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
except:
    # fall back to numpy loadtxt
    readcsv = lambda f, c, t=np.float64: np.loadtxt(f, usecols=c, delimiter=',',

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
print("recall_score", recall_score(predict_labels, predictedLabel, average="micr
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
因此micro方法下的precision和recall都等于accuracy
"""
```

```
/opt/anaconda3/envs/daal/lib/python3.7/site-packages/ipykernel_launch
er.py:12: DeprecationWarning: time.clock has been deprecated in Pytho
n 3.3 and will be removed from Python 3.8: use time.perf_counter or t
ime.process_time instead
  if sys.path[0] == '':
/opt/anaconda3/envs/daal/lib/python3.7/site-packages/ipykernel_launch
er.py:28: DataConversionWarning: A column-vector y was passed when a
1d array was expected. Please change the shape of y to (n_samples, ),
for example using ravel().
```

```
time 0.35871600000000003
accuracy_score 0.9655
precision_score 0.9650319748408597
```

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
except:
    # fall back to numpy loadtxt
    readcsv = lambda f, c, t=np.float64: np.loadtxt(f, usecols=c, delimiter=',',

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))


```

```
time 0.13607600000000009
accuracy_score 0.986
precision_score 0.9864077669902913
recall_score 0.9859719438877755
f1_score 0.985996415082261
confusion_matrix
 [[1002    0]
 [  28  970]]

/opt/anaconda3/envs/daal/lib/python3.7/site-packages/ipykernel_launche
r.py:12: DeprecationWarning: time.clock has been deprecated in Python
3.3 and will be removed from Python 3.8: use time.perf_counter or tim
e.process_time instead
  if sys.path[0] == '':
/opt/anaconda3/envs/daal/lib/python3.7/site-packages/sklearn/utils/val
idation.py:73: DataConversionWarning: A column-vector y was passed whe
n a 1d array was expected. Please change the shape of y to (n_samples,
), for example using ravel().
  return f(**kwargs)
/opt/anaconda3/envs/daal/lib/python3.7/site-packages/ipykernel_launche
r.py:26: DeprecationWarning: time.clock has been deprecated in Python
3.3 and will be removed from Python 3.8: use time.perf_counter or tim
e.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, delimite
except:
    # fall back to numpy loadtxt
    read_csv = lambda f, c, t=np.float64: np.loadtxt(f, usecols=c, delimiter=','

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))


```

```
/opt/anaconda3/envs/daal/lib/python3.7/site-packages/ipykernel_launche
r.py:12: DeprecationWarning: time.clock has been deprecated in Python
3.3 and will be removed from Python 3.8: use time.perf_counter or tim
e.process_time instead
  if sys.path[0] == '':
/opt/anaconda3/envs/daal/lib/python3.7/site-packages/sklearn/utils/val
idation.py:73: DataConversionWarning: A column-vector y was passed whe
n a 1d array was expected. Please change the shape of y to (n_samples,
), for example using ravel().
  return f(**kwargs)

time 0.21331999999999995
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   0 104   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  95    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  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  92    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    0 107    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    0 115    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    0 109    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
97
     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   99]]
```

```
/opt/anaconda3/envs/daal/lib/python3.7/site-packages/ipykernel_launche
r.py:25: DeprecationWarning: time.clock has been deprecated in Python
3.3 and will be removed from Python 3.8: use time.perf_counter or tim
e.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, delimite
except:
    # fall back to numpy loadtxt
    read_csv = lambda f, c, t=np.float64: np.loadtxt(f, usecols=c, delimiter=','

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 obser
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

bdt = AdaBoostClassifier(DecisionTreeClassifier(max_depth=2, min_samples_split=2
                         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))


```

```
/opt/anaconda3/envs/daal/lib/python3.7/site-packages/ipykernel_launche
r.py:14: DeprecationWarning: time.clock has been deprecated in Python
3.3 and will be removed from Python 3.8: use time.perf_counter or tim
e.process_time instead


time 0.22340399999999994
accuracy_score 1.0
precision_score 1.0
recall_score 1.0
f1_score 1.0
confusion_matrix
 [[1599    0]
 [   0  401]]


/opt/anaconda3/envs/daal/lib/python3.7/site-packages/sklearn/utils/val
```

```
idation.py:73: DataConversionWarning: A column-vector y was passed whe
n a 1d array was expected. Please change the shape of y to (n_samples,
), for example using ravel().
  return f(**kwargs)
/opt/anaconda3/envs/daal/lib/python3.7/site-packages/ipykernel_launche
r.py:31: DeprecationWarning: time.clock has been deprecated in Python
3.3 and will be removed from Python 3.8: use time.perf_counter or tim
e.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, delimite
except:
    # fall back to numpy loadtxt
    read_csv = lambda f, c, t=np.float64: np.loadtxt(f, usecols=c, delimiter=','

# 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))


```

```
/opt/anaconda3/envs/daal/lib/python3.7/site-packages/ipykernel_launche
r.py:14: DeprecationWarning: time.clock has been deprecated in Python
3.3 and will be removed from Python 3.8: use time.perf_counter or tim
e.process_time instead

/opt/anaconda3/envs/daal/lib/python3.7/site-packages/ipykernel_launche
r.py:26: DataConversionWarning: A column-vector y was passed when a 1d
array was expected. Please change the shape of y to (n_samples,), for
example using ravel().

time 3.9982549999999994
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]]

/opt/anaconda3/envs/daal/lib/python3.7/site-packages/ipykernel_launche
r.py:29: DeprecationWarning: time.clock has been deprecated in Python
3.3 and will be removed from Python 3.8: use time.perf_counter or tim
e.process_time instead
/opt/anaconda3/envs/daal/lib/python3.7/site-packages/sklearn/metrics/_
classification.py:1221: UndefinedMetricWarning: Precision is ill-defin
ed and being set to 0.0 in labels with no predicted samples. Use `zero
_division` parameter to control this behavior.
  _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

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))


```

```
/opt/anaconda3/envs/daal/lib/python3.7/site-packages/ipykernel_launche
r.py:3: DeprecationWarning: time.clock has been deprecated in Python
3.3 and will be removed from Python 3.8: use time.perf_counter or tim
e.process_time instead
  This is separate from the ipykernel package so we can avoid doing im
ports until

time 0.020326999999999984
accuracy_score 0.3333333333333333
precision_score 0.30303030303030304
recall_score 0.3333333333333333
f1_score 0.31746031746031744
confusion_matrix
 [[ 0 50  0]
 [45  0  5]
 [ 0  0 50]]

/opt/anaconda3/envs/daal/lib/python3.7/site-packages/ipykernel_launche
r.py:12: DeprecationWarning: time.clock has been deprecated in Python
3.3 and will be removed from Python 3.8: use time.perf_counter or tim
e.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, delimite
except:
    # fall back to numpy loadtxt
    read_csv = lambda f, c, t=np.float64: np.loadtxt(f, usecols=c, delimiter=','


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 obse
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))
```

```
/opt/anaconda3/envs/daal/lib/python3.7/site-packages/ipykernel_launche
r.py:15: DeprecationWarning: time.clock has been deprecated in Python
3.3 and will be removed from Python 3.8: use time.perf_counter or tim
e.process_time instead
  from ipykernel import kernelapp as app
/opt/anaconda3/envs/daal/lib/python3.7/site-packages/ipykernel_launche
r.py:25: DataConversionWarning: A column-vector y was passed when a 1d
array was expected. Please change the shape of y to (n_samples,), for
example using ravel().

time 0.28624399999999994
MAE 2.511535433070865
MSE 12.542506133858259

/opt/anaconda3/envs/daal/lib/python3.7/site-packages/ipykernel_launche
r.py:28: DeprecationWarning: time.clock has been deprecated in Python
3.3 and will be removed from Python 3.8: use time.perf_counter or tim
e.process_time instead
```

In [ ]:

```

```

In [ ]:

```
1
```