8B_LR_SVM

January 28, 2021

[ ]:

[1]:
```
import numpy as np
import pandas as pd
import plotly
import plotly.figure_factory as ff
import plotly.graph_objs as go
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot
init_notebook_mode(connected=True)
```

[2]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
```

[3]:
```
data = pd.read_csv('/content/drive/My Drive/AAIC/Assignments/10.Behavior of␣
 ↪Linear Models/practice/task_b.csv')
data=data.iloc[:,1:]
```

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

[4]:

	f1	f2	f3	y
0	-195.871045	-14843.084171	5.532140	1.0
1	-1217.183964	-4068.124621	4.416082	1.0
2	9.138451	4413.412028	0.425317	0.0
3	363.824242	15474.760647	1.094119	0.0
4	-768.812047	-7963.932192	1.870536	0.0

[13]:
```
data.corr()
```

[13]:

	f1	f2	f3	y
f1	1.000000	0.065468	0.123589	0.067172
f2	0.065468	1.000000	-0.055561	-0.017944
f3	0.123589	-0.055561	1.000000	0.839060
y	0.067172	-0.017944	0.839060	1.000000

[5]:
```
data.corr()['y']
```

[5]:
```
f1    0.067172
f2   -0.017944
f3    0.839060
y     1.000000
Name: y, dtype: float64
```

[19]:
```

```

[19]:
```

```

[12]:
```
data.describe()
```

[12]:

	f1	f2	f3	y
count	200.000000	200.000000	200.000000	200.000000
mean	10.180031	1299.986739	5.001840	0.500000
std	488.195035	10403.417325	2.926662	0.501255
min	-1662.579110	-29605.563847	0.076763	0.000000
25%	-303.220980	-5626.637315	2.508042	0.000000
50%	4.684317	2611.405803	5.029256	0.500000
75%	312.239850	8075.864754	7.436617	1.000000
max	1130.609573	24131.360720	9.933769	1.000000

[8]:
```
X=data[['f1','f2','f3']].values
Y=data['y'].values
print(X.shape)
print(Y.shape)
```

```
(200, 3)
(200,)
```

1 What if our features are with different variance

[9]:
```
from sklearn.linear_model import SGDClassifier
```

[10]:
```
from sklearn.linear_model import SGDClassifier
X=data[['f1','f2','f3']].values
Y=data['y'].values
lr = SGDClassifier(loss='log',penalty='l1').fit(X,Y)
svm = SGDClassifier(loss='hinge',penalty='l1').fit(X,Y)
print('Logistic-Regression',(lr.coef_),'\n')
print('SVM',(svm.coef_),'\n')
print('*'*70)
print('After col.standz\'n','\n')
X = StandardScaler().fit_transform(X)
lr1 = SGDClassifier(loss='log',penalty='l1').fit(X,Y)
svm1 = SGDClassifier(loss='hinge',penalty='l1').fit(X,Y)
print('Logistic-Regression',(lr1.coef_),'\n')
print('SVM',(svm1.coef_),'\n')
```

```
Logistic-Regression [[  831.05633127  2940.11073688 21279.92730916]]

SVM [[14320.95904934 -8466.37870461 26372.7955545 ]]

**********************************************************************
After col.standz'n

Logistic-Regression [[ 0.          0.87370007 13.3888022 ]]

SVM [[ 0.         -1.56989774 14.8007958 ]]
```

[11]:
```
from sklearn.linear_model import SGDClassifier
X=data[['f1','f2','f3']].values
Y=data['y'].values
lr = SGDClassifier(loss='log',penalty='l2').fit(X,Y)
svm = SGDClassifier(loss='hinge',penalty='l2').fit(X,Y)
print('Logistic-Regression',(lr.coef_),'\n')
print('SVM',(svm.coef_),'\n')
print('*'*70)
print('After col.standz\'n','\n')

X = StandardScaler().fit_transform(X)
lr1 = SGDClassifier(loss='log',penalty='l2').fit(X,Y)
svm1 = SGDClassifier(loss='hinge',penalty='l2').fit(X,Y)
print('Logistic-Regression',(lr1.coef_),'\n')
print('SVM',(svm1.coef_),'\n')
```

```
Logistic-Regression [[15090.66050727 -8127.46385949 11085.67179861]]

SVM [[  4495.7844023  -14900.61423946  10460.81707621]]

**********************************************************************
After col.standz'n

Logistic-Regression [[-0.69458829 -0.27203536 13.65444974]]

SVM [[-7.71014128 -4.17324229 18.42629538]]
```

[19]:

Task1:

1. The assumption in LogisticRegression is "Features are Independent".
2. we use pertubation technique to check whether features are collinear or not.Here the features are non-collinear.

3. we know that var(F2)>>var(F1)>>Var(F3).
4.

Make sure you write the observations for each task, why a particular feautre got more importance than others

OBS

The assumption in LogisticRegression is "Features are Independent".

We use the Perturbation Technique to check whether features are collinear or not. Here the features are non-collinear.

Even though the var(F2)>>var(F1)>>Var(F3), the model gives more weightage to the features whose correlation with the y-label is more.

Even after column standardization, these correlations don't change much

[ ]:

[ ]:

[ ]: