

# Some Notes

:)

Updated on 2025-11-11

## 1 DS/ML 面试问题准备

## 2 Definition of 峰度 (Kurtosis) & 偏度 (Skewness)

### 2.1 Kurtosis ( $K$ )

Kurtosis describes the "tailedness" of the distribution. Mathematically:

$$K = \frac{\mathbb{E}[(X - \mu)^4]}{\sigma^4} - 3$$

where  $\mu$  is the mean,  $\sigma$  is the standard deviation.

- $K < 0$ : Flatter than the normal distribution (platykurtic).
- $K > 0$ : More peaked or thinner-tailed than the normal distribution (leptokurtic).
- $K \approx 0$ : Close to normal distribution (mesokurtic).

### 2.2 Skewness ( $S$ )

Skewness measures the asymmetry of the distribution:

$$S = \frac{\mathbb{E}[(X - \mu)^3]}{\sigma^3}$$

- $S > 0$ : Right-skewed (longer tail on the right, peak shifted left).
- $S < 0$ : Left-skewed (longer tail on the left, peak shifted right).
- $S \approx 0$ : Close to normal distribution.

Measure	Sign	Interpretation
Kurtosis $K$	$< 0$	Flatter than normal
	$> 0$	Peaked/thinner than normal
	$\approx 0$	Close to normal
Skewness $S$	$> 0$	Right-skewed (tail right, peak left)
	$< 0$	Left-skewed (tail left, peak right)
	$\approx 0$	Close to normal

表 1: Summary of Kurtosis and Skewness

## 2.3 Summary Table

## 3 Bias–Variance Tradeoff

**Question:** What is the bias–variance tradeoff?

**Answer:**

The bias–variance tradeoff describes how model complexity affects prediction error. High bias means the model is too simple and underfits, while high variance means the model is too complex and overfits. For example, in a recommendation model, a shallow collaborative filtering model may have high bias, whereas a deep neural model without regularization can have high variance. I usually balance them by cross-validation, regularization, and early stopping.

中文关键词辅助记忆:

- 偏差-方差权衡 (Bias–Variance Tradeoff)
- 高偏差→ 欠拟合 (High Bias → Underfitting)
- 高方差→ 过拟合 (High Variance → Overfitting)
- 推荐系统例子: 浅层协同过滤 (Shallow Collaborative Filtering)
- 推荐系统例子: 深度神经网络无正则化 (Deep NN w/o Regularization)
- 平衡策略: 交叉验证、正则化、提前停止 (Cross-validation, Regularization, Early Stopping)

## 4 Choosing a Loss Function for a Ranking Model

**Question:** How would you choose a loss function for a ranking model?

**Answer:**

It depends on the objective. Pointwise losses (like MSE or logistic loss) optimize individual relevance scores. Pairwise losses (like hinge loss or pairwise logistic loss) focus on the ordering between two items. Listwise losses (like softmax cross-entropy or LambdaRank) optimize the entire ranked list. For search and recommendation ranking, I often prefer pairwise or listwise losses since they directly reflect user behavior and ranking metrics such as NDCG.

### Definitions of Loss Functions:

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

$$L_{\text{MSE}} = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2$$

- Logistic Loss (for binary relevance):

$$L_{\text{logistic}} = -\frac{1}{N} \sum_{i=1}^N [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)]$$

- **Pairwise Losses:** - Hinge Loss:

$$L_{\text{hinge}} = \sum_{(i,j)} \max(0, 1 - (\hat{y}_i - \hat{y}_j))$$

- Pairwise Logistic Loss:

$$L_{\text{pairwise-logistic}} = \sum_{(i,j)} \log(1 + \exp(-(\hat{y}_i - \hat{y}_j)))$$

- **Listwise Losses:** - Softmax Cross-Entropy (over a list):

$$L_{\text{listwise}} = -\sum_{i=1}^N y_i \log \frac{\exp(\hat{y}_i)}{\sum_j \exp(\hat{y}_j)}$$

- LambdaRank (simplified):

$$L_{\text{LambdaRank}} = \sum_{(i,j)} |\Delta \text{NDCG}_{ij}| \cdot \log(1 + \exp(-(\hat{y}_i - \hat{y}_j)))$$

### How to judge model performance:

- Metrics closer to 1 (like NDCG@k, Precision@k, MAP) indicate better ranking quality.
- Loss value: lower loss generally indicates better fit to the objective.

### 中文关键词辅助记忆:

- 目标导向 (Depends on Objective)
- Pointwise → 优化单个相关性 (Individual relevance)

- Pairwise  $\rightarrow$  优化两两排序 (Pairwise ordering)
- Listwise  $\rightarrow$  优化整个列表 (Entire ranked list)
- 推荐系统偏好 Pairwise/Listwise (Reflect user behavior, NDCG)
- 模型评价: NDCG、Precision、MAP 越接近 1 越好 (Closer to 1  $\rightarrow$  better)
- Loss 越小越好 (Lower loss  $\rightarrow$  better)

## 5 Feature Leakage and How to Detect It

**Question:** Explain feature leakage and how to detect it.

**Answer:**

Feature leakage happens when information from the future or from the target leaks into the training features, giving the model unrealistic predictive power. For example, including “number of clicks in the next week” when predicting current engagement can create leakage. I usually check for leakage by using time-based validation, performing feature correlation analysis with the label, and simulating the real-time feature generation process to ensure features only include information available at prediction time.

**How to detect feature leakage:**

- **Time-based validation:** Ensure that training data only contains past information relative to validation/test sets.
- **Feature correlation analysis:** Check correlations between features and the target; unusually high correlations may indicate leakage.
- **Simulate real-time feature generation:** Recreate the process of feature computation in production to confirm that no future information is included.

**中文关键词辅助记忆:**

- 特征泄漏 (Feature Leakage)
- 来自未来或目标的信息泄漏 (Future/Target info leaks)
- 模型预测能力不真实 (Unrealistic predictive power)
- 例子: 预测当前活跃度时用 “下周点击量” (Next-week clicks)
- 检测方法:
  - 时间切分验证 (Time-based validation)

- 特征与标签相关性分析 (Feature-label correlation)
- 模拟实时特征生成流程 (Simulate real-time feature generation)