Попарный подход

Линейное ранжирование

Recap: SVM для бинарной классификации.

$$x_i \in \mathbb{R}^D,\ y_i \in \{-1, +1\},\ i \in \{1, \dots, n\}$$
$$w \in \mathbb{R}^D,\ w_0 \in \mathbb{R}$$
$$f(x_i) = \text{sign}\left(w^T x_i - w_0\right)$$

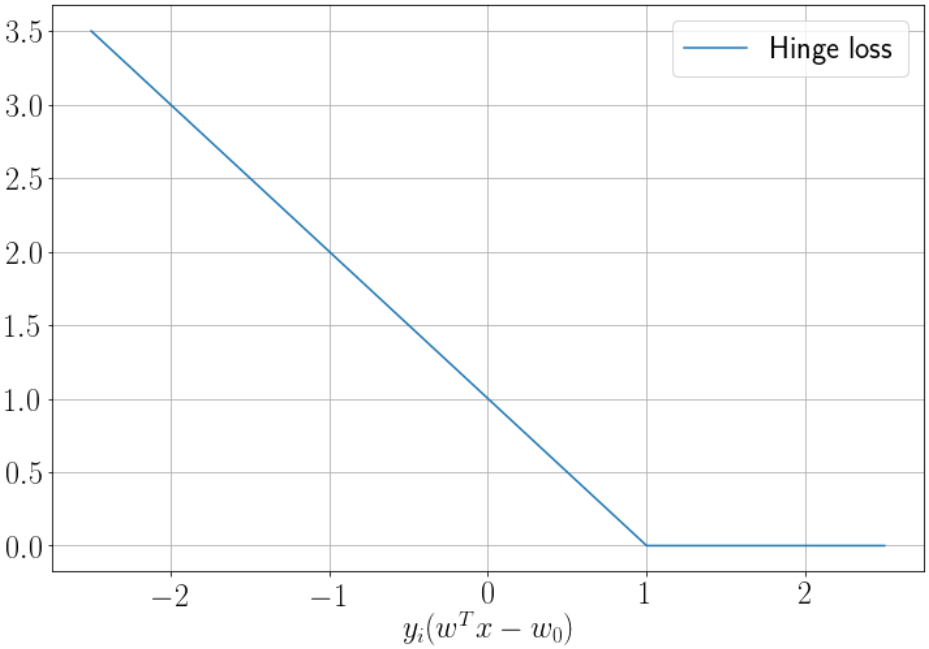
Задача оптимизации с ограничениями:

$$\begin{cases} \frac{1}{2}\|w\|^2 + C\sum\limits_{i=1}^{n}\xi_i \to \min\limits_{w, w_0, \xi} \\ y_i(w^T x_i - w_0) \geq 1 - \xi_i, \quad i \in \{1, \dots, n\} \\ \xi_i \geq 0, \quad i \in \{1, \dots, n\} \end{cases}$$

Та же задача в безусловном варианте:

$$\mathcal{L}(w, w_0, y) = \sum_{i=1}^{n}\max(0, 1 - y_i(w^T x_i - w_0)) + \frac{1}{2C}\|w\|^2 \to \min_{w, w_0}$$

Hinge loss:



Ранжирование линейной моделью:

$$f(x_j^{(i)}) = w^T x_j^{(i)} + w_0$$

Для каждой пары документов $j, k \in \{1, \dots, n^{(i)}\}$ мы хотим, чтобы выражение

$$f(x_j^{(i)}) - f(x_k^{(i)}) = w^T (x_j^{(i)} - x_k^{(i)})$$

имело бы такой же знак, что и $y_j^{(i)} - y_k^{(i)}$. Значит во-первых, нам не важен intercept w_0, во-вторых мы можем искать разделяющую гиперплоскость w в пространстве пар документов, относящихся к одному и тому же запросу.

Формулировка Ranking SVM:

$$f(x_j^{(i)}) = w^T x_j^{(i)}, \quad i \in \{1, \dots, m\}, j \in 1, \dots, n^{(i)}$$

$$\begin{cases} \dfrac{1}{2}\|w\|^2 + C \sum\limits_{i,j,k:\ d_j^{(i)} \prec d_k^{(i)}} \xi_{jk}^{(i)} \to \min\limits_{w,\xi} \\ w^T(x_j^{(i)} - x_k^{(i)}) \geq 1 - \xi_{jk}^{(i)}, \quad i,j,k:\ d_j^{(i)} \prec d_k^{(i)} \\ \xi_{jk}^{(i)} \geq 0, \quad i,j,k:\ d_j^{(i)} \prec d_k^{(i)} \end{cases}$$

Безусловный вариант:

$$\mathcal{L}(w, y) = \sum_{i,j,k: d_j^{(i)} \prec d_k^{(i)}} \max(0, 1 - w^T(x_j^{(i)} - x_k^{(i)})) + \frac{1}{2C}\|w\|^2 \to \min_{w, w_0}$$

Для каждого запроса можно преобразовать выборку $\{x_j^{(i)}, y_j^{(i)}\}_{j=1}^{n^{(i)}}$ в выборку пар:

$$\left\{x_j^{(i)} - x_k^{(i)}, [y_j^{(i)} \prec y_k^{(i)}]\right\}_{i,j,k:\ d_j^{(i)} \prec d_k^{(i)} \vee d_j^{(i)} \succ d_k^{(i)}}$$

Тогда на такой выборке можно обучить любую линейную модель (без intercept), которая будет обладать ранжирующими свойствами.

RankNet, LambdaRank

Для любой пары документов $d_j^{(i)}, d_k^{(i)}$ смоделируем вероятность того, что документ j должен быть выше документа k в ранжированию с помощью выходов алгоритма $s_j^{(i)}, s_k^{(i)}$ как сигмоиду их разницы:

$$\mathbb{P}_{jk}^{(i)} = \mathbb{P}(d_j^{(i)} \prec d_k^{(i)}) = \frac{1}{1 + e^{-\sigma(s_j^{(i)} - s_k^{(i)})}}$$

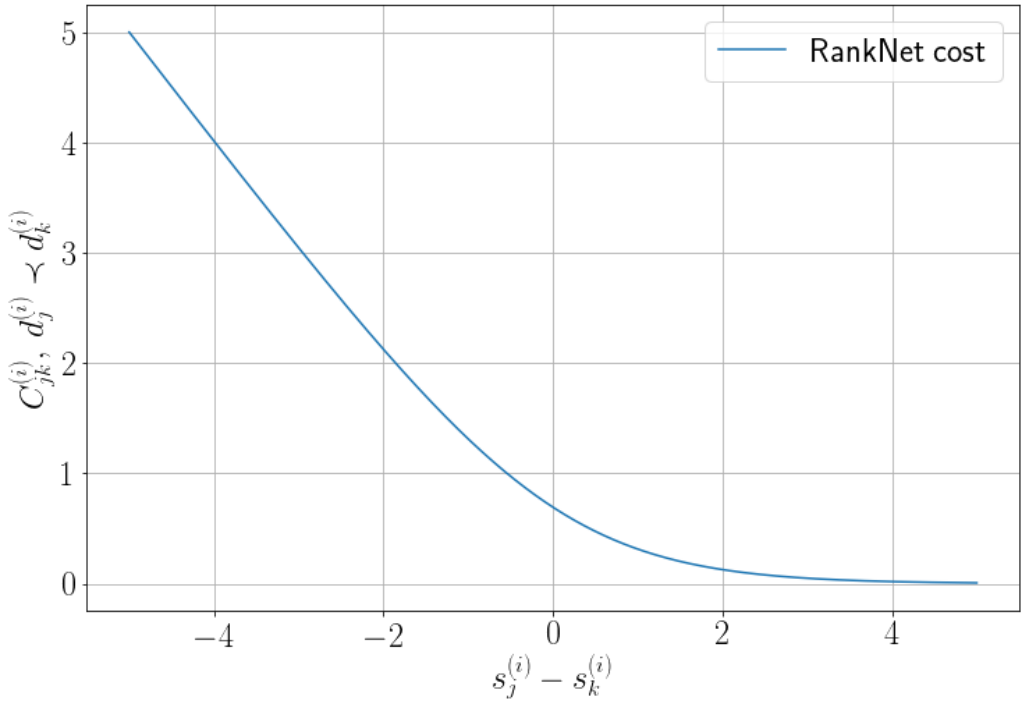
Ground truth этой вероятности обозначим через $\overline{\mathbb{P}}_{jk}^{(i)}$. Будем считать её равной 1 для $d_j^{(i)} \prec d_k^{(i)}$, 0 для $d_j^{(i)} \succ d_k^{(i)}$, и неопределённой для остальных случаев. Тогда для каждой пары, для которой имеет смысл сравнение, можно задать кросс-энтройпийный cost:

$$C_{jk}^{(i)} = -\overline{\mathbb{P}}_{jk}^{(i)} \log \mathbb{P}_{jk}^{(i)} - (1 - \overline{\mathbb{P}}_{jk}^{(i)}) \log(1 - \mathbb{P}_{jk}^{(i)})$$

$$C_{jk}^{(i)} = \begin{cases} \log(1 + e^{-\sigma(s_j^{(i)} - s_k^{(i)})}) & d_j^{(i)} \prec d_k^{(i)} \\ \log(1 + e^{-\sigma(s_k^{(i)} - s_j^{(i)})}) & d_j^{(i)} \succ d_k^{(i)} \end{cases}$$

$$C_{jk}^{(i)} = \log(1 + e^{-\sigma[d_j^{(i)} \prec d_k^{(i)}](s_j^{(i)} - s_k^{(i)})})$$

$$C_{jk}^{(i)} = C_{kj}^{(i)}$$



Общий loss модели для всех пар можно определить через сумму парных costs:

$$\mathcal{L}(\{(s^{(i)}, y^{(i)})\}_{i=1}^{m}) = \frac{1}{m} \sum_{i,j,k:\ d_j^{(i)} \prec d_k^{(i)}} C_{jk}^{(i)} = \frac{1}{m} \sum_{i,j,k:\ d_j^{(i)} \prec d_k^{(i)}} \log(1 + e^{-\sigma(s_j^{(i)} - s_k^{(i)})})$$

Будем использовать этот loss для обучения моделей с помощью градиентного метода. Для этого посчитаем частные производные:

$$\frac{\partial C_{jk}^{(i)}}{\partial s_j^{(i)}} = \frac{-\sigma}{1 + e^{\sigma(s_j^{(i)} - s_k^{(i)})}} = -\frac{\partial C_{jk}^{(i)}}{\partial s_k^{(i)}}; \quad d_j^{(i)} \prec d_k^{(i)}$$

Если мы моделируем нашу ранжирующую функцию нейросетью $f(x_j^{(i)}, w)$, то апдейт весов для батча из одного запроса $q^{(i)}$:

$$\frac{\partial \mathcal{L}(s^{(i)}, y^{(i)})}{\partial w} = \sum_{d_j^{(i)} \prec d_k^{(i)}} \frac{\partial C_{jk}^{(i)}}{\partial w} =$$

$$= \sum_{d_j^{(i)} \prec d_k^{(i)}} \left(\frac{\partial C_{jk}^{(i)}}{\partial s_j^{(i)}} \cdot \frac{\partial s_j^{(i)}}{\partial w} + \frac{\partial C_{jk}^{(i)}}{\partial s_k^{(i)}} \cdot \frac{\partial s_k^{(i)}}{\partial w} \right) = \sum_{d_j^{(i)} \prec d_k^{(i)}} \frac{\partial C_{jk}^{(i)}}{\partial s_j^{(i)}} \left(\frac{\partial s_j^{(i)}}{\partial w} - \frac{\partial s_k^{(i)}}{\partial w} \right) =$$

$$\sum_{d_j^{(i)} \prec d_k^{(i)}} \frac{-\sigma}{1 + e^{\sigma(s_j^{(i)} - s_k^{(i)})}} \left(\frac{\partial s_j^{(i)}}{\partial w} - \frac{\partial s_k^{(i)}}{\partial w} \right)$$

Нейросеть с такой функцией потерь называют моделью RankNet https://www.microsoft.com/en-us/research/wp-content/uploads/2005/08/icml_ranking.pdf.

Перепишем градиент в виде:

$$\frac{\partial \mathcal{L}}{\partial w} = \sum_{d_j^{(i)} \prec d_k^{(i)}} \lambda_{jk}^{(i)} \left(\frac{\partial s_j^{(i)}}{\partial w} - \frac{\partial s_k^{(i)}}{\partial w} \right); \quad \lambda_{jk}^{(i)} \equiv \frac{-\sigma}{1 + e^{\sigma(s_j^{(i)} - s_k^{(i)})}}$$

$$\frac{\partial \mathcal{L}}{\partial w} = \sum_j \lambda_j^{(i)} \frac{\partial s_j^{(i)}}{\partial w}; \quad \lambda_j^{(i)} \equiv \sum_{k:\, d_j^{(i)} \prec d_k^{(i)}} \lambda_{jk}^{(i)} - \sum_{k:\, d_j^{(i)} \succ d_k^{(i)}} \lambda_{kj}^{(i)}$$

Fig. 1 A set of urls ordered for a given query using a binary relevance measure. The light gray bars represent urls that are not relevant to the query, while the dark blue bars represent urls that are relevant to the query. Left: the total number of pairwise errors is thirteen. Right: by moving the top url down three rank levels, and the bottom relevant url up five, the total number of pairwise errors has been reduced to eleven. However for IR measures like NDCG and ERR that emphasize the top few results, this is not what we want. The (black) arrows on the left denote the RankNet gradients (which increase with the number of pairwise errors), whereas what we'd really like are the (red) arrows on the right.

Для того, чтобы "сила", прилагаемая к каждому документу больше отражала, интересующую нас метрику ранжирования, а именно делала больший акцент на начало списка, предлагается подхачить градиент эвристикой https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/lambdarank.pdf:

$$\lambda_{jk}^{(i)} \equiv \frac{-\sigma}{1 + e^{\sigma(s_j^{(i)} - s_k^{(i)})}} \cdot |\Delta Z_{jk}^{(i)}|$$

Где $\Delta Z_{jk}^{(i)}$ это разница в целевой ранжирующей метрике, которая получается, если переставить j-й и k-й документы местами:

$$|\Delta Z_{jk}^{(i)}| = |Z(s_{j \leftrightarrow k}^{(i)}, y^{(i)}) - Z(s^{(i)}, y^{(i)})|$$

На примере nDCG:

$$|\Delta \text{nDCG}_{jk}^{(i)}| = \Big| \frac{2^{y_j^{(i)}} - 2^{y_k^{(i)}}}{\text{IDCG}^{(i)}} \Big(\frac{1}{\log_2(t_j^{(i)} + 1)} - \frac{1}{\log_2(t_k^{(i)} + 1)} \Big) \Big|$$

LambdaMART

LambdaMART – это просто применение функции потерь RankNet в градиентном бустинге над решающими деревьями. Для использования ф-и потерь в таких моделях помимо градиентов нам нужны вторые производные по выходам модели. Для их вычисления определим суррогатный loss, в котором мы "притворимся", что $|\Delta Z_{jk}^{(i)}|$ не зависит от выходов модели, а просто является весом для пары документов $d_j^{(i)}, d_k^{(i)}$:

$$\mathcal{L}(\{(s^{(i)}, y^{(i)})\}_{i=1}^m) = \frac{1}{m} \sum_{i,j,k:\ d_j^{(i)} \prec d_k^{(i)}} |\Delta Z_{jk}^{(i)}| \log(1 + e^{-\sigma(s_j^{(i)} - s_k^{(i)})})$$

Тогда:

$$\frac{\partial \mathcal{L}}{\partial s_j^{(i)}} = \lambda_j^{(i)} = \sum_{d_j^{(i)} \prec d_k^{(i)}} \frac{-\sigma |\Delta Z_{jk}^{(i)}|}{1 + e^{\sigma(s_j^{(i)} - s_k^{(i)})}} - \sum_{d_j^{(i)} \succ d_k^{(i)}} \frac{-\sigma |\Delta Z_{jk}^{(i)}|}{1 + e^{\sigma(s_k^{(i)} - s_j^{(i)})}} =$$

$$= \sum_{d_j^{(i)} \prec d_k^{(i)}} -\sigma |\Delta Z_{jk}^{(i)}| \rho_{jk}^{(i)} - \sum_{d_j^{(i)} \succ d_k^{(i)}} -\sigma |\Delta Z_{jk}^{(i)}| \rho_{kj}^{(i)};$$

$$\rho_{jk}^{(i)} = \frac{-\lambda_{jk}^{(i)}}{\sigma |Z_{jk}^{(i)}|} = \frac{1}{1 + e^{\sigma(s_j^{(i)} - s_k^{(i)})}} = 1 - \rho_{kj}^{(i)}$$

$$\frac{\partial^2 \mathcal{L}}{\partial (s_j^{(i)})^2} = \sum_{d_j^{(i)} \prec d_k^{(i)}} \sigma^2 |\Delta Z_{jk}^{(i)}| \rho_{jk}^{(i)} (1 - \rho_{jk}^{(i)}) + \sum_{d_j^{(i)} \succ d_k^{(i)}} \sigma^2 |\Delta Z_{jk}^{(i)}| \rho_{kj}^{(i)} (1 - \rho_{kj}^{(i)}) =$$

$$= 2 \sum_{d_j^{(i)} \prec d_k^{(i)}} \sigma^2 |\Delta Z_{jk}^{(i)}| \rho_{jk}^{(i)} (1 - \rho_{jk}^{(i)})$$