hw-assignment-3-khan-inan

February 21, 2023

[3]:

```
from sklearn.linear_model import LogisticRegression

from sklearn.model_selection import train_test_split

from sklearn import metrics

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn import datasets

import pandas as pd
```

[43]:

```
iris = datasets.load_iris()
iris = datasets.load_iris()
iris_df=pd.DataFrame(iris.data)
iris_df.columns=['sepal_len', 'sepal_wid', 'petal_len', 'Species']
iris_df.dropna(how="all", inplace=True)
iris_X=iris_df.iloc[:,[0,1,2,3]]
print(iris_df)
```

```
     sepal_len  sepal_wid  petal_len  Species
0          5.1        3.5        1.4      0.2
1          4.9        3.0        1.4      0.2
2          4.7        3.2        1.3      0.2
3          4.6        3.1        1.5      0.2
4          5.0        3.6        1.4      0.2
..         …          …          …        …
145        6.7        3.0        5.2      2.3
146        6.3        2.5        5.0      1.9
147        6.5        3.0        5.2      2.0
148        6.2        3.4        5.4      2.3
149        5.9        3.0        5.1      1.8
```

```
[150 rows x 4 columns]
```

[29]:

```
from sklearn import preprocessing
from sklearn import utils
iris_cols = ['sepal_len', 'sepal_wid',]

X = iris_df[iris_cols]

y = iris_df.Species

lab = preprocessing.LabelEncoder()
y = lab.fit_transform(y)
```

[30]:

```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.
 ↪25,random_state=0)
logreg =  LogisticRegression(solver='liblinear')
logreg.fit(X_train,y_train)
y_pred=logreg.predict(X_test)
y_pred
```

[30]:

```
array([ 9,  9,  1, 19,  1,  1,  1,  9,  9,  9,  9,  9,  9,  9,  9,  1,  9,
        9,  1,  1,  9,  1,  1,  1,  9,  1,  1,  9,  9,  1,  9,  1,  1,  9,
        9,  1,  1,  9])
```

[33]:

```
from sklearn import preprocessing
from sklearn import utils
iris_cols = ['sepal_len', 'sepal_wid', 'petal_len']

X = iris_df[iris_cols]

y = iris_df.Species

lab = preprocessing.LabelEncoder()
y = lab.fit_transform(y)
```

[34]:

```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.
 ↪25,random_state=0)
logreg =  LogisticRegression(solver='liblinear')
logreg.fit(X_train,y_train)
y_pred=logreg.predict(X_test)
y_pred
```

[34]:

```
array([14,  9,  1, 19,  1, 14,  1,  9,  9,  9, 14,  9,  9,  9,  9,  1,  9,
        9,  1,  1, 14, 14,  1,  1,  9,  1,  1,  9,  9,  1, 14, 14,  1, 14,
       19,  9,  1, 19])
```

[45]:

```
from sklearn import preprocessing
from sklearn import utils
iris_cols = ['sepal_len', 'sepal_wid', 'petal_len', 'Species']

X = iris_df[iris_cols]

y = iris_df.Species

lab = preprocessing.LabelEncoder()
y = lab.fit_transform(y)
```

[46]:

```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.
 ↪25,random_state=0)
logreg =  LogisticRegression(solver='liblinear')
logreg.fit(X_train,y_train)
y_pred=logreg.predict(X_test)
y_pred
```

[46]:

```
array([19,  9,  1,  9,  1, 19,  1,  9,  9,  9, 14,  9,  9,  9,  9,  1,  9,
        9,  1,  1, 19, 14,  1,  1, 19,  1,  1,  9,  9,  1, 14, 14,  1, 19,
       19,  9,  1, 14])
```

Summarize your results (i.e, what's the best accuracy you can obtain for each of the 11 cases you considered, how many iterations does it take to converge, anything else you think is relevant and important) in a table.

The best accuracy in my opinion does not seem to differ in accordance with the number of features. This is most likely because a logistic regression model can improve with more data, but simply having more categories would not affect any machine learning model in a way that would be useful to us.

Discuss your findings. Does using more dimensions help when trying to classify the data in this dataset? How important is regularization in these cases?

Too many dimesions overfits the data in accordance to the training data. Regularisation is really important in this case because it allows us to fix errors due to generalization while at the same time preserving the machine learning and training