05.04.2018

Вычислительные модели с использованием научных библиотек Python

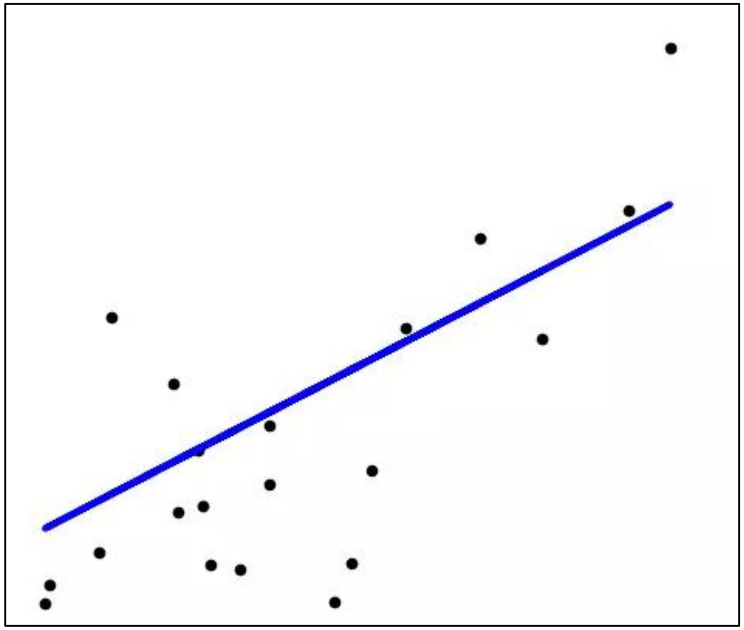
Алгоритмы регрессии

Линейная регрессия

$$\mathrm{RSS}(\beta) = \sum_{i=1}^{N}(y_i - f(x_i))^2$$
$$= \sum_{i=1}^{N}\Big(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j\Big)^2$$

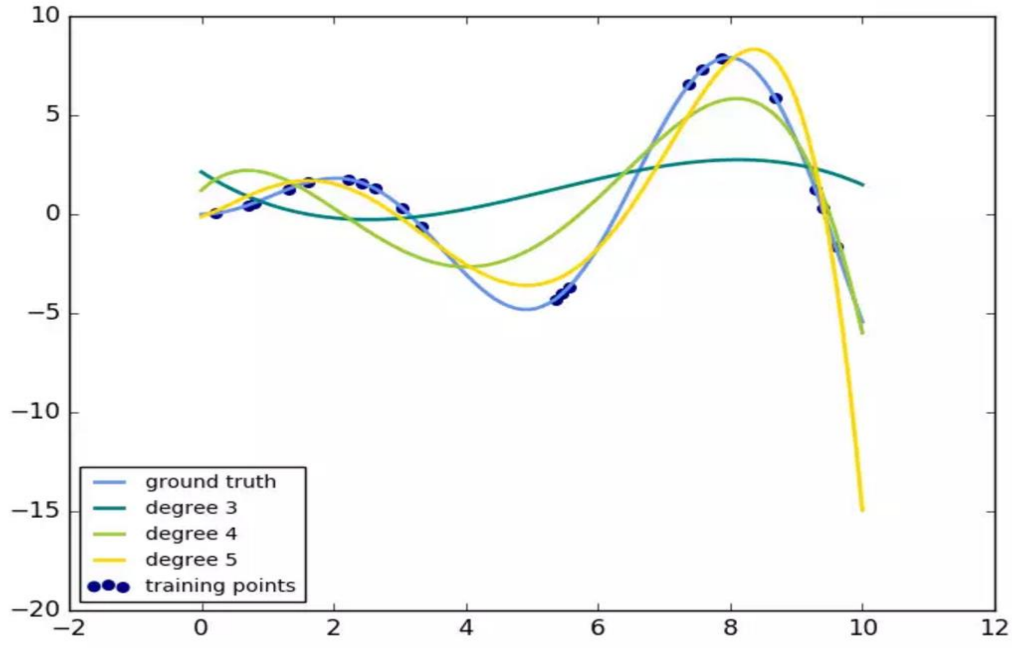
$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

```
>>> from sklearn import linear_model
>>> reg = linear_model.LinearRegression()
>>> reg.fit ([[0, 0], [1, 1], [2, 2]], [0, 1])
>>> reg.coef_
array([ 0.5, 0.5])
```



Полиномиальная регрессия

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import PolynomialFeatures
……
model = PolynomialFeatures(degree)
model.fit(X_train, y_train)
y_test = model.predict(X_test)
```



Мультиколлинеарность признаков

$$\mu(\Sigma) = \|\Sigma\| \|\Sigma^{-1}\| = \frac{\max\limits_{u:\, \|u\|=1} \|\Sigma u\|}{\min\limits_{u:\, \|u\|=1} \|\Sigma u\|} = \frac{\lambda_{\max}}{\lambda_{\min}},$$

Матрица ковариации $\mu(\Sigma) \gtrsim 10^2 \dots 10^4.$

<u>Следствие</u>: неустойчивость МНК (небольшое изменение входных данных сильно влияет на решение)

Ridge - регрессия

$$\mathrm{RSS}(\lambda) = (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta) + \lambda\beta^T\beta,$$

$$\hat{\beta}^{\mathrm{ridge}} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}$$

```
>>> from sklearn import linear_model
>>> reg = linear_model.Ridge (alpha = .5)
>>> reg.fit ([[0, 0], [0, 0], [1, 1]], [0, .1, 1])
>>> reg.coef_
 array([ 0.34545455, 0.34545455])
>>> reg.intercept_
0.13636...
```

```
>>> from sklearn import linear_model
>>> reg = linear_model.RidgeCV(alphas=[0.1, 1.0, 10.0])
>>> reg.fit([[0, 0], [0, 0], [1, 1]], [0, .1, 1])
>>> reg.alpha_
0.1
```

LASSO - регрессия

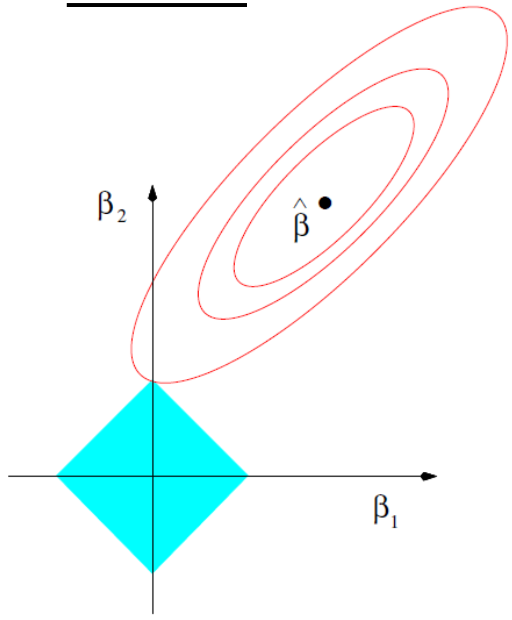
$$\hat{\beta}^{\text{lasso}} = \underset{\beta}{\operatorname{argmin}} \left\{ \frac{1}{2} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\}$$

```
>>> from sklearn import linear_model
>>> reg = linear_model.Lasso(alpha = 0.1)
>>> reg.fit([[0, 0], [1, 1]], [0, 1])
>>> reg.predict([[1, 1]])
 array([ 0.8])
```

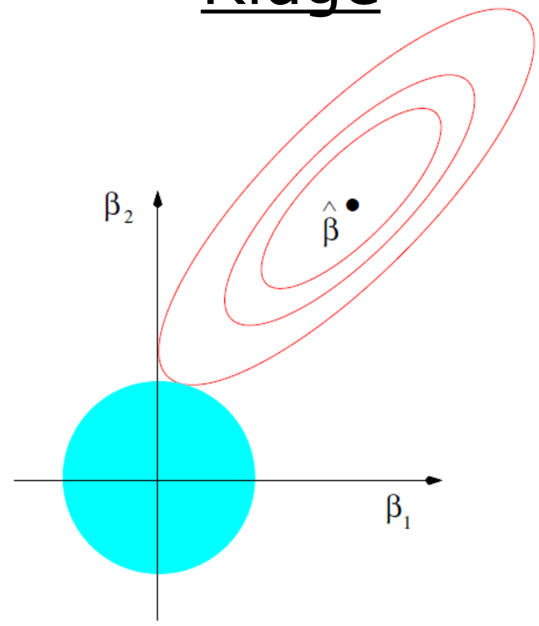
```
import numpy as np
from sklearn import datasets
from sklearn.linear_model import Lasso
from sklearn.model_selection import cross_val_score
diabetes = datasets.load_diabetes()
X = diabetes.data[:150]
y = diabetes.target[:150]
lasso = Lasso(random_state=0)
alphas = np.logspace(-4, -0.5, 30)
scores = list()
scores_std = list()
n_folds = 3
for alpha in alphas:
	lasso.alpha = alpha
	this_scores = cross_val_score(lasso, X, y, cv=n_folds, n_jobs=1)
	scores.append(np.mean(this_scores))
	scores_std.append(np.std(this_scores))
scores, scores_std = np.array(scores), np.array(scores_std)
```

Сравнение Ridge и Lasso

Lasso

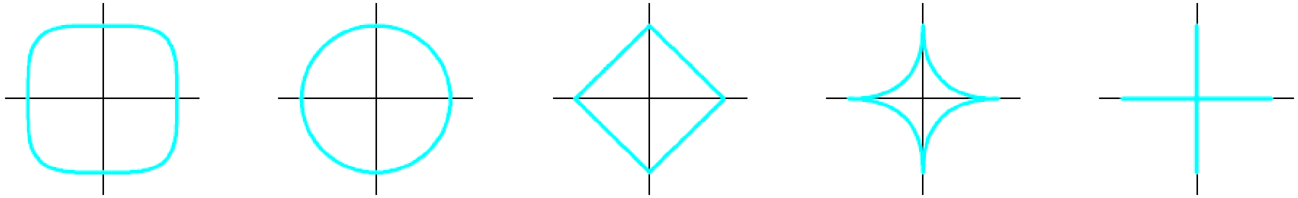


Ridge



$$\tilde{\beta} = \underset{\beta}{\operatorname{argmin}} \left\{ \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j|^q \right\}$$

$q = 4$ $q = 2$ $q = 1$ $q = 0.5$ $q = 0.1$



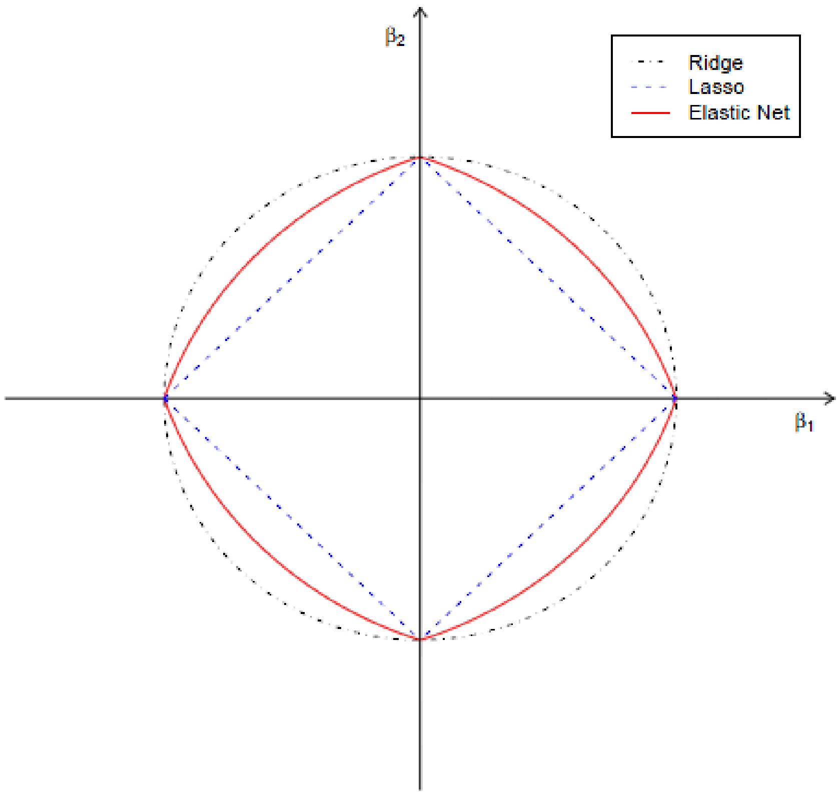
Elastic Net

$$\min_{w} \frac{1}{2n_{samples}} ||Xw - y||_2^2 + \alpha\rho||w||_1 + \frac{\alpha(1-\rho)}{2}||w||_2^2$$

```
import numpy as np
from sklearn import datasets
from sklearn.linear_model import ElasticNetCV
from sklearn.model_selection import
cross_val_score

diabetes = datasets.load_diabetes()
X = diabetes.data[:150]
y = diabetes.target[:150]

alphas=[0.1, 1.0, 10.0]
reg = linear_model.ElasticNetCV(l1_ratio=0.1,
alphas=alphas)
error = np.abs(cross_val_score(reg, X, y,
cv=5)).mean()
```



Логистическая регрессия

$$\Pr(G=1|X=x) = \frac{\exp(\beta_0 + \beta^T x)}{1 + \exp(\beta_0 + \beta^T x)},$$
$$\Pr(G=2|X=x) = \frac{1}{1 + \exp(\beta_0 + \beta^T x)}.$$

$$\log \frac{\Pr(G=1|X=x)}{\Pr(G=2|X=x)} = \beta_0 + \beta^T x.$$

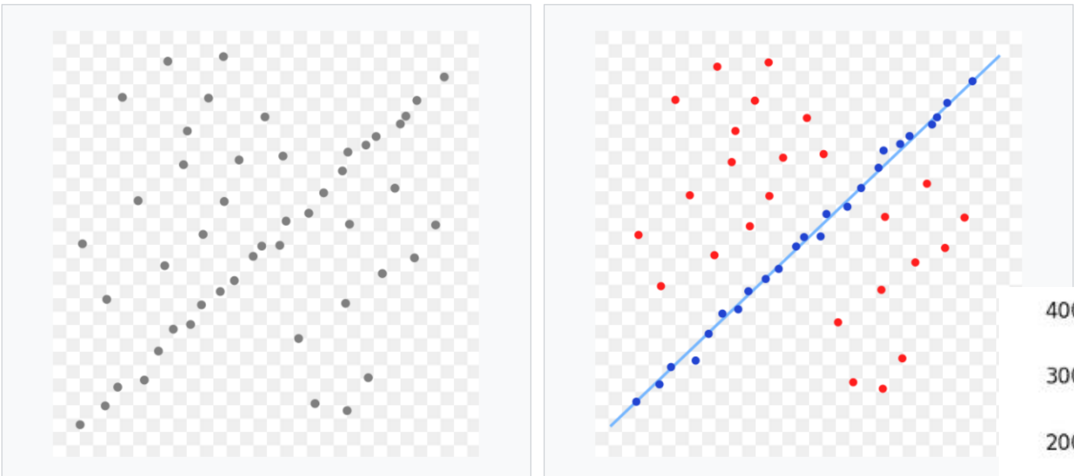
L2: $$\min_{w,c} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \log(\exp(-y_i(X_i^T w + c)) + 1)$$

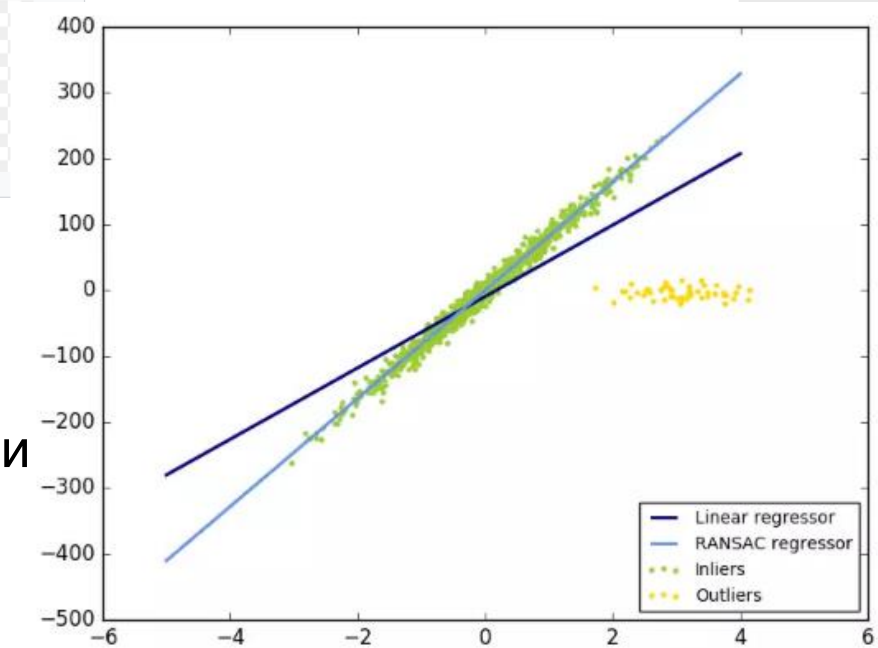
L1: $$\min_{w,c} \|w\|_1 + C \sum_{i=1}^{n} \log(\exp(-y_i(X_i^T w + c)) + 1).$$

Case	Solver
Small dataset or L1 penalty	“liblinear”
Multinomial loss or large dataset	“lbfgs”, “sag” or “newton-cg”
Very Large dataset	“sag”

Робастная регрессия

RANdom SAmple Consensus



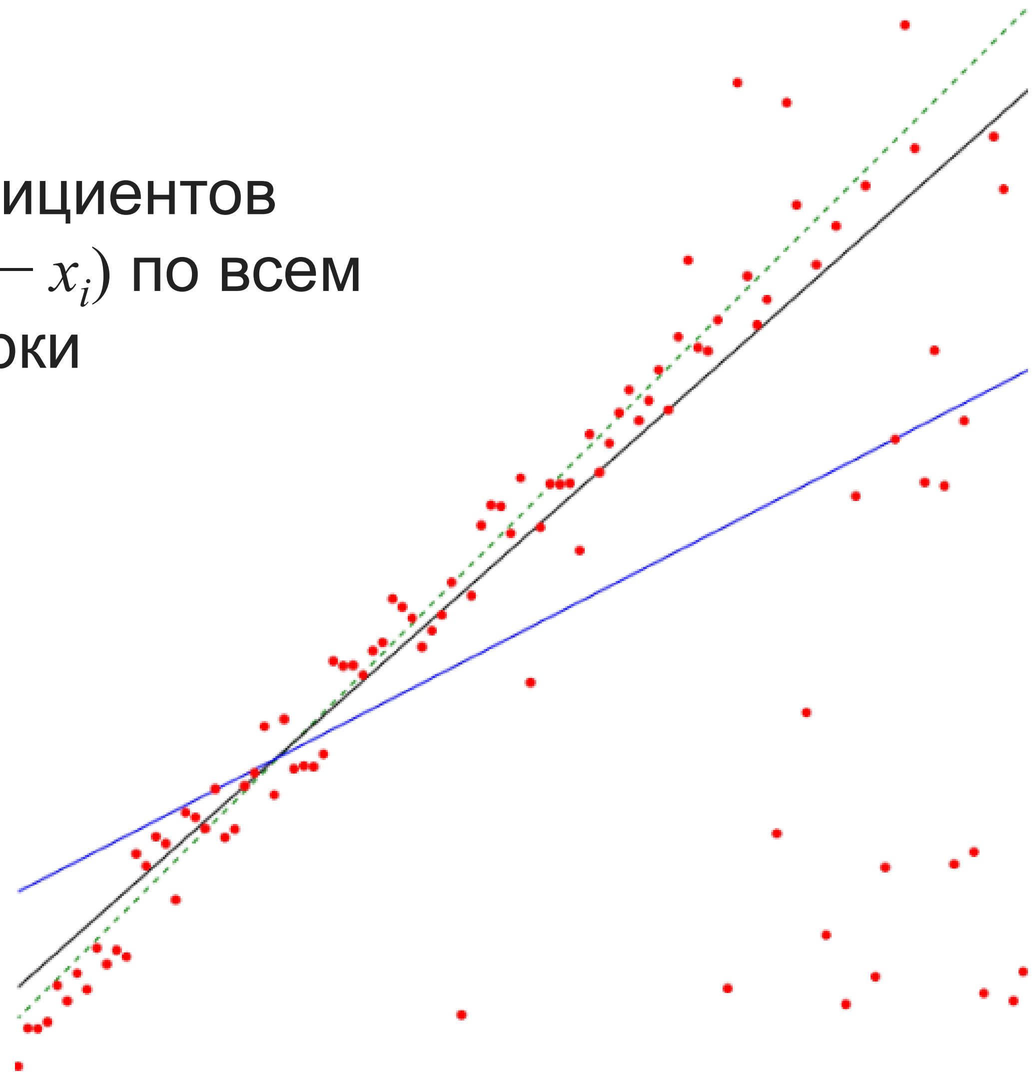


- Выбор случайного поднабора точек
- Построение модели регрессии
- Точка классифицируется выбросом при ошибке модели больше заданного порога
- Выбор модели с минимальным количеством выбросов

Робастная регрессия

Функция Тейла-Сена

Медиана m коэффициентов наклона $(y_j - y_i)/(x_j - x_i)$ по всем парам точек выборки



Робастная регрессия

Функция Хьюбера

$$\min_{w,\sigma} \sum_{i=1}^{n} \left(\sigma + H_m \left(\frac{X_i w - y_i}{\sigma} \right) \sigma \right) + \alpha ||w||_2{}^2$$

$$H_m(z) = \begin{cases} z^2, & \text{if } |z| < \epsilon, \\ 2\epsilon|z| - \epsilon^2, & \text{otherwise} \end{cases}$$