



NEW
PRO
LAB

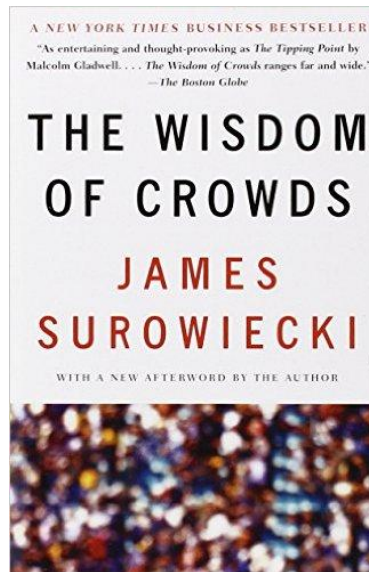
The logo consists of a red square containing the text 'NEWPROLAB' in white, stacked in three lines. The square is positioned over several overlapping horizontal bars in green, blue, and orange.

Ensemble Learning in ML

NEWPROLAB.COM

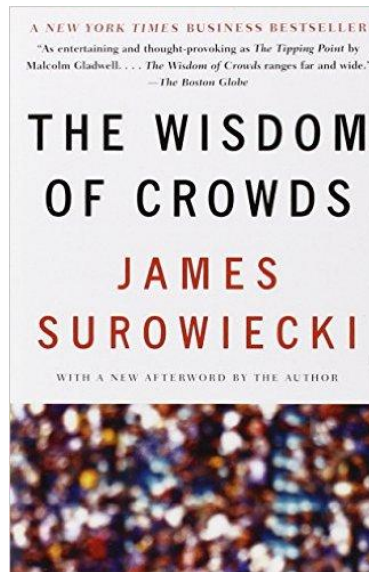
Интуиция—1

- Человек мыслит аналогиями
- В **определённых** ситуациях решения, принятые группой, превосходят по точности индивидуальные решения
- Люди **иногда** принимают решения, взвешивая мнения других
- Пример — консилиумы врачей и учёных
- Не подходит для творческих решений



Интуиция—2

- Различия в точках зрения участников
- Различия в интерпретациях, допускаются эксцентричные идеи и взгляды
- Независимость принятия решений
- Специализация участников, доменные знания
- Наличие алгоритма агрегации отдельных решений в коллективное решение



Демо: голосование

- 3 классификатора с точностью 70% каждый (“0” — ошибка)
- При большинстве голосов улучшают точность до 78%
- 5 clf. Точность: 83%

All three are correct

$$0.7 * 0.7 * 0.7 = 0.3429$$

Two are correct

$$0.7 * 0.7 * 0.3 + 0.7 * 0.3 * 0.7 + 0.3 * 0.7 * 0.7 = 0.4409$$

Two are wrong

$$0.3 * 0.3 * 0.7 + 0.3 * 0.7 * 0.3 + 0.7 * 0.3 * 0.3 = 0.189$$

All three are wrong

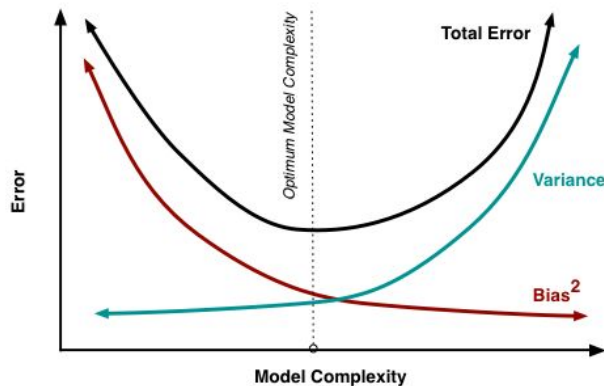
$$0.3 * 0.3 * 0.3 = 0.027$$

Bias-variance tradeoff

$$Err(x) = E[(Y - \hat{f}(x))^2]$$

$$Err(x) = \text{Bias}^2 + \text{Variance} + \text{Irreducible Error}$$

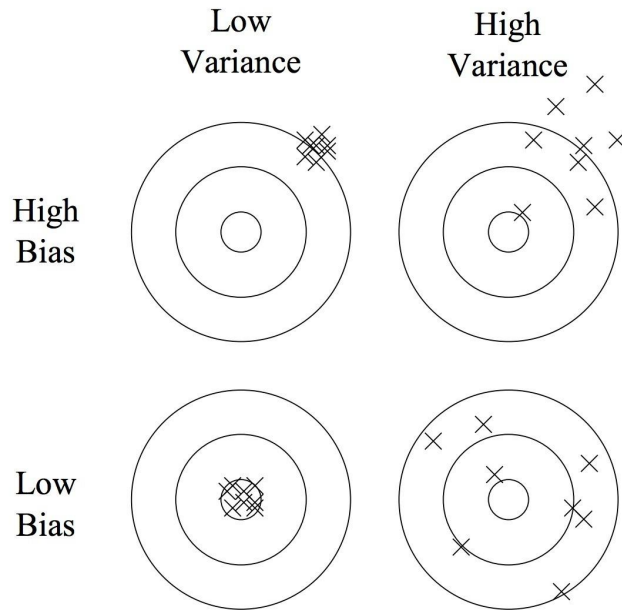
$$Err(x) = (E[\hat{f}(x)] - f(x))^2 + E[(\hat{f}(x) - E[\hat{f}(x)])^2] + \sigma_e^2$$



- Смещение — среднее отклонение прогноза модели от прогноза идеальной модели
- Разброс — дисперсия ответов модели, обученных на разных подвыборках
- Неустраняемая ошибка внутри самих данных

Bias-variance tradeoff

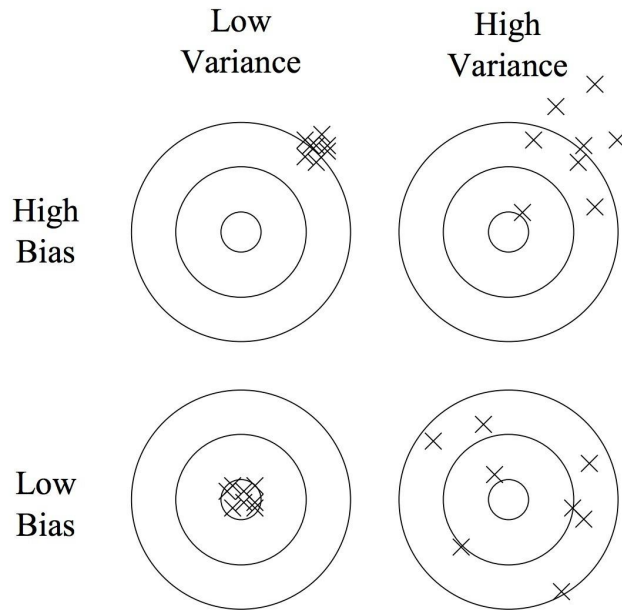
- Компромисс между качеством на train и силой модели на test
- Low bias — сложные модели => **переобучение**
- Low variance — модели, не учитывающие сложность данных => **недообучение**



Bias-variance tradeoff

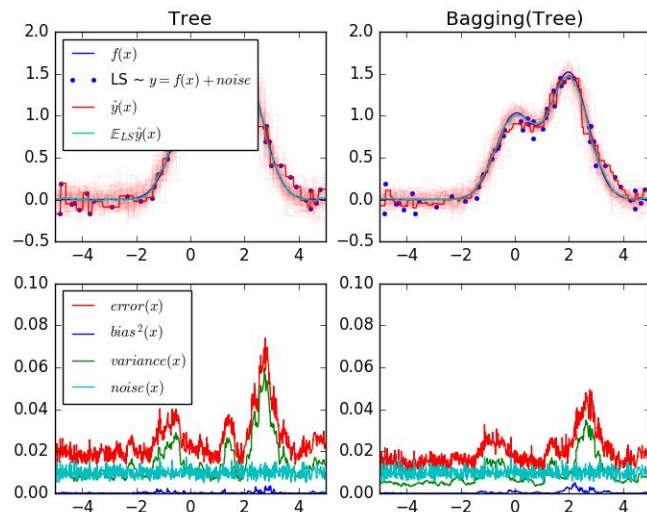
Резюме:

- Ошибка модели раскладывается на смещение и разброс + шум в данных
- Корреляция между предсказаниями моделей играет важную (и негативную) роль
- Разброс можно уменьшить, снизив корреляцию между моделями в ансамбле



Ансамбли моделей

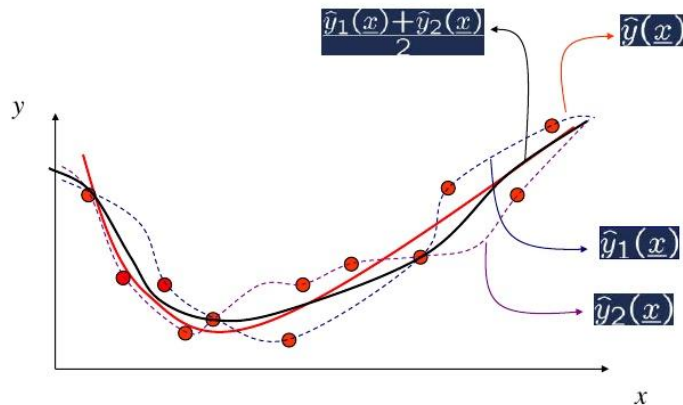
- Помогают снизить разброс моделей
- Виды:
 - Бутстрап (или subsampling) + усреднение — бэггинг
 - Использование случайных подпространств
 - Направленное построение ансамбля (бустинг)
 - Метамоделли (Wolpert (1992) Stacked Generalization)



Бэггинг–1

- Bagging: **bootstrap aggregating**
- Используем бутстрап / подмножества тренировочной выборки с **возвращением**
- Часто этого недостаточно. Тогда дополнительно используем рандомизированные подмножества признаков
- Агрегируем модели одного типа
- Каждая модель считается в ансамбле с одинаковым весом
- BaggingClassifier в sklearn

Illustration on Bagging



Бэггинг–2: Бутстрап

- Непараметрический метод.
- Позволяет оценить эмпирическую функцию распределения на основе многократной генерации псевдовыборок с возвращением (методом Монте-Карло)
- По эмпирической функции распределения можно вычислять нужные статистики: дисперсию, квантили и т.д.

Бэггинг—3: OOBЕ

- Out of Bag Error: «встроенная» в бутстрап возможность кросс-валидироваться на не попавших в выборку примерах
- В выборку не попадает $(1 - \frac{1}{\ell})^\ell$ примеров, что в пределе означает, что дерево обучается на 63% выборки, а 37% можно оставить на валидацию.

Случайный лес

- Ещё более рандомизированный вариант бэггинга
- Алгоритм:
 - Выделяем бутстрапом N случайных подвыборок размера X_n
 - Для каждой подвыборки независимо строим CART-дерево заданной глубины (либо до тех пор, пока в листе не окажется заданное число объектов)
 - При каждом сплите рандомизируем подпространство признаков
 - Объединяем в ансамбль для классификации:

$$a(x) = \text{sign} \frac{1}{N} \sum_{n=1}^N b_n(x).$$

Бустинг—1

- Концепция weak learners (базовых алгоритмов): натренировать слабые классификаторы намного проще (bias-variance tradeoff)
- Из множества weak сделать один strong learner
- Направленное построение ансамбля, в отличие от RF
- Стратегия:
 - Базовые алгоритмы строятся последовательно, один за другим
 - Каждый следующий алгоритм строится так, чтобы исправлять общую ошибку текущего ансамбля

Бустинг–2

- Пример — задача регрессии. Функция ошибки — MSE:

$$MSE(a, X) = \frac{1}{\ell} \sum_{i=1}^{\ell} (a(x_i) - y_i)^2.$$

- Инициализируем первый базовый алгоритм (b_1):

$$b_1(x) = \operatorname{argmin}_b \frac{1}{\ell} \sum_{i=1}^{\ell} (b(x_i) - y_i)^2.$$

и минимизируем MSE через GD.

- Строим второй алгоритм (b_2) так, чтобы композиция $b_1(x_i) + b_2(x_i)$ имела наименьший MSE:

$$b_2(x) = \operatorname{argmin}_b \frac{1}{\ell} \sum_{i=1}^{\ell} (b_1(x_i) + b(x_i) - y_i)^2 :$$

- Повторяем процесс до нужной ошибки $b_1(x) + \dots + b_N(x)$

Градиентный бустинг

- Определить функцию потерь $L(y, z)$; y — истинный ответ, z — прогноз. Для регрессии: $L(y, z) = (y - z)^2$
- Инициализация базового алгоритма (регр.): $b_0(x) = \frac{1}{\ell} \sum_{i=1}^{\ell} y_i$
- Постановка задачи на каждом шаге:

$$\sum_{i=1}^{\ell} L(y_i, a_{N-1}(x_i) + b(x_i)) \rightarrow \min_b.$$

- Оптимизируем вектор сдвигов s :

$$F(s) = \sum_{i=1}^{\ell} L(y_i, a_{N-1}(x_i) + s_i) \rightarrow \min_s, \quad s = -\nabla F = \begin{pmatrix} -L'_z(y_1, a_{N-1}(x_1)), \\ \dots \\ -L'_z(y_{\ell}, a_{N-1}(x_{\ell})) \end{pmatrix}$$

- n -базовый алгоритм: $b_n(x) = \operatorname{argmin}_b \frac{1}{\ell} \sum_{i=1}^{\ell} (b(x_i) - s_i)^2$

- Добавление в композицию: $a_n(x) = \sum_{m=1}^n b_m(x)$

Метамодели

Netflix Prize

COMPLETED[Home](#) [Rules](#) [Leaderboard](#) [Update](#) [Download](#)

Leaderboard

Showing Test Score. [Click here to show quiz score](#)Display top leaders.

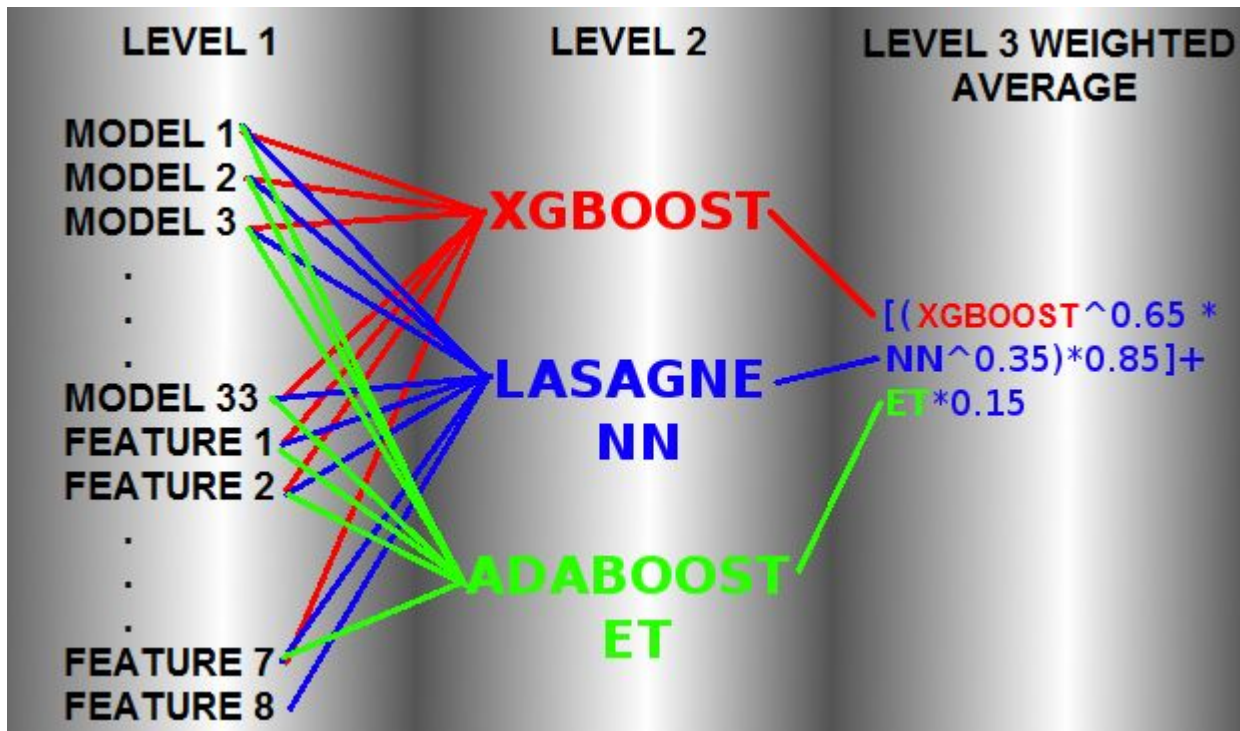
| Rank | Team Name | Best Test Score | % Improvement | Best Submit Time |
|---|---|-----------------|---------------|---------------------|
| Grand Prize - RMSE = 0.8567 - Winning Team: BellKor's Pragmatic Chaos | | | | |
| 1 | BellKor's Pragmatic Chaos | 0.8567 | 10.06 | 2009-07-26 18:18:28 |
| 2 | The Ensemble | 0.8567 | 10.06 | 2009-07-26 18:38:22 |
| 3 | Grand Prize Team | 0.8582 | 9.90 | 2009-07-10 21:24:40 |
| 4 | Opera Solutions and Vandelay United | 0.8588 | 9.84 | 2009-07-10 01:12:31 |
| 5 | Vandelay Industries I | 0.8591 | 9.81 | 2009-07-10 00:32:20 |
| 6 | PragmaticTheory | 0.8594 | 9.77 | 2009-06-24 12:06:56 |
| 7 | BellKor in BigChaos | 0.8601 | 9.70 | 2009-05-13 08:14:09 |
| 8 | Dace | 0.8612 | 9.59 | 2009-07-24 17:18:43 |

“This is a truly impressive compilation and culmination of years of work, blending hundreds of predictive models to finally cross the finish line.

We evaluated some of the new methods offline but the additional accuracy gains that we measured did not seem to justify the engineering effort needed to bring them into a production environment.”

Netflix Engineers

Метамодели



Метамодели

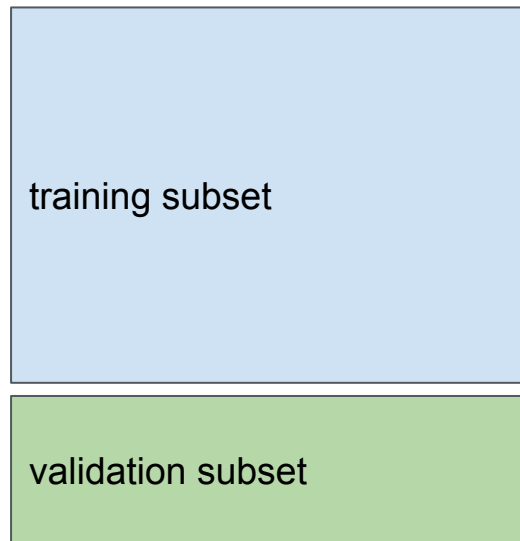
Blending

Stacking

Метамодели

Blending

- Как и стекинг, тесно связан с понятием кросс-валидации
- Большой train-датасет делится на 2 части: training и validation
- Лучшая по валидации модель делает предсказания для [1] validation subset, [2] test set
- Строим несколько моделей и используем validation subset как train для метамодели

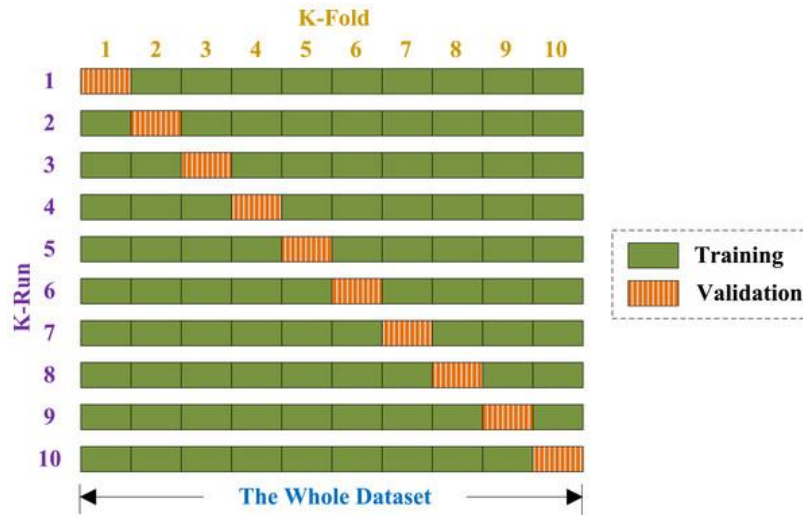


Blending

- +: Просто реализовать (можно даже в Spark RDD, используйте `pyspark.RDD.randomSplit`)
- +: Защищает от дата-лика
- +: Один валидационный сет можно использовать для любых классификаторов — удобно для командной работы
- —: validation subset не используется в обучении, этими данными придётся пожертвовать
- —: возможность переобучения на validation subset
- —: нужно проверять соответствие распределений в validation и training subsets

Stacking

- Думайте в терминах k-fold cross-validation
- В k раз сложнее для расчёта, чем простой блендинг
- Задача — сделать предсказание всего train-сета, используя k-валидационных фолдов; каждый oof-классификатор предсказывает **весь** test-сет



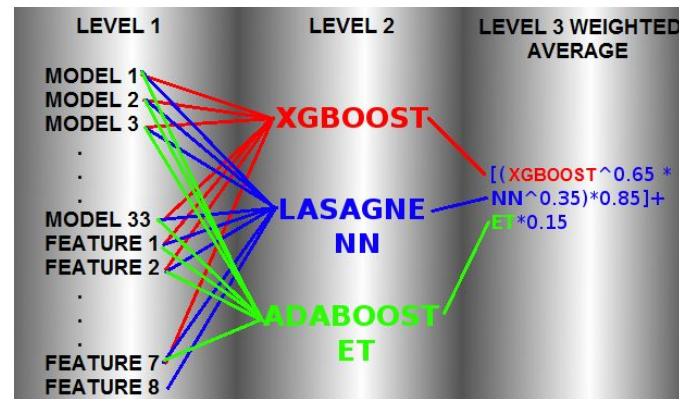
Stacking

- +: Использует все данные из train-сета => обычно даёт лучше результат, чем блендинг
- +: Можно сделать StratifiedKFoldSplit
- +: Лучше защищён от переобучения на валидации
- —: значительно тяжеловеснее, чем блендинг
- —: у всех в команде должны быть одинаковые фолды train-сета
- —: обычно включает ещё и bagged folds, что ещё **на порядок** увеличивает сложность

Уровни

- L1: уровень отдельных моделей и агрегированных фич => фичи для L2
- L2: метамоделі, обучающиеся на выходах валидации / oof L1
- L3: метамоделі, обучающиеся на выходах моделей L2

На вход метамоделям можно подавать фичи, полученные через unsupervised learning: k-means кластеры, pca components, t-sne вектора



<http://blog.kaggle.com/2015/06/09/otto-product-classification-winners-interview-2nd-place-alexander-guschin/>

<https://www.kaggle.com/c/otto-group-product-classification-challenge/discussion/14295>

