# Логистическая регрессиия

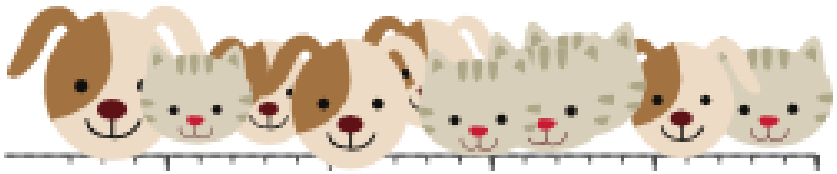
Методы анализа данных

Москва, МФТИ, 2020

# План

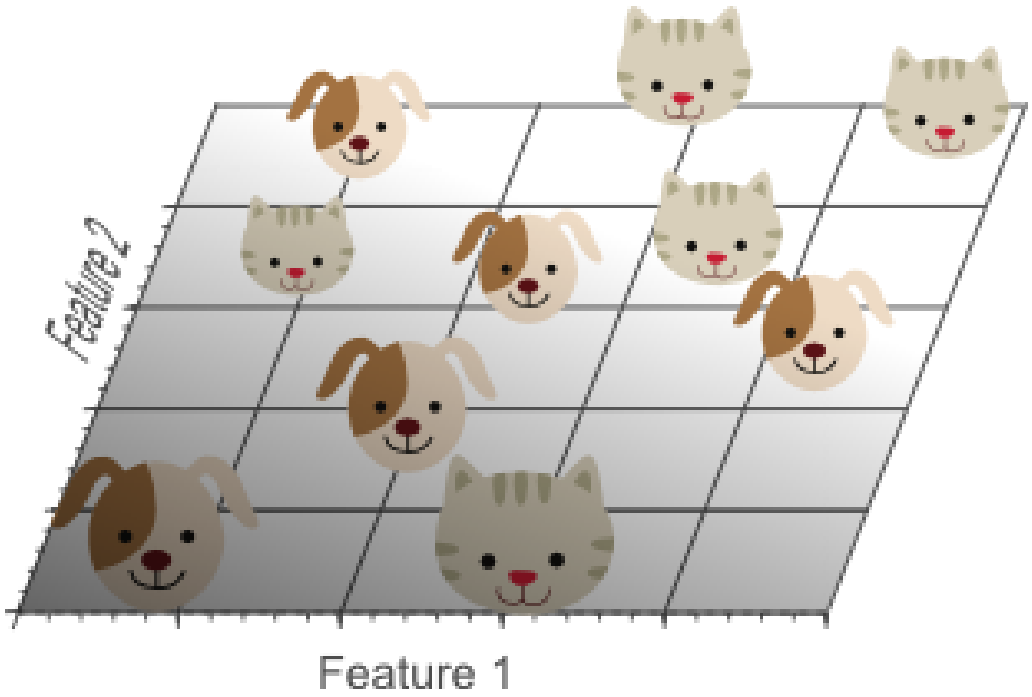
- Линейные модели
- Логистическая регрессия
- Обучение логистической регрессии
- Практические советы при обучении

# 1D разделимость

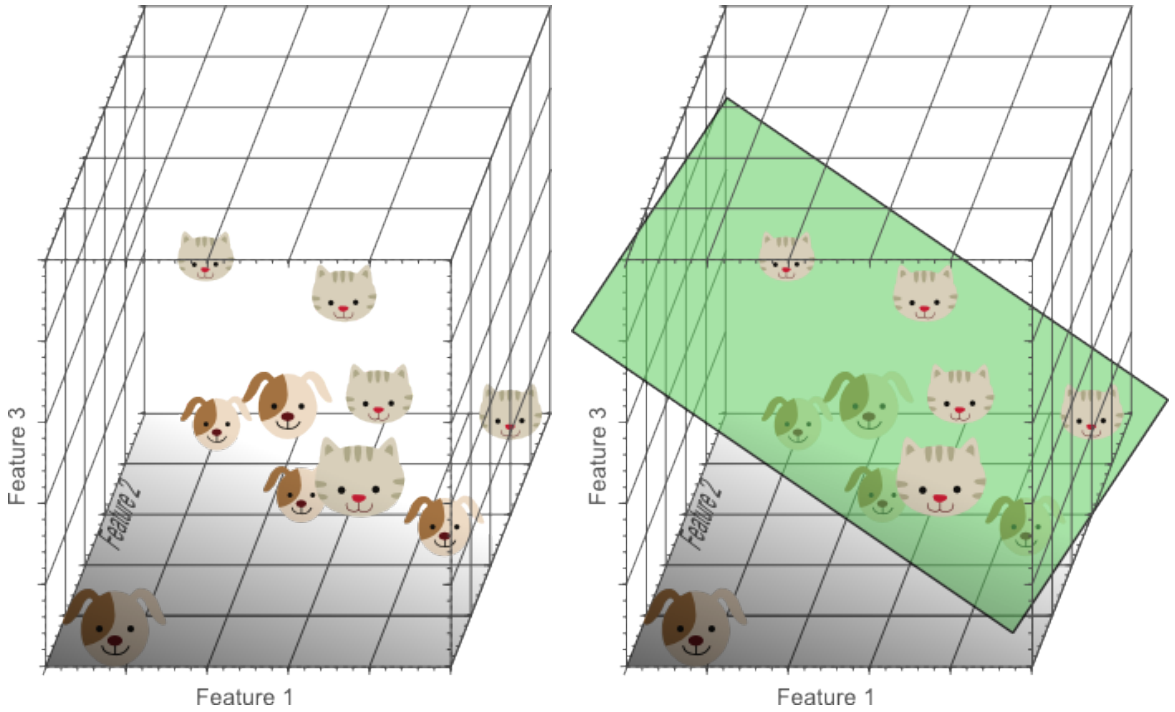


Feature 1

# 2D разделимость



# 3D разделимость



# Модель для линейной регрессии

$$a(x) = w_0 + \sum_{j=1}^{d} w_j x^j \tag{1}$$

Обозначения:

- $w_0$ - свободный коэффициент
- $x^j$ - признаки
- $w_j$ - веса признаков

$$a(x) = \sum_{j=1}^{d+1} w_j x^j = w \cdot x \tag{2}$$

# Модель для логистической регрессии

Линейная регрессия

$$a(x) = w_0 + \sum_{j=1}^{d} w_j x^j \tag{3}$$

Данная прямая задает нормаль для разделяющей гиперпоскости
Логистическая регрессия

$$h(x) = a(x) > 0 \tag{4}$$

Данная гиперпоскость разделяет классы

- $a(x) > 0 \Rightarrow x \in K_0$ - нулевой класс
- $a(x) \leq 0 \Rightarrow x \in K_1$ - первый первый класс

# Пример линейной модели для классфиикации

Стоит ли пойти работать в Яндекс?

- Зарплата: 100000
- Расстояние до дома: 20км
- ДМС: Да
- Я ленивый?: Да

# Пример линейной модели для классфиикации

Стоит ли пойти работать в Яндекс?

- Зарплата: 100000
- Расстояние до дома: 20км
- ДМС: Да
- Я ленивый?: Да

Построим модель (свободный член в начале)

- $x = (1, 100000, 20, 1, 1)$ - вектор объекта
- $w = (20, 2/10000, -1, 30, -45)$ - вектор весов
- $a(x) = 20 + 20 - 20 + 30 - 45 = 5 > 0$

Вердикт - идем!

# Обучение логистической регрессии

Введем метки классов:

- $y_i = 1$ - класс $K_1$
- $y_i = -1$ - класс $K_0$

Отступ (не путать с зазором в SVM)

$$M(x_i) = y_i \langle x_i, w \rangle \tag{5}$$

Отступ положителен если мы верно классифицировалии $x_i$ и отрицателен если нет

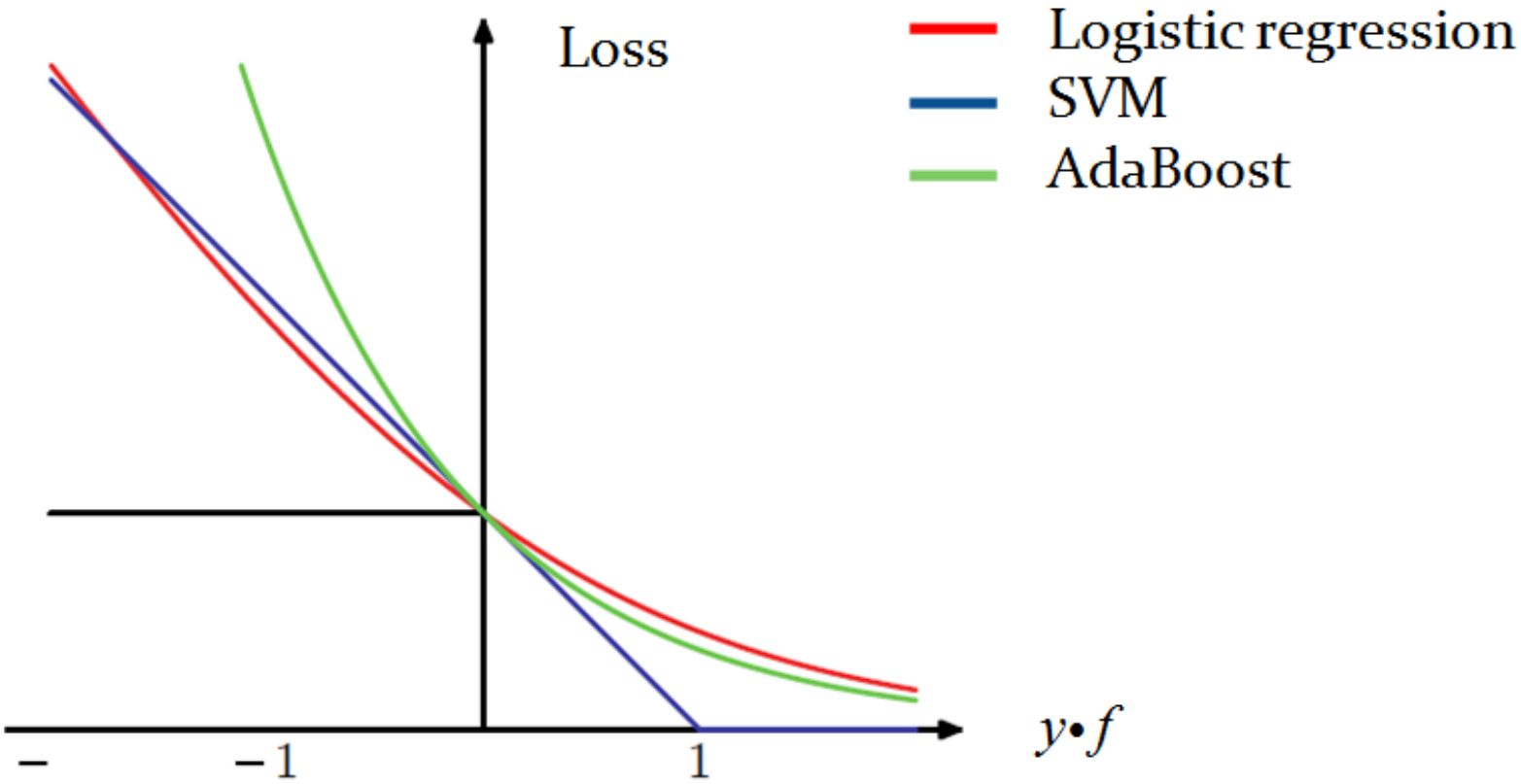
# Эмперический риск

Эмперический риск - сумма ошибок классификации

$$Q(w) = \sum_{i=1}^{m} \left[ a(x_i, w) \neq y_i \right] = \sum_{i=1}^{m} \left[ M(x_i) < 0 \right] \to \min_w \tag{6}$$

Цель обучения - минимизировать эмпирический риск на обучающей выборке. Минимизация эмперического риска - сложная комбинаторная задача.

# Аппроксимация эмперического риска




В случае логистической регрессии $\left[M < 0\right] \leq \log_2(1 + e^{-M})$

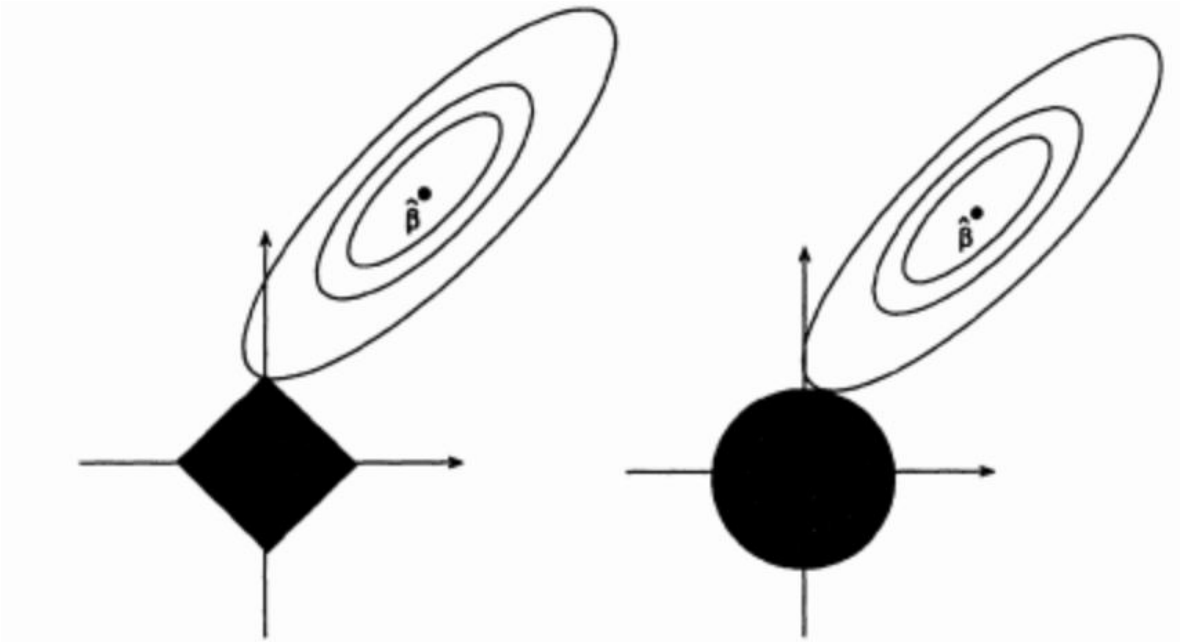
# Функция потерь логистической регрессии

$$L_{log}(X, \vec{y}, \vec{w}) = \sum_{i=1}^{\ell} \log(1 + \exp^{-y_i \langle x_i, w \rangle}) \tag{7}$$

Данная функция минимизируется различными алгоритмами градиентного спуска, в том чиисле стохастического (SGD)

Обзор разных алгоритмов оптимизации
`https://ruder.io/optimizing-gradient-descent/`

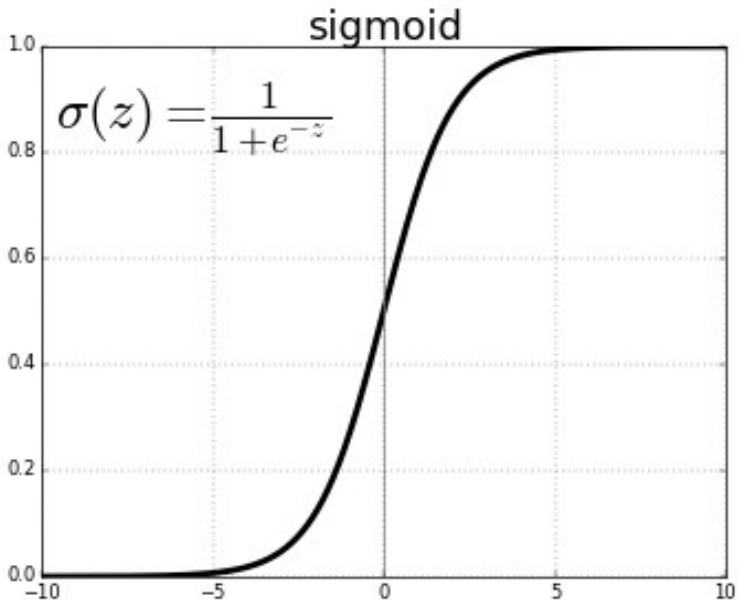
# Регуляризация




$$\arg\min_{\vec{w}} L_2(X, \vec{y}, \vec{w}) = \arg\min_{\vec{w}} \left(C \sum_{i=1}^{\ell} \log(1 + \exp^{-y_i \langle x_i, w \rangle}) + |\vec{w}|^2\right) \quad (8)$$

$$\arg\min_{\vec{w}} L_1(X, \vec{y}, \vec{w}) = \arg\min_{\vec{w}} \left(C \sum_{i=1}^{\ell} \log(1 + \exp^{-y_i \langle x_i, w \rangle}) + |\vec{w}|\right) \quad (9)$$

# Сигмоида как оценка вероятности




Хотелось бы использовать данную оценку вероятности для $i$-го объекта

$$p_i = \sigma(\langle x_i, w \rangle) = \frac{1}{1 + \exp^{-\langle x_i, w \rangle}} \tag{10}$$

# Связь с линейной регрессией

Задача обучения линейной регрессии сводится к задаче минимизации ошибки:

$$\min_w \frac{1}{\ell} \sum_{j=1}^{\ell} (\langle x_i, w \rangle - y_i)^2 \tag{11}$$

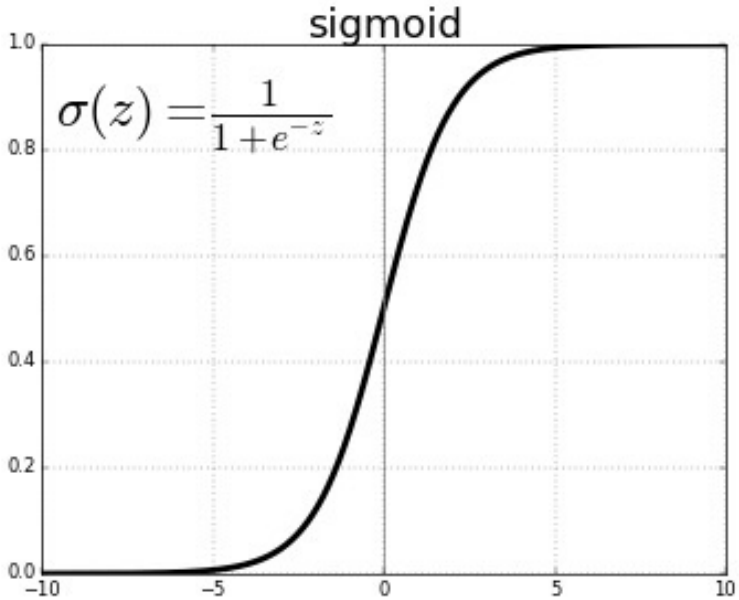
- $p_i = \sigma(\langle x_i, w \rangle) \in [0, 1]$ - вероятность
- $log(\frac{p_i}{1-p_i}) \in \mathbb{R}$ - логарифм отношения вероятностей, его и будем предсказывать линейной регрессией

Таким образом

$$log(\frac{p_i}{1-p_i}) = \langle x_i, w \rangle \tag{12}$$

Отсюда получаем $p_i = \sigma(\langle x_i, w \rangle)$

# Сигмоида как оценка вероятности




Оценка вероятности для $i$-го объекта

$$p_i = \sigma(\langle x_i, w \rangle) = \frac{1}{1 + \exp^{-\langle x_i, w \rangle}} \tag{13}$$

Функция правдоподобия

$$P(\vec{y} \mid X, \vec{w}) = \prod_{i=1}^{\ell} P(y = y_i \mid \vec{x_i}, \vec{w}) \tag{14}$$

# LogLoss

Пусть $p_i$ - вероятность принадлежности в 1 класс
Пусть $y_i \in 0, 1$ Функция правдоподобия через схему Бернулли

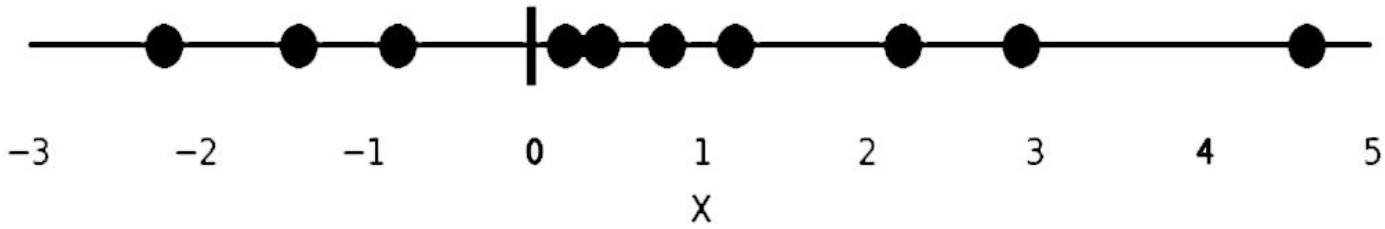
$$P(\vec{y} \mid X, \vec{w}) = \prod_{i=1}^{\ell} P(y = y_i \mid \vec{x_i}, \vec{w}) = \prod_{i=1}^{\ell} p_i^{y_i}(1 - p_i)^{1-y_i} \to max \quad (15)$$

Прологорифмируем и получим LogLoss

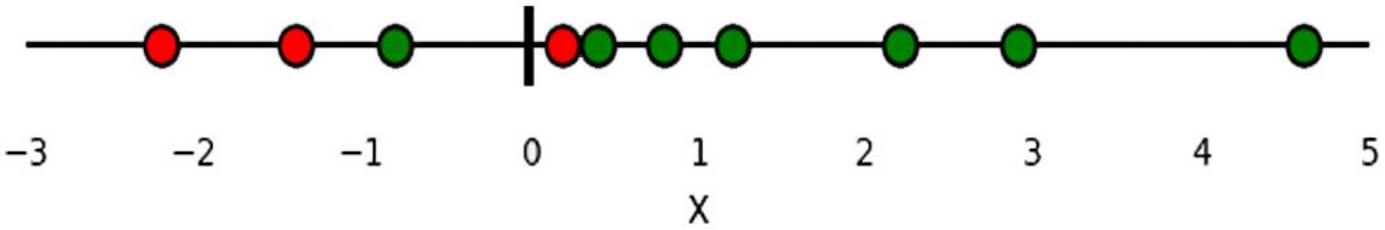
$$logloss = \sum_{i=1}^{\ell} (-y_i log(p_i) - (1 - y_i) log(1 - p_i)) \to min \quad (16)$$

# LogLoss - пример

Рассматорим задачу из одного признакаазом

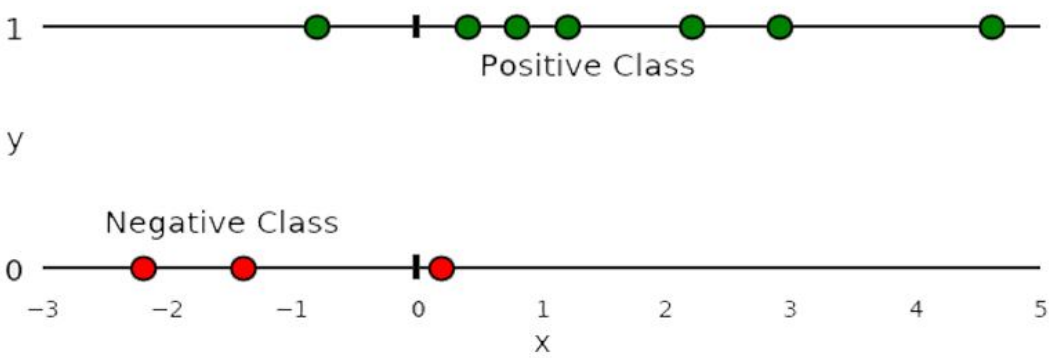



Пускай объекты разделены на классы следующим образом

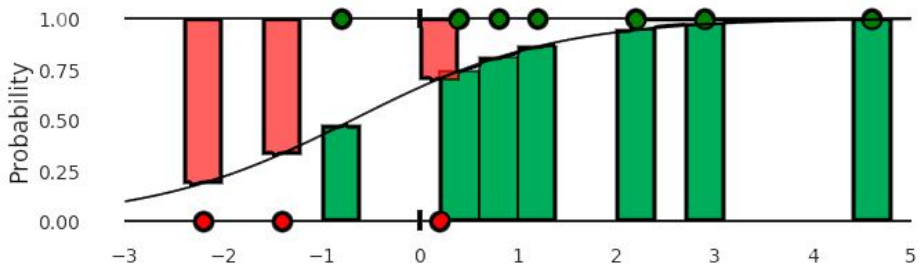


# LogLoss - пример

На данной бинарной задаче

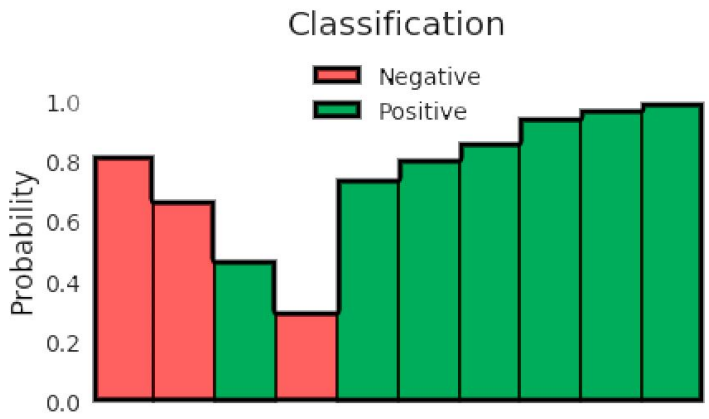



Обучим логистическую регрессию

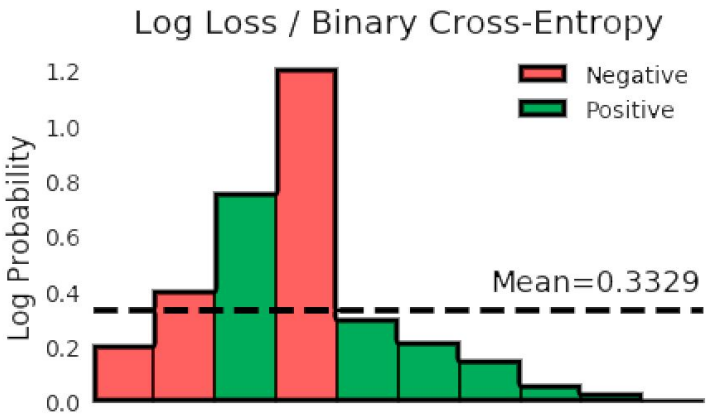


# LogLoss - пример

Получим вероятности

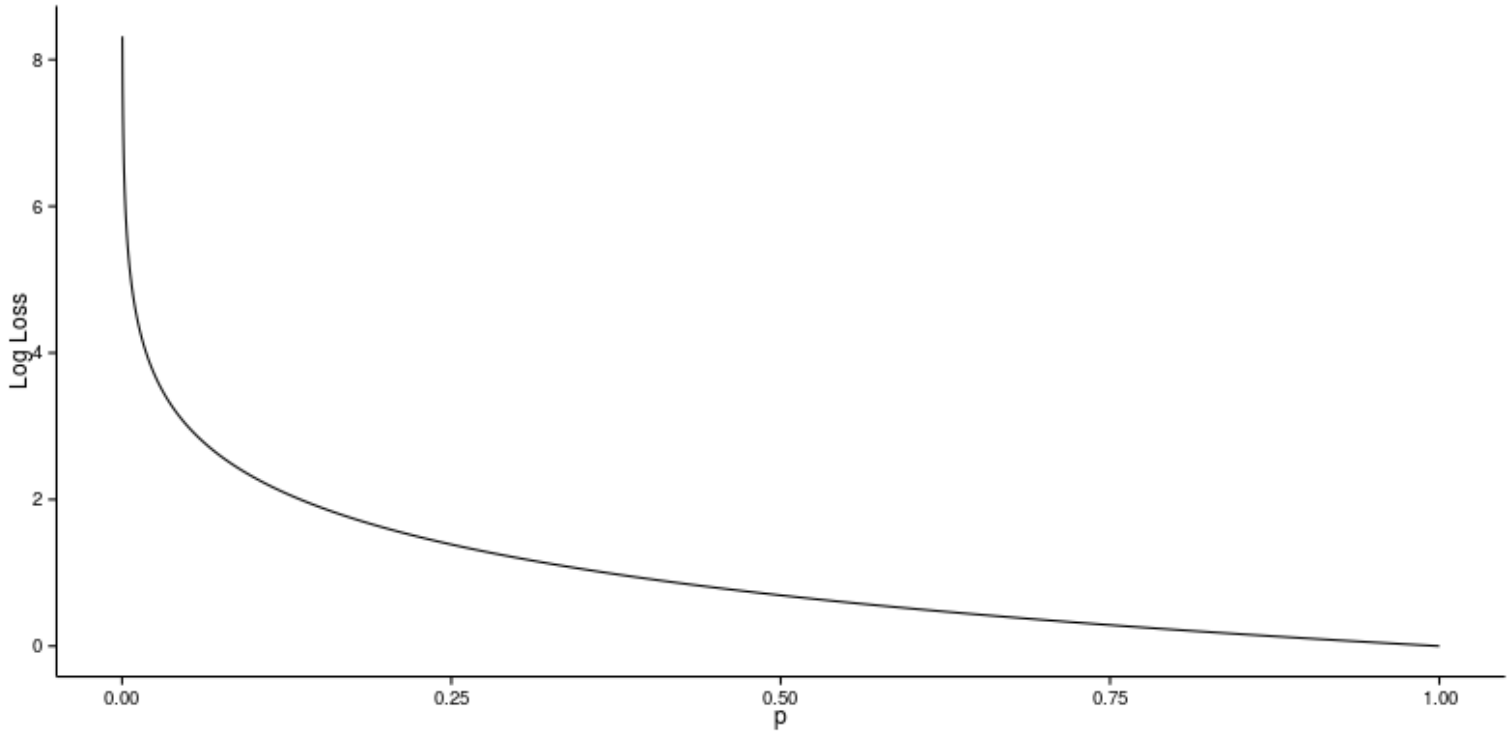



И прологорифмируем их



# LogLoss - пример

График логлосса для одного объекта (0 класс)



# LogLoss в случае нескольких классов

Бинарная классификация

$$logloss = \frac{1}{\ell}\sum_{i=1}^{\ell}(-y_i log(p_i) - (1-y_i) log(1-p_i)) \tag{17}$$

Многоклассовая классификация

$$logloss = \frac{1}{\ell}\sum_{i=1}^{\ell}\sum_{j=1}^{K}(-y_{i,j} log(p_{i,j})) \tag{18}$$

# Логистическая регрессия - подготовка

- Заполнение NaN
- StandardScaler/MinMaxScaler (повышает точность и ускоряет)
- Замена кат признаков на OHE, TargetEncoding (LabelEncoding обычно плохо работает, так как у меток нет порядка)
  Но иногда можно, с учетом смысла:
  `education = {'No':0, 'School': 1, 'University': 2, 'PhD': 3}` лы

# Логистическая регрессия - подготовка

- Признаки с нелинейной зависимостью можно разбить на интервалы

```
df['temp_good'] = df.loc[(df['temp'] < 37)&(df['temp'] > 36), 'temp'].fillna(0)
df['temp_bad'] = df.loc[~((df['temp'] < 37)&(df['temp'] > 36)), 'temp'].fillna(0)
df.fillna(0)
```

| | temp | temp_good | temp_bad |
|---|---|---|---|
| 0 | 34.5 | 0.0 | 34.5 |
| 1 | 39.0 | 0.0 | 39.0 |
| 2 | 36.6 | 36.6 | 0.0 |
| 3 | 36.1 | 36.1 | 0.0 |
| 4 | 35.0 | 0.0 | 35.0 |
| 5 | 41.0 | 0.0 | 41.0 |

# Логистическая регрессия - интерпретация результатов

- Коэффициенты признаков
- Вклад признаков (помноженный на значения признаков)

eli5: `https://github.com/TeamHG-Memex/eli5`

| y=0 top features | | y=1 top features | | y=2 top features | |
|---|---|---|---|---|---|
| Weight[?] | Feature | Weight[?] | Feature | Weight[?] | Feature |
| +0.772 | keith | +1.096 | graphics | +0.948 | rutgers |
| +0.656 | okcforum | +0.637 | software | +0.817 | christians |
| +0.625 | mathew | +0.609 | image | +0.754 | church |
| +0.593 | atheism | +0.586 | host | +0.734 | clh |
| +0.574 | writes | +0.573 | nntp | +0.681 | christ |
| +0.541 | psuvm | +0.529 | 42 | +0.610 | athos |
| +0.523 | wingate | +0.510 | tiff | +0.534 | christian |
| +0.511 | umd | +0.506 | looking | +0.528 | 1993 |
| +0.504 | benedikt | +0.501 | files | +0.495 | petch |
| +0.501 | islamic | +0.481 | ftp | +0.482 | love |
| +0.482 | psu | +0.473 | card | +0.450 | bassili |
| *... 10732 more positive ...* | | *... 12994 more positive ...* | | +0.424 | geneva |
| *... 16774 more negative ...* | | *... 14512 more negative ...* | | *... 12074 more positive ...* | |
| -0.475 | organization | -0.472 | jesus | *... 15432 more negative ...* | |
| -0.480 | christ | -0.506 | writes | -0.463 | tin |
| -0.548 | lines | -0.539 | okcforum | -0.549 | software |
| -0.554 | thanks | -0.584 | keith | -0.550 | newsreader |
| -0.554 | christians | -0.606 | church | -0.566 | article |
| -0.591 | graphics | -0.642 | christian | -0.793 | posting |
| -0.764 | rutgers | -0.674 | bible | -0.904 | graphics |
| -0.844 | subject | -0.754 | people | -0.960 | nntp |
| -0.901 | <BIAS> | -0.822 | god | -1.013 | host |

Shap: `https://github.com/slundberg/shap`

# Логистическая регрессия - плюсы

- Легко интерпретируема “на пальцах”
- Дает ответ на вопрос “почему ответ 0”/”что сделать чтобы ответ стал 1”
- Быстрое предсказание
- Просто реализуется на любом языке, многие корпоративные пакеты ее содержат
- Можно быстро найти зависимые признаки (научиться двумя признаками предсказывать третий)

# Логистическая регрессия - минусы

- Требует подготовки данных
- Как правило, не лучший результат
- В случае линейной регрессии нужен постпроцессинг (отрицательные продажи)