ML Models and Architectures You Should Know

Model/Architecture	Brief Description	When Used
Linear Regression	Predicts continuous values by fitting a linear relationship.	Used for simple relationships between variables, assumes linearity.
Polynomial Regression	Extends linear regression by adding polynomial terms.	Used when the relationship is nonlinear but can be approximated with polynomials.
Logistic Regression	Predicts probabilities for classification tasks.	Used for binary and multiclass classification when features are linearly separable.
Decision Trees	Splits data into branches based on feature conditions.	Used for classification and regression when interpretability is needed.
Random Forests	An ensemble of decision trees to improve accuracy.	Used for reducing overfitting and improving prediction robustness.
Gradient Boosting Machines (GBM)	Boosts weak learners (decision trees) iteratively.	Used when higher accuracy is needed at the cost of training time.
XGBoost	An optimized version of GBM with regularization.	Used for structured data problems with high performance.
LightGBM	A gradient boosting algorithm optimized for efficiency.	Used for large datasets and faster training.
CatBoost	Boosting method optimized for categorical data.	Used for datasets with categorical variables and minimal preprocessing.
K-Means	Partitions data into k clusters based on distance.	Used for market segmentation, anomaly detection, and clustering tasks.
Hierarchical Clustering	Builds a tree of clusters using a bottom-up approach.	Used when the number of clusters is unknown and hierarchical structure is useful.
DBSCAN	Groups data based on density and noise detection.	Used for anomaly detection and irregularly shaped clusters.
Gaussian Mixture Models (GMM)	Probabilistic clustering using Gaussian distributions.	Used when clusters have overlapping distributions and need soft clustering.

Principal Component Analysis (PCA)	Reduces dimensions by projecting onto principal components.	Used to remove redundancy and speed up learning in high-dimensional data.
Independent Component Analysis (ICA)	Finds independent sources in mixed signals.	Used in signal processing and separating independent data sources.
t-SNE	Projects high-dimensional data into 2D or 3D for visualization.	Used for visualizing complex datasets, not for learning models.
UMAP	Similar to t-SNE but preserves more global structure.	Used for visualizing clusters in high-dimensional space efficiently.
Multi-Layer Perceptron (MLP)	A basic neural network with multiple layers.	Used for general function approximation and classification.
Convolutional Neural Networks (CNN)	Uses convolutional layers for feature extraction.	Used for image recognition and processing tasks.
Recurrent Neural Networks (RNN)	Processes sequential data using memory states.	Used for time series prediction, language modeling, and speech recognition.
Long Short-Term Memory (LSTM)	A type of RNN with memory cells to avoid vanishing gradients.	Used for long-term dependencies in sequence prediction tasks.
Gated Recurrent Unit (GRU)	A simplified LSTM with fewer parameters.	Used when efficiency is needed while maintaining sequence information.
Transformer	Processes entire sequences at once using self-attention.	Used in NLP tasks like machine translation, BERT, and GPT.
BERT	A transformer-based model pre-trained for NLP tasks.	Used for sentiment analysis, question answering, and text classification.
GPT	A generative model that produces human-like text.	Used for text generation and conversational AI applications.
Q-Learning	A value-based reinforcement learning method.	Used for simple RL tasks where a value function is sufficient.
Deep Q-Networks (DQN)	Uses deep learning to approximate Q-values.	Used for complex RL tasks like playing video games.

Policy Gradient Methods	Directly optimizes the policy without Q-values.	Used when action spaces are large or continuous.
Actor-Critic Methods	Combines policy gradients with value-based methods.	Used for stable and efficient reinforcement learning training.
Naive Bayes	Probabilistic classifier using Bayes' theorem.	Used for text classification and spam filtering.
Bayesian Networks	Represents variables and dependencies probabilistically.	Used in probabilistic reasoning and decision making.
Hidden Markov Models (HMM)	Uses a sequence of hidden states to model time series.	Used for speech recognition and financial modeling.
Isolation Forest	Detects anomalies by isolating outliers in trees.	Used for fraud detection and network security.
Autoencoders	Neural networks trained to reconstruct inputs.	Used for anomaly detection and feature compression.
GANs (Generative Adversarial Networks)	Uses a generator and discriminator for realistic data generation.	Used for image synthesis, deepfake creation, and data augmentation.
Variational Autoencoders (VAE)	Generative models that learn latent space representations.	Used for generative modeling and anomaly detection.

Here's a **comprehensive list** of general **theorems, concepts, and ideas** you **must** know** for an ML fundamentals technical interview**:

---

1. Bias-Variance Tradeoff

Concept

- **Bias**: Error due to overly simplistic assumptions; leads to **underfitting**.
- **Variance**: Sensitivity to fluctuations in training data; leads to **overfitting**.
- **Tradeoff**: Increasing model complexity **reduces bias but increases variance**, and vice versa.

Key Takeaways

- **High bias, low variance** → Underfitting (model is too simple).
- **Low bias, high variance** → Overfitting (model memorizes training data).
- **Optimal balance** → Good generalization.

Formula

Total Error = **Bias^2 + Variance + Irreducible Error**

2. Overfitting vs Underfitting

Concept

- **Underfitting**: Model is too simple and fails to learn patterns in the training data.
- **Overfitting**: Model is too complex and memorizes the training data but does not generalize.

How to Handle?

- **For underfitting**:
  - Use a **more complex model** (e.g., from linear to polynomial regression).
  - Train **longer**.
  - Use **more features**.
- **For overfitting**:
  - Use **regularization** (L1, L2).
  - Get **more training data**.
  - Use **dropout** (for neural networks).
  - Reduce model complexity.

---

3. Curse of Dimensionality

Concept

- As the number of features (dimensions) increases, the data becomes sparse, making it harder for models to learn patterns.

Implications

- **Distance-based algorithms (KNN, SVM, Clustering) struggle in high dimensions.**
- **Exponential growth of computation.**

Solutions

- **Dimensionality reduction** (PCA, t-SNE, LDA).
- **Feature selection** (removing irrelevant features).

4. Central Limit Theorem (CLT)

Concept

- If you take **many random samples** from a population and compute their means, the distribution of these means **approaches a normal distribution**, regardless of the original distribution.

Why is this important?

- Justifies using **Gaussian-based models**.
- Essential for **confidence intervals** and **hypothesis testing**.

5. No Free Lunch Theorem

Concept

- **No single algorithm works best for all problems.**
- The best model depends on **the dataset and problem type**.

Implications

- Must **experiment with multiple algorithms**.
- **Cross-validation** is essential to select the best model.

---

6. Law of Large Numbers (LLN)

Concept

- As the sample size **increases**, the sample mean **converges** to the true population mean.

Implications

- A small dataset can lead to **high variance**.
- More data generally leads to **better generalization**.

7. Regularization (L1, L2, ElasticNet)

Concept

- Regularization **adds a penalty term** to the loss function to prevent overfitting.

Types

- **L1 (Lasso)**: Shrinks some coefficients to **zero**, leading to **feature selection**.
- **L2 (Ridge)**: Shrinks coefficients but **doesn’t eliminate them**.
- **ElasticNet**: Combines L1 and L2.

When to Use?

- **L1**: When you want **sparse features** (feature selection).
- **L2**: When you want to **prevent large coefficients**.
- **ElasticNet**: When both sparsity and small coefficients are needed.

8. Gradient Descent and Variants

Concept

- **Optimization algorithm** used to minimize the loss function by iteratively updating model parameters.

Types

- **Batch Gradient Descent**: Uses the entire dataset.
- **Stochastic Gradient Descent (SGD)**: Uses **one data point** at a time.
- **Mini-batch Gradient Descent**: Uses a **small subset** of data.

When to Use?

- **Batch GD**: More stable but slow for large datasets.
- **SGD**: Faster but noisier.
- **Mini-batch**: Best compromise.

9. Bayes' Theorem

Concept

- **Formula:**

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

- Used in **Naïve Bayes**, **Bayesian Inference**, and **ML probabilistic models**.

Key Insights

- Helps update **prior beliefs** based on new data.
- Works well when **features are independent**.

10. Cross-Validation

Concept

- **Splits data into multiple folds** to train and test the model on different subsets.

Types

- **K-Fold CV**: Divides data into `K` parts and trains on `K−1`, testing on the last one.
- **Stratified K-Fold CV**: Ensures class balance across folds.
- **Leave-One-Out CV (LOO-CV)**: Uses **one sample** for testing and the rest for training (expensive).

Why Use It?

- Reduces **overfitting**.
- Ensures **robust evaluation**.

11. ROC Curve and AUC

Concept

- **ROC Curve**: Plots **True Positive Rate (TPR) vs False Positive Rate (FPR)**.
- **AUC (Area Under the Curve)**: Measures model performance.

Key Takeaways

- **AUC close to 1** → Good model.
- **AUC ≈ 0.5** → Random guessing.

---

12. Loss Functions (Common Types)

Regression

- **Mean Squared Error (MSE)**: Penalizes large errors.
- **Mean Absolute Error (MAE)**: Penalizes errors linearly.

Classification

- **Binary Cross-Entropy (Log Loss)**: For binary classification.
- **Categorical Cross-Entropy**: For multi-class classification.
- **Hinge Loss**: For SVM.

13. Confusion Matrix Metrics

Concept

A **confusion matrix** summarizes classification performance with:

- **True Positives (TP)**: Correctly predicted positive.
- **False Positives (FP)**: Incorrectly predicted positive.
- **True Negatives (TN)**: Correctly predicted negative.
- **False Negatives (FN)**: Incorrectly predicted negative.

Derived Metrics

- **Precision** = $\frac{TP}{TP+FP}$
- **Recall** = $\frac{TP}{TP+FN}$
- **F1 Score** = $2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$

14. Principal Component Analysis (PCA)

Concept

- **Reduces dimensionality** by projecting data onto principal components.
- Finds directions of **maximum variance**.

When to Use?

- When there are **many correlated features**.
- When reducing **computational cost**.

---

15. Reinforcement Learning Fundamentals

Concept

- Learning from **reward-based feedback**.
- Uses **Markov Decision Processes** (MDP).

Key Components

- **Agent** (learner).
- **Environment** (world).
- **State** (current situation).
- **Action** (decision taken).
- **Reward** (feedback signal).

---

1. Classification Metrics

1.1 Precision vs. Recall

Precision (Positive Predictive Value)

- **Definition:**

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

- **Interpretation:** Out of all the **predicted positives**, how many were actually correct?
- **When to prioritize?**
  - When **False Positives (FP)** are **costly**.
  - Example: **Spam detection** → If a legitimate email (ham) is classified as spam (FP), it's a problem.

Recall (Sensitivity / True Positive Rate)

- **Definition:**

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

- **Interpretation:** Out of all the **actual positives**, how many were correctly identified?
- **When to prioritize?**
  - When **False Negatives (FN)** are **costly**.
  - Example: **Cancer detection** → Missing a cancer case (FN) is worse than a false alarm.

Tradeoff Between Precision and Recall

- Increasing **Precision** lowers **Recall** (and vice versa).
- Example: A **strict spam filter** (high precision) may **miss some spam emails** (low recall).
- **Solution:** Use **F1-score** to balance both.

F1-Score (Harmonic Mean of Precision & Recall)

- **Definition:**

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

- **When to Use?**
  - When there is **an imbalance** between **FP and FN costs**.
  - Useful in **imbalanced datasets** (e.g., fraud detection, medical diagnosis).

Other Classification Metrics

Accuracy

- **Definition:**

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

- **Problem:**
    - Misleading when classes are **imbalanced** (e.g., 99% non-fraud, 1% fraud → Always predicting non-fraud gives 99% accuracy).
- **Better Alternatives?**
    - **F1-score, Precision-Recall AUC, ROC AUC.**

Specificity (True Negative Rate)

- **Definition:**

$$\text{Specificity} = \frac{TN}{TN + FP}$$

- **Interpretation:** Out of all the **actual negatives**, how many were correctly classified?

Balanced Accuracy

- **Definition:**

$$\text{Balanced Accuracy} = \frac{\text{Sensitivity} + \text{Specificity}}{2}$$

- **When to Use?**
  - When classes are **imbalanced**.

Matthews Correlation Coefficient (MCC)

- **Definition:**

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

- **When to Use?**
  - Works well for **imbalanced classification**.

1.2 ROC Curve and AUC

Receiver Operating Characteristic (ROC) Curve

- **Definition:** Plots **True Positive Rate (TPR) vs. False Positive Rate (FPR)** for different thresholds.
- **AUC (Area Under Curve)**:
  - **1.0** → Perfect model.
  - **0.5** → Random guessing.

Precision-Recall (PR) Curve

- **Alternative to ROC** for **imbalanced datasets**.
- AUC-PR is more informative when **positives are rare**.

---

2. Regression Metrics

For regression, we measure **errors** instead of classification accuracy.

2.1 Mean Squared Error (MSE)

- **Definition:**

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

- **When to Use?**
  - When large errors should be **penalized more**.

2.2 Root Mean Squared Error (RMSE)

- **Definition:**

$$RMSE = \sqrt{MSE}$$

- **When to Use?**
  - When errors should be interpreted in the same units as **y**.

2.3 Mean Absolute Error (MAE)

- **Definition:**

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i|$$

- **When to Use?**
  - When we want to **treat all errors equally** (linear penalty).

2.4 R-Squared (R^2 Score)

- **Definition:**

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2}$$

- **Interpretation:**
  - **1.0** → Perfect prediction.
  - **0.0** → No improvement over mean prediction.
  - **Negative** → Worse than predicting the mean.

---

3. Ranking & Information Retrieval Metrics

3.1 Mean Average Precision (MAP)

- Used in **search engines**.
- Measures how well relevant documents are ranked.

3.2 Mean Reciprocal Rank (MRR)

- Used in **question-answering**.
- Measures how soon the correct answer appears in results.

Choosing the Right Metric

Problem Type	Best Metrics
Binary Classification	Precision, Recall, F1-score, ROC AUC
Imbalanced Classification	Precision-Recall AUC, F1-score, MCC
Multiclass Classification	F1-score (macro/micro), Accuracy
Regression	RMSE, MAE, R^2
Ranking/Search	MAP, MRR

Final Thoughts

- Choose metrics based on the problem type.
- Avoid using accuracy in imbalanced datasets.
- ROC AUC is good but PR AUC is better when positives are rare.
- Regression should use RMSE if penalizing large errors more is important.