FOR THE
IB DIPLOMA
PROGRAMME

IN COOPERATION WITH ib

Computer Science

Paul Baumgarten
Ioana Ganea
Carl Turland

hachette
LEARNING

A4.1 Machine learning fundamentals

What principals and approaches should be considered to ensure machine learning models produce accurate results ethically?

SYLLABUS CONTENT

By the end of this chapter, you should be able to:

- A4.1.1 Describe the types of machine learning and their applications in the real world
- A4.1.2 Describe the hardware requirements for various scenarios where machine learning is deployed

◆ **Generative AI:** a form of artificial intelligence capable of generating text, images, audio, video and other digital artefacts, usually in response to a prompt. It is a form experiencing rapid advances at the time of writing.

◆ **Machine learning:** a branch of AI where computers learn from data and experiences to perform specific tasks or solve specific problems, without being explicitly programmed to do so.

◆ **Artificial intelligence:** computer technology able to perform tasks and make decisions in a manner that imitates human intelligence. There are two main forms of AI: narrow (or weak) AI is designed to perform specific tasks or solve specific types of problems; general (or strong) AI processes human-level intelligence and can operate across a range of domains. While speculation persists that general AI is "close", at this time only narrow AI technology is available.

A4.1.1 Types of machine learning and their applications

TOK

What counts as knowledge?

Machine learning models "learn" from data, which raises questions about what constitutes knowledge.

Views on knowledge often distinguish between knowledge gained through experience (empirical) and knowledge gained through reasoning (rational). Machine learning models acquire knowledge empirically by processing vast amounts of data. However, unlike humans, machines do not "understand" or reason about this data in the human sense. This raises the question: Can the patterns and predictions that machines generate be considered "knowledge", or are they simply data-processed outputs?

Welcome to the world of machine learning! We live in a time of exciting growth and rapid innovation in machine learning. **Generative AI** is making global headlines and has changed the way we live and work in a very short timeframe. Speculation is rife that "general AI" is not far from becoming reality. Certainly, it is an exciting topic, but what are machine learning and artificial intelligence, and how do they work? Gaining an understanding of what is happening behind the scenes is the goal of this chapter.

This chapter will not seek to dissect the details of the latest, greatest, news-making developments in the field. That would be a fool's errand as it would be obsolete before the book is printed. Instead, the aim is to give you a solid understanding of the core theories and techniques that form the basis of the entire field of machine learning. From these foundations, you will be in a much stronger position to understand the true implications of modern developments occurring in the field.

Before proceeding any further, it is important to clarify and differentiate between the terms **machine learning** (ML) and **artificial intelligence** (AI). Artificial intelligence is a broad field that seeks to create systems capable of performing tasks that typically require human intelligence.

This can include, but is not limited to, reasoning, learning, perception, problem-solving, understanding and interaction. Machine learning is a subset of artificial intelligence that focuses on the learning aspect of AI. It seeks to teach computers to learn from data, identify the patterns in that data and make decisions based on what it has learned, with minimal human intervention. Implementing machine learning programmatically is heavily reliant on the mathematics of statistics, linear algebra and calculus.

Machine learning applications are being increasingly used throughout commerce, industry, research and government. They are used for everything from market analysis to robotics; from generative art to diagnosing medical conditions. The applications for machine learning will only grow as the technology continues to develop.

Within machine learning, there are many further subcategories we will consider in A4.3 Machine learning approaches. These can be broadly described as:

- supervised learning: linear regression
- supervised learning: classification
- unsupervised learning: clustering
- unsupervised learning: association rule
- reinforcement learning
- genetic algorithms
- artificial neural networks
- convolutional neural networks.

Top tip!

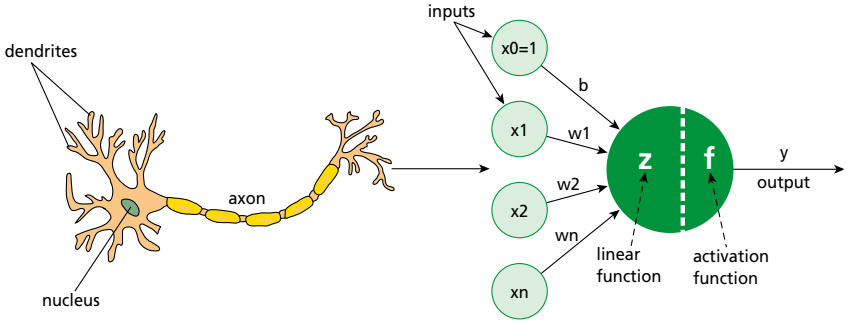
Take the time to appreciate the differences between types of machine learning: supervised, unsupervised, reinforcement, deep learning and transfer learning. Know what scenarios each is best suited for, and the typical algorithms used in each category. In this topic, terms and definitions are foundational for answering theoretical questions accurately. Using terminology in an incorrect context will cost marks.

Deep learning

◆ Neural network: a computer algorithm that imitates the design of the human brain by using a set of interconnected nodes for the processing and analysing of data.

The term "deep learning" is used to imply the use of a **neural network** within a machine learning algorithm. There are a variety of machine learning techniques that work perfectly fine without the need for a neural network, so the "deep learning" term is used to distinguish between those that do and those that do not make use of a neural network. For example, you can refer to "reinforcement learning" and "deep reinforcement learning".

A neural network is where algorithms and data structures have been constructed in such a manner as to replicate biology's understanding of how the brain functions: as an interconnected network of neurons, each of which has various input connections and generates an output on the basis of the combination of inputs.



Comparison of a biological neuron with that used by artificial neural networks

A more detailed examination of how neural networks function will be provided in A4.3.8 Artificial neural networks.

Common mistake

Deep learning is a subset of machine learning. Deep learning is not separate from machine learning, but rather is a specific approach within it. It utilizes layers of neural networks to extract progressively higher-level features from the input. Machine learning includes many other types of algorithms that do not require neural networks.

Supervised learning

Supervised learning refers to an algorithm that is trained on labelled data sets. These data sets comprise example input values, and the correct output response that should be given if the algorithm sees something resembling that input. Generally, the larger and better the data set, the more accurate the results that will be produced by the supervised learning algorithm. Data sets used by the major technology companies contain many millions of records.

Supervised learning can be used for regression and classification tasks.

A **regression** task is where the algorithm is predicting a numerical value for the output within an allocated range, for example:

- A grade-prediction algorithm might take inputs of hours studied, attendance record, class participation, scores on previous tests, hours spent on homework; and output a final predicted grade in the range 0–100.
- A weather-forecasting algorithm might take inputs of historical temperatures for each day over the last week, humidity, wind speed, air pressure; and output a predicted temperature for the coming day in a given range.

A **classification** task is where the algorithm predicts which category the input item belongs to, for example an image recognition algorithm might input an image and seek to classify it as either a dog or a muffin.

◆ **Supervised learning:** when a machine learning algorithm is provided a data set of pairs of items, where the pair comprises a value and what response the network should provide if it sees that value. By learning the answers to the values given, the network will make generalizations to be able to estimate the answer when given a previously unseen value.

◆ **Regression:** machine learning where the output generated should be a numerical value.

◆ **Classification:** machine learning where the output generated should be a category, chosen from among a discrete set of categories available.

Common mistake

Confusing the goals of regression and classification

Be clear about the difference in outputs between regression and classification tasks in supervised learning. Regression models predict a continuous output (numerical values), whereas classification models predict categorical outputs (class labels). For example, predicting the price of a house based on its features (like size and location) is a regression problem because price is a continuous variable. On the other hand, determining whether an email is spam or not spam is a classification problem because there are discrete categories (spam or not spam) to choose from.

A music genre classification algorithm may input song tempo, rhythm, pitch, instruments used; and output the music genre as either pop, rock, hip-hop, classical, and so on.

A handwriting recognition algorithm may input an image of a character and seek to classify it as an individual letter, number or punctuation mark.

Unsupervised learning

◆ **Unsupervised learning:** a method of machine learning where the data set does not include the "answers" or expected outputs for the data provided. The algorithm will attempt to discover the patterns on its own.

◆ **Reinforcement learning:** machine learning by trial and error. Based on what it has learned at any moment in time, the algorithm selects an action to take in a given environment. The environment provides feedback (called a "reward"), which the algorithm will use to learn from and refine its decision-making process moving forward.

Unsupervised learning is where the algorithm is constructed to identify patterns or structures within its data sets without being provided with an explicit label indicating the correct output. This may be because the nature of the data involved doesn't lend itself to having a "correct" response paired with it, or because the algorithm is constantly learning based on user interactions that don't have a fixed right or wrong answer. Examples include:

- An algorithm that seeks to identify a user's social group: The input data may consist of social-media activity such as likes, comments and follows. The algorithm could analyse this data to identify other users with mutual acquaintances or similar interests. Interestingly, this type of social-group analysis can take place without needing any content from the messages or chats between the parties involved. This is why social-media companies such as WhatsApp are perfectly happy to offer end-to-end encrypted messaging as, even without the message content, just knowing how many messages are exchanged between each pair of users is enough to perform social-group analysis.
- Retail stores use unsupervised learning to find associations and correlations between the different products that customers purchase, and identify similarities in purchasing behaviour and preferences. The reason that so many brands run customer loyalty schemes is it allows them to build a profile of data to match against other customers, from which they can tailor marketing strategies.
- Media companies such as Netflix, Spotify and YouTube use unsupervised learning to train recommendation systems to refine their suggestions to users for future watching or listening.

Reinforcement learning

Reinforcement learning is where the algorithm looks at its input data and decides on a particular output, and is then informed how good or bad that decision was after the fact. It uses that information to refine future actions when presented with a similar situation. Reinforcement learning can be thought of as learning from trial and error.

Some common situations where reinforcement learning is used include:

- **Gaming:** Reinforcement learning algorithms can be trained to act as AI players or bots within computer games.
- **Robotics:** Reinforcement learning can be used to teach a robot how to walk, pick up objects or perform other mechanical tasks. As a subtype of robotics, autonomous self-driving cars also make use of reinforcement learning to better and more safely navigate the complexities of roads and traffic.
- **Finance:** Reinforcement learning bots can trade securities on the market and receive feedback based on whether the bot made or lost money on the trade.
- **Recommendation systems:** Reinforcement learning can also be part of a suite of algorithms used in generating user recommendations. The engagement of the user (did they watch or listen to the suggested item?) can be used to provide feedback to the algorithm to refine future recommendations.

◆ **Transfer learning:** when a previously trained machine learning model is applied to a similar yet new situation, context or problem. The goal is to speed up the training process by using an already trained model, even if the problem is slightly different.

Transfer learning

Transfer learning is where the knowledge gained from solving one problem can be used to help solve a different but somewhat related problem. The benefit of transfer learning is that it requires less data, as the algorithm is already partially trained and may just require a little fine-tuning for the new task being asked of it.

Consider the following examples:

- **Image recognition:** Given a model that has been trained on a massive data set such as ImageNet (over a million labelled images and 1000 different categories), transfer learning could take that model and fine-tune it to recognize specific types of objects, such as a species of flower or breed of dog. The model would already be adept at processing images and easily able to identify features such as edges and shapes, so it would only need to learn how to distinguish between the new categories.
- **Speech recognition:** Using a generalized model that has been trained on spoken language to transcribe it into text, transfer learning can be used to adapt it to work with particular accents or specialized jargon for use within a particular industry.
- **Customized chatbot:** By using a publicly available pre-trained LLaMA (large language model meta AI), a company might fine-tune it by training it on customer-service logs to create a chatbot that can be added to its website for handling domain-specific queries.
- **Customized image generators:** Pre-trained models for tools such as Stable Diffusion can be further extended and fine-tuned to generate images that mimic a particular artistic style, or be specialized in images for a particular industry or domain. This can be done relatively quickly and easily without the burden of redoing the massive task of original training that went into the underlying model.

Common mistake

Transfer learning is not just about using a pre-trained model. It involves adapting a model developed for one task to solve a related one; not just reusing an existing model without modifications. It's crucial where data is scarce or similar tasks are involved.

A4.1.2 Hardware requirements

The hardware required for machine learning purposes will continue to innovate and evolve throughout the lifetime of this text. Accordingly, this section is not going to make recommendations as to specific model numbers of processors, but will rather discuss the broad categories of hardware technology available and their various use cases.

Computing platforms

Standard laptops

The starting point is obviously the standard laptop available on the retail market. At the time of writing, this might be an i7 processor with 16 GB or 32 GB of RAM, or an Apple Silicon equivalent.

These machines are generally limited to small-scale machine learning tasks, such as the development and testing of a simple machine learning model. For educational purposes, there is a lot that can be done with a standard laptop, but you would not want to be training a commercial-grade machine learning model with such equipment as it would be too slow, and lack sufficient memory or storage.

Some recent developments do aim to improve the capacity of standard laptops when it comes to machine learning. One is the introduction of Apple Silicon M processors into Apple MacBooks. Apple integrates the CPU, GPU, neural engine and other components into a single system-on-a-chip (SoC) structure, allowing better performance and energy efficiencies. By integrating the CPU and GPU functions on to a single chip, they pool and share the same

memory. This is in contrast to the traditional approach of GPUs having their own dedicated memory, separate from the RAM used by the CPU. This is why those with an Apple Silicon-based computer are often able to perform machine learning tasks that traditional Intel laptop owners are unable to do without access to a dedicated GPU.

Not wanting to allow Windows users to be left behind, Microsoft has launched its Microsoft Copilot AI-supported branding, which requires laptops to have an integrated neural processing unit (NPU), which is discussed further in the section regarding CPUs coming up.

Dedicated workstation

After a standard laptop, the next step would be the purchase of a dedicated desktop workstation with a GPU, such as an NVIDIA RTX.

Having a true GPU can offer an order of magnitude improvement in processing speeds for machine learning calculations and would serve as an excellent platform for some quite sophisticated projects.

The primary advantage of a GPU is its parallel processing capabilities, which come from having thousands of small processing cores that are optimized for parallel processing. Machine learning algorithms often involve performing the same computations on large amounts of data. GPUs can perform these same calculations on different values simultaneously, whereas a CPU has to queue them up for processing one by one.

Edge devices

Edge devices refer to computing systems that perform data processing at or near the location where data is being generated, rather than relying on centralized computing resources such as the cloud.

Processing data locally reduces the need to send data back and forth to a distant data centre. This reduction in data being transmitted has the added benefit of improving privacy and security.

The downside is that you are still committed to investing in the physical hardware infrastructure yourself, along with all the maintenance workload associated with it.

Cloud-based platforms

To perform training on large or complex models generally requires the use of online cloud-based platforms (in lieu of investing in the massive infrastructure yourself). Cloud platforms are accessible over the internet and provide services on demand to users worldwide.

These cloud providers allow you to vary the combination and specifications of CPUs, GPUs and Tensor Processing Units (TPUs) available for your project on demand. They can also scale to provide large quantities of RAM, storage and network connectivity, as required. The cloud-based services are also useful for deployment of your model as an API for other systems to access.

The main downside with cloud-based platforms is the dependency and reliance your project will have on an external provider. You have to trust their data and network security arrangements; you have to transmit your data to their network to have it perform tasks for you; and you are committing yourself to the monthly subscription costs involved. The flexibility of cloud-based systems always comes with a cost, and this should not be treated lightly.

At the time of writing, the major industry leaders that provide cloud-based platforms with machine learning specialist equipment available include AWS, Google Cloud and Microsoft Azure. A good tool for getting started with minimal set-up requirements is Google Colab; it allows you to create a Python Notebook and utilize GPU or TPU technology just by changing the settings in the Runtime menu.

High-performance computing (HPC) centres

In contrast to the publicly available, user-pays approach of cloud-based providers, HPC centres are dedicated facilities designed to support large-scale scientific or academic research objectives. In this way, access to an HPC is more restricted, often requiring membership; affiliation with an academic or research institution; or specific research grants or time allocation processes.

They are data centres that have been designed to be suitable for highly demanding workloads that require sustained high-performance computing resources. They are built around a model of catering to resource-intensive computational tasks, not an as-a-service model.

Many universities have made investments in their own HPCs for use by their research students.

Processors for machine learning

Having considered the various platforms available for accessing the computing power necessary for machine learning, it is time to review the electronics within the computers that make machine learning happen.

Central processing units (CPUs)

CPUs are the generalized processors inside all modern computer systems. They are designed to perform a wide range of computing operations, are highly flexible and can process complex tasks. They are not specialized devices designed specifically for machine learning. While it is feasible to perform some introductory machine learning tasks with a CPU, they are generally limited to tasks that do not require intensive parallel processing.

Neural processing units (NPUs) have recently been integrated alongside traditional CPUs in consumer-level laptops. NPUs are specialized processors designed specifically to handle the computations required for neural networks and deep learning, such as matrix and vector operations. By having specialized processors in the computing device, it provides faster processing times and lower power consumption for AI-related tasks, compared to general-purpose CPUs.

As of 2024, laptops marketed as being Microsoft Copilot AI-supported include NPUs with a minimum capability of 40 TOPS (trillion operations per second).

Graphics processing units (GPUs)

GPUs contain hundreds or thousands of small cores designed for highly parallel tasks such as rendering graphics. The GPU allows all the cores to perform the same calculation on different values simultaneously, so if there are large arrays that need processing, where every element requires the same operation performed, GPUs provide significant time savings. GPUs excel at parallel processing of matrix and vector operations, which is the very mathematics that forms the basis of neural networks.

The presence of a dedicated GPU can often produce training speed improvements of up to ten times over using just a CPU.

◆ **Tensor:** a mathematical term for an array with three or more dimensions. A single number (no dimensions) is known as a "scalar". A one-dimensional array of numbers is known as a "vector". A two-dimensional array of numbers is known as a "matrix". Three or more dimensions is known as a "tensor".

Tensor Processing Units (TPUs)

Building on the idea of the GPU, the TPU was custom-designed by Google specifically for **tensor** computations. They are optimized for high volume, low precision calculations to increase the efficiency of neural network tasks. Low precision in this context typically means calculations occur at a maximum of 16 bits, in contrast to the 32 bits or 64 bits in a normal GPU. Machine learning generally does not require that level of precision, so 16 bits or even 8 bits will do the job.

At the heart of a TPU is a large matrix multiplication unit. Matrix multiplication is fundamental to neural networks, so having a unit within the processor specifically optimized for this task helps make TPUs well suited for machine learning.

The TensorFlow library is tailored to make use of TPUs when available, and Google Cloud services, such as Google Colab, make TPUs easily available for the general public.

Application-specific integrated circuits (ASICs)

ASICs are custom-designed for a specific use rather than general-purpose computing. They are engineered to perform a particular set of tasks with optimal efficiency. They offer peak performance and efficiency for these tasks, but lack the general-purpose flexibility of a CPU.

If your machine learning workload can be precisely defined and won't change much over time, an ASIC may perform these tasks faster than a GPU or TPU as, while these are optimized for parallelism, they are still generalized processors.

Due to the degree of specialization involved, ASICs tend to be more energy efficient and have lower operating costs over the long term. The downside is that the upfront cost is typically very high as the chips require custom design and development. This means they are really only viable where a machine learning application is going to be deployed on a very large scale, as the per-unit cost of the ASIC will decrease significantly with scale when mass-produced.

Examples of well-known, mass-produced ASICs include the Apple A-series chips used in iPhones and Qualcomm's Snapdragon.

You should conduct some research into the current state-of-the-art ASICs available for machine learning operations at the time of reading, and be familiar with what differentiates them from just using a typical GPU or TPU.

Field-programmable gate arrays (FPGAs)

FPGAs can be programmed and reprogrammed to perform specialized computing tasks, offering a balance between the flexibility of CPUs / GPUs and the efficiency of ASICs.

As such, they are ideal for prototyping machine learning models or applications that require custom hardware acceleration, however that may change over time.

FPGAs are used for high-frequency trading systems where microseconds can make a significant difference in the profitability of trades.

Common mistake

Confusing the differences between each of the processor types

There are a lot of separate technologies listed in this topic, many of which you will not have had personal hands-on experience with. That makes it harder to have an intuitive understanding of the differences between them.

- **ASICs** are designed for specific tasks and are not reprogrammable.
- **FPGAs** are versatile and can be reprogrammed.
- **GPUs** are great for parallel processing tasks.
- **TPUs** are specialized chips designed by Google, optimized for tensor calculations in deep learning for large-scale models.
- **NPUs** are designed to accelerate neural network computations for consumer-grade devices.

Top tip!

Adapt the following as a guide to help determine which is the best device for a given scenario.

- For large and complete models, does it require real-time processing?
  - ☐ Yes: Consider GPUs for their parallel-processing capabilities
  - ☐ No: TPUs might be a better choice for batch processing with high efficiency in tensor operations
- For real-time inference (using a model for decision-making after training), is the model deployed on edge devices?
  - ☐ Yes: NPUs or ASICs, for optimized power and efficiency
  - ☐ No: Consider FPGAs for flexibility or ASICs for efficiency if the task won't change
- For models requiring future flexibility, are future updates expected?
  - ☐ Yes: FPGAs, due to their reprogrammability
  - ☐ No: ASICs or GPUs, depending on whether the task is more about speed or parallel processing
- Is low cost more important than cutting-edge performance?
  - ☐ Yes: Consider older generation GPUs or cloud-based solutions where hardware costs can be easily absorbed
- Will there be a need to quickly scale processing power?
  - ☐ Yes: Cloud GPUs or TPUs can offer scalable resources as required

REVIEW QUESTIONS

1 A hospital is integrating a system that can automatically diagnose diseases from patient-imaging data.
   - **a** Describe whether this system should be classified as artificial intelligence, machine learning or deep learning.
   - **b** Distinguish between regression-based and classification-based machine learning.

2 An email client uses a program to sort incoming emails into "Primary", "Social", "Promotions" and "Spam" folders.
   - **a** Identify whether this is an example of supervised or unsupervised learning.
   - **b** Describe your reasoning for this choice.

3 An autonomous vehicle company transfers the knowledge from a model trained in one city to a new model designed to navigate another city.
   - **a** Define "transfer learning".
   - **b** Outline how this is an example of transfer learning.
   - **c** Outline one possible limitation to the effectiveness of this approach.
   - **d** The original model was trained from thousands of hours of driving on roads under human supervision to monitor and correct it when required. Describe the form of machine learning used for the original model.

4 A tech start-up is planning to deploy a large-scale machine learning system to predict stock prices in real time.
   - **a** Identify one type of hardware that would be critical for processing large volumes of real-time data in this context.
   - **b** Outline one reason that this type of hardware is suitable for real-time data processing in machine learning applications.
   - **c** Discuss one potential limitation of the identified hardware when used for machine learning.

5 A university plans to implement an AI-driven system to analyse video lectures for enhancing online learning experiences.
   - **a** Identify two types of hardware that could be used for conducting machine learning processing of video data in real time.
   - **b** For the two types of hardware identified, outline one possible reason for selecting each device over the other.

A4.2 Data preprocessing (HL)

Top tip!

Spend significant time on data preprocessing, visualization and analysis. Understanding the data is as important as understanding the algorithms.

SYLLABUS CONTENT

By the end of this chapter, you should be able to:

- A4.2.1 Describe the significance of data cleaning
- A4.2.2 Describe the role of feature selection
- A4.2.3 Describe the importance of dimensionality reduction

A4.2.1 Data cleaning

High-quality data builds high-quality models. If the training data is full of errors or redundant features, the model will learn from these inaccuracies and make poor predictions.

Taking the time to ensure your data is as clean as possible will reap rewards with respect to efficiency and accuracy. There are several steps that may be useful for cleaning your data set.

1 **Handling outliers**: Statistical methods, such as using the interquartile range or Z-scores, can detect outlying data. Once found, depending on the context, outlying data may be capped, transformed or removed as appropriate.

◆ **Outlier:** a data point that deviates from the typical pattern of values in a data set, indicating a possible unusual or erroneous value that should be discounted.

Python

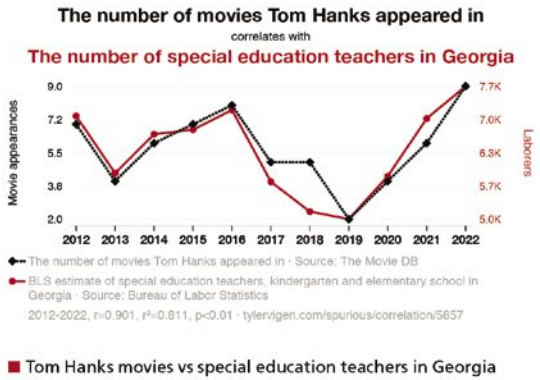
```
import numpy as np
# Create random array of values between 0 and 100
# Set one extreme value to act as an outlier
data = np.random.randint(0, 100, size=1000)
data[999] = 937
# Calculate outliers via Z-scores
mea = np.mean(data)
std_dev = np.std(data)
z_scores = (data - mean) / std_dev
threshold = 3 # Outliers if 3 stddev from mean
outliers = data[np.abs(z_scored) > threshold]
print("mean", mean, "stddev", std_dev)
print("Outliers:", outliers)
# Calculate outliers via IQR
q1 = np.percentile(data, 25)
q3 = np.percentile(data, 75)
iqr = q3-q1
cutoff = 1.5 * iqr
lower_bound = q1 - cutoff
upper_bound = q3 + cutoff
outliers = data[(data < lower_bound) | (data > upper_bound)]
print("Outliers:", outliers)
```

2 **Removing duplicate data:** Identifying and removing duplicate data will assist in preventing the model from becoming biased towards over-represented values. For data sets where individual records contain a large number of variables, calculating and comparing SHA256 hash values can be a useful mechanism for detecting duplicates (see Section B4.1.6 for more about hash values). Depending on the context of the model, near-duplicate data may also need to be consolidated into a single record.

3 **Identifying incorrect data:** Process your data through validation rules to ensure obviously incorrect data can be found and removed. This may mean checking the ranges given for dates and times, or amounts given for currency values, and so on. Set sensible limits and have your program detect anomalies for possible manual checking.

4 **Filtering irrelevant data:** If there is no measurable correlation between an input variable and the outcome variable, it may be completely irrelevant and contribute nothing to the predictive power of the model. Keeping such data in the training process is only going to make the process less efficient and less accurate.

Additionally, just because data may appear to be correlated doesn't mean it is. As the Spurious Correlations website demonstrates, if you compare enough unrelated data sets, you will find correlations that are, in fact, not.



■ Tom Hanks movies vs special education teachers in Georgia

■ Robberies in Alaska vs professor salaries

5 **Transform improperly formatted data:** Data may be incorrectly formatted but easily correctable to ensure consistency in what is presented to the machine learning model, for example:

- ☐ Ensure all dates are in a consistent style (not having a mix of day / month / year, month / day / year, or ISO yyyy-mm-dd formats).
- ☐ Ensure numerical values are formatted, and to the same level of precision.
- ☐ Ensure images are correctly rotated and oriented, and of matching ratio and size.

6 **Missing data:** Sometimes it may be necessary to use models to predict missing values to ensure full coverage of the data set. Mean / mode imputation, k-nearest neighbours or regression models could be used for this, if required.

7 **Normalization and standardization:** Many machine learning algorithms will benefit from completing preprocessing of data by performing the statistical operations of normalization and standardization to scale data to a standard range or distribution.

- ☐ Normalization can be used to rescale input data to a range of [0,1] or [–1,1], which is useful when various features (input variables) have different scales.
- ☐ Standardization can be used to transform the input data to have a mean score of 0 and standard deviation of 1 (Gaussian distribution). (Note that it is not mathematically possible for the range to be [–1,1] and to have a standard deviation of 1; you need to determine which is required for your model.)

Common mistake

Ignoring the important role that normalization and standardization play

Recognize that normalization (scaling data to a range) and standardization (scaling data to have zero mean and unit variance) are crucial for many algorithms to perform optimally. Apply these transformations consistently across all data used in the model.

Python

```
import numpy as np
data = np.array([10, 20, 30, 40, 50])
# Normalize the data to have a mean of 0, and have range [-1, 1]
data_mean_centered = data - np.mean(data)
max_abs_val = np.max(np.abs(data_mean_centered))
normalized_data = data_mean_centered/max_abs_val
print(normalized_data)
# Standardize the data to have a mean of 0, and std dev of 1
standardized_data = (data - np.mean(data))/np.std(data)
print(standardized_data)
```

A4.2.2 Feature selection

TOK

How does the way that we organize or classify knowledge affect what we know?

The structuring of data sets and the choice of features directly influence the insights gained from machine learning algorithms.

The way data is structured can significantly determine what the machine learning model can learn. For instance, missing values; the inclusion or exclusion of certain data points; or the way categories are defined and labelled can all skew or bias the model's outputs. This structuring determines how the machine "views" and "understands" the world, directly influencing the patterns it recognizes and the predictions it makes.

The features chosen can amplify or suppress certain patterns within the data. For example, in a model predicting creditworthiness, choosing features like income might reflect economic factors, whereas including features like zip code could inadvertently introduce socio-economic biases related to geographical areas.

The decisions made in data structuring and feature selection are not value-neutral. They reflect the biases, perspectives and priorities of those who design the data sets and algorithms.

◆ **Feature:** a numeric property that can be used to contribute a data point for a machine learning algorithm to train on. Think of it as a variable in your data set.

Feature selection refers to taking care to select only the most relevant features for use in your machine learning models. In the context of machine learning, a **feature** is a variable that you wish to use as input values for generating predictions. While it may seem like a lot of additional effort to perform manual feature selection, the process can dramatically impact the overall performance and accuracy of your machine learning model.

Removal of irrelevant detail will result in a more generalized model that is better suited to processing new, previously unseen data.

Three commonly used methods to help determine which features to select are filter methods, wrapper methods and embedded methods.

Common mistake

Don't underestimate the importance of feature selection and engineering. Good features are often more important than the choice of model itself.

See also

For more detail on these approaches, along with example code, search online for scikit-learn's section 1.13 "Feature selection" documentation (https://scikit-learn.org/stable/modules/feature_selection.html).

Filter methods

So-called as they help "filter out" features, filter methods involve applying a statistical metric to determine which features are best to be retained and which should be removed from the model. Features are ranked by their score, and those that don't meet the threshold can be filtered out.

As a purely statistical measure, using filter methods is less computationally expensive than retaining the feature in the model for full training. The downside is this does not detect interaction between features. That is, if one feature is affecting another, then a filter may suggest deleting a feature that is actually important. This is where manual appreciation of the context of your model is always important.

The most common, and easy-to-use, filter is to calculate the r value of the correlation (Pearson's product moment correlation coefficient). The r value of a data set may be calculated using

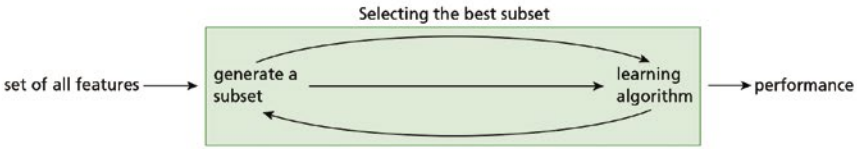
$$r = \frac{\Sigma(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\Sigma(x_i - \bar{x})^2(y_i - \bar{y})^2}}$$

where x_i and y_i are your individual data points and $\bar{x}$ and $\bar{y}$ are the mean of each data series.

Once calculated, records with r values beyond a given threshold can be flagged for deletion.

Wrapper methods

Wrapper methods involve iterating over different combinations of the input features and comparing which subset produces optimal performance.



Wrapper methods

This can be a time-consuming and computationally expensive process, especially when compared to filter methods. There is also an increased risk of overfitting the model. The benefit, however, can be a very quick and efficient final model at the end of the process.

For further study on suitable techniques, do some research into recursive feature elimination (RFE), and sequential feature selection (forward selection, backward elimination). The scikit-learn library (online) provides functionality for both.

ACTIVITY

Research skills: Select and analyse an existing open-source data set relevant to a specific machine learning problem. Learn about the data cleaning and feature selection process used by these "professional" projects, and make recommendations for students learning to use data-cleaning methods for the first time.

Embedded methods

Embedded methods draw on both filter and wrapper methods, but incorporate them directly into the model training algorithm. This means that the feature selection is performed simultaneously with the model training, rather than as a separate step before training.

Embedded methods can be more computationally efficient since they don't require separate iteration of the data prior to training. An embedded method will automatically assess the relevance or importance of features and adjust their weights or inclusion in the model accordingly during the training process.

While embedded methods can save manual labour by eliminating the need for feature selection processes prior to training, they typically require more computational time compared to simpler filter methods. The effectiveness of embedded methods depends on the model's ability to accurately assess feature relevance during the training process.

A4.2.3 Dimensionality reduction

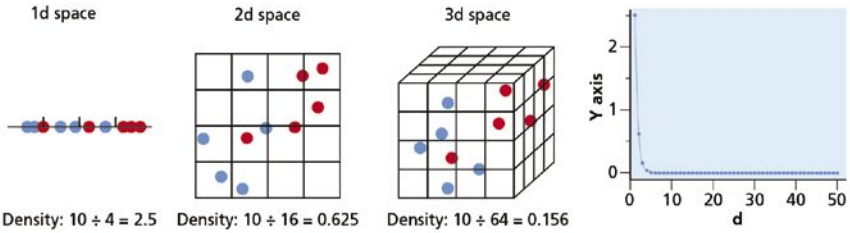
When getting started with machine learning, it is easy to make the mistake of giving too much data to your model. While more quality entries in your data set is usually good, supplying too many features for each entry can easily cause more harm than good.

A typical way of thinking about this as a beginner would be "The more attributes or features I supply, the more detail about my data the model will learn, and perhaps it'll discover a pattern that I hadn't thought of". The problem is that machine learning algorithms are at their best when they are able to make generalizations about the training data. If there is too much detail in each item, and not enough items overall to compensate for that extra detail, then challenges arise.

◆ **Curse of dimensionality:** each feature in a machine learning model adds another dimension to the overall model the algorithm is attempting to map and create generalizations about; the curse of dimensionality refers to the problem that occurs when there are too many dimensions relative to the quantity of data available, so that patterns cannot be meaningfully observed.

◆ **Data sparsity:** how "spread out" data points are from each other in a model.

These challenges are known as the **curse of dimensionality**, and describe the problems that arise in highly dimensional data. The following visualization is a useful way to help understand the problem.



■ Data increases in sparsity as more dimensions are added

In the first panel, there are 10 data points in one dimension, which represents one feature or variable that the model is training with. With 10 points spread across a range of [0,4], there are 2.5 data points per unit. Visually, you can see it is quite crowded, meaning there is a lot of data available to make conclusions and generalizations from.

In the second panel, the same 10 data points are now spread across two dimensions. While both dimensions still have the range [0,4], the effect of the extra dimension is that it squares the space available, so those 10 data points now spread out such that there are only 0.625 data points per unit.

In the third panel, the third dimension is added. With three dimensions, representing three features or variables, there is now only one data point per 0.156 units of space.

The additional detail that comes from adding the extra dimensions acts to spread the data out, making it a lot more difficult for the model to find the generalizations it needs to be useful. To keep the ratio of data points to space consistent, the third panel needs 160 items in its training data instead of just 10. If you don't compensate for additional dimensions with additional quantity of data, the quality of your model will deteriorate.

The empty cells in the diagram above are an example of **data sparsity**, which is where the data points are too far from each other, and the data set contains a high number of empty values. If asked to generate a prediction when given those values that are empty in the training set, the model will have no basis on which to make an accurate estimation.

Sparsity is problematic as it makes it difficult for models to find patterns without overfitting, which is where the model effectively memorizes the individual items in the data set, including the noisy little details.