8B_LR_SVM_Assignment

June 12, 2020

[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]:
```
data = pd.read_csv('task_b.csv')
data=data.iloc[:,1:]
```

[3]:
```
data.head()
```

[3]:
```
            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
```

[4]:
```
data.corr()['y']
```

[4]:
```
f1    0.067172
f2   -0.017944
f3    0.839060
y     1.000000
Name: y, dtype: float64
```

[5]:
```
data.std()
```

[5]:
```
f1      488.195035
f2    10403.417325
f3        2.926662
y         0.501255
```

```
dtype: float64
```

[6]:

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

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

0.1 TASK 1

[7]:

```
from sklearn import linear_model
clf = linear_model.SGDClassifier(loss='log',tol=1e-3,alpha=0.0001,eta0=0.
↪0001,learning_rate='constant')
clf.fit(X,Y)
```

[7]:

```
SGDClassifier(alpha=0.0001, average=False, class_weight=None,
              early_stopping=False, epsilon=0.1, eta0=0.0001,
              fit_intercept=True, l1_ratio=0.15, learning_rate='constant',
              loss='log', max_iter=1000, n_iter_no_change=5, n_jobs=None,
              penalty='l2', power_t=0.5, random_state=None, shuffle=True,
              tol=0.001, validation_fraction=0.1, verbose=0, warm_start=False)
```

[8]:

```
clf.coef_
```

[8]:

```
array([[ 1.00791569, -0.67485403,  0.20901138]])
```

[9]:

```
from sklearn import linear_model
clf = linear_model.SGDClassifier(loss='hinge',tol=1e-3,alpha=0.0001,eta0=0.
↪0001,learning_rate='constant')
clf.fit(X,Y)
```

[9]:

```
SGDClassifier(alpha=0.0001, average=False, class_weight=None,
              early_stopping=False, epsilon=0.1, eta0=0.0001,
              fit_intercept=True, l1_ratio=0.15, learning_rate='constant',
              loss='hinge', max_iter=1000, n_iter_no_change=5, n_jobs=None,
              penalty='l2', power_t=0.5, random_state=None, shuffle=True,
              tol=0.001, validation_fraction=0.1, verbose=0, warm_start=False)
```

[10]:

```
clf.coef_
```

[10]:

```
array([[-0.03823947,  2.20709409,  0.14636524]])
```

0.2 TASK 2

[11]:

```
scaler = StandardScaler()
scaler.fit(X)
X=scaler.transform(X)
```

[12]:

```
from sklearn import linear_model
clf = linear_model.SGDClassifier(loss='log',tol=1e-3,alpha=0.0001,eta0=0.
↪0001,learning_rate='constant')
clf.fit(X,Y)
```

[12]:

```
SGDClassifier(alpha=0.0001, average=False, class_weight=None,
              early_stopping=False, epsilon=0.1, eta0=0.0001,
              fit_intercept=True, l1_ratio=0.15, learning_rate='constant',
              loss='log', max_iter=1000, n_iter_no_change=5, n_jobs=None,
              penalty='l2', power_t=0.5, random_state=None, shuffle=True,
              tol=0.001, validation_fraction=0.1, verbose=0, warm_start=False)
```

[13]:

```
clf.coef_
```

[13]:

```
array([[ 0.03848544, -0.00552488,  0.88963741]])
```

Observations:

In case of Logistic regression (SGDClassifier with logloss)

before standardization

1.f1 have high positive weight value than f3 postive weight value and f2 negative weight value.so f1 is more important than f3 and f2. 2.f2 have high negative weight value than f3 positive weight value.so f2 is also very important feature than f3.

after standardization

1.f3 have high positive weight value than f1.so f3 is more important than f1 after standardization. 2.negative weight value of f2 is less than f1 postive weight value.so, f3 and f1 have high feature importance than f2.

1.Large positive values of weight signify higher importance of that feature in predicting postive class 2.simillarly large negative values of weight signify higher importance of that feature in predicting negitive class

[14]:

```
from sklearn import linear_model
clf = linear_model.SGDClassifier(loss='hinge',tol=1e-3,alpha=0.0001,eta0=0.
↪0001,learning_rate='constant')
clf.fit(X,Y)
```

[14]:

```
SGDClassifier(alpha=0.0001, average=False, class_weight=None,
              early_stopping=False, epsilon=0.1, eta0=0.0001,
              fit_intercept=True, l1_ratio=0.15, learning_rate='constant',
              loss='hinge', max_iter=1000, n_iter_no_change=5, n_jobs=None,
              penalty='l2', power_t=0.5, random_state=None, shuffle=True,
              tol=0.001, validation_fraction=0.1, verbose=0, warm_start=False)
```

[15]:

```
clf.coef_
```

[15]:

```
array([[0.0424252 , 0.02589553, 1.07325136]])
```

Observations:

In case of SVM (SGDClassifier with hinge)

before standardization

1.f2 have high positive weight value than f3 postive weight value.so f2 is more important than f3. 2.f3 have high positive weight value than f1 negitive weight value.so f3 is very important feature than f1.

after standardization

1.f3 have high positive weight value than f1.so f3 is more important than f1 after standardization. 2.positive weight value of f2 is less than f1 postive weight value.so, f3 and f1 have high feature importance than f2.

[ ]:

```

```