Assignment 2

MAHENDRA ENGINEERING COLLEGES FOR WOMEN

NAME:SASIKALA.S

CLASS:4 YEAR IT

SUBJECT:IBM

REGISTER NUMBER:611419205028

1

0 1 1 101348.88 1

HasCrCard IsActiveMember EstimatedSalary Exited

1

4 850 Spain Female 43 2 125510.82

2

3 699 France Female 39 1 0.00

3

2 502 France Female 42 8 159660.80

1 608 Spain Female 41 1 83807.86

1

0 619 France Female 42 2 0.00

NumOfProducts \

CreditScore Geography Gender Age Tenure Balance

df.head()

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

0 1 0 0

dtypes: float64(2), int64(7), object(2)

memory usage: 859.5+ KB

df["Geography"].unique()

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

df["Gender"].unique()

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

geo=pd.get\_dummies(df["Geography"],drop\_first=False)

geo.head()

France Germany Spain

10 Exited 10000 non-null int64

1 0 0 1

2 1 0 0

3 1 0 0

4 0 0 1

gen=pd.get\_dummies(df["Gender"],drop\_first=False)

df=pd.concat([df, geo,gen], axis=1)

df

CreditScore Geography Gender Age Tenure Balance

NumOfProducts \

0 CreditScore 10000 non-null int64

2 1 0 113931.57 1

3 0 0 93826.63 0

4 1 1 79084.10 0

df.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 10000 entries, 0 to 9999

Data columns (total 11 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

1 0 1 112542.58 0

1 Geography 10000 non-null object

2 Gender 10000 non-null object

3 Age 10000 non-null int64

4 Tenure 10000 non-null int64

5 Balance 10000 non-null float64

6 NumOfProducts 10000 non-null int64

7 HasCrCard 10000 non-null int64

8 IsActiveMember 10000 non-null int64

9 EstimatedSalary 10000 non-null float64

9995 1 0 96270.64 0 1

0

1 0 1 112542.58 0 0

0

2 1 0 113931.57 1 1

0

3 0 0 93826.63 0 1

0

4 1 1 79084.10 0 0

0

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

...

0 1 1 101348.88 1 1

0

9996 1 1 101699.77 0 1

0

9997 0 1 42085.58 1 1

0

9998 1 0 92888.52 1 0

1

9999 1 0 38190.78 0 1

0

Spain Female Male

0 0 1 0

9995 771 France Male 39 5 0.00

1

1 608 Spain Female 41 1 83807.86

1

2 502 France Female 42 8 159660.80

3

3 699 France Female 39 1 0.00

2

4 850 Spain Female 43 2 125510.82

1

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

...

0 619 France Female 42 2 0.00

2

9996 516 France Male 35 10 57369.61

1

9997 709 France Female 36 7 0.00

1

9998 772 Germany Male 42 3 75075.31

2

9999 792 France Female 28 4 130142.79

1

HasCrCard IsActiveMember EstimatedSalary Exited France

Germany \

0 0

1

1 1 112542.58 0 0 0 1

1

2 0 113931.57 1 1 0 0

1

3 0 93826.63 0 1 0 0

1

4 1 79084.10 0 0 0 1

1

Male

0 1 101348.88 1 1 0 0

1 0

2 0

3 0

4 0

x=df.drop('Exited',axis=1)

x

CreditScore Age Tenure Balance NumOfProducts HasCrCard \

0 619 42 2 0.00 1 1

1 608 41 1 83807.86 1 0

df.drop(["Geography","Gender"], axis=1, inplace=True)

2 0 1 0

3 0 1 0

4 1 1 0

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

9995 0 0 1

9996 0 0 1

9997 0 1 0

9998 0 0 1

9999 0 1 0

[10000 rows x 16 columns]

1 1 1 0

df.head()

CreditScore Age Tenure Balance NumOfProducts HasCrCard \

0 619 42 2 0.00 1 1

1 608 41 1 83807.86 1 0

2 502 42 8 159660.80 3 1

3 699 39 1 0.00 2 0

4 850 43 2 125510.82 1 1

IsActiveMember EstimatedSalary Exited France Germany Spain

Female \

y

1

9996 1 101699.77 1 0 0 0

1

9997 1 42085.58 1 0 0 1

0

9998 0 92888.52 0 1 0 0

1

9999 0 38190.78 1 0 0 1

0

[10000 rows x 13 columns]

y=df['Exited']

9995 0 96270.64 1 0 0 0

0 1

1 0

2 1

3 0

4 0

..

9995 0

9996 0

9997 1

9998 1

0

3 699 39 1 0.00 2 0

4 850 43 2 125510.82 1 1

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

9995 771 39 5 0.00 2 1

9996 516 35 10 57369.61 1 1

9997 709 36 7 0.00 1 0

9998 772 42 3 75075.31 2 1

9999 792 28 4 130142.79 1 1

IsActiveMember EstimatedSalary France Germany Spain Female

Male

0 1 101348.88 1 0 0 1

2 502 42 8 159660.80 3 1

1 1 112542.58 0 0 1 1

0

2 0 113931.57 1 0 0 1

0

3 0 93826.63 1 0 0 1

0

4 1 79084.10 0 0 1 1

0

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

...

1.09168714, -1.09168714],

sc = StandardScaler()

x\_train = sc.fit\_transform(x\_train)

x\_train

array([[ 0.16958176, -0.46460796, 0.00666099, ..., 1.74309049,

1.09168714, -1.09168714],

[-2.30455945, 0.30102557, -1.37744033, ..., -0.57369368,

-0.91601335, 0.91601335],

[-1.19119591, -0.94312892, -1.031415 , ..., -0.57369368,

from sklearn.preprocessing import StandardScaler

...,

[ 0.9015152 , -0.36890377, 0.00666099, ..., -0.57369368,

-0.91601335, 0.91601335],

[-0.62420521, -0.08179119, 1.39076231, ..., 1.74309049,

1.09168714, -1.09168714],

[-0.28401079, 0.87525072, -1.37744033, ..., -0.57369368,

1.09168714, -1.09168714]])

x\_test = sc.transform(x\_test)

x\_train,x\_test, y\_train,y\_test = train\_test\_split(x,y,

Name: Exited, Length: 10000, dtype: int64

df.shape

(10000, 14)

x.shape

(10000, 13)

y.shape

(10000,)

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

9999 0

test\_size=0.2,random\_state=0)

x\_train.shape

(8000, 13)

x\_test.shape

(2000, 13)

y\_test.shape

(2000,)

x\_test

array([[-0.55204276, -0.36890377, 1.04473698, ..., -0.57369368,

1.09168714, -1.09168714],

[-1.31490297, 0.10961719, -1.031415 , ..., -0.57369368,

1.09168714, -1.09168714],

[ 0.57162971, 0.30102557, 1.04473698, ..., 1.74309049,

1.09168714, -1.09168714],

...,

[-0.74791227, -0.27319958, -1.37744033, ..., 1.74309049,

-0.91601335, 0.91601335],

[-0.00566991, -0.46460796, -0.33936434, ..., -0.57369368,

-0.91601335, 0.91601335],

[-0.79945688, -0.84742473, 1.04473698, ..., -0.57369368,

-0.91601335, 0.91601335]])