApYB+wcAJLkjPYB7pbWJkaak2GFp/9x4M4kpwLPAlczOGDcneQa4Ft8sM8e8DlwQH
8DxwVet7P3AFMAe82vpSVt1IVHR1KFJGkoQ4V+VX0Dz8A65fdlL2DEBoktP0LeA65fY
zh5gzwrGJ0kaIb+RK0kdMfQlqSOGvqS9JHTH0J

akjhr4kdcTQl6SOGPqS1BFDX5I6YuhLUkcMfUnqiKEvSR0x9CWpI4a+JHXE0Jekjhj6ktQRQ
1+SOmLoS1JHhgr9JM8IeSLJN5I80trOTLI/yTPt9YzWniS3JpIL8niSCxdtZ3vr/0yS7WtTkiRpK
Ss50/9gVb2vqqbb/C7gQFVtBg60eYDLgc3tZyfwRRgcJIAbgYuBi4AbFw4UkqTxWM3InW3A3
ja9F7hyUfsdNXAQOD3J2cBIwP6qOlpVLwP7ga2r2L8kaYWGDf0C/kuSR5PsbG1TVfVim34J
mGrTG4EXFq17qLUt1S1S5JGpMNQ/b7G1V1OMlPAvuT/MHihVVVSWoUA2oHlZ0AU1NTzM7
OnvS25ufnV7X+etNbvWDNvZhUzTdccHzs+1ywVjUPFfpVdbi9HknyFQbX5L+T5OyqerFdvjn
Suh8GzI20+jmt7TAw87r22RPsazewG2B6erpmZmZe32Vos7OzrGb99aa3esGaezGpmq/ddd/
Y97ng9q2nrUnNy17eSXJakncuTANbgG8C+4CFO3C2A/e26X3ANe0unkuAY+0y0APAliRntA
9wt7Q2SdKYDHOmPwV8JcIC/9+sqt9L8jBwd5IdwPPAVa3//cAVwBzwKnAdQFUdTXIz8HDrd
1NVHR1ZJZKkZS0b+IX1LPDeE7R/F7j0BO0FXL/EtvYAe1Y+TEnSKPiNXEnqiKEvSR0x9CW
pI4a+JHXE0Jekjhj6ktQRQ1+SOmLoS1JHDH1J6oihL0kdMfQIqSOGviR1xNCXpI4Y+pLUEU
Nfkjpi6EtSRwx9SeqIoS9JHTH0Jakjhr4kdcTQl6SOGPqS1BFDX5I6YuhLUkeGDv0kpyR5LM
nvtvnzkjyUZC7JI5Oc2trf1ubn2vJNi7bxydb+rSSXjbwaSdIbWsmZ/ieApxfN3wJ8tqreAw7wM7Gjt
O4CXW/tnWz+SnA9cDfBw/ALw79t8gA8B97Que4Er2/RNk
9bfmnrvw24q6p+UFXfBuaAi0ZQgyRpSBuG7PdvgX8Gvq+4rTbhO6DxUuLIJj4naraOtDvF/3eT
c51vpvBA4u2ubidV6TZCewE2BqaorZ2dkhh/ij5ufnV7X+etNbvWDNvZhUzTdccHz5TmtkrWp
eNvST/B3gSFU9mmRm5CN4naraDewGmJ6erpmZk9/l7Owsq1I/vemtXrDmXkyq5mt33Tf2fS
64fetpa1LzMGf6Pwd8OMkVwNuBHwc+B5yeZ4eZEM72z8HONz6HwbOBQ4l2QC8C8C/juovYFi9e
RJI3Bstf0q+qTVXVOVW1i8EHsV6vqY8CDwDdWMmAvW16H3P4unLf9qVrXVVCvrd3XMesBn4+s
gqkSQta9hr+ifyz4G7kvwK8BhwW2u/DfidJHPAUYZ6nDuiCeaA48D1VfWgBvf+agA48D1VfXDVexfkrRCKwr
9qpoFZtv0s5zg7puPyzR0OSr4h3eZqyfZC2wcAzi6rStVj6MMYnlOV0v+m6Rt5KFXA4nHFnNkNnYfiyHMOWxvUstQ
KwB9ezVH4Tb5OjGJTs9lFQ0Yp+VxkpXTW2nl8/Wa5Q7/gVO6Xa6S0XJQ8wC7Fv3Xf/6vyG/GJcI4UfZaR9c
IHTBp+z57ex2qpJOV3A06TSfPUkG9QMGhI/J76P4DX+V/BTTlOXTXpqKJjIlBoOvSZbDo+1Zgg4ggrBYH
8JiB+oZuN2+tb+KqY4gkG5CmI+EC87XnHSkYmD4vZJuRxvWrPIS5CmsKMH+rsF7TdJ/6SZ9XPakEjPWrHlZqAx
kpZbWvzfvI+ETozVrWqcWQTE9Ndf4xIr9JPXtvr7Vwr9WO9P+fe5NZ+bFtOZ2b8lPRsdnU3dA5rz/DoEPO/MTK
ZOyXbGyXp8mFcVeP74aNu7NaCuPg5HJXFXB7yKUw7jv3kaFvj6f/SaO5NU+NrNTkW/UerwyJI6RjBqpL9Po
OL3/dmRFJvEVjaqbJCkIVN1NK3IOW4tJTfv8pv8z6tVb6MQFWlYpdnVoXN1LqHSSnw/VFsJ5TbcLWuqFkhy
ze4KvWaXtl2m3jpqh+JXNA1e0a93BnQ+kYCM3PKj/7eRfQvmJvFcgW1bXH2Egk+iXePgEvmvG9JgHhQXq
jtHNpLTB0Km5NVtNaZ8r7oR2eI7L4DjKOjR5gQkEHn4/Gy7VHHoaxhGKbCVL18vKsxfXU5D2bIYYEsPIVmvs8x
TWVGyWJlrJUMIzu4UWe/2uo+WZvaRSlsMRZOkUE9MYhlZ9e06egK0uOzdmh5c4M5E+HSsoaXi2CPSVHG
pn4Q6r4gI0jnxhLu35m1LTvtQm73s3/3Lgt0z3XBcJmAqBdb1TVpWjF/L9sUyaLT8BJnqd14kk9ojt2JM9fIaxN
n+1QnkbTIlNv54KrMS+L6sNlvTwE91L9+VdaSnSTQVxgb/XImoTHU3vLfLFj1tBjcoGQzR4m2+n8nSqgLGb
Z+Ns5Q+eEPOk4Fp5+2SK9xvxcAzJLIYnjLgewS/8BEf7uqVC3P4Jc2gJwh6+ZcKnxOuMAPtBfzG0CYHbfoX
LiGwhUWn2+AzF7WSaOv5tKAaAD+Dh7M+QlhdVHSj2UdZsRuwqwYZn3lhnl3e4J+6bIMpFffAEFTcBqv/Z5

AIklaYxtOcr2pqnqxTb8ETLXpjcALi/odam1Ltf+IJDsZvEtgamqK2dnZkxwiTL0Dbrjg+Emvf7JW
M+bVmJ+fn9i+J8Wa+zCpmieRHwvWquaTDf3XVFUlqVEMpm1vN7AbYHp6umZmZk56W5+
/814+88SqS1yx5z42M/Z9wuBgs5rf13pkzX2YVM3X7rpv7PtccPvW09ak5pO9e+c77bIN7fVIa
z8MnLuo3zmtbal2SdIYnWzo7wMW7sDZDty7qP2adhfPJcCxdhnoAWBLkjPaB7hbWpskaYy
WvfaR5LeAGeCsJIcY3IXzaeDuJDuA54GrWvf7gSuAOeBV4DqAqjqa5Gbg4dbvpqp6/YfDkq
Q1tmzoV9VHI1h06Qn6FnD9EtvZA+xZ0egkSSPlN3IlqSOGviR1xNCXpI4Y+pLUEUNfkjpi6Et
SRwx9SeqIoS9JHTH0Jakjhr4kdcTQl6SOGPqS1BFDX5I6YuhLUkcMfUnqiKEvSR0x9CWpI4
a+JHXE0Jekjhj6ktQRQ1+SOmLoS1JHDH1J6oihL0kdGXvoJ9ma5FtJ5pLsGvf+JalnYw39JK
cAXwAuB84HPprk/HGOQZJ6Nu4z/YuAuap6tqr+FLgL2DbmMUhStzaMeX8bgRcWzR8CLl7c
IclOYGebnU/yrVXs7yzgT1ax/knJLePe42smUu+EWXMfuqv5g7esqua/stSCcYf+sqpqN7B7F
NtK8khVTY9iW+tBb/WCNffCmkdn3Jd3DgPnLpo/p7VJksZg3KH/MLA5yXlJTgWuBvaNeQyS
1K2xXt6pquNJ/iHwAHAKsKeqnlzDXY7kMtE60lu9YM29sOYRSVWtxXYlSW9CfiNXkjpi6EtS
R9Z96C/3WIckb0vy5bb8oSSbJjDMkRqi5n+S5Kkkjyc5kGTJe3bXi2Ef35Hk7yapJOv+9r5hak
5yVftbP5nkN8c9xlEb4t/2X07yYJLH2r/vKyYxzlFJsifJkSTfXGJ5ktzafh+PJ7lw1TutqnX7w+DD
4D8E/ipwKvD7wPmv6/MPgF9v01cDX570uMdQ8weBv9Smf7mHmlu/dwJfAw4C05Me9xj+zp
uBx4Az2vxPTnrcY6h5N/DLbfp84LlJj3uVNf88cCHwzSWWXwH8ZyDAJcBDq93nej/TH+axDt
uAvW36HuDSJBnjGEdt2Zqr6sGqerXNHmTwfYj1bNjHd9wM3AJ8f5yDWyPD1Pz3gS9U1csA
VXVkzGMctWFqLuDH2/S7gP81xvGNXFV9DTj6Bl22AXfUwEHg9CRnr2af6z30T/RYh41L9a
mq48Ax4N1jGd3aGKbmxXYwOFNYz5atub3tPbeq7hvnwNbQMH/nnwJ+Ksl/S3lwydaxjW5tD
FPzvwJ+Kckh4H7g4+MZ2sSs9L/3Zb3pHsOg0UnyS8A08DcnPZa1lOQtwK8C1054KOO2gc
ElnhkG7+a+luSCqnplkoNaYx8Fbq+qzyT5APClJD9bVf9v0gNbL9b7mf4wj3V4rU+SDQzeEn5
3LKNbG0M9yiLJ3wL+JfDhqvrBmMa2Vpar+Z3AzwKzSZ5jcO1z3zr/MHeYv/MhYF9V/d+q+jb
wPxkcBNarYWreAdwNUFX/HXg7g4ex/UU18kfXrPfQH+axDvuA7W36I8BXq31Csk4tW3OS9
wP/jkHgr/frvLBMzVV1rKrOqqpNVbWJwecYH66qRyYz3JEY5t/2f2Jwlk+Ssxhc7nl2jGMctWFq
/iPgUoAkf51B6P/xWEc5XvuAa9pdPJcAx6rqxdVscF1f3qklHuuQ5CbgkaraB9zG4C3gHIMT
K6e3IhXb8ia/w3wY8Bvt8+s/6iqPjyxQa/SkDX/hTJkzQ8AW5I8BfwQ+KdVtW7fxQ5Z8w3AbyT
5xww+1L12PZ/EJfktBgfus9rnFDcCbwWoql9n8LnFCcAc8Cpw3ar3ar3uY5/X5KkFVrvl3ckSStg6
EtSRwx9SeqIoS9JHTH0Tqw4AAAABJRU5ErkJggg==\n",
      "text/plain": [
       "<Figure size 432x288 with 1 Axes>"
      ]
     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    }
   ],
   "source": [
    "df['Zero Balance'].hist()"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 22,
   "id": "c1f6ddba",
   "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></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>Num Of Products</th>\n",
   "      <th>Has Credit Card</th>\n",
   "      <th>Is Active Member</th>\n",
   "      <th>Estimated Salary</th>\n",
   "      <th>Zero Balance</th>\n",
   "    </tr>\n",
   "    <tr>\n",
   "      <th>Churn</th>\n",
   "      <th>Geography</th>\n",
   "      <th></th>\n",
   "      <th></th>\n",
   "      <th></th>\n",
   "      <th></th>\n",
   "      <th></th>\n",
   "      <th></th>\n",
   "      <th></th>\n",
   "      <th></th>\n",
   "      <th></th>\n",
   "      <th></th>\n",
   "      <th></th>\n",
```

```
"    </tr>\n",
"  </thead>\n",
"  <tbody>\n",
"    <tr>\n",
"      <th rowspan=\"3\" valign=\"top\">0</th>\n",
"      <th>0</th>\n",
"      <td>2064</td>\n",
"      <td>2064</td>\n",
"      <td>2064</td>\n",
"      <td>2064</td>\n",
"      <td>2064</td>\n",
"      <td>2064</td>\n",
"      <td>2064</td>\n",
"      <td>2064</td>\n",
"      <td>2064</td>\n",
"      <td>2064</td>\n",
"      <td>2064</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>1</th>\n",
"      <td>1695</td>\n",
"      <td>1695</td>\n",
"      <td>1695</td>\n",
"      <td>1695</td>\n",
"      <td>1695</td>\n",
"      <td>1695</td>\n",
"      <td>1695</td>\n",
"      <td>1695</td>\n",
"      <td>1695</td>\n",
"      <td>1695</td>\n",
"      <td>1695</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th>2</th>\n",
"      <td>4204</td>\n",
"      <td>4204</td>\n",
"      <td>4204</td>\n",
"      <td>4204</td>\n",
"      <td>4204</td>\n",
"      <td>4204</td>\n",
"      <td>4204</td>\n",
"      <td>4204</td>\n",
"      <td>4204</td>\n",
"      <td>4204</td>\n",
"      <td>4204</td>\n",
"    </tr>\n",
"    <tr>\n",
"      <th rowspan=\"3\" valign=\"top\">1</th>\n",
```

```
 "      <th>0</th>\n",
 "      <td>413</td>\n",
 "      <td>413</td>\n",
 "      <td>413</td>\n",
 "      <td>413</td>\n",
 "      <td>413</td>\n",
 "      <td>413</td>\n",
 "      <td>413</td>\n",
 "      <td>413</td>\n",
 "      <td>413</td>\n",
 "      <td>413</td>\n",
 "      <td>413</td>\n",
 "    </tr>\n",
 "    <tr>\n",
 "      <th>1</th>\n",
 "      <td>814</td>\n",
 "      <td>814</td>\n",
 "      <td>814</td>\n",
 "      <td>814</td>\n",
 "      <td>814</td>\n",
 "      <td>814</td>\n",
 "      <td>814</td>\n",
 "      <td>814</td>\n",
 "      <td>814</td>\n",
 "      <td>814</td>\n",
 "      <td>814</td>\n",
 "      <td>814</td>\n",
 "    </tr>\n",
 "    <tr>\n",
 "      <th>2</th>\n",
 "      <td>810</td>\n",
 "      <td>810</td>\n",
 "      <td>810</td>\n",
 "      <td>810</td>\n",
 "      <td>810</td>\n",
 "      <td>810</td>\n",
 "      <td>810</td>\n",
 "      <td>810</td>\n",
 "      <td>810</td>\n",
 "      <td>810</td>\n",
 "      <td>810</td>\n",
 "      <td>810</td>\n",
 "    </tr>\n",
 "  </tbody>\n",
 "</table>\n",
 "</div>"
],
"text/plain": [
 "            Surname CreditScore Gender  Age  Tenure  Balance  \\\n",
 "Churn Geography                                                \n",
```

```
    "0     0           2064        2064    2064  2064    2064     2064   \n",
    "      1           1695        1695    1695  1695    1695     1695   \n",
    "      2           4204        4204    4204  4204    4204     4204   \n",
    "1     0            413         413     413   413     413      413   \n",
    "      1            814         814     814   814     814      814   \n",
    "      2            810         810     810   810     810      810   \n",
    "\n",
    "                Num Of Products  Has Credit Card  Is Active Member  \\\n",
    "Churn Geography                                                     \n",
    "0     0                  2064             2064             2064   \n",
    "      1                  1695             1695             1695   \n",
    "      2                  4204             4204             4204   \n",
    "1     0                   413              413              413   \n",
    "      1                   814              814              814   \n",
    "      2                   810              810              810   \n",
    "\n",
    "                Estimated Salary  Zero Balance  \n",
    "Churn Geography                                 \n",
    "0     0                  2064         2064  \n",
    "      1                  1695         1695  \n",
    "      2                  4204         4204  \n",
    "1     0                   413          413  \n",
    "      1                   814          814  \n",
    "      2                   810          810  "
   ]
  },
  "execution_count": 22,
  "metadata": {},
  "output_type": "execute_result"
 }
],
"source": [
 "df.groupby(['Churn', 'Geography']).count()"
]
},
{
"cell_type": "markdown",
"id": "8f4dfee5",
"metadata": {},
"source": [
 "# Define Label and Features"
]
},
{
"cell_type": "code",
"execution_count": 23,
"id": "d32bd416",
"metadata": {},
```

```
 "outputs": [
  {
   "data": {
    "text/plain": [
     "Index(['Surname', 'CreditScore', 'Geography', 'Gender', 'Age', 'Tenure',\n",
     "       'Balance', 'Num Of Products', 'Has Credit Card', 'Is Active Member',\n",
     "       'Estimated Salary', 'Churn', 'Zero Balance'],\n",
     "      dtype='object')"
    ]
   },
   "execution_count": 23,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "df.columns"
 ]
},
{
 "cell_type": "code",
 "execution_count": 24,
 "id": "61b78b91",
 "metadata": {},
 "outputs": [],
 "source": [
  "x = df.drop(['Surname', 'Churn'], axis = 1)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 25,
 "id": "0e3d321e",
 "metadata": {},
 "outputs": [],
 "source": [
  "y = df['Churn']"
 ]
},
{
 "cell_type": "code",
 "execution_count": 26,
 "id": "4dffe1c7",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
```

```
      "((10000, 11), (10000,))"
     ]
    },
    "execution_count": 26,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "x.shape, y.shape"
  ]
 },
 {
  "cell_type": "markdown",
  "id": "1f91fdc2",
  "metadata": {},
  "source": [
   "# Sampling The Data"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 27,
  "id": "1ae9467e",
  "metadata": {},
  "outputs": [
   {
    "data": {
     "text/plain": [
      "0    7963\n",
      "1    2037\n",
      "Name: Churn, dtype: int64"
     ]
    },
    "execution_count": 27,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "df['Churn'].value_counts()"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 28,
  "id": "07d4aca4",
  "metadata": {},
```

"outputs": [
{
"data": {
"text/plain": [
"<AxesSubplot:xlabel='Churn', ylabel='count'>"
]
},
"execution_count": 28,
"metadata": {},
"output_type": "execute_result"
},
{
"data": {
"image/png":
"iVBORw0KGgoAAAANSUhEUgAAAYsAAAEGCAYAAACUzrmNAAAAOXRFWHRTb2Z0d2
FyZQBNYXRwbG90bGliIHZlcnNpb24zLjQuMywgaHR0cHM6Ly9tYXRwbG90bGliLm9yZy/M
nkTPAAAACXBIWXMAAAsTAAALEwEAmpwYAAAUhElEQVR4nO3df5Bd5X3f8ffHYOzYTS
wBW4VIUBFbsQenMeAtkLrtOJYjBE0smtoUT11Uqhqhnl D5rGTacN9I8qgTC1p04xuA0ZTZAtP
ClYJiGoCWOqCruZTsOPJSaYH2G0xsaSBtAaAXZMTSL67R/32XARuzrXYs/uin2/Zu7cc77n
Oec8OyP7w3nOc89JVSFJ0pG8YaE7IEla/AwLSVInw0KS1MmwkCR1MiwkSZ2OX+gO9OH
kk0+u1atXL3JOmYcv/993+7qsZm2va6DIvVq1czMTGx0N2QpGNKkidm2+YwlCSpk2EhS
epkWEiSOhkWkqROhoUkqVOvYZHkXyd5OOMlDSW5O8uYkpye5J8J8Ikki8kOaG1fVNbn2zbVw
8d58pWfyzJ+X32WZL0ar2FRZKVwL8CxqvqJ4HjgEuATwLXBTBTW2XTcCzrX5ta0eS
M9p+7wbWA7+V5Li++i1JerW+h6GOB34yfhAW34oyfHAW4AngQ8At7bt24GL2vKjGtk7bvjZJWv2Wqn
qxqr4BTALn9NxvSdKQ3sKiqvYqvYDnwK+xSAkngfuB56rqkOt2T5T5gVexHt+x5q7U8ars+wz19
LsjnJRJKJqampuf+DJGkJ6+0X3EmWM7hL0+3KJ6+0X3EmWM7gqOB14Dvgig2GkXlTVVmArwPj4+Gt+o9N7/+1Nr7
lPev25/z9dutBdkBZEn8NQHwS+UVVTVfVXwO8D7wOOWtWEpgFXA/ra8HzgVoVoG1/G/DMcH
2GfSRJ86DPsPgWcF6St7R7D2uAR4B7gY+0NuuB29ryjrZO2/7HVeyeNvy5pBPYgTqT7x/ADu2txHdxNgpalyqX4v0t9ASZRxtd/rqVwS2Zl
5Lkg7T2zBUVd2T5FbgT4ZYbmqT4FDwFcZCDP9EXBLkt9otKnq4sqQ4qXcX
MluLyqXuqr35KkV+v1qbNVtQXYcjj5cWaYzVRV3wc+MstxrgGumfMOSpJG4i+4JUdA1JU
ifDQpLUybCQJHUyLCRJnQwLiyTvTPLA0OcVLSVnw0KS1MmwkCR1MiwkSZ0MC0k5LSad/LW/skuT7ZJZIHk
5w9dKyNrf2eJBtnP6skqQ+9hUVPVZVZ1bVmcB7gReA4Ax24gN1VtQbNbN1VtQbY3dYBLgDWtM9
AaAJCcyeDXruQxex7plOmAkSfNjvoah1gjO1jJr6ongwA3A3A9lbfDlzCyJKcA5wO7qup
gVT0L7ALWzk1O/JUnMX1hcAtzZ1lf4Ukm5NMJmMmpqam9Aof/HYPUWbYPW0k6UxVmMJmYmpqay
75L0pLXe1gkOQH4MPC/W1dH4EPDFw7d8d2m9f1p9WXW/xfVgBe7PNGJFRdKm47xSdo/4IkCdg5JDV8LuFBF6apVpgjDyc9GZJkfQIxGGEr29j4Jba+7sk+NyeVfi6z3NTQKt3GnkfcdJ7H1GsQo5jw5ZqufQ8jA6
CgVT0L7ALWzk1O/JUnMX1hcAtzZ1lf4Ukm5NMJmMmpqam9Aof/HYPUWbYPW0k6UxVmMJmYmpqay
75L0pLXe1gkOQH4EPDFw7d8d2m9f1p9WXYTNk/kIi4/y8hAUwE5gekbTRuD2ofqIbVbUecDzbbjqjUq
Xtxva6VpMkzZPj+zx4krcCPwv84ID5E8COJJuAJ4CLW/W4CLW/0O4EJgksHHMqcsAqupgkqB1q7q6
rqYJ/9IiS9Uq9hUVXfA046rPYMg9lRh7ct4PJZjrMN2NZHHyVJ3fwFtySpk2EhSepkWEiSOhk
WkqROhoUkqZNhIUnqZFhIkjoZFpKkToaFJKmTYSFJ6mRYSJI6GRaSpE6GhSSPpE6GhSSpk2EhSepk
WEiSOhkWkqROhoUkqVOvYZFkWZJbk/x5kkeT/HSSE5PsJjvbkD6P5zTtk4vHvycK5ANBwCaoKzJbk/s
Ta29nuSbJz9jKjKkPvR9ZXEd4Wqehm4HcRBZXBRe4P1WIYdIt4WxxKlYkIiTwp7bjWASSlyO4GlRaSpE6Ghi
gHGDLdMBIkuZHb2GR5G3A3wC2AGC5gDeMDSb8BbXh8BKqiv6yq6BwcAwP7UzMlJ6BfmxM7ANX
Kqqg1X1LLALWNLAWN9XvyVJVr9bnlcXpaAbbrXTpsBXx/ZeMwDANnAsNdqjGzgDpsIaDTtdTxbQcaj
TjKRZGJqamqO/xRJWtr6DIvjgbfGgbOBG6rqLOAV4KPz5QNHzu5cGQgfPDGIUSafJLgQu5r1pJVkS2EIxJiSBIkZ23UuAAiRl0EbVxtAnxgiCkb1fHdlkyzrV0gqUD4aVcGvJt2
1fYbFPmBfVd3T1m9lEB7fbsNLtO/PDm0/dWj/VcBTRypLkuZJb2FRVU8D30ryzlZaAzwC7ACmn2haD9zWlncAl7anos4Dnm/DVHcAa5Msbze217aaJGmeHN/z8X8J+FKSE4DHgcsYBNRXk2wEngAubm13AhcCk8ALrS1VdTDJ1cC9rd2
nGW1u7d2e9Lm+6zzQUB9l/e+ZLFsgzKQbW80B2Ayiiu1WYuoz6b07R9xj2ftnvKaABOrMm8E5wF/rdyOpIbmzKObNEdyOZn+o/Gm6ljwRKPlZmLA+nM8n9W7
YBTwAXt7Z3ABCcK8ALrS1VdTDJ1cB9rd1VVXWw535J0pd0DWqbuDGFYwb8KRXVD9TCAYNzrdbJEmQwcFrDZoWbBpCeiL6E6nwQa9zxGlNVrGyY2bLsgvkfzNm3VmMDcwXFtXgvQnNdXA4TbVV1TRZqtJpBGxYT0pfVH0WtvFBNlVedgqa8+B0CevSJ+
bBnU4yvG8C5ca1Y/Lsf6SvDNN1X0XhUKX9kbzfcszVwIK1ej+tuNXeBkeYDoN7a6TV7AQCP4Pr+2v7FHy9XqBGWs00PRGjLAMlUSkZLzLP4RO69THvHHHvhMhqmXbd4m6LLtTQrqjx7rb/TzwwaFmzlJXYGEJ7rO
YBTwAXt7Z3ABcCk8ALrS1VdTDJ1cB9rd1VVXWw535Lkob0GhZV9QAwPsOmtTO0LeDyDWy

Y6zDdg2p52TJI3MX3BLkjoZFpKkToaFJKmTYSFJ6mRYSJI6GRaSpE6GhSSpk2EhSepkW
EiSOhkWkqROhoUkqZNhIUnqZFhIkjoZFpKkToaFJKmTYSFJ6mRYSJI69RoWSb6Z5GtJHk
gy0WonJtmVZE/7Xt7qSXJ9kskkDyY5e+g4G1v7PUk2znY+SVI/5uPK4meq6syqmn696hXA7
qpaA+xu6wAXAGvaZzNwAwzCBdgCnAucA2yZDhhJ0vxYiGGoDcD2trwduGiofIMN3A0sS3I
KcD6wq6oOVtWzwC5g/Tz3WZKWtL7DooD/keT+JJtbbUVVPdmWnwJWtOWWVwN6hffe12m
z1V0iyOcIEkompqam5/Bskack7vufj/72q2p/kbwK7kvz58MaqqiQ1Fyeqqq3AVoDx8fE5OaYka
aDXK4uq2t++DwC3Mbjn8HQbXqJ9H2jN9wOnDu2+qtVmq0uS5kIvYZHkrUI+eHoZWAc8BO
wEpmc0bQRub8s7gUvbrKjzgOfbcNWdwLoky9uN7XWtJkmaJ30OQ60AbksyfZ7/VIVfSnIfsC
PJJuAJ4OLW/g7gQmASeAG4DKCqDia5Grivtbuqqg722G9J0mF6C4uqehx4zwz1Z4C1M9Q
LuHyWY20Dts11HyVJo/EX3JKkToaFJKmTYSFJ6mRYSJI6jRQWSXaPUpMkvT4dcTZUkjcD
bwFObr9xSNv0I8zwyA1J0utT19TZXwQ+DvwYcD8vh8V3gP/SX7ckSYvJEcOiOiqq4DrkvyS1X
1mXnqkyRpkRnpR3IV9ZkkfxdYPbxPVd3UU78kSYvISGGR5PPA04AN4CTB/P8zgJeK6qDrX1fbz8q
f9zEOnNEeySFJWmJG/Z3FQ8CP9tkRSdLiNeqVxcnAl5PcC7w4XayqD/fSK0nSojJqWPyrPjshSVrcRgqLqvqzvjsiSVq8Rp0N9V0Gs58ATgJOBL5XVT/SV8ckSYvHqFcWb51eThJgA3BeX52SJC0uR/3U2Rr4EwYvJpIkHQdGHYb6yNDqGxj87uIHvfRIkrTojDob6meHlg8B32QwFCVJOg6Mes/isr47IklavEZ9+dGqJLcl2d8+f5RkVd+dkyQtDqPe4P4DBm+y+7H2+eNWkyQdB0YNi7Gq+oOqOtQ+NwJjPfZLkrSIjBoWzyT5aJIT2uejwDN9dkyStHiMGha/DlwMPAU8CfwSA5vH/TUL0mLzKhTZ398GqE7P+zDhUCKqwFfbTaw9/F6eavg/8oZ+z8z+/pBNIXDKDewEhbnNtWI2xzXXHcHZ7C7o8HK3pvl+A0jLZpG/s1ZYDkYP5IHQ5bzXE2iIiP/8jZCYxTxHNr9brycrmJKdYx5XGKqUoIHvN6ZrkSANQ6TSWajKX4mIxZ/V5SZmzbHHprYQAxAXqDS9UwEQpYo52Dn15ZuYtQAeGuyoH+5LXLzwuZYqZyywqWRYlRhu+GufLrFCp60ncpl32ElNeimJbAdfdXCsvqFaVs8tXtyBCDkD4uvTmx09+uPXpxrOfDefb9dLDgl0vX/6U/TahKm8zBX3ECNszhA6JyRjTt1bKP/XzQ9Z3dkeLMKxIP47h5q2XP32B6bPm5zm2HxZ4yL/U7IkbzLw3OzDB83tH0Jo1i9aBW1ZTt4TUvyP8uTs87bnqCG8xzPI6d/mfJqpR3TGWF5MRmoAgTyYItv/j3/xwqmWHWaNz0oq6WyEY3UoLoWH1G9SJ9S/xyD3MWuSX5yRAgFVT1kV+OMbvxqVvsU+yzXBq/jNSvuEkWPTbLXr/tDAOPZ9+HGy98UbKJlXBYhlUqLqGr3fhV+XUFjpeUuS+hgR73+WDEpu9MsxbeZ2H2u7Xp5jzOdQqPFRlPx5vfsUvcs/wJyu2P1WnlHDRRxhGs4Qc84qTfbJXXjD4PAOIXzyaZ/qPKw5FHCP2lYL4wOcbQEd5g7yUVI/GO0IQ67CGvX9e/mkYvYAt//Ov8OjLXf6oivkh/AdqYS/qH7tjGlNqkjqQcb2MB1CaYFp+NbhBmoOZnb89+s+MXCfrBuxyMqdUjGYJoYgSIPmDcZ3Zo+nYZyuqSPJgf9BZeEV8UUL4pBmr9vgLQu3NHtH9+0/DfeE1C+OR/b/S6M+i4IJunkYsRvH6Wxnm3FbPWf9xxLIF3jT1PsdBJfxGTpTNz2h96mFbFQ+t1yx5iTDvCeW/i+R7H5+cC1/R6o79gFnKD4J16uepWLMn8/UXGVxF86SsaPd8AxMoImZ7Go/JB6M9vJCFQ2XaWAZkJ9gh2b/5MtDK4jUK29wKU8xOvvYEeSxvCXQK5vK9bC19GZaCGWZPtvVXNkwYUaS0/EVQ2/a/fwyScHI2XcMxmUuK8bGX4xKSx0MoVzfs5cyZFh5SlO0dR+o2KCAnCAHtFJyZFGJnOJ

31QDdwPLkpwCnA/sqqqDVfUssItXB5AkqWfzfc9iRVU92ZafAla05ZXA3qF2+1pttvqrJNmcZ
CLJxNTU1Nz2WpKWuAW7wV1VBdQcHm9rVY1X1fjY2NhcHVaSxPyHxdNteIn2faDV9wOn
DrVb1Wqz1SVJ82i+w2InMD2jaSNw+1D90jYr6jzg+TZcdSewLsnydmN7XatJkubR8X0dOMn
NwPuBk5PsYzCr6RPAjiSbgCeAi1vzO4ALgUngBeAygKo6mORq4L7W7qqqOvymuSSpZ72F
RVV9dJZNa2doW8DlsxxnG7BtDrsmSfoB+QtuSVInw0KS1MmwkCR1MiwkSZ0MC0lSJ8NC
ktTJsJAkdTlsJEmdDAtJUifDQpLUybCQJHUyLCRJnQwLSVInw0KS1MmwkCR1MiwkSZ0M
C0lSp2MmLJKsT/JYkskkVyx0fyRpKTkmwiLJccB/BS4AzgA+muSMhe2VJC0dx0RYAOcAk1
X1eFX9JXALsGGB+yRJS8bxC92BEa0E9g6t7wPOHW6QZDOwua3+RZLH5qlvS8HJwLcXu
hOLQT61caG7oFfy3+a0LZmLo/yt2TYcK2HRqaq2AlsXuh+vR0kmqmp8ofshHc5/m/PnWBm
G2g+cOrS+qtUkSfPgWAmL+4A1SU5PcgJwCbBzgfskSUvGMTEMVVWHkvxL4E7gOGBbV
T28wN1aShze02Llv815kqpa6D5Ikha5Y2UYSpK0gAwLSVInw0JH5GNWtBgl2ZbkQJKHFrov
S4VhoVn5mBUtYp8D1i90J5YSw0JH4mNWtChV1R8DBxe6H0uJYaEjmekxKysXqC+SFpBhI
UnqZFjoSHzMiiTAsNCR+ZgVSYBhoSOoqkPA9GNWHgV2+JgVLQZJbgb+BHhnkn1JNi10n1
7vfNyHJKmTVxaSpE6GhSSpk2EhSepkWEiSOhkWkqROx8Sb8qTFKMmPAp8G/g7wHPA08
AfAh6rq5xasY1IPvLKQjkKSALcBX6mqt1fVe4ErgRWv8bj+B5wWJf9hSkfnZ4C/qqrfni5U1Z8l
WQ6sTXIr8JPA/cDHqqqSfBMYr6pvJxkHPlVV70/ya8DbgR8HvXkMeC0tn4a8Omqun4+/zjp
cF5ZSEdnOghmchbwcQbvAPlx4H0jHO8M4INV9dG2/i7gfAaPid+S5I2vqbfSa2RYSHPv3qra
V1X/D3gAWD3CPjur6v8Orf9RVb1YVd8GDvAah7ek18qwkI7Ow8B7Z9n24tDyS7w83Hull/83
9+bD9vneiMeQFoRhIR2du4A3Jdk8XUjyU8DfP8l+TlgPnH/XVNmnuGhXQUavAEzn8EfD
DJ15M8DPxH4Kkj7PbrwHVJJhhcLUjHDJ86K0nq5JWFJKmTYSFJ6mRYSJI6GRaSpE6GhS
Spk2EhSepkWEiSOv1/LUDH8nPzzJEAAAAASUVORK5CYII=\n",
      "text/plain": [
       "<Figure size 432x288 with 1 Axes>"
      ]
     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    }
   ],
   "source": [
    "sns.countplot(x = 'Churn', data = df)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 29,
   "id": "6cb398ee",
   "metadata": {},
   "outputs": [
    {
     "data": {
      "text/plain": [
       "((10000, 11), (10000,))"
      ]
     },
     "execution_count": 29,

```
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "x.shape, y.shape"
 ]
},
{
 "cell_type": "markdown",
 "id": "e8a290bb",
 "metadata": {},
 "source": [
  "# Random UnderSampling"
 ]
},
{
 "cell_type": "code",
 "execution_count": 30,
 "id": "30d183eb",
 "metadata": {},
 "outputs": [],
 "source": [
  "from imblearn.under_sampling import RandomUnderSampler"
 ]
},
{
 "cell_type": "code",
 "execution_count": 31,
 "id": "98d6d00e",
 "metadata": {},
 "outputs": [],
 "source": [
  "rus = RandomUnderSampler(random_state = 2529)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 32,
 "id": "cd295471",
 "metadata": {},
 "outputs": [],
 "source": [
  "x_rus, y_rus = rus.fit_resample(x, y)"
 ]
},
{
 "cell_type": "code",
```

```
 "execution_count": 33,
 "id": "c03b9903",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "((4074, 11), (4074,), (10000, 11), (10000,))"
    ]
   },
   "execution_count": 33,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "x_rus.shape, y_rus.shape, x.shape, y.shape"
 ]
},
{
 "cell_type": "code",
 "execution_count": 34,
 "id": "ef2343f4",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "0    7963\n",
     "1    2037\n",
     "Name: Churn, dtype: int64"
    ]
   },
   "execution_count": 34,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "y.value_counts()"
 ]
},
{
 "cell_type": "code",
 "execution_count": 35,
 "id": "f19d5f7c",
 "metadata": {},
 "outputs": [
```

{
"data": {
"text/plain": [
"0   2037\n",
"1   2037\n",
"Name: Churn, dtype: int64"
]
},
"execution_count": 35,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"y_rus.value_counts()"
]
},
{
"cell_type": "code",
"execution_count": 36,
"id": "ac9dc5cb",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"<AxesSubplot:ylabel='Frequency'>"
]
},
"execution_count": 36,
"metadata": {},
"output_type": "execute_result"
},
{
"data": {
"image/png":
"iVBORw0KGgoAAAANSUhEUgAAAYsAAAD4CAYAAAAdIcpQAAAAOXRFWHRTb2Z0d2F
yZQBNYXRwbG90bGliIHZlcnNpb24zLjQuMywgaHR0cHM6Ly9tYXRwbG90bGliLm9yZy/Mn
kTPAAAACXBIWXMAAAsTAAALEwEAmpwYAAAV+klEQVR4nO3dfbRddX3n8fdH8IkKBZrI0
CQ04Aq0ERXhFpnIYHVoEXBKsJ2hsKo8DIuIwqxxZM0UbNdAdVhL2yIjU4vGkgWxyoNSJD
PFoYGxsjpLHi6QCU9SAgRJjOQWHGjFgYLf+ePsK8d4b/ZJcs8593Ler7XOunt/997nfDc35J
O9f/vsnapCkqRtedWwG5AkzX6GhSSSplWEhSWplWEiSWhkWkqRWuw67gX6gX6ZN29eLV68e
NhtSNKcccdddd/19Vc2fatkrNiwWL17M+Pj4sNuQpDkl+IQMZ5Ezi+jZbb2NSmq0/1OcuTjCcZn5iYmLkdkKQR
qRWhoUkqZVhIUlqZVhIkloZFpKkVoaFJKlV377BnWQRsBnWnewDXAYmADcG
JV/SBJgM8CxwHPPAadV1d3Ne50K/SBFHz1v+Iqq7sV98Ai8/7q36+/bQ2fOp9Q/IcSTPvIfb3SD+P
LF4Ezq2qpcARwNlJLgAbdU1RLgImYe4FhgSfNaDlw8HwBAMmDwyuBI4oZleBqyjuAPZPsC7wXWFNVT1fVD4A96
VNXmySODqvoH4EFgAbAMmDwyuBI4oZleBqyjtuAPZPsC7wXWFNVT1fVD4A1wDH96Iu
S9LMGMmaRZDHwduB2YO+q2tws+j6wdzO9AHi8a7ONTW26+iSVCRxHRRyBUeh
S9LMGMmaRZDHwduB2YJ+q2tws+j6d01TQCZInujbb2NSmq0/1OcuTjCcZn5iYmLkdkKQR

1/ewSPIG4Drgo1X1bPeyqio64xkzoqpWVNVYVY3Nnz/IXXYISTugr2GR5NV0guLLVfWXTfnJ
5vQSzc8tTX0TsKhr84VNbbq6JGIA+hYWzdVNlwMPVtVnuhatBk5tpk8Fbuiqn5KOI4BnmtNV
NwFHJ9mrGdg+uqIJkgaknw8/eifwQeDeJGub2seBTwHXJjkDeBw4sVI2I53LZtfTuXT2dICqejr
JJ4E7m/U+UVVP97FvSdJW+hYWVfW3QKZZfNQU6xdw9jTvtRJYOXPdSZK2h9/gIiS1Miwk
Sa0MC0ISK8NCktTKsJAktTIsJEmtDAtJUivDQpLUyrCQJLUyLCRJrQwLSVIrw0KS1MqwkC
S1MiwkSa0MC0ISK8NCktTKsJAkternM7hXJtmS5L6u2jVJ1javDZOPW02yOMmPupZ9vmub
w5Lcm2R9kkubZ3tLkgaon8/gvgL4U2DVZKGqfmdyOsnDwDNd6z9SVYdM8T6XAWcCt9N5T
vcxwDdmvI1J0nT6dmRRVbcCT0+1rDk6OBG4ealvvvkWRfYI+quq15Rvcq4IQZbIWS1GJYYxZ
HAk9W1cNdtf2T3JPkW0mObGoLgI1d62xsaINKsjzJeJLxiYmJme9akkbUsMLiZH76qGIzsF9
VvR34GPCVJHts75tW1YqqGquqsfnz589Qq5Kkfo5ZTCnJrsBvAYdN1qrqeeD5ZvquJI8ABwBP
bgIVdmy9sapKkARrGkcWvA9+pqp+cXkoyP8kuzuQBwBLg0acFZVTQ6OfwT4c2A98AheCSVJA9e301BVdfI0
9dOmqF0HXDfN+uPAbCjJxiRnNItO4mcHtt8FrGsupfBcFZVTQ6OfwT4c2A98AheCSVJA9e301BVdfI0
9dOmqF0HXDfN+uPAwTPanCRpu/gNbkISK8NCktTKsJAktTIsJEmtDAtJUivDQpLUyrCQJLU
yLCRJrQwLSVIrw0KS1MqwkCS1MiwkSa0MC0ISK8NCktTKsJAktTIsJEmt+vmkvJVJtiS5r6t2
YZJNSdY2r+O6Ip2fZH2Sh5K8t6t+TFNbn+S8fvUrSZpeP48srgCOmaj+SVUdZM0rxuBEiylM7jVt
/cbPNnSXZpnsv9OeBYYClwcrOuJGmA+vIY1VuTLO5pu/TLO5x9WXA1VX1PPBYkvXAsc2y9VX1KE
CSq5t1H5jpfiVJ0xvGmMiVW1dL7/Jsn2t1V7j9JZGIlmDDo
vLgDcBhwCbgYtn8s2ol2Pz582fyrSvppPXtNNRUqurJyekllwFQXwT+RzO7CVjUtd
Y26pKkARnokUWSfbpmfxOYvFJqNXBSkp2S7A8sAe4A7gSWJNk/yUw6g+CrB9mzJKmPRxZJvg68B9gryUbgAuA9SQ4BCtgAfASgqu5Pcg2dgeutwFlV9bPmfc4G/gSYB6ysqvv71bMkaWp9u89imIzGhurx8fFhtyFJs0qSu6pqbKpl3sEtSWplWEiSWhkWkqRWhoUkqZVhIUlqZVhIkloZFpKkVoaFJKmVYSFJamVYSJJaGRaSpFaGhSSplWEhSWplWEiSWhkWkqRWhoUkqZVhIUlqZVhIkloZFpKkVoaFJKmVYSFJamVYSJJaGRaSpFaGhSSplWEhSWrVt7BIsjLJliT3ddX+Y5L/nWRdkuuTzG3qi5P8Q5K1zesLXdscluTeJOuTXJok/epZkjS1fh5ZXAEc01NbAxxUVW8D/hI4v2vZI1V1SPP6aFf9MuBMYEnz6n1PSVKf9S0squpW4Ome2jeramszuwZY+HLvkWQfYI+quq2qCrgSOGGm2v00hPIOT/K8b+rYyuZ49pvkE8LcnP3fqknHSPWKsGP9s/TNzVf9/1cQCFUZI9YgZH9uvHCBPwFsjd29cw3FXP+tRJVMqFXSnm6fiB/4aSeFHyE0bnCJ8hVBCJDMiVR6Kn8qfZ4GHw9OH/+ndM5vTmFoIcpJwtMCCzN7hJtbgpqxyBxhHW3F/AWQoLoGIMkm2vEbf9c+9JXJoeyzGL4rdW00AbuTnMvXTxVIl/kVGPSPmWbnXNLH1OMmzFpYOiYN9P+jtnrw2Yb0zeJGrhTqnnaJeiKdk9N4/ujk2WNc4B+yLdsaqFFQ+CDaxCNaQZNtWq3sLr5rAhqsIFr5W1Uz99jbhaw+zZZQmVfT23xG+/PNd4h+iuzAKWFQyZdRIIdOJnG6NXjuE8zd5sb9DeHV3vGO3flW8zjDz5IVh0+jt86RPQpvuqSJa9MZtzvDCXgV7e0rHm+ROuVJvwB2oX6b5D7gsXs4uJPpwbTqDSqHtePzfy7Xcm5jlkbAb6exyPYyMB3fJ/dtN3y+ysT96UlGHEIdYpDcmPwP

fY/svmZUkzVG9hsULzQB0AST5uf61JEmabXoNi2uTfAHYM8mZwM34ICRJGhmtV0MlCXA
N8MvAs8BBwH+uqjV97k2SNEu0hkVz/6Ybq+otgAEhSSOo19NQdyf51b52IkmatXr9Bvc7gA
8k2UDniqjQOeh4a78akyTNHtsMiyT7VdV3gfcOqB9J0izUdmTxdTp3m308yXVV9dsD6EmSN
Mu0jVmka/qAfjYiSZq92sKippmWJI2QttNQb0vyLJ0jjNc30/DyAPcefe1OkjQrbDMsqmqXQTUi
SZq9tucW5TMiyUFJ1na9nk3y0SQXJtnUVT+ua5vzk6xP8lASr8ySpAHr9XsWM6aqHgIOAUi
yC7AJuJ7OY1Qvqao/6V4/yVLgJODNwC8CNyc5sKpeGmTfkjTKBn5ksZWjgEeq6vFtrLMMu
Lqqnq+qx+g8o/vwgXQnSQKGHxYnAVd1zZ+TZF2SlUn2amoLgCe61tnY1CRJAzK0sEjyGjo
PVPpqU7oMeBOdU1SbgYt34D2XJxlPMj4xMTFTrUrSyBvmkcWxwN1V9SRAVT1ZVS9V1Y/
pPCtj8lTTJmBR13YLm9rPqKoVVTVWVWPz58/vY+uSNFqGGRYn03UKKsm+XcveD9zXTK
8GTkry2iT7A0uAOwbWpSRp8FdDwU8ey/obwIe6yn+U5BA63xTfMLmsqu5Pci3wAPAicLZX
QknSYA0lLKrqh8AvbFX74DbWvwi4qN99SZKmNuyroSRJc4BhIUlqZVhIkloZFpKkVoaFJKm
VYSFJamVYSJJaGRaSpFaGhSSplWEhSWplWEiSWhkWkqRWhoUkqZVhIUlqZVhIkloZFpK
kVoaFJKnV0MIiyYYk9yZZm2S8qe2dZE2Sh5ufezX1JLk0yfok65IcOqy+JWkUDfvI4j1VdUhVj
TXz5wG3VNUS4JZmHuBYYEnzWg5cNvBOJWmEDTsstrYMuLKZvhI4oau+qjpuA/ZMsu8Q+
pOkkTTMsCjgr5PclWR5U9unqjY3098H9mmmFwBPdG27san9lCTLk4wnGZ+YmOhX35I0cn
Yd4mf/i6raIOSNwJok3+leWFWVpLbnDatqBbACYGxsbLu2lSRNb2hHFlW1qfm5BbgeOBx4c
vL0UvNzS7P6JmBR1+YLm5okaQCGEhZJfi7J7pPTwNHAfcBq4NRmtVOBG5rp1cApzVVRR
wDPdJ2ukiT12bBOQ+0DXJ9ksoevVNX/THIncG2SM4DHgROb9W8EjgPWA88Bpw++ZUka
XUMJi6p6FHjbFPWngKOmqBdw9gBakyRNYbZdOitJmoUMC0lSK8NCktTKsJAktTIsJEmtD
AtJUivDQpLUyrCQJLUyLCRJrQwLSVIrw0KS1MqwkCS1MiwkSa0MC0lSK8NCktTKsJAktRp
4WCRZlOSbSR5Icn+Sf9/UL0yyKcna5nVc1zbnJ1mf5KEk7x10z5l06obxpLwXgXOr6u7mOdx
3JVnTLLukqv6ke+UkS4GTgDcDvwjcnOTAqnppoF1L0ggb+JFFVW2uqrub6X8AHgQWbGO
TZcDVVfV8VT1G5znch/e/U0nSpKGOWSRZDLwduL0pnZNkXZKVSfZqaguAJ7o228g04ZJk
eZLxJOMTExP9aluSRs7QwiLJG4DrgI9W1bPAZcCbgEOAzcDF2/ueVbWiqsaqamz+/Pkz2a4
kjbShhEWSV9MJii9X1V8CVNWTVfVSVf0Y+CIvn2raBCzq2nxhU5MkDcgwroYKcDnwYFV9p
qu+b9dq7wfua6ZXAycleW2S/YElwB2D6leSNJyrod4JfBC4N8napvZx4OQkhwAFbAA+BFBV
9ye5FniAzpVUZ3sllCQN1sDDoqr+FsgUi27cxjYXARf1rSlJ0jb5DW5JUivDQpLUyrCQJLUyLC
RJrQwLSVIrw0KS1MqwkCS1MiwkSa0MC0lSK8NCktTKsJAktTIsJEmtDAtJUivDQpLUyrCQJ
LUyLCRJrQwLSVKrORMWSY5J8lCS9UnOG3Y/kjRK5kRYJNkF+BxwLLCUzvO6lw63K0kaH
XMiLIDDgfVV9WhVvQBcDSwbck+SNDJ2HXYDPVoAPNE1vxF4x9YrJVkOLG9m/zHJQzv4e
fOAv9/BbXdYPj3oT/wpQ9nnIRu1fR61/YUR3Od8eqf2+ZemWzBXwqInVbUCWLGz75NkvKr
GZqClOcN9fuUbtf0F93kmzZXTUJuARV3zC5uaJGkA5kpY3AksSbJ/ktcAJwGrh9yTJI2MOX
EaqqpeTHIOcBOwC7Cyqu7v40fu9KmsOch9fuUbtf0F93nGpKr68b6SpFeQuXIaSpI0RIaFJK
nVSIdF2y1Ekrw2yTXN8tuTLB5CmzOmh/39WJIHkqxLckuSaa+5nit6vU1Mkt9OUknm/GWWv
exzkhOb3/X9Sb4y6B5nWg9/tvdL8s0k9zR/vo8bRp8zJcnKJFuS3DfN8iS5tPnvsS7JoTv9oVU
1ki86A+WPAAcArwH+D7B0q3U+Any+mT4JuGbYffd5f98D7NZMf3gu72+v+9ystztwK3AbMD
bsvgfwe14C3APs1cy/cdh9D2CfVwAfbqaXAhuG3fdO7vO7gEOB+6ZZfhzwDSDAEcDtO/uZo
3xk0cstRJYBVzbTXwOOSpIB9jiTWve3qr5ZVc81s7fR+T7LXNbrbWI+CXwa+H+DbK5Petnn
M4HPVdUPAKpqy4B7nGm97HMBezTTPw98b4D9zbiquhV4ehurLANWVcdtwJ5J9t2ZzxzlsJj
qFilLplunql4EngF+YSDdzbxe9rfbGXT+ZTKXte5zc3i+qKr+apCN9VEvv+cDgQOT/O8ktyU5Z
mDd9Ucv+3wh8IEkG4EbgX83mNaGZnv/f281J75nocFK8gFgDPi1YffST0leBXwGOR2ztnn
nRORb2bztHjrUneUlX/d5hN9dnJwBVVdXGSfw58yTpJdqVz+
PrUQLqbeT3dMiXJrwO/DxxfVc8PqLd+advn3YGDgb9JsoHOud3Vc3yQu5ff80ZgdVX9U1U9
BvwdnfCYq3rZ5zOAawGAawGq6tvA6+6cZPCVasZvkTTKYdHLLURWA6c20/8a+F/jvjTOt0ln7
8LHAWQ5FfohMXEQLscrNXAKc1VUUcAz1TV5p15w5E9DVXT3ElkySeA8apaDVxO53B1P
Z3BpJOG1/HO6XF//xh4A/DVZhz/u1V1/NCa3kk97vMrSo/7fBNwdJIHgJeA/1hVc/20/8a+F/VjB7NQa37m+
TtwBfoBMVcP48NLftcVc9U1byqWlxVi+mM0xxfVePDaXdG9PLn+ut0jipIMo/OaalHB9jjTOtln7
8LHAWQ5FfohMXEQLscrNXAKc1VUUcAz1TV5p15w5E9DVXT3ElkySeA8apaDVxO53B1P
Z3BpJOG1/HO6XF//xh4A/DVZhz/u1V1/NCa3kk97vMrSo/7fBNwdJIHgJeA/1hVc/Wludd9Phf
4YpL/QGew+7Q5/A8/klxFJ/DnNeMwFwCvBqiqz9MZlzkOWA88B5y+0585h/97SZIGZJRPQ0

mSemRYSJJaGRaSpFaGhSSplWEhSWplWEiSWhkWkqRW/x8pxbKVCKJKaAAAAABJRU5ErkJggg==\n",
      "text/plain": [
       "<Figure size 432x288 with 1 Axes>"
      ]
     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    }
   ],
   "source": [
    "y_rus.plot(kind = 'hist')"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "30756ae5",
   "metadata": {},
   "source": [
    "# Random OverSampling"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 37,
   "id": "be55b4b5",
   "metadata": {},
   "outputs": [],
   "source": [
    "from imblearn.over_sampling import RandomOverSampler"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 38,
   "id": "13780001",
   "metadata": {},
   "outputs": [],
   "source": [
    "ros = RandomOverSampler(random_state = 2529)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 39,
   "id": "d9a4b5b5",

```
 "metadata": {},
 "outputs": [],
 "source": [
  "x_ros, y_ros = ros.fit_resample(x, y)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 40,
 "id": "6c198103",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "((15926, 11), (15926,), (10000, 11), (10000,))"
    ]
   },
   "execution_count": 40,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "x_ros.shape, y_ros.shape, x.shape, y.shape"
 ]
},
{
 "cell_type": "code",
 "execution_count": 41,
 "id": "cf4bd942",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "0    7963\n",
     "1    2037\n",
     "Name: Churn, dtype: int64"
    ]
   },
   "execution_count": 41,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "y.value_counts()"
```

]
},
{
"cell_type": "code",
"execution_count": 42,
"id": "59aec23e",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"1   7963\n",
"0   7963\n",
"Name: Churn, dtype: int64"
]
},
"execution_count": 42,
"metadata": {},
"output_type": "execute_result"
}
],
"source": [
"y_ros.value_counts()"
]
},
{
"cell_type": "code",
"execution_count": 43,
"id": "41129e4e",
"metadata": {},
"outputs": [
{
"data": {
"text/plain": [
"<AxesSubplot:ylabel='Frequency'>"
]
},
"execution_count": 43,
"metadata": {},
"output_type": "execute_result"
},
{
"data": {
"image/png":
"iVBORw0KGgoAAAANSUhEUgAAAYsAAAD4CAYAAAAdIcpQAAAAOXRFWHRTb2Z0d2F
yZQBNYXRwbG90bGliIHZlcnNpb24zLjQuMywgaHR0cHM6Ly9tYXRwbG90bGliLm9yZy/Mn
kTPAAAACXBIWXMAAAsTAAALEwEAmpwYAAAXTklEQVR4nO3de7RedX3n8fdHIiJWIUjM
MAk2dEy1qFXpEXCctirKzSlhVpXBVYeUlWW6WqZT284FnK6JA7KWrJlKZValpiXTwFS5VS

VTaWlErWtmlUu4FLmUyZGLJHI5JVyqVDD0O388v8hjyGE/kLOfk8N5v9Z61vPbv/3be383gX
zY1ydVhSRJz+UIs12AJGnPZ1hIkjoZFpKkToaFJKmTYSFJ6rRgtgvow4EHHIjLIi2b7TIkaU65
4YYb/q6qFu1q3osyLJYtW8amTZtmuwxJmlOS3DvdPE9DSZI6GRaSpE6GhSSpk2EhSepk
WEiSOvUaFkI+M8ltSW5N8vkk+yQ5JMm1SSaTXJJk7zb2ZW16ss1fNrSeM1r/nUmO6bNmS
dKz9RYWSZYA/w6YqKo3AXsBJwPAOdW1euAR4BVbZFVwCOt/9w2jiSHtuxCbJX
n3VLUI6tr5PQy0AXp5kAbAvcD/wHuDyNn89cGJrr2jTtPlHJUnrv7iqnqyqu4FJ4PCe65YkDekt
LKpqK/DfgW8zCInHgBuBAR6tqexu2BVjS2kuA+9qy29v4Vw/372KZH0qyOsmmJJumpqZmfoc
kaR7r7QnuJAsZHBUcAjwKXMbgNFIvqmotsBZgYmJit37RadnpX56Rmp6vez75/InZrqSZNVt/
h0B/f4/0eRrqvcDdVTVVVT8AvgC8E9i/nZYCWApsbe2twMEAbf5+wMPD/btYRpI0Bn2GxbeB
I5Ps2649HAXcDnwN+EAbsxK4orU3tGna/K/W4DdSe9nYaqqm
uTXA7cCGwHbmJwmujLwMVJPtH6LmiLXABcIGQS2MbgDiiq6rYklzIImu3AaVX1dF91S5Ker
de3zlbVGmDNTt13sYu7marq+8AHp1nP2cDZM16gJGkkPsEtSepkWEiSOhkWkqROhoUkqZ
NhIUnqZFhIkjoZFpKkToaFJKmTYSFJ6mRYSJI6GRaSpE6GhSSpk2EhSepkWEiSOhkWkqR
OhoUkqZNhIUnq1FtYJHI9kpuHPo8n+WiSA5JsTLK5fS9s45PkvCSTSW5JctjQula28ZuTrJx+
q5KkPvQWFIV1Z1W9tareCvwM8M8aYBjgOWt89q4HyAJAcw+GnWIxj8
HOuaHQEjSRqPcZ2GOgr4VlXdC6wA1rf+9cCJrb0bH2T3IQcAywwsaq2VdUjwEbg2D
HVLUIifGFxMvD51I5cVfe39gPA4tZeAtw3tMyW1jdd/49IsjsMyW1jdd/49IsjrJpiSbpqamZrJ2SZr3eg+LJHsDJw
CX7TyvqgqomdhOVa2tqomqmli0aNFMrFKS1IzjyOl44MaMaqerBNP9hOL9G+H2r9W4GDh5Zb
2vqm65ckjck4wuJDPHMKCmADsPMbTSoOppXAFUP9p7a7oo4EHmunq64BliVZ2C5sL2t9kqQxWdDnypO8AngP8NtD3Z8ELk2yCrgXOKn1XwWcAEwyuHPqwwBVtS3JWcANbdwnqmpbn3VLkn5U
ZEGfK0/yCuB9wK8MdSuBuDTJKuBe4KTWfyVwPDDJ4c6pUwGqaluaSs4Dr27gzq2pbn3VL
kn5Ur2FRVd8DXr1T38MMTT38MM7o7aeWwBp02znnXAuj5qICR18wIuSVInw0KS1MmwkCR1MiwkS
Z0MC0ISJ8NCktTJsJAkdTIsJEmdDAtJUifDQpLUybCQJHUyLCRJnQwLSVInw0KS1MmwkC
R1MiwkSZ0MC0ISp17DIsn+SS5P8rdJ7kjgiQHJNmYZHP7Xtj0GJslsSSaT3JLksH1rGzjNyd
ZOf0WJUI96PvI4tPAX1TVTwFvAe4ATgeurarlwLVtGuB4YHn7rAbOB0hyALAGOAI4HFiza
ZETCSpPHoLSyS7Af8HHABQFU9VVWPAiuA9W3YeuDE1l4BXFQD1wP7JzkIOAbYWFXbquoRYCNwbq
uoRYCNwbF91S5Kerc8ji0OAKeB/JrkpyR8R8leQWqWrGvD1OIRb39EXCYbQ2bTbX+ckGwWLc9b34+v8tgHhTWQzvKkubUbVlHadZAMPanPyXnAwCy
Yl2TQ1NTXDuyJJ81ufYbEAOw4Aw4v6reBnyXZ40xAvABcA61BlRV9bIIhYqklboa0XWKUDcZS3+KeUypatgqgacNmvlbjeLb+gd0SZSMzV3pz8jnTPV2pRfjRNRzJUy5RHnqSVqjWq3TqkSkLGqmqtrEWsUgqVYzgzJx8WAu8pNKhbmfQDGGrL6vg3Bfkb5ydm8izCv3vTdUcTwRlexzIuzSBmuwaRx6UmpxB9U8N/X6l9QtWv11A9WT6LkqulozXQDNiDuIqzaaQXcEwv
Iq+gyLLcCWqrq2TV/OIDwebWJ1O0jIAQRlGRVqiKBgXWCzLx3H0LQqIl8MN6JR6LVDCGfJNbk7KAvkl1RxNCGmVxQKZ11sN7XUwOwAhzPJbdd
A7LijaSVwRWtvAE5pd0UdTzWTIddBRydZGG7sH1065MkjcmCntf/68CfJNkbLkbuAs4IUFAXZ
pkFXAvcFIbeyVwPDAJPNHGUIXbkpwFXN/GnVlV23quW5OpNewqKqbgYldzDpqF2MLOG2
a9awD1s1ocZKkkfkEtySpk2EhSepkWEiSOhkWkqROhoUkqZNhIUnqZFhIkjoZFpKkToaFJK
mTYSFJ6mRYSJI6GRaSpE6GhSSpk2EhSepkWEiSOhkWkqROhoUkqVOvYZHkniTzTfHJzk
k2t74AkG5Nsbt8LW3+3SnqtnnCU8t9HwG+0QzgS+UhjG3P4Lm1Y6f
Vz0duLqqlgNXt2mA44DDQjiwiSL1Sbh1z+e3XDxPPnfbXPq2kjFzfp5c2x3d3hFDkC2Ehu2kJDt2Qr1QvVPzvb9M4JNbcjR2bOuOX1nqTswr3fVGRCOAADmfuNEQuuj6xDQkaqIyjIv3MmlSPehwGhUX
39rrgROH+i+sgWuA/ZMcBBwDbKyqbVX1CLAROHY3a5AkPQ8jhUVVUVS/SzwS8DBwA1JPpfkfyfkf
aMsCvxIkhuSrG59i6vq/tZ+AFjc2kuA+9Qz8i6vq/tZ+AFjc2kuA+4aW3dL6puv/EUlWJ9mUZNPU1NQouyVJGtGCUQd
W1eYkvwNsAs4D3pYkwMeq6gvTLPrXPjE7U7JO
SdLAqNcsfjrJucAdwd74HuAX6iqE5JcU6I9EF6FNBRhyhD1ZOQl2aH4cUL7sC7daWZFp/5oZF6uAGFJ8/bKOp3FJjY32jW6SVyBw5S28GqRVzoR
0JPNZOV10FHJ1kYbuwfXTrkySNyaino4P/ENVPQ2Q5CXAPlX1vi5n4KDeVHu+NQ0Q5CXAPlX1yZzrKnU9R5Gd4+vnhQzpUsd1RFX9Dds5UsQD
4XFX9RZLrgUuTrALuBU5q468EbGmFCA47gzAaezUGqaluSs4Dr27gzq2rb89lJSdLuGTUsvgK8F/
um94X+Evgn0+3QFXdBbxlF/0PQ2c9SrdM2bF3KCZoAmi8dY4kSPc3vn4fapAqqpVw8Jv/0eKPbz9nSq9
zahh8b2dnqj+GeAf+iIJkrSnGfU01Ee5By5J8BwjT4B/3VdRkqQ9y0hyhUVXXJ3kwhb2wD8Pz9QlL9NGFj4aHmEc
/aC/siRJe5KRH8oD3g4sa8sdXlb4g4vD4R5aNQiwOB25NcBzy5o7O7TuilKknSHmXmz94
6+1dJfhxYXIVfSbIvsFe/pUmS9hSjwjR1LP4qyceAI44D9IMTgNX
0VJUnas4waFk9W1RPhSRpHhg1hLP4qyceAI7ff3r4M+N/9ISVJ2pOMeOMGhanA1PA
N4FfYfBDRbv8hTxTJ0ovPqHdD/SPwh+0jSZnRr0b6u4kd+GXHZvZLcWpnOTMp2vQhSa5NMp

nkkiR7t/6XtenJNn/Z0DrOaP13JjnmBeynJGk3jHoaaoLBW2ffDvwscB7wv0Zc9jeAO4amzwH
OrarXAY8Aq1r/KuCR1n9uG0eSQ4GTgTcCxwKfSeIzHpI0RiOFRVU9PPTZWlW/B7y/a7kkS9
u4P2rTAd7D4JkNgPXAia29ok3T5h/Vxq8ALq6qJ6vqbmASOHyUuiVJM2PUV5QfNjT5EgZH
GqMs+3vAfwRe2aZfDTxaVdvb9BYGD/jRvu8DqKrtSR5r45cA1wytc3iZ4RpXA6sBXvva145Q
miRpVKO+G+p3h9rbgXuAk55rgST/Enioqm5I8q4XUtzzUVVrgbUAExMTPgMiSTNo1Luh3v0
C1v1O4IQkxwP7AK8CPg3sn2RBO7pYCmxt47cCBwNb2kN/+wEPD/XvMLyMJGkMRj0N9V
vPNb+qPrWLvjOAM9ry7wL+fVX9UpLLgA8AFwMrgSvaSvaIhva9F+3+V+tqkqyAfhckk8B/xRYDI
w3St2SpJnxfH4p7+0M/kIH+AUGf2FvfgHb/E/AxUk+AdwEXND6LwAuSjIJbGNwBxRVdVuSS
4HbGZwCO62qnn72aiVJfRk1LJYCh1XV3wMk+Tjw5ar68Xga+39CgLV9XXga+39I3s4m6mqvo+8MF
plj8bOHvEWiVJM2zU5ywWA08NTT/V+iRJ88CoRxYXAtcI+WKbPpFnnomQJL3IjXo31NlJ/pz
B09sAp1bVTf2VJUnak4x6GgpgX+Dxqvo0g9bD+mpJknSHmbUFwmuYXAX0xmt66WM/m4
oSdIcN+qRxb8CTgC+B1BV3+GZV3hIkI7kRg2Lp6qqaD+ImuQV/ZUkSdrTjBoWIyb5LINXdX
wE+Ar+EJIkzRudd0O114RfArwBeBx+i/sXwBFyJzBfqmpjz7VJkvYQnWdqaDGvV9qaD+ImuQV/ZUkSdrTjBoWIyb5LINXdX
wE+Ar+EJIkzRudd0O114RfArwBeBx4PfBfqmpjz7VJkvYQnWHR3s90ZVW9GTAgJGkeGvU0
1I1J3t5rJZKkPdaoT3AfAXw4yT0M7ogKg4OOn+6rMEnSnuM5wyLJa6vqL4ALa6vq28AxY6pHkrQH6jq
y+BKDt83em+RPq+oXx1CTCTJGkP03XNIkPtn+izEEnSnqsrLGqatiRpHuk6DfWWJI8zOMJ4e
WvDMxe4X9VrdZKkPcJzHIlU1V5V9aqqemVVLWjtHdPPGRRJ9klyXZK/SXJbkv/a+g9Jcm2S
ySSXJNm79b+sTU+2+cuG1nVbgLcCxyY5EjgHOLeqXg
c8Aqxq41cBj7T+c9s4khzK4Pe43wgcC3wmyV491i1J2kIvYVED322TL22fAt4DXN761zP41
AFTzz63uXA0e1V42sAC6uqier6m5gkI38hrckqT99HImQZK8kNwMPMXhVyLeAR6tq
jS2kuA+wDa/MeAVw/372KZ4W2tTrIpyaapqake9kaS5q9ew6Kqnq6qtwJLGRwNvGRwNvKHHba2tqo
mqmli0aFFm5GkeanXsNihqh4Fvga8g8FrznfchbUU2Nra3k8FRygQ8UdDbYSV3ccf
ski5Ls39ovB94H3MEgND7Qhq0ErmjtDW2aNv+r7QeXNgAnt7uIDgGWA9f1Vbck6dIGfZHgC
3EQsL7dufQS4NKq+rMktwMXJ/kEcBNwQRt/AXBRkklgG4M7oKiq25JcCtwObAdOq6ne6x
bkrST3sKiqm4B3raL/rvYxd1MVfV94IPTrOts4OyZrIGSNJqxXLOQJM1thoUkqZNhIUnqZFhIkj
oZFpKkToaFJKmTYSFJ6mRYSJI6GRaSpE6GhSSpk2EhSepkWEiSOhkWkqROhoUkqZNhI
UnqZFhIkjoZFpKkTn3+BvfBSb6W5PYkdZ/QJKNSTa374WtP0nOSzKZ5JYkhw2ta2Ubv
znJyum2KUnqR59HFtuB366qQ4EjgdOSHAqcAXy1qpYDX23TAMcBy9tnFXA+DMIFOAs4AsFp
Pg51jU7AkaSNB69hUAVA6Nmw9cGJrrwAurIFrgP2P2THAQcAycA2ysq
m1V9QiwETi2r7olSc82lmsWZYBwOuBRZX1f1t1gPA4tZ+CDBUm+Y1VUzTnl2nC41jNbKOdY+bKjKb5
1NTM7sDkjTP9R4WSX4M+FPgo1X1+PC8qioGLeqSG0b8fAgE8dm9DjWWfV4aT8rdNBjo2GHc
YOg+JOq+kLrfrCdXqJ9P9T6twIHDy2+tPVN1y9JGpM+YKcAFwR1V9amwaWBmDHHU0rgS
uG+k9pd0UdCTzWTIddBRydZGG7sH1065MkjcmCHtf9TuDfAN9McnPr+xjxXeDSJKuuAe4G
T2rwrgeOBSeAJ4FSAqtW5Czg+jbuzvZ+L+hjbRXuL2sOb++N0xLHlyRn82aUdHcnQGxt8s1/rKv+vKf/vKqfchwu6gEJnvp52dkvhcLf/+k12p18mMsDrEi1l9bjqLbkJqBh
PVSZKeD5/gliR1MiwkSZ0MC0ISJ8NCktTJsJAkdTIsJEmdDAtJUifDQpLUybCQJHUyLCRJn
QwLSVInw0KS1MmwkCR1MiwkSZ0MC0ISJ8NCktTJsJAkderzN7jXJXkoya1/te
2PqT5Lwkk0IuSXLY0DIr2/jNSVbuaIuSpH71eWTxuvv/MvQt2qHGNzl0QjdEfCwrWmbaGJ
GiyxKgg6Dl0gFgPt8UJbYWQtW2kPTtMUrrQSh4m6SDr4i0GUJ5h+TbmXXiUfyerfmz7OtcOVYYrXn
MpgUuTrAw1Dr7

WktuT3JLk6iTT3nM9V4z6mpgkv5ikksz52yxH2eckJ7U/69uSfG7cNc60Ef7dfm2SryW5qf37ffx
s1DlTkqxL8lCSW6eZnyTntX8etyQ5bLc3WlXz7sPgIvm3gJ8A9gb+Bjh0pzG/BvxBa58MXDLb
dY9hn98N7Nvavzof9rmNeyXwDeAaYGK26x7Dn/Ny4CZgYZt+zWzXPYZ9Xgv8amsfCtwz23
Xv5j7/HHAYcOs0848H/hwIcCRw7e5uc74eWYzy+pAVwPrWvhw4KknGWONM69znqvpaVT
3RJq9h8DzLXDbqa2LOAs4Bvj/O4noyyj5/BPj9qnoEoKoeGnONM22UfS7gVa29H/CdMdY34
6rqG8C25xiyAriwBq4B9k9y0O5sc76Gxa5eH7JkujFVtR14DHj1WKrrxyj7PGwVg/8zmcs697k
dnh9cVV8eZ2E9GuXP+SeBn0zyf5Nck+TYsVXXj1H2+ePAh5NsAa4Efn08pc2a5/vfe6c58Zy
FxivJh4EJ4Odnu5Y+JXkJ8Cngl2e5lHFbwOBU1LsYHD1+I8mbq+rR2SyqZx8C/riqfjfJO4CLkr
ypqv5xtgubK+brkcUorw/54ZgkCxgcuj48lur6MdIrU5K8F/jPwAlV9eSYautL1z6/EngT8PUk9zA
4t7thjl/kHuXPeQuwoap+UFV3A/+PQXjMVaPs8yrgUoCq+mtgHwYvGXyxmvFXJM3XsBjl9SE
bgJWt/QHgq9WuHM1Rnfuc5G3AZxkExVw/jw0d+1xVj1XVgVW1rKqWMbhOc0JVbZqdcmfE
KP9uf4nBUQVJDmRwWuquMdY400bZ528DRwEk+SkGYTE11irHawNwSrsr6kjgsaq6f3dW
OC9PQ9U0rw9Jciawqao2ABcwOFSdZHAh6eTZq3j3jbjP/w34MeCydi3/21V1wqwVvZtG3Oc
XlRH3+Srg6CS3A08D/6Gq5uxR84j7/NvAHyb5TQYXu395Lv/PX5LPMwj8A9t1mDXASwGq6
g8YXJc5HpgEngBO3e1tzuF/XpKkMZmvp6EkSc+DYSFJ6mRYSJI6GRaSpE6GhSSpk2EhS
epkWEiSOv1/euBuZREVkgAAAAASUVORK5CYII=\n",
      "text/plain": [
       "<Figure size 432x288 with 1 Axes>"
      ]
     },
     "metadata": {
      "needs_background": "light"
     },
     "output_type": "display_data"
    }
   ],
   "source": [
    "y_ros.plot(kind = 'hist')"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "28a4438d",
   "metadata": {},
   "source": [
    "# Train Test Split"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 44,
   "id": "8ab9c13f",
   "metadata": {},
   "outputs": [],
   "source": [
    "from sklearn.model_selection import train_test_split"
   ]
  },

```
 {
  "cell_type": "markdown",
  "id": "7bb0a2f3",
  "metadata": {},
  "source": [
   "# Split Original Data"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 45,
  "id": "ed463468",
  "metadata": {},
  "outputs": [],
  "source": [
   "x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.3, random_state =
2529)"
  ]
 },
 {
  "cell_type": "markdown",
  "id": "9626e83f",
  "metadata": {},
  "source": [
   "# Split Random Under Sample Data"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 46,
  "id": "592a083c",
  "metadata": {},
  "outputs": [],
  "source": [
   "x_train_rus, x_test_rus, y_train_rus, y_test_rus = train_test_split(x_rus, y_rus, test_size =
0.3, random_state = 2529)"
  ]
 },
 {
  "cell_type": "markdown",
  "id": "f1473ec7",
  "metadata": {},
  "source": [
   "# Split Random Over Sample Data"
  ]
 },
 {
  "cell_type": "code",
```

```
  "execution_count": 47,
  "id": "5c232f8a",
  "metadata": {},
  "outputs": [],
  "source": [
   "x_train_ros, x_test_ros, y_train_ros, y_test_ros = train_test_split(x_ros, y_ros, test_size =
0.3, random_state = 2529)"
  ]
 },
 {
  "cell_type": "markdown",
  "id": "9a7f692b",
  "metadata": {},
  "source": [
   "# Standardize Features"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 48,
  "id": "3c366c7a",
  "metadata": {},
  "outputs": [],
  "source": [
   "from sklearn.preprocessing import StandardScaler"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 49,
  "id": "e1aeb20d",
  "metadata": {},
  "outputs": [],
  "source": [
   "sc = StandardScaler()"
  ]
 },
 {
  "cell_type": "markdown",
  "id": "7b3a51a6",
  "metadata": {},
  "source": [
   "# Standardize Original Data"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 50,
```

```
   "id": "c6f2e144",
   "metadata": {},
   "outputs": [],
   "source": [
    "x_train[['CreditScore', 'Age', 'Tenure', 'Balance', 'Estimated Salary']] =
sc.fit_transform(x_train[['CreditScore', 'Age', 'Tenure', 'Balance', 'Estimated Salary']])"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 51,
   "id": "f0b29530",
   "metadata": {},
   "outputs": [],
   "source": [
    "x_test[['CreditScore', 'Age', 'Tenure', 'Balance', 'Estimated Salary']] =
sc.fit_transform(x_test[['CreditScore', 'Age', 'Tenure', 'Balance', 'Estimated Salary']])"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "bb736d39",
   "metadata": {},
   "source": [
    "# Standardize Random Under Sample Data"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 52,
   "id": "ddd5551c",
   "metadata": {},
   "outputs": [],
   "source": [
    "x_train_rus[['CreditScore', 'Age', 'Tenure', 'Balance', 'Estimated Salary']] =
sc.fit_transform(x_train_rus[['CreditScore', 'Age', 'Tenure', 'Balance', 'Estimated Salary']])"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 53,
   "id": "67a368f6",
   "metadata": {},
   "outputs": [],
   "source": [
    "x_test_rus[['CreditScore', 'Age', 'Tenure', 'Balance', 'Estimated Salary']] =
sc.fit_transform(x_test_rus[['CreditScore', 'Age', 'Tenure', 'Balance', 'Estimated Salary']])"
   ]
```

```
  },
  {
   "cell_type": "markdown",
   "id": "75d603df",
   "metadata": {},
   "source": [
    "# Standardize Random Over Sample Data"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 54,
   "id": "bc0893d8",
   "metadata": {},
   "outputs": [],
   "source": [
    "x_train_ros[['CreditScore', 'Age', 'Tenure', 'Balance', 'Estimated Salary']] =
sc.fit_transform(x_train_ros[['CreditScore', 'Age', 'Tenure', 'Balance', 'Estimated Salary']])"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 55,
   "id": "235d469c",
   "metadata": {},
   "outputs": [],
   "source": [
    "x_test_ros[['CreditScore', 'Age', 'Tenure', 'Balance', 'Estimated Salary']] =
sc.fit_transform(x_test_ros[['CreditScore', 'Age', 'Tenure', 'Balance', 'Estimated Salary']])"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "268b82f0",
   "metadata": {},
   "source": [
    "# Support vector machine Classifier"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 56,
   "id": "35830bc2",
   "metadata": {},
   "outputs": [],
   "source": [
    "from sklearn.svm import SVC"
   ]
```

```
},
{
 "cell_type": "code",
 "execution_count": 57,
 "id": "d439e53b",
 "metadata": {},
 "outputs": [],
 "source": [
  "svc = SVC()"
 ]
},
{
 "cell_type": "code",
 "execution_count": 58,
 "id": "4b4b329f",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "SVC()"
    ]
   },
   "execution_count": 58,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "svc.fit(x_train, y_train)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 59,
 "id": "194bb8cf",
 "metadata": {},
 "outputs": [],
 "source": [
  "y_pred = svc.predict(x_test)"
 ]
},
{
 "cell_type": "markdown",
 "id": "ce8cfdc5",
 "metadata": {},
 "source": [
  "# Model Accuracy"
```

```
 ]
},
{
 "cell_type": "code",
 "execution_count": 60,
 "id": "3e1ce831",
 "metadata": {},
 "outputs": [],
 "source": [
  "from sklearn.metrics import confusion_matrix, classification_report"
 ]
},
{
 "cell_type": "code",
 "execution_count": 61,
 "id": "1cf3b579",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "array([[2381,   33],\n",
     "       [ 436,  150]], dtype=int64)"
    ]
   },
   "execution_count": 61,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "confusion_matrix(y_test, y_pred)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 62,
 "id": "04053ee4",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "              precision    recall  f1-score   support\n",
    "\n",
    "           0       0.85      0.99      0.91      2414\n",
    "           1       0.82      0.26      0.39       586\n",
```

```
      "\n",
      "    accuracy                           0.84      3000\n",
      "   macro avg       0.83      0.62      0.65      3000\n",
      "weighted avg       0.84      0.84      0.81      3000\n",
      "\n"
     ]
    }
   ],
   "source": [
    "print(classification_report(y_test, y_pred))"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "ca48c401",
   "metadata": {},
   "source": [
    "# Hyperparameter Tunning"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 63,
   "id": "1338af67",
   "metadata": {},
   "outputs": [],
   "source": [
    "from sklearn.model_selection import GridSearchCV"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 64,
   "id": "c8da9523",
   "metadata": {},
   "outputs": [],
   "source": [
    "param_grid = {'C' : [0.1, 1, 10], 'gamma' : [1, 0.1, 0.01], 'kernel' : ['rbf'], 'class_weight' :
['balanced']}"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 65,
   "id": "27784340",
   "metadata": {},
   "outputs": [],
   "source": [
```

```
   "grid = GridSearchCV(SVC(), param_grid, refit = True, verbose = 2, cv = 2)"
  ]
 },
 {
  "cell_type": "code",
  "execution_count": 66,
  "id": "dceebb46",
  "metadata": {},
  "outputs": [
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "Fitting 2 folds for each of 9 candidates, totalling 18 fits\n",
     "[CV] END ..C=0.1, class_weight=balanced, gamma=1, kernel=rbf; total time=   1.3s\n",
     "[CV] END ..C=0.1, class_weight=balanced, gamma=1, kernel=rbf; total time=   1.3s\n",
     "[CV] END C=0.1, class_weight=balanced, gamma=0.1, kernel=rbf; total time=   0.9s\n",
     "[CV] END C=0.1, class_weight=balanced, gamma=0.1, kernel=rbf; total time=   0.9s\n",
     "[CV] END C=0.1, class_weight=balanced, gamma=0.01, kernel=rbf; total time=
1.0s\n",
     "[CV] END C=0.1, class_weight=balanced, gamma=0.01, kernel=rbf; total time=
1.0s\n",
     "[CV] END ....C=1, class_weight=balanced, gamma=1, kernel=rbf; total time=   1.0s\n",
     "[CV] END ....C=1, class_weight=balanced, gamma=1, kernel=rbf; total time=   1.1s\n",
     "[CV] END ..C=1, class_weight=balanced, gamma=0.1, kernel=rbf; total time=   0.8s\n",
     "[CV] END ..C=1, class_weight=balanced, gamma=0.1, kernel=rbf; total time=   0.8s\n",
     "[CV] END .C=1, class_weight=balanced, gamma=0.01, kernel=rbf; total time=   0.9s\n",
     "[CV] END .C=1, class_weight=balanced, gamma=0.01, kernel=rbf; total time=   0.9s\n",
     "[CV] END ...C=10, class_weight=balanced, gamma=1, kernel=rbf; total time=   1.0s\n",
     "[CV] END ...C=10, class_weight=balanced, gamma=1, kernel=rbf; total time=   1.0s\n",
     "[CV] END .C=10, class_weight=balanced, gamma=0.1, kernel=rbf; total time=   0.8s\n",
     "[CV] END .C=10, class_weight=balanced, gamma=0.1, kernel=rbf; total time=   0.8s\n",
     "[CV] END C=10, class_weight=balanced, gamma=0.01, kernel=rbf; total time=   0.8s\n",
     "[CV] END C=10, class_weight=balanced, gamma=0.01, kernel=rbf; total time=   0.8s\n"
    ]
   },
   {
    "data": {
     "text/plain": [
      "GridSearchCV(cv=2, estimator=SVC(),\n",
      "             param_grid={'C': [0.1, 1, 10], 'class_weight': ['balanced'],\n",
      "                         'gamma': [1, 0.1, 0.01], 'kernel': ['rbf']},\n",
      "             verbose=2)"
     ]
    },
    "execution_count": 66,
    "metadata": {},
    "output_type": "execute_result"
```

```
  }
 ],
 "source": [
  "grid.fit(x_train, y_train)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 67,
 "id": "fbdbc65b",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "SVC(C=10, class_weight='balanced', gamma=1)\n"
   ]
  }
 ],
 "source": [
  "print(grid.best_estimator_)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 68,
 "id": "194cee17",
 "metadata": {},
 "outputs": [],
 "source": [
  "grid_predictions = grid.predict(x_test)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 69,
 "id": "f2759f00",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "array([[2159,  255],\n",
     "       [ 343,  243]], dtype=int64)"
    ]
   },
   "execution_count": 69,
```

```
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "confusion_matrix(y_test, grid_predictions)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 70,
 "id": "ab7940f0",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "              precision    recall  f1-score   support\n",
    "\n",
    "           0       0.86      0.89      0.88      2414\n",
    "           1       0.49      0.41      0.45       586\n",
    "\n",
    "    accuracy                           0.80      3000\n",
    "   macro avg       0.68      0.65      0.66      3000\n",
    "weighted avg       0.79      0.80      0.79      3000\n",
    "\n"
   ]
  }
 ],
 "source": [
  "print(classification_report(y_test, grid_predictions))"
 ]
},
{
 "cell_type": "markdown",
 "id": "af158ff3",
 "metadata": {},
 "source": [
  "# Model with Random Under Sampling"
 ]
},
{
 "cell_type": "code",
 "execution_count": 71,
 "id": "ddb101a1",
 "metadata": {},
 "outputs": [],
```

```
 "source": [
  "svc_rus = SVC()"
 ]
},
{
 "cell_type": "code",
 "execution_count": 72,
 "id": "510e180f",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "SVC()"
    ]
   },
   "execution_count": 72,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "svc_rus.fit(x_train_rus, y_train_rus)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 73,
 "id": "769da95e",
 "metadata": {},
 "outputs": [],
 "source": [
  "y_pred_rus = svc_rus.predict(x_test_rus)"
 ]
},
{
 "cell_type": "markdown",
 "id": "58c7bb0c",
 "metadata": {},
 "source": [
  "# Model Accuracy"
 ]
},
{
 "cell_type": "code",
 "execution_count": 74,
 "id": "d4dbedfc",
 "metadata": {},
```

```
 "outputs": [
  {
   "data": {
    "text/plain": [
     "array([[470, 157],\n",
     "       [174, 422]], dtype=int64)"
    ]
   },
   "execution_count": 74,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "confusion_matrix(y_test_rus, y_pred_rus)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 75,
 "id": "95246dac",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "              precision    recall  f1-score   support\n",
    "\n",
    "           0       0.73      0.75      0.74       627\n",
    "           1       0.73      0.71      0.72       596\n",
    "\n",
    "    accuracy                           0.73      1223\n",
    "   macro avg       0.73      0.73      0.73      1223\n",
    "weighted avg       0.73      0.73      0.73      1223\n",
    "\n"
   ]
  }
 ],
 "source": [
  "print(classification_report(y_test_rus, y_pred_rus))"
 ]
},
{
 "cell_type": "markdown",
 "id": "16ecfe2f",
 "metadata": {},
 "source": [
```

```
    "# Hyperparameter Tunning"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 76,
   "id": "4e84f039",
   "metadata": {},
   "outputs": [],
   "source": [
    "param_grid = {'C' : [0.1, 1, 10], 'gamma' : [1, 0.1, 0.01], 'kernel' : ['rbf'], 'class_weight' :
['balanced']}"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 77,
   "id": "6c5f0e36",
   "metadata": {},
   "outputs": [],
   "source": [
    "grid_rus = GridSearchCV(SVC(), param_grid, refit = True, verbose = 2, cv = 2)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 78,
   "id": "3e8f5290",
   "metadata": {},
   "outputs": [
    {
     "name": "stdout",
     "output_type": "stream",
     "text": [
      "Fitting 2 folds for each of 9 candidates, totalling 18 fits\n",
      "[CV] END ..C=0.1, class_weight=balanced, gamma=1, kernel=rbf; total time=   0.2s\n",
      "[CV] END ..C=0.1, class_weight=balanced, gamma=1, kernel=rbf; total time=   0.1s\n",
      "[CV] END C=0.1, class_weight=balanced, gamma=0.1, kernel=rbf; total time=   0.1s\n",
      "[CV] END C=0.1, class_weight=balanced, gamma=0.1, kernel=rbf; total time=   0.1s\n",
      "[CV] END C=0.1, class_weight=balanced, gamma=0.01, kernel=rbf; total time=
0.1s\n",
      "[CV] END C=0.1, class_weight=balanced, gamma=0.01, kernel=rbf; total time=
0.1s\n",
      "[CV] END ....C=1, class_weight=balanced, gamma=1, kernel=rbf; total time=   0.1s\n",
      "[CV] END ....C=1, class_weight=balanced, gamma=1, kernel=rbf; total time=   0.1s\n",
      "[CV] END ..C=1, class_weight=balanced, gamma=0.1, kernel=rbf; total time=   0.0s\n",
      "[CV] END ..C=1, class_weight=balanced, gamma=0.1, kernel=rbf; total time=   0.1s\n",
      "[CV] END .C=1, class_weight=balanced, gamma=0.01, kernel=rbf; total time=   0.1s\n",
```

```
    "[CV] END .C=1, class_weight=balanced, gamma=0.01, kernel=rbf; total time=   0.1s\n",
    "[CV] END ...C=10, class_weight=balanced, gamma=1, kernel=rbf; total time=   0.1s\n",
    "[CV] END ...C=10, class_weight=balanced, gamma=1, kernel=rbf; total time=   0.1s\n",
    "[CV] END .C=10, class_weight=balanced, gamma=0.1, kernel=rbf; total time=   0.0s\n",
    "[CV] END .C=10, class_weight=balanced, gamma=0.1, kernel=rbf; total time=   0.1s\n",
    "[CV] END C=10, class_weight=balanced, gamma=0.01, kernel=rbf; total time=   0.0s\n",
    "[CV] END C=10, class_weight=balanced, gamma=0.01, kernel=rbf; total time=   0.1s\n"
   ]
  },
  {
   "data": {
    "text/plain": [
     "GridSearchCV(cv=2, estimator=SVC(),\n",
     "             param_grid={'C': [0.1, 1, 10], 'class_weight': ['balanced'],\n",
     "                         'gamma': [1, 0.1, 0.01], 'kernel': ['rbf']},\n",
     "             verbose=2)"
    ]
   },
   "execution_count": 78,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "grid_rus.fit(x_train_rus, y_train_rus)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 79,
 "id": "4737140f",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "SVC(C=1, class_weight='balanced', gamma=0.1)\n"
   ]
  }
 ],
 "source": [
  "print(grid_rus.best_estimator_)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 80,
```

```
 "id": "541389d2",
 "metadata": {},
 "outputs": [],
 "source": [
  "grid_pred_rus = grid_rus.predict(x_test_rus)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 81,
 "id": "7e55bd4c",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "array([[476, 151],\n",
     "       [172, 424]], dtype=int64)"
    ]
   },
   "execution_count": 81,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "confusion_matrix(y_test_rus, grid_pred_rus)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 82,
 "id": "76a4b497",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "              precision    recall  f1-score   support\n",
    "\n",
    "           0       0.73      0.76      0.75       627\n",
    "           1       0.74      0.71      0.72       596\n",
    "\n",
    "    accuracy                           0.74      1223\n",
    "   macro avg       0.74      0.74      0.74      1223\n",
    "weighted avg       0.74      0.74      0.74      1223\n",
    "\n"
```

```
   ]
  }
 ],
 "source": [
  "print(classification_report(y_test_rus, grid_pred_rus))"
 ]
},
{
 "cell_type": "markdown",
 "id": "d5483903",
 "metadata": {},
 "source": [
  "# Model with Random Over Sampling"
 ]
},
{
 "cell_type": "code",
 "execution_count": 83,
 "id": "fcd3a270",
 "metadata": {},
 "outputs": [],
 "source": [
  "svc_ros = SVC()"
 ]
},
{
 "cell_type": "code",
 "execution_count": 84,
 "id": "d3b6de44",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "SVC()"
    ]
   },
   "execution_count": 84,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "svc_ros.fit(x_train_ros, y_train_ros)"
 ]
},
{
 "cell_type": "code",
```

```
 "execution_count": 85,
 "id": "2ccdbd6a",
 "metadata": {},
 "outputs": [],
 "source": [
  "y_pred_ros = svc_ros.predict(x_test_ros)"
 ]
},
{
 "cell_type": "markdown",
 "id": "b0e950c6",
 "metadata": {},
 "source": [
  "# Model Accuracy"
 ]
},
{
 "cell_type": "code",
 "execution_count": 86,
 "id": "067c7443",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "array([[1823,  556],\n",
     "       [ 626, 1773]], dtype=int64)"
    ]
   },
   "execution_count": 86,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "confusion_matrix(y_test_ros, y_pred_ros)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 87,
 "id": "ccb0d59e",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
```

```
      "              precision    recall  f1-score   support\n",
      "\n",
      "           0       0.74      0.77      0.76      2379\n",
      "           1       0.76      0.74      0.75      2399\n",
      "\n",
      "    accuracy                           0.75      4778\n",
      "   macro avg       0.75      0.75      0.75      4778\n",
      "weighted avg       0.75      0.75      0.75      4778\n",
      "\n"
     ]
    }
   ],
   "source": [
    "print(classification_report(y_test_ros, y_pred_ros))"
   ]
  },
  {
   "cell_type": "markdown",
   "id": "f36ef9b3",
   "metadata": {},
   "source": [
    "# Hyperparameter Tunning"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 88,
   "id": "f4bb15fd",
   "metadata": {},
   "outputs": [],
   "source": [
    "param_grid = {'C' : [0.1, 1, 10], 'gamma' : [1, 0.1, 0.01], 'kernel' : ['rbf'], 'class_weight' :
['balanced']}"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 89,
   "id": "f8287e69",
   "metadata": {},
   "outputs": [],
   "source": [
    "grid_ros = GridSearchCV(SVC(), param_grid, refit = True, verbose = 2, cv = 2)"
   ]
  },
  {
   "cell_type": "code",
   "execution_count": 90,
```

```
  "id": "9011e315",
  "metadata": {},
  "outputs": [
   {
    "name": "stdout",
    "output_type": "stream",
    "text": [
     "Fitting 2 folds for each of 9 candidates, totalling 18 fits\n",
     "[CV] END ..C=0.1, class_weight=balanced, gamma=1, kernel=rbf; total time=   3.4s\n",
     "[CV] END ..C=0.1, class_weight=balanced, gamma=1, kernel=rbf; total time=   3.3s\n",
     "[CV] END C=0.1, class_weight=balanced, gamma=0.1, kernel=rbf; total time=   2.4s\n",
     "[CV] END C=0.1, class_weight=balanced, gamma=0.1, kernel=rbf; total time=   2.3s\n",
     "[CV] END C=0.1, class_weight=balanced, gamma=0.01, kernel=rbf; total time=
2.7s\n",
     "[CV] END C=0.1, class_weight=balanced, gamma=0.01, kernel=rbf; total time=
2.6s\n",
     "[CV] END ....C=1, class_weight=balanced, gamma=1, kernel=rbf; total time=   2.7s\n",
     "[CV] END ....C=1, class_weight=balanced, gamma=1, kernel=rbf; total time=   2.8s\n",
     "[CV] END ..C=1, class_weight=balanced, gamma=0.1, kernel=rbf; total time=   2.0s\n",
     "[CV] END ..C=1, class_weight=balanced, gamma=0.1, kernel=rbf; total time=   2.0s\n",
     "[CV] END .C=1, class_weight=balanced, gamma=0.01, kernel=rbf; total time=   2.3s\n",
     "[CV] END .C=1, class_weight=balanced, gamma=0.01, kernel=rbf; total time=   2.2s\n",
     "[CV] END ...C=10, class_weight=balanced, gamma=1, kernel=rbf; total time=   2.5s\n",
     "[CV] END ...C=10, class_weight=balanced, gamma=1, kernel=rbf; total time=   2.5s\n",
     "[CV] END .C=10, class_weight=balanced, gamma=0.1, kernel=rbf; total time=   2.1s\n",
     "[CV] END .C=10, class_weight=balanced, gamma=0.1, kernel=rbf; total time=   2.1s\n",
     "[CV] END C=10, class_weight=balanced, gamma=0.01, kernel=rbf; total time=   2.2s\n",
     "[CV] END C=10, class_weight=balanced, gamma=0.01, kernel=rbf; total time=   2.1s\n"
    ]
   },
   {
    "data": {
     "text/plain": [
      "GridSearchCV(cv=2, estimator=SVC(),\n",
      "             param_grid={'C': [0.1, 1, 10], 'class_weight': ['balanced'],\n",
      "                         'gamma': [1, 0.1, 0.01], 'kernel': ['rbf']},\n",
      "             verbose=2)"
     ]
    },
    "execution_count": 90,
    "metadata": {},
    "output_type": "execute_result"
   }
  ],
  "source": [
   "grid_ros.fit(x_train_ros, y_train_ros)"
  ]
 },
```

```
{
 "cell_type": "code",
 "execution_count": 91,
 "id": "d55881b1",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "SVC(C=10, class_weight='balanced', gamma=1)\n"
   ]
  }
 ],
 "source": [
  "print(grid_ros.best_estimator_)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 92,
 "id": "87bfddc6",
 "metadata": {},
 "outputs": [],
 "source": [
  "grid_pred_ros = grid_ros.predict(x_test_ros)"
 ]
},
{
 "cell_type": "code",
 "execution_count": 93,
 "id": "3b60b9bd",
 "metadata": {},
 "outputs": [
  {
   "data": {
    "text/plain": [
     "array([[2047,  332],\n",
     "       [  68, 2331]], dtype=int64)"
    ]
   },
   "execution_count": 93,
   "metadata": {},
   "output_type": "execute_result"
  }
 ],
 "source": [
  "confusion_matrix(y_test_ros, grid_pred_ros)"
```

```
 ]
},
{
 "cell_type": "code",
 "execution_count": 94,
 "id": "406a5ca3",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "              precision    recall  f1-score   support\n",
    "\n",
    "           0       0.97      0.86      0.91      2379\n",
    "           1       0.88      0.97      0.92      2399\n",
    "\n",
    "    accuracy                           0.92      4778\n",
    "   macro avg       0.92      0.92      0.92      4778\n",
    "weighted avg       0.92      0.92      0.92      4778\n",
    "\n"
   ]
  }
 ],
 "source": [
  "print(classification_report(y_test_ros, grid_pred_ros))"
 ]
},
{
 "cell_type": "markdown",
 "id": "44dedf84",
 "metadata": {},
 "source": [
  "# Lets Compare"
 ]
},
{
 "cell_type": "code",
 "execution_count": 95,
 "id": "74aaec6c",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "              precision    recall  f1-score   support\n",
    "\n",
```

```
    "           0       0.85      0.99      0.91      2414\n",
    "           1       0.82      0.26      0.39       586\n",
    "\n",
    "    accuracy                           0.84      3000\n",
    "   macro avg       0.83      0.62      0.65      3000\n",
    "weighted avg       0.84      0.84      0.81      3000\n",
    "\n"
   ]
  }
 ],
 "source": [
  "print(classification_report(y_test, y_pred))"
 ]
},
{
 "cell_type": "code",
 "execution_count": 96,
 "id": "f9dc6052",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "              precision    recall  f1-score   support\n",
    "\n",
    "           0       0.86      0.89      0.88      2414\n",
    "           1       0.49      0.41      0.45       586\n",
    "\n",
    "    accuracy                           0.80      3000\n",
    "   macro avg       0.68      0.65      0.66      3000\n",
    "weighted avg       0.79      0.80      0.79      3000\n",
    "\n"
   ]
  }
 ],
 "source": [
  "print(classification_report(y_test, grid_predictions))"
 ]
},
{
 "cell_type": "code",
 "execution_count": 97,
 "id": "c4c90355",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
```

```
   "output_type": "stream",
   "text": [
    "              precision    recall  f1-score   support\n",
    "\n",
    "           0       0.73      0.75      0.74       627\n",
    "           1       0.73      0.71      0.72       596\n",
    "\n",
    "    accuracy                           0.73      1223\n",
    "   macro avg       0.73      0.73      0.73      1223\n",
    "weighted avg       0.73      0.73      0.73      1223\n",
    "\n"
   ]
  }
 ],
 "source": [
  "print(classification_report(y_test_rus, y_pred_rus))"
 ]
},
{
 "cell_type": "code",
 "execution_count": 98,
 "id": "b63889ae",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "              precision    recall  f1-score   support\n",
    "\n",
    "           0       0.73      0.76      0.75       627\n",
    "           1       0.74      0.71      0.72       596\n",
    "\n",
    "    accuracy                           0.74      1223\n",
    "   macro avg       0.74      0.74      0.74      1223\n",
    "weighted avg       0.74      0.74      0.74      1223\n",
    "\n"
   ]
  }
 ],
 "source": [
  "print(classification_report(y_test_rus, grid_pred_rus))"
 ]
},
{
 "cell_type": "code",
 "execution_count": 99,
 "id": "dce8dffe",
```

```
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "              precision    recall  f1-score   support\n",
    "\n",
    "           0       0.74      0.77      0.76      2379\n",
    "           1       0.76      0.74      0.75      2399\n",
    "\n",
    "    accuracy                           0.75      4778\n",
    "   macro avg       0.75      0.75      0.75      4778\n",
    "weighted avg       0.75      0.75      0.75      4778\n",
    "\n"
   ]
  }
 ],
 "source": [
  "print(classification_report(y_test_ros, y_pred_ros))"
 ]
},
{
 "cell_type": "code",
 "execution_count": 100,
 "id": "f9450c04",
 "metadata": {},
 "outputs": [
  {
   "name": "stdout",
   "output_type": "stream",
   "text": [
    "              precision    recall  f1-score   support\n",
    "\n",
    "           0       0.97      0.86      0.91      2379\n",
    "           1       0.88      0.97      0.92      2399\n",
    "\n",
    "    accuracy                           0.92      4778\n",
    "   macro avg       0.92      0.92      0.92      4778\n",
    "weighted avg       0.92      0.92      0.92      4778\n",
    "\n"
   ]
  }
 ],
 "source": [
  "print(classification_report(y_test_ros, grid_pred_ros))"
 ]
},
```

```
  {
   "cell_type": "code",
   "execution_count": null,
   "id": "99502e33",
   "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": "3.8.12"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 5
}
```