Step 1

import pandas as pd

import seaborn as sns

Step 2

df=pd.read\_csv("/content/Churn\_Modelling.csv")

df

RowNumber CustomerId Surname CreditScore Geography Gender Age Tenure Balance NumOfProducts HasCrCard IsActiveMember EstimatedSalary Exited

0 1 15634602 Hargrave 619 France Female 42 2 0.00 1 1 1 101348.88 1

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

2 3 15619304 Onio 502 France Female 42 8 159660.80 3 1 0 113931.57 1

3 4 15701354 Boni 699 France Female 39 1 0.00 2 0 0 93826.63 0

4 5 15737888 Mitchell 850 Spain Female 43 2 125510.82 1 1 1 79084.10 0

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

9995 9996 15606229 Obijiaku 771 France Male 39 5 0.00 2 1 0 96270.64 0

9996 9997 15569892 Johnstone 516 France Male 35 10 57369.61 1 1 1 101699.77 0

9997 9998 15584532 Liu 709 France Female 36 7 0.00 1 0 1 42085.58 1

9998 9999 15682355 Sabbatini 772 Germany Male 42 3 75075.31 2 1 0 92888.52 1

9999 10000 15628319 Walker 792 France Female 28 4 130142.79 1 1 0 38190.78 0

10000 rows × 14 columns

df.dtypes

RowNumber int64

CustomerId int64

Surname object

CreditScore int64

Geography object

Gender object

Age int64

Tenure int64

Balance float64

NumOfProducts int64

HasCrCard int64

IsActiveMember int64

EstimatedSalary float64

Exited int64

dtype: object

df.describe()

RowNumber CustomerId CreditScore Age Tenure Balance NumOfProducts HasCrCard IsActiveMember EstimatedSalary Exited

count 10000.00000 1.000000e+04 10000.000000 10000.000000 10000.000000 10000.000000 10000.000000 10000.00000 10000.000000 10000.000000 10000.000000

mean 5000.50000 1.569094e+07 650.528800 38.921800 5.012800 76485.889288 1.530200 0.70550 0.515100 100090.239881 0.203700

std 2886.89568 7.193619e+04 96.653299 10.487806 2.892174 62397.405202 0.581654 0.45584 0.499797 57510.492818 0.402769

min 1.00000 1.556570e+07 350.000000 18.000000 0.000000 0.000000 1.000000 0.00000 0.000000 11.580000 0.000000

25% 2500.75000 1.562853e+07 584.000000 32.000000 3.000000 0.000000 1.000000 0.00000 0.000000 51002.110000 0.000000

50% 5000.50000 1.569074e+07 652.000000 37.000000 5.000000 97198.540000 1.000000 1.00000 1.000000 100193.915000 0.000000

75% 7500.25000 1.575323e+07 718.000000 44.000000 7.000000 127644.240000 2.000000 1.00000 1.000000 149388.247500 0.000000

max 10000.00000 1.581569e+07 850.000000 92.000000 10.000000 250898.090000 4.000000 1.00000 1.000000 199992.480000 1.000000

df.isnull().any()

RowNumber False

CustomerId False

Surname False

CreditScore False

Geography False

Gender False

Age False

Tenure False

Balance False

NumOfProducts False

HasCrCard False

IsActiveMember False

EstimatedSalary False

Exited False

dtype: bool

df.isna().sum()

RowNumber 0

CustomerId 0

Surname 0

CreditScore 0

Geography 0

Gender 0

Age 0

Tenure 0

Balance 0

NumOfProducts 0

HasCrCard 0

IsActiveMember 0

EstimatedSalary 0

Exited 0

dtype: int64

df.skew()

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

"""Entry point for launching an IPython kernel.

RowNumber 0.000000

CustomerId 0.001149

CreditScore -0.071607

Age 1.011320

Tenure 0.010991

Balance -0.141109

NumOfProducts 0.745568

HasCrCard -0.901812

IsActiveMember -0.060437

EstimatedSalary 0.002085

Exited 1.471611

dtype: float64

df.boxplot(column="EstimatedSalary")

<matplotlib.axes.\_subplots.AxesSubplot at 0x7fd194d2a150>

df=df[(df.EstimatedSalary > 25000) & (df.EstimatedSalary<175000)]

df

RowNumber CustomerId Surname CreditScore Geography Gender Age Tenure Balance NumOfProducts HasCrCard IsActiveMember EstimatedSalary Exited

0 1 15634602 Hargrave 619 France Female 42 2 0.00 1 1 1 101348.88 1

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

2 3 15619304 Onio 502 France Female 42 8 159660.80 3 1 0 113931.57 1

3 4 15701354 Boni 699 France Female 39 1 0.00 2 0 0 93826.63 0

4 5 15737888 Mitchell 850 Spain Female 43 2 125510.82 1 1 1 79084.10 0

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

9995 9996 15606229 Obijiaku 771 France Male 39 5 0.00 2 1 0 96270.64 0

9996 9997 15569892 Johnstone 516 France Male 35 10 57369.61 1 1 1 101699.77 0

9997 9998 15584532 Liu 709 France Female 36 7 0.00 1 0 1 42085.58 1

9998 9999 15682355 Sabbatini 772 Germany Male 42 3 75075.31 2 1 0 92888.52 1

9999 10000 15628319 Walker 792 France Female 28 4 130142.79 1 1 0 38190.78 0

7523 rows × 14 columns

df.NumOfProducts.unique()

array([1, 3, 2, 4])

df.boxplot(column="Balance")

<matplotlib.axes.\_subplots.AxesSubplot at 0x7fd191cc9b50>

df=df[df.Balance < 150000]

df

RowNumber CustomerId Surname CreditScore Geography Gender Age Tenure Balance NumOfProducts HasCrCard IsActiveMember EstimatedSalary Exited

0 1 15634602 Hargrave 619 France Female 42 2 0.00 1 1 1 101348.88 1

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

3 4 15701354 Boni 699 France Female 39 1 0.00 2 0 0 93826.63 0

4 5 15737888 Mitchell 850 Spain Female 43 2 125510.82 1 1 1 79084.10 0

5 6 15574012 Chu 645 Spain Male 44 8 113755.78 2 1 0 149756.71 1

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

9995 9996 15606229 Obijiaku 771 France Male 39 5 0.00 2 1 0 96270.64 0

9996 9997 15569892 Johnstone 516 France Male 35 10 57369.61 1 1 1 101699.77 0

9997 9998 15584532 Liu 709 France Female 36 7 0.00 1 0 1 42085.58 1

9998 9999 15682355 Sabbatini 772 Germany Male 42 3 75075.31 2 1 0 92888.52 1

9999 10000 15628319 Walker 792 France Female 28 4 130142.79 1 1 0 38190.78 0

6782 rows × 14 columns

df.boxplot(column="Age")

<matplotlib.axes.\_subplots.AxesSubplot at 0x7fd191bab050>

df=df[(df.Age <50) & (df.Age >20)]

df

RowNumber CustomerId Surname CreditScore Geography Gender Age Tenure Balance NumOfProducts HasCrCard IsActiveMember EstimatedSalary Exited

0 1 15634602 Hargrave 619 France Female 42 2 0.00 1 1 1 101348.88 1

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

3 4 15701354 Boni 699 France Female 39 1 0.00 2 0 0 93826.63 0

4 5 15737888 Mitchell 850 Spain Female 43 2 125510.82 1 1 1 79084.10 0

5 6 15574012 Chu 645 Spain Male 44 8 113755.78 2 1 0 149756.71 1

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

9995 9996 15606229 Obijiaku 771 France Male 39 5 0.00 2 1 0 96270.64 0

9996 9997 15569892 Johnstone 516 France Male 35 10 57369.61 1 1 1 101699.77 0

9997 9998 15584532 Liu 709 France Female 36 7 0.00 1 0 1 42085.58 1

9998 9999 15682355 Sabbatini 772 Germany Male 42 3 75075.31 2 1 0 92888.52 1

9999 10000 15628319 Walker 792 France Female 28 4 130142.79 1 1 0 38190.78 0

5783 rows × 14 columns

df.boxplot(column="CreditScore")

<matplotlib.axes.\_subplots.AxesSubplot at 0x7fd191b53110>

df=df[(df.CreditScore>500) & (df.CreditScore<790)]

df

RowNumber CustomerId Surname CreditScore Geography Gender Age Tenure Balance NumOfProducts HasCrCard IsActiveMember EstimatedSalary Exited

0 1 15634602 Hargrave 619 France Female 42 2 0.00 1 1 1 101348.88 1

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

3 4 15701354 Boni 699 France Female 39 1 0.00 2 0 0 93826.63 0

5 6 15574012 Chu 645 Spain Male 44 8 113755.78 2 1 0 149756.71 1

8 9 15792365 He 501 France Male 44 4 142051.07 2 0 1 74940.50 0

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

9990 9991 15798964 Nkemakonam 714 Germany Male 33 3 35016.60 1 1 0 53667.08 0

9995 9996 15606229 Obijiaku 771 France Male 39 5 0.00 2 1 0 96270.64 0

9996 9997 15569892 Johnstone 516 France Male 35 10 57369.61 1 1 1 101699.77 0

9997 9998 15584532 Liu 709 France Female 36 7 0.00 1 0 1 42085.58 1

9998 9999 15682355 Sabbatini 772 Germany Male 42 3 75075.31 2 1 0 92888.52 1

4993 rows × 14 columns

churn\_yes=df['Tenure'][df.Exited == 1]

churn\_no=df['Tenure'][df.Exited == 0]

import matplotlib.pyplot as plt

plt.xlabel("Tenure")

plt.ylabel("No of Customers")

plt.title("churn prediction ")

plt.hist([churn\_yes,churn\_no],color=["green","red"],label=["churn=yes","churn=no"])

plt.legend()

plt.show()

/usr/local/lib/python3.7/dist-packages/numpy/core/fromnumeric.py:3208: VisibleDeprecationWarning: Creating an ndarray from ragged nested sequences (which is a list-or-tuple of lists-or-tuples-or ndarrays with different lengths or shapes) is deprecated. If you meant to do this, you must specify 'dtype=object' when creating the ndarray.

return asarray(a).size

/usr/local/lib/python3.7/dist-packages/matplotlib/cbook/\_\_init\_\_.py:1376: VisibleDeprecationWarning: Creating an ndarray from ragged nested sequences (which is a list-or-tuple of lists-or-tuples-or ndarrays with different lengths or shapes) is deprecated. If you meant to do this, you must specify 'dtype=object' when creating the ndarray.

X = np.atleast\_1d(X.T if isinstance(X, np.ndarray) else np.asarray(X))

sns.pairplot(data=df,hue='Exited')

<seaborn.axisgrid.PairGrid at 0x7fd191908a10>

df.columns

Index(['RowNumber', 'CustomerId', 'Surname', 'CreditScore', 'Geography',

'Gender', 'Age', 'Tenure', 'Balance', 'NumOfProducts', 'HasCrCard',

'IsActiveMember', 'EstimatedSalary', 'Exited'],

dtype='object')

feature=df[['CreditScore','Geography',

'Gender', 'Age', 'Tenure', 'Balance', 'NumOfProducts', 'HasCrCard',

'IsActiveMember', 'EstimatedSalary']]

feature

CreditScore Geography Gender Age Tenure Balance NumOfProducts HasCrCard IsActiveMember EstimatedSalary

0 619 France Female 42 2 0.00 1 1 1 101348.88

1 608 Spain Female 41 1 83807.86 1 0 1 112542.58

3 699 France Female 39 1 0.00 2 0 0 93826.63

5 645 Spain Male 44 8 113755.78 2 1 0 149756.71

8 501 France Male 44 4 142051.07 2 0 1 74940.50

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

9990 714 Germany Male 33 3 35016.60 1 1 0 53667.08

9995 771 France Male 39 5 0.00 2 1 0 96270.64

9996 516 France Male 35 10 57369.61 1 1 1 101699.77

9997 709 France Female 36 7 0.00 1 0 1 42085.58

9998 772 Germany Male 42 3 75075.31 2 1 0 92888.52

4993 rows × 10 columns

label=df['Exited']

label

0 1

1 0

3 0

5 1

8 0

..

9990 0

9995 0

9996 0

9997 1

9998 1

Name: Exited, Length: 4993, dtype: int64

from sklearn.compose import ColumnTransformer

from sklearn.preprocessing import OneHotEncoder

ct = ColumnTransformer([("oh",OneHotEncoder(),[1,2])],remainder="passthrough")

feature\_onehot= ct.fit\_transform(feature)

feature\_onehot

array([[1.0000000e+00, 0.0000000e+00, 0.0000000e+00, ..., 1.0000000e+00,

1.0000000e+00, 1.0134888e+05],

[0.0000000e+00, 0.0000000e+00, 1.0000000e+00, ..., 0.0000000e+00,

1.0000000e+00, 1.1254258e+05],

[1.0000000e+00, 0.0000000e+00, 0.0000000e+00, ..., 0.0000000e+00,

0.0000000e+00, 9.3826630e+04],

...,

[1.0000000e+00, 0.0000000e+00, 0.0000000e+00, ..., 1.0000000e+00,

1.0000000e+00, 1.0169977e+05],

[1.0000000e+00, 0.0000000e+00, 0.0000000e+00, ..., 0.0000000e+00,

1.0000000e+00, 4.2085580e+04],

[0.0000000e+00, 1.0000000e+00, 0.0000000e+00, ..., 1.0000000e+00,

0.0000000e+00, 9.2888520e+04]])

len(feature\_onehot)

4993

feature\_onehot[0]

array([1.0000000e+00, 0.0000000e+00, 0.0000000e+00, 1.0000000e+00,

0.0000000e+00, 6.1900000e+02, 4.2000000e+01, 2.0000000e+00,

0.0000000e+00, 1.0000000e+00, 1.0000000e+00, 1.0000000e+00,

1.0134888e+05])

df["Geography"].unique()

array(['France', 'Spain', 'Germany'], dtype=object)

df["Gender"].unique()

array(['Female', 'Male'], dtype=object)

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

trainX,testX,trainY,testY = train\_test\_split(feature\_onehot,label,test\_size=0.2,random\_state=0)

trainX

array([[0.0000000e+00, 1.0000000e+00, 0.0000000e+00, ..., 1.0000000e+00,

0.0000000e+00, 8.4300400e+04],

[0.0000000e+00, 0.0000000e+00, 1.0000000e+00, ..., 1.0000000e+00,

1.0000000e+00, 1.4203307e+05],

[1.0000000e+00, 0.0000000e+00, 0.0000000e+00, ..., 1.0000000e+00,

1.0000000e+00, 1.6737626e+05],

...,

[0.0000000e+00, 1.0000000e+00, 0.0000000e+00, ..., 1.0000000e+00,

0.0000000e+00, 3.8270470e+04],

[1.0000000e+00, 0.0000000e+00, 0.0000000e+00, ..., 1.0000000e+00,

0.0000000e+00, 1.1812088e+05],

[1.0000000e+00, 0.0000000e+00, 0.0000000e+00, ..., 1.0000000e+00,

1.0000000e+00, 9.7755290e+04]])

testX

array([[1.0000000e+00, 0.0000000e+00, 0.0000000e+00, ..., 1.0000000e+00,

1.0000000e+00, 1.1045799e+05],

[0.0000000e+00, 1.0000000e+00, 0.0000000e+00, ..., 1.0000000e+00,

0.0000000e+00, 6.3981370e+04],

[1.0000000e+00, 0.0000000e+00, 0.0000000e+00, ..., 1.0000000e+00,

0.0000000e+00, 1.1343608e+05],

...,

[1.0000000e+00, 0.0000000e+00, 0.0000000e+00, ..., 1.0000000e+00,

0.0000000e+00, 2.6450570e+04],

[0.0000000e+00, 0.0000000e+00, 1.0000000e+00, ..., 1.0000000e+00,

0.0000000e+00, 5.4947510e+04],

[0.0000000e+00, 1.0000000e+00, 0.0000000e+00, ..., 1.0000000e+00,

0.0000000e+00, 1.6318162e+05]])

trainY

3935 1

34 0

4189 0

5100 0

5918 0

..

9864 0

6541 0

3333 0

5298 0

5530 0

Name: Exited, Length: 3994, dtype: int64

testY

2378 0

8392 1

8410 0

4970 0

7674 0

..

7618 0

5529 0

2262 1

7122 0

7061 0

Name: Exited, Length: 999, dtype: int64

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

trainX\_scale = scaler.fit\_transform(trainX)

testX\_scale = scaler.transform(testX)

trainX\_scale

array([[-1.0305103 , 1.81765764, -0.58139784, ..., 0.64211021,

-0.97918504, -0.37178651],

[-1.0305103 , -0.55015861, 1.71999262, ..., 0.64211021,

1.02125744, 0.97885865],

[ 0.97039302, -0.55015861, -0.58139784, ..., 0.64211021,

1.02125744, 1.57175791],

...,

[-1.0305103 , 1.81765764, -0.58139784, ..., 0.64211021,

-0.97918504, -1.44864824],

[ 0.97039302, -0.55015861, -0.58139784, ..., 0.64211021,

-0.97918504, 0.41943738],

[ 0.97039302, -0.55015861, -0.58139784, ..., 0.64211021,

1.02125744, -0.05701184]])

testX\_scale

array([[ 0.97039302, -0.55015861, -0.58139784, ..., 0.64211021,

1.02125744, 0.24016548],

[-1.0305103 , 1.81765764, -0.58139784, ..., 0.64211021,

-0.97918504, -0.84714647],

[ 0.97039302, -0.55015861, -0.58139784, ..., 0.64211021,

-0.97918504, 0.30983735],

...,

[ 0.97039302, -0.55015861, -0.58139784, ..., 0.64211021,

-0.97918504, -1.72517262],

[-1.0305103 , -0.55015861, 1.71999262, ..., 0.64211021,

-0.97918504, -1.05849196],

[-1.0305103 , 1.81765764, -0.58139784, ..., 0.64211021,

-0.97918504, 1.47362508]])

trainY

3935 1

34 0

4189 0

5100 0

5918 0

..

9864 0

6541 0

3333 0

5298 0

5530 0

Name: Exited, Length: 3994, dtype: int64

testY

2378 0

8392 1

8410 0

4970 0

7674 0

..

7618 0

5529 0

2262 1

7122 0

7061 0

Name: Exited, Length: 999, dtype: int64