KARPOV.COURSES >>>
КОНСПЕКТ

Конспект > 12 урок > ROC, PR-кривые. AUC-ROC, AUC-PR. Калибровка

> ROC-кривая

Бизнес требования к модели могут быть зараненее неизвестны, например, какой порог t в итоге потребуется. Поэтому лучше научиться оценивать качество работы модели не для одного трешхолда t, а в целом по всем порогам.

Например, на таблице внизу представлены значения recall и precision для разных значений t. **Как агрегировать все эти значения в единую метрику?**

t	*recall*	*precision*
1	0	1
0.9	0.05	0.97
0.8	0.2	0.86
0.7	0.23	0.88

Для этого используется ROC (receiver operating characteristic)-кривая. Она состоит из двух метрик — FPR (false positive rate) и TPR (true positive rate).

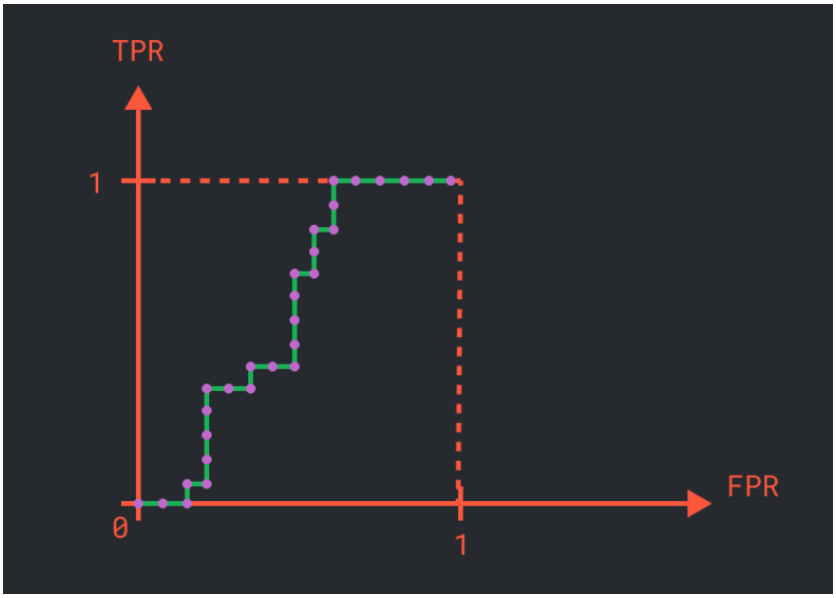
FPR по сути своей является перевернутым precision и находится по формуле $FPR = \frac{false\ positive}{false\ positivetrue\ negative}$. Данная метрика показывает, сколько отрицательных объектов модель неправильно классифицировала как положительные.

TPR по сути своей является recall и находится по формуле $TPR = \frac{true\ positive}{false\ negative+true\ positive}$. Данная метрика показывает, сколько положительных объектов модель классифицировала правильно.

	$y = +1$	$y = -1$
$a(x) = +1$	True Positive	False Positive
$a(x) = -1$	False Negative	True Negative

Данные метрики, аналогично precision и recall, принимают разные значения в зависимости от выбранного порога t. Поэтому мы можем перебрать всевозможные пороги t для построенной модели и получить в итоге множество точек (FPR_i, TPR_i), которые можно нанести на график с $OX = FPR$, $OY = TPR$ и последовательно соединить их между собой. Полученный график и будет являться ROC-кривой.

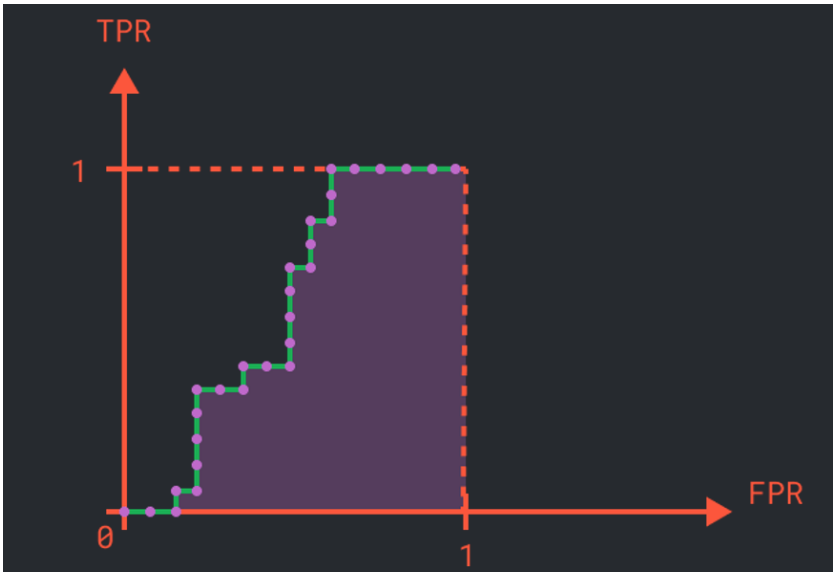
$a(x) = sgn(\frac{1}{1+e^{-\langle \beta, xi \rangle}}$



> ROC-AUC

Как получить из ROC-кривой хорошую метрику?

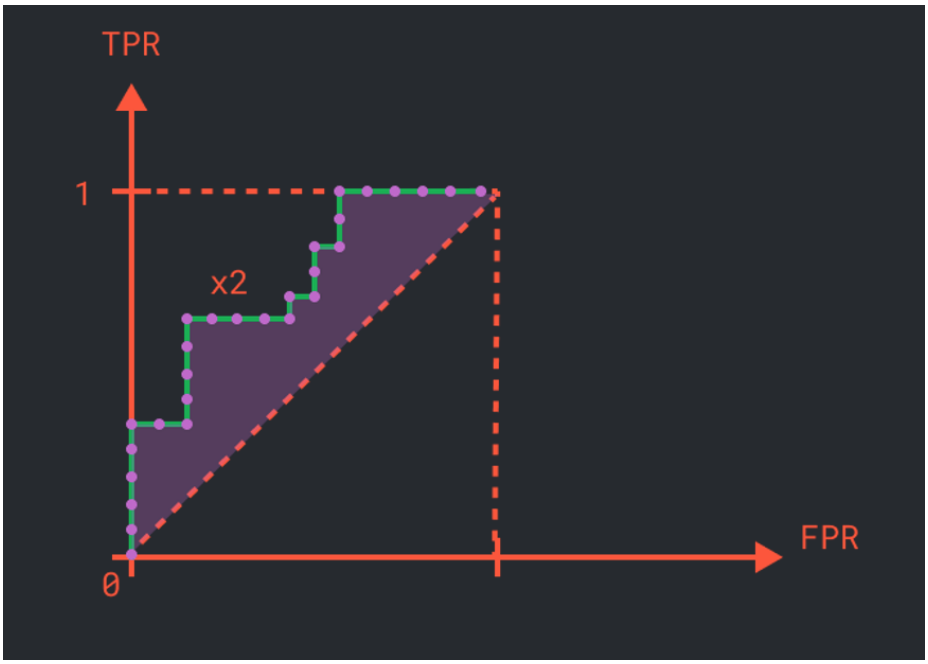
Можно взять площадь под кривой — AUC (Area under the ROC Curve).



Чем больше ROC-AUC модели, тем лучше модель в среднем. У идеальной модели $ROC - AUC = 1$, у плохой она близка к 0, а если $ROC - AUC = 0$, значит модель классифицирует положительный класс как отрицательный и наоборот. Если же $ROC - AUC = 0.5$, значит модель случайным образом размечает классы.

Есть другая вариация ROC-AUC — коэффициент Джини. Метрика была разработана социологом Коррадо Джини и изначально использовалась для оценки степени расслоения общества данной страны или региона по какому-либо изучаемому признаку.

Коэффициент Джини может быть линейно выражен через AUC следующим образом: $Gini = 2 \cdot AUC - 1$



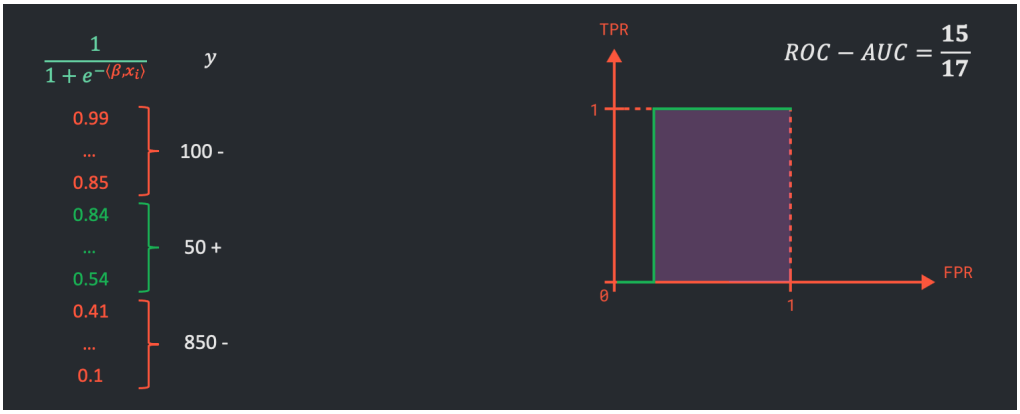
Совершенно неважно, какой из коэффициентов использовать и оптимизировать, так как они линейно зависят друг от друга.

Нахождение ROC-AUC в sklearn

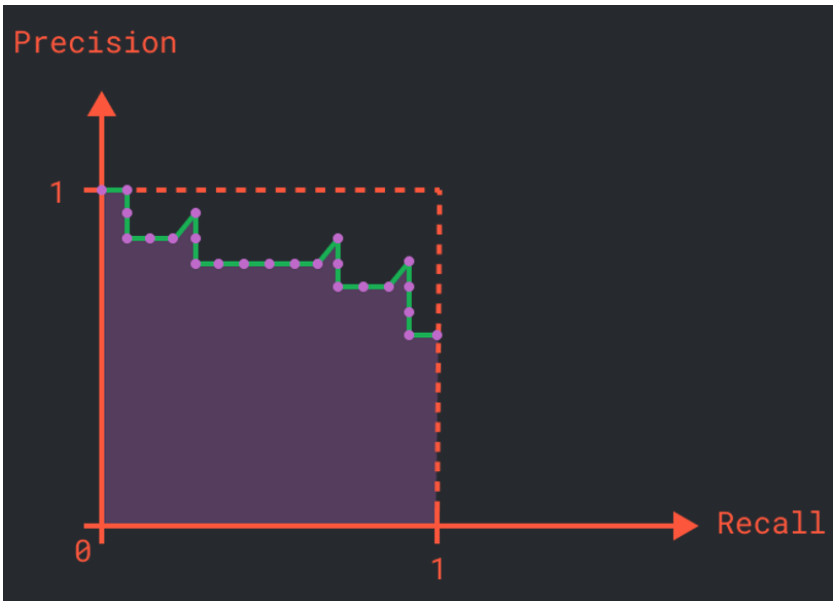
```
# Получим всевозможные пары FPR, TPR

from sklearn.metrics import roc_curve

fpr, tpr, thresholds = roc_curve(Y_test, pipe.predict_proba(X_test)[:, 1])

# И нарисуем ROC-кривую

from sklearn.metrics import RocCurveDisplay

RocCurveDisplay(fpr=fpr, tpr=tpr).plot()

# Посчитаем ROC-AUC

from sklearn.metrics import auc

auc(fpr, tpr)
```

> PR-кривая. Сравнение ROC-кривой и PR-кривой.

Метрика ROC-AUC является чувствительной к несбалансированным классам. Например, если из 1000 объектов положительными являются всего 50 из них, а наша не очень хорошая модель неправильно отнесла к положительному классу еще 100 объектов, то ROC-AUC будет равен $\frac{15}{17}$, что является достаточно высоким значением.



Для работы с несбалансированными классами используется другая кривая — PR (Precision-Recall) -кривая. Она считается аналогично ROC-кривой, только на метриках precision и recall. Аналогично находится и PR-AUC.



Нахождение PR-AUC в sklearn

```
# Получим всевозможные пары Precision, Recall

precision, recall, thresholds = precision_recall_curve(Y_test, pipe.predict_proba(X_test)[:, 1])

# И нарисуем PR-кривую

from sklearn.metrics import PrecisionRecallDisplay

PrecisionRecallDisplay(precision=precision, recall=recall).plot()

# Посчитаем PR-AUC
auc(recall, precision)
```

Резюме:

- ROC-кривая используется тогда, когда главная задача – как можно больше объектов правильно отранжировать с точки зрения вероятностей.
- PR-кривая используется тогда, когда присутствует ситуация дисбаланса классов. Важнее всего для данной кривой является положительный класс.

> Калибровка вероятностей

Обычно мы оцениваем вероятность того, что объект принадлежит к определенному классу, с помощью построения логистической регрессии.

Но что если есть задача найти не логистические (линейные) потери, а например, экспоненциальные или сигмоидные?

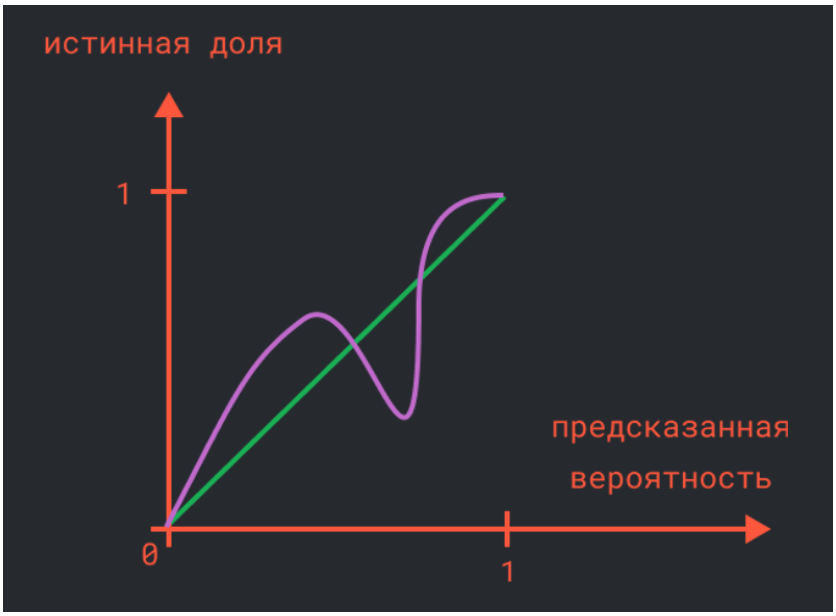
Предположим, что у нас есть 100 объектов, для которых вероятность принадлежности к положительному классу модель оценила в 80%. Сколько, как мы ожидаем, из этих 100 объектов будут отнесены к положительному, а сколько к отрицательному классу?

В идеальной ситуации 80% будут отнесены к положительному классу, а 20% к отрицательному классу. А если для 100 объектов модель оценила вероятность в

50%? Тогда, если ожидаем корректную оценку вероятностей, доля в идеале должна быть 50/50. Аналогично и для всех остальных вероятностей.

Возникает вопрос, как понять, что вероятность, которую предсказывает модель, соответствует реальному распределению классов?

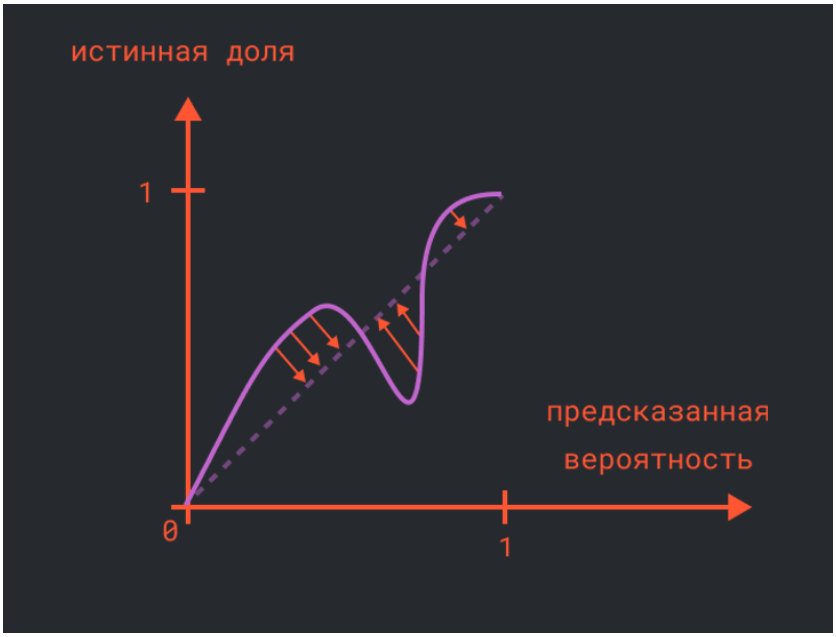
Можно изобразить на графике предсказанную вероятность и истинную долю объектов как множество точек. В идеальном сценарии график будет представлять собой биссектрису угла.



Обычно, конечно, модели оказываются неидеальными, как видно из графика выше.

Как можно откалибровать кривую?

Другими словами, как можно найти такую функцию поверх выходов нашего алгоритма, которая сгладит нашу калибровочную кривую до биссектрисы?



Существует несколько методов для калибровки кривой. Здесь мы рассмотрим один из них, а именно калибровку Платта.

Пусть имеем некоторый алгоритм $a(x)$, который оценивает качество модели. В качестве $a(x)$ можно взять как просто , так и оценку вероятности $p(yi = +1 \mid xi)$.

Рассмотрим калибровку на примере сигмоидной функции $Calibration = \frac{1}{1+e^{b \cdot a(x)+c}}$, где b, c — калибровочные параметры, на которые умножаются выходы алгоритма. Чтобы подобрать параметры данной функции, можно, например, использовать метод максимального правдоподобия.

Таким образом, мы откалибруем кривую и сделаем её более похожей на биссектрису.

Построение калибровочной кривой в sklearn

```
Нарисуем калибровочную кривую для модели LR

from sklearn.calibration import CalibrationDisplay
CalibrationDisplay.from_estimator(pipe, X_test, Y_test)
```