Quick-Start Guide to Machine Learning and NexusMax

For Analysts, Product Managers, Business Intelligence Professionals

Introduction

Digital Transformation is at the core of every enterprise today. Digitization, Robotic Process Automation and Artificial Intelligence are the foundational pillars of this transformation. Enterprises are producing more data than ever before in their history. The availability of inexpensive cloud computing and storage have made it possible to apply Artificial Intelligence (AI) and Machine Learning (ML) to derive uncommon insights from this data, drive intelligent actions and deliver impressive results. In traditional software development inputs are transformed to outputs through algorithms defined and programmed by software developers. On the other hand, ML and AI engines take inputs and empirical outputs and derive the algorithm that will transform inputs into output; these algorithms are called models. In order to develop effective models, we would develop hypotheses, run experiments, and iterate to arrive at the most optimal model that can transform inputs into observed outputs.

The value that machine learning brings is that it can learn from historical and current data and predict the outcome of the future. Whether it is predicting changes in the stock market, giving users directions through maps, or even predicting the price of real-estate through a set of features, machine learning algorithms and models are being used across several industries to improve processes, reduce costs, optimize finances and also improve the lives of the employees within the organization itself.

About this Document

This document aims at helping you as a business user to define *what* Artificial Intelligence is, what the *benefits* of using ML are and guide you *how* it can be used for your specific business needs.

NexusMax is an integrated self-service AI platform that enables business users, business scientists and data scientists to harness the power of AI and Machine Learning, to solve their business problems irrespective of their expertise in ML. NexusMax delivers an intuitive and collaborative drag and drop experience on top of the best-in-class data and AI technologies to deliver results in hours and not days or weeks.

We will be taking a deeper look at the power of self-service NexusMax AI platform in later sections of the document and how it can help business users achieve their business goals, although they have familiarity with but not mastery of the underlying complexities of the science and technology.

1. What is Artificial Intelligence and Machine Learning?

A brief History of Artificial Intelligence

Artificial intelligence is not a new technology. It has been around for decades, even though it has caught wind in recent years with the promise of conversational chat interfaces like ChatGPT and its applications. It has been touted as the "next big thing" to unlock productivity and the world is seeing early proof from services like GitHub CoPilot, Microsoft Office365 CoPilot, etc., The current state of Artificial Intelligence is more suitable for solving very specific problems (Narrow AI) and is not ready to solve broad amorphous problems (General AI).

While we are still far away from replacing the wonders of the human brain, AI applications have started to fill our daily lives and power our electronic devices, home automation systems, security alarms and so forth, the list goes on.

As humans we have always tried to understand the world around us and bend nature to meet our goals; we have relied on many external tools to amplify/supplement our brain's capacity. The abacus was probably one such tool invented about 5000 to 6000 years ago to help make mathematical calculations easy, and it did have its lasting effect until recent times. Many kids still attend abacus lessons to get better at math. It wasn't until the late 1960's that the first machines that could add and subtract were invented and we have come a long way from the calculator being a great invention.

The words *artificial* and *intelligence* were first coined on August 31, 1955, when professor John McCarthy from Dartmouth College, together with M.L Minsky from Harvard University, N. Rochester from IBM, and C. E. Shannon from Bell Telephone Laboratories, asked the Rockefeller Foundation to fund a summer of research on artificial intelligence[1] which gave rise to two primary groups of artificial intelligence that will be beneficial to define, for a deeper understanding:

- *General AI (strong AI)* – An artificial intelligence capable of performing every task that is presented to it, with minimal human supervision and assistance. This is the holy grail of artificial intelligence, think *Skynet* from the *Terminator* movies. This is a distant dream, and quite possibly impossible to achieve.
- *Narrow AI* – An artificial intelligence capable of solving a single, well defined task that requires historical data, a domain expert (computer programmer), a training model which the *AI* learns from to better predict data and a set of features upon which the algorithm is built. This is what drives our current *AI* revolution, and it is known as *Machine Learning*.

[1] *Zero to AI: A non-technical, hype free guide to prospering in the AI era, Gianluca Mauro & Nicolò Valigi, May 2020*

2. Understanding the engine behind AI: Machine Learning

The simplest definition to *machine learning* dates back to 1959 when American AI pioneer Arthur Samuel defined it as:

> *"Machine learning is the field of study that gives computers the ability to learn without explicitly being programmed".*

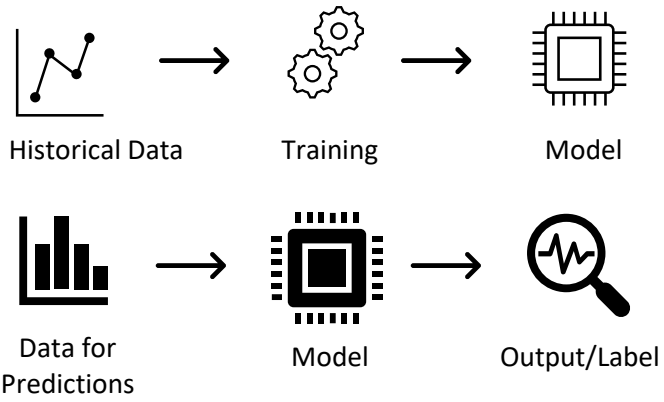
Machine learning is achieved by relying on historical *Data.* Images, spreadsheets, words, videos are all examples of Data. And ML aims to derive patterns from all this data and uses this knowledge to predict outcomes. Data is not only produced by humans, but also by devices.

The main goal of ML is to predict results by learning patterns from incoming data (inputs and results) available on a given topic. In traditional computing, an algorithm is first created based on the understanding of how something is supposed to work, and then inputs are fed to the algorithm to generate output. With ML, input and output data from past observations are fed as inputs, and the computer generates an algorithm via a process called Training. More later!

Now it's time to define some of the common technical terms that have been used in trying to understand what machine learning is.

- *Historical Data* – Collected data about past events pertaining to a particular subject. This includes input data and output results that were observed.
- *Data for predictions* – Data that is split into training and testing, usually in 80-20 ratio.
- *Features* – A set of characteristics of an object that the algorithm can learn from.
- *Label* – The output or target we want the algorithm to predict.
- *Training* – A phase in which the machine learning algorithm is fed with Historical Data and learns from them.
- *Model* – The output of the training phase – a self-contained computer program that can predict an output (*Label)*, given a set of input data.

The illustration below should give you an idea of the core workings of ML, with two steps:



3. A Case Study in Regression: Steps involved in Machine Learning

It is time to take a deeper look into a case study and internalize all the processes that go into a machine learning model in order to truly understand how machine learning works, and later in the document we will also discuss Use Cases, which will give you a better idea of how it can be integrated into your specific business. But for now, let's look at a simple example, below.

We are asked to come up with an ML model that can predict *how far a person can run in one hour*. We will start with defining attributes that determine how far a person can run: *Age in Years, Average pace in minutes/mile, elevation gain in meters, Heart rate in beats/min and Distance in miles.*

1. Gathering the Data

The first step is gathering historical data. You can either have yourself and a bunch of friends go out for a run for an hour and measure all five features along the way for each of you.

For now, let us use data that we generated for you, below:

Age	Average Pace (minutes/mile)	Elevation Gain	Heart Rate(beats/minute)	Distance
23 years	13.23 min/mile	182.4	143 bpm	1.61 miles
27 years	14.34 min/mile	175.8	154 bpm	3.21 miles
30 years	12.99 min/mile	199	156 bpm	4.83 miles
34 years	14.58 min/mile	178.2	143 bpm	1.61 miles
37 years	13.20 min/mile	187	155 bpm	3.22 miles

In our example, all of this is historical data which will be used for training. The first 4 columns are called ***features*** which are inputs and Distance is the Output/Result. We will use this historical data in training an ***ML model***.

2. Data Preparation

After we have collected our ***training data***, we can now move onto the next crucial step in the ML process: *Data preparation* – this is where all the data we have gathered will be loaded into a suitable place and used to train our machine learning model.

Making visual representations of the data we have collected is a good way to gather relevant relationships between different variables you can take advantage of, as well as show if there are any imbalances in the data. For example, if there is more data on Distance covered approximately 1.6 miles, then the model will end up predicting an average distance covered as 1.6 miles since there is less data on instances running for lesser or greater miles based on our imbalanced data. We can achieve this by making bar charts representing the amount of data.

We will also need to split our data into two parts. The first part which is used in our training *model* will make up majority of the dataset. The second part of the data will be used for evaluating the performance of our *model*. We don't want the same data we used in our *model* for the evaluation segment too. This would just make the model memorize the questions and we end up with an inefficient *model*.

Sometimes when there are projects with even more data, we would expect even further *data cleaning* like normalization, error correction and more. But for the sake of our simple project, so we don't have any more data preparation needs and can move on.

3. Choosing a Model

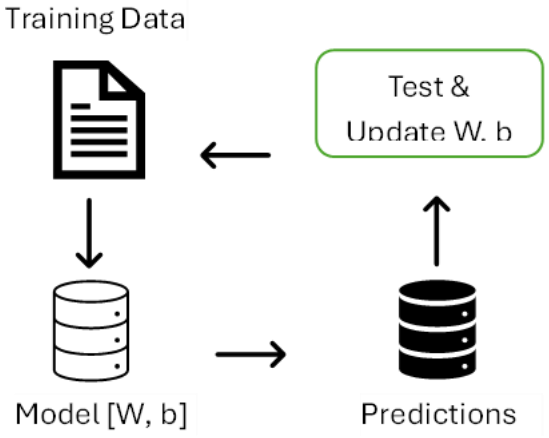
The next step in our workflow would be to choose an appropriate model for our machine learning algorithm. There are many models that researchers and data scientists have created over the years. Some are suitable for image processing, some for text, music and numerical data. In our case since we just have four *features*, age (years), average pace (mins/mile), elevation gain and heart rate (bpm). Therefore, a simple *multi-variate linear* model should do the trick.

4. Training the model

Now we move onto a crucial step in building our machine learning algorithm – the *training.* In this step we will use our data to incrementally improve our *model's* ability to predict the output as Distance in 1 hour. Think of it as someone learning how to ride a bike. The first time around they are really bad at it, most often times they end up falling from the bike and not being able to maintain the balance to move forward. But once they practice enough and get in a few hours of cycling, we get a resultant biker who is quite adept at cycling. The act of cycling and reacting to real-world data has adapted their cycling abilities, honing their skills.

We will apply a similar approach to training our *model*, but at a much smaller scale with a simple formula for a straight line $y = m*x + b;$ where x is the input, m is the slope of the line, b is the is the *y-intercept* and y is the value of the line at position x. The values at our disposal that we can tamper or *train* our model with are m and b. There are no other values in the equation for the line, expect for the input x and the output y.

In machine learning, there are many values for m ($m_1, m_2 \ldots m_n$) since there are many *features* in the dataset. All the values of m are grouped into a matrix denoted *W for* a "weights" matrix. Similarly, for the matrix b, the values are grouped together, and this is called "biases".



The training process involves initializing some random values for W and b and attempting to predict the output/target with those values, in our case the output is distance jogged in 1 hour. At the first few attempts, as can be expected, the model does quite poorly. But then, as this in iterative process, we can compare the values of our predictions to the output it should have produced, and then update the values of W and B such that we have better predictions.

This process then repeats itself, and each iteration or cycle of updating the Weights or biases is called one training "step". Let's take a deeper look at what that means in our case, more specifically, for our dataset.

When we first start the training process, it's like we drew a random line through the data we have for jogging. Then as each step of the training progresses, the line moves, step by step, closer to our ideal prediction of the distance jogged in 1 hour. The first few trials may give us bad results, but as the training takes place iteratively the algorithm gets better at predictions.

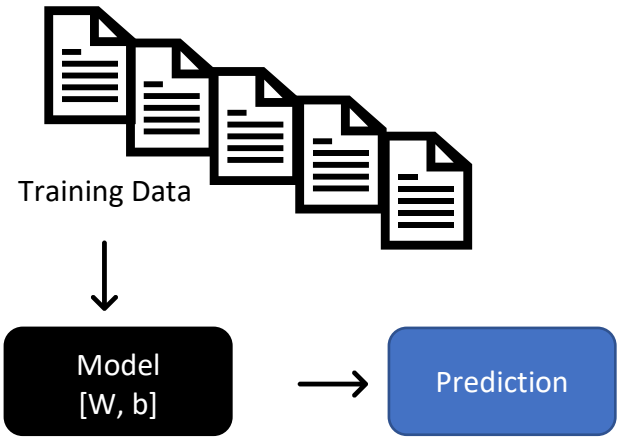
5. Evaluation

After the training is complete, we want to find out whether our model is working, using Evaluation steps. This is where the dataset that we set aside earlier will come into play. Evaluation allows us to test the model with data that was never before used for training an algorithm. This allows us to see how our model might perform against data that it has not seen yet. This is supposed to be a representation of how the model would function in a real-world setting.

6. Parameter Tuning

Once the evaluation is complete, it is possible that you want to make some improvements to your training in any way possible. We can achieve this by *parameter tuning.* There were a few parameters we implicitly assumed when we did our training and it is time to test those assumptions and try other values.

One example is how many times we run through the training dataset during the training phase. This means we have to run the numerical data through the *model* a couple of times, and the more times it runs on this data we can get a more accurate result on the next try. It's like giving the model some practice runs, to get a value that's more suitable for us.

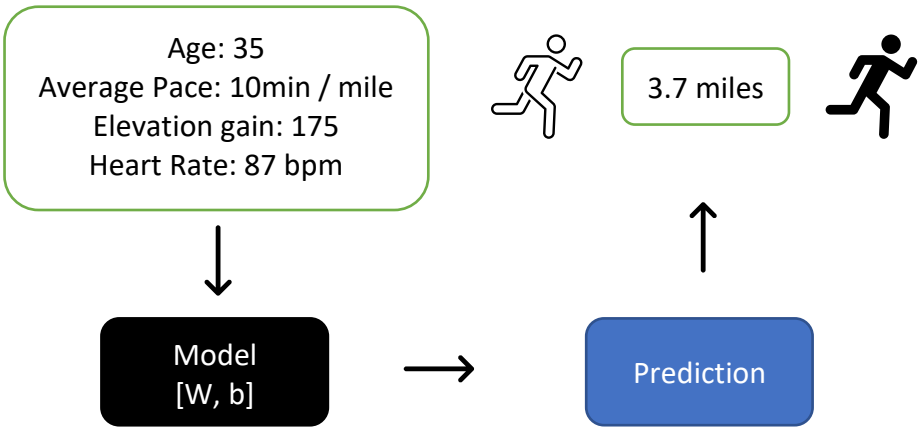


There is another parameter called "*learning rate"* which essentially defines how far we shift the line during each step, based on information from the previous training step. This plays a role in determining how accurate our model can become, and how long it takes for the training to occur. The parameters in this step are known as "*hyperparameters"* which often times have a value between 0.0 and 1.0. There are many adjustments to consider in the tuning process, and how well one can adjust them will determine the efficiency of the model we have trained.

After we have trained our model and tuned our hyperparameters to our content, guided by the evaluation step, it is time to move onto the more important goal here: prediction! This is where we find out how well our machine learning algorithm is working.

7. Prediction

Machine Learning is using data to answer questions[2]. This is the step where prediction or inference takes place. We get to see the answer to our question: *How far can a person jog in 1 hour?* This is where the value of machine learning is realized.



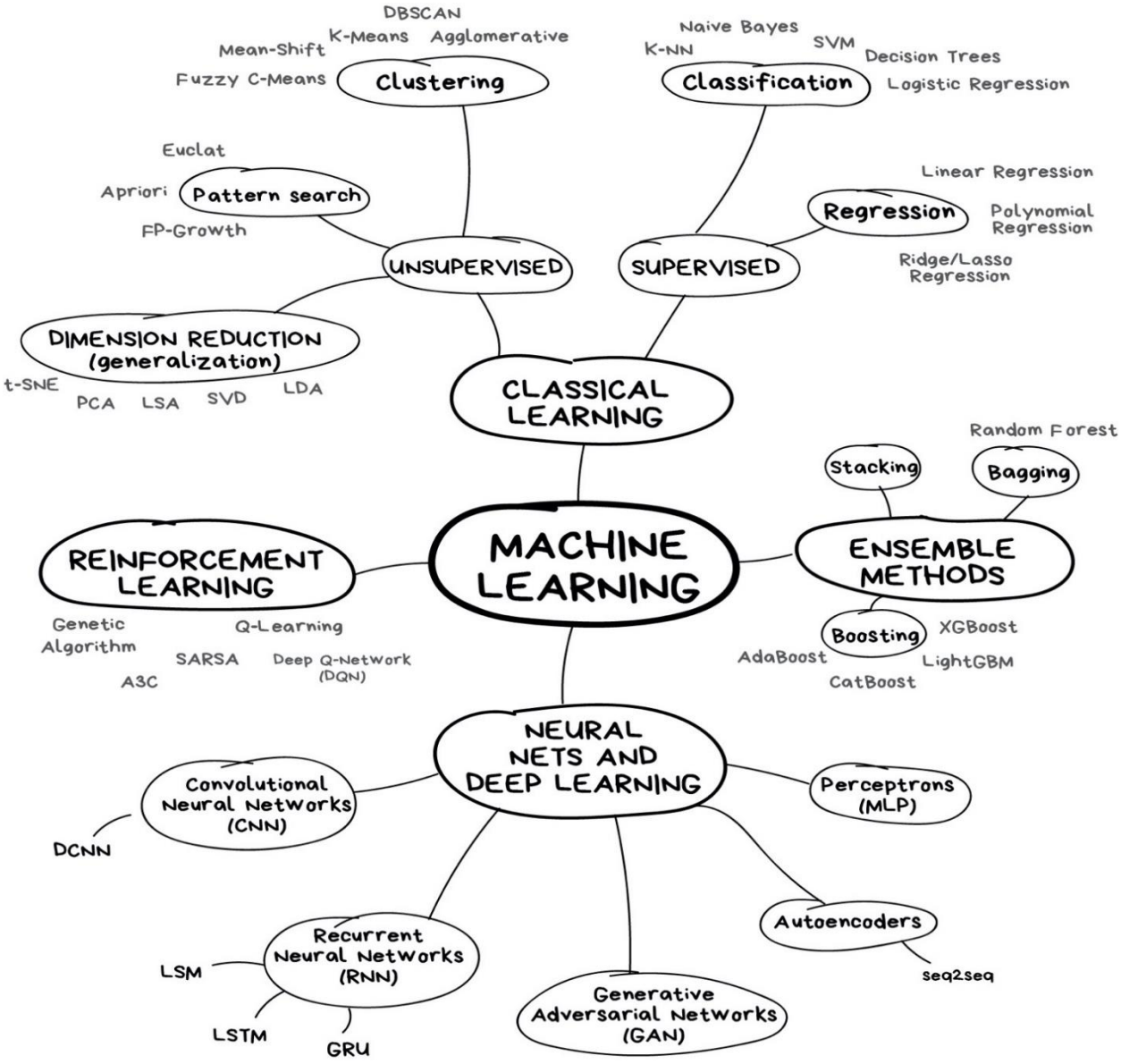
As can be seen in the representation above, after following the 7 steps involved in building a machine learning algorithm, we will have a model that can predict a certain output based on the initial values or *features* we provide. Of course, on the surface it looks a little easy but often times it's hard to find data on particular topics and if the problem requires complex data then we cannot use the same algorithm (*multi-variate linear regression)* in every case. This is where machine learning splits into many variants or types which we will be focusing on the next section. But for now, let's sum up the key steps involved in building the machine learning algorithm we used for our case study in figuring out how far a jogger would run in 1 hour:

- Data Gathering
- Data preparation
- Choosing a *Model*
- Training the *Model*
- Evaluation
- Parameter Tuning
- Prediction

This should give you a big picture when it comes to choosing the right model for your machine learning algorithm. But as mentioned before, what if the data does not fit the model you have in mind? Let's look at the different streams of machine learning to get perspective.

[2] *Yufeng G, Developer and Advocate for @googlecloud, Machine Learning August 31st, 2017*

4. Types of Machine Learning: The Map of the Machine Learning World



Types of Machine Learning: Always important to remember -there is no one way to solve a problem in the machine learning world. There are always several algorithms that fit, but you have to choose the one that fits best.

As represented above, Machine learning is broadly categorized into 3 main types: *Classical Machine Learning* which also has two sub-groups called *Supervised Learning* and *Unsupervised Learning,* which is the original way of using ML. This also is the way we do things at *NexusMax*, which we will discuss in further sections of the document (not there yet). Then there is *Reinforcement Learning, and Deep learning and Neural Networks* which are more advanced. Let's focus on *Classical Machine learning* for now.

Classical Machine Learning: Supervised vs Unsupervised Learning

Classical machine learning is often divided into two major categories – **Supervised** & **Unsupervised Learning** and the terms are pretty self-explanatory.

In the first category, the machine learning algorithm has a supervisor or trainer that gives the machine all the answers in advance, and the algorithm learns from this data to predict future outcomes. Like for instance if we want the machine to tell whether it's an apple or banana in the picture. The trainer has already labelled the data into apples and bananas and the machine learns from this data, one by one and predicts the next outcome.

Unsupervised learning is as it sounds, there is not supervisor and the machine is left on its own to figure out from a pile of data. The data is not labelled, and the machine tries to find patterns on its own. We will talk more about unsupervised learning in the following sections.

5. Supervised Learning

As it is quite obvious, when there is a trainer or supervisor the machine learning algorithm gives better predictions, simply because there is much more data involved that is labeled by the supervisor. It is also more widely used in real-life tasks. There are two types of supervised machine learning: **classification – an objects category prediction and regression – prediction of a specific point on a numerical axis,** much like the case study we discussed about in the previous section.

- Classification

Classification is the process of splitting objects based on one of the attributes known beforehand. For example, separating socks based on length, documents based on template, and music by genre. One of the first use cases of classification was to detect spam emails, although that is a very rudimental problem these days, it was where it started. Some other areas of use for classification are churn predictions *&* *fraud detection.* Machine learning is mostly about classifying things into categories based on certain *attributes* and over time a few algorithms have been developed to tackle these problems.

- Naïve Bayes Algorithm

In spam filtering the Naive Bayes algorithm was used widely. The machine counts the number of times a word that is potentially a red flag is mentioned, for instance “money”, in spam and normal email. Then it multiplies both probabilities using Bayes theorem, sums the results and filters out spam email. The Naïve bayes theorem is as follows:

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}$$

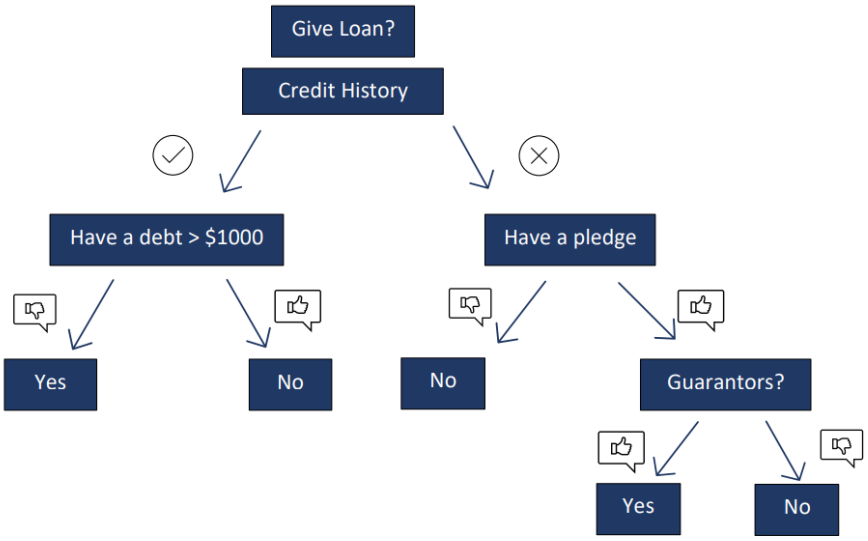
After discovering their mail gets filtered, spammers learned how to deal with spam filters by adding in lots of “safe” or “good” words that aren’t suspicious. This was called Bayesian poisoning, which is basically a workaround to confuse the machine to classify spam as safe.

- Decision Tree Algorithm

Another algorithm widely used in the supervised learning division of ML is *Decision Trees*. This is a great algorithm that can combine numeric data with *yes* or *no* questions based on how the algorithm is programmed. This is primarily also what banks use to approve a certain individual for loan approvals, say based on their credit score or loan repayment history.

The machine presents the banker with certain questions about the candidate applying for credit, say about age, education, occupation and more importantly whether he/she has paid her debts in the past. And based on the answers that the person gives, the tree leads to a simple *yes/no* answer at the end which helps the bank decide.

Decision trees have been used in many ways, and more recently they are also being used in the Gaming industry. The player is presented with a series of questions, which lead to different outcomes during the whole course of the game. Many AAA titles like *The Witcher 3: Wild Hunt, Stars Wars: Jedi Fallen Order* and *Detroit: Become Human* have neatly integrated this algorithm into their video games to provide gamers with a more unique and immersive experience. Here is an example of how a simplified version of a Decision Tree looks like:



- Regression

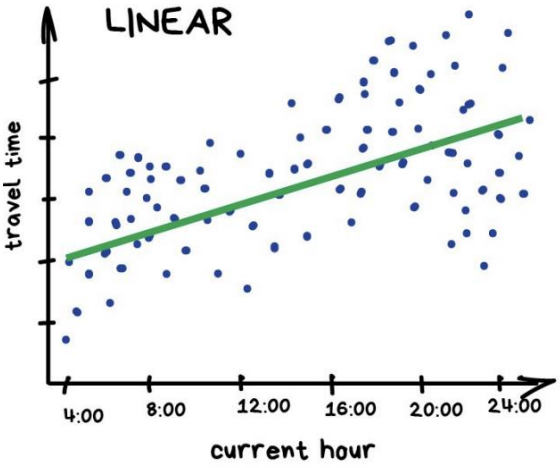
Regression is basically like classification, but instead of forecasting categories here a numerical value is predicted. It works best when variables of our interest to predict changes due to change in another variable, much like the *Jogging Distance* calculation algorithm we have discussed in case study of this document. Some other examples of regression-based models are predicting the value of a car based on mileage, traffic by the time of day, demand volume by company growth etc.

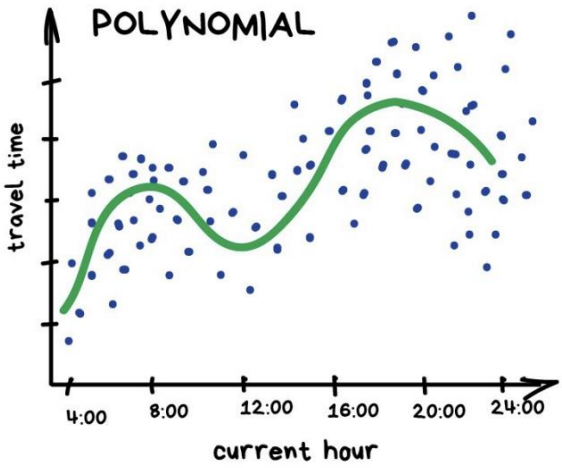
From the case study we must have a decent understanding of what Regression is, but let's look at another example before that given below is a representation of the two types of regression: *Liner regression* and *Polynomial Regression.*

Let's say we have some data on the traffic times by the hour, and we use this data to make a plot. As we know, regression tries to *predict a continuous numerical value* Y based on a discrete/ continuous variable X. A continuous variable is something that always changes over time, like the time it takes to travel. A discrete variable is something that has a finite change over time, like the number of hours in the day.

[3]So, using the data we have on traffic times we can plot a regression plot that looks something like this:

[3] Machine Learning for Everyone, Steve Nouri, blog





Graphically we are trying to estimate the *Line of Best Fit* which can be drawn in two ways as can be seen above. If the line is straight is a *Linear Regression* and the equation is as below, where x is our current hour and y is the value to be predicted based on x which is travel time:
y = b0+b1x

However, in the in the right image it is observed that the line is curved. For cases, where data is changes in a non-linear way and the line is curved, it is a *Polynomial Regression.* The equation for changes as below:
$y = b_0 + b_1 x + b_2 x^2 + b_3 x^3 + \ldots$

6. Unsupervised Learning

To find the underlying structure of a dataset, what do you do? How do you summarize it and group it most effectively? And how can you compress the data so that it does not take up too much space? All these questions can be answered using *Unsupervised Learning*, which is called *"unsupervised"* because there is no trainer or teacher overseeing the process and also since the data used here is **unlabeled** data. This means that we have a dataset about an unknown source, and it's up to the machine to figure out and categorize this data.
There is no Y component in unsupervised learning.
There are mainly two types of Unsupervised machine learning: clustering and association.

Clustering

This type of unsupervised learning describes techniques that attempt to divide data into groups that are clustered closet together. An example of clustering is grouping customers by how much they typically spend on a certain product. Some of the algorithms used in clustering are K-means clustering, Mean-shift or DBSCAN.

An excellent example of clustering is to increase the sales of a store, we would like to strategize marketing techniques based on customer's preferences. Using clustering, we can devise a strategy based on purchasing patterns of each group and share relevant marketing advertisements.

Dimensionality Reduction (Generalization)

Dimensionality reduction is a lot like compression. It is mainly about trying to reduce the complexity of data whilst keeping the core structure in-tact. Let's say you have a sample image of the size 120 x 120 x 4 pixels (length x width x RGB value), that's 57,600 dimensions of data. If you are able to reduce the dimensions of the space in which this data resides without compromising on the core meaningful data, then you have done a good job with dimension reduction. That's the main aim, anyway. This can be achieved using two main forms of techniques: ***principal component analysis*** and ***singular value decomposition.***

Principal component analysis (PCA) identifies uncorrelated variables called principal components from large sets of data. The objective 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) combines information from several vectors and forms basis vectors that can explain most of the variances in data.

Unsupervised learning is often used to preprocess data to compress it in a way that retains the core structure with PCA and SVD before feeding it to a Deep Neural Net.

7. Understanding decisions made by the machine learning models

EXPLANABILITY INTRODUCTION

Machine learning models take on real-world tasks that require safety measures and testing. An algorithm trains a model that produces the predictions. Each step can be evaluated in terms of transparency or interpretability.

After once, the model is trained, it is important to explain the predictions of a machine learning model. A machine or algorithm that explains its predictions will find more acceptance. Explanations are used to manage social interactions. Machines must “persuade” us, so that they can achieve their intended goal. And Machine learning models can only be debugged and audited when they can be interpreted. By default, machine learning models pick up biases from the training data

And **Interpretability** is a useful debugging tool for detecting bias in machine learning models. When a model is used in a product, things can go wrong. An interpretation for an erroneous prediction helps to understand the cause of the error.

Methods for machine learning interpretability can be classified according to various criteria.
Local or global? Does the interpretation method explain an individual prediction or the entire model behavior? Or is the scope somewhere in between?

You could describe a model as interpretable if you can comprehend the entire model at once. Global model interpretability helps to understand the distribution of your target outcome based on the features. While Local Interpretability helps to understand predictions for a single instance.

Local surrogate models are interpretable models that are used to explain individual predictions of black box machine learning models. SHALP and LIME are algorithms that generate these explanations. Surrogate models are trained to approximate the predictions of the underlying black box model. Instead of training a global surrogate model, LIME focuses on training local surrogate models to explain individual predictions.

LOCAL INTERPRETABILITY

The goal is to understand why the machine learning model made a certain prediction. LIME tests what happens to the predictions when you give variations of your data into the machine learning model. LIME generates a new dataset consisting of perturbed samples and the corresponding predictions of the black box model. On this new dataset LIME then trains an interpretable model, which is weighted by the proximity of the sampled instances to the instance of interest. The interpretable model can be anything from Lasso to a decision tree.

Here explanation relates to the feature values of an instance to its predictions in a humanly understandable way. Or even relates to set of data instances to its predictions.

For example, the intrinsically interpretable models such as a linear regression is already equipped with explanation in the form of weights assigned to each feature. However, Surrogate models such as classification, clustering or time series and so on, there are algorithms to generate explanations. Linear regression, logistic regression and the decision tree are commonly used interpretable models.

GLOBAL INTERPRETABILITY

Global method explains the average behavior of a machine learning model. It is often expressed as expected values based on the distribution of the data. For instance, the partial dependence plot, a

feature effect plot, is the expected prediction when all other features are marginalized out. Since global interpretation methods describe average behavior, they are particularly useful when the modeler wants to understand the general mechanisms in the data or debug a model.

8. Reinforcement Learning

Reinforcement learning is one of the most advanced streams in machine learning that closely relates to the world's version of *artificial intelligence,* you know the argument where robots take over the world? That is because it is a behavioral learning method. Let's say we have a robot thrown into a maze and we have it figure its way out on its own. It would make a lot of errors for sure, but somehow it would come out in once piece at the end. The system isn't trained on a sample data set but rather learns to deal with real world problems on the fly. Therefore, a sequence of successful decisions will result in the process being "reinforced" as it best solves the problem at hand, by trial and error.

The most effective way to train these robots is by exposing them to a virtual world first, where it can make all the mistakes it wants before figuring out an optimal solution to the obstacles around it. This is how Tesla's auto-pilot feature is also trained. That's a great segway to how *self-driving cars* are using the reinforcement learning algorithm to train their models. The idea of a self-driving car is a very complex one, since the system faces so many real-world scenarios that are unpredictable that it is bound to make errors. But this is how the system learns, and even with the complex scenarios the algorithm can be optimized over time to find ways to adapt to the state where actions are *rewarded.* A simple way to think about reinforcement learning is the way an animal is trained to take actions based on rewards. If the dog gets a treat every time it fetches on command, he will take this action each time. Take a look at this example about a robot mouse trying to find its way out of a maze with rewards along the way that is so well explained by *Machine Learning for Humans*.

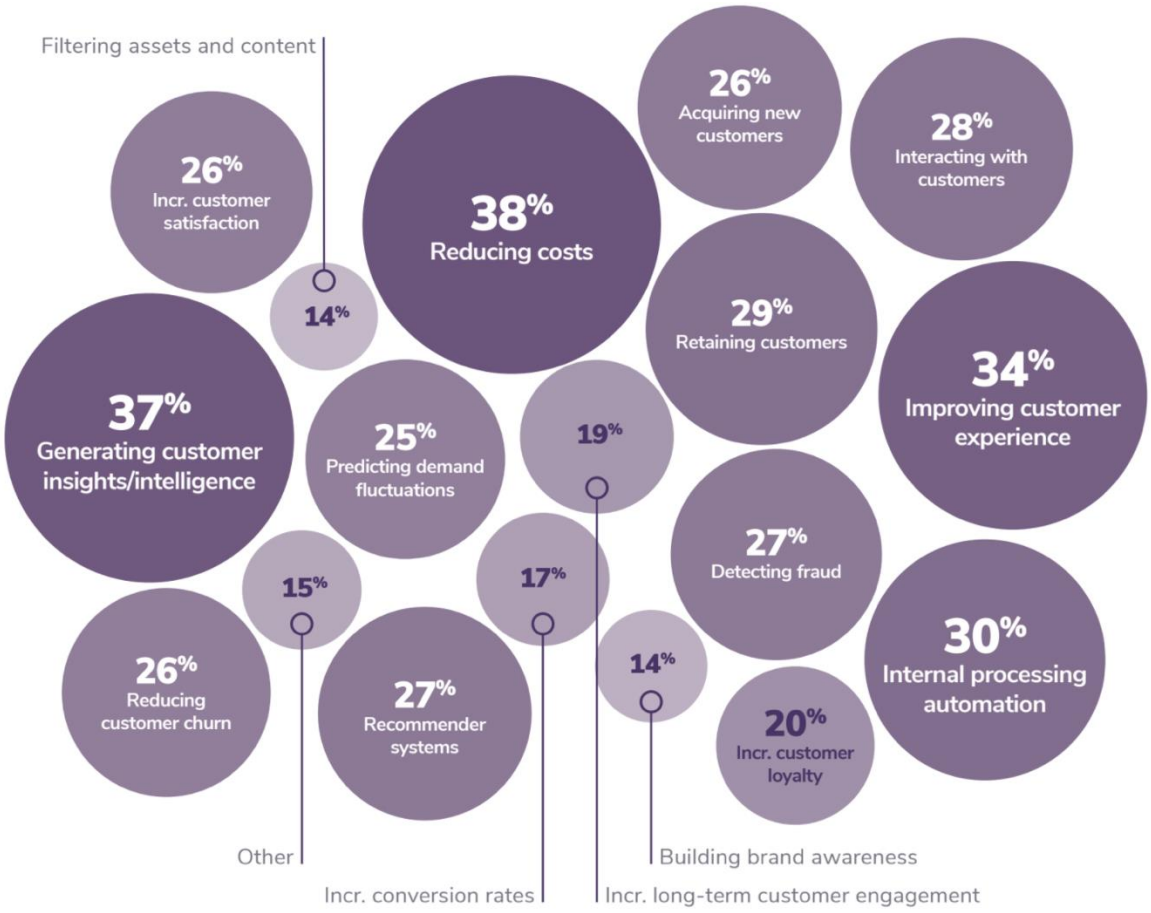
Summary: The story so far

You have made it this far, congratulations! This means you did not zone out in between and the document was not intense and geeky, and that's a good thing. The main aim of this document is to get you to understand *AI* and *Machine Learning* in the simplest way possible. So, let's take a look at what we have been talking about so far, before moving forward to the next topic:

- What AI and Machine learning is, and what differentiates them?
- What Machine learning is and what are the core terminology to be aware of.
- The different types of machine learning: *Supervised, Unsupervised, Reinforcement, Ensemble methods and Neural Networks*
- Case study on a *Jogging Distance predictor* algorithm; what is the process of designing a machine learning algorithm and the steps involved.

As you could have already guessed right off the bat, there has been a lot of emphasis on the "*what"* of things, to get you familiarized with the whole concept. Now we will move onto the "*how"* of things and how it can help you in your particular situation, depending on the scenario. The next sections will focus on the *AI Use Cases* and *Scenarios* of machine learning depending of different industry verticals such as *Sales, Marketing, Operations, Customer Service, Finance and Talent Acquisition.*

9. Machine Learning Use Cases depending on Industry Vertical



To answer the questions that might arise based on your particular situation, whether you are from sales or marketing or banking, we are going to look at a few examples at machine learning in the industry, with some real-life examples to be able to wrap ours heads around it better and to think about how it can be used in your scenario.

The information in the representation above is from a survey conducted by [4]*Algorithmia* on companies of varying sizes, and the top three machine learning use cases across the board were:

- Reducing company costs
- Generating customer insights and intelligence
- Improving customer experience

Breakdown of use cases by industry

Understandably, industries with customer-facing products or services (retail, manufacturing, healthcare, etc.)prioritize ML applications that improve customer service, and industries involved with security, compliance laws, and proprietary data (financial institutions, government agencies, insurers, etc.) focus more so on ML use cases that help solve those challenges. The following are a few noteworthy examples:

- Reducing *customer churn* using ML is a priority for groups of people who work in consulting and professional services industries.
- Interacting with customers using ML was a top priority in the Education/EdTech sectors.
- Increasing customer satisfaction was the main use case in the Healthcare, Pharma and biotech industries with them suggesting that customer churn or customer dissatisfaction may be a perpetual challenge.
- The Information Technology sector (IT) mainly uses ML to acquire new customers.
- Software development organizations prioritize ML recommender systems to guide viewers toward new features or products to buy.
- As we will see through as example in the following section, Banks and financial services firms are focusing their ML power toward retaining customers and detecting fraud – keeping the customers happy and mitigating vulnerabilities to the company.
- Lastly, the energy sector, including utility companies, are using ML to forecast demand fluctuations likely to prevent power outages, reduce response times during disruptions of service, and plan for power consumption for coming years (NeuralDesigner).

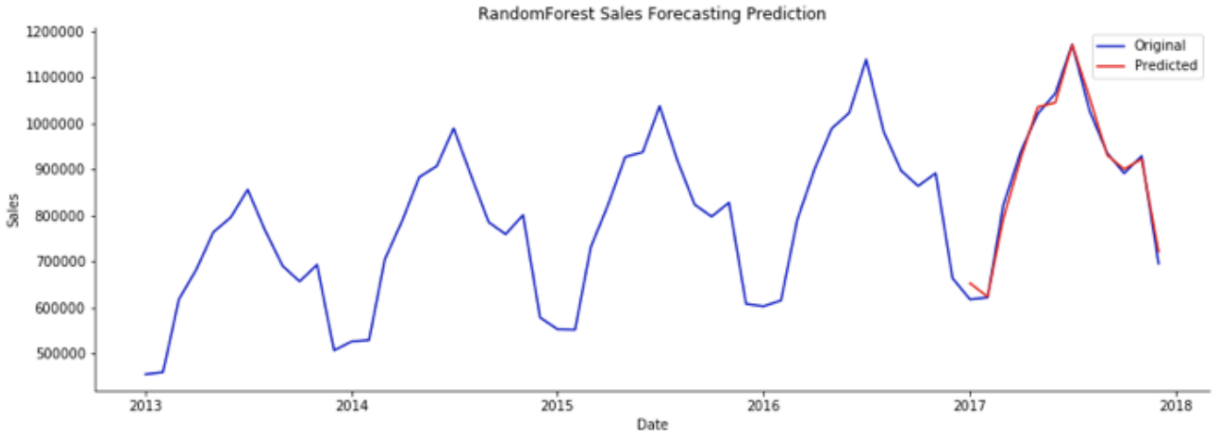
[5]Forecasting sales is an essential technique that Retailers and Marketers are using in the past ML era to identify benchmarks and determine incremental impacts of new initiatives, plan resources in response to expected demand, and project future budgets. Here is an example for how a sales forecasting model looks like, after crunching the numbers and doing all the geeky ML coding stuff:

[4] *Algorithmia, 2020 state of enterprise machine learning*
[5] *5 Machine Learning Techniques for Sales Forecasting, Molly Liebeskind, April 22nd 2020*

Sales and Marketing Example

SEO and Keyword optimization In Marketing

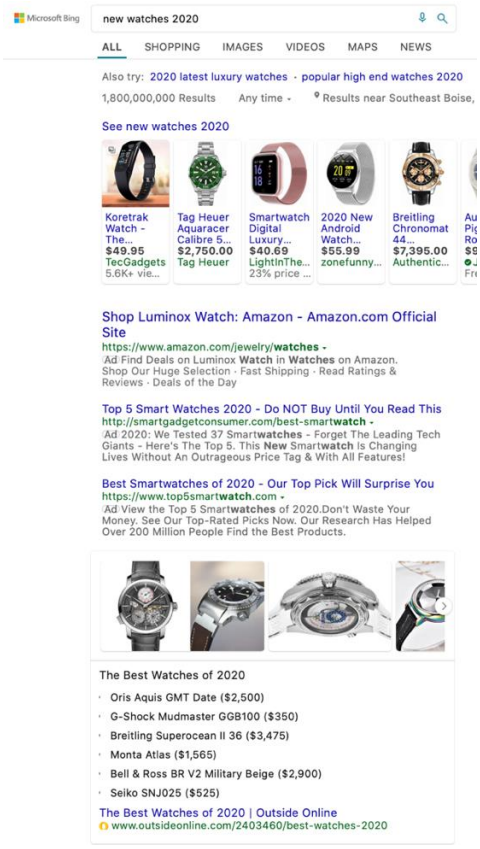
Machine learning is utilized in the marketing space in the most obvious of ways, that is to utilize SEO (search engine optimization) and keyword optimization. Notice when you are googling for a new watch and get multiple advertisements on Google that lead you to various vendors like

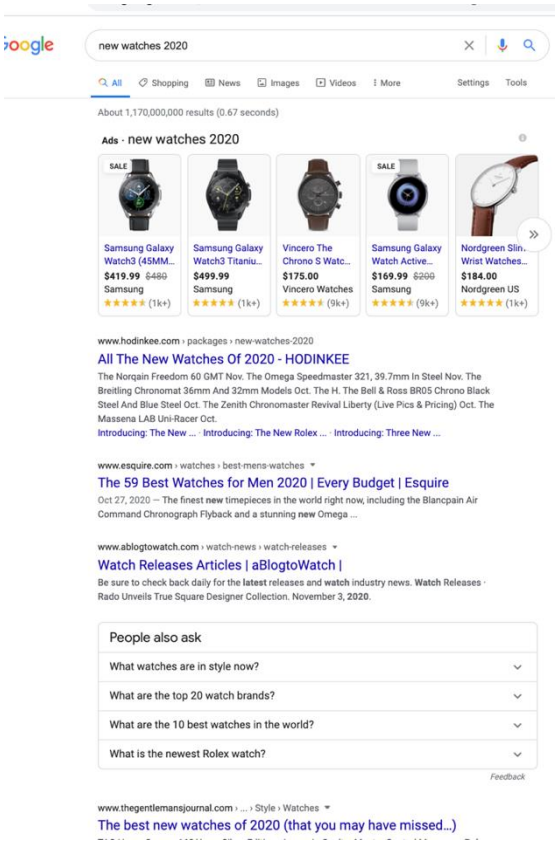


Amazon, eBay, BestBuy etc.? Well, that's SEO in full cry. We don't have the time to get into the nitty gritty of the core details about SEO, but from a bird's eye view, all we need to know that it adds to Google's capabilities an incredible speed of:

- Data accumulation
- Interpretation
- Reaction

Here's an example of the same search conducted on Google Chrome and Microsoft's Bing browsers for the same product, *new watches 2020* but notice how we get different results, this is an example of how SEO works in its simplest form:





Predicting Customer Churn

One of the biggest challenges that company's face in the modern world is losing their customer base. With the onset of superior technologies and greater options, we see customers going out the back door. Luckily for you as a marketer or business analyst, there is a way to predict this too. Churn quantifies the number of customers who have unsubscribed or canceled their service contract with a company they are relying on for a certain service.

Let's look at it through a simple definition and example:

$$Customers\ lost\ during\ fixed\ period / Total\ customers\ at\ start\ of\ fixed\ period = Customer\ Churn\ Rate\ in\ \%$$

Let's say you have 1000 customers on January 1st. 100 of them are not using your service at the end of the month. Also, you have acquired 500 new customers through January 31st, where 125 directly churn out. The simplest churn calculation is as such:

100 + 125 / 1000 = 0.225 = 22.5 % Churn Rate

Of course, that's not all it takes to calculate churn, but we should be able to give you a better model at *NexusMax.*

Machine Learning in the Healthcare Value Chain – Revenue Cycle Management

Incorporating AI and machine learning into revenue cycle practices can lead to drastic improvements in business outcomes. Here are some examples use cases of how ML can change the Revenue cycle management landscape, but also, we will cover a lot of topics on RCM in the Scenarios table coming up in the next section:

- Personalized patient engagement, segmentation for payment plans, financing, charity, digital touch points, and communication strategies
- Reduced staff workload, re-allocation of resources to higher value tasks
- Increased consumer experience
- Increased **Customer Lifetime Value** and retention
- Increased accuracy, forecastability and scale of revenue models
- Improved budget allocation
- Increased mobile and digital adoption coupled with increased features and functionality
- Better, faster scalability of services

Examples of Use Cases by Industry

Retail		
Application	**ML Type**	**Use Case Description/ Links to references**
Fraud Detection	Classification	Credit Card Fraud Detection
Inventory forecasting / Maximize Sales and minimize Returns	Regression	Grupo Bimbo Inventory Demand
Store Sales prediction	Regression	Google Analytics Customer Revenue Production
Which customers are happy customers	Classification	Customer Satisfaction
Which promotion is best for the customer?	Regression	Sales Prediction
Forecasting Website traffic	Time Series	Web Traffic Time Series Forecasting

Churn Prediction	Classification	KKBox's Churn Prediction Challenge
Movie Recommendation	Clustering	Unsupervised Classification Project
Movie Recommendation	Recommendation	Movie Recommender
Classify products correctly	Classification	Classify products into the correct category
Recommend items for shopping cart	Recommendation	Shopping Cart Recommender
Video Games sales Prediction	Regression	Video Games sales for Ps4 and Xbox
Super Stores Sales	Time Series	Predicting Sales of Super Stores

Real Estate		
Application	**ML Type**	**Use Case Description/ Links to references**
Zestimate prediction (Kaggle competition)	Regression	Zillow's Home Value Prediction
Property Inspection Risk Prediction	Classification	Property Inspection Prediction
Zillow Rent Index	Time Series	Zillow Rent Index, 2010 -2016

Marketing		
Application	**ML Type**	**Use Case Description/ Links to references**
Predicting where user will check-in	Recommendation	Identify the correct place for check ins
Customer Segmentation	Classification	Santander Customer Transaction Prediction
Which content will get clicked?	Recommendation	Predict which recommended content each user will click
Personalized Hotel Recommendations	Recommendation	Predict which hotel type will an Expedia customer book
Market Basket Analysis	Clustering	Market Basket Analysis using mall customer data
Should I send this direct mail to the customer	Classification	Determine whether to send a direct mail piece to a customer
Which Shopper is more valuable	Classification	Predict which shoppers will become repeat buyers
Will the user click this ad	Classification	Predict if context ads will earn a user's click
Predicting click-through rates	Classification	Predict whether a mobile ad will be clicked
Telco Churn Prediction	Classification	Everything you need to know about Churn

Manufacturing		
Application	**ML Type**	**Use Case Description/ Links to references**
Reduce manufacturing failures	Classification	Reduce manufacturing failures
Fault detection	Computer Vision	Power Line Fault Detection

Finance		
Application	**ML Type**	**Use Case Description/ Links to references**
Market Segmentation	Clustering	Financial Market Segmentation using Clustering
Matching product with customer	Recommendation	Santander Product Recommendation
Insurance customer risk prediction	Classification	Prudential Life Insurance Assessment
Loan Early Pay-off Prediction	Classification	https://www.kaggle.com/datasets/zhijinzhai/loandata
Money Laundering Detection		How can ML detect Money Laundering?
Credit Card Fraud Detection	Classification	Credit card fraud detection model
Stock Exchange Prediction	Regression	New York Stock Exchange Prediction Model

Technology		
Application	**ML Type**	**Use Case Description/ Links to references**
Malware/Spam classification	Classification	Microsoft Malware Prediction - I
Predicting resource utilization/optimization	Time Series	Predicting Resource Utilization for Web Services

Logistics		
Application	**ML Type**	**Use Case Description/ Links to references**
Grouping of orders for delivery to reduce costs	Clustering	Optimizing Order Fulfillment
Maximizing Truck Utilization		Machine Learning for Fleet Management

Healthcare		
Application	**ML Type**	**Use Case Description/ Links to references**
Hospital Length-of-stay prediction	Time Series	Microsoft-Hospital Length of Stay - Github
Fetal Health	Regression/Classification	Fetal Health Classification model - Kaggle
Patient Charges	Clustering / Regression	A guide on predicting Patient Charges - Kaggle
Heart Failure Prediction	Classification	Predicting Hear failure - Kaggle
Healthcare Expenditure	Time Series	AI in Healthcare: Time Series forecasting using ML
Heart Failure Prediction	Classification	Heart Failure Prediction Model - Kaggle

RCM – Revenue Cycle Management		
Insurance Discovery, Demographic Verification, Insurance Verification, Self-Pay Analysis, Deductible Monitoring, Claims Monitoring		Revenue cycle management solutions by Zoll AR
RCM - Predictive Collections & Denial Management		Claims denial management
RCM - Utilization review and case management, Scheduling/pre-registration, Remittance processing and Rejections, Claims Submissions		Healthcare RCM basics and applications
RCM - Charge Capture and Coding Optimization		Key Insights to RCM
Charge Capture Advisor, Claims Lifecycle - Denial Propensity Scoring, Optimized Healthcare System - Payment Accuracy		Beckers Healthcare in RCM analysis

Energy		
Application	**ML Type**	**Use Case Description/ Links to references**
Predict Energy Consumption	Time Series	ASHRAE - Great Energy Predictor

10. NexusMax Platform Introduction

NexusMax is an integrated self-service AI platform that enables business users, business scientists and data scientists to harness the power of AI and Machine learning, to solve their business problems irrespective of their ML expertise. NexusMax provides an intuitive and collaborative drag and drop experience on top of the best in-class data and AI technologies to deliver results in hours and not days or weeks.

NexusMax AI platform puts the business user in the driver's seat. Our built-in automation accelerates the entire machine learning pipeline – data ingestion, data transformation, feature engineering, model training, model deployment – and integrates them with line-of-business tools, dramatically reducing time to insight for your business. It enables business users to collaborate with Data Scientists by generating high quality Python code within Jupyter Notebooks that they can optimize further.

Benefits of the NexusMax Platform

- You focus on solving the business problem, we automate the rest
- Visual tools for analysts, Jupyter and Python for Data Scientists
- Business impact in hours, not days or weeks
- Actionable insights right from the tools of your business
- The best of computing, data & AI technologies; on-perm & cloud

Goals of the NexusMax Platform for the User

- **Simple and Easy – By Design**: Analyst focused experience: For analysts who want a rapid solution to their business problem, we created a very simple and easy interface.
- For those who are versed in AI & ML, we expose more flexibility and power!
- **Visual Feature Engineering**: ***Auto Feature Engineering***: Fully automated feature engineering for business users. Manual Override: Citizen data scientists and power business users can flex their feature engineering muscle with the manual override option.
- **Visual Model Training**: ***Auto ML***: Automatically identifies the best ML model, with manual override. All model metrics in one place, visualized graphically for easier understanding. Retrain AI Models easily: Want to iterate for better results, retrain with a click of a button.
- **Deploy and Manage Models Easily**: ***Model Management***: View, edit and delete models on a single dashboard across Development, Staging & Production environments. Model Deployment: Single click ML model deployment into any environment.

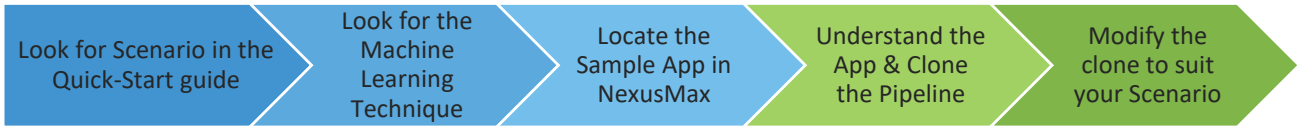
- **Integration into Business & Data Science Tools**: ***Collaborate with Data Scientists***: Generate Python code that can be modified with Code Extensions. Intelligence where you need it: Seamless integration with business tools like Excel, Power BI, Tableau etc.

11. Sample data use description: Setting up a Sample Application

As part of the NexusMax Trial version, we have included a set of sample data applications that can be used to get a sense of what the platform can do. You will have the opportunity to play around with some of the features that are available to you, and this should help you visualize what machine learning is like.

The goal with this section is to provide you with a set of steps to follow on how to make this process easier on your part. You can leave the rest of the machine learning details to us since you know, since comprehensive applications will run at the backend of NexusMax. All you will need to do is press a few buttons, literally, and you will get results delivered to you.

You can achieve this is by following these simple steps for setting up and modifying your sample data application in the NexusMax platform:



Since you are reading this, you already know where to go to be able to access the Scenarios table. The table is a great way for you to start exploring the many different applications of Machine learning in the real world, so I would suggest you take some time familiarizing yourself through the Quick-Start guide and the scenarios table to get a bird's eye view of what goes around in the ML world, and it's completely okay if things start getting a bit geeky, the lucky part is that you don't have to focus on all that backend data analysis fiasco.

Moving onto the next section, we will discuss how each step can help you design your very first ML application using NexusMax. There is also description at the end of this section which I highly recommend you watch; it will give you some perspective on what you are dealing with! Now read on!

Here are the steps in detail so you can follow to make the most of the sample apps:

Step 1: Look for your Scenario in the Quick-Start Guide

In the previous section on Use Cases, we have a Tabular representation of Machine learning scenarios grouped by industry vertical. You can use this table to pin-point where your focus is. This folder is also present in the NexusMax environment in a README file, for your convenience.

For example, if you were from Health Care industry and wanted to use an app that can group Covid-19 patients using Clustering ML type, you would refer to the scenarios table and then understand where it falls in the hierarchy and then go from there.

Step 2: Look at the Type of Machine Learning Technique

Once you figure out what application you are trying to use, the table should give you an idea of what type of ML technique is applicable in that example.
Let's say you are from Retail and want to predict your stores sales, you would look at the table and understand that this type of problem uses a Time Series ML technique, and would follow the workflow accordingly.

Step 3: Locate the Sample App in NexusMax Environment

So, you now know what your Scenario is and what type of ML technique can be applied to that scenario. Great! You are a person with such clarity, it is not so easy to be so clear about what you want in life.

Now that you have those in place, you can move onto the NexusMax platform and locate the sample app related to your criteria in the Projects tab on the home screen. These are the steps:

NexusMax home > Projects > Sample > Applications > Desired Application

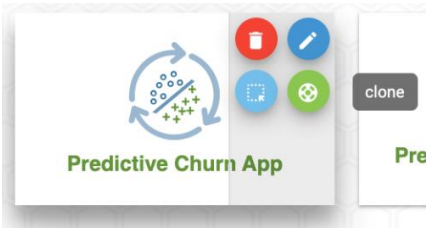
Step 4: Understand the App and Clone the Pipeline

Once you click the application you so desire, you will be taken into the NexusMax platforms design workspace, where all the magic happens!

Here you can start your very first Machine Learning project and take some time to understand how it works. Luckily, the user interface within the platform makes it very easy for you to understand the workflow, so that's a win.

Once you are in, you should take some time trying to understand the steps involved in the buildup to the model training and results and try to play around with the features.

If you feel that this is a system that fits your framework, you can go back to the applications dashboard and *Clone* the application and paste it into a new project file created by you (named to whatever you'd like).



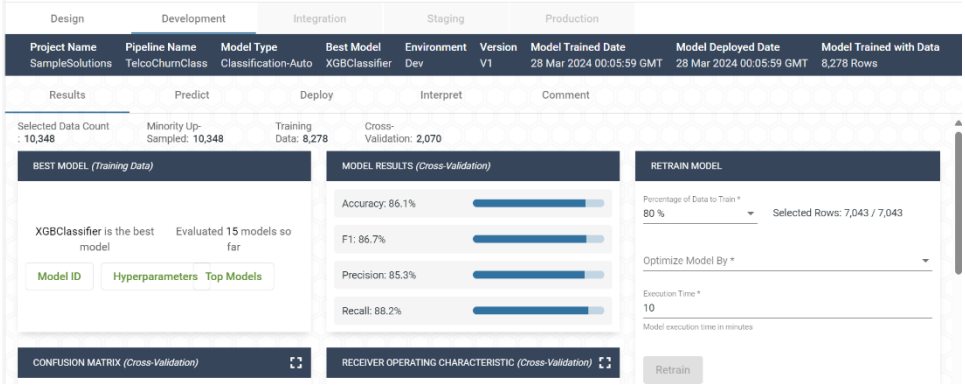
These are the steps you will take:

Locate the App > Hit on the Clone button > Select the desired Directory you want it cloned to

Step 5: Modifying the Clone to Suit your Exact Scenario

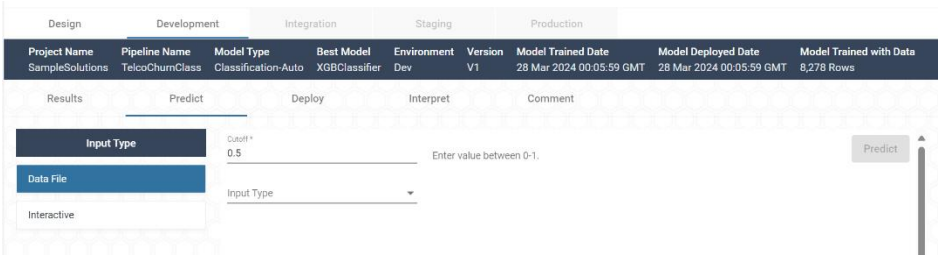
Once you have the *Clone* application at your disposal you can modify the application as you please. The workflow is simple enough on NexusMax for you to edit each step of the way, as shown in the sample workflow below:

Next you just need to run the cloned pipeline to train the model. The performance of the model will be displayed in the next "Results" tab.

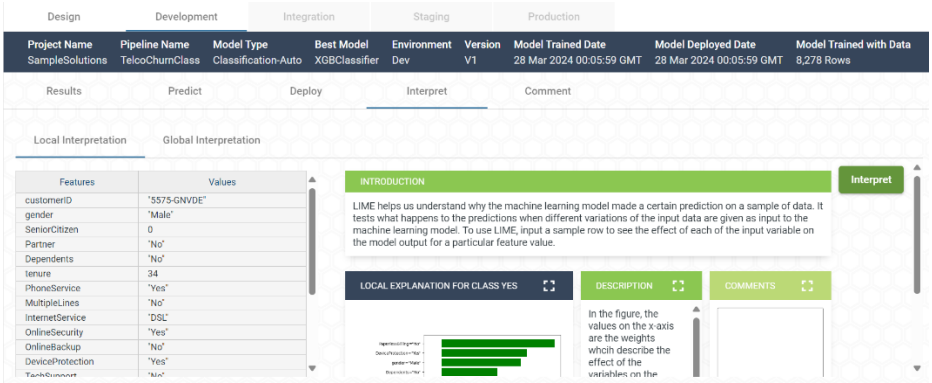


Step 6: Predictions and Interpretations

Once the model is trained, you can get the required predictions of future data with the model that has been trained. All you would need is to select the next tab called “Predict”, import the suitable file and hit the predict button. Your predictions will be displayed right away.



On completion of prediction, the next step would be to interpret the model. With a simple click on the next tab “Interpret”, NexusMax offers both local and global Interpretations options. They have been explained in detail in the Appendix.



Demo Use Case Walkthrough

If this step-by-step guide was not enough for you and you feel like you need more insights to get a better sense of what NexusMax's Machine learning platform can offer, then we have some good news!

We have put efforts into making a visual guide, through a video which you can check out here. This movie is a walkthrough of a simple Demo Use Case in the NexusMax platform using Supervised learning. It focuses on a dataset which contains information on customers who have churned earlier from a telecom service. It has data of Customer's general information of the services availed and relevant historical details. I mean who doesn't like a good visual representation to go with all this textual learning, right?

Appendix

The Banking Example: Always-on Chatbots

It is quite mind boggling to think that in the 21st century the banking industry is essentially just a world of computers and networks. The bulk of the worlds wealth is stored in databases and transactions these days are simply an exchange of information between networks that banks own. As amazing or dubious as that sounds, artificial intelligence aims at further improving the relationships between banks and customers. Banks never seem to be open when we want them to; for instance, if we go after work hours, you're most likely to find that banks are closed. Our money doesn't sleep, so why should the banks?

This is why banks are now using AI powered conversational assistants to or *chatbots* to engage customers 24/7. Customers are using these chatbots to handle many operations which would otherwise require a phone call or a visit to the bank. They are also using chatbots for private conversations regarding bank transactions, bank services and other tasks that don't really require human intervention.

A good example of this is Bank of Americas *Erica* chatbot, that essentially gives insights about a customer's spending, transactions history and also other functions like locking your credit card or showing transactions from a particular vendor. You can either speak commands to it, or chat with it using conventional texting. Many other banks have followed suit, like American Express' *Amex bot, EVA* by HDFC and so on. If you are in the banking business, maybe this might be a good option for you to consider? And maybe NexusMax can help? Food for thought.

Local and Global Interpretation Dashboards

The following are the interpretation techniques that we use at NexusMax:

Firstly, let us understand local interpretability of a customer who is likely to churn or not. The red indicates probability of 0(Not churning), and the green indicates probability of customer likely to churn. For instance, a customer having a tenure of 1 year is highly likely to not churn with a probability of 0.25 approx. The same customer has other features inclined towards predictions of not likely to churn.

Further, in NexusMax for global interpretation, we provide explainer dashboards which are equipped with interactive dashboards for analyzing and explaining the predictions and workings of machine learning models, including xgboost, catboost and lightgbm. This makes the model transparent and explainable.

FEATURE IMPORTANCES

It is to explain how the value of a specific feature influenced the outcome of an individual prediction — instead of the overall model. There are 2 types that NexuMax has used to understand feature importance. A graph is plotted which can be used to visualize or compare how each feature plays a role in the outcome of the predictions.

- **SHAP**: Shapley Additive Explanation — A prediction can be explained by assuming that each feature value of the instance is a “player” in a game where the prediction is the payout. Shapley values – a method from coalitional game theory – tells us how to fairly distribute the “payout” among the features. It is a model explainability algorithm which operates on single prediction level. The graph visualizes the average SHAP contribution of each feature. It gives how much has each feature value contributed to the prediction compared to the average prediction.
- **Permutation Importances**: Permutation feature importance measures the increase in the prediction error of the model after we permuted the feature’s values, which breaks the relationship between the feature and the true outcome. Here the importance of a feature is

measured by calculating the increase in the model's prediction error after permuting the feature. A feature is "important" if shuffling its values increases the model error, because in this case the model relied on the feature for the prediction.

REGRESSION STATS/ TREE EXPLAINER

In this tab the overall summary of model performance in terms of metrics could be viewed.

INDIVIDUAL PREDICTIONS

- **Partial Dependence Plot:** The partial dependence plot shows the marginal effect one or two features have on the predicted outcome of a machine learning model. A partial dependence plot can show whether the relationship between the target and a feature is linear, monotonic or more complex.

- **Contributions Plot and Table**: Shows the contribution each feature had on a prediction. The contributions add up to an instance of a predicted value.

WHAT IF..

Here there are options to change the input values of each feature and check for the change in predicted value. The contribution of each feature to the predictions can also be visualized.

FEATURE DEPENDENCE

- **Shap Summary**: The Shap summary summarizes the Shap values per feature.
- **Shap Dependence**: The plot in Shap dependence shows the relationship between feature values and shap values. This allows to investigate the general relationship between the feature value and the impact on the prediction. We can verify if the model uses features inline with the intuition of the user.

FEATURE INTERACTIONS

Interactions Summary: Shows shap interaction values. Each shap value can be decomposed into direct effect and indirect effect. The indirect effect is due to interactions of the features with each other. Interaction Dependence: The plot shows the relationship between feature values and shap interaction values. This allows to investigate the interactions between the features in determining the predictions of the model.