Part 1

MACHINE LEARNING INTERVIEW QUESTIONS

SWIPE →

Q1: What is the difference between supervised and unsupervised machine learning?

Supervised learning requires training labelled data.

For example, in order to do classification (a supervised learning task), you'll need to first label the data you'll use to train the model to classify data into your labelled groups.

Unsupervised learning, in contrast, does not require labelling data explicitly.

Q2: What is regularization? Can you give some examples of regularization techniques?

Regularization is any technique that aims to improve the validation score, sometimes at the cost of reducing the training score.

Some regularization techniques:

- L1 tries to minimize the absolute value of the parameters of the model. It produces sparse parameters.
- L2 tries to minimize the square value of the parameters of the model. It produces parameters with small values.

Q3: What is stratified cross-validation and when should we use it?

Cross-validation is a technique for dividing data between training and validation sets. On typical cross-validation, this split is done randomly. But in stratified cross-validation, the split preserves the ratio of the categories on both the training and validation datasets.

Stratified cross-validation may be applied in the following scenarios:

- On a dataset with multiple categories
- On a dataset with data of different distributions

Q4: What is deep learning, and how does it contrast with other machine learning algorithms?

Deep learning is a subset of machine learning that is concerned with neural networks: how to use backpropagation and certain principles from neuroscience to more accurately model large sets of unlabelled or semi-structured data.

In that sense, deep learning represents an unsupervised learning algorithm that learns representations of data through the use of neural nets.

Q5: How would you handle an imbalanced dataset?

An imbalanced dataset is when you have, for example, a classification test and 90% of the data is in one class. That leads to problems: an accuracy of 90% can be skewed if you have no predictive power on the other category of data! Here are a few tactics to get over the hump:

- Collect more data to even the imbalances in the dataset.
- Resample the dataset to correct for imbalances.
- Try a different algorithm altogether on your dataset.

Q6: Why do ensembles typically have higher scores than individual models?

An ensemble is the combination of multiple models to create a single prediction. The key idea for making better predictions is that the models should make different errors. That way the errors of one model will be compensated by the right guesses of the other models and thus the score of the ensemble will be higher.

Many winning solutions to data science competitions are ensembles. However, in real-life machine learning projects, engineers need to find a balance between execution time and accuracy.

Q7: What evaluation approaches would you work to gauge the effectiveness of a ML model?

You would first split the dataset into training and test sets, or perhaps use cross-validation techniques to further segment the dataset into composite sets of training and test sets within the data.

You should then implement a choice selection of performance metrics. You could use measures such as the F1 score, the accuracy, and the confusion matrix. What's important here is to demonstrate that you understand the nuances of how a model is measured and how to choose the right performance measures for the right situations.

Q8: What's the "kernel trick" and how is it useful?

The Kernel trick involves kernel functions that can enable in higher-dimension spaces without explicitly calculating the coordinates of points within that dimension: instead, kernel functions compute the inner products between the images of all pairs of data in a feature space.

This allows them the very useful attribute of calculating the coordinates of higher dimensions while being computationally cheaper than the explicit calculation of said coordinates. Many algorithms can be expressed in terms of inner products. Using the kernel trick enables us effectively run algorithms in a high-dimensional space with lower-dimensional data.

Data-Driven Science

FOUND IT HELPFUL?

Let us know in the comments

At DDS, we will help you build a career in Data Science

LINK IN BIO