# LendingClubLoanData

February 11, 2023

[1]:
```
# import required library
import pandas as pd
import numpy as np
pd.set_option('display.max_rows',None)
pd.set_option('display.max_columns',None)
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

[2]:
```
#loading data
df_loan=pd.read_csv("loan_data.csv")
df_loan.head()
```

[2]:

| | credit.policy | purpose | int.rate | installment | log.annual.inc | dti | fico | days.with.cr.line | revol.bal | revol.util | inq.last.6mths | delinq.2yrs | pub.rec | not.fully.paid |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | debt_consolidation | 0.1189 | 829.10 | 11.350407 | 19.48 | 737 | 5639.958333 | 28854 | 52.1 | 0 | 0 | 0 | 0 |
| 1 | 1 | credit_card | 0.1071 | 228.22 | 11.082143 | 14.29 | 707 | 2760.000000 | 33623 | 76.7 | 0 | 0 | 0 | 0 |
| 2 | 1 | debt_consolidation | 0.1357 | 366.86 | 10.373491 | 11.63 | 682 | 4710.000000 | 3511 | 25.6 | 1 | 0 | 0 | 0 |
| 3 | 1 | debt_consolidation | 0.1008 | 162.34 | 11.350407 | 8.10 | 712 | 2699.958333 | 33667 | 73.2 | 1 | 0 | 0 | 0 |
| 4 | 1 | credit_card | 0.1426 | 102.92 | 11.299732 | 14.97 | 667 | 4066.000000 | 4740 | 39.5 | 0 | 1 | 0 | 0 |

[3]:
```
#shape of data
df_loan.shape
```

[3]: (9578, 14)

[4]:
```
#info of data
df_loan.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9578 entries, 0 to 9577
Data columns (total 14 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   credit.policy      9578 non-null   int64  
 1   purpose            9578 non-null   object 
 2   int.rate           9578 non-null   float64
 3   installment        9578 non-null   float64
 4   log.annual.inc     9578 non-null   float64
 5   dti                9578 non-null   float64
 6   fico               9578 non-null   int64  
 7   days.with.cr.line  9578 non-null   float64
 8   revol.bal          9578 non-null   int64  
 9   revol.util         9578 non-null   float64
 10  inq.last.6mths     9578 non-null   int64  
 11  delinq.2yrs        9578 non-null   int64  
 12  pub.rec            9578 non-null   int64  
 13  not.fully.paid     9578 non-null   int64  
dtypes: float64(6), int64(7), object(1)
memory usage: 1.0+ MB
```

[5]:
```
#description of data
df_loan.describe()
```

[5]:

| | credit.policy | int.rate | installment | log.annual.inc | dti | \ |
|---|---|---|---|---|---|---|
| count | 9578.000000 | 9578.000000 | 9578.000000 | 9578.000000 | 9578.000000 | |
| mean | 0.804970 | 0.122640 | 319.089413 | 10.932117 | 12.606679 | |
| std | 0.396245 | 0.026847 | 207.071301 | 0.614813 | 6.883970 | |
| min | 0.000000 | 0.060000 | 15.670000 | 7.547502 | 0.000000 | |
| 25% | 1.000000 | 0.103900 | 163.770000 | 10.558414 | 7.212500 | |
| 50% | 1.000000 | 0.122100 | 268.950000 | 10.928884 | 12.665000 | |
| 75% | 1.000000 | 0.140700 | 432.762500 | 11.291293 | 17.950000 | |
| max | 1.000000 | 0.216400 | 940.140000 | 14.528354 | 29.960000 | |

| | fico | days.with.cr.line | revol.bal | revol.util | \ |
|---|---|---|---|---|---|
| count | 9578.000000 | 9578.000000 | 9.578000e+03 | 9578.000000 | |
| mean | 710.846314 | 4560.767197 | 1.691396e+04 | 46.799236 | |
| std | 37.970537 | 2496.930377 | 3.375619e+04 | 29.014417 | |
| min | 612.000000 | 178.958333 | 0.000000e+00 | 0.000000 | |
| 25% | 682.000000 | 2820.000000 | 3.187000e+03 | 22.600000 | |
| 50% | 707.000000 | 4139.958333 | 8.596000e+03 | 46.300000 | |

```
75%     737.000000        5730.000000  1.824950e+04    70.900000
max     827.000000       17639.958330  1.207359e+06   119.000000

       inq.last.6mths  delinq.2yrs      pub.rec  not.fully.paid
count     9578.000000  9578.000000  9578.000000     9578.000000
mean         1.577469     0.163708     0.062122        0.160054
std          2.200245     0.546215     0.262126        0.366676
min          0.000000     0.000000     0.000000        0.000000
25%          0.000000     0.000000     0.000000        0.000000
50%          1.000000     0.000000     0.000000        0.000000
75%          2.000000     0.000000     0.000000        0.000000
max         33.000000    13.000000     5.000000        1.000000
```

[6]:
```
#value count on purpose column
df_loan['purpose'].value_counts()
```

[6]:
```
debt_consolidation    3957
all_other             2331
credit_card           1262
home_improvement       629
small_business         619
major_purchase         437
educational            343
Name: purpose, dtype: int64
```

[7]:
```
#groupby purpose on revolving balance to get the amount due in various cate
df_loan.groupby('purpose')['revol.bal'].sum()
```

[7]:
```
purpose
all_other             30030366
credit_card           29253186
debt_consolidation    67849534
educational            3714312
home_improvement      10899788
major_purchase         3181995
small_business        17072765
Name: revol.bal, dtype: int64
```

[8]:
```
#purpose is categorical column
#applying label encoder to convert categorical to numerocal values.
from sklearn.preprocessing import LabelEncoder
LB = LabelEncoder()
df_loan['purpose']=LB.fit_transform(df_loan['purpose'])
```

[9]:
```
df_loan.head()
```

[9]:
```
   credit.policy  purpose  int.rate  installment  log.annual.inc    dti  fico  \
0              1        2    0.1189       829.10       11.350407  19.48   737
1              1        1    0.1071       228.22       11.082143  14.29   707
2              1        2    0.1357       366.86       10.373491  11.63   682
3              1        2    0.1008       162.34       11.350407   8.10   712
4              1        1    0.1426       102.92       11.299732  14.97   667

   days.with.cr.line  revol.bal  revol.util  inq.last.6mths  delinq.2yrs  \
0        5639.958333      28854        52.1               0            0
1        2760.000000      33623        76.7               0            0
2        4710.000000       3511        25.6               1            0
3        2699.958333      33667        73.2               1            0
4        4066.000000       4740        39.5               0            1

   pub.rec  not.fully.paid
0        0               0
1        0               0
2        0               0
3        0               0
4        0               0
```

[10]:
```
df_loan.tail()
```

[10]:
```
      credit.policy  purpose  int.rate  installment  log.annual.inc    dti  \
9573              0        0    0.1461       344.76       12.180755  10.39
9574              0        0    0.1253       257.70       11.141862   0.21
9575              0        2    0.1071        97.81       10.596635  13.09
9576              0        4    0.1600       351.58       10.819778  19.18
9577              0        2    0.1392       853.43       11.264464  16.28

      fico  days.with.cr.line  revol.bal  revol.util  inq.last.6mths  \
9573   672       10474.000000     215372        82.1               2
9574   722        4380.000000        184         1.1               5
9575   687        3450.041667      10036        82.9               8
9576   692        1800.000000          0         3.2               5
9577   732        4740.000000      37879        57.0               6

      delinq.2yrs  pub.rec  not.fully.paid
9573            0        0               1
9574            0        0               1
9575            0        0               1
9576            0        0               1
9577            0        0               1
```

[11]:
```
#value counts to credit policy
df_loan['credit.policy'].value_counts()
```

[11]:
```
1    7710
0    1868
Name: credit.policy, dtype: int64
```

[12]:
```
#APPLYING ONE HOT ENCODING ON CREDIT POLICY
df_loan=pd.get_dummies(df_loan, columns=['credit.policy'])
df_loan.head()
```

[12]:

| | purpose | int.rate | installment | log.annual.inc | dti | fico | \ |
|---|---|---|---|---|---|---|---|
| 0 | 2 | 0.1189 | 829.10 | 11.350407 | 19.48 | 737 | |
| 1 | 1 | 0.1071 | 228.22 | 11.082143 | 14.29 | 707 | |
| 2 | 2 | 0.1357 | 366.86 | 10.373491 | 11.63 | 682 | |
| 3 | 2 | 0.1008 | 162.34 | 11.350407 | 8.10 | 712 | |
| 4 | 1 | 0.1426 | 102.92 | 11.299732 | 14.97 | 667 | |

| | days.with.cr.line | revol.bal | revol.util | inq.last.6mths | delinq.2yrs | \ |
|---|---|---|---|---|---|---|
| 0 | 5639.958333 | 28854 | 52.1 | 0 | 0 | |
| 1 | 2760.000000 | 33623 | 76.7 | 0 | 0 | |
| 2 | 4710.000000 | 3511 | 25.6 | 1 | 0 | |
| 3 | 2699.958333 | 33667 | 73.2 | 1 | 0 | |
| 4 | 4066.000000 | 4740 | 39.5 | 0 | 1 | |

| | pub.rec | not.fully.paid | credit.policy_0 | credit.policy_1 |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 2 | 0 | 0 | 0 | 1 |
| 3 | 0 | 0 | 0 | 1 |
| 4 | 0 | 0 | 0 | 1 |

[13]:
```
#creating dependent and independent features
x=df_loan.drop('not.fully.paid', axis=1)
y=df_loan['not.fully.paid']
```

[14]:
```
np.random.seed(12345)
```

[15]:
```
#train test split
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test=train_test_split(x, y,
                                                  stratify=y,
                                                  test_size=0.2,
                                                  random_state=12345)
```

[16]:
```
#shape of train test data
print(f'training data shape {x_train.shape} {y_train.shape}')
print(f'testing data shape {x_test.shape}  {y_test.shape}')
```

```
training data shape (7662, 14) (7662,)
```

```
testing data shape (1916, 14)  (1916,)
```

[17]:
```
x_train.head()
```

[17]:

| | purpose | int.rate | installment | log.annual.inc | dti | fico | \ |
|---|---|---|---|---|---|---|---|
| 1279 | 2 | 0.1412 | 167.76 | 10.385914 | 23.85 | 682 | |
| 262 | 0 | 0.0933 | 319.54 | 10.505068 | 19.46 | 727 | |
| 7922 | 1 | 0.1438 | 359.95 | 11.245046 | 9.98 | 642 | |
| 1959 | 3 | 0.0963 | 176.52 | 10.650887 | 15.63 | 727 | |
| 4095 | 0 | 0.0859 | 221.28 | 11.350407 | 3.92 | 762 | |

| | days.with.cr.line | revol.bal | revol.util | inq.last.6mths | delinq.2yrs | \ |
|---|---|---|---|---|---|---|
| 1279 | 3569.958333 | 2379 | 68.0 | 1 | 0 | |
| 262 | 4470.041667 | 827 | 6.6 | 2 | 0 | |
| 7922 | 4679.958333 | 15918 | 101.4 | 6 | 0 | |
| 1959 | 4200.041667 | 21002 | 36.7 | 1 | 0 | |
| 4095 | 1896.000000 | 17390 | 86.1 | 1 | 0 | |

| | pub.rec | credit.policy_0 | credit.policy_1 |
|---|---|---|---|
| 1279 | 1 | 0 | 1 |
| 262 | 0 | 0 | 1 |
| 7922 | 0 | 1 | 0 |
| 1959 | 0 | 0 | 1 |
| 4095 | 0 | 0 | 1 |

[18]:
```
#applying scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
```

[19]:
```
x_train.columns
```

[19]:
```
Index(['purpose', 'int.rate', 'installment', 'log.annual.inc', 'dti', 'fico',
       'days.with.cr.line', 'revol.bal', 'revol.util', 'inq.last.6mths',
       'delinq.2yrs', 'pub.rec', 'credit.policy_0', 'credit.policy_1'],
      dtype='object')
```

[20]:
```
columns_for_scaling=['purpose','int.rate', 'installment', 'log.annual.inc',␣
↪'dti', 'fico',
       'days.with.cr.line', 'revol.bal', 'revol.util', 'inq.last.6mths',
       'delinq.2yrs', 'pub.rec']
```

[21]:
```
x_train_sc=pd.DataFrame(data=sc.fit_transform(x_train.drop(['credit.
↪policy_0','credit.policy_1'],axis=1)), columns=columns_for_scaling)
x_train_sc.head()
```

[21]:

| | purpose | int.rate | installment | log.annual.inc | dti | fico | \ |
|---|---|---|---|---|---|---|---|
| 0 | 0.045953 | 0.697111 | -0.732563 | -0.879480 | 1.630647 | -0.758297 | |

```
1 -1.151643 -1.097370     0.000840       -0.687085  0.992340  0.432807
2 -0.552845  0.794514     0.196102        0.507739 -0.386055 -1.817057
3  0.644751 -0.984980    -0.690234       -0.451634  0.435457  0.432807
4 -1.151643 -1.374596    -0.473953        0.677862 -1.267180  1.359221

   days.with.cr.line  revol.bal  revol.util  inq.last.6mths  delinq.2yrs  \
0          -0.396417  -0.488924    0.722720       -0.267287    -0.306244
1          -0.035727  -0.542009   -1.391680        0.189642    -0.306244
2           0.048393  -0.025831    1.872899        2.017358    -0.306244
3          -0.143924   0.148064   -0.355142       -0.267287    -0.306244
4          -1.067223   0.024517    1.346020       -0.267287    -0.306244

    pub.rec
0  3.471782
1 -0.243413
2 -0.243413
3 -0.243413
4 -0.243413
```

[22]:
```
appending_columns= x_train[['credit.policy_0','credit.policy_1']].reset_index()
appending_columns.drop('index', axis=1).head()
```

[22]:
```
   credit.policy_0  credit.policy_1
0                0                1
1                0                1
2                1                0
3                0                1
4                0                1
```

[23]:
```
x_train_sc=pd.concat([x_train_sc, appending_columns], axis=1)
x_train_sc.head()
```

[23]:
```
    purpose  int.rate  installment  log.annual.inc       dti      fico  \
0  0.045953  0.697111    -0.732563       -0.879480  1.630647 -0.758297
1 -1.151643 -1.097370     0.000840       -0.687085  0.992340  0.432807
2 -0.552845  0.794514     0.196102        0.507739 -0.386055 -1.817057
3  0.644751 -0.984980    -0.690234       -0.451634  0.435457  0.432807
4 -1.151643 -1.374596    -0.473953        0.677862 -1.267180  1.359221

   days.with.cr.line  revol.bal  revol.util  inq.last.6mths  delinq.2yrs  \
0          -0.396417  -0.488924    0.722720       -0.267287    -0.306244
1          -0.035727  -0.542009   -1.391680        0.189642    -0.306244
2           0.048393  -0.025831    1.872899        2.017358    -0.306244
3          -0.143924   0.148064   -0.355142       -0.267287    -0.306244
4          -1.067223   0.024517    1.346020       -0.267287    -0.306244

    pub.rec  index  credit.policy_0  credit.policy_1
```

```
0  3.471782   1279                0                1
1 -0.243413    262                0                1
2 -0.243413   7922                1                0
3 -0.243413   1959                0                1
4 -0.243413   4095                0                1
```

[24]:
```
x_test.columns
```

[24]:
```
Index(['purpose', 'int.rate', 'installment', 'log.annual.inc', 'dti', 'fico',
       'days.with.cr.line', 'revol.bal', 'revol.util', 'inq.last.6mths',
       'delinq.2yrs', 'pub.rec', 'credit.policy_0', 'credit.policy_1'],
      dtype='object')
```

[25]:
```
x_test.shape
```

[25]:
```
(1916, 14)
```

[26]:
```
x_test.head()
```

[26]:

| | purpose | int.rate | installment | log.annual.inc | dti | fico | \ |
|---|---|---|---|---|---|---|---|
| 8806 | 2 | 0.1791 | 86.66 | 11.107210 | 2.45 | 647 | |
| 8676 | 5 | 0.0976 | 128.62 | 10.752484 | 7.29 | 707 | |
| 7411 | 6 | 0.0788 | 112.61 | 10.712059 | 11.79 | 717 | |
| 2791 | 1 | 0.1189 | 238.79 | 10.692036 | 6.98 | 712 | |
| 2501 | 0 | 0.0932 | 111.82 | 11.373663 | 13.50 | 727 | |

| | days.with.cr.line | revol.bal | revol.util | inq.last.6mths | delinq.2yrs | \ |
|---|---|---|---|---|---|---|
| 8806 | 720.000000 | 1601 | 50.0 | 3 | 0 | |
| 8676 | 990.000000 | 0 | 19.3 | 0 | 0 | |
| 7411 | 2820.041667 | 1773 | 20.9 | 3 | 1 | |
| 2791 | 5040.000000 | 9144 | 25.5 | 0 | 0 | |
| 2501 | 5011.000000 | 18695 | 53.7 | 3 | 0 | |

| | pub.rec | credit.policy_0 | credit.policy_1 |
|---|---|---|---|
| 8806 | 0 | 1 | 0 |
| 8676 | 0 | 1 | 0 |
| 7411 | 0 | 0 | 1 |
| 2791 | 0 | 0 | 1 |
| 2501 | 0 | 0 | 1 |

[27]:
```
x_test_sc = sc.transform(x_test.drop(['credit.policy_0', 'credit.policy_1'],␣
↪axis=1))
x_test_sc = pd.DataFrame(x_test_sc, columns=columns_for_scaling)
x_test_sc.shape
```

[27]:
```
(1916, 12)
```

[28]:
```
appending_columns=x_test[['credit.policy_0', 'credit.policy_1']].reset_index()
appending_columns.drop('index', axis=1).head()
```

[28]:

| | credit.policy_0 | credit.policy_1 |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 1 | 0 |
| 2 | 0 | 1 |
| 3 | 0 | 1 |
| 4 | 0 | 1 |

[29]:
```
x_test_sc=pd.concat([x_test_sc, appending_columns], axis=1)
x_test_sc.head()
```

[29]:

| | purpose | int.rate | installment | log.annual.inc | dti | fico | \ |
|---|---|---|---|---|---|---|---|
| 0 | 0.045953 | 2.116960 | -1.124439 | 0.285179 | -1.480919 | -1.684712 | |
| 1 | 1.842347 | -0.936278 | -0.921687 | -0.287588 | -0.777182 | -0.096573 | |
| 2 | 2.441145 | -1.640584 | -0.999048 | -0.352861 | -0.122880 | 0.168117 | |
| 3 | -0.552845 | -0.138316 | -0.389345 | -0.385192 | -0.822256 | 0.035772 | |
| 4 | -1.151643 | -1.101116 | -1.002865 | 0.715414 | 0.125754 | 0.432807 | |

| | days.with.cr.line | revol.bal | revol.util | inq.last.6mths | delinq.2yrs | \ |
|---|---|---|---|---|---|---|
| 0 | -1.538481 | -0.515535 | 0.102863 | 0.646571 | -0.306244 | |
| 1 | -1.430284 | -0.570296 | -0.954337 | -0.724216 | -0.306244 | |
| 2 | -0.696931 | -0.509652 | -0.899239 | 0.646571 | 1.583000 | |
| 3 | 0.192672 | -0.257532 | -0.740831 | -0.724216 | -0.306244 | |
| 4 | 0.181051 | 0.069154 | 0.230278 | 0.646571 | -0.306244 | |

| | pub.rec | index | credit.policy_0 | credit.policy_1 |
|---|---|---|---|---|
| 0 | -0.243413 | 8806 | 1 | 0 |
| 1 | -0.243413 | 8676 | 1 | 0 |
| 2 | -0.243413 | 7411 | 0 | 1 |
| 3 | -0.243413 | 2791 | 0 | 1 |
| 4 | -0.243413 | 2501 | 0 | 1 |

[30]:
```
x_train_sc.drop('index', axis=1, inplace=True)
```

[31]:
```
x_test_sc.drop('index', axis=1, inplace=True)
```

[32]:
```
print(x_train_sc.shape, x_train.shape)
print(x_test_sc.shape, x_test.shape)
```

```
(7662, 14) (7662, 14)
(1916, 14) (1916, 14)
```

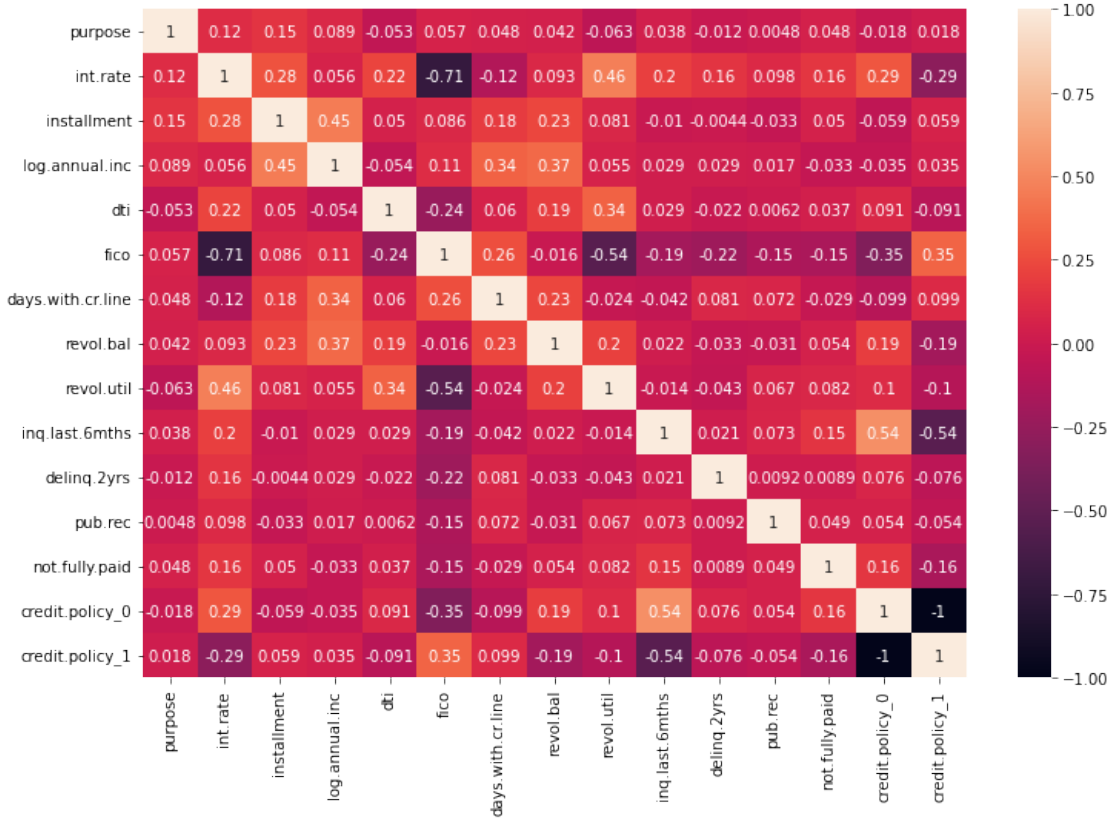
[33]:
```
y_train.head()
```

[33]:
```
1279    0
262     1
7922    1
1959    0
4095    0
Name: not.fully.paid, dtype: int64
```

[34]:
```
y_test.head()
```

[34]:
```
8806    0
8676    0
7411    0
2791    0
2501    0
Name: not.fully.paid, dtype: int64
```

[35]:
```
#correlation matrix
plt.figure(figsize=(12,8))
sns.heatmap(df_loan.corr(), annot=True)
plt.show()
```



[36]:
```
#checking vif score for correlation among variables
vif_data = pd.DataFrame()
vif_data['features'] = x_train.columns
from statsmodels.stats.outliers_influence import variance_inflation_factor as␣
↪vif
vif_data['score'] = [vif(x_train_sc.values, i)
                     for i in range(len(x_train_sc.columns))]

vif_data
```

[36]:

| | features | score |
|---|---|---|
| 0 | purpose | 1.069554 |
| 1 | int.rate | 2.852813 |
| 2 | installment | 1.600678 |
| 3 | log.annual.inc | 1.561518 |
| 4 | dti | 1.214252 |
| 5 | fico | 3.378618 |
| 6 | days.with.cr.line | 1.315642 |
| 7 | revol.bal | 1.459020 |
| 8 | revol.util | 1.767070 |
| 9 | inq.last.6mths | 1.470540 |
| 10 | delinq.2yrs | 1.142994 |
| 11 | pub.rec | 1.051614 |
| 12 | credit.policy_0 | 1.564075 |
| 13 | credit.policy_1 | 1.136516 |

[37]:
```
#there is no multicolinearity
```

[38]:
```
#building model
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Input
```

[39]:
```
model=Sequential()
```

[40]:
```
model.add(Input(shape=(x_train_sc.shape[1],)))
```

[41]:
```
#first hidden layer
model.add(Dense(units=128,
                activation="relu"))
```

[42]:
```
#second hidden layer
model.add(Dense(units=64,
                activation='relu'))
#third hidden layer
model.add(Dense(units=32,
                activation='relu'))
```

```
#output layer
model.add(Dense(units=1,
                activation='sigmoid'))
```

[43]:
```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 128)               1920

 dense_1 (Dense)             (None, 64)                8256

 dense_2 (Dense)             (None, 32)                2080

 dense_3 (Dense)             (None, 1)                 33

=================================================================
Total params: 12,289
Trainable params: 12,289
Non-trainable params: 0
_________________________________________________________________
```

[44]:
```
#compilation
model.compile(optimizer='adam',
              loss='binary_crossentropy',
              metrics=['accuracy'])
```

[45]:
```
#fitting data
result = model.fit(x_train_sc,
                   y_train,
                   validation_data=(x_test_sc, y_test),
                   epochs=100)
```

```
Epoch 1/100
240/240 [==============================] - 1s 3ms/step - loss: 0.4338 -
accuracy: 0.8354 - val_loss: 0.4175 - val_accuracy: 0.8403
Epoch 2/100
240/240 [==============================] - 0s 2ms/step - loss: 0.4137 -
accuracy: 0.8403 - val_loss: 0.4166 - val_accuracy: 0.8387
Epoch 3/100
240/240 [==============================] - 0s 2ms/step - loss: 0.4080 -
accuracy: 0.8423 - val_loss: 0.4181 - val_accuracy: 0.8361
```

```
Epoch 4/100
240/240 [==============================] - 0s 2ms/step - loss: 0.4060 -
accuracy: 0.8429 - val_loss: 0.4191 - val_accuracy: 0.8387
Epoch 5/100
240/240 [==============================] - 0s 2ms/step - loss: 0.4024 -
accuracy: 0.8426 - val_loss: 0.4198 - val_accuracy: 0.8382
Epoch 6/100
240/240 [==============================] - 0s 2ms/step - loss: 0.3982 -
accuracy: 0.8433 - val_loss: 0.4190 - val_accuracy: 0.8377
Epoch 7/100
240/240 [==============================] - 0s 2ms/step - loss: 0.3962 -
accuracy: 0.8440 - val_loss: 0.4192 - val_accuracy: 0.8340
Epoch 8/100
240/240 [==============================] - 0s 2ms/step - loss: 0.3930 -
accuracy: 0.8446 - val_loss: 0.4183 - val_accuracy: 0.8398
Epoch 9/100
240/240 [==============================] - 0s 2ms/step - loss: 0.3910 -
accuracy: 0.8447 - val_loss: 0.4236 - val_accuracy: 0.8377
Epoch 10/100
240/240 [==============================] - 0s 2ms/step - loss: 0.3873 -
accuracy: 0.8470 - val_loss: 0.4341 - val_accuracy: 0.8392
Epoch 11/100
240/240 [==============================] - 0s 2ms/step - loss: 0.3834 -
accuracy: 0.8480 - val_loss: 0.4297 - val_accuracy: 0.8377
Epoch 12/100
240/240 [==============================] - 0s 2ms/step - loss: 0.3792 -
accuracy: 0.8493 - val_loss: 0.4328 - val_accuracy: 0.8351
Epoch 13/100
240/240 [==============================] - 0s 2ms/step - loss: 0.3761 -
accuracy: 0.8530 - val_loss: 0.4350 - val_accuracy: 0.8351
Epoch 14/100
240/240 [==============================] - 0s 2ms/step - loss: 0.3697 -
accuracy: 0.8533 - val_loss: 0.4510 - val_accuracy: 0.8241
Epoch 15/100
240/240 [==============================] - 0s 2ms/step - loss: 0.3682 -
accuracy: 0.8537 - val_loss: 0.4371 - val_accuracy: 0.8335
Epoch 16/100
240/240 [==============================] - 0s 2ms/step - loss: 0.3603 -
accuracy: 0.8587 - val_loss: 0.4507 - val_accuracy: 0.8272
Epoch 17/100
240/240 [==============================] - 0s 2ms/step - loss: 0.3590 -
accuracy: 0.8570 - val_loss: 0.4493 - val_accuracy: 0.8335
Epoch 18/100
240/240 [==============================] - 0s 2ms/step - loss: 0.3492 -
accuracy: 0.8645 - val_loss: 0.4548 - val_accuracy: 0.8314
Epoch 19/100
240/240 [==============================] - 0s 2ms/step - loss: 0.3450 -
accuracy: 0.8645 - val_loss: 0.4752 - val_accuracy: 0.7985
```

```
Epoch 20/100
240/240 [==============================] - 0s 2ms/step - loss: 0.3395 -
accuracy: 0.8671 - val_loss: 0.4643 - val_accuracy: 0.8184
Epoch 21/100
240/240 [==============================] - 0s 2ms/step - loss: 0.3358 -
accuracy: 0.8717 - val_loss: 0.4867 - val_accuracy: 0.8299
Epoch 22/100
240/240 [==============================] - 0s 2ms/step - loss: 0.3269 -
accuracy: 0.8733 - val_loss: 0.4751 - val_accuracy: 0.8262
Epoch 23/100
240/240 [==============================] - 0s 2ms/step - loss: 0.3209 -
accuracy: 0.8750 - val_loss: 0.4903 - val_accuracy: 0.8220
Epoch 24/100
240/240 [==============================] - 0s 2ms/step - loss: 0.3150 -
accuracy: 0.8810 - val_loss: 0.5064 - val_accuracy: 0.8100
Epoch 25/100
240/240 [==============================] - 0s 2ms/step - loss: 0.3110 -
accuracy: 0.8794 - val_loss: 0.4996 - val_accuracy: 0.8147
Epoch 26/100
240/240 [==============================] - 0s 2ms/step - loss: 0.3031 -
accuracy: 0.8845 - val_loss: 0.5166 - val_accuracy: 0.8168
Epoch 27/100
240/240 [==============================] - 0s 2ms/step - loss: 0.3015 -
accuracy: 0.8824 - val_loss: 0.5120 - val_accuracy: 0.8105
Epoch 28/100
240/240 [==============================] - 0s 2ms/step - loss: 0.2929 -
accuracy: 0.8891 - val_loss: 0.5032 - val_accuracy: 0.8199
Epoch 29/100
240/240 [==============================] - 0s 2ms/step - loss: 0.2859 -
accuracy: 0.8900 - val_loss: 0.5255 - val_accuracy: 0.8225
Epoch 30/100
240/240 [==============================] - 0s 2ms/step - loss: 0.2805 -
accuracy: 0.8927 - val_loss: 0.5355 - val_accuracy: 0.8163
Epoch 31/100
240/240 [==============================] - 0s 2ms/step - loss: 0.2757 -
accuracy: 0.8948 - val_loss: 0.5565 - val_accuracy: 0.8173
Epoch 32/100
240/240 [==============================] - 0s 2ms/step - loss: 0.2716 -
accuracy: 0.8960 - val_loss: 0.5515 - val_accuracy: 0.8168
Epoch 33/100
240/240 [==============================] - 0s 2ms/step - loss: 0.2611 -
accuracy: 0.9016 - val_loss: 0.5834 - val_accuracy: 0.8194
Epoch 34/100
240/240 [==============================] - 0s 2ms/step - loss: 0.2588 -
accuracy: 0.9015 - val_loss: 0.5880 - val_accuracy: 0.8058
Epoch 35/100
240/240 [==============================] - 0s 2ms/step - loss: 0.2573 -
accuracy: 0.9016 - val_loss: 0.5699 - val_accuracy: 0.8194
```

```
Epoch 36/100
240/240 [==============================] - 0s 2ms/step - loss: 0.2503 -
accuracy: 0.9038 - val_loss: 0.5823 - val_accuracy: 0.8090
Epoch 37/100
240/240 [==============================] - 0s 2ms/step - loss: 0.2439 -
accuracy: 0.9071 - val_loss: 0.6260 - val_accuracy: 0.7865
Epoch 38/100
240/240 [==============================] - 0s 2ms/step - loss: 0.2416 -
accuracy: 0.9103 - val_loss: 0.6248 - val_accuracy: 0.8126
Epoch 39/100
240/240 [==============================] - 0s 2ms/step - loss: 0.2343 -
accuracy: 0.9118 - val_loss: 0.6630 - val_accuracy: 0.7970
Epoch 40/100
240/240 [==============================] - 0s 2ms/step - loss: 0.2279 -
accuracy: 0.9171 - val_loss: 0.6332 - val_accuracy: 0.8095
Epoch 41/100
240/240 [==============================] - 0s 2ms/step - loss: 0.2284 -
accuracy: 0.9165 - val_loss: 0.6389 - val_accuracy: 0.8022
Epoch 42/100
240/240 [==============================] - 0s 2ms/step - loss: 0.2197 -
accuracy: 0.9169 - val_loss: 0.6498 - val_accuracy: 0.8011
Epoch 43/100
240/240 [==============================] - 0s 2ms/step - loss: 0.2115 -
accuracy: 0.9221 - val_loss: 0.6792 - val_accuracy: 0.8079
Epoch 44/100
240/240 [==============================] - 0s 2ms/step - loss: 0.2084 -
accuracy: 0.9203 - val_loss: 0.7099 - val_accuracy: 0.8085
Epoch 45/100
240/240 [==============================] - 0s 2ms/step - loss: 0.2066 -
accuracy: 0.9242 - val_loss: 0.7295 - val_accuracy: 0.8178
Epoch 46/100
240/240 [==============================] - 0s 2ms/step - loss: 0.2044 -
accuracy: 0.9244 - val_loss: 0.7026 - val_accuracy: 0.7954
Epoch 47/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1965 -
accuracy: 0.9244 - val_loss: 0.7290 - val_accuracy: 0.7933
Epoch 48/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1972 -
accuracy: 0.9239 - val_loss: 0.7230 - val_accuracy: 0.7928
Epoch 49/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1927 -
accuracy: 0.9287 - val_loss: 0.7480 - val_accuracy: 0.8053
Epoch 50/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1825 -
accuracy: 0.9317 - val_loss: 0.8061 - val_accuracy: 0.8011
Epoch 51/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1812 -
accuracy: 0.9315 - val_loss: 0.7618 - val_accuracy: 0.8048
```

```
Epoch 52/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1761 -
accuracy: 0.9353 - val_loss: 0.7982 - val_accuracy: 0.7891
Epoch 53/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1777 -
accuracy: 0.9329 - val_loss: 0.8046 - val_accuracy: 0.7469
Epoch 54/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1707 -
accuracy: 0.9342 - val_loss: 0.8339 - val_accuracy: 0.7891
Epoch 55/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1702 -
accuracy: 0.9370 - val_loss: 0.8499 - val_accuracy: 0.7975
Epoch 56/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1642 -
accuracy: 0.9366 - val_loss: 0.8470 - val_accuracy: 0.8001
Epoch 57/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1620 -
accuracy: 0.9379 - val_loss: 0.8580 - val_accuracy: 0.7944
Epoch 58/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1560 -
accuracy: 0.9428 - val_loss: 0.8610 - val_accuracy: 0.8085
Epoch 59/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1508 -
accuracy: 0.9427 - val_loss: 0.8885 - val_accuracy: 0.7996
Epoch 60/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1531 -
accuracy: 0.9437 - val_loss: 0.9064 - val_accuracy: 0.7938
Epoch 61/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1464 -
accuracy: 0.9441 - val_loss: 0.9695 - val_accuracy: 0.7803
Epoch 62/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1572 -
accuracy: 0.9410 - val_loss: 0.9323 - val_accuracy: 0.7970
Epoch 63/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1479 -
accuracy: 0.9423 - val_loss: 0.9498 - val_accuracy: 0.7975
Epoch 64/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1365 -
accuracy: 0.9504 - val_loss: 0.9447 - val_accuracy: 0.7891
Epoch 65/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1332 -
accuracy: 0.9509 - val_loss: 0.9749 - val_accuracy: 0.8085
Epoch 66/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1361 -
accuracy: 0.9473 - val_loss: 1.0072 - val_accuracy: 0.7897
Epoch 67/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1322 -
accuracy: 0.9509 - val_loss: 0.9980 - val_accuracy: 0.8001
```

```
Epoch 68/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1252 -
accuracy: 0.9528 - val_loss: 1.0841 - val_accuracy: 0.7771
Epoch 69/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1291 -
accuracy: 0.9504 - val_loss: 1.0276 - val_accuracy: 0.7850
Epoch 70/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1249 -
accuracy: 0.9528 - val_loss: 1.0384 - val_accuracy: 0.7996
Epoch 71/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1160 -
accuracy: 0.9568 - val_loss: 1.0892 - val_accuracy: 0.7902
Epoch 72/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1230 -
accuracy: 0.9571 - val_loss: 1.0959 - val_accuracy: 0.7923
Epoch 73/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1359 -
accuracy: 0.9492 - val_loss: 1.0810 - val_accuracy: 0.7918
Epoch 74/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1158 -
accuracy: 0.9578 - val_loss: 1.1351 - val_accuracy: 0.7944
Epoch 75/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1098 -
accuracy: 0.9598 - val_loss: 1.1556 - val_accuracy: 0.7777
Epoch 76/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1212 -
accuracy: 0.9545 - val_loss: 1.1484 - val_accuracy: 0.7912
Epoch 77/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1054 -
accuracy: 0.9616 - val_loss: 1.2054 - val_accuracy: 0.7891
Epoch 78/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1098 -
accuracy: 0.9598 - val_loss: 1.1755 - val_accuracy: 0.7923
Epoch 79/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1100 -
accuracy: 0.9598 - val_loss: 1.1954 - val_accuracy: 0.7766
Epoch 80/100
240/240 [==============================] - 0s 2ms/step - loss: 0.0988 -
accuracy: 0.9648 - val_loss: 1.2216 - val_accuracy: 0.7865
Epoch 81/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1075 -
accuracy: 0.9601 - val_loss: 1.2687 - val_accuracy: 0.8027
Epoch 82/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1052 -
accuracy: 0.9601 - val_loss: 1.2950 - val_accuracy: 0.7505
Epoch 83/100
240/240 [==============================] - 0s 2ms/step - loss: 0.0959 -
accuracy: 0.9646 - val_loss: 1.2649 - val_accuracy: 0.7996
```

```
Epoch 84/100
240/240 [==============================] - 0s 2ms/step - loss: 0.0869 -
accuracy: 0.9704 - val_loss: 1.2908 - val_accuracy: 0.7797
Epoch 85/100
240/240 [==============================] - 0s 2ms/step - loss: 0.0862 -
accuracy: 0.9699 - val_loss: 1.2717 - val_accuracy: 0.7745
Epoch 86/100
240/240 [==============================] - 0s 2ms/step - loss: 0.0888 -
accuracy: 0.9676 - val_loss: 1.3066 - val_accuracy: 0.7902
Epoch 87/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1066 -
accuracy: 0.9572 - val_loss: 1.3012 - val_accuracy: 0.7876
Epoch 88/100
240/240 [==============================] - 0s 2ms/step - loss: 0.0975 -
accuracy: 0.9666 - val_loss: 1.3595 - val_accuracy: 0.7970
Epoch 89/100
240/240 [==============================] - 0s 2ms/step - loss: 0.1064 -
accuracy: 0.9582 - val_loss: 1.3252 - val_accuracy: 0.7860
Epoch 90/100
240/240 [==============================] - 0s 2ms/step - loss: 0.0886 -
accuracy: 0.9653 - val_loss: 1.2910 - val_accuracy: 0.7938
Epoch 91/100
240/240 [==============================] - 0s 2ms/step - loss: 0.0759 -
accuracy: 0.9722 - val_loss: 1.3367 - val_accuracy: 0.7876
Epoch 92/100
240/240 [==============================] - 0s 2ms/step - loss: 0.0780 -
accuracy: 0.9697 - val_loss: 1.3501 - val_accuracy: 0.7818
Epoch 93/100
240/240 [==============================] - 0s 2ms/step - loss: 0.0771 -
accuracy: 0.9704 - val_loss: 1.3407 - val_accuracy: 0.7756
Epoch 94/100
240/240 [==============================] - 0s 2ms/step - loss: 0.0759 -
accuracy: 0.9726 - val_loss: 1.4218 - val_accuracy: 0.7897
Epoch 95/100
240/240 [==============================] - 0s 2ms/step - loss: 0.0871 -
accuracy: 0.9680 - val_loss: 1.4595 - val_accuracy: 0.7959
Epoch 96/100
240/240 [==============================] - 0s 2ms/step - loss: 0.0872 -
accuracy: 0.9684 - val_loss: 1.4751 - val_accuracy: 0.7876
Epoch 97/100
240/240 [==============================] - 0s 2ms/step - loss: 0.0752 -
accuracy: 0.9736 - val_loss: 1.5155 - val_accuracy: 0.8006
Epoch 98/100
240/240 [==============================] - 0s 2ms/step - loss: 0.0714 -
accuracy: 0.9749 - val_loss: 1.5118 - val_accuracy: 0.7970
Epoch 99/100
240/240 [==============================] - 0s 2ms/step - loss: 0.0610 -
accuracy: 0.9781 - val_loss: 1.5168 - val_accuracy: 0.7818
```

```
Epoch 100/100
240/240 [==============================] - 0s 2ms/step - loss: 0.0922 -
accuracy: 0.9679 - val_loss: 1.5578 - val_accuracy: 0.7839
```

[46]:
```
#prediction
y_train_pred = model.predict(x_train_sc)
y_test_pred = model.predict(x_test_sc)
```

[47]:
```
#accuracy
from sklearn.metrics import confusion_matrix, accuracy_score
```

[48]:
```
#confusion matrix on training data
confusion_matrix(y_train_pred >=0.5,
                 y_train)
```

[48]:
```
array([[6356,  169],
       [  80, 1057]])
```

[49]:
```
#confusion matrix on test data
confusion_matrix(y_test_pred>= 0.5,
                 y_test)
```

[49]:
```
array([[1448,  253],
       [ 161,   54]])
```

[50]:
```
# accuracy score on training data
accuracy_score(y_train_pred >=0.5, y_train)
```

[50]:
```
0.9675019577133908
```

[51]:
```
#accuracy score on test data
accuracy_score(y_test_pred >=0.5, y_test)
```

[51]:
```
0.7839248434237995
```

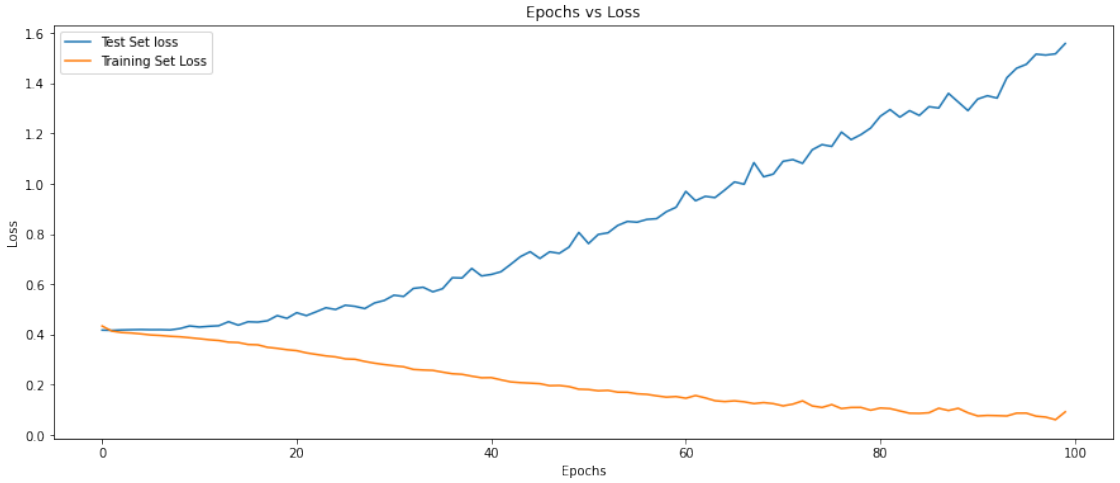
[52]:
```
history = pd.DataFrame(result.history)
history.head()
```

[52]:

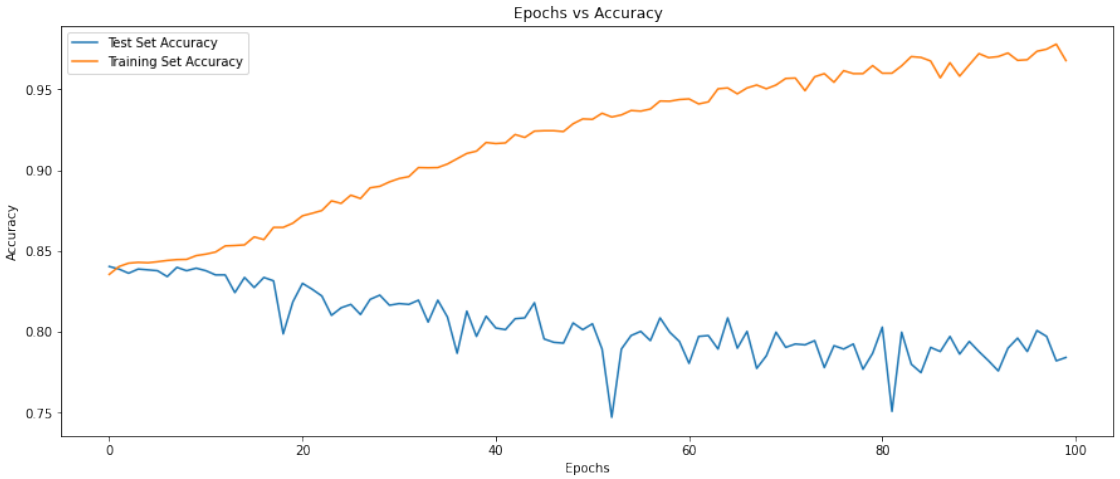
| | loss | accuracy | val_loss | val_accuracy |
|---|---|---|---|---|
| 0 | 0.433824 | 0.835422 | 0.417523 | 0.840292 |
| 1 | 0.413673 | 0.840251 | 0.416624 | 0.838727 |
| 2 | 0.408014 | 0.842339 | 0.418101 | 0.836117 |
| 3 | 0.405994 | 0.842861 | 0.419053 | 0.838727 |
| 4 | 0.402447 | 0.842600 | 0.419753 | 0.838205 |

[53]:
```
plt.figure(figsize= (15,6))
plt.plot(history.val_loss, label='Test Set loss')
```

```
plt.plot(history.loss, label='Training Set Loss')
plt.title('Epochs vs Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```




[54]:
```
plt.figure(figsize= (15,6))
plt.plot(history.val_accuracy, label='Test Set Accuracy')
plt.plot(history.accuracy, label='Training Set Accuracy')
plt.title('Epochs vs Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



[55]:
```
# model is overfitting
# lets apply regularization
model_1 = Sequential()
```

[56]:
```
model_1.add(Input((x_train_sc.shape[1],)))
```

[57]:
```
model_1.add(Dense(units=128, activation='relu', kernel_regularizer = tf.keras.
↪regularizers.L1()))
```

[58]:
```
from tensorflow.keras.layers import Dropout
```

[59]:
```
model_1.add(Dropout(0.2))
```

[60]:
```
model_1.add(Dense(units=128, activation='relu'))
model_1.add(Dropout(0.2))

model_1.add(Dense(units=32, activation='relu'))

model_1.add(Dense(units=1, activation='sigmoid'))
```

[61]:
```
model_1.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_4 (Dense)             (None, 128)               1920

 dropout (Dropout)           (None, 128)               0

 dense_5 (Dense)             (None, 128)               16512

 dropout_1 (Dropout)         (None, 128)               0

 dense_6 (Dense)             (None, 32)                4128

 dense_7 (Dense)             (None, 1)                 33

=================================================================
Total params: 22,593
Trainable params: 22,593
Non-trainable params: 0
_________________________________________________________________
```

[75]:
```
model_1.compile(optimizer='adam',
                loss= 'binary_crossentropy',
                metrics=['accuracy'])
```

[77]:
```
result_1 = model_1.fit(x_train_sc,
                       y_train,
                       epochs=100,
                       validation_data = (x_test_sc , y_test)
                       )
```

```
Epoch 1/100
240/240 [==============================] - 1s 3ms/step - loss: 0.4470 -
accuracy: 0.8400 - val_loss: 0.4425 - val_accuracy: 0.8398
Epoch 2/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4426 -
accuracy: 0.8400 - val_loss: 0.4405 - val_accuracy: 0.8398
Epoch 3/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4400 -
accuracy: 0.8400 - val_loss: 0.4369 - val_accuracy: 0.8398
Epoch 4/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4376 -
accuracy: 0.8400 - val_loss: 0.4418 - val_accuracy: 0.8398
Epoch 5/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4364 -
accuracy: 0.8400 - val_loss: 0.4350 - val_accuracy: 0.8398
Epoch 6/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4376 -
accuracy: 0.8400 - val_loss: 0.4361 - val_accuracy: 0.8398
Epoch 7/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4353 -
accuracy: 0.8400 - val_loss: 0.4357 - val_accuracy: 0.8398
Epoch 8/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4334 -
accuracy: 0.8400 - val_loss: 0.4364 - val_accuracy: 0.8398
Epoch 9/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4349 -
accuracy: 0.8400 - val_loss: 0.4332 - val_accuracy: 0.8398
Epoch 10/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4323 -
accuracy: 0.8400 - val_loss: 0.4343 - val_accuracy: 0.8398
Epoch 11/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4326 -
accuracy: 0.8400 - val_loss: 0.4350 - val_accuracy: 0.8398
Epoch 12/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4328 -
accuracy: 0.8400 - val_loss: 0.4351 - val_accuracy: 0.8398
Epoch 13/100
```

```
240/240 [==============================] - 1s 2ms/step - loss: 0.4311 -
accuracy: 0.8400 - val_loss: 0.4314 - val_accuracy: 0.8398
Epoch 14/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4314 -
accuracy: 0.8400 - val_loss: 0.4343 - val_accuracy: 0.8398
Epoch 15/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4311 -
accuracy: 0.8400 - val_loss: 0.4320 - val_accuracy: 0.8398
Epoch 16/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4318 -
accuracy: 0.8400 - val_loss: 0.4310 - val_accuracy: 0.8398
Epoch 17/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4300 -
accuracy: 0.8400 - val_loss: 0.4326 - val_accuracy: 0.8398
Epoch 18/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4314 -
accuracy: 0.8400 - val_loss: 0.4283 - val_accuracy: 0.8398
Epoch 19/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4288 -
accuracy: 0.8400 - val_loss: 0.4304 - val_accuracy: 0.8398
Epoch 20/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4308 -
accuracy: 0.8400 - val_loss: 0.4311 - val_accuracy: 0.8398
Epoch 21/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4300 -
accuracy: 0.8400 - val_loss: 0.4297 - val_accuracy: 0.8398
Epoch 22/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4293 -
accuracy: 0.8400 - val_loss: 0.4325 - val_accuracy: 0.8398
Epoch 23/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4291 -
accuracy: 0.8400 - val_loss: 0.4247 - val_accuracy: 0.8398
Epoch 24/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4285 -
accuracy: 0.8400 - val_loss: 0.4317 - val_accuracy: 0.8398
Epoch 25/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4284 -
accuracy: 0.8400 - val_loss: 0.4300 - val_accuracy: 0.8398
Epoch 26/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4290 -
accuracy: 0.8400 - val_loss: 0.4304 - val_accuracy: 0.8398
Epoch 27/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4288 -
accuracy: 0.8400 - val_loss: 0.4329 - val_accuracy: 0.8398
Epoch 28/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4281 -
accuracy: 0.8400 - val_loss: 0.4295 - val_accuracy: 0.8398
Epoch 29/100
```

```
240/240 [==============================] - 1s 2ms/step - loss: 0.4297 -
accuracy: 0.8400 - val_loss: 0.4296 - val_accuracy: 0.8398
Epoch 30/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4280 -
accuracy: 0.8400 - val_loss: 0.4313 - val_accuracy: 0.8398
Epoch 31/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4264 -
accuracy: 0.8400 - val_loss: 0.4291 - val_accuracy: 0.8398
Epoch 32/100
240/240 [==============================] - 1s 3ms/step - loss: 0.4284 -
accuracy: 0.8400 - val_loss: 0.4276 - val_accuracy: 0.8398
Epoch 33/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4266 -
accuracy: 0.8400 - val_loss: 0.4325 - val_accuracy: 0.8398
Epoch 34/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4288 -
accuracy: 0.8400 - val_loss: 0.4281 - val_accuracy: 0.8398
Epoch 35/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4279 -
accuracy: 0.8400 - val_loss: 0.4298 - val_accuracy: 0.8398
Epoch 36/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4273 -
accuracy: 0.8400 - val_loss: 0.4278 - val_accuracy: 0.8398
Epoch 37/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4281 -
accuracy: 0.8400 - val_loss: 0.4289 - val_accuracy: 0.8398
Epoch 38/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4282 -
accuracy: 0.8400 - val_loss: 0.4261 - val_accuracy: 0.8398
Epoch 39/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4280 -
accuracy: 0.8400 - val_loss: 0.4291 - val_accuracy: 0.8398
Epoch 40/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4264 -
accuracy: 0.8400 - val_loss: 0.4297 - val_accuracy: 0.8398
Epoch 41/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4273 -
accuracy: 0.8400 - val_loss: 0.4262 - val_accuracy: 0.8398
Epoch 42/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4275 -
accuracy: 0.8400 - val_loss: 0.4275 - val_accuracy: 0.8398
Epoch 43/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4277 -
accuracy: 0.8400 - val_loss: 0.4293 - val_accuracy: 0.8398
Epoch 44/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4260 -
accuracy: 0.8400 - val_loss: 0.4376 - val_accuracy: 0.8398
Epoch 45/100
```

```
240/240 [==============================] - 1s 2ms/step - loss: 0.4298 -
accuracy: 0.8400 - val_loss: 0.4302 - val_accuracy: 0.8398
Epoch 46/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4262 -
accuracy: 0.8400 - val_loss: 0.4289 - val_accuracy: 0.8398
Epoch 47/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4263 -
accuracy: 0.8400 - val_loss: 0.4290 - val_accuracy: 0.8398
Epoch 48/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4270 -
accuracy: 0.8400 - val_loss: 0.4297 - val_accuracy: 0.8398
Epoch 49/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4284 -
accuracy: 0.8400 - val_loss: 0.4283 - val_accuracy: 0.8398
Epoch 50/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4303 -
accuracy: 0.8400 - val_loss: 0.4266 - val_accuracy: 0.8398
Epoch 51/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4279 -
accuracy: 0.8400 - val_loss: 0.4265 - val_accuracy: 0.8398
Epoch 52/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4285 -
accuracy: 0.8400 - val_loss: 0.4293 - val_accuracy: 0.8398
Epoch 53/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4272 -
accuracy: 0.8400 - val_loss: 0.4290 - val_accuracy: 0.8398
Epoch 54/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4290 -
accuracy: 0.8400 - val_loss: 0.4297 - val_accuracy: 0.8398
Epoch 55/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4282 -
accuracy: 0.8400 - val_loss: 0.4293 - val_accuracy: 0.8398
Epoch 56/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4277 -
accuracy: 0.8400 - val_loss: 0.4290 - val_accuracy: 0.8398
Epoch 57/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4276 -
accuracy: 0.8400 - val_loss: 0.4297 - val_accuracy: 0.8398
Epoch 58/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4260 -
accuracy: 0.8400 - val_loss: 0.4254 - val_accuracy: 0.8398
Epoch 59/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4264 -
accuracy: 0.8400 - val_loss: 0.4296 - val_accuracy: 0.8398
Epoch 60/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4289 -
accuracy: 0.8400 - val_loss: 0.4263 - val_accuracy: 0.8398
Epoch 61/100
```

```
240/240 [==============================] - 1s 2ms/step - loss: 0.4279 -
accuracy: 0.8400 - val_loss: 0.4263 - val_accuracy: 0.8398
Epoch 62/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4285 -
accuracy: 0.8400 - val_loss: 0.4292 - val_accuracy: 0.8398
Epoch 63/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4267 -
accuracy: 0.8400 - val_loss: 0.4338 - val_accuracy: 0.8398
Epoch 64/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4259 -
accuracy: 0.8400 - val_loss: 0.4269 - val_accuracy: 0.8398
Epoch 65/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4280 -
accuracy: 0.8400 - val_loss: 0.4278 - val_accuracy: 0.8398
Epoch 66/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4267 -
accuracy: 0.8400 - val_loss: 0.4298 - val_accuracy: 0.8398
Epoch 67/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4274 -
accuracy: 0.8400 - val_loss: 0.4261 - val_accuracy: 0.8398
Epoch 68/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4280 -
accuracy: 0.8400 - val_loss: 0.4291 - val_accuracy: 0.8398
Epoch 69/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4268 -
accuracy: 0.8400 - val_loss: 0.4248 - val_accuracy: 0.8398
Epoch 70/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4279 -
accuracy: 0.8400 - val_loss: 0.4288 - val_accuracy: 0.8398
Epoch 71/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4276 -
accuracy: 0.8400 - val_loss: 0.4258 - val_accuracy: 0.8398
Epoch 72/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4267 -
accuracy: 0.8400 - val_loss: 0.4267 - val_accuracy: 0.8398
Epoch 73/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4278 -
accuracy: 0.8400 - val_loss: 0.4291 - val_accuracy: 0.8398
Epoch 74/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4259 -
accuracy: 0.8400 - val_loss: 0.4288 - val_accuracy: 0.8398
Epoch 75/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4275 -
accuracy: 0.8400 - val_loss: 0.4294 - val_accuracy: 0.8398
Epoch 76/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4284 -
accuracy: 0.8400 - val_loss: 0.4304 - val_accuracy: 0.8398
Epoch 77/100
```

```
240/240 [==============================] - 1s 2ms/step - loss: 0.4269 -
accuracy: 0.8400 - val_loss: 0.4278 - val_accuracy: 0.8398
Epoch 78/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4270 -
accuracy: 0.8400 - val_loss: 0.4286 - val_accuracy: 0.8398
Epoch 79/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4264 -
accuracy: 0.8400 - val_loss: 0.4279 - val_accuracy: 0.8398
Epoch 80/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4281 -
accuracy: 0.8400 - val_loss: 0.4286 - val_accuracy: 0.8398
Epoch 81/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4274 -
accuracy: 0.8400 - val_loss: 0.4291 - val_accuracy: 0.8398
Epoch 82/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4280 -
accuracy: 0.8400 - val_loss: 0.4329 - val_accuracy: 0.8398
Epoch 83/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4266 -
accuracy: 0.8400 - val_loss: 0.4272 - val_accuracy: 0.8398
Epoch 84/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4266 -
accuracy: 0.8400 - val_loss: 0.4290 - val_accuracy: 0.8398
Epoch 85/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4279 -
accuracy: 0.8400 - val_loss: 0.4269 - val_accuracy: 0.8398
Epoch 86/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4275 -
accuracy: 0.8400 - val_loss: 0.4292 - val_accuracy: 0.8398
Epoch 87/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4285 -
accuracy: 0.8400 - val_loss: 0.4264 - val_accuracy: 0.8398
Epoch 88/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4274 -
accuracy: 0.8400 - val_loss: 0.4264 - val_accuracy: 0.8398
Epoch 89/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4269 -
accuracy: 0.8400 - val_loss: 0.4264 - val_accuracy: 0.8398
Epoch 90/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4252 -
accuracy: 0.8400 - val_loss: 0.4252 - val_accuracy: 0.8398
Epoch 91/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4286 -
accuracy: 0.8400 - val_loss: 0.4272 - val_accuracy: 0.8398
Epoch 92/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4287 -
accuracy: 0.8400 - val_loss: 0.4304 - val_accuracy: 0.8398
Epoch 93/100
```

```
240/240 [==============================] - 1s 2ms/step - loss: 0.4271 -
accuracy: 0.8400 - val_loss: 0.4277 - val_accuracy: 0.8398
Epoch 94/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4272 -
accuracy: 0.8400 - val_loss: 0.4283 - val_accuracy: 0.8398
Epoch 95/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4270 -
accuracy: 0.8400 - val_loss: 0.4260 - val_accuracy: 0.8398
Epoch 96/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4264 -
accuracy: 0.8400 - val_loss: 0.4315 - val_accuracy: 0.8398
Epoch 97/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4258 -
accuracy: 0.8400 - val_loss: 0.4275 - val_accuracy: 0.8398
Epoch 98/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4287 -
accuracy: 0.8400 - val_loss: 0.4288 - val_accuracy: 0.8398
Epoch 99/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4263 -
accuracy: 0.8400 - val_loss: 0.4290 - val_accuracy: 0.8398
Epoch 100/100
240/240 [==============================] - 1s 2ms/step - loss: 0.4262 -
accuracy: 0.8400 - val_loss: 0.4270 - val_accuracy: 0.8398
```

[78]:
```
# predictions
y_pred_tr = model_1.predict(x_train_sc)
y_pred_ts = model_1.predict(x_test_sc)
```

[79]:
```
# confusion matrix on training data
confusion_matrix(y_pred_tr >=0.5, y_train)
```

[79]:
```
array([[6436, 1226],
       [   0,    0]])
```

[80]:
```
# confusion matrix on test data
confusion_matrix(y_pred_ts >= 0.5, y_test)
```

[80]:
```
array([[1609,  307],
       [   0,    0]])
```

[81]:
```
# accuracy score on training data
accuracy_score(y_pred_tr >=0.5, y_train)
```

[81]:
```
0.839989558861916
```

[82]:
```
# accuracy score on test data
accuracy_score(y_pred_ts >=0.5, y_test)
```

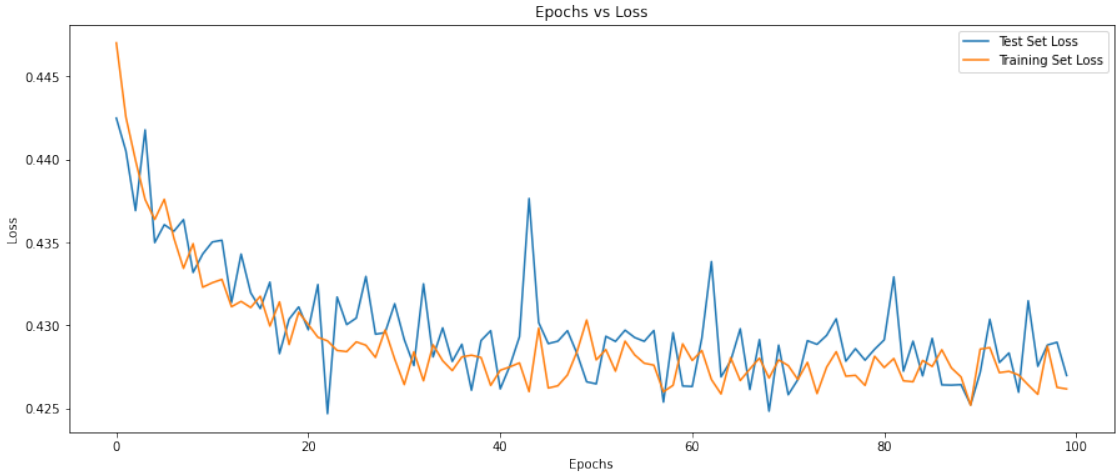
[82]:
```
0.8397703549060542
```

[87]:
```
# history
history_1 = pd.DataFrame(result_1.history)
history_1.head()
```

[87]:

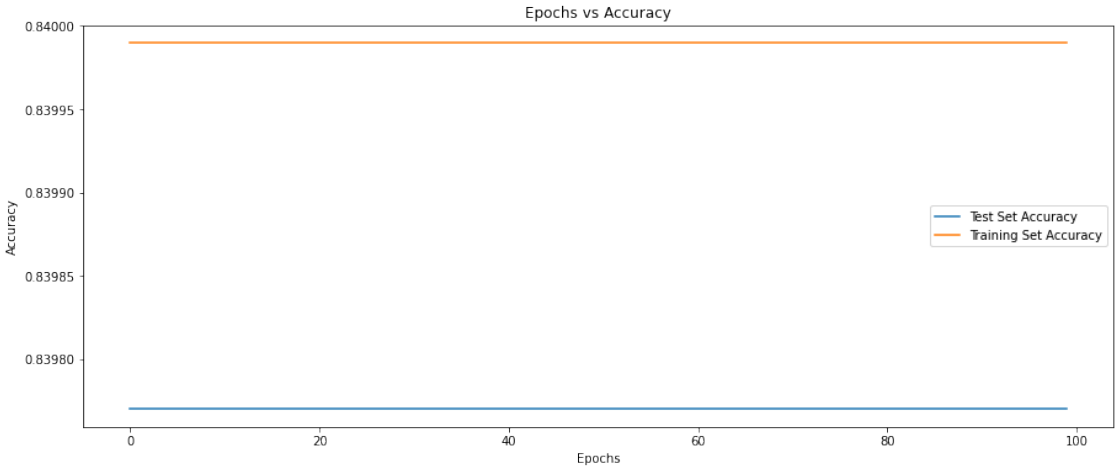
| | loss | accuracy | val_loss | val_accuracy |
|---|---|---|---|---|
| 0 | 0.447038 | 0.83999 | 0.442501 | 0.83977 |
| 1 | 0.442559 | 0.83999 | 0.440511 | 0.83977 |
| 2 | 0.439956 | 0.83999 | 0.436915 | 0.83977 |
| 3 | 0.437589 | 0.83999 | 0.441788 | 0.83977 |
| 4 | 0.436383 | 0.83999 | 0.434987 | 0.83977 |

[88]:
```
# plotting loss vs epochs
plt.figure(figsize = (15,6))
plt.plot(history_1.val_loss, label = 'Test Set Loss')
plt.plot(history_1.loss, label = 'Training Set Loss')
plt.title('Epochs vs Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```




[89]:
```
# plot Epochs vs Accuracy
plt.figure(figsize = (15, 6))
plt.plot(history_1.val_accuracy, label = 'Test Set Accuracy' )
plt.plot(history_1.accuracy, label = 'Training Set Accuracy')
plt.title('Epochs vs Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
```

```
plt.show()
```


Epochs vs Accuracy
0.84000
0.83995
0.83990
0.83985
0.83980
Accuracy
0
20
40
60
80
100
Epochs
Test Set Accuracy
Training Set Accuracy