zfmlhw05

1. 阅读作业

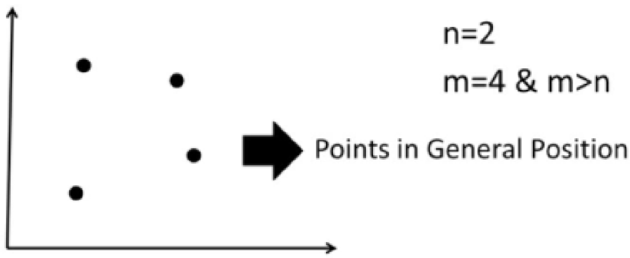
在本次课程中，你学习了VC维这个概念。“对于非线性分类器，VC维非常难于计算，在学术研究领域，这仍然是一个有待回答的开放性问题。但对于线性分类器，VC维是可以计算的。” 请你阅读下面博文中 http://blog.csdn.net/baimafujinji/article/details/44856089 (http://blog.csdn.net/baimafujinji/article/details/44856089) 的第二部分“VC维”，以了解对于一个线性分类器，我们该如何计算其VC维，特别注意“Points in General Position和Shatter”这两个概念。提交一张你阅读的界面截图。

二、VC维

对于一个给定的分类器或者Hypothsis，还如何确定VC维呢？一个不好的消息是，对于非线性分类器，VC维非常难于计算，在学术研究领域，这仍然是一个有待回答的开放性问题。一个好消息是，对于线性分类器，VC维是可以计算的，所以下面我们主要讨论线性分类器的VC维。但在此之前，我们还需要先了解两个关键概念，即Points in General Position和Shatter。

在一个n维特征空间中，一个包含m个点的集合（m>n）is in general position 当且仅当没有包含n+1个点的子集落在n-1维的超平面上。

来看两个具体的例子，如下图所示，在一个二维空间中，有4个点，显然其中不存在包含3个点的子集都落在一个1维的超平面上的情况，所以说这些点都是in general position。



另外，你还需要阅读如下博文中的关于NFL原理的部分（注意：你只需要阅读文章中的第一部分暨NFL原理的部分）

http://blog.csdn.net/baimafujinji/article/details/6475824 (http://blog.csdn.net/baimafujinji/article/details/6475824)

原 NFL原理与Hoeffding不等式

2009年09月23日 13:07:00 白马负金羁 阅读数：2415 标签： 没有免费午餐定理 Hoeffding不等式 更多

一、没有免费午餐（NFL，No Free Lunch）原理

天下没有免费的午餐。这句常理在数学中有一个定理专门描述它，这个定理就叫做没有免费午餐定理。这个定理的严格证明可参见文献【2】和【3】。如果你有兴趣读了这两篇经典文献中的任何一篇就会知道，最原始的没有免费午餐定理是在最优化理论中出现的（而后又推广到例如机器学习这样的领域）。为了解释清楚这个定理，先来说说什么是最优化。

最优化，其实就是寻找函数最大（或最小）值的办法。我们在微积分中已经接触过类似的数学知识，彼时我们更关注函数是否存在最值。但搞其他应用学科的人往往要知道这个最值在哪里，多大以及如果计算。例如在机器学习中常常采样迭代法计算极值。当然，计算机的计算能力始终是有限的，要尽可能的花费更少的时间和资源来算出最值，这也成为最优化算法尤其要解决的一个问题。

提交一张你阅读的界面截图。

同时，请你思考一下，**NFL**原理和我们讲的**VC**维有什么相通的地方或者有什么联系，用两三句话简单总结一下你的认识或者理解。

NFL表述的意思是没有放之四海皆准的方法。如果一个算法对某些问题非常好，那么一定存在另一些问题，对于这些问题，该算法比随机猜测还要差。
VC维表述的就是用来衡量一个算法适用于一类数据的最大边界。

你不需要回答，但请你仔细思考一下文章最后给出的结论：d维空间中的线性分类器之VC维等于d+1，如果你自己能够理解清楚这背后的道理，说明你对VC维的理解已经足够深入了！

2. 编程实践题（*20%）

在之前的作业中我们已经给出了countries_data数据，现在请你利用此数据建立最大间隔分类器（也就是SVM模型）来对两类国家进行分类。具体要求如下：
1）使用MATLAB, Python或者R。
2）通过代码读入一个csv文件的方式来导入数据。
3）评估你的分类器（使用Accuracy、Precision、Recall和F1-Score）。
4）用图形化的方式展示你的分类结果。

In [14]:

```
import pandas as pd
from sklearn import svm
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report
import matplotlib.pyplot as plt
```

In [2]:

```
df = pd.read_csv('countries_data.csv',encoding='gbk')
df.head()
```

Out[2]:

	countries	Services_of_GDP	ages65_of_total	label
0	Belgium	76.7	18	1
1	France	78.9	18	1
2	Denmark	76.2	18	1
3	Spain	73.9	18	1
4	Japan	72.6	25	1

In [8]:

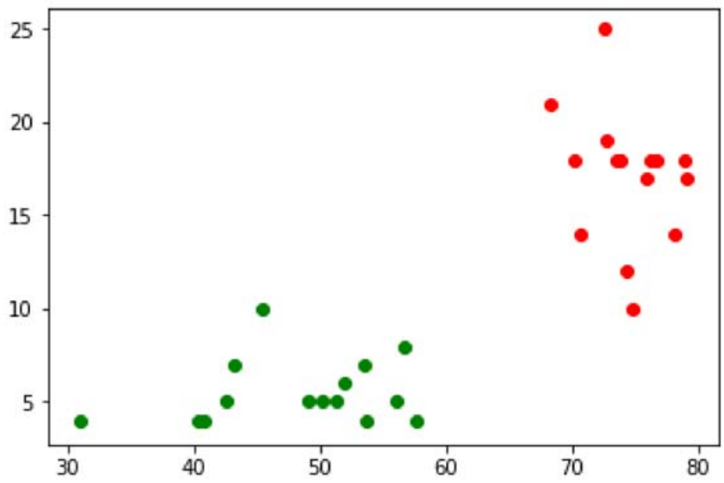
```
X = df.iloc[:,1:3]
Y = df.iloc[:,3]

#X.as_matrix()
#FutureWarning: Method .as_matrix will be removed in a future version. Use .values
X = X.values
Y = Y.values
```

In [18]:

```
plt.scatter([X[:,0][i] for i in range(len(Y)) if Y[i]==1],[X[:,1][i] for i in range
plt.scatter([X[:,0][i] for i in range(len(Y)) if Y[i]==0],[X[:,1][i] for i in range
```

Out[18]:

<matplotlib.collections.PathCollection at 0x7fa9bb8f7978>



In [9]:

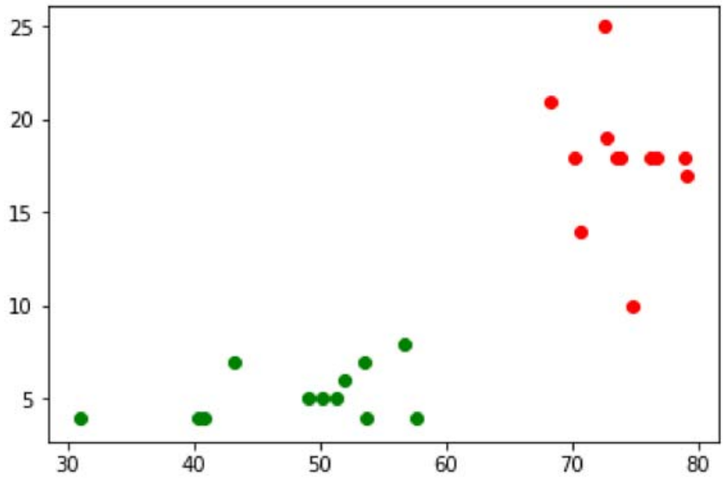
```
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2, random_sta
```

In [30]:

```
or i in range(len(y_train)) if y_train[i]==1],[X_train[:,1][i] for i in range(len(y_
or i in range(len(y_train)) if y_train[i]==0],[X_train[:,1][i] for i in range(len(y_
```

Out[30]:

```
<matplotlib.collections.PathCollection at 0x7fa9bb610a58>
```

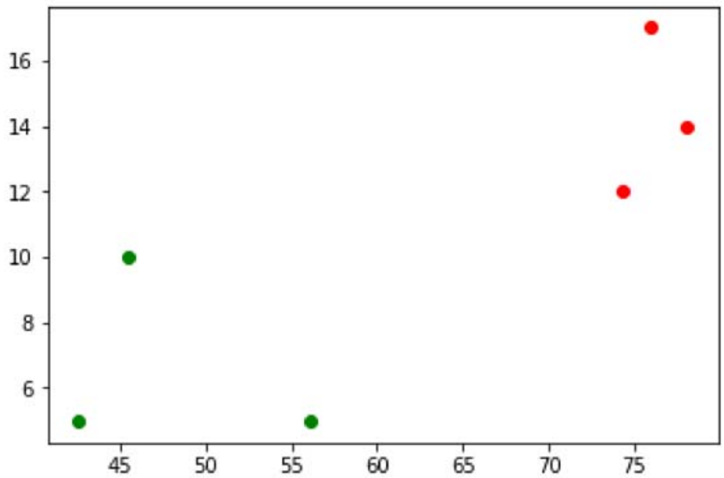


In [31]:

```
plt.scatter([X_test[:,0][i] for i in range(len(y_test)) if y_test[i]==1],[X_test[:,
plt.scatter([X_test[:,0][i] for i in range(len(y_test)) if y_test[i]==0],[X_test[:,
```

Out[31]:

```
<matplotlib.collections.PathCollection at 0x7fa9bb5e6940>
```



In [24]:

```
model = svm.SVC()
```

In [25]:

```
model.fit(X_train, y_train)
```

Out[25]:

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

In [26]:

```
prediction = model.predict(X_test)
prediction
```

Out[26]:

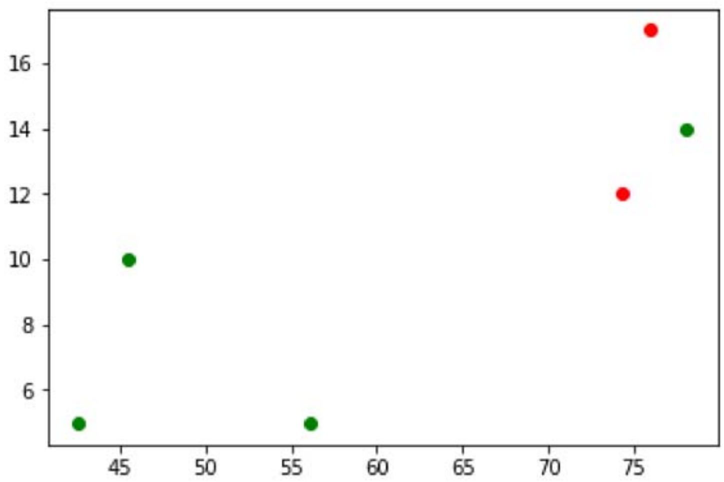
```
array([0, 0, 0, 0, 1, 1])
```

In [32]:

```
plt.scatter([X_test[:,0][i] for i in range(len(prediction)) if prediction[i]==1],[X
plt.scatter([X_test[:,0][i] for i in range(len(prediction)) if prediction[i]==0],[X
```

Out[32]:

```
<matplotlib.collections.PathCollection at 0x7fa9bb5491d0>
```



In [27]:

```
print("accuracy score: ")
print(accuracy_score(y_test, prediction))
print(classification_report(y_test, prediction))
```

```
accuracy score:
0.8333333333333334
             precision    recall  f1-score   support

          0       0.75      1.00      0.86         3
          1       1.00      0.67      0.80         3

avg / total       0.88      0.83      0.83         6
```

3. 数学推导题 （*35%）

推导PPT中第24页最下方的等式：

$$\mathcal{L}(\boldsymbol{w},b,\boldsymbol{\alpha})=\frac{1}{2}\|\boldsymbol{w}\|^2-\sum_{i=1}^{k}\alpha_i[y_i(\boldsymbol{w}^T\boldsymbol{x}_i+b)-1]=\sum_{i=1}^{k}\alpha_i-\frac{1}{2}\sum_{i,j=1}^{k}\alpha_i\alpha_j y_i y_j (x_i)^T x_j$$

$$\vec{w}=\sum_{i=1}^{k}\alpha_i y_i \vec{x}_i$$

$$\sum_{i=1}^{k}\alpha_i y_i=0$$

$$L(\vec{w},b,\vec{\alpha})=\frac{1}{2}\|\vec{w}\|^2-\sum_{i=1}^{k}\alpha_i[y_i(\vec{w}^T\vec{x}_i+b)-1]$$

$$=\frac{1}{2}\vec{w}^T\cdot\vec{w}-\sum_{i=1}^{k}\alpha_i y_i\left(\sum_{j=1}^{k}\alpha_j y_j(\vec{x}_j)\right)^T\vec{x}_i$$

$$-b\sum_{i=1}^{k}\alpha_i y_i+\sum_{i=1}^{k}\alpha_i$$

$$=\frac{1}{2}\sum_{i,j=1}^{k}\alpha_i\alpha_j y_i y_j(\vec{x}_i)^T\vec{x}_j-\sum_{i,j=1}^{k}\alpha_i\alpha_j y_i y_j(\vec{x}_i)^T\vec{x}_j+\sum_{i=1}^{k}\alpha_i$$

$$=\sum_{i=1}^{k}\alpha_i-\frac{1}{2}\sum_{i,j=1}^{k}\alpha_i\alpha_j y_i y_j(\vec{x}_i)^T\vec{x}_j$$

4. 证明题（*35%）

对于带等式约束的优化问题，我们可以使用“拉格朗日乘数法”。拉格朗日乘数法在机器学习（甚至图像处理）中都有较多应用，例如SVM中的凸优化、回归分析中的正则化、以及最大熵模型的推导。
为了强化你对拉格朗日乘数法的理解，最后这个问题可以帮助你亲身体验一下它的应用。请你运用拉格朗日乘数法来证明几何-算术均值不等式。注意：这个不等式的证明方法很多，本题的意思是要求你仅仅使用拉格朗日乘数法来证明之，如果你采用其它方法，则会被判定为“答非所问”。 几何均值不等式：

for any list of n nonnegative real numbers $x_1, x_2, \ldots, x_n$, we have

$$\frac{x_1 + x_2 + \cdots + x_n}{n} \geq \sqrt[n]{x_1 \cdot x_2 \cdots x_n},$$

and that equality holds if and only if $x_1 = x_2 = \cdots = x_n$.

end

end

end

end

end

end

end

end

end

end

end

end