In [1]:

```
import os,sys,time
import numpy as np
from sklearn.metrics import accuracy_score
from sklearn.metrics import f1_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import f1_score
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import mean_squared_error

from sklearn import datasets
from sklearn.model_selection import train_test_split
```

In [2]:

```
# KNN
from sklearn.neighbors import KNeighborsClassifier

# let's try to use pandas' fast csv reader
try:
    import pandas
    readcsv = lambda f, c, t=np.float64: pandas.read_csv(f, usecols=c, delimiter=',', header=None,
except:
    # fall back to numpy loadtxt
    readcsv = lambda f, c, t=np.float64: np.loadtxt(f, usecols=c, delimiter=',', ndmin=2)

start = time.clock() # 开始计时

# Input data set parameters
train_file = os.path.join('data', 'k_nearest_neighbors_train.csv')
predict_file  = os.path.join('data', 'k_nearest_neighbors_test.csv')

# Read data. Let's use 5 features per observation
nFeatures = 5
nClasses = 5
train_data   = readcsv(train_file, range(nFeatures))
train_labels = readcsv(train_file, range(nFeatures, nFeatures+1))
predict_data = readcsv(predict_file, range(nFeatures))
predict_labels = readcsv(predict_file, range(nFeatures, nFeatures+1))

knn = KNeighborsClassifier()#得到分类器

knn.fit(train_data, train_labels)#训练模型

predictedLabel = knn.predict(predict_data) # 进行预测
end=time.clock() #结束计时
print("time", end-start)

print("accuracy_score", accuracy_score(predict_labels, predictedLabel))
print("precision_score", precision_score(predict_labels, predictedLabel, average="macro"))
print("recall_score", recall_score(predict_labels, predictedLabel, average="micro"))
print("f1_score", f1_score(predict_labels, predictedLabel, average="micro"))
print("confusion_matrix\n", confusion_matrix(predict_labels, predictedLabel))
"""
micro算法是指把所有的类放在一起算，具体到precision，就是把所有类的TP加和，再除以所有类的TP和FN的加和
因此micro方法下的precision和recall都等于accuracy
"""
```

```
f:\pearl\anaconda3\envs\daal\lib\site-packages\ipykernel_launcher.py:12: Depreca
tionWarning: time.clock has been deprecated in Python 3.3 and will be removed fr
om Python 3.8: use time.perf_counter or time.process_time instead
  if sys.path[0] == '':
f:\pearl\anaconda3\envs\daal\lib\site-packages\ipykernel_launcher.py:28: DataCon
versionWarning: A column-vector y was passed when a 1d array was expected. Pleas
e change the shape of y to (n_samples, ), for example using ravel().
```

```
time 0.5116148000000003
accuracy_score 0.9655
```

In [3]:

```
# 支持向量机
from sklearn.svm import SVC

# let's try to use pandas' fast csv reader
try:
    import pandas
    readcsv = lambda f, c, t=np.float64: pandas.read_csv(f, usecols=c, delimiter=',', header=None,
except:
    # fall back to numpy loadtxt
    readcsv = lambda f, c, t=np.float64: np.loadtxt(f, usecols=c, delimiter=',', ndmin=2)

start = time.clock() # 开始计时
# input data file
infile = "./data/svm_two_class_train_dense.csv"
testfile = "./data/svm_two_class_test_dense.csv"

data   = readcsv(infile, range(20))
labels = readcsv(infile, range(20,21))
pdata = readcsv(testfile, range(20))
plabels = readcsv(testfile, range(20,21))

clf = SVC()
clf.fit(data, labels)
predictedLabel = clf.predict(pdata)

end=time.clock() #结束计时
print("time", end-start)

print("accuracy_score", accuracy_score(plabels, predictedLabel))
print("precision_score", precision_score(plabels, predictedLabel, average="macro"))
print("recall_score", recall_score(plabels, predictedLabel, average="macro"))
print("f1_score", f1_score(plabels, predictedLabel, average="macro"))
print("confusion_matrix\n", confusion_matrix(plabels, predictedLabel))
```

```
f:\pearl\anaconda3\envs\daal\lib\site-packages\ipykernel_launcher.py:12: Deprecatio
nWarning: time.clock has been deprecated in Python 3.3 and will be removed from Pyt
hon 3.8: use time.perf_counter or time.process_time instead
  if sys.path[0] == '':
f:\pearl\anaconda3\envs\daal\lib\site-packages\sklearn\utils\validation.py:73: Data
ConversionWarning: A column-vector y was passed when a 1d array was expected. Pleas
e change the shape of y to (n_samples, ), for example using ravel().
  return f(**kwargs)

time 0.19032150000000314
accuracy_score 0.986
precision_score 0.9864077669902913
recall_score 0.9859719438877755
f1_score 0.985996415082261
confusion_matrix
 [[1002    0]
 [  28  970]]

f:\pearl\anaconda3\envs\daal\lib\site-packages\ipykernel_launcher.py:26: Deprecatio
nWarning: time.clock has been deprecated in Python 3.3 and will be removed from Pyt
hon 3.8: use time.perf_counter or time.process_time instead
```

In [4]:

```
# 朴素贝叶斯
from sklearn.naive_bayes import GaussianNB

# let's try to use pandas' fast csv reader
try:
    import pandas
    read_csv = lambda f, c, t=np.float64: pandas.read_csv(f, usecols=c, delimiter=',', header=None
except:
    # fall back to numpy loadtxt
    read_csv = lambda f, c, t=np.float64: np.loadtxt(f, usecols=c, delimiter=',', ndmin=2)

start = time.clock() # 开始计时

# input data file
infile = "./data/naivebayes_train_dense.csv"
testfile = "./data/naivebayes_test_dense.csv"
data   = readcsv(infile, range(20))
labels = readcsv(infile, range(20,21))
pdata = readcsv(testfile, range(20))
plabels = readcsv(testfile, range(20,21))

gnb = GaussianNB()
predictedLabel = gnb.fit(data, labels).predict(pdata)

end=time.clock() #结束计时
print("time", end-start)

print("accuracy_score", accuracy_score(plabels, predictedLabel))
print("precision_score", precision_score(plabels, predictedLabel, average="macro"))
print("recall_score", recall_score(plabels, predictedLabel, average="macro"))
print("f1_score", f1_score(plabels, predictedLabel, average="macro"))
print("confusion_matrix\n", confusion_matrix(plabels, predictedLabel))
```

```
f:\pearl\anaconda3\envs\daal\lib\site-packages\ipykernel_launcher.py:12: Deprecatio
nWarning: time.clock has been deprecated in Python 3.3 and will be removed from Pyt
hon 3.8: use time.perf_counter or time.process_time instead
  if sys.path[0] == '':

time 0.3533451999999997
accuracy_score 1.0
precision_score 1.0
recall_score 1.0
f1_score 1.0
confusion_matrix
 [[ 83   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0]
 [  0  90   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0]
 [  0   0 104   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0]
 [  0   0   0  91   0   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0]
 [  0   0   0   0  95   0   0   0   0   0   0   0   0   0   0   0   0   0
    0   0]
 [  0   0   0   0   0  93   0   0   0   0   0   0   0   0   0   0   0   0
    0   0]
 [  0   0   0   0   0   0  97   0   0   0   0   0   0   0   0   0   0   0
```

```
[  0   0   0   0   0   0  97   0   0   0   0   0   0   0   0   0   0
   0   0]
 [  0   0   0   0   0   0   0 106   0   0   0   0   0   0   0   0   0   0
   0   0]
 [  0   0   0   0   0   0   0   0  92   0   0   0   0   0   0   0   0   0
   0   0]
 [  0   0   0   0   0   0   0   0   0 105   0   0   0   0   0   0   0   0
   0   0]
 [  0   0   0   0   0   0   0   0   0   0 107   0   0   0   0   0   0   0
   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0  98   0   0   0   0   0   0
   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0 115   0   0   0   0   0
   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0 103   0   0   0   0
   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0 109   0   0   0
   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0 109   0   0
   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0 106   0
   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  97
   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 101   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
   0  99]]
```

```
f:\pearl\anaconda3\envs\daal\lib\site-packages\sklearn\utils\validation.py:73: Data
ConversionWarning: A column-vector y was passed when a 1d array was expected. Pleas
e change the shape of y to (n_samples, ), for example using ravel().
  return f(**kwargs)
f:\pearl\anaconda3\envs\daal\lib\site-packages\ipykernel_launcher.py:25: Deprecatio
nWarning: time.clock has been deprecated in Python 3.3 and will be removed from Pyt
hon 3.8: use time.perf_counter or time.process_time instead
```

In [5]:

```
# adaboost
from sklearn.ensemble import AdaBoostClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.datasets import make_gaussian_quantiles

# let's try to use pandas' fast csv reader
try:
    import pandas
    read_csv = lambda f, c, t=np.float64: pandas.read_csv(f, usecols=c, delimiter=',', header=None
except:
    # fall back to numpy loadtxt
    read_csv = lambda f, c, t=np.float64: np.loadtxt(f, usecols=c, delimiter=',', ndmin=2)

start = time.clock() # 开始计时

infile = "./data/adaboost_train.csv"
testfile = "./data/adaboost_test.csv"

# Read data. Let's have 20 independent, and 1 dependent variable (for each observation)
indep_data = readcsv(infile, range(20))
dep_data   = readcsv(infile, range(20,21))
pdata = readcsv(testfile, range(20))
predict_labels = np.loadtxt(testfile, usecols=range(20,21), delimiter=',', ndmin=2)

bdt = AdaBoostClassifier(DecisionTreeClassifier(max_depth=2, min_samples_split=20, min_samples_leaf=
                         algorithm="SAMME",
                         n_estimators=200, learning_rate=0.8)
bdt.fit(indep_data, dep_data)
predictedLabel = bdt.predict(pdata)

end=time.clock() #结束计时
print("time", end-start)

print("accuracy_score", accuracy_score(predict_labels, predictedLabel))
print("precision_score", precision_score(predict_labels, predictedLabel))
print("recall_score", recall_score(predict_labels, predictedLabel))
print("f1_score", f1_score(predict_labels, predictedLabel))
print("confusion_matrix\n", confusion_matrix(predict_labels, predictedLabel))
```

```
f:\pearl\anaconda3\envs\daal\lib\site-packages\ipykernel_launcher.py:14: Deprecatio
nWarning: time.clock has been deprecated in Python 3.3 and will be removed from Pyt
hon 3.8: use time.perf_counter or time.process_time instead


time 0.2419916999999998
accuracy_score 1.0
precision_score 1.0
recall_score 1.0
f1_score 1.0
confusion_matrix
 [[1599    0]
 [   0  401]]

f:\pearl\anaconda3\envs\daal\lib\site-packages\sklearn\utils\validation.py:73: Data
ConversionWarning: A column-vector y was passed when a 1d array was expected. Pleas
e change the shape of y to (n_samples, ), for example using ravel().
```

```
    return f(**kwargs)
```

```
f:\pearl\anaconda3\envs\daal\lib\site-packages\ipykernel_launcher.py:31: Deprecatio
nWarning: time.clock has been deprecated in Python 3.3 and will be removed from Pyt
hon 3.8: use time.perf_counter or time.process_time instead
```

In [6]:

```
# 随机森林
from sklearn.ensemble import RandomForestClassifier
from sklearn.datasets import make_classification
# let's try to use pandas' fast csv reader
try:
    import pandas
    read_csv = lambda f, c, t=np.float64: pandas.read_csv(f, usecols=c, delimiter=',', header=None
except:
    # fall back to numpy loadtxt
    read_csv = lambda f, c, t=np.float64: np.loadtxt(f, usecols=c, delimiter=',', ndmin=2, dtype=t

# Get Intel(R) Data Analytics Acceleration Library (Intel(R) DAAL) version

start = time.clock() # 开始计时

# input data file
infile = "./data/df_classification_train.csv"
testfile = "./data/df_classification_test.csv"
# Read data. Let's use 3 features per observation
data   = readcsv(infile, range(3), t=np.float32)
labels = readcsv(infile, range(3,4), t=np.float32)
pdata = readcsv(testfile, range(3), t=np.float32)
plabels = readcsv(testfile, range(3,4), t=np.float32)

clf = RandomForestClassifier(max_depth=3, random_state=0)
clf.fit(data, labels)
predictedLabel = clf.predict(pdata)

end=time.clock() #结束计时
print("time", end-start)

print("accuracy_score", accuracy_score(plabels, predictedLabel))
print("precision_score", precision_score(plabels, predictedLabel, average="macro"))
print("recall_score", recall_score(plabels, predictedLabel, average="macro"))
print("f1_score", f1_score(plabels, predictedLabel, average="macro"))
print("confusion_matrix\n", confusion_matrix(plabels, predictedLabel))
```

```
f:\pearl\anaconda3\envs\daal\lib\site-packages\ipykernel_launcher.py:14: Deprecatio
nWarning: time.clock has been deprecated in Python 3.3 and will be removed from Pyt
hon 3.8: use time.perf_counter or time.process_time instead

f:\pearl\anaconda3\envs\daal\lib\site-packages\ipykernel_launcher.py:26: DataConver
sionWarning: A column-vector y was passed when a 1d array was expected. Please chan
ge the shape of y to (n_samples,), for example using ravel().

time 4.456374499999999
accuracy_score 0.363
precision_score 0.17200956937799045
recall_score 0.24420454545454545
f1_score 0.16618381618381614
confusion_matrix
 [[  0   0 163   0   0]
 [  0   0 160   0   0]
 [  0   0 318   2   0]
 [  0   0 153  45   0]
 [  0   0 118  41   0]]
```

```
f:\pearl\anaconda3\envs\daal\lib\site-packages\ipykernel_launcher.py:29: Deprecatio
nWarning: time.clock has been deprecated in Python 3.3 and will be removed from Pyt
hon 3.8: use time.perf_counter or time.process_time instead
f:\pearl\anaconda3\envs\daal\lib\site-packages\sklearn\metrics\_classification.py:1
221: UndefinedMetricWarning: Precision is ill-defined and being set to 0.0 in label
s with no predicted samples. Use `zero_division` parameter to control this behavio
r.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [7]:

```
# EM算法
from sklearn.mixture import GaussianMixture
start = time.clock() # 开始计时

#鸢尾花数据集
X, y = datasets.load_iris(return_X_y=True)
#X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y, test_size=0.7, random_state=4

gmm = GaussianMixture(n_components=3)

predictedLabel = gmm.fit_predict(X)
end=time.clock() #结束计时
print("time", end-start)

print("accuracy_score", accuracy_score(y, predictedLabel))
print("precision_score", precision_score(y, predictedLabel, average="macro"))
print("recall_score", recall_score(y, predictedLabel, average="macro"))
print("f1_score", f1_score(y, predictedLabel, average="macro"))
print("confusion_matrix\n", confusion_matrix(y, predictedLabel))
```

```
time 0.02340820000000221
accuracy_score 0.0
precision_score 0.0
recall_score 0.0
f1_score 0.0
confusion_matrix
 [[ 0 50  0]
 [ 5  0 45]
 [50  0  0]]

f:\pearl\anaconda3\envs\daal\lib\site-packages\ipykernel_launcher.py:3: Deprecation
Warning: time.clock has been deprecated in Python 3.3 and will be removed from Pyth
on 3.8: use time.perf_counter or time.process_time instead
  This is separate from the ipykernel package so we can avoid doing imports until
f:\pearl\anaconda3\envs\daal\lib\site-packages\ipykernel_launcher.py:12: Deprecatio
nWarning: time.clock has been deprecated in Python 3.3 and will be removed from Pyt
hon 3.8: use time.perf_counter or time.process_time instead
  if sys.path[0] == '':
```

In [8]:

```
#随机森林 回归

from sklearn.ensemble import RandomForestRegressor
from sklearn.datasets import make_regression

# let's try to use pandas' fast csv reader
try:
    import pandas
    read_csv = lambda f, c, t=np.float64: pandas.read_csv(f, usecols=c, delimiter=',', header=None
except:
    # fall back to numpy loadtxt
    read_csv = lambda f, c, t=np.float64: np.loadtxt(f, usecols=c, delimiter=',', ndmin=2, dtype=n


start = time.clock() # 开始计时
infile = "./data/df_regression_train.csv"
testfile = "./data/df_regression_test.csv"
# Read data. Let's have 13 independent, and 1 dependent variables (for each observation)
indep_data = readcsv(infile, range(13), t=np.float32)
dep_data   = readcsv(infile, range(13,14), t=np.float32)
pdata = readcsv(testfile, range(13), t=np.float32)
ptdata = readcsv(testfile, range(13,14), t=np.float32)

regr = RandomForestRegressor(n_estimators=100, random_state=0)
regr.fit(indep_data, dep_data)
predict_result = regr.predict(pdata)

end=time.clock() #结束计时
print("time", end-start)

print("MAE", mean_absolute_error(ptdata, predict_result))
print("MSE", mean_squared_error(ptdata, predict_result))
```

```
f:\pearl\anaconda3\envs\daal\lib\site-packages\ipykernel_launcher.py:15: Deprecatio
nWarning: time.clock has been deprecated in Python 3.3 and will be removed from Pyt
hon 3.8: use time.perf_counter or time.process_time instead
  from ipykernel import kernelapp as app
f:\pearl\anaconda3\envs\daal\lib\site-packages\ipykernel_launcher.py:25: DataConver
sionWarning: A column-vector y was passed when a 1d array was expected. Please chan
ge the shape of y to (n_samples,), for example using ravel().

time 0.2740158000000008
MAE 2.511535433070865
MSE 12.542506133858259

f:\pearl\anaconda3\envs\daal\lib\site-packages\ipykernel_launcher.py:28: Deprecatio
nWarning: time.clock has been deprecated in Python 3.3 and will be removed from Pyt
hon 3.8: use time.perf_counter or time.process_time instead
```

In [ ]:

```

```

In [ ]:

```

```