# MACHINE LEARNING INTERVIEW

## CHEAT SHEET

EMMA ding

EMMA ding

emmading.com

info@datainterviewpro.com

# Machine Learning Interview Cheat Sheet

This comprehensive guide serves as a quick reference for various concepts, algorithms, and techniques. It walks you through the following question types in Machine Learning interviews:

1. **Data and feature engineering**
2. **Pros and cons of machine learning models**
3. **Model comparison**
4. **Loss functions**
5. **Evaluation metrics**

Data and feature engineering
- Missing Values
- Feature Selection
- Imbalanced Data

Pros and Cons of ML Algorithms
- Linear Models
- Tree-based Models
- Clustering Models

Model Comparison
- Random Forest vs. Gradient Boosting
- Decision Tree vs. Random Forest
- Linear Regression vs. Logistic Regression
- SVMs vs. Logistic Regression
- Logistic Regression vs. Random Forest
- K-means vs. Gaussian Mixture Model

Loss functions
- Cross-Entropy
- Hinge Loss
- Mean Squared Error (MSE)
- Root Mean Square Error (RMSE)
- Mean Absolute Error (MAE)

Evaluation metrics
- Regression

Confusion Matrix
Accuracy
Precision
Recall
Specificity
F1 Score
ROC Curve and Precision-Recall Curve

# Data and feature engineering

## ▼ Missing Values

- **Gather more data:** See if there is any way to backfill the data or join with external datasets.
- **Imputation:** Infer the missing values by leveraging our prior knowledge of the existing data.
  - Model-based imputation: use non-missing data to predict missing data with KNN/linear regression/trees.
- **Omission:** removing missing observations or features with lots of missing values in the dataset.

## ▼ Feature Selection

Select a **subset** of the original features for model training. It is usually used as a pre-processing step before doing the actual learning.

### Intrinsic Feature Selection Methods

Have feature selection naturally **embedded** with the training process of models such as tree-based models and regularization models.

**Pros**

✔️ • Fast and no external feature tool is needed.

• Provides a direct connection between feature selection and the object function.

**Cons**

⚠️ • Model-dependent and the choice of models is limited.

### Filter Feature Selection Methods

Select features that correlate well with target variables, such as univariate statistical analysis and feature importance scores. The process is performed only once and is independent of the model algorithm.

**Pros**

**Cons**

| | |
|---|---|
| ✓ • Fast and Simple.<br>• can be effective at capturing large trends in the dataset. | ⚠️ • Tend to select redundant features.<br>• Ignore relationships among features. |

### Wrapper Feature Selection Methods

An iterative process, such as sequential feature selection, repeatedly adds subset features to the model and then uses the resulting model performance to guide the selection of the next subset.

| **Pros** | **Cons** |
|---|---|
| ✓ • Search for a wider variety of feature subsets than other methods. | ⚠️ • Significant computation time when the number of features is large.<br>• Have the most potential to overfit the features to the training data. |

## ▼ Imbalanced Data

An imbalanced dataset is a dataset where one or more labels make up the majority of the dataset, leaving far fewer examples of other labels. This problem applies to both **classification** and **regression** tasks. A few ways to deal with imbalanced data:

### Resampling

Change the distribution of the training data to reduce the level of class imbalance by either upsampling the minority class or downsampling the majority class.

Methods include:

- Over-sampling
- Under-sampling
- Generate synthetic examples
    - SMOTE (synthetic minority oversampling technique)

| **Pros** | **Cons** |
|---|---|
| ✓ • Simple and fast. | ⚠️ • Risk of overfitting training data (over-sampling) and losing important information from removing data (under-sampling). |

### Model-level Methods

Make the model more robust to class imbalance by penalizing wrong classifications of the minority class more or by selecting appropriate algorithms (e.g. tree-based models).

**Pros**

✔

- Does not change the distribution of the training data.

**Cons**

⚠

- Methods are model specific.

### Evaluation Metrics

Choose appropriate evaluation metrics for the task.

**Metrics to Consider**

✔

- **Precision-Recall curve** gives more importance to the positive class and is helpful for dealing with imbalanced data.
- **Precision**, **recall**, and **F1** measure a model’s performance with respect to the positive class in a binary classification problem.

**Metrics to avoid**

⚠

- **Accuracy** is misleading when classes are imbalanced.
- **AUC of the ROC curve** treats both classes equally and is less sensitive to model improvement on minority class.

# Pros and Cons of ML Algorithms

## ▼ Linear Models

### Linear Regression

**Pros**

✔

- Simple. Easy to explain, implement and interpret.

**Cons**

⚠

- Makes linear assumptions between the features and the target.
- Prone to overfitting in high-dimensional data.
- Sensitive to outliers.

### Support Vector Machine

**Pros**

**Cons**

✔️

- Effective in high dimensional spaces and where the number of dimensions is greater than the number of samples.

- Versatile: different Kernel functions can be specified for the decision function. It is also possible to specify custom kernels.

- Memory efficient since the decision boundary depends on a few support vectors.

- Can handle outliers and overlapping classes.

⚠️

- If the number of features is much greater than the number of samples, avoid overfitting in choosing Kernel functions and regularization terms are crucial.

- Slow to train on large datasets because of the need to choose the appropriate kernel and its parameters.

- Do not directly provide probability estimates.

### Logistic Regression

**Pros**

✔️

- Easy to explain, implement and interpret.
- Outputs the probability of class membership.

**Cons**

⚠️

- Maximize the conditional likelihoods of the training data, which makes it prone to outliers.
- Prone to overfitting in high-dimensional data.

## ▼ Tree-based Models

### Decision Trees

**Pros**

✔️

- Easy to understand and interpret.
- Requires little data pre-processing.
- Doesn't require feature selection.
- Efficient in prediction: the cost of one prediction is logarithmic in the number of examples used to train the tree.

**Cons**

⚠️

- Prone to overfitting.
- Sensitive to noise.
- Not good at extrapolation.

### Random Forest

**Pros**

✓ • Has a better generalization performance than an individual decision tree due to randomness.
• Doesn’t require much parameter tuning.
• Doesn't require feature selection.
• Less sensitive to outliers in the dataset.
• It generates feature importance which is helpful when interpreting the results.

**Cons**

⚠ • Computationally expensive.

### Gradient Boosting

**Pros**

✓ • It produces very accurate models, it outperforms random forest in accuracy.
• No data pre-processing is required.
• Handles missing data - imputation not required.

**Cons**

⚠ • Gradient boosting is a sequential process that can be slow to train.
• Computationally expensive - often require many trees (>1000) which can be time and memory exhaustive.
• Sacrifices interpretability for accuracy - less interpretative in nature.

## ▼ Clustering Models

### K-means

**Pros**

✓ • Easy to implement.
• Computationally efficient.

**Cons**

⚠ • The number of clusters, has to be determined.
• Stability: Initial positions of centroids influence the final position.
• The shapes of clusters can only be circular with equal sizes.

### Gaussian Mixture Model

**Pros**

✔ • A soft clustering algorithm. Each data point is assigned a probability of belonging to each cluster.
• Clusters can have different shapes and sizes.

**Cons**

⚠ • Less efficient to train due to its flexibility.

# Model Comparison

## ▼ Random Forest vs. Gradient Boosting

Both are ensemble learning methods — training a group of models and combining their predictions to make a more accurate final prediction.

| Differences | **Random Forest** | **Gradient Boosting** |
|---|---|---|
| Training | Each learner is trained independently. | Each learner is trained sequentially to correct the errors made by the previous tree. |
| Making Predictions | The final prediction is an average of the predictions of all trees. | The overall prediction is given by a weighted sum of the collection. |
| Optimization | Mainly reduces variance. | Reduces both bias and variance. |
| Overfitting | Less prone to overfitting as it averages the predictions of many different trees. | More prone to overfitting as it's trained sequentially and can continue to fit the data until it is perfectly fit. |
| Hyperparameter | Fewer hyperparameters. | More hyperparameters. |
| Parallelization | Can be trained in parallel. | Cannot be trained in parallel. |

## ▼ Decision Tree vs. Random Forest

Both can be used for classification and regression tasks.

| Differences | Decision Tree | Random Forest |
|---|---|---|
| Training | A single tree is trained on the dataset. | An ensemble of decision trees. Each tree is trained independently. |
| Prediction | Makes predictions based on a series of rules trained on the dataset. | The final prediction is an average of the predictions of all trees. |

| Differences | Decision Tree | Random Forest |
|---|---|---|
| Overfitting | More prone to overfitting as the depth of the tree grows. | Less prone to overfitting as it averages the predictions of many different trees. |
| Hyperparameter | Fewer hyperparameters | More hyperparameters |
| Interpretability | More interpretable as it’s based on a series of simple rules that can be followed to make a prediction. | Models can be interpreted based on the feature importance which is less straightforward compared with decision trees. |

## ▼ Linear Regression vs. Logistic Regression

Both are parametric models that learn a set of parameters from the dataset to make predictions.

| Differences | Linear regression | Logistic regression |
|---|---|---|
| Type of prediction | Is used when the target is continuous. | Is used when the target is binary. E.g. 0 or 1, true or false, etc. |
| Prediction | Gives real value prediction. | Uses a logistic function as the prediction function that gives a value between 0 and 1, which can be interpreted as a probability. |
| Loss function | Uses mean squared error (MSE) in most applications. | Uses the cross-entropy loss as the cost function that measures the difference between the predicted probability distribution and the observed distribution. |
| Assumptions | Assumes there is a linear relationship between each input variable and the target. | Does not make any assumptions about the distribution of the input variables. |
| Sensitivity to outliers | More sensitive to outliers. | Less sensitive to outliers. |

## ▼ SVMs vs. Logistic Regression

Both are parametric models that learn a set of parameters from the dataset to make predictions. Both are classification models.

| Differences | SVM | Logistic regression |
|---|---|---|
| Prediction | Uses a linear function to make predictions. Can only predict class labels instead of probabilities. | Uses a logistic function as the prediction function that gives a value between 0 and 1, which can be interpreted as a probability. |

| Differences | SVM | Logistic regression |
|---|---|---|
| Loss function | Uses hinge loss as the loss function which is a measure of the amount by which the model's prediction is incorrect. | Uses the cross-entropy loss as the cost function that measures the difference between the predicted probability distribution and the observed distribution. |
| Decision boundary | Finds the separating hyperplane that maximizes the distance of the closest points to the margin. Can learn non-linear decision boundaries by using kernel tricks. | Finds linear decision boundary that focuses on maximizing the likelihood of the data — the distance from the data to the decision boundary. |

## ▼ Logistic Regression vs. Random Forest

| Differences | Random forest | Logistic regression |
|---|---|---|
| Structure | An ensemble of decision trees that work together to make a prediction. | A simple, linear model that is used to predict the probability of a binary outcome. |
| Training process | Involve training many decision trees independently. | Involves estimating the parameters of the model by maximizing the likelihood of the observed data. |
| Interpretability | The model can be interpreted based on the feature importance. | The model can be interpreted by the value and sign of the estimated coefficients. |
| Overfitting | More prone to overfitting if individual trees are allowed to grow infinitely. | Less prone to overfitting than random forests, as it is a simple, linear model with few parameters. |

## ▼ K-means vs. Gaussian Mixture Model

Both are unsupervised models that are used to group data into a predetermined number of clusters. Both require users to specify the number of clusters to be used.

| Differences | K-means | Gaussian Mixture Model (GMM) |
|---|---|---|
| Hard vs. soft clustering | A hard clustering algorithm. Each data point is assigned to a single cluster. | A soft clustering algorithm. Each data point is assigned a probability of belonging to each cluster. |
| Initialization | Random initialization. | Can use either random initialization or K-mean for initialization. |

| Differences | K-means | Gaussian Mixture Model (GMM) |
|---|---|---|
| Convergence | Converges when the centroids of the clusters do not change significantly between iterations. | Converges when the parameters of the model (mean, covariance, and mixing coefficients) do not change significantly between iterations. |
| Shape of clusters | Clusters are circular and sizes are equal. | Clusters can have different shapes and sizes. |
| Efficiency | More efficient due to its simplicity. | Less efficient due to its flexibility. |

# Loss functions

## ▼ Cross-Entropy

Cross-entropy loss measures the performance of a binary classification model by comparing the output distribution to observations.

$$L = ylog(p) + (1-y)log(1-p)$$

where $y$ is the target (data label), and $p$ is the predicted probability.

## ▼ Hinge Loss

Hinge loss measure of the amount by which the model's prediction is incorrect. It’s used in SVM.

$$L = \max(0, 1-y)$$

where $y$ is the predicted output.

## ▼ Mean Squared Error (MSE)

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

where $n$ is the number of samples, $y_i$ is the true value of the i-th sample, and $\hat{y_i}$ is the predicted value.

## ▼ Root Mean Square Error (RMSE)

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

where $n$ is the number of samples, $y_i$ is the true value of the i-th sample, and $\hat{y_i}$ is the predicted value.

## ▼ Mean Absolute Error (MAE)

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

where $n$ is the number of samples, $y_i$ is the true value of the i-th sample, and $\hat{y}_i$ is the predicted value.

# Evaluation metrics

## ▼ Regression

### ▼ Mean Squared Error (MSE)

Measures the average squared distance between predictions and true values.

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

| Pros | Cons |
|---|---|
| ✔️ • Penalizes large errors.<br>• Mathematically convenient to obtain gradient. | ⚠️ • Sensitive to outliers since outliers.<br>• Hard to interpret because the unit is squared. |

### ▼ Mean Absolute Error (MAE)

Measures the average absolute difference between predictions and true values.

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

| Pros | Cons |
|---|---|
| ✔️ • MAE is relatively simple to interpret.<br>• MAE is less sensitive to outliers compared to MSE. | ⚠️ • MAE is not differentiable at 0, which makes it more difficult to optimize. |

### ▼ $R^2$

$R^2$ describes the percentage of the target variable **variation** that is explained by the model.

- $R^2 = 0$: The model fails to accurately model the data at all.
- $R^2 = 1$: The model is a perfect fit for the data.

$$R^2 = \frac{RSS}{TSS} = 1 - \frac{ESS}{TSS} = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

**Pros**

✔️ • R-squared is easy to interpret.
• Easy to compare the performance of different models.

**Cons**

⚠️ • Always increases upon adding a new variable.
• Does not show the predicting power of the model.

## ▼ Adjusted $R^2$

Adjusted $R^2$ incorporates the model's degree of freedom.

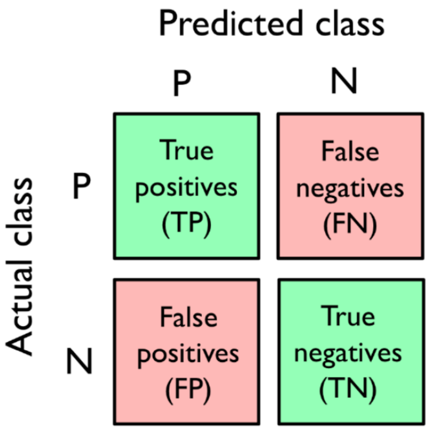
$$\bar{R}^2 = 1 - \frac{RSS/df_r}{TSS/df_t}$$

- $df_r$: degrees of freedom of the estimate of the variance around the **model**.
  - $n - p$
  - $n$: total number of observations
  - $p$: total number of variables (i.e. features) in the model

| | $R^2$ | adjusted $R^2$ |
|---|---|---|
| Add a feature (column) | Increases | Increases only if the feature improves the accuracy of the model |
| Add observation (row) | may increase, decrease or stay the same | may increase, decrease or stay the same |

# Classification

## ▼ Confusion Matrix



| | | Predicted class | |
|---|---|---|---|
| | | P | N |
| Actual class | P | True positives (TP) | False negatives (FN) |
| | N | False positives (FP) | True negatives (TN) |

### Accuracy

% predicted labels that match true labels.

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

### Precision

% of the number of accurate positives the model claims compared to the total number of positives it claims.

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

### Recall

% of the number of positives the model claims compared to the actual number of positives in the data.

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

### Specificity

% of the number of negatives the model claims compared to the actual number of negatives in the data. (a.k.a true negative rate)

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

$$\text{False positive rate} = 1 - \text{specificity}$$

### F1 Score

The harmonic mean of precision and recall.

- 0 → worst, 1 → best (perfect precision and recall).

$$F_1 = \frac{2}{\frac{1}{\text{precision}} + \frac{1}{\text{recall}}} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

## ROC Curve and Precision-Recall Curve

### ROC Curve

🎷 • **x-axis:** False Positive Rate
• **y-axis:** Recall (TPR)
• The area under a ROC curve is a more **systematic metric** comparing binary classification models because it’s independent of how we set the threshold.

### Precision-Recall Curve

🛼 • **x-axis**: Recall
• **y-axis**: Precision
• Puts emphasis on predictions the model got right out of the total number it predicted to be positive (i.e. precision).
• Precision-Recall curves give more importance to the **positive class** which is useful when the classes are imbalanced.