Qafza Advanced Training
Task 02
Chapter 2 Summary
Lubna Nedhal Kan’an

Designing Machine Learning (ML) systems involves integrating various components to work together effectively. This holistic approach includes understanding business needs, managing data, selecting the right technology, implementing the system, and monitoring performance.

The first step is to establish clear objectives aligned with business goals, which guide the development of ML models. Next, foundational requirements must be set, focusing on:

- **Reliability:** Ensuring consistent performance.
- **Scalability:** Handling increased workloads.
- **Maintainability:** Facilitating easy updates and fixes.
- **Adaptability:** Allowing flexibility for changes.

An iterative design process follows, ensuring alignment with objectives and allowing for continuous improvement. Clearly framing the problem to be solved is crucial, as this impacts the complexity of the work.

Finally, since data is fundamental to ML, its significance in system design cannot be overstated. The ongoing debate in the ML field centers on whether quality data or advanced algorithms are more important for success. By grasping these core principles, practitioners can create effective ML systems that meet business demands and adapt to technological changes.

Business and Machine Learning Objectives

When starting a Machine Learning (ML) project, it's important to focus on the right objectives. Data scientists often get caught up in ML-specific metrics like accuracy and performance improvements. However, companies care more about how these projects impact their business goals, such as increasing profits and customer satisfaction.

A successful ML project should connect its results to business performance metrics. For instance, if you work for an e-commerce site focused on boosting the purchase-through rate, transitioning to real-time recommendations could help users make purchases more effectively. Improved accuracy in recommendations can lead to higher sales.

Some common ML applications, like predicting ad click-through rates, show clear links to business outcomes. Companies may create their own metrics to connect ML performance with business results. For example, Netflix measures its recommendation system's effectiveness through its "take-rate," which helps understand user engagement and subscription retention.

Understanding how ML metrics affect business metrics often requires experiments, such as A/B testing. These tests help determine which models perform better in driving business success, even if they don't have the best ML metrics.

However, it can be challenging to trace the impact of ML models on business results. For example, in a cybersecurity context, ML may help detect anomalies, but if a threat slips through, it’s hard to know whether the ML system was at fault.

Many companies promote themselves as "AI-powered," but simply using ML doesn't guarantee success. Expectations should be realistic; while ML can offer significant benefits, results usually take time and sustained investment. Companies like Google have reaped the rewards of long-term investment in ML.

The success of ML projects often depends on how long a company has been using it. More experienced companies can deploy models faster and more efficiently, leading to better returns on investment. In contrast, newer adopters may face longer deployment times and higher costs, impacting their overall success with ML.

Key Characteristics of Machine Learning Systems

Building a successful machine learning (ML) system involves meeting specific requirements, which can differ based on the use case. However, there are four main characteristics that most ML systems should have: reliability, scalability, maintainability, and adaptability.

Reliability

An ML system should consistently perform its functions correctly, even when faced with challenges like hardware failures or human mistakes. Determining if a system is correct can be tricky, especially when the predictions it makes can’t be easily verified against known outcomes. Unlike traditional software that might show clear errors, ML systems can fail silently, making it hard for users to realize something is wrong.

Scalability

ML systems need to handle growth in several ways, including:

1. **Complexity**: As models become more complex, they may require more computational resources.
2. **Traffic Volume**: The number of predictions requested can increase significantly as a user base grows.
3. **Model Count**: Initially, a system might use one model, but as needs expand, multiple models may be required for different tasks.

Managing this growth involves scaling resources up or down as needed and automating the monitoring and updating processes to handle multiple models effectively.

Maintainability

Different team members, including engineers and subject matter experts, will work on an ML system. It’s essential to organize the workload so everyone can use tools they are comfortable with. This includes documenting code, versioning data and models, and ensuring that the system is reproducible. Effective collaboration and problem-solving are vital for maintaining the system.

Adaptability

ML systems should be flexible enough to adjust to changes in data and business needs. Since both the code and the data can evolve quickly, it's important for the system to update without causing disruptions. This adaptability is linked closely to maintainability, as both aspects focus on ensuring the system can grow and change over time.

The Iterative Process

Developing an ML system is an ongoing, iterative process. Once a system is deployed, it requires constant monitoring and updates. The workflow includes:

1. **Project Scoping**: Establishing goals, resources, and stakeholders.
2. **Data Engineering**: Collecting and preparing data for training models.
3. **Model Development**: Extracting features and creating models.
4. **Deployment**: Making the model accessible to users.
5. **Monitoring and Continual Learning**: Ensuring models perform well and adapt to changes.
6. **Business Analysis**: Evaluating model performance against business objectives and making strategic decisions.

Overall, successfully building and maintaining an ML system requires careful attention to these characteristics and a commitment to ongoing improvement.

Framing Machine Learning Problems

Imagine you're a tech lead at a bank that wants to improve its customer service for millennial users. Your boss learns that a rival bank is using machine learning (ML) to process customer requests faster and wants your team to do the same. While slow customer support is a problem, it's not directly an ML problem. To define it as an ML problem, you need to identify inputs, outputs, and the goal of the ML model.

Upon investigation, you find that the main issue is how to route customer requests to the correct department—accounting, inventory, HR, or IT. You can solve this by creating an ML model that predicts which department should handle each request. This turns into a classification problem, where the input is the customer request, the output is the department, and the goal is to accurately predict the correct department.

Types of ML Tasks

The output of your ML model defines the type of task. The two main types of ML tasks are classification and regression.

- **Classification** involves categorizing inputs into different classes, like deciding if an email is spam or not.

- **Regression** involves predicting continuous values, such as estimating a house price.

You can convert a regression problem into a classification problem by grouping continuous values into categories, and vice versa by turning categories into a continuous output.

Binary vs. Multiclass Classification

Classification problems can be simple or complex. Binary classification has only two classes, like determining if a transaction is fraudulent. It's easier than multiclass classification, where there are more than two classes. The more classes you have, the harder it can be to manage data and make accurate predictions, especially if some classes have very few examples.

For complex cases with many classes, hierarchical classification helps. You first classify into broad categories and then into specific subcategories.

Multiclass vs. Multilabel Classification

In multiclass classification, each example belongs to only one class. In multilabel classification, an example can belong to multiple classes. For example, an article could be tagged as both tech and finance. There are two ways to handle multilabel classification: treat it like multiclass or create separate binary models for each class.

Multilabel classification can be challenging due to varying numbers of classes per example, leading to complexities in labeling and prediction.

Framing Problems Differently

How you frame a problem can significantly affect its difficulty. For example, predicting which app a user will open next could initially be framed as a multiclass classification problem, predicting the likelihood for each app. However, it's more effective to frame it as a regression problem that gives a score for each app's likelihood, simplifying updates when new apps are added.

Objective Functions

An ML model needs an objective function, also called a loss function, to guide its learning. This function measures how well the model's predictions match the actual labels. Common loss functions are used depending on the type of task—like mean squared error for regression or cross-entropy for classification.

Decoupling Objectives

Framing problems can be complicated when you need to balance multiple objectives. For example, if you're ranking posts on a newsfeed to maximize engagement, you might unintentionally promote extreme content. To address this, you might add objectives like filtering out spam and misinformation. This can create conflicts between the goals of maximizing

engagement and ensuring content quality, making it important to carefully define and balance these objectives in your ML model.

Mind Versus Data in Machine Learning

Over the last ten years, it has become clear that the success of machine learning (ML) systems largely depends on the quality and quantity of the data they are trained on. Instead of just improving ML algorithms, many companies are now focusing on how to manage and enhance their data.

Despite the effectiveness of models that use vast amounts of data, some experts remain skeptical about relying too heavily on data. At academic conferences, there have been ongoing debates about the importance of human insight versus data. Proponents of using human reasoning argue that intelligent design can outperform models that simply process lots of data. However, there is a trade-off: focusing on one often limits time spent on the other.

Dr. Judea Pearl, a respected figure in the field, argues that "data is profoundly dumb" and believes that over-reliance on data-centric methods may leave those in the field outdated in a few years. In contrast, Professor Christopher Manning from Stanford suggests that using enormous amounts of data with basic algorithms leads to poor learning outcomes, and emphasizes the need for smarter system designs that can learn effectively from smaller datasets.

On the other side of the argument, experts like Professor Richard Sutton highlight that methods which effectively utilize computation will be more successful in the long run. Google’s Peter Norvig backs this up by stating that Google Search excels not because of superior algorithms, but because of the vast amounts of data it processes. Dr. Monica Rogati reinforces that data is fundamental to data science and essential for improving products and processes.

The debate centers not on whether we need data but on whether large datasets alone are sufficient for success. Historically, the growth in dataset sizes used for training ML models has been staggering—from 0.8 billion tokens in 2013 to 500 billion tokens for OpenAI's GPT-3. However, it's important to note that simply having more data doesn’t guarantee better performance. Low-quality data can hinder model effectiveness.

This discussion emphasizes the need for a clear understanding of the goals of any ML project. Each project should start with a business objective, which then informs the ML objectives guiding the development of models. The success of an ML system depends on various requirements, including reliability, scalability, maintainability, and adaptability.

Building an ML system is not a one-time task but an ongoing process. The role of data in these systems remains critical, and while the debate over the supremacy of data versus intelligent design continues, the evidence shows that access to large datasets has driven significant advancements in ML.

In summary, the next chapters of this book will delve into specific aspects of building effective ML systems, starting with the basics of data engineering. By addressing the challenges discussed here, future examples will help clarify these concepts.