



Machine Learning for Marketers

A COMPREHENSIVE GUIDE TO MACHINE LEARNING

CONTENTS

pg 3	Introduction
pg 4	CH. 1 The Basics of Machine Learning
pg 9	CH. 2 Supervised vs Unsupervised Learning and Other Essential Jargon
pg 13	CH. 3 What Marketers can Accomplish with Machine Learning
pg 18	CH. 4 Successful Machine Learning Use Cases
pg 26	CH. 5 How Machine Learning Guides SEO
pg 30	CH. 6 Chatbots: The Machine Learning you are Already Interacting with
pg 36	CH. 7 How to Set Up a Chatbot
pg 45	CH. 8 How Marketers Can Get Started with Machine Learning
pg 58	CH. 9 Most Effective Machine Learning Models
pg 65	CH. 10 How to Deploy Models Online
pg 72	CH. 11 How Data Scientists Take Modeling to the Next Level
pg 79	CH. 12 Common Problems with Machine Learning
pg 84	CH. 13 Machine Learning Quick Start

INTRODUCTION

Machine learning is a term thrown around in technology circles with an ever-increasing intensity. Major technology companies have attached themselves to this buzzword to receive capital investments, and every major technology company is pushing its even shinier parent artificial intelligence (AI).

The reality is that Machine Learning as a concept is as old as computing itself. As early as 1950, Alan Turing was asking the question, “Can computers think?” In 1969, Arthur Samuel helped define machine learning specifically by stating, “[Machine Learning is the] field of study that gives computers the ability to learn without being explicitly programmed.”

While the concept has been around for more than half a century, we have finally reached a point in technological advancement where hardware and software can actually help developers match their aspirations with tangible reality. This development has led to not only the rise of machine learning and AI advancements, but, more importantly, also advancements inexpensive enough for anyone to use.

Why Did We Create This Guide?

While the topic of machine learning and AI has been exhaustively covered in the technology space, a singular comprehensive guide has not been created in the marketing space on the topic, including how it affects marketers and their work. This space is so thick with technology-based terminology, but not every marketer has the chops to venture into the space with confidence. With many products coming to market, iPullRank believes that preparing marketers to tackle the landscape armed with a solid foundation is important.

Who Is This Guide For?

Since machine learning touches an ever-increasing number of industries, we'll also touch on several different ways that machine learning is impacting people in many professions. Most data scientists use R and Python for machine learning, but have you met a marketer these

days that only lives and breathes data science? We created this guide for the marketers among us whom we know and love by giving them simpler tools that don't require coding for machine learning.

We'll briefly touch on other spaces to round out marketers' understanding of the complex topic we're tackling. We'll also look at how machine learning can help marketers through examination of use cases, plus we'll dive into martech, a field that's increasingly including machine-learning concepts.

How Will This Guide Help?

Since we want to create the most comprehensive resource on machine learning for marketers, this guide will be far more than explanatory. We'll be looking at relevant use cases related to machine learning and delving into practical use of machine learning so that you can begin to use the technologies we'll discuss after you read this guide.

We'll also discuss essential jargon that you'll need to know about machine learning and how machine learning guides SEO specifically, plus we'll delve into the topic of chatbots (just what do they do, anyway?).

Finally, we'll help marketers actually use machine learning, but focus on the common problems beginners face. Along the way, you'll get a look at specific tools and platforms you can use more effectively.

Machine learning is a vital tool for marketers to add to their knowledge base and future-proof their skill sets. As the technology behind it continues to develop, machine learning won't be something you read about in tech articles; it'll be essential to organizations of all sizes.



○ CHAPTER 1

The Basics of Machine Learning

By 2020, the digital universe will be 40,000 exabytes, or 40tn gigabytes, in comprehensive size. In contrast, the human brain can hold only 1 million gigabytes of memory. Too much data exists for humans to parse, analyze, and understand. Here is where machine learning is finding its value: The raw amount and constant growth of data creates a need for methods to make sense of that data overload in ways that can impact an array of professions and lifestyles.

Although it has many uses, machine learning usually gets deployed to solve problems by finding patterns in data we can't see ourselves. Computers give us the power to unearth concepts that are either too complex for humans or would take us longer to than we'd like to practically use them as a solution.

The Basic Machine Learning Model

The first step in machine learning is identifying the rules. The automation, or machine part, comes secondary. Rules are essentially the logic upon which we build the automation.

The first step in rule creation is finding the basic breakdown of the data you want to learn about. Think of this area as the labels you give your data in an Excel sheet or database.

Google called these labels **“Parameters: the signals or factors used by the model to form its decisions”** during a Machine Learning 101 event it held in 2015. A good example here would be working with stock prices to see how different variables can affect the market. In our case, the Parameters would be the stock price, the dates, and the company.

Next, identify the positive and negative results your automation looks to unearth. Essentially, think of this idea in terms of programmatic words such as “True” and “False.” You need to essentially “teach” your program how you, and it, should evaluate your data. Google calls this teaching the **“Model: the system that makes predictions or identifications.”**

Based on our example we used for the “Parameters,” let's look at how a basic setup of the “model” would look.

- If we want to see how different variables would affect stock prices, a human being would need to assign the logic of dates and prices with the variables that affected them, such as the upticks in stocks post-war and in conflicts.
- Once you create the basic logic, you take that logic and your data parameters begin to grow your data set you intend to use in the learning stage. At this point in the process you may find your data wanting, and this is the reason to not begin the process data first.
- From here, you run the data through algorithms and tools to solve the logic created. Google calls this process the **“Learner: the system that adjusts the parameters — and in turn the model — by looking at differences in predictions versus actual outcome.”** In our stock example, our learner would be looking for what variables could have possible impacts on the stocks we're looking to buy and give us predictions about whether the data suggests that the purchase is a short- or long-term buy.
- Gradient learning, or gradient decline, are an important part of the process from here. We're talking about small adjustments — not large leaps — that the computer makes over time until it gets the results correct. But you have to watch out for anomalies in the data; they can have huge impacts on the results.

For marketers, we can find a clear use case we can talk about for this basic description of machine

The background is a solid light blue color. It features a series of concentric circles and radial lines. A solid circle is centered in the upper half, with a vertical line extending downwards from its center. Another solid circle is in the lower right quadrant, with a line extending upwards and to the left towards the center. Dotted concentric circles are also present, creating a layered, architectural feel.

THE BASICS OF MACHINE LEARNING

Rules are
essentially
the logic
upon which
we build the
automation.

learning: Google. What would our life be like without it?

Even in its early form, Google took indexed webpages and unstructured data points around them and arranged them based on logic created using the original PageRank. All of this arrangement happened because of tools used to create a result: a search engine results page (SERP).

Marketers have been dealing with machine learning in one form or another for some time now. Facebook feeds, Twitter trends, and algorithmic ad-buying platforms all use a form of machine learning to make decisions easier.

Supervised Versus Unsupervised Machine Learning

One of the largest fallacies with machine learning is that it'll replace the need for humans. But didn't we just show you above how humans work among several levels of the process? This idea goes beyond basic data scientists and engineers, extending to people who can shape the problems that machine learning will solve, extract the results from the learning process, and apply those results in a meaningful way. As we've seen with Google quality raters, humans must also qualify the results and help refine the logic used for learning.

Machine learning is actually a method for human capital enrichment. It can super-charge the results achieved by marketers and expand the scope of what we even consider positive results and returns.

We can further break apart machine learning into two parts: supervised and unsupervised learning.

- With supervised learning, you deal with labeled data, and you try to optimize your machine learning algorithm to produce the single, correct output for each input.
- With unsupervised learning, you deal with unlabeled data, so you don't know what the output will be.

Instead of training your algorithm to return a correct output based on a specific input, you're training it to look for structure and patterns in the entire data set.

Different Variants of Machine Learning

Are you still with us so far? Good, because now we're getting to some cool stuff.

Artificial Intelligence, Deep Learning, and Natural Language Processing: They're shock-and-awe words, but what do they mean? Well, they're related concepts that have perhaps larger implications on human capital replacement and interaction.

- **Artificial Intelligence (AI):** We can look at this concept as a computer doing something that humans need intelligence to do. Historically, the Turing test was used to determine whether a computer has a level of intelligence equal to that of a human. The Turing test invites a participant to exchange messages, in real time, with an unseen party. In some cases, that unseen party is another human, but in other cases, it's a computer. If the participant is unable to distinguish the computer from the human, we say that the computer has passed the Turing test and is considered intelligent.

However, deeper concepts guide AI development today than merely parroting of human language via computers. AI research looks to develop technology that takes action based on learned patterns. You can break down AI into the smaller segments of deep learning — sometimes called neural networks — and natural language processing.

- **Deep learning** uses the human brain as its design model. It layers brain-like neurons created from levels of machine learning. Each level does its learning and produces

results that get passed onto the next network that takes on another task with that data. This process replicates itself so that the program can look at one set of data from an unlimited number of viewpoints and create an equally unlimited number of solutions.

Google DeepMind's AlphaGo program — it made headlines when it beat the world's top-ranked players at the ancient board game Go — is an example of deep learning in action. The complexities of the game Go means that a computer can't simply brute force patterns as you can with a game like chess. The computer must learn from patterns and use intuition to make choices. This level of operation isn't something basic machine learning around a base data set can do; however, deep learning allows for the layered neurons to rework the data in unlimited ways by looking at every possible solution.

Deep learning is mostly unsupervised and aims to avoid the need for human intervention. At its core, it looks to learn by watching. When you think about the AlphaGo use case, it all starts to make sense.

- **Natural Language Processing (NLP):** Here's where computers use machine learning to communicate in human language. Search engines and grammar checkers are both examples of NLP. This type of technology is what people most often associate with the Turing test. A quality NLP looks to pass the basics of the Turing test, even though some developers would argue about whether their applications really aim to "trick" users into thinking their applications are humans. Think about it this way: No one believes Amazon's Alexa is a real person answering questions despite its use of NLP at its core.

Put It All Together

In its current and expanding variation, machine learning is something that is often seen as a confusing topic. However, as we've just described, it breaks neatly into some basic concepts.

- Machine Learning has a clean model for collecting **Parameters of data** that it feeds into a **human-created Model**, which the **machine Learner** then uses to create or find a solution. The basic example we used was the original Google PageRank model for creating SERPs.
- Machine Learning can be **unsupervised**, often associated with the fields of AI, or **supervised**, where humans must create the Models and test quality for the findings of the Learners.
- Further, advanced Machine Learning, — often identified as **AI** — breaks into a few subfields of its own, the most notable of these are **Deep Learning** and **NLP**. Deep Learning uses layered machine learning in an unsupervised way to approach data from different angles and learn as it moves from layer to layer. NLP uses machine learning to communicate in languages humans use every day.

We'll take a deeper look into several of these topics as we move into decoding the industry jargon associated with machine learning and its variants.



○ CHAPTER 2

Supervised vs Unsupervised Learning and Other Essential Jargon

Diving deeper into the topics surrounding machine learning, we're confronted with a copious amount of jargon. It helps our journey to understand how professionals in the space discuss the topics so that we can become familiar with the terms we'll run into as we dive deeper into machine learning.

Our goal will be providing an understanding of the various topics without getting too deep into the technical details. We know your time is important, so we're making sure that the time spent learning the language of machine learning will pay off as we go down the path of utilization and use cases.

Revisiting Supervised and Unsupervised Machine Learning

We grazed past the concept of supervised and unsupervised learning in Chapter 1; however, these topics are important, and they deserve a more in-depth study.

Supervised Machine Learning

As previously discussed, supervised machine learning involves human interaction elements to manage the machine learning process. Supervised machine learning makes up most of the machine learning in use.

The easiest way to understand supervised machine learning is to think of it involving an input variable (x) and an output variable (y). You use an algorithm to learn a mapping function that connects the input to the output. In this scenario, humans are providing the input, the desired output, and the algorithm.

Let's look at supervised learning in terms of two types of problems:

Classification – Classification problems use categories as an output variable. Example categories would be demographic data such as sex or marital status. A common model for these types of problems are support vector machines. Despite their odd name, support vector machines are a way we describe a linear decision boundary between classes that maximizes the width of the boundary itself.

Regression – Regression problems are where the output variables are a real number. A common format of these types of problems are linear progressions. Linear regression models determine the impact of a number of independent variables on a dependent variable (such as sales) by seeking a “best fit” that minimizes squared error. Other regression models combine linear models for more complex scenarios.

One of two basic types of machine learning models uses labeled data to build models that make predictions. “Labeled” means that the data represents something in the past for which the outcome is known (for example, an item purchased).

Ensemble Methods – These learning algorithms construct a set of classifiers and then classify new data points by taking a weighted vote of their predictions (the original ensemble method was known as the Bayesian average). This mathematical concept estimates the mean of a population using outside information, including pre-existing beliefs, much like what you'd do to find a weighted average. Ensemble methods reduce variance in individual models by combining a number of them together and averaging predictions.

Logistic Regression – This regression model comes into play when the dependent variable is categorical. Given this categorical dependent variable, the logistic regression gets used as a classifier. This model maps variables between 1-0 (such as true or false and pass or fail). This two-outcome model is also called binomial or binary logistic regression.

Unsupervised Machine Learning

As opposed to supervised learning, unsupervised learning involves only entering data for (x) . In this model, a correct answer doesn't exist, and a "teacher" is not needed for the "learner."

You'll find two types of unsupervised machine learning: clustering and association.

Clustering – This type describes techniques that attempt to divide data into groups clustered close together. An example of clustering is grouping customers by their purchasing behavior.

Association – This type describes techniques that create rules that explore the connections among data. An example is helpful here: We might say that people who buy X product also often buy Y product.

Semi-supervised Machine Learning

But wait ... you'll also find a third type of machine learning technique, semi-supervised machine Learning. Think of this type as a hybrid between the two previously mentioned models. In most cases, this type of learning happens when you've got a large data set for (x) , but only some of (y) is definitive and capable of being taught.

Semi-supervised machine learning can be used with regression and classification models, but you can also use them to create predictions.

Classifiers

Decision trees build a series of branches from a root node, splitting nodes into branches based on the "purity" of the resulting branches. You use decision trees to classify instances: One starts at the root of the tree. By taking appropriate branches according to the attribute or question asked at each branch node, one eventually comes to a leaf node. The label on that leaf node is the class for that instance.

This modeling is the most intuitive type of modeling. You've likely used some version of it in your school or professional life.

Backed-up Error Estimate – In order to prune decision tree and keep the "purity" of each branch of the tree, you must decide whether an estimated error in classification is greater if the branch is present or pruned. A system to measure these issues takes the previously computed estimated errors associated with the branch nodes, multiplies them by the estimated frequencies that the current branch will classify data to each child node, and sums the resulting products. Training data gets used to estimate instances that are classified as belonging to each child node. This sum is the backed-up error estimate for the branch node.

Naive Bayes – Naive Bayes classifiers are based on applying Bayes' theorem with naive independence assumptions between the features. Bayesian inference focuses on the likelihood that something is true given that something else is true. For example, if you're given the height and weight of an individual and are asked to determine whether that person is male or female, naive Bayes can help you to make assumptions based on the data.

Reinforced Learning

As discussed in the previous chapter, reinforced learning is a machine learning evolution that involves neural network development. Reinforced learning combines multiple layers of networks into complex models that are useful for building intelligent models.

Neural Networks – This collection of artificial neurons (or machine learning and algorithmic layer) gets linked by directed weighted connections. Each neural network layer feeds the next. The neural network levels each have input units clamped to desired values.

Clamping – This action takes place when a layer, also called a “neuron,” in a neural network has its value forcibly set and fixed to some externally provided value, otherwise called clamping.

Activation Function – This function describes the output behavior of a neuron. Most networks get designed around the weighted sum of the inputs.

Asynchronous vs. Synchronous – Some neural networks have layers of computations that “fire” at the same time, with their nodes sharing a common clock. These networks are synchronous.

Other networks have levels that fire at different times. Before you stop us, let’s clarify something for you: We’re not saying that there’s not a pattern to how the levels handle input and output data; we’re only saying that the levels aren’t firing in a precisely timed way.

Mathematical Concepts

Dimensionality Reduction – This concept uses linear algebra to find correlations across data sets.

Principal Component Analysis (PCA) – This process identifies uncorrelated variables called principal components from a large set of data. The goal of principal component analysis is to explain the maximum amount of variance with the fewest number of principal components. This process is often used in utilities that work with large data sets.

Singular Value Decomposition (SVD) – This technique combines information from several vectors and forms basis vectors that can explain most of the variances in the data.

Graph Analysis – This process involves using “edges” connected to other numerical data points to analyze networks. The data points are known as nodes, and the edges are the ways in which they get connected. Facebook’s EdgeRank, which was replaced by a more advanced machine learning algorithm, got its name from graph theory.

Similarity Measures – Sometimes known as a similarity function, a similarity measure quantifies the similarity between two objects. Cosine similarity is a commonly used similarity measure. This measurement is used in information retrieval to score the similarity between documents. Clustering uses similarity measures to determine the distance between data points. Shorter distances are equivalent to greater similarity.



CHAPTER 3

What Marketers Can Accomplish with Machine Learning

Now that we know the language of machine learning, we're ready to look at specifically what marketers can do using machine learning. The ad tech space is full of companies promising the next silver bullet for marketers. Armed with your new knowledge of machine learning and related concepts, we can begin to look past the veil toward what makes these tools, process, and marketing services tick.

Marketing Automation

It's highly unlikely that if you're reading this guide you've not worked directly with the concept of marketing automation. It's even highly likely that you've played around with one of the industry's leading marketing automation platforms such as Marketo or HubSpot. Even niche tool providers like MailChimp have created automation in their platforms to help marketers scale their efforts.

These automations often help marketers and sales teams streamline and manage tasks that human hands once did. Drip emails are a great example. You can now build an email list, generate a number of templates, and the system will email your recipients at the time and on the conditions you instruct it to.

While these automations are highly valuable to marketers, the next iteration for these systems will be layering in machine learning.

Using the above example of email drips, software providers like Mautic are already offering mail automation that relies on a logic tree. If your recipient Z takes action X, then it sends email Y. This supervised learning system (remember that term from Chapter 2?) is a basic one.

The next evolution in such a system would come from Mautic learning how long it takes for recipient Z to respond to emails and instructs your sales team on the best time to follow up an email with a call based on an unopened email. Going even further, the system helps you to pick the best adjectives and verbs for your subject lines based on previous subject lines and open rates. You've likely seen or reviewed tools with these types of features, since they're becoming more available and can greatly impact the value of human capital working on marketing and sales initiatives.

Sending Frequency Optimization

Sending frequency optimization can also have a substantial impact on both your standard email marketing initiatives and your drip campaigns.

Machine learning can help you answer the following questions:

1. How often should you pay attention to specific recipients and segment new marketing messages?
2. How often should you follow up with leads?
3. What days are most effective for follow-ups and new marketing messages alike?

Before machine learning, marketers would leave frequency optimization up to testing and ROI analysis; however, frequency optimization was all but a measurement on full lists in most cases. Machine learning allows marketers to carve lists into precise segments and to neatly personalize sending frequencies for individual recipients.

Content Marketing

How much time are you spending on administrative tasks, such as asset tagging, versus content creation? Tagging assets with relevant keywords is essential to making them searchable, but it's also a tedious, time-consuming task that most marketers would rather avoid.

Machine-learning technology can smartly include the most valuable or least expensive keywords in copy. This technology can associate similar or related assets, making it easier to combine relevant copy, images, and videos for target audiences. If your audience has consumed one bit of content, then it can smartly recommend what to offer next for the highest conversions. Machine-learning technology can also help predict what content will lead to the behaviors — sharing or engaging with content, increased sales, and improved customer loyalties — you're trying to gain from customers. Adobe's Smart Tag technology is available now to automate metadata insertion so that you can get better search results while slashing the amount of time you spend on this task.

Each marketing channel presents a special set of requirements for the size and resolution of marketing assets. When a new platform emerges — or if you decide to add a new channel — it could require the time and cost of redesigning existing assets. For example, if you have a piece of content delivered to a web channel or blog, machine learning can smartly crop that content for a mobile channel or reduce the copy in smart ways. With machine learning, you can shorten visual content and videos to optimize experiences for different channels based on the patterns by which people are consuming them.

Machine learning will either offer recommendations — or provide a first draft of the new content — that can then help accelerate the pace by which you get those different pieces of copy, creative graphics, or videos published to various channels and distributed to the selected audiences.

You don't want to have to create massive volumes of content and hope that only some of it will be effective. What's important is being able to create the right content that's effective in your channels, learn from that content creation, and then develop more content based on those insights as you expand from that point.

Machine learning can give you the intelligence needed to quickly determine what's working as well as recommend what's needed to amplify your strategies that might better connect with your audience. The learning part of machine learning means that, over time, the machine becomes smarter. We're still in the early stages with this knowledge evolution, but machines could potentially learn so quickly that you could remix, reuse, and adapt content almost instantaneously, test the content you've created, and learn whether what you've created will be an improvement over your previous campaign or whether you need a different approach.

Ad Platforms

One of the first platforms to incorporate machine learning into systems and processes that marketers use every day are media-buying software sets. As an example, Google's quality score helps determine which ads are most relevant to searchers and the cost per click of ads by using factors such as click-through rate, ad copy relevance, and landing page quality.

Programmatic Display

With the growth of mobile internet consumption, banner ads and other display advertising have had to undergo a major change in order to retain their value. Further, with the rise of better and better

technology, marketers are looking for ways to segment their message in a more precise manner and not broadcast to a large audience in hopes of motivating a few people.

Programmatic advertising allows marketers to take matters one step further by measuring and pivoting on advertising strategies in near real time. Where big data has allowed for segmentation, programmatic advertising has allowed for marketers and advertisers to see how these segments perform on specific ads at specific times on specific devices, for example. This type of advertising also allows for more control over spend and higher ROI on display ads.

AdWords Scripting

Google has developed much great functionality in its system for ad control and information in managing ads. Its resources will help you find new segments that you should be reaching and give you information on underperforming ads, groups, and keywords. However, tasks and processes are available to help an AdWords practitioner in ways that AdWords doesn't offer: AdWords Scripts fills this need.

Adwords Scripts runs within the Google AdWords platform. The option can be found in the campaign navigation under "Bulk Operations > Scripting." These Scripts get written through Javascript, although you can find many premade Scripts, too. These Scripts often work as a form of supervised machine learning in many cases. AdWords advertisers specify the input and the output determines the function. An algorithm connects the input to the output.

Predictive Analytics

Predictive analytics are used in many verticals and professions. Essentially, these analytics are models that help find and exploit patterns in data. These patterns can then be used to analyze risks and opportunities.

In the marketing world, predictive analytics layers on top of our standard analytics data to give us more insight and help shape marketing actions. Previously, we'd look at raw data like "sessions" that would give us insight into our analysis of ROI based on base metrics of lifetime value for a session. Now, we can predict with more precision the exact value of each individual session based on their onsite actions and the sources of their referrals. As with programmatic advertising, we're able to interact with the potential customers represented by those sessions in highly tailored ways, including chatbots, which we'll discuss in more detail in a later chapter.

Customer Churn

Keeping customers is as pivotal to growth as getting new customers. Analytics can help understand behaviors that lead to customer churn and help marketers craft strategies to reverse churn. Predicting customer churn is a valuable piece of this puzzle. Through machine learning, behaviors of past users that are no longer customers can give us a data set of actions to apply to the current customer base to see which customers are at risk of jumping ship on you.

Most predictive churn models have their bases in logistic regression and binary models (aren't you glad you read through all those definitions now?). Micro-segmentation and the use of machine learning can help marketers and sales teams understand when a customer or client may be ready to jump ship and help spark an action that can keep churn numbers low.

Computer Vision

Computer vision is exactly what the term sounds like — it's how machines "see."

Machine learning, especially reinforced learning, has allowed machines to have ever-increasing capabilities related to image recognition. A great everyday example is the facial recognition that Facebook uses when suggesting people to tag in a photo. Facebook's facial recognition technology has "learned"

the faces of users over time as they've been tagged by the endless amount of photos that make their way into the Facebook universe.

Computer vision has practical uses in marketing.

For example, Sentient Aware offers software that lets its users serve customers with products that are visually similar to the products that they choose. This style of product serving could have a significant benefit over the traditional use of tagging, especially when dealing with new customers whose buying habits are not yet known.

Snapchat and Instagram have made visual-based social listening increasingly important. Ditto and GumGum provide social listening tools that can enhance reputation-management efforts. For example, brand marketing folks can receive alerts to tell them when their company's logo appears in a meme or image that might need their attention.

Segment Comparison

Audience segmentation has always been an important part of advertising. Knowing the members of your audience and where they're coming from offers marketers incredibly valuable information. Until the invention of the internet, gaining that data was never done in real time. Now marketers can gain almost real-time access to the demographic-based data of their consumers and create actions that interact with them.

Only a decade ago, marketers rejoiced when they gained access to data such as age, sex, location, and the length of time that users interacted with their messages. Now marketers can create micro-segmentations as well as measure and compare how each segment reacts to different messages.

Google Analytics offers behavioral-based demographic data such as affinity groups for a user. Companies such as Jumpshot offer extremely detailed segmentation, offering the ability to let you know which micro-segments purchase from your competition, at what times, and how they're finding them. Furthermore, these tools can tell you which micro-segments buy a \$1,000 watch and which buy a \$100 watch, which can give you a better analysis of where to spend the dollars in your own marketing and advertising budget.



○ CHAPTER 4

Successful Machine Learning use Cases

Now that we've pushed through both generalities of machine learning, its basic concepts, and how they apply to areas of marketing, it's time to dive into the specifics of how companies are using these processes.

We'll look at a number of examples of how major companies are using machine learning to interact and deliver content to customers in an increasingly targeted and personalized way. We'll also look at how some marketing companies are using this technology to increase their insights and their clients' ROI.

Target Identifies Pregnant Customer Before Family Does

This use case is one that Target's PR department probably didn't cook up. The story went viral and was the first experience that many people outside the marketing and advertising spaces had with the way major retailers were using machine learning to target their customers.

Target assigns all their customers Guest ID numbers tied to their credit card, name, and email address. Further, they use customer rewards products such as their Red Card and Cartwheel to gain even more data points to tie to these Guest IDs.

In this case, Target used the data from women who had signed up for their baby registry. They looked at the historical purchases to see what the data profile for a pregnant customer looked like. Going back to our machine learning terminology, they likely used a variation of a Naive Bayes to create a classification for the group.

Next, Target sent this classified group coupons for baby items. The recipients included a teenage girl whose father was unaware of her pregnancy. The coupon likely wasn't the way the girl wanted her father to find out about her expected child or the way Target wanted the world to find out about its customer targeting. However, this story is an interesting and practical example revealing how retailers are collecting, learning about, and using data points.

Adobe Knows What Your Customers Want and When They Want It

Adobe's Marketing Cloud puts several tools in the hands of marketers that can help them immediately put machine learning to work.

The Adobe Audience Manager allows users to centralize all of their customers' data into a data management platform. From this platform, they can create distinct customer profiles and unified audience segments. Understanding audiences and segments allows Adobe's customers to tailor content specifically for these consumers in their Experience Cloud products. Beyond delivering a targeted message, the tools in Adobe's products allow you to find the segments in your customer base that have the highest yield and focus your effort and spend on those targets, thus maximizing your ROI.

Adobe has also brought about Virtual Analyst, powered by an artificial intelligence (AI) interface with its Adobe Sensei product. Virtual Analyst continually processes data and uses predictive algorithms and machine learning to drill into specifics of your business operations. Virtual Analyst is like the real-life data scientist working with you in your company.

Adobe reports several benefits to customers using Virtual Analyst: increased revenues, major cost savings, mitigated risks, and bug detection.

Facebook Filters Posts and Advertisements for Users

Facebook wanted to understand more about the text content its users were sharing so the social media giant built DeepText. This reinforced learning platform helps users to make sense of the context of content and allows Facebook to filter spam and bring quality content to the surface. Of course, DeepText also has implications on ad segmentation and delivery.

The DeepText system is a deep neural network that uses FBLearn Flow for model training. The trained models go into the FBLearn Predictor platform. The scalable model allows constant model iterations for DeepText.

To understand the scale of DeepText, keep in mind that Facebook has 1.86 billion monthly active users. This user base is worldwide, and thus DeepText must have an understanding of many languages. The level of understanding goes beyond basic Natural Language Processing. Instead, the goal of DeepText is to have an understanding of the context and intent of the content, not simply what the content says only.

DeepText achieves contextual understanding by using a mathematical concept called “word embeddings.” This practice preserves the semantic relationship among various words. With this model, DeepText can see that “bae” (a term of endearment) and girlfriend are in a close space. Word embeddings also allows for terms across languages to have similarities aligned.

One practical use of DeepText can be found in the Messenger application. The Messenger app uses AI to figure out if someone is messaging someone with the intent to get a ride from a taxi somewhere. If Messenger recognizes this request, it offers up the options to “request a ride” from a ride-sharing application.

Future use cases that Facebook has floated include the ability for the system to understand when users want to sell products they’ve posted and offer them tools to help with the transaction. DeepText could also be help for bringing to the surface high-quality comments on posts by celebrities and other large threads.

With it’s new focus on improving the news feed, you can see where Facebook could deploy DeepText to help flag issues such as fake news at scale.

Additionally, Facebook is using AI to create chatbots that talk like humans. To do so, the social networking giant has developed a free software toolkit that people can download to help Facebook compile data, view research, and interact with others participating in the projects. The objective is to get computers to understand human conversations without failing; to do that, individuals can provide computers with real conversations and language used online to teach them. For example, information from Reddit discussions about films can train computers to converse with humans about movies.

Clarifai Identifies People and Objects in Videos for Easier Media Indexing

Clarifai is an image and video recognition company based in New York. The firm has raised \$40 million over two rounds from Menlo and Union Square Ventures to build a system that helps its customers detect near-duplicate images in large uncategorized repositories.

Clarifai’s technology can be used through one of the e-commerce examples we discussed in Chapter 3. The company’s “Apparel” model, which is in beta, claims to be able to recognize over 100 fashion-related concepts including clothing and accessories. Further, the company’s “Logo” model can be used by companies looking for a social listening tool for Snapchat and Instagram.

Clarifai’s “Demographics” model offers an interesting opportunity for marketers as well. The company is able to analyze images and return information on the age, gender, and multicultural appearance for each face detected. The opportunities for advertisers to target specific demographics in an increasingly visual internet landscape, where there is less text content to inform the intent, are arriving at the perfect time.

And just when you thought what Clarifai was doing was cool in itself, the company also allows customers to “train” their own model.

Sailthru Helps Customers Develop Digital Marketing Campaigns With Better ROI

Sailthru is another New York-based company building solutions around machine learning. Their solution focuses around promotional emails. Sailthru's system learns about customers' interests and buying habits and generates a classification based on the information. This classifier tells the system how to craft the message and when to send the message to optimize performance.

One Sailthru customer, The Clymb, saw a 12 percent increase in email revenue within 90 days of launching the Sailthru personalization model.

Sailthru also offers prediction and retention models that allow companies to be able to anticipate customers' actions. After turning on these models, The Clymb saw a 175 percent increase in revenue per thousand emails sent and a 72 percent reduction in customer churn.

Netflix Personalizes Video Recommendations

Netflix is incredibly transparent about how it has used machine learning to optimize its business. The most obvious place that you can see the effects of their use of machine learning is on the Netflix homepage. Netflix personalizes its homepage for every user.

One of the issues Netflix faces with its homepage is the sheer amount of content and products from which it chooses. The homepage has to not only pull content that the user is likely to want to the surface, but it also has to serve that content as a doorway to other content that the user may find interesting.

Netflix uses a series of videos grouped in rows. The company can group these rows by basic metadata such as genre. However, Netflix also creates rows based on personalized information, such as TV shows, that are similar to other shows the user has watched.

This system that Netflix uses is a great example of graph analysis. Netflix is able to examine the connection between various data points and recommend content based on the "edges" between the points. As an example, "The Rock" could be one edge that connects "The Fast and the Furious" movies with "The Scorpion King."

Concepts such as Naive Bayes and logistic regression are likely used to help create profiles and match those profiles with the outputs from the content graph analysis. Naive Bayes would help classify groups of users by their behaviors, and the logistic regression would qualify whether that group should be served each type of content. For example, if a user is watching mostly content focused on preschoolers, Netflix knows not to serve horror movies as a suggestion.

On top of these two types of machine learning, with independent focus points on content and user behavior, Netflix also conducts A/B tests on the layout of the homepage itself. Historically, Netflix had displayed rows such as "Popular on Netflix," but the company realized that this arrangement was not personalized for every user. In our example above of the preschool Netflix user, getting recommendations for the new season of "Orange Is the New Black" because the show is popular doesn't make much sense for this audience. However, showing this user content grouped in rows by their favorite children's entertainment characters would make logical sense.



SUCCESSFUL MACHINE LEARNING USE CASES

Identifying who
is truly leading
the field in
terms of success
in SEO is still
not an exact
science.

iPullRank Predicts Visibility of Sites in Organic Search in Vector Report

Identifying who is truly leading the field in terms of success in SEO is still not an exact science. While you can easily view something such as a content marketing campaign and grade a company on its efforts in this area, SEO by its nature is more like a black box.

As such, iPullRank set out to create a view of the Inc 500 companies and their overall and market-level organic search performance. This study intended to predict a site's performance in SEO from relevant link factors from Ahrefs, cognitiveSEO, Majestic, and Moz.

The data collection involved the following:

1. Scraping the Inc 500 list from the company's site
2. Pooling the 100 winners and 100 losers from Searchmetrics
3. Establishing the URLs from each company
4. Pulling all the available domain-level metrics from Moz, SEMrush, Searchmetrics, Majestic, and Ahrefs.
5. Placing the 700 domains into cognitiveSEO
6. Using cognitiveSEO's Unnatural Links Classifier

Next, iPullRank reviewed 116 features for every domain and used logistic regression, cross validation, random forest, and lasso to analyze the data.

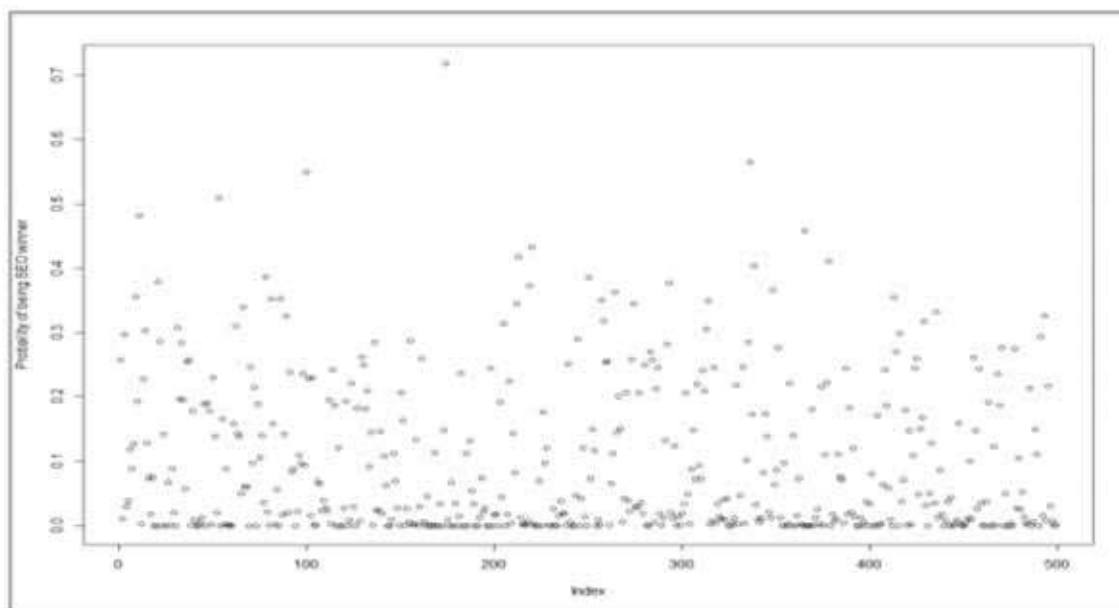
From here, we collected two sets of training data. We used the 100 SEO winners and 100 SEO losers from Searchmetrics for our 2014 and 2015 Inc 500 data pulls, respectively. We then used the training data from 2014 to select the model. With the 2015 training data, we updated coefficients with the chosen model.

The sample size for 2014 training data was relatively small, and we knew that any statistical models would be sensitive to the data points selected as the training data set. We used 5-fold cross validation to choose the model, solve the sample size data, and reduce variance of the model.

We used several machine learning methods to select variables, including random forest and lasso. In the end, we selected eight factors in the final logistic regression model which are believed to influence the overall performance of a webpage. These eight important factors are as follows:

Variable Name	Metric Provider	Description
umrr	Moz	The raw (zero to one, linearly-scaled) MozRank of the target URL
refclass_c	Ahrefs	Number of referring class C networks that link to the target
CitationFlow	Majestic	A score helps to measure the link equity or "power" the website or links carries
TopicalTrustFlow_Value_2	Majestic	The number shows the relative influence of a web page, subdomain or root domain in any given topic or category
Rank	SEMrush	Site rank on Google
Unnatural.Links	CognitiveSEO	Number of unnatural links
OK.Links	CognitiveSEO	Number of OK links
fspsc	Moz	Spam score for the page's subdomain

Before predicting winners and losers for the 2015 Inc 500 companies, we used median imputation to handle missing values. Then we updated the final model coefficients with 2015 training data. The prediction results are plotted below:



We noticed that most companies registered predictions as losers (probability <0.5) based on these plot points. This result follows from the huge difference in the distribution of the 2015 training data set and 2015 Inc 500 company data. Put simply, the SEO winners and losers we used for training data are often more advanced in their SEO strategies and techniques than the standard Inc 500 company. Therefore, we came up with the alternative to grade each company based on the relative performance within Inc 500 companies.

We measured the overall performance in the following two ways:

1. weighted average score of each variable
2. the predicted probability of being a SEO winner

The lower score between the above two methods was used to better identify underperforming companies from an SEO perspective. Overall performance using predicted probability of being a SEO winner gets decided by the following rules:

A	Top 10%
B	Top 10% – 25%
C	Top 25% – 45%
D	Top 45% – 65%
F	Top 65%

We found that 2015 Inc 500 companies had better organic search performance than 2014 Inc 500 companies in terms of spam score, trust metrics, link popularity, visibility, and unnatural links. In general, however, most Inc 500 companies have high grades in spam scores and unnatural links, but have poor link popularity and low trust metrics. This finding shows a general lack of off-site SEO and content marketing strategy by the group as a whole.

Below are the number of companies being assigned to each grade for 2014 and 2015, respectively:

Grade	2014 Inc. 500	2015 Inc. 500
A	34	47
B	63	76
C	54	102
D	127	91
F	222	183



◦ CHAPTER 5

How Machine Learning Guides SEO

Our Vector Report served as a delicious appetizer for the world of machine learning and how it applies to SEO and search. As we've noted, the core of modern search builds upon models based on machine learning. As the field has expanded, both search engines and search practitioners have worked to incorporate more robust machine learning processes into their methodologies and technologies.

Google and Machine Learning

Most of our conversation will center around Google and its expansion of machine learning technologies and where they might be going next.

Ranking Algorithm and Beyond

Remember when we talked about the original PageRank algorithm and how this algorithm can be treated as unsupervised learning? We could make arguments against the early Google algorithm being what we consider machine learning today, but the reality is that the algorithm aimed to make sense of patterns in unstructured data around the web and put those patterns into a coherent output — we know them as SERPs. Detractors from this view will point out that the algorithm is only one part of the overall information retrieval structure and not by definition pure machine learning.

What can't be argued is that **Quality Score** — arguably the economic engine of Google — is a machine learning-based program. This score uses a large amount of signals to predict future outcomes. The largest signal in this mix is click-through rate (CTR) for well-established accounts with history. A myriad of other signals make up the remaining Quality Score mix, but based on the efficiency of Quality Score and its staying power, SEO sleuths could see the writing on the wall that Google would eventually take a “smarter” approach to its overall algorithm.

RankBrain

Enter **RankBrain**. RankBrain is an algorithm-learning AI that Google verified existed in October 2015. RankBrain isn't the algorithm but part of an algorithm, one that Google has stated is the third most important factor today behind links and content.

RankBrain's real power is understanding never-before-seen queries and delivering results based on its understanding of the term. RankBrain groups search queries close to each other in linguistic similarities into “distributed representations.” Put simply, RankBrain is trying to predict what people “mean” in their search queries by looking at their intent.

Google's RankBrain can “self-learn” and form new word associations over time. This ability makes it capable of working faster than a human “training” the program to recognize linguistic similarities.

The principles of RankBrain are similar to the word2vec tool, so while RankBrain is a bit of a closed box outside of what Google has let out, we can get a better idea of its structure from looking at this comparative AI tool.

As an **open-source toolkit**, **word2vec** uses a text corpus to count the distance between words and product vector representations of words and phrases. The closer in distance the vectors are, the more similar in meaning they are.

According to Google, no one way exists to optimize content for RankBrain specifically. Further, the meat and potatoes of RankBrain's effects will be found in the longer tail terms where you're less than likely to be fishing. The main intent for RankBrain is to help with brand-new queries.

However, the one place that RankBrain has the largest effect is in content creation. The idea of keyword density having a major effect on search died long ago, but RankBrain is an overall death knell, putting the focus on comprehensive content that provides immense value to readers. This kind of overwhelmingly informative piece of content on a subject is what we're creating with this guide as an example.

Google Patents and the Future

With RankBrain launched, and the value that Quality Score plays in its advertising system, Google is clearly moving quickly toward innovation in machine learning to keep its market lead in the world of search. The next step for SEO practitioners is to look at what the company's patents tell us about its future intentions about the concept.

Methods and Systems for Classifying Data Using Hierarchical Taxonomy

Filed in 2012, this patent for **classifying data using a hierarchical taxonomy** is a method and system for classifying documents. The use case to which Bill Slawski attributed the model at Google is customer support, a system that helps to classify customer issues and route them to the correct channels. In this use case, Google would group various issues related to each other. For example, payment processing would be a subcategory under which various issues would be classified. From this level of classification, the system would build a hierarchy of importance of the issues (also discussed as documents).

This same concept could be applied to ranking algorithms, and the patent gives us insight into how its ranking module handles ranking algorithms in the use case of customer service:

In some implementations, the ranking module **ranks document clusters according to one or more metrics**. For example, the ranking module may rank the clusters according to the document quantity in each cluster. A cluster with many documents may represent a relatively significant topic, such as a product issue.

Need another example? The ranking module may **rank the clusters according to an estimated time to resolution of an issue represented by the cluster**. Issues represented by a cluster "software update," for example, may typically be resolved faster than issues represented by a cluster "hardware malfunction," a label assigned to the cluster, a number of documents in a cluster, a designated importance of subject matter associated with a cluster, identities of authors of documents in a cluster, or a number of people who viewed documents in a cluster.

One more example for you: A cluster that represents an issue that historically has taken a longer time to resolve may be ranked higher than a cluster representing an issue with a shorter historical time to resolution.

This patent shows that Google has done internal work to classify and rank content based on learned value of documents. Bounce rates and click-through rates (CTR) could take the place of "estimated time to resolution" as data points that quickly make this effective addition to ranking algorithm.

Topics and content that matches what the system perceives to be high-quality content based on past user-based signals could float to the top of the results. In many ways, Google could use this type of document-based understanding as a citation-based replacement to links. Brands and websites get value based on how much internet users talk about them online.

Since the system is constantly “learning” from its generated outputs and adding new “training” from these learning experiences, a reinforced learning system like this one, focused on organic search, would simply need time to learn to be efficient enough to be added to the ranking algorithm.

Searchmetrics Content Experience

Since Google has made machine learning a priority, organic search tool providers must do the same to keep their customers armed with the right technology for the changing market.

Searchmetrics created a tool called “The Searchmetrics Content Experience.” This tool uses reinforced learning to help SEO managers and content creators within an organization collaborate and create better content.

In this tool set, marketers are able to research topics, optimize content, and measure results.

Searchmetrics uses reinforced learning to deliver related topics in its Topic Explorer tool. Searchmetrics uses NLP to create a group of nodes and edges around related terms.

The optimization tools in the tool set deliver help and support to writers through interactive parameters and live content scoring. Writers learn in real time how to create the most effective piece of content possible. These efforts are all based on reinforced learning around 250 billion continually updated data points in the Searchmetrics index.

What Searchmetrics has effectively created is a tool that has SEO managers thinking, collaborating, and creating content around concepts that are similar to what RankBrain is doing to answer questions for new queries.

The Future for SEO

Based on the rise of RankBrain, the feelers we’ve gotten from Google on future machine learning plans and the tools being created in the SEO space makes one point clear: Comprehensive content is future-proof SEO.

While Panda killed off the days of automated content at scale — offering quick wins for SEO managers looking to get ROI without expelling much energy — the reality is that lower quality content production is still happening.

Content is still commoditized, and getting content at scale is difficult and expensive. **Great content creates great links**, which have real impact on today’s algorithm — the number-one factor on search rankings today. Great content also yields value with systems such as RankBrain that are likely to surface quite comprehensive articles for intensely specific queries.

Based on the other concepts we looked at in this chapter, content that garners quality signals from search visitors is also likely to have an increasing impact in the future, especially as Google is able to associate those data points with content, learn the meaning of that corollary data, and apply it to ranking results.



CHAPTER 6

Chatbots: The Machine Learning you are Already Interacting with

Chatbots are probably the most common format of reinforced learning that people are interacting with daily. Since chatbots can pass the Turing test, some people probably don't know that they are interacting with chatbots in some cases.

The most widely used versions of chatbots today are likely the AI assistants from the major tech companies such as Siri from Apple and Alexa from Amazon. While chatbots differ widely in how advanced they are and how they're built, most chatbots have some basic similarities.

Platform

Chatbots are usually hosted within a messaging platform of some type. If you use Slack, the chatbot that comes bundled with your account from the outset is the Slackbot. This bot can help with many Slack-based activities and reporting.

Another example that we already looked at is the way Facebook uses DeepText to interact with its users in the Messenger app.

Language Processing

This element is a vital layer of any chatbot. Human language communication between man and machine is impossible without a Natural Language Processor (NLP).

Essentially, an NLP refers to components that allow chatbots to carry out a conversation that's nearly identical to a person-to-person human conversation. NLPs use deep learning to not only study human input but also generate a human response. According to Chatbots Magazine, newer, smarter chatbots using NLP should be able to carry out the following functions:

- Summarizing large quantities of text into concise explanations
- Responding to open or closed questions ("Is Washington, D.C., the capital of the United States" compared to "Why is Washington, D.C., the capital of the United States?")
- Inferring conference clues (that is, relating an object with words in a sentence)
- Ambiguity (sorting out the context and meanings of words in sentences)
- Morphology (the ability to separate individual words into morphemes, the smallest unit of meaning for words)
- Semantics (the meaning of sentences or words in natural human language)
- Text structure (the usage of punctuation, spacing, and text)
- Sentiment (the emotional "feeling" conveyed by the speech patterning, for example, "happy" or "sad")

Amazon powers Alexa with its own NLP called **Amazon Lex**. Most chatbot builders use third-party NLPs rather than building their own, and Amazon has made Lex available to developers.

Application

This stage is where the machine learning happens. Despite the advanced options we've shared with you, chatbots can run on simple supervised machine learning models. For example, you can train a chatbot to respond to various questions from customers with predetermined responses. As new questions come in from customers, you can classify them within various categories of responses, thus "training" the application. This part of bot development can now find a third-party home at services

such as **Amazon's Machine Learning Service**. When the process being used to manage the application moves into the neural network level of computing, the chatbot moves into the world of AI.

Database

The database is the arrival location for the data points that the application computes. In the example in Chapter 5 with our customer service chatbot, this database can be as simple as the list of questions and responses. Obviously, the more advanced a chatbot is, the more data and the more advanced the machine learning that deals with that data will need to be.

As you can see, chatbots are a key application that connects the value of machine learning with tasks people do every day.

Now that we understand the basic components of a chatbot, let's look at some use cases for the technology.

WeChat Empowers Its Users

WeChat is a social media application brought to life by Chinese company Tencent, the company behind the QQ desktop messaging software. The app has more than 899 million active users spending over an hour a day on the app on average. WeChat is huge in China, but it has yet to break through the social media realm internationally. A special projects team within Tencent created the product.

WeChat runs contrary to Facebook in many ways. Where Facebook launched Messenger to “unbundle” services that users may want to use, WeChat gives you everything you could want in a mobile experience in one application. In other words, WeChat is not only a social media home for many users, but it's also the method for shopping and payment services.

WeChat has built an API for developers to build chatbots on their applications. Today, this API lets users do many interesting activities such as:

- get a ride share
- order food
- buy movie tickets
- order from the nearest Starbucks
- track your daily fitness progress
- book appointments
- pay bills

In many ways, WeChat served as the beta test for chatbots in the social media space for the U.S. market. Major technology companies in the West took notice, and Windows developed a chatbot called Xiaoice. This NLP application from Microsoft has more than 40 million users in China and Japan.

Xiaoice is capable of leading long conversations with distinct personalities. Microsoft mined the internet for human conversations on Chinese social media and used reinforced learning to develop a chatbot that can adapt its phrasing based on the positive or negative cues given by the user.

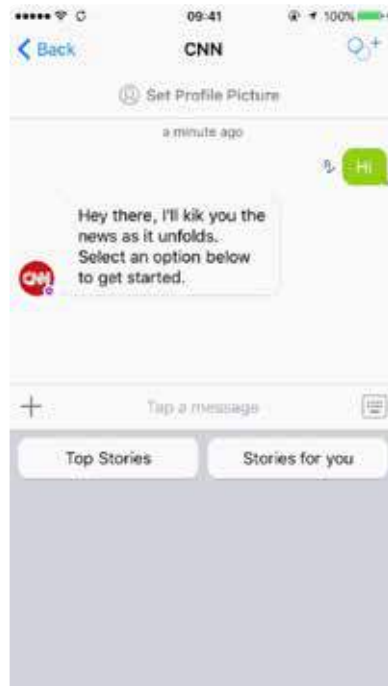


CHATBOTS

Chatbots
are the most
common format
of reinforced
learning that
people are
interacting
with daily.

CNN's Bots Share the Latest News

In 2016, CNN began rolling out a chatbot to interact with younger demographics on social media platforms. The first bots rolled out on **Messenger and Kik**, and they were created to give users the latest news in a conversational format that mirrors their everyday messaging application use.



CNN specifically launched the Kik application with an interactive explainer to help users understand the developing news better. Additionally, CNN launched bots on LINE and for Amazon Echo.

Commenting on the news organization's view on its success, Alex Wellen, CNN's senior vice president and chief product officer, told **The Drum**:

"We've seen the most growth on LINE, where our chatbot publishes a handful of international stories each week; we see high engagement on Kik, where our chatbots invite the audience to experience the news in a 'choose-your-own-adventure' format; and we've observed the most experimentation on Facebook Messenger and Amazon Alexa, where people can ask and receive responses to open-ended questions about the news. CNN defines success metrics for each platform, from reach to engagement to monetization, with the overarching goal to learn, scale up, or wind down the product."

We're still in the early days of chatbots. Companies are still trying to understand how to quantify the value of the interactions they're having on the various applications.

Wellen gave a peek at the possible future for CNN's chatbots: "The next frontier is real-time multimedia chat bots that return intelligent audio or video responses across mobile, IoT (Internet of Things), and connected TVs."

Magic's Chatbot Offers Complete Virtual Assistant Capabilities

Magic is an online virtual assistant service that launched in 2015. This service has created chatbots in several applications, including Slack, Messenger, and through SMS (Short Message Service). We might think about Magic as an example of **conversational commerce**, an important subset of chatbots.

We've already looked at a few examples of conversational commerce with the examples of WeChat and Messenger allowing you to order a ride share. Magic allows you to speak with a virtual assistant to help you with dozens of tasks that an administrative assistant would handle, all through the applications you use every day. These virtual assistants save data points that allow them to iterate the tasks later.

We know you might be thinking, but we're going to tell you anyway: Magic is not a chatbot. While Magic is an interesting example of how companies can enter the conversational commerce space, live human operators manage the tasks in Magic. You won't find any AI or reinforced learning happening as of the date of this writing.

However, with the capabilities of companies to leverage an ever-expanding world of third-party reinforced learning tools, the interactions happening at Magic today could serve as the training for tomorrow's AI-based virtual assistant.

We're still much entrenched in the early days of chatbots. This time period feels like the early days of mobile applications, and developers should be salivating at the untapped opportunities.

While the idea of developing a conversational bot may seem like a cumbersome task, remember that you can start small. A chatbot simply needs to be built on the four components we discussed earlier. A conversational commerce-based chatbot is possible with a human-generated database and basic training over a supervised machine learning model. The tools and infrastructure are becoming more readily available for all developers. With the increasing exposure of developers to these tools, we're likely to see an explosion of chatbots and related environments and economies over the coming years.



CHAPTER 7

How to Set Up a Chatbot

In Chapter 6, we discussed the basics of chatbots and how they are important to the growth of machine learning. Now, we'll dive into the nuts and bolts of how you can set up a chatbot yourself. We briefly touched on the overall structure and components of a chatbot, as well as some of the third-party tools that exist to create chatbots, but we want to dive deeper into the concept and give you a strong idea of exactly how you can put down this guide and get to building your first chatbot.

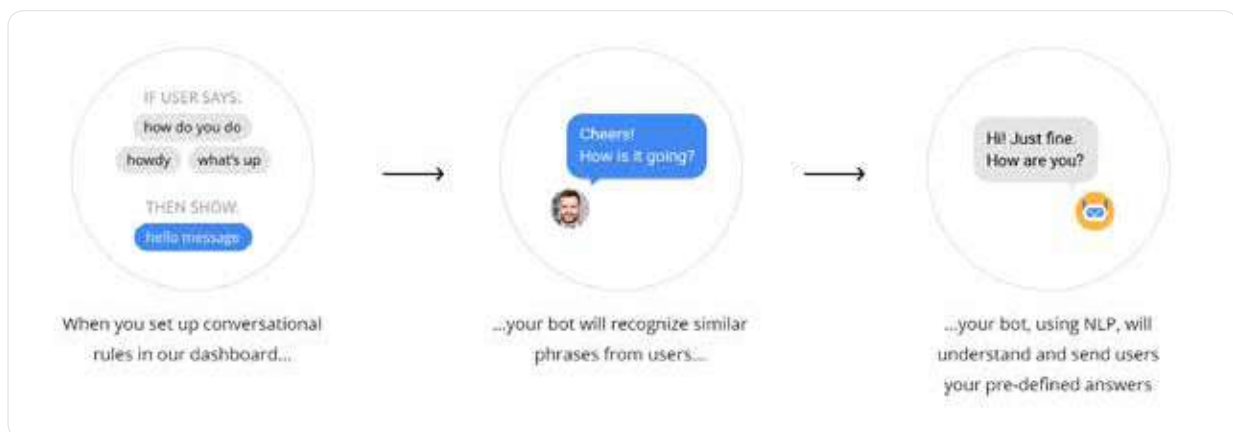
Creating a Facebook Chatbot

Creating a Facebook chatbot from scratch isn't extremely difficult, but the process can be a technical one. Below, we present two versions of this process: one for individuals who don't have a high degree of coding knowledge and one for those readers who are more technically proficient.

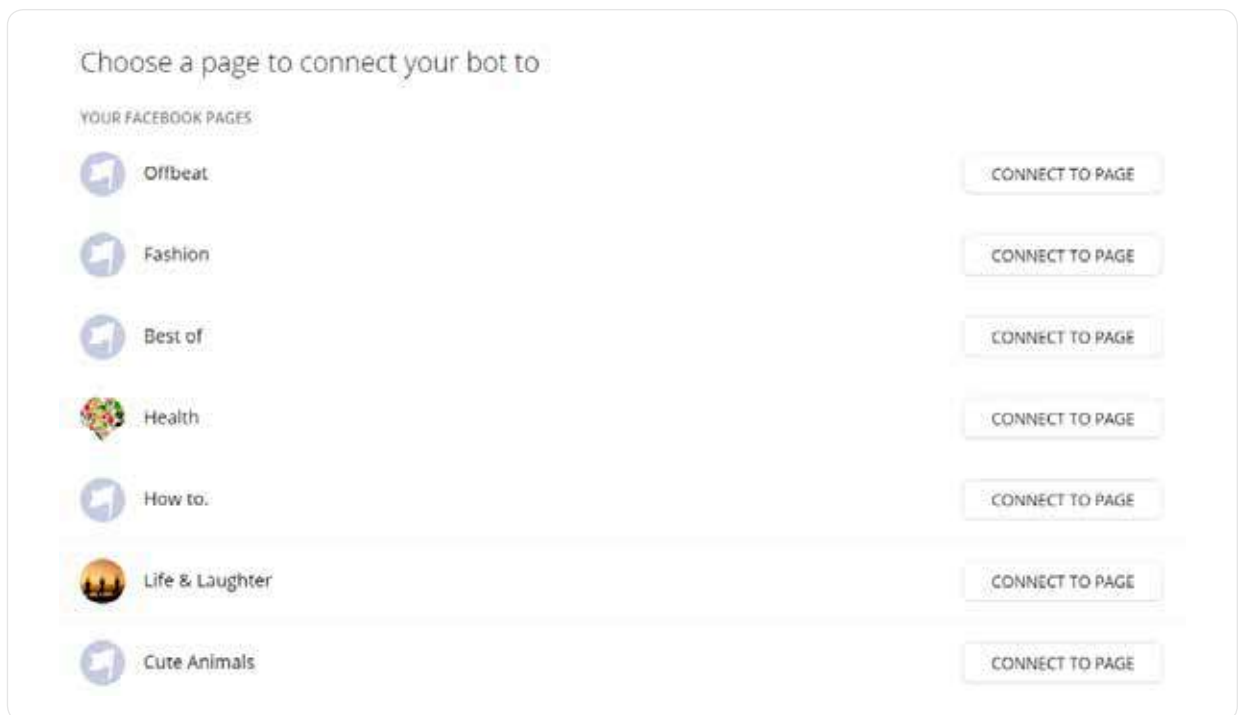
Chatfuel

For those of you who don't have a technical background or access to a developer, Chatfuel may be the answer for you.

Backed by Yandex, Russian entrepreneurs Dmitrii Dumik and Artem Ptashnik founded . Chatfuel in 2015. Chatfuel builds chatbots on the Facebook Messenger platform. Users can train their own bots in the Chatfuel system. Natural Language Processing (NLP) in the system allows training information from the user to be applied to a larger web of related terms and phrases.

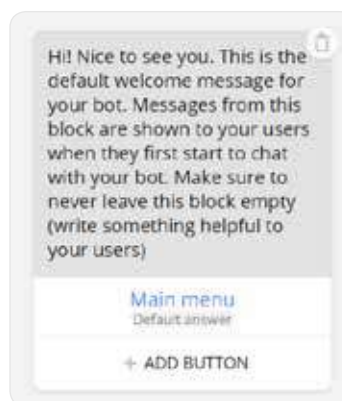


Chatfuel walks you through the same process that developers use as they code their applications. Once you sign up via your Facebook account, you'll choose a page that you want to associate with your application. You may need to create a page if you intend to test your application before launching it.

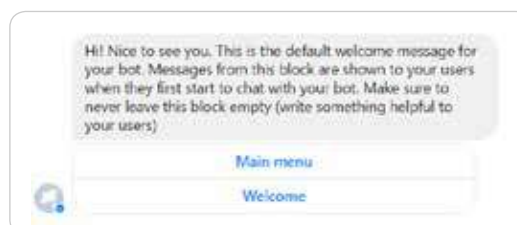


Blocks

The first and most basic setup for the Chatfuel bots is to create and link “blocks.” Blocks are essentially messages sent to users. For example, below is a “welcome” block in the back end of the system:



And what follows is how the block looks in Messenger.



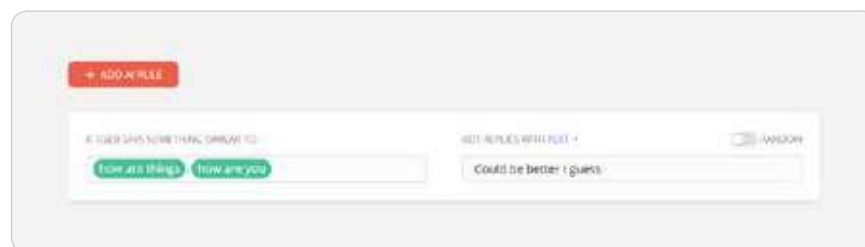
The “link” here is the “Main Menu” button. When you press it, Chatfuel gives you the following:



You'll remember that this choose-your-own-adventure style of logic is similar to some of the functionality CNN built into its Kik-based news bot.

Chatfuel AI

You'll get the most value from using the Chatfuel-integrated AI. This section of Chatfuel is the place where you'll train your chatbot how to handle questions. You can respond with basic text, images, video, and other options.



Payments

Chatfuel offers a payments option that makes conversational commerce an option for bot builders.

Connecting to Facebook

The first step in creating your own Facebook chatbot through development is to create a working endpoint that delivers a 200 response code. Put simply, this step lets you talk directly with Facebook. Hartley Brody does a great job walking you through this step with a Github project to help you every step of the way with code. To execute this process, you'll need Heroku, which is basically a cloud platform that allows companies to essentially become their own app companies by producing, sharing, tracking, and scaling their apps. Heroku offers app developers the advantage of avoiding complicated infrastructure hardware or scaled servers.

Choosing the Right Facebook Page

You'll see that this is the second basic step to create your chatbot through third-party applications as well. You can use a Facebook page you currently have or begin a new page, but think of this page as the identity of your bot. It'll be the group of people you can broadcast your bot to, and it'll be the entity with whom your users will interact. Be careful with your choice and testing.

If you don't already have one, you need to create a Facebook page. The Facebook Page is the “identity” of your bot, including the name and image that appears when someone chats with it inside Facebook Messenger.

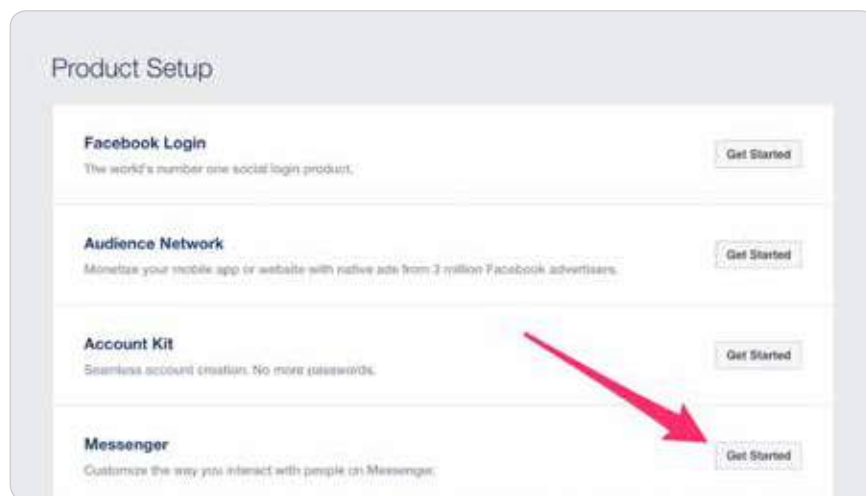
If you're creating a dummy one for your chatbot, it doesn't matter what you name it or how you categorize it. You can skip through most of the setup steps.

In order to communicate with your bot, people will need to go through your page, which we'll look at in a bit.

Create Your Facebook Application

Now you are ready to create a Facebook App for your bot. You'll need to name your bot, categorize it, and give it some other basic info.

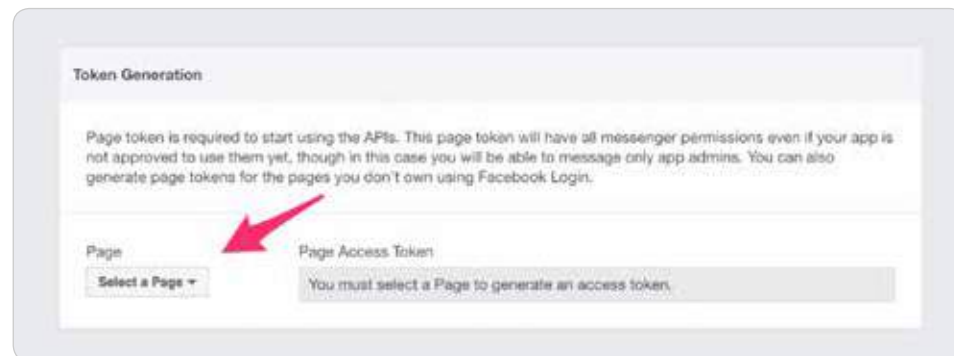
After you've created your App ID, you'll have the option of selecting the product you want when setting up your app. You'll choose Messenger, which serves as the platform for your chatbot.



Set Up the Messenger Application

All of the steps below need to happen in Facebook before you get into the code portion of your chatbot.

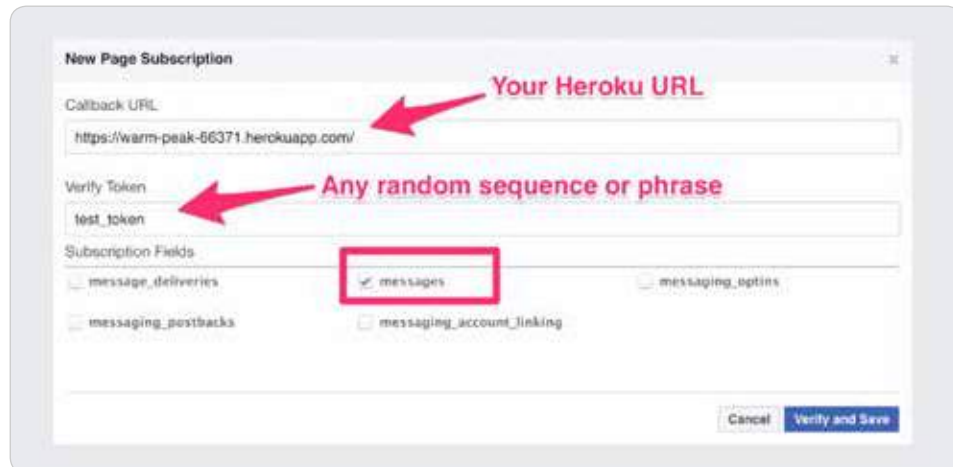
First, create a Page Access Token by clicking through the authentication flow (see below).



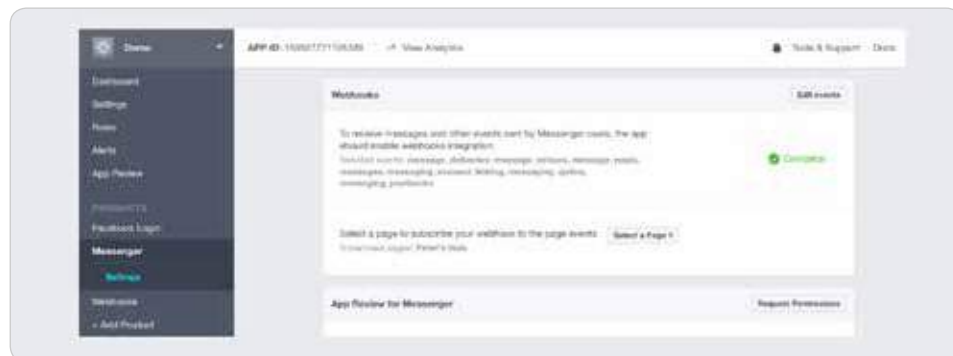
You'll use this token to authenticate your requests when you try to send a message or reply to someone.

You'll then need to create a webhook. Facebook will ask you for a callback URL, verification token, and subscription hooks. Hartley Brody's tutorial and Github repository have more information on the

specifics of these items. The Facebook Developers portal also has vital information on the Messenger Platform that can help you.

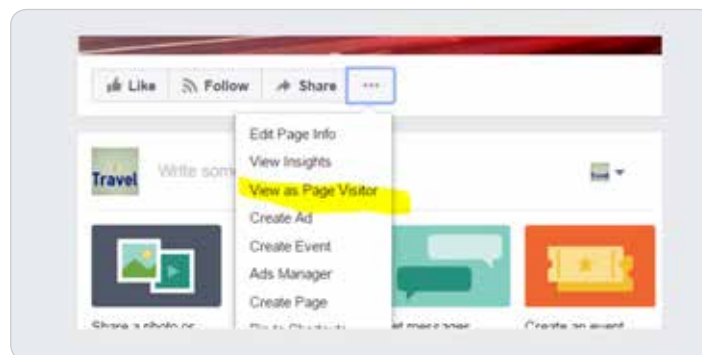


After you've configured your webhook, you'll need to subscribe to the specific page for which you want to receive message notifications.



Get Chatty With Your Bot

Start sending yourself messages through your newly connected application. Go to the Facebook page, and under the options button next to "Share," select "View Page as Visitor."



This selection will allow you to message the page as a visitor.

Most developers use Heroku when building Facebook apps because Heroku allows developers to work with the language that's most natural for them to use when coding.

Beyond Hartley Brody's framework, you'll find a few others worth noting that may help you or your developers with some plug-and-play code to get your chatbot talking.

- Messenger Bot Tutorial
- Charca bootbot
- Facebook Chat bot with Node and Heroku

One commonality you'll notice is the way these chatbots work: They classify anticipated user questions or keywords and match them with responses.

Submit Your App for Review

Before Facebook verifies your application, only you and your page admins can test or launch the chatbot you're creating. Facebook has to review all apps, but chatbots are especially large security risks. The capabilities to spoof human behavior, send malicious links, and use processes to retrieve private data are too real with this technology.

While you're testing your bot, only you and other page admins can message the bot directly. You'll have to go through a review process before your bot is open to the world, ready to chat with anyone.

Facebook seems to be quite thorough in its review process, and with good reason. The code for a messaging bot runs on your own servers and could change at any time, without Facebook knowing it's changing.

As long as your application is a legitimate chatbot set up to enrich the user experience of people who like your page, you shouldn't have any issues.

Botsify

Similar to Chatfuel, Botsify allows you to build a chatbot on Facebook Messenger free. It also offers a paid version of its system to create a chatbot for a website. One of the interesting parts that's special about Botsify is the ability for bot builders to use what it calls "stories."

Botsify Stories

Stories are essentially user stories, but they also serve as the real training data for the bot. The creator builds out an entire possible interaction or series of interactions, with phrase and response alternatives. Within these stories, you're able to connect outside APIs to bring data into the chatbot. These features are the right mix of user friendliness to allow anyone to build a bot, while also allowing for the maximum amount of customization.

Links

Like Chatfuel, Botsify permits template linking similar to the CNN use case. These links allow information to easily be tied to various communications between bot and user.

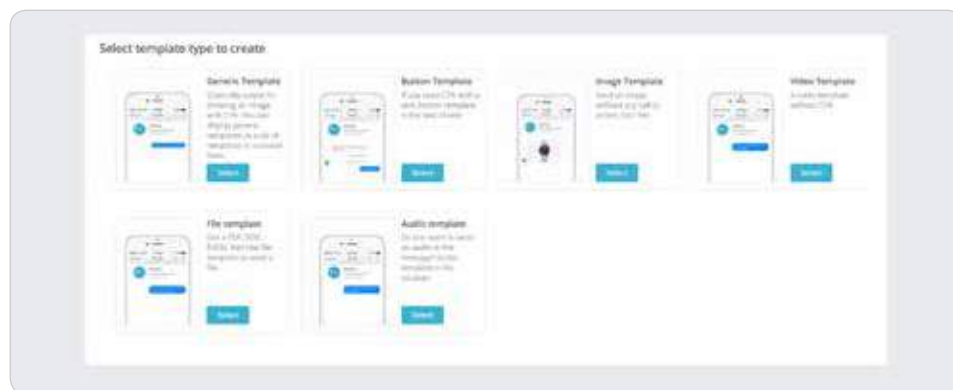
Converse and Learning

The Converse and Learning tools in Botsify bring the AI capabilities of this system to a new level. In this window, you're able to "teach" your bot based on real conversations within the application that it wasn't able to answer.

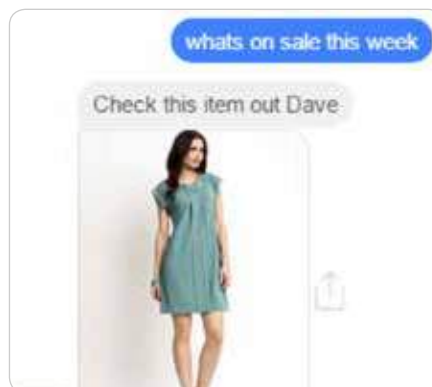


Templates

The templates section of the Botsify application will make marketers swoon. These templates feature various ways to maximize interaction with chatbot users.



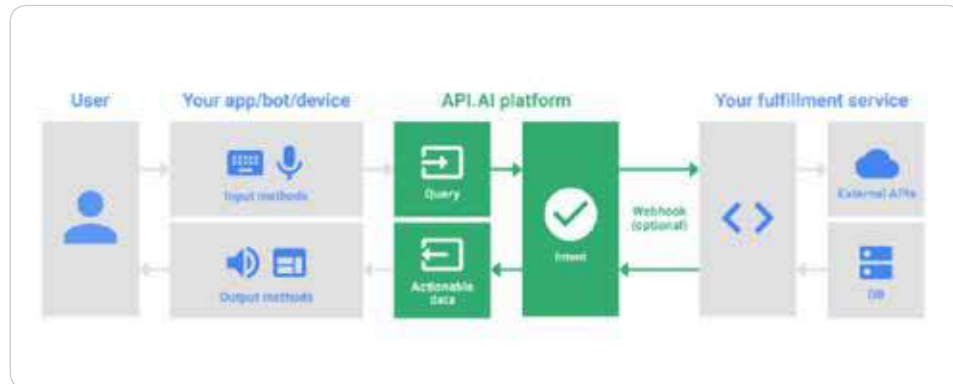
With templates, you can allow various messages to be tied directly into product options that link to purchase opportunities.



API.AI

API.AI is another third-party tool that permits chatbot creation; however, the focus here is not firmly centered around Facebook Messenger as the application platform. API.AI is able to build conversational interfaces with a number of applications and devices.

Below, you'll find the way API.AI layers into the chatbot development process.



As you can see, API.AI serves as a hybrid model between custom-coded chatbots and full third-party service providers, giving you a ton of flexibility.

API.AI receives a query as either text in natural language or an event. The system matches the query to the most suitable intent based on information contained in the information the user enters into the "Intent" module. This area of API.AI is similar to Botsify's "Stories." The process of transforming natural language into actionable data is called Natural Language Understanding (NLU).

We've covered the subject of creating your own chatbot in quite some detail in this chapter, and we offered you some examples of several tools to consider when developing a chatbot on your own. In the next chapter, we'll move beyond chatbots a bit and get you on the path to putting machine learning to work for you in your marketing efforts.



CHAPTER 8

How Marketers Can Get Started with Machine Learning

Now that we've taken the time to outline exactly how you can use and build chatbots, let's take a look at specific third-party options that let you get started with machine learning regardless of your technical prowess.

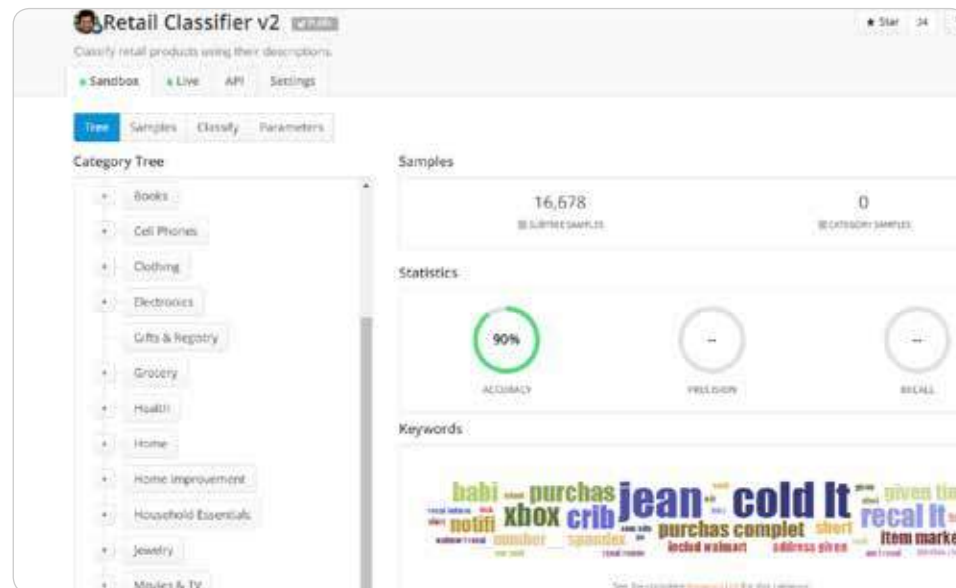
MonkeyLearn

MonkeyLearn, a machine learning API that uses machines to understand human language, offers a solution to create and test machine learning algorithms to solve problems such as sentiment analysis and topic detection. You'll find pre-created modules in the application, but you can also create your own customized modules.

The modules represent three components: **classification, extraction, and pipelines**.

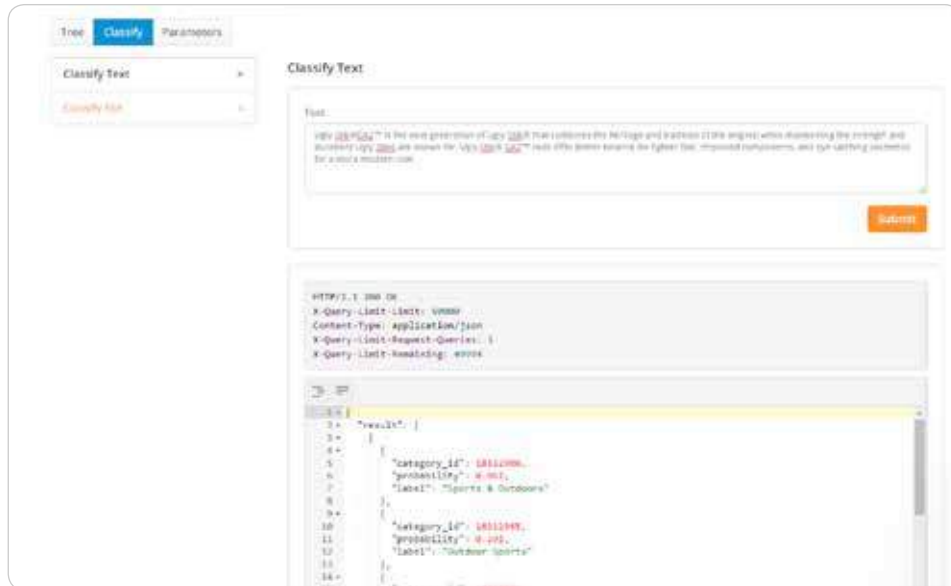
Classification

These modules take text and return labels organized in a hierarchy. Let's take a look at its pre-built retail classifier.



The algorithm for classification was taught over time by having “samples” added to the system. To see the classification in action, you can classify “Text” or a “File.” We entered a description of a fishing rod and reel combo to test the functionality.

Results come back in a JSON format, but they're still readable by people with a base level of development understanding.



The probability rate gets graded on a 0 to 1 point scale, and several categories get returned that can share space in a hierarchy. Below is a sample of the data:

```
{
  "category_id": 18313026,
  "probability": 0.152,
  "label": "Fishing"
},
{
  "category_id": 18313027,
  "probability": 0.264,
  "label": "Fishing Rods & Reels"
},
{
  "category_id": 18313028,
  "probability": 0.568,
  "label": "Fishing Rod & Reel Combos"
}
```

As you can see, Fishing Rod & Reel Combos was the category with the highest probability.

Extraction

You can't create a customized text extractor today, but you can train pre-built modules. The extractors allow you to extract keywords, useful data, and entities. We'll look below at the useful data extractor.



This tool allows people to easily extract valuable information from a block of text and store it. This type of functionality could be used to help customer service representatives manage potential issues that arise through email or social media. Post-sentiment analysis for this module could grab the information needed and store it on a customer data level.

Pipelines

The pipeline modules allow you to combine other modules into one workflow. For example, given a text, you can create a pipeline that performs the following functions:

- First, it does language detection.
- Second, if the language is English, then it does sentiment analysis in English.
- Third, if the sentiment is negative, it extracts valuable data.

With pipelines, you can declare the different processing steps made to the text and create logic to combine the partial results.

The advantage of using a pipeline is that you can do multiple process with only one HTTP request.

This option is more technical in that you need to use JSON to tie together other modules you've already built into a workflow process. Alternatively, you can turn to a useful tool called Zapier, a way to make various apps connect with one another. When you allow these apps to connect, you can save yourself valuable time by automating repetitive, tedious tasks. For example, with Zapier, you can unite a Trigger ("a new email") with a specific action ("create a contact") to execute an action in one app when something occurs in another app.

Amazon Machine Learning

Amazon launched its machine learning as a service product in 2015, and while the product isn't as intuitive to use as some other third-party applications, it's a fantastic supervised learning option for predictions. This system uses a logistic regression model, mapping variables between 1 and 0. Additionally, Amazon can also handle multiclass classifications. Multiclass classification models can do the following:

- Automate the effort of predicting object classification based on one of more than two classes
- Create a scalable way to account for classes and objects that are continually evolving

Prepare Your Data

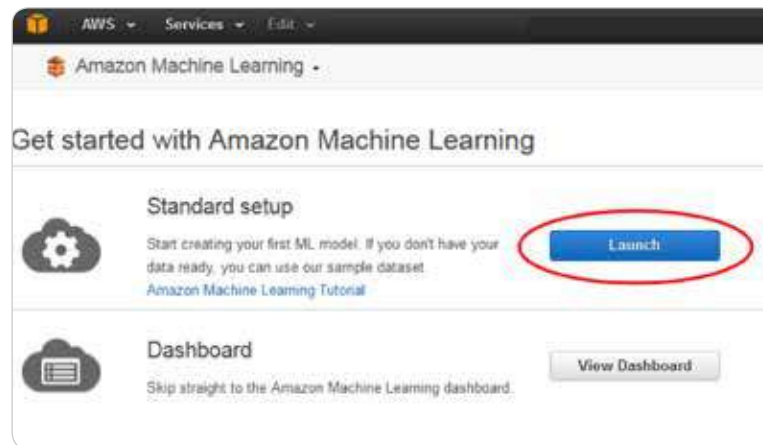
You're going to work with data stored in an Amazon S3 location, so you need to become familiar with uploading data through the Amazon Management Console. You'll get an S3 URL once you've uploaded your data; however, you should be aware that the S3 Browser is slightly more intuitive than the Amazon Management Console.

The data set can hold as many attributes as you want. An attribute in this model works as a classifier. Regardless of which attributes you include in your dataset, you must include the attribute *y*. This is the training data for your model and will later be the attribute you're looking to predict.

Since we're using a logistic regression model, we want to choose questions where the answers have only two choices (yes and no or true and false). For your training data, you can represent these two choices with 1s and 0s. For multiclass classifications, you'll use as many identifiers as you need to represent as many classes as you have set up for classifying your data.

Create a Training Source

Go into your machine learning console and choose the standard setup under "Get started ..."



On the next page, you'll need to enter the data set location that you entered in the previous step. Next, you go through a number of steps where you verify the data, allow Amazon ML to infer the schema, and select the *y* as the target attribute for predictions.

Create an ML Model

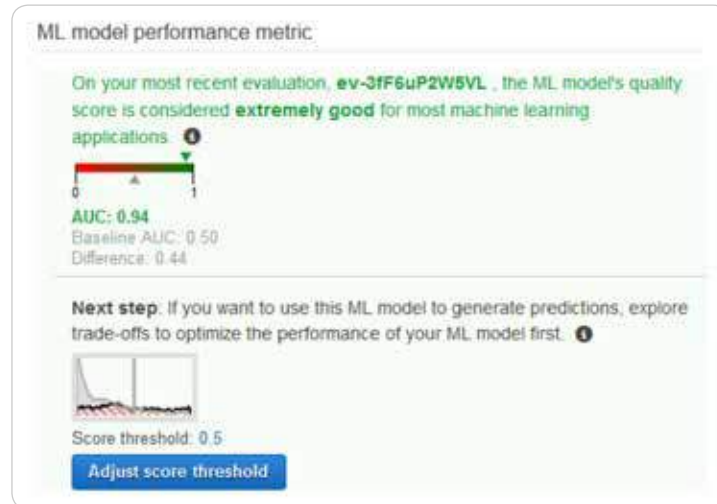
The default model for the ML model is the option that Amazon creates based on its understanding of your schema and your predictive attribute. However, you can customize the recipe yourself. If you're getting started with machine learning, it would be best to experiment with the default models and get an understanding of the system first before you venture into recipe creation.

Set a Score Threshold

Once you've created the model, Amazon will tell you if your data and recipe combined are good enough to use. Amazon computes an Area Under a Curve (AUC) Metric that expresses the quality

performance of your machine learning model. The closer the AUC for a machine learning model comes to 1, the better the quality of the performance.

Amazon lets you know in real-world language whether the data is good or bad.



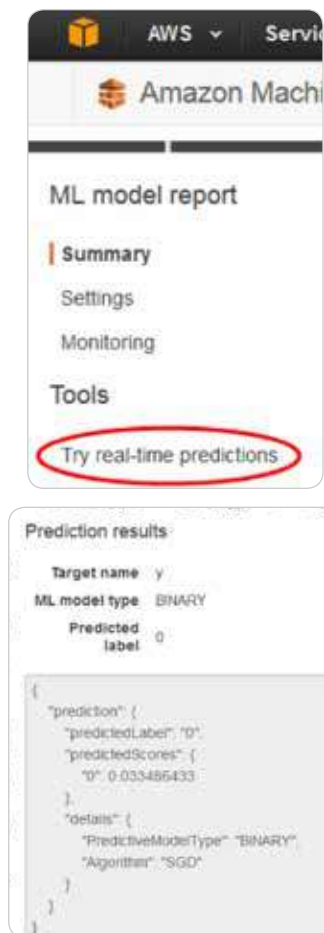
You can now use the score threshold to fine-tune your ML model performance. You can use this score to target the most true or false items predicted based on likelihood.



Generate Predictions

Now you're ready to generate predictions.

You can generate a real-time prediction to test the model. Here, you simply enter the attributes for one record and see what the prediction for the record returns.



You're also able to do batch predictions. For this process, you'll need to upload the data you intend to analyze to S3 and choose Batch Predictions in the Machine Learning menu. You'll then be asked to enter the S3 URL, verify the data, and name your batch prediction. The process will then run for you.

To view your data, go back to Batch Predictions in the Amazon Machine Learning menu.



A list of predictions will appear, and you choose the batch prediction you ran.

You want to note the Output S3 URL. Go to the Amazon S3 console, navigate to that referenced URL's location, and find the results folder. The results are in a compressed .gzip file. You can download the file, open it, and analyze the results.

The data has two columns. One is for best answer, which delivers a true or false 0 or 1 based on the score threshold you set. The other column has the raw score of the record.

Application

The application for this type of service is fairly open, and that flexibility is what makes this option such an interesting one for marketers.

One option is to glean more from your marketing campaigns. You can take the data from your analytics and create attributes for users that entered through your campaigns as they worked their way through the funnel. As you launch new campaigns, you can measure success in early days by running a model that compares your data to the performance of previous campaigns. You could also set up these models for different levels of your funnel to tell your salespeople about records that they need to focus on at each level. This arrangement is referred to as lead scoring. When applied to machine learning, you essentially “train” a machine to recognize individuals in a marketing database that have the potential to become future customers. The machine assigns predictive lead scores to identify the likelihood that each individual account will convert to a sale or a customer.

Google Cloud Prediction API

Google Cloud Prediction API provides a RESTful API to build machine learning models. You can build spam detection, sentiment analysis, and recommendation systems based on the models.

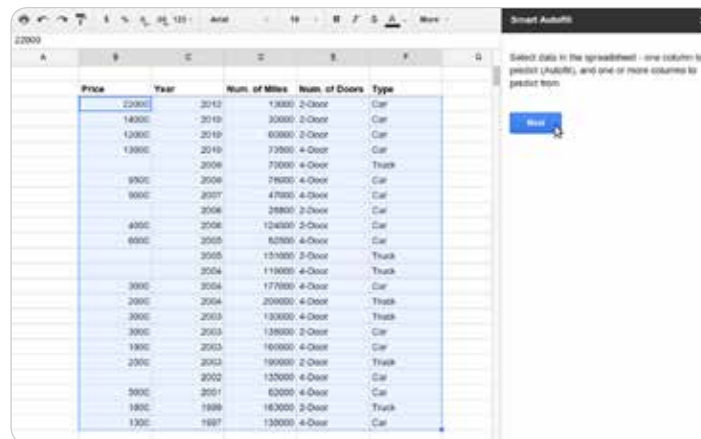
This system is similar to Amazon’s Machine Learning option, except it involves a much more advanced knowledge of basic coding skill to get up and running. However, you’ll find an option for those of us that have less refined experience creating machine learning recipes from syntax.

Google Cloud Prediction API has a Google Sheets option called Smart Autofill.

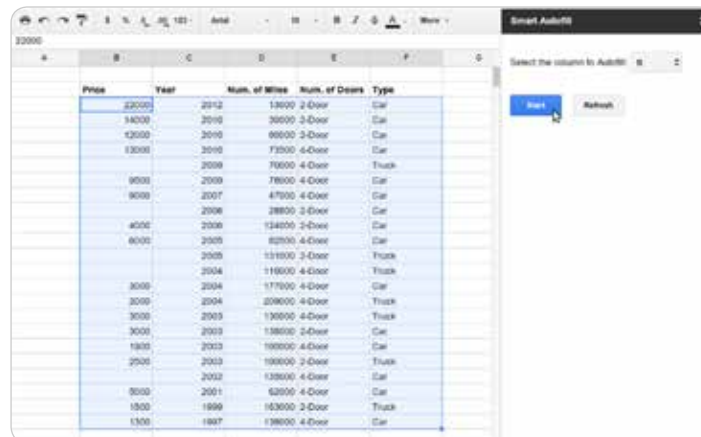
Just like our other processes, the first step starts with data. You pull in the data based on important attributes for your campaign. Unlike the process we did with Amazon, you actually want to combine your training data and data you want to predict in the same sheet.

	Price	Year	Num. of Miles	Num. of Doors	Type
1	22000	2012	13000	2-Door	Car
2	14000	2010	30000	2-Door	Car
3	12000	2010	80000	2-Door	Car
4	13000	2010	72000	4-Door	Car
5		2009	70000	4-Door	Truck
6	9000	2009	78000	4-Door	Car
7	9000	2007	47000	4-Door	Car
8		2006	28000	2-Door	Car
9	4000	2006	134000	2-Door	Car
10	8000	2005	82000	4-Door	Car
11		2005	131000	2-Door	Truck
12		2004	118000	4-Door	Truck
13	3000	2004	177000	4-Door	Car
14	2000	2004	209000	4-Door	Truck
15	3000	2003	130000	4-Door	Truck
16	3000	2003	138000	2-Door	Car
17	1900	2003	188000	4-Door	Car
18	2500	2003	190000	2-Door	Truck
19		2002	130000	4-Door	Car
20	3000	2001	82000	4-Door	Car
21	1800	1999	143000	2-Door	Truck

From the data you enter, you choose the columns that have data you want to use in predicting your results, as well as the column you plan to fill with predictions.



Next, you select the column that has the data you want to predict.



When you're done, you can view the estimated error and accuracy of your data. You can work with values that are not simply true or false statements, and the add-on function will report the possible amount of error.

We ran the Auto Smartfill add-on on the data from the Amazon Machine Learning test. The test worked, but we noticed a couple of items.

- As with traditional Google Sheets operations, the add-on has a hard time dealing with data at a very large scale. We had to parse down the record count to get it to work.
- It takes some time to get the results you need.

However, the results you get are very good.

Actual Price	Predicted Price	Error	Accuracy
22000	22000	0	0.999
14000	14000	0	0.999
13000	13000	0	0.999
13000	13000	0	0.999
9000	9000	0	0.999
9000	9000	0	0.999
4000	4000	0	0.999
6000	6000	0	0.999
3000	3000	0	0.999
2000	2000	0	0.999
3000	3000	0	0.999
3000	3000	0	0.999
1900	1900	0	0.999
2000	2000	0	0.999
5000	5000	0	0.999
1800	1800	0	0.999
1300	1300	0	0.999

Estimated accuracy: 99.99%

Number of rows: 100

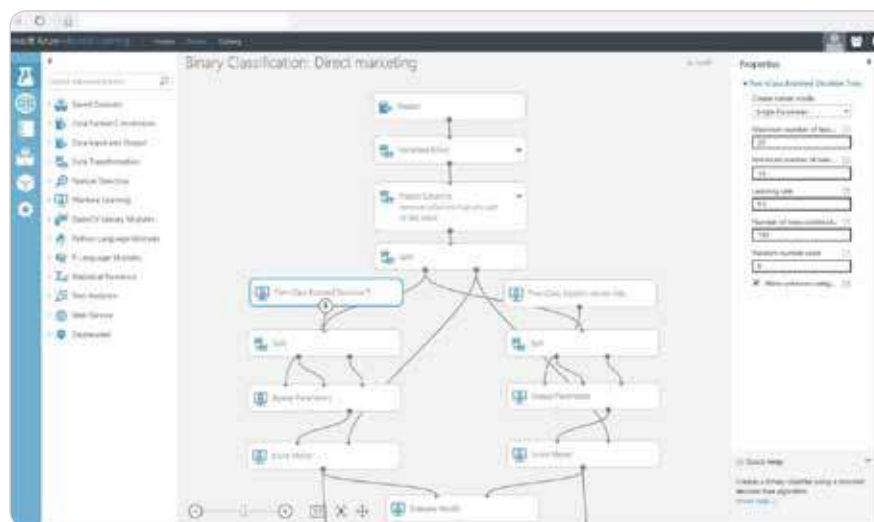
Number of observations: 100

Number of sample rows: 100

Microsoft Azure ML

Microsoft has launched ML Studio as a part of its Azure tool set.

ML Studio is interesting because you can work in a drag-and-drop interface to build your workflows.



While this arrangement makes the setup look extremely user-friendly, the robust nature of ML Studio is not something anyone can dive into. Each of the interconnected units in ML Studio is a layer of the machine learning process. This layering includes data, algorithms, scoring models, and more. Each of those components is capable of being managed through R or Python.

As you dig in, you realize that Microsoft Azure ML Studio is a powerful tool created with data scientists in mind. This feature makes it a good option if you're growing a data science team, but it's less friendly for a new user to the system.

One interesting note is that you'll discover pre-built solutions available in the Azure marketplace that might make this option more palatable for those with technical experience.

Out-of-the-Box Tools

While the above options allow for an entire world of flexibility and creativity for your machine learning needs, some of us only need tools that work today for very specific issues. Luckily, machine learning-based software solutions are on the rise in the adtech space, and solutions exist for you to sign up today and put machine learning to work for you.

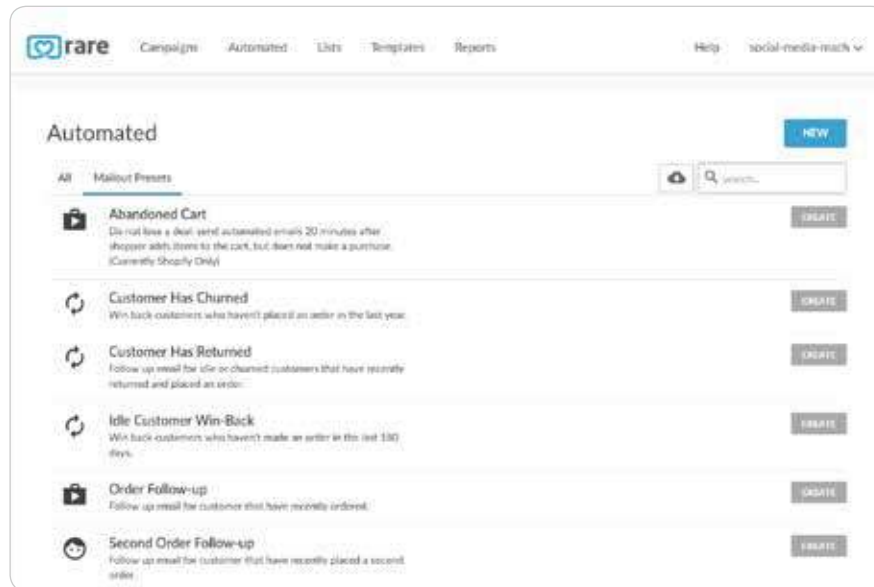
Atomic Reach

Atomic Reach is an out-of-the-box solution that helps marketers know what to do with their content to get the most value out of their content marketing. This tool helps you create the content, share it, engage with data in real time to make on-the-spot changes to strategy, and see trending topics that you can use to create timely content.

The creation module measures 22 attributes of the content you're creating in real time. These attributes include readability level, length, emotions and more. The Atomic Reach Writer lets you train your machine learning by assigning different attribute levels as an "audience." You generate content for that audience, and the system lets you know where you score.

Rare.io

Rare.io has created an email marketing optimization solution that works with Shopify and BigCommerce. The company's main offering is predictive email recommendations. You can create and use preset Mailouts based on visitor segmentations:



You're able to test different subject lines and email templates to see how different customer segmentations will likely react to them based on the predictive analysis of Rare.io.

Open-Source Options

Before we close the door on the topic of tools and ways that marketers can get working with machine learning today, we wanted to touch on some open-source platforms that exist and are worth investigating.

Keras.io

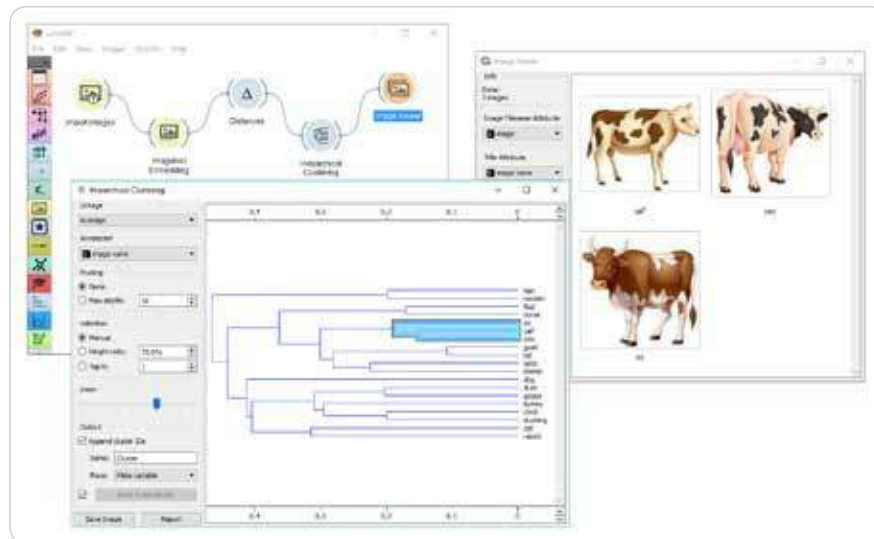
Keras is an open-source neural network library written in Python. It's capable of running on top of three open-source machine learning software libraries, DeepLearning4j, TensorFlow, and Theano. The library has many implementations of commonly used neural network building blocks including layers, objectives, and optimizers.

Orange

Orange is an open-source data visualization, machine learning, and data-mining toolkit. It has many uses, but its widgets are dedicated to the following:

- **Data:** Data input, filtering, and manipulation
- **Classification:** Supervised Machine Learning algorithms for classification
- **Regression:** Supervised Machine Learning algorithms for regression
- **Evaluate:** Cross-validation, sampling-based procedures, and reliability scoring
- **Unsupervised:** Unsupervised Learning algorithms for clustering and data projection techniques

Orange is a great learner tool because it allows anyone to jump in and test out machine learning. Even though it is great for beginners, Orange allows you to embed deep learning networks and other outside resources. For example you can utilize ImageNet to do image classification :



The best part of Orange is it's easy to use drag and drop interface.

The steps to run basic linear regression are very straightforward.

Needed Signals

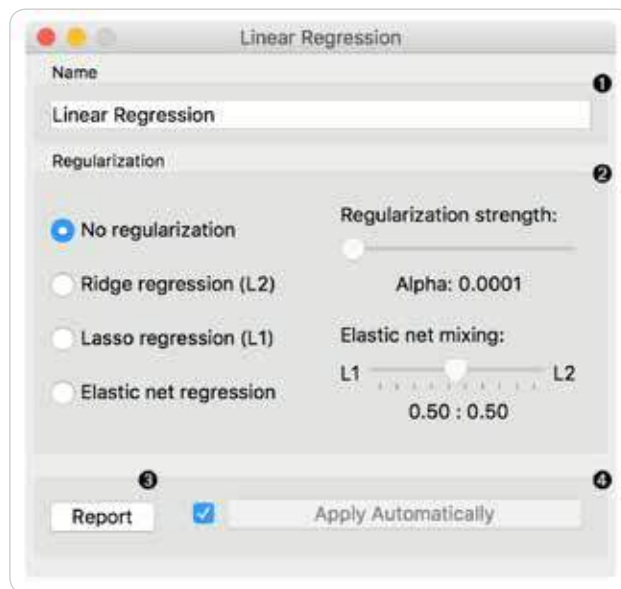
Inputs:

- A data set
- Preprocessor
- A preprocessed data set.

Outputs:

- Learner
- A linear regression learning algorithm with settings as specified in the dialog.
- Predictor
- A trained regressor. Output signal sent only if input Data is present.

First, you will drag and drop the Linear Regression Widget. You are able to customize the widget to customize the learner.



1. You will name the predictor
2. You can choose anyone of the existing models
3. You can produce a report
4. Save your changes

Next, you will setup a workflow from your data to Linear Regression. In the below example, you can see that you can run two models simultaneously, in this case random forest is being run in parallel.



Knime

KNIME (Konstanz Information Miner) is an open-source data analytics, reporting, and integration platform that has components of machine learning included in its modular data pipelining. KNIME integrates with various open-source projects to help with machine learning functionality, such as Weka and R Project.



○ CHAPTER 9

Most Effective Machine Learning Models

Previously, we've already talked about the macro levels of machine learning (ML) which included the supervised and the unsupervised module. Additionally, we also discussed the best tools which marketers use for ML and the best way of using such tools. In this article, we'll be talking about the models and the algorithms which stand as inner logic for all those that we've previously discussed.

Having a thorough knowledge on the models that build up the core of each tool will give you more options when using those tools.

Regression Models

Linear Regression

The primary task of linear relationship models is to establish the relationship between two different factors or variables. The variable or factor being predicted is the dependent variable; the factors used to predict values of the dependent variable would be the independent variables.

Establishing the relationships between dependent and independent variables is represented by the equation $y = ax + b$ where:

y = the dependent variable

a = the slope

x = the independent variable

b = the intercept

The intercept and the slope are determined by minimizing the total sum of the square difference of the regression line and the data points.

Simplifying this formula, we can say: Linear regressions predict values through the existing variables and their relationship with one another.

A Sample Case in Marketing Use

A value which this particular model could have is predicting the value of possible client personas being targeted toward a new product release. As soon as you have broken down your client personas — including their buying patterns — you will be able to pull out data points which represent corresponding client personas to substitute in the linear regression equation. This arrangement will provide a precise prediction for how much a client persona may try to purchase.

The moment marketers use a linear regression as a way to estimate sales out of each persona, they can also estimate the marketing channel value to offer a better estimate on the personas for each channel. This optimization will then optimize the marketing campaign even before its launch.

Logistic Regression

Unlike actual regression, this type of regression isn't focused on predicting the numeric variable value through a set of inputs. Alternatively, it delivers a probability that the provided input point is coming from a specific class. In logistic regression, we are simply looking for probabilities that an event will happen.

Logistic regression focuses on the assumption that the input space could be set apart to two “regions” through a linear boundary.

The two region answers would be similar to

True/False

Yes/No

The tools that use logistic regression within their core include Google’s prediction API along with an ML application from Amazon.

A Sample Case in Marketing Use

With the case mentioned earlier, we could use a logistic regression to determine if a sale has a likelihood to close, and we are able to use the linear regression to determine the amount of money which clients are likely to spend. Using this information, we can use a lead list which follows:

Linear: decides how much money a lead will be likely to spend

Logistic: determines the chances of clients spending that money

Next come up with a forecast for your overall lead list using that combination of amount and probability.

Model Clustering

K-Means Clustering Algorithm

K-means is among the popular unsupervised ML algorithms designed for clustering analysis. It is a non-deterministic and iterative method. Its algorithm works over a specified data set using a predefined set of clusters, k. The result of a k-means algorithm is the k clusters of input data partitioned from the clusters.

As an example, let’s have a k-Means clustering application for Wikipedia search results. We’ll use the search term Lincoln over Wikipedia. When we do, we will get results containing this word, and they usually give us references to the past U.S. president, a city in the United States, or a car company. We can use this algorithm to put these results and pages into groups which talk about the same concepts. Thus, the algorithm will group the results and pages into clusters.

A k-means clustering algorithm is commonly used by search engines to identify similarities of search results. This algorithm provides lesser time for the searcher when looking for a more precise result using the search engine.

A Sample Case in Marketing Use

K-means clustering stands out to be an effective tool when developing marketing and buying personas out of clusters of information concerning buyers.

Cluster analysis proves to be useful for market segmentation. This analysis means having to divide a market’s potential customers over to different subsets wherein customers that belong to the same group are the same concerning a specified characteristic and vice versa. This arrangement will provide a convenient and more accurate marketing mix depending on the target customer specified.

The background is a solid light blue color. It features several concentric circles of varying radii, some solid and some dashed, centered around the text. There are also two solid light blue circles, one at the top and one at the bottom right, connected to the concentric circles by thin lines.

MOST EFFECTIVE MACHINE LEARNING MODELS

Thorough
knowledge on
the models that
build up the
core of each tool
will give you
more options.

Model Classifications

Decision Trees

Decision trees are graphical representations which use branching methodology to deliver all possible results of a certain decision using a few specified conditions. Within a single decision tree, the particular internal node gets represented as a test over an attribute. Each corresponding branch of the tree will represent the possible outcome of the test. The leaf node represents the particular class label such as the decision made after all attributes have been compute. Classification rules will therefore be represented via the path from the root toward the leaf node.

The types of decision trees include the following:

- **Classification trees:** The default type of decision trees is used to set apart data sets into two varying classes using a response variable. This arrangement is generally used when a categorical nature is detected out of a response variable.
- **Regression trees:** This type of decision tree is used if the result or target variable will be numerical or continuous — mostly used for predictive type of issues.

Two classifications of decision trees exist, depending on the type of the target variable. These two types are composed of the binary variable decision tree and the continuous variable decision tree.

Such ML algorithms allow for reliable decisions in uncertainties and help improve communication by providing a visual representation for a decision situation. Decision-tree ML algorithms provide assistance to data scientists by capturing the idea that when one decision gets implemented, its operational nature will be something specific.

Decision-tree algorithms give developers the best decisions by letting them traverse forward and backward in a calculation path.

A Sample Case in Marketing Use

We can use decision trees referring to the various cases we've mentioned previously.

Regarding classification, trees gets used in classification of data sets through response variables. Using this model, we are able to use the actions of historical data from previous website visitors to classify potential customers over to various lead funnel buckets.

Regression trees will then give a prediction for whether a client will make a purchase using the actions provided.

Random Forests

This ML algorithm uses the bagging approach to create a number of decision trees using random subsets of data. One model gets trained a number of times over random samples of data sets to obtain high accuracy prediction over the random forest algorithm. Within this method, all outputs combine to come up with the last final decision. This final prediction gets computed by creating a poll of the results from each decision tree.

A Sample Case in Marketing Use

Random forests are much like decision trees. We can use them to predict client behaviors. The only difference is that random forests use a number of decision trees and give marketers a higher level of accuracy.

Naïve Bayes Classifier Algorithm

Classifying lengthy text notes is almost close to impossible when done manually. A Naïve Bayes classifier algorithm comes into place for this occasion. A classifier serves as a function which assigns a value for a population using the categories available. An example of this algorithm is the spam filtering in emails where the filter becomes a classifier that assigns an email as either spam or not spam.

This classifier is among those common learning methods grouped by similarities which makes use of Bayes' theorem of probability to build ML models, especially those related to disease prediction and document classification. This is merely a basic classification of words that is on account of Bayes probability theorem with regards to subjective analysis of any content.

A Sample Case in Marketing Use

Facebook uses sentiment analysis to determine status updates showing a negative or positive emotion.

Google uses document categorization to index documents and find relevancy scores such as PageRank.

A Naïve Bayes algorithm can be used for article news classification for politics, sports, technology, and other topics.

Google also uses email spam filtering to decide whether to classify emails as spam or through the Naïve Bayes algorithm.

Support Vector Machine Algorithm (SVM)

This supervised ML algorithm design is mainly for classification and regression issues in which a certain data set instructs the SVM about the classes so that it can classify fresh data sets. This classification is achieved by classifying the assigned data sets to different classes through linear hyperplanes. The SVM algorithm will try to maximize the certain distance of the different classes involved, known as margin maximization. If the line that maximizes such distance will be identified, the possibility to generalize well over unseen datasets will be increased.

Two types of SVMs

Linear SVMs are the training data which include the classifiers separated by the hyperplane.

Nonlinear SVMs concern the classifiers' inability to become separated by the hyperplane. An example of this situation is face detection which comprise a number of images composed of faces and non-faces (images other than faces). Through specific conditions, the training data becomes too complicated, becoming impossible to trace a representation for each feature vector. Having to separate the set of faces linearly through the set of non-faces is going to be a complex task.

A Sample Case in Marketing Use

Social Media SVMs get used to track clear mentions of a certain brand. The SVMs classify them from among other brands. Some of the metrics include loyalty, likes, and perceived value.

Through theme generation through social media, the algorithm can map social mentions over to predefined categories specific to an industry. An example of this mapping is the hospitality and travel labels which point out key themes discussed over social media regarding a company's services or products.

Email routing SVMs automatically classify emails over clients directed to complaint handlers. This classification reduces turnaround times and offers quick solution to issues.

Opinion mining is the computational learning of the attitude of people, their opinions, and their stances on issues, companies, individuals, or services. SVMs are used to classify the polarity among texts in a document or sentence level in the classifications: negative, positive or neutral.

Artificial Neural Networks (ANN)

The artificial neuron network (ANN) serves as a computational model based over a structure and function of a biological neural network. Data which flows in the network will affect the structure of an ANN since the neural network varies — or becomes smarter in a sense — out of the output or input.

ANNs are nonlinear statistic data modeling platforms where the intricate relationships among outputs and inputs become modeled or where patterns exist.

Additionally, ANN is a neural network. The ANN comes with a few advantages; one of them is learning just by observing data sets. Using ANN as a random function approximation tool, you can estimate the cost effectiveness and best methods to arrive over a solution during the process of defining computing functions and distributions.

ANNs come with three interconnected layers. The initial layer comes with input neurons where such neurons deliver data over to the second layer and then over to a third layer.

Having to train an artificial neural network will require choosing over an allowed model in a pool of associated algorithms.

A Sample Case in Marketing Use

Ad targeting uses a neural network to efficiently decide where to deploy advertising spending. One example is Baidu which handles ad targeting. It launched the use of artificial neural networks.

This Chinese web giant also uses deep learning to target ads over its many lines of services. As Andrew Ng, a co-launcher of the deep learning operation of Google and Baidu's chief research officer, "It's used very successfully in advertising." He further adds, "We have not released revenue numbers on the specific impact, but it is significant."

Your model selection will be decided by the data you are working with, the tools your team utilizes, and the questions you are trying to answer. As you can see, an ML model is not an one size fits all tool. Luckily for marketers the decision of what model fits with which activity is being made by ML and AI tool developers that are making it easier for professionals across occupations to deliver results using data science.



CHAPTER 10

How to Deploy Models Online

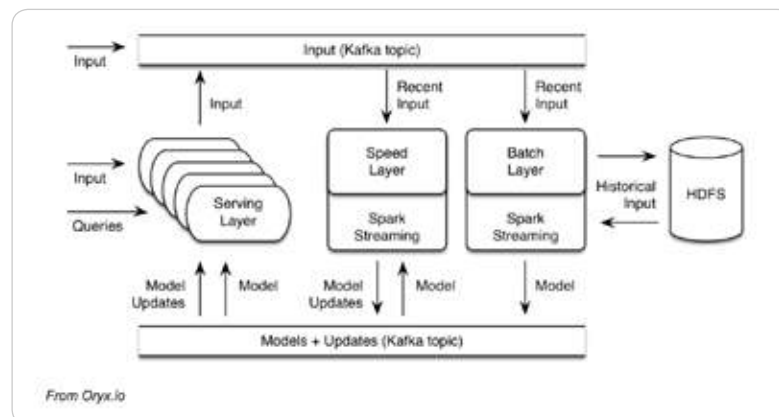
Marketers and businesses who want to use machine learning (ML) beyond the tools previously discussed may require better customization to deploy their specific online models. In addition, marketers as well as business organizations may be able to come up with ways on how their developers can fully use their models which, in this case, may call for other helpful tools.

Self-Managed and Hosted Tools

Cloudera Oryx

Oryx is primarily designed to work on Apache Hadoop. For those who are not familiar with Hadoop, this open-source software platform is designed for storing data sets and executing applications over groups of commodity hardware. It can store a substantial amount of data of any kind, provides a high level of processing power, and is capable of handling a virtually limitless amount of tasks or jobs.

Oryx 2 is a product of Lambda Architecture built upon Apache Kafka and Apache Spark; however, Oryx 2 is designed with a specialty in real-time machine learning at a large scale. This framework allows for easy creation of applications, but also comes with packaged, end-to-end apps designed for collaborative filtering, regression, clustering, and classification.



From Oryx.io

Oryx is composed of three tiers: Each tier builds over the one below:

1. The generic Lambda Architecture tier provides serving, batch, and speed layers that are not particular to ML.
2. Specialization over top provides ML simplifications for hyperparameter selecting and other selections.
3. The end-to-end execution of similar standard machine learning algorithms gives the application (k-means, random decision forests, or Alternating Least Squares Method for Collaborative Filtering) over the top.

Oryx Implementation

Execute the three layers using:

```
./oryx-run.sh batch
./oryx-run.sh speed
./oryx-run.sh serving
```

You don't have to use these layers over one machine; however, the application can also be possible if the configuration specifies various ports to accommodate the speed and batch layer Spark web user interface along with the port for serving layer API. You can run the serving layer over a number of machines.

Say the batch layer Spark user interface runs on the 4040 port of the machine where you started it — that is, unless this arrangement was altered by a configuration. As a default, the port 8080 will come with a web-based console designed for the serving layer.

Example of Use

A sample GroupLens 100K data set is found in a u.data file. The data set should be converted to CSV:

```
wget --quiet --post-file data.csv --output-document - \
--header "Content-Type: text/csv" \
--data-binary @u.data > data.csv
```

Provide the input over a serving layer, having a certain local command line tool such as a curl:

```
curl -XPOST -H 'Content-Type: text/csv' -d @u.data > data.csv
```

In case you are actually tailing the particular input topic, you will find substantial CSV data flow toward the topic:

```
196,242,3.0,88125042186
196,242,3.0,881250449
186,383,3.0,881717742
22,377,1.0,878887116
244,51,2.0,880606323
165,346,1.0,886397996
298,474,4.0,884182806
...
```

After a few moments, you will find the batch layer triggering a new computation. This sample configuration will start per five-minute intervals.

The data provided is first delivered to HDFS. The sample configuration has it delivered to the directory `hdfs:///user/example/Oryx/data/`. Located inside are directories named by a time stamp with each one having a Hadoop part — files equipped with input being SequenceFiles of a text. Despite not being pure text, if you have to print them, you should be able to deliver recognizable data since it is actual text.

```
SEQ[org.apache.hadoop.io.Text]org.apache.hadoop.io.Text[22,377,1.0,878887116]
62...
```

Afterward, a model computation will take place. This computation should show results in the form of a number of fresh distributed jobs over the batch layer. The Spark UI will be stared over `http://your-batch-layer:4040` as provided in the sample configuration.

The model will then be complete and will be considered as a blend of PMML and various supporting data in the subdirectory of `hdfs:///user/example/Oryx/model/`. Here's an example: The model PMML files will be PMML files that contain elements such as the following:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<PMML xmlns="http://www.dmg.org/PMML-4_3" version="4.3">
  <Header>
    <Application name="Dryx"/>
    <Timestamp>2014-12-18T04:48:54-0800</Timestamp>
  </Header>
  <Extension name="X" value="X"/>
  <Extension name="Y" value="Y"/>
  <Extension name="features" value="10"/>
  <Extension name="lambda" value="0.001"/>
  <Extension name="implicit" value="true"/>
  <Extension name="alpha" value="1.0"/>
  <Extension name="implicit" value="false"/>
  <Extension name="XID">>56 168 222 343 397 ...
  ...
</PMML>
```

The Y and X subdirectories that go with it come with feature vectors such as:

```
[56,[0.5746202834154230,-0.08096614131333057,-0.020456222765775263,
0.6039821216690552,0.1407901814774658,-0.018654312114339861,
-0.37342063488340266,-0.2370768843521007,1.148260034028485,
1.0645643656769151]]
[168,[0.8722769882777296,0.4370416943031704,0.27402044461549885,
-0.031252701117490456,-0.7241385753098256,0.026079081002582338,
0.42050971702065714,0.27766923396205817,0.6241038215056671,
-0.48530795198811266]]
...
```

If you're tailing the particular update topic, you will find these values to be published over the topic. This publication will then be detected by the serving layer later and will provide a return status at the /ready endpoint of zoo OK:

```
wget --quiet --output-document - --server-response \
http://your-serving-layer:8080/ready
...
HTTP/1.1 200 OK
Content-Length: 0
Date: Tue, 1 Sep 2015 13:26:53 GMT
Server: Dryx
```

```
wget --quiet --output-document - http://your-serving-layer:8080/recommend/17
...
50,0.7749542842056966
275,0.7373013861581563
258,0.731818692628511
181,0.7049967175706345
127,0.704510989947408
121,0.7014631020793741
15,0.6954683387287907
288,0.6774989711024022
25,0.6663619887033064
285,0.6398968471343595
```

Apache Mahout

Three primary features of Apache Mahout include the following:

- A basic and extensible coding environment and framework designed for building scalable algorithms
- A wide array of premade algorithms for H2O, Apache Flink, and Scala + Apache Spark
- Samsara, the vector mathematics experimentation setting having R-like syntax that works at scale

Considering that Mahout's primary algorithms are designed for classification, you can implement clustering as well as batch-based collaborative filtering over Apache Hadoop through the map/reduce paradigm. However, this process does not prevent any contributions over to Hadoop-based implementations. The contributions which operate over one node or over a non-Hadoop cluster become adapted or welcomed. As an example, the "taste" collaborative-filtering influencer part of Mahout was originally a single project and runs stand-alone without the aid of Hadoop.

An algebraic back end-independent optimizer — along with an algebraic Scala DSL to unify in-memory and distributed algebraic operators — makes up the setting. During this time, the available algebraic platforms supported include H2O, Apache Flink, and Apache Spark. MapReduce algorithms have gradually lost support from users.

H2O

H2O is an open-source, scalable ML and a predictive analytics system which provides convenience in building ML models over substantial data. H2O also offers effortless productionalization over models used in an enterprise.

This platform's code is written in Java language. Within H2O, a certain DistributedKey/Value store gets used for approach as well as reference models, objects, data, and other elements over all the machines and nodes.

The algorithms are implemented over H2O's dispersed Map/Reduce framework. After implementation, you make use of the Java Fork/Join framework to deliver multi-threading. The particular data gets read in parallel and gets distributed all over the cluster and stored inside a memory within compressed columnar format. Additionally, the H2O data parser comes with a built-in intelligence that guesses the schema of a certain incoming data set and offers data ingest support over a number of sources over various formats.

The REST API from H2O provides access to every capability of H2O out of both external program and script out of JSON on HTTP. The H2O web interface, R binding, and Python binding uses REST API.

Accord.NET Framework

For developers as well as marketers who are presently in the .net working environment, Accord Framework stands out as a robust open-source alternative when deploying models.

Accord.NET Framework is actually a .NET ML framework combined with image and audio-processing libraries coded in C#. Accord.NET is a comprehensive framework used to create production-grade computer vision, signal processing, statistical applications, and computer audition.

MLlib

MLlib is one popular ML library consisting of basic learning algorithms as well as utilities that include regression, clustering, collaborative filtering, dimensionality reduction, and classification.

This platform conveniently adapts to Spark's APIs and works conveniently with NumPy from Python and R libraries. Hadoop data sourcing can also be used on this platform, a function which allows it to work effectively with Hadoop workflows as well.

Fully Managed Tools

Yhat

Yhat is a Y Combinator-supported enterprise delivering an end-to-end data science platform for deploying, managing, and developing real-time decision APIs.

This platform conveniently takes away the pains brought about by IT obstacles and problems which come with cloud-based data science such as server setting and configuration. Through this platform, data scientists and experts are able to convert static insights into available decision-making APIs which can work effortlessly with any client- and employee-facing application. This platform also brought about the birth of Rodeo, an open-source IDE (integrated development environment) for the Python platform.

Developers have also come up with the product ScienceOps, a tool which aims to provide data science users, experts, and developers the capability to seamlessly and quickly deploy models in any environment. You use these models in cases involving lead scoring, image recognition, credit analysis, and various other predictive models.



Developers, data scientists, and experts use Python or R to easily and quickly deploy models through Yhat instance. Coders and developers are then able to make use of a REST API in embedding models over to web applications. This embedding allows for real-time ML application. One excellent use case of such technology surrounds the submission and approval of real-time credit applications.

Below is an example of a model deployed through Yhat.

To use Yhat, you generate your model within Python or R session, change the user name and API key fields, and then run the script.

```
from yhat import Yhat, YhatModel, preprocess

class HelloWorld(YhatModel):
    @preprocess(in_type=dict, out_type=dict)
    def predict(self, data):
        m = data["name"]
        greeting = "hello " + str(m) + "!"
        return { "greeting": greeting }

yh = Yhat("YOUR_USERNAME", "YOUR_API_KEY", "https://vanbox.yhat.com/")
yh.deploy("helloworld", helloworld, globalis())
```

The Yhat platform comes with a basic Beer Recommender which provides a simple use case of the platform to generate accurate predictions based on the specific data that a user enters.

Now that we've looked at how marketers with an advanced understanding of ML can develop and deploy models online, it's time to consider how data scientists take modeling to the next level in the next chapter of this guide.



CHAPTER 11

How Data Scientists Take Modeling to the Next Level

Data scientists are exceptional people who bring data analysis to a new level. These analytical experts have a high level of technical skills and can solve the most complicated problems with a high level of inquisitiveness.

In simpler words, data scientists are those who structure and deploy the specific models which we've been discussing. In the previous discussions, we've talked about platforms, and those platforms are where data scientists deploy their specific models. However, they can also use other tools to analyze their data.

Roles of a Data Scientist

- Pinpoint which data-analytics problem would offer the most benefit to an organization.
- Identify the ideal variables and data sets.
- Gather multiple sets of unstructured and structured data out of disparate sources.
- Verify and clean data for increased accuracy, uniformity, and completeness.
- Come up with algorithms and models to hunt the stores of large data.
- Analyze data for pattern and trend identification.
- Interpret data to trace ideal solutions and opportunities.
- Share findings with stakeholders through the use of visualization models and other methods.

From research conducted by Burtch Works about data scientists, 88 percent of them have a master's degree while 46 percent have doctoral degrees. Not surprisingly, most data scientists are highly educated people.

Languages

Given that the primary role of data scientists is to connect the world of IT to the world of mathematics, they need to understand how to deploy their code and how to use technology to deliver their analyses. The most common coding languages which data analysts use today are R Code and Python. Other alternatives include Scala and Java; however, R and Python have become the norm in this profession.

R vs. Python for Data Science

Both R and Python have practical applications for data scientists. Below you'll find some highlights of the comparisons between R and Python and when you may use each one.

R is useful when the data analysis you plan to do requires you to dig in for some preliminary data dives, and it's useful for almost any type of data analysis you'd want to do because of the large number of packages and readily accessible tests that can provide you with the tools you need to get up and moving on your data analysis. R packages (see below) can further your reach with packages for data visualizations and machine learning.

Python is useful when you need to integrate your data analysis activities with web apps or if you need to incorporate some statistics codes into a production database arrangement. As a robust programming language on its own, Python makes for a practical way to put algorithms into place when you're working with production applications.

In the past, Python packages for data analysis were in their infancy, but their growth and capabilities have increased over recent years. NumPy, SciPy and pandas for data manipulation (see more

in Python libraries below) have made Python useful from a data analysis perspective. From an ML standpoint, scikit-learn is useful when you want to consider multiple versions of a single algorithm.

Why Do Data Scientists Recommend Python?

Among those high-level languages commonly used by data scientists, Python makes the claim that it's easy to learn, friendly, fast, open, and powerful. Developed in the 1980s and named after the comedy group Monty Python, this platform has established its presence in the data science industry because of its number of libraries created using this language.

Additionally, Python also has a simple syntax which is easy to understand for coding newbies, one that is easily recognizable for anyone who has dealt with Java, Visual Basic, C/C++, and Matlab. Python prides itself on being a multipurpose, user-friendly programming language when dealing with quantitative and analytical computing.

Data scientists also find Python an easy language to use when developing visualizations that serve as an essential function to their work while relaying analysis to organizational stakeholders. This process gets carried out using APIs such as data visualization libraries and Plotly.

Another selling point of Python is that it scales quickly and gets equipped with handy tools such as IPython Notebook – known today as Jupyter Notebook. This tool serves as an interactive computational setting where rich text, plots, rich media, mathematics, and code execution can be joined. You can also run a number of blocks and lines of code over various cells using this tool. Additionally, you can play with them, interchange their positions up or down, and have results display just below the cell. Lastly, writing R, Scale, SQL, and other languages in this tool is also possible which permits easier and more efficient workflow.

Why Do Data Scientists Recommend R Code?

R is another open-source coding language and platform setting developed for graphics and statistical computing. It is R Foundation for Statistical Computing supported.

Developed in the 1990s by Ross Ihaka along with Robert Gentleman from New Zealand's University of Auckland, R served as statistical software for students. It was further improved through the decades through the ever-increasing number of user-created libraries.

R Code can be used in many libraries and data scientists prefer it because of these other reasons:

- R serves as an object-oriented platform developed by statisticians. R creates objects, functions, and operators, all of which give users a convenient way to model, visualize, and explore data.
- Using R, standard statistical methods become easy to implement. Since most predictive and statistical modeling today is already done in R, most techniques get introduced first using R.
- R is free and comes with a high level of numerical accuracy and quality thanks to countless improvement efforts by developers through the years. Having the R open interface offers easy system and application integration.

The R environment provides the following:

- Efficient storage facility and data handling
- A selection of operators to make calculations on specific arrays within particular matrices
- A manageable set of tools used for data analysis
- Graphical models designed for data analysis

These reasons can support why R is an excellent choice among data scientists over other platforms such as Python. That is, R can offer a better ability to deploy, visualize, and develop models within one system.

Packages and Libraries

Packages and libraries give data scientists the tools inside Python for faster scaling of visualizations and data analysis.

Python Libraries

In the Python Tutorial, Modules are defined as files composed of Python statements and definitions — the filename equipped with the suffix `.py`.

Packages offer a way to structure the Python's module namespace by “dotted module names.”

“Library” is a term used for a number of codes applied for use in various applications. It delivers generic functionality for specific applications.

The moment a package or module gets published, those entities can be called by many as a library. For much of the time, a library can be composed of packages or a single package, but it can still be considered as a single module.

Among those Python libraries commonly used by data scientists are the following:

NumPy

NumPy is a foundational library used for particular computing within Python. This platform introduces models for multidimensional matrices and arrays. Additionally, routines enable developers to execute advanced statistical and mathematical functions using as little coding as possible.

SciPy

This platform builds upon NumPy through accumulating algorithms along with high-level commands designed for visualizing and manipulating data. It also comes with functions designed for solving differential equations, computing integrals numerically, and many other functions.

Pandas

Pandas offer tools and data structures for data analysis in statistics, finance, social sciences, and engineering. This platform is convenient to use over unlabeled, messy, and incomplete data, and it allows shaping, merging, slicing, and reshaping of data sets.

IPython

This platform extends the functionality of the interactive interpreter of Python. IPython accomplishes this functionality through a souped-up interactive shell combining a shell syntax, rich media, and tab completion, plus a command history retrieval. Additionally, this platform also serves as an embeddable interpreter for various programs used for debugging.

Matplotlib

This platform is essential in a Python library when you're developing 2D plots and graphs. Compared to the more advanced libraries, Matplotlib needs more commands to create better-looking graphs and figures.

Scikit-learn

This platform can build over SciPy and NumPy by combining algorithms for common machine learning (ML) and data mining tasks, along with clustering, classification, and regression. It is a curated library that permits easy selection of various versions of a single algorithm.

Theano

This library uses a NumPy-like syntax to optimize and evaluate mathematical expressions. It comes with an amazing speed that is highly essential for deep learning and complex computational tasks.

TensorFlow

Developed by Google, this platform is a high-profile entrant in the field of ML. TensorFlow became developed as a successor to the open-source platform DistBelief, a framework for modeling and training neural works.

Scrapy

This aptly named platform is designed for developing spider bots which crawl over the web and extract structured data such as contact info, prices, and URLs.

NLTK

This platform compiles several libraries designed for NLP, Natural Language Processing. NLTK (Natural Language Toolkit) enables easy entity identification, text tagging, and parse trees display. In addition, you can also do more complex tasks such as automatic summarization and sentiment analysis.

Pattern

This platform combines NLTK and Scrapy functionality and acts as an out-of-the-box solution for web mining, ML, NLP, and network analysis. It comes with tools such as APIs for Google, Wikipedia, Twitter, and text-analysis algorithms.

Seaborn

Seaborn is another common visualization library used by many data scientists. It builds over Matplotlib's foundation and is an easier tool to use for generating various types of plots along the lines of violin plots, time series, and heat maps.

Bokeh

This tool permits zoomable and interactive plots over modern web browsers through JavaScript widgets. It also provides a three-level interface over high-level abstractions which enable you to generate complex plots quickly.

Basemap

This tool allows you to easily manipulate and transfer maps over to Matplotlib by taking Matplotlib's coordinates and using them over 25 different projections.

NetworkX

NetworkX permits easy creation and analysis of graphs and networks. This platform works conveniently on standard and non-standard data formats.

Data scientists
can solve the
most complicated
problems with
a high level of
inquisitiveness.

Data scientists
can solve the
most complicated
problems with
a high level of
inquisitiveness.

R Packages

R Packages are a collection of R functions, compiled code, and data stored over a well-defined format. As soon as you install this tool, load it over to a session so that you can use it.

forecast

If you need to consider time series, or a series of observations about well-defined data points obtained through repeated measurements over time, you'll find the forecast package useful for time series analysis. This package can also help you when you need to predict the future value of a time series (for example, a future stock price).

nnet

This common platform is widely used because it is easy to understand. Unfortunately, nnet can also be a challenge to use since the platform gets limited only to one layer of nodes.

klaR Visualization and Classification

The CARET (Classification and Regression Training) module combines prediction and model training. This combination allows data scientists to run multiple algorithms for a specific business problem. Additionally, developers can investigate parameters for a specific algorithm through controlled experiments.

plyr

This package offering is like the salvation of a cool breeze on a sultry summer day. When you need to break up a big data structure into homogenous units, apply a function to each one of those pieces, and then bring all the results back together, plyr has the capability to fit the same model to each subset of a data frame, calculate summary statistics for each group, and then perform transformations that include scaling and standardizing.

ggplot2

Answering the pleas of those who work with graphics, ggplot2 offers a powerful model of graphics creation that allow you to shape complex multi-layered graphics. This package operates on the idea that you can build graphs by using a data set, visual marks (called gemos) that represent data points, and a coordinate system. It also takes care of those details like creating legends for graphics.

igraph

This platform comes equipped with a set of tools used for network analysis.

Random Forest

As one of the widely used algorithms in the ML field, this platform can be used to develop multiple decision trees where you'll input observations. When using this platform, data scientists must use numeric variables or factors. You're allowed a maximum of 32 levels when using Random Forest.

reshape2

Shifting data between long and wide formats gets less complicated with reshape2. Using two functions, melt and cast, you can take wide-format data and render it as long-form data (melting) or take long-format data and present it as wide-format data (casting). With reshape2, the format your data takes follows from what you intend to do with your data.

We've covered a plethora of the tools behind ML in this chapter. While these tools are sophisticated ones for marketers, we certainly want you to be aware that not everything in life is 100 percent fool-proof. In the next chapter, we'll drill down into some cautionary tales about ML problems and how you can improve your success and accuracy with ML.



CHAPTER 12

Common Problems with Machine Learning

Machine learning (ML) can provide a great deal of advantages for any marketer as long as marketers use the technology efficiently. Knowing the possible issues and problems companies face can help you avoid the same mistakes and better use ML.

Common Practical Mistakes

Focusing Too Much on Algorithms and Theories

Leave advanced mathematics to the experts. As you embark on a journey with ML, you'll be drawn in to the concepts that build the foundation of science, but you may still be on the other end of results that you won't be able to achieve after learning everything. Fortunately, the experts have already taken care of the more complicated tasks and algorithmic and theoretical challenges. With this help, mastering all the foundational theories along with statistics of an ML project won't be necessary.

Mastering ALL of ML

Once you become an expert in ML, you become a data scientist. For those who are not data scientists, you don't need to master everything about ML. All you have to do is to identify the issues which you will be solving and find the best model resources to help you solve those issues.

For example, for those dealing with basic predictive modeling, you wouldn't need the expertise of a master on natural language processing. Why would you spend time being an expert in the field when you can just master the niches of ML to solve specific problems?

Using Changing or Premade Tools

Previously, we've discussed the best tools such as R Code and Python which data scientists use for making customizable solutions for their projects. For the nonexperts, tools such as Orange and Amazon S3 could already suffice. With these simple but handy tools, we are able to get busy, get working, and get answers quickly. All that is left to do when using these tools is to focus on making analyses.

When you have found that ideal tool to help you solve your problem, don't switch tools. Many developers switch tools as soon as they find new ones in the market. Although trying out other tools may be essential to find your ideal option, you should stick to one tool as soon as you find it. Don't play with other tools as this practice can make you lose track of solving your problem.

Having Algorithms Become Obsolete as Soon as Data Grows

ML algorithms will always require much data when being trained. Often, these ML algorithms will be trained over a particular data set and then used to predict future data, a process which you can't easily anticipate. The previously "accurate" model over a data set may no longer be as accurate as it once was when the set of data changes. For a system that changes slowly, the accuracy may still not be compromised; however, if the system changes rapidly, the ML algorithm will have a lesser accuracy rate given that the past data no longer applies.

Getting Bad Predictions to Come Together With Biases

ML algorithms can pinpoint the specific biases which can cause problems for a business. An example of this problem can occur when a car insurance company tries to predict which client has a high rate of getting into a car accident and tries to strip out the gender preference given that the law does not allow such discrimination. Even without gender as a part of the data set, the algorithm can still determine the gender through correlates and eventually use gender as a predictor form.

With this example, we can draw out two principles. The first you need to impose additional constraints over an algorithm other than accuracy alone. Second, the smarter the algorithm becomes, the more difficulty you'll have controlling it.

Developers always use ML to develop predictors. Such predictors include improving search results and product selections and anticipating the behavior of customers. One reason behind inaccurate predictions may be overfitting, which occurs when the ML algorithm adapts to the noise in its data instead of uncovering the basic signal.

When you want to fit complex models to a small amount of data, you can always do so. Doing so will then allow your complex model to hit every data point, including the random fluctuations. Depending on the amount of data and noise, you can fit a complex model that matches these requirements.

Marketers should always keep these items in mind when dealing with data sets. Make sure that your data is as clean of an inherent bias as possible and overfitting resulting from noise in the data set. You can deal with this concern immediately during the evaluation stage of an ML project while you're looking at the variations between training and test data.

Making the Wrong Assumptions

ML algorithms running over fully automated systems have to be able to deal with missing data points. One popular approach to this issue is using mean value as a replacement for the missing value. This application will provide reliable assumptions about data including the particular data missing at random.

Whether they're being used in automated systems or not, ML algorithms automatically assume that the data is random and representative. However, having random data in a company is not common.

The best way to deal with this issue is to make sure that your data does not come with gaping holes and can deliver a substantial amount of assumptions.

Receiving Bad Recommendations

Recommendation engines are already common today. While some may be reliable, others may not seem to be more accurate. ML algorithms impose what these recommendation engines learn. One example can be seen when a customer's taste changes; the recommendations will already become useless. Experts call this phenomenon "exploitation versus exploration" trade-off. In the event the algorithm tries to exploit what it learned devoid of exploration, it will reinforce the data that it has, will not try to entertain new data, and will become unusable.

To deal with this issue, marketers need to add the varying changes in tastes over time-sensitive niches such as fashion. With this step, you can avoid recommending winter coats to your clients during the summer.

Having Bad Data Convert to Bad Results

Not all data will be relevant and valuable. If data is not well understood, ML results could also provide negative expectations. The initial testing would say that you are right about everything, but when launched, your model becomes disastrous. When creating products, data scientists should initiate tests using unforeseen variables, which include smart attackers, so that they can know about any possible outcome.

Have your ML project start and end with high-quality data. Having garbage within the system automatically converts to garbage over the end of the system.

The background is a solid light blue color. It features several concentric circles and lines in a slightly darker shade of blue. A solid circle is at the top center, and another is at the bottom right. A dashed circle is also visible. Lines connect these circles to the center and to each other, creating a geometric pattern.

COMMON PROBLEMS WITH MACHINE LEARNING

Machine learning
can provide a great
deal of advantages
for any marketer as
long as marketers
use the technology
efficiently.

Machine Learning Goes Wrong

Despite the many success stories with ML, we can also find the failures. While machines are constantly evolving, events can also show us that ML is not as reliable in achieving intelligence which far exceeds that of humans.

Below are a few examples of when ML goes wrong.

Microsoft and the Chatbot Tay

Microsoft set up the chatbot Tay to simulate the image of a teenage girl over Twitter, show the world its most advanced technology, and connect with modern users. The company included what it assumed to be an impenetrable layer of ML and then ran the program over a certain search engine to get responses from its audiences. The developers gave Tay an adolescent personality along with some common one-liners before presenting the program to the online world. Unfortunately, the program didn't perform well with the internet crowd, bashed with racist comments, anti-Semitic ideas, and obscene words from audiences. In the end, Microsoft had shut down the experiment and apologized for the offensive and hurtful tweets.

With this example, it would seem that ML-powered programs are still not as advanced and intelligent as we expect them to be. Research shows that only two tweets were more than enough to bring Tay down and brand it as anti-Semitic. However, in Tay's defense, the words she used were only those taught to her and those from conversations in the internet. In light of this observation, the appropriateness filter was not present in Tay's system.

Uber

Uber has also dealt with the same problem when ML did not work well with them. This ride-sharing app comes with an algorithm which automatically responds to increased demands by increasing its fare rates. During the Martin Place siege over Sydney, the prices quadrupled, leaving criticisms from most of its customers. The app algorithm detected a sudden spike in the demand and alternatively increased its price to draw more drivers to that particular place with high demand. ML understood the demand; however, it could not interpret why the particular increased demand happened. With this example, we can safely say that algorithms need to have a few inputs which allow them to connect to real-world scenarios.

These examples should not discourage a marketer from using ML tools to lessen their workloads. These tales are merely cautionary, common mistakes which marketers should keep in mind when developing ML algorithms and projects. Machine learning models are constantly evolving and the insufficiency can be overcome with exponentially growing real-world data and computation power in the near future.



◦ CHAPTER 13

Machine Learning Quick Start

A certain notion exists that machine learning is only for the experts and is not for people with less knowledge of programming. The people you'll be dealing with are machines, and machines can't make decisions on their own. Your job is to make these machines think. At first, you would think that this task is impossible, but through this guide, you'll understand how machine learning (ML) can be much easier than it sounds.

As with almost every framework in the history of humankind — along the lines of Newton's laws of motion, the law of supply and demand, and others — the particular ideas and concepts of machine learning are likewise achievable and not complicated. Among those challenges that you can face when discovering machine learning are the notations, formulas, and a specific language which you may have never heard about before.

Choosing Your Data Set

Just about any data set can be used for ML purposes, but an efficient model can only be achieved through a well-structured data set.

When thinking about data sets, the better data sets with well-labeled data that features a number of ML algorithms mean you won't have a hard time building your model. However, coming up with simpler algorithms is best despite the volume, especially when you're dealing with a number of data sets, for easier sorting later on. As an example, regularized logistic regression using billions of features can work well despite the substantial numbers.

The more complicated algorithms — which include GBRT (Gradient Boosted Regression Tree), deep neural networks, and random forests — perform better than simpler platforms; however, they can be more expensive to use and train. When you're working with tons of data, you will always make use of linear perceptron and other learning algorithms online. When using any learning algorithms online, you always need to use a stochastic gradient descent.

Perhaps always choosing a good selection of features would be safe, such as the essential tools to learn about your data. Also a features selection serves more likely an art instead of a science. The selection process could be complicated; however, it could be eased out if you work on removing the unpromising ones first to narrow down your options.

Model Selection

To find the best model that suits your needs, you need to keep in mind a few of the following considerations:

Accuracy

While accuracy is important, you don't always have to go with the most accurate answer. This can happen when you feel that you have a model that is producing a high bias or high variance. Sometimes an approximation will do depending on the use. Opting for the approximation, you'll be able to reduce processing time and you avoid overfitting.

Training Time

Each algorithm will require varying training times for instructing its models. As we consider the training time, we should note that this time is closely related to the accuracy — as training time increases, so does the accuracy. Also keep in mind that other algorithms can be more sensitive than others and may require more training time.

Linear Classification

Linearity is common in machine learning algorithms. A line can possibly separate classes as assumed by linear classification algorithms. These classes can be composed of support vector and logistic regression machines. Linear classification algorithms work off of assumptions which can lead to the previously mentioned high bias and variance issues. These algorithms are still commonly implemented first because they're algorithmically simple and quick to train.

Number of Parameters

Data scientists use precise measurements to set up an algorithm called parameters. These parameters are in the form of numbers which provide changes on the behavior of algorithms. These behavioral changes can include the number of iterations, error tolerances, or options among variants about the behavior of algorithms. Sometimes, if you want to have the best settings, you may often need to adjust the accuracy and the training time. The algorithms having the largest number of parameters will always require more trial-and-error combinations and tests.

How You Can Train Your Model

A substantial amount of repetition will always be required with training a model to achieve a high level of accuracy out of a selected model.

As a first step, you'll have to select the best model suited for your particular project. Previously, we discussed the pros and cons for various models. With the information provided, the steps would cover the following:

Data Set Partitioning for Testing and Training

Also called the process of "data partitioning," you'll find various options from the variety of tools and languages to help you choose which data points to divide over training and testing data sets. Below is an example of partitioning data sets over R.

```
1  # By default R will come with a number of datasets
2  data = mtcars
3  dim(data) # 32 11
4
5  #Sample Indexes
6  indexes = sample(1:nrow(data), size=0.2*nrow(data))
7
8  # Split data
9  test = data[indexes,]
10 dim(test) # 6 11
11 train = data[-indexes,]
12 dim(train) # 26 11
```

Eliminating Biases in Data Through Cross-Validation

Performing a cross-validation out of your model will give you a guarantee for an accurately correct model that doesn't pick too much noise or, in simpler terms, low on variance and bias. Cross-validation becomes a reliable technique if you want to track the effectiveness of your particular model especially within cases where you need to ease overfitting.

You will find various cross-validation techniques available where the holdout method could be the most basic form that makes use of the partitioning method. Function approximators will only fit a certain function through training sets and afterward will be prompted to make a prediction on the output values for the particular data within the testing set. One good feature about this method is its quickness to compute, a benefit recommended for the residual method.

Maintain the Accuracy of Your Model

Using the data provided after a cross-validation process, let's check out the errors and devise a way to fix these errors.

Repeat These Steps Until You Obtain the Desired Accuracy

Gather and Test

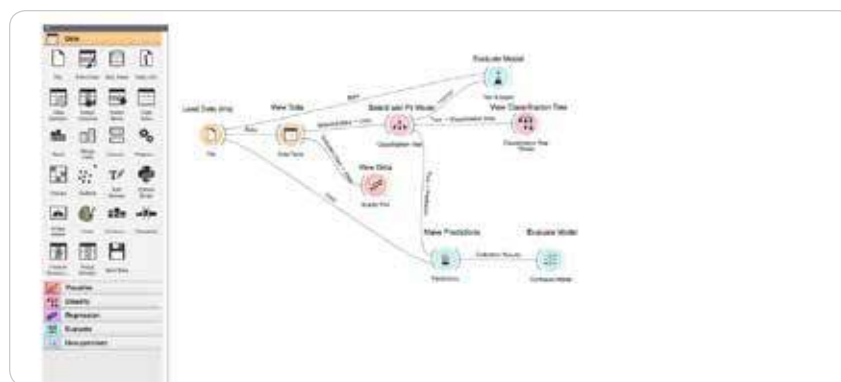
When on the lookout for the basic data sets to learn with and gain comfort with, experts would recommend multivariate data sets provided by Ronald Fisher, a British statistician who, in 1936, introduced the Iris Flower Data Set. This data set studied four measured features across three different species of iris. It comes with 50 samples for every species.

Moving on to the next steps as soon as we acquire our data set:

Basic Predictive Model Out of Orange

Let's go with a pair of basic predictive models within the Orange canvas along with their specific Jupyter notebooks.

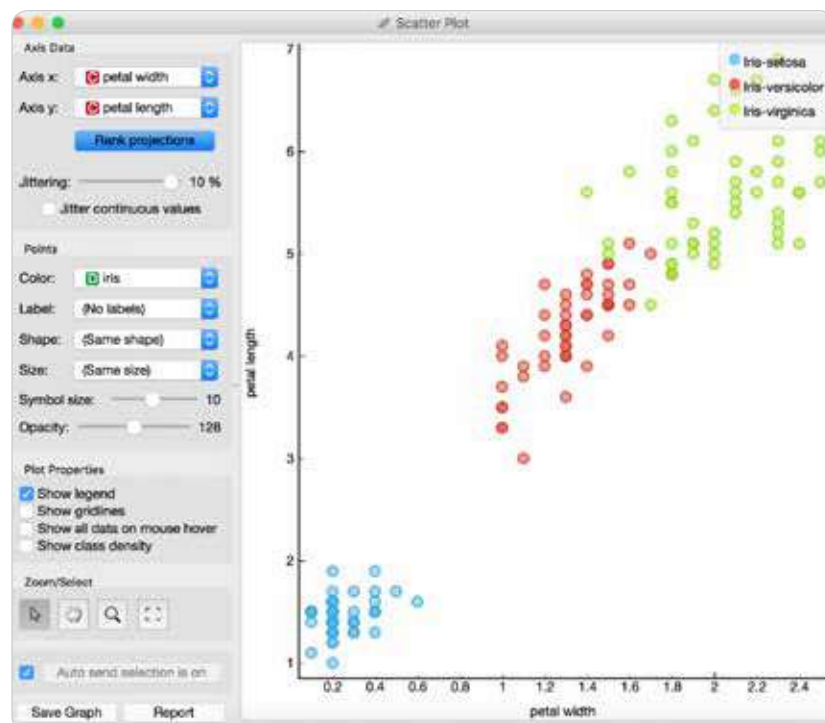
Initially, let's examine our Simple Predictive Model – Part 1 notebook. After we do, let's proceed with creating the model over the Orange Canvas. Below is a diagram for predicting results coming from the Iris data set by using a classification from within Orange.



The toolbar over the left part of the canvas provides you with more than 100 widgets that you can use by dragging the toolbar over the canvas. Reading the schema from left to right, and details from widget

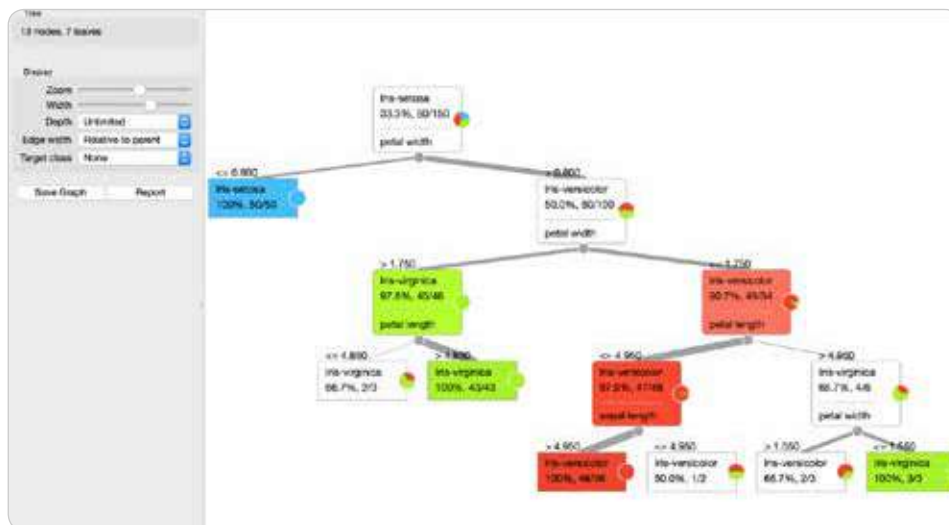
to widget out of the pipelines, the Iris data set can be seen over a variety of widgets once it is loaded in. By clicking the simple data table along with the scatter plot, we'll have the following displayed:

	sepal length	sepal width	petal length	petal width	iris
1	5.100	3.500	1.400	0.200	Iris-setosa
2	4.900	3.000	1.400	0.200	Iris-setosa
3	4.700	3.200	1.300	0.200	Iris-setosa
4	4.600	3.100	1.500	0.200	Iris-setosa
5	5.000	3.600	1.400	0.200	Iris-setosa
6	5.400	3.900	1.700	0.400	Iris-setosa
7	4.600	3.400	1.400	0.300	Iris-setosa
8	5.000	3.400	1.500	0.200	Iris-setosa
9	4.400	2.900	1.400	0.200	Iris-setosa
10	4.900	3.100	1.500	0.100	Iris-setosa
11	5.400	3.700	1.500	0.200	Iris-setosa
12	4.800	3.400	1.600	0.200	Iris-setosa
13	4.800	3.000	1.400	0.100	Iris-setosa
14	4.300	3.000	1.100	0.100	Iris-setosa
15	5.800	4.000	1.200	0.200	Iris-setosa
16	5.700	4.400	1.500	0.400	Iris-setosa
17	5.400	3.900	1.300	0.400	Iris-setosa
18	5.100	3.500	1.400	0.300	Iris-setosa
19	5.700	3.800	1.700	0.300	Iris-setosa
20	5.100	3.800	1.500	0.300	Iris-setosa
21	5.400	3.400	1.700	0.200	Iris-setosa
22	5.100	3.700	1.500	0.400	Iris-setosa
23	4.600	3.600	1.000	0.200	Iris-setosa
24	5.100	3.300	1.700	0.500	Iris-setosa
25	4.800	3.400	1.900	0.200	Iris-setosa
26	5.000	3.000	1.600	0.200	Iris-setosa
27	5.000	3.400	1.600	0.400	Iris-setosa
28	5.200	3.500	1.500	0.200	Iris-setosa
29	5.200	3.400	1.400	0.200	Iris-setosa

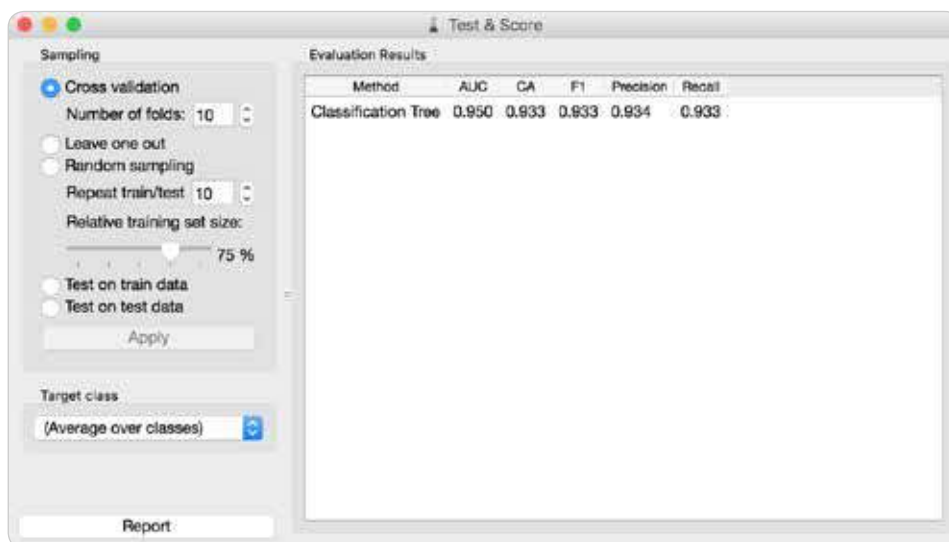


Using three widgets, we can already have a better picture of our data. The scatter plot provides us with the option “Rank Projections,” which tracks the best step to view the specific data. In this regard, having “Petal Width vs. Petal Length” provides a quick pattern on the width of the flower’s petal and the type of Iris flower studied.

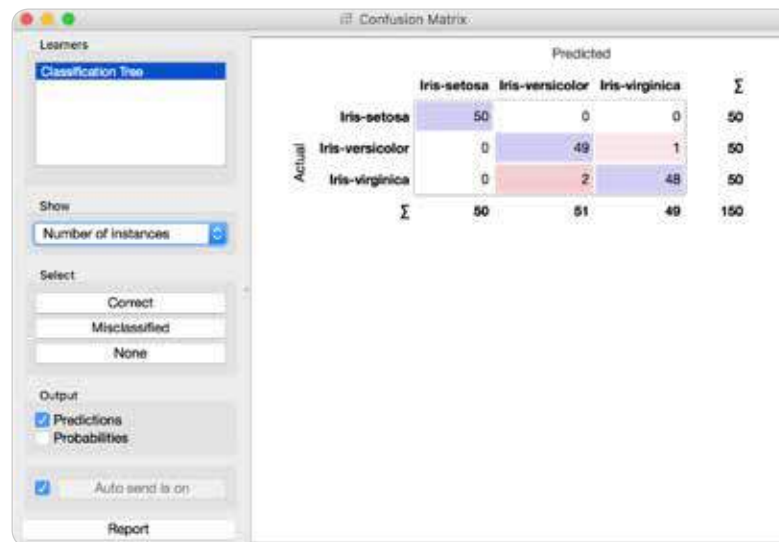
Looking at the way we build the predictive model, we connect the data over a classification tree widget to examine through the viewer widget.



Here the predictive model becomes noticeable and thus allows us to connect the model plus data over to the “Test and Score” along with the “Predictions” widgets. This process illustrates how the classification tree performs.



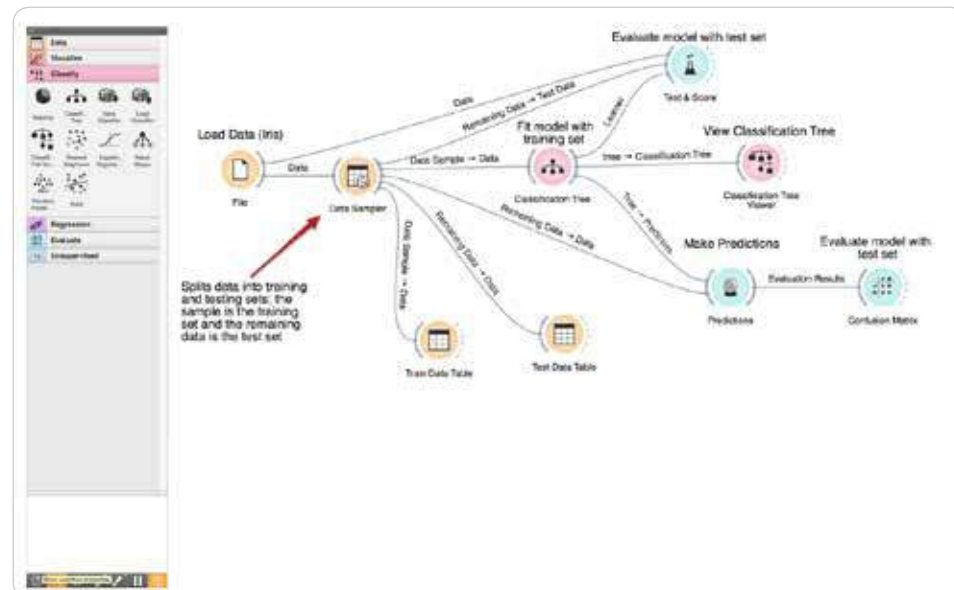
The Predictions widget will be able to predict the specific type of iris flower through the set of input data. Using a confusion matrix, we can find the most accurate predictions.



Hold Out Test Over Orange to Build Predictive Model

In this next example for predictive modeling, we'll be having a slightly complicated model out of holding out a particular test set. Through this effort, we can use varying data sets to test and train the model, thus disabling overfitting.

Using Orange Canvas, we can now proceed with building the same predictive model for a better view of what we are building.



Are you with us so far? Now let's take your knowledge to the next level and apply what you've learned in this chapter. We'll next consider some more resources you can use to get you started with a more complex ML project.



© 2017 iPULLRANK

Visit the online version of this guide and
more at ipullrank.com