In [1]:
```
from matplotlib.colors import ListedColormap
from sklearn import model_selection, datasets, metrics, neighbors

%matplotlib inline

import numpy as np
```

In [3]:
```
breast_cancer = datasets.load_breast_cancer()
```

In [4]:
```
digits = datasets.load_digits()
```

In [9]:
```
digits.data[:3]
```

Out[9]:
```
array([[  0.,   0.,   5.,  13.,   9.,   1.,   0.,   0.,   0.,   0.,  13.,
         15.,  10.,  15.,   5.,   0.,   0.,   3.,  15.,   2.,   0.,  11.,
          8.,   0.,   0.,   4.,  12.,   0.,   0.,   8.,   8.,   0.,   0.,
          5.,   8.,   0.,   0.,   9.,   8.,   0.,   0.,   4.,  11.,   0.,
          1.,  12.,   7.,   0.,   0.,   2.,  14.,   5.,  10.,  12.,   0.,
          0.,   0.,   0.,   6.,  13.,  10.,   0.,   0.,   0.],
       [  0.,   0.,   0.,  12.,  13.,   5.,   0.,   0.,   0.,   0.,   0.,
         11.,  16.,   9.,   0.,   0.,   0.,   0.,   3.,  15.,  16.,   6.,
          0.,   0.,   0.,   7.,  15.,  16.,  16.,   2.,   0.,   0.,   0.,
          0.,   1.,  16.,  16.,   3.,   0.,   0.,   0.,   0.,   1.,  16.,
         16.,   6.,   0.,   0.,   0.,   0.,   1.,  16.,  16.,   6.,   0.,
          0.,   0.,   0.,   0.,  11.,  16.,  10.,   0.,   0.],
       [  0.,   0.,   0.,   4.,  15.,  12.,   0.,   0.,   0.,   0.,   3.,
         16.,  15.,  14.,   0.,   0.,   0.,   0.,   8.,  13.,   8.,  16.,
          0.,   0.,   0.,   0.,   1.,   6.,  15.,  11.,   0.,   0.,   0.,
          1.,   8.,  13.,  15.,   1.,   0.,   0.,   0.,   9.,  16.,  16.,
          5.,   0.,   0.,   0.,   0.,   3.,  13.,  16.,  16.,  11.,   5.,
          0.,   0.,   0.,   0.,   3.,  11.,  16.,   9.,   0.]])
```

In [11]:
```
breast_cancer.data[:3]
```

```
Out[11]: array([[  1.79900000e+01,   1.03800000e+01,   1.22800000e+02,
                   1.00100000e+03,   1.18400000e-01,   2.77600000e-01,
                   3.00100000e-01,   1.47100000e-01,   2.41900000e-01,
                   7.87100000e-02,   1.09500000e+00,   9.05300000e-01,
                   8.58900000e+00,   1.53400000e+02,   6.39900000e-03,
                   4.90400000e-02,   5.37300000e-02,   1.58700000e-02,
                   3.00300000e-02,   6.19300000e-03,   2.53800000e+01,
                   1.73300000e+01,   1.84600000e+02,   2.01900000e+03,
                   1.62200000e-01,   6.65600000e-01,   7.11900000e-01,
                   2.65400000e-01,   4.60100000e-01,   1.18900000e-01],
                [  2.05700000e+01,   1.77700000e+01,   1.32900000e+02,
                   1.32600000e+03,   8.47400000e-02,   7.86400000e-02,
                   8.69000000e-02,   7.01700000e-02,   1.81200000e-01,
                   5.66700000e-02,   5.43500000e-01,   7.33900000e-01,
                   3.39800000e+00,   7.40800000e+01,   5.22500000e-03,
                   1.30800000e-02,   1.86000000e-02,   1.34000000e-02,
                   1.38900000e-02,   3.53200000e-03,   2.49900000e+01,
                   2.34100000e+01,   1.58800000e+02,   1.95600000e+03,
                   1.23800000e-01,   1.86600000e-01,   2.41600000e-01,
                   1.86000000e-01,   2.75000000e-01,   8.90200000e-02],
                [  1.96900000e+01,   2.12500000e+01,   1.30000000e+02,
                   1.20300000e+03,   1.09600000e-01,   1.59900000e-01,
                   1.97400000e-01,   1.27900000e-01,   2.06900000e-01,
                   5.99900000e-02,   7.45600000e-01,   7.86900000e-01,
                   4.58500000e+00,   9.40300000e+01,   6.15000000e-03,
                   4.00600000e-02,   3.83200000e-02,   2.05800000e-02,
                   2.25000000e-02,   4.57100000e-03,   2.35700000e+01,
                   2.55300000e+01,   1.52500000e+02,   1.70900000e+03,
                   1.44400000e-01,   4.24500000e-01,   4.50400000e-01,
                   2.43000000e-01,   3.61300000e-01,   8.75800000e-02]])
```

In [12]:
```
from sklearn.naive_bayes import BernoulliNB
clf_1 = BernoulliNB()
from sklearn.naive_bayes import MultinomialNB
clf_2 = MultinomialNB()
from sklearn.naive_bayes import GaussianNB
clf_3 = GaussianNB()
```

In [15]:
```
cross_val_score_digits = []
cross_val_score_cancer = []
```

In [16]:
```
for clf in [clf_1, clf_2, clf_3]:
    cross_val_score_digits.append(model_selection.cross_val_score(clf, digits
.data, digits.target, cv=5).mean())
    cross_val_score_cancer.append(model_selection.cross_val_score(clf, breast
_cancer.data, breast_cancer.target, cv=5).mean())
```

In [17]:
```
cross_val_score_digits
```

Out[17]:
```
[0.82477104598047846, 0.87147030254753344, 0.80652075555522984]
```

In [18]:
```
cross_val_score_cancer
```

Out[18]:
```
[0.62742593305117356, 0.89637552904963447, 0.9403770681031165]
```

Наивные байесовские классификаторы справляются в целом неплохо. Ясно, что Бернулли справляется хуже, так как он считает признаки бинарными. Бинаризовывать мы умеем категориальные признаки, а количественные - нет, так как их неограниченно много. Из-за этого понятно, почему Бернулли справился лучше с интовыми значениями: возможно, взял минимум, максимум, увидел, что их конечное число и бинаризовал. С флотовыми значениями такое не проходит, и этим объясняется всего score в 62 процента

1. Каким получилось максимальное качество классификации на датасете breast_cancer?

0.94 с помощью GaussianNB

2. Каким получилось максимальное качество классификации на датасете digits?

0.87 с помощью MultinomialNB

Какие утверждения верны?

c, d

In [ ]: