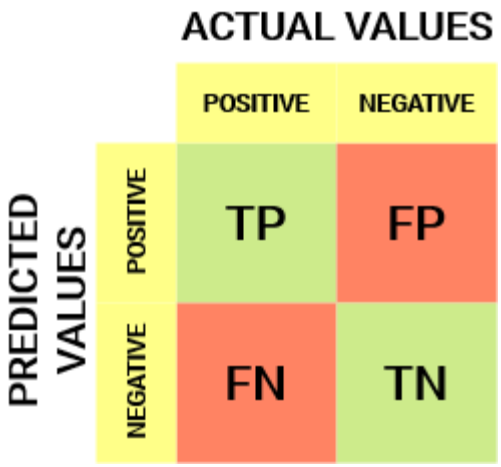
Лекция 3

- Метрики в задачах классификации
- ROC-анализ
- Кривая обучения
- Калибровка классификаторов

Метрики в задачах классификации

https://habr.com/ru/company/ods/blog/328372/

Матрица ошибок (confusion matrix) используется с целью сопоставления предсказаний и реальности в Data Science. Это таблица с 4 различными комбинациями прогнозируемых и фактических значений. Прогнозируемые значения описываются как положительные и отрицательные, а фактические – как истинные и ложные



- TP (**True Positives**) — верно классифицированные положительные примеры (так называемые истинно положительные случаи).
- TN (**True Negatives**) — верно классифицированные отрицательные примеры (истинно отрицательные случаи).
- FN (**False Negatives**) — положительные примеры, классифицированные как отрицательные (ошибка I рода). Это так называемый «ложный пропуск» — когда интересующее нас событие ошибочно не обнаруживается (ложно отрицательные примеры).
- FP (**False Positives**) — отрицательные примеры, классифицированные как положительные (ошибка II рода). Это ложное обнаружение, т.к. при отсутствии события ошибочно выносится решение о его присутствии (ложно положительные случаи).

Ошибка I-го рода (FP): чужого приняли за своего

Ошибка II-го рода (FN): Своего приняли за чужого

В распознавании

Что является положительным событием, а что — отрицательным, зависит от конкретной задачи. *Например, если мы прогнозируем вероятность наличия заболевания, то положительным исходом будет класс «Больной пациент», отрицательным — «Здоровый пациент». И наоборот, если мы хотим определить вероятность того, что человек здоров, то положительным исходом будет класс «Здоровый пациент», и так далее.*

Accuracy

Интуитивно понятной, очевидной и почти неиспользуемой метрикой является **accuracy** — доля правильных ответов алгоритма:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

НО эта метрика бесполезна в задачах с неравными классами, и это легко показать на примере.

Допустим, *мы хотим оценить работу спам-фильтра почты. У нас есть 100 не-спам писем, 90 из которых наш классификатор определил верно (True Negative = 90, False Positive = 10), и 10 спам-писем, 5 из которых классификатор также определил верно (True Positive = 5, False Negative = 5).*

Тогда **accuracy**:

$$accuracy = \frac{5 + 90}{5 + 90 + 10 + 5} = 86,4$$

Однако если мы просто будем предсказывать все письма как не спам, то получим более высокую **accuracy**:

$$accuracy = \frac{0 + 100}{0 + 100 + 0 + 10} = 90,9$$

При этом, наша модель совершенно не обладает никакой предсказательной силой, так как изначально мы хотели определять письма со спамом. Преодолеть это нам поможет переход с общей для всех классов метрики к отдельным показателям качества классов.

Precision, recall и F-мера

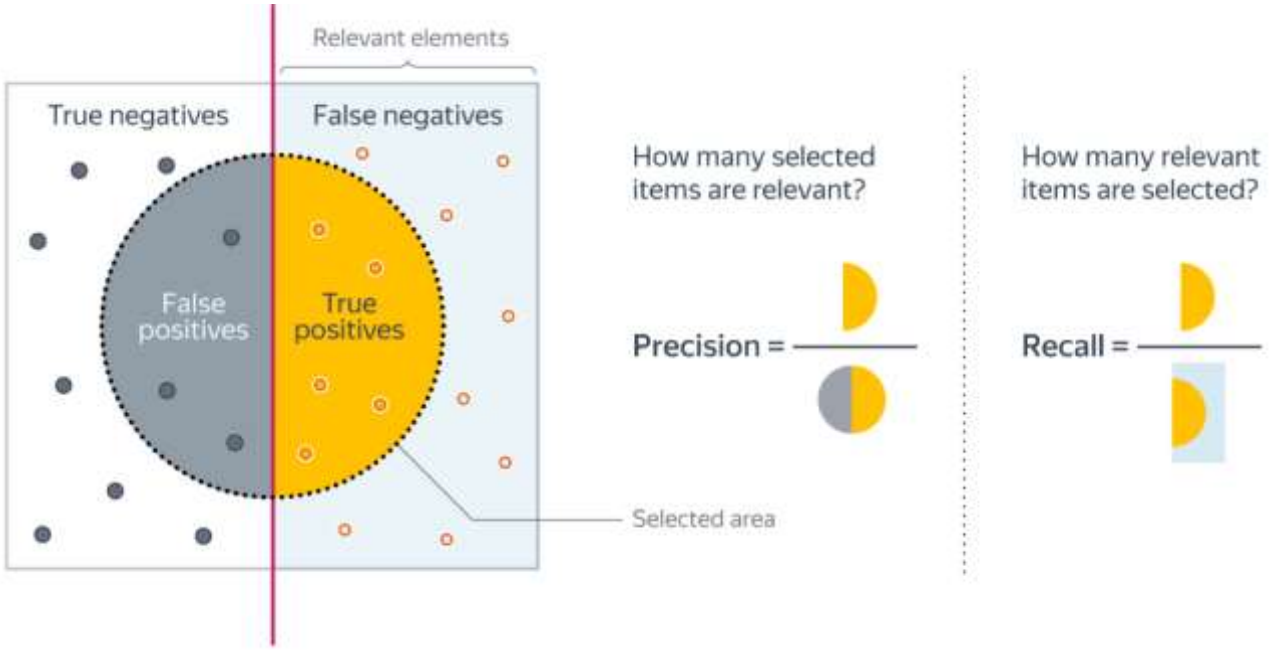
Для оценки качества работы алгоритма на каждом из классов по отдельности введем метрики **precision** (точность) и **recall** (полнота).

Precision можно интерпретировать как долю объектов, названных классификатором положительными и при этом действительно являющимися положительными.

$$precision = \frac{TP}{TP + FP}$$

Recall показывает, какую долю объектов положительного класса из всех объектов положительного класса нашел алгоритм.

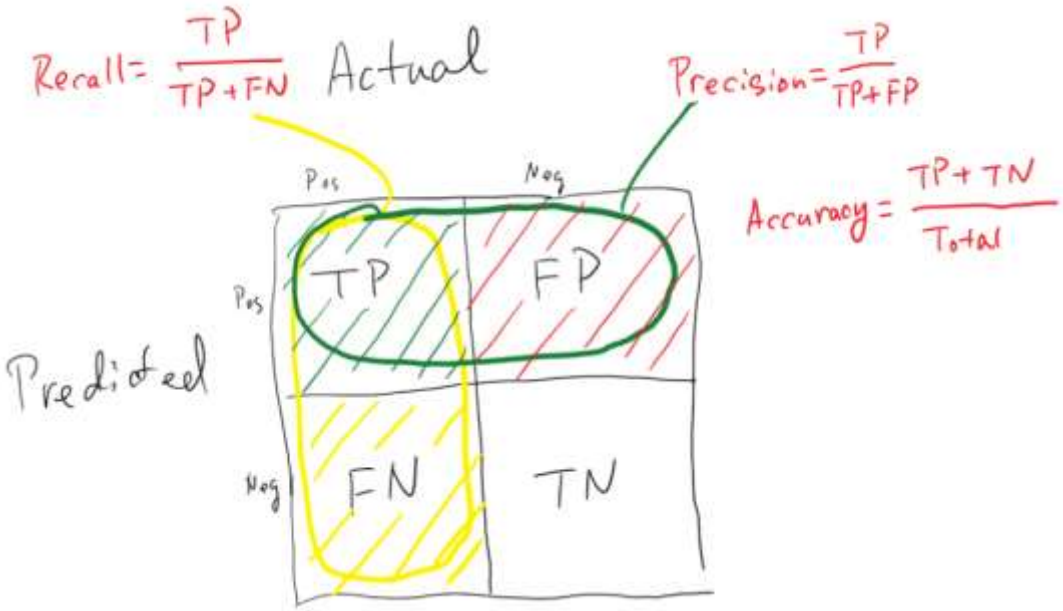
$$recall = \frac{TP}{TP + FN}$$



Именно введение **precision** не позволяет нам записывать все объекты в один класс, так как в этом случае мы получаем рост уровня False Positive.

Recall демонстрирует способность алгоритма обнаруживать данный класс вообще, а **precision** — способность отличать этот класс от других классов.

Precision и **recall** не зависят, в отличие от accuracy, от соотношения классов и потому применимы в условиях несбалансированных выборок.



Источник картинки https://hranalytic.ru/kak-ponyat-matrica-nesootvetstvij-confusion-matrix/

Часто в реальной практике стоит задача найти оптимальный (для заказчика) баланс между этими двумя метриками.

Обычно при оптимизации гиперпараметров алгоритма (например, в случае перебора по сетке GridSearchCV) используется одна метрика, улучшение которой мы и ожидаем увидеть на тестовой выборке.

Существует несколько различных способов объединить precision и recall в агрегированный критерий качества. **F-мера** (в общем случае $F_ß$) — среднее гармоническое precision и recall :

$$F_\beta = (1 + \beta^2) \cdot \frac{precision \cdot recall}{(\beta^2 \cdot precision) + recall}$$

ß в данном случае определяет вес точности в метрике, и при ß = 1 это среднее гармоническое (с множителем 2, чтобы в случае precision = 1 и recall = 1 иметь $F_1 = 1$)

F-мера достигает максимума при полноте и точности, равными единице, и близка к нулю, если один из аргументов близок к нулю.

В *sklearn* есть удобная функция *_metrics.classificationreport*, возвращающая recall, precision и F-меру для каждого из классов, а также количество экземпляров каждого класса.

```
report = classification_report(y_test, lr.predict(X_test), target_names=['Non-
churned', 'Churned'])
print(report)
```

class	precision	recall	f1-score	support
Non-churned	0.88	0.97	0.93	941
Churned	0.60	0.25	0.35	159
avg / total	0.84	0.87	0.84	1100

Support (поддержка) эти значения показывают сколько объектов принадлежит к каждому классу в тестовом наборе данных.

```
>>> from sklearn.metrics import classification_report
>>> y_true = [0, 1, 2, 2, 0]
>>> y_pred = [0, 0, 2, 1, 0]
>>> target_names = ['class 0', 'class 1', 'class 2']
>>> print(classification_report(y_true, y_pred, target_names=target_names))
              precision    recall  f1-score   support

     class 0       0.67      1.00      0.80         2
     class 1       0.00      0.00      0.00         1
     class 2       1.00      0.50      0.67         2

    accuracy                           0.60         5
   macro avg       0.56      0.50      0.49         5
weighted avg       0.67      0.60      0.59         5
```

Macro avg позволяет оценить общую способность классификатора находить образцы из всех классов.

Weighted avg позволяет оценить, как хорошо классификатор обрабатывает множества образцов разных классов.

ROC-анализ

ROC-кривая (Receiver Operator Characteristic) часто используется для представления результатов бинарной классификации в машинном обучении. Название пришло из систем обработки сигналов. Поскольку классов два, один из них называется классом с положительными исходами, второй — с отрицательными исходами. **ROC-кривая показывает зависимость количества верно классифицированных положительных примеров** (истинно положительное множество) **от количества неверно классифицированных отрицательных примеров** (ложно отрицательное множество).

Предполагается, что **у классификатора имеется некоторый параметр, варьируя который, мы будем получать то или иное разбиение на два класса. Этот параметр часто называют порогом,** или точкой отсечения (cut-off value). В зависимости от него будут получаться различные величины ошибок I и II рода.

В логистической регрессии порог отсечения изменяется от 0 до 1 — это и есть расчетное значение уравнения регрессии. Будем называть его рейтингом.

При анализе чаще оперируют не абсолютными показателями, а относительными — долями (rates), выраженными в процентах:

- Доля истинно положительных примеров (**True Positives Rate**):

$$TPR = \frac{TP}{TP + FN} \cdot 100\,\%$$

- Доля ложно положительных примеров (**False Positives Rate**):

$$FPR = \frac{FP}{TN + FP} \cdot 100\,\%$$

Введем еще два определения: **чувствительность** и **специфичность модели**. Ими определяется объективная ценность любого бинарного классификатора.

Чувствительность (Sensitivity) —доля истинно положительных случаев:

$$S_e = TPR = \frac{TP}{TP + FN} \cdot 100\,\%$$

Специфичность (Specificity) — доля истинно отрицательных случаев, которые были правильно идентифицированы моделью:

$$S_p = \frac{TN}{TN + FP} \cdot 100\,\%$$

Попытаемся разобраться в этих определениях.

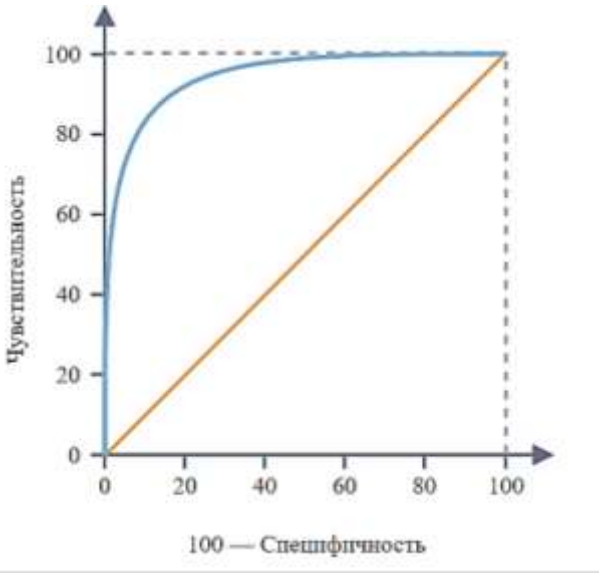
Модель с высокой чувствительностью часто дает истинный результат при наличии положительного исхода (обнаруживает положительные примеры).

Наоборот, **модель с высокой специфичностью чаще дает истинный результат при наличии отрицательного исхода** (обнаруживает отрицательные примеры). Если рассуждать в терминах медицины — задачи диагностики заболевания, где модель классификации пациентов на больных и здоровых называется диагностическим тестом (*болен – положительный исход, здоров – отрицательный*), то получится следующее:

- ✓ **Чувствительный диагностический тест проявляется в гипердиагностике** — максимальном предотвращении пропуска больных.
- ✓ **Специфичный диагностический тест диагностирует только доподлинно больных.** Это важно в случае, когда, например, лечение больного связано с серьезными побочными эффектами и гипердиагностика пациентов не желательна.

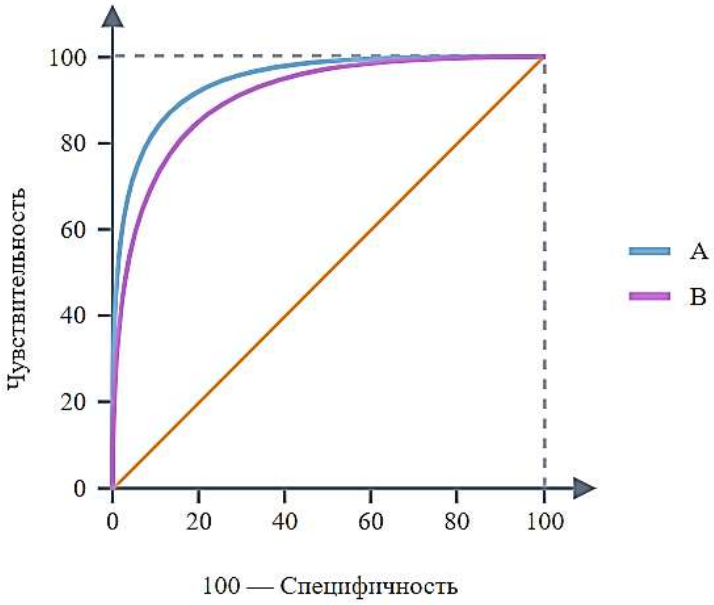
ROC-кривая получается следующим образом:

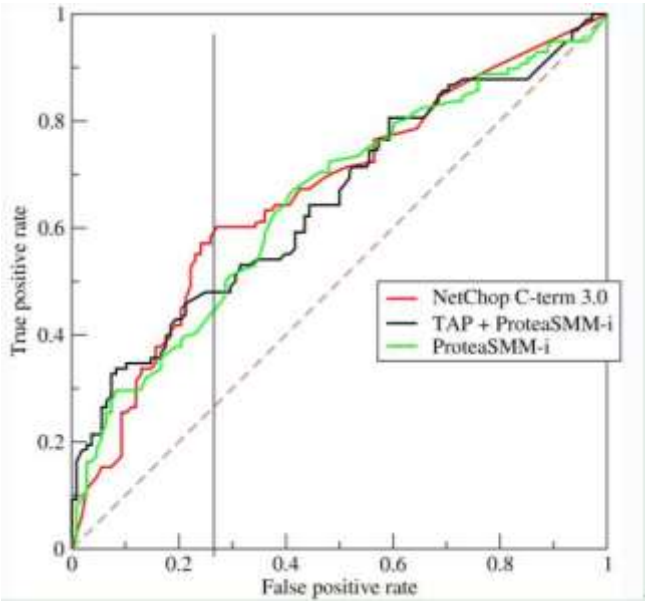
- ✓ Для каждого значения порога отсечения, которое меняется от 0 до 1 с шагом d_x (например, 0.01) рассчитываются значения чувствительности Se и специфичности Sp. В качестве альтернативы порогом может являться каждое последующее значение примера в выборке.
- ✓ Строится график зависимости: по оси Y откладывается чувствительность Se, по оси X — $FPR = 100 - Sp$ — доля ложно положительных случаев.



Для идеального классификатора график ROC-кривой проходит через верхний левый угол, где доля истинно положительных случаев составляет 100% или 1,0 (идеальная чувствительность), а доля ложно положительных примеров равна нулю. Поэтому чем ближе кривая к верхнему левому углу, тем выше предсказательная способность модели. Наоборот, чем меньше изгиб кривой и чем ближе она расположена к диагональной прямой, тем менее эффективна модель. Диагональная линия (y=x) соответствует «бесполезному» классификатору, т.е. полной неразличимости двух классов.

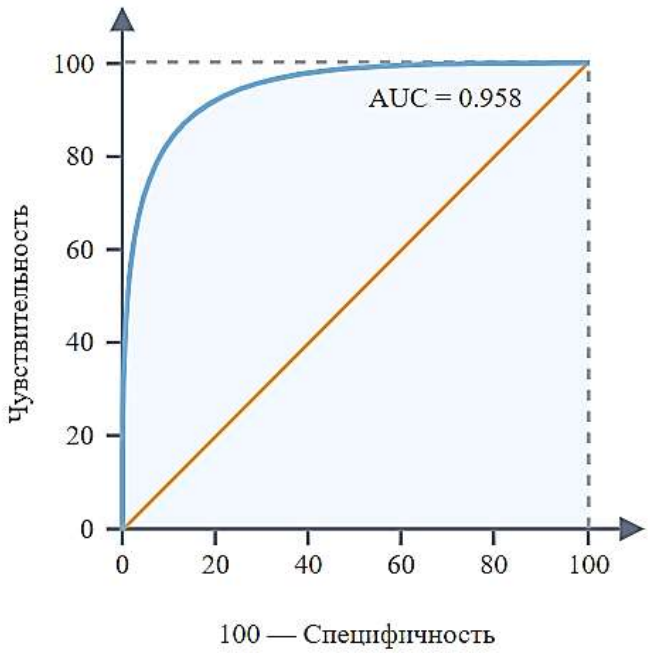
При визуальной оценке ROC-кривых расположение их относительно друг друга указывает на их сравнительную эффективность. Кривая, расположенная выше и левее, свидетельствует о большей предсказательной способности модели. Так, на рисунке видно, что модель «А» лучше.





Однако визуальное сравнение кривых ROC не всегда позволяет выявить наиболее эффективную модель (см. рис.). Своеобразным методом сравнения ROC-кривых является **оценка площади под кривыми**. Теоретически она изменяется от 0 до 1,0, но, поскольку модель всегда характеризуются кривой, расположенной выше положительной диагонали, то обычно говорят об изменениях от 0,5 («бесполезный» классификатор) до 1,0 («идеальная» модель).

Эта оценка может быть получена непосредственно вычислением площади под многогранником, ограниченным справа и снизу осями координат и слева вверху — экспериментально полученными точками. Численный показатель площади под кривой называется AUC (Area Under Curve).

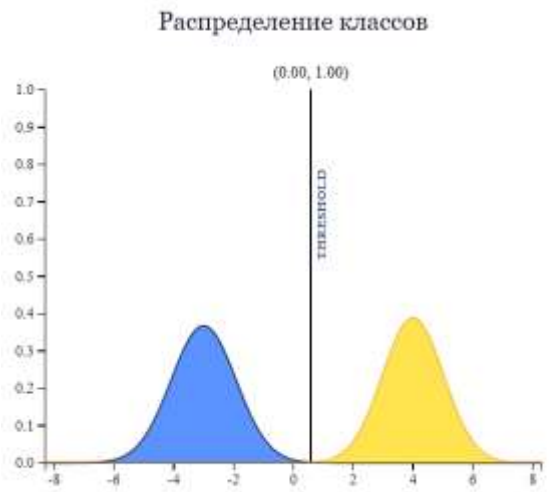


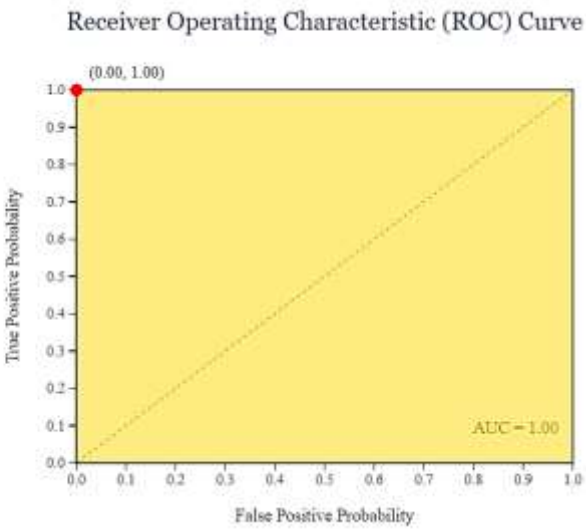
С большими допущениями можно считать, что чем больше показатель AUC, тем лучшей прогностической силой обладает модель. Однако следует знать, что:

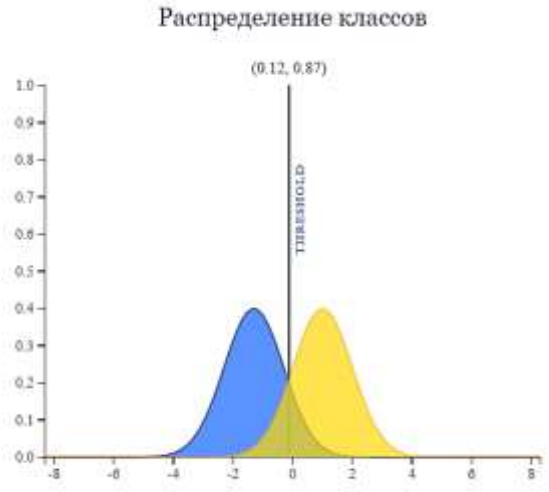
- показатель AUCAUC предназначен скорее для сравнительного анализа нескольких моделей;
- AUCAUC не содержит никакой информации о чувствительности и специфичности модели.

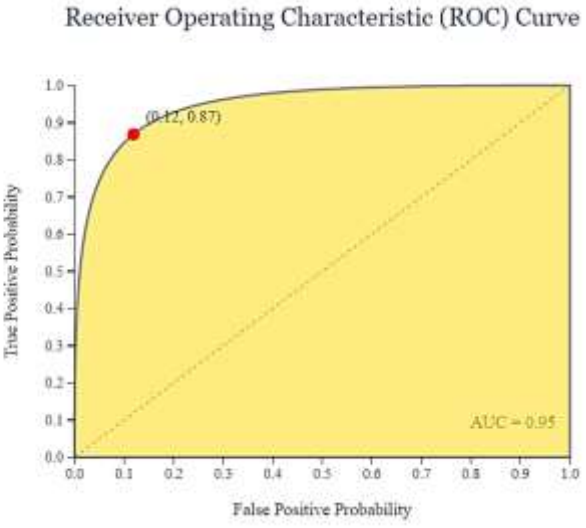
В литературе иногда приводится следующая экспертная шкала для значений AUCAUC, по которой можно судить о качестве модели:

Интервал AUC	Качество модели
0,9-1,0	Отличное
0,8-0,9	Очень хорошее
0,7-0,8	Хорошее
0,6-0,7	Среднее
0,5-0,6	Неудовлетворительное





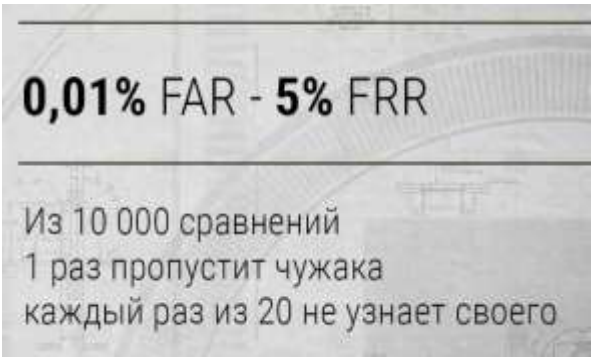




Данный симулятор распределения классов и построения ROC-кривой можно найти по адресу https://ml-handbook.ru/chapters/model_evaluation/intro

Идеальная модель обладает 100% чувствительностью и специфичностью. Однако на практике добиться этого невозможно, более того, невозможно одновременно повысить и чувствительность, и специфичность модели. Компромисс находится с помощью порога отсечения, т.к. пороговое значение влияет на соотношение Se и Sp. Можно говорить о **задаче нахождения оптимального порога отсечения** (optimal cut-off value).

Например, в системах распознавания при уменьшении ошибки второго рода резко возрастает ошибка первого рода



- ✓ FAR (False Acceptance Rate) — процентный порог, определяющий вероятность того, что один человек может быть принят за другого (коэффициент ложного доступа) (ошибка 2 рода). Величина **1−FAR** называется специфичность.
- ✓ FRR (False Rejection Rate) — вероятность того, что человек может быть не распознан системой (коэффициент ложного отказа в доступе) (ошибка 1 рода). Величина **1−FRR** называется чувствительность.

Порог отсечения нужен для того, чтобы применять модель на практике: относить новые примеры к одному из двух классов. **Для определения оптимального порога нужно задать критерий его определения**, т.к. в разных задачах присутствует своя оптимальная стратегия. Критериями выбора порога отсечения могут выступать:

- ✓ Требование минимальной величины чувствительности (специфичности) модели. Например, нужно обеспечить чувствительность теста не менее 80%. В этом случае оптимальным порогом будет максимальная специфичность (чувствительность), которая достигается при 80%.
- ✓ Требование максимальной суммарной чувствительности и специфичности модели, т.е.

$$Cutt\_off_o = \max_k(Se_k + Sp_k)$$

В этом случае значение порога обычно предлагается пользователю по умолчанию.

- ✓ Требование баланса между чувствительностью и специфичностью, т.е. когда Se ≈ Sp

$$Cutt\_off_o = \min_k |Se_k - Sp_k|$$

В этом случае порог есть точка пересечения двух кривых, когда по оси X откладывается порог отсечения, а по оси Y — чувствительность или специфичность модели (рис. 5).

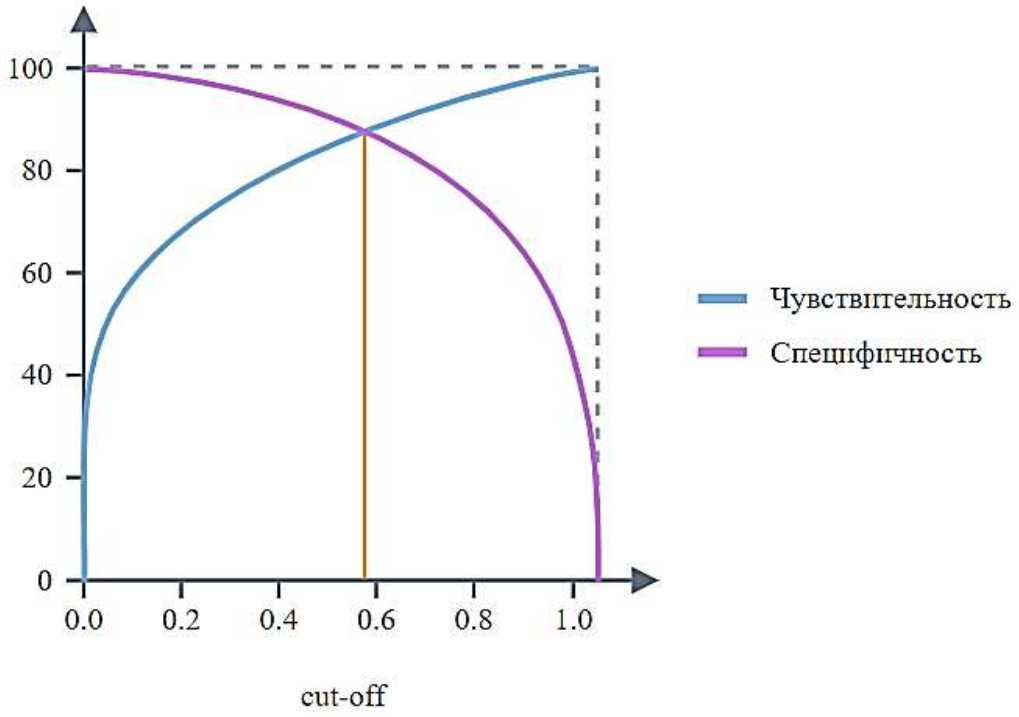
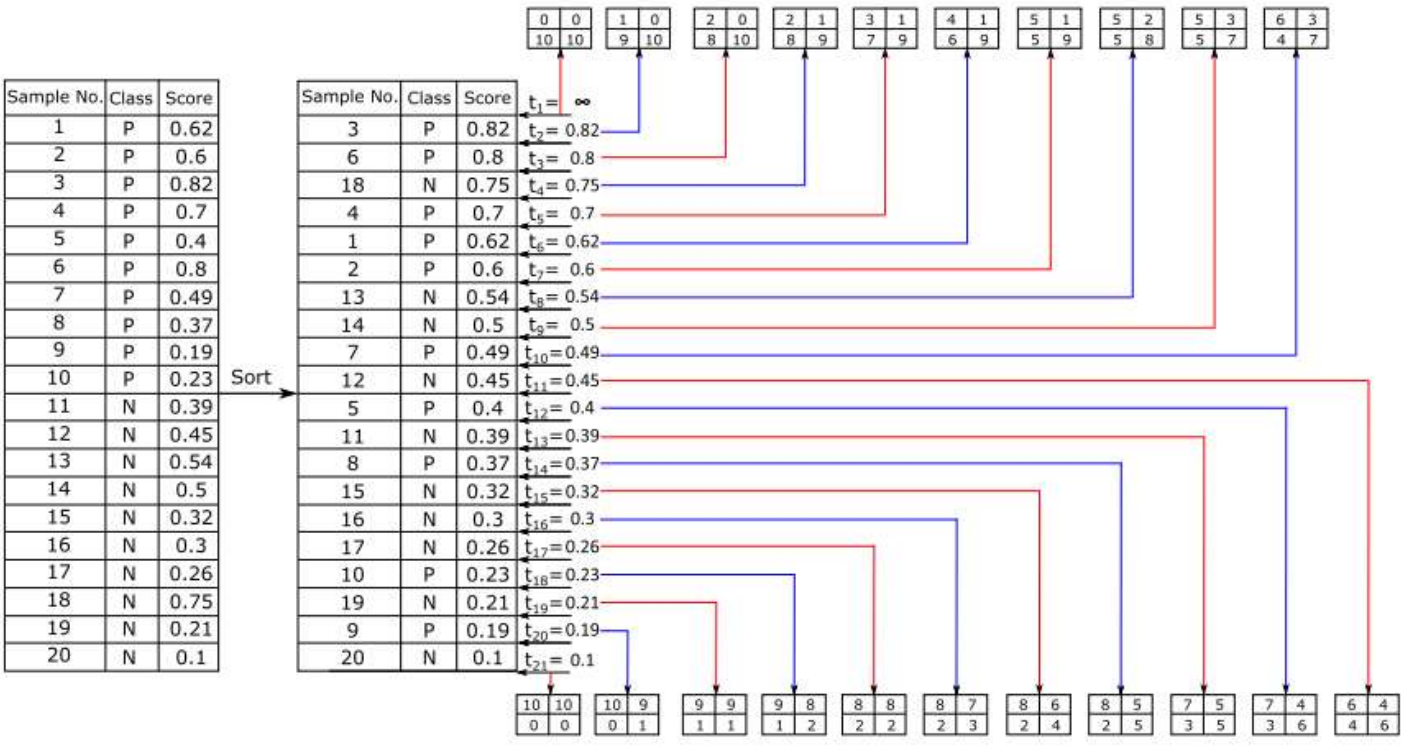
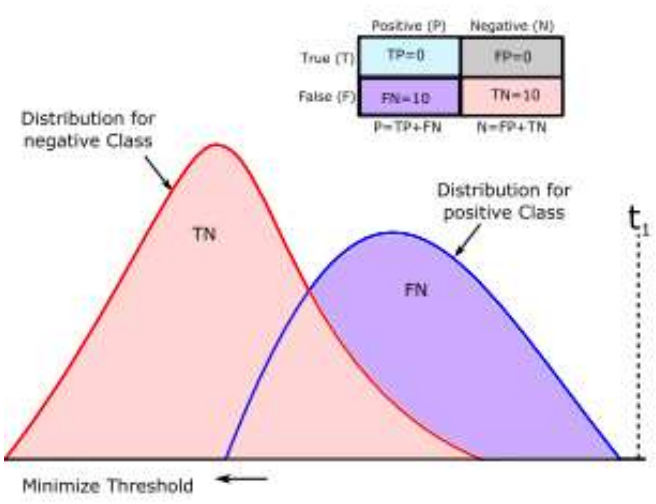


Рис. «Точка баланса» между чувствительностью и специфичностью

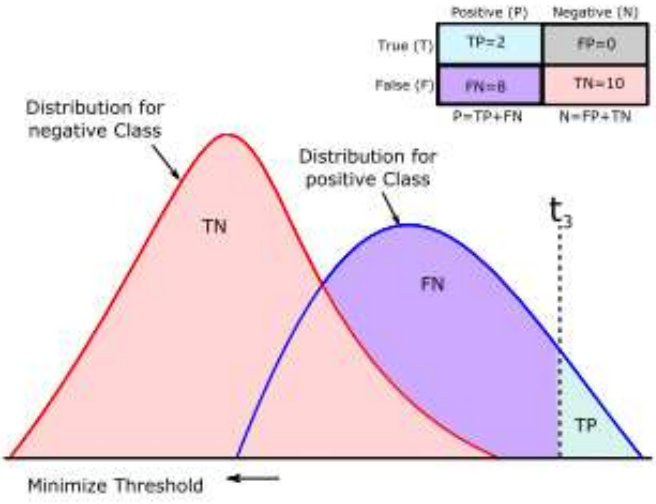
Наглядный пример расчета TPR и FPR при изменении порогового значения.

https://www.researchgate.net/publication/327148996_Classification_Assessment_Methods_a_detailed_tutorial

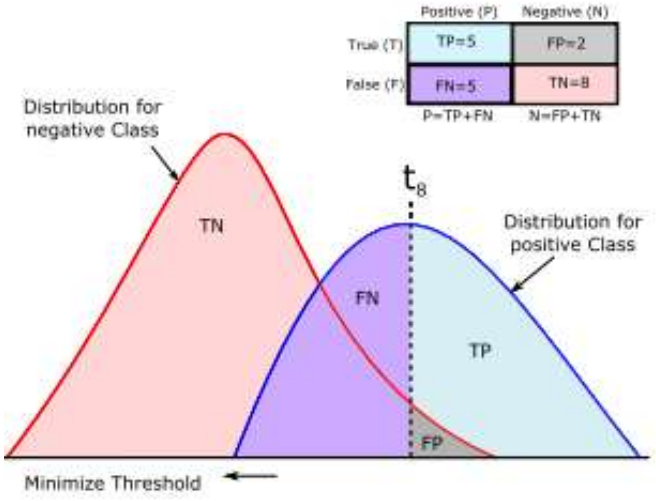



Positive (P)
Negative (N)
True (T)
False (F)
TP=0
FP=0
FN=10
TN=10
P=TP+FN
N=FP+TN
Distribution for negative Class
Distribution for positive Class
TN
FN
t_1
Minimize Threshold

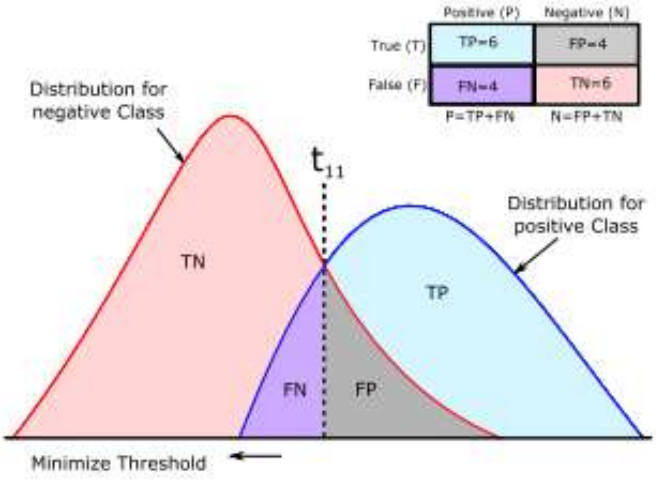
(a)


Positive (P)
Negative (N)
True (T)
False (F)
TP=2
FP=0
FN=8
TN=10
P=TP+FN
N=FP+TN
Distribution for negative Class
Distribution for positive Class
TN
FN
TP
t_3
Minimize Threshold

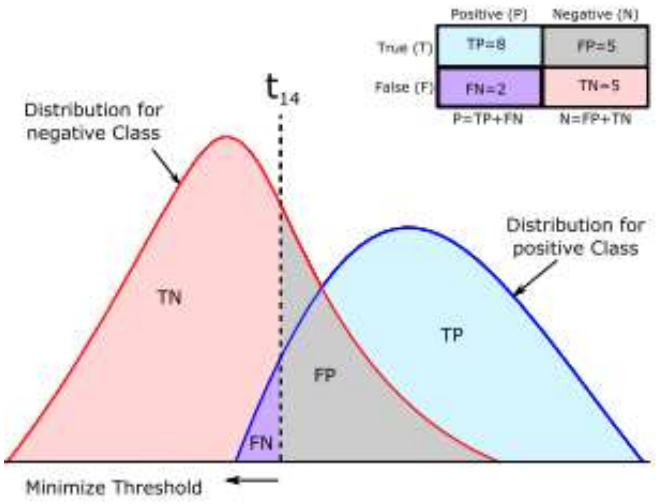
(b)

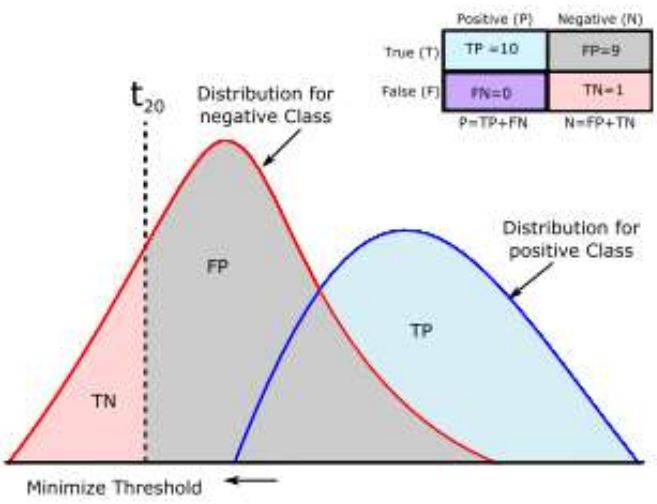

Positive (P)
Negative (N)
True (T)
False (F)
TP=5
FP=2
FN=5
TN=8
P=TP+FN
N=FP+TN
Distribution for negative Class
Distribution for positive Class
TN
FN
TP
FP
t_8
Minimize Threshold

(c)


Positive (P)
Negative (N)
True (T)
False (F)
TP=6
FP=4
FN=4
TN=6
P=TP+FN
N=FP+TN
Distribution for negative Class
Distribution for positive Class
TN
FN
TP
FP
t_{11}
Minimize Threshold

(d)


Positive (P)
Negative (N)
True (T)
False (F)
TP=8
FP=5
FN=2
TN=5
P=TP+FN
N=FP+TN
Distribution for negative Class
Distribution for positive Class
TN
FN
TP
FP
t_{14}
Minimize Threshold


Positive (P)
Negative (N)
True (T)
False (F)
TP =10
FP=9
FN=0
TN=1
P=TP+FN
N=FP+TN
Distribution for negative Class
Distribution for positive Class
TN
FP
TP
t_{20}
Minimize Threshold

Кривые обучения

Кривые обучения - это широко используемый диагностический инструмент в машинном обучении для алгоритмов, которые постепенно обучаются на основе обучающего набора данных. Модель может быть оценена на наборе обучающих данных и на тестовом наборе данных после каждого обновления во время обучения, измеренная производительность отображается на графиках для отображения кривых обучения.

Анализ кривых обучения моделей может использоваться для диагностики проблем с обучением, таких как недостаточная или избыточная модель, а также для определения того, являются ли наборы данных для обучения и проверки надлежащим образом репрезентативными.

Кривая обучения - это график эффективности обучения модели в зависимости от опыта или времени. По оси **x** (как правило) откладывают время или опыт, а по оси **y** обучение или улучшения.

Эффективность обучения может оцениваться

- ✓ **точностью классификации**, тогда большим значениям точности соответствует лучшее обучение
- ✓ или **величиной ошибки**, тогда лучшее обучение соответствует меньшим значениям ошибки.

Оценка текущего состояния модели **на обучающем наборе** данных, показывает насколько хорошо модель «**обучается**».

Оценка **на тестовом наборе** данных проверки дает представление о том, насколько хорошо модель «**обобщает**».

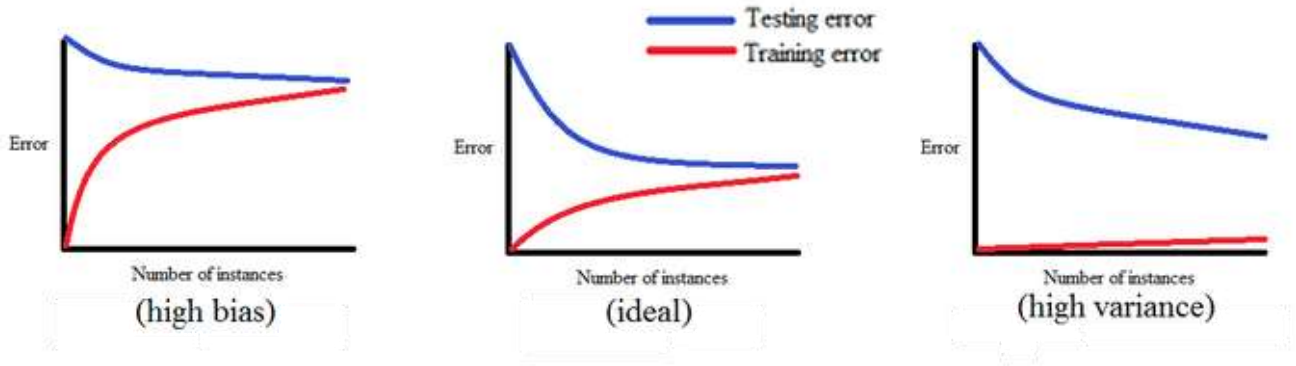
Диагностика поведения модели

Форма и динамика кривой обучения могут использоваться для диагностики поведения модели машинного обучения и, в свою очередь, предлагать тип изменений конфигурации, которые могут быть внесены для улучшения обучения и / или производительности.

Есть три общих динамики, которые можно заметить в кривых обучения:

- ✓ недообучение
- ✓ переобучение
- ✓ хорошо подходит.

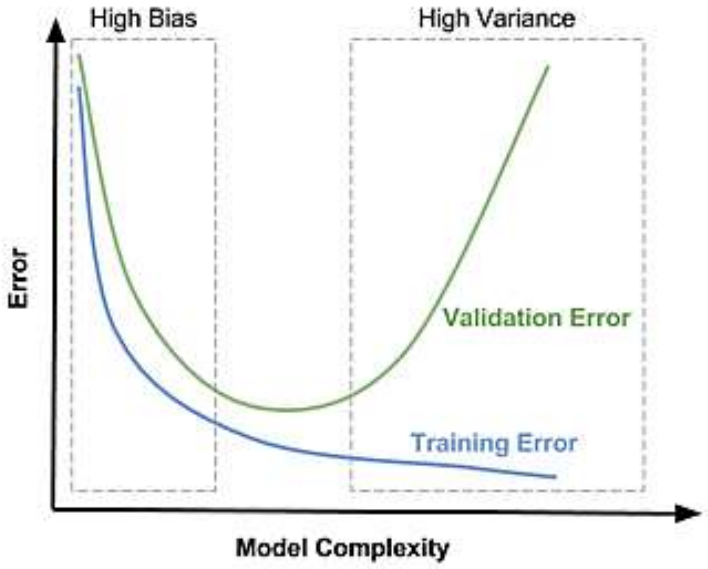
Обычно результаты обучения и тестирования/валидации строятся вместе, чтобы мы могли диагностировать компромисс между смещением и дисперсией (т. е. определить, выиграем ли мы от добавления большего количества обучающих данных, и оценить сложность модели, контролируя регуляризацию или количество функций).



Смещение (bias) — это погрешность оценки, возникающая в результате ошибочного предположения в алгоритме обучения. В результате большого смещения алгоритм может пропустить связь между признаками и выводом (недообучение).

Дисперсия (variance) — это ошибка чувствительности к малым отклонениям в тренировочном наборе. При высокой дисперсии алгоритм может как-то трактовать случайный шум в тренировочном наборе, а не желаемый результат (переобучение).

Компромисс отклонение-дисперсия — конфликт при попытке одновременно минимизировать эти два источника ошибки, которые мешают алгоритмам обучения с учителем делать обобщение за пределами тренировочного набора. При уменьшении одного из негативных эффектов, приводит к увеличению другого. Данная дилемма проиллюстрирована на рис. При небольшой сложности модели мы наблюдаем большое смещение (high bias). При усложнении модели смещение уменьшается, но дисперсия увеличится, что приводит к проблеме отклонение-дисперсия.



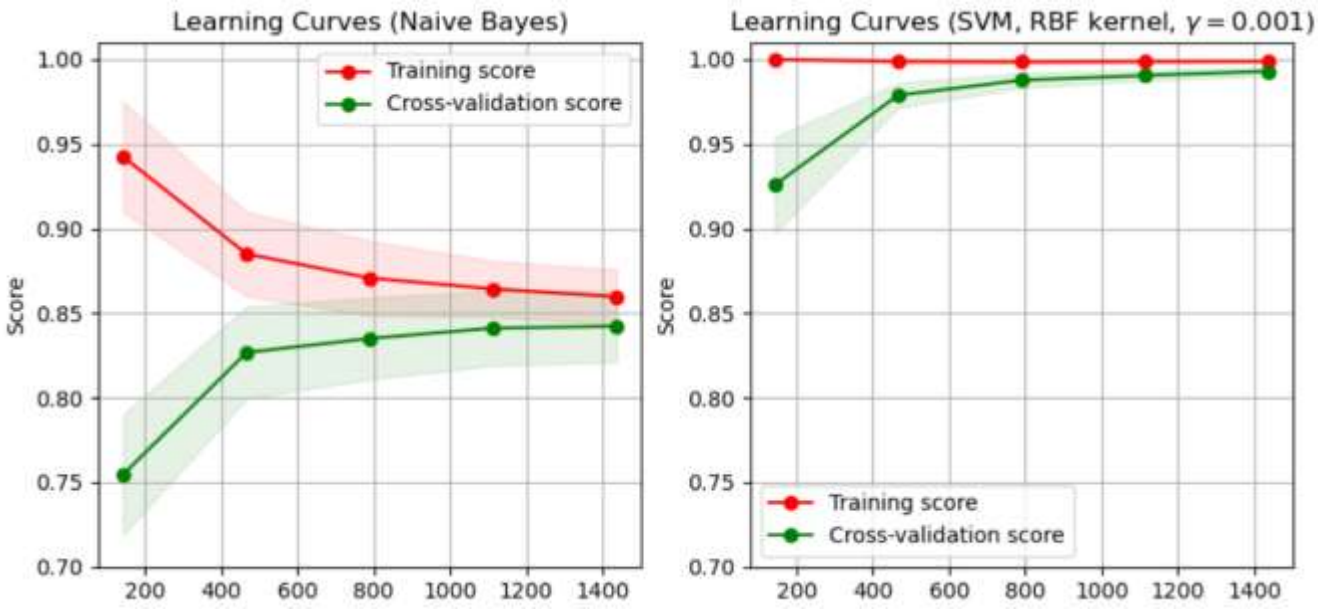
Возможные решения при переобучении

- ✓ Увеличение количества данных в наборе;
- ✓ Уменьшение количества параметров модели;
- ✓ Добавление регуляризации / увеличение коэффициента регуляризации.

Возможные решения при недообучении

- ✓ Добавление новых параметров модели;
- ✓ Использование для описания модели функций с более высокой степенью;
- ✓ Уменьшение коэффициента регуляризации.

Вопрос: какая модель лучше?



https://scikit--learn-org.translate.goog/stable/auto_examples/model_selection/plot_learning_curve.html?_x_tr_sl=en&_x_tr_tl=ru&_x_tr_hl=ru&_x_tr_pto=nui,sc

Калибровка классификаторов

https://ichi.pro/ru/kalibrovka-klassifikatora-137489987687506

Для оценки полученной модели классификатора мы используем вероятность, с которой модель определяет класс для каждого объекта. Но всегда ли это отражает реальность?

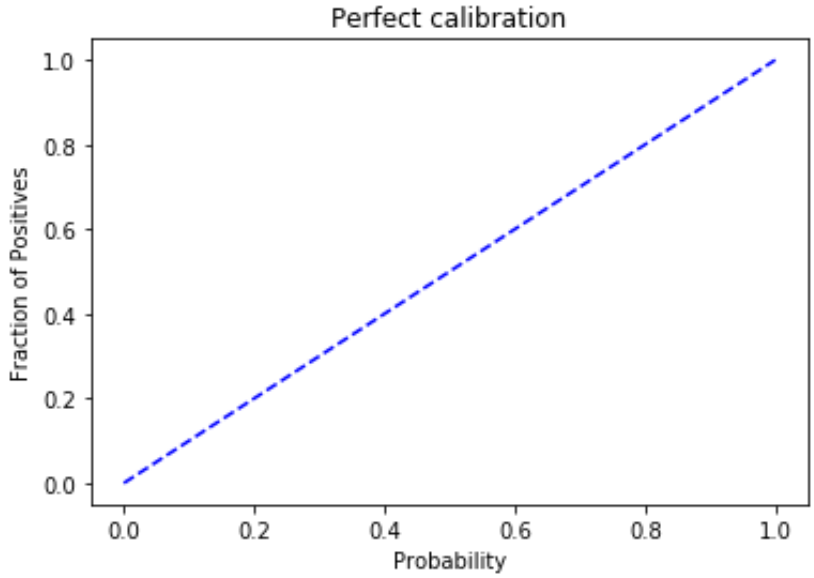
Представьте, что у нас есть два бинарных классификатора; модель A и модель B. Модель A имеет точность 85% и уверенность 0,86 для каждого сделанного прогноза. С другой стороны, модель B также на 85% точна, но достоверна на 0,99 для каждого из своих прогнозов. Как вы думаете, какая модель лучше?

Модель A лучше, потому что считает себя точной в 86% случаев, и это действительно так. Напротив, модель B слишком уверена в своих прогнозах. Этот пример демонстрирует интуицию вероятности и калибровки модели.

Калибровка модели относится к процессу, в котором мы берем модель, которая уже обучена, и применяем операцию постобработки, которая улучшает ее оценку вероятности. Таким образом, если бы мы проверили образцы, которые были оценены как положительные с вероятностью 0,85, мы бы ожидали, что 85% из них действительно положительные.

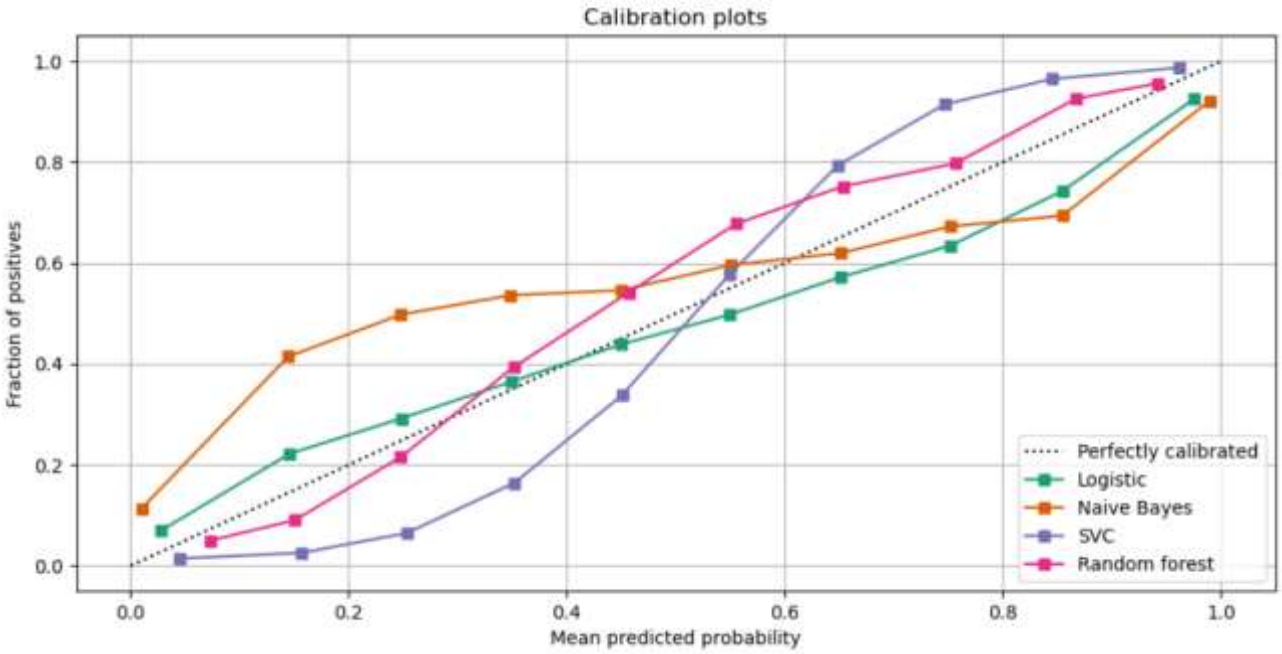
Формально модель идеально откалибрована, если для любого значения вероятности ***p*** уверенное предсказание класса в процентах случаев ***p*** является правильным ***100*p***.

Если построить каждое значение ***p*** в интервале от 0 до 1, мы ожидаем получить идеальную линейную связь между вычисленной вероятностью и долей положительных результатов.



Калибровочные кривые (также известные как диаграммы надежности) позволяют сравнить, насколько хорошо откалиброваны вероятностные прогнозы двоичного классификатора. Он отображает истинную частоту положительной метки против ее прогнозируемой вероятности для прогнозов с разбиением на

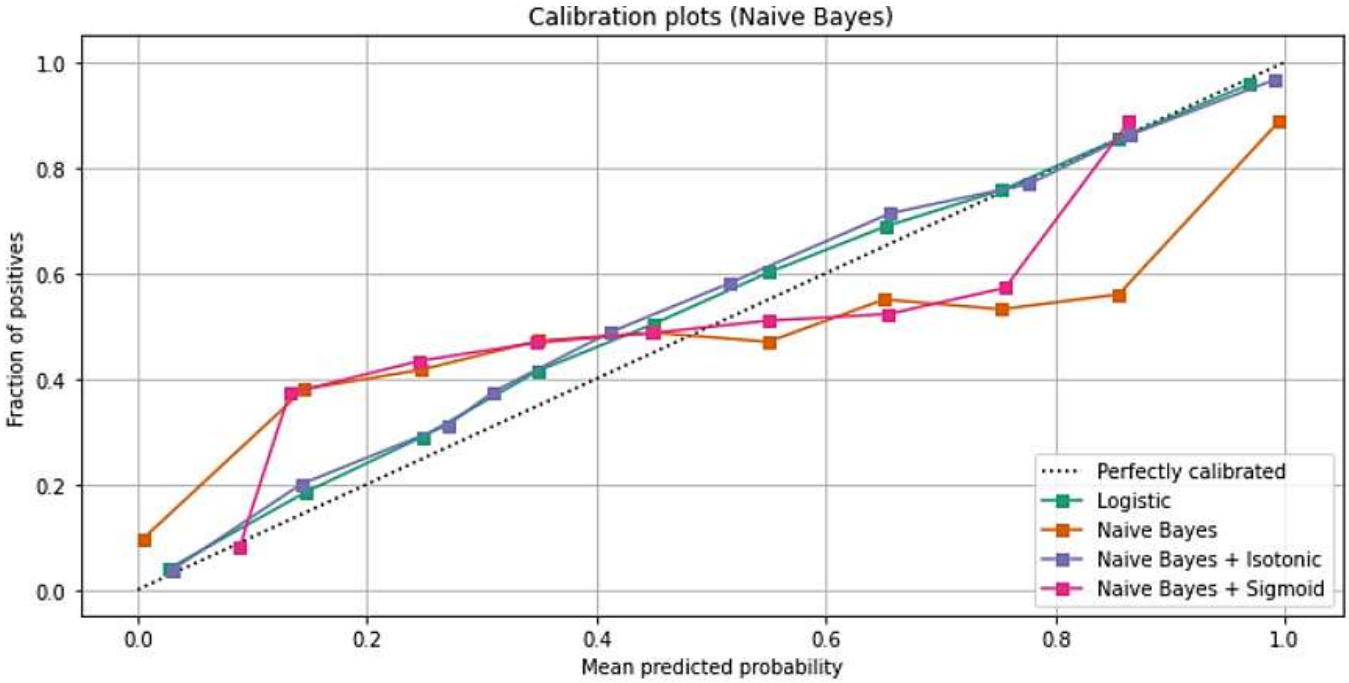
интервалы. Ось x представляет собой среднюю прогнозируемую вероятность в каждом интервале. По оси ординат отложена доля положительных результатов, т. е. доля образцов, класс которых является положительным (в каждой ячейке). График верхней калибровочной кривой создается с помощью CalibrationDisplay.from_estimators, который используется calibration_curve для расчета средней прогнозируемой вероятности для каждого бина и доли положительных результатов. CalibrationDisplay.from_estimator принимает в качестве входных данных подобранный классификатор, который используется для вычисления прогнозируемых вероятностей. Таким образом, классификатор должен иметь pred_proba метод. Для нескольких классификаторов, у которых нет метода pred_proba, его можно использовать CalibratedClassifierCV для калибровки выходных данных классификатора по вероятностям.



CalibratedClassifierCV - Класс используется для калибровки классификатора.

CalibratedClassifierCV поддерживает использование двух «калибровочных» регрессоров: «**сигмовидной**» и «**изотонической**».

```
gnb = GaussianNB()
gnb_isotonic = CalibratedClassifierCV(gnb, cv=2, method="isotonic")
gnb_sigmoid = CalibratedClassifierCV(gnb, cv=2, method="sigmoid")
```



Так ли важна калибровка модели?

Калибровка модели важна только в том случае, если вы заботитесь о вероятностях, которые вычисляет ваша модель. Например, предположим, что вы создаете механизм рекомендаций, который ранжирует продукты в соответствии с предпочтениями пользователя. Если ваша модель оценивает, что пользователь **M** купит продукт **a** с вероятностью 0,9, а элемент – **b** с вероятностью 0,7, вы можете сначала подать продукт **a**. Калибровать эту модель не нужно.

Однако, если вы создаете критически важное приложение, которое вычисляет вероятность того, что человек заболел, фактическое значение вероятности имеет большое значение.

Можно еще почитать сравнение классификаторов

https://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html?highlight=gaussiannb