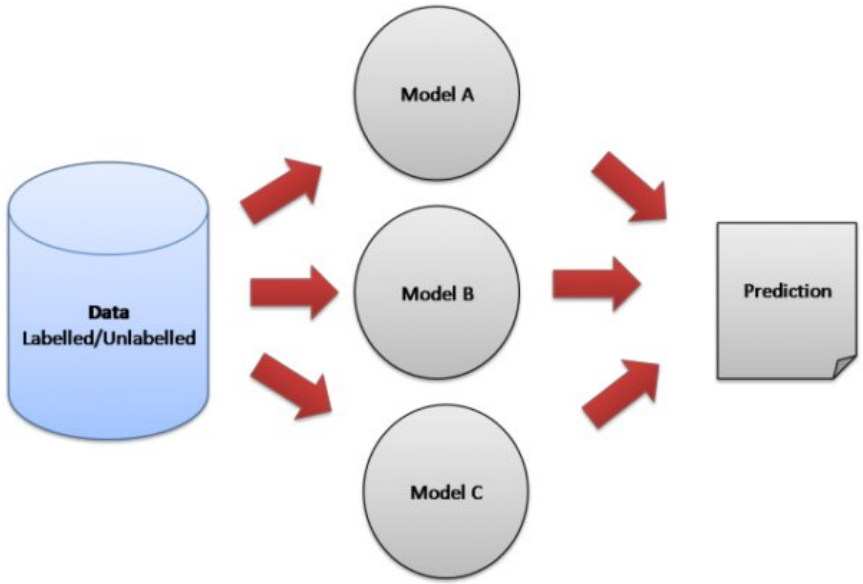
Композиции алгоритмов

Виктор Китов

victorkitov.github.io

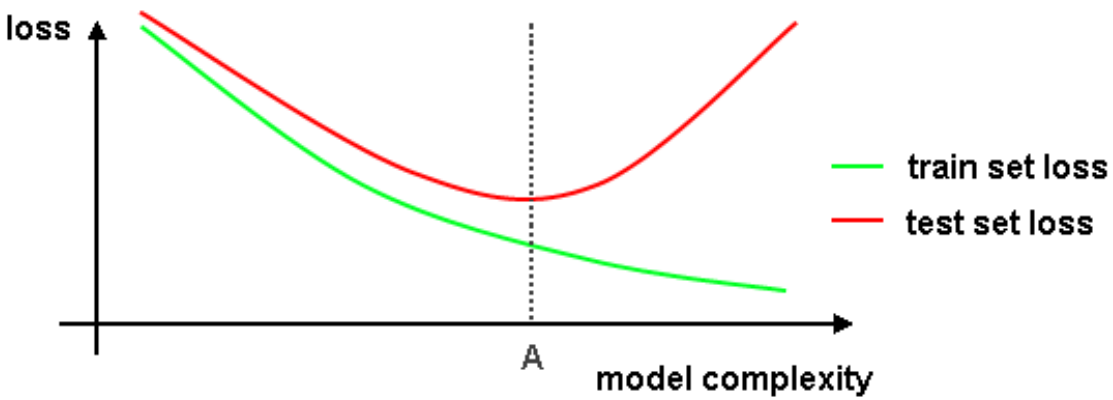


Курс поддержан фондом 'Интеллект'

Победитель конкурса VK среди курсов по IT

Содержание

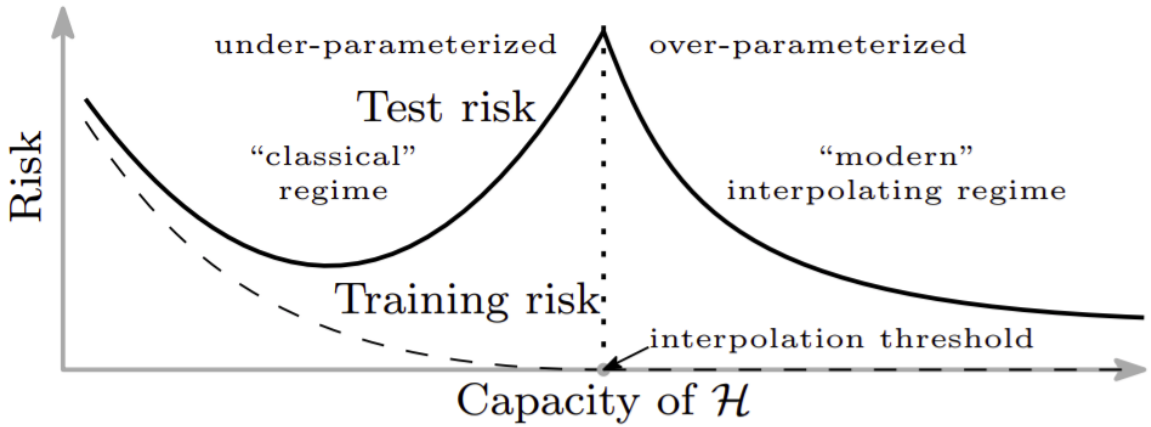
Средние потери в зависимости от сложности модели



Комментарии:

- ожидаемые потери на тестовой выборке выше потерь на обучающей.
- слева от A: модель слишком простая, недообучение.
- справа от A: модель слишком сложная, переобучение

Дальнейшее усложнение модели



- Некоторые эмпирические наблюдения свидетельствуют, что для слишком сложных моделей (многослойные нейросети) качество выше ожиданий.
- Феномен double descent risk curve[1].

[1] Belkin, M., Hsu, D., Ma, S., & Mandal, S. (2019). Reconciling modern machine-learning practice and the classical bias–variance trade-off.

Разложение на смещение и разброс

- Распределение реальных данных $y = f(x) + \varepsilon$
  - шум не зависит от x и старых наблюдений
  - хотим оценить $f(x)$
- Зависимость оценивается по $(X, Y) = \{(x_n, y_n),\ n = 1, 2...N\}$.
- Восстановленная зависимость $\widehat{f}(x)$.
- x - фикс. объект для прогноза.
- Шум ε не зависит от X, Y, $\mathbb{E}\varepsilon = 0$

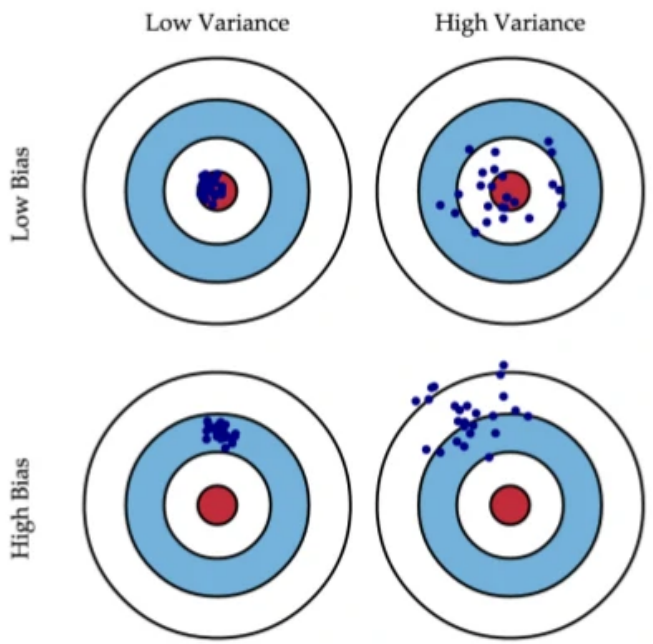
Разложение на смещение и разброс (bias-variance decomposition)

$$\mathbb{E}_{X,Y,\varepsilon}\{[\widehat{f}(x) - y(x)]^2\} = \left(\mathbb{E}_{X,Y}\{\widehat{f}(x)\} - f(x)\right)^2 + \mathbb{E}_{X,Y}\left\{[\widehat{f}(x) - \mathbb{E}_{X,Y}\widehat{f}(x)]^2\right\} + \mathbb{E}\varepsilon^2$$

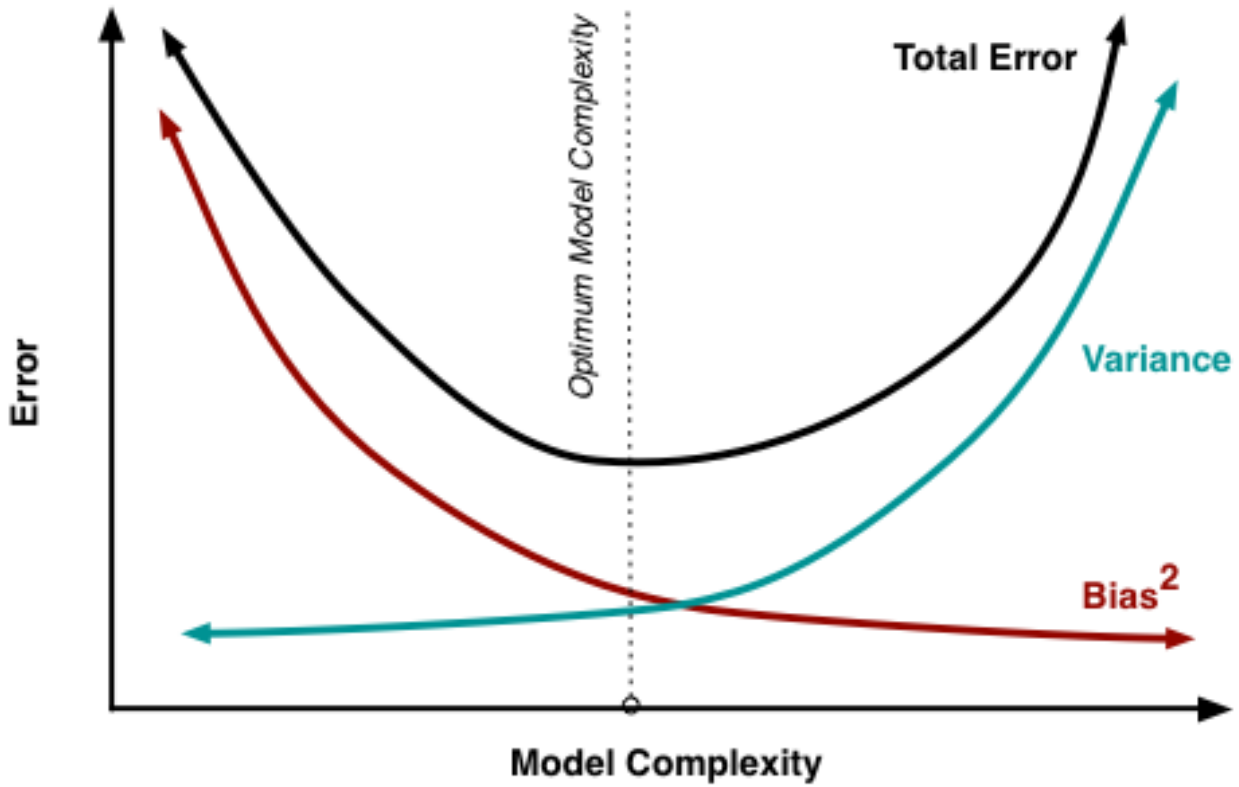
- Интуиция:
  MSE = смещение^2 + дисперсия + неснижаемая ошибка

Интуиция разложения

$$\mathbb{E}_{X,Y,\varepsilon}\{[\widehat{f}(x) - y(x)]^2\} = \left(\mathbb{E}_{X,Y}\{\widehat{f}(x)\} - f(x)\right)^2 + \mathbb{E}_{X,Y}\left\{[\widehat{f}(x) - \mathbb{E}_{X,Y}\widehat{f}(x)]^2\right\} + \mathbb{E}\varepsilon^2$$



Средние потери в зависимости от сложности модели



Доказательство разложения

Обозначим для краткости $f = f(x)$, $\widehat{f} = \widehat{f}(x)$, $\mathbb{E} = \mathbb{E}_{X,Y,\varepsilon}$.

Доказательство разложения

Обозначим для краткости $f = f(x)$, $\widehat{f} = \widehat{f}(x)$, $\mathbb{E} = \mathbb{E}_{X,Y,\varepsilon}$.

$$\begin{aligned}\mathbb{E}\left(\widehat{f}-f\right)^2 &= \mathbb{E}\left(\widehat{f}-\mathbb{E}\widehat{f}+\mathbb{E}\widehat{f}-f\right)^2 = \mathbb{E}\left(\widehat{f}-\mathbb{E}\widehat{f}\right)^2 + \left(\mathbb{E}\widehat{f}-f\right)^2 \\ &+ 2\mathbb{E}\left[(\widehat{f}-\mathbb{E}\widehat{f})(\mathbb{E}\widehat{f}-f)\right] \\ &= \mathbb{E}\left(\widehat{f}-\mathbb{E}\widehat{f}\right)^2 + \left(\mathbb{E}\widehat{f}-f\right)^2\end{aligned}$$

т.к. $(\mathbb{E}\widehat{f}-f)$ - константа относительно X, Y,
$\mathbb{E}\left[(\widehat{f}-\mathbb{E}\widehat{f})(\mathbb{E}\widehat{f}-f)\right] = (\mathbb{E}\widehat{f}-f)\mathbb{E}(\widehat{f}-\mathbb{E}\widehat{f}) = 0.$

$$\begin{aligned}\mathbb{E}\left(\widehat{f}-y\right)^2 &= \mathbb{E}\left(\widehat{f}-f-\varepsilon\right)^2 = \mathbb{E}\left(\widehat{f}-f\right)^2 + \mathbb{E}\varepsilon^2 - 2\mathbb{E}\left[(\widehat{f}-f)\varepsilon\right] \\ &= \mathbb{E}\left(\widehat{f}-\mathbb{E}\widehat{f}\right)^2 + \left(\mathbb{E}\widehat{f}-f\right)^2 + \mathbb{E}\varepsilon^2\end{aligned}$$

$\mathbb{E}\left[(\widehat{f}-f)\varepsilon\right] = \mathbb{E}\left[(\widehat{f}-f)\right]\mathbb{E}\varepsilon = 0$, поскольку ε не зависит от X, Y.

Содержание

Композиции алгоритмов

- Композиция алгоритмов (ансамбль моделей, ensemble learning):

$$\widehat{y}(x) = G(f_1(x), ...f_M(x))$$

  - $f_1(x), ...f_M(x)$ - базовые модели=признаки для $G(\cdot)$
  - $G(\cdot)$ - агрегирующая модель, мета-модель
- Используется в
  - обучении с учителем (регрессия, классификация)
  - без учителя (кластеризация)

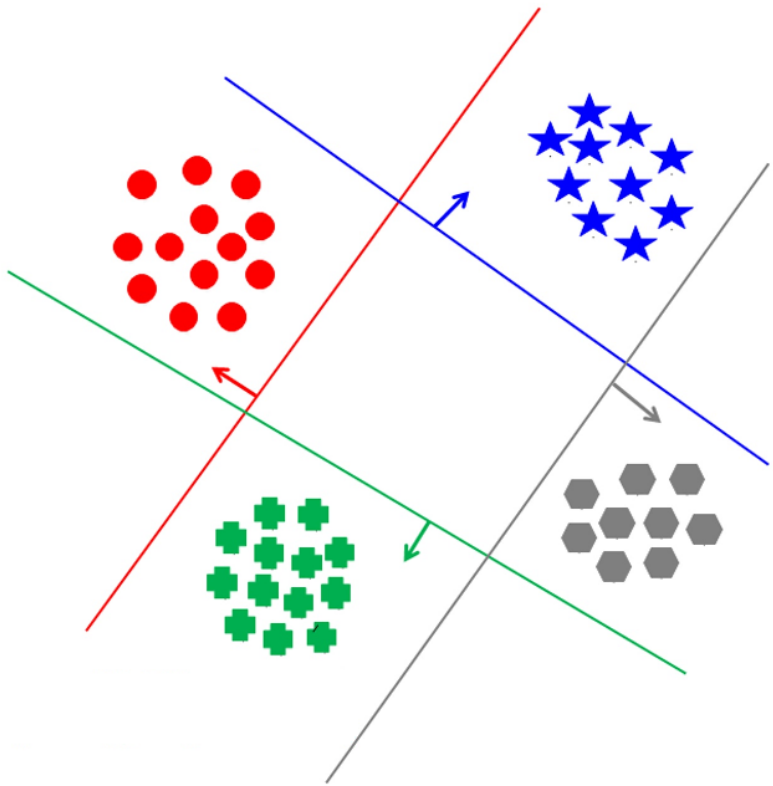
Мотивация композиций

Мотивация:

- борьба с переобучением $f_1(x), ...f_M(x)$: простая $G(\cdot)$
- борьба с недообучением $f_1(x), ...f_M(x)$: сложная $G(\cdot)$
- каждая $f_1(x), ...f_M(x)$ отвечает за свою область признакового пространства (mixture of experts)
  - $G(\cdot)$ назначает одного из экспертов
- построение $\widehat{y}(x)$ декомпозируется на решение подзадач $f_1(x), ...f_M(x)$
- ускорение обучения
  - например, усреднение ядерных SVM на подвыборках

Многоклассовая классификация

Многоклассовая классификация бинарными классификаторами (один против всех, один против одного, коды, исправляющие ошибки):



Последовательное решение, признаки разной природы

Последовательное решение

- Разделим классы: 1,2,"3+4"
  - если "3+4", применим модель, разделяющую 3 от 4.
- Прогнозирование стоимости квартир:
  - определяем тип покупки: для жилья/для инвестиций
  - для жилья: комфорт, индивидуальные вкусы и т.д.
  - для инвестиций: обменные курсы, процент по вкладам, рост рынка акций и т.д.
- Определение людей по фото:
  - определяем ракурс: фас/профиль
  - одна модель определяет людей по фото в фас
  - другая определяет людей по фото в профиль

Признаки разной природы

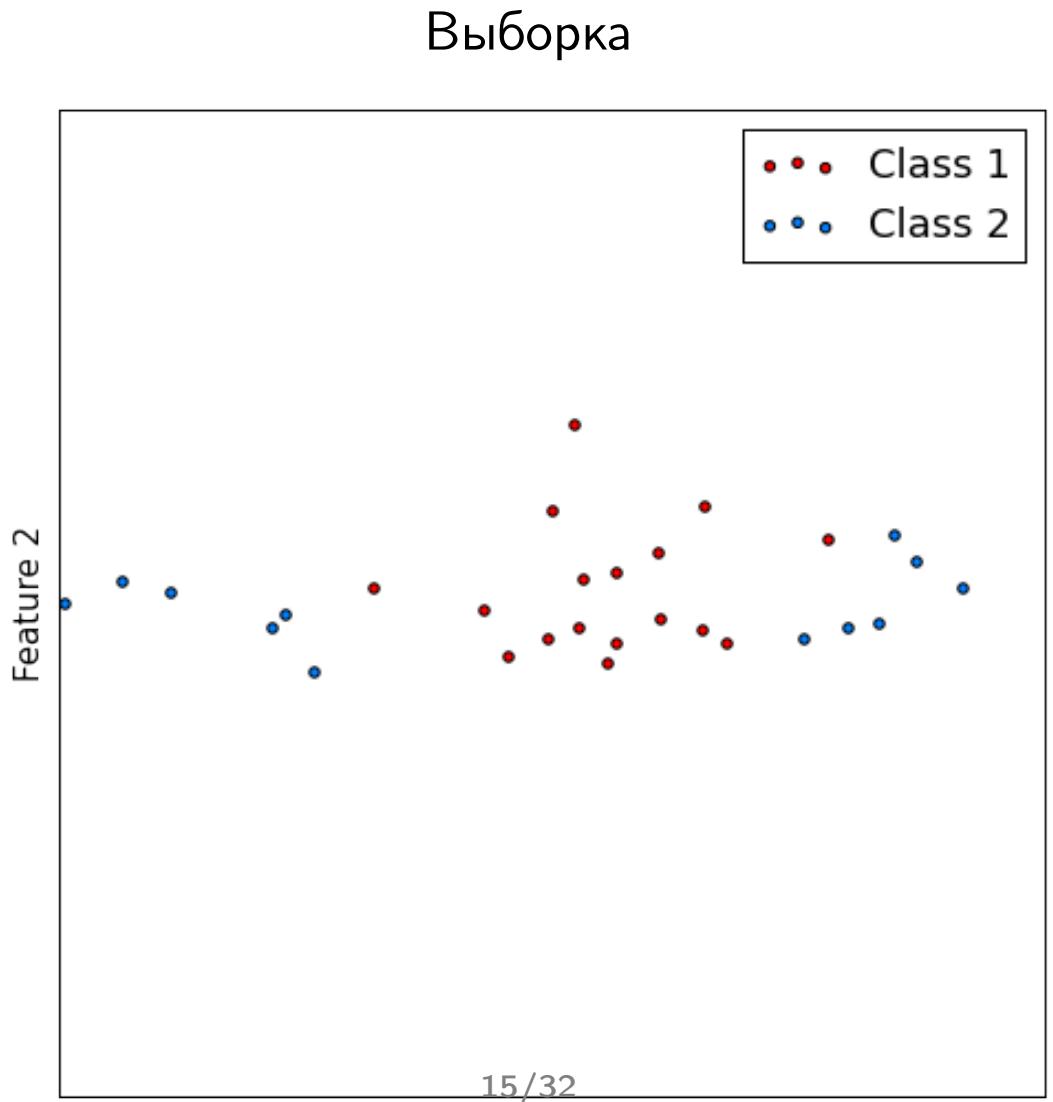
- Идентификация человека по разнородной информации: по голосу, по лицу, по поведению, и т.д.

Борьба с переобучением

- Предположим $f_1(x), ...f_M(x)$ - слишком простые модели.
- Можем повысить сложность, применяя сложную мета-модель:

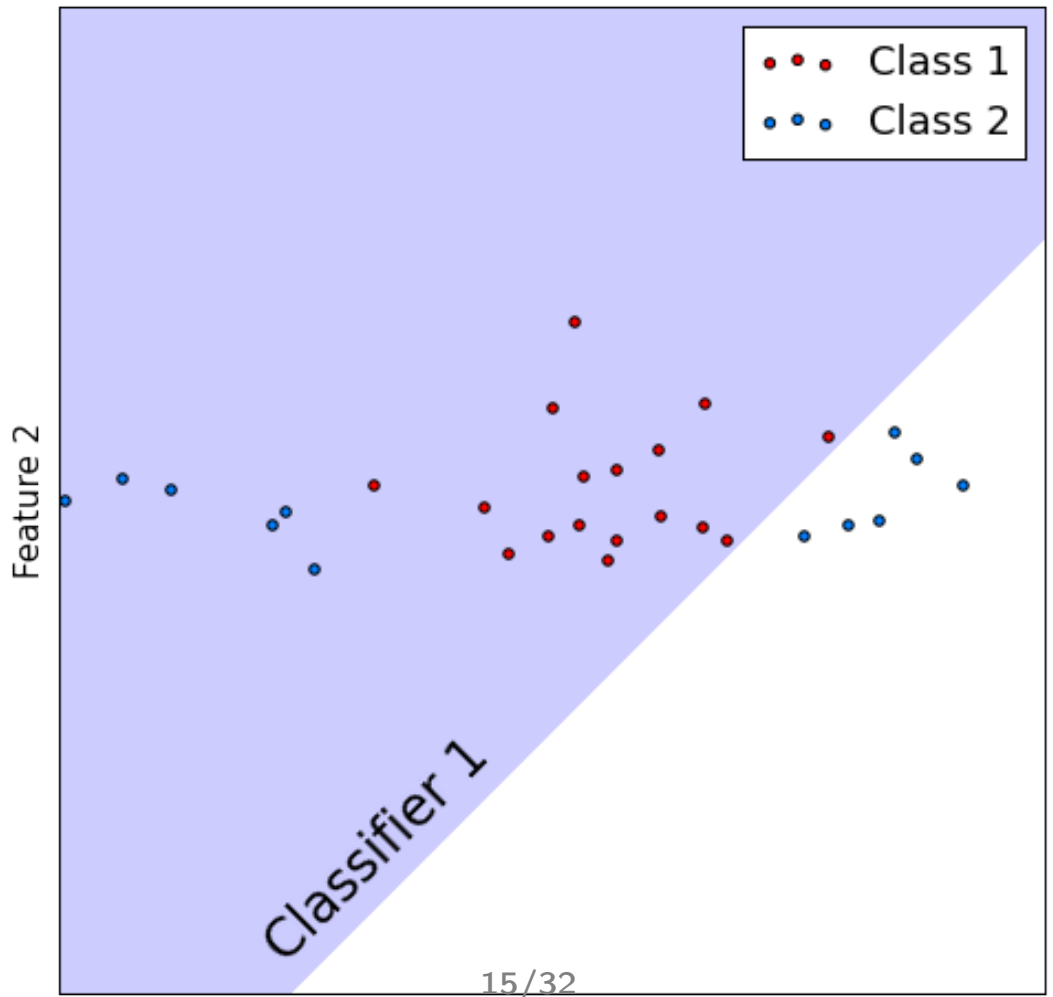
$$\widehat{y}(x) = G(f_1(x), ...f_M(x))$$

Пример



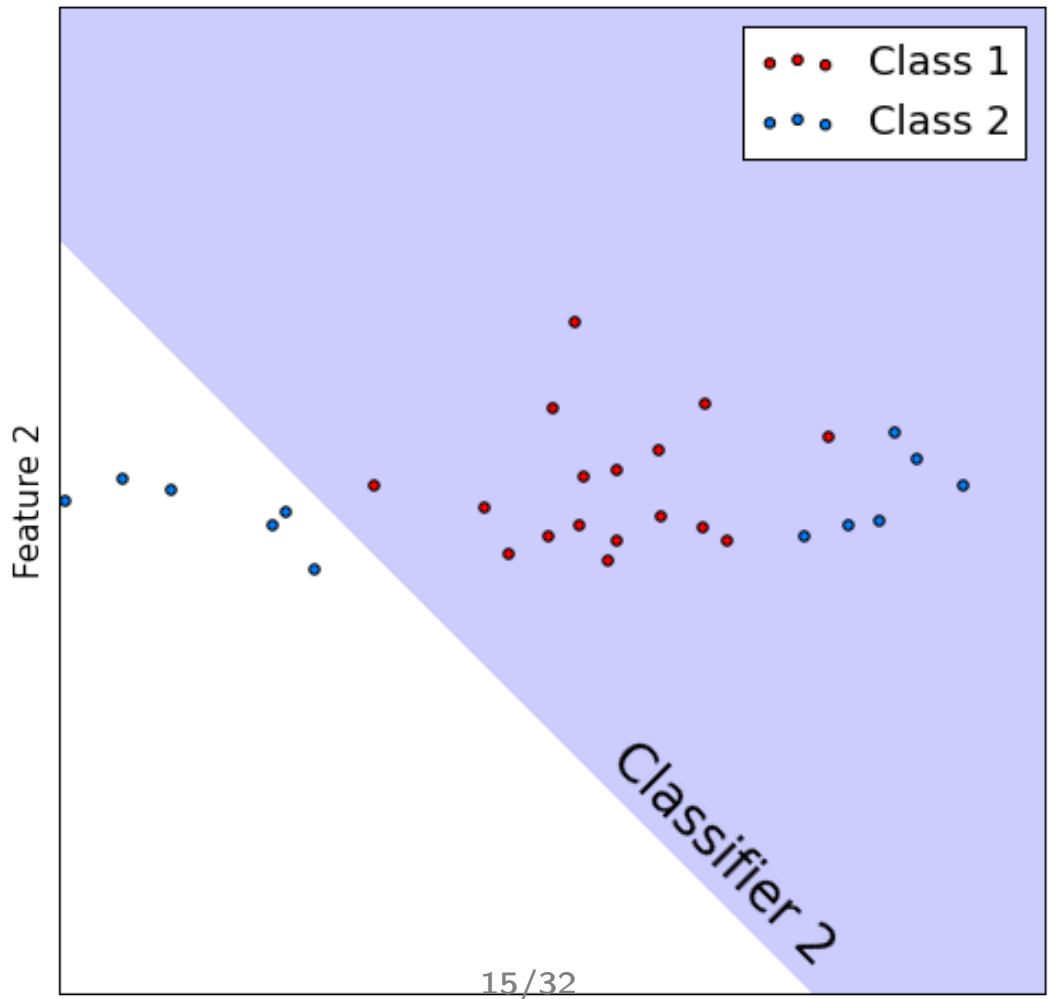
Пример

Классификатор 1



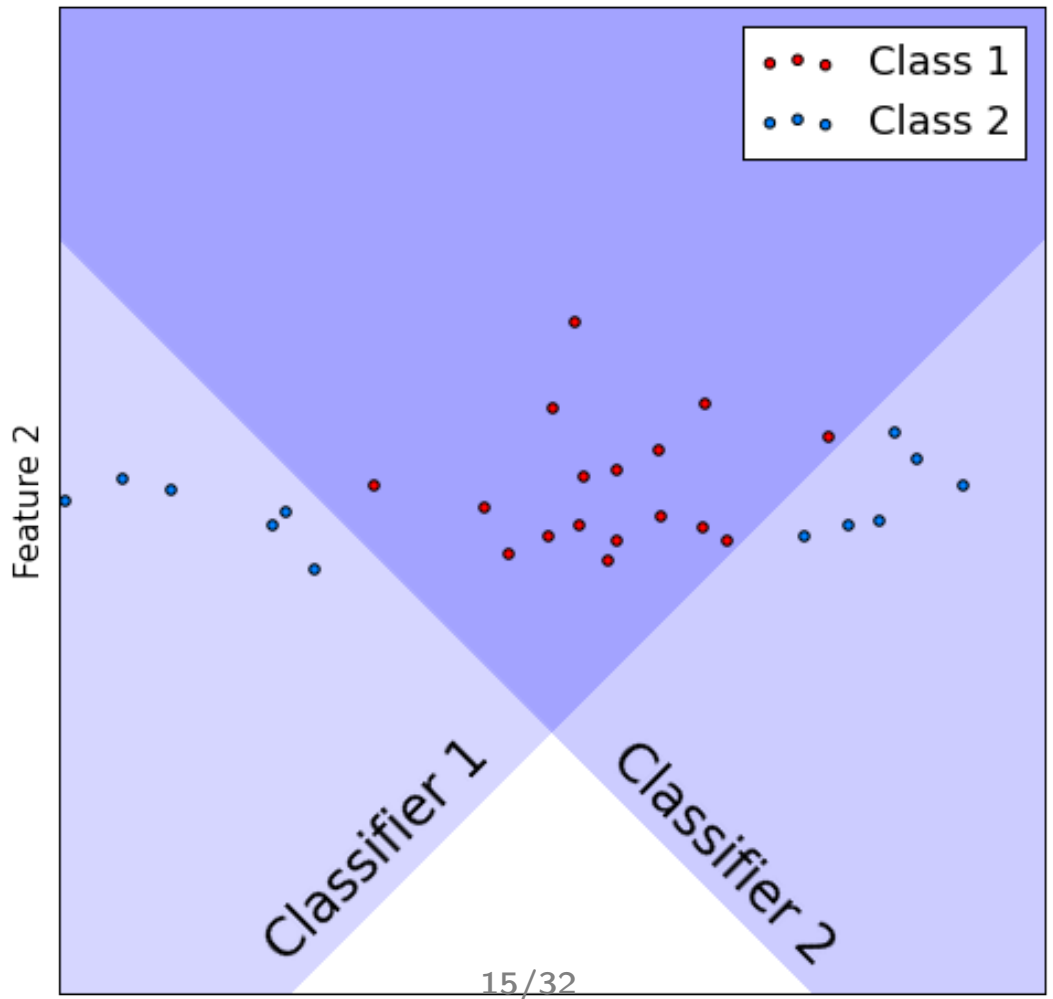
Пример

Классификатор 2



Пример

(Классификатор 1) AND (классификатор 2)



Борьба с переобучением

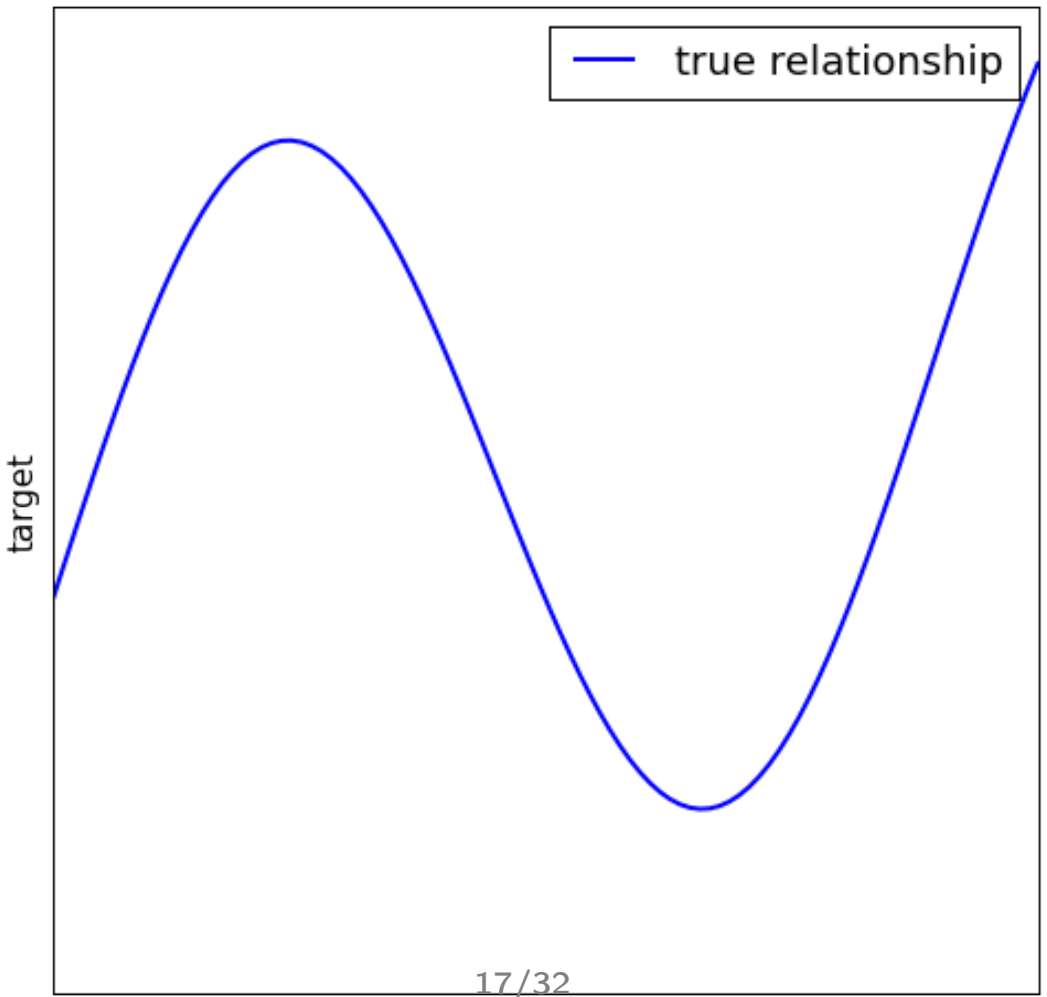
- $f_1(x), ...f_M(x)$ слишком сложные (переобученные модели)
  - решающие деревья большой глубины на разных подвыборках
  - глубокие нейросети
    - обученные из разных начальных приближений
    - разной архитектуры
- Регрессия: сделаем устойчивый прогноз за счет усреднения

$$\widehat{y}(x) = \frac{1}{M}\sum_{m=1}^{M} f_m(x)$$

- Классификация: прогноз=самый частый класс из $\{f_1(x), ...f_M(x)\}$.

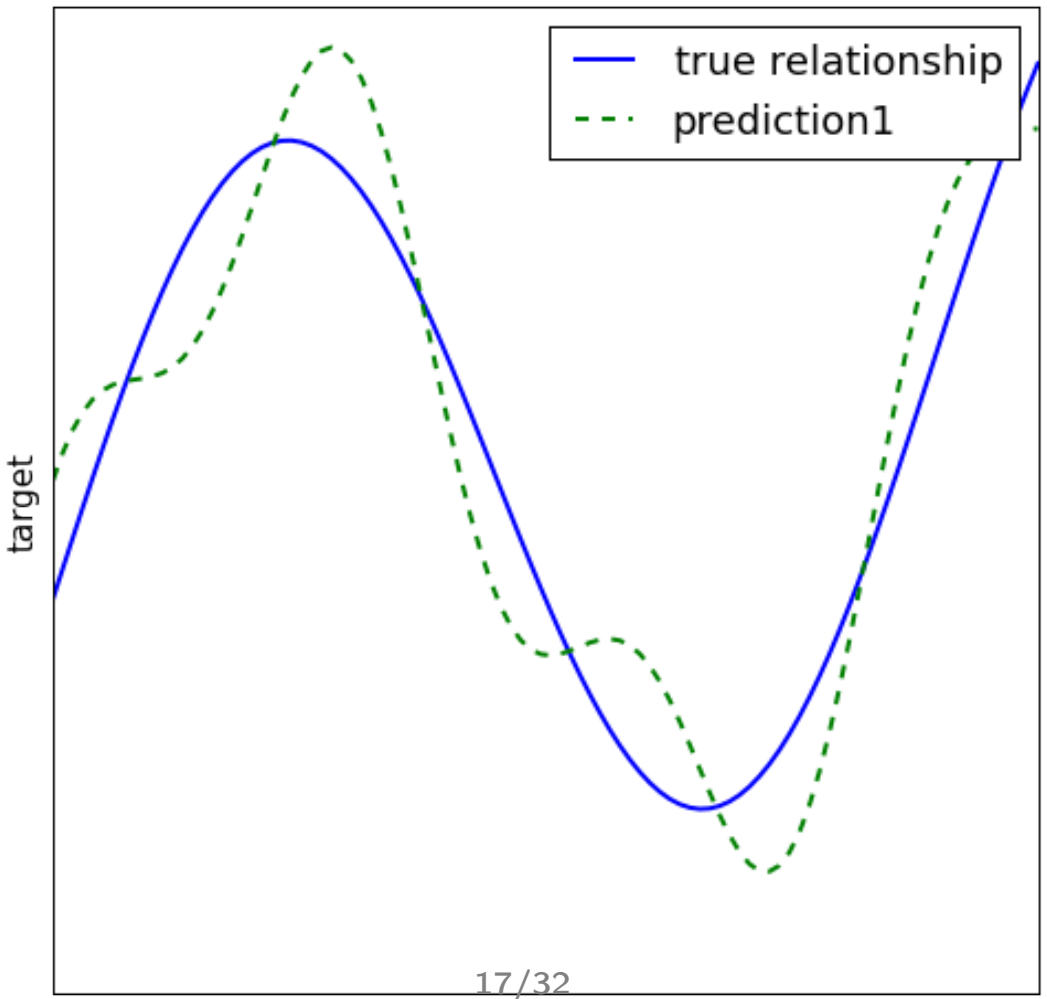
Регрессия: переобучение

Целевая зависимость



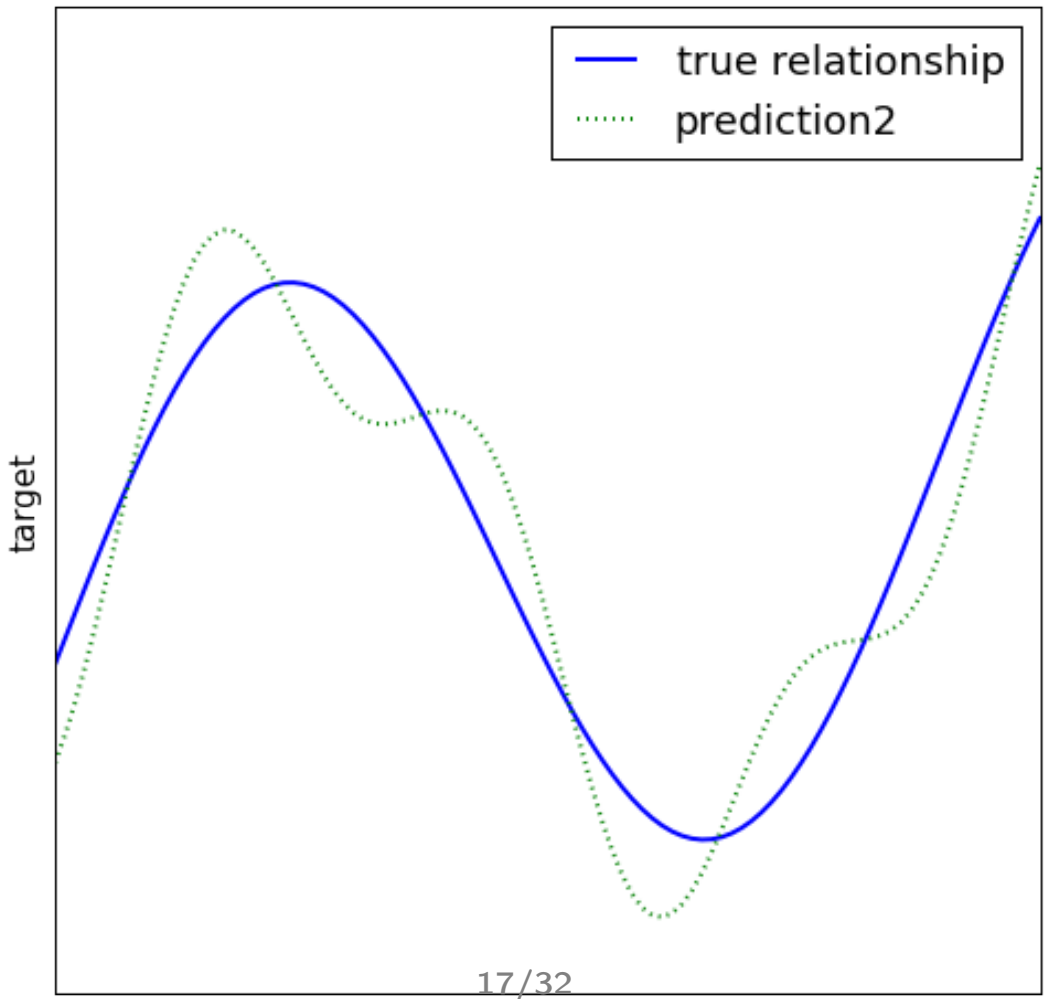
Регрессия: переобучение

Модель 1.



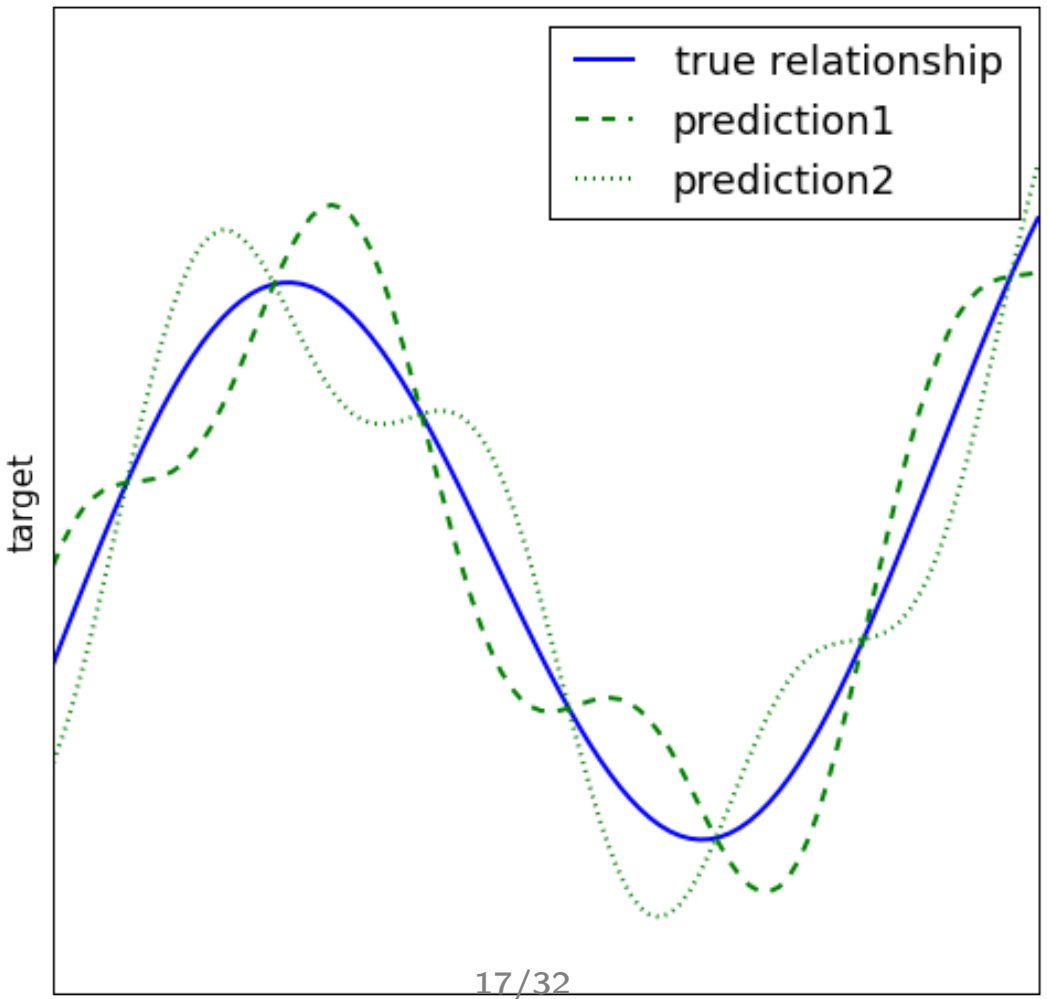
Регрессия: переобучение

Модель 2.



Регрессия: переобучение

Среднее модели 1 и 2 дает более точный прогноз.



Усреднение ошибок

- Рассмотрим задачу регрессии с усредняющим ансамблем:

$$\widehat{y}(x) = \frac{1}{M}\sum_{m=1}^{M} f_m(x)$$

- Пусть $\varepsilon_1, ...\varepsilon_M$ - ошибки прогнозирования базовыми моделями $f_1(x), ...f_M(x)$ со средним ноль, дисперсией $\mathbb{D}\varepsilon_i = \sigma^2$ и пусть эти ошибки имеют попарную корреляцию ρ (соответственно, ковариация будет $\mathbb{E}\varepsilon_i\varepsilon_j = \rho\sigma^2$)
- Тогда ожидаемый квадрат ошибки отдельных моделей будет

$$\mathbb{E}\left\{(f_i(x) - y(x))^2\right\} = \mathbb{E}\varepsilon_i^2 = \sigma^2$$

Усреднение ошибок

- А ожидаемый квадрат ошибки усредняющего ансамбля:

$$\mathbb{E}\left\{(\widehat{y}(x)-y(x))^2\right\}=\mathbb{E}\left\{\left(\frac{\sum_{m=1}^{M}(f_m(x)-y(x))}{M}\right)^2\right\}$$

$$=\frac{1}{M^2}\mathbb{E}\left\{\left(\sum_{m=1}^{M}\varepsilon_m\right)^2\right\}=\frac{1}{M^2}\mathbb{E}\left\{\sum_{m=1}^{M}\varepsilon_m^2+\sum_{i\neq j}\varepsilon_i\varepsilon_j\right\}$$

$$=\frac{M}{M^2}\sigma^2+\frac{M^2-M}{M^2}\rho\sigma^2=\frac{\sigma^2}{M}+\left(1-\frac{1}{M}\right)\rho\sigma^2$$

- При $\rho=0$ квадрат ошибки в M раз меньше!
- Может быть еще меньше при $\rho<0$.

Голосование большинства (против переобучения)

- Рассмотрим M классификаторов $f_1(x), ...f_M(x)$.
- Пусть $p(f_m(x) \neq y) = p < 0.5 \, \forall m$
- Пусть модели ошибаются независимо друг от друга.
- Пусть $G(x)$ - выбор самого частого класса.
- Тогда $p(G(x) \neq y) \to 0$ при $M \to \infty$[2]

[2] Докажите это утверждение.

Взвешенное усреднение (против переобучения)

- **Разложение неоднозначности** (ambiguity decomposition): пусть (x, y) прогнозируется с помощью регрессии $G(x) = \sum_{m=1}^{M} w_m f_m(x)$, $w_m \geq 0$, $\sum_m w_m = 1$. Тогда

$$\underbrace{(G(x) - y)^2}_{\text{ensemble error}} = \underbrace{\sum_m w_m \left(f_m(x) - y\right)^2}_{\text{base learner error}} - \underbrace{\sum_m w_m \left(f_m(x) - G(x)\right)^2}_{\text{ambiguity}}$$

- Композиция дает точные прогнозы когда:
  - $f_m(x)$ достаточно точны
  - индивидуальные прогнозы $\{f_m(x)\}_m$ сильно различаются
    - поэтому полезно усреднять по разным моделям

Доказательство разложения неоднозначности

Доказательство:

$$\sum_m w_m \left(f_m(x) - G(x)\right)^2 = \sum_m w_m \left(f_m(x) - y + y - G(x)\right)^2$$

$$= \sum_m w_m \left(f_m(x) - y\right)^2 + \sum_m w_m \left(y - G(x)\right)^2 + 2\sum_m w_m \left(f_m(x) - y\right)\left(y - G(x)\right)$$

$$= \sum_m w_m \left(f_m(x) - y\right)^2 + \left(G(x) - y\right)^2 + 2\left(y - G(x)\right)\sum_m w_m \left(f_m(x) - y\right)$$

$$= \sum_m w_m \left(f_m(x) - y\right)^2 + \left(G(x) - y\right)^2 + 2\left(y - G(x)\right)\left(G(x) - y\right)$$

$$= \sum_m w_m \left(f_m(x) - y\right)^2 + \left(G(x) - y\right)^2 - 2\left(G(x) - y\right)^2$$

$$= \sum_m w_m \left(f_m(x) - y\right)^2 - \left(G(x) - y\right)^2$$

Выпуклые потери

Выпуклые потери поощряют использование взвешенных прогнозов вместо индивидуальных.

- Рассмотрим регрессию с выпуклой ф-цией потерь $\mathcal{L}(\widehat{y}-y)$.
- Учитываем $f_1(x),...f_M(x)$ с весами $w_1,...w_M$.
- Для фикс. x рассмотрим 2 стратегии прогнозирования:

1. сэмплировать $m \sim Categorical(w_1,...w_M)$, $\widehat{y}(x)=f_m(x)$.
2. усреднять $\widehat{y}(x)=\sum_{m=1}^{M} w_m f_m(x)$

Какая стратегия в среднем по m будет давать меньшие ожидаемые потери?

Содержание

Регрессия

$$\widehat{y}(x) = \frac{1}{M}\sum_{m=1}^{M} f_m(x)$$

$$\widehat{y}(x) = \frac{1}{\sum_{m=1}^{M} w_m}\sum_{m=1}^{M} w_m f_m(x)$$

- Взвешенное усреднение лучше, если модели сильно отличаются по точности.
- Веса $w_1 \geq 0, ...w_M \geq 0$ нужно настраивать на отдельной выборке (не той, на которой обучали $f_1(x), ...f_M(x)$)
- Альтернатива: медиана/взвешенная медиана

Классификаторы выдают **вероятности**

- Пусть $p_y^m(x)$ - вероятность класса y по мнению классификатора m.
- Равномерная агрегация:

$$p_c(x) = \frac{1}{M}\sum_{m=1}^{M} p_c^m(x)$$

- Взвешенная агрегация:

$$p_c(x) = \frac{1}{\sum_{m=1}^{M} w_m}\sum_{m=1}^{M} w_m p_c^m(x)$$

- Взвешенное усреднение лучше, если модели сильно отличаются по точности.
- Веса $w_1 \geq 0, ...w_M \geq 0$ нужно настраивать на отдельной выборке (не той, на которой обучали $f_1(x), ...f_M(x)$)

Классификаторы выдают **метки классов**

- Голосование по большинству (majority vote)
  - возможен взвешенный учет классификаторов
- Бинарная классификация: $\widehat{y} = +1$ <=>
  - $\geq k$ классификаторов выдают +1 (k-out-of-N)
    - возможен взвешенный учет классификаторов
  - все классификаторы выдают +1 (AND, N-out-of-N)
  - хотя бы один выдает +1 (OR, 1-out-of-N)

Классификаторы выдают **рейтинги**

- Пусть $g_y^m(x)$ - рейтинг класса y в модели m.
- Проблема: рейтинги несравнимы для разных моделей.
- Решение (Brier scores):
  1. Стандартизованный рейтинг - #классов ниже по рейтингу:

$$s_y^m(x) = \sum_{i \neq y} \mathbb{I}[g_y^m(x) > g_i^m(x)]$$

$$s_y^m(x) \in \{1, 2, \dots C-1\}$$

  2. Предскажем класс с максимальным усредненным по моделям рейтингом:

$$\widehat{y}(x) = \arg\max_y \frac{1}{M} \sum_{m=1}^{M} s_y^m(x)$$

Использование голосования (регрессия)

```
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import VotingRegressor
from sklearn.metrics import mean_absolute_error

X_train, X_test, Y_train, Y_test =
    get_demo_classification_data()
knn = KNeighborsRegressor(n_neighbors=100)
tree_model = DecisionTreeRegressor()

ensemble = VotingRegressor(     # усредняющий ансамбль
    estimators=[('K-NN', knn), ('DT', tree_model)],
    weights=[0.5,0.5])
ensemble.fit(X_train, Y_train) # обучение базовых м-лей
Y_hat = ensemble.predict(X_test) # построение прогнозов
print(f'Средний модуль ошибки (MAE): {
    mean_absolute_error(Y_test, Y_hat):.2f}')
```

Больше информации. Полный код.

Использование голосования (классификация) 1

```
rom sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import VotingClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import brier_score_loss

X_train, X_test, Y_train, Y_test =
    get_demo_classification_data()
# инициализация базовых моделей
log_model = LogisticRegression()
tree_model = DecisionTreeClassifier()
```

Использование голосования (классификация) 2

```
# Голосование по большинству (для меток классов)
ensemble = VotingClassifier(
    estimators=[('logistic regression', log_model),
                ('decision tree', tree_model)],
    voting='hard', weights=[0.5,0.5])
# обучение базовых моделей ансамбля:
ensemble.fit(X_train, Y_train)
# построение прогнозов:
Y_hat = ensemble.predict(X_test)
print(f'Точность VotingClassifier: \
{100*accuracy_score(Y_test, Y_hat):.1f}%')
```

Использование голосования (классификация) 3

```
# Ансамбль, усредняющий вероятности классов
ensemble = VotingClassifier(
    estimators=[('logistic regression', log_model), ('
     decision tree', tree_model)],
    voting='soft', weights=[0.5,0.5])
# обучение базовых моделей ансамбля:
ensemble.fit(X_train, Y_train)
# построение прогнозов:
Y_hat = ensemble.predict(X_test)
print(f'Точность VotingClassifier: \
{100*accuracy_score(Y_test, Y_hat):.1f}%')

P_hat = ensemble.predict_proba(X_test)  # вероятности
    классов
loss = brier_score_loss(Y_test, P_hat[:,1])
print(f'Мера Бриера отклонения вер-тей: {loss:.2f}')
```

Больше информации. Полный код.