**Лекция №8. Оценка модели.**

Оценка качества моделей машинного обучения – это процесс измерения и оценки того, насколько хорошо модель способна предсказывать или классифицировать данные. Она позволяет определить, насколько точно модель может предсказывать значения целевой переменной или классифицировать объекты.

Оценка качества моделей машинного обучения является важной частью процесса разработки и применения моделей. Она позволяет оценить эффективность модели и сравнить ее с другими моделями или алгоритмами. Оценка качества моделей также помогает определить, насколько модель может быть полезной в реальных задачах и принимать решения на основе предсказаний.

Важно использовать несколько показателей оценки для оценки модели. Потому что модель может работать хорошо, используя одну метрику оценки, в то время как производительность может снизиться для другой метрики оценки.

**Почему для успешной модели необходима оценка?**

Переоснащение и недообучение – две основные причины низкой производительности алгоритмов машинного обучения.



[**Переобучение**](https://en.wikipedia.org/wiki/Overfitting)**:**происходит, когда модель хорошо работает с определенным набором данных (известные данные) и, следовательно, может не соответствовать дополнительным данным (неизвестные данные). или надежно прогнозировать будущие наблюдения.



Переобучение происходит, если модель слишком сильно полагается только на один параметр, то есть вес одной фичи затмевает остальные.



[**Недостаточное соответствие**](https://www.datarobot.com/wiki/underfitting/)**:** происходит, когда модель не может адекватно отразить базовую структуру данных.

[**Обобщение**](https://towardsdatascience.com/generalization-regularization-overfitting-bias-and-variance-in-machine-learning-aa942886b870)**:** относится к тому, насколько хорошо концепции, изученные моделью машинного обучения, применимы к конкретным примерам, которые модель не видела во время обучения.

**Нормальный вариант:**



**1. Метрики классификации**

Матрица путаницы – это таблица, которая часто используется для **описания производительности модели классификации** (или «классификатора») на наборе тестовых данных, для которых известны истинные значения.



***Истинно положительный (TP) –*** истинный положительный результат – это результат, при котором модель правильно предсказывает положительный класс.

***Истинно отрицательный (TN) –*** истинно отрицательный результат – это результат, при котором модель правильно предсказывает отрицательный класс.

***Ложноположительный результат (FP) –*** ложноположительный результат – это результат, при котором модель неверно предсказывает положительный класс.

***Ложноотрицательный результат (FN) –*** ложноотрицательный результат – это результат, при котором модель неверно предсказывает отрицательный класс.

**Точность**

**Точность (Accuracy)** – это одна из метрик для оценки моделей классификации. Формально точность можно определить как отношение количества правильных прогнозов к общему количеству прогнозов.



Написав это с точки зрения истинно положительного и истинно отрицательного,



Давайте рассмотрим один пример и попробуем найти его точность.

Предположим, вы пытаетесь построить простую бинарную классификацию, чтобы отличить человеческий образ от нечеловеческого. Скажем, у вас есть 1100 изображений (100 изображений человека и 1000 изображений не человека) с приведенной ниже матрицей ошибок.



Вывод из матрицы ошибок:

1. Из **100 изображений людей** модель предсказала **30 изображений как людей**(TP), а оставшиеся **70 изображений как нечеловеческие** (FN).

2. Из **1000 нечеловеческих изображений** модель спрогнозировала **980 изображений как нечеловеческих**(TN) и оставшиеся **20 изображений как Человек**(FP).

Теперь вычислим точность для модели,



Итак, точность модели составляет 91,81%, что очень хорошо.

Однако, при неравномерном распределении классов accuracy обманчива.

Предположим, что в почтовом ящике 99% обычных писем (класс 0) и 1% спама (класс 1). Тогда, если возвращать всегда 0, то модель будет иметь accuracy 0.99.

**Точность (Precision) и полнота**

**Точность (Precision)** – это метрика, которая измеряет способность модели предсказывать только правильные положительные примеры. Она вычисляется как отношение числа правильно предсказанных положительных примеров к общему числу предсказанных положительных примеров.





**Полнота (Recall)** – это метрика, которая измеряет способность модели обнаруживать все положительные примеры. Она вычисляется как отношение числа правильно обнаруженных положительных примеров к общему числу положительных примеров.



Высокая точность означает, что алгоритм вернул значительно более релевантные результаты, чем нерелевантные. Высокая полнота означает, что алгоритм вернул большинство релевантных результатов.

**Оценка F1**

В случае пары Precision-Recall существует популярный способ скомпоновать их в одну метрику - взять их среднее гармоническое. Данный показатель эффективности исторически носит название **F1-меры (F1-measure)**.



Стоит иметь в виду, что F1-мера предполагает одинаковую важность Precision и Recall, если одна из этих метрик для вас приоритетнее, то можно воспользоваться мерой:



**Кривая ROC и площадь под кривой (AUC)**

Другим способом оценки качества классификации является построение кривой ROC (Receiver Operating Characteristic) и вычисление площади под этой кривой (AUC – Area Under the Curve). Кривая ROC показывает зависимость между долей верно классифицированных объектов положительного класса (True Positive Rate) и долей ложно классифицированных объектов отрицательного класса (False Positive Rate) при изменении порога классификации.

Чем ближе площадь под кривой к 1, тем лучше качество классификации. Значение AUC равное 0.5 говорит о случайном предсказании классов, а значение меньше 0.5 – о неправильной классификации.



**2 Метрики регрессии**

**Среднеквадратическая ошибка** (MSE), пожалуй, самый популярный показатель, используемый для задач регрессии. По сути, он находит среднеквадратичную ошибку между прогнозируемыми и фактическими значениями. MSE – это мера качества оценки –она всегда неотрицательна, а значения ближе к нулю – лучше.



где ***n*** –количество точек данных, ***yᵢ*** –наблюдаемое значение, а ***ŷ***ᵢ –прогнозируемое значение.

Чем ниже MSE, тем лучше прогноз.

Иногда для того, чтобы показатель эффективности MSE имел размерность исходных данных, из него извлекают квадратный корень и получают показатель эффективности RMSE.

**Средняя абсолютная ошибка** (MAE) Использовать MSE для сравнения моделей на выборках с большим количеством выбросов может быть неудобно. В таких случаях прибегают к также знакомой вам в качестве функции потери метрике **MAE** (**mean absolute error**):



где ***n*** –количество точек данных, ***xᵢ*** –истинное значение, а ***yᵢ*** –прогнозируемое значение.

Вспомним, что причина, по которой мы разбиваем наши данные на обучающий и тестовый наборы, заключается в том, что нас интересует, насколько хорошо наша модель обобщает результат на новые, ранее неизвестные данные. Нас интересует не качество подгонки модели к обучающим данным, а правильность ее прогнозов для данных, не участвовавших в обучении. В этой лекции мы подробнее остановимся на одном из аспектов оценки.

**Перекрестная проверка** представляет собой статистический метод оценки обобщающей способности, который является более устойчивым и основательным, чем разбиение данных на обучающий и тестовый наборы. В перекрестной проверке данные разбиваются несколько раз и строится несколько моделей. Наиболее часто используемый вариант перекрестной проверки – k-блочная кросс-проверка (k-fold cross-validation), в которой k – это задаваемое пользователем число, как правило, 5 или 10. При выполнении пятиблочной перекрестной проверки данные сначала разбиваются на пять частей (примерно) одинакового размера, называемых блоками (folds) складками. Затем строится последовательность моделей. Первая модель обучается, используя блок 1 в качестве тестового набора, а остальные блоки (2-5) выполняют роль обучающего набора. Модель строится на основе данных, расположенных в блоках 2-5, а затем на данных блока 1 оценивается ее правильность. Затем происходит обучение второй модели, на этот раз в качестве тестового набора используется блок 2, а данные в блоках 1, 3, 4, и 5 служат обучающим набором. Этот процесс повторяется для блоков 3, 4 и 5, выполняющих роль тестовых наборов. Для каждого из этих пяти разбиений (splits) данных на обучающий и тестовый наборы мы вычисляем правильность. В итоге мы зафиксировали пять значений правильности. Процесс показан на рисунке ниже



Как правило, первая пятая часть данных формирует первый блок, вторая пятая часть данных формирует второй блок и так далее.

**Преимущества перекрестной проверки**

По сравнению с однократным разбиением данных на обучающий и тестовый наборы использование перекрестной проверки имеет несколько преимуществ. Во-первых, вспомним что train\_test\_split выполняет случайное разбиение данных. Представьте себе, что при выполнении случайного разбиения данных нам «повезло», и все трудно классифицируемые примеры в конечном итоге попали в обучающий набор. В этом случае в тестовый набор попадут только «легкие» примеры, и правильность на тестовом наборе будет неправдоподобно высокой. И, наоборот, если нам «не повезло», все трудно классифицируемые примеры попадают в тестовый набор и поэтому мы получаем неправдоподобно низкую правильность. Однако при использовании перекрестной проверки на каждой итерации в тестовый набор, использующийся для проверки модели, попадают разные примеры. Таким образом, модель должна хорошо обобщать все примеры в наборе данных, чтобы все значения правильности (или их среднее) были высокими. Кроме того, наличие нескольких разбиений дает определенную информацию о том, насколько наша модель чувствительна к выбору обучающего набора данных. Для набора данных iris мы увидели разброс значений правильности от 90% до 100%. Это довольно широкий диапазон значений, и он позволяет нам судить о том, как модель будет работать в худшем и лучшем случае, когда мы применим ее к новым данным. Еще одно преимущество перекрестной проверки по сравнению с однократным разбиением данных заключается в том, что мы используем наши данные более эффективно. Применяя train\_test\_split, мы обычно используем 75% данных для обучения и 25% данных для оценки качества. Применяя пятиблочную перекрестную проверку, на каждой итерации для подгонки модели мы можем использовать 4/5 данных (80%). При использовании 10-блочной перекрестной проверки мы можем использовать для подгонки модели 9/10 данных (90%). Больший объем данных, как правило, приводит к построению более точных моделей. Основной недостаток перекрестной проверки – увеличение стоимости вычислений. Поскольку теперь мы обучаем k моделей вместо одной модели, перекрестная проверка будет выполняться примерно в k раз медленнее, чем однократное разбиение данных.