1. 阅读作业（*10%）

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

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。

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

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。

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！

学习是自己的事情，请认真理解其中内容，切勿只截图而不阅读。

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：

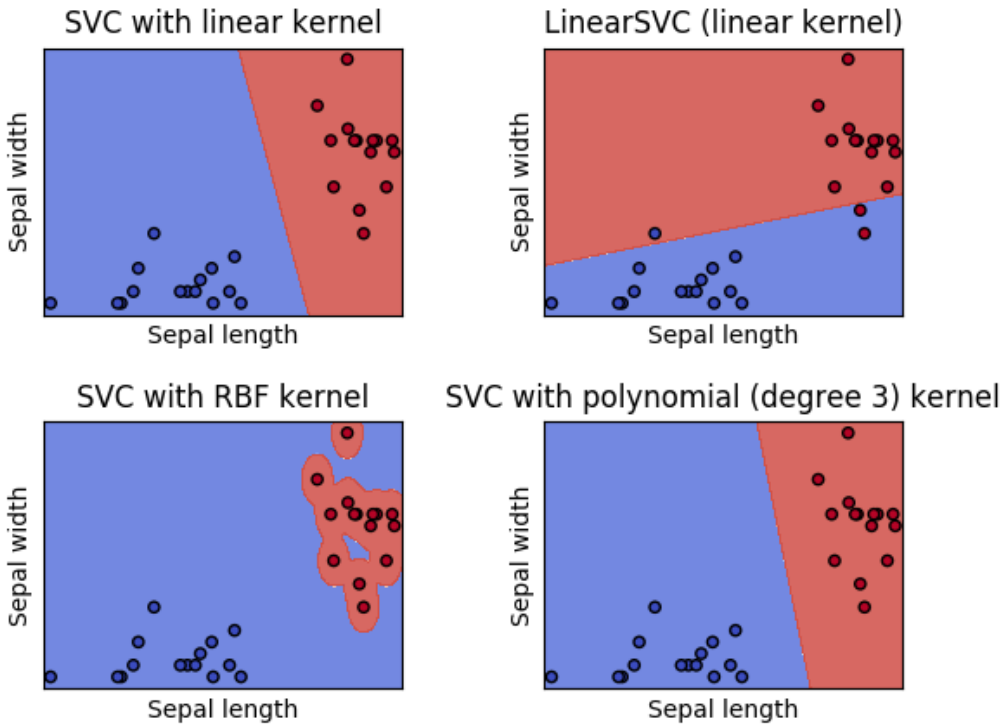
1）使用 MATLAB, Python 或者 R。

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。

```
(env-cpu) testuser@testuser-System-Product-Name:~/Documents/jupyter/dataguru/ml_zuofei$ python lecture_05_hm_scikit-learn.py
None
             precision    recall  f1-score   support

          0       1.00      1.00      1.00        15
          1       1.00      1.00      1.00        15

avg / total       1.00      1.00      1.00        30

[[15  0]
 [ 0 15]]
             precision    recall  f1-score   support

          0       0.88      0.93      0.90        15
          1       0.93      0.87      0.90        15

avg / total       0.90      0.90      0.90        30

[[14  1]
 [ 2 13]]
             precision    recall  f1-score   support

          0       1.00      1.00      1.00        15
          1       1.00      1.00      1.00        15

avg / total       1.00      1.00      1.00        30

[[15  0]
 [ 0 15]]
             precision    recall  f1-score   support

          0       1.00      1.00      1.00        15
          1       1.00      1.00      1.00        15

avg / total       1.00      1.00      1.00        30

[[15  0]
 [ 0 15]]
```



代码

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

```
from sklearn import metrics, svm

def extract_data(filename):
    df = pd.read_csv(filename)
    df_fvecs = df[['Services_of_GDP', 'ages65_of_total']]
    fvecs = df_fvecs.as_matrix()
    labels = df[['label']].as_matrix()
    labels = labels.reshape(1, labels.size).flatten()
    return fvecs,labels


def make_meshgrid(x, y, h=.02):
    x_min, x_max = x.min() - 1, x.max() + 1
    y_min, y_max = y.min() - 1, y.max() + 1
    xx, yy = np.meshgrid(np.arange(x_min, x_max, h),
                         np.arange(y_min, y_max, h))
    return xx, yy


def plot_contours(ax, clf, xx, yy, **params):
    Z = clf.predict(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)
    out = ax.contourf(xx, yy, Z, **params)
    return out


X,y = extract_data('countries_data.csv')

C = 1.0
models = (svm.SVC(kernel='linear', C=C),
          svm.LinearSVC(C=C),
          svm.SVC(kernel='rbf', gamma=0.7, C=C),
          svm.SVC(kernel='poly', degree=3, C=C))

for clf in models:
  clf.fit(X, y)
  # metrics
  predicted = clf.predict(X)
  print(metrics.classification_report(y, predicted))
  print(metrics.confusion_matrix(y, predicted))

titles = ('SVC with linear kernel',
          'LinearSVC (linear kernel)',
          'SVC with RBF kernel',
```

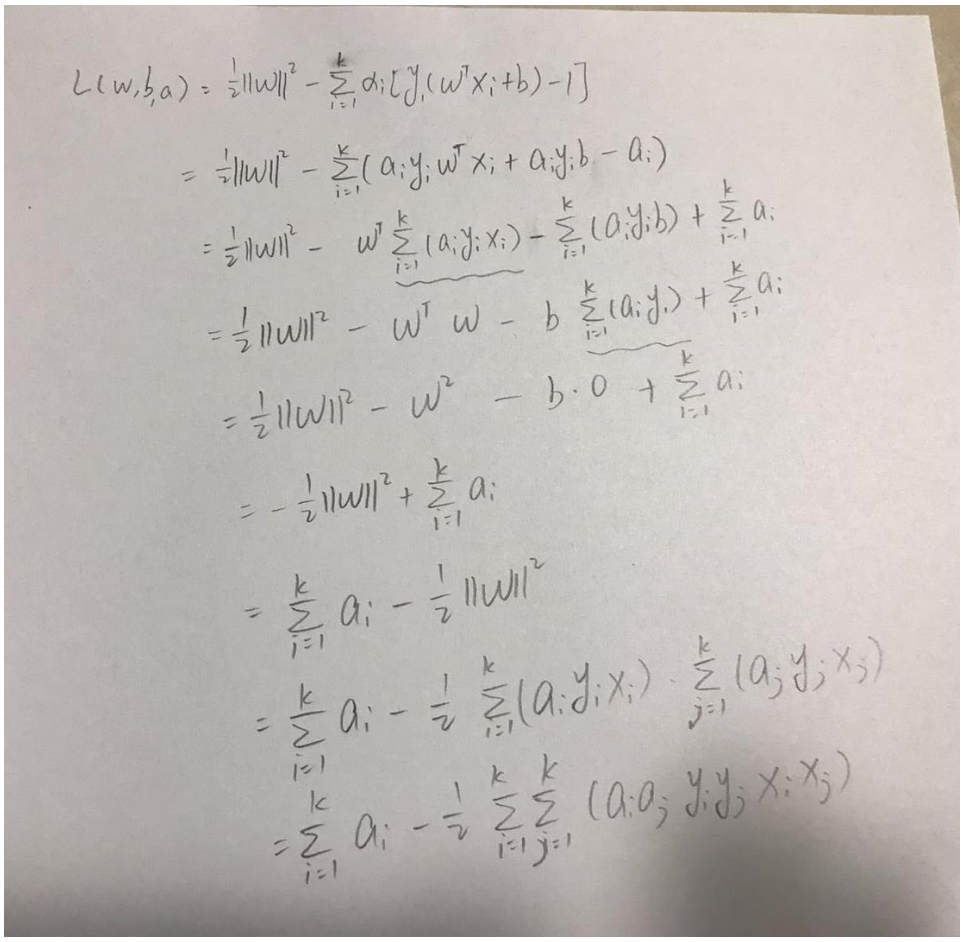
```
        'SVC with polynomial (degree 3) kernel')

fig, sub = plt.subplots(2, 2)
plt.subplots_adjust(wspace=0.4, hspace=0.4)

X0, X1 = X[:, 0], X[:, 1]
xx, yy = make_meshgrid(X0, X1)

for clf, title, ax in zip(models, titles, sub.flatten()):
    plot_contours(ax, clf, xx, yy, cmap=plt.cm.coolwarm, alpha=0.8)
    ax.scatter(X0, X1, c=y, cmap=plt.cm.coolwarm, s=20,
edgecolors='k')
    ax.set_xlim(xx.min(), xx.max())
    ax.set_ylim(yy.min(), yy.max())
    ax.set_xlabel('Sepal length')
    ax.set_ylabel('Sepal width')
    ax.set_xticks(())
    ax.set_yticks(())
    ax.set_title(title)

plt.show()
```

3. 数学推导题 (*35%)

推导 PPT 中第 24 页最下方的等式：

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

提交完整的证明过程。答案会随下次课程资料一同公布。



$$L(w,b,a) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{k} a_i[y_i(w^T x_i + b) - 1]$$

$$= \frac{1}{2}\|w\|^2 - \sum_{i=1}^{k}(a_i y_i w^T x_i + a_i y_i b - a_i)$$

$$= \frac{1}{2}\|w\|^2 - w^T \sum_{i=1}^{k}(a_i y_i x_i) - \sum_{i=1}^{k}(a_i y_i b) + \sum_{i=1}^{k} a_i$$

$$= \frac{1}{2}\|w\|^2 - w^T w - b \sum_{i=1}^{k}(a_i y_i) + \sum_{i=1}^{k} a_i$$

$$= \frac{1}{2}\|w\|^2 - w^2 - b \cdot 0 + \sum_{i=1}^{k} a_i$$

$$= -\frac{1}{2}\|w\|^2 + \sum_{i=1}^{k} a_i$$

$$= \sum_{i=1}^{k} a_i - \frac{1}{2}\|w\|^2$$

$$= \sum_{i=1}^{k} a_i - \frac{1}{2}\sum_{i=1}^{k}(a_i y_i x_i) \cdot \sum_{j=1}^{k}(a_j y_j x_j)$$

$$= \sum_{i=1}^{k} a_i - \frac{1}{2}\sum_{i=1}^{k}\sum_{j=1}^{k}(a_i a_j y_i y_j x_i x_j)$$

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：

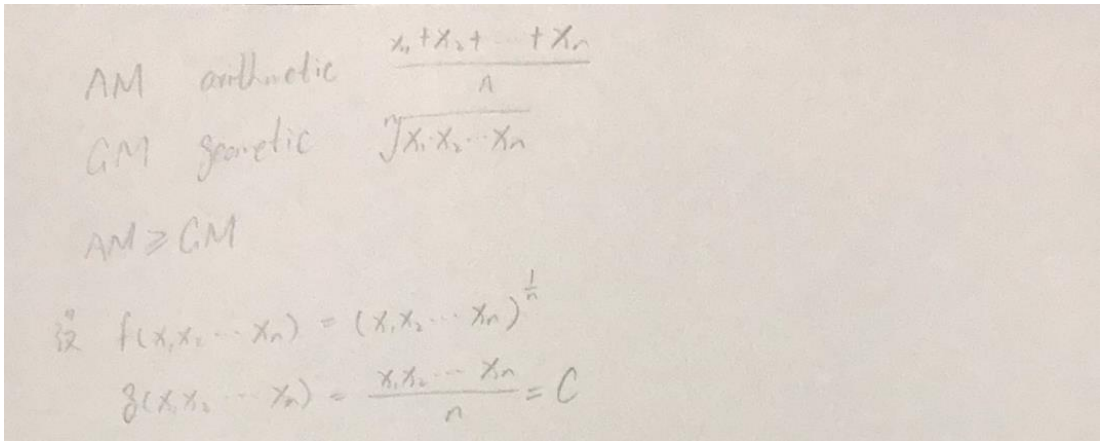
for any list of n nonnegative real numbers $x_1, x_2, \ldots, x_n$, we have

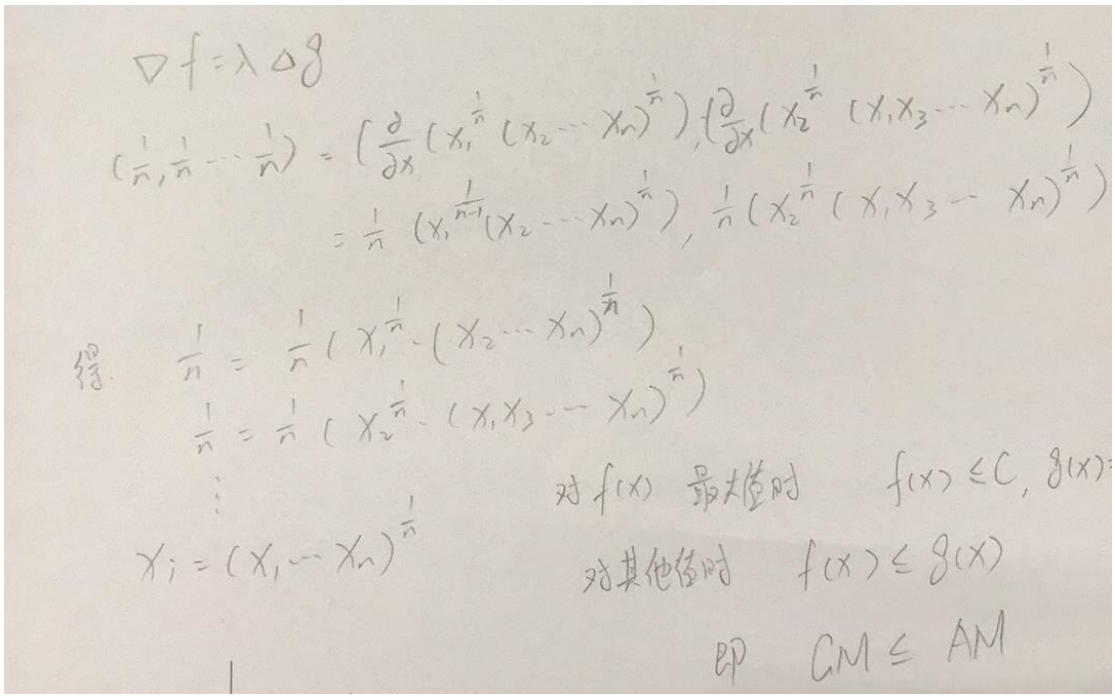
$$\frac{x_1 + x_2 + \cdots + x_n}{n} \geq \sqrt[n]{x_1 \cdot x_2 \cdots x_n},$$

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

关于这个不等式的更多内容，你还可以参考（如果你有兴趣的话）：

https://en.wikipedia.org/wiki/Inequality_of_arithmetic_and_geometric_means





---

* 难度系数及所占比例