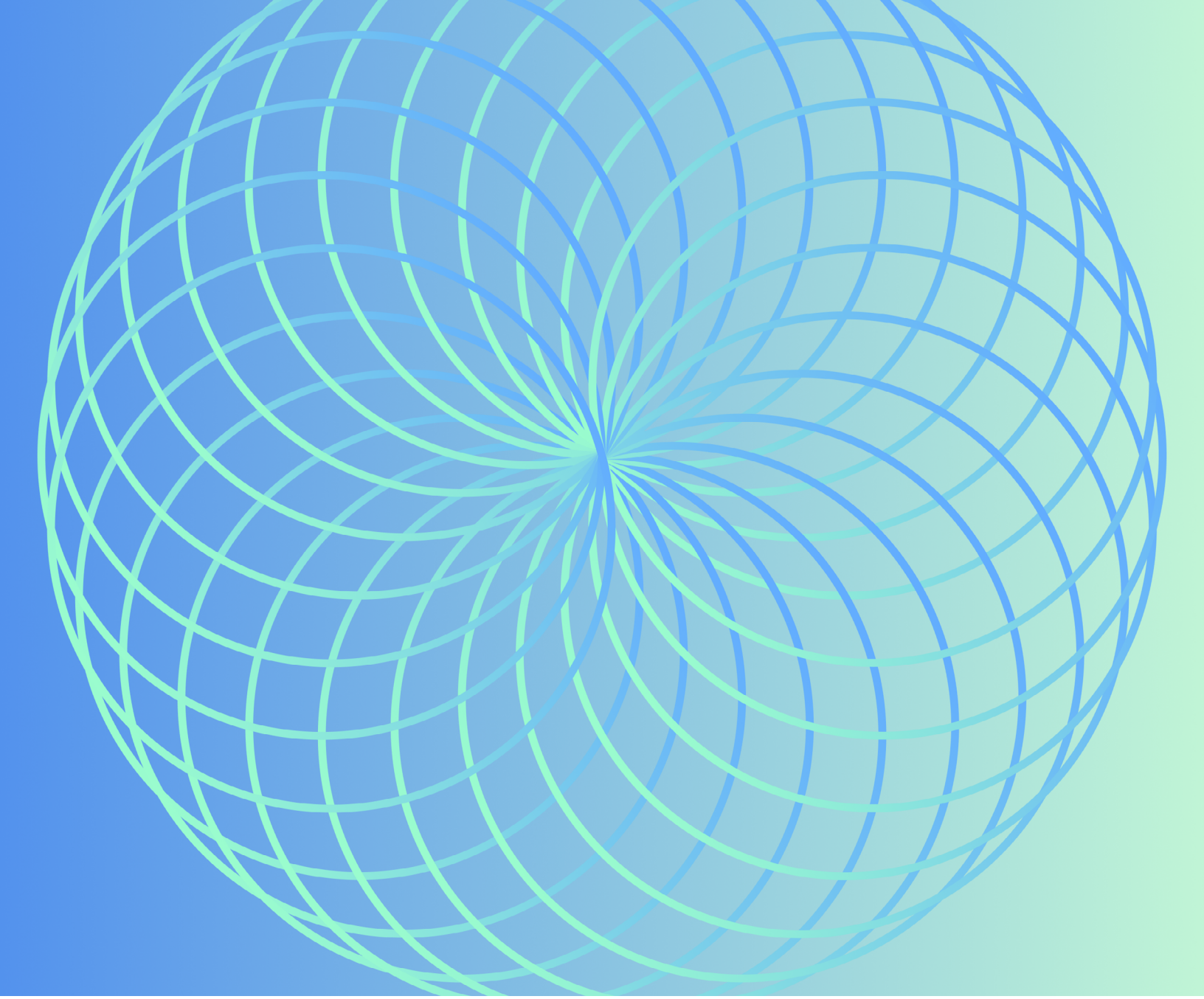


THE COMPLETE BEGINNER'S GUIDE TO

Machine Learning

A comprehensive guide to the foundations and applications of machine learning.

By Akkio

Table of Content

What is Machine Learning?

Machine learning is a branch of computer science that allows computers to automatically infer patterns from data without being explicitly told what these patterns are. These inferences are often based on using algorithms to automatically examine the statistical properties of the data and creating mathematical models to represent the relationship between different quantities.

Let's contrast this with traditional computing, which relies on *deterministic* systems, wherein we explicitly tell the computer a set of rules to perform a specific task. This method of programming computers is referred to as being rules-based. Where machine learning differs from and supersedes, rules-based programming is that it's capable of inferring these rules on its own.

Say you're a bank manager, and you'd like to figure out whether a loan applicant is likely to default on their loan. In a rules-based approach, the bank manager (or other experts) would explicitly tell the computer that if the applicant's credit score is less than a threshold, reject the application.

A machine learning algorithm, however, would simply take in historical data on the credit scores of customers and their loan outcomes and figure out, *on its own*, what this threshold should be. In doing so, the machine is **learning from historical data and creating its own rules.**

This is just an introduction to machine learning, of course, as real-world machine learning models are generally far more complex than a simple threshold. Still, it's a great example of just how powerful machine learning can be.

Any organizational KPI can be optimized as long as you have the relevant data. Given a historical customer dataset, for example, you could predict which of your current customers are in danger of leaving, so you can stop churn before it happens.

Modern approaches to machine learning have made great strides and can accomplish a lot more than just that. From self-driving cars to voice recognition to the automated email filtering systems that flag the spam in your inbox, machine learning algorithms form the basis of many of the advances in technology that we've come to depend on today.

Next, let's consider the different types of machine learning algorithms and the specific types of problems they can solve.

Types of Machine Learning

Machine learning algorithms are often divided into three general categories (though other classification schemes are also used): supervised learning, unsupervised learning, and reinforcement learning.

Supervised Learning

Supervised machine learning refers to classes of algorithms where the machine learning model is given a set of data with explicit labels for the quantity we're interested in (this quantity is often referred to as the response or target).

Semi-supervised learning uses a combination of labeled and unlabeled data to train AI models.

If you're dealing with unlabeled data, you'll need to do data labeling. Labeling is the process of annotating examples to help the training of a machine learning model. Labeling is typically performed by humans, which can be expensive and time-consuming. However, there are ways to automate the labeling process.

A great example of supervised learning is the loan applications scenario we considered earlier. Here, we had historical data about past loan applicants' credit scores (and potentially income levels, age, etc.) alongside explicit labels which told us if the person in question defaulted on their loan or not.

Supervised learning algorithms can be further subdivided into **regression** and **classification.** This difference refers to the type of quantity our target is.

If the target is a choice between a few discrete categories — for example, will the applicant default or not, is this a picture of a cat, a dog, or a human, etc. — then the problem is referred to as **classification**, as we're trying to determine the class that a given data point belongs to.

If, however, our target variable is continuous, then the problem is referred to as **regression**. For example, predicting the price of a house given the number of bedrooms and its location.

Unsupervised Learning

In unsupervised learning problems, the data we're given has no labels, and we're simply looking for patterns. For example, say you're Amazon. Given customers' purchase history, can we identify any clusters (groups of similar customers)?

In this scenario, even though we don't have explicit, definitive data on what a person's interests are, just identifying that a particular group of customers purchase similar items can allow us to make purchase recommendations based on what other people in the cluster have also purchased. Similar systems are what power Amazon's "you might also be interested in" carousel.

K-means clustering is a type of clustering model that takes the different groups of customers and assigns them to various clusters, or groups, based on similarities in their behavior patterns. On a technical level, it works by finding the centroid for each cluster, which is then used as the initial mean for the cluster. New customers are then assigned to clusters based on their similarity to other members of that cluster.

Additionally, once we've identified the clusters, we could then study their characteristics. For example, suppose we see that a given cluster is buying many video games. In that case, we can make an educated guess that this group of customers are gamers, even though no one actually told us so.

Once we've done this form of analysis, we could potentially even use the labels from unsupervised learning to create supervised learning models that may, for example, allow us to predict how much money a 25-year-old gamer is likely to spend with us compared to a 50-year-old fishing enthusiast.

Reinforcement Learning

Reinforcement learning is a class of machine learning algorithms where we assign a computer agent to perform some task without giving it much guidance on precisely what to do.

Instead, the computer is allowed to make its own choices and, depending on whether those choices lead to the outcome we want or not, we assign penalties and rewards. We repeat this process multiple times, allowing the computer to learn the optimal way of doing something by trial and error and repeated iterations.

Think of this as the carrot-and-stick approach to machine learning. It's almost like the computer is playing a video game and discovering what works and what doesn't.

Interestingly, playing games is precisely the application where reinforcement learning has shown the most astonishing results. Google's infamous AlphaGo model, which trounced even the highest-ranked human players of Go, was built using reinforcement learning.

Google has since extended the same technology to AlphaZero, a successor to the original AlphaGo used as a reference by chess players to determine the best strategies.

Deep Learning

If you've seen machine learning in the news, you almost certainly have also heard about deep learning. And you might be wondering at this point where deep learning fits into the above paradigm.

And the answer is all of them.

Deep learning is a subset of machine learning that breaks a problem down into several 'layers' of 'neurons.' These neurons are very loosely modeled on how neurons in the human brain work.

This class of machine learning is referred to as deep learning because the typical artificial neural network (the collection of all the layers of neurons) often contains many layers.

While deep learning was initially used for supervised learning problems, recent advances have extended its capabilities to unsupervised and reinforcement learning problems.

And they have shown tremendous results. Many of the latest advances in computer vision, which self-driving cars and facial recognition systems depend on, are rooted in the use of deep learning models. Natural language processing, which allows computers to understand natural human conversations and powers Siri and Google Assistant, also owes its success to deep learning.

Today's AI boom is largely thanks to the pioneers of deep learning: Geoffrey Hinton, Yann LeCun, and Yoshua Bengio. These AI engineers were awarded the Turing Award for their breakthrough advancements in deep neural networks.

What is the Difference Between Artificial Intelligence and Machine Learning?

If you've ever looked at a tech company's website or watched the keynote for Apple's latest iPhones, you might have seen terms like artificial intelligence (AI) and machine learning (ML) popping up everywhere.

These "buzzwords" are often treated interchangeably, though there are subtle and important differences between them. So, let's look at what both of these terms imply, exactly, and how they all relate to each other.

To start off, let's first define each of these terms and then circle back to the question of how they're related.

What is Artificial Intelligence?

While it's possible to write a book on AI covering computer science, history, philosophy, and the very nature of intelligence, let's keep things simple. The easiest and most accessible way of defining artificial intelligence is simply looking at the words: it's an attempt to create intelligence.

The discipline of AI studies the theory and practice of intelligent systems, especially automated decision making and learning.

In less abstract terms, it's an attempt at allowing computers to mimic both humans' perception of the world as well as our ability to reason with it.

That is a tall order, of course, but it sums up the ultimate goal of AI research rather well. Consider the Terminator. This was an imaginary machine perfectly capable of navigating our world, incorporating new information about its surroundings and the very dynamic nature of both the world and its inhabitants, and making independent decisions without needing any instructions from a human.

This definition also makes it abundantly clear that we are a long, long way away from achieving true artificial intelligence.

Still, the contributions of today's AI are almost limitless. The benefits of AI are already being felt in many industries, including medicine, agriculture, manufacturing, or simply sales and marketing. AI is changing the way we work, play, and engage with one another, from the tools we use to the ways we communicate to the organizations we form.

The Three Types of Artificial Intelligence

But while a genuinely independent machine perfectly capable of handling itself in all situations is the holy grail of research in this field, we have already made significant progress in allowing computers to exhibit human-like capability when performing at least very specific tasks.

To distinguish between these different levels of intelligence, researchers in the field often divided artificial intelligence into two or three types:

- Artificial Narrow Intelligence (ANI)
- Artificial General Intelligence (AGI)
- Artificial Super Intelligence (ASI)

ANI is often referred to as **weak AI**, as it is designed to exhibit "intelligence" or human-like ability in performing a specific task. One of the next frontiers in ANI is maximizing the efficiency of models. This includes optimizing training, inference, and deployment, as well as enhancing the performance of each.

AGI or **strong AI** refers to systems that are capable of matching human intelligence in general (i.e., in more than a few specific tasks), while an artificial super intelligence would be able to surpass human capabilities.

For now, these comparisons are largely relegated to schools of thought, as all deployed AI models are examples of Artificial Narrow Intelligence (not AGI or ASI).

There are a number of factors that are accelerating the emergence of AGI, including the increasing availability of data, the development of better algorithms, and progress in computer processing.

The AI in the movie "Her" is a cultural example of an AGI. Samantha, the artificial intelligence character in the movie, has her own thoughts and opinions. She's not a subservient robot, but rather an independent being. Samantha is capable of using voice and speech recognition, natural language processing, computer vision, and more.

These are good examples of artificial narrow intelligence, as they show a machine performing a single task really well. However, the beauty of general AI is that it's capable of integrating all of these individual elements into a single, holistic system that can do everything a human can.

And while we haven't achieved the latter, we have achieved remarkable progress with the former. Consider self-driving cars, for example. They're an example of ANI, as they excel at a specific task (navigation), and are generally quite capable of identifying elements in their environment (other cars, pedestrians, etc.) and incorporating that information into a decision (e.g., how to make a turn or when to use the brakes to avoid a collision).

Virtual assistants like Siri and Google Assistant are examples of the great strides we've made in creating robust ANI systems that are capable of creating actual value for businesses and individuals.

These assistants use speech recognition, an AI-enabled technology that allows an individual to input voice commands and receive a response. This is achieved through a machine learning model which learns and understands the structure of language by processing sound waves.

In any AI system, data is collected and processed in order to make predictions. This data is then cleaned and converted into a format that can be used by the model. The model will then generate a prediction, which can be viewed as a response to some input. The input may be a question or task, and the response can be considered an answer or a solution.

Other examples include facial and image recognition systems, speech-to-text, machine translation (Google Translate), and recommendation engines (how Amazon or Netflix know which products you'd like).

And this is also where machine learning comes in, as the majority of these advances have been made possible thanks to machine learning (and deep learning).

Whether or not AGI emerges, AI of the future will be embedded everywhere and will touch every part of society, from smart devices to loan applications to phone apps. With the rapid growth of AI, practically all industries are exploring how they can take advantage of this new technology.

What is Machine Learning?

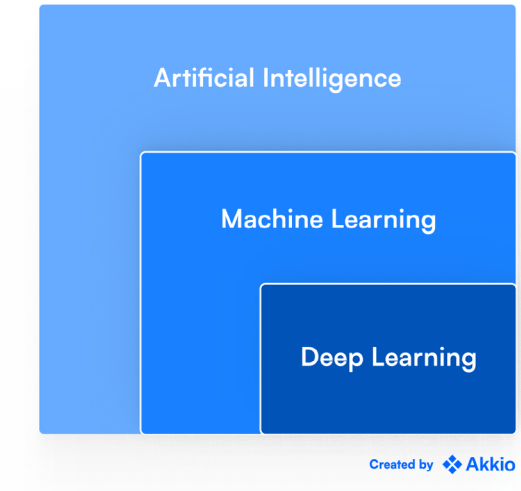
As we discussed in this guide's introduction, "machine learning is a branch of computer science that allows computers to automatically infer patterns from data without being explicitly told what these patterns are. "

As such, machine learning is one way for us to achieve artificial intelligence — i.e., systems capable of making independent, human-like decisions. Unfortunately, these systems have, thus far, been restricted to only specific tasks and are therefore examples of narrow AI.

In the last two decades, many of the most exciting machine learning applications have come from a subset of the field referred to as **Deep Learning**. As discussed in the deep learning section of this guide, deep learning algorithms have achieved state-of-the-art performance in image recognition and natural language processing problems. They have also shown incredible promise in forecasting and reinforcement learning problems. Let's circle back and look at how AI, ML, and DL relate to each other.

How are AI and ML Related?

The following graphic explains the relationship between artificial intelligence, machine learning, and deep learning rather well.



AI is the most general of the three and could almost be thought of as the overarching goal of this field of research: creating systems capable of mimicking human decision making.

A common misconception is that AI is learning. In reality, AI is programmed by humans to complete tasks and offer predictions. AI can mimic intelligence, but it cannot independently learn like a person. The goal of AI engineers today is to make machines think more like humans and less like machines.

Another goal of AI researchers today is to make AI *behave more like* humans. This is particularly challenging, as behavior is thought of as the joint product of predisposition and environment, which are entirely different concepts between people and machines.

Machine learning is one way of achieving artificial intelligence, while deep learning is a subset of machine learning algorithms which have shown the most promise in dealing with problems involving unstructured data, such as image recognition and natural language.

Machine learning is commonly used as a part of hybrid systems. Hybrid systems are a mix of human and machine intelligence that seeks to combine the best of both worlds, such as machine learning models that send predictions to humans to be analyzed.

It is important to distinguish between machine learning and AI, however, because machine learning is not the only means for us to create artificially intelligent systems — just the most *successful thus far.*

Symbolic AI

In the early years of research into this field, for example, researchers focused on building Symbolic AI systems — also referred to as classical *AI or good old-fashioned AI (GOFAI)*.

This approach to creating intelligent systems focuses on representing the world as a collection of symbols, translating real-world problems into symbolic propositions, and then allowing the computer to use propositional logic to solve these problems .

These efforts were based on the observation that humans (and our languages) use symbols to represent both objects in the real world and how they relate to each other. “John” and “pizza” are symbols, while “eat” is the relationship between these two objects/symbols.

Suppose we could represent the entire universe (or at least, all of the information pertaining to a specific domain, such as medicine) into such symbols and relations. In that case, a computer could then solve these problems using logic.

We could also link different propositions together using if-then rules. For example, IF HUNGRY(John) THEN EAT(John, Pizza). This is an example of an extremely simple rule-based symbolic AI.

Of course, while this simplistic example only uses a few symbols and a single rule, a real computer system can store billions of such symbols, propositions, and rules. Such rule-based systems formed the basis for what are known as **expert systems**, AI tools that rely on a hierarchy of rules to provide solutions to problems.

For example, consider a doctor diagnosing a patient. These diagnoses are often also rule-based: i.e., if the patient has X and Y symptoms, if their blood sugar is greater than Z, then they have disease A. As a result, AI has had a big impact on cancer diagnosis, treatment, and prevention. Researchers have shown that algorithms are better than humans at classifying cells as cancerous or not.

Or consider the loan application problem that we discussed in the machine learning article. A panel of experts could easily represent this problem into a series of symbols and rules (e.g., IF credit score > X AND loan amount < Y THEN approve the loan). This could then be used to create an AI expert system that could potentially replace a doctor or a loan officer for making these decisions.

Symbolic AI enjoys several advantages over machine learning. While machine learning systems practice pattern recognition on historical data, symbolic systems only require an expert to define the problem space in terms of symbols, propositions, and rules. Thus, it requires next to no training data.

Additionally, since symbolic AI systems comprise a hierarchy of human-readable rules, they’re much easier to interpret than, say, deep neural networks, which are famously opaque and difficult to interpret.

Lastly, an ideal symbolic AI, with all the knowledge of the world that a human possesses, could potentially be an example of an artificial general (or super) intelligence capable of genuinely reasoning like a human.

With that said, while it theoretically makes sense to argue that we could potentially express all knowledge as symbols, the reality is that our understanding of the world is incredibly complex and explicitly stating all of human knowledge and common sense as a series of symbols and relations would be a Herculean task.

Some pieces of information may also be difficult to represent as symbols. For example, consider image classification. How does one describe a '2' in image form as a symbol? While neural networks excel at these tasks, simply translating the problem into a symbolic system is difficult.

This was one of the major limitations of symbolic AI research in the 70s and 80s. These systems were often considered brittle (i.e., unable to handle problems that were out of the norm), lacking common sense, and therefore "toy" solutions.

These limitations were among the primary drivers of the first "AI winter", a period of time when most funding into AI systems was withdrawn, as research failed to satisfactorily address these problems.

As a result, aside from some niche applications, symbolic AI has generally fallen out of fashion in favor of machine learning, which focused on specific tasks (i.e., narrow AI) but provided far more robust solutions.

Advances in computing power and the proliferation of data in the internet era have also been a significant boosting factor in enabling machine learning systems, whose performance is often limited by the amount (and quality) of the data available to them.

In recent years, however, researchers have started looking at combining machine learning systems, especially neural networks, with symbolic AI in an attempt to capitalize on the strengths of both these approaches to AI. This is referred to as **neural-symbolic computing.**

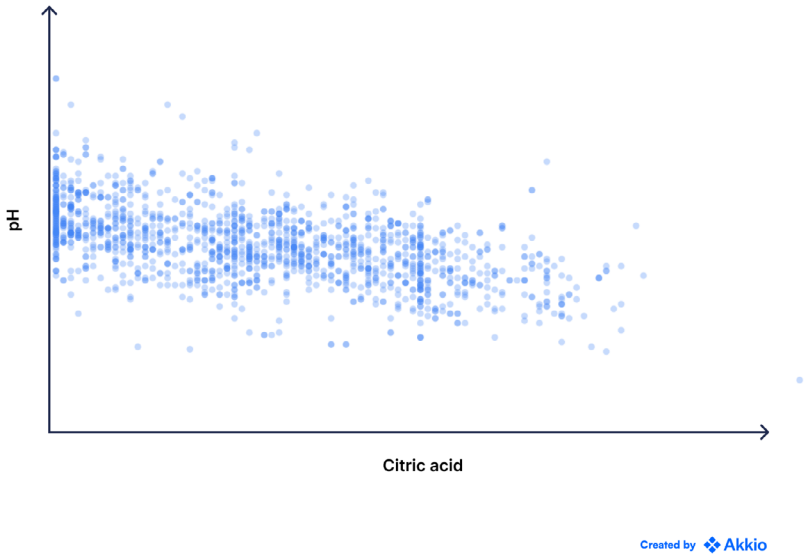
ML Applications: Regression

Many business applications require **predicting a continuous quantity**. For example, “what is the lifetime value of a customer with a given age and income level?”, or, “what is the probability of customer churn?” These are generally referred to as **regression problems.**

In this article, we will go over several machine learning algorithms used for solving regression problems. While we won’t cover the math in depth, we will at least briefly touch on the general mathematical form of these models to provide you with a better understanding of the intuition behind these models.

Linear Regression

The most common method for solving regression problems is referred to as **linear regression.** Say you’re given the following data about the relationship between pH and Citric acid to determine wine quality.



You can clearly see a linear relationship between the two, but as with all real data, there is also some noise. Since the relationship is linear, it makes sense to model this using a straight line.

> You might recall from high school math that the equation of a straight line is given as:

$$y = c + mx$$

where

- y is the response,
- x is the predictor,
- m is the slope of the line (or the coefficient/weight of x), and
- c is the y-intercept

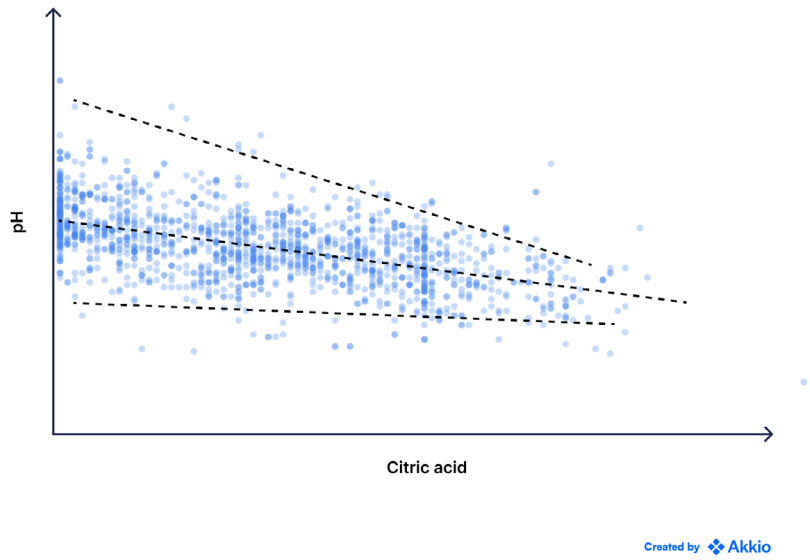
We can extend this to multiple predictors as follows, which is also the general form of linear regression:

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p$$

where

- $\hat{y}$ is our model's prediction
- β_0 is the intercept
- β_i is the coefficient of the x_i-th predictor, and
- p is the total number of predictors

But we could potentially draw many straight lines and because of the noise, it's not entirely clear which one is the 'best line'. For example, of the three lines below, which is the better one?



This requires us to define a criterion for what is, mathematically, considered 'good' vs. 'bad.'

Since we're using this model to predict quantities, it makes sense to use the **error** of our predictions as our yardstick, where the error is defined as the difference between the actual value and our prediction. A line that minimizes the overall prediction error is 'good,' while a line that has a large overall error is 'bad.'

There are different ways of calculating errors. For our purposes, we'll use the **sum of squared errors (SSE)**. As a simple example of this, consider the following example of predicting the weather:

Actual	Prediction	Error	Squared Error
30	33	-3	9
25	21	4	16
40	41	-1	1
	Totals	**0**	**26**

At this point, you might be wondering why we're taking the square of the errors, and not just the actual value. That's because we don't want negative and positive errors to cancel each other out.

If we just summed the error values in the above example, we'd get 4 - 3 - 1 = 0. This would suggest the model is perfect and give us a false sense of confidence in our model. Using squared errors prevents this from happening.

> The mathematical formula below describes the aforementioned sum of squared errors. When using the sum of squared errors, it's important to understand that this isn't a perfect indicator of how well a model fits the data, but it's simple to understand, and thus widely used, as it relies on just three values: The number of data points, the actual values, and the predictions.

$$SSE = \sum_{i=1}^{n} (y_i - y_i\hat{})^2 = \sum_{i=1}^{n} [y_i - (\beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p)]^2$$

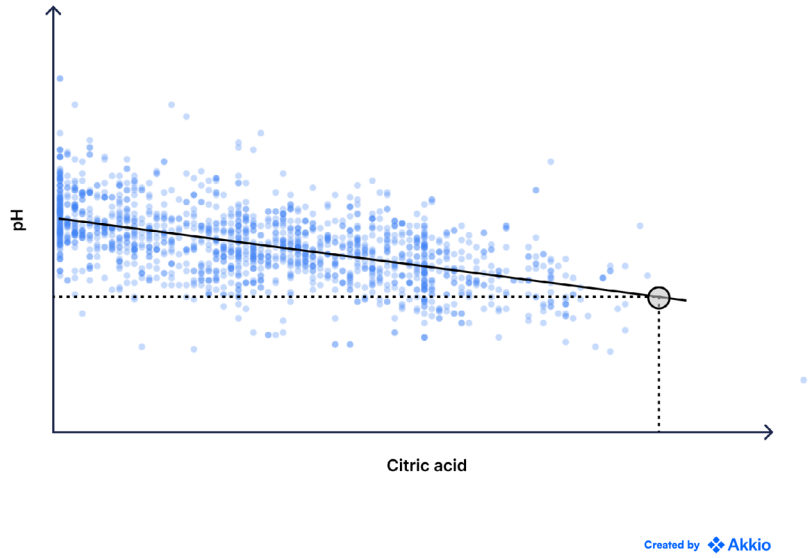
where

- n the number of data points we have
- y_i is the actual value of the response for data point i
- y_i is our prediction for data point i

Thus, of all the possible lines we can draw, we would select the line with the lowest SSE. This is referred to as an **objective function** — i.e., some value we want to either minimize or maximize. In this case, we want to minimize the SSE.

While we won't go into the mathematical details here, this problem can easily be solved using **optimization theory**, thereby allowing us to find the 'best' line which minimizes the sum of squared errors.

Once we have found the best-fit line, we can make predictions for any new input point by interpolating its value from the straight line. For example, while none of our data points have a citric acid of 0.8, we can predict that when citric acid value is 0.8, the pH is ~3.

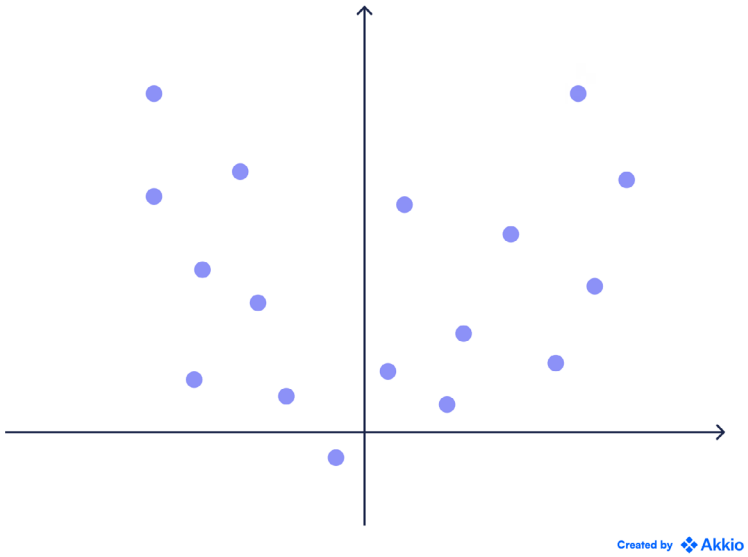


While the above example was extremely simple with only one response and one predictor, we can easily extend the same logic to more complex problems involving higher dimensions (i.e., more predictors).

Nonlinear Regression Methods

Real-world regression problems are often **nonlinear**. There are many ways to deal with such problems, either by extending the linear regression model itself or using other modeling constructs.

For example, say the data we have looks like this:



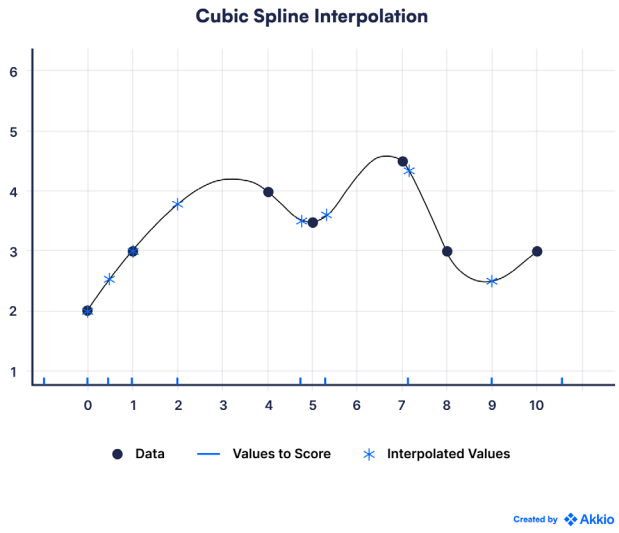
While there's some noise, you can see that this resembles a quadratic curve. Let's say we know that the true relationship is given by the equation:

$$y = 2x + 3x^2$$

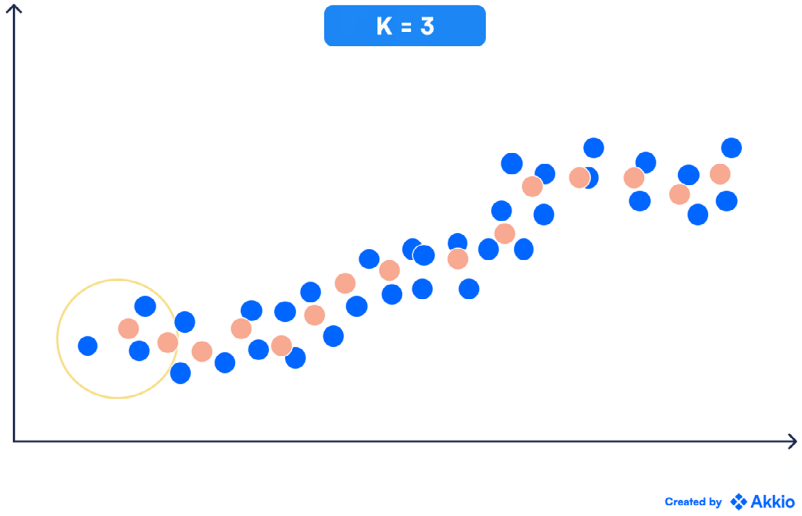
We could easily extend the linear regression model to this problem by simply taking the square of the dependent variable and adding it as another predictor for the linear regression model. We could do the same for higher-order terms, and this is referred to as **polynomial regression**.

Other, more complex methods include the use of **splines**. While we won't go into the theory or the math behind this in any detail, at a fundamental level, splines allow us to fit different nonlinear functions to different parts of the input space, while ensuring that the functions are smooth (i.e., connected) at the boundaries between these regions.

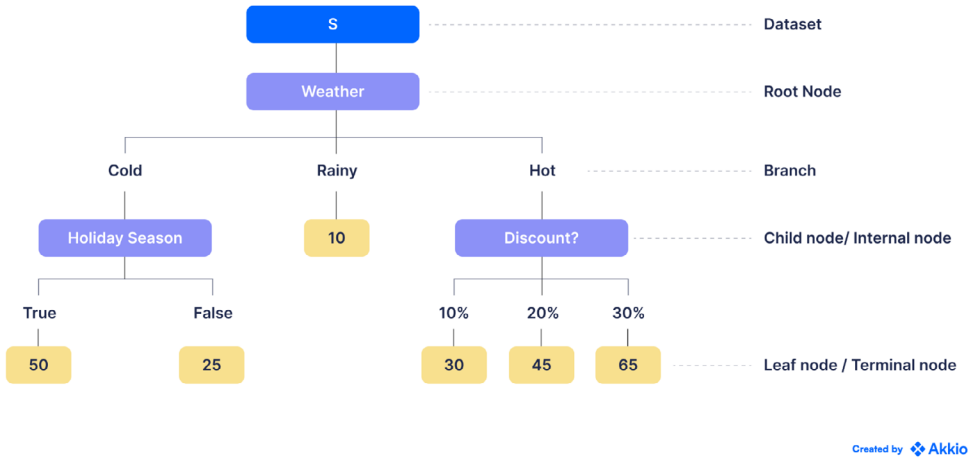
The result is a highly flexible model that can fit nonlinear data more closely. However, this may come at the expense of **overfitting** as the model may be fitting to random noise instead of the actual patterns. As a result, splines and polynomial regression should be used with care and evaluated using **cross-validation** to ensure that the model we train can be generalized.



We may also use **nonparametric methods** for regression problems. The simplest of these may just be **K-nearest neighbor regression.** In this method, given historical data and a new data point we want a prediction for, we simply find the k data points closest to this new point and predict its value to be the mean of these k points.



We can also use **decision trees** for regression problems. Here, we split the data into different subsets based on a set of criteria. We may then assign a fixed value to each leaf node as its prediction (e.g., the mean of all the data points belonging to that leaf node). See the example below of using decision tree regression for predicting the number of hours played based on different weather conditions:



Alternatively, we could also fit a separate linear regression model for each of the leaf nodes.

As with many other machine learning problems, we can also use **deep learning** and neural networks to solve nonlinear regression problems.

Predicting Probabilities with Logistic Regression

Let's extend the idea of predicting a continuous variable to probabilities. Say we wanted to predict the probability of a customer canceling their subscription to our service.

Since probability is a continuous variable, it naturally extends itself to regression. However, it's a continuous variable capped by two constraints: a probability can neither be negative nor greater than 1. Regular linear regression is incapable of abiding by these constraints, and thus the **logistic model** was born.

Logistic regression is an extension of linear regression, which straddles the line between regression and classification. It works on the same principle as linear regression but with a key difference: the response is the **natural log of the odds of an event occurring.**

Odds, in statistics, refers to the ratio of the probability of an event occurring to the probability of it not occurring:

$$odds = \frac{prob.\ of\ occurring}{prob.\ of\ not\ occurring}$$

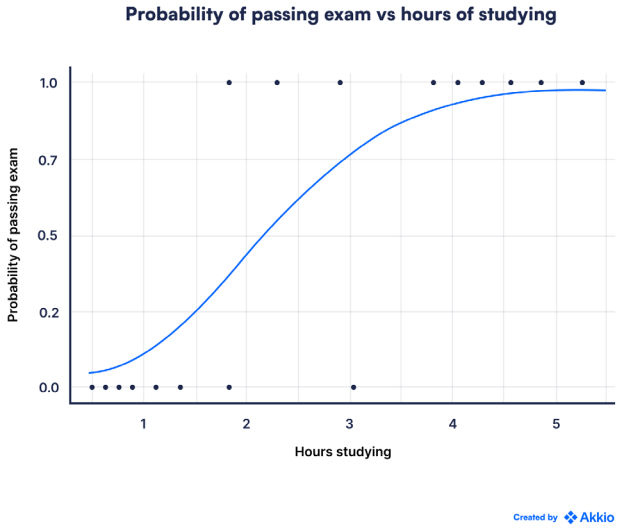
For example, say the probability of Barcelona winning their next match is 30%. Then, the **odds for** their victory are 3/7 or 3:7. This is also the nomenclature used in gambling, though gambling sites often show odds against an event and not odds for. The **odds against** Barcelona, in this case, would be 7:3.

> The log of odds (or log-odds) is often referred to as logit(p), where p is the probability of an event occurring. The logistic model is thus represented by the following equation:

$$logit(p) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots$$

Statistically, the most salient aspect of using the natural log of odds is that while the output of the regression model is still unconstrained, when we convert the log odds back to probabilities, these are capped to between 0 and 1, thus solving our problem!

While we won't go into the mathematical details of why that is the case, you can see below a graph of the output probability p as the value of the independent variable changes:



Thus, we've successfully extended the linear regression model to predict probabilities. Once we have an estimate for the probability of an event occurring, **classification** is just one step away.

If we set a certain probability as a **threshold**, we can classify each data point (e.g., each customer) into one of two classes. Choosing this threshold is largely dependent on the application.

For example, a luxury carmaker that operates on high margins and low volumes may want to be highly proactive and personally check in with customers with even a 20% probability of churn. If churn is not mission-critical or we simply don't have the resources to handle individual customers, we may want to set this threshold much higher (e.g., 90%) so we are alerted to only the most urgent prospects.

ML Applications: Classification

In the previous section, we dealt with examples of **regression problems**, where we want to predict a continuous variable. The second major type of **supervised learning** problem is classification, where we want to assign each sample into one of **two (or more) categories.**

For example, a bank may want to determine if a loan applicant will repay their loan or not. Or an email provider might want to build a system that filters out spam from your inbox.

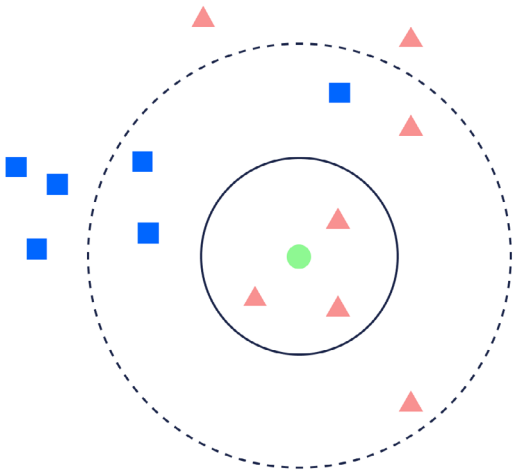
In both these cases, we have only two possible classes/categories, but it's also possible to handle problems with multiple options. For example, a lead-scoring system might want to distinguish between hot, neutral, and cold leads. Computer vision problems are often also **multi-class problems**, as we wish to identify multiple types of objects (cars, people, traffic signs, etc.).

In this article, we'll examine some of the algorithms used for classification problems. However, the focus here will be on building intuition, and so we won't be covering the math behind these algorithms in any detail. We'll also focus on only binary classification problems (i.e., those with only two options) for simplicity.

K-Nearest Neighbours (KNN)

One of the simplest classification algorithms is **KNN classification**. Say we have historical data with labels and a new point whose label we want to determine. In this method, we simply find the k points closest to the new point and assign its label to be the mode (the most commonly occurring class) of these k points.

For example, consider the image below. If k=3, then the label of the green point is a red triangle because, among the three points closest to it, the majority (⅔) are red triangles.

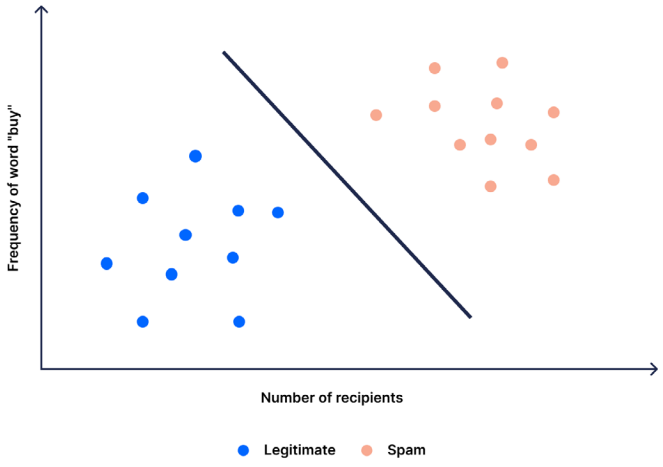


Created by Akkio

As we discussed in the **regression section**, the KNN algorithm can also solve nonlinear regression problems.

Support Vector Machines (SVM)

Another commonly used classification algorithm is SVM. Consider the following example where we want to filter out spam emails. The x-axis is the number of times the word “buy” appears in the email, and the y-axis is the number of other people who have received the same email. When plotted, the data looks like this:



Created by Akkio

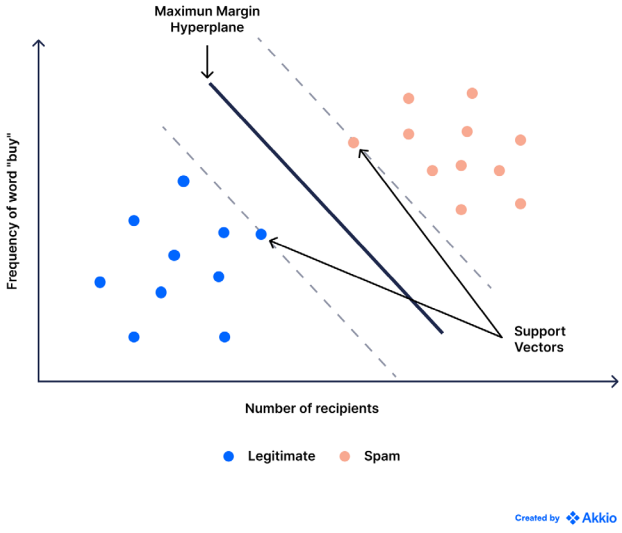
The blue points are legitimate emails, and the red points are spam. Spam emails presumably want you to buy things and are sent to more people, so it makes sense that emails sent out to a large number of people with many mentions of the word “buy” are spam.

More importantly, we see that we can clearly separate the two classes using a straight line, but as with linear regression, this raises the question: which line is the best?

As shown below, we can draw many possible lines, all of which perfectly separate between the two classes.

We may want to, thus, think about defining what makes one line better than another. This is somewhat dependent on the problem we’re trying to solve, and we’ll revisit this point later on.
For now, though, a reasonable criterion may be to choose the line which maximizes the **margin** between the two classes — that is, the line which is as far away as possible from the most extreme examples of either class.

This raises another question: how can we translate this into a mathematical problem instead of just doing it by eye? Consider the following diagram:



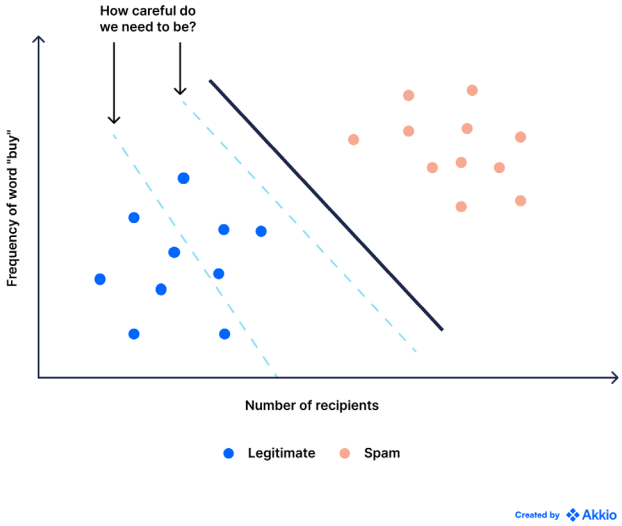
We can find the ‘best’ line by first drawing two lines that only touch the outermost points of each class. Note also that both of these lines are parallel to each other. These lines are called **support vectors**; hence the name of the algorithm.

The ‘best’ line is then a line that is parallel to both of these lines and also equidistant from them (i.e., it’s the same distance from each). The distance between the support vectors and the classifier line is called the **margin**, and we want to maximize this.

This is the most common (or default) way in which SVM selects the best classifier line. This may not always be the ideal way of doing things, however.

For example, say we were working on determining if a tumor is benign or malignant. In this case, the cost of making a mistake is not the same for each class. If we classify a malignant tumor as benign, it could potentially cost the patient their life, while mistaking a benign tumor as malignant might only require further testing. Clearly, one mistake is worse than the other.

Depending on the application and how careful we want to be, we may choose to assign a greater weight to either type of mistake. As such, we may decide to move the line further away from one class or even deliberately mislabel some of the data points simply because we want to be extremely cautious about making a mistake.



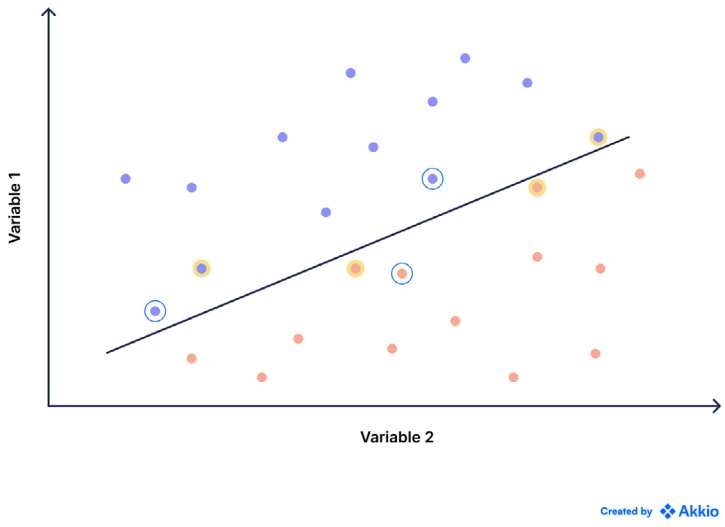
Next, let’s consider scenarios where the two classes cannot be cleanly separated via a straight line.

Soft vs. Hard Classifiers

Sometimes, it may not be possible to perfectly classify points using a straight line. We could, then, resort to nonlinear methods (discussed later), but for now, let's stick to only straight lines.

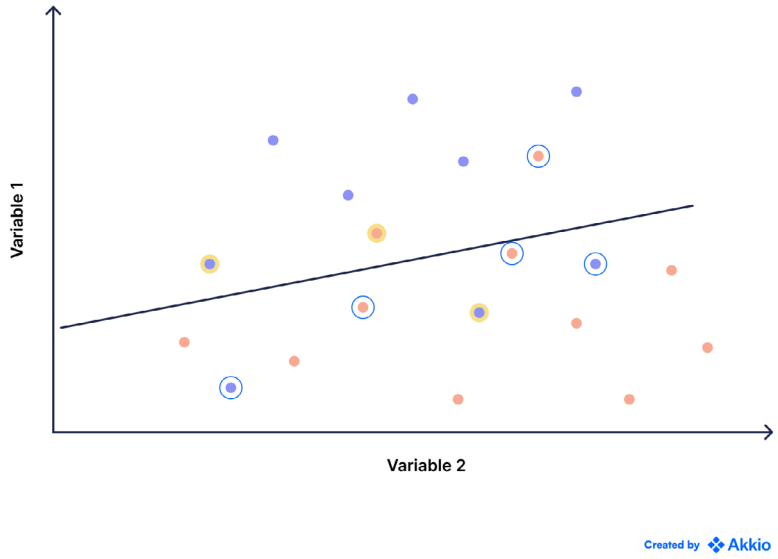
In that case, we may be willing to take an imperfect classifier. This is also called a **soft classifier**, as it does not classify all points correctly. On the other hand, a hard classifier would refer to the examples we've discussed thus far, which perfectly classify all data points.

Consider the following example:



In the above image, we see that the soft classifier we've selected misclassifies three points (highlighted in yellow). At the same time, we also see two blue points and two red points (circled in blue) that are extremely close to the line and are near-mistakes. Thus, our classifier has a very small margin between the two classes.

Consider a second possible classifier we could draw for the same data:



In this case, we have five misclassified points (compared to three before), but the line has a wider margin and very few points that are near-misses or extremely close to the line.

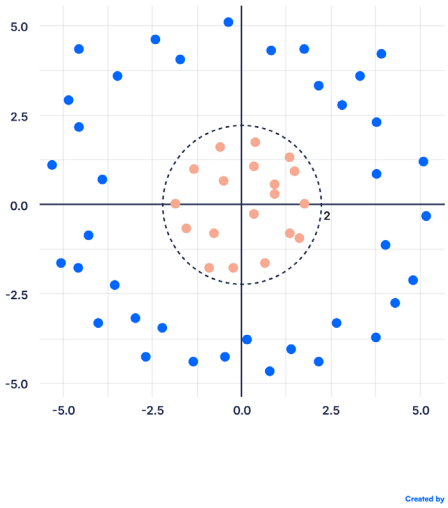
This demonstrates an inherent tradeoff with soft classifiers. We can choose to either minimize errors or maximize the margin between the two classes.

While we won't show the mathematical details here, we can assign different weights to either of these options depending on how important higher accuracy is compared to having a cleaner, less ambiguous boundary.

This is an example of a **model hyperparameter**: a variable that we specify for the algorithm and which defines or in some way constrains the form that our model will take.

Nonlinear SVM Classifiers

Now, consider the following example:

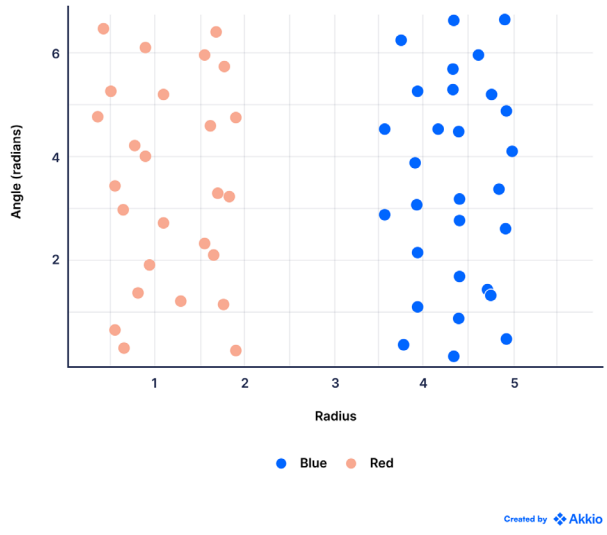


In this case, we see that while a straight line cannot separate these points, a circle can. How can we solve this problem? As we've seen above, one option may be to use nonlinear methods like KNN classification or classification trees.

Since these are **nonparametric methods** and don't specify a particular form for the model (e.g., that it needs to be a straight line), they are particularly well-suited for nonlinear problems.

However, SVM can also be extended to solving this problem by **transforming the data** to achieve linear separation between the classes. For example, we can see that all the points within a circle of radius 2 are red and those outside it are blue.

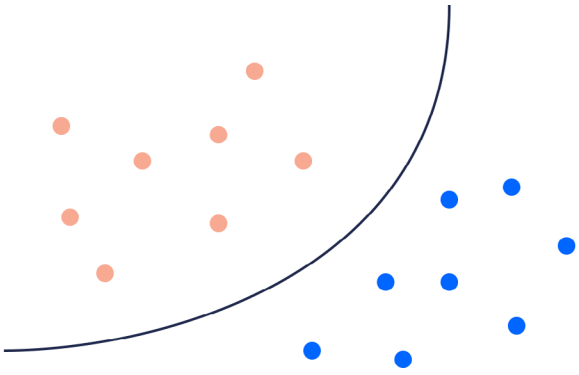
In a simple case like this, if we convert the data from Cartesian to polar coordinates. The resulting plot is shown below, where the x-axis is the radial distance from the origin, and the y-axis is the angle in radians:



As can be seen, the classes are now easily separated using a straight line. Thus, we would simply feed the SVM algorithm this transformed version of the data.

In more complex scenarios, especially when we have multi-dimensional problems and don't know that the ideal classifier is, for example, a circle, we may not know which transformation to use. In other cases, the transformation may be computationally inefficient.

In these cases, we can transform the problem by adding more dimensions to it. This is referred to as the **kernel trick** or **Kernel SVM** and allows us to create nonlinear classification boundaries like the one below:



An explanation of the mechanics or the math of how and why kernel SVM works is beyond the scope of this article. Still, it's an important detail to know in order for you to have a comprehensive understanding of the kinds of problems the SVM algorithm can solve.

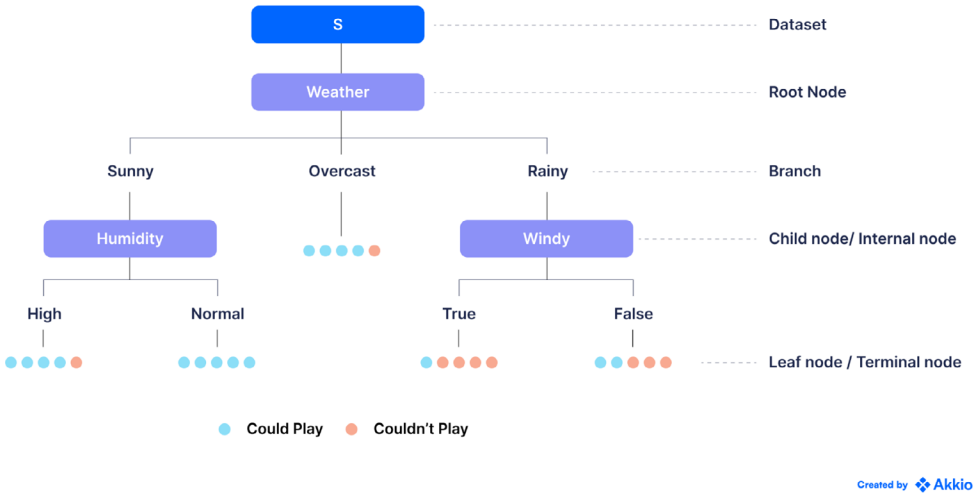
Kernel methods should be used with SVM with caution, however. By adding more dimensions to the problem and allowing for nonlinear boundaries, we are creating a more **flexible model**. This can easily lead to **overfitting**.

Classification Trees

Another means of solving classification problems — and one that's exceptionally well-suited to nonlinear problems — is the use of a **decision tree.**

Since decision trees can be used for both classification and regression problems (see the regression section), the algorithm is sometimes referred to as **CART (Classification and Regression Trees)**.

In this method, we divide the data into smaller and smaller subsets based on a series of binary (yes/no) questions. Consider the following decision tree for deciding if we should play football or not, based on how the weather affected past games:



The balls at the leaf nodes indicate whether we were able to successfully finish a game (blue) with the given weather conditions or whether the game had to be interrupted due to poor weather (red).

We see that on most rainy days with wind, we were forced to cancel our games. As such, the mode of this leaf node is red, and we would classify any future rainy and windy days as red (i.e., we probably shouldn't play on those days).

Note that decision trees are also an excellent example of how machine learning methods differ from more traditional forms of AI. You might recall that in the ***What is the difference between machine learning and AI*** section, we discussed something called **expert systems**, which are a hierarchy of if/else rules that allow a computer to make a decision.

A decision tree is also a hierarchy of binary rules, but the key difference between the two is that the rules in an expert system are defined by a human expert. On the other hand, decision trees figure out what the splitting criteria at stage (i.e., the rules) should be by themselves — which is why we say that the machine is learning.

How does it do that? You might have noticed that each of the leaf nodes consists mostly of one class — for example, the Sunny + Normal Humidity node is mostly blue, while the Rainy + Windy node is mostly red.

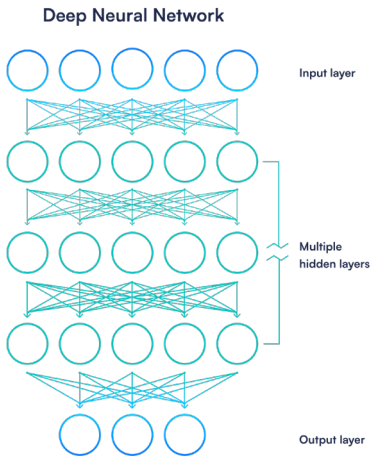
That's by design. At each stage of building the decision tree, the computer will look at all of the possible options it has and choose the splitting criterion that minimizes the impurity of the subsequent nodes — that is, it tries to ensure that each of the nodes has points belonging to only one class, if possible.

Of course, if we allow the computer to keep splitting the data into smaller and smaller subsets (i.e., a deep tree), we might eventually end up with a scenario where each leaf node only contains one (or very few) data points. This might lead to **overfitting**. Therefore the **maximum allowable depth** is one of the most important hyperparameters when using tree-based methods.

Deep Learning

Deep learning is another excellent example of a classification method. In fact, deep learning models are great at solving problems with multiple classes.

They're also particularly effective at dealing with nonlinear relationships and unstructured problems as they're able to tease out the more abstract interactions between different terms.



In the **What is Machine Learning section** of the guide, we considered the example of a bank trying to determine whether a loan applicant is likely to default or not. This is an example of a problem where we have relatively structured data. We know, for each applicant, specific values of different metrics that we think are important and relevant to solving their problem (e.g., their income, credit score, etc.). These metrics are often referred to as features or predictors.

But what about a facial recognition problem? Say we have two pictures of the same person looking in different directions. If we simply fed these two images as a string of pixels to a classical ML algorithm, it might not recognize that they are the same person because the string of pixels that it received might be quite different based on lightning conditions, the direction in which the person is looking and so on.

Instead, it would make far more sense for us to try and extract useful features from the image first and then feed these as the inputs to the algorithm.

For example, we might want to determine what the colour of their skin is, the shape of their face, the length of their nose, the colour of their eyes and so on. Since these will remain the same regardless of lighting conditions or the orientation of their face, this might be a far more robust solution.

This, however, raises another problem as we might need another machine learning algorithm to, for example, distinguish between the person's face and hair. Once we've identified the hair, we may then need a second machine learning algorithm to distinguish between the different types of hair colours (since hair colours aren't discrete and 'red' hair can be many different colours in reality).

As such, we may need to break down the problem into 'layers' of smaller sub-problems (also solved using machine learning) to first extract the relevant, structured features before we can feed them to the final algorithm which actually classifies faces.

Deep learning, on the other hand, tries to circumvent this problem as it doesn't require us to determine these intermediate features. Instead, we can simply feed it the raw, unstructured image and it can figure out, on its own, what these relevant features might be.

In doing so, it presents two significant benefits compared to classical machine learning algorithms:

- we may not always know which features are relevant; for example, is the length of the eyelashes relevant? Instead of having to figure out which features are relevant ourselves, a deep learning model can do the job for us and potentially also identify features we might have never thought of; and
- even if we know what the relevant features are, we no longer need to spend time working on actually extracting these features ourselves as the neural network does this work for us.

This is also why deep learning algorithms are often considered black boxes. The complexity of their structure and the large number of layers in them means we can't precisely extract information about specific features as we might do with a linear regression model, where the coefficient for each feature imparts direct and easily interpretable information about the linear relationship between the features and the response.

Why are neural networks deep?

As we've discussed before, a neural network is 'deep' when it contains multiple layers. While different practitioners might differ on exactly what the threshold for a 'deep' neural network is, a neural network with more than three layers is often considered as being 'deep'.

This begs the question, however, why do neural networks need to be deep?

To answer this question, recall how, in the previous section, we discussed that solving a facial recognition problem may require the creation of a pipeline with multiple layers of sub-problems in order to use classical ML algorithms.

Well, it turns out that that's more or less also how deep learning algorithms work. For example, in an image classification problem, research has shown that each of the layers (or a group of them) will tend to specialize toward extracting specific pieces of information about the image. For example, some layers might focus on the shapes in the image, while others might focus on colors.

Adding more layers can, therefore, allow neural networks to more granularly extract information — that is, identify more types of features.

Deeper layers also allow the neural network to learn about the more abstract interactions between different features. For example, the impact credit score has on a person's ability to repay a loan may be very different based on whether they're a student or a business owner.

In a regression setting, the data scientist would need to manually specify any such interaction terms. But as we discussed before, we may not always know which interaction terms are relevant, while a deep neural network would be able to do the job for us.

Its prowess with unstructured data has allowed deep learning to produce massive advances in the fields of computer vision, object detection, and natural language processing, all of which involve unstructured data and classification (for example, classifying the different objects in an image as a car or a pedestrian).

Machine Learning Training Data Sources

Machine Learning works by recognizing the patterns in past data, and then using them to predict future outcomes. To build a successful predictive model, you need data that is relevant to the outcome of interest. This data can take many forms - from number values (temperature, cost of a commodity, etc) to time values (dates, elapsed times) to text, images, video and audio. Fortunately the explosion in computing and sensor technology combined with the internet has enabled us to capture and store data at exponentially increasing rates. The trick is getting the right data for any particular problem - most businesses capture this in their existing technology stacks, and a lot of this data is available for free online.

Structured vs Unstructured Data

Structured versus unstructured data is a common topic in the field of data science, where a structured dataset typically has a well-defined schema and is organized in a table with rows and columns. Unstructured data, on the other hand, is often messy and difficult to process.

Structured and unstructured data can both be the fuel for successful machine learning models.

Let’s dive into the details of structured versus unstructured data, including data formats, data storage, data sources, analysis, and more.

Structured vs Unstructured Data Formats

Structured data is quantifiable and easy to search and analyze, and comes in predefined formats such as CSV, Excel, XML, or JSON, while unstructured data can be in a variety of less well-defined formats including PDFs, images, audio, or video.

Structured data is typically a result of a well-defined schema, which is often created by human experts. It's easy for people to add or change the schema of structured data, but it can be very difficult to do so with unstructured data.

In short, structured data is searchable and organized in a table, making it easy to find patterns and relationships. It's also possible to analyze and gain value from unstructured data, such as by using text extraction on PDFs, followed by text classification, but it's a much more difficult task.

Structured Data Sources

Many popular business tools, like Hubspot, Salesforce, or Snowflake, are sources of structured data.

Akkio's sample datasets, which are in CSV format, are also examples of structured data. More broadly speaking, any well-defined CSV or Excel file is an example of structured data, millions of examples of which are available on sites like Kaggle or Data.gov.

Unstructured Data Sources

For the purpose of predictive modeling, the most common type of unstructured data is text. This includes text forms, like customer feedback forms, as well as emails, comments on social media sites, product reviews, or even notes taken during sales calls or business meetings.

As we've highlighted, unstructured data goes beyond text, and includes audio and video. For example, YouTube reviews are another source of unstructured data. YouTube videos also include AI-generated transcriptions or speech-to-text. Given that text data, text classification could be used to mine those reviews for insights.

Structured vs Unstructured Data Storage

Structured data is often stored in data warehouses while unstructured data is stored in data lakes. A warehouse stores structured datasets and typically relies on more traditional databases like SQL Server and Oracle for storage, while a data lake stores less well-defined datasets.

Structured Data in Real-World AI

Some of the most well-known machine learning models in use today are fueled by structured data.

For example, Amazon uses its database of customer purchasing patterns and preferences to recommend items that are likely to be of interest to a particular customer.

Unstructured Data in Real-World AI

Other machine learning models are fueled by unstructured data.

Tesla uses its fleet of self-driving cars to collect data about driving patterns and conditions. The data is used for teaching self-driving cars how to avoid collisions and navigate through varying driving conditions.

Another example is seen in Google Photos. When you take a photo, Google's machine learning models scan the image, an unstructured data type, to find what category it falls into. Then, users can search their own, previously unlabeled photos by categories like "Nature" or "People."

Structured Data Analysis

Most analytics tools are designed for structured data, making it easier than ever to analyze and gain value from structured data.

With Akkio, for instance, you can upload structured data to build and deploy AI models in minutes. In the background, machine learning algorithms scan and digest the tabular data to find patterns, creating a model that can be deployed to find those patterns in new data.

Unstructured Data Analysis

Unstructured data analysis is a less-common task, but it's still highly important for businesses looking to gain value from their PDFs, image and audio data, and so on.

Analyzing unstructured data is a complicated task, which is why it's ignored by many businesses.

Unstructured data can be difficult to process and understand because it's messy and in a variety of formats. Unstructured data may also be qualitative instead of quantitative, making it even harder to analyze.

One use-case for unstructured data is to analyze reviews and comments on social media, both from your own company and from competitors, to inform competitive strategy.

Another use-case is market analysis to find new opportunities. By analyzing unstructured market data, such as social media posts that mention customer needs, businesses can uncover opportunities for new products and features that may meet the needs of these potential customers.

Quantitative vs Qualitative/ Categorical Data

Quantitative data is a numerical set of information, such as the height and weight of each person in a group, alongside the size of the group. Quantitative data can be further divided into two sub-categories: Discrete and continuous data.

Discrete data does not include measurements, which are along a spectrum, but instead refers to counting numbers, like the number of products in a customer's shopping cart, or a count of financial transactions. Continuous data, on the other hand, refers to data that can meaningfully be broken down into smaller units, or placed on a scale, like a customer's income, an employee's salary, or the dollar size of a financial transaction.

Qualitative data is non-numeric, such as whether or not a transaction is fraudulent, whether a review has positive or negative sentiment, or whether a sales deal has a high or low likelihood of being closed. Qualitative data is largely categorical, but it also includes things like text, whether it's a tweet, a customer support ticket, or documentation. By the very meaning of the word, categorical data is simply data relating to categories, while quantitative data relates to quantities.

Let's dive more deeply into the differences between quantitative and qualitative data, with the latter focusing on categorical data.

How to know whether your data is quantitative or categorical

It can be difficult to determine if your data is categorical or quantitative, but there are a few steps you can take to find out.

If your data has a numerical range of values, like income, age, transaction size, or similar, it's quantitative. If, on the other hand, there are categories, like "Yes," "Maybe," and "No," it's categorical.

You should also consider the type of answers you're expecting from your data. Are you expecting an answer that has a range of values, or just one set of values? If you're expecting one set of values, like "Fraud" or "Not Fraud," then it's categorical. If you're expecting a range of values, like a certain dollar amount, then it's quantitative.

Examples of AI models you can make with quantitative data

Quantitative data can be used to fuel a wide range of AI models. Let's explore a few examples.

- Forecasting site traffic, given historical traffic data (e.g. if you're going to run Google Ads on a Saturday night, what are your expected traffic numbers?)

- Determining the number of customers that will buy something, given historical transaction frequencies (e.g. if you are running a promotion, how many people will purchase an item?)
- Predicting how much money you will make, given historical revenue (e.g. how many people will click on an ad, and then make a purchase?)
- Determining what your inventory levels should be, given historical sales figures (e.g. what should your inventory levels look like given your sales figures?)

Quantitative machine learning algorithms can use various forms of regression analysis, for instance, to find the relationship between variables.

To give a simple example, if one variable is the weight of a patient and the other variable is the height of a patient, then the relationship between these variables can be found by running regression analysis on a set of patients.

Examples of AI models you can make with categorical data

Categorical data can also fuel a wide range of AI use-cases. Here are just a couple of examples.

- Classifying customers into different categories based on the groups of behavior they fall into (e.g. what type of device do they use to browse your website? Do they buy clothes or shoes?)
- Classifying your ads into different categories based on their effectiveness (e.g. does this ad attract more clicks than another ad?)

Categorical machine learning algorithms including clustering algorithms are used to identify groups within a dataset, where the groups are based on similarity. The technical algorithm names include Naïve Bayes and K-nearest neighbors.

Understanding the intricacies of these complex algorithms used to be a prerequisite to AI modeling, but you can now build and deploy these models in minutes, with no technical expertise needed.

What's better: Quantitative or categorical data?

There are pros and cons to each type of data, and which data type to use depends on the situation.

Quantitative data is inherently more precise than categorical data, as there's greater granularity in quantitative data. For example, a height of "72.5 inches" is a lot more precise than the category "tall." An income of "$12,000" is a lot more precise than the category "poor."

By using categories, some information can be lost.

For instance, one American with an annual income of $0 and another with an annual income of $12,000 are both classified in the same legal category—poverty—even with significant differences in living situations. Similarly, someone with a net worth of $30 million and someone with a net worth of $100 billion are both classified as Ultra-High-Net-Worth Individuals, even while there are tens of thousands of individuals in the former category and just a few in the latter category.

One disadvantage of quantitative data is that it's harder to make sense of and model than categorical data. Categorical data inherently simplifies data by reducing the number of data points.

What's more common: Quantitative or categorical data?

There's no simple answer as to what's the more common data type.

Categorical data is often easier to collect. For example, given someone's Facebook profile, you can likely get data on their race, gender, their favorite food, their interests, their education, their political party, and more, which are all examples of categorical data.

On the other hand, you probably wouldn't be able to find out their exact income, their weight, their spending habits, or other exact quantitative metrics (with some exceptions like age).

Under the hood, however, the situation is quite different, as Facebook collects vast amounts of data on each of its users, much of which is quantitative, such as the amount of time spent viewing a post, the number of posts viewed, the number of profile views, the number of link clicks, the number of application opens, and so on.

Ultimately, we create large amounts of both data types every day, with virtually every action we take. When you pick up a new smartphone, sensors recognize that it was picked up, by tracking the exact spatial location of your phone at any point in time, which is an example of quantitative data. Then, as it recognizes that your phone was picked up, it may change a variable like "Status" to be "Active" instead of "Inactive," causing your phone's lock screen to light up.

Time Series

Time series data is a type of data that records events happening over time, which is especially useful in predicting future events.

To give a very simple example, here's a time series dataset with three data points: In 1975, Earth's global surface temperature was anomalous by 0.0 degrees Celsius, in 1995 it was +0.5 degrees Celsius higher than normal, and in 2015 it was +0.9 degrees Celsius higher than normal.

One of the key tenets of time series data is that when something happens is as important as what happens. In marketing, for example, the time it takes a customer to go through the steps of the marketing funnel is an important predictor of revenue.

Common Applications

One of the most important uses of time series data is forecasting. This is because the past is the best predictor of the future. Let's explore some common applications of time-series data, including forecasting and more.

Marketing Journey

Marketing is a journey, and the customer's journey through the marketing funnel can seem unpredictable.

However, there are many ways to predict the customer's journey and reach them at the appropriate time to increase customer engagement and conversion rates. By understanding customer journeys, marketers can also create a more relevant and compelling content experience for each stage of the journey.

For example, if you are running a marketing campaign on Instagram and want to know how many clicks your advertisements will receive, you could forecast clicks based on historical data.

To give another example, time series forecasting can be used to predict when customers will make their next purchase. This allows companies to make decisions about when to launch new products and when to send emails or other consumer messaging.

Revenue Run-Rate

Revenue run-rate is predicting revenue based on what has happened in the past.

This is an important metric for companies because it helps them plan for future revenue needs. Revenue run-rate is an annual metric, which is traditionally calculated by multiplying the average revenue per month by 12, or the average revenue per quarter by 4. This will give a rough estimate of how much revenue the company will have per year.

That said, this is a very rough method of estimating revenue, which can be highly inaccurate. For example, businesses like fitness centers typically out-perform in January, due to New Year's resolutioners, so they wouldn't be able to accurately forecast revenue with traditional means. The opposite situation holds true for a landscaping company, which likely won't see much business in January.

A number of other variables impact revenue as well, from dynamic budgets to new competitors or new product innovation. Traditional calculations, which are based purely on multiplying historical revenue, are ignoring all these other factors.

Using Akkio's forecasting, you can accurately predict revenue run-rate based on any number of complex variables in your data.

Stock or Crypto Values

Predicting stock and crypto prices is notoriously difficult, especially considering the technical difficulties of manually building and deploying forecasting models.

That said, for investors who are interested in forecasting assets, time series data and machine learning are must-haves. With Akkio, you can connect time series data of stock and crypto assets to forecast prices.

It's important to remember that stocks and crypto are different types of investments, as crypto markets are much smaller and more volatile. Investors should be wary of their own emotions when investing in stocks and crypto.

Device Health

Manufacturers are using time series AI for predictive maintenance and monitoring equipment health. The AI systems are able to identify when changes need to be made to improve efficiency. They are also able to predict when equipment will break down and send alerts before it happens.

These technologies are saving manufacturers money by not having to spend on unexpected repairs or urgently replace machinery when it is no longer working.

Time Series Datasets

For non-experts, finding high-quality time series datasets is a challenge. Fortunately, there are a huge amount of free, high-quality time series dataset sources available online.

Let's explore a couple of time series dataset sources.

UCI Time-Series Repository

The UCI repository features 48 time-series datasets, ranging from air quality to sales forecasting data.

Most of the data is offered in CSV format, so it's easy to read with tools like Akkio, with no manual pre-processing needed. Just connect a dataset, and you're good to go!

World Bank's World Development Indicators

The World Bank provides a wildly extensive databank featuring 79 databases for 264 countries with data as far back as 1960.

The World Development Indicators database, for example, includes over 1,440 data columns to pick from, ranging from high-level indicators like "percent access to electricity" to very niche indicators like "rural population living in areas where elevation is below 5 meters." The Education Statistics database includes almost 4,000 data columns.

There's no easy answer to how many time-series datasets are offered, but if you treat each potential time-series dataset as a univariate problem, then there are millions of datasets from this source alone (79 databases across 264 countries with an average of 2,000 data columns).

Special considerations for time series data

Time series data can be a particularly tricky data type to work with, for a number of reasons. We've highlighted some special considerations to keep in mind when working with time-series data.

1. Time series data is sequential, but many algorithms for predicting the future are not.

In a time-series dataset, the temporal aspect is crucial, but many machine learning algorithms don't use this temporal aspect, which creates misleading models that aren't actually predictive of the future.

For example, a "random walk" model is a stochastic process, which means that it's simply not possible for it to accurately predict future outcomes from historical data.

To give another example, basic regression models ignore temporal correlation in the observed data and predict the next value of the time series based merely on linear regression methods.

Moreover, many time series models can easily "overfit" to the data, by finding spurious correlations, instead of causal variables.

For example, there's a positive correlation between ice cream sales and murder, but obviously not because eating ice cream makes you want to murder people. This is what's known as a "spurious correlation."

In the case of ice cream sales and murder, what's happening is that ice cream sales increase in the summer, which is when more people go outside, causing a natural increase in crime (fewer crimes are committed when everyone's bundled up inside during the winter, versus, say, when there's a sports events in the summer with 50,000 attendees packed in a stadium).

2. It is a lot of work to produce a model that predicts the future from time-series data.

Modeling time series data is an intensive effort, requiring pre-processing, data cleaning, stationarity tests, stationarization methods like detrending or differencing, finding optimal parameters, and more.

Doing this manually requires a high degree of technical expertise, not to mention a large time commitment. With Akkio, these complex processes are automated in the back-end, so you can forecast data effortlessly.

3.Time series data is often not as accurate when it comes to predicting the future, because many things that have happened in the past are simply no longer relevant to the future.

If you've ever considered investing, you've likely read a financial disclaimer along the lines of: "Past performance is no guarantee of future results."

It's actually a legal requirement for asset management firms to give such a disclaimer, because, well, there's really no way to know what the future holds. The best we can do is assign probabilities to certain values.

Indeed, even generating accurate probabilities is immensely challenging, as the world is constantly changing. Predicting COVID-19 cases is a great example of the challenges of time series forecasting, as virtually all forecasts failed.

Even now, accurate forecasts are extremely difficult, considering that much past data is no longer relevant for the future, given new vaccines, new strains, and ever-changing regulations around travel, social distancing, quarantines, and so on.

Feature engineering for time series data

Feature engineering is the process of creating new features from existing data.

One challenge with time series data is that it's often not stationary. Stationarity means that a time series is a sequence of observations of the same variable, taken at equally spaced times. If the observations are equally spaced in time and do not contain any trends or seasonality, then it's stationary.

Creating stationary data is a form of feature engineering, and the two most common techniques for transforming time series into stationary data are differencing and transforming.

That said, with no-code AI tools like Akkio, you can build and deploy time series models without any manual feature engineering needed, as this is all done automatically after a dataset is connected.

How much data do I need to train an ML model?

Data is the fuel that makes machine learning tick. For the most part, the more data you have, the more accurate your model will be, but there are many cases where you can get by with less.

Machine learning models are pattern matching machines. They can only capture and predict patterns that have been seen before. This is the one big catch with machine learning. If you want to predict what happens with new data, the model has to have seen similar data before.

It's also important to note that there's no golden rule for how much data you need. For example, while Akkio's lead scoring demo dataset has over 40,000 rows of data, the text classification demo dataset has just 1,000 rows of data, and both achieve roughly 90% accuracy. Meanwhile, the credit card fraud demo dataset has nearly 300,000 rows of data!

It's best to explore the modeling process for your dataset and see what it takes to get high accuracy.

Do you have too little data?

Accurate machine learning models can be made with as little as a few hundred rows of data. If you truly have extremely little data, say less than a few hundred rows, you can try a few things.

One is data augmentation: A process where data is generated by adding in fake data examples. You can also merge in other datasets, whether internal or external, on shared columns to increase the overall dataset size.

For example, suppose you're building a model to classify customer support tickets based on urgency. If you need more data, you'll want to ensure that you have a pipeline in place that's generating this data for you. In such a case, your support teams should be tagging the urgency of incoming tickets, so you can later export this data to fuel your machine learning model.

Depending on the use-case, you can even turn to crowdsourcing platforms like Amazon Mechanical Turk. These platforms allow you to hire people from all over the world to do small tasks for you at low prices, like collecting and labelling data. You may not want to do this if you're a small company with limited resources, but if you're a large company and want more data quickly, this may be a good option for you.

Yet another method is to scrape data from the Internet, which is again use-case dependent, but potentially an easy way to boost your dataset size, given the open nature of a lot of Internet data, such as social media posts.

Do you have too much data?

There are instances when it feels like you could have too much data. If your dataset is too large, it becomes difficult to explore and understand what the data is telling you. This is particularly the case with big data in the order of many gigabytes, or even terabytes, which cannot be analyzed with regular tools like Excel or even typical Python Pandas code.

Given that it's possible to make high-quality machine learning models with much smaller datasets, this problem can be solved by sampling from the larger dataset, and using the derived, smaller sample to build and deploy models.

ML models at any size

A good example of a massive AI model is Google's latest language model, which is an incredible 1.6 trillion parameters in size—too large for us to practically comprehend, though for comparison, there are just 86 billion neurons in the human brain.

At the same time, it's possible to build machine learning models that are around 10 orders of magnitude smaller than Google's language model.

For example, the perceptron is a classifier that was developed in the 1950s. These single-layer neural networks are trained by assigning inputs to different outputs, with the network adjusting its weights until it can correctly predict the output for new inputs. The perceptron is limited by its lack of memory and by not being able to extrapolate relationships between data points that it might not have seen, but at its core, it can be the basis of a functioning model with just a few parameters.

Quantity isn't everything

It's important to remember that quantity isn't everything when it comes to data. Even if you have a lot of data, your model may not work well. In order to have high quality models, you need high quality data. This means that your data needs to be clean and easy to work with so that it can be used effectively.

In other words, it's better to have a small, high-quality dataset that's indicative of the problem that you're trying to solve, than a large, generic dataset riddled with quality issues.

After all, not all data is valuable. As Nate Silver, Founder of FiveThirtyEight, says: "Every day, three times per second, we produce the equivalent of the amount of data that the Library of Congress has in its entire print collection, right? But most of it is like cat videos on YouTube or 13-year-olds exchanging text messages about the next Twilight movie."

Experiment to find out how much data you need

Machine learning is getting easier and faster. There's no need to waste a lot of time on preparation, as a huge dataset isn't a prerequisite. As Adam Savage puts it: "In the spirit of science, there really is no such thing as a 'failed experiment.'" Simply experiment and see how much data you need.

In the last few years, machine learning and AI tools have been getting simpler and faster. The days of waiting weeks or months for building and deploying models are over. With Akkio, you can build a model in as little as 10 seconds, which means that the process of figuring out how much data you really need for an effective model is quick and effortless.

With traditional machine learning, you typically need a large dataset in order to get sufficient training data. But with Akkio, it's possible to create compelling models with as little as 100 or 1000 examples. As we've explored, if you find that you're not getting great results with a small dataset, you can always try merging on new data, data augmentation, crowdsourcing platforms, or simply turning to online dataset sources.

Data preparation for machine learning

Preparing your data for the training of a machine learning model can range from simply connecting your existing business operations technology platforms (Salesforce, Marketo, and Hubspot, etc) and data-stores (Snowflake, Google Big Query, etc) to business wide data hygiene programs that take months but yield clean data for optimum performance. You also need to narrow down the dataset used for training so it only has the information available to you when you want to predict a key outcome. We have designed Akkio to work with messy data as well as clean - and are firm believers in capturing 90% of the value of machine learning at a fraction of the cost of a data hygiene initiative. To learn more about preparing your data for machine learning click here.

Data augmentation for machine learning

The performance of a machine learning model is primarily dependent on the predictive accuracy of its training dataset with respect to the outcome of interest. If you were able to know everything about a system (quantum physics aside) you would be able to perfectly predict its future state. In reality most datasets contain a small subset of information about a system - but that is often more than enough to build a valuable ML model. That said, adding in additional data can often help improve predictive performance. This is called data augmentation. To learn more about data augmentation for machine learning click here.

Bias in Machine Learning: What is it and how can it be avoided?

One very important thing to be aware of when using machine learning is that biases in the dataset used to train the model will be reflected in the decision making of the model itself. Sometimes these biases are not obvious in your data - take for example zip or postal codes. Location information encodes a lot of information that might not be obvious at first glance - everything from weather to population density to income, housing, to demographics information like age and ethnicity. These patterns can be helpful, but also have the potential to be harmful when the models are used in ways that reinforce unwanted discriminatory outcomes (both ethically and legally). Click here to learn more about bias in machine learning and how to minimize it.

Use Cases of Machine Learning

Machine learning is a subset of artificial intelligence that is focused on systems that can learn from data.

While we'll explore some of the top applications of machine learning across a number of industries, the academic world is also using AI, largely for research in areas such as biology, chemistry, and materials science.

Energy

Renewable Energy

Renewable energy is one of the fastest-growing sources of power generation worldwide. In 2020, it accounted for 80 percent of new power capacity globally.

AI is critical for successful adoption. AI can balance electricity supply and demand needs in real-time, optimize energy use and storage to reduce rates, and help integrate new, clean sources into existing infrastructures. AI can also predict and prevent power outages in the future by learning from past events.

For example, when a grid is overwhelmed by demand, AI can forecast the trajectory for that grid's flow of energy and power usage, then act to prevent a power outage. AI can also predict when a power outage will occur in the future, so utilities can take proactive measures to minimize the outage's effects.

Additionally, AI can even help with wind energy. The power of the wind is ever-present, but harnessing it is not easy. Windmills have been used for centuries to capture wind power, but this process is difficult and costly.

But now AI can change the game. AI can calculate how wind turbines should be rotated so that as few turbines as possible are in the wind shadow of the other. Using data collected from the terrain, the height and size of the turbines, and meteorological data, AI can work out how wind turbines should be rotated to harness the wind.

Insurance

Insurance Pricing

The insurance industry is highly competitive. The simple fact is that if you are not consistently profitable, you will be driven out of the market. To maintain profitability, insurance firms must be able to accurately predict high-risk, high-cost individuals.

Indeed, data shows that 70 percent of new North American insurance companies fail within 10 years. This is the status quo, as insurance firms often cannot accurately price their plans, leading to tremendous losses.

AI has been shown to be highly accurate when it comes to predicting future claims costs. This accuracy allows you to assess the risk of insuring an individual based on their past claims history and use this information to correctly price your premiums.

This is crucial because it will allow you to stay profitable in a high-risk industry where you are always at risk of being driven out of business by adverse selection.

With Akkio, AI-powered cost modeling can be done in clicks, enabling insurers to leapfrog competitors that are stuck using traditional, laborious, and inaccurate cost models. This cost modeling solves one of the biggest problems insurers face today: Choosing who to insure, and at what rates.

Claim Development Modeling

In the insurance industry, it's all about risk management. And when you're making predictions about risk, you want to do it right. In the past, the industry relied on outdated modeling techniques that often led to under- or over-pricing claims. That led to higher premiums for consumers and a host of other problems.

But AI is solving this problem. With these new machine learning techniques, it's possible to accurately predict a claim cost and build accurate prediction models within minutes. Not only that, but insurers can even build models to predict how claims costs will change, and account for case estimation changes.

That means insurance companies can price their policies more accurately and offer lower premiums for consumers, leading to lower costs of coverage for everyone. It also helps insurers be more competitive and attract more customers, which is especially important as the industry faces stiff competition.

Akkio's platform makes this possible by enabling users to create models based on their own data, and then deploy them across any number of environments with just a few clicks. This reduces the need for costly and time-consuming custom development work, and translates into lower costs for the company overall.

It also enables insurers to respond faster to a changing insurance market, which provides a critical edge against competitors that are still relying on outdated techniques like regression modeling in Excel. The result is an improved customer experience that translates into higher sales volume and happier shareholders.

Claim Payment Automation Modeling

Claims are a major expense for insurance companies and a frustrating process for policyholders. At the same time, insurance claims are extremely common, as by the age of 34, every person driving since they were 16 are likely to have filed at least one car insurance claim.

The inefficiencies in processing claims is bad for both parties: the customer wastes time and the insurance company spends more on processing than they could have spent on settling the claim. Akkio's no-code machine learning can model when it's best to pay off claims automatically, so that you can minimize wait times for customers and maximize ROI for your business.

Predicting when a customer will make a claim is not simple. Your risk profile changes over time, and so does the competitiveness of your market. Given the right historical data, Akkio's machine learning models take all of this into account, making it easy to find the optimal solution for your specific needs.

Simply upload your data, and let Akkio do the heavy lifting, giving you more time to focus on what really matters: running your business.

Insurance Conversion Modeling

Insurance companies are always searching for new ways to attract new customers, and they need to optimize their marketing efforts to help them grow.

A key problem that many insurance companies are struggling with is how to make accurate pricing decisions. Given that insurance is sold by quoting a policy, accurately estimating the conversion rate from quote to policy is essential. Akkio allows you to gather historical data, make estimates about the probability of conversion, and then use those predictions to drive your pricing decisions.

Accurately modeling insurance conversion is key because it is an important determinant of insurance company profitability.

A key benefit of an AI-based approach is that it allows insurance companies to adjust prices for customer segments without manually creating and testing a wide range of pricing variants. This ensures that marketing dollars are spent effectively and efficiently on segments where there is the greatest chance of conversion.

Fraudulent Claim Modeling

With over $40 billion in insurance fraud in the US alone, according to FBI statistics, it's no wonder that insurers are looking for ways to reduce fraudulent payouts. One solution is to use machine learning to create models that can predict the probability of a claim being legitimate or not.

Fraudulent claim modeling is an excellent example of how predictive modeling can be used to analyze fraud in the insurance industry. Using a model built on past payouts, an insurer could, for instance, apply a scoring system to claims and automatically reject or flag those with high probability of being fraudulent.

Fraudulent claims don't just reduce the bottom-line for insurers, they can even lead directly to corporate bankruptcy, as research indicates. Moreover, fraud hurts consumers, who pay up to $700 a year in the form of increased premiums, in the US.

The traditional means of detecting fraud are inefficient and ineffective, as it's impossible for humans to manually analyze vast amounts of data at scale, which lets fraud slip through the cracks.

Akkio's potential in this area goes beyond the insurance industry. Modeling fraud is a popular use-case in the financial sector as well, for example to help eliminate fraudulent credit card applications and transactions.

Life Insurance Underwriting for Impaired Customers

Many life insurance companies do not underwrite customers who suffered from some serious diseases such as cancer. This is because it requires them to spend a long and expensive medical assessment process on the customer.

In insurance, the term "impaired" refers to applicants who don't meet the standard criteria to obtain a very affordable rate. As a result, impaired applicants are often un- or under-insured.

It's a wise business decision to increase the coverage for impaired customers and Akkio's AI is able to provide that capability.

While many who suffer from a serious disease can be accurately identified through a questionnaire, Akkio can achieve an even higher degree of accuracy by integrating the applicant's medical history and conditions. AI-driven predictive models use these factors to predict the risk of underwriting a serious disease survivor. The model predicts the risk of death, which is the ultimate impairment in insurance.

For insurers, it's possible to build the model in just minutes, opening up a new line of business and boosting the bottom line.

FinTech and Banking

Credit Card Fraudulent Transactions

Credit card fraud is a huge problem costing billions of dollars per year. Fraudulent transactions cost $28 billion in 2018, and they continue to grow rapidly. In fact, annual losses are expected to exceed $40 billion by the end of the decade.

With Akkio's no-code machine learning, the likelihood of fraudulent transactions can be predicted effortlessly. This reduces the number of fraudulent transactions, while at the same time increases customer satisfaction. For banks, this means less cost per transaction and more revenue and profit.

Akkio's fraud detection for credit card transactions is one example of how Akkio can help banks. By using a historical transaction dataset, machine learning models identify suspicious patterns and account for factors that are often overlooked in credit card transactions, like IP address changes, high-risk browsing behavior, or a low level of engagement with the transaction.

By using proprietary AI training methods, Akkio can be used to build fraudulent transaction models in minutes, which can be deployed in any setting via API.

Credit Default Rates

Credit default rate is the percentage of loans that default. The credit default rate problem is difficult to model due to its complexity, with many factors influencing an individual's or company's likelihood of default, such as industry, credit score, income, and time.

Understanding the factors that lead to credit card defaults can help lenders better assess the risk of lending to borrowers, and ultimately boost the bottom-line. Credit risk is a measure of the likelihood that a person will be unable to repay a debt, and this is what lenders use to determine whether or not to offer credit. In finance, credit risk is the risk of default on an obligation that arises due to the uncertainty of future cash flow.

Akkio's API can help any organization that needs accurate credit risk models in a fraction of the time it would take to build them on their own. Akkio makes it easy to build a model that predicts the likelihood of default based on data from the past.

In addition, Akkio can be used for automatic model retraining, so that once a model is built, it's easy to maintain and update as needed. This makes it possible for organizations not just to save time on predictive modeling tasks but also to be confident in their models at all times.

Digital Wealth Management

Digital Wealth Management is a competitive field. In this market, it's not just about having the best investment products, but also about how to distribute them effectively while managing client assets. Akkio's machine learning algorithms can be deployed to constantly analyze data from your existing clients' portfolios to find new opportunities and assign values for each of your prospects.

It's important to diversify your portfolio to make sure you are investing in the right technologies and companies. AI can help diversity portfolios by finding new investment opportunities

Akkio helps asset managers learn which customers are more likely to invest in particular categories based on their previous investments and demographic information, as well as information like their risk appetite.

AI can even be used to automate investment analysis, by ingesting financial data from sources like a securities market to predict the probability of stock prices rising or falling. These predictions can then provide real-time strategy recommendations for individuals or institutional investors.

The result? A successful asset management strategy that attracts new clients and captures a greater share of existing client assets at the same time.

Further, algorithms have been used in stock trading for decades. For example, a 1986 New York Times article titled “Wall Street’s Tomorrow Machine” discussed the use of computers for evaluating new trading opportunities.

Today’s AI trading is a form of automated trading that uses algorithms to find patterns in the market and make trades. AI traders can also be used to optimize portfolios with respect to risk and return objectives and are often used in trading organizations.

AI-powered trading systems can also use sentiment analysis to identify trading opportunities in the securities market. Sophisticated AI algorithms can find buy and sell signals based on the tone of social media posts.

Blockchain

A blockchain is a decentralized database that stores information in blocks of data. The blocks are linked together through cryptography to create a history of all transactions. The system relies on consensus among the users of the network about the validity of information and data, making blockchains more secure than other types of databases.

However, as blockchain technology becomes more popular, security threats are also increasing. Larger blockchains like Bitcoin and Ethereum are practically impossible to attack due to the sheer amount of resources required. That said there are hundreds of smaller blockchains at risk.

MIT Technology Review reports, “marketing slogans and headlines that called the technology ‘unhackable’ were dead wrong,” as blockchains can be rewritten if an attacker is able to muster over 51% of the computational power defending a network allowing the attacker to reallocate ownership of funds. One such example is when Ethereum Classic (a fork off of Ethereum) suffered a 51% attack 3 times in a single month. In 2020, there were over 120 blockchain attacks, leading to losses to the tune of nearly $4 billion.

While preventing 51% attacks depends on distributed participants allocating compute resources to chain defense, users and exchanges need to be able to detect anomalous behavior when it happens on a chain (so they can attempt to minimize loss of funds).

Akkio’s machine learning algorithms can detect anomalies in real-time, alerting you and enabling you to take action quickly before additional damage is done. With Akkio’s AutoML, it only takes minutes to build a fraud detection system tailored to your needs.

Healthcare

Drug Delivery Optimization

The pharmaceutical supply chain is notoriously fragile, leading to shortages, higher costs, and safety issues. Part of these issues lie in under-optimized drug delivery systems.

Pharmaceutical firms spend millions of dollars shipping drug samples to doctors and hospitals. Simple analyses uncover situations for order consolidation, such as when the same location requests two or more drug samples. However, manually looking at the data for order consolidation quickly becomes infeasible at scale.

AI helps optimize supply chain delivery processes by predicting which orders can be consolidated, no matter how complex or how many orders there are to process. That's the killer advantage of AI: It's incredibly fast and accurate compared to traditional techniques.

AI can be used to find the best locations for consolidated shipping, estimate cost savings, and improve customer satisfaction. Instead of putting out fires related to unoptimized supply chains, health systems can now focus on what truly matters: Helping patients.

Disease Propensity

In a world of virtually unlimited data and powerful analytics, it's easy to see why health systems are looking for ways to better understand the health of their patients. With AI platforms, teams can connect to various data sources, like lab results and HIE, and use machine learning models to predict the severity of a patient's condition and what type of care they will need.

Medical professionals should consider screening patients that may have a higher likelihood for a particular disease. If they see a patient that could be predisposed to developing an illness, treating them right away will lead to better health outcomes, in addition to being more fiscally responsible than not seeing them until they're carrying it.

Ultimately, using AI to automate disease propensity modeling has the potential to save hospitals and other healthcare providers millions of dollars per year by reducing unnecessary emergency room visits and readmissions.

Modeling ICU Occupancy

Staffing and budgeting for a hospital ICU is always a difficult decision, and it's even harder when you don't know how quickly the patient load will change. With machine learning, hospitals can easily make projections about their occupancy by modeling historic data to account for trends.

Exceeding capacity limits, as has happened in ICU rooms around the world as of late, often results directly in patient death. Higher occupancy rates are clearly correlated to higher death rates.

With AI, hospitals can quickly create a model that forecasts occupancy rates, which consequently leads to more accurate budgeting and staffing decisions. Machine learning models help hospitals save lives, reduce staffing inefficiencies, and better prepare for incoming patients.

Forecasting models also help hospitals make better decisions about what services they need to offer their patients. Healthcare has been rapidly changing over the last few years, with an increased focus on providing holistic care and individualized treatment plans. Further, forecasting can help hospitals anticipate patient needs and provide the right services to meet expectations.

Ultimately, machine learning algorithms make it easy for hospitals to predict the next step in their operations and make more informed decisions about future staffing needs. The result is healthier, happier patients and a stronger bottom line for hospitals.

Estimating Sepsis Risk

Sepsis is a life-threatening condition that can develop suddenly and with devastating consequences. It is a leading cause of death in intensive care units and in hospital settings, and the incidence of sepsis is on the rise. Doctors and nurses are constantly challenged by the need to quickly assess patient risk for developing sepsis, which can be difficult when symptoms are non-specific.

Decades ago, sepsis wasn't much of a concern. Today, sepsis accounts for almost a fifth of human deaths.

AI complements medical professionals' expertise by providing data-driven insights to identify patients at high risk for developing sepsis. Medical professionals can leverage the power of machine learning to aggregate patient data and generate automated alerts tailored to each patient's unique needs.

Machine learning models are designed to learn from historical data, which can include past sepsis cases, to provide accurate predictions, enabling healthcare professionals to confidently identify patients at high risk for developing sepsis.

Hospital Readmission Risk

The average cost of a hospital readmission ranges from $15,000 to $25,000, which leads to wasted resources, unnecessary tests, potentially harmful treatments, delayed patient care, and other damaging consequences.

Machine learning can help in reducing readmission risk via predictive analytics models that identify at-risk patients. By feeding in historical hospital discharge data, demographics, diagnosis codes, and other factors, medical professionals can calculate the probability that the patient will have a readmission.

AI makes it easy for hospitals to identify which patients are most at-risk for readmissions. No-code AI tools don't require any IT work or coding, so hospitals can save money and improve the quality of care they provide.

Ultimately, AI's hospital readmission risk use-case can help hospitals reduce their costs and increase the quality of care they are able to provide to their patients.

Public Sector

Counterterrorism

Terrorism is a top concern for intelligence and law enforcement agencies around the world. After 9/11, preventing terrorist attacks became a heavily-funded, prime directive for a number of government agencies.

As described in a United Nations Office of Counter-Terrorism report on AI, government agencies can use predictive modeling to identify red flags of radicalization, detect the spread of terrorist misinformation, and counter terrorist narratives.

Machine learning isn't just for marketing; it can also be used to help prevent terror attacks by identifying patterns in past events and predicting future ones, saving lives, and making the world a safer place.

Fraud detection

Fraud is an issue that is costly, not only to the government and its citizens, but to companies as well. Every government agency from the IRS to the Social Security Administration suffers significant losses from fraud.

In fact, as explored in an Association of Certified Fraud Examiners report, a study of nearly 3,000 cases of occupational fraud found that government entities "were the most represented sectors among the fraud cases analyzed." While much public discourse centers around governments as perpetrators of fraud, the reality is often that government employees and agencies are often the targets of a wide-range of fraudulent activities.

Fraudulent activities can be difficult to detect, costing agencies valuable time and resources. Ultimately, AI makes it easy for government agencies to detect fraudulent activities as they happen, saving them time and resources while also safeguarding taxpayer dollars.

Insider threat

In the age of digital transformation, attack vectors are getting ever larger. As a result, even government agencies are at risk of being breached by insiders (or ex-employees) who want to use their data for malicious purposes.

At the same time, there are a number of insider threats that can seem innocuous in nature, but costly nonetheless, such as sending company information over a personal account, or even accidentally misconfiguring access credentials.

For example, while cybersecurity firms like to keep their exact techniques private, research shows that AI can accurately identify malicious emails, which cost governments billions of dollars if undetected.

To make sure that firms don't have to pay for these kinds of internal breaches, agencies need to proactively block any potential misuse, using machine learning to identify risks.

Cybersecurity

Cyberattacks are on the rise, with real-world consequences for everyday people. Recently, for instance, hackers stopped gasoline and jet fuel pipelines and closed off beef and pork production at a leading US supplier. These are just a couple of examples of the tens of thousands of annual cybersecurity attacks.

One of the main challenges in cybersecurity today is an ever-growing attack vector. As more and more of our world goes digital, there's more data to keep track of, and it's easier for hackers to go unnoticed. Manually combing through this data can only get you so far, but AI can scan massive amounts of data in real-time.

No-code AI enables security teams to build, deploy and refresh models to predict incoming threats in real-time, whether it's scanning incoming emails for malicious threats or flagging concerning IP activity, so they can prevent a breach before it happens.

Ultimately, this enables security teams to reduce their risk exposure and prepare for an increasingly hostile cyber landscape. Teams that fail to deploy AI for cybersecurity will be more vulnerable to attacks compared to other market players who do.

Customer Support

Support Ticket Topic Classification

Good customer service is of universal importance, with surveys indicating that 96% of customers feel customer service is important in their choice of loyalty to a brand.

Customer service is also a major factor in customer retention. In other words, people are more likely to stay with a company if they're satisfied with the service they receive.

AI-based classification of customer support tickets can help companies respond to queries in an efficient manner. By combining natural language processing and machine learning, AI can be used to automatically group queries into predefined categories, making it easy for customer support teams to select the appropriate department to handle a query based on their area of expertise.

Essentially, by digesting past queries to find patterns in terms of content, AI can learn how to classify new tickets more accurately and efficiently. This means that with time, AI-based ticket classification will become an integral part of any organization's customer service strategy.

Support Ticket Prioritization

Customer support teams need to handle a huge number of customer queries in a limited time, and they're often not sure which tickets need to be addressed first. Machine learning models can rank tickets according to their urgency, with the most urgent tickets addressed first. This relieves teams of the burden of deciding which tickets require the most attention, freeing up more time for actually addressing tickets and satisfying customers.

Predictive analytics is also useful for identifying patterns in the data so that customer queries can be more accurately met with answers, and it allows teams to improve their customer experience by responding faster.

Social Media Sentiment Analysis

Social media is an invaluable tool for marketing and customer support teams, but it's a complicated and fast-moving landscape. Every day, millions of people post their thoughts, opinions, and suggestions to social media about brands they're interacting with. From a raving comment to a scathing review, social media posts can have a big impact on your company's success.

Machine learning can help teams make sense of the vast amount of social media data, by automatically classifying the sentiment of posts in real-time thanks to models trained on historical data. This enables teams to respond faster and more effectively to customer feedback.

Ultimately, this allows marketers and customer service teams to identify early warning signs of dissatisfaction before they spiral out of control and needlessly drive away customers.

Sales

Finding Duplicate Customer Records in Your Database

In the process of data entry, we know that errors will be made. Humans are not perfect and this includes those who code the data: editing mistakes can occur such as inverting an "S" or a "Z" in the input document. It is reasonable to assume that there may be multiple copies of your records in which different people may have typed one letter wrong or did not notice inconsistent formatting, such as "smith" versus "Smith," before saving it as a new version.

Additionally, data can be brought in by multiple systems, with different column values, such that duplicates won't be found by traditional means (e.g. one system has the first and last name, while another system has their email).

Detecting duplicates is notoriously difficult, requiring manual intervention to identify duplicate records. This can be time-consuming and prone to human error. AI is different: it's fully automated and can detect duplicates for all types of fields with high accuracy.

AI is essential for complex deduplication tasks, because the same record could show up multiple times throughout your database. With AI, you can detect these duplicates even if they have different data fields - making it easy to clean up your database so that it adheres to best practices without any manual intervention.

Lead Scoring

Lead scoring is a powerful way to determine which leads are most in need of your attention. AI enables teams to automatically predict the likelihood that each lead will become a paying customer. Armed with these insights, marketing teams can decide which leads to pursue and spend time on, and which to put on the back-burner.

Today's lead scoring is powered by machine learning that leverages any historical data, whether from Salesforce, Snowflake, Google Sheets, or any other source, to predict the likelihood a given lead will convert.

This insight helps marketing teams to identify leads that are in need of more attention, as well as those that are likely to be a waste of time for the team.

Sales Forecasting

As a business, forecasting is one of your most important tasks. It's what allows you to plan ahead and make better use of your budget.

Machine learning can help you do that with unparalleled accuracy, even in unpredictable economic environments. No-code AI can be used to quickly build a model from past sales data and predict the sales you're likely to receive in the future. With no-code AI, you can get accurate forecasts in a matter of seconds by uploading your product catalog and past sales data.

Instead of relying on rules of thumb or gut feelings, AI offers a more scientific approach that lets you make better decisions about your budget, staff hiring, and promotional campaigns.

This is essential for businesses that need to know how to budget for the future or optimize their limited resources. Forecasting models can be deployed through a web-based interface, API, Salesforce, or even through Zapier, making it easy to get started in any setting without requiring any data science know-how.

Marketing

Direct Marketing

The way we consume goods has changed. In the past, we would go to the store, pick out what we needed, and purchase it. Nowadays, we can order what we need from the comfort of our own home and have it delivered to our door.

As a result, the way we are marketed to has changed. Direct marketing is an excellent way for businesses to reach their potential customers, and it's a largely under-utilized opportunity.

That said, it's often difficult to determine which prospects are the most likely to purchase. Marketing to uninterested leads isn't just a waste of time and money - it can be a huge turn-off to those leads from ever deciding to make a purchase decision.

That's where data-driven AI comes in.

AI can find the best prospects among a particular group and determine the best way to reach them. This means you can quickly and easily identify the most valuable leads, and then contact them with a personalized message that speaks to their particular needs.

With no-code AI, you can effortlessly prioritize and classify leads based on their likelihood of converting, all at a fraction of the time and cost that traditional methods require.

Loyalty Program Usage

A loyalty program is a reward program that gives points or other awards to customers who shop at a particular establishment. A typical example might be a program that provides each customer with ten points for every dollar spent at the store, and if a customer collects 1,000 points, they are given $10 off their purchase.

Loyalty programs are designed to incentivize customers to shop with the company on a regular basis, and they usually consist of various tiers of rewards, depending on how much the customer spends each time. The most effective type of loyalty program is one that provides increased benefits based on the amount of money spent, as customers are more likely to be motivated by the prospect of an increased reward.

Unfortunately, even if you have a good understanding of your customers' behaviors and preferences, it is not easy to predict which rewards will incentivize them most effectively. While your neighborhood coffee shop might offer a free coffee for every fifth visit, the scale and complexity of loyalty programs are orders of magnitude greater for large, data-driven firms.

Machine learning algorithms can analyze past data and detect which customer segments are most likely to respond positively to certain rewards. This helps managers make informed decisions about which rewards to offer and when, increasing the likelihood that they will convert.

Next Best Offer

One of the best ways that marketers can create a personalized experience for customers is by considering the "next best offer." This requires marketers to take into account all of the possible actions they could take with that customer and then select the most appropriate one.

As an example, suppose that a customer visits a website for information on renting. The customer can't decide between a studio or one-bedroom apartment, so she searches for more information on both and cannot find any definitive information. In this case, the "next best offer" could be to create a personalized email with links to articles and videos from both types of apartments, so the customer can decide which one is better for her.

Doing this manually is clearly impossible at scale. Businesses can use AI to offer the right product to the right person at the right time.

Businesses can automatically make recommendations in real-time, using predictive models that account for customer preferences, price sensitivity, and product availability, or any data provided for training.

Predicting the right offer for the right person at the right time is a huge undertaking, but AI makes it easy for retailers to optimize their operations. Best of all, retailers don't need any data scientists or AI specialists to deploy predictive models - no-code AI automatically powers recommendations with no coding required.

Multichannel Marketing Attribution

If your marketing budget includes advertising on social media, the web, TV, and more, it can be difficult to tell which channels are most responsible for driving sales. With machine learning-driven attribution modeling, teams can quickly and easily identify which marketing activities are driving the most revenue.

Marketing attribution models are traditionally built through large-scale statistical analysis, which is time-consuming and expensive. No-code AI platforms can build accurate attribution models in just seconds, and non-technical teams can deploy the models in any setting.

This lets marketing teams keep costs down while still pinpointing exactly where to allocate their marketing budget to optimize for the best ROI. Ultimately, this makes it easier to ensure that every dollar spent on marketing is worth it, so you're consistently getting the most out of your marketing budget.

By automating attribution, marketers can overcome the boring stuff and get more creative with what really matters. Armed with knowledge on how specific channels are performing, marketers can finally double-down on high-performing channels, eliminate the laggards, and strategize how to move forward.

Product Personalization

Consumers today expect personalized products and content.

Machine learning enables businesses to finally target consumers with the right message, at the right time, and on the right channel.

For example, rather than using one message to reach everyone on your website, machine learning could be used for sentiment analysis of customer reviews on your site, or your CRM or social media tools, to present different customer segments with different messages.

In addition, AI platforms can be trained on historical product purchase data to build a product recommendations model. For example, if a customer has purchased a certain product in the past, an AI API can be deployed to recommend related products that the customer is likely to be interested in.

This can be a powerful propellant for the bottom line, as research shows that 80% of consumers are more likely to make a purchase when brands offer personalized experiences.

Beyond personalized experiences, AI can even be used for personalizing products and services themselves.

While today, many of these individualized products are created by an individual designer or a custom order, personalized AI will make this process much more efficient, tailoring the product to an individual customer's needs and delivering it in a matter of days.

Customer Churn

The churn rate, also known as the rate of attrition, is the number of customers who discontinue their subscriptions within a given time period. For a company to grow, it must acquire more new customers than its churn rate.

It's quite a challenge to prevent customer churn, which is why it's so important for companies to be proactive.

Fortunately, AI has the power to do just that. Machine learning algorithms can identify the data patterns common among customers who are likely to churn, such as those with a high cost of acquisition or those that are misaligned with your ideal customer persona.

Armed with this knowledge, you can optimize your retention strategy by targeting high-risk customers with personalized offers or incentives before they leave. Moreover, marketing teams can tailor their strategies to avoid hi-gh-churn-profile leads.

The more data you have, the better. AI platforms like Akkio allow you to work with your data sources wherever they are - your CRM system, data warehouses, and other databases - to create the best model for predicting churn for your business.

Next Best Action

When it comes to marketing, there are always more tactics to explore than time or resources allow. Trying to decide which channel or activity to focus on that will have the biggest impact on revenue means you're forced to make guesses.

AI can put those guesses to the test. Machine learning algorithms can be fed with data from all of your marketing channels, as well as customer lifecycle information, to identify which activities are most likely to move each individual customer closer to purchase.

A/B testing is a great way to figure out how best to allocate marketing resources, but only if you can measure success accurately. That's where machine learning excels: it's able not only to measure and predict sales, but also predict what might happen if you try any given marketing tactic.

Google AdWords Bidding

Google AdWords is a huge part of most advertising budgets, but it can be difficult to get bidding right. If you bid too low, you lose out on opportunities. If you bid too high, your marketing ROI will dwindle.

However, machine learning can make this process easier by building a model off of past marketing and sales activities to predict the sales volume attributable to each AdWord, making it easy to determine the optimal price to bid to achieve your target ROI while avoiding losing the word to a competitor.

It is incredibly difficult and time-consuming for teams to build auction models that can capture complex human behavior. But no-code AI can be used to build accurate models with just a few clicks. Companies can deploy these models easily with an API in any setting or even with no-code tools like Zapier.

Ultimately, this enables marketing teams to boost the effectiveness of their ad spend, which is critical for success in an ever-more competitive landscape for consumer attention. Teams that fail to deploy AI for AdWords bidding will lose directly to their competitors that are using data-driven strategies.

Lead Scoring

Lead scoring is a crucial part of any marketing campaign because it helps you focus your time and resources on the potential customers that are most likely to become paying customers. In other words, an accurate lead scoring model helps you go where the money is. In fact, over two-thirds of marketers point to lead scoring as a top revenue contributor.

Accurate lead scoring can be tough, though. It's not easy to measure how well a customer will interact with your product without knowing much about them, so traditional lead scoring models rely on interest from the prospect to determine the score. Traditional approaches are highly limited, since they don't necessarily indicate the prospect's ability or true probability of making a purchase.

That's where AI comes in. Machine learning models use a wide range of factors to score marketing leads. With data-driven lead scoring models, you can have more confidence in your marketing decisions because you're looking at more data points than just interest from the prospect.

Employee Retention

Studies have shown that attracting and retaining top talent is one of the most important factors in a company's success. After all, the average employee exit costs an entire third of their annual salary.

However, as employee-employer relationships are shifting, the challenge of getting and keeping top talent is getting tougher. Year after year, employee attrition is increasing, and some are calling this crisis "The Great Resignation."

But there's hope: data. No-code AI platforms let HR professionals scan massive amounts of data - from hiring pipelines to employee history or performance reviews - to uncover insights to keep your best people working for your team.

With no-code AI, you can use machine learning algorithms to create predictive models that let you predict when an employee might be considering a job change, when they might be considering leaving their current position, or if they're simply unsatisfied.

This data-driven approach illuminates potential issues before they become major problems, giving HR teams the high-quality insights they need for more informed decision-making. With tools like Zapier, HR teams can even deploy predictive models in any setting without writing code.

How can I create and deploy a machine learning model?

For many, machine learning might as well be magic. But the truth is, as we've seen, that it's really just advanced statistics, empowered by the growth of data and more powerful computers.

Having said that, machine learning models are incredibly versatile tools that can add tremendous value across business units. We saw earlier, for example, how finance teams can use machine learning to predict fraud, marketing teams can score leads or predict churn, HR teams can predict attrition, and more.

Building the machine learning models to make these use-cases possible was once an arduous, resource-intensive task, requiring technical experts for data engineering, building pipelines, coding, maintaining infrastructure, and more.

As we've explored, no-code AI allows anyone to create and deploy machine learning models on their own, without needing programming skills. However, to become truly AI-driven, getting AI to work for you is not a one-time upgrade. It is a journey that will require an understanding of data management and the use of machine learning.

Another reason that code-based AI is problematic is that there is a shortage of programmers, and the shortfall is expected to grow as the AI industry grows. As ACM reports, there's actually a recent decrease in computer science graduates, in spite of increasing demand for them, fueled by delays in student visa processing, limited access to educational loans, and travel embargos.

Start with data

As we've seen, data is the fuel that powers machine learning engines, which is why data preparation is so important when building a model.

The expression "the more the merrier" holds true in machine learning, which typically performs better with larger, high-quality datasets. With Akkio, you can connect this data from a number of sources, such as a CSV file, an Excel sheet, or from Snowflake (a data warehouse) or Salesforce (a Customer Relationship Manager).

For example, suppose you'd like to use AI to score sales leads. If your business uses Salesforce, you can directly connect your sales dataset, and then select a column that relates to whether or not a deal was closed.

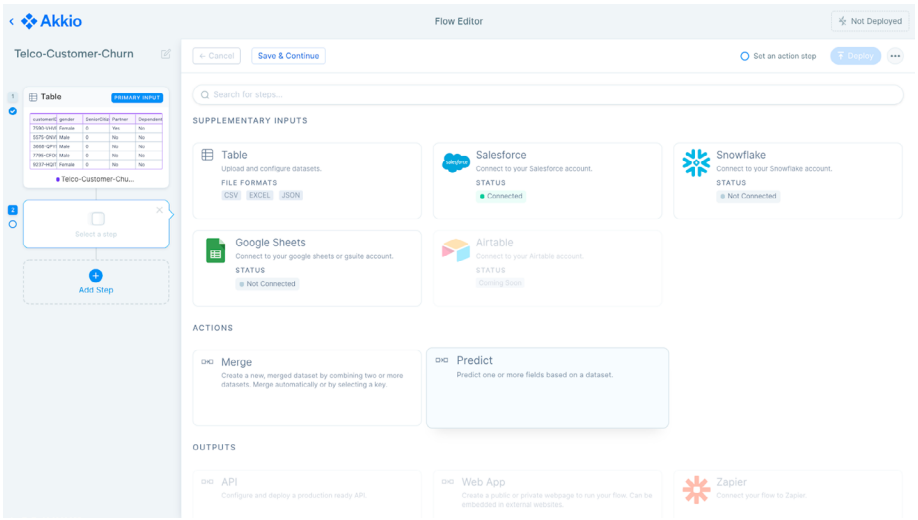
Many smaller sales teams keep it simple, using Google Sheets or Excel to organize lead data. Both of these sources can be easily connected to Akkio as well, and you'd build the model in the same way—by selecting the column you'd like to predict.

On the other end of the spectrum, some larger firms use Snowflake for handling massive amounts of sales data, which can be easily integrated with Akkio as well.

Train a model

We've explored how machine learning models are mathematical algorithms that are used to find patterns in data. To train a machine learning model, you need a high-quality dataset that is representative of the problem you're trying to solve. Let's walk through a practical example.

In Akkio, you can train a model by hitting "Add Step" once a dataset is connected, and then "Predict." Then, simply select the column to predict.



Generally speaking, there are two kinds of models you can train: Classification models and regression models.

A few examples of classification include fraud prediction, lead conversion prediction, and churn prediction. The output values of these examples are all "Yes" or "No," or similar such classes.

On the other hand, regression models are used to predict a range of output variables, such as sales revenue or costs.

After selecting "Predict," training either kind of model is the same: You'll select the column name you want to predict, whether it's called conversion, churn, attrition, fraud, or any other metric. You also have the option to select a "Training Mode," which ranges from 10 seconds of training time to 5 minutes, where longer training times may lead to more accurate models.

Behind the scenes

While the training process is done in just a couple clicks, a lot of work is done in the background.

It starts with software engineering to lay the groundwork for the platform itself. Software engineering is a branch of engineering that deals with the design, development, operation, and maintenance of software. Most of today's software development activities are performed by a team of engineers.

But that's not all. DevOps is used to help bring AI applications to production.

DevOps is a software development method that focuses on the collaboration between software developers and other IT professionals. It aims to shorten the time between the software's conception and its adoption by end users.

In order to build the AI pattern recognition models themselves, a number of different approaches are used. Pattern recognition is the ability to identify a pattern in data and match that pattern in new data. This is a key part of machine learning, and it can be either supervised or unsupervised.

The Bayesian approach to AI is a probabilistic approach to making decisions. Bayesian methods are used to estimate the probability of a hypothesis, based on prior knowledge and new evidence.

Another technique is dimensionality reduction, a process that reduces the number of dimensions of a dataset by identifying which are important and removing those that are not.

K-means clustering and PCA, or Principle Component Analysis, are two methods commonly used together. In order to group associated data points, k-means finds the partition in data, while PCA finds the cluster membership vector.

Random forest is another common method. A random forest is a machine learning method that generates multiple decision trees on the same input features. The hierarchy of decision trees is built by randomly selecting observations to root each tree.

Gradient descent is a commonly used technique in various model training methods. It's used to find the local minimum in a function through an iterative process of "descending the gradient" of error.

These AI methods are often built with tools like TensorFlow, ONNX, and PyTorch.

TensorFlow is an open-source software library for Machine Intelligence that provides a set of tools for data scientists and machine learning engineers to build and train neural nets. It is one of the most popular deep learning frameworks.

ONNX is an open-source modeling language for neural networks that was created to make it easier for AI developers to transfer their algorithms between systems and applications. This open-source AI framework was made to be widely available to anyone who wants to use it.

PyTorch is an open source machine learning library for Python, based on Torch. PyTorch provides GPU acceleration and can be used either as a command line tool or through Jupyter Notebooks. PyTorch has been designed with a Python-first approach, allowing researchers to prototype models quickly.

All of these model training processes are iterative, and many technical model training considerations are accounted for.

One of these concerns is overfitting, which happens when a model tries to predict every individual input that it might get instead of just being able to predict certain patterns in the data.

There are best practices that can be followed when training machine learning models in order to prevent these mistakes from happening. One of these best practices is regularization, which helps with overfitting by shrinking parameters (e.g., weights) until they make less impact on predictions. An additional best practice for successful training is using cross validation.

Another concern is called the “curse of dimensionality.” This happens when the number of inputs to a model gets too large for it to work properly, especially if many inputs are not statistically relevant to the outcome being predicted. A way to get around this is by simplifying or reducing the number of features or dimensions being used in order to make more accurate predictions - this is known as “dimensionality reduction.”

One technique for dimensionality reduction is called Principal Component Analysis, or PCA. PCA turns a large amount of data into a few categories that are most useful for describing the properties of what you’re measuring.

Evaluate model performance

Not all machine learning models are made equal. There’s a popular saying in the AI world: “Garbage in, garbage out.” If low-quality data is used to build a machine learning model, then the model will generate low-quality predictions as well.

There are a number of metrics you can use to evaluate the performance of a model. After making any model in Akkio, you get a model report, including a “Prediction Quality” section.

Classification

If you’ve built a classification model, the quality metrics include percentage accuracy, precision, recall, and F1 score, as well as the number of values predicted correctly and incorrectly for each class.

Here are what these fields mean:

- Accuracy: Accuracy measures how often a prediction is correct, and is calculated by dividing the number of correct predictions by the total number of predictions.
- Precision: Precision is the fraction of true positives out of the predicted positives. This is useful to consider when the cost of a false positive is high, such as in email spam detection. If an important email is incorrectly classified as spam, you’ll lose important information.
- Recall: Recall is how many of the actual positives your model captures. This is useful to consider when the cost of a false negative is high, such as in malignant cancer prediction.
- F1 Score: The F1 score combines precision and recall into one metric and weights them, in order to balance the consideration of false positives and false negatives.

Forecasting

Because forecasting is used to predict a range of values, as opposed to a limited set of classes, there are different evaluation metrics to consider.

After building a forecasting model, such as for cost modeling, you'll see an RMSE value and a field called "usually within."

RMSE stands for Root Mean Square Error, which is the standard deviation of the residuals (prediction errors). The "usually within" field provides values that are simpler to understand in context, such as a cost model that's "usually within" $40 of the actual value.

Deploy a model and make predictions

VentureBeat reports that 87% machine learning models never make it into production. This is affirmed by a separate study indicating that just 14.6% of firms have deployed AI capabilities in production.

We can't blame them. AI is a difficult task, and many companies try to reinvent the wheel by building their own data pipelines, model infrastructure, and more. At the same time, a McKinsey survey found that just 8% of respondents engaged in effective scaling practices. What this means is that many firms are building models, but are unable to deploy them, particularly at scale.

With Akkio, businesses can effortlessly deploy models at scale in a range of environments. More technical users can use our API to serve predictions in practically any setting, while business users can deploy predictions directly in Salesforce, Snowflake, Google Sheets, and thousands of other apps with the power of Zapier.

The term API is short for "application programming interface," and it's a way for software to talk to other software. APIs are often used in cloud computing and IoT applications to connect systems, services, and devices.

By querying Akkio's API endpoints, businesses can send data to any model and get a prediction back in the form of a JSON data structure.

For context, data structures refer to the way data is organized in a computer program. Data structures are built on two concepts: data types and data manipulation. Data types define the type of data in the structure, such as number, word, or image. Data manipulation defines how data is organized in the structure, such as linear, hierarchical, or tree.

Models can even be deployed via web app to instantly get a URL to share with others. When you hit “Deploy” for a web app, you’ll also get an iFrame embed (an inline frame), which is an HTML tag that can be embedded in any site.

Users who deploy models can take advantage of cloud storage that scales to accommodate unlimited data uploads. AI is the next growth engine for cloud storage, with a massive annual growth rate.

Further, these cloud servers are home to huge Graphical Processing Unit (GPU) clusters. AI algorithms that require a lot of mathematical calculations, such as neural networks, are well suited to GPU processing, such that cloud servers enable unlimited scalability of model predictions.

Continuous Learning (what it is and why it matters)

The importance of continuous learning in machine learning cannot be overstated. Continuous learning is the process of improving a system’s performance by updating the system as new data becomes available. Continuous learning is the key to creating machine learning models that will be used years down the road.

The process of updating a system with new data, or “learning”, is something that is done by people all the time. Continuous learning seeks to replicate this process in a machine. The key to building robust models that continue to be valuable in the future is to learn from new information as it becomes available. This would allow the machine to adjust its behavior accordingly when responding to new information, just like humans do.

The more data a machine has, the more effective it will be at responding to new information. The extent to which continuous learning is applied will help determine how intelligent the system is and how well it responds to new situations.

ML Operations

Machine Learning Operations (MLOps) is the compendium of services and tools that an organization uses to help train and deploy machine learning models.

MLOps services help businesses and developers to get started with AI, with service offerings that include data preparation, model training, hyper-parameter tuning, model deployment, and ongoing monitoring and maintenance. Organizations with a large training pipeline need MLOps to efficiently scale training and production operations.

These services allow developers to tap into the power of AI without having to invest as much in the infrastructure and expertise that are required to build AI systems.

With Akkio, machine learning operations are standardized, streamlined, and automated in the background, allowing non-technical users to have access to the same caliber of features as industry experts.

Data Preparation

To recap, data preparation is the process of transforming raw data into a format that is appropriate for modeling, which makes it a key component of machine learning operations. This process typically includes splitting the data into parts for training and validation, and normalizing the data.

This means randomly splitting the data into a set of two subsets, known as “training data” and “testing data” (this is called stratified sampling). The first subset is then trained to try and find patterns in the data, but the model doesn’t know what’s coming next. The second subset is used as new input the AI has never seen before, which helps better predict outcomes.

That way, when you create predictions on new inputs using this model, they’re more accurate, because you’re using examples that have not already been seen by the model.

Data preparation can also include normalizing values within one column so that each value falls between 0 and 1 or belongs to a particular range of values (a process known as binning).

For example, if someone were providing demographic information about people who visit their website and are able to purchase goods online, it would be helpful to split them into male or female; under 18 or over 18 years old; and so on, in order to classify their behavior while browsing based on these groupings.

Model Training

The training phase is where machine learning models are generated out of algorithms. The algorithm may determine which features of the data are most predictive for the desired outcome. This phase can be divided into several sub-steps, including feature selection, model training, and hyperparameter optimization.

The goal of feature selection is to find a subset of features that still captures variability in the data, while excluding those features that are irrelevant or have a weak correlation with the desired outcome.

Machine learning algorithms are supported by inferential statistics to “train” the model, such that it is able to make “inferences” about new data.

Machine learning will often operate via a feedback loop whereby input data starts with an empty algorithm, which then finds patterns in that data over the course of multiple iterations. That information is fed back into the algorithm which modifies its parameters and goes through another iteration for refinement, until the optimal model is found.

Finally, hyperparameter optimization determines what set of hyperparameter settings should be used based on some criteria, such as cost or computational efficiency. Factors to consider when evaluating model hyperparameter tunings can include:

- Accuracy vs speed tradeoff
- The degree of robustness against overfitting and underfitting due to a large number of tunable parameters vs accuracy tradeoff

Model Deployment

The process of deploying an AI model is often the most difficult step of MLOps, which explains why so many AI models are built, but not deployed.

The process of deploying an AI model is often the most difficult step of MLOps, which explains why so many AI models are built, but not deployed.

There are a number of different considerations to plan for, including: How will data be queried? What product or service will the AI model be embedded into? How do we ensure that all pieces of the model will continue to work together as expected over time?

These are just some of many questions which must be addressed before deployment. With Akkio, teams can deploy models without having to worry about these considerations, and can select their deployment environment in clicks.

Nowadays, there are many creative ways to deploy AI. For instance, you can deploy models on mobile phones with limited bandwidth, or even offline-capable AI servers. Offline AI is a model deployment option that can be used to serve predictions locally, or "at the edge," for use-cases like smart CCTVs that might be in a wireless dead zone, or even AI-powered medical diagnostic apps that deal with sensitive health data.

Summary

Building and deploying any type of AI model can seem daunting, but with no-code AI tools like Akkio, it's truly effortless.

As long as teams have the data, which can come from tools like Salesforce, Snowflake, or even just a Google Sheets file, they can effortlessly train and deploy intelligent models, for everything from churn prediction to sales funnel optimization.