In [275...
```
import numpy as np
from scipy import stats
import matplotlib.pyplot as plt
import pandas as pd
import scipy.stats as stats
import statistics
from tqdm import tqdm
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score, roc_auc_score
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.linear_model import Ridge,Lasso,LogisticRegression
import seaborn as sns
data_df=pd.read_csv("movieReplicationSet.csv")
movies=data_df[data_df.columns[:400]]
movies2=movies.copy() #data after cleaning
movie_names=movies.columns
s=movies.shape
```

In [276...
```
for m in range(s[1]):
    for user in range(s[0]):
        if np.isnan(movies.iloc[user,m]):
            movies2.iloc[user,m]=(movies.iloc[:,m].mean()+movies.iloc[user,:

movies2=movies2.drop(896,axis="index")
s=movies2.shape

movies2.to_csv("new.csv",index=False)
```

1

In [277...
```
df=pd.read_csv("new.csv")
```

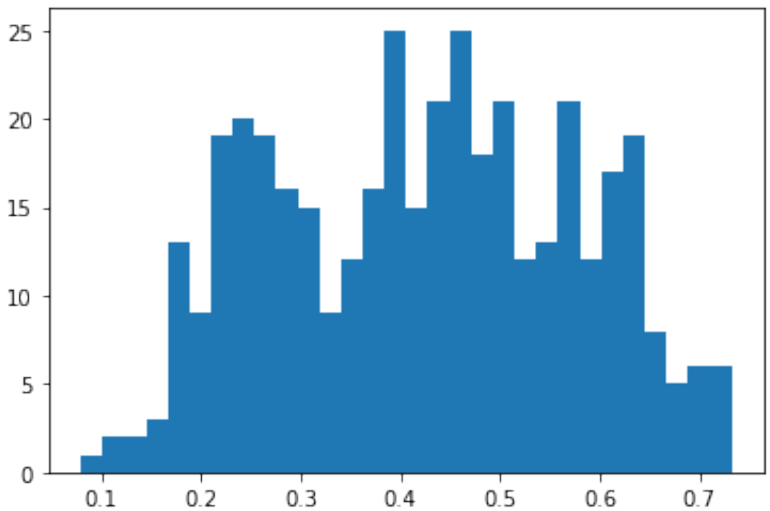
In [278...
```
cod=[]
for m in tqdm(range(s[1])):
    temp=[]
    for mm in range(s[1]):
        if m!=mm:
            x=df.iloc[:,mm].to_numpy()
            y=df.iloc[:,m].to_numpy()
            reg = LinearRegression().fit(x.reshape(-1,1), y)  #mm predict m
            y_hat = reg.predict(x.reshape(-1,1))
            r2 = r2_score(y,y_hat)
            temp.append(r2)
        else:
            temp.append(0)
    cod.append(temp)
```

```
100%|██████████| 400/400 [00:53<00:00,  7.47it/s]
```

In [279…
```
best=[np.max(cod[i]) for i in range(s[1])]
print(np.mean(best))
```

0.42378171067196035

In [280…
```
plt.hist(best,bins=30)
plt.show()
```



In [281…
```
rank=np.argsort(best)
data=[[movie_names[r],np.max(cod[r]), movie_names[cod[r].index(max(cod[r]))]
data2=[[movie_names[r],np.max(cod[r]),movie_names[cod[r].index(max(cod[r]))]
for i in data2:
    data.append(i)
ans1 = pd.DataFrame(data, columns=['movies',"cod","best predictor"])
```

In [282…
```
ans1
```

Out[282]:

	movies	cod	best predictor
0	Heavy Traffic (1973)	0.692734	Ran (1985)
1	The Final Conflict (1981)	0.700188	The Lookout (2007)
2	The Straight Story (1999)	0.700569	Congo (1995)
3	Congo (1995)	0.700569	The Straight Story (1999)
4	The Bandit (1996)	0.711222	Best Laid Plans (1999)
5	Best Laid Plans (1999)	0.711222	The Bandit (1996)
6	Patton (1970)	0.713554	The Lookout (2007)
7	The Lookout (2007)	0.713554	Patton (1970)
8	I.Q. (1994)	0.731507	Erik the Viking (1989)
9	Erik the Viking (1989)	0.731507	I.Q. (1994)
10	Avatar (2009)	0.079485	Bad Boys (1995)
11	Interstellar (2014)	0.111343	Torque (2004)
12	Black Swan (2010)	0.117080	Sorority Boys (2002)
13	Clueless (1995)	0.141426	Escape from LA (1996)
14	The Cabin in the Woods (2012)	0.143887	The Evil Dead (1981)
15	La La Land (2016)	0.148514	The Lookout (2007)
16	Titanic (1997)	0.154136	Cocktail (1988)
17	13 Going on 30 (2004)	0.160164	Can't Hardly Wait (1998)
18	The Fast and the Furious (2001)	0.168991	Terminator 3: Rise of the Machines (2003)
19	Grown Ups 2 (2013)	0.171119	The Core (2003)

2

In [283…

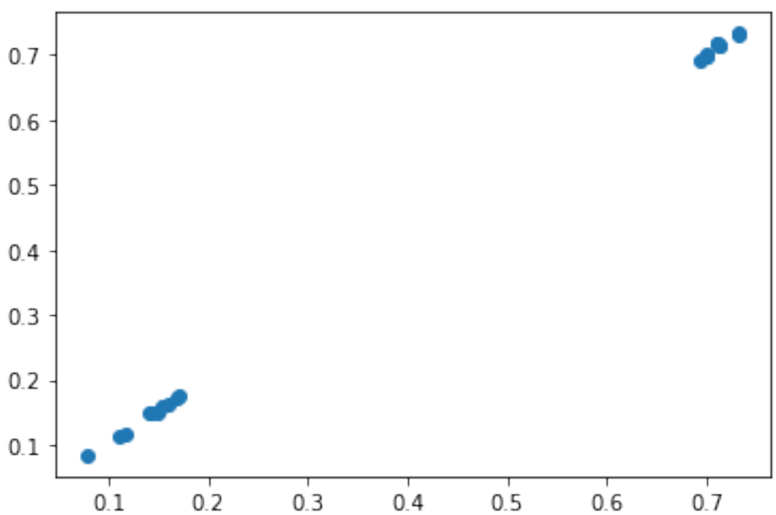
```
data_df=data_df.drop(896,axis="index")
```

In [286…

```
cod_new=[]
cod_old=[]
gender=data_df.iloc[:,-3].to_numpy()
sib=data_df.iloc[:,-2].to_numpy()
social=data_df.iloc[:,-1].to_numpy()
for m in range(20):
    best_p=ans1.iloc[m,2]
    x_best_p=df[best_p].to_numpy()
    x_with0 = np.concatenate((gender.reshape(-1,1), sib.reshape(-1,1), socia
    x=[]
    y_old=df[ans1["movies"][m]].to_numpy()
    y=[]
    for i in range(len(x_with0)):
        if np.isnan(x_with0[i][0])==False:
            x.append(x_with0[i])
            y.append(y_old[i])
    reg = LinearRegression().fit(x,y)
    y_hat = reg.predict(x)
    r2 = r2_score(y,y_hat)
    cod_new.append(r2)
for i in data:
    cod_old.append(i[1])
plt.scatter(cod_old,cod_new)
```

Out[286]:

```
<matplotlib.collections.PathCollection at 0x7fd4e87e75b0>
```



3

In [109…
```
names=[]
names2=[]
for r in rank[185:215]:
    names.append(movie_names[r])#movies name in middle cod
for r in rank[216:226]:
    names2.append(movie_names[r])
mov30=df.loc[:, names]
mov10=df.loc[:, names2]
a=[int(x) for x in np.linspace(start = 1, stop = 200, num = 200)]
rmse_m=[]
r_beta=[]
r_alpha=[]
```

In [111…
```
def rmse(predictions, targets):
    return np.sqrt(np.mean((predictions-targets)**2))
```

In [112…
```
for i in tqdm(range(30)):
    X_train, X_test, Y_train, Y_test = train_test_split(mov10, mov30.iloc[:,
    param_grid = {'alpha': a}
    model=Ridge()
    Ridge_reg= GridSearchCV(model, param_grid, scoring='neg_mean_squared_err
    Ridge_reg.fit(X_train, Y_train)
    pred=Ridge_reg.predict(X_test)
    rmse_m.append(rmse(pred,Y_test))
    r_alpha.append(Ridge_reg.best_params_['alpha'])
```

```
100%|██████████| 30/30 [00:36<00:00,  1.20s/it]
```

In [113…
```
for i in tqdm(range(30)):
    X_train, X_test, Y_train, Y_test = train_test_split(mov10, mov30.iloc[:,
    param_grid = {'alpha': a}
    model=Ridge(r_alpha[i])
    model.fit(X_train, Y_train)
    r_beta.append(np.round(model.coef_, 3))
```

```
100%|██████████| 30/30 [00:00<00:00, 645.76it/s]
```

In [118…
```
ans3_1=pd.DataFrame({'Movie': names, 'Best Alpha': r_alpha, 'RSME': np.round
ans3_2 = pd.DataFrame(r_beta, columns = names2)
ans3 = pd.concat([ans3_1, ans3_2], axis = 1)
```

In [119…
```
ans3
```

Out[119]:

	Movie	Best Alpha	RSME	There's Something About Mary (1998)	Predator (1987)	Toy Story (1995)	Shrek 2 (2004)	Shrek (2001)	Just Like Heaven (2005)	St By (19
0	Gone in Sixty Seconds	24	0.338	0.177	0.064	0.019	-0.000	0.006	0.166	0.

	(2000)									
1	Crossroads (2002)	41	0.382	0.156	0.027	-0.002	0.024	0.024	0.164	0.
2	Austin Powers: The Spy Who Shagged Me (1999)	53	0.568	0.240	0.057	0.031	0.013	0.061	0.064	0.
3	Austin Powers in Goldmember (2002)	80	0.496	0.215	0.109	-0.005	0.017	0.071	0.075	0.
4	Goodfellas (1990)	52	0.377	0.175	0.111	0.094	-0.028	0.006	0.038	0.
5	The Big Lebowski (1998)	73	0.367	0.085	0.088	0.034	0.020	0.052	0.083	0.
6	Twister (1996)	25	0.367	0.114	0.126	-0.003	-0.012	0.032	0.106	0.
7	Blues Brothers 2000 (1998)	83	0.393	0.104	0.073	0.035	0.028	-0.000	0.105	0.
8	Dances with Wolves (1990)	71	0.319	0.056	0.156	0.046	0.023	0.012	0.059	0.
9	28 Days Later (2002)	56	0.370	0.114	0.159	0.050	0.005	0.010	0.030	0.
10	Knight and Day (2010)	42	0.450	0.059	0.100	-0.006	0.050	-0.030	0.126	0.
11	The Evil Dead (1981)	52	0.378	0.076	0.169	0.085	-0.012	0.010	0.044	0.
12	The Machinist (2004)	94	0.353	0.057	0.132	0.027	0.016	0.013	0.079	0.
13	The Blue Lagoon (1980)	52	0.312	0.096	0.116	0.022	0.014	0.008	0.099	0.
14	Uptown Girls (2003)	29	0.419	0.262	0.040	0.048	0.017	0.018	0.139	0.
15	Men in Black (1997)	104	0.539	0.070	0.099	0.121	0.038	0.067	0.069	0.
16	Men in Black II (2002)	34	0.505	0.140	0.264	0.066	0.041	0.021	0.020	0.

17	The Green Mile (1999)	71	0.421	0.101	0.139	0.048	0.013	0.016	0.075	0.
18	The Rock (1996)	65	0.399	0.097	0.098	0.057	0.035	0.015	0.165	0.
19	You're Next (2011)	61	0.337	0.097	0.147	0.027	-0.020	0.046	0.083	0.
20	The Poseidon Adventure (1972)	87	0.225	0.107	0.060	0.048	0.040	0.012	0.124	0.
21	The Good the Bad and the Ugly (1966)	63	0.354	0.157	0.039	0.027	-0.028	0.044	0.166	0.
22	Let the Right One In (2008)	20	0.319	0.145	0.113	0.009	0.003	0.013	0.195	0.
23	Equilibrium (2002)	42	0.310	0.066	0.019	0.016	0.032	-0.004	0.147	0.
24	Just Married (2003)	52	0.371	0.118	0.112	0.044	0.028	-0.007	0.146	0.
25	The Mummy Returns (2001)	97	0.487	0.066	0.127	0.040	0.010	0.044	0.018	0.
26	The Mummy (1999)	47	0.571	0.046	0.221	0.078	0.031	0.043	0.040	-0.
27	Reservoir Dogs (1992)	61	0.402	0.109	0.195	0.050	-0.026	0.014	0.030	0.
28	Man on Fire (2004)	31	0.328	0.149	0.032	0.011	0.001	0.034	0.183	0.
29	The Prestige (2006)	60	0.344	0.134	0.158	0.065	0.003	0.024	0.084	0

4

In [131…

```
rmse_m2=[]
l_beta=[]
l_alpha=[]
a2=[round(x,5) for x in np.linspace(start = 0.0001, stop = 0.1, num = 1000)]
for i in tqdm(range(30)):
    X_train, X_test, Y_train, Y_test = train_test_split(mov10, mov30.iloc[:,
    param_grid = {'alpha': a2}
    model=Lasso()
    Lasso_reg= GridSearchCV(model, param_grid, scoring='neg_mean_squared_err
    Lasso_reg.fit(X_train, Y_train)
    pred=Lasso_reg.predict(X_test)
    rmse_m2.append(rmse(pred,Y_test))
    l_alpha.append(Lasso_reg.best_params_['alpha'])
```

```
100%|██████████| 30/30 [03:10<00:00,  6.33s/it]
```

In [132…

```
for i in tqdm(range(30)):
    X_train, X_test, Y_train, Y_test = train_test_split(mov10, mov30.iloc[:,
    param_grid = {'alpha': a}
    model=Lasso(l_alpha[i])
    model.fit(X_train, Y_train)
    l_beta.append(np.round(model.coef_, 3))
```

```
100%|██████████| 30/30 [00:00<00:00, 613.85it/s]
```

In [133…

```
ans4_1=pd.DataFrame({'Movie': names, 'Best Alpha': l_alpha, 'RSME': np.round
ans4_2 = pd.DataFrame(l_beta, columns = names2)
ans4 = pd.concat([ans4_1, ans4_2], axis = 1)
ans4
```

Out[133]:

	Movie	Best Alpha	RSME	There's Something About Mary (1998)	Predator (1987)	Toy Story (1995)	Shrek 2 (2004)	Shrek (2001)	Just Like Heaven (2005)	S B (1
0	Gone in Sixty Seconds (2000)	0.0061	0.341	0.183	0.050	0.016	0.000	0.001	0.164	0
1	Crossroads (2002)	0.0032	0.384	0.170	0.003	-0.000	0.021	0.020	0.183	(
2	Austin Powers: The Spy Who Shagged Me (1999)	0.0046	0.572	0.294	0.036	0.018	0.005	0.064	0.025	(
3	Austin Powers in Goldmember (2002)	0.0106	0.499	0.284	0.100	-0.000	0.004	0.070	0.051	0

4	Goodfellas (1990)	0.0036	0.375	0.210	0.121	0.091	-0.024	0.000	0.005
5	The Big Lebowski (1998)	0.0016	0.365	0.088	0.095	0.021	0.016	0.052	0.088
6	Twister (1996)	0.0044	0.367	0.108	0.123	-0.000	-0.000	0.019	0.096
7	Blues Brothers 2000 (1998)	0.0010	0.395	0.121	0.070	0.025	0.025	-0.006	0.125
8	Dances with Wolves (1990)	0.0045	0.321	0.036	0.199	0.036	0.023	0.004	0.036
9	28 Days Later (2002)	0.0041	0.372	0.123	0.191	0.042	0.000	0.006	0.000
10	Knight and Day (2010)	0.0061	0.457	0.037	0.088	-0.000	0.034	-0.013	0.132
11	The Evil Dead (1981)	0.0072	0.382	0.069	0.203	0.083	-0.000	0.000	0.013
12	The Machinist (2004)	0.0037	0.368	0.035	0.170	0.013	0.009	0.008	0.061
13	The Blue Lagoon (1980)	0.0038	0.316	0.095	0.120	0.013	0.009	0.003	0.096
14	Uptown Girls (2003)	0.0021	0.419	0.297	0.024	0.043	0.016	0.014	0.148
15	Men in Black (1997)	0.0037	0.536	0.065	0.110	0.128	0.027	0.069	0.067
16	Men in Black II (2002)	0.0077	0.506	0.146	0.298	0.061	0.038	0.016	0.000
17	The Green Mile (1999)	0.0002	0.422	0.120	0.183	0.038	0.012	0.010	0.086
18	The Rock (1996)	0.0011	0.398	0.107	0.113	0.055	0.033	0.009	0.220
19	You're Next (2011)	0.0042	0.335	0.104	0.179	0.016	-0.013	0.038	0.082
20	The Poseidon Adventure (1972)	0.0016	0.225	0.126	0.049	0.043	0.040	0.002	0.160
21	The Good the Bad and	0.0050	0.364	0.188	0.010	0.018	-0.020	0.033	0.208

	the Ugly (1966)									
22	Let the Right One In (2008)	0.0024	0.323	0.151	0.116	0.006	0.000	0.013	0.208	C
23	Equilibrium (2002)	0.0035	0.315	0.052	0.000	0.010	0.024	0.000	0.158	C
24	Just Married (2003)	0.0052	0.374	0.123	0.114	0.035	0.018	-0.000	0.172	(
25	The Mummy Returns (2001)	0.0102	0.481	0.052	0.149	0.027	0.000	0.042	0.000	(
26	The Mummy (1999)	0.0095	0.572	0.003	0.244	0.068	0.026	0.039	0.000	-C
27	Reservoir Dogs (1992)	0.0057	0.399	0.116	0.244	0.040	-0.015	0.000	0.000	C
28	Man on Fire (2004)	0.0051	0.330	0.152	0.010	0.006	0.000	0.031	0.192	(
29	The Prestige (2006)	0.0060	0.344	0.157	0.195	0.060	0.000	0.020	0.085	(

5

In [261…

```
X = np.mean(movies, axis = 1).dropna().values
movie_m=np.mean(movies, axis = 0).dropna().values
```

In [272…

```
s=df.shape[0]
temp=np.argsort(movie_m)
mov4_names_ind=temp[198:202]
mov4_names=[]
mov4_m=[]
label=[]
for i in mov4_names_ind:
    mov4_names.append(movie_names[i])
for m in mov4_names_ind:
    temp=[]
    median=df[movie_names[m]].median()
    for i in range(s):
        if df.iloc[i,m]<median:
            temp.append(0)
        else:
            temp.append(1)
    label.append(temp)
auc=[]
beta=[]
df5=pd.DataFrame(np.array(label).T)
```

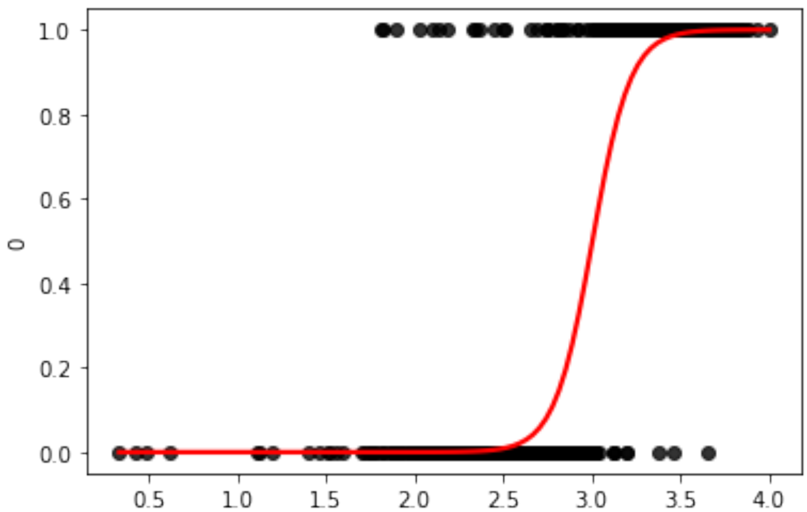
In [273…

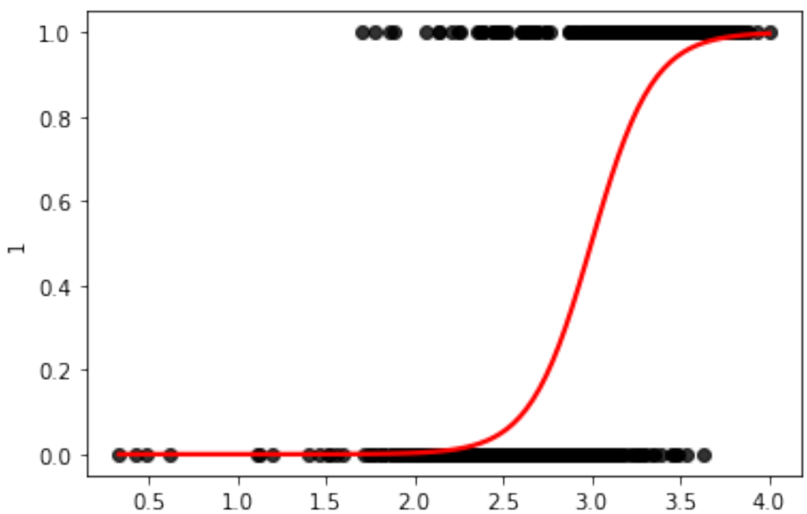
```
for i in range(4):
    X_train, X_test, Y_train, Y_test = train_test_split(X.reshape(-1,1), df5
    params = {'penalty': ['l2', 'none']}
    model = LogisticRegression()
    l = GridSearchCV(model, params)
    l=l.fit(X_train, Y_train)
    pred = l.predict(X_test)
    auc.append(roc_auc_score(Y_test, pred))
    beta.append(l.best_estimator_.fit(X_train, Y_train).coef_)
    plt.figure(i)
    print(mov4_names[i])
    sns.regplot(x = X.reshape(-1,1), y = df5.iloc[:,i], logistic = True, ci
auc, beta
```

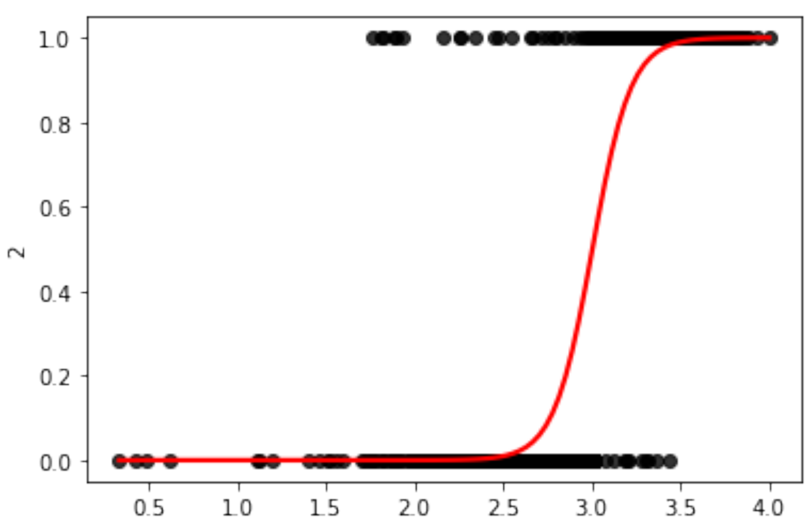
```
Fahrenheit 9/11 (2004)
Happy Gilmore (1996)
Diamonds are Forever (1971)
Scream (1996)
```

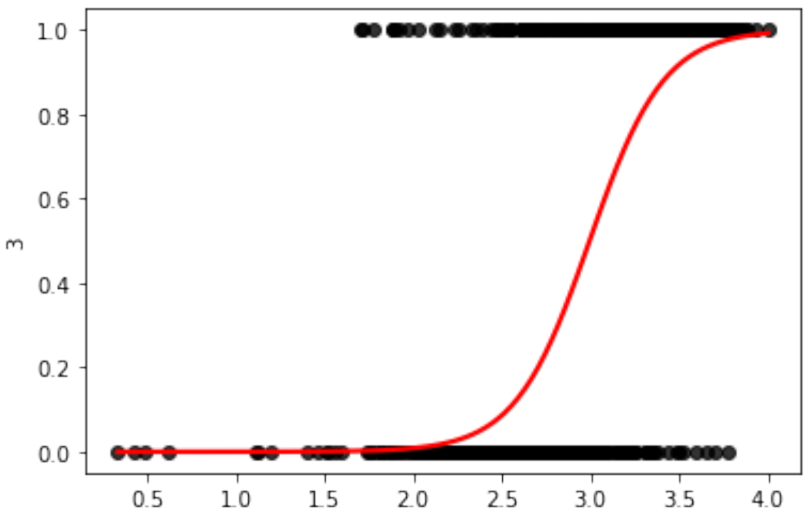
Out[273]:

```
([0.9545454545454546,
  0.8584078119827872,
  0.9540153833429824,
  0.8590909090909089],
 [array([[10.28532939]]),
  array([[5.96005022]]),
  array([[10.33151309]]),
  array([[4.97839277]])])
```


0
0.0
0.2
0.4
0.6
0.8
1.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0


1
0.0
0.2
0.4
0.6
0.8
1.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0


2
0.0
0.2
0.4
0.6
0.8
1.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0



Extra Credit

In [305…
```
cry=data_df.iloc[:,-13].fillna(0).to_numpy()
```

In [308…
```
X_train,X_test, Y_train,Y_test=train_test_split(X.reshape(-1,1),cry,test_siz
model=LinearRegression()
m=model.fit(X_train,Y_train)
y_pred=m.predict(X_test)
print(r2_score(Y_test,y_pred))
```

```
-0.022796791886914O8
```

In [ ]: