lending club

March 31, 2023

1 Lending Club Loan Data Analysis

1.0.1 Create a model that predicts whether or not a loan will be default using the historical data.

[1]:
```
# import necessory librarys
```

[2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
print("All library importted")
```

```
All library importted
```

[3]:
```
# load the dataset
df = pd.read_csv("loan_data.csv")
```

[4]:
```
df.head()
```

[4]:

	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	0	0	0

```
2            0        0               0
3            0        0               0
4            1        0               0
```

[5]:
```
df.shape
```

[5]:
```
(9578, 14)
```

[6]:
```
df.describe()
```

[6]:
```
       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
```

[7]:
```
#missing values
df.isna().sum().any()
```

[7]:
```
False
```

[8]:
```
df['not.fully.paid'].value_counts()
# 0- means fully paid , 1-means not paid
```

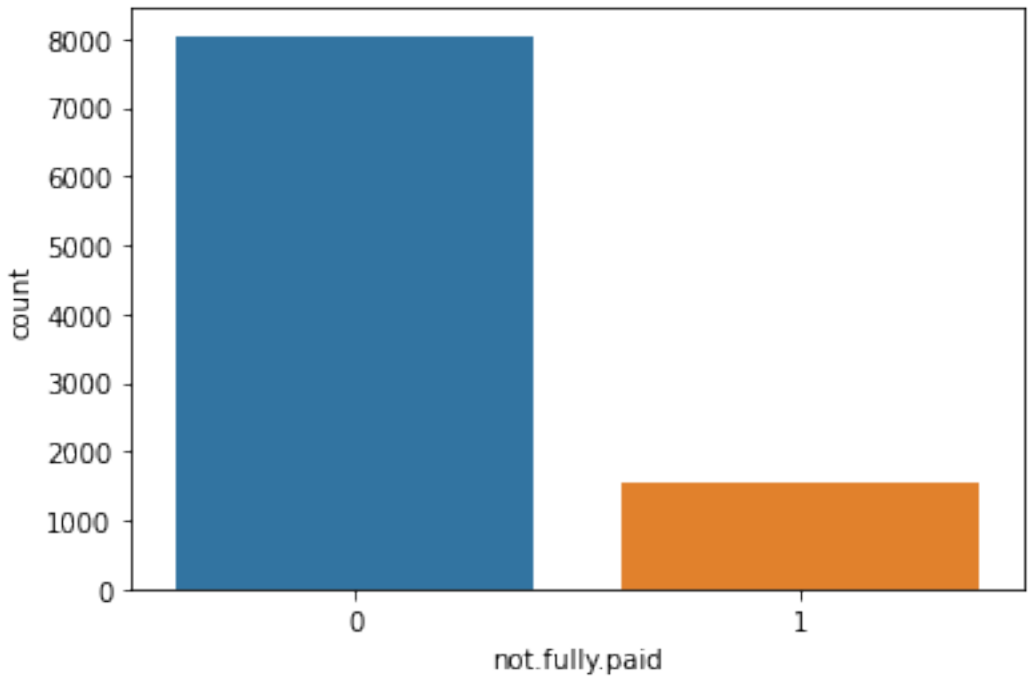
```
# my observation is imbalance data
```

[8]:
```
0    8045
1    1533
Name: not.fully.paid, dtype: int64
```

1.0.2 Exploratory data analysis of different factors of the dataset.

[9]:
```
sns.countplot(df['not.fully.paid'])
plt.show()
```

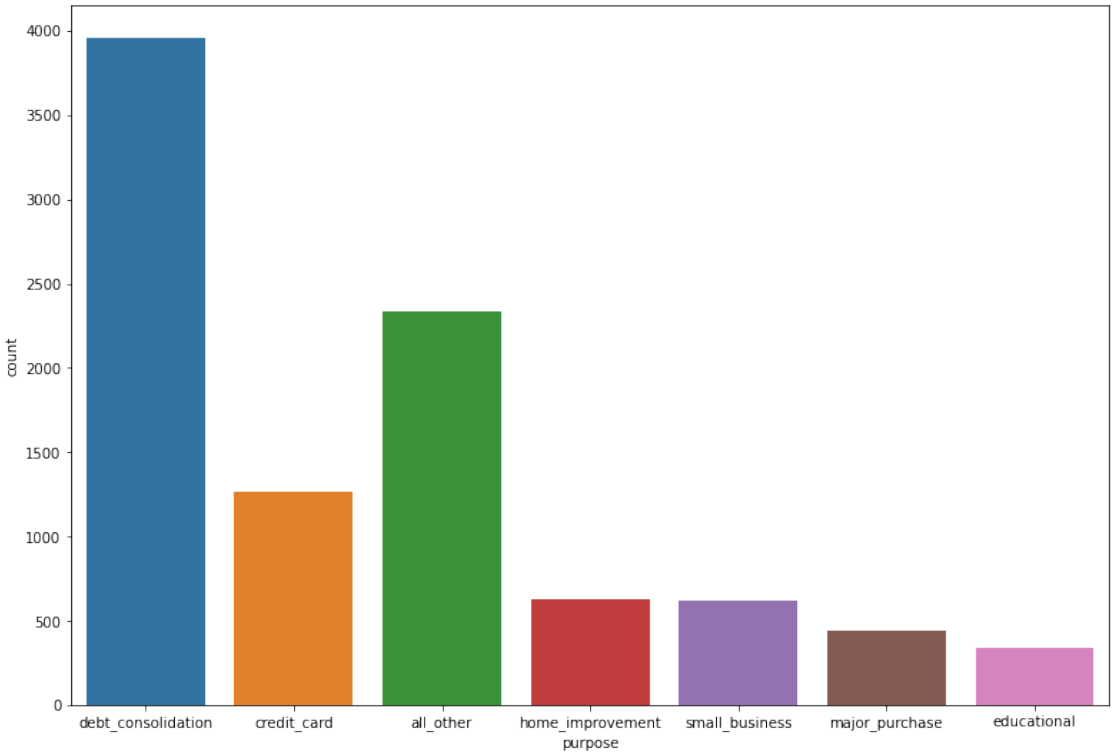
```
/usr/local/lib/python3.7/site-packages/seaborn/_decorators.py:43: FutureWarning:
Pass the following variable as a keyword arg: x. From version 0.12, the only
valid positional argument will be `data`, and passing other arguments without an
explicit keyword will result in an error or misinterpretation.
  FutureWarning
```



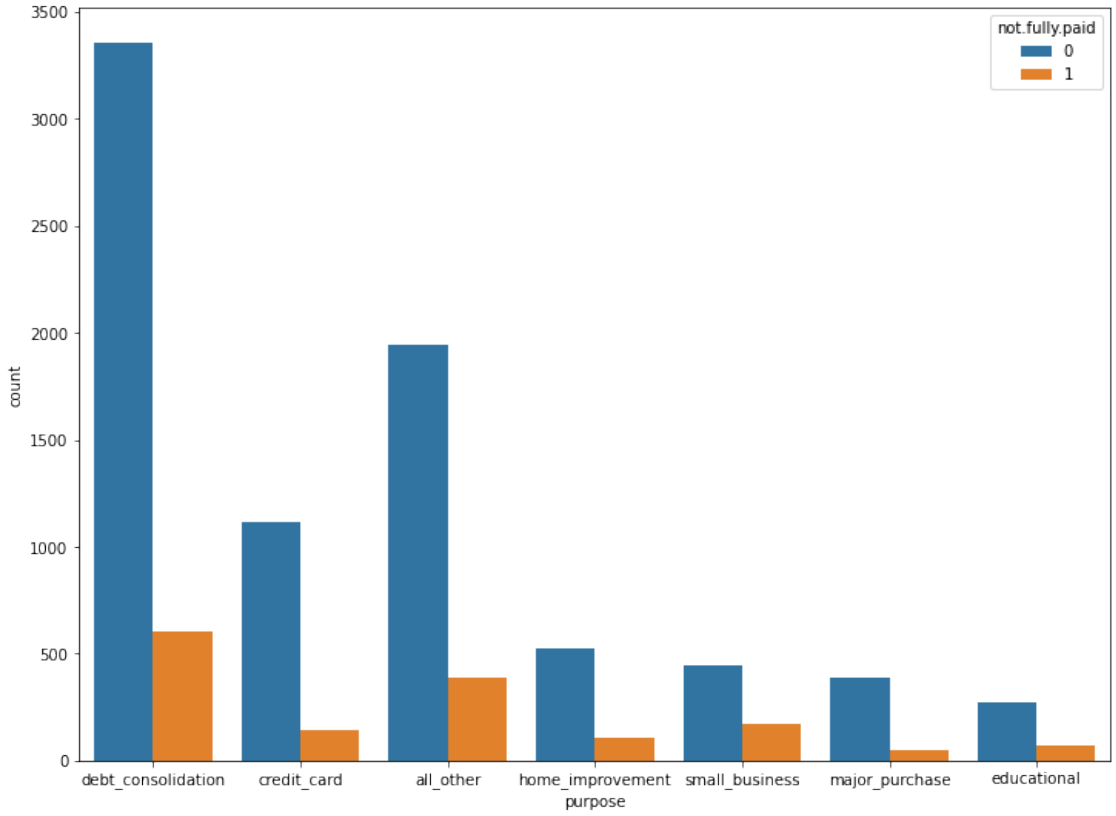
[10]:
```
plt.figure (figsize=(13,9))
sns.countplot(df['purpose'])
plt.show()
```

```
/usr/local/lib/python3.7/site-packages/seaborn/_decorators.py:43: FutureWarning:
Pass the following variable as a keyword arg: x. From version 0.12, the only
valid positional argument will be `data`, and passing other arguments without an
```

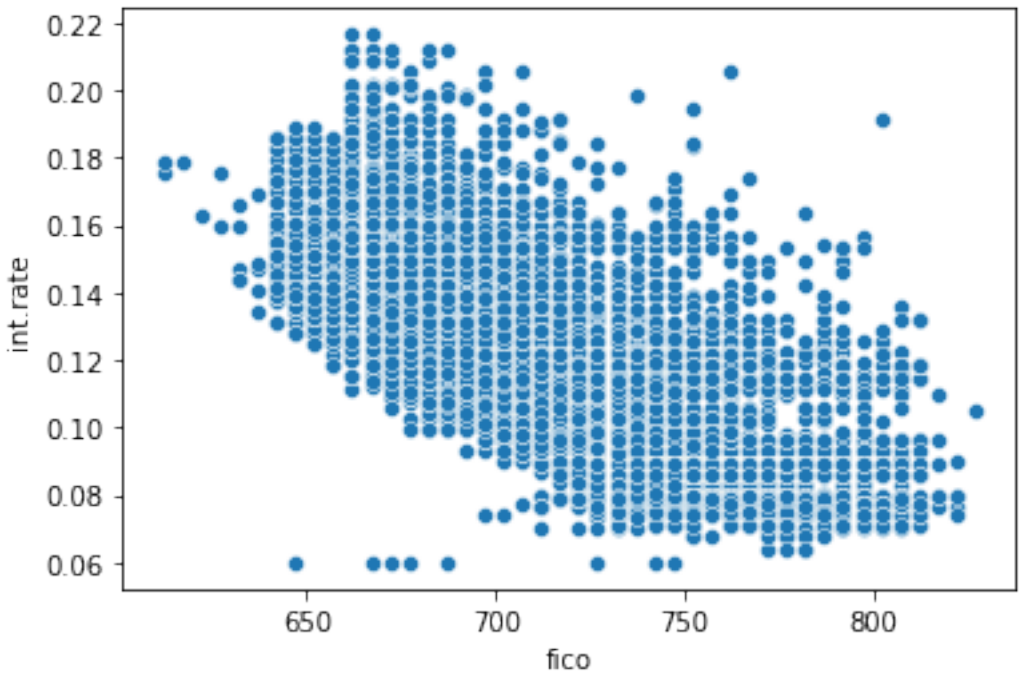
```
explicit keyword will result in an error or misinterpretation.
  FutureWarning
```



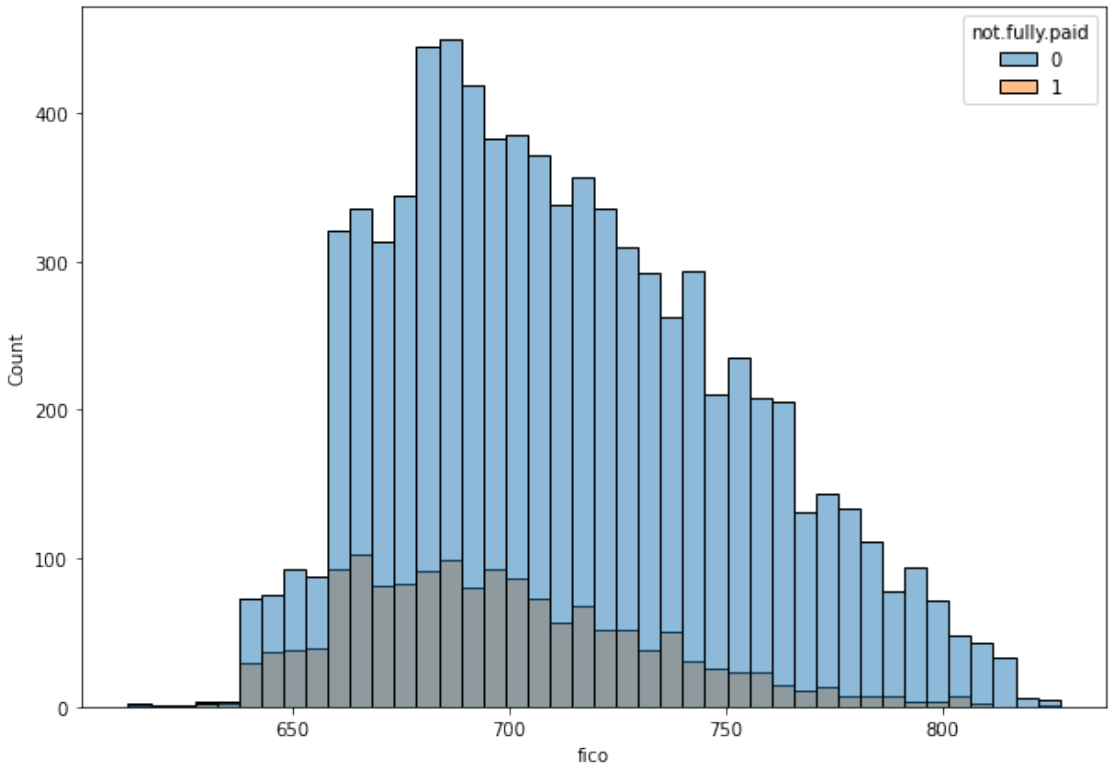
[11]:
```
# purpose --not fully paid
plt.figure(figsize=(12,9))
sns.countplot(x='purpose',hue='not.fully.paid',data=df)
plt.show()
```



[12]:
```
sns.scatterplot(x='fico',y='int.rate',data=df)
plt.show()
```



[13]:
```
plt.figure(figsize=(10,7))
sns.histplot(x='fico',hue='not.fully.paid',data=df)
plt.show()
```



1.1 Feature Transformation

1.2 Transform categorical values into numerical values (discrete)

[14]:
```
df['not.fully.paid'].value_counts()
```

[14]:
```
0    8045
1    1533
Name: not.fully.paid, dtype: int64
```

[15]:
```
not_fully_paid_0=df[df['not.fully.paid']==0]
not_fully_paid_1=df[df['not.fully.paid']==1]
```

[16]:
```
# resample
from sklearn.utils import resample
df_minor_upsample=resample(not_fully_paid_1,replace=True,n_samples=8045)
```

[17]:
```
new_df=pd.concat([not_fully_paid_0,df_minor_upsample])
```

[18]:
```
# shuffle
from sklearn.utils import shuffle
```

```
new_df=shuffle(new_df)
```

[19]:
```
new_df['not.fully.paid'].value_counts()
```

[19]:
```
1    8045
0    8045
Name: not.fully.paid, dtype: int64
```

[20]:
```
new_df.shape
```

[20]:
```
(16090, 14)
```

[21]:
```
new_df.dtypes
```

[21]:
```
credit.policy          int64
purpose               object
int.rate             float64
installment          float64
log.annual.inc       float64
dti                  float64
fico                   int64
days.with.cr.line    float64
revol.bal              int64
revol.util           float64
inq.last.6mths         int64
delinq.2yrs            int64
pub.rec                int64
not.fully.paid         int64
dtype: object
```

[22]:
```
# convert purpose into numerical data
from sklearn.preprocessing import LabelEncoder
le =LabelEncoder()
```

[23]:
```
for i in new_df.columns:
    if new_df[i].dtypes=='object':
        new_df[i]=le.fit_transform(new_df[i])
```

[24]:
```
new_df.head()
```

[24]:

	credit.policy	purpose	int.rate	installment	log.annual.inc	dti	\
8782	0	0	0.1412	119.83	10.714418	10.19	
3492	1	2	0.1221	666.30	11.184421	14.63	
3821	1	0	0.1095	163.57	11.603680	15.75	
8257	0	2	0.1103	818.83	12.323856	12.66	
4877	1	0	0.1600	98.45	10.416191	14.77	

```
      fico  days.with.cr.line  revol.bal  revol.util  inq.last.6mths  \
8782   657        3210.000000       9771        86.4               4
3492   727        4320.000000      28585        60.8               0
3821   717        4680.000000      34267        67.5               2
8257   712        4409.000000     242194        56.0               0
4877   662        1319.958333        339        12.1               2

      delinq.2yrs  pub.rec  not.fully.paid
8782            0        0               0
3492            0        0               1
3821            0        0               0
8257            0        0               1
4877            0        0               1
```

1.3 Additional Feature Engineering

[25]:
```
new_df.corr()
```

[25]:
```
                    credit.policy   purpose  int.rate  installment  \
credit.policy            1.000000  0.009097 -0.288040     0.051985
purpose                  0.009097  1.000000  0.155618     0.198676
int.rate                -0.288040  0.155618  1.000000     0.278762
installment              0.051985  0.198676  0.278762     1.000000
log.annual.inc           0.023310  0.108969  0.088776     0.470014
dti                     -0.107404 -0.038988  0.213374     0.032384
fico                     0.372500  0.064936 -0.679289     0.109998
days.with.cr.line        0.101963  0.060905 -0.096354     0.187717
revol.bal               -0.183294  0.065036  0.087130     0.244774
revol.util              -0.095301 -0.071921  0.417074     0.059197
inq.last.6mths          -0.543335  0.049193  0.183889    -0.016854
delinq.2yrs             -0.059199  0.003047  0.151028     0.000810
pub.rec                 -0.062898  0.008493  0.106161    -0.027508
not.fully.paid          -0.194886  0.062390  0.205697     0.069890

                    log.annual.inc       dti      fico  days.with.cr.line  \
credit.policy             0.023310 -0.107404  0.372500           0.101963
purpose                   0.108969 -0.038988  0.064936           0.060905
int.rate                  0.088776  0.213374 -0.679289          -0.096354
installment               0.470014  0.032384  0.109998           0.187717
log.annual.inc            1.000000 -0.027590  0.105770           0.342234
dti                      -0.027590  1.000000 -0.222464           0.093253
fico                      0.105770 -0.222464  1.000000           0.263120
days.with.cr.line         0.342234  0.093253  0.263120           1.000000
revol.bal                 0.372457  0.160022  0.009329           0.251895
revol.util                0.078417  0.326170 -0.500435           0.014484
inq.last.6mths            0.036594  0.030055 -0.188686          -0.031736
```

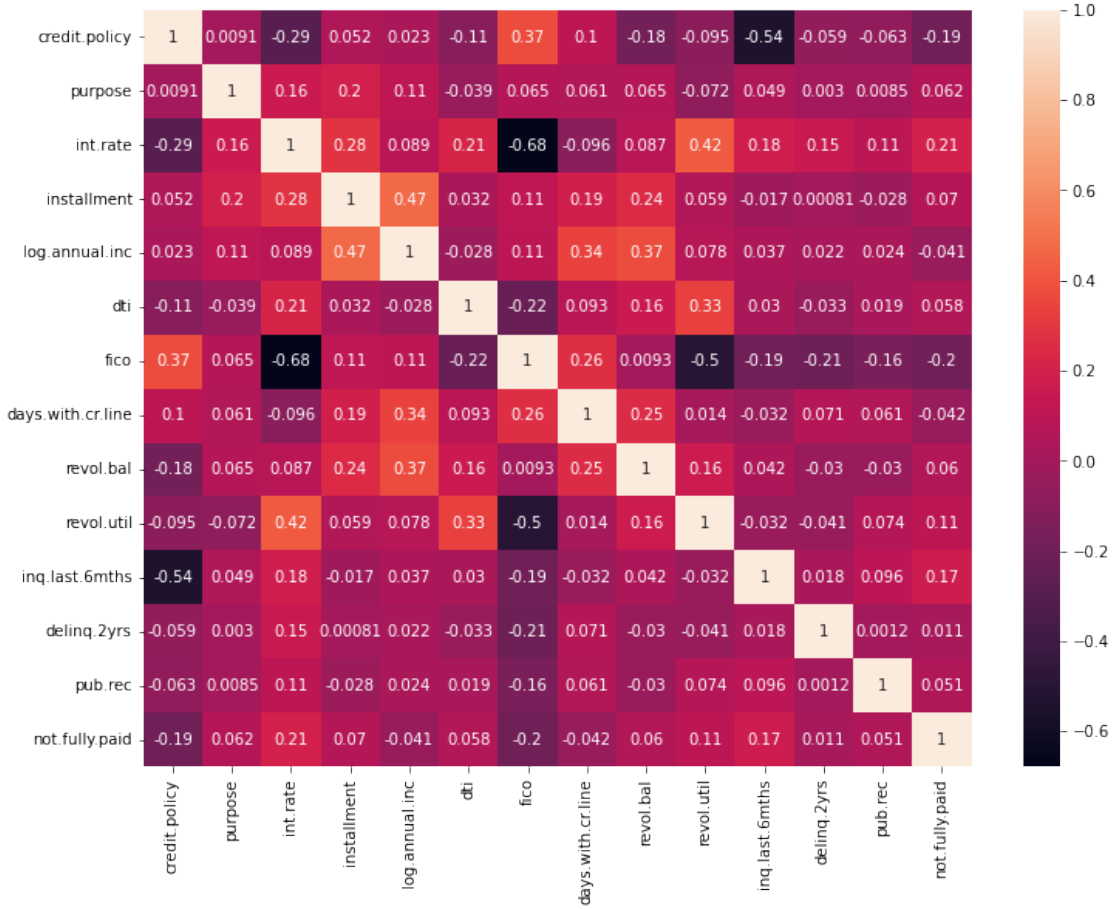
```
delinq.2yrs              0.021930 -0.033153 -0.210431           0.071191
pub.rec                  0.024055  0.019376 -0.155474           0.061456
not.fully.paid          -0.041026  0.058015 -0.199179          -0.041605

                    revol.bal  revol.util  inq.last.6mths  delinq.2yrs  \
credit.policy      -0.183294   -0.095301       -0.543335    -0.059199
purpose             0.065036   -0.071921        0.049193     0.003047
int.rate            0.087130    0.417074        0.183889     0.151028
installment         0.244774    0.059197       -0.016854     0.000810
log.annual.inc      0.372457    0.078417        0.036594     0.021930
dti                 0.160022    0.326170        0.030055    -0.033153
fico                0.009329   -0.500435       -0.188686    -0.210431
days.with.cr.line   0.251895    0.014484       -0.031736     0.071191
revol.bal           1.000000    0.161268        0.042036    -0.030257
revol.util          0.161268    1.000000       -0.032231    -0.040760
inq.last.6mths      0.042036   -0.032231        1.000000     0.018464
delinq.2yrs        -0.030257   -0.040760        0.018464     1.000000
pub.rec            -0.030296    0.074139        0.095552     0.001157
not.fully.paid      0.059934    0.106028        0.171867     0.011340

                     pub.rec  not.fully.paid
credit.policy      -0.062898       -0.194886
purpose             0.008493        0.062390
int.rate            0.106161        0.205697
installment        -0.027508        0.069890
log.annual.inc      0.024055       -0.041026
dti                 0.019376        0.058015
fico               -0.155474       -0.199179
days.with.cr.line   0.061456       -0.041605
revol.bal          -0.030296        0.059934
revol.util          0.074139        0.106028
inq.last.6mths      0.095552        0.171867
delinq.2yrs         0.001157        0.011340
pub.rec             1.000000        0.050653
not.fully.paid      0.050653        1.000000
```

[26]:
```
plt.figure(figsize=(12,9))
sns.heatmap(new_df.corr(),annot=True)
```

[26]:
```
<AxesSubplot:>
```



[27]:
```
# see the sorted corrleation result

new_df.corr().abs()['not.fully.paid'].sort_values(ascending=False)
```

[27]:
```
not.fully.paid       1.000000
int.rate             0.205697
fico                 0.199179
credit.policy        0.194886
inq.last.6mths       0.171867
revol.util           0.106028
installment          0.069890
purpose              0.062390
revol.bal            0.059934
dti                  0.058015
pub.rec              0.050653
days.with.cr.line    0.041605
log.annual.inc       0.041026
delinq.2yrs          0.011340
```

```
Name: not.fully.paid, dtype: float64
```

[28]:
```
# take columns with respect to corrleation
X=new_df[['credit.policy','purpose', 'int.rate', 'installment','fico','revol.
 ↪bal','revol.util','inq.last.6mths','pub.rec']]
```

[29]:
```
y=new_df['not.fully.paid']
```

[30]:
```
# create a train test splt
from sklearn.model_selection import train_test_split
```

[31]:
```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=.2,random_state=42)
```

[32]:
```
# Apply scaling
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
```

[33]:
```
X_train=sc.fit_transform(X_train)
X_test=sc.transform(X_test)
```

[34]:
```
X_train.shape
```

[34]:
```
(12872, 9)
```

[35]:
```
X_test.shape
```

[35]:
```
(3218, 9)
```

2 Create a deep learning keras with tensorflow

[36]:
```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense,Dropout
from tensorflow.keras.callbacks import EarlyStopping
```

[37]:
```
# create the architecture
model=Sequential()
model.add(Dense(19,activation='relu',input_shape=[9]))
model.add(Dropout(0.20))

model.add(Dense(10,activation='relu'))
model.add(Dropout(0.20))

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

[38]:
```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 19)                190

 dropout (Dropout)           (None, 19)                0

 dense_1 (Dense)             (None, 10)                200

 dropout_1 (Dropout)         (None, 10)                0

 dense_2 (Dense)             (None, 1)                 11

=================================================================
Total params: 401
Trainable params: 401
Non-trainable params: 0
_________________________________________________________________
```

[39]:
```
#compile model
model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])
```

[40]:
```
early_stop =EarlyStopping( monitor='val_loss',min_delta=0.
↪001,mode='min',patience=10,verbose=1)
```

[41]:
```
history=model.fit(X_train,y_train,
          epochs=50,
          batch_size=256,
          validation_data=(X_test,y_test),
          callbacks=[early_stop])
```

```
Epoch 1/50
51/51 [==============================] - 1s 4ms/step - loss: 0.7168 - accuracy:
0.5332 - val_loss: 0.6653 - val_accuracy: 0.5914
Epoch 2/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6817 - accuracy:
0.5761 - val_loss: 0.6578 - val_accuracy: 0.6022
Epoch 3/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6785 - accuracy:
0.5754 - val_loss: 0.6554 - val_accuracy: 0.6109
Epoch 4/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6678 - accuracy:
0.5851 - val_loss: 0.6536 - val_accuracy: 0.6085
Epoch 5/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6684 - accuracy:
```

```
0.5852 - val_loss: 0.6529 - val_accuracy: 0.6085
Epoch 6/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6633 - accuracy:
0.5938 - val_loss: 0.6519 - val_accuracy: 0.6122
Epoch 7/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6619 - accuracy:
0.5958 - val_loss: 0.6511 - val_accuracy: 0.6103
Epoch 8/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6607 - accuracy:
0.5954 - val_loss: 0.6510 - val_accuracy: 0.6088
Epoch 9/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6585 - accuracy:
0.5959 - val_loss: 0.6503 - val_accuracy: 0.6081
Epoch 10/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6562 - accuracy:
0.6045 - val_loss: 0.6496 - val_accuracy: 0.6094
Epoch 11/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6567 - accuracy:
0.6052 - val_loss: 0.6491 - val_accuracy: 0.6112
Epoch 12/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6566 - accuracy:
0.6047 - val_loss: 0.6487 - val_accuracy: 0.6134
Epoch 13/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6557 - accuracy:
0.6071 - val_loss: 0.6486 - val_accuracy: 0.6125
Epoch 14/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6543 - accuracy:
0.6057 - val_loss: 0.6480 - val_accuracy: 0.6112
Epoch 15/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6572 - accuracy:
0.6060 - val_loss: 0.6481 - val_accuracy: 0.6137
Epoch 16/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6526 - accuracy:
0.6096 - val_loss: 0.6473 - val_accuracy: 0.6109
Epoch 17/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6528 - accuracy:
0.6103 - val_loss: 0.6471 - val_accuracy: 0.6181
Epoch 18/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6558 - accuracy:
0.6067 - val_loss: 0.6469 - val_accuracy: 0.6147
Epoch 19/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6530 - accuracy:
0.6098 - val_loss: 0.6466 - val_accuracy: 0.6122
Epoch 20/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6519 - accuracy:
0.6106 - val_loss: 0.6459 - val_accuracy: 0.6140
Epoch 21/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6511 - accuracy:
```

```
0.6116 - val_loss: 0.6459 - val_accuracy: 0.6156
Epoch 22/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6539 - accuracy:
0.6132 - val_loss: 0.6457 - val_accuracy: 0.6168
Epoch 23/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6514 - accuracy:
0.6135 - val_loss: 0.6452 - val_accuracy: 0.6134
Epoch 24/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6497 - accuracy:
0.6161 - val_loss: 0.6452 - val_accuracy: 0.6137
Epoch 25/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6504 - accuracy:
0.6148 - val_loss: 0.6451 - val_accuracy: 0.6140
Epoch 26/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6508 - accuracy:
0.6112 - val_loss: 0.6447 - val_accuracy: 0.6153
Epoch 27/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6524 - accuracy:
0.6119 - val_loss: 0.6446 - val_accuracy: 0.6150
Epoch 28/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6510 - accuracy:
0.6141 - val_loss: 0.6443 - val_accuracy: 0.6144
Epoch 29/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6496 - accuracy:
0.6140 - val_loss: 0.6443 - val_accuracy: 0.6168
Epoch 30/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6499 - accuracy:
0.6154 - val_loss: 0.6441 - val_accuracy: 0.6168
Epoch 31/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6487 - accuracy:
0.6119 - val_loss: 0.6442 - val_accuracy: 0.6153
Epoch 32/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6504 - accuracy:
0.6127 - val_loss: 0.6439 - val_accuracy: 0.6181
Epoch 33/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6503 - accuracy:
0.6081 - val_loss: 0.6438 - val_accuracy: 0.6178
Epoch 34/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6494 - accuracy:
0.6125 - val_loss: 0.6434 - val_accuracy: 0.6144
Epoch 35/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6490 - accuracy:
0.6169 - val_loss: 0.6434 - val_accuracy: 0.6178
Epoch 36/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6491 - accuracy:
0.6152 - val_loss: 0.6435 - val_accuracy: 0.6147
Epoch 37/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6475 - accuracy:
```

```
0.6120 - val_loss: 0.6427 - val_accuracy: 0.6200
Epoch 38/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6497 - accuracy:
0.6176 - val_loss: 0.6434 - val_accuracy: 0.6172
Epoch 39/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6483 - accuracy:
0.6154 - val_loss: 0.6432 - val_accuracy: 0.6203
Epoch 40/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6497 - accuracy:
0.6171 - val_loss: 0.6430 - val_accuracy: 0.6172
Epoch 41/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6484 - accuracy:
0.6146 - val_loss: 0.6428 - val_accuracy: 0.6187
Epoch 42/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6470 - accuracy:
0.6193 - val_loss: 0.6429 - val_accuracy: 0.6159
Epoch 43/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6488 - accuracy:
0.6137 - val_loss: 0.6430 - val_accuracy: 0.6165
Epoch 44/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6487 - accuracy:
0.6123 - val_loss: 0.6430 - val_accuracy: 0.6187
Epoch 44: early stopping
```

[42]:
```
history
```

[42]:
```
<keras.callbacks.History at 0x7f419e248710>
```

[43]:
```
history=model.fit(X_train,y_train,
          epochs=50,
          batch_size=256,
          validation_data=(X_test,y_test))
```

```
Epoch 1/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6479 - accuracy:
0.6149 - val_loss: 0.6425 - val_accuracy: 0.6196
Epoch 2/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6475 - accuracy:
0.6179 - val_loss: 0.6425 - val_accuracy: 0.6184
Epoch 3/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6477 - accuracy:
0.6127 - val_loss: 0.6430 - val_accuracy: 0.6181
Epoch 4/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6445 - accuracy:
0.6202 - val_loss: 0.6423 - val_accuracy: 0.6193
Epoch 5/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6487 - accuracy:
0.6133 - val_loss: 0.6430 - val_accuracy: 0.6193
```

```
Epoch 6/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6461 - accuracy:
0.6187 - val_loss: 0.6426 - val_accuracy: 0.6147
Epoch 7/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6468 - accuracy:
0.6161 - val_loss: 0.6422 - val_accuracy: 0.6165
Epoch 8/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6464 - accuracy:
0.6185 - val_loss: 0.6419 - val_accuracy: 0.6218
Epoch 9/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6482 - accuracy:
0.6173 - val_loss: 0.6421 - val_accuracy: 0.6237
Epoch 10/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6462 - accuracy:
0.6210 - val_loss: 0.6419 - val_accuracy: 0.6206
Epoch 11/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6468 - accuracy:
0.6172 - val_loss: 0.6417 - val_accuracy: 0.6203
Epoch 12/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6472 - accuracy:
0.6182 - val_loss: 0.6422 - val_accuracy: 0.6178
Epoch 13/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6479 - accuracy:
0.6177 - val_loss: 0.6417 - val_accuracy: 0.6162
Epoch 14/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6473 - accuracy:
0.6184 - val_loss: 0.6417 - val_accuracy: 0.6218
Epoch 15/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6446 - accuracy:
0.6220 - val_loss: 0.6413 - val_accuracy: 0.6215
Epoch 16/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6447 - accuracy:
0.6220 - val_loss: 0.6412 - val_accuracy: 0.6262
Epoch 17/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6457 - accuracy:
0.6155 - val_loss: 0.6411 - val_accuracy: 0.6271
Epoch 18/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6472 - accuracy:
0.6198 - val_loss: 0.6421 - val_accuracy: 0.6209
Epoch 19/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6446 - accuracy:
0.6193 - val_loss: 0.6416 - val_accuracy: 0.6234
Epoch 20/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6451 - accuracy:
0.6223 - val_loss: 0.6416 - val_accuracy: 0.6243
Epoch 21/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6469 - accuracy:
0.6171 - val_loss: 0.6417 - val_accuracy: 0.6246
```

```
Epoch 22/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6463 - accuracy:
0.6186 - val_loss: 0.6416 - val_accuracy: 0.6209
Epoch 23/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6443 - accuracy:
0.6195 - val_loss: 0.6411 - val_accuracy: 0.6215
Epoch 24/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6455 - accuracy:
0.6193 - val_loss: 0.6409 - val_accuracy: 0.6190
Epoch 25/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6452 - accuracy:
0.6196 - val_loss: 0.6414 - val_accuracy: 0.6224
Epoch 26/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6444 - accuracy:
0.6229 - val_loss: 0.6410 - val_accuracy: 0.6227
Epoch 27/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6453 - accuracy:
0.6223 - val_loss: 0.6409 - val_accuracy: 0.6231
Epoch 28/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6446 - accuracy:
0.6225 - val_loss: 0.6404 - val_accuracy: 0.6252
Epoch 29/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6448 - accuracy:
0.6209 - val_loss: 0.6403 - val_accuracy: 0.6231
Epoch 30/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6439 - accuracy:
0.6211 - val_loss: 0.6407 - val_accuracy: 0.6200
Epoch 31/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6445 - accuracy:
0.6273 - val_loss: 0.6403 - val_accuracy: 0.6227
Epoch 32/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6456 - accuracy:
0.6241 - val_loss: 0.6403 - val_accuracy: 0.6215
Epoch 33/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6438 - accuracy:
0.6219 - val_loss: 0.6401 - val_accuracy: 0.6212
Epoch 34/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6450 - accuracy:
0.6210 - val_loss: 0.6396 - val_accuracy: 0.6187
Epoch 35/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6444 - accuracy:
0.6211 - val_loss: 0.6403 - val_accuracy: 0.6240
Epoch 36/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6432 - accuracy:
0.6255 - val_loss: 0.6399 - val_accuracy: 0.6203
Epoch 37/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6422 - accuracy:
0.6262 - val_loss: 0.6397 - val_accuracy: 0.6243
```

```
Epoch 38/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6440 - accuracy:
0.6261 - val_loss: 0.6397 - val_accuracy: 0.6231
Epoch 39/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6433 - accuracy:
0.6228 - val_loss: 0.6397 - val_accuracy: 0.6215
Epoch 40/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6430 - accuracy:
0.6261 - val_loss: 0.6397 - val_accuracy: 0.6224
Epoch 41/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6431 - accuracy:
0.6196 - val_loss: 0.6393 - val_accuracy: 0.6196
Epoch 42/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6434 - accuracy:
0.6258 - val_loss: 0.6397 - val_accuracy: 0.6240
Epoch 43/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6449 - accuracy:
0.6234 - val_loss: 0.6398 - val_accuracy: 0.6255
Epoch 44/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6436 - accuracy:
0.6231 - val_loss: 0.6398 - val_accuracy: 0.6227
Epoch 45/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6431 - accuracy:
0.6248 - val_loss: 0.6395 - val_accuracy: 0.6240
Epoch 46/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6437 - accuracy:
0.6217 - val_loss: 0.6396 - val_accuracy: 0.6209
Epoch 47/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6429 - accuracy:
0.6228 - val_loss: 0.6394 - val_accuracy: 0.6221
Epoch 48/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6420 - accuracy:
0.6228 - val_loss: 0.6388 - val_accuracy: 0.6255
Epoch 49/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6449 - accuracy:
0.6210 - val_loss: 0.6402 - val_accuracy: 0.6190
Epoch 50/50
51/51 [==============================] - 0s 2ms/step - loss: 0.6422 - accuracy:
0.6200 - val_loss: 0.6394 - val_accuracy: 0.6215
```

[44]:
```
model.evaluate(X_test,y_test)
```

```
101/101 [==============================] - 0s 828us/step - loss: 0.6394 -
accuracy: 0.6215
```

[44]:
```
[0.6394385695457458, 0.6215040683746338]
```

[45]:
```
y_pred=model.predict(X_test)
```

[46]:
```
y_pred
```

[46]:
```
array([[0.49597514],
       [0.6011866 ],
       [0.36741394],
       …,
       [0.4140299 ],
       [0.6708174 ],
       [0.6873083 ]], dtype=float32)
```

[47]:
```
predictions=(y_pred>0.5).astype('int')
```

[48]:
```
predictions
```

[48]:
```
array([[0],
       [1],
       [0],
       …,
       [0],
       [1],
       [1]])
```

[49]:
```
y_test
```

[49]:
```
201     1
5025    1
4066    1
8642    0
7866    1
       ..
8580    1
9340    1
6737    0
9071    1
9021    1
Name: not.fully.paid, Length: 3218, dtype: int64
```

[50]:
```
from sklearn.metrics import␣
 ↪accuracy_score,confusion_matrix,classification_report
accuracy_score(predictions,y_test)
```

[50]:
```
0.6215040397762586
```

[51]:
```
print(classification_report(predictions,y_test))
```

```
              precision    recall  f1-score   support
```

```
           0       0.64      0.62      0.63      1659
           1       0.61      0.63      0.62      1559

    accuracy                           0.62      3218
   macro avg       0.62      0.62      0.62      3218
weighted avg       0.62      0.62      0.62      3218
```

[52]:
```
model.save('loan_default1.h5')
```

[53]:
```
# batch Normalization
from tensorflow.keras.layers import BatchNormalization
```

[54]:
```
# create the architecture model2
# First ANN layer
model1=Sequential()
model1.add(Dense(128,activation='relu',input_shape=[9]))
model1.add(BatchNormalization())
model1.add(Dropout(0.20))

# Second ANN layer
model1.add(Dense(64,activation='tanh'))
model1.add(BatchNormalization())
model1.add(Dropout(0.20))


# third ANN layer
model1.add(Dense(32,activation='relu'))
model1.add(BatchNormalization())
model1.add(Dropout(0.20))

# output layer
model1.add(Dense(1,activation='sigmoid'))
```

[55]:
```
model1.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_3 (Dense)             (None, 128)               1280

 batch_normalization (BatchN  (None, 128)              512
 ormalization)

 dropout_2 (Dropout)         (None, 128)               0

 dense_4 (Dense)             (None, 64)                8256
```

```
batch_normalization_1 (Batc  (None, 64)              256
hNormalization)

dropout_3 (Dropout)          (None, 64)              0

dense_5 (Dense)              (None, 32)              2080

batch_normalization_2 (Batc  (None, 32)              128
hNormalization)

dropout_4 (Dropout)          (None, 32)              0

dense_6 (Dense)              (None, 1)               33

=================================================================
Total params: 12,545
Trainable params: 12,097
Non-trainable params: 448
_________________________________________________________________
```

[56]:
```
# compile the model
model1.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])
```

[57]:
```
history=model1.fit(X_train,y_train,
          epochs=100,
          batch_size=256,
          validation_data=(X_test,y_test))
```

```
Epoch 1/100
51/51 [==============================] - 1s 10ms/step - loss: 0.7692 - accuracy:
0.5635 - val_loss: 0.6613 - val_accuracy: 0.5970
Epoch 2/100
51/51 [==============================] - 0s 4ms/step - loss: 0.7133 - accuracy:
0.5794 - val_loss: 0.6523 - val_accuracy: 0.6106
Epoch 3/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6928 - accuracy:
0.5883 - val_loss: 0.6472 - val_accuracy: 0.6134
Epoch 4/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6764 - accuracy:
0.5999 - val_loss: 0.6439 - val_accuracy: 0.6165
Epoch 5/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6672 - accuracy:
0.6015 - val_loss: 0.6395 - val_accuracy: 0.6212
Epoch 6/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6646 - accuracy:
0.6071 - val_loss: 0.6388 - val_accuracy: 0.6271
Epoch 7/100
```

```
51/51 [==============================] - 0s 4ms/step - loss: 0.6594 - accuracy:
0.6101 - val_loss: 0.6368 - val_accuracy: 0.6249
Epoch 8/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6602 - accuracy:
0.6078 - val_loss: 0.6357 - val_accuracy: 0.6342
Epoch 9/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6498 - accuracy:
0.6178 - val_loss: 0.6348 - val_accuracy: 0.6374
Epoch 10/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6501 - accuracy:
0.6193 - val_loss: 0.6342 - val_accuracy: 0.6349
Epoch 11/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6473 - accuracy:
0.6241 - val_loss: 0.6349 - val_accuracy: 0.6374
Epoch 12/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6457 - accuracy:
0.6200 - val_loss: 0.6332 - val_accuracy: 0.6377
Epoch 13/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6421 - accuracy:
0.6269 - val_loss: 0.6324 - val_accuracy: 0.6417
Epoch 14/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6430 - accuracy:
0.6276 - val_loss: 0.6322 - val_accuracy: 0.6417
Epoch 15/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6398 - accuracy:
0.6304 - val_loss: 0.6318 - val_accuracy: 0.6401
Epoch 16/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6413 - accuracy:
0.6318 - val_loss: 0.6312 - val_accuracy: 0.6485
Epoch 17/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6397 - accuracy:
0.6314 - val_loss: 0.6288 - val_accuracy: 0.6445
Epoch 18/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6378 - accuracy:
0.6312 - val_loss: 0.6298 - val_accuracy: 0.6411
Epoch 19/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6384 - accuracy:
0.6338 - val_loss: 0.6307 - val_accuracy: 0.6489
Epoch 20/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6366 - accuracy:
0.6302 - val_loss: 0.6304 - val_accuracy: 0.6445
Epoch 21/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6355 - accuracy:
0.6370 - val_loss: 0.6281 - val_accuracy: 0.6523
Epoch 22/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6342 - accuracy:
0.6372 - val_loss: 0.6282 - val_accuracy: 0.6479
Epoch 23/100
```

```
51/51 [==============================] - 0s 4ms/step - loss: 0.6342 - accuracy:
0.6329 - val_loss: 0.6287 - val_accuracy: 0.6448
Epoch 24/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6323 - accuracy:
0.6388 - val_loss: 0.6281 - val_accuracy: 0.6510
Epoch 25/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6335 - accuracy:
0.6397 - val_loss: 0.6274 - val_accuracy: 0.6510
Epoch 26/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6316 - accuracy:
0.6380 - val_loss: 0.6252 - val_accuracy: 0.6504
Epoch 27/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6328 - accuracy:
0.6410 - val_loss: 0.6245 - val_accuracy: 0.6554
Epoch 28/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6268 - accuracy:
0.6448 - val_loss: 0.6263 - val_accuracy: 0.6520
Epoch 29/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6292 - accuracy:
0.6416 - val_loss: 0.6240 - val_accuracy: 0.6510
Epoch 30/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6258 - accuracy:
0.6445 - val_loss: 0.6232 - val_accuracy: 0.6607
Epoch 31/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6275 - accuracy:
0.6407 - val_loss: 0.6219 - val_accuracy: 0.6572
Epoch 32/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6267 - accuracy:
0.6471 - val_loss: 0.6215 - val_accuracy: 0.6613
Epoch 33/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6257 - accuracy:
0.6471 - val_loss: 0.6212 - val_accuracy: 0.6563
Epoch 34/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6237 - accuracy:
0.6536 - val_loss: 0.6216 - val_accuracy: 0.6566
Epoch 35/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6228 - accuracy:
0.6493 - val_loss: 0.6193 - val_accuracy: 0.6579
Epoch 36/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6221 - accuracy:
0.6493 - val_loss: 0.6176 - val_accuracy: 0.6591
Epoch 37/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6221 - accuracy:
0.6529 - val_loss: 0.6177 - val_accuracy: 0.6591
Epoch 38/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6207 - accuracy:
0.6545 - val_loss: 0.6188 - val_accuracy: 0.6576
Epoch 39/100
```

```
51/51 [==============================] - 0s 4ms/step - loss: 0.6209 - accuracy:
0.6509 - val_loss: 0.6166 - val_accuracy: 0.6563
Epoch 40/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6175 - accuracy:
0.6565 - val_loss: 0.6146 - val_accuracy: 0.6628
Epoch 41/100
51/51 [==============================] - 0s 5ms/step - loss: 0.6179 - accuracy:
0.6514 - val_loss: 0.6129 - val_accuracy: 0.6669
Epoch 42/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6195 - accuracy:
0.6586 - val_loss: 0.6130 - val_accuracy: 0.6684
Epoch 43/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6153 - accuracy:
0.6576 - val_loss: 0.6157 - val_accuracy: 0.6678
Epoch 44/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6160 - accuracy:
0.6579 - val_loss: 0.6118 - val_accuracy: 0.6728
Epoch 45/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6140 - accuracy:
0.6639 - val_loss: 0.6106 - val_accuracy: 0.6718
Epoch 46/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6155 - accuracy:
0.6567 - val_loss: 0.6099 - val_accuracy: 0.6681
Epoch 47/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6089 - accuracy:
0.6636 - val_loss: 0.6078 - val_accuracy: 0.6737
Epoch 48/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6095 - accuracy:
0.6693 - val_loss: 0.6083 - val_accuracy: 0.6750
Epoch 49/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6107 - accuracy:
0.6632 - val_loss: 0.6067 - val_accuracy: 0.6706
Epoch 50/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6093 - accuracy:
0.6633 - val_loss: 0.6050 - val_accuracy: 0.6743
Epoch 51/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6074 - accuracy:
0.6666 - val_loss: 0.6044 - val_accuracy: 0.6768
Epoch 52/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6088 - accuracy:
0.6665 - val_loss: 0.6056 - val_accuracy: 0.6746
Epoch 53/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6110 - accuracy:
0.6627 - val_loss: 0.6020 - val_accuracy: 0.6762
Epoch 54/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6068 - accuracy:
0.6648 - val_loss: 0.6005 - val_accuracy: 0.6750
Epoch 55/100
```

```
51/51 [==============================] - 0s 4ms/step - loss: 0.6056 - accuracy:
0.6730 - val_loss: 0.6002 - val_accuracy: 0.6796
Epoch 56/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6060 - accuracy:
0.6648 - val_loss: 0.5983 - val_accuracy: 0.6818
Epoch 57/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6019 - accuracy:
0.6738 - val_loss: 0.5963 - val_accuracy: 0.6833
Epoch 58/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6023 - accuracy:
0.6735 - val_loss: 0.5938 - val_accuracy: 0.6917
Epoch 59/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6019 - accuracy:
0.6702 - val_loss: 0.5955 - val_accuracy: 0.6892
Epoch 60/100
51/51 [==============================] - 0s 4ms/step - loss: 0.6024 - accuracy:
0.6696 - val_loss: 0.5947 - val_accuracy: 0.6852
Epoch 61/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5980 - accuracy:
0.6805 - val_loss: 0.5928 - val_accuracy: 0.6830
Epoch 62/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5997 - accuracy:
0.6733 - val_loss: 0.5924 - val_accuracy: 0.6914
Epoch 63/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5973 - accuracy:
0.6821 - val_loss: 0.5930 - val_accuracy: 0.6886
Epoch 64/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5970 - accuracy:
0.6757 - val_loss: 0.5887 - val_accuracy: 0.6855
Epoch 65/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5961 - accuracy:
0.6778 - val_loss: 0.5887 - val_accuracy: 0.6883
Epoch 66/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5941 - accuracy:
0.6778 - val_loss: 0.5897 - val_accuracy: 0.6930
Epoch 67/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5950 - accuracy:
0.6753 - val_loss: 0.5870 - val_accuracy: 0.6942
Epoch 68/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5942 - accuracy:
0.6789 - val_loss: 0.5879 - val_accuracy: 0.6917
Epoch 69/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5943 - accuracy:
0.6750 - val_loss: 0.5865 - val_accuracy: 0.6933
Epoch 70/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5942 - accuracy:
0.6815 - val_loss: 0.5859 - val_accuracy: 0.6942
Epoch 71/100
```

```
51/51 [==============================] - 0s 4ms/step - loss: 0.5881 - accuracy:
0.6861 - val_loss: 0.5835 - val_accuracy: 0.6952
Epoch 72/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5891 - accuracy:
0.6816 - val_loss: 0.5829 - val_accuracy: 0.6976
Epoch 73/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5885 - accuracy:
0.6869 - val_loss: 0.5809 - val_accuracy: 0.7035
Epoch 74/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5883 - accuracy:
0.6853 - val_loss: 0.5806 - val_accuracy: 0.6992
Epoch 75/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5866 - accuracy:
0.6855 - val_loss: 0.5801 - val_accuracy: 0.6933
Epoch 76/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5874 - accuracy:
0.6862 - val_loss: 0.5800 - val_accuracy: 0.6973
Epoch 77/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5847 - accuracy:
0.6862 - val_loss: 0.5797 - val_accuracy: 0.6958
Epoch 78/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5872 - accuracy:
0.6855 - val_loss: 0.5784 - val_accuracy: 0.6995
Epoch 79/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5855 - accuracy:
0.6868 - val_loss: 0.5780 - val_accuracy: 0.6939
Epoch 80/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5817 - accuracy:
0.6919 - val_loss: 0.5750 - val_accuracy: 0.6998
Epoch 81/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5819 - accuracy:
0.6899 - val_loss: 0.5756 - val_accuracy: 0.6992
Epoch 82/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5804 - accuracy:
0.6913 - val_loss: 0.5752 - val_accuracy: 0.7054
Epoch 83/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5784 - accuracy:
0.6927 - val_loss: 0.5720 - val_accuracy: 0.7119
Epoch 84/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5838 - accuracy:
0.6843 - val_loss: 0.5740 - val_accuracy: 0.7039
Epoch 85/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5770 - accuracy:
0.6916 - val_loss: 0.5718 - val_accuracy: 0.7091
Epoch 86/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5788 - accuracy:
0.6936 - val_loss: 0.5720 - val_accuracy: 0.7098
Epoch 87/100
```

```
51/51 [==============================] - 0s 4ms/step - loss: 0.5777 - accuracy:
0.6923 - val_loss: 0.5729 - val_accuracy: 0.7042
Epoch 88/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5808 - accuracy:
0.6896 - val_loss: 0.5733 - val_accuracy: 0.7048
Epoch 89/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5755 - accuracy:
0.6966 - val_loss: 0.5690 - val_accuracy: 0.7054
Epoch 90/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5752 - accuracy:
0.6919 - val_loss: 0.5671 - val_accuracy: 0.7060
Epoch 91/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5719 - accuracy:
0.6985 - val_loss: 0.5683 - val_accuracy: 0.7070
Epoch 92/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5731 - accuracy:
0.6990 - val_loss: 0.5667 - val_accuracy: 0.7107
Epoch 93/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5733 - accuracy:
0.6983 - val_loss: 0.5653 - val_accuracy: 0.7147
Epoch 94/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5711 - accuracy:
0.7011 - val_loss: 0.5650 - val_accuracy: 0.7213
Epoch 95/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5755 - accuracy:
0.6982 - val_loss: 0.5649 - val_accuracy: 0.7144
Epoch 96/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5718 - accuracy:
0.6960 - val_loss: 0.5640 - val_accuracy: 0.7091
Epoch 97/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5715 - accuracy:
0.6990 - val_loss: 0.5633 - val_accuracy: 0.7119
Epoch 98/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5738 - accuracy:
0.6980 - val_loss: 0.5637 - val_accuracy: 0.7116
Epoch 99/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5652 - accuracy:
0.6978 - val_loss: 0.5585 - val_accuracy: 0.7116
Epoch 100/100
51/51 [==============================] - 0s 4ms/step - loss: 0.5690 - accuracy:
0.6980 - val_loss: 0.5569 - val_accuracy: 0.7172
```

[58]:
```
model1.evaluate(X_test,y_test)
```

```
101/101 [==============================] - 0s 941us/step - loss: 0.5569 -
accuracy: 0.7172
```

[58]: [0.5568888187408447, 0.7172156572341919]

[59]:
```
model1.evaluate(X_train,y_train)
```

```
403/403 [==============================] - 0s 936us/step - loss: 0.4978 -
accuracy: 0.7685
```

[59]: [0.4978392720222473, 0.7684897184371948]

3 Hyperparameter tuning in keras

[60]:
```
!pip install keras-tuner
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: keras-tuner in /usr/local/lib/python3.7/site-
packages (1.1.2)
Requirement already satisfied: ipython in /usr/local/lib/python3.7/site-packages
(from keras-tuner) (7.13.0)
Requirement already satisfied: kt-legacy in /usr/local/lib/python3.7/site-
packages (from keras-tuner) (1.0.4)
Requirement already satisfied: tensorboard in /usr/local/lib/python3.7/site-
packages (from keras-tuner) (2.8.0)
Requirement already satisfied: numpy in /usr/local/lib/python3.7/site-packages
(from keras-tuner) (1.21.5)
Requirement already satisfied: packaging in /usr/local/lib/python3.7/site-
packages (from keras-tuner) (21.0)
Requirement already satisfied: requests in /usr/local/lib/python3.7/site-
packages (from keras-tuner) (2.23.0)
Requirement already satisfied: prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0 in
/usr/local/lib/python3.7/site-packages (from ipython->keras-tuner) (3.0.5)
Requirement already satisfied: traitlets>=4.2 in /usr/local/lib/python3.7/site-
packages (from ipython->keras-tuner) (5.1.1)
Requirement already satisfied: setuptools>=18.5 in
/usr/local/lib/python3.7/site-packages (from ipython->keras-tuner) (41.2.0)
Requirement already satisfied: jedi>=0.10 in /usr/local/lib/python3.7/site-
packages (from ipython->keras-tuner) (0.16.0)
Requirement already satisfied: backcall in /usr/local/lib/python3.7/site-
packages (from ipython->keras-tuner) (0.1.0)
Requirement already satisfied: pickleshare in /usr/local/lib/python3.7/site-
packages (from ipython->keras-tuner) (0.7.5)
Requirement already satisfied: pexpect in /usr/local/lib/python3.7/site-packages
(from ipython->keras-tuner) (4.8.0)
Requirement already satisfied: pygments in /usr/local/lib/python3.7/site-
packages (from ipython->keras-tuner) (2.6.1)
Requirement already satisfied: decorator in /usr/local/lib/python3.7/site-
packages (from ipython->keras-tuner) (4.4.2)
```

```
Requirement already satisfied: pyparsing>=2.0.2 in
/usr/local/lib/python3.7/site-packages (from packaging->keras-tuner) (2.4.6)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in
/usr/local/lib/python3.7/site-packages (from requests->keras-tuner) (1.25.8)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/site-
packages (from requests->keras-tuner) (2.9)
Requirement already satisfied: chardet<4,>=3.0.2 in
/usr/local/lib/python3.7/site-packages (from requests->keras-tuner) (3.0.4)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.7/site-packages (from requests->keras-tuner) (2019.11.28)
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in
/usr/local/lib/python3.7/site-packages (from tensorboard->keras-tuner) (0.4.1)
Requirement already satisfied: protobuf>=3.6.0 in /usr/local/lib/python3.7/site-
packages (from tensorboard->keras-tuner) (3.20.1)
Requirement already satisfied: google-auth<3,>=1.6.3 in
/usr/local/lib/python3.7/site-packages (from tensorboard->keras-tuner) (1.13.1)
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.7/site-
packages (from tensorboard->keras-tuner) (3.2.1)
Requirement already satisfied: wheel>=0.26 in /usr/local/lib/python3.7/site-
packages (from tensorboard->keras-tuner) (0.34.2)
Requirement already satisfied: werkzeug>=0.11.15 in
/usr/local/lib/python3.7/site-packages (from tensorboard->keras-tuner) (1.0.1)
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in
/usr/local/lib/python3.7/site-packages (from tensorboard->keras-tuner)
(1.6.0.post2)
Requirement already satisfied: grpcio>=1.24.3 in /usr/local/lib/python3.7/site-
packages (from tensorboard->keras-tuner) (1.27.2)
Requirement already satisfied: tensorboard-data-server<0.7.0,>=0.6.0 in
/usr/local/lib/python3.7/site-packages (from tensorboard->keras-tuner) (0.6.1)
Requirement already satisfied: absl-py>=0.4 in /usr/local/lib/python3.7/site-
packages (from tensorboard->keras-tuner) (0.9.0)
Requirement already satisfied: six in /usr/local/lib/python3.7/site-packages
(from absl-py>=0.4->tensorboard->keras-tuner) (1.14.0)
Requirement already satisfied: cachetools<5.0,>=2.0.0 in
/usr/local/lib/python3.7/site-packages (from google-
auth<3,>=1.6.3->tensorboard->keras-tuner) (4.0.0)
Requirement already satisfied: pyasn1-modules>=0.2.1 in
/usr/local/lib/python3.7/site-packages (from google-
auth<3,>=1.6.3->tensorboard->keras-tuner) (0.2.8)
Requirement already satisfied: rsa<4.1,>=3.1.4 in /usr/local/lib/python3.7/site-
packages (from google-auth<3,>=1.6.3->tensorboard->keras-tuner) (4.0)
Requirement already satisfied: requests-oauthlib>=0.7.0 in
/usr/local/lib/python3.7/site-packages (from google-auth-
oauthlib<0.5,>=0.4.1->tensorboard->keras-tuner) (1.3.0)
Requirement already satisfied: parso>=0.5.2 in /usr/local/lib/python3.7/site-
packages (from jedi>=0.10->ipython->keras-tuner) (0.6.2)
Requirement already satisfied: wcwidth in /usr/local/lib/python3.7/site-packages
(from prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0->ipython->keras-tuner)
```

```
(0.1.9)
Requirement already satisfied: ptyprocess>=0.5 in /usr/local/lib/python3.7/site-
packages (from pexpect->ipython->keras-tuner) (0.6.0)
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in
/usr/local/lib/python3.7/site-packages (from pyasn1-modules>=0.2.1->google-
auth<3,>=1.6.3->tensorboard->keras-tuner) (0.4.8)
Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python3.7/site-
packages (from requests-oauthlib>=0.7.0->google-auth-
oauthlib<0.5,>=0.4.1->tensorboard->keras-tuner) (3.1.0)
WARNING: You are using pip version 22.0.3; however, version 23.0.1 is
available.
You should consider upgrading via the '/usr/local/bin/python3 -m pip install
--upgrade pip' command.
```

[61]:
```
import keras_tuner
import tensorflow
```

[62]:
```
def build_model(hp):
    model=Sequential()

    # first hidden layer
    model.add(Dense(units=hp.Int('units',min_value=32,max_value=1024,step=16),
                    activation=hp.
 ↪Choice('activation',['relu','tanh']),input_shape=[9]))

    model.add(BatchNormalization())
    model.add(Dropout(hp.Float('rate',min_value=0.1,max_value=0.5,step=0.1)))


    # Second hidden layer
    model.add(Dense(units=hp.Int('units',min_value=32,max_value=1024,step=16),
                    activation=hp.Choice('activation',['relu','tanh'])))

    model.add(BatchNormalization())
    model.add(Dropout(hp.Float('rate',min_value=0.1,max_value=0.5,step=0.1)))

    # third hidden layer
    model.add(Dense(units=hp.Int('units',min_value=32,max_value=1024,step=16),
                    activation=hp.Choice('activation',['relu','tanh'])))

    model.add(BatchNormalization())
    model.add(Dropout(hp.Float('rate',min_value=0.1,max_value=0.5,step=0.1)))

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

```
    learning_rate=hp.Float('learning_rate',min_value=0.001,max_value=0.1,step=0.
↪01)

    model.compile(loss='binary_crossentropy',
                  optimizer=tensorflow.keras.optimizers.Adam(learning_rate),
                  metrics=['accuracy'])
    return model
```

[63]:
```
import keras_tuner as kt
```

[64]:
```
build_model(kt.HyperParameters())
```

[64]:
```
<keras.engine.sequential.Sequential at 0x7f416454b3d0>
```

[65]:
```
rtuner=kt.RandomSearch(hypermodel=build_model,
                       objective='val_accuracy',
                       max_trials=10
                      )
```

```
INFO:tensorflow:Reloading Oracle from existing project
./untitled_project/oracle.json
INFO:tensorflow:Reloading Tuner from ./untitled_project/tuner0.json
```

[66]:
```
rtuner.search(X_train,y_train,
              epochs=50,validation_data=(X_test,y_test),
              verbose=2)
```

```
INFO:tensorflow:Oracle triggered exit
```

[68]:
```
models=rtuner.get_best_models()
```

[69]:
```
len(models)
```

[69]:
```
1
```

[70]:
```
models[0].summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 416)               4160

 batch_normalization (BatchN  (None, 416)              1664
 ormalization)

 dropout (Dropout)           (None, 416)               0
```

```
dense_1 (Dense)              (None, 416)               173472

batch_normalization_1 (Batc  (None, 416)               1664
hNormalization)

dropout_1 (Dropout)          (None, 416)               0

dense_2 (Dense)              (None, 416)               173472

batch_normalization_2 (Batc  (None, 416)               1664
hNormalization)

dropout_2 (Dropout)          (None, 416)               0

dense_3 (Dense)              (None, 1)                 417

=================================================================
Total params: 356,513
Trainable params: 354,017
Non-trainable params: 2,496
_________________________________________________________________
```

[71]:
```
y_pred=models[0].predict(X_test)>=0.5
```

[72]:
```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

[72]:
```
0.7992541951522685
```

[ ]:
```

```