DS/ML 面试问题准备

:)

Updated on 2025-11-12

1 Definition of 峰度 (Kurtosis) & 偏度 (Skewness)

1.1 Kurtosis (K)

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

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

where μ is the mean, σ 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).

1.2 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.

1.3 Summary Table

1.4 中文辅助记忆点

- **Kurtosis (峰度)**: 描述分布尾部的尖锐程度或平坦程度

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

表 1: Summary of Kurtosis and Skewness

  - $K < 0$: 平坦 (platykurtic) → 尾部轻，峰低
  - $K > 0$: 尖锐/尾重 (leptokurtic) → 尾部重，峰高
  - $K \approx 0$: 接近正态分布 (mesokurtic)

- **Skewness (偏度)**: 描述分布左右偏斜
  - $S > 0$: 右偏 (tail right, peak left)
  - $S < 0$: 左偏 (tail left, peak right)
  - $S \approx 0$: 接近对称

- **记忆小技巧**:
  - 峰度想“高低胖瘦” → 尾部轻重 峰高低
  - 偏度想“左右倾斜” → 尾部长在哪边

2 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)

3 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 → 优化两两排序 (Pairwise ordering)
- Listwise → 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 → better)
- Loss 越小越好 (Lower loss → better)

4 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)

5 Handling Cold-Start Problems in Recommendation Systems

Question: How do you handle cold-start problems in recommendation systems?

Answer:

I handle cold-start problems using content-based or hybrid approaches. For new users, I rely on demographic or session-based features, while for new items, I use item metadata embeddings such as category or description text. Once enough interactions accumulate, I switch to collaborative signals. In production, we often blend cold-start scores and collaborative filtering (CF) scores using a weighted strategy to ensure smooth recommendations.

Common Strategies:

- **New users:** Use demographic data (age, location) or session behaviors to estimate preferences.
- **New items:** Leverage item metadata, embeddings from categories, descriptions, or textual features.
- **Transition:** Gradually incorporate collaborative signals as user-item interactions grow.
- **Production blending:** Combine cold-start model and CF-based model with weighted averaging or ensemble.

中文关键词辅助记忆:

- 冷启动问题 (Cold-Start Problem)
- 新用户→ 人口统计/会话特征 (New users → Demographic/Session features)

- 新物品→ 元数据嵌入 (New items → Metadata embeddings)
- 后期→ 协同过滤信号 (Later → Collaborative signals)
- 生产环境→ 加权融合冷启动 CF (Production → Weighted blend Cold-start CF)

6 Evaluating Ranking Models

Question: How do you evaluate ranking models?

Answer:

I use metrics that reflect ranking quality, such as NDCG@K, MAP, Recall@K, and Precision@K. NDCG is my favorite because it considers both relevance and position. Offline evaluation is typically followed by A/B testing to validate real-world performance, such as CTR or conversion uplift. Additionally, confusion tables can help evaluate ranking-related classification tasks if items are binarized as relevant/non-relevant.

Definitions and Formulas:

- **Confusion Table (for binary relevance):**

	Predicted Relevant	Predicted Non-Relevant
Actual Relevant	True Positive (TP)	False Negative (FN)
Actual Non-Relevant	False Positive (FP)	True Negative (TN)

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

- **TPR / FPR (True Positive Rate / False Positive Rate):**

$$\text{TPR} = \frac{TP}{TP+FN}, \quad \text{FPR} = \frac{FP}{FP+TN}$$

- **NDCG@K (Normalized Discounted Cumulative Gain):**

$$\text{DCG@K} = \sum_{i=1}^{K} \frac{2^{rel_i}-1}{\log_2(i+1)}, \quad \text{NDCG@K} = \frac{\text{DCG@K}}{\text{IDCG@K}}$$

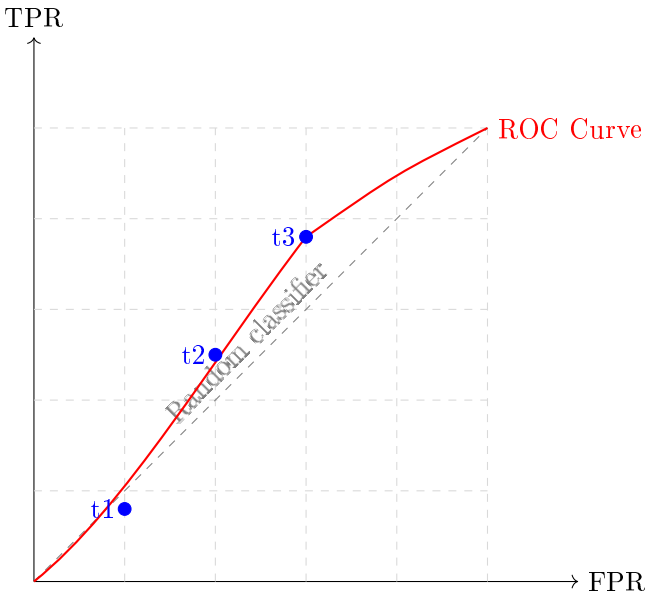
  where rel_i is the relevance of item i and IDCG@K is the ideal DCG.

- **MAP (Mean Average Precision):**

$$\text{AP} = \frac{\sum_{k=1}^{N} P(k) \cdot rel(k)}{\text{number of relevant items}}, \quad \text{MAP} = \frac{1}{M}\sum_{j=1}^{M} \text{AP}_j$$

where $P(k)$ is the precision at position k, $rel(k)$ is 1 if relevant, else 0, and M is the number of queries.

ROC Curve (示意图):



中文关键词辅助记忆:

- 排序指标 (Ranking metrics)
- NDCG@K → 考虑相关性 排名位置
- MAP → 平均精度
- Recall@K, Precision@K → 前K命中率 精准率
- 混淆表 (Confusion Table) → TP, FP, TN, FN
- Accuracy → 准确率, Precision → 精确率, Recall → 召回率, F1 → 综合指标
- TPR / FPR → 真阳性率 / 假阳性率
- ROC 曲线→ 评估二分类排序能力
- 离线评估→ A/B 测试验证 CTR/Conversion

7 Designing Features for User Behavior Logs

Question: How do you design features for user behavior logs?

Answer:

I start by defining the prediction target — for example, next-click or purchase intent. Then I extract user-level, item-level, and contextual features such as recency, frequency, dwell time, device type, etc. For time-related data, I use time decay or sequence embedding to capture temporal patterns. I also ensure feature freshness by automating updates via data pipelines.

Definitions of Key Techniques:

- **Time Decay:** Weights more recent user actions higher to reflect current interests. A common formulation:

  $$w_i = e^{-\lambda \Delta t_i}$$

  where Δt_i is the time elapsed since action i, and λ is the decay rate.

- **Sequence Embedding:** Represents a sequence of user actions as a dense vector, capturing temporal patterns and dependencies. Example using embedding function f:

  $$\mathbf{v}_u = f(a_1, a_2, ..., a_T)$$

  where a_t are sequential actions, and $\mathbf{v}_u$ is the user sequence embedding.

中文关键词辅助记忆:

- 特征设计目标 (Define prediction target)
- 用户级特征 (User-level features)
- 物品级特征 (Item-level features)
- 上下文特征 (Contextual features) → recency, frequency, dwell time, device
- 时间相关特征 (Time-related features)
  - Time Decay → 最近行为权重更高 (Recent actions weighted more)
  - Sequence Embedding → 行为序列向量表示 (Sequence of actions → dense vector)
- 特征更新自动化 (Feature freshness via pipelines)

8 Handling Large Categorical Features

Question: How do you handle large categorical features like millions of product IDs?

Answer:

I usually use embedding representations instead of one-hot encoding. For high-cardinality IDs, we can apply frequency thresholding, hashing, or learn embeddings via models like Word2Vec or DeepFM. This approach not only reduces dimensionality but also captures semantic relationships between items.

Key Techniques:

- **Embedding Representations:** Map each categorical ID to a dense vector in lower-dimensional space.

$$\mathbf{v}_i \in \mathbb{R}^d \quad \text{for each item } i$$

- **Frequency Thresholding:** Replace rare IDs with an "other" category to reduce sparsity.
- **Hashing:** Map IDs into a fixed-size hash space to avoid huge one-hot vectors.
- **Learned Embeddings:** Use models like Word2Vec, DeepFM, or other neural networks to learn embeddings that capture item relationships.

中文关键词辅助记忆:

- 大类目特征 (Large categorical features)
- One-hot 编码太大 (One-hot too sparse/high-dimensional)
- Embedding → 降维 + 捕捉语义关系 (Dimensionality reduction + semantic capture)
- 高基数处理方法 (High-cardinality methods):
  - 频次阈值 (Frequency thresholding)
  - 哈希 (Hashing)
  - 学习型嵌入 (Learned embeddings, e.g., Word2Vec, DeepFM)

9 Handling Models that Perform Poorly Online Despite Good Training

Question: What’s your approach when your model trains well but performs poorly online?

Answer:

I would first check for data mismatch between training and serving (feature drift). Then I’d verify that offline metrics align with online KPIs and ensure the A/B test setup is valid. Often the issue lies in feature staleness, data leakage, distributional shift, or model overfitting.

Potential Causes and Checks:

- **Feature Drift:** The distribution of features changes between training and serving.

$$P_{\text{train}}(X) \neq P_{\text{serving}}(X)$$

- **Data Leakage:** Information from the future or the target leaks into training features, giving unrealistic predictive power.
- **Distributional Shift:** Overall change in input or target distribution between training and deployment.
- **Model Overfitting:** Model fits training data too well but generalizes poorly.
- **Offline vs Online Misalignment:** Offline metrics may not fully reflect online KPIs like CTR or conversion.

中文关键词辅助记忆:

- 训练好但线上差 (Good training, poor online)
- 数据不匹配→ 特征漂移 (Feature drift)
- 特征过期 (Feature staleness)
- 数据泄漏 (Data leakage)
- 分布变化 (Distributional shift)
- 模型过拟合 (Overfitting)
- 离线指标与线上 KPI 对齐 (Offline vs Online metric alignment)
- A/B 测试验证是否正确 (Validate A/B test setup)

10 Monitoring Model Degradation or Drift in Production

Question: How do you monitor model degradation or drift in production?

Answer:

I monitor both input feature drift (using statistical tests like KS divergence) and output drift (changes in prediction distributions). I also track business metrics such as CTR, conversion, and latency. When drift exceeds predefined thresholds, we trigger retraining or human review through the MLOps pipeline.

Key Concepts and Definitions:

- **Feature Drift / Input Drift:** Changes in the input feature distribution over time. Can be measured using tests like Kolmogorov–Smirnov (KS) divergence.

$$D_{KS} = \sup_x |F_{\text{train}}(x) - F_{\text{prod}}(x)|$$

- **Output Drift / Prediction Drift:** Changes in the distribution of model predictions, which may indicate performance degradation.
- **Conversion:** A business metric indicating successful user action (e.g., purchase, signup).

$$\text{Conversion Rate} = \frac{\text{Number of Conversions}}{\text{Number of Impressions or Visitors}}$$

- **Latency:** Time taken for the model to generate predictions, usually measured in milliseconds.

中文关键词辅助记忆:

- 监控模型退化/漂移 (Monitor degradation/drift)
- 特征漂移 (Feature/Input drift) → KS 检验
- 输出漂移 (Output/Prediction drift)
- 业务指标 (Business metrics) → CTR, 转化率 (Conversion), 延迟 (Latency)
- 阈值触发 (Threshold) → 重新训练或人工审核
- MLOps 流程 (MLOps pipeline)

11 Designing and Interpreting an A/B Test for a Recommendation Model

Question: How do you design and interpret an A/B test for a recommendation model?

Answer:

I split traffic randomly into control and treatment groups, ensuring similar user distributions. Key metrics include CTR, conversion rate, dwell time, etc. I calculate p-values and confidence intervals to test statistical significance. If the uplift is consistent across segments and significant, we proceed to rollout. Otherwise, I check the experiment setup, seasonality, and sample size.

中文关键词辅助记忆:

- 随机分流 (Random traffic split) → control vs treatment
- 保证用户分布相似 (Ensure similar user distributions)
- 关键指标 (Key metrics) → CTR, 转化率 (Conversion rate), 停留时间 (Dwell time)
- 统计显著性 (Statistical significance) → p-value, 置信区间 (Confidence interval)
- 分段一致性 (Uplift consistent across segments)
- 若不显著→ 检查实验设置、季节性、样本量 (Check setup, seasonality, sample size)

12 P-test and Z-test Definitions and Relationship

12.1 Z-test

Z-test is a statistical test used to determine if there is a significant difference between sample mean(s) and population mean (or between two sample means) when the population variance is known.

Formula (one-sample Z-test):

$$Z = \frac{\bar{X} - \mu_0}{\sigma/\sqrt{n}}$$

where

- $\bar{X}$ = sample mean
- μ_0 = population mean
- σ = population standard deviation
- n = sample size

12.2 P-value (P-test)

P-value is the probability of observing a test statistic at least as extreme as the one observed, under the null hypothesis H_0. It quantifies the evidence against H_0.

Decision rule:

- If $p \leq \alpha$ (significance level), reject H_0.
- If $p > \alpha$, fail to reject H_0.

12.3 Relationship between Z-test and P-value

- Z-test produces a test statistic Z.
- The P-value is computed from Z as the probability of observing a value as extreme or more extreme under H_0:

$$p = 2 \cdot P(Z \geq |z_{\text{obs}}|) \quad \text{(two-tailed)}$$

- Therefore, Z-test provides the test statistic, and P-test (P-value) provides the significance measure.

12.4 中文关键词辅助记忆

- Z检验 (Z-test) → 样本均值与总体均值差异检验 (Known population variance)
- P值 (P-value) → 在零假设下观测极端值的概率 (Evidence against H_0)
- 决策规则 (Decision rule) → $p \leq \alpha$ 拒绝 H_0
- 关系 (Relationship) → Z-test 产生统计量，P-test 给出显著性

13 Handling Popularity Bias in Recommendation Systems

Question: How would you handle “popularity bias” in a recommender?

Answer:

I’d reweight training data or apply negative sampling to reduce dominance of popular items. Another way is to regularize by item exposure frequency, or use fairness-aware objectives. In ranking, I might mix a small fraction of exploration items to ensure long-tail coverage.

中文关键词辅助记忆:

- 流行度偏差 (Popularity bias) → 热门物品占主导，冷门被忽略
- 重加权训练数据 (Reweight training data) → 降低热门权重
- 负采样 (Negative sampling) → 平衡正负样本比例
- 按曝光频次正则化 (Regularize by item exposure frequency)
- 公平性目标 (Fairness-aware objectives)
- 排序中加入探索项 (Exploration items) → 扩展长尾覆盖 (Long-tail coverage)

14 Feature Selection Methods

Question: How do you select important features for your model?

Answer:

I combine domain knowledge and data-driven selection. Statistically, I use correlation analysis, permutation importance, or SHAP values. In tree-based models, feature importance is straightforward; in deep models, I rely on embedding visualization and attention weights. I also prune redundant features to reduce overfitting and serving latency.

中文关键词辅助记忆:

- 专业知识结合数据驱动 (Domain knowledge + data-driven)
- 统计方法：相关分析 (Correlation), Permutation Importance, SHAP
- 树模型：特征重要性直观 (Feature importance)
- 深度模型：嵌入可视化与注意力权重 (Embedding visualization, attention weights)
- 特征剪枝 (Feature pruning)：减少过拟合与推理延迟

15 Collaborative Filtering vs. Matrix Factorization

Question: What's the difference between collaborative filtering and matrix factorization?

Answer:

Collaborative filtering (CF) uses user-item interactions to infer similarity, while matrix factorization (MF) explicitly learns latent vectors for users and items such that their dot product approximates observed interactions. MF is essentially a parameterized form of CF that's easier to extend with regularization and side features.

中文关键词辅助记忆:

- CF：基于用户–物品交互的相似性 (user-item similarity)
- MF：学习用户与物品的潜在向量 (latent vectors)，用内积近似交互
- 关系：MF 是 CF 的参数化版本，可扩展正则化与侧信息 (regularization, side features)

16 Combining Offline and Online Evaluation for Personalization Models

Question: How do you combine offline and online evaluation for personalization models?

Answer:

Offline metrics like NDCG or AUC help filter out poor candidates quickly. However, only A/B tests capture true business impact. I usually promote a model only if offline metrics improve and online KPIs (like CTR, conversion) show statistically significant uplift.

中文关键词辅助记忆:

- Offline：NDCG、AUC 等指标→ 快速筛选模型质量
- Online：A/B 测试→ 衡量实际业务效果 (CTR, Conversion)
- 策略：离线提升 + 在线显著提升→ 才晋级上线

17 Common Sources of Data Leakage in User Engagement Data

Question: What are some common sources of data leakage in user engagement data?

Answer:

- Using post-event features like “total clicks after purchase”.
- Aggregations that include future time windows.
- Improper joins between train/test sets.

I prevent leakage by building time-based splits and verifying that only past data is accessible during training.

中文关键词辅助记忆:

- 未来信息泄露：例如「购买后点击数」或「未来窗口聚合」。
- 错误的连接：训练集与测试集之间不当的 join。
- 预防策略：基于时间的切分 (time-based split)，确保模型训练只访问历史数据。

18 Designing a Hybrid Recommender

Question: How would you design a hybrid recommender?

Answer:

I’d blend collaborative filtering and content-based signals. For example, CF captures behavioral similarity, while content models handle cold-start. The blending can be linear (weighted average) or learned via a meta-model that optimizes CTR. In Sam’s Club, this can help balance personalization with item diversity.

中文关键词辅助记忆:

- **结合两类信号：** 协同过滤 (CF) + 内容特征 (Content-based)。
- **互补优势：** CF 擅长个性化与行为相似度；内容模型解决冷启动问题。
- **融合方式：** 线性加权或通过元模型 (meta-model) 学习最优组合。
- **业务意义：** 在 Sam’s Club 等场景中平衡推荐的个性化与多样性。

19 Regularization and Its Importance

Question: What is regularization and why is it important?

Answer:

Regularization adds a penalty to model complexity, helping to prevent overfitting. L1 promotes sparsity and feature selection; L2 smooths weights and stabilizes training. In deep models, techniques like dropout or batch normalization also act as regularizers.

中文关键词辅助记忆:

- **目的：** 防止过拟合（overfitting），控制模型复杂度。
- **L1 正则化：** 促进稀疏性（sparsity），用于特征选择。
- **L2 正则化：** 平滑权重（smooth weights），提升训练稳定性。
- **深度模型中的正则化手段：** Dropout（随机失活）和 Batch Normalization（批归一化）。
- **核心思路：** 降低模型对训练数据噪声的敏感性，提高泛化能力。

20 Handling Missing or Noisy Data in Behavioral Datasets

Question: How do you handle missing or noisy data in behavioral datasets?

Answer:

For missing values, the approach depends on the data and task context. If the dataset is large, dropping a small portion of rows with missing values usually has minimal impact, so I might choose to remove them. If dropping data is not feasible, I'd consider imputation strategies — for example, using the mean or median for numerical features, or applying more advanced interpolation methods if appropriate.

For noisy data, my first step is to check if the noise can be directly filtered out — for example, by fitting a normal distribution to detect outliers or applying denoising techniques like FFT. If that's not possible, I analyze the characteristics of the noise and consider smoothing techniques. If neither works, I use models that are robust to noise, such as tree-based models or regularized regression, depending on the task.

中文关键词辅助记忆:

- **缺失值 (Missing Data):**
  - 数据量大时可直接删除（drop rows）。
  - 不可删除时使用 **插补 (imputation):**

    - 均值 / 中位数填充 (mean/median imputation)
    - 插值法 (interpolation)

- **噪声数据 (Noisy Data):**
  - 通过正态分布拟合检测异常值 (fit normal distribution for outlier detection)。
  - 使用 **FFT（快速傅里叶变换）** 去噪。
  - 使用平滑方法 (smoothing)。
  - 采用对噪声鲁棒的模型：
    - Tree-based models（如 XGBoost, Random Forest）
    - Regularized regression（正则化回归）
- **核心思路：** 缺失值→ 补；噪声→ 滤或换稳健模型。（Filter noise, Fill missing, Choose robust models）

21 Common Recommendation Algorithms

Question: 请讲讲你熟悉的推荐算法有哪些？（CF, MF, Deep Learning-based等）

Answer:

I'm familiar with several types of recommendation algorithms.

- **Collaborative Filtering (CF):** Includes user-based and item-based approaches. Predicts a user's preference based on similar users or items. Matrix Factorization (MF) methods like **SVD** or **ALS** learn latent factors representing users and items to capture hidden interactions.
- **Content-Based Methods:** Recommend items similar to what a user liked before, based on explicit item features (e.g., category, description, or embedding vectors).
- **Deep Learning-based Methods:** Modern neural architectures improve representation learning and personalization:
  - **Wide & Deep**, **DeepFM** — effective for CTR prediction, combining memorization and generalization.
  - **GRU4Rec**, **SASRec** — sequential models capturing temporal dependencies in user sessions.
  - **LightGCN** — graph-based model capturing high-order connectivity between users and items.
- **Hybrid Methods:** Combine CF and content-based models to balance personalization, diversity, and handle the cold-start problem.

中文关键词辅助记忆：

- **协同过滤 (CF):** 用户-物品相似度；分为User-based / Item-based。
- **矩阵分解 (MF):** SVD, ALS —— 提取隐向量 (latent factors)。
- **内容推荐 (Content-based):** 根据物品特征找相似内容。
- **深度学习推荐:**
  - Wide&Deep, DeepFM → CTR预测；
  - GRU4Rec, SASRec → 序列建模；
  - LightGCN → 图关系挖掘。
- **混合推荐 (Hybrid):** 综合优点；解决冷启动 + 提升多样性。

22 Cold-Start User Recommendation Strategies

Question: 如果一个用户是新注册的（cold-start），你会怎么解决推荐问题？

Answer:

For cold-start users with no interaction history, there are several approaches I' d consider:

- **Demographic or Content-Based Recommendation:** Use registration information such as age, gender, or interest tags to match users with similar profiles or recommend relevant items based on content similarity.
- **Popularity-Based or Trending Items:** Recommend globally popular or trending products when personalization is not yet possible. This approach is simple and ensures user engagement at early stages, though personalization is limited.
- **Onboarding Questionnaires or Surveys:** Collect initial user preferences during registration through short surveys or quizzes, then assign users to predefined clusters or segments for personalized recommendations.
- **Contextual Recommendations:** Utilize contextual data such as location, device type, time of day, or session behavior to recommend items that are relevant in the current context.
- **Embedding-Based Approach:** If we have a learned user-attribute embedding model, we can map the new user's attributes into the latent space and recommend nearest-neighbor items, achieving better personalization than pure popularity-based methods.

中文关键词辅助记忆:

- **人口学/内容推荐：** 用年龄、性别、兴趣等信息匹配相似用户或物品。

- **热门推荐：** 推荐热门或趋势物品（简单但无个性化）。
- **问卷收集偏好：** 注册时通过问卷或选择标签收集初步兴趣。
- **上下文推荐：** 利用地理位置、设备类型、时间等上下文信息。
- **Embedding推荐：** 用属性嵌入映射到隐空间，找最近邻物品。

23 Hyperparameter Optimization for Random Forest and XGBoost

Question: How to optimize Hyperparameters for Random Forest and XGB?

Answer:

Random Forest (RF) Key Hyperparameters:

- **n_estimators:** Number of trees in the forest.
- **max_depth:** Maximum depth of each tree.
- **min_samples_split / min_samples_leaf:** Control overfitting by limiting splits and leaf size.
- **max_features:** Number of features considered when splitting nodes.

XGBoost (XGB) Key Hyperparameters:

- **n_estimators / num_boost_round:** Number of trees.
- **max_depth:** Maximum depth of each tree.
- **learning_rate / eta:** Step size shrinkage to prevent overfitting.
- **subsample / colsample_bytree:** Control overfitting by row/column sampling.
- **reg_alpha / reg_lambda:** L1/L2 regularization terms.

Optimization Approaches:

- Grid Search: Exhaustively search a predefined parameter grid.
- Random Search: Randomly sample hyperparameters; faster for large search spaces.
- Bayesian Optimization / Hyperopt / Optuna: Model-based, sample-efficient optimization.

Practical Tips:

- Start with coarse search, then fine-tune promising ranges.
- For RF: Increase n_estimators and adjust max_depth to reduce overfitting.
- For XGB: Tune learning_rate with n_estimators, and adjust regularization parameters to control overfitting.

中文关键词辅助记忆:

- **RF超参数:** 树数量、最大深度、最小分裂/叶子样本数、特征数
- **XGB超参数:** 树数量、最大深度、学习率、采样比率、正则化参数
- **调参方法:** Grid Search (网格搜索), Random Search (随机搜索), Bayesian Optimization / Hyperopt / Optuna (贝叶斯优化)
- **实践经验:** 粗调→ 精调；RF: 增加树数量 + 调整 max_depth；XGB: 调整 learning_rate + n_estimators，正则化控制过拟合