Машинное обучение (Machine Learning)

Функции потерь и показатели качества моделей

Уткин Л.В.

Санкт-Петербургский политехнический университет Петра Великого

Функции потерь

Функции потерь - Loss functions

Определение функции потерь

- Цель задачи МО - научить функцию f, которая преобразует вход, определяемый входным пространством Φ, в желаемый результат, определяемый выходным пространством $\mathcal{Y}$:

$$f : \Phi \to \mathcal{Y}$$

- f может быть аппроксимирована моделью f_Θ с параметрами Θ.
- Дано множество векторов $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_n\} \in \Phi$ множество меток $\mathbf{Y} = \{\mathbf{y}_1, ..., \mathbf{y}_n\} \in \mathcal{Y}$. Функция потерь L определяет близость $f(\mathbf{x}_i)$ и $\mathbf{y}_i$.
- Суммируя по всем $i = 1, ..., n$, получаем

$$\mathcal{L}(f|\mathbf{X}, \mathbf{Y}) = \frac{1}{n}\sum_{i=1}^{n} L(f(\mathbf{x}_i), \mathbf{y}_i)$$

Определение функции потерь

- Суммируя по всем $i = 1, \ldots, n$, получаем

$$\mathcal{L}(f|\mathbf{X}, \mathbf{Y}) = \frac{1}{n}\sum_{i=1}^{n} L(f(\mathbf{x}_i), \mathbf{y}_i)$$

- С регуляризацией

$$\min_f \frac{1}{n}\sum_{i=1}^{n} L(f(\mathbf{x}_i), \mathbf{y}_i) + R(f)$$

- С параметрами

$$\min_\Theta \frac{1}{n}\sum_{i=1}^{n} L(f_\Theta(\mathbf{x}_i), \mathbf{y}_i) + R(\Theta)$$

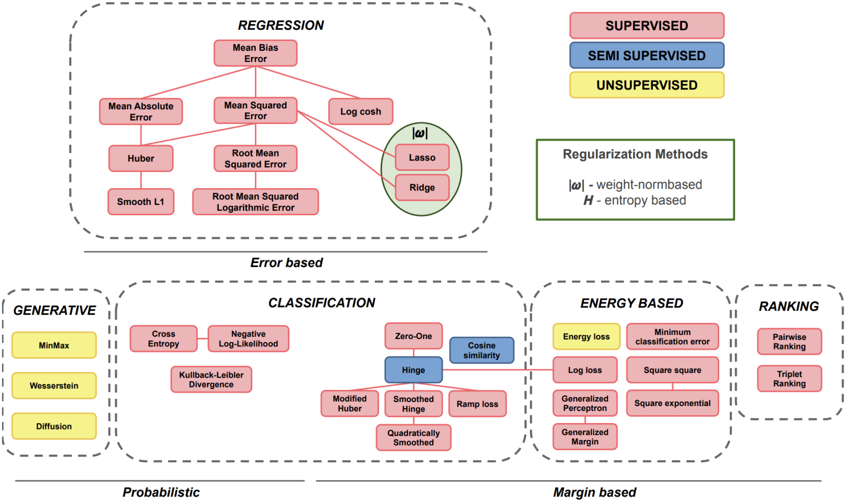
Градиентный алгоритм

Algorithm. Gradient Descent

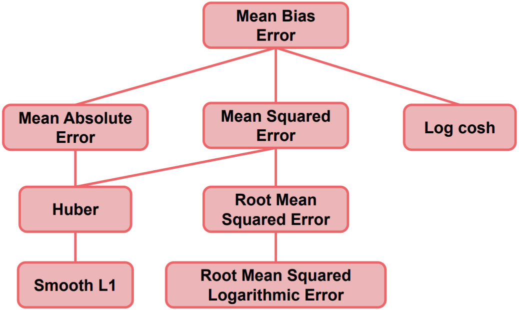
Input: initial parameters $\Theta^{(0)}$, number of iterations T, learning rate α
Output: final learning $\Theta^{(T)}$

1. **for** $t = 0$ **to** $T - 1$
2. estimate $\nabla \mathcal{L}(\Theta^{(t)})$
3. compute $\Delta\Theta^{(t)} = -\nabla \mathcal{L}(\Theta^{(t)})$
4. $\Theta^{(t+1)} := \Theta^{(t)} + \alpha \Delta\Theta^{(t)}$
5. **return** $\Theta^{(T)}$

Виды функций потерь



Функции потерь для регрессии (1)



Функции потерь для регрессии (2)

- **Mean Absolute Error Loss**:

$$\mathcal{L}_{MAE} = \frac{1}{n}\sum_{i=1}^{n} |f(\mathbf{x}_i) - y_i|$$

- **Mean Squared Error Loss**:

$$\mathcal{L}_{MSE} = \frac{1}{n}\sum_{i=1}^{n} (f(\mathbf{x}_i) - y_i)^2$$

- **Root Mean Squared Error Loss**:

$$\mathcal{L}_{MSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (f(\mathbf{x}_i) - y_i)^2}$$

Функции потерь для регрессии (3)

- **Huber loss** - равна MSE, когда ошибка мала $|f(\mathbf{x}_i) - y_i| \le \delta$, иначе получаем MAE:

$$L_{Huber} = \begin{cases} \frac{1}{2}\left(f(\mathbf{x}_i) - y_i\right)^2, & |f(\mathbf{x}_i) - y_i| \le \delta, \\ \delta\left(|f(\mathbf{x}_i) - y_i| - \frac{1}{2}\delta\right), & . \end{cases}$$

- **Log-cosh loss**:

$$\mathcal{L}_{Logcosh} = \frac{1}{n}\sum_{i=1}^{n} \log\left(\cosh\left(f(\mathbf{x}_i) - y_i\right)\right)$$

  - все преимущества потерь Хьюбера без гиперпараметра, но высокие выч. затраты.
  - везде дифференцируема дважды, это плюс, когда, требуется вторая производная.
  - $\log(\cosh(\mathbf{x})) \approx \mathbf{x}^2/2$ при малых $\mathbf{x}$ (MSE), $\log(\cosh(\mathbf{x})) \approx |\mathbf{x}| - \log(2)$ при больших $\mathbf{x}$ (MAE).

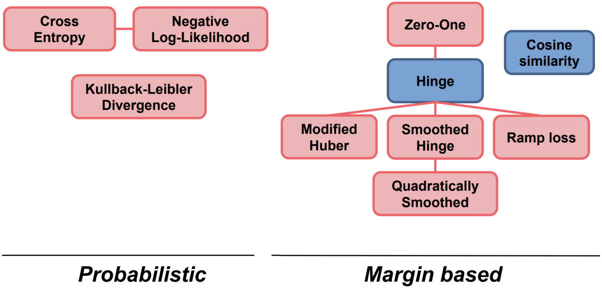
Функции потерь для регрессии (4)

- **Root Mean Squared Logarithmic Error Loss**:

$$\mathcal{L}_{RMSLE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\log(y_i + 1) - \log\left(f(\mathbf{x}_i) + 1\right)\right)^2}$$

  - единственное отличие от RMSE - логарифм применяется как к предсказаниям, так и к меткам
  - плюс 1 внутри логарифма позволяет $f(\mathbf{x}_i)$ быть равными 0
  - из-за свойств логарифма ошибка между $f(\mathbf{x}_i)$ и y_i является относительной, что делает RMSLE более устойчивым к выбросам, т.к. величина RMLSE не масштабируется соответственно величине ошибки, и данные с большими остатками меньше штрафуются, когда $f(\mathbf{x}_i)$ и y_i имеют большие значения.

Функции потерь для классификации



Функции потерь Margin Based (1)

- **Zero-One loss**:

$$L_{ZeroOne}(f(\mathbf{x}), y) = \begin{cases} 1, & f(\mathbf{x}) \cdot y < 0, \\ 0, & . \end{cases}$$

не используется так как невыпуклая и недифференцируема

- **Hinge loss**:

$$L_{Hinge}(f(\mathbf{x}), y) = \max(0, 1 - (f(\mathbf{x}) \cdot y))$$

линейно штрафует каждое предсказание, где результирующее соответствие $<= 1$

- **Perceptron loss:**

$$L_{Perceptron}(f(\mathbf{x}), y) = \max(0, -(f(\mathbf{x}) \cdot y))$$

не штрафует примеры внутри полосы (margin) разделяющей гиперплоскости.

Функции потерь Margin Based (2)

- **Smoothed Hinge loss:**

$$L_{Smooth}(f(\mathbf{x}), y) = \begin{cases} \frac{1}{2} - f(\mathbf{x}) \cdot t, & f(\mathbf{x}) \cdot y \leq 0 \\ \frac{1}{2}\left(1 - (f(\mathbf{x}) \cdot t)\right)^2, & 0 < f(\mathbf{x}) \cdot y < 1 \\ 0, & f(\mathbf{x}) \cdot y \geq 0 \end{cases}$$

сглаженная дифференцируемая версия Hinge loss

- **Quadratically Smoothed Hinge loss:**

$$L_{QSm}(f(\mathbf{x}), y) = \begin{cases} \frac{1}{2\gamma} \max\left(0, -f(\mathbf{x}) \cdot y\right)^2, & f(\mathbf{x}) \cdot y \geq 1 - \gamma \\ 1 - \frac{\gamma}{2} - (f(\mathbf{x}) \cdot y), \end{cases}$$

параметр γ определяет степень сглаженности, при $\gamma \to 0$ функция равна Hinge loss.

Функции потерь Margin Based (3)

- **Ramp loss** или **Truncated Hinge**:

$$L_{Ramp}(f(\mathbf{x}), y) = \begin{cases} L_{Hinge}(f(\mathbf{x}), y), & f(\mathbf{x}) \cdot y \geq -1 \\ 1, & \end{cases}$$

робастная для многоклассовой классификации

- **Cosine Similarity loss**:

$$L_{Cos-Si\,m}(f(\mathbf{x}), y) = 1 - \frac{\mathbf{y} \cdot f(\mathbf{x})}{\|\mathbf{y}\| \, \|f(\mathbf{x})\|}$$

  - когда $\mathbf{y}$ и $f(\mathbf{x})$ - векторы
  - $L_{Cos-Si\,m}(f(\mathbf{x}), y) \in [0, 1]$

Вероятностные функции потерь (Cross Entropy)

- **Cross Entropy loss**:
  - Оценка максимального правдоподобия (MLE).
  - Для датасета $\mathcal{D}$ максимизируем MLE:

$$P(\mathcal{D}|\Theta) = \prod_{i=1}^{n} f_\Theta(\mathbf{x}_i)^{y_i} \left(1 - f_\Theta(\mathbf{x}_i)\right)^{1-y_i}$$

  - переход к логарифму

$$\log P(\mathcal{D}|\Theta) = \sum_{i=1}^{n} (y_i \log f_\Theta(\mathbf{x}_i) + (1 - y_i) \log(1 - f_\Theta(\mathbf{x}_i)))$$

  - Функция потерь (Cross Entropy loss)

$$\mathcal{L}_{NLL} = -\sum_{i=1}^{n} (y_i \log f_\Theta(\mathbf{x}_i) + (1 - y_i) \log(1 - f_\Theta(\mathbf{x}_i)))$$

Вероятностные функции потерь (Cross Entropy)

- Минимизация $\mathcal{L}_{NLL}$ по Θ:

$$\begin{aligned}
\min_{\Theta} \mathcal{L}_{NLL} &= \min_{\Theta} \frac{1}{N} \prod_{i=1}^{n} f_{\Theta}(\mathbf{x}_i)^{n y_i} \\
&= \sum_{i=1}^{n} q(\mathbf{x}_i) \log f_{\Theta}(\mathbf{x}_i) \\
&= -H(q, f_{\Theta})
\end{aligned}$$

- q - распределение вероятностей данных.

Вероятностные функции потерь (Cross Entropy)

- Классический подход - добавление в качестве окончательной активации модели функции softmax, определенной в соответствии с количеством рассматриваемых K классов. Учитывая оценку для каждого класса $f_k(\mathbf{x}) = s$:

$$\widehat{f_k}(\mathbf{x}_i) = f_S(f_k(\mathbf{x})),$$

где

$$f_S(s_i) = \frac{\exp(s_i)}{\sum_{j=1}^{K} \exp(s_j)}.$$

- Отсюда

$$\mathcal{L}_{CCE} = -\frac{1}{K}\sum_{i=1}^{n} \log \widehat{f_k}(\mathbf{x}_i)$$

Вероятностные функции потерь (KL)

- **Kullback-Leibler divergence**

$$KL(g||f_\Theta) = \int q(\mathbf{x}) \log \frac{q(\mathbf{x})}{f_\Theta(\mathbf{x})} \mathrm{d}\mathbf{x}$$
$$= -\int q(\mathbf{x}) \log f_\Theta(\mathbf{x}) \mathrm{d}\mathbf{x} + \int q(\mathbf{x}) \log q(\mathbf{x}) \mathrm{d}\mathbf{x}$$

- Второй интеграл не зависит от Θ:

$$\min_\Theta KL(g||f_\Theta) = \min_\Theta \left(-\int q(\mathbf{x}) \log f_\Theta(\mathbf{x}) \mathrm{d}\mathbf{x} \right)$$
$$= \min_\Theta (-H(q, f_\Theta))$$

Функции потерь ранжирования (ranking loss)

- Цель функции потерь - формировать относительные расстояния между данными, а не учиться предсказывать метку. Это *метрическое обучение (metric learning)*.
- **Pairwise Ranking loss**:
  - используются полож. и негатив. пары обучающих данных
  - положительные пары состоят из якорного примера $\mathbf{x}_a$ и положит. примера $\mathbf{x}_p$, который аналогичен $\mathbf{x}_a$
  - негативные пары состоят из якорного примера $\mathbf{x}_a$ и негативного примера $\mathbf{x}_n$, который не похож на $\mathbf{x}_a$
  - цель - обучить представления с небольшим расстоянием d между ними для положит. пар и большим расстоянием для негативн. пар.

Pairwise Ranking loss

- Даны эмбединги $\mathbf{r}_a$, $\mathbf{r}_p$, $\mathbf{r}_n$ примеров $\mathbf{x}_a$, $\mathbf{x}_p$, $\mathbf{x}_n$ и d - расстояние

$$L_{pairwise} = \begin{cases} d(\mathbf{r}_a, \mathbf{r}_p), & \text{пара положит,} \\ \max(0, m - d(\mathbf{r}_a, \mathbf{r}_n)), & \text{пара отрицат.} \end{cases}$$

- Другое представление:

$$L_{pairwise}(\mathbf{r}_0, \mathbf{r}_1, y) = y \left\|\mathbf{r}_0 - \mathbf{r}_1\right\| + (1 - y) \max(0, m - \left\|\mathbf{r}_0 - \mathbf{r}_1\right\|)$$

$y = 0$ для отрицат. пары, $y = 1$ для положит. пары

Triplet Ranking loss

- Тройки примеров из датасета вместо пар могут повысить точность.
- Тройка состоим из якоря $\mathbf{x}_a$, положительного примера $\mathbf{x}_p$ и отрицательного примера $\mathbf{x}_n$
- Цель - $d(\mathbf{r}_a, \mathbf{r}_n) > d(\mathbf{r}_a, \mathbf{r}_p)$:

$$L_{triplet}(\mathbf{r}_a, \mathbf{r}_p, \mathbf{r}_n) = \max(0, m + d(\mathbf{r}_a, \mathbf{r}_p) - d(\mathbf{r}_a, \mathbf{r}_n))$$

Три варианта Triplet Ranking

- Простая тройка (Easy Triplet): $d(\mathbf{r}_a, \mathbf{r}_n) > d(\mathbf{r}_a, \mathbf{r}_p) + m$: $L_{triplet} = 0$
- Сложная тройка (Hard Triplet): $d(\mathbf{r}_a, \mathbf{r}_n) < d(\mathbf{r}_a, \mathbf{r}_p)$: $L_{triplet} > m$
- Полусложная тройка (Semi-Hard Triplet): $d(\mathbf{r}_a, \mathbf{r}_p) < d(\mathbf{r}_a, \mathbf{r}_n) < d(\mathbf{r}_a, \mathbf{r}_p) + m$: $0 < L_{triplet} < m$

Метрики качества

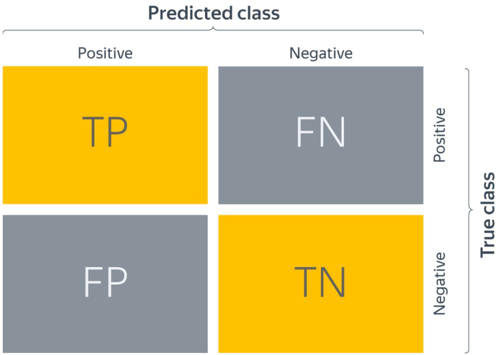
Метрики качества
Классификация

Confusion matrix (матрица ошибок)

Для каждого объекта в выборке возможны 4 ситуации:

- предсказали положительную метку и угадали: true positive (TP) (true - потому что предсказали мы правильно, а positive – потому что предсказали положительную метку);
- предсказали положительную метку, но ошиблись: false positive (FP) (false - потому что предсказание было неправильным);
- предсказали отрицательную метку и угадали: true negative (TN);
- предсказали отрицательную метку, но ошиблись: false negative (FN).

Confusion matrix (матрица ошибок)



Accuracy (точность)

- **Точность:**

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

- **Ошибка** (error rate):

$$ER = 1 - Acc = \frac{FP + FN}{TP + TN + FP + FN}$$

Точность и полнота (precision и recall)

1. Ситуация: пусть FP - доля доброкачественных опухолей, которым ошибочно присваивается метка злокачественной, а FN - доля злокачественных опухолей, которые классификатор пропускает. FN важнее, так как связана с неправильным диагнозом и жизнью
2. Ситуация: рассмотрим задачу: по данным о погоде предсказать, будет ли успешным запуск спутника. FN - это ошибочное предсказание неуспеха, то есть не более, чем упущенный шанс. С FP все серьезней: если предсказать удачный запуск , а он потерпит крушение из-за погоды, то потери в разы существеннее.

Точность и полнота (precision и recall)

3. Ситуация: положительный класс - редкое событие. Пример поисковой системы - в хранилище хранятся миллиарды документов, а релевантных к конкретному поисковому запросу на несколько порядков меньше. Это задача бинарной классификации: “документ d релевантен по запросу q”. Благодаря большому дисбалансу, объявляющего все документы нерелевантными, Accuracy близка к 1, что обеспечено TN, в то время для пользователей более важен высокий TP.

Точность и полнота (precision и recall)

- **Точность** (precision) - доля объектов, названных классификатором положительными, и при этом действительно являющимися положительными:

$$\text{Precision} = \frac{TP}{TP + FP}$$

- Чем меньше ложноположительных срабатываний будет допускать модель, тем больше будет ее Precision.

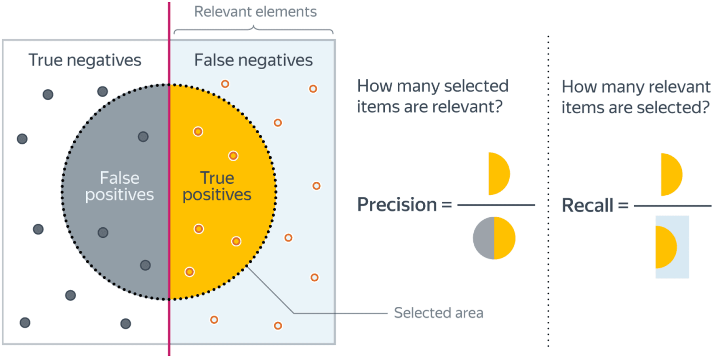
Точность и полнота (precision и recall)

- **Полнота** (recall) - доля объектов положительного класса из всех объектов положительного класса, которые нашел алгоритм:

$$\text{Recall} = \frac{TP}{TP + FN}$$

- Чем меньше ложно отрицательных срабатываний, тем выше Recall модели.
- В задаче предсказания злокачественности опухоли Precision показывает, сколько из определенных нами как злокачественные опухолей действительно являются злокачественными, а Recall - какую долю злокачественных опухолей нам удалось выявить.

Точность и полнота (precision и recall)



F1-мера

- Идея: пару precision и recall скомпоновать, чтобы было одно число, например, взять их среднее гармоническое.
- **F1**-measure (F1-мера):

$$F1 = 2\frac{\text{Recall} \cdot \text{Precision}}{\text{Recall} + \text{Precision}} = \frac{TP}{TP + \frac{FP+FN}{2}}$$

F1-мера - обобщение

- F1-мера предполагает одинаковую важность Precision и Recall.
- Если одна из этих метрик приоритетнее, то можно использовать меру F_β:

$$F_\beta = (\beta^2 + 1)\frac{\text{Recall} \cdot \text{Precision}}{\text{Recall} + \beta^2 \text{Precision}}$$

Чувствительность и специфичность (1)

- **Чувствительность** - доля положительных результатов, которые правильно идентифицированы как таковые. Вероятность того, что больной будет классифицирован именно как больной.
- **Специфичность** отражает долю отрицательных результатов, которые правильно идентифицированы как таковые. Вероятность того, что не больные субъекты будут классифицированы именно как не больные.

Чувствительность и специфичность (2)

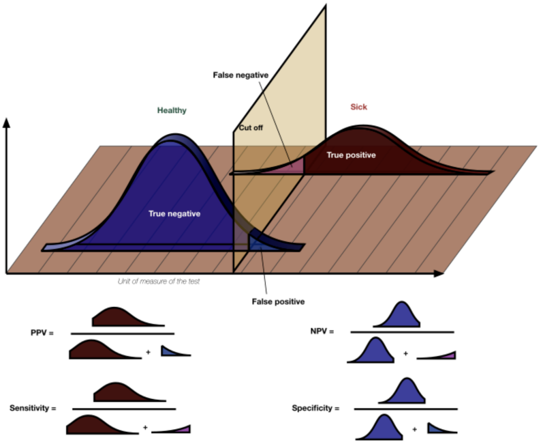
- Чувствительность (true positive rate, recall)

$$TPR = \frac{TP}{P} = \frac{TP}{TP + FN}$$

- Специфичность (true negative rate)

$$TNR = \frac{TN}{N} = \frac{TN}{TN + FP}$$

Чувствительность и специфичность (3)



https://en.wikipedia.org/wiki/Sensitivity_and_specificity

Precision и Recall, Sensitivity и Specificity

- Precision и Recall: рассматривают True Positives (TP)
  - Precision: TP / Predicted positive
  - Recall: TP / Real positive
- Sensitivity и Specificity: рассматривают Correct Predictions.
  - SNIP (SeNsitivity Is Positive): TP / (TP + FN)
  - SPIN (SPecificity Is Negative): TN / (TN + FP)

AUC

- Пусть бинарная классификация с вероятностями классов. Как оценить качество предсказываемых вероятностей (калибровка)?
- При уменьшении порога отсечения находим (правильно предсказываем) все большее число положительных объектов, но также и неправильно предсказываем положительную метку на все большем числе отрицательных объектов.
- Естественным кажется ввести две метрики
- TPR (true positive rate) и FPR (false positive rate)

TPR и FPR

- **TPR** (true positive rate) – это полнота, доля положительных объектов, правильно предсказанных положительными:

$$TPR = \text{Recall} = \frac{TP}{P} = \frac{TP}{TP + FN}$$

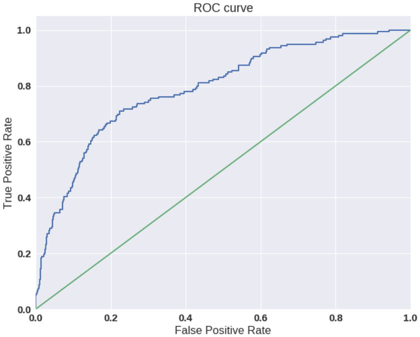
- **FPR** (false positive rate) – это доля отрицательных объектов, неправильно предсказанных положительными:

$$FPR = \frac{FP}{N} = \frac{FP}{FP + TN}$$

- Уменьшая порог, увеличиваем TPR и FPR

ROC curve

- Уменьшая порог, увеличиваем TPR и FPR
- Кривая в осях TPR/FPR, которая получается при варьировании порога, называется **ROC**-кривой (receiver operating characteristics curve, ROC curve)



ROC curve

- Если классификатор идеальный, то получаем ROC-кривую (0,0)->(0,1)->(1,1), площадь под которой равна 1.
- Если классификатор случайный (предсказывает одинаковые метки положительным и отрицательным объектам), то получаем ROC-кривую (0,0)->(1,1), площадь под которой равна 0.5.
- Чем лучше классификатор разделяет два класса, тем больше площадь (area under curve) под ROC-кривой. Эта площадь используется в качестве метрики.

AUC

- **AUC** равен доле пар объектов вида (объект класса 1, объект класса 0), которые алгоритм верно упорядочил, т.е. предсказание классификатора на первом объекте больше

$$AUC = \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} \mathbf{1}[y_i < y_j] I^*[f(\mathbf{x}_i) < f(\mathbf{x}_j)]}{\sum_{i=1}^{n}\sum_{j=1}^{n} \mathbf{1}[y_i < y_j]}$$

$$I^*[f(\mathbf{x}_i) < f(\mathbf{x}_j)] = \begin{cases} 0, & f(\mathbf{x}_i) > f(\mathbf{x}_j) \\ 0.5, & f(\mathbf{x}_i) = f(\mathbf{x}_j) \\ 1, & f(\mathbf{x}_i) < f(\mathbf{x}_j) \end{cases}$$

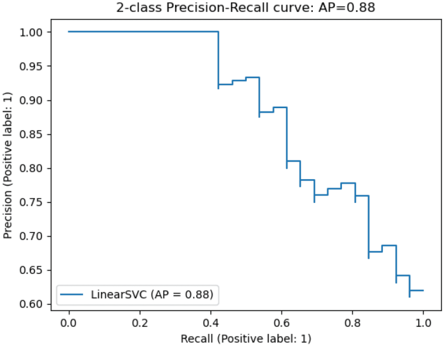
AUC - когда лучше использовать

- В любой задаче, где важна не метка сама по себе, а правильный порядок на объектах, имеет смысл применять AUC.

Average Precision

- Будем постепенно уменьшать порог бинаризации.
- Полнота будет расти от 0 до 1, так как будет увеличиваться количество объектов, которым мы приписываем положительный класс (а количество объектов, на самом деле относящихся к положительному классу, очевидно, меняться не будет).
- Про точность нельзя сказать ничего определенного, но скорее всего она будет выше при более высоком пороге отсечения (мы оставим только объекты, в которых модель “уверена” больше всего).
- Варьируя порог и пересчитывая значения Precision и Recall на каждом пороге, мы получим некоторую кривую примерно следующего вида:

Average Precision



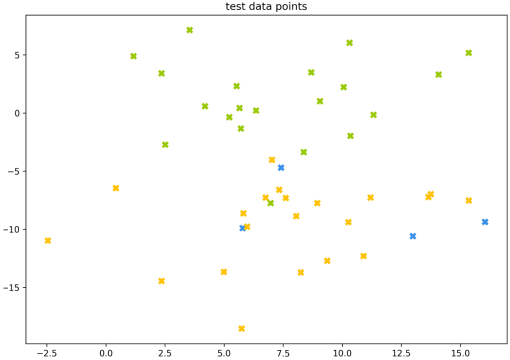
Среднее значение точности AP равно площади под кривой точность-полнота.

Многоклассовая классификация

- K классов: ставится как K задач об отделении класса i от остальных $(i = 1, ..., K)$, для каждой из них можно посчитать свою матрицу ошибок.
- 2 варианта вычисления итоговой метрики:
  - Усредняем элементы матрицы ошибок (TP, FP, TN, FN) между бинарными классификаторами, например $TP = K^{-1} \sum_{i=1}^{K} TP_i$. Затем по одной усредненной матрице ошибок считаем Precision, Recall, F-меру. Это *микроусреднение.*
  - Считаем Precision, Recall для каждого классификатора отдельно, а потом усредняем. Это *макроусреднение*.
- Порядок усреднения влияет на результат в случае дисбаланса классов

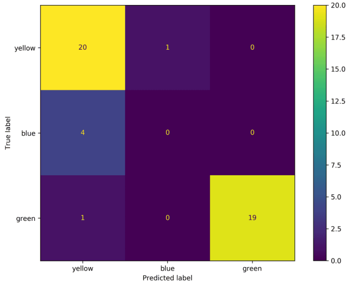
Многоклассовая классификация - пример

- Датасет из объектов трех цветов: желтого, зеленого и синего. Желтого и зеленого цветов почти поровну - 21 и 20 объектов соответственно, а синих объектов 4.



Многоклассовая классификация - пример

- Модель по очереди для каждого цвета пытается отделить объекты этого цвета от объектов других двух цветов. Результаты в матрице ошибок. Модель "покрасила" в желтый 25 объектов, 20 из которых были действ-но желтыми (левый столбец). В синий - только 1 объект, который на самом деле желтый (средний столбец). В зеленый - 19, все на самом деле зеленые (правый столбец).



Многоклассовая классификация - пример

- Precision классификации:
    - микроусреднение

$$\text{Precision} = \frac{(20+0+19)/3}{(20+0+19)/3+(5+1+0)/3} = 0.87$$

    - макроусреднение

$$\text{Precision} = \frac{1}{3}\left(\frac{20}{20+5}+\frac{0}{0+1}+\frac{19}{19+0}\right) = 0.6$$

- макроусреднение лучше отражает тот факт, что синий цвет, которого в датасете совсем мало, модель практически игнорирует.

Метрики качества

Метрики качества
Регрессия

Регрессия - MSE, RMSE, MAE, MAPE

- **MSE** (Mean Squared Error):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(f(\mathbf{x}_i) - y_i)^2$$

- **RMSE** (Root Mean Squared Error): $RMSE = \sqrt{MSE}$
- **MAE** (Mean Absolute Error):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}|f(\mathbf{x}_i) - y_i|$$

- **MAPE** (Mean Absolute Percentage Error):

$$MAPE = 100\% \times \frac{1}{n}\sum_{i=1}^{n}\frac{|f(\mathbf{x}_i) - y_i|}{|y_i|}$$

Регрессия - коэффициент детерминации

- **Коэффициент детерминации** R^2:

$$R^2 = 1 - \frac{\sum_{i=1}^{n} \left(f\left(\mathbf{x}_i\right) - y_i\right)^2}{\sum_{i=1}^{n} \left(\overline{y}_i - y_i\right)^2}$$

- $\overline{y}_i$ - среднее арифметическое меток
- У идеального регрессора $R^2 = 0$.
- R^2 измеряет долю дисперсии, объясненную моделью, в общей дисперсии целевой переменной. Это нормированная среднеквадратичная ошибка.

Метрики качества

Метрики качества
Кластеризация

Внутренние меры оценки качества (1)

- Оценивают качество структуры кластеров опираясь только непосредственно на нее, не используя внешней информации, например о классах.
- **Компактность кластеров** (Cluster Cohesion) - M кластеров, чем ближе друг к другу объекты внутри кластеров, тем лучше разделение:

$$WSS = \sum_{j=1}^{M} \sum_{i=1}^{|C_j|} (x_{ij} - \overline{x}_j)^2 \rightarrow \min$$

- **Отделимость кластеров** (Cluster Separation) - чем дальше друг от друга объекты разных кластеров, тем лучше:

$$BSS = n \sum_{j=1}^{M} (\overline{x} - \overline{x}_j)^2 \rightarrow \max$$

Внутренние меры оценки качества (2)

- Пусть $\delta(C_i, C_j)$ - метрика межкластерного расстояния между кластерами C_i и C_j
- Δ_k - среднее расстояние между парами точек в кластере C_k
- **Индекс Данна** (Dunn Index):

$$DI_m = \frac{\min_{1 \leq i < j \leq M} \delta(C_i, C_j)}{\min_{1 \leq k \leq M} \Delta_k}$$

- Высокий индекс Данна указывает на лучшую кластеризацию

Вопросы

?