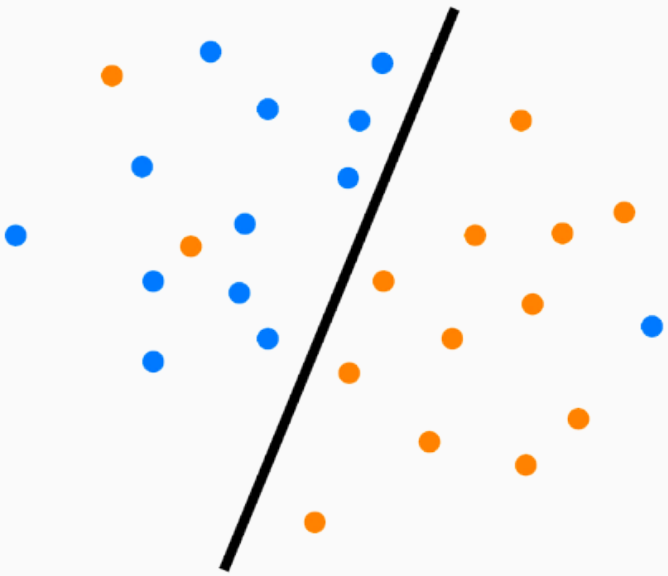
Семинар 8. Линейная классификация

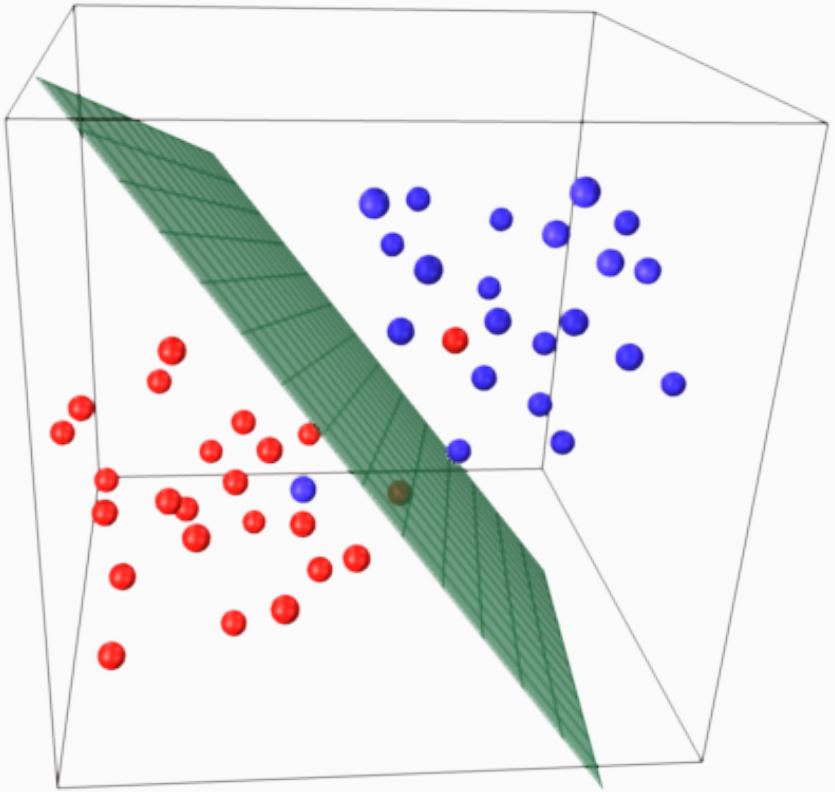
Даулбаев Талгат

15 марта 2016 г.

Линейная классификация: картинки

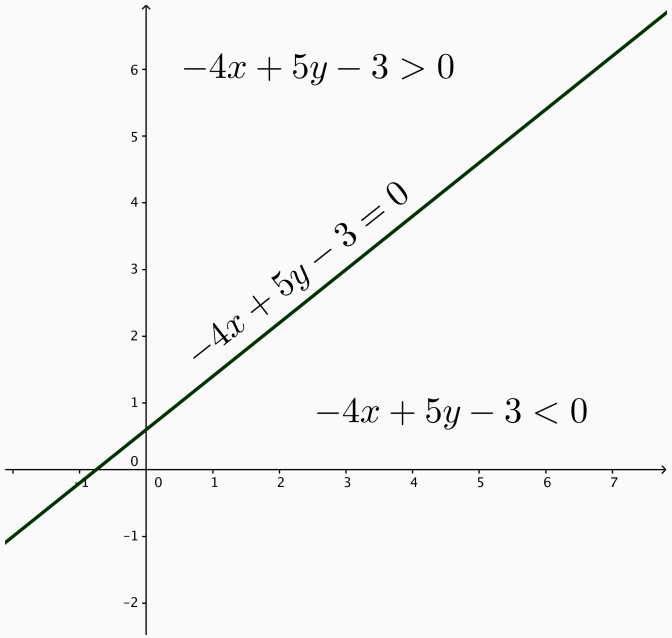


Пространство двух признаков

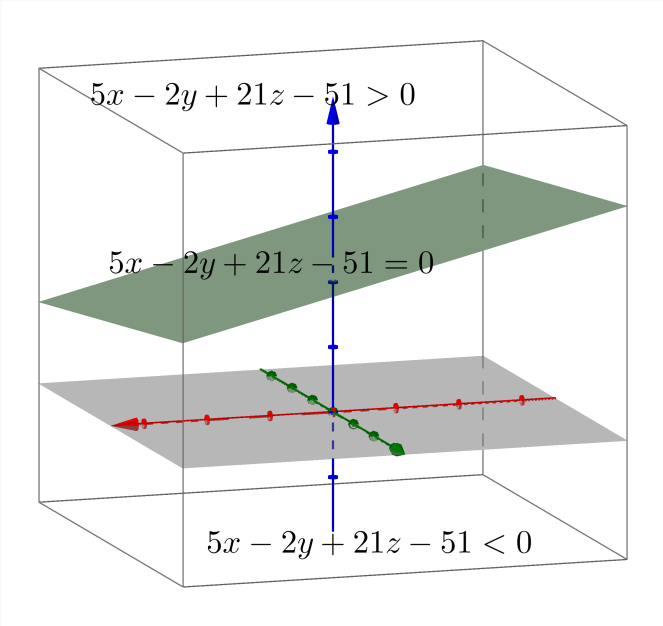


Пространство трёх признаков

Линейная классификация: картинки



$d = 2$



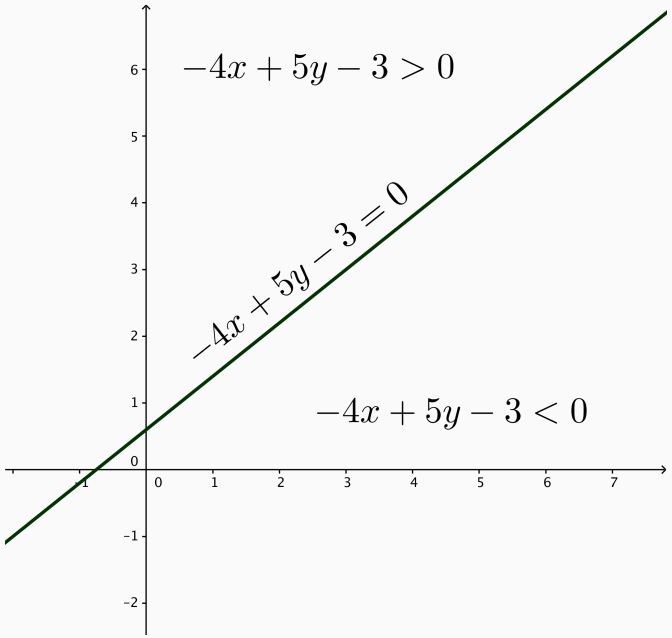
$d = 3$

$\mathbb{X} = \mathbb{R}^d$ — множество объектов, $\mathbb{Y} = \{-1, +1\}$ — множество ответов

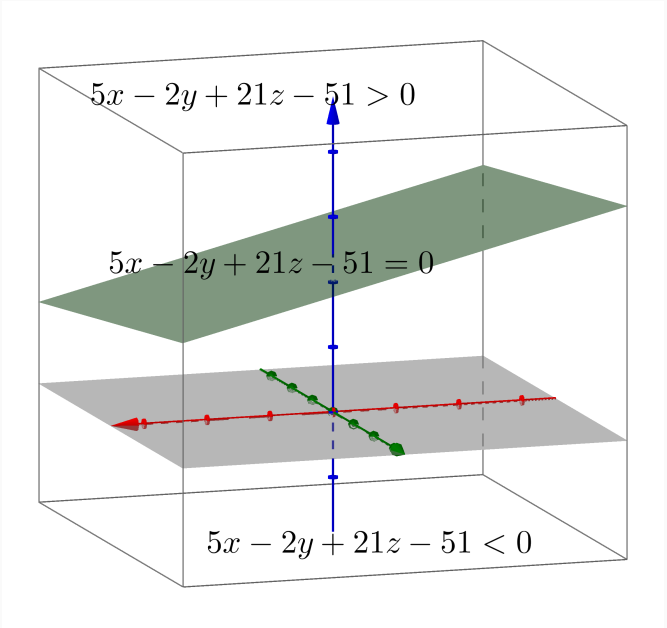
$X = \{(x_1, y_1), (x_2, y_2), \ldots, (x_\ell, y_\ell)\}$ — обучающая выборка.

Вопрос: как выглядит формула линейного классификатора $a(x)$?

Линейная классификация: картинки



$d = 2$



$d = 3$

$y_i \in \{-1, +1\}$

$$\boxed{a(x) = \text{sign}\left(\langle w, x\rangle + w_0\right)}$$

Далее будем считать, что добавлен константный признак, и $w_0 = 0$.

Как найти вектор w?

Для каждого объекта обучающей выборки x_i введём понятие отступа (margin):

$$M_i(w) = y_i \langle w, x_i \rangle.$$

Так как $y_i \in \{-1, +1\}$, $a(x) = \text{sign}\langle w, x \rangle$, то

- $M_i(w) < 0 \quad \Rightarrow \quad$ классификатор выдал неверный ответ
- $M_i(w) > 0 \quad \Rightarrow \quad$ классификатор выдал правильный ответ

Хотим минимизировать количество ошибок на обучающей выборке. Математически это записывается как:

$$Q(w) = \sum_{i=1}^{\ell} [M_i(w) < 0] \to \min_w$$

Здесь квадратные скобки обозначают так называемую нотацию Айверсона.

Аппроксимация пороговой функции

$$Q(w) = \sum_{i=1}^{\ell} [M_i(w) < 0] \to \min_w$$

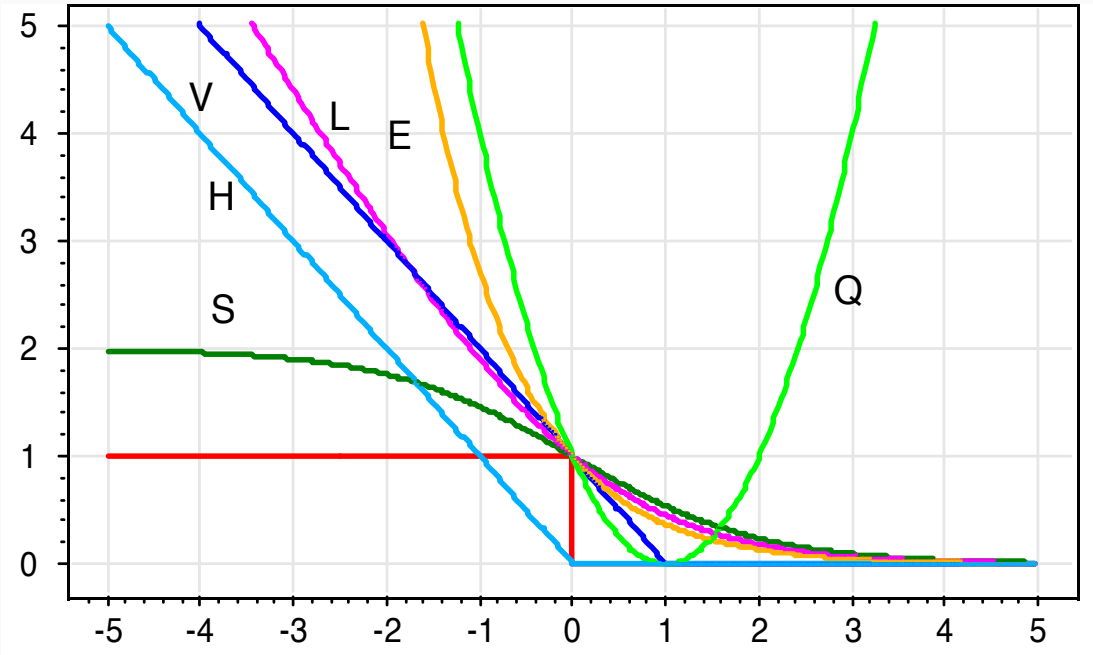
— сложно решать, потому что $Q(w)$ недифференцируемая.

Ограничим $[M_i(w) < 0]$ сверху некоторой более гладкой, невозрастающей, неотрицательной функцией $\mathscr{L}(M_i(w))$:

$$Q(w) = \sum_{i=1}^{\ell} [M_i(w) < 0] \leq \widetilde{Q}(w) = \sum_{i=1}^{\ell} \mathscr{L}(M_i(w)) \to \min_w$$

Получили задачу непрерывной оптимизации, которая решается градиентными методами!

Аппроксимация пороговой функции



$Q(M) = (1 - M)^2$ — квадратичная

$V(M) = (1 - M)_+$ — кусочно-линейная (SVM)

$S(M) = 2(1 + e^M)^{-1}$ — сигмоидная (нейросети)

$L(M) = \log(1 + e^{-M})$ — логарифмическая (Logistic Regression)

$E(M) = e^{-M}$ — экспоненциальная (AdaBoost)

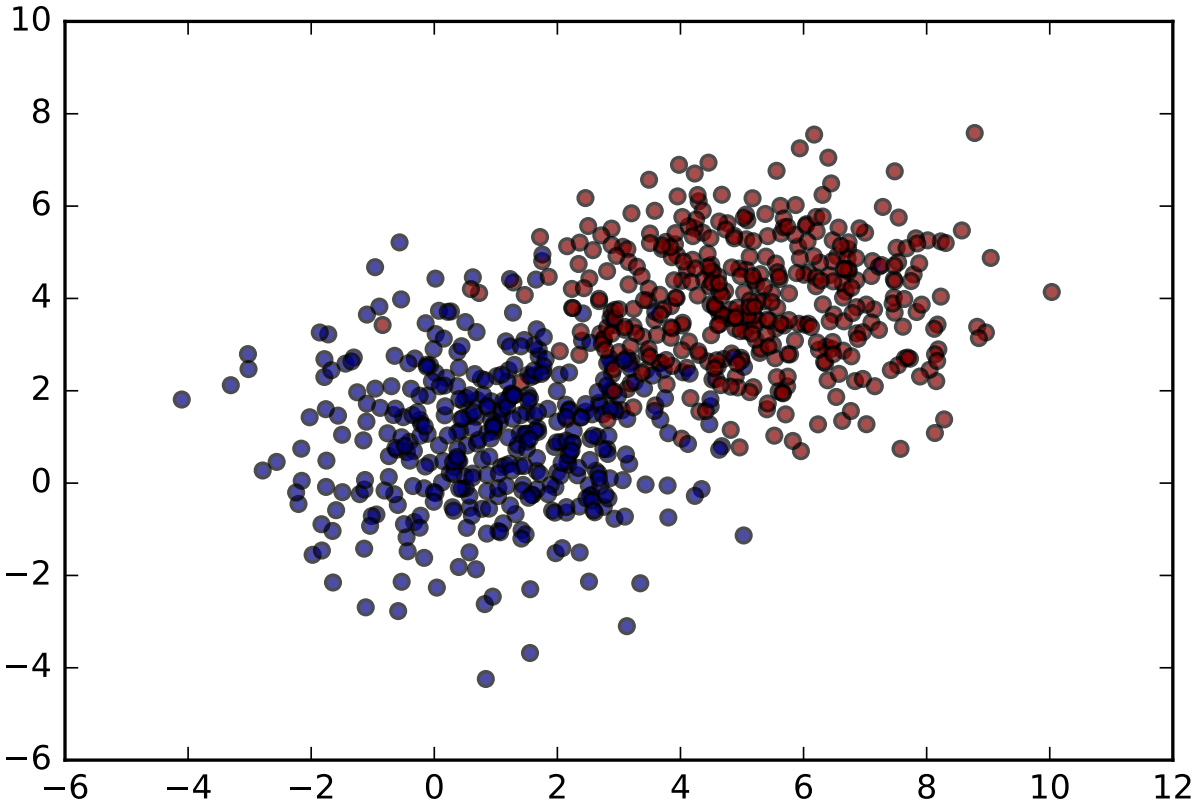
Данные

Сгенерируем данные из нормального распределения

```
Sigma = np.array([[3, 0.1], [0.5, 2]])
mu_1 = np.array([1, 1])
mu_2 = np.array([5, 4])
n = 500
X1 = np.random.multivariate_normal(mu_1, Sigma, n)
X2 = np.random.multivariate_normal(mu_2, Sigma, n)
X = np.vstack((X1, X2))
y = np.hstack((np.zeros(n), np.ones(n)))
volume = 2 * n
train_volume = int(0.8 * volume)
idx = np.arange(volume)
np.random.shuffle(idx)
X_train, y_train = X[idx[:train_volume], :], y[idx[:train_volume]]
X_test, y_test = X[idx[train_volume:], :], y[idx[train_volume:]]
```

Данные

```
# Нарисуем картинки:
plt.scatter(X_train[:, 0], X_train[:, 1], c=y_train, alpha=0.7)
plt.savefig('./normal.pdf')
```



Логистическая регрессия (logistic regression, logit regression)

$$\widetilde{Q}(w) = \frac{1}{\ell}\sum_{i=1}^{\ell} \log\left(1 + \exp\{-y_i\langle w, x_i\rangle\}\right) \to \min_w$$

Пусть $\pi(x)$ — вероятность того, что объект x принадлежит классу $+1$.

$$\log\frac{\pi(x)}{1-\pi(x)} = \langle w, x\rangle \quad \Rightarrow \quad \pi(x) = \frac{1}{1+\exp\{-\langle w, x\rangle\}}$$

Поэтому логистическая регрессия может выдавать оценки вероятности принадлежности объекта классу.

Логистическая регрессия

Логистическая регрессия — это метод для классификации, а не для регрессии!

Логистическая регрессия и регуляризация

L2-регуляризация:

$$\frac{1}{\ell}\sum_{i=1}^{\ell}\log\left(1+\exp\{-y_i\langle w,x_i\rangle\}\right)+\frac{1}{C}\|w\|_2^2\to\min_w$$

L1-регуляризация:

$$\frac{1}{\ell}\sum_{i=1}^{\ell}\log\left(1+\exp\{-y_i\langle w,x_i\rangle\}\right)+\frac{1}{C}\|w\|_1\to\min_w$$

C — гиперпараметр. **Вопрос:** как его выбирать?

Логистическая регрессия в sklearn

```
LogisticRegression(self, penalty='l2', tol=0.0001,
    C=1.0,
    class_weight=None, # balanced
    max_iter=100,
    multi_class='ovr')

# Атрибуты:
.coef_
.intercept_

# Методы:
.fit(X, y)
.predict_proba(X)
.predict(X)
```

Логистическая регрессия в sklearn

```
lr = LogisticRegression()
lr.fit(X_train, y_train)
y_pred = lr.predict(X_test)
print("accuracy:", np.mean(y_pred == y_test))

proba = lr.predict_proba(X_test)
print(proba.shape)
```

У меня получилась точность 0.935

Подбор гиперпараметра C

```
from sklearn.linear_model import LogisticRegressionCV

# Cs - это либо список значений, либо число
# Если число, то перебираем по сетке из Cs значений
lr = LogisticRegressionCV(Cs=100)
lr.fit(X_train, y_train)
# Оптимальное значение:
lr.C_
```

Матрица ошибок

	$y = 1$	$y = -1$
$a(x) = 1$	True Positive (TP)	False Positive (FP)
$a(x) = -1$	False Negative (FN)	True Negative (TN)

Доля правильный ответов (accuracy):

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{FP} + \text{TN}}$$

Точность:

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Полнота:

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Слайд о том, почему иногда нужно выбирать разный порог

$$a(x) = [b(x) > t]$$

$b(x)$ — оценка вероятностей принадлежности классу, t — порог.

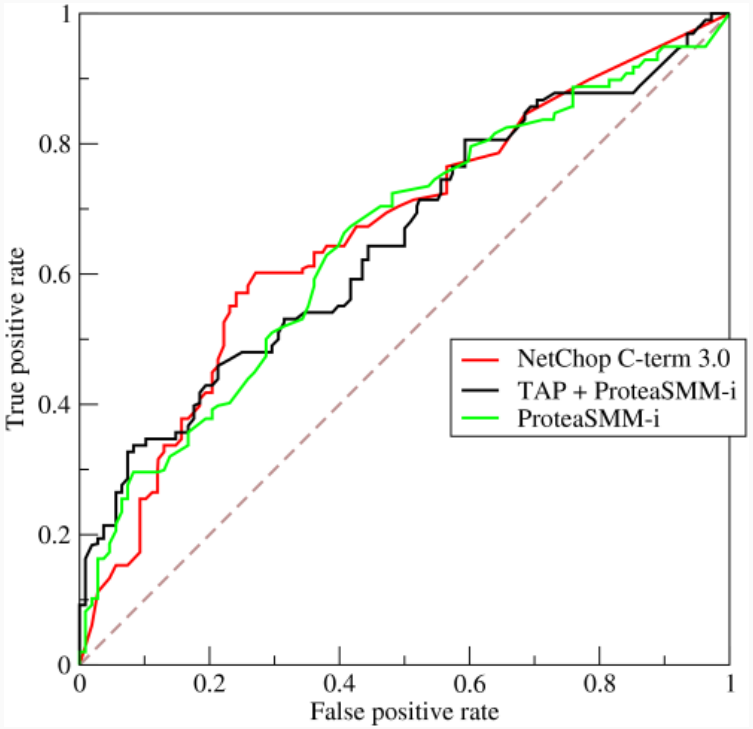
Примеры из лекций:

- Кредитный скоринг
- Медицинская диагностика

ROC (Receiver Operating Curves)

False Positive Rate (FPR) и True Positive Rate (TPR):

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}, \quad \text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$



$a(x) = [b(x) > t]$, каждому t соответствует точка, а всего разных порогов $\ell + 1$.

ROC и AUC

У хорошего алгоритма ROC проходит как можно «выше и правее». В идеале: график, похожий на букву «Г».

Поэтому оценивать качество алгоритма можно по площади под ROC-кривой, эта метрика называется AUC (area under curve).

ROC и AUC

- Продолжим говорить про метрики качества и их вычисление в sklearn
- Поговорим пр предобработку признаков: one-hot-encoding, tf-idf, ...
- Обсудим, как придумывать признаки

Вопросы?