Доверительные интервалы для оценки среднего

In [1]:
```
from sklearn import cross_validation, datasets, linear_model, metrics

import numpy as np
```

In [2]:
```
%pylab inline
```

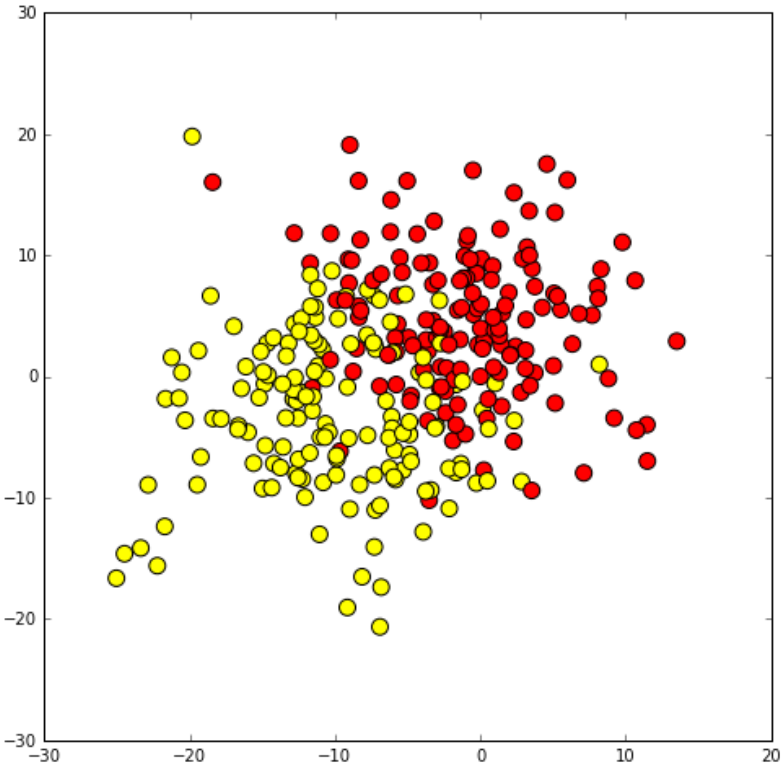
```
Populating the interactive namespace from numpy and matplotlib
```

Генерация данных

In [3]:
```
blobs = datasets.make_blobs(300, centers = 2, cluster_std = 6, random_state=1
```

In [4]:
```
pylab.figure(figsize(8, 8))
pylab.scatter(map(lambda x: x[0], blobs[0]), map(lambda x: x[1], blobs[0]), c
              s=100)
```

Out[4]:
```
<matplotlib.collections.PathCollection at 0x968b240>
```



Сравнение линейных моделей

Точечная оценка

In [5]:
```
train_data, test_data, train_labels, test_labels = cross_validation.train_te
```

In [6]:
```
ridge_model = linear_model.RidgeClassifier()
ridge_model.fit(train_data, train_labels)
metrics.roc_auc_score(test_labels, ridge_model.predict(test_data))
```

Out[6]: 0.88888888888888884

In [7]:
```
sgd_model = linear_model.SGDClassifier(random_state = 0)
sgd_model.fit(train_data, train_labels)
metrics.roc_auc_score(test_labels, sgd_model.predict(test_data))
```

Out[7]: 0.88888888888888884

Оценка среднего

In [8]:
```
sgd_auc_scores = cross_validation.cross_val_score(linear_model.SGDClassifier
                                    blobs[0], blobs[1], scoring = 'roc_auc',
                                    cv = 20)
```

In [9]:
```
ridge_auc_scores = cross_validation.cross_val_score(linear_model.RidgeClassi
                                    blobs[0], blobs[1], scoring = 'roc_auc',
                                    cv = 20)
```

Точечная оценка среднего

In [10]:
```
print "sgd model auc: mean %.3f, std %.3f" % (sgd_auc_scores.mean(), sgd_auc_
print "ridge model auc: mean %.3f, std %.3f" % (ridge_auc_scores.mean(), ridg
```

```
sgd model auc: mean 0.904, std 0.117
ridge model auc: mean 0.948, std 0.054
```

Интервальная оценка среднего

In [11]:
```
from statsmodels.stats.weightstats import _zconfint_generic, _tconfint_gener
```

In [12]:
```
sgd_mean = sgd_auc_scores.mean()
ridge_mean = ridge_auc_scores.mean()
```

z-интервал

Допустим, нам откуда-то известно, что дисперсия auc_scores $\sigma^2 = 0.25$. Построим доверительные интервалы для средних вида

$$\bar{X}_n \pm z_{1-\frac{\alpha}{2}} \frac{\sigma}{\sqrt{n}}$$

In [13]:
```
print "sgd model mean auc 95%% confidence interval", _zconfint_generic(sgd_me
                                                                sqrt(0.25/
                                                                0.05, 'two

print "ridge model mean auc 95%% confidence interval", _zconfint_generic(ridg
                                                                  sqrt(0.2
                                                                  0.05, 'tw
```

```
sgd model mean auc 95%% confidence interval (0.68512638016197691, 1.12338
76504502678)
ridge model mean auc 95%% confidence interval (0.72886043628442598, 1.167
1217065727166)
```

t-интервал

Вместо гипотетической теоретической дисперсии σ^2, которую мы на самом деле в данном случае не знаем, используем выборочные дисперсии, и построим доверительные интервалы вида

$$\bar{X}_n \pm t_{1-\frac{\alpha}{2}} \frac{S}{\sqrt{n}}$$

In [14]:
```
type(sgd_auc_scores)
```

Out[14]:
```
numpy.ndarray
```

In [15]:
```
sgd_mean_std = sgd_auc_scores.std(ddof=1)/sqrt(len(sgd_auc_scores))
ridge_mean_std = ridge_auc_scores.std(ddof=1)/sqrt(len(ridge_auc_scores))
```

In [16]:
```
print "sgd model mean auc 95%% confidence interval", _tconfint_generic(sgd_me
                                                                       len(sg
                                                                       0.05,

print "ridge model mean auc 95%% confidence interval", _tconfint_generic(ridg
                                                                         len
                                                                         0.0
```

```
sgd model mean auc 95%% confidence interval (0.84939983366532035, 0.95911
419694692435)
ridge model mean auc 95%% confidence interval (0.92287832593023689, 0.973
10381692690573)
```