INTERVIEW QUESTIONS

1.How does logistic regression differ from linear regression?

2.What is the sigmoid function?

3.What is precision vs recall?

4.What is the ROC-AUC curve?

5.What is the confusion matrix?

6.What happens if classes are imbalanced?

7.How do you choose the threshold?

8.Can logistic regression be used for multi-class problems?

1.How does logistic regression differ from linear regression?

Explanation (intuition).

- **Linear regression** predicts a **continuous** value $\hat{y} \in \mathbb{R}$. It models the conditional mean $E[y|x] = x^\top w$.
- **Logistic regression** predicts a **probability** that an example belongs to class 1: $P(y = 1 \mid x) \in (0, 1)$. It models the **log-odds** (logit) as a linear function of x: $\text{logit}(p) = \log \frac{p}{1-p} = x^\top w$. Use a threshold (usually 0.5) on the probability to get a class label.

Key formulas.

Linear regression (prediction):

$$\hat{y} = x^\top w \quad \text{(often fit by minimizing MSE)}$$

MSE loss:

$$L_{\text{MSE}}(w) = \frac{1}{2n} \sum_{i=1}^{n} (y_i - x_i^\top w)^2$$

Logistic regression (probability):

$$p(x) = P(y = 1 \mid x) = \sigma(x^\top w) = \frac{1}{1 + e^{-x^\top w}}$$

Log-loss (negative log-likelihood):

$$L_{\log}(w) = -\frac{1}{n} \sum_{i=1}^{n} \Big[y_i \log p(x_i) + (1 - y_i) \log(1 - p(x_i)) \Big]$$

Differences to remember:

- Output type: real value vs probability in $[0, 1]$.
- Loss: MSE (squared errors) vs log-loss (cross-entropy / negative log-likelihood).
- Interpretation: linear predicts mean; logistic predicts odds/probability.
- Predictions from linear can fall outside $[0, 1]$; logistic is constrained to $[0, 1]$.

Example (toy):

Suppose $x^\top w = 1.2$.

- Linear prediction: $\hat{y} = 1.2$ (not a probability).
- Logistic probability: $p = \sigma(1.2) = \frac{1}{1+e^{-1.2}} \approx 0.768$. This can be interpreted directly as probability of class 1.

2.What is the sigmoid function?

Explanation.

The sigmoid (logistic) function $\sigma(z)$ squashes any real number z into the interval $(0, 1)$. In logistic regression we set $z = x^\top w$ and interpret $\sigma(z)$ as a probability.

Formula:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Useful identities:

- $\sigma(0) = 0.5$.
- Derivative: $\sigma'(z) = \sigma(z)\big(1 - \sigma(z)\big)$. This is handy for gradients in optimization.
- Logit (inverse): $\mathrm{logit}(p) = \log \frac{p}{1-p}$ so $\sigma(\mathrm{logit}(p)) = p$.

Example values:

- $\sigma(0) = 0.5$
- $\sigma(2) \approx 0.8808$ (so a large positive score → high probability).
- $\sigma(-2) \approx 0.1192$ (large negative score → low probability).

Graphical intuition: S-shaped curve; near $-\infty$ it → 0, near $+\infty$ it → 1, center at 0.

3.What is precision vs recall?

Definitions (from the confusion matrix):

- True Positive (TP): predicted 1 and true 1.
- False Positive (FP): predicted 1 but true 0.
- False Negative (FN): predicted 0 but true 1.
- True Negative (TN): predicted 0 and true 0.

Formulas:

$$\text{Precision} = \frac{TP}{TP + FP} \qquad \text{Recall (Sensitivity, TPR)} = \frac{TP}{TP + FN}$$

Also useful:

$$\text{F1 score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Interpretation:

- **Precision** = “when I predict positive, how often am I right?” — controls false positives.
- **Recall** = “of all actual positives, how many did I detect?” — controls false negatives.
  Which matters depends on application (e.g., recall is critical for disease screening; precision is critical when false alarms are costly).

Numeric example: (confusion matrix)

- TP = 40, FP = 10, FN = 5, TN = 45 (100 samples total)
  - Precision = $40/(40 + 10) = 0.80$
  - Recall = $40/(40 + 5) \approx 0.8889$
  - Accuracy = $(40 + 45)/100 = 0.85$
  - F1 ≈ 0.8421

This shows high recall (we catch most positives) and good precision (most predicted positives are correct).

4.What is the ROC-AUC curve?

ROC (Receiver Operating Characteristic) curve:

- Plot of **True Positive Rate** (TPR = recall) on the y-axis versus **False Positive Rate** (FPR = FP/(FP+TN)) on the x-axis as you vary the classification threshold. Each threshold gives one (FPR, TPR) point — the ROC curve connects these.

AUC (Area Under Curve):

- $\mathrm{AUC} = \int_0^1 \mathrm{TPR}(\mathrm{FPR}^{-1}(x))\,dx$ — geometrically the area under the ROC curve.
- Interpretation: AUC = probability that a randomly chosen positive example has a higher predicted score than a randomly chosen negative example:

$$\mathrm{AUC} = P\big(\mathrm{score}(X^+) \; > \; \mathrm{score}(X^-)\big)$$

Why it's useful:

- It measures ranking quality independent of a chosen threshold and of class imbalance to some extent. Values range from 0.0 (worst) to 1.0 (perfect); 0.5 indicates random ranking.

Small example & computing AUC by trapezoids (illustration):

Given a small dataset with true labels y and probabilities p,

- $y = [1, 1, 0, 0, 1, 0]$
- $p = [0.9, 0.4, 0.7, 0.6, 0.3, 0.2]$

If you compute ROC points (FPR, TPR) for thresholds and then integrate using trapezoids, you get an AUC ≈ 0.5556 (i.e. slightly better than random in this toy example). The AUC can be computed numerically by:

$$\mathrm{AUC} \approx \sum_{i=1}^{m-1} (\mathrm{FPR}_{i+1} - \mathrm{FPR}_i) \cdot \frac{\mathrm{TPR}_i + \mathrm{TPR}_{i+1}}{2}.$$

Caveat: When classes are heavily imbalanced, Precision–Recall curves are often more informative than ROC curves.

5.What is the confusion matrix?

Definition (2×2 table):

	Pred = 1	**Pred = 0**
True = 1	TP	FN
True = 0	FP	TN

Common metrics expressed from it:

- Accuracy $= \dfrac{TP + TN}{TP + TN + FP + FN}$
- Precision $= \dfrac{TP}{TP + FP}$
- Recall / Sensitivity $= \dfrac{TP}{TP + FN}$
- Specificity $= \dfrac{TN}{TN + FP}$
- False Positive Rate $= \dfrac{FP}{FP + TN} = 1 - \text{Specificity}$
- Negative Predictive Value $= \dfrac{TN}{TN + FN}$
- F1 score $= 2\frac{\text{Precision}\cdot\text{Recall}}{\text{Precision}+\text{Recall}}$

Example (same as question 3): TP=40, FP=10, FN=5, TN=45 → you can compute all metrics from the table (see earlier).

Confusion matrix is the fundamental tool to compute and reason about the trade-offs (precision vs recall, etc.).

6.What happens if classes are imbalanced?

Problem: When one class (say negatives) far outnumbers the other (positives), common issues arise:

- **Accuracy becomes misleading.** A classifier always predicting the majority class can have high accuracy but be useless (e.g., 99% accuracy in a 99:1 dataset by always predicting the majority).
- **ROC may be optimistic**, while **Precision–Recall (PR) curve** is more sensitive to class imbalance for the minority (positive) class.
- **Training** may under-emphasize the minority class because the loss is dominated by the majority.

Techniques to handle imbalance:

1. **Resampling**
   - Oversample minority (random oversampling, SMOTE) or undersample majority.
2. **Use class weights** in the loss (e.g., in scikit-learn `class_weight='balanced'`) — gives more penalty for misclassifying minority class.
3. **Use appropriate metrics**: Precision, Recall, F1, PR-AUC, Matthews correlation coefficient (MCC), balanced accuracy:

$$\text{Balanced accuracy} = \frac{\text{TPR} + \text{TNR}}{2}$$

4. **Threshold tuning** to meet recall/precision targets.
5. **Specialized algorithms**: anomaly-detection methods if the positive class is extremely rare.

Example: If positives = 1% and you care about detecting positives (recall), optimize thresholds or use sampling/weights — otherwise model may just predict all negatives.

7.How do you choose the threshold?

Default: 0.5 — but this is arbitrary.

Practical methods to choose a threshold:

1. **Business-cost based:** choose threshold that minimizes expected cost:

$$\text{Expected cost}(t) = \text{Cost}_{FP} \cdot FP(t) + \text{Cost}_{FN} \cdot FN(t)$$

   Choose t minimizing expected cost (costs come from domain knowledge).

2. **Youden's J statistic:** choose t that maximizes

$$J(t) = \text{Sensitivity}(t) + \text{Specificity}(t) - 1.$$

   This maximizes overall correct separation.

3. **Maximize F1** (or other metric): pick t that maximizes F1 if you want a balance between precision and recall.
4. **Meet a constraint:** choose smallest t such that recall ≥ required level (useful in screening where recall must be high).
5. **ROC/PR curve inspection:** pick a point on curve that gives the best trade-off visually or according to metric.

Concrete threshold example (toy):
Labels $y = [1, 0, 1, 0, 1, 0]$, scores $p = [0.9, 0.4, 0.7, 0.6, 0.2, 0.1]$.

- At threshold $t = 0.5$: predictions = [1,0,1,1,0,0] → TP=2, FP=1, FN=1, TN=2 → precision=0.667, recall=0.667.
- At threshold $t = 0.3$: predictions = [1,1,1,1,0,0] → TP=2, FP=2, FN=1, TN=1 → precision=0.5, recall=0.667.

Lowering the threshold often increases recall and decreases precision. Choose t depending on whether you want fewer false negatives (favor recall) or fewer false positives (favor precision).

8.Can logistic regression be used for multi-class problems?

Yes — two standard approaches:

1. **One-vs-Rest (OvR / One-vs-All):**
   - Train K binary logistic regressors for K classes. For class k the model learns $P(y = k \mid x)$ vs others. At prediction time choose class with highest score (or probability). Simple and works well.
2. **Multinomial logistic regression (softmax):**
   - A single model that outputs a K-way probability distribution using the **softmax** function. This is sometimes called "multinomial" logistic regression and optimizes the multinomial (cross-entropy) loss directly.

Softmax formula (for K classes):

Let $z_j = w_j^\top x$ be the score for class j. Then

$$P(y = j \mid x) = \frac{\exp(z_j)}{\sum_{k=1}^{K} \exp(z_k)}.$$

Multiclass cross-entropy loss:

$$L(w) = -\frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{K} \mathbf{1}\{y_i = j\} \log P(y_i = j \mid x_i).$$

Example (softmax numeric):

Scores $z = [1.2,\ 0.3,\ -0.5]$ for classes 1..3.

$$\text{softmax}(z)_j = \frac{e^{z_j}}{e^{1.2} + e^{0.3} + e^{-0.5}}$$

Numerically:

$$P \approx [0.6292,\ 0.2558,\ 0.1149]$$

So the model would pick class 1 with probability ≈ 0.63.

Practical note: scikit-learn's `LogisticRegression` supports both `multi_class='ovr'` and `multi_class='multinomial'` (the latter with `solver='lbfgs'` etc.). Multinomial often performs better if the true problem is inherently multi-class.

Quick summary / cheat-sheet

- Logistic regression => models $\sigma(x^\top w)$ → probability; use log-loss.
- Sigmoid: $\sigma(z) = 1/(1+e^{-z})$; derivative $\sigma(z)(1-\sigma(z))$.
- Precision = $TP/(TP+FP)$ (correctness of positive predictions).
- Recall = $TP/(TP+FN)$ (coverage of actual positives).
- ROC plots TPR vs FPR; AUC measures ranking quality (probability a positive outranks a negative).
- Confusion matrix gives TP/FP/FN/TN; all standard metrics derive from it.
- Imbalanced classes → accuracy misleading; use class weights, resampling, PR curves.
- Threshold selection should be driven by business cost or chosen metric (Youden's J, F1, cost minimization, constraints).
- Multi-class: OvR or softmax (multinomial logistic) — both supported in libraries.