Exploring Neural Networks

Digital Media Master Thesis

October 2020

By Pranoti Tumkar

Acknowledgement

I would like to thank my advisors Prof. Peter von Maydell and Prof. Dennis Paul for their guidance, advice and invaluable feedback for the work. I would like to thank my friends and family for supporting me during this entire period.

Contents

Introduction

Your life is already affected by machine learning - whether you are aware of it or not. It is used in the USA to make important decisions such as screening resumes of job applicants, to suggest how long a criminal should be incarcerated. It is also used in shopping websites to suggest which product you should buy next or which advertisement you should see. To develop your own opinion on how it should be used to impact your life, you need to know more about what they are, how they work, and when it is actually useful to use them. The word 'Artificial Intelligence' (AI) is often used to paint a utopian or a dystopian future. The term AI has different meanings over time, but for this thesis, I will use it the way programmers today use it - to refer to a type of computer program called machine learning algorithm.

Machine learning is not about teaching computers how to think like humans. It is about applying math to large quantities of information. Machine learning provides computer systems the ability to automatically learn and improve from experience without being explicitly programmed. The process of learning begins with observations or data, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. The primary aim is to allow the computers to learn automatically without human intervention or assistance and adjust actions accordingly. A neural network is a type of machine learning which models itself after the human brain.

A peek behind the curtains to see how machine learning works to enable us to start having a discussion about its various aspects impacting life. I believe that machine learning has the potential to change the world for the better. But to understand how it can, it is important to know and be aware of its pitfalls. I look to explain key aspects of machine learning to anyone without prior knowledge, so as to start a conversation. A few key aspects to be aware of: opacity, scale and damage. Some of which can be countered with: transparency, controlled by the user, and personal.

My work is not about the positive or negative aspects of machine learning, but rather about taking an unbiased approach. I developed a tool that people can use to introduce and make the topic of machine learning more approachable and provide hands on experience in a short seminar/workshop setting. The seminar/workshop can be conducted by anyone who already has knowledge about machine learning and uses the website to present. The text can be read by the participants or work as a guide to the person conducting it.

Target participants include teachers who want to give a glimpse of this topic to their high school students, team leads in a non-tech company who want to encourage their team members to be open to digital change. It is not for people who already have a serious interest in this topic, but to get them interested. It is targeted to take up 30mins of the participants time.

A lot of the available information and discussion in my thesis is based on the use of Machine Learning algorithms in the United States of America (USA). The examples may not apply to other parts of the world.

Age of data

In the 20th century, value was placed in intangibles such as brand and intellectual property. This was a change from the previously held values related to physical infrastructure such as land and factories. This is further expanding to data, which is considered the oil of the information economy. It will probably be a matter of time until data is found on corporate balance sheets.

Around the early 2000s, data was no longer considered stale or something which was not needed once it’s original task was completed. Rather, data became a raw material used to create new forms of economic value. There are significant strides being made in this field at a very fast pace. The way the telescope allowed us to understand the universe while the microscope enabled us to understand germs, big data will help us analyse huge quantities of information and enable us to make sense of our world in different ways. To comprehend the degree to which the information revolution can affect us, let us consider astronomy and the effect of the telescope on this field. When the Sloan Digital Sky Survey began in 2000, its telescope in New Mexico collected more data in its first few weeks than what was collected previously in the entire history of astronomy. By 2010, it had collected 140 terabytes of information. A successor, the Large Synoptic Survey Telescope in Chile, which started in 2016, acquires that quantity of data every five days.

Most of our institutions were established based on the understanding that human decisions are based on small, exact, and causal information. But the current situation involves data that is huge, processes quickly and tolerates inexactitude. It is not just the size of the data which makes it big data. It also has to do with analysing the entire data rather than just a sample of it.

What is Machine Learning?

Machine learning is not about teaching computers how to think like humans. It is about applying math to large quantities of information. Machine learning provides computer systems the ability to automatically learn and improve from experience without being explicitly programmed. In regular programming, you tell the computer what to look for, and you tell it what to do. You must list every condition and define every action. The issue with this is that you need to be exhaustive and think about and cater to every possible scenario.

Machine learning still requires code, but that code isn't a step-by-step procedure to solve the problem, like in traditional programming. Instead, the code in machine learning tells the computer how to crunch the data, so that the computer can solve the problem by itself. The primary aim is to allow the computers to learn automatically without human intervention or assistance and adjust actions accordingly. The process of learning begins with observations or data, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. Machine learning is used for prediction, pattern recognition, and generating new content.

Let us consider the example of a program learning to play a video game. The algorithm learns how to play by trial and error. It starts by giving random commands: “decelerate,” “shoot,” “rotate,” and so on. The algorithm remembers instances when it leads to a success - a high score. The algorithm also makes a note of the times it leads to failure - such as death. It also keeps track of the state of the game - where are the enemies, the obstacles, and the power-ups, health status and so on. After this, whenever it encounters a similar game state, the algorithm is a bit more likely to try the successful actions than the unsuccessful ones. After many cycles of trial and error, such a program would become a competent player. In 2013, a system using this approach reached superhuman skills in a bunch of old Atari games.

Janelle Shane talks about the Five Principles of machine learning’s weirdness:

- The danger of machine learning is not that it’s too smart but that it’s not smart enough
- Machine learning has the approximate brainpower of a worm
- Machine learning does not really understand the problem you want it to solve
- But: machine learning will do exactly what you tell it to. Or at least it will try its beast
- And machine learning will take the path of least resistance

What is an Algorithm

The word algorithm comes up often while reading anything about machine learning. Here, we can look a bit deeper into what it is. It is defined differently in mathematics and computer science fields. An informal definition could be "a set of rules that precisely defines a sequence of operations". It can include computer programs or even a cook-book recipe.

According to Cathy O'Neil, an algorithm is an opinion embedded in math. An algorithm is a general concept of what we do in our heads everyday. To build an algorithm we need only 2 things mainly - a historical dataset and a definition of success. An example of an algorithm can be the one you use to cook dinner for your family everyday. The data you use on a daily basis is the ingredients in your kitchen, time, and how much motivation you have to cook. You have the power to determine after dinner if the meal was successful. For instance if your kids ate vegetables, you might call it a success. But if your kids were in charge, they would determine it differently. Over a month, your definition of success is different from if your kids had to determine the success. Similarly, every time an algorithm is built, the data is curated and success is determined based on the person’s values. Essentially, algorithms are not very objective but biased based on the motivations of the person building it.

Why should you care about machine learning?

Some places you might have noticed the effects of machine learning are in your online experience. It determines which ads you see and suggest which videos you should watch next. But you might have realised that these are not flawless - not by a long shot. You might be haunted by ads for a pair of shoes you already bought long ago. Email spam filters sometimes mark important emails as “spam” at the most unfortunate time. People often claim machine learning can do things which are more in the realm of science fiction. Others advertise their products as unbiased while in reality they have been found to be significantly biased. As consumers and citizens of this planet, we need to stop being duped. We need to understand and have a say in how our lives should be affected.

Benefits of using Machine Learning

If we place all the world’s information on CD ROMs, they would stack up and stretch to the moon in 5 separate piles. Going through such large quantities of information needs new methodologies. It is not feasible to go through vast amounts of digital data using the same methods as those used for analog information. The amount of stored information grows 4 times faster than the economy, on the other hand, the processing power of computers grows 9 times faster. Machine learning is a current way to go through vast amounts of data and find patterns. We can consider the analogy of the photograph to the movie. The movie is fundamentally, multiple photographs captured at the speed of 24 frames per second. Big data is similar, where by increasing the quantity, we change the essence. Similar to movies, with big data, we can do new things which were not possible before.

Tasks which have a lot of information to be sorted through and where buggy performance or mistakes can be tolerated, are a good candidate for automation. Tasks such as spam filtering in emails, hyper personalization of movie and music recommendations.

Some aspects to consider while deciding if a task can be automated or not include:

1. C-3PO versus your toaster. Machine learning performs well with narrow or focussed tasks. It's easy to pass the Turing test if you can make the topic of conversation

narrow enough. Eg: chatbots on Facebook for customer service. Problems appear when the task is too broad, such as with self-driving cars.

2. Slow learning and data needed. Machine learning programs learn slowly. If you show a human a pic of a new animal called wug, they would be able to identify all the pictures that contain a wug based on that one picture. But a machine learning program might need thousands or hundreds of thousands of varied wug pictures before it can semi reliably identify wugs.
3. Don’t ask it to remember. Easier to solve a problem if it doesn’t require much memory. This is often seen in text generation where machine learning programs cannot write entire books with a coherent story line.

Helping optimize a cockroach farm in Xichang, China, is something machine learning is likely to be good at. There’s a lot of data to parse, but these algorithms are good at finding the trends in huge datasets. Machine learning programs don’t mind repetitive tasks (unlike humans). Cockroaches reproduce quickly, so it doesn’t take long to see the effects of variable tweaking. And it's a specific, narrow problem rather than one that’s complex and open-ended.

Lab technicians spend hours everyday looking at blood samples under a microscope, counting platelets or white or red blood cells or examining tissue samples for abnormal cells. This can be automated with machine learning analyzing medical images. But the stakes are higher when these algorithms leave the research lab and start working in hospitals, where the consequences of a mistake are much more serious.

During the 2009 H1N1 flu, the Center for Disease Control and Prevention (CDC) in the USA realised that there was a lag of 2 weeks before they got the data from the doctors about the spread of infection. This was too late for acting quickly enough. But Google had the processing power and know-how to correlate frequency of search queries with the area or location of the people. They would correspond it to the data from the CDC about seasonal flu from 2003 to 2008. It wasn’t about looking for search terms like ‘flu medicine’. Their software processed 450 million different mathematical models and found a combination of 45 search terms that when used together in a mathematical model, had a strong correlation with the spread of flu. This provided the CDC with realtime information about where the flu was spreading and they could act accordingly.

Another example is that of models used in baseball. They are constantly learning where the model went wrong, investigate and tweak it. They are based on concrete values such as balls, strikes and hits and there is a constant back - and - forth between the real world activities and the models information.

When machine learning is not helpful

Machine learning is changing how people interact with many products. The possibilities and applications for it are endless. But this is still early stages of machine learning and it’s currently useful at executing a narrow set of tasks. Some scenarios where using machine learning is not a good idea and it will fail include:

1. The problem is too hard. Problems such as hiring the right person for a job is really difficult. Even humans have trouble identifying good candidates.
2. The problem is not what we thought it was. The problem with designing a program to screen candidates for us: we aren’t really asking machine learning to identify the best candidates. We’re asking it to identify the candidates that most resemble the ones our human hiring managers liked in the past.
3. There are sneaky shortcuts. Machine learning identifying rulers in skin cancer detection. Machine learning programs take sneaky shortcuts all the time - they just don’t know any better.
4. Machine learning tried to learn from flawed data. There’s an old computer-science saying: garbage in, garbage out. If the program’s goal is to imitate humans who make flawed decisions, perfect success would be to imitate those decisions exactly, flaws and all. In fact, warning sign numbers 1 through 3 are most often evidence of problems with data.

ExamSoft’s facial recognition technology was used to identify fraud among students while taking the state bar exam remotely in the USA. This software could not identify the faces of people with color clearly and kept asking them to sit in a room with better lighting. Some students managed to overcome this by shining a bright light on their face for the entire duration of their 2 days exam. Not only did students of color have to worry about their exams, but also surviving headaches brought on by the lights. Since people do not have access to the developed software and no way to address these issues, it raises a concern of perpetuating inequality.

The intent behind developing machine learning has the biggest impact on whether there is a positive or negative affect on people. Three aspects of machine learning which lead to them having a negative influence on people’s lives are (i) their widespread use - **scale**, (ii) mysterious workings - **opacity**, and (iii) being destructive - **damage** caused. The main factor is that they are unfair to some groups of people.

Cyclical

When we consider, the machine learning used to determine how long a criminal should be incarcerated and the probability of them committing a crime again are based on various factors. One of them being how many other people they know who have also committed a crime. For people who are from poor neighbourhoods, the likelihood of them knowing someone with a criminal history is higher than for someone from an affluent neighbourhood. This leads to them being incarcerated for much longer leading to them not being able to reintegrate into society, commiting crime, and continuing the cycle.

When the intent is to make a positive impact on people, the same machine learning used to determine the likelihood of people committing crime again, can instead be used to identify people who are from poor neighbourhoods and surrounded by crime and offer them opportunities that enable them to break out of the cycle. This would lead to identifying and helping those who would have otherwise fallen through the cracks without receiving help.

The effect of online marketing advertisements algorithms learn and enable people to be in the bubbles of their beliefs. The poor people are more likely to look at predatory ads for subprime loans or for-profit schools. This data feeds other algorithms and increases their rates for mortgages, car loans, and every kind of insurance imaginable. The same algorithms that abuse the poor also place the comfortable classes of society in their own marketing silos by recommending similar vacations, cafes, and universities. The quiet and personal nature of this targeting keeps society’s winners from seeing how the very same models are destroying lives, sometimes just a few blocks away. It’s a silent war that hits the poor hardest but also hammers the middle class. Its victims for the most part lack economic power, access to lawyers, or well-funded political organizations to fight their battles. The result is widespread damage that too often passes for inevitability.

Scale

There is a change of scale due to so much data being generated. When scale increases, the number of inaccuracies increase as well. A small store can count the money in the register to the penny, while the same cannot be said for the Gross Domestic Product of a country or the census. In 1934, Jerzy Neyman, a Polish statistician showed that random sampling from a data set was more representative of the data. This led to the census conducted in the USA using random sampling rather than the brute force method of going and talking to each person. Sampling solved the problem of an earlier time when collecting large samples of data was very hard. This went on to be used in industries such as quality control, surveys, business decision making, etc. Random sampling does not work when you want to look into sub-categories as this increases the chances of errors. With big data, we need to move away from the mindset of exactitude. With the present day implementation of big data, we need to be satisfied with a general sense of direction rather than knowing the phenomenon down to the inch, penny or atom. What we lose in accuracy at the micro level, we gain in insight at the macro level.

Having the full data set or close to full, allows one to look at the information up close or from different angles. This is not possible with just a sample of the data. When having only a small sample of data, exactitude is important, since you want to make sure the small amount of data is accurate. Embracing messiness is an important aspect of using big data. Large messy data with simple models provides much better results than elaborate models used on less data. Accuracy is sacrificed for scale.

Messiness in action can be found in datasets based on Flicker. Which is an online platform where people upload images and tag them. Tagging has become the standard for classification of images, videos and music. This has no proper standards as anyone can create a tag for their image. Often the tags may be wrong, but the relatively small amount of errors are a trade-off for vast amounts of data.

Opacity: patterns over causality

With big data, we can identify patterns and correlations but have no idea about the ‘why’ behind it - the causality. Amazon’s recommendation systems did not compare people, they tracked all the data such as how long people looked at a product, what they bought, etc, and just found correlations between the products themselves. In 2013, a third of the sales on Amazon were due to the recommendation and personalisation systems which show valuable correlations without knowing the underlying causes. Following Amazon’s lead other companies capitalize on this, such as Netflix recommending what to watch next, social media platforms recommending who to follow or add as a friend.

When we start using machine learning with big data, we will need to stop considering causality and focus on correlations. With big data, we know the ‘what’, but not the ‘why’. This is unlike centuries of experience we humans have with interacting with the world. We are used to expecting the ‘why’ - the reason behind an occurrence or decision. With correlations, there is no certainty, only probability. But if correlation is strong, the likelihood of a link is high. By letting us identify a really good proxy for a phenomenon, correlations help us capture the present and predict the future. If A often occurs with B, we need to look at B, to predict that A will happen. This is useful when A is not easy to observe or measure.

Walmart analysed their shopping information and noticed that prior to a hurricane in the USA, consumers stocked up on flashlights and poptarts - a sugary American breakfast snack. They placed these items at the front of the store along with the hurricane supplies, which led to an increase in their sales. In this example, the cost of being wrong with the correlation is low. There would've been fewer pop tarts sold. We should be careful about using this for decision making which makes a big impact on people’s lives, when we have no idea about the ‘why’ or the reasoning behind a decision. People are unable to push back on a decision which cannot be explained.

Since the 2000s, the shipping company UPS, has been using predictive analytics to monitor their fleet of 60,000 vehicles in the USA. A breakdown can cause major delays in shipments. The cost of collecting and analysing data is lower than the cost caused due to an outage. The company has saved millions of dollars monitoring and measuring individual parts, and fixing them before they break down. This is an instance where 'what' is good enough. To know the 'why', a person would have to manually check the vehicle to figure out the problem.

Potential regulatory measures

It is hard to regulate a software that designs itself. Machine learning programs often learn on their own and identify patterns which humans do not see. Products must be designed to evaluate and test for the implicit biases. It would take a lot of creativity to identify ways in which such self-learning systems can be monitored and held accountable. It could also be in the form of creating artificial self-control: a built-in system of limits on the types of improvements it can make to itself and the rules the software uses to decide if it can make them. Even if machine learning is on a long leash, it means there is someone on the other end holding the leash. Although, we talk about how it "learns" or "makes decisions", that is not entirely true. It is usually the responsibility of the developers who build these systems.

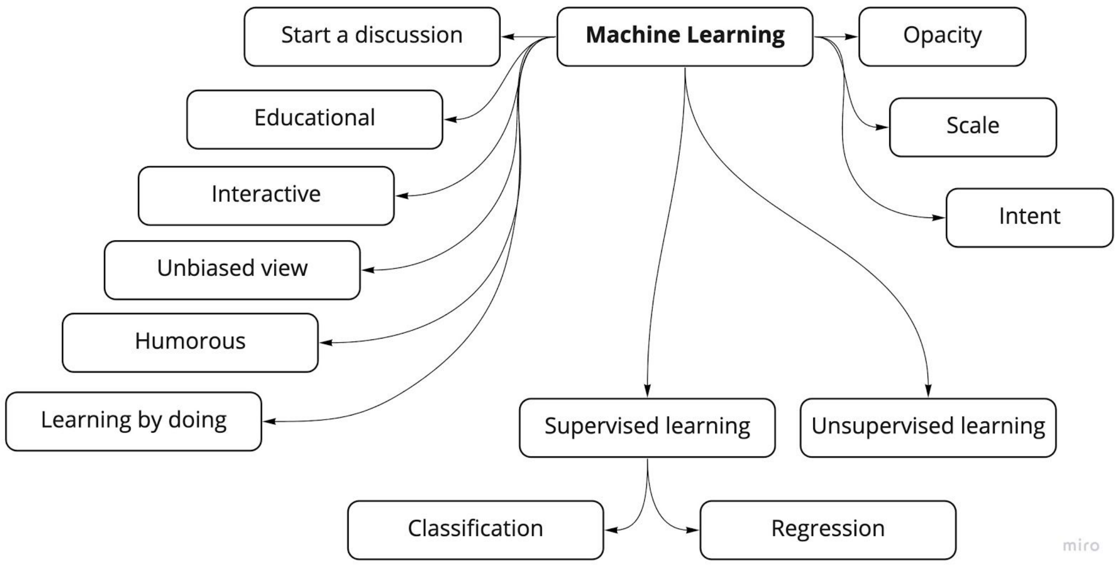
Following the market crash of 2008, two financial engineers, Emanuel Derman and Paul Wilmott, drew up an oath.

The Modelers' Hippocratic Oath

- I will remember that I didn't make the world, and it doesn't satisfy my equations.
- Though I will use models boldly to estimate value, I will not be overly impressed by mathematics.
- I will never sacrifice reality for elegance without explaining why I have done so.
- Nor will I give the people who use my model false comfort about its accuracy. Instead, I will make explicit its assumptions and oversights.
- I understand that my work may have enormous effects on society and the economy, many of them beyond my comprehension.

Presentation: Early brainstorming

My early brainstorming on what to present can be seen below. I wanted to combine educational material with hands-on interactions which enables one to learn more about machine learning.



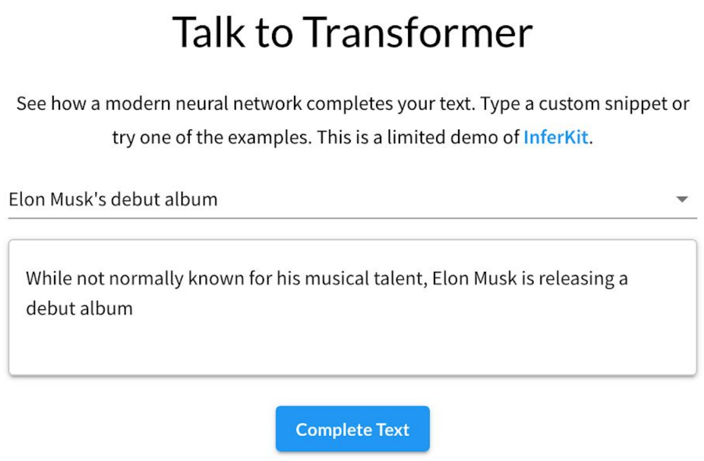
Initially I wanted to have an in-person exhibition. But since the COVID-19 pandemic occurred during my thesis, I decided to switch to a fully online based presentation. The below inspirations and concepts are all for supporting a completely online presentation which people can join from their laptops anywhere.

I wanted to start a discussion with my audience about machine learning where they could learn more about it with an unbiased perspective. I do not take a stance that it has positive or negative connotations. I believe that when used for the right purpose, machine learning has a lot of benefits.

Inspiration from existing projects

An early source of significant inspiration for my thesis is from Janelle Shane's blog https://aiweirdness.com/ . She believes that pranking a machine learning program - giving it a task and watching it fail, is a great way to learn about it. In her blog she writes about her artificial intelligence experiments and the sometimes hilarious, sometimes unsettling ways that algorithms get things wrong. Some of her experiments were computer programs that tried to come up with cat names or one which generated a personality quiz.

I enjoyed projects which were self-sufficient within the website. I could play around and see results within the website without having to go through a lot of effort. One such example is Talk to Transformer https://app.inferkit.com/demo . It is a text generator which generates text based on the prompt you provided. Although it's demo on the website is very limited, it is still a fun example.

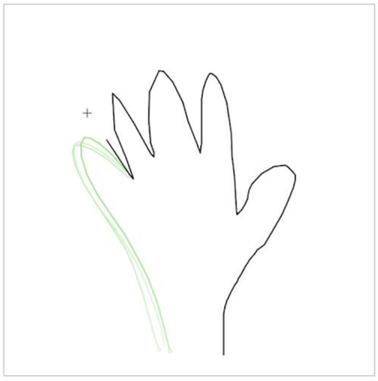

Talk to Transformer

See how a modern neural network completes your text. Type a custom snippet or try one of the examples. This is a limited demo of InferKit.

Elon Musk's debut album

While not normally known for his musical talent, Elon Musk is releasing a debut album

Some simple experiments to start exploring machine learning, through pictures, drawings, language, music, and others can be found at Experiments with Google https://experiments.withgoogle.com/collection/ai . You can try them on the website itself without having to download any other software.


SCRYING PEN

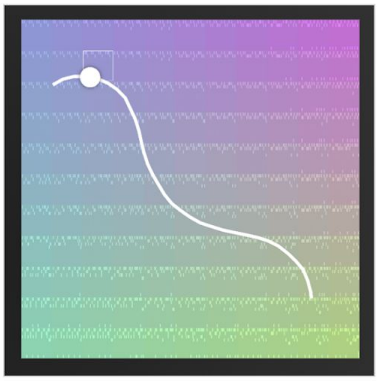
by Andy Matuschak

A realtime implementation of SketchRNN which predicts future strokes while you draw

EMOJI SCAVENGER HUNT

by Google Brand Studio

Identify emojis in the real world with your phone's camera


BEAT BLENDER

by Creative Lab + Magenta

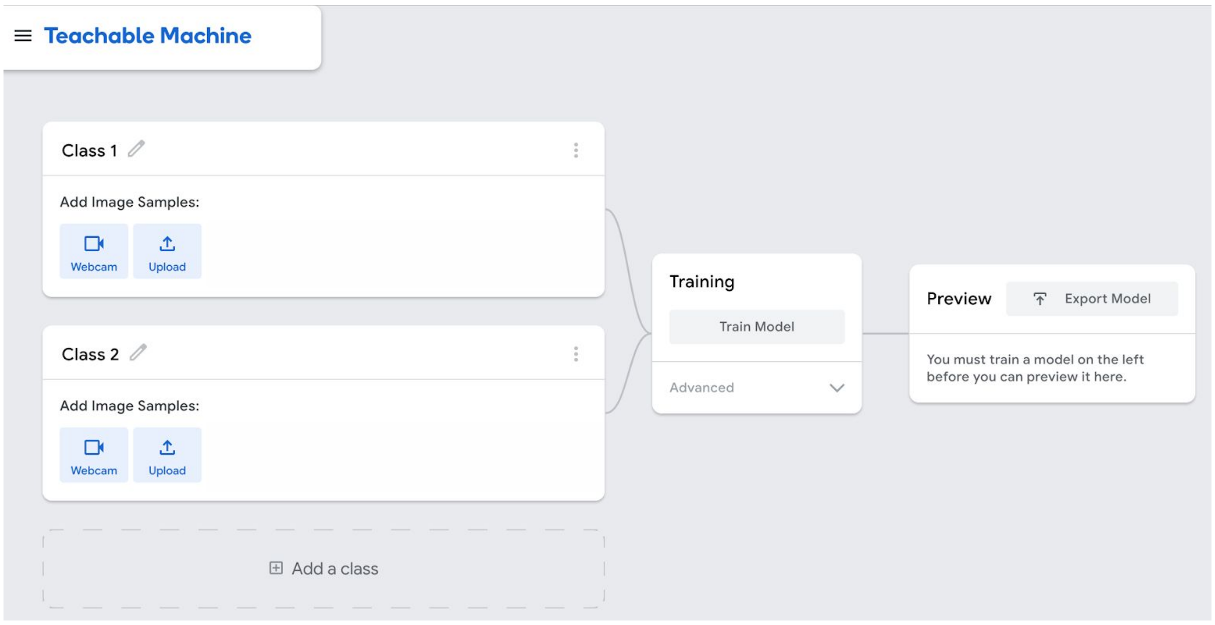
Blend beats using machine learning to create music in a fun new way.

Based on these early inspirations, I looked for projects which were similar to what I wanted to create - interacting and learning about machine learning on the website without having to download any software or have previous knowledge about programming.

Teachable Machine

Source: https://teachablemachine.withgoogle.com/

This project is a web-based tool by Google. You can create machine learning models for your sites, and apps, with no expertise or coding required. It is for educators, artists, students, innovators, makers or anyone else.



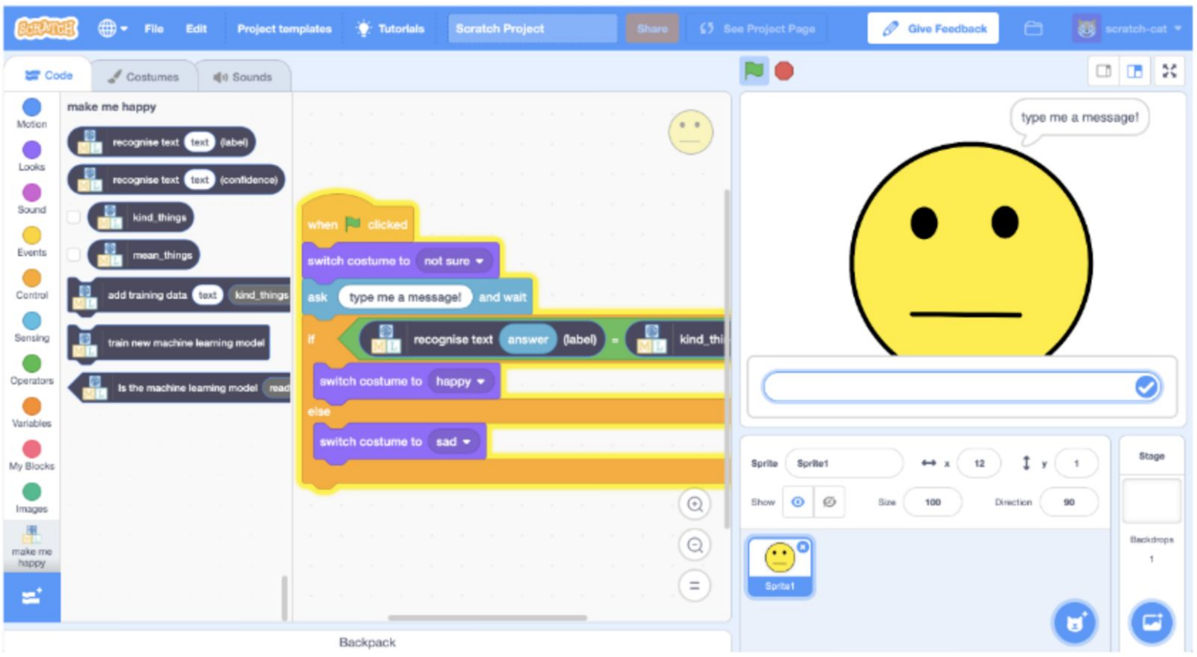
They use the following three steps:

1. Gather: Gather and group your examples into classes, or categories, that you want the computer to learn.
2. Train: Train your model, then instantly test it out to see whether it can correctly classify new examples.
3. Export: Export your model for your projects: sites, apps, and more. You can download your model or host it online for free.

Machine Learning for kids

Source: https://machinelearningforkids.co.uk/#!/welcome
This project is about teaching machine learning to kids. They offer downloadable step-by-step guides, with explanations and colour screenshots for students to follow. They use the following three steps:

1. Collect examples of things you want to be able to recognise
2. Use the examples to train a computer to be able to recognise them
3. Make a game in Scratch that uses the computer's ability to recognise them



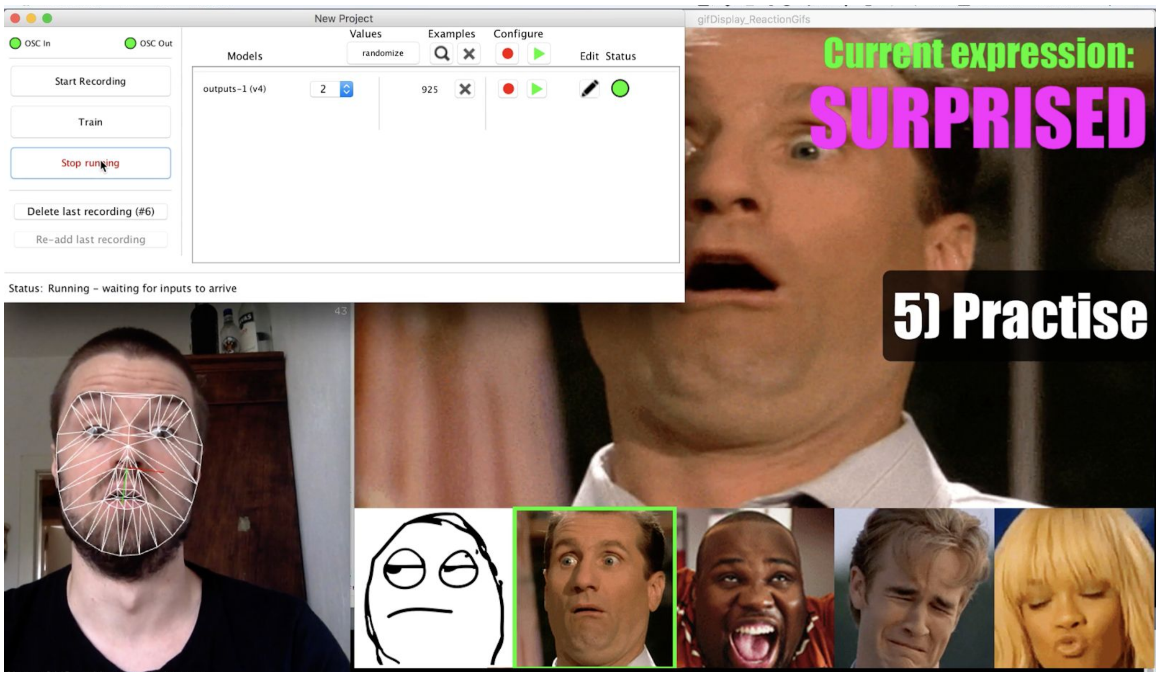
```
when [flag] clicked
switch costume to (not sure ▾)
ask (type me a message!) and wait
if <(recognise text (answer) (label)) = (kind_thi
    switch costume to (happy ▾)
else
    switch costume to (sad ▾)
```

They use Scratch. Scratch is a block-based visual programming language and website targeted primarily at children to help learn code. Users of the site can create online projects using a block-like interface. The service is developed by the MIT Media Lab.

They use visual programming to teach kids about machine learning. The teaching and programming is not fully self-contained in their own website.

Turn your facial expressions into memes by Andreas Refsgaard

Source: https://vimeo.com/175947130
Silly little test showing how to classify facial expressions using Wekinator, FaceOSC and Processing. Wekinator is a software meant for musicians and artists to use machine learning for their artistic work.



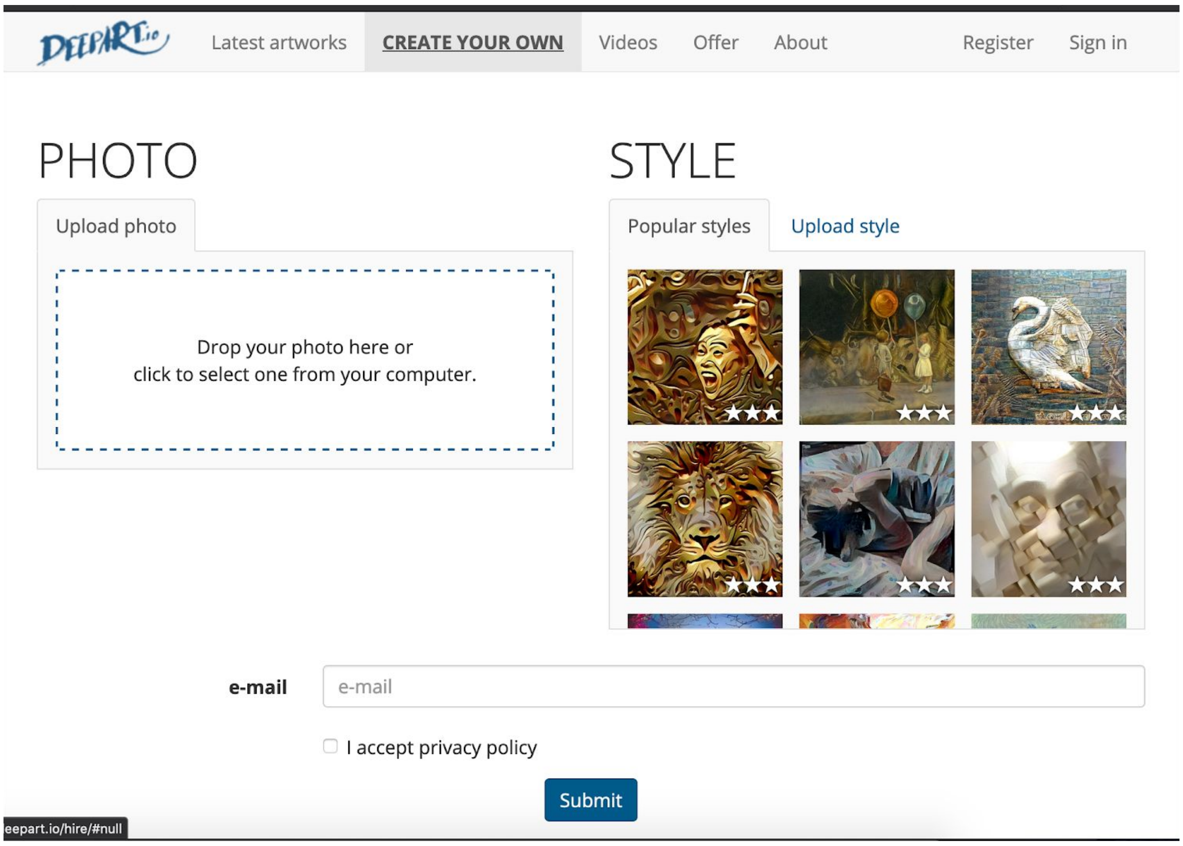
Andreas Refsgaard is an artist and creative coder based in Copenhagen. He uses algorithms and machine learning to make unconventional connections between inputs and outputs, such as enabling people to play music using only eye movement and facial gestures,or control games by making silly sounds.

Deepart

Source: https://deepart.io/hire/
It enables users to create artistic pictures with just a few clicks. They do it using the following three steps:

1. Upload photo. The first picture defines the scene you would like to have painted.

2. Choose style. Choose among predefined styles or upload your own style image.
3. Submit. Their servers paint the image for you. You get an email when it's done.



Their free version renders images in 500 × 500 pixels. For higher resolutions, it is a paid version. Their algorithm uses a neural network and is based on the 19-layer VGG network. University of Tübingen has a pending patent application for the Neural Art technology.

Building a machine learning application

We can look at building machine learning to make knock-knock jokes where it involves machine learning to figure out the rules for itself. It just gets a set of existing knock-knock jokes and needs to come up with more similar ones. It just has a bunch of random letters and punctuation. Initially it tries to guess the first few letters of the knock-knock joke, completely random. The result might be something like "kjshdoa sdoiso?cd". It then looks at the jokes it is provided and realizes that it is wrong and tries to adjust on it's own. It continues to make rounds of corrections until it is something similar. At no point does the programmer explain what the rules of a knock-knock joke are to machine learning.

Sometimes machine learning’s brilliant problem solving relies on mistaken assumptions. For example, Microsoft’s image recognition product identified sheep in an image of a landscape with no sheep. Because it recognised sheep based on the background rather than the shape of an actual sheep. Sheep on a leash were identified as “dogs”.

There are four basic steps for building a machine learning application (or model). These are typically performed by data scientists working closely with the business professionals for whom the model is being developed. A *model* is an abstract representation of a process - an oil company's supply chain, a baseball game, or a movie theatre's attendance. It is similar to a model we all carry in our head - it takes what we know and uses it to predict responses in various situations.

Step 1: Select and prepare a training data set
Training data is a data set representative of the data the machine learning model is provided with to solve the problem it’s designed to solve. The training data needs to be properly prepared—randomized, de-duped, and checked for imbalances or biases that could impact the training.

Step 2: Choose an algorithm to run on the training data set
An algorithm is a set of statistical processing steps. Picking an algorithm is based on the type (labeled or unlabeled) and amount of data in the training data set as well as the type of problem to be solved. Common types of machine learning algorithms for use with labeled data include the following - Regression algorithms, Decision trees, Instance-based algorithms. Algorithms for use with unlabeled data include the following- Clustering algorithms, Association algorithms, Neural networks.

Step 3: Training the algorithm to create the model
This is an iterative process- involving running variables through the algorithm, comparing the output with the results it should have produced, adjusting weights and biases within the algorithm that might yield a more accurate result, and running the variables again until the algorithm returns the correct result most of the time. The resulting trained, accurate algorithm is the machine learning model—an important distinction

to note, because 'algorithm' and 'model' are incorrectly used interchangeably, even by machine learning mavens.

Step 4: Using and improving the model
The final step is to use the model with new data and, in the best case, for it to improve in accuracy and effectiveness over time. Where the new data comes from will depend on the problem being solved.

Neural Networks and how they work

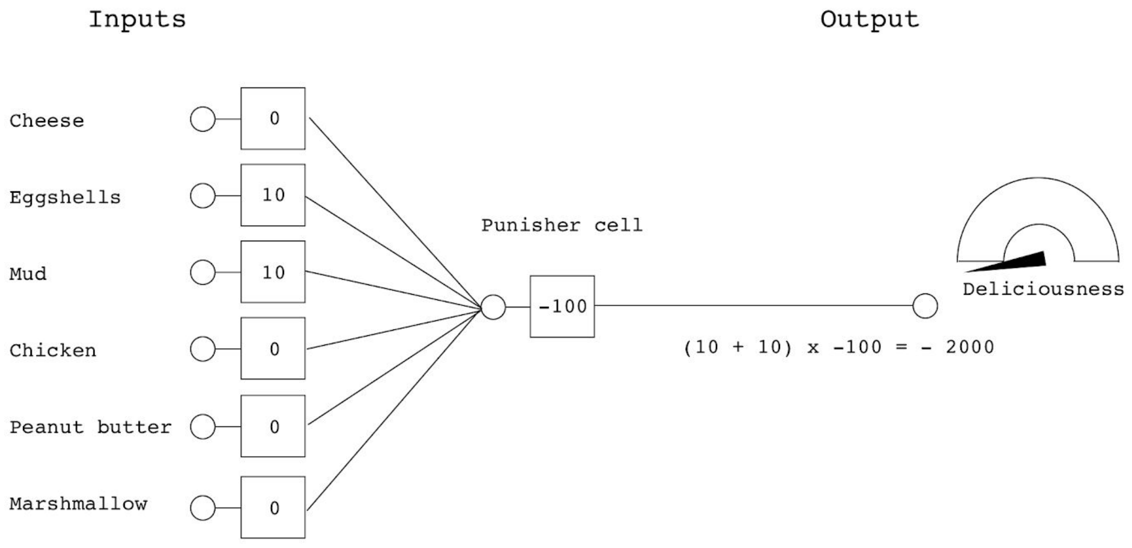
Nature inspired a lot of man’s inventions, such as birds which inspired us to fly. It seemed logical that people might look to the brain’s architecture to be inspired to design an intelligent machine. This is the logic that sparked Artificial Neural Networks (ANNs): an ANN is a machine learning model inspired by the networks of biological neurons found in our brains. Although birds inspired flying, the planes do not have to flap wings. Similarly, ANNs gradually were designed quite differently from their biological cousins. There are different ways to build neural networks based on their application. But they are all loosely modelled after the way the brain works - their cousins are called biological neural networks - are the original, far more complex models. It was built in the 1950s, originally to test theories about how the brain works. They are built from a bunch of simple chunks of software, each able to perform very simple math - called cells or neurons. The power of the neural network lies in how these cells are connected.

The early successes of ANNs led to the widespread belief that we would soon be conversing with truly intelligent machines. When it became clear in the 1960s that this promise would go unfulfilled, they lost funding, and ANNs did not receive much attention. In the early 1980s, new architectures were invented and better training techniques were developed, sparking a revival of interest in them. But progress was slow, and by the 1990s other powerful machine learning techniques were invented. The current interest we are experiencing may not die out so easily and are making a significant impact on our lives.

To get a better understanding about the architecture of neural networks, we will look at an example for making a sandwich. All the inputs (cheese, eggshells, mud, chicken, peanut butter, marshmallow) are connected to an output which is a "deliciousness" scale. Each input has a weight of 1 if it's good and 0 if it's bad. This is a simple one-layered neural network. If a sandwich contains mud, but enough other good ingredients, it might still rate it as good. To get a better neural network, we are going to need another layer of cells.

Each ingredient is now connected to the new layer of cells, and each cell is connected to the output. The new layer is called a hidden layer, because the user only sees the inputs and the outputs. This isn't deep learning yet, but we are getting there. Deep learning is the approach of lots of hidden layers for lots of complexity is known as deep learning.

Back to the sandwich neural network. We can avoid bad ingredients by connecting them to a cell that we'll call the punisher. We'll give that cell a huge negative weight (let's say -100) and connect everything bad to it with a weight of 10. Let's make the first cell the punisher and connect the mud and eggshells to it:



Now, no matter what happens in the other cells, a sandwich is likely to fail if it contains eggshells or mud. We can do other things with the rest of the cells - which ingredient combos work. Let's use the second cell to recognise chicken-and-cheese type sandwiches (deli sandwich cell). We connect the chicken and cheese to it with weights of 1 and connect everything else to it with weights of 0. This cell connects to the output with a modest weight of 1. Higher weight might make the punisher cell less powerful.

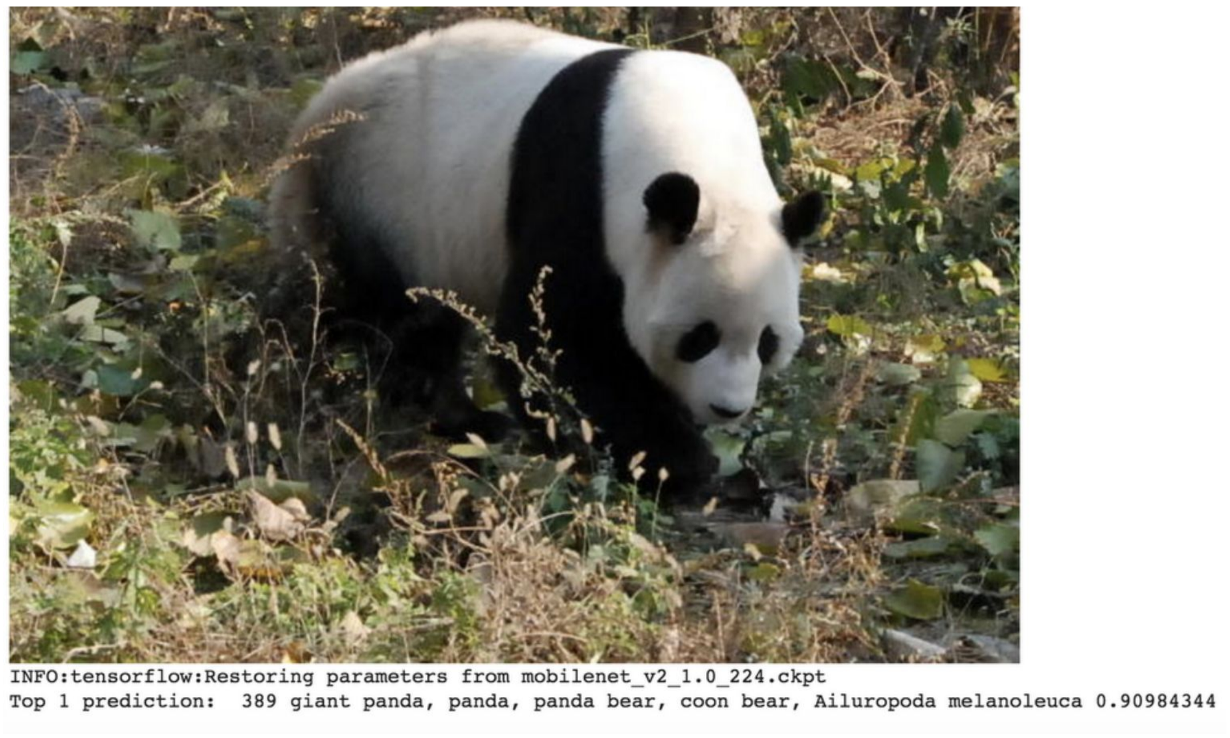
The training process. The point of using machine learning is that we don't have to set up the neural network by hand. It should be able to configure itself. When a neural net starts from scratch, it's very bad and starts with random weights assigned. We need to correct it with examples - it needs to compare its ratings against those of already rated. It learns over time with tons of examples and adjusts its weights.

Failed attempts

Initially, I spent a lot of time trying to identify the right software and workflow which would enable me to achieve what I want. I wanted to build a self-contained work where everything would be available within the website without external software needed. I abandoned a lot of avenues when I realised that they had certain roadblocks which I could not overcome. Below are some of the ones I tried but failed to get them to work the way I wanted.

Google Colab

Previously known as Google Seedbank, I had used Google Colab for text generation. But using it for images seemed to require programming skills beyond my abilities. I was also unable to figure out how to connect it to my website. After a while, I gave up on using this and looked for alternatives.

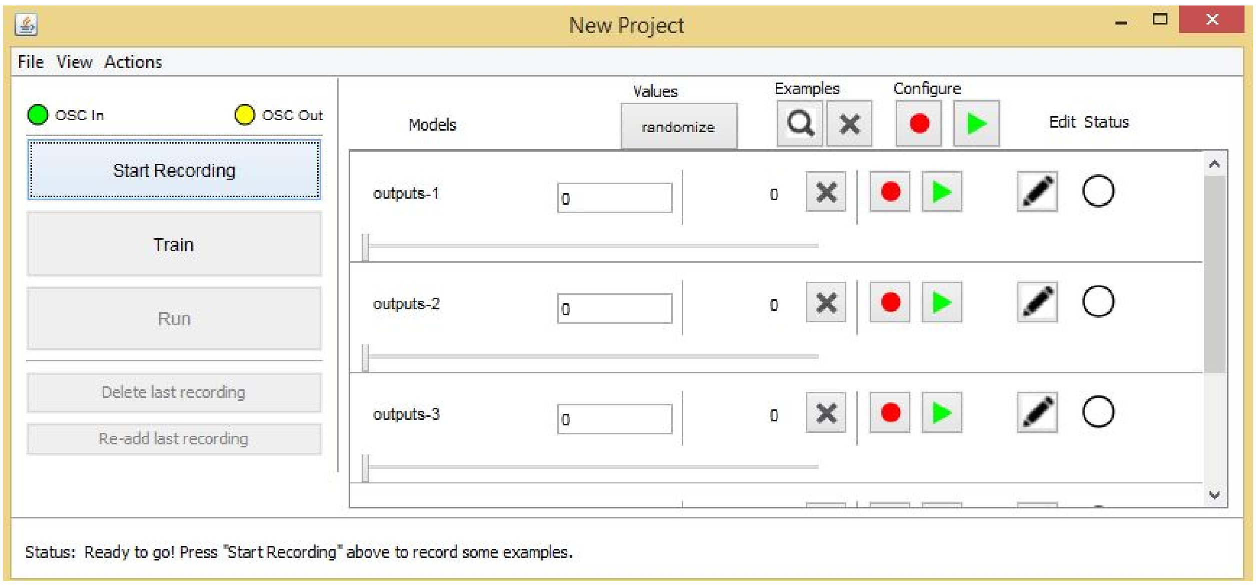


```
INFO:tensorflow:Restoring parameters from mobilenet_v2_1.0_224.ckpt
Top 1 prediction:  389 giant panda, panda, panda bear, coon bear, Ailuropoda melanoleuca 0.90984344
```

They had the MobileNets which I wanted to use for a demonstration. It can be found here: https://aihub.cloud.google.com/p/products%2Fafc076d6-3017-49b2-a5ee-0af4ba514f53

Wekinator

This software is built for artists and musicians. I had my hopes pinned on this and was sure I could use this software for my work. It allows building a wide variety of interactive systems such as musical instruments, gesturally-controlled animations and games, and easily connects to sensors. It does not need knowledge of programming. In the beginning, one needs to understand the concepts taught in this course on Kadenze: https://www.kadenze.com/courses/machine-learning-for-musicians-and-artists/info. After that, creating an interactive system is mainly using sliders and buttons. Although this seemed to be a perfect fit for my project, I could not figure out how to get it to work within a website.



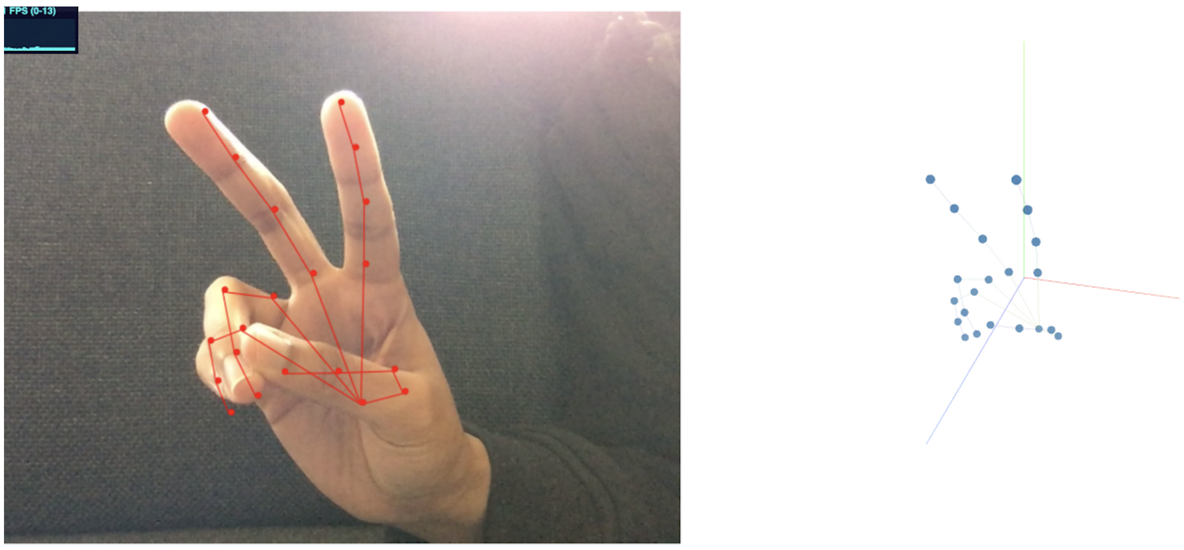
Status: Ready to go! Press "Start Recording" above to record some examples.

More details about it can be found here: http://www.wekinator.org/

MediaPipe

MediaPipe Handpose is a lightweight machine learning pipeline consisting of two models: A palm detector and a hand-skeleton finger tracking model. It predicts 21 3D hand keypoints per detected hand. Given an input, the model predicts whether it contains a hand. If so, the model returns coordinates for the

bounding box around the hand, as well as 21 key points within the hand, outlining the location of each finger joint and the palm. It was built using TensorFlow and could be used in a website. TensorFlow is a free and open-source library used for machine learning applications. But using TensorFlow required significant programming and machine learning skills to use compared to what I know, so I discarded this option. MediaPipe details can be found here:
https://blog.tensorflow.org/2020/03/face-and-hand-tracking-in-browser-with-mediapipe-and-tensorflowjs.html



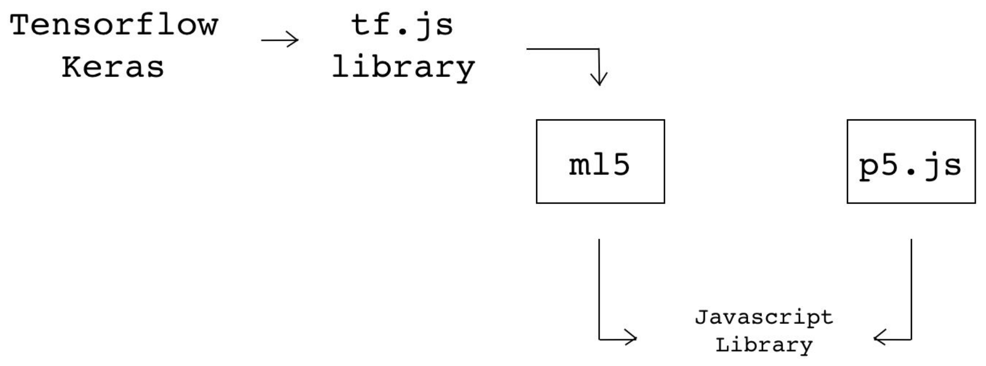
I found that it worked well even when fingers overlap. Live demo of this model can be found here:
https://storage.googleapis.com/tfjs-models/demos/handpose/index.html

Final presentation

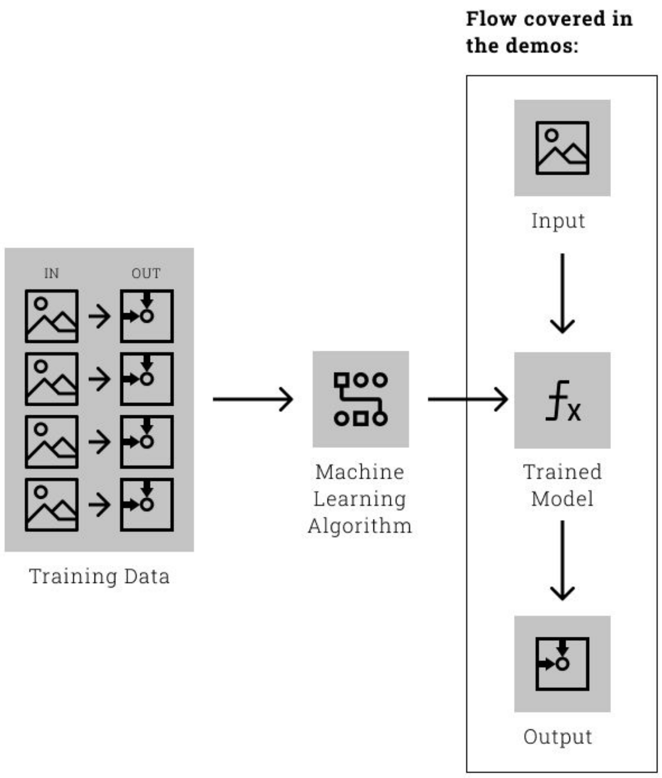
My website is a tool that people can use to discuss/educate others about Machine Learning. Possible target audience include teachers who want to give a glimpse of this topic to their high school students, team leads in a company who want to convince their team members to pay more attention to the topic. It is not for people who have a serious interest in this topic, but to get them interested.

Learning by doing in a specific situation where the user learns the mechanisms adequate to that situation. This is done irrespective of the user’s prior knowledge and experience with machine learning. The presentation format is about doing machine learning in the web browser.

The setup I used is a combination of p5.js and ml5.js. The p5.js is a free and open-source JavaScript library for creative coding, with a focus on making coding accessible and inclusive for artists, designers, educators, beginners, and anyone else. It has a full set of drawing functionality including text, input, video, webcam, and sound. I used this to define a lot of the interactive elements on the website. The ml5.js offers machine learning for the web - it can be directly used in the website. The library provides access to machine learning algorithms and models in the browser, building on top of TensorFlow.js with no other external dependencies. The ml5.js provides immediate access in the browser to pre-trained models for detecting human poses, generating text, styling an image with another, composing music, pitch detection, and common English language word relationships, and much more. Since ml5.js is built in such a way that it works with p5.js, I successfully used it for my demonstrations.



Only a part of the main flow of machine learning is covered in the demos. The initial part of collecting and cleaning training data, selecting a machine learning algorithm and training the model is not shown. This part requires significant knowledge of machine learning to achieve and is not something I wanted to teach through my tool. The later part which interacts with a trained model by taking an input and giving an output is only covered in my demonstrations. The whole flow is shown below diagrammatically and I have marked the parts I cover in a box:



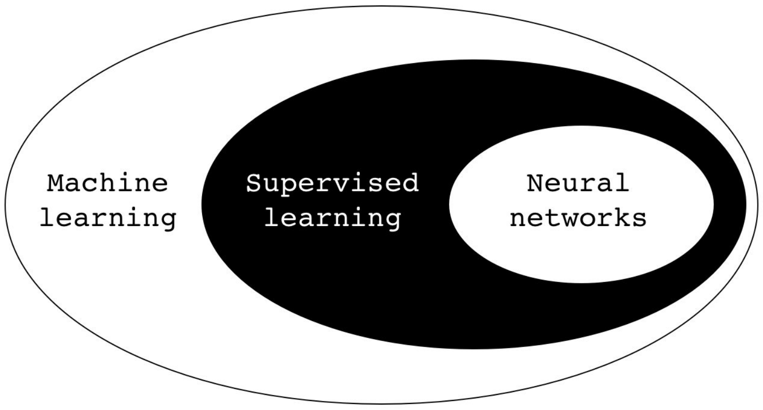
Selecting what to demonstrate

There are two main categories of machine learning - Supervised learning and unsupervised learning.

In *supervised learning*, the algorithm is given a bunch of inputs and a bunch of desired outputs. The algorithm in it’s own, figures out how to give the desired output based on the given input. So when there is a new input the algorithm has not seen before, it will give an output on it’s own. Machine learning algorithms that learn from input/output pairs are called supervised learning algorithms because a “teacher” provides supervision to the algorithms in the form of the desired outputs for each example that they learn from. Neural networks are a popular architecture for supervised learning. We can consider the example of classifying emails as spam or not spam. The algorithm is given a large number of emails (input) which are marked as spam or not spam (desired output). Later when it is given a new email, the algorithm will predict if it is spam or not spam.

In *unsupervised learning*, the algorithm is given the input data but not given the desired output. The learning algorithm is just shown the input data and asked to extract knowledge from this data.Though this is used in multiple applications, it is harder to evaluate and understand. An example of this is segmenting customers into groups with similar preferences. For a shopping site, these might be “parents,” “bookworms,” or “gamers.” Because you don’t know in advance what these groups might be, or even how many there are, you have no known outputs.

Since supervised learning is the most commonly used type of machine learning, I picked this to explain using my interactive demonstrations. There are two major types of supervised machine learning problems, called *classification* and *regression*. In classification, the goal is to predict an output, which is a choice from a predefined list of possibilities. For regression tasks, the goal is to predict a continuous number, or a floating-point number in programming terms (or real number in mathematical terms). Predicting a person’s annual income from their education, their age, and where they live is an example of a regression task. An easy way to distinguish between classification and regression tasks is to ask whether there is some kind of continuity in the output. If there is continuity between possible outcomes, then the problem is a regression problem.



Demonstration 1, Uncovering opacity is a classification task. It classifies the object shown in the webcam into one of the categories it is trained on. Demonstration 2, Train your own Neural Network is a regression task. It moves the ball between the left and right end. There is continuity between possible outcomes.

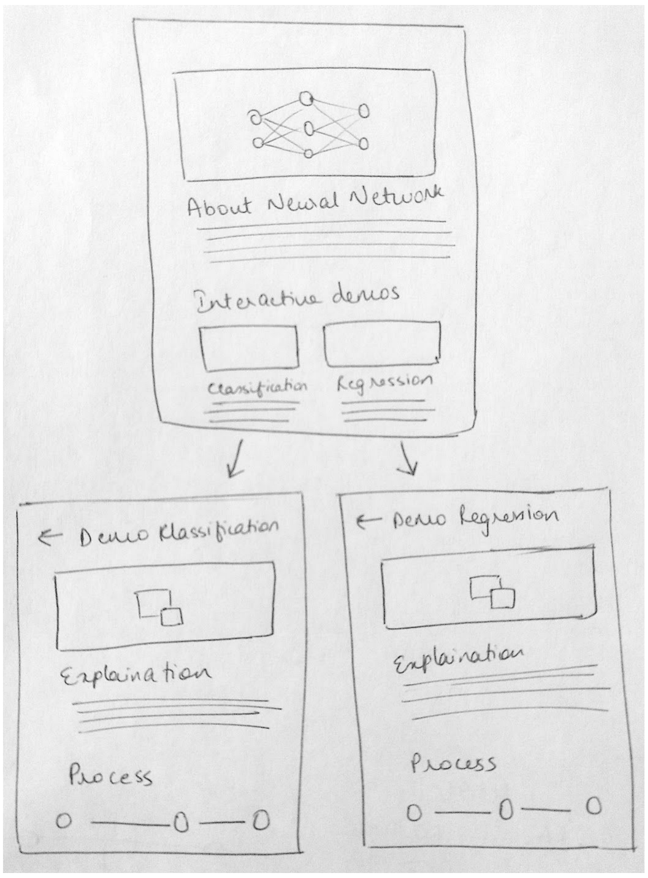
Active learning

I chose to present by work using the teaching concept of “active learning”. It is based on the idea that students use their knowledge to tackle a problem, and in doing so, test and extend their understanding. According to Bonwell and Eison (1991), *Active learning implies that students are engaged in their own learning. Active teaching strategies have students do something other than taking notes or following directions … they participate in activities … [to] construct new knowledge and build new scientific skills.* They require that students do something that needs higher-order thinking. The definition is quite broad and they knew that a range of activities fell into this category.

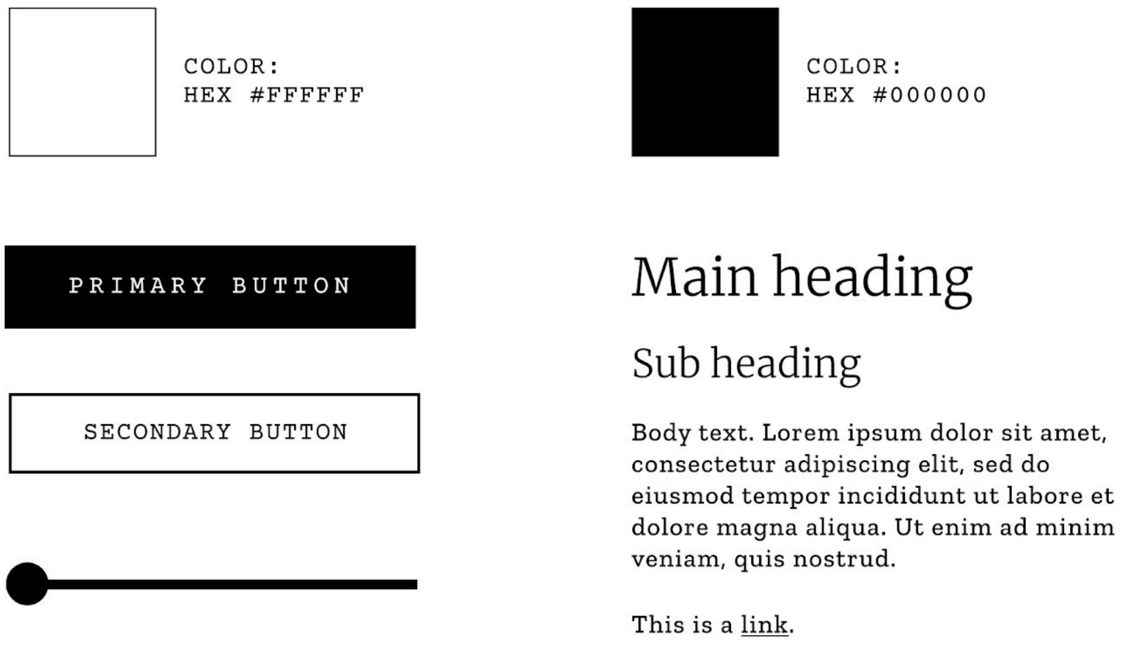
I chose to focus my demonstrations on this method of teaching, to explain two commonly used types of machine learning. This should enable people to learn the concepts and form a basis for discussion with a larger audience.

Website format

I chose to present in a website format since I wanted the information and interactive demos to be accessible anywhere, without being restricted to a single art exhibition. The early wireframe for the website shown below, evolved to include more information and iterated based on feedback. The homepage below covered the main information, but expanded to include more, based on feedback from users who tried it.



The navigation is kept simple with just two levels deep. The key interaction elements used in the website are those that are commonly found in digital products such as buttons, sliders, and links. I wanted the visual aspects to take a backseat to the information shown. I went for a black and white aesthetic with simple square edged buttons. The primary buttons are filled in for more emphasis while the secondary buttons are ghost buttons.



Body text. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud.

This is a link.

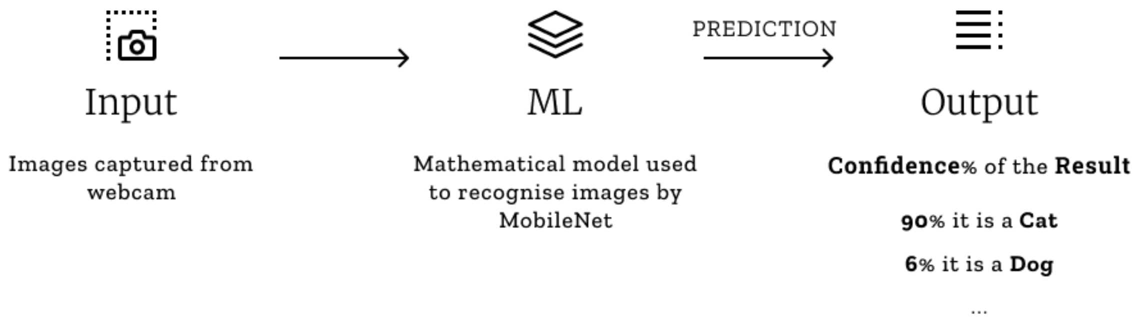
The website is hosted for free on Github. It can be found here: https://pranotit.github.io/Masters_Thesis/. Since it is the free version, it has a "github.io" in the name. The web design code is available on Github: https://github.com/PranotiT/Masters_Thesis.

Demonstration 1: Uncovering opacity

A machine learning model is a mathematical representation of a real-world process. In this demonstration, we use a model which is already trained to recognise images - a pre-trained model. Pre-trained models are used in real-world applications to make decisions about various aspects of our lives. I also use pre-trained models in this project to show how they are limited by the data they are trained on. Yet, they are used to influence important aspects of our lives. When using a pretrained model, it is important to ask who trained that model, why did they train it, what data was used to train it, and how is that data collected? Pre-trained model in my demonstration has learnt based on a fixed dataset. If I show it an image not from it’s dataset, it will not know about it. Below is an early version of the demo.

This hands-on demonstration focuses on the usage of “pre-trained models” in machine learning using image classification. In classification, information is categorized into predefined categories. For example, an email can either be ‘spam’ or ‘not spam’. The user can hold up an object to the webcam and the model will classify the object. Sometimes, the recognition is not accurate. In this demonstration, the user can search through the list of images this model is trained on, to see if the object they are showing is a part of it.

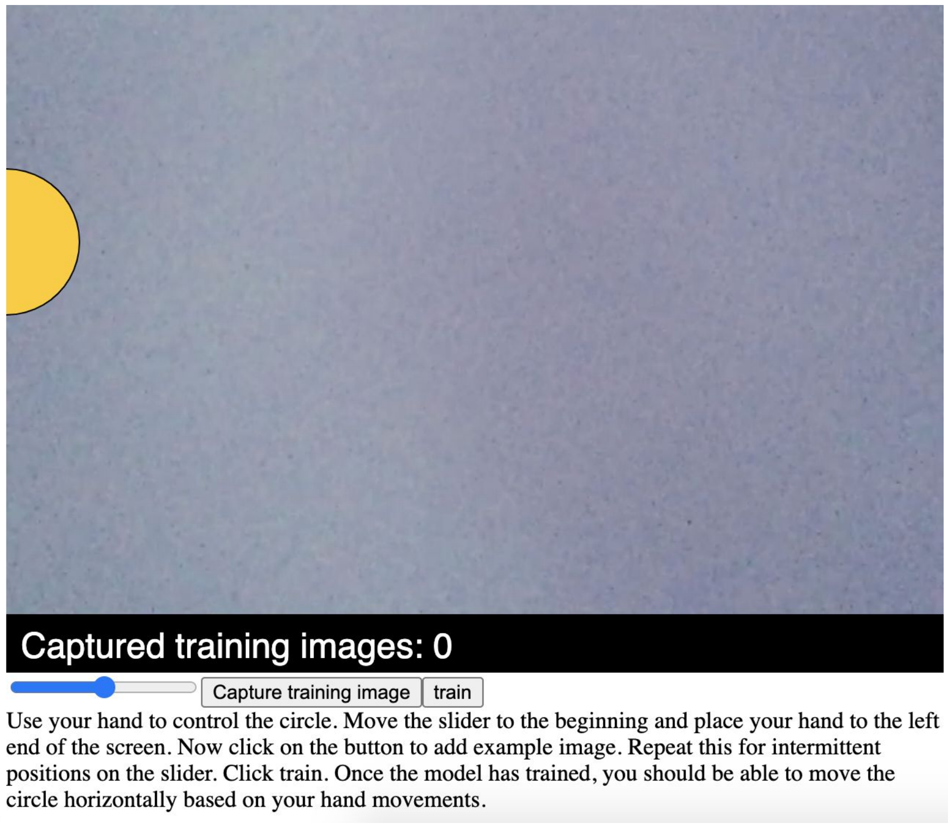
This is a classification problem. The pre-trained MobileNet model has a fixed number of things it knows about - the images it is trained on. It will classify the object you are showing it based on the original 1,000 image categories it is trained on. It is not aware of any other information. Which is why, you will find that oftentimes, the items shown through the webcam are not classified accurately - such as your face! But the result you see will be something from the original 1,000 categories. The training set is from a database of images called ImageNet made up of almost 15 million images. The below diagram shows how the objects are recognized:


Input
Images captured from webcam
ML
Mathematical model used to recognise images by MobileNet
PREDICTION
Output
Confidence% of the **Result**
90% it is a **Cat**
6% it is a **Dog**
...

Real world application: We have often encountered classification problems where machine learning is used. For example, the spam filters in your email. They are trained on a huge database of e-mails and classify if an email is "spam" or "not spam". It adapts over time and learns as you mark or unmark new emails as "spam" or "not spam". This is an example where machine learning makes it easy and it would be nearly impossible to have humans do this task. The sheer number of emails each one of us gets is huge and would be tough for humans to sort through each one.

Demonstration 2: Train your own Neural Network

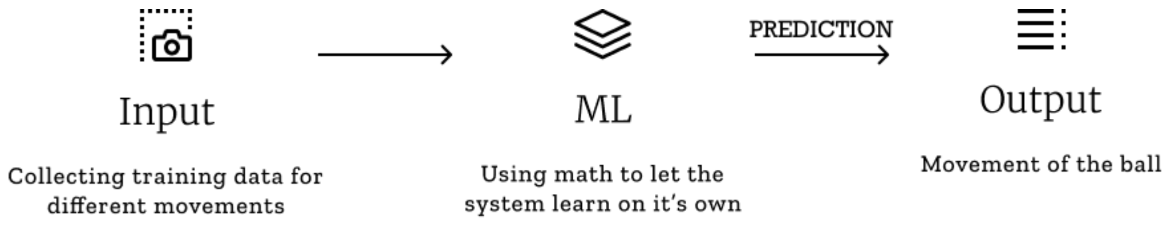
This is a hands-on project where the user can train their own Neural Network starting from a blank state, to get a brief insight into the steps involved. In the previous project on opacity, we looked at classification. Here we will look at an example of regression. Regression is where the output is on a scale or a dial. For example, instead of giving an output that someone is "happy" or "sad" (which is classification), regression gives an output of how happy or sad someone is. Below is an early version of the demo.



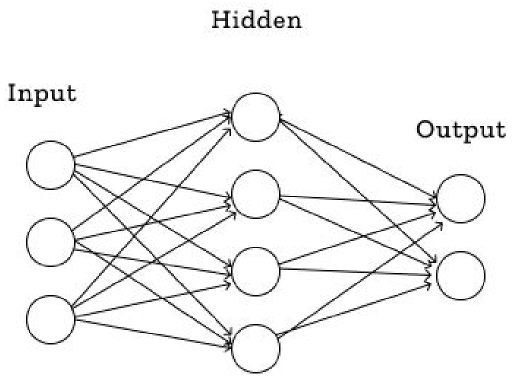
Use your hand to control the circle. Move the slider to the beginning and place your hand to the left end of the screen. Now click on the button to add example image. Repeat this for intermittent positions on the slider. Click train. Once the model has trained, you should be able to move the circle horizontally based on your hand movements.

The user can control the movement of a circle with your body. They go through the following main steps: (1) collecting training data, (2) training the model, and (3) prediction or making an inference. The slider can be used to move the circle from left to right. The aim is to replace the slider and control the movement using our own body. For this project, the movement is restricted from the left to the right. In the steps mentioned on the website, I mention moving your hand left to right, but the user can use any other part of your body such as legs, head movement, etc. The user can also use another object in your hand, as long as it is a progressive motion.

The user trains it using the following steps:


Input
Collecting training data for different movements
ML
Using math to let the system learn on it's own
PREDICTION
Output
Movement of the ball

Structure of a Neural Network:



Real world application: We have often encountered regression problems where Machine Learning is used - such as in forecasting or predictions. Weather forecasting often uses this method. Based on data, weather is predicted. It does not just say that the weather will be hot or cold, but rather how hot or how cold and give the temperature value on a scale. For a human to crunch through the vast amount of data and predict the weather for every location on the planet, everyday would be nearly impossible. This is a situation where machine learning is useful to use.

Live website

The final website can be found at:
https://pranotit.github.io/Masters_Thesis/

Through the interactive demonstrations, I was able to provide a peek into how machine learning works. It also offered some hints on what type of questions to ask, such as what the dataset consists of, or what was the motivation behind designing the program. I also provide some links to further reading to those who are seriously interested in the topic.

Feedback from users

I asked people who interacted with my website to answer the below questionnaire. The aim of this questionnaire was to understand if my aim through the thesis was achieved and feedback. I used Google forms to conduct it.

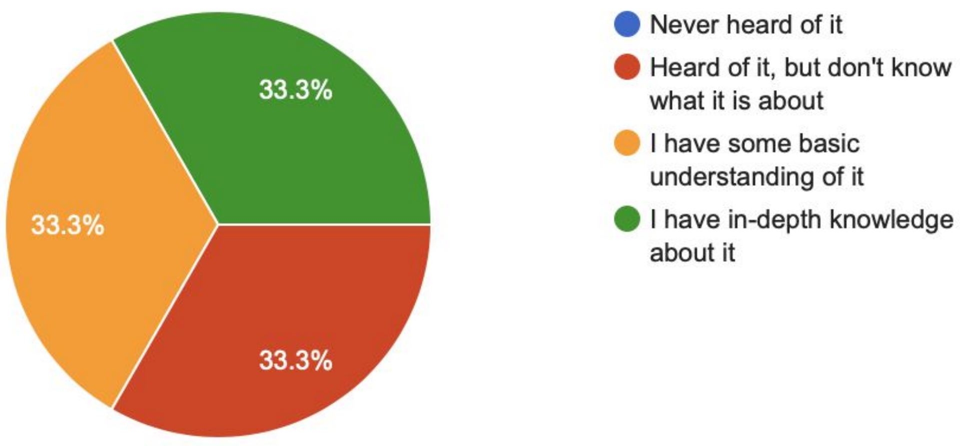
Disclaimer: This feedback is only partially representative, since it is not conducted in a seminar/workshop setting as aimed. The 6 participants used the website on their own unobserved.

The entire questionnaire and responses in excel format can be found here:
https://docs.google.com/spreadsheets/d/15faoxMJhqDq2HdoSzU0mc56DoQiBEfGqMk6ifytg1sc/edit?usp=sharing

Questionnaire with key responses:

1. **How familiar are you with machine learning before interacting with the website?**
    a. Never heard of it
    b. Heard of it, but don't know what it is about
    c. I have some basic understanding of it
    d. I have in-depth knowledge about it
    e. Other

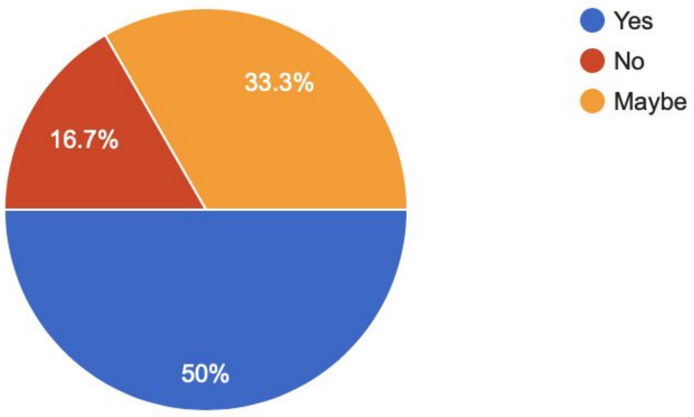
Replies: The participants varied in their knowledge equally about machine learning from not knowing much to being experts in the field.



2. **What is your take-away from the website?**

Replies: Some replies found it nice that they learnt about machine learning, and it can be taught without having to know a lot of scientific information and that the website explains some of it to the layperson. Another mentioned that neural networks help identify patterns in the real world.

3. **Did the interactive demos help you understand better?**
    a. Yes
    b. No
    c. Maybe



Replies: 50% of the participants said "yes", 16.7% said "no" while 33.3% said Maybe.

4. **If you meet the creator of a machine learning program, what question/s would you ask them about their program?**

Replies: They varied from asking about how it works, why, to ethics.

5. **How would you explain machine learning to someone else?**

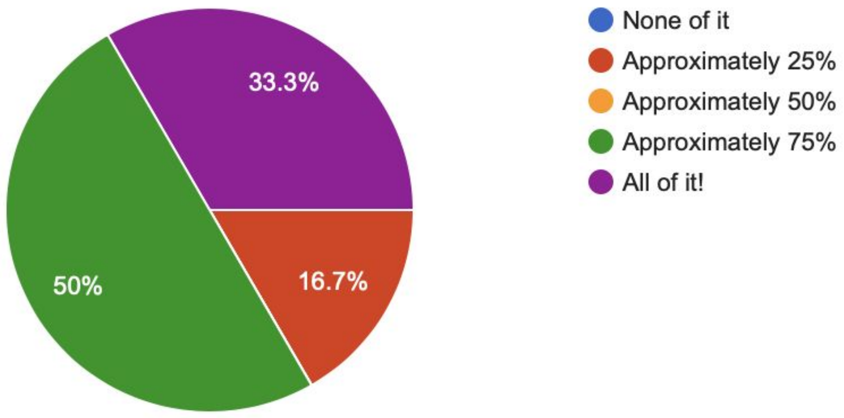
Replies: Most of the answers were along similar lines of training computers to go through large amounts of data.

6. Do you have feedback about the website?

Replies: Some mentioned that it was too much text to read and another participant mentioned that the information could probably delve a bit deeper. Some asked for better optimization of the website for mobile usage.

7. How much of the text on the website did you read?

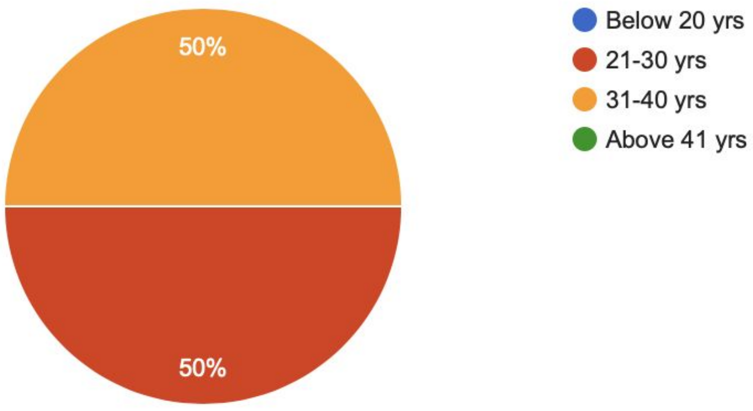
a. None of it
b. Approximately 25%
c. Approximately 50%
d. Approximately 75%
e. All of it!



Replies: 50% of the participants read approximately 75% of the text. Surprisingly, 33.3% read all of it, and 16.7% read approximately 25% of the text.

8. What is your age group?

a. Below 20 yrs
b. 21-30 yrs
c. 31-40 yrs
d. Above 41 yrs



Replies: The respondents were evenly divided between the age groups of 21-30yrs and 31-40yrs.

Conclusion

Based on feedback received through the questionnaire, I feel that I was successful in achieving the goals I set out to. This is taking into consideration my limited skill set with programming and designing based on it. Reading a lot of text was a consistent negative feedback received. But since the goal of the presentation is that someone conducts a workshop and not just sits and reads the text on their own, it is something I have left unchanged for now. There is a lot of room for improvement and further testing in real workshop scenarios to improve the work.

What I learnt

Finding the right software/tool to use to achieve what I want, proved to be very difficult. It took a lot of research, trial and errors, and failed attempts. It helped me learn the limitations of the various tools and know that finding the right tool is important.

Learning by doing is a great way for people to learn about a concept. During my tests with users, I found that while they interacted with the demo, it brought up different topics of conversation and the demo provided a good context to talk about it. During demo 1: "uncovering opacity", people always liked to try showing different objects from around them to see if the program was able to classify the object correctly. This brought up the conversation why, and I was able to explain the importance of the data set it is trained on and about pre-trained models.

The importance of getting the UX right, is something I faced with demo 2: "Training your own neural network". I found that people did not often read the instructions and tried to already interact with the demo. When they read the instructions, they were forced to remember it and scroll down since the instructions and actual interaction were now side by side. I solved this partly by placing text only instructions near the interaction. This helped partly but has room for further improvement.

References

Shane, Janelle. (2019) *You Look Like a Thing and I Love You: How Artificial Intelligence Works and Why It's Making the World a Weirder Place.* ISBN-13 : 978-0316525244

O'Neil, Cathy. (2016) *Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy.* ISBN-13 : 978-0553418811

Mayer-Schönberger, Viktor, and Cukier, Kenneth. (2014) *Big Data: A Revolution That Will Transform How We Live, Work, and Think.* ISBN-13 : 978-0544227750

Müller, Andreas, Guido, Sarah, (2016) *Introduction to Machine Learning with Python.* ISBN: 9781449369415

Géron, Aurélien, (2019) *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition.* ISBN: 9781492032649

Perrotta, Paolo, (2020) *Programming Machine Learning,* ISBN: 9781680506600

Fenner, Mark (2019) *Machine Learning with Python for Everyone,* ISBN: 9780134845708

Shiffman, Daniel, (2015) *Learning Processing: A Beginner's Guide to Programming Images, Animation, and Interaction.* ISBN-13 : 978-0123944436

Brame, Cynthia, (2019) *Science Teaching Essentials,* ISBN: 9780128147030

Anzai, Y., & Simon, H. A. (1979). The theory of learning by doing. Psychological Review, 86(2), 124–140. https://doi.org/10.1037/0033-295X.86.2.124

Howard, A., Zhu, M., et all. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. https://arxiv.org/abs/1704.04861

IBM Cloud Education. IBM Cloud Learn Hub > Machine Learning. As edited on: 15 July 2020 https://www.ibm.com/cloud/learn/machine-learning

Image database ImageNet. 2016 Stanford Vision Lab, Stanford University, Princeton University. http://image-net.org/

ml5 library built on top of TensorFlow.js with no other external dependencies. https://ml5js.org/

Johnson, K., (2020) ExamSoft's remote bar exam sparks privacy and facial recognition concerns. https://venturebeat.com/2020/09/29/examsofts-remote-bar-exam-sparks-privacy-and-facial-recognition-concerns/

Surette, S., (2020) How should the FDA regulate adaptive AI, software that designs itself? https://www.statnews.com/2020/10/02/how-should-fda-regulate-adaptive-ai/