



Логика ML kNN

Политология, 3 курс
Татьяна Рогович



Что узнаем?

1. Повторим терминологию ML
2. Разберемся с логикой ML
3. Поговорим про метрики ошибки
4. Узнаем, что такое недообучение и переобучение и как с этим бороться
5. Поговорим про алгоритм kNN




Терминология

- **Example:** объект или наблюдение
- **Features, inputs:** набор независимых переменных
- **Labels, output, target:** значения целевой (зависимой) переменной.
В задачи классификация - категория объекта.
В регрессии - действительные числа.
- **Training data:** данные, на которых обучаем алгоритм.
- **Test data:** данные, на которых тестируем алгоритм и принимаем решение о его пригодности.



Терминология

- **Loss function:** функция, которая в теории статистических решений характеризует потери при неправильном принятии решений на основе наблюдаемых данных.
- **y :** вектор со значениями целевой переменной
- **X :** матрица всех значений признаков (независимых переменных)
- **x_i :** наблюдение (набор значений независимых переменных для одного объекта выборки)
- **\hat{y} с крышкой:** предсказанное моделью значение целевой переменной



Обучение алгоритма ML

- **Define / Определи.**
- **Fit / Обучи.**
- **Predict / Предскажи.**
- **Evaluate / Оцени.**

Если модель прошла оценивание успешно, то ее можно отправлять в бой (на неведомые данные, для которых значения целевой переменной мы не знаем).

Для некоторых данных в будущем мы можем узнать значение целевой переменной и еще раз проверить качество нашей модели.



Какие бывают метрики ошибки?

$$MSE = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2$$

$$MAE = \frac{1}{n} \sum_{i=1}^n |Y_i - \hat{Y}_i|$$

Для задач регрессии самыми популярными метриками ошибками являются mean absolute error и mean squared error (а также корни из них).

В задачах регрессии мы предсказываем значение непрерывной переменной, и рассчитывая ошибку как раз находим разницу между нашими предсказаниями и реальными значениями.



Какие бывают метрики ошибки?

Самой простой метрики для задачи классификации будет доля правильных ответов (accuracy). Мы просто считаем долю правильных ответов нашей модели — тех предсказаний, которые совпадают с реальными значениями.

		Predicted	
		Negative	Positive
Actual	Negative	True Negative	False Positive
	Positive	False Negative	True Positive

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{Total}}$$

Какие бывают метрики ошибки?

		Predicted	
		Negative	Positive
Actual	Negative	True Negative	False Positive
	Positive	False Negative	True Positive

$$\begin{aligned}\text{Precision} &= \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} \\ &= \frac{\text{True Positive}}{\text{Total Predicted Positive}}\end{aligned}$$

Precision (точность) - метрика, которая оценивает долю правильно предсказанных классов 1 среди всех предсказаний 1.

Эта метрика полезна, когда у нас высока цена ложноположительного результата. Например, если у вас есть телефон с Face ID, наверное, вас больше расстроит, если кто-то сможет разблокировать ваш телефон, а не то, что телефон не узнает вас самого.



Какие бывают метрики ошибки?

		Predicted	
		Negative	Positive
Actual	Negative	True Negative	False Positive
	Positive	False Negative	True Positive

$$\begin{aligned}\text{Recall} &= \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} \\ &= \frac{\text{True Positive}}{\text{Total Actual Positive}}\end{aligned}$$

Recall (полнота) учитывает долю ложноотрицательных результатов. Эта метрика полезна, когда высока ложноотрицательной ошибки.

Например, если мы пытаемся обнаруживать мошеннические банковские операции, где цена не остановленной злонамеренной операции гораздо выше вашего временного неудобства из-за блокировки карты.




Какие бывают метрики ошибки?

		Predicted	
		Negative	Positive
Actual	Negative	True Negative	False Positive
	Positive	False Negative	True Positive

F1 - мера объединяющая точность и полноту, их гармоническое среднее. Такая метрика может использоваться как альтернатива доли правильных ответов, когда у нас нет необходимости сильнее штрафовать тот или иной вид ошибки.

$$F1 = 2 \times \frac{Precision * Recall}{Precision + Recall}$$



Качество модели может зависеть от метрики ошибки!

		Predicted/Classified	
		Negative	Positive
Actual	Negative	998	0
	Positive	1	1

		Predicted	
		Negative	Positive
Actual	Negative	True Negative	False Positive
	Positive	False Negative	True Positive

Качество модели может зависеть от метрики ошибки!

		Predicted/Classified	
		Negative	Positive
Actual	Negative	998	0
	Positive	1	1


Accuracy = 0.99

Precision = 1

Recall = 0.5

F1 = 0.67

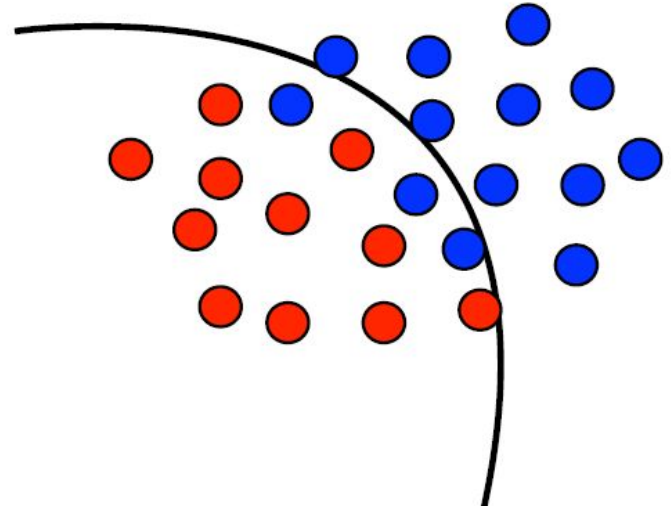
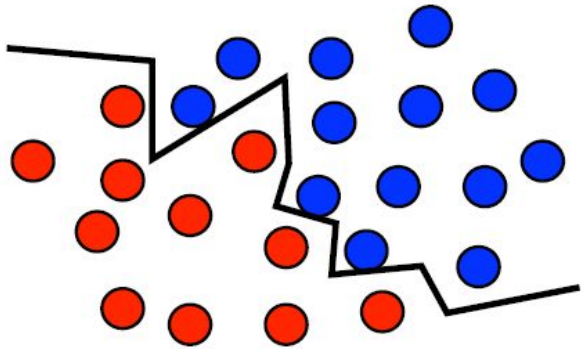
		Predicted	
		Negative	Positive
Actual	Negative	True Negative	False Positive
	Positive	False Negative	True Positive



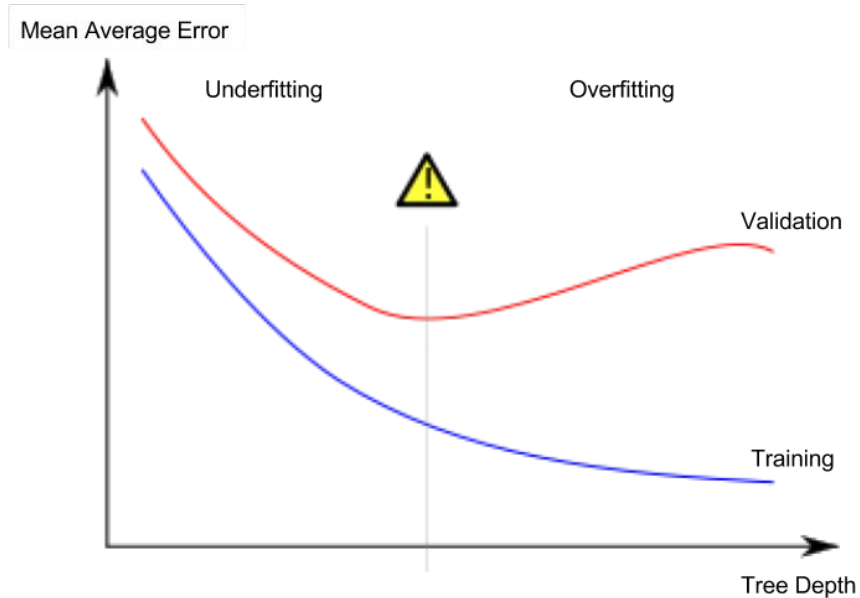
Алгоритм должен обобщать данные, не запоминать их!

- Мы стремимся уменьшить ошибку предсказания на тренировочном наборе данных, но не полностью. 100% точности - это плохо, значит ваша модель запомнила данные “наизусть”.
- Очень сложные модели с большим количеством переменных обычно переобучаются (**overfit**).
- С другой стороны, если наш алгоритм слишком общий, то модель не отражает главные особенности по которым можно разделить данные - недообученная модель (**underfit**).

Какая модель переобучилась?



Какая модель переобучилась?



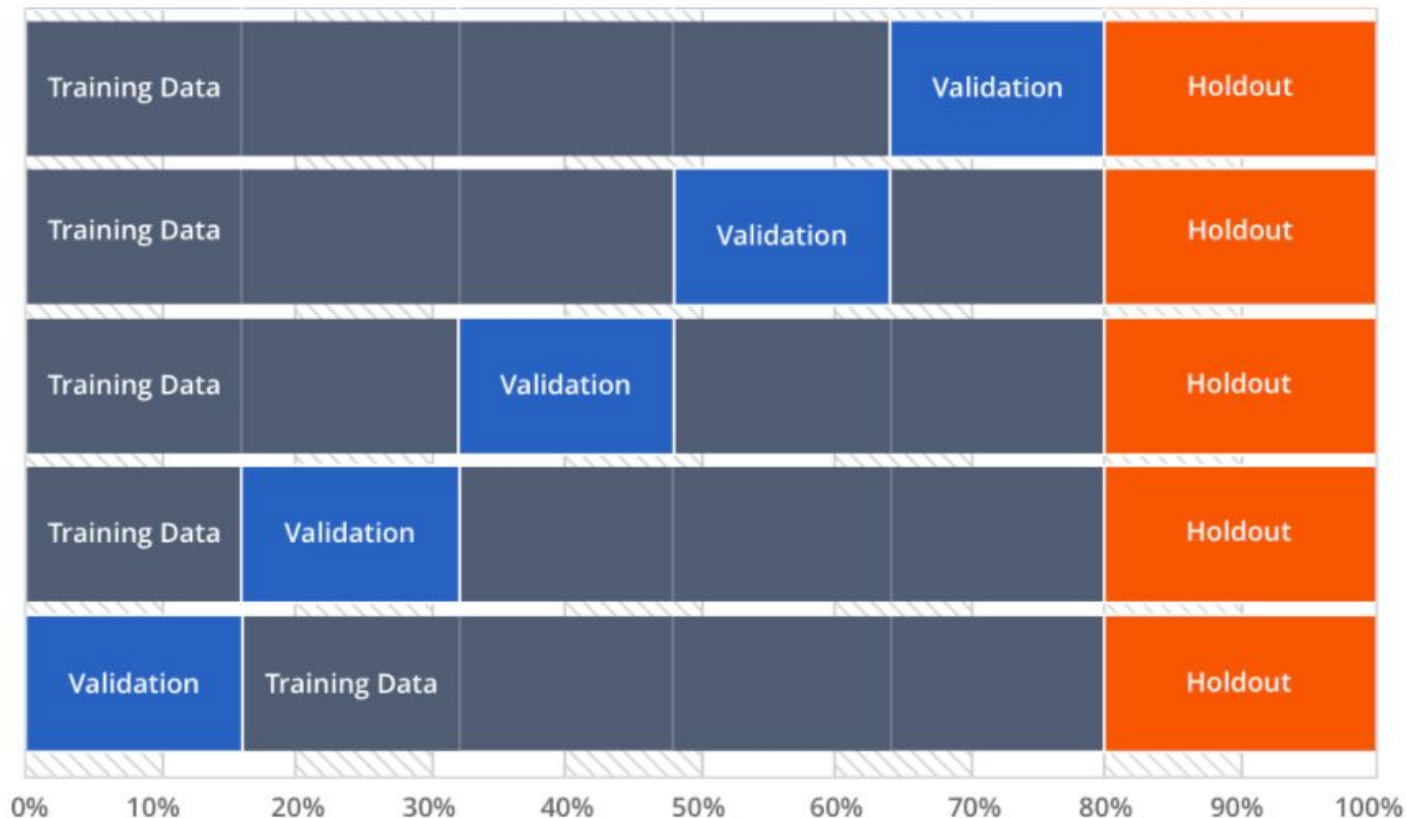
<https://www.kaggle.com/dansbecke/r/underfitting-and-overfitting>




Улучшение качества модели

- **Перекрестная проверка / Cross Validation (CV):** такое разбиение тестовых и тренировочных данных на части, чтобы можно было обучиться несколько раз на этих данных и выбрать лучшую модель. Очень при маленьких размерах тренировочной выборки.
- **Поиск параметров по сетке / GridSearch:** перебор параметров модели для поиска таких параметров, при которых значение функции потерь минимально.

Перекрестная проверка / Cross Validation (CV)





Обучение алгоритма ML

- Возьмите набор данных с известным значением целевой переменной и разбейте его на тренировочные и тестовые данные.
- Выберите модель для теста.
- Используйте тренировочный набор данных для обучения ML алгоритма.
- Предскажите целевую переменную на тренировочном наборе данных. Сравните точность предсказания (мы уже знаем это значение!).
- При необходимости улучшите модель (GridSearch, cross-validation)
- Проверьте модель на тестовых данных.
- Примите решение о качестве алгоритма - используйте его для остальных данных или выберите другую модель.



Резюме

1. Мы исследуем наши данные, чтобы понять, какие признаки важные.
2. Мы обучаем нашу модель на выбранных признаках и оцениваем ее качество на основе ошибки.
3. Метрики ошибки выбираются и в зависимости от типа задач и от самих данных / исследовательского вопроса.

Метод k-ближайших соседей

—



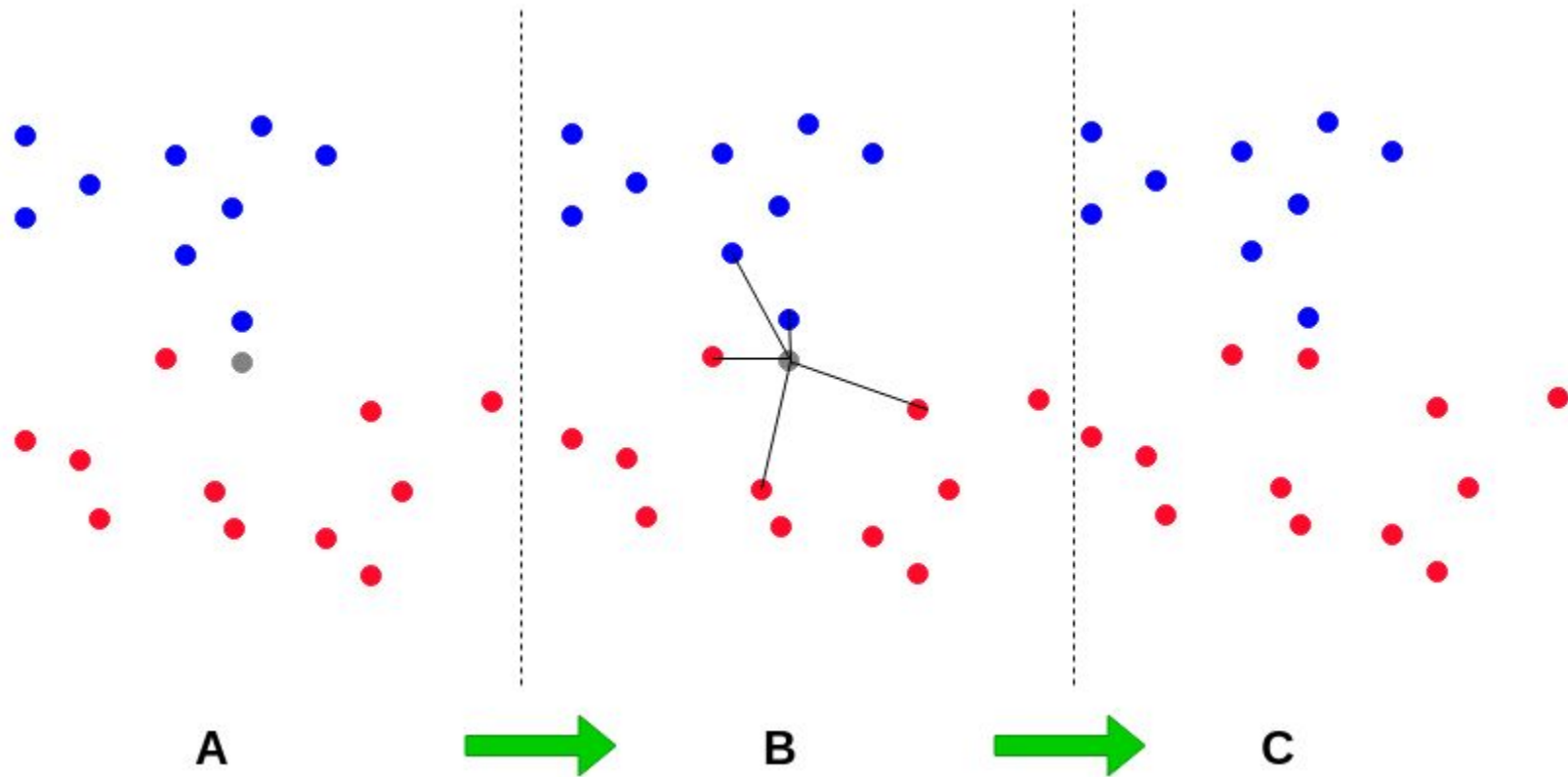
Какие бывают алгоритмы?

Алгоритмов ML достаточно много. Причем у многих есть вариации, которые позволяют решать и задачи регрессии, и классификации. Такие узконаправленные специалисты, как логистическая регрессия, скорее исключения.



Метод k-ближайших соседей

- В случае использования метода для **классификации** объект присваивается тому классу, который является наиболее распространённым среди соседей данного элемента, классы которых уже известны.
- В случае использования метода для **регрессии**, объекту присваивается среднее значение по ближайшим к нему объектам, значения которых уже известны.





Какие параметры можно настраивать?

- **Количество соседей**
- **Веса**

При взвешенном способе во внимание принимается не только количество попавших в область определённых классов, но и их удалённость от нового значения.

- **Метрику дистанции**

Если объекты описываются числовыми векторами, часто берут евклидову метрику. Этот выбор, как правило, ничем не обоснован — просто это первое, что приходит в голову.

При этом необходимо помнить, что все признаки должны быть измерены «в одном масштабе», а лучше всего — отнормированы (но есть нюансы).