—----------import important libraries—----------
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from keras.models import Sequential
from keras.layers import Dense
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEnocder,LabelEncoder
```

—----------read dataset—----------
```
data = pd.read_csv('Downloads/forestfires.csv')
data
```

	month	day	FFMC	DMC	DC	ISI	temp	RH	wind	rain	...	monthfeb	monthjan	monthjul	monthjun	monthmar	monthr
0	mar	fri	86.2	26.2	94.3	5.1	8.2	51	6.7	0.0	...	0	0	0	0	1	
1	oct	tue	90.6	35.4	669.1	6.7	18.0	33	0.9	0.0	...	0	0	0	0	0	
2	oct	sat	90.6	43.7	686.9	6.7	14.6	33	1.3	0.0	...	0	0	0	0	0	
3	mar	fri	91.7	33.3	77.5	9.0	8.3	97	4.0	0.2	...	0	0	0	0	1	
4	mar	sun	89.3	51.3	102.2	9.6	11.4	99	1.8	0.0	...	0	0	0	0	1	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
512	aug	sun	81.6	56.7	665.6	1.9	27.8	32	2.7	0.0	...	0	0	0	0	0	
513	aug	sun	81.6	56.7	665.6	1.9	21.9	71	5.8	0.0	...	0	0	0	0	0	
514	aug	sun	81.6	56.7	665.6	1.9	21.2	70	6.7	0.0	...	0	0	0	0	0	
515	aug	sat	94.4	146.0	614.7	11.3	25.6	42	4.0	0.0	...	0	0	0	0	0	
516	nov	tue	79.5	3.0	106.7	1.1	11.8	31	4.5	0.0	...	0	0	0	0	0	

```
data.describe()
```

	FFMC	DMC	DC	ISI	temp	RH	wind	rain	area	dayfri	...	monthdec	monthfeb
count	517.000000	517.000000	517.000000	517.000000	517.000000	517.000000	517.000000	517.000000	517.000000	517.000000	...	517.000000	517.000000
mean	90.644681	110.872340	547.940039	9.021663	18.889168	44.288201	4.017602	0.021663	12.847292	0.164410	...	0.017408	0.038685
std	5.520111	64.046482	248.066192	4.559477	5.806625	16.317469	1.791653	0.295959	63.655818	0.371006	...	0.130913	0.193029
min	18.700000	1.100000	7.900000	0.000000	2.200000	15.000000	0.400000	0.000000	0.000000	0.000000	...	0.000000	0.000000
25%	90.200000	68.600000	437.700000	6.500000	15.500000	33.000000	2.700000	0.000000	0.000000	0.000000	...	0.000000	0.000000
50%	91.600000	108.300000	664.200000	8.400000	19.300000	42.000000	4.000000	0.000000	0.520000	0.000000	...	0.000000	0.000000

```
enc = LabelEncoder()
enc.fit(df['month'])
```

```
LabelEncoder()
```

```
enc.classes_
```

```
array(['apr', 'aug', 'dec', 'feb', 'jan', 'jul', 'jun', 'mar', 'may',
       'nov', 'oct', 'sep'], dtype=object)
```

data['monthencoded'] = enc.transform(data['month'])
data.head()

	X	Y	month	day	FFMC	DMC	DC	ISI	temp	RH	wind	rain	area	Log-area	month_encoded
0	7	5	mar	fri	86.2	26.2	94.3	5.1	8.2	51	6.7	0.0	0.0	0.0	7
1	7	4	oct	tue	90.6	35.4	669.1	6.7	18.0	33	0.9	0.0	0.0	0.0	10
2	7	4	oct	sat	90.6	43.7	686.9	6.7	14.6	33	1.3	0.0	0.0	0.0	10
3	8	6	mar	fri	91.7	33.3	77.5	9.0	8.3	97	4.0	0.2	0.0	0.0	7
4	8	6	mar	sun	89.3	51.3	102.2	9.6	11.4	99	1.8	0.0	0.0	0.0	7

enc.fit(data['day'])

```
LabelEncoder()
```

enc.classes_

```
array(['fri', 'mon', 'sat', 'sun', 'thu', 'tue', 'wed'], dtype=object)
```

data['dayencoded'] = enc.transform(data['day'])
data.head(16)

	X	Y	month	day	FFMC	DMC	DC	ISI	temp	RH	wind	rain	area	Log-area	month_encoded	day_encoded
0	7	5	mar	fri	86.2	26.2	94.3	5.1	8.2	51	6.7	0.0	0.0	0.0	7	0
1	7	4	oct	tue	90.6	35.4	669.1	6.7	18.0	33	0.9	0.0	0.0	0.0	10	5
2	7	4	oct	sat	90.6	43.7	686.9	6.7	14.6	33	1.3	0.0	0.0	0.0	10	2
3	8	6	mar	fri	91.7	33.3	77.5	9.0	8.3	97	4.0	0.2	0.0	0.0	7	0
4	8	6	mar	sun	89.3	51.3	102.2	9.6	11.4	99	1.8	0.0	0.0	0.0	7	3
5	8	6	aug	sun	92.3	85.3	488.0	14.7	22.2	29	5.4	0.0	0.0	0.0	1	3
6	8	6	aug	mon	92.3	88.9	495.6	8.5	24.1	27	3.1	0.0	0.0	0.0	1	1
7	8	6	aug	mon	91.5	145.4	608.2	10.7	8.0	86	2.2	0.0	0.0	0.0	1	1
8	8	6	sep	tue	91.0	129.5	692.6	7.0	13.1	63	5.4	0.0	0.0	0.0	11	5
9	7	5	sep	sat	92.5	88.0	698.6	7.1	22.8	40	4.0	0.0	0.0	0.0	11	2
10	7	5	sep	sat	92.5	88.0	698.6	7.1	17.8	51	7.2	0.0	0.0	0.0	11	2

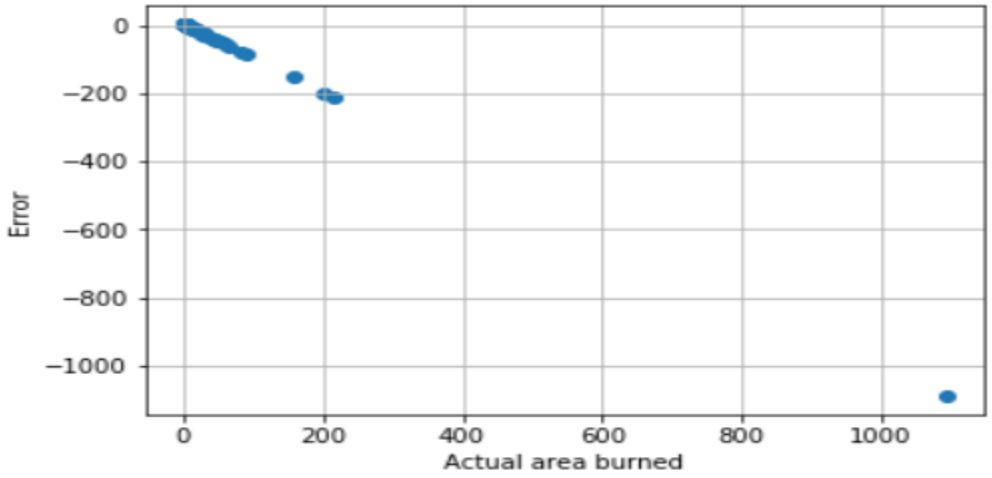
xdata = data.drop(['area','Log-area','month','day'],axis=1)
ydata = data['Log-area']

```
x_train,x_test,y_train,y_test = train_test_split(xdata,ydata,test_size=0.3)
y_train = y_train.reshape(y_train.size,1)

def rec(m,n,tol):
    if type(m)!='numpy.ndarray':
        m=np.array(m)
            if type(n)!='numpy.ndarray':
        n=np.array(n)
        l=m.size
        percent=0
for i in range(1):
    if np.abs(10**m[i]-10**n[i]<=tol:
                percent+=1
                return 100*(percent/1)

tol_max=10

model = Sequential()
model.add(Dense(12,kernel_initializer='uniform',activation='relu'))
model.add(Dense(8,kernel_initializer='uniform',activation='relu'))
model.add(Dense(1,kernel_initializer='uniform',activation='sigmoid'))

model.compile(loss='binary_crossentropy',optimizer='adam',metrics='accuracy')

data=x_train
target = y_train
model.fit(data,target,batch_size=10,verbose=0,epochs=100)

a = model.predict(x_test)
print('Deep Network:',np.sqrt(np.mean((y_test-a.reshape(a.size))**2)))
```

```
Deep Network: 0.610289206294
```

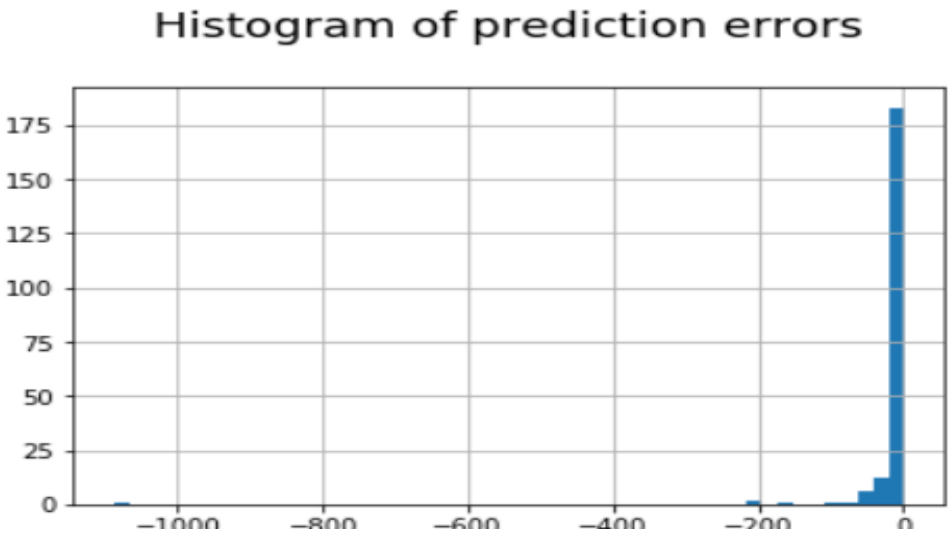
```
plt.xlabel('Actual area burned')
plt.ylabel('Error')
plt.scatter(10**(y_test),10**(a.reshape(a.size,))-10**(y_test))
```

```
<matplotlib.collections.PathCollection at 0x1aa6ba28d68>
```




```
plt.title('Histogram of prediction errors\n',fontsize=18)
plt.xlabel('Prediction error ',fontsize=16)
plt.hist(10**(a.reshape(a.size,))-10**(y_test),bins=50)
```

```
(array([   1.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
          0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
          0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
          0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
          0.,    0.,    0.,    0.,    2.,    0.,    1.,    0.,    0.,
          1.,    1.,    6.,   12.,  183.]),
 array([-1089.03599032, -1067.21491772, -1045.39384512, -1023.57277251,
        -1001.75169991,  -979.9306273 ,  -958.1095547 ,  -936.2884821 ,
         -914.46740949,  -892.64633689,  -870.82526428,  -849.00419168,
         -827.18311908,  -805.36204647,  -783.54097387,  -761.71990126,
         -739.89882866,  -718.07775606,  -696.25668345,  -674.43561085,
         -652.61453824,  -630.79346564,  -608.97239304,  -587.15132043,
         -565.33024783,  -543.50917522,  -521.68810262,  -499.86703002,
         -478.04595741,  -456.22488481,  -434.4038122 ,  -412.5827396 ,
         -390.761667  ,  -368.94059439,  -347.11952179,  -325.29844918,
         -303.47737658,  -281.65630398,  -259.83523137,  -238.01415877,
         -216.19308616,  -194.37201356,  -172.55094096,  -150.72986835,
         -128.90879575,  -107.08772314,   -85.26665054,   -63.44557794,
```



```
rec_NN=[]
for i in range(tol_max):
    rec_NN.append(rec(a,y_test,i))
plt.title('REC Curve for the Deep Network\n')
plt.xlabel('Absolute error (tolerance) in prediction')
plt.ylabel('Percentage of correct prediction')
plt.xticks([i for i in range(0,tol_max+1,5)])
plt.ylim(-10,100)
plt.yticks([i*20 for i in range(6)])
plt.plot(range(tol_max),rec_NN)
```