SVM

December 10, 2017

```
In [1]: import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import seaborn as sns
        %matplotlib inline
```

```
In [4]: letters = pd.read_csv('letter-recognition.csv', names= ['lettr','x-box','y-box','width',
```

```
In [5]: letters.head()
```

Out[5]:

	lettr	x-box	y-box	width	high	onpix	x-bar	y-bar	x2bar	y2bar	xybar	\
0	T	2	8	3	5	1	8	13	0	6	6	
1	I	5	12	3	7	2	10	5	5	4	13	
2	D	4	11	6	8	6	10	6	2	6	10	
3	N	7	11	6	6	3	5	9	4	6	4	
4	G	2	1	3	1	1	8	6	6	6	6	

	x2ybr	xy2br	x-ege	xegvy	y-ege	yegvx
0	10	8	0	8	0	8
1	3	9	2	8	4	10
2	3	7	3	7	3	9
3	4	10	6	10	2	8
4	5	9	1	7	5	10

```
In [6]: letters.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20000 entries, 0 to 19999
Data columns (total 17 columns):
lettr    20000 non-null object
x-box    20000 non-null int64
y-box    20000 non-null int64
width    20000 non-null int64
high     20000 non-null int64
onpix    20000 non-null int64
x-bar    20000 non-null int64
y-bar    20000 non-null int64
x2bar    20000 non-null int64
```

```
y2bar    20000 non-null int64
xybar    20000 non-null int64
x2ybr    20000 non-null int64
xy2br    20000 non-null int64
x-ege    20000 non-null int64
xegvy    20000 non-null int64
y-ege    20000 non-null int64
yegvx    20000 non-null int64
dtypes: int64(16), object(1)
memory usage: 2.6+ MB
```

```
In [7]: from sklearn.model_selection import train_test_split

In [8]: X= letters.drop('lettr', axis=1)

In [10]: y= letters['lettr']

In [16]: X_train, X_test, y_train, y_test = train_test_split( X, y, test_size=0.2, random_state=

In [17]: from sklearn.svm import SVC

In [26]: model= SVC(kernel='linear')

In [27]: model.fit(X_train, y_train)

Out[27]: SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
           decision_function_shape='ovr', degree=3, gamma='auto', kernel='linear',
           max_iter=-1, probability=False, random_state=None, shrinking=True,
           tol=0.001, verbose=False)

In [28]: predictions= model.predict(X_test)

In [29]: print(predictions)

['Z' 'L' 'A' ..., 'Q' 'Y' 'Y']

In [30]: from sklearn.metrics import classification_report, confusion_matrix

In [31]: print(classification_report(y_test, predictions))

             precision    recall  f1-score   support

          A       0.88      0.94      0.91       149
          B       0.79      0.87      0.83       153
          C       0.87      0.85      0.86       137
          D       0.78      0.90      0.84       156
          E       0.83      0.90      0.86       141
          F       0.80      0.91      0.85       140
```

```
          G       0.77      0.81      0.79       160
          H       0.63      0.59      0.61       144
          I       0.88      0.87      0.88       146
          J       0.83      0.87      0.85       149
          K       0.75      0.80      0.78       130
          L       0.92      0.87      0.90       155
          M       0.95      0.93      0.94       168
          N       0.93      0.89      0.91       151
          O       0.87      0.80      0.83       145
          P       0.95      0.83      0.88       173
          Q       0.85      0.78      0.81       166
          R       0.77      0.81      0.79       160
          S       0.75      0.73      0.74       171
          T       0.90      0.90      0.90       163
          U       0.95      0.90      0.92       183
          V       0.91      0.90      0.90       158
          W       0.90      0.95      0.93       148
          X       0.93      0.90      0.92       154
          Y       0.94      0.89      0.91       168
          Z       0.86      0.82      0.84       132

avg / total       0.86      0.85      0.85      4000
```

In [25]:
```
print(confusion_matrix(y_test, predictions))
```

```
[[149   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]
 [  0 150   0   1   0   0   1   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   1   0   0   0   0]
 [  0   0 131   0   1   0   1   1   0   0   0   0   0   0   3   0   0   0
    0   0   0   0   0   0   0   0]
 [  0   0   0 153   0   0   0   2   0   0   0   0   0   0   1   0   0   0
    0   0   0   0   0   0   0   0]
 [  0   0   0   0 139   0   0   0   0   0   0   1   0   0   0   0   0   0
    0   0   0   0   0   0   0   1]
 [  0   0   0   0   1 136   0   0   0   0   0   0   0   0   0   2   0   0
    0   0   0   0   0   1   0   0]
 [  0   0   1   2   1   0 154   0   0   0   0   0   0   0   0   0   0   0
    1   0   0   1   0   0   0   0]
 [  0   2   0   2   0   0   0 126   0   0   4   0   0   0   1   0   0   8
    0   0   1   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0 138   7   0   0   0   0   0   0   0   0
    0   0   0   0   0   1   0   0]
 [  0   0   0   0   0   1   0   0   1 147   0   0   0   0   0   0   0   0
    0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   1   0   0 122   0   0   0   0   0   0   5
```

```
   0   0   0   0   0   2   0   0]
 [  0   0   0   0   1   0   1   0   0   0   0 152   0   0   0   0   0   1
    0   0   0   0   0   0   0   0]
 [  0   1   0   0   0   0   0   0   0   0   0   0 167   0   0   0   0   0
    0   0   0   0   0   0   0   0]
 [  0   0   0   1   0   0   0   1   0   0   0   0   1 144   1   0   0   3
    0   0   0   0   0   0   0   0]
 [  0   0   0   3   0   0   0   0   0   0   0   0   0   0 140   0   1   0
    0   0   1   0   0   0   0   0]
 [  0   0   0   0   0   6   0   0   0   0   0   1   0   0   0 165   1   0
    0   0   0   0   0   0   0   0]
 [  0   0   0   1   0   0   0   0   0   0   0   0   0   0   1   1 163   0
    0   0   0   0   0   0   0   0]
 [  0   4   0   2   0   0   0   1   0   0   1   0   0   0   0   0   0 152
    0   0   0   0   0   0   0   0]
 [  0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0
  170   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
    0 163   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   1   0   0   0   0   1   0   0   0   0   0
    0   0 181   0   0   0   0   0]
 [  0   5   0   0   0   1   0   0   0   0   0   0   0   0   0   1   0   0
    0   0   0 150   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   0 148   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    1   0   0   0   0 153   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   0 168   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1
    0   0   0   0   0   0   0 131]]
```

```
In [32]: model= SVC()
```

```
In [33]: model.fit(X_train, y_train)
```

```
Out[33]: SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
           decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf',
           max_iter=-1, probability=False, random_state=None, shrinking=True,
           tol=0.001, verbose=False)
```

```
In [34]: predictions2 = model.predict(X_test)
```

```
In [35]: print(predictions2)
```

```
['T' 'L' 'A' ..., 'Q' 'Y' 'Y']
```

```
In [37]: print(classification_report(y_test, predictions2))
```

```
              precision    recall  f1-score   support

           A       1.00      1.00      1.00       149
           B       0.93      0.98      0.95       153
           C       0.99      0.96      0.97       137
           D       0.93      0.98      0.95       156
           E       0.97      0.99      0.98       141
           F       0.94      0.97      0.96       140
           G       0.98      0.96      0.97       160
           H       0.95      0.88      0.91       144
           I       0.99      0.95      0.97       146
           J       0.95      0.99      0.97       149
           K       0.96      0.94      0.95       130
           L       0.99      0.98      0.98       155
           M       0.99      0.99      0.99       168
           N       1.00      0.95      0.98       151
           O       0.95      0.97      0.96       145
           P       0.98      0.95      0.96       173
           Q       0.99      0.98      0.98       166
           R       0.89      0.95      0.92       160
           S       0.99      0.99      0.99       171
           T       1.00      1.00      1.00       163
           U       0.99      0.99      0.99       183
           V       0.99      0.95      0.97       158
           W       0.99      1.00      1.00       148
           X       0.97      0.99      0.98       154
           Y       1.00      1.00      1.00       168
           Z       0.99      0.99      0.99       132

avg / total       0.97      0.97      0.97      4000
```

In [38]:
```
print(confusion_matrix(y_test, predictions2))
```

```
[[149   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]
 [  0 150   0   1   0   0   1   0   0   0   0   0   0   0   0   0   0   0
    0   0   0   1   0   0   0   0]
 [  0   0 131   0   1   0   1   1   0   0   0   0   0   0   3   0   0   0
    0   0   0   0   0   0   0   0]
 [  0   0   0 153   0   0   0   2   0   0   0   0   0   0   1   0   0   0
    0   0   0   0   0   0   0   0]
 [  0   0   0   0 139   0   0   0   0   0   0   1   0   0   0   0   0   0
    0   0   0   0   0   0   0   1]
 [  0   0   0   0   1 136   0   0   0   0   0   0   0   0   0   2   0   0
    0   0   0   0   0   1   0   0]
 [  0   0   1   2   1   0 154   0   0   0   0   0   0   0   0   0   0   0
```

```
   1   0   0   1   0   0   0   0]
[  0   2   0   2   0   0   0 126   0   0   4   0   0   0   1   0   0   8
   0   0   1   0   0   0   0   0]
[  0   0   0   0   0   0   0   0 138   7   0   0   0   0   0   0   0   0
   0   0   0   0   0   1   0   0]
[  0   0   0   0   0   1   0   0   1 147   0   0   0   0   0   0   0   0
   0   0   0   0   0   0   0   0]
[  0   0   0   0   0   0   0   1   0   0 122   0   0   0   0   0   0   5
   0   0   0   0   0   2   0   0]
[  0   0   0   0   1   0   1   0   0   0   0 152   0   0   0   0   0   1
   0   0   0   0   0   0   0   0]
[  0   1   0   0   0   0   0   0   0   0   0   0 167   0   0   0   0   0
   0   0   0   0   0   0   0   0]
[  0   0   0   1   0   0   0   1   0   0   0   0   1 144   1   0   0   3
   0   0   0   0   0   0   0   0]
[  0   0   0   3   0   0   0   0   0   0   0   0   0   0 140   0   1   0
   0   0   1   0   0   0   0   0]
[  0   0   0   0   0   6   0   0   0   0   0   1   0   0   0 165   1   0
   0   0   0   0   0   0   0   0]
[  0   0   0   1   0   0   0   0   0   0   0   0   0   0   1   1 163   0
   0   0   0   0   0   0   0   0]
[  0   4   0   2   0   0   0   1   0   0   1   0   0   0   0   0   0 152
   0   0   0   0   0   0   0   0]
[  0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0
 170   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
   0 163   0   0   0   0   0   0]
[  0   0   0   0   0   0   0   1   0   0   0   0   1   0   0   0   0   0
   0   0 181   0   0   0   0   0]
[  0   5   0   0   0   1   0   0   0   0   0   0   0   0   0   0   1   0   0
   0   0   0 150   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   0 148   0   0   0]
[  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
   1   0   0   0   0 153   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   0 168   0]
[  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1
   0   0   0   0   0   0   0 131]]
```