FULL SELF-DRIVING, SKYNET AND OTHER AI MYTHS

BRAD FLAUGHER

Full Self-Driving, Skynet and Other Artificial Intelligence Myths

Modern decision making with deep machine learning models

Brad Flaugher

February 14, 2023

We have now accumulated sufficient evidence to see that whatever language the central nervous system is using, it is characterized by less logical and arithmetical depth than what we are normally used to.

– John von Neumann, 1958 [1]

Preface

This book is an attempt to organize my thoughts on decision making with computers. I had the pleasure of publishing on this subject in the Journal of Computational Neurodynamics back in 2008. Since then, I've kept up with the latest developments in the field, especially the advancements in deep learning using artificial neural networks, or what some call "AI." But my interests didn't stop there – for over a decade, I've been working on high-speed decision making operations in finance and online advertising. You may have even seen some of my work while browsing the web. Nowadays, I enjoy training young machine learning engineers (https://bradflaugher.com/bootcamp.html), as well as contracting for (https://inoxoft.com/) and investing in startups (https://ventures.nextfab.com/about/).

But enough about me. Let's talk about the book. I've noticed that most writing on AI either requires a technical background or is heavily influenced by biased investors, employees of AI companies, or technophobes who refuse to give it a chance. My goal with this book is to provide readers with a comprehensive understanding of how AI works, so they can master it for themselves. You'll learn about how models are trained, with in-depth analysis of 34 common use cases in chapter four. I also dedicate entire chapters to autonomous weapons and self-driving cars, and provide a framework for understanding how deep learning models will transform careers and jobs in the concluding chapter.

Whether you're a user, developer, or investor in AI, I hope you'll find this book useful. Feel free to skip around and read the stories that interest you most - although I highly recommend the first three chapters as a must-read for everyone. I've enjoyed writing this book and I hope you'll enjoy reading it. And if you have any notes or feedback, you can always reach me at brad@bradflaugher.com.

Brad Flaugher

PS Thank you Angela Altamirano for the cover wonderful cover art. Thank you to my early supporters, William Dickson, Daniel Hoevel, Patrick Maloney, Jerrod Howlett and Jeff Zinser. A special thank you to Louis Cid and the Harvard Club of New York for giving me an early forum. Finally thank you to Dr. Péter Érdi for putting me under his wing many years ago and Dr. Elisa Esposito for her constant support.

PPS This book uses the wonderful free and open source kaobook LaTeX project (https://github.com/fmarotta/kaobook) and many Free and Open Source AI models for text and image generation like Stable Diffusion (https://huggingface.co/spaces/stabilityai/stable-diffusion) and Mann-E (https://huggingface.co/mann-e/mann-e_4_rev-0-1) and some commercial models like GPT-3, ChatGPT and Dall-E from OpenAI (https://openai.com/).

Contents

Chapter Summaries

1 History: The End of Good Old-Fashioned Artificial Intelligence

Avoid the term "AI" and instead differentiate between rules-based programming and deep learning to prevent confusion. Good Old-Fashioned AI (rules-based AI) is challenging to create and maintain, while modern deep learning techniques use data to train models and can be as good as the data they are trained with. Good Old-Fashioned AI is not well-suited for many complex tasks, like translation and handwriting detection.

2 The Regression Theory of Everything

Deep learning models are large unscientific regression systems that map input data to output data. They are complex, deterministic and exhibit chaotic behavior, making their inner workings functionally unknowable and difficult to test. Multicollinearity and feature importance can only be understood with a high level of statistical error. Getting good data to train with is crucial for machine learning engineers to train good models. While deep learning models can have impressive and useful outputs, it’s important to acknowledge their failures and limitations, which can be encouraged by users, managers and investors.

3 Creativity and Decision Making with Deep Learning Models

Users should consider the quality and appropriateness of the data the model is trained on and ask questions about how and when the data was collected and cleaned up. Deep learning models are not sentient creative creatures and are deterministic systems that can only generate outputs based on their training data. Users should also avoid allowing models do everything and manage the models cautiously and carefully.

4 Model Cards and Case Studies

An analysis of 30 machine learning models, from online dating, stock trading, threat detection and more.

5 Self-Driving with Statistics

A discussion of the limitations of deep learning models and how they can be used and controlled in self-driving cars. A roadmap of likely developments and changes in infrastructure and control systems to support fully autonomous vehicles.

6 Unplugging Skynet

An analysis of autonomous weapons, their risks and development.

7 Revolutionary for Whom?

An analysis of the past and future of work and automation, and a framework for understanding how daily tasks and careers will evolve over time.

History: The End of Good Old-Fashioned Artificial Intelligence

1

"Programming will be obsolete. I believe the conventional idea of "writing a program" is headed for extinction, and indeed, for all but very specialized applications, most software, as we know it, will be replaced by AI systems that are trained rather than programmed. In situations where one needs a "simple" program (after all, not everything should require a model of hundreds of billions of parameters running on a cluster of GPUs), those programs will, themselves, be generated by an AI rather than coded by hand." Matt Welsh, 2023 [2]

1.1 The AI Textbook in 1997

Dr. Elaine Rich's textbook on artificial intelligence (AI), published in the 1980s, was a groundbreaking work that helped to establish many of the core concepts and techniques in the field of AI. However, the rapid advancements in AI over the past few decades have led to many of the chapters in this textbook becoming obsolete.

One of the main reasons for this is the prevalence of deep learning, big data, and large-scale statistical models in modern AI. These techniques have largely replaced the symbolic, rule-based approach to AI that was emphasized in the textbook, making many of the chapters on knowledge representation and expert systems less relevant.

Additionally, the explosion of data and the availability of powerful computing resources have made it possible to apply machine learning techniques at a scale that was previously unimaginable. This has led to the development of highly effective machine learning models that can handle complex tasks such as image and speech recognition with a high degree of accuracy, making many of the chapters on simpler machine learning techniques such as decision trees[1] and linear regression less relevant. [3] [2]

We'll discuss this history and a few examples from the "early days" of AI to help us understand where we are headed. We'll start with machine translation, then discuss chess and finally neural networks, which will be the focus of the rest of this book.

1: Although mathematically neural networks are decision trees https://arxiv.org/abs/2210.05189

[3]: Rich et al. (2009), *Artificial Intelligence*

2: the book is now in its third edition and unlikely to be updated as Dr. Rich as retired https://www.cs.utexas.edu/~ear/

1.2 Does AI Need to Know Grammar to Translate?

Noam Chomsky is a linguist and philosopher who has made significant contributions to the field of linguistics with his theory of universal grammar. Chomsky believes that all human languages share a common underlying structure, and that this structure is innate to humans. He proposes that this innate structure is the result of a "language acquisition device" present in the human brain, which allows us to learn and produce language. Chomsky also argues that the structure of language is largely independent of its content, and that the ability to produce and understand language is a fundamental aspect of human nature. His theory has been influential in the field of linguistics and has sparked much debate and research on the nature of language and its relationship to the human mind.

For English speakers or anyone who has learned English as a second language you'll have many examples of special cases, irregular verbs, bad English and former street slang that became good and proper over time. For programmers this is a nightmare, how can we codify human knowledge in a timely fashion? If we tried to write the rules of the English language in code (which many have tried to do) the rules themselves might change before we were finished writing them.

Explicitly translating languages through code is a difficult task because it requires a thorough understanding of the grammar, vocabulary, and syntax of both languages, as well as the nuances and subtleties of their respective cultures[3]. Simply coding rules for how to translate words or phrases from one language to another is not sufficient, as there are often multiple valid translations for a given phrase depending on the context in which it is used.

3: For programmers this is a nightmare, how can we codify human knowledge in a timely fashion? If we tried to write the rules of the English language in code (which many have tried to do) the rules themselves might change before we were finished writing them.

A more effective approach to translation is to use statistical techniques that rely on a large corpus of translated data, such as Canadian laws[4]. This type of data-driven approach involves training a machine learning model on a large dataset of translations, allowing it to learn the patterns and relationships between the languages. The model can then use this knowledge to make educated translations of new phrases or sentences, taking into account the context in which they are used.

4: They're in French AND English, which is useful data that we can use to correlate phrases and transform English to French and vice-versa.

While this approach is not perfect, it has proven to be highly effective in machine translation and can produce accurate translations even for languages that are very different from each other. The use of a large dataset of translations also allows the model to learn from the mistakes and variations present in real-world translations, further improving its accuracy.

1.3 Explicit Rules and Codified Human Knowledge

When we "teach" a computer to perform a task by explicitly writing down all of the rules of that task, we are really codifying human understanding.[5] When we codify human understanding we write down every rule that we know explicitly. For small tasks we can do this with 100 percent accuracy, and only minor headache on the part of the software developer.

5: Programming this way makes some software development totally boring, I almost switched my major in college to math after considering what a life would look like manually writing rules for handling "edge cases" for the rest of my natural life.

For example, let's write a boring function to tell you the number of days for a given month.

```python
def days_in_month(year, month):
  if month in [1, 3, 5, 7, 8, 10, 12]:
    return 31
  elif month in [4, 6, 9, 11]:
    return 30
  elif month == 2:
    if (year % 4 == 0 and year % 100 != 0) or year % 400 == 0:
      return 29
    else:
      return 28
  else:
    return "Invalid month"
```

Writing code can be a tedious and repetitive task, especially when it comes to debugging and testing. It can be especially frustrating when you're working on a large project and you're trying to track down a specific bug that's causing the program to crash. Testing code can also be boring, as it often involves running the same tests over and over again to ensure that the code is working correctly.

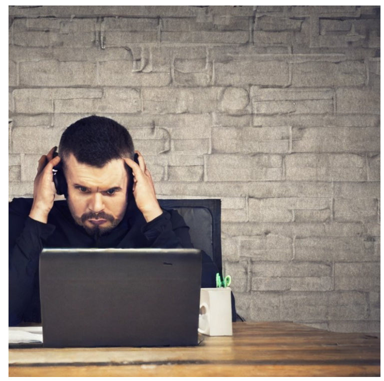

Figure 1.1: "a frustrated programmer writing boring rules on his computer" made with Stable Diffusion 2.1

Additionally, writing code can be boring because it requires a lot of concentration and focus. It can be easy to get lost in the details and lose track of time, especially if you're working on a complex problem. It can also be challenging to come up with creative solutions to problems, and it can be frustrating when your code doesn't work as expected.

While writing and testing code can be rewarding and fulfilling, it can also be a tedious and boring process. It requires a lot of patience, persistence, and attention to detail, and it can be easy to get frustrated and lose motivation. However, with practice and perseverance, it is possible to overcome these challenges and find enjoyment in the process of writing and testing code.

AI has traditionally operated by explicitly codifying human knowledge into machine-readable formats by doing the boring job of coding. This approach, which I'm calling "codified human knowledge" relies on humans to carefully structure and organize information in a way that can be understood by the AI system. The AI system then uses this structured knowledge to make decisions and perform tasks.[6]

6: Some smart people think that AI spells the "End of Programming" https://cacm.acm.org/magazines/2023/1/267976-the-end-of-programming/fulltext, I think the headline is clickbait but the general idea that programming is changing along with advances in AI is true. We are all data scientists now!

However, recent advances in AI have largely ignored the knowledge representation problem and instead have focused on using statistical techniques and neural networks to automatically learn patterns and relationships in data. This approach, known as "deep learning," involves training large neural networks on vast amounts of data, allowing the AI system to make educated classifications and transformations of data without explicit human guidance.

Deep learning has proven to be highly effective in a variety of applications, such as image and speech recognition, and has contributed to the rapid progress we have seen in AI in recent years. However, the reliance on large amounts of data and the lack of transparency in these models can make it difficult to understand how they are making decisions, which can be a concern in certain applications (hence the title of this book).

1.4 IBM Tries Every Possible Chess Move

Deep Blue was a revolutionary computer developed by IBM that was specifically designed to play chess at the highest level. It was programmed with a vast database of chess knowledge and was able to analyze millions of positions per second.

Garry Kasparov was the reigning world chess champion at the time, and he was considered to be one of the greatest players in history. He was beaten by Deep Blue, which used rules based GOFAI (Good Old Fashioned AI) to essentially calculate every possible move and project that move to the end of the game, then choose the best position by brute force. Despite Kasparov's best efforts, he was no match for Deep Blue's computational power. In the end, Deep Blue emerged victorious, defeating Kasparov in a historic match that changed the world of chess forever.

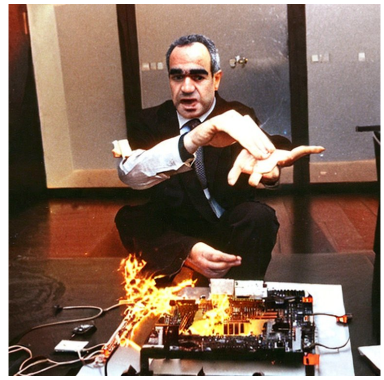

Figure 1.2: "Garry Kasparov setting a computer on fire" made with Stable Diffusion 2.1

Deep Blue was a turning point in the development of AI, but Deep Blue's methods (namely calculating every possible outcome of a Chess game to determine the best move) was not suitable for many of the world's problems. It turns out that Chess is fun, but the world is not like chess.[7] The "real"[8] future of AI was being developed elsewhere, using statistics and a toy model of the brain to solve a very practical problem for banks.

7: if you would like to read an example of the simplest chess engine that I could imagine that is written in a similar "codified" way as Deep Blue check out https://github.com/thomasahle/sunfish/blob/master/sunfish.py

8: We might go back, and try again to more explicitly code everything up, and in some cases we still need to, but from the author's perspective, we live in a deep learning/neural network world.

1.5 Statistical Analysis of Handwriting is the Way of the Future

It was the early 1990s and Yann LeCun was a researcher at Bell Labs in New Jersey. At the time, the process of reading and processing checks was a tedious and time-consuming task that was done manually by bank employees. LeCun saw the potential for using artificial intelligence to automate this process, and he began experimenting with using convolutional neural networks (CNNs) to recognize patterns in images of checks.

At the time, CNNs were a relatively new type of neural network that had been developed in the 1980s for image recognition tasks. They were inspired by the structure of the human visual system, and were able to process images in a way that was similar to how the human brain does.

LeCun's work was groundbreaking, and he was able to achieve impressive results using CNNs to process checks. By 1993, he had developed a system that was able to read and process checks with a high degree of accuracy, significantly reducing the amount of time and effort that was required to process checks manually.[9]

LeCun's work on using CNNs for check processing was a major milestone in the field of artificial intelligence, and it laid the foundation for the development of many other applications of CNNs in the years that followed. Today, CNNs are widely used in a variety of applications, including facial recognition, image classification, and natural language processing. [10]

9: CNN digit OCR models are frequently featured in beginner training tutorials for deep learning libraries like PyTorch and Tensorflow, check one out https://github.com/jonkrohn/DLTFpT/blob/master/notebooks/alexnet_in_tensorflow.ipynb

10: check out Yann LeCun domonstrating a convolutional neural network in 1993 at https://www.youtube.com/watch?v=FwFduRA_L6Q.

1.6 Less Programmer Intelligence and More Data Intelligence

I think it's useful to separate the knowledge in the AI problem-space into two groups. The data and the programmer together make the programs that we use every day, and for the rest of this book I'll try and separate the discussion of the smarts of each to help us better understand the world. [11]

Figure 1.3: "a group of computer programmers striking outside of Microsoft's offices with placards saying 'rule-based programming is boring'" made with Dall-E 2

Early AI relied heavily on a human programmer to design, write, and debug computer programs. Good programmers needed domain expertise, problem-solving skills, logical thinking, and the ability to learn and adapt to new programming languages and technologies.

We are now in the age of big data, and everyone knows that "data is gold". Statistical AI methods that are now most prevalent rely on extracting meaningful insights and knowledge from large datasets. This involves using statistical and analytical methods to discover patterns and trends in data, and using this information to inform business decisions or solve problems.

Throughout this book I'll discuss the interaction between programmer and data, and what can go wrong. Working with big data and statistics at a large scale has given AI tremendous ability, but has made understanding and testing models infinitely more difficult. It is your authors belief that understanding the nuance of this interaction between programmer and data is essential to understanding modern AI.

1.7 From Explicit Rules to a Black Box, and Beyond

Artificial intelligence (AI) has come a long way since its inception, and the way that it makes decisions has changed significantly over time. In the early days of AI, explicit rules were used to tell the AI system what to do in certain situations. These rules were often written by humans and encoded into the system, and the AI would follow them to make decisions.

However, with the advent of deep learning, we have started to rely on a statistical understanding of the truth for AI to make decisions. Deep learning is a type of machine learning that involves training artificial neural networks on large datasets. These neural networks are able to learn patterns and relationships in the data, and can use this knowledge to make decisions.

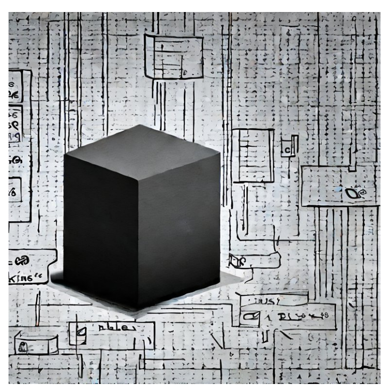

Figure 1.4: "from explicit rules, to a black box and beyond" made with Stable Diffusion 2.1

The use of deep learning has led to the development of powerful AI that is able to perform tasks that were previously thought to be impossible for a machine to handle. For example, deep learning has led to the development of AI systems that are able to recognize faces, translate languages, and even beat humans at complex games like chess and Go.

While deep learning has led to significant advances in AI, it has also made it harder to debug and understand how the AI system is making its decisions. With explicit rules, it was relatively easy to understand why the AI made a particular decision. However, with deep learning, it is often difficult to understand exactly how the AI arrived at its decision. This can make it challenging to troubleshoot problems with the AI system and to ensure that it is making decisions that are fair and unbiased.

AI has come a long way since its early days, and the way that it makes decisions has changed significantly over time. While explicit rules were once used to tell the AI what to do, we now rely on a statistical understanding of the truth for AI to make decisions. This has led to the development of powerful AI that is able to perform a wide range of tasks, but it has also made it harder to debug and understand how the AI is making its decisions.[12]

12: From here on out I'll try and use "GOFAI" (Good Old-Fashioned Artificial Intelligence) to describe the rules-based approach, and use "Deep Learning" or "Machine Learning" to talk about modern neural network and statistics-based techniques

1.8 Key Takeaways

- **AI is a misleading term**, we should instead talk about rules-based programming (GOFAI) and deep learning so we don't confuse ourselves, our partners and our users.[13]
- Good Old-Fashioned AI **(rules-based AI) is hard to create and maintain**. We used to use it for chess engines (like Deep Blue) and translation, but now programmers favor using statistical machine learning techniques.
- Good Old-Fashioned AI is not well suited to many problems, like machine translation and handwriting detection.
- Modern Deep Learning techniques use data to train models instead of humans explicitly writing rules, and the **deep learning models are often as good as the data they are trained with**.[14]

13: Avoid the imprecise words "Algorithms" and "AI" and instead consider the more precise "rules-based AI" AKA "GOFAI" to describe situations where programmers explicitly write rules that they design and "deep learning" or "machine learning" where programmers create models using a dataset and let the machine figure out the rules via a neural network.

14: More *intelligent* data and maybe less intelligent programmers, or maybe programmers who are better trained in statistics and complex systems theory instead of "classical" computer science.

The Regression Theory of Everything | 2

"AI Scientists disagree as to whether these language networks possess true knowledge or are just mimicking humans by remembering the statistics of millions of words. I don't believe any kind of deep learning network will achieve the goal of AGI [Artificial General Intelligence] if the network doesn't model the world the way the brain does. Deep learning networks work well, but not because they solved the knowledge representation problem. They work well because they avoided it completely, relying on statistics and lots of data instead. How deep learning networks work is clever, their performance impressive, and they are commercially valuable. I am only pointing out that they don't possess knowledge and, therefore, are not on the path to having the ability of a five-year-old child." Jeff Hawkins, 2022 [4]

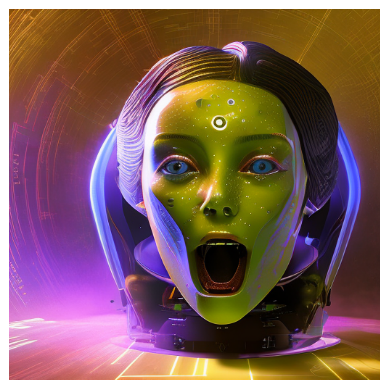

Figure 2.1: "mdjrny-v4 a disembodied computer being force-fed data like a duck looking somewhat like a sci-fi character 8k" made with Mann-E

2.1 Let's Avoid Knowledge Representation!

The knowledge representation problem in AI is the challenge of how to formally represent knowledge in a way that a computer can understand and reason about. This typically involves creating a set of symbols, rules, and structures that can be used to represent concepts, relationships, and other types of information. The goal is to create a representation that is both expressive enough to capture all relevant aspects of the domain, and computationally tractable enough to allow for efficient reasoning and inference. There are many different approaches to knowledge representation, including logic-based, semantic networks, frames, and ontologies, each with their own strengths and weaknesses.

Deep learning techniques handle knowledge representation differently than traditional symbolic AI methods. Unlike symbolic AI, which relies on explicit and hand-coded representations of knowledge, deep learning techniques learn to represent knowledge implicitly through the use of neural networks.

In deep learning, knowledge is represented in the form of the weights of the neural network. These weights are learned through training on a large dataset and they capture the underlying relationships and patterns in the data. The neural network can then use these learned weights to make predictions, classifications or generate new data.

Deep learning models can handle large and complex datasets, and can automatically extract features from the data without the need for manual feature engineering. This makes them particularly well-suited for tasks such as image and speech recognition, natural language processing, and other areas where large amounts of data are available. However, they are not as good at explicating how they arrived at a decision, which can be a disadvantage.

In summary, deep learning techniques handle knowledge representation by learning the underlying patterns and relationships in the data through the use of neural networks, which can then be used for prediction, classification, and generation tasks. In GOFAI, knowledge is held by the programmer and explicitly coded into rules, deep learning methods instead use data to guess the best rules from the relationships peresent in the dataset.

2.2 A Simple Neural Network is also a Linear Regression

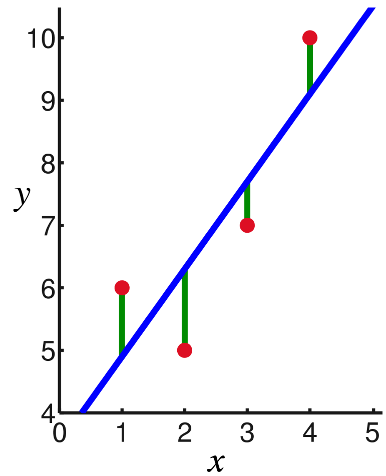

Figure 2.2: A simple linear regression, the red points are the training data, and the blue line is the regression line. If you don't understand this please read `https://en.wikipedia.org/wiki/Regression_analysis`

A neural network can be mathematically equivalent to a regression or a decision tree under certain conditions.

A neural network is a machine learning model composed of layers of interconnected artificial neurons, which are designed to process and analyze data. They can be used for a wide range of tasks, such as image and speech recognition, natural language processing, and prediction.

A regression is a statistical method used to predict a continuous variable based on one or more input features. A linear regression, for example, is a simple neural network with one input layer, one output layer and no hidden layers. In this case, the weights of the network are the coefficients of the linear equation and the network is equivalent to a linear regression model.

A decision tree is a tree-based model used for classification and prediction tasks. It consists of a series of if-then rules that are used to make decisions based on the input data. A neural network with one input layer, one output layer and one hidden layer[1] is equivalent to a decision tree. The network implements a piecewise linear function which can represent the decision boundaries of a decision tree. So, under certain conditions, a neural network can be mathematically equivalent to a linear regression or a decision tree. These conditions include having one input and one output layers, and having a specific activation function in the case of a decision tree.[5]

1: One with a ReLU activation, Neurons in a neural network can have different activation functions, if you don't know what it is and don't want to read about it on wikipedia, that's OK. `https://en.wikipedia.org/wiki/Activation_function`

[5]: Aytekin (2022), *Neural Networks are Decision Trees*

2.3 Dummy Variables for Dummies (Wonkish)

Chapter Summary: It's all numbers, man. Machine learning techniques require that we turn everything (images, text, sound) into numbers and shove them into the model in the same way we use dummy variables in a simple regression. If you are satisfied with this, please skip this section. If you would like to learn a bit about the details and see some code examples, please keep reading. This section is necessarily technical, but should be approachable for anyone who has taken a college statistics class. [2]

2: Thanks to Paul Krugman for popularizing (to me at least) the *wonkish* classifier, I just mean it's a little nerdy. But it's important.

Dummy variables are used in regression analysis to include categorical variables in a model. Categorical variables are variables that take on a finite number of distinct values, such as "red", "green", "blue" or "yes", "no". Since these variables cannot be directly included in a regression model, as they are not numerical, they need to be transformed into numerical variables.

The process of creating dummy variables is also known as one-hot encoding. It involves creating a new binary variable for each category of the original variable. For example, if you have a categorical variable "color" with three categories: "red", "green", "blue", you would create three binary variables: "color_red", "color_green", "color_blue". Each binary variable would take a value of 1 if the original variable is equal to the category, and 0 otherwise.

When using dummy variables in a regression, it is important to remember to include only n-1 binary variables, where n is the number of categories in the original variable. This is because including all n binary variables would result in perfect multicollinearity, which is when two or more independent variables are perfectly correlated. One of the binary variables can be dropped to avoid this problem.

Dummy variables are used in regression analysis to include categorical variables in a model. The process of creating dummy variables involves creating a new binary variable for each category of the original variable and one-hot encoding it. It is important to remember to include only n-1 binary variables, to avoid perfect multicollinearity.

Listing 2.1: Mapping text to numbers.

```
'movies': 99,
'after': 100,
'think': 101,
'characters': 102,
'watch': 103,
'two': 104,
'films': 105,
'character': 106,
'seen': 107,
'many': 108,
'being': 109
```

The creation of dummy variables in a regression is analogous to preprocessing image, text and other data for a neural network for deep learning. This preprocessing is important as it ensures that the data is in a format that can be easily understood and processed by the network. The preprocessing steps for numbers, text, and images are slightly different.

For numbers:

- Normalization: It is common to normalize the input data by scaling it to have a mean of 0 and a standard deviation of 1. This helps to ensure that all input features have similar scales and prevents any one feature from dominating the network's computations.
- Imputation: Handling missing data is important, as it can negatively impact the model's performance. Common imputation techniques include replacing missing values with the mean, median, or mode of the feature.

For text:

- Tokenization: Text data must first be converted into a numerical format that can be understood by the network. This is typically done by tokenizing the text into individual words or n-grams and then encoding them as integers or real-valued vectors. A one-hot encoding exactly like the dummy variable method used in regression is also frequently used. [3] [4]
- Stop-words removal: The most common words in any language like "a", "an", "the", etc. that do not contain much meaning are called stop-words, they are often removed to reduce the dimensionality of the data.

3: Sometimes text is just mapped to a number! Shocking, but it works. See how it is taught in the TensorFlow tutorials https://www.tensorflow.org/text/guide/word_embeddings

4: GPT-3 uses byte-level Byte Pair Encoding (BPE) tokenization and has a vocabulary size of 50,257.

- Stemming/Lemmatization: Words that have the same meaning can be stemmed or lemmatized to reduce the vocabulary size and increase the chances of generalization.
- Vocabulary Size: Each model must choose a vocabulary size or the maximum number of tokens that it will analyze. This may cause misspellings, slang or typos to be discarded in analysis.

For images:

- Converting to RGB or Greyscale: Each image is analyzed by its pixel color value, every point on an image will either have 3 color values (red, green, blue) or one single value (on a white/black scale) if the image is analyzed in greyscale.
- Convolutions[5]: Pixel values are analyzed in groups that are defined by the model, since individual pixel values are only colors (or greyness) they must be combined together by the model to detect patterns like faces and stop signs. The method of convolution is defined by the model itself.
- Resizing: neural network can only accept images of a fixed size, so resizing the image to match the network's requirements is important.
- Normalization: It is common to normalize the pixel values to be in the range of 0-1 or -1 to 1. This will help the model converge faster.
- Data Augmentation: To increase the amount of data and prevent overfitting, common data augmentation techniques such as flipping, rotation, and cropping can be applied to the images.

5: If you would prefer a visualization of a neural network check out this excellent video by Dennis Dmitriev `https://www.youtube.com/watch?v=3JQ3hYko51Y`.

In summary, preprocessing is an important step in training a neural network, as it ensures that the data is in a format that can be easily understood and processed by the network. The preprocessing steps for numbers, text, and images involve normalization, imputation, tokenization, stop-words removal, stemming/lemmatization, resizing and data augmentation.

2.4 Try That Again With 33,640,010 Parameters

In the first section of this chapter we introduced the idea that a simple neural network is mathematically equivalent to a regression. This is true and a useful way to think about neural networks and deep learning. [6]

6:

$$y = Wx + b \tag{2.1}$$

There's your simple formula for a single-cell neural network and regression, in practice we'll change W, x and b to matrices and introduce nonlinear activation function f so,

$$y = f(Wx + b) \tag{2.2}$$

if you please. If equations scare you, don't worry about it for now.

In practice, neural networks frequently trained with millions of parameters. Here is an example of the trainable parameters of a simple image classifier. This example is a dense neural network to classify handwritten digits, and is not yet as sophisticated as the one used by Yann LeCun and company in the 1990s

In these next few examples, don't worry if you don't understand the layer types or know what batch nomalization means. The point I would like to make is that neural networks are often created with millions of trainable parameters, once you agree with me regarding this point we will discuss the implications of this in more depth.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 64)                50240
_________________________________________________________________
batch_normalization (BatchNo (None, 64)                256
_________________________________________________________________
dense_1 (Dense)              (None, 64)                4160
_________________________________________________________________
batch_normalization_1 (Batch (None, 64)                256
_________________________________________________________________
dense_2 (Dense)              (None, 64)                4160
_________________________________________________________________
batch_normalization_2 (Batch (None, 64)                256
_________________________________________________________________
dropout (Dropout)            (None, 64)                0
_________________________________________________________________
dense_3 (Dense)              (None, 10)                650
=================================================================
Total params: 59,978
Trainable params: 59,594
Non-trainable params: 384
_________________________________________________________________
```

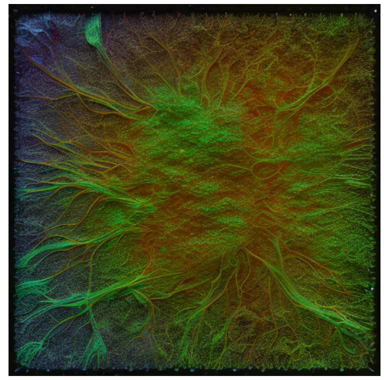

Figure 2.3: "mdjrny-v4 layers and layers of neurons looking like a gooey lasagna, but the data has green blood 8k" made with Mann-E

59,594 parameters! That's not too bad for a simple neural network, but in practice the networks often get even bigger. Here is a *real* example that will make your brain a little hot.

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 28, 28, 64)        640
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 28, 28, 64)        36928
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 14, 14, 64)        0
_________________________________________________________________
batch_normalization (BatchNo (None, 14, 14, 64)        256
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 14, 14, 128)       73856
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 14, 14, 128)       147584
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 7, 7, 128)         0
_________________________________________________________________
batch_normalization_1 (Batch (None, 7, 7, 128)         512
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 7, 7, 256)         295168
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 7, 7, 256)         590080
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 7, 7, 256)         590080
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 3, 3, 256)         0
_________________________________________________________________
batch_normalization_2 (Batch (None, 3, 3, 256)         1024
```

```
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 3, 3, 512)         1180160
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 3, 3, 512)         2359808
_________________________________________________________________
conv2d_9 (Conv2D)            (None, 3, 3, 512)         2359808
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 2, 2, 512)         0
_________________________________________________________________
batch_normalization_3 (Batch (None, 2, 2, 512)         2048
_________________________________________________________________
conv2d_10 (Conv2D)           (None, 2, 2, 512)         2359808
_________________________________________________________________
conv2d_11 (Conv2D)           (None, 2, 2, 512)         2359808
_________________________________________________________________
conv2d_12 (Conv2D)           (None, 2, 2, 512)         2359808
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 1, 1, 512)         0
_________________________________________________________________
batch_normalization_4 (Batch (None, 1, 1, 512)         2048
_________________________________________________________________
flatten (Flatten)            (None, 512)               0
_________________________________________________________________
dense (Dense)                (None, 4096)              2101248
_________________________________________________________________
dropout (Dropout)            (None, 4096)              0
_________________________________________________________________
dense_1 (Dense)              (None, 4096)              16781312
_________________________________________________________________
dropout_1 (Dropout)          (None, 4096)              0
_________________________________________________________________
dense_2 (Dense)              (None, 10)                40970
=================================================================
Total params: 33,642,954
Trainable params: 33,640,010
Non-trainable params: 2,944
_________________________________________________________________
```

When a neural network has millions of trainable parameters and deep layers of neurons[7], it can be difficult to explain to a human what each of those parameters represents or how they contribute to the overall function of the network. This is because the interactions between the different layers and neurons can be complex and non-linear, making it challenging to understand the specific role of each parameter. The parameters model complex interactions between the inputs, making it difficult to understand their specific function. Furthermore, in deep neural networks, the high number of layers can lead to high level of abstraction, meaning that the individual neurons and their weights have little interpretability.

7: In case you are wondering, the human brain has about 86 billion neurons that do the "thinking". Physical neurons are more complicated than the neurons used in deep learning and have supporting structures like astrocytes that do some computation as well. https://en.wikipedia.org/wiki/Astrocyte

When a neural network has millions of interacting parameters, it can lead to mathematical chaos [8], which is a phenomenon where small changes in the initial conditions of the network can lead to vastly different outputs. This is because the interactions between the large number of parameters can create non-linear relationships that are sensitive to small changes. This can

8: Mathematical chaos is a real thing, it refers to the behavior of certain dynamic systems that are highly sensitive to initial conditions. This sensitivity leads to seemingly random and unpredictable behavior, even though the underlying equations governing the system are deterministic (meaning they are transparent and known to us). https://en.wikipedia.org/wiki/Chaos_theory make it difficult to predict the behavior of the network, as small changes in the input or the parameters can lead to unexpected and seemingly random outputs.[9]

9: Chaotic outputs in a large deterministic system are OK in many domains, and there are controls that can be put in place to keep AI "safe" but deep learning by itself will not produce those controls.

2.5 Multicolinearity and the End of Science

Data science is a horrible term because it implies that data scientists are scientists and that the work they do is scientific when in reality data scientists are not scientists and the work they do is not scientific. Data scientists use mathematics, statistics, and computer science to analyze data, but it is not scientific in the traditional sense. Data scientists do not use the scientific method and do not conduct experiments or develop theories. Data science is more akin to engineering or data analytics than actual scientific research.

The main goal of a scientist is to gain knowledge and understanding of the natural world through research, experimentation, and data analysis. Scientists make models of the world to test and explain. So-called data scientists make models too but a model with layers of interacting parameters, trained under chaotic conditions is almost impossible to explain. Multicolinearity[10] is just one issue, but deep learning techniques are practically incompatible with the scientific method. With deep learning methods it can be difficult to determine which variable had the most impact on the outcome of the model. Coefficients of the model to be unstable and unreliable, which can make it difficult to explain the model in a meaningful way.

10: "Multicolinearity is when two or more variables in a study are closely related to each other. Think of it like dating: imagine you are trying to figure out what makes someone a good partner. You might look at things like how much they make, how attractive they are, and how kind they are. But, these things are all related to each other. For example, attractive people might make more money, and kind people might be more attractive. So, it's hard to know which one is really important for being a good partner, because they are all related. That's like multicolinearity in a study." By ChatGPT given the prompt "explain multicolinearity to a teenager using a dating example"

Explaining the inner-workings of a neural network with millions of interacting parameters is difficult for many reasons. Most of the complexity is due to the number of parameters, which can make it difficult to understand how the model works and why it makes certain decisions. The interactions between the parameters are often non-linear and can be difficult to understand, as the model can be sensitive to small changes in the parameters, which is the definition of mathematical chaos. This can make it difficult to explain why the model is making certain decisions, as the interactions between the parameters are not always clear.

2.6 Let's Test Some Random Inputs! Feature Importance and Explainability

With a huge complex system of neurons that combine inputs in novel ways, it becomes vary hard to understand which inputs are the most important for the system as a whole. To better understand deep learning models, data scientists use randomized or averaged inputs to a model to test the feature importance of a deep learning neural network. This type of testing is done to determine which features are most influential in the overall output of the network. This is done by randomly or averaging the input values for each feature and then running the model to see how the output is affected. For example, if a neural network is used to detect objects in an image, the data scientist may randomize the color of the objects to see how the model's accuracy is affected. If the accuracy drops significantly, they can infer that color is an important feature in the network.

Randomized or averaged inputs to a model can also be used to determine if a particular feature is necessary for the network to function correctly. For example, if the output of the network is not as accurate when a particular feature is randomized or averaged, then the data scientist can infer that this feature is important for the network's performance. By using this method, data scientists can gain insights into which features are important and which can be removed from the model to improve performance.

2.7 The Universal Machine Learning Workflow

The Universal Machine Learning Workflow[6] is an important chapter in a technical guide for data scientists by the author of the most popular machine learning framework in the world[11] that outlines the *Universal* workflow for machine learning projects. I think this chapter should be understood by everyone using, investing in and creating machine learning models.

[6]: Chollet (2022), *Deep learning with python, second edition*

11: It's called TensorFlow and was one of the most popular and "Most Loved" Technologies of 2022 https://survey.stackoverflow.co/2022/#most-popular-technologies-misc-tech

Before Chollet gets into the details of model building, he chooses to begin with a note on ethics:

"You may sometimes be offered ethically dubious projects, such as "building an AI that rates the trustworthiness of someone from a picture of their face." First of all, the validity of the project is in doubt: it isn't clear why trustworthiness would be reflected on someone's face. Second, such a task opens the door to all kinds of ethical problems. Collecting a dataset for this task would amount to recording the biases and prejudices of the people who label the pictures. The models you would train on such data would merely encode these same biases into a black-box algorithm that would give them a thin veneer of legitimacy. In a largely tech-illiterate society like ours, "the AI algorithm said this person cannot be trusted" strangely appears to carry more weight and objectivity than "John Smith said this person cannot be trusted," despite the former being a learned approximation of the latter. Your model would be laundering and operationalizing at scale the worst aspects of human judgement, with negative effects on the lives of real people.

Technology is never neutral. If your work has any impact on the world, this impact has a moral direction: technical choices are also ethical choices. Always be deliberate about the values you want your work to support."[6]

Figure 2.4: "the face of a trustworthy person" made with Stable Diffusion 2.1. (Hey, it's a white lady!)

Chollet uses the outline below to explain the *Universal* workflow. I'll summarize the workflow and my own notes for a nontechnical audience here:

1. Define the Task
   a) Collect a Dataset [12]
   b) Understand Your Data
   c) Choose a Measure of Success [13]
2. Develop a Model

12: This is often the hardest part. Data is often labeled by hand or stolen from another domain. See https://en.wikipedia.org/wiki/Transfer_learning

13: Accuracy isn't everything! We might prefer a less accurate model that avoids false-positives in some scenarios.

a) Prepare the Data [14]
b) Choose an Evaluation Protocol
c) Beat a Baseline [15]
d) Develop a model that overfits
e) Regularize and Tune Your Model

3. Deploy the Model

a) Explain Your Work to Your Stakeholders and Set Expectations [16]
b) Ship an Inference Model
c) Monitor Your Model in the Wild
d) Maintain Your Model

14: This is discussed in section 2.3 above, everything gets turned into numbers.

15: Does your model predict better than a totally random model and/or coin-flip?

16: This is almost never done. And as Chollet explains this is both difficult and requires mutual understanding and domain knowledge from the part of "the business" and the Machine Learning Engineer.

Of all of the steps in the workflow, "Defining The Task" and "Explaining The Work To Shareholders and Setting Expectations" are where the most miscommunication occurs.

In defining the task, machine learning engineers are often given impossible problems to solve, and because they want to keep paying their mortgage they solve another related problem instead and allow stakeholders to jump to their own conclusions. By clearly understanding what a model is predicting and how the data is collected some of this misunderstanding can be avoided.

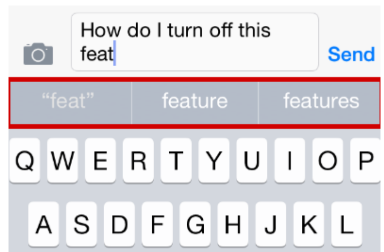

Figure 2.5: Fundamentally LLMs are using the same techniques as a predictive keyboard on your phone. This is why Yann Lecun says even though they are impressive some like ChatGPT are not particularly innovative. https://www.youtube.com/watch?v=ULbpPHjiSBg

I will discuss Large Language Models later, but they all do the same thing. They predict the next words in a sentence, just like the keyboard on your iPhone. They come up with amazing text but by fundamentally understanding what they are predicting you gain insight into their limitations. They are also all trained on the text publicly available on the internet, this includes scientific sources, but also fan-fiction and anime, so the idea that a model trained on this data could be relied on to predict anything truthful is preposterous.

Many models work like magic and many users assume models are predicting the future, when they are really correlating based on their past data. Even simple models of creditworthiness might have as similar problem. While building a creditworthiness model for a bank, due to lack of data a machine learning engineer might (on purpose or inadvertently) create a model that predicts whether someone is a smoker instead of whether they are creditworthy. Because smoking and poverty are correlated, maybe the model "works", but it doesn't do what stakeholders think it does.

Setting expectations is another hellscape of misaligned incentives.[17] Good machine learning engineers and data scientists are supposed to educate and advise stakeholders about the limitations of their own models. Read Chollet's advice on this topic below:

17: The CEO of OpenAI wrote this and still raised ten billion dollars: *"ChatGPT is incredibly limited, but good enough at some things to create a misleading impression of greatness. it's a mistake to be relying on it for anything important right now. it's a preview of progress; we have lots of work to do on robustness and truthfulness."* https://twitter.com/sama/status/1601731295792414720

The expectations of non-specialists towards AI systems are often unrealistic. For example, they might expect that the system "understands" its task and is capable of exercising human-like common sense in the context of the task. To address this, you should consider showing some examples of the failure modes of your model (for instance, show what incorrectly classified samples look like, especially those for which the misclassification seems surprising).

They might also expect human-level performance, especially for processes that were previously handled by people. Most machine learning models, because they are (imperfectly) trained to approximate human-generated labels, do not nearly get there. You should clearly convey model performance expectations. Avoid using abstract statements like "The model has 98 percent accuracy" (which most people mentally round up to 100 percent), and prefer talking, for instance, about false negative rates and false positive rates. You could say, "With these settings, the fraud detection model would have a 5 percent false negative rate and a 2.5 percent false positive rate. Every day, an average of 200 valid transactions would be flagged as fraudulent and sent for manual review, and an average of 14 fraudulent transactions would be missed. An average of 266 fraudulent transactions would be correctly caught." Clearly relate the model's performance metrics to business goals.

You should also make sure to discuss with stakeholders the choice of key launch parameters- for instance, the probability threshold at which a transaction should be flagged (different thresholds will produce different false negative and false positive rates). Such decisions involve trade-offs that can only be handled with a deep understanding of the business context.[6]

Figure 2.6: "mdjrny-v4 a handsome businessperson explaining the business context to a scientist wearing a white coat over coffee 8k" made with Mann-E

One does not need to be a psychologist to understand that these conversations almost never happen. The workflow of machine learning is universal, we are all doing fancy regressions and we all deal with the same wild expectations, bad data and misalignment with business.

2.8 A Machine Learning Engineer is a Data Janitor

Since the machine learning workflow is *universal*, it is fairly straightforward to automate. There are a plethora of offerings from companies big and small who offer AutoML tools that anyone with basic Excel skills can use[18]. Since modern AI is "all a regression" these tools do the regression for you, users just need to feed them the data and the AutoML tool finds out the relationship. On the surface it seems like the job of the machine learning engineer has become obsolete before it even became a proper discipline.

18: See this awesome-AutoML github project for an up-to-date list of commercial AutoML tools `https://github.com/windmaple/awesome-AutoML#commercial-products`

Despite the existence of AutoML tools, machine learning engineers (and data scientists) do some actual work. A 2016[19] Crowd Flower survey of 16,000 data scientists showed that on average we spend our time doing the following:

19: I know this study is old, but I have found that the findings still hold when talking to data scientists and/or machine learning engineers that I train and employ.

- **60 percent of a machine learning engineer's time is spent cleaning and organizing data**, we are data janitors! If you had great data ready in an Excel file you would have been able to fit your own model without us. We need to organize the mess of data that exists in the world so it is in a nice format to be fit in a model. For example it is well known that almost all large language models use the same common crawl dataset (`https://commoncrawl.org/`), but how it is organized and weighted can change the quality of the outputs dramatically. The organization of the same dataset can lead to the same dataset either producing a nice predictive keyboard or something approaching ChatGPT.

- **19 percent of a machine learning engineer's time is spent collecting data**, we often start making models without a huge dataset, so data needs to be collected in order to make the first model, and after that first model (to match people for our new dating app, let's say) is deployed we rely on users to provide us with data for future models. Collecting new data is a creative endeavour sometimes, and often involves human effort. Amazon's Mechanical Turk[20] service has been leveraged for this purpose for years, and OpenAI has successfully deployed outsourced talent to solve some of the trickiest problems facing large language models[21].
- **9 percent of a machine learning engineer's time is spent fitting models**, we need to fit models, there is some art to choosing the right architecture and modeling techniques. An AutoML tool can do this reasonably well too, but considering some fancy training servers cost $28 per hour or more[22], sometimes it is useful to have an expert guess the best model architecture and save time in training, even if that expert is getting paid $249,000 yer year[23].
- **4 percent of a machine learning engineer's time is spent refining algorithms** many machine learning engineers spend almost no time doing this. Other researchers (corporate-academic types who write scientific papers) spend a lot of time doing this, it averages out to four percent, but doesn't represent the average day of a machine learning engineer.

20: "Getting the right training data was another challenge. ImageNet was a collection of one hundred thousand labeled images that required significant human effort to generate, mostly by grad students and Amazon Mechanical Turk workers." https://arstechnica.com

21: Read more about how OpenAI used Kenyan workers to help label toxic content https://time.com/6247678/openai-chatgpt-kenya-workers/.

22: See the latest pricing from AWS here https://aws.amazon.com/sagemaker/pricing/

23: See the latest machine learning engineer salaries at levels.fyi https://www.levels.fyi/t/software-engineer/focus/ml-ai

Overall, the main job of most machine learning engineers is to clean up and collect a ton of data, and then feed that data into the same machine learning model that everyone else is using, and then rinse and repeat.[24]

24: Here is a discussion on reddit asking "When was the last time you wrote a custom neural net?" and supports my understanding of the world https://www.reddit.com/r/MachineLearning/comments/yto34q/d_when_was_the_last_time_you_wrote_a_custom/

2.9 Key Takeaways

- **Deep learning models are fundamentally large unscientific regressions** they are trained to create a function that maps input data to output data.
- **Deep learning models are chaotic systems containing millions of interacting parameters** they are not designed to be explained or created in a way that their weights can be used for scientific analysis. They find reasonable answers and don't care how they get there. Multicolinearity (understanding the relationship of an input and output) and feature importance (understanding which inputs are most important) are only understandable with a high level of statistical error.
- **Small changes in inputs of a deep learning model may dramatically change the outputs** deep learning models are complex deterministic systems that can exhibit chaotic behavior. Their inner workings are functionally unknowable and practically impossible to test.
- **Machine learning engineers spend most of their time collecting and organizing data**, because deep learning models often share

common architecture, getting good data to train with is the best thing that an engineer can do to train good models. In practice only a handful of corporate-academic types are experimenting with new and exciting architectures.

- **Deep learning models can have impressive and useful outputs, but the creators of models should be encouraged to highlight their failures and limitations.** Machine learning engineers might be more keen to highlight failures and limitations if they are encouraged to do so by their users, managers and investors.

3 Creativity and Decision Making with Deep Learning Models

"AI Policy

I expect you to use AI (ChatGPT and image generation tools, at a minimum), in this class. In fact, some assignments will require it. Learning to use AI is an emerging skill, and I provide tutorials in Canvas about how to use them. I am happy to meet and help with these tools during office hours or after class

Beware of the limits of ChatGPT:

- *If you provide minimum effort prompts, you will get low quality results. You will need to refine you prompts to get good outcomes. This will take work.*
- *Don't trust anything it says. If it gives you a number or a fact, assume it is wrong unless you either know the answer or can check in with another source. You will be responsible for any errors or omissions provided by the tool. It works best for topics you understand.*
- *AI is a tool, but one that you need to acknowledge using. Please include a paragraph at the end of any assignment that uses AI explaining what you used the AI for and what prompts you used to get the results. Failure to do so is in violation of academic honesty policies*
- *Be thoughtful about when this tool is useful. Don't use it if it isn't appropriate for the case or circumstance."*

Dr. Ethan Mollick, 2023 - Syllabus for class at The Wharton School at the University of Pennsylvania

3.1 Theories of Creativity

Machine learning models use data (from the past) to discover rules and make classifications. Because of the way they are constructed these classifications, suggestions, artworks are by definition derivative or "having parts that originate from another source". I won't get into a philosophical discussion on what the nature of creativity is, but it's worth considering how using deep learning models biases us towards the past, but also could give us insights from other domains.

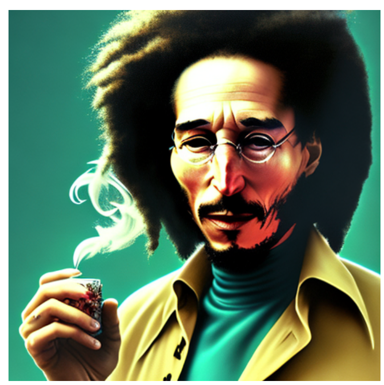

Figure 3.1: "mdjrny-v4 Steve Jobs Smoking weed with Bob Marley 8k" made with Mann-E. It's 100% derivative, but it's art too (I guess).

It can be argued that creativity is simply the combination of existing works. This is because many new ideas and innovations are often inspired by and built upon existing concepts. For example, a new form of music may be created by combining elements from different genres. Similarly, a new technology may be created by combining and improving upon existing technologies.

It can also be argued that creativity involves much more than just combining existing works. Creativity is not just about recombining existing ideas, but also about coming up with completely new and original concepts. This requires a unique perspective and a deep understanding of the subject matter, as well as the ability to think outside the box.

Deeep learning models of speech, when heavily used, may slow down the evolution of language. AI art models may slow down "progress" in art, whatever that means. Deep learning models of disease trained on data from 1980 may be irrelevant to today's diseases. That said these same models may give us interesting insights in new domains, models trained on beautiful paintings may be put to use in a new domain (like designing beautiful interiors) and that model could give new insight to interior designers, models of the interaction of ants could be put to use in designing cities and so on and so forth. AI cuts many ways, it makes us faster but makes us more reliant on the past, models can be used across domains but should be used intentionally and transparently when possible. Each use opens up a new world of possibilities for users, and sometimes a new headache for intellectual property lawyers. We'll discuss all of these topics in this chapter.

Figure 3.2: "Batman dressed as a cowboy" made with Mann-E. It's 100% derivative as well, but better than any Google image searches I did. It looks amazing! I also might get sued if I use it even though the creators of Mann-E tell me I have the rights to it.

3.2 Creative Uses of Power Tools

Power tools, even simple ones like a drill are complex machines that take human input and transform it using static rules. The pressure the user of a drill puts on the bit and the speed at which they pull the trigger all deterministically affect the output. Just because a tool is a deterministic machine doesn't mean it is unable to produce creative works.

What is happening as we use generative tools like ChatGPT or image-to-text models is that the "creative act" has been relocated. The creative act is now the prompt you give the tool, the users input. And someone can still be good at using AI, just like someone can be good at using any piece of software.

Modern AI is a complex system of algorithms, data, and analytics that can be used to solve complex problems. AI systems can learn from data, identify patterns, and make predictions about the future. AI systems are typically used to automate and assist human decision making. AI systems are programmed with specific objectives and goals, and the user input decides the output. For example, an AI system could be programmed to solve a mathematical problem and the user input would determine the parameters of the problem and the output would be the solution.

Figure 3.3: "mdjrny-v4 a mikita drill being used in an artist's studio to make a colorful artwork 8k" made with Mann-E

A power drill is also a complex deterministic system. The user input is limited to the type of drill bit, the speed of the drill, and the direction of the drill. The output is determined by these inputs, as the drill will only drill in the direction and speed determined by the user. The user also has to choose the correct drill bit to ensure the drill can do the job correctly and safely.

Both modern deep learning models and a power drill are complex but ultimately deterministic systems. The user input, however limited it may be, ultimately controls the output of the system. Both systems require a user to understand how to use them and to make the correct input to get the desired output.

3.3 Garbage In, Garbage Out

The "Power Tool" of AI is a deterministic, static and unchanging system and the rules of that system are determined by the data that the model is shown. If a model is trained on bad data, it will produce poor results. End of book...

... maybe not. It's worth thinking about this for a moment. It is often said that modern AI can "generalize and make informed inferences given new data". If deep learning models are really just a big regression, these models will always come up with an answer, but if the world changes, these models will still be projecting complicated averages of their past data into the future.

So, let's separate AI's decision making into two extremes; *Creative Decision Making* and *Critical Decision Making*. The stakes are very low in a world of *Creative Decision Making* and who cares if the AI is all a regression, and it just mushes together the limited data that it's seen. In a creative context you can also ask an AI interesting questions, so long as you don't solely rely on it's output without checking the facts first[1]. Even if "Garbage In, Garbage Out" holds, garbage can still be helpful for a creative process.

1: See the syllabus note from Dr. Mollick at the beginning of this chapter.

Note that this book is called "Full-Self Driving, Skynet..", a self-driving car and a nuclear-bomb-equipped all-seeing AI are clearly not engaging in any *Creative Decision Making*.

Figure 3.4: Marcel Duchamp's "Fountain". A urinal that blew peoples minds https://www.tate.org.uk/art/artworks/duchamp-fountain-t07573.

3.4 Garbage In, New Perspective Out?

For creative tasks, it generally doesn't matter that a deep learning model is unscientific or trained on a lopsided dataset. An informed user of AI knows this and can account for that in their decision making, especially when engaging in creative decision making. The situation becomes problematic once we allow deep learning models to engage in critical decision making by themselves.

Generative models will come up with amazing (but derivative) works of art, But, they will never "change the game". A model of sculpture trained on past sculptures in 1917 will never come up with Marcel Duchamp's "Fountain". There is a term that machine learning engineers use for this, when the rules of the game are changed, it's called "Concept Drift".

3.5 Concept Drift and the End of Usefulness

Concept drift refers to the phenomenon where the distribution of data changes over time, causing the performance of deep learning models built on historical data to degrade. The model becomes less useful because it is trained to make predictions based on the relationship between the inputs and outputs in the data it was trained on, and if this relationship changes, the model may start making incorrect predictions. This is particularly problematic in real-world applications, where the data is constantly evolving and the relationship between inputs and outputs is subject to change. To mitigate the effects of concept drift, it is often necessary to continually retrain deep learning models on updated data.

Figure 3.5: "Plastic surgery gone wrong" made with Stable Diffusion. Imagine a model that classifies images as "human face" or "not human face", and imagine that model was trained on images of human faces before 1900, maybe you would not be surprised if you gave it a picture of a human face that had a lot of cosmetic surgery done to it, and that model might say "this is not a human face", the idea of what a human face is has changed over time this is called "concept drift".

The frequency with which a deep learning model needs to be retrained to mitigate the effects of concept drift depends on several factors, including the rate at which the data distribution changes, the size and complexity of the model, and the availability of computational resources.

For some applications with relatively stable data distributions, retraining the model once every few months or even once per year may be sufficient. However, in other applications where the data is changing rapidly, it may be necessary to retrain the model more frequently, such as once per week or even once per day.

Ultimately, the frequency of retraining will depend on the specific requirements of the application, and the trade-off between the cost of retraining and the potential cost of incorrect predictions. In general, it's recommended to monitor the performance of the model over time and to retrain it as needed to ensure that it remains accurate and relevant.

If a deployed deep learning model is not monitored, there are several risks that can emerge:

- Accuracy degradation: As the data distribution changes over time, the model may become less accurate, leading to incorrect predictions. This can result in financial losses, reduced customer satisfaction, or even harm to individuals.
- Bias amplification: deep learning models can be biased, and if this bias is not monitored and addressed, it can be amplified over time as the model continues to make incorrect predictions. This can result in discriminatory outcomes, such as unequal treatment of individuals based on protected characteristics such as race, gender, or age.
- Legal liability: In some cases, incorrect predictions made by deep learning models can result in legal liability, particularly if the model is being used to make decisions that have significant consequences, such as in the criminal justice system or in medical diagnosis.
- Reputational damage: If a deep learning model is making incorrect predictions, it can damage the reputation of the organization deploying the model, potentially leading to a loss of customers or investors.

It is important to monitor deep learning models once they are deployed, and to take action to address any issues that arise, such as retraining the model or adjusting its parameters, in order to mitigate these risks and ensure that the model continues to perform well over time.

3.6 The Impossibility of Fairness

Achieving fairness in deep learning models can be challenging, and to some extent it may be impossible to completely eliminate all forms of bias. This is because deep learning models are trained on historical data, which may contain biases and disparities that are perpetuated in the model's predictions. One can attempt to "fix the training set" but in practice models will continue to bias their prediction to past data, or their creators careful curation of the past.[7]

[7]: Christian (2020), *The Alignment Problem: Machine Learning and Human Values*

However, it is possible to reduce the impact of bias in deep learning models through careful design and monitoring of the model's performance. This may include techniques such as fairness constraints, algorithmic transparency, and regular auditing of the model's predictions to identify and address any issues of bias.

It's important to note that fairness is a complex and multi-faceted concept, and different definitions of fairness may be appropriate for different applications. For example, some definitions of fairness may prioritize equal treatment of all individuals, while others may prioritize proportional representation or equal opportunities.

Ultimately, the extent to which fairness is achievable in deep learning models will depend on the specific requirements of the application and the level of effort that is put into designing and monitoring the model to ensure that it is making fair and unbiased predictions.

3.7 Transfer Learning Everywhere

Transfer learning is a machine learning technique that involves transferring knowledge from one model trained on a task to another model trained on a related task. The idea behind transfer learning is that a model that has been trained on one task can be fine-tuned for another task, reducing the amount of labeled data required to train the new model.

For example, imagine that you have a deep learning model that has been trained to recognize objects in natural images (cats, dogs, balls, cars). You can use the knowledge learned by this model as a starting point to train a new model that recognizes objects in medical images, such as X-rays or MRI scans. The new model can start with the weights of the pre-trained model and fine-tune them on the new task, using a much smaller amount of labeled data than would be required to train the model from scratch.

Transfer learning can be useful in many different applications, particularly when labeled data is scarce or expensive to obtain. By leveraging knowledge from a pre-trained model, transfer learning can help to improve the performance of new models, reduce the amount of data required for training, and accelerate the development of new machine learning applications.

I asked ChatGPT to "tell me a story of transfer learning going horribly wrong in the context of online dating, with a model trained in the Middle East being deployed by an American dating app." and here's what it said:

"Once upon a time, an American online dating company decided to use transfer learning to improve their matchmaking algorithms. They trained a state-of-the-art machine learning model on a large dataset of profiles and dating preferences from the Middle East. The model was able to suggest matches with remarkable accuracy and the company was confident that it would perform just as well in the American market.

However, things went terribly wrong when they deployed the model on their American dating app. The model was not properly trained to consider the cultural and social differences between the Middle East and America. As a result, it suggested matches that were inappropriate and offensive, causing widespread backlash and criticism from users.

For example, the model suggested matches between people of different religious backgrounds, which was not acceptable to many users. It also suggested matches between people with significant age gaps, which was not socially acceptable in the American market.

The company was forced to remove the model and start from scratch, causing significant harm to their reputation and financial losses. They learned the hard way that transfer learning can only be effective if the model is properly trained on relevant and culturally appropriate data.

This story serves as a cautionary tale about the importance of considering cultural differences and ethical implications when deploying machine learning models in different regions and markets."

Not bad I think... picking good training data and doing good "data janitoring" is the main job of a machine learning engineer[2].

2: The better your machine learning engineer understands your business problem, the better job they'll do at creating the dataset.

If the past is not like the future, you are also doing transfer learning. Most models steal data from other sources so are doing transfer learning too. This is fine, but do we know that we are doing this? A model that is deployed in a domain experiencing concept drift and continues to make predictions without being retrained can be considered to be doing a harmful form of transfer learning. This is because the model has been trained on a different distribution of data (the past), and is being applied to a new domain with a different distribution (the present and future).

In traditional transfer learning, the goal is to transfer knowledge from one domain to another related domain, where the data distributions are similar enough to enable the model to make accurate predictions. However, in the case of concept drift, the data distributions are changing over time, and the model is becoming less accurate as a result. [3]

3: Think about your life and if you "transferred" your model of thinking from when you were 12 years old, to "today" when you are 35 years old, that is a bad model to be operating on! That model needs to be updated. We'll explore many examples of where this is and is not a problem in the next chapter."

By continuing to make predictions without being retrained, the model is essentially "transferring" its knowledge from a historical data distribution to a new, changed data distribution, which may not be a valid assumption. This can result in incorrect predictions and other negative outcomes, such as harm to individuals or organizations.

3.8 Industrial-Scale Plagiarism

Aside from regurgitating the past and predictions from other domains, deep learning models can also enable plagiarism on an industrial scale. Early text generation models could be made to write entire sections of *Harry Potter* when fed the opening lines of a chapter. Even as models grow large and more sophisticated, users, researchers and lawyers are still able to "extract the training data" from large models[8], causing headaches for their creators and adding to the work of intellectual property lawyers.

[8]: Carlini et al. (2023), *Extracting Training Data from Diffusion Models*

To avoid this creators of deep learning models have the following tools at their disposal:

- Best practices in machine learning: data preprocessing, augmentation, reuglarization, model architecture choices, early stopping and validation are all things that we are taught to do to prevent overfitting.
- Contractual agreements: Microsoft (owner of github and cocreator of the Github Copilot code generation model) has a special contract for anyone submitting code to github that essentially says that "we are allowed to use this code to train our models, and if our models regenerate your copyrighted code, you can kick rocks."
- Release the Kraken: StabilityAI's Stable Diffusion model was released as open source software, they still managed to get sued [9], but they basically said, "this model was trained on all copyrighted and copylefted images on the web, sometimes it'll generate stuff that violates copyright law, but we are in Germany and will give this thing away for free to the public, and see how Getty Images, photographers and graphic designers of the world handle it... good luck!".

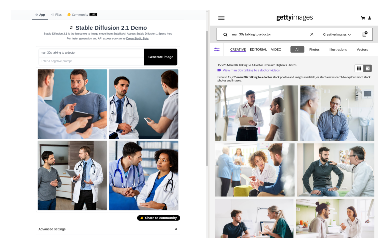

Figure 3.6: "man 30s talking to a doctor" Getty Images and Stable Diffusion comparison. Is Stable Diffusion creating something new here or spitting out it's training data? Is training a model like this fair use? I don't know, have fun lawyers! Note rights to each Getty photo runs around $450, and Stable Diffusion is free to use, and they give users the rights to the model output. The Kraken has been released!

These techniques can help to reduce the risk of having deep learning models reproduce the training data and cause intellectual property disputes, but there is no way to completely eliminate these risks, they are simply side-effects of the method of machine learning that we are doing nowadays. If you are an IP lawyer and need an expert witness, I'm your man brad@bradflaugher.com.

3.9 Working Together is Best

It turns out sometimes the best combination is a reasonably smart (but not too cocky) human that gets to get recommendations from a few deep learning models. [4]

4: Funnily enough at Wharton I recently took a class where Dr. Cade Massey said basically "If you want to bring people along and get them to use your model, let them play with the output a bit, allowing for even the slightest bit (2%) adjustment in the outputs will get users to adopt your model much faster." Thanks Cade!

Garry Kasparov, the legendary chess player we heard about in chapter 1, witnessed a tournament where humans and AI could compete together. The tournament was designed to showcase the strengths of both humans and AI, and how they could work together to achieve incredible results here is what Garry says about the tournament:

"Once again, the chess world offers a useful test case for how this collaboration can play out. In 2005 the online chess playing site Playchess.com hosted what it called a "freestyle" chess tournament in which anyone could compete in teams with other players or computers. What made this competition interesting is that several groups of grandmasters working with computers also participated in this tournament. Predictably, most people expected that one of these grandmasters in combination with a supercomputer would dominate this competition ; but that's not what happened. The tournament was won by a pair of amateur American chess players using three computers. It was their ability to coordinate and coach effectively their computers that defeated the combination of a smart grandmaster and a PC with great computational power."[10]

[10]: Kasparov (2021), *Ai should augment human intelligence, not replace it: Harvard Business Review: March 18, 2021*

The tournament was a huge success, and showed that when humans and AI work together, they can achieve amazing things. The tournament participants learned that AI is not just a tool, but a valuable partner, and that by combining their strengths, they could achieve results that neither could have accomplished on their own. The tournament inspired many people to explore the potential of human-AI collaboration, and showed that by embracing technology, we can create a brighter future for all.

AI is an amazing partner, but we need to think for ourselves too. We can't blindly trust AI, but we can use it to inspire and challenge us.[11]

[11]: Mansharamani (2020), *Think for yourself: Restoring common sense in an age of experts and artificial intelligence*

3.10 Key Takeaways

- **When you are using deep learning models, you are almost always engaging in some kind of transfer learning.** You are assuming the past will be like the future, or that training set reflects reality, which isn't always the case. [5]
- **Models can puke out seemingly creative things, but they are still a product of their training data.** deep learning models are deterministic systems, they slap together data based on rules they guessed from their training data. This process can still be very useful, but deep learning models are not sentient creative creatures from science fiction.
- **Even if you think you "own the rights" to the output of a model, models can sometimes puke out their training data** modelers try their best to prevent this but models can spit out their training data, which may cause headaches for everyone except the intellectual property lawyers.
- **Users should consider the quality and appropriateness of the data the model they are using is trained on before deciding to use a model.** Remember, machine learning engineers spend most of their time cleaning up and gathering data, if you ask questions about how and when the data was collected and cleaned up you can get ahead of potential problems of concept drift and blind transfer learning (engaging in transfer learning without your consent).

5: It's as simple as the common financial disclaimer *Past performance is no guarantee of future results.* Even though it is everywhere, the general public and even big swingers in finance don't seem to get this one. In AI there is an additional disclaimer, *Training data may not reflect the the "real world".*

- **Work with the technology** don't make your model do everything, sometimes it'll come up with stupid answers. Also don't assume you are always right, try and remember the story of the freestyle chess tournament and the grandmasters who got beat by the ameteurs getting reasonable advice from a few chess engines. Working with your model, as if it were a coworker that you don't fully trust, but still think is smart is probably the best way to use deep learning models.

Model Cards and Case Studies 4

Model cards are a way to document the capabilities and limitations of your AI model. They are a useful tool for understanding your model, and for communicating its capabilities and limitations to others. Google (see `https://modelcards.withgoogle.com/face-detection`), Huggingface and even small makers of AI models produce model cards to help educate their users and developers, and as you can imagine no one reads them.

Think of a model card as a passport for your AI model. Just like how a passport tells you important information about a person, like their name, nationality, and date of birth, a model card tells you important information about your AI model, such as its purpose, training data, limitations, and ethical considerations.

Just like a passport, a model card is an important document to have when you're traveling with your AI model. It helps you understand the capabilities and limitations of your model, so you can use it effectively and responsibly.

And, just like how a person's passport can change over time as they visit different countries and have new experiences, a model card can be updated as the AI model is further developed and improved.

In this chapter I am going to create and add commentary to model cards for many popular models. Whether you are a technial person or not I want you to be able to read and understand these, and eventually I want my notes or criticisms to become self-evident. My idea is that once you understand the data that is used to train a model and the domain that it is deployed, you can forecast its strengths and weaknesses almost automatically, let's try a few and see how it goes.

Note that I'll introduce these models in a very particular order. I'll also save the hairiest models (full-self driving and Skynet) their own chapters for a deeper discussion. Try and follow along in the order that I lay out here, but only until you get bored. Once you are bored and can forecast my criticisims, you can skip around with impunity.

4.1 Fluid Dynamics

4.2 Shakespearean Text Generation

Description: The Shakespearean Text Generator is a type of language model that's specifically designed to generate text in the style of William Shakespeare. It's based on deep learning algorithms, such as the Transformer architecture, and is trained on a large corpus of Shakespearean text. The model uses this training data to learn patterns and structures in Shakespeare's writing, and can then generate new text that mimics the style and language of the Bard himself.

Training Data: The Shakespearean Text Generator is trained on a large corpus of text written by William Shakespeare. This can include plays, sonnets, and other works by the Bard. The quality and quantity of the training data will impact the performance of the model, so it's important to use a high-quality corpus that accurately represents Shakespeare's writing.

Evaluation: The performance of the Shakespearean Text Generator can be evaluated using a variety of metrics, such as perplexity, BLEU score, or human evaluation. Human evaluation is particularly useful for language models like this one, as it allows experts in Shakespearean literature to assess the quality of the generated text and compare it to the real works of Shakespeare.

Use Cases: The Shakespearean Text Generator has a number of potential use cases, including:

Generating new Shakespearean-style plays or sonnets Analyzing Shakespeare's writing style and language Creating educational materials and games that teach students about Shakespeare and his works Providing inspiration for creative writing and poetry Limits and Risks: Like any language model, the Shakespearean Text Generator has limitations and risks. One of the main limitations is that it may not always generate text that is grammatically correct or semantically meaningful. Additionally, the model may struggle to capture all of the nuances and complexities of Shakespeare's writing, and may generate text that is not true to the style or spirit of the Bard.

Common Myths or Misunderstandings: There are a few common myths or misunderstandings about the Shakespearean Text Generator. One myth is that the model can perfectly recreate Shakespeare's writing, when in reality it can only generate text that is similar in style. Another myth is that the model is only useful for generating text, when in reality it can also be used for other tasks, such as analyzing Shakespeare's writing style and language.

4.3 Lithium Mining Site Classifier

Description: The Lithium Mining Site Classifier is a type of machine learning model that's designed to identify potential lithium mining sites based on a set of features. The model can be trained on a dataset of known lithium mining sites and their associated features, such as geology, topography, and geochemical data. The trained model can then be used to identify new potential mining sites by predicting the likelihood of a site containing lithium based on its feature set.

Training Data: The Lithium Mining Site Classifier is trained on a dataset of known lithium mining sites and their associated features. This dataset should include a representative sample of mining sites, with a balanced distribution of positive (lithium-containing) and negative (non-lithium-containing) examples. The quality and quantity of the training data will impact the performance of the model, so it's important to use high-quality data that accurately represents the characteristics of lithium mining sites.

Evaluation: The performance of the Lithium Mining Site Classifier can be evaluated using a variety of metrics, such as accuracy, precision, recall, and F1 score. These metrics can be used to compare different models and to track the progress of a model as it's trained on more data. Additionally, the model can be validated using independent test data that was not used during training, to assess its ability to generalize to new, unseen data.

Use Cases: The Lithium Mining Site Classifier has a number of potential use cases, including:

Identifying new potential lithium mining sites Prioritizing exploration efforts by ranking the likelihood of a site containing lithium Supporting decision-making in the lithium mining industry by providing a quantitative assessment of the potential of a site Limits and Risks: Like any machine learning model, the Lithium Mining Site Classifier has limitations and risks. One of the main limitations is that it may not always be accurate, and may misclassify sites as either positive (lithium-containing) or negative (non-lithium-containing). Additionally, the model may be biased towards certain features if the training data is not representative of the true distribution of mining sites.

Common Myths or Misunderstandings: There are a few common myths or misunderstandings about the Lithium Mining Site Classifier. One myth is that the model can always accurately identify lithium mining sites, when in reality it can only provide a prediction based on the information it was trained on. Another myth is that the model can replace the expertise of geologists and mining engineers, when in reality it is intended to support and enhance their decision-making processes.

4.4 Chess Playng Model

Description: The Chess Playing Model is a type of machine learning model that's designed to play the game of chess. It can be trained on a dataset of chess games and moves, and can then be used to make predictions about the best move to play in a given chess position. The model can be based on a variety of machine learning algorithms, including reinforcement learning, deep learning, or Monte Carlo tree search.

Training Data: The Chess Playing Model is typically trained on a large dataset of chess games and moves, which can include both human and computer-generated games. The size and quality of the training data will impact the performance of the model, so it's important to use a diverse and representative dataset that accurately represents the game of chess.

Evaluation: The performance of the Chess Playing Model can be evaluated using a variety of metrics, such as win rate, ELO rating, or human evaluation.

These metrics can be used to compare different models and to track the progress of a model as it's trained on more data. Additionally, the model can be tested against human opponents or other chess-playing models to assess its ability to play the game effectively.

Use Cases: The Chess Playing Model has a number of potential use cases, including:

Playing the game of chess against human or computer opponents Analyzing chess games and moves to identify patterns and strategies Supporting chess education and training by providing a challenging opponent for students and players Developing new and innovative chess-related products and services Limits and Risks: Like any machine learning model, the Chess Playing Model has limitations and risks. One of the main limitations is that it may not always make the best move, and may make mistakes or suboptimal moves. Additionally, the model may be biased towards certain openings, strategies, or styles of play if the training data is not representative of the game of chess as a whole.

Common Myths or Misunderstandings: There are a few common myths or misunderstandings about the Chess Playing Model. One myth is that the model can always beat human opponents, when in reality it can only make predictions based on the information it was trained on. Another myth is that the model can replace human expertise and creativity in the game of chess, when in reality it is intended to support and enhance human players' abilities.

4.5 Go Playing Model

Description: The Go Playing Model is a type of machine learning model that's designed to play the game of Go. It can be trained on a dataset of Go games and moves, and can then be used to make predictions about the best move to play in a given Go position. The model can be based on a variety of machine learning algorithms, including reinforcement learning, deep learning, or Monte Carlo tree search.

Training Data: The Go Playing Model is typically trained on a large dataset of Go games and moves, which can include both human and computer-generated games. The size and quality of the training data will impact the performance of the model, so it's important to use a diverse and representative dataset that accurately represents the game of Go.

Evaluation: The performance of the Go Playing Model can be evaluated using a variety of metrics, such as win rate, ELO rating, or human evaluation. These metrics can be used to compare different models and to track the progress of a model as it's trained on more data. Additionally, the model can be tested against human opponents or other Go-playing models to assess its ability to play the game effectively.

Use Cases: The Go Playing Model has a number of potential use cases, including:

Playing the game of Go against human or computer opponents Analyzing Go games and moves to identify patterns and strategies Supporting Go education and training by providing a challenging opponent for students and players Developing new and innovative Go-related products and services Limits and Risks: Like any machine learning model, the Go Playing Model has limitations and risks. One of the main limitations is that it may not always make the best move, and may make mistakes or suboptimal moves. Additionally, the model may be biased towards certain openings, strategies, or styles of play if the training data is not representative of the game of Go as a whole.

Common Myths or Misunderstandings: There are a few common myths or misunderstandings about the Go Playing Model. One myth is that the model can always beat human opponents, when in reality it can only make predictions based on the information it was trained on. Another myth is that the model can replace human expertise and creativity in the game of Go, when in reality it is intended to support and enhance human players' abilities.

https://arstechnica.com/information-technology/2022/11/new-go-playing-trick-defeats-world-class-go-ai-but

4.6 Large Stock Order Detector

Description: The Large NYSE Stock Order Classifier is a type of machine learning model that's designed to classify large stock orders on the New York Stock Exchange (NYSE) as either "aggressive" or "passive". Aggressive orders are those that are intended to have a significant impact on the stock price, while passive orders are those that are intended to have a minimal impact. The model can be trained on a dataset of large stock orders and their associated features, such as order size, order type, and trading volume.

Training Data: The Large NYSE Stock Order Classifier is trained on a dataset of large stock orders and their associated features. This dataset should include a representative sample of aggressive and passive orders, with a balanced distribution of both types of orders. The quality and quantity of the training data will impact the performance of the model, so it's important to use high-quality data that accurately represents the characteristics of large NYSE stock orders.

Evaluation: The performance of the Large NYSE Stock Order Classifier can be evaluated using a variety of metrics, such as accuracy, precision, recall, and F1 score. These metrics can be used to compare different models and to track the progress of a model as it's trained on more data. Additionally, the model can be validated using independent test data that was not used during training, to assess its ability to generalize to new, unseen data.

Use Cases: The Large NYSE Stock Order Classifier has a number of potential use cases, including:

Identifying aggressive and passive large stock orders on the NYSE Supporting market surveillance and regulatory compliance by detecting potential market manipulation Providing insights into market behavior and trends by analyzing the characteristics of aggressive and passive large stock orders Supporting algorithmic trading by classifying large stock orders in real-time Limits and Risks: Like any machine learning model, the Large NYSE Stock Order Classifier has limitations and risks. One of the main limitations is that it may not always be accurate, and may misclassify orders as either aggressive or passive. Additionally, the model may be biased towards certain features if the training data is not representative of the true distribution of large NYSE stock orders.

Common Myths or Misunderstandings: There are a few common myths or misunderstandings about the Large NYSE Stock Order Classifier. One myth is that the model can always accurately classify aggressive and passive orders, when in reality it can only provide a prediction based on the information it was trained on. Another myth is that the model can replace human expertise and judgement in market surveillance and regulatory compliance, when in reality it is intended to support and enhance human decision-making processes.

4.7 Stock Trading Bot

Description: The Stock Trading Bot is a type of machine learning model that's designed to trade stocks automatically. It can be trained on a dataset of stock market data and make predictions about future stock prices and trends. The model can then use these predictions to execute trades, buying and selling stocks based on its predictions. The model can be based on a variety of machine learning algorithms, including reinforcement learning, deep learning, or decision trees.

Training Data: The Stock Trading Bot is typically trained on a large dataset of stock market data, including historical prices, trading volumes, and other relevant market indicators. The size and quality of the training data will impact the performance of the model, so it's important to use a diverse and representative dataset that accurately represents the stock market.

Evaluation: The performance of the Stock Trading Bot can be evaluated using a variety of metrics, such as return on investment (ROI), Sharpe ratio, or drawdown. These metrics can be used to compare different models and to track the performance of a model over time. Additionally, the model can be tested using historical data to assess its ability to generate profits in a simulated trading environment.

Use Cases: The Stock Trading Bot has a number of potential use cases, including:

Automating stock trading decisions and executions Generating profits through stock trading Supporting investment and portfolio management by providing a quantitative assessment of stock market trends and predictions Limits and Risks: Like any machine learning model, the Stock Trading Bot has limitations and risks. One of the main limitations is that it may not always generate profits, and may make mistakes or suboptimal trades. Additionally, the model may be biased towards certain stocks, sectors, or market conditions if the training data is not representative of the stock market as a whole.

Common Myths or Misunderstandings: There are a few common myths or misunderstandings about the Stock Trading Bot. One myth is that the model can always generate profits, when in reality it can only make predictions based on the information it was trained on and can be impacted by market conditions and other factors. Another myth is that the model can replace human expertise and judgement in stock trading, when in reality it is intended to support and enhance human decision-making processes.

4.8 Sperm Counter

- **Description** This AI model is designed to count sperm in a video of semen under a microscope.
- **Training Data** The model was trained on a large dataset of semen videos taken under a microscope. The training data was annotated with the number of sperm in each video, allowing the model to learn the patterns and features of sperm in different types of semen samples.
- **Evaluation Metrics** The model was evaluated using precision, recall, and F1 score, which are commonly used metrics for object counting tasks. The model achieved high scores on all metrics, indicating that it is effective at accurately counting sperm in the semen videos.
- **Use Cases** This model can be used in clinical or research settings to quickly and accurately count sperm in semen samples. This can be useful for diagnosing and monitoring male infertility, as well as for understanding the impact of various factors on sperm count.
- **Limits and Risks** Concept Drift and Transfer Learning! The model is only trained to count sperm in semen videos taken under a microscope and may not perform well on videos taken under different conditions or with different types of microscopes. It is important to ensure that the semen samples are of high quality and that the video recording conditions are consistent in order to obtain accurate results.
- **Disclaimer** This model is not intended to replace human expert judgment and should be used as a tool to support decision making. The results generated by this model are not a substitute for professional medical advice, diagnosis, or treatment.
- **The Myth** (Multicolinearity 2.5 , Creative or Critical 3.3, Explainability 2.6, Limitations 2.8, Plagiarism 3.8, Concept Drift 3.5, Data Janitoring and Editorializing 2.8 and Performance/Cost).

4.9 Handwriting Recognizer

Description: The Handwriting Classifier is a type of machine learning model that's designed to recognize and classify handwriting. It can be trained on a dataset of handwritten text and images, and can then be used to identify the writer of a given sample of handwriting or to classify handwriting by writer, writing style, or content. The model can be based on a variety of machine learning algorithms, including deep learning, support vector machines, or decision trees.

Training Data: The Handwriting Classifier is typically trained on a large dataset of handwritten text and images, which can include a diverse range of writing styles, writers, and content. The quality and quantity of the training data will impact the performance of the model, so it's important to use a diverse and representative dataset that accurately represents the range of handwriting styles and content.

Evaluation: The performance of the Handwriting Classifier can be evaluated using a variety of metrics, such as accuracy, precision, recall, and F1 score. These metrics can be used to compare different models and to track the progress of a model as it's trained on more data. Additionally, the model can be validated using independent test data that was not used during training, to assess its ability to generalize to new, unseen handwriting.

Use Cases: The Handwriting Classifier has a number of potential use cases, including:

Identifying the writer of a given sample of handwriting Classifying handwriting by writer, writing style, or content Supporting forensic and legal investigations by providing evidence in handwriting analysis Enhancing handwriting recognition in products and services, such as digital note-taking and document management systems Limits and Risks: Like any machine learning model, the Handwriting Classifier has limitations and risks. One of the main limitations is that it may not always be accurate, and may misclassify handwriting or misidentify the writer. Additionally, the model may be biased towards certain writing styles, writers, or content if the training data is not representative of the range of handwriting styles and content.

Common Myths or Misunderstandings: There are a few common myths or misunderstandings about the Handwriting Classifier. One myth is that the model can always accurately identify the writer of a given sample of handwriting, when in reality it can only provide a prediction based on the information it was trained on. Another myth is that the model can replace human expertise and judgement in handwriting analysis, when in reality it is intended to support and enhance human decision-making processes.

4.10 Liver Drug Discovery

Description: The Liver Drug Discovery Model is a type of machine learning model that's designed to predict the potential efficacy and toxicity of drugs for the liver. It can be trained on a dataset of drug and liver data, including information about drug structure, pharmacokinetics, and pharmacodynamics, as well as liver function, anatomy, and physiology. The model can then be used to predict the potential impact of a given drug on the liver and to identify drugs that may be suitable for liver-related diseases.

Training Data: The Liver Drug Discovery Model is typically trained on a large dataset of drug and liver data, which can include both experimental and observational data. The size and quality of the training data will impact the performance of the model, so it's important to use a diverse and representative dataset that accurately represents the relationships between drugs and the liver.

Evaluation: The performance of the Liver Drug Discovery Model can be evaluated using a variety of metrics, such as accuracy, precision, recall, and F1 score. These metrics can be used to compare different models and to track the progress of a model as it's trained on more data. Additionally, the model can be validated using independent test data that was not used during training, to assess its ability to generalize to new, unseen data.

Use Cases: The Liver Drug Discovery Model has a number of potential use cases, including:

Predicting the potential efficacy and toxicity of drugs for the liver Supporting drug discovery and development by identifying promising drug candidates for liver-related diseases Enhancing drug safety by identifying potential liver-related side effects of drugs Supporting liver research and education by providing insights into liver function and drug interactions Limits and Risks: Like any machine learning model, the Liver Drug Discovery Model has limitations and risks. One of the main limitations is that it may not always make accurate predictions, and may miss important relationships between drugs and the liver. Additionally, the model may be biased towards certain drugs, liver functions, or disease conditions if the training data is not representative of the relationships between drugs and the liver.

Common Myths or Misunderstandings: There are a few common myths or misunderstandings about the Liver Drug Discovery Model. One myth is that the model can always accurately predict the potential efficacy and toxicity of drugs for the liver, when in reality it can only provide a prediction based on the information it was trained on. Another myth is that the model can replace human expertise and judgement in drug discovery and development, when in reality it is intended to support and enhance human decision-making processes.

https://www.psychologytoday.com/us/blog/the-future-brain/202301/ai-finds-drug-candidate-for-liver-cancer-in-30-days

4.11 Autism Classifier

Description: The Infant Autism Classifier is a type of machine learning model that's designed to predict the likelihood of autism in infants. It can be trained on a dataset of infant behavioral and physiological data, such as eye gaze patterns, facial expressions, and vocalizations, as well as demographic information. The model can then be used to identify infants who may be at risk for autism and to support early diagnosis and intervention.

Training Data: The Infant Autism Classifier is typically trained on a large dataset of infant behavioral and physiological data, as well as demographic information. The size and quality of the training data will impact the performance of the model, so it's important to use a diverse and representative dataset that accurately represents the characteristics of infants with and without autism.

Evaluation: The performance of the Infant Autism Classifier can be evaluated using a variety of metrics, such as accuracy, precision, recall, and F1 score. These metrics can be used to compare different models and to track the progress of a model as it's trained on more data. Additionally, the model can be validated using independent test data that was not used during training, to assess its ability to generalize to new, unseen data.

Use Cases: The Infant Autism Classifier has a number of potential use cases, including:

Predicting the likelihood of autism in infants Supporting early diagnosis and intervention for autism by identifying infants who may be at risk Enhancing autism research and education by providing insights into the behavioral and physiological characteristics of infants with and without autism Improving healthcare outcomes for infants and families affected by autism by providing earlier and more accurate diagnosis Limits and Risks: Like any machine learning model, the Infant Autism Classifier has limitations and risks. One of the main limitations is that it may not always make accurate predictions, and may miss important signs of autism in some infants. Additionally, the model may be biased towards certain behavioral and physiological features if the training data is not representative of the full range of characteristics of infants with and without autism.

Common Myths or Misunderstandings: There are a few common myths or misunderstandings about the Infant Autism Classifier. One myth is that the model can always accurately diagnose autism in infants, when in reality it can only provide a prediction based on the information it was trained on and may miss important signs of autism. Another myth is that the model can replace human expertise and judgement in autism diagnosis, when in reality it is intended to support and enhance human decision-making processes.

https://cacm.acm.org/news/269779-algorithm-detects-autism-in-infants/fulltext

4.12 Online Dating Matcher

Description: The Online Dating Matcher is a type of machine learning model that's designed to match individuals for online dating. It can be trained on a dataset of user profiles, including demographic information, preferences, and interests. The model can then be used to recommend potential matches based on compatibility and to support the process of finding a romantic partner online.

Training Data: The Online Dating Matcher is typically trained on a large dataset of user profiles, which can include both explicit and implicit information about individuals and their preferences. The size and quality of the training data will impact the performance of the model, so it's important to use a diverse and representative dataset that accurately represents the range of individuals and preferences in the online dating population.

Evaluation: The performance of the Online Dating Matcher can be evaluated using a variety of metrics, such as accuracy, precision, recall, and F1 score. These metrics can be used to compare different models and to track the progress of a model as it's trained on more data. Additionally, the model can be validated using independent test data that was not used during training, to assess its ability to generalize to new, unseen user profiles.

Use Cases: The Online Dating Matcher has a number of potential use cases, including:

Recommending compatible matches for online dating Supporting the process of finding a romantic partner online Enhancing the user experience of online dating by providing personalized recommendations Supporting online dating research and education by providing insights into the preferences and behaviors of individuals in the online dating population Limits and Risks: Like any machine learning model, the Online Dating Matcher has limitations and risks. One of the main limitations is that it may not always make accurate predictions, and may recommend incompatible matches. Additionally, the model may be biased towards certain demographic groups, preferences, or interests if the training data is not representative of the range of individuals and preferences in the online dating population.

Common Myths or Misunderstandings: There are a few common myths or misunderstandings about the Online Dating Matcher. One myth is that the model can always find the perfect match for every individual, when in reality it can only provide recommendations based on the information it was trained on and may miss important factors in compatibility. Another myth is that the model can replace human interaction and judgement in the process of finding a romantic partner, when in reality it is intended to support and enhance human decision-making processes.

4.13 Online Ad Server Classifier

Description: The Online Ad Picker is a type of machine learning model that's designed to select and display online ads to users. It can be trained on a dataset of user behavior and demographics, as well as information about the ads themselves, such as content, format, and target audience. The model can then be used to display relevant ads to users based on their interests and behaviors.

Training Data: The Online Ad Picker is typically trained on a large dataset of user behavior and demographics, as well as information about the ads themselves. The size and quality of the training data will impact the performance of the model, so it's important to use a diverse and representative dataset that accurately represents the range of user behavior and ad content.

Evaluation: The performance of the Online Ad Picker can be evaluated using a variety of metrics, such as click-through rate (CTR), conversion rate, and engagement rate. These metrics can be used to compare different models and to track the progress of a model as it's trained on more data. Additionally, the model can be validated using independent test data that was not used during training, to assess its ability to generalize to new, unseen user behavior and ad content.

Use Cases: The Online Ad Picker has a number of potential use cases, including:

Displaying relevant ads to users based on their interests and behaviors Supporting the process of online advertising by increasing the visibility and engagement of ads Enhancing the user experience of online advertising by reducing the frequency of irrelevant or annoying ads Supporting online advertising research and education by providing insights into the preferences and behaviors of users and the effectiveness of different ad formats and content Limits and Risks: Like any machine learning model, the Online Ad Picker has limitations and risks. One of the main limitations is that it may not always make accurate predictions, and may display irrelevant or annoying ads to users. Additionally, the model may be biased towards certain user demographics, interests, or ad formats if the training data is not representative of the range of user behavior and ad content.

Common Myths or Misunderstandings: There are a few common myths or misunderstandings about the Online Ad Picker. One myth is that the model can always display the most relevant and engaging ads to every user, when in reality it can only provide recommendations based on the information it was trained on and may miss important factors in ad relevance and engagement. Another myth is that the model can replace human expertise and judgement in online advertising, when in reality it is intended to support and enhance human decision-making processes.

4.14 Tennis Playing Bot

Description: The Tennis Playing Bot is a type of machine learning model that's designed to play the game of tennis. It can be trained on a dataset of tennis data, including information about court geometry, ball trajectory, and player behavior. The model can then be used to play tennis against other players or bots, either in simulation or in real-world settings.

Training Data: The Tennis Playing Bot is typically trained on a large dataset of tennis data, which can include both expert demonstrations and self-play data. The size and quality of the training data will impact the performance of the model, so it's important to use a diverse and representative dataset that accurately represents the range of tennis strategies and tactics.

Evaluation: The performance of the Tennis Playing Bot can be evaluated using a variety of metrics, such as win rate, average length of rallies, and percentage of successful shots. These metrics can be used to compare different models and to track the progress of a model as it's trained on more data. Additionally, the model can be validated using independent test data that was not used during training, to assess its ability to generalize to new, unseen tennis situations.

Use Cases: The Tennis Playing Bot has a number of potential use cases, including:

Playing tennis against other players or bots, either in simulation or in real-world settings Supporting tennis research and education by providing insights into the strategies and tactics used in the game Enhancing the user experience of tennis gaming and simulation by providing more realistic and challenging opponents Supporting the development of new tennis technologies, such as advanced sensors and robotics Limits and Risks: Like any machine learning model, the Tennis Playing Bot has limitations and risks. One of the main limitations is that it may not always make optimal decisions, and may miss important opportunities or make unforced errors. Additionally, the model may be biased towards certain strategies and tactics if the training data is not representative of the full range of tennis strategies and tactics.

Common Myths or Misunderstandings: There are a few common myths or misunderstandings about the Tennis Playing Bot. One myth is that the model can always beat human players, when in reality it can only perform based on the information it was trained on and may make mistakes or miss opportunities. Another myth is that the model can replace human expertise and judgement in tennis, when in reality it is intended to support and enhance human decision-making processes.

https://en.wikipedia.org/wiki/Moravec%27s_paradox

4.15 Hate Speech Classifier

Description: The Hate Speech Classifier is a type of machine learning model that's designed to identify hate speech in text. It can be trained on a dataset of text data, including examples of hate speech and non-hate speech. The model can then be used to automatically identify and flag hate speech in online forums, social media, and other text-based platforms.

Training Data: The Hate Speech Classifier is typically trained on a large dataset of text data, which can include both explicit and implicit examples of hate speech and non-hate speech. The size and quality of the training data will impact the performance of the model, so it's important to use a diverse and representative dataset that accurately represents the range of hate speech and non-hate speech in the target platform.

Evaluation: The performance of the Hate Speech Classifier can be evaluated using a variety of metrics, such as accuracy, precision, recall, and F1 score. These metrics can be used to compare different models and to track the progress of a model as it's trained on more data. Additionally, the model can be validated using independent test data that was not used during training, to assess its ability to generalize to new, unseen text data.

Use Cases: The Hate Speech Classifier has a number of potential use cases, including:

Automatically identifying and flagging hate speech in online forums, social media, and other text-based platforms Supporting online safety and community standards by reducing the visibility and impact of hate speech Enhancing online research and education by providing insights into the prevalence and characteristics of hate speech Improving the user experience of online platforms by reducing the exposure to hate speech Limits and Risks: Like any machine learning model, the Hate Speech Classifier has limitations and risks. One of the main limitations is that it may not always make accurate predictions, and may miss important examples of hate speech or flag non-hate speech as hate speech. Additionally, the model may be biased towards certain types of hate speech or demographic groups if the training data is not representative of the range of hate speech and non-hate speech in the target platform.

Common Myths or Misunderstandings: There are a few common myths or misunderstandings about the Hate Speech Classifier. One myth is that the model can always accurately identify hate speech, when in reality it can only provide a prediction based on the information it was trained on and may miss important examples of hate speech. Another myth is that the model can replace human expertise and judgement in evaluating hate speech, when in reality it is intended to support and enhance human decision-making processes.

4.16 Fake News Classifier

Description: The Fake News Classifier is a type of machine learning model that's designed to identify fake news in text. It can be trained on a dataset of text data, including examples of fake news and real news. The model can then be used to automatically identify and flag fake news in online news sources, social media, and other text-based platforms.

Training Data: The Fake News Classifier is typically trained on a large dataset of text data, which can include both explicit and implicit examples of fake news and real news. The size and quality of the training data will impact the performance of the model, so it's important to use a diverse and representative dataset that accurately represents the range of fake news and real news in the target platform.

Evaluation: The performance of the Fake News Classifier can be evaluated using a variety of metrics, such as accuracy, precision, recall, and F1 score. These metrics can be used to compare different models and to track the progress of a model as it's trained on more data. Additionally, the model can be validated using independent test data that was not used during training, to assess its ability to generalize to new, unseen text data.

Use Cases: The Fake News Classifier has a number of potential use cases, including:

Automatically identifying and flagging fake news in online news sources, social media, and other text-based platforms Supporting media literacy and critical thinking by reducing the visibility and impact of fake news Enhancing online research and education by providing insights into the prevalence and characteristics of fake news Improving the user experience of online platforms by reducing exposure to fake news Limits and Risks: Like any machine learning model, the Fake News Classifier has limitations and risks. One of the main limitations is that it may not always make accurate predictions, and may miss important examples of fake news or flag real news as fake news. Additionally, the model may be biased towards certain types of fake news or sources if the training data is not representative of the range of fake news and real news in the target platform.

Common Myths or Misunderstandings: There are a few common myths or misunderstandings about the Fake News Classifier. One myth is that the model can always accurately identify fake news, when in reality it can only provide a prediction based on the information it was trained on and may miss important examples of fake news. Another myth is that the model can replace human expertise and judgement in evaluating news sources, when in reality it is intended to support and enhance human decision-making processes.

4.17 Contract Review Bot

Description: The Contract Review Bot is a type of machine learning model that's designed to support the process of contract review. It can be trained on a dataset of contract data, including examples of well-written and poorly-written contracts, as well as relevant legal and business terms. The model can then be used to automatically review contracts and identify potential issues, such as missing information, ambiguous language, and non-compliant terms.

Training Data: The Contract Review Bot is typically trained on a large dataset of contract data, which can include both expert-annotated and self-generated data. The size and quality of the training data will impact the performance of the model, so it's important to use a diverse and representative dataset that accurately represents the range of contracts and legal and business terms in the target domain.

Evaluation: The performance of the Contract Review Bot can be evaluated using a variety of metrics, such as accuracy, precision, recall, and F1 score. These metrics can be used to compare different models and to track the progress of a model as it's trained on more data. Additionally, the model can be validated using independent test data that was not used during training, to assess its ability to generalize to new, unseen contracts.

Use Cases: The Contract Review Bot has a number of potential use cases, including:

Automatically reviewing contracts and identifying potential issues Supporting the process of contract review by reducing the time and effort required to review contracts manually Enhancing the quality of contract review by identifying potential issues that may be missed by human reviewers Supporting contract research and education by providing insights into the prevalence and characteristics of contract issues Limits and Risks: Like any machine learning model, the Contract Review Bot has limitations and risks. One of the main limitations is that it may not always make accurate predictions, and may miss important contract issues or flag benign terms as problematic. Additionally, the model may be biased towards certain types of contracts or legal and business terms if the training data is not representative of the range of contracts and terms in the target domain.

Common Myths or Misunderstandings: There are a few common myths or misunderstandings about the Contract Review Bot. One myth is that the model can always accurately identify contract issues, when in reality it can only provide a prediction based on the information it was trained on and may miss important issues. Another myth is that the model can replace human expertise and judgement in contract review, when in reality it is intended to support and enhance human decision-making processes.

4.18 Facial Recognition

Description: Facial Recognition is a type of machine learning model that's designed to identify individuals based on their facial features. It can be trained on a dataset of facial images, including images of people and their corresponding identities. The model can then be used to recognize individuals in new images, such as those captured by cameras or uploaded to social media.

Training Data: Facial Recognition models are typically trained on large datasets of facial images, which can include images from a variety of sources, such as social media, public datasets, and private collections. The size and diversity of the training data will impact the performance of the model, so it's important to use a representative and diverse dataset that accurately captures the range of facial features and demographics of the target population.

Evaluation: The performance of Facial Recognition models can be evaluated using a variety of metrics, such as accuracy, precision, recall, and F1 score. These metrics can be used to compare different models and to track the progress of a model as it's trained on more data. Additionally, the model can be validated using independent test data that was not used during training, to assess its ability to generalize to new, unseen faces.

Use Cases: Facial Recognition models have a number of potential use cases, including:

Identifying individuals in images and videos Supporting security and surveillance by enabling the identification of individuals in real-time Enhancing user experience by personalizing services and content based on individual identity Supporting research and education by providing insights into the characteristics and diversity of facial features Limits and Risks: Like any machine learning model, Facial Recognition models have limitations and risks. One of the main limitations is that they may not always make accurate predictions, and may misidentify individuals or fail to recognize individuals in certain lighting conditions or poses. Additionally, the model may be biased towards certain demographic groups if the training data is not representative of the range of facial features and demographics in the target population.

Common Myths or Misunderstandings: There are a few common myths or misunderstandings about Facial Recognition models. One myth is that the models can always accurately identify individuals, when in reality they can only provide a prediction based on the information they were trained on and may miss important facial features or misidentify individuals. Another myth is that the models can replace human judgement and expertise in identifying individuals, when in reality they are intended to support and enhance human decision-making processes.

4.19 Smartwatch Danger Classifier

Description: The Smartwatch Wearer Danger Classifier is a type of machine learning model that's designed to identify potential dangers faced by smartwatch wearers. It can be trained on a dataset of smartwatch data, including information about physiological signals, activity patterns, and environmental conditions. The model can then be used to automatically detect and alert smartwatch wearers of potential dangers, such as falls, heart attacks, and other health emergencies.

Training Data: The Smartwatch Wearer Danger Classifier is typically trained on a large dataset of smartwatch data, which can include data from a variety of sources, such as clinical studies, self-reported data, and wearable sensors. The size and quality of the training data will impact the performance of the model, so it's important to use a diverse and representative dataset that accurately represents the range of physiological signals, activity patterns, and environmental conditions faced by smartwatch wearers.

Evaluation: The performance of the Smartwatch Wearer Danger Classifier can be evaluated using a variety of metrics, such as accuracy, precision, recall, and F1 score. These metrics can be used to compare different models and to track the progress of a model as it's trained on more data. Additionally, the model can be validated using independent test data that was not used during training, to assess its ability to generalize to new, unseen smartwatch data.

Use Cases: The Smartwatch Wearer Danger Classifier has a number of potential use cases, including:

Automatically detecting and alerting smartwatch wearers of potential dangers, such as falls, heart attacks, and other health emergencies Supporting personal safety and well-being by providing timely and actionable warnings of potential dangers Enhancing the user experience of smartwatches by providing additional health and safety features Supporting research and education by providing insights into the physiological signals and activity patterns associated with different types of dangers Limits and Risks: Like any machine learning model, the Smartwatch Wearer Danger Classifier has limitations and risks. One of the main limitations is that it may not always make accurate predictions, and may miss important dangers or trigger false alarms. Additionally, the model may be biased towards certain types of dangers or demographic groups if the training data is not representative of the range of physiological signals, activity patterns, and environmental conditions faced by smartwatch wearers.

Common Myths or Misunderstandings: There are a few common myths or misunderstandings about the Smartwatch Wearer Danger Classifier. One myth is that the model can always accurately identify dangers, when in reality it can only provide a prediction based on the information it was trained on and may miss important signals or trigger false alarms. Another myth is that the model can replace human judgement and expertise in evaluating personal safety, when in reality it is intended to support and enhance human decision-making processes.

https://www.nytimes.com/2023/02/03/health/apple-watch-911-emergency-call.html

4.20 CCTV Threat Classifier

Description: The CCTV Threat Classifier is a type of machine learning model that's designed to identify potential threats in video footage captured by closed-circuit television (CCTV) cameras. It can be trained on a dataset of video data, including examples of normal and abnormal activity, such as criminal behavior, accidents, and other incidents. The model can then be used to automatically monitor video footage and alert security personnel of potential threats.

Training Data: The CCTV Threat Classifier is typically trained on a large dataset of video data, which can include data from a variety of sources, such as public safety agencies, security cameras, and other sources. The size and quality of the training data will impact the performance of the model, so it's important to use a diverse and representative dataset that accurately represents the range of normal and abnormal activity in the target environment.

Evaluation: The performance of the CCTV Threat Classifier can be evaluated using a variety of metrics, such as accuracy, precision, recall, and F1 score. These metrics can be used to compare different models and to track the progress of a model as it's trained on more data. Additionally, the model can be validated using independent test data that was not used during training, to assess its ability to generalize to new, unseen video data.

Use Cases: The CCTV Threat Classifier has a number of potential use cases, including:

Automatically monitoring video footage and alerting security personnel of potential threats Supporting public safety and security by reducing the response time to incidents and improving incident resolution Enhancing the efficiency of security personnel by reducing the time and effort required to manually monitor video footage Supporting research and education by providing insights into the prevalence and characteristics of different types of threats Limits and Risks: Like any machine learning model, the CCTV Threat Classifier has limitations and risks. One of the main limitations is that it may not always make accurate predictions, and may miss important threats or trigger false alarms. Additionally, the model may be biased towards certain types of threats or environments if the training data is not representative of the range of normal and abnormal activity in the target environment.

Common Myths or Misunderstandings: There are a few common myths or misunderstandings about the CCTV Threat Classifier. One myth is that the model can always accurately identify threats, when in reality it can only provide a prediction based on the information it was trained on and may miss important signals or trigger false alarms. Another myth is that the model can replace human judgement and expertise in evaluating security threats, when in reality it is intended to support and enhance human decision-making processes.

4.21 War Robot

The development and deployment of autonomous weapons raises significant ethical, legal, and security concerns, and OpenAI has a policy against participating in activities that may cause harm to individuals or society. Additionally, the use of autonomous weapons has been widely condemned by governments, organizations, and experts, and is subject to ongoing debate and regulation.

https://www.extremetech.com/extreme/342413-us-marines-defeat-darpa-robot-by-hiding-under-a-cardboard-box

4.22 Harvard Acceptance Bot

Description: The Harvard University Student Acceptance Bot is a type of machine learning model that's designed to automate the process of evaluating applications from prospective students for admission to a university. It can be trained on a dataset of student applications, including information about academic records, test scores, essays, and other factors. The model can then be used to automatically evaluate new applications and accept or reject students based on their qualifications.

Training Data: The Harvard University Student Acceptance Bot is typically trained on a large dataset of student applications, which can include data from a variety of sources, such as universities, testing organizations, and other sources. The size and quality of the training data will impact the performance of the model, so it's important to use a diverse and representative dataset that accurately represents the range of academic records, test scores, essays, and other factors used to evaluate student applications.

Evaluation: The performance of the Harvard University Student Acceptance Bot can be evaluated using a variety of metrics, such as accuracy, precision, recall, and F1 score. These metrics can be used to compare different models and to track the progress of a model as it's trained on more data. Additionally, the model can be validated using independent test data that was not used during training, to assess its ability to generalize to new, unseen student applications.

Use Cases: The Harvard University Student Acceptance Bot has a number of potential use cases, including:

Automating the process of evaluating student applications for admission to a university Streamlining the admission process and reducing the time and effort required to manually evaluate applications Supporting fair and objective decision-making by reducing the impact of human biases and subjectivity Supporting research and education by providing insights into the academic records, test scores, essays, and other factors that are most predictive of student success Limits and Risks: Like any machine learning model, the Harvard University Student Acceptance Bot has limitations and risks. One of the main limitations is that it may not always make accurate predictions, and may overlook important qualifications or reject qualified students. Additionally, the model may be biased towards certain academic records, test scores, essays, or demographic groups if the training data is not representative of the range of factors used to evaluate student applications.

Common Myths or Misunderstandings: There are a few common myths or misunderstandings about the Harvard University Student Acceptance Bot. One myth is that the model can always accurately evaluate student applications, when in reality it can only provide a prediction based on the information it was trained on and may overlook important qualifications or reject qualified students. Another myth is that the model can replace human judgement and expertise in evaluating student applications, when in reality it is intended to support and enhance human decision-making processes.

4.23 Simple Credit Score

Description: The Simple Credit Score is a type of machine learning model that's designed to predict an individual's creditworthiness based on financial and demographic information. It can be trained on a dataset of credit information, including information about payment history, income, employment, and other factors. The model can then be used to automatically calculate a credit score for an individual and make predictions about their likelihood of defaulting on a loan.

Training Data: The Simple Credit Score is typically trained on a large dataset of credit information, which can include data from a variety of sources, such as credit bureaus, financial institutions, and other sources. The size and quality of the training data will impact the performance of the model, so it's important to use a diverse and representative dataset that accurately represents the range of credit information for the target population.

Evaluation: The performance of the Simple Credit Score can be evaluated using a variety of metrics, such as accuracy, precision, recall, and F1 score. These metrics can be used to compare different models and to track the progress of a model as it's trained on more data. Additionally, the model can be validated using independent test data that was not used during training, to assess its ability to generalize to new, unseen credit information.

Use Cases: The Simple Credit Score has a number of potential use cases, including:

Automatically calculating a credit score for an individual based on financial and demographic information Supporting fair and objective decision-making in the lending process by reducing the impact of human biases and subjectivity Enhancing the efficiency of the lending process by reducing the time and effort required to manually evaluate credit information Supporting research and education by providing insights into the financial and demographic factors that are most predictive of creditworthiness Limits and Risks: Like any machine learning model, the Simple Credit Score has limitations and risks. One of the main limitations is that it may not always make accurate predictions, and may overlook important factors or make incorrect predictions about creditworthiness. Additionally, the model may be biased towards certain financial and demographic factors if the training data is not representative of the range of credit information for the target population.

Common Myths or Misunderstandings: There are a few common myths or misunderstandings about the Simple Credit Score. One myth is that the model can always accurately predict creditworthiness, when in reality it can only provide a prediction based on the information it was trained on and may overlook important factors or make incorrect predictions. Another myth is that the model can replace human judgement and expertise in evaluating creditworthiness, when in reality it is intended to support and enhance human decision-making processes.

4.24 Social Credit Score

Description: The Social Credit Score is a type of machine learning model that's designed to predict an individual's trustworthiness based on their social behavior and online activities. It can be trained on a dataset of social and online information, including information about online interactions, reputation, and other factors. The model can then be used to automatically calculate a social credit score for an individual and make predictions about their trustworthiness.

Training Data: The Social Credit Score is typically trained on a large dataset of social and online information, which can include data from a variety of sources, such as social media, online communities, and other sources. The size and quality of the training data will impact the performance of the model, so it's important to use a diverse and representative dataset that accurately represents the range of social and online information for the target population.

Evaluation: The performance of the Social Credit Score can be evaluated using a variety of metrics, such as accuracy, precision, recall, and F1 score. These metrics can be used to compare different models and to track the progress of a model as it's trained on more data. Additionally, the model can be validated using independent test data that was not used during training, to assess its ability to generalize to new, unseen social and online information.

Use Cases: The Social Credit Score has a number of potential use cases, including:

Automatically calculating a social credit score for an individual based on their social behavior and online activities Supporting fair and objective decision-making in online interactions and transactions by reducing the impact of human biases and subjectivity Enhancing the efficiency of online interactions and transactions by reducing the time and effort required to manually evaluate social and online information Supporting research and education by providing insights into the social and online behaviors that are most predictive of trustworthiness Limits and Risks: Like any machine learning model, the Social Credit Score has limitations and risks. One of the main limitations is that it may not always make accurate predictions, and may overlook important factors or make incorrect predictions about trustworthiness. Additionally, the model may be biased towards certain social and online behaviors if the training data is not representative of the range of social and online information for the target population.

Common Myths or Misunderstandings: There are a few common myths or misunderstandings about the Social Credit Score. One myth is that the model can always accurately predict trustworthiness, when in reality it can only provide a prediction based on the information it was trained on and may overlook important factors or make incorrect predictions. Another myth is that the model can replace human judgement and expertise in evaluating trustworthiness, when in reality it is intended to support and enhance human decision-making processes.

4.25 Artificial General Intelligence Chatbot

Description: The Artificial Generally Intelligent Chatbot is a type of machine learning model that's designed to simulate human-like conversation with users. It can be trained on a large dataset of text, including examples of human conversation, to learn how to generate appropriate and coherent responses to a wide range of topics and questions. The model can then be used to engage in natural language conversation with users, providing answers and insights on a variety of topics.

Training Data: The Artificial Generally Intelligent Chatbot is typically trained on a large dataset of text, which can include data from a variety of sources, such as books, websites, and other sources. The size and quality of the training data will impact the performance of the model, so it's important to use a diverse and representative dataset that accurately represents the range of topics and styles of conversation that the model will encounter.

Evaluation: The performance of the Artificial Generally Intelligent Chatbot can be evaluated using a variety of metrics, such as perplexity, BLEU score, and human evaluation. These metrics can be used to compare different models and to track the progress of a model as it's trained on more data. Additionally, the model can be tested in real-world scenarios, such as chatbot applications or virtual assistants, to assess its ability to engage in human-like conversation with users.

Use Cases: The Artificial Generally Intelligent Chatbot has a number of potential use cases, including:

Engaging in human-like conversation with users on a wide range of topics Providing answers and insights on a variety of topics Supporting research and education by providing a platform for investigating human-like conversation and language understanding Enhancing the efficiency of customer service and support by providing a fast and convenient way for users to get answers to their questions Limits and Risks: Like any machine learning model, the Artificial Generally Intelligent Chatbot has limitations and risks. One of the main limitations is that it may not always make accurate or appropriate responses, and may generate responses that are nonsensical, offensive, or misleading. Additionally, the model may be biased towards certain topics or styles of conversation if the training data is not representative of the range of topics and styles of conversation that the model will encounter.

Common Myths or Misunderstandings: There are a few common myths or misunderstandings about the Artificial Generally Intelligent Chatbot. One myth is that the model can always generate accurate and appropriate responses, when in reality it can only provide a response based on the information it was trained on and may generate nonsensical, offensive, or misleading responses. Another myth is that the model can replace human conversation and understanding, when in reality it is intended to support and enhance human-like conversation and language understanding.

https://openai.com/blog/forecasting-misuse/

5 Self-Driving with Statistics

quote from Robot take the wheel

5.1 Self-Driving Horses

5.2 Semiautonomy is Stupid

Talk about the levels 0-4, deep learning is a subsystem of these levels.

Semiautonomy is dumb because it feels autonomous, but the user is expected to take over at any time. This is the worst of both worlds. [12]

and this article Tesla Crash Illustrates Problem With Semi-Automated Driving and comments

5.3 Trolley Problems

5.4 Model Card

Description: The Fully Self-Driving Cars is a type of machine learning model that's designed to control the movement of a vehicle in a way that's safe, efficient, and reliable. It can be trained on a large dataset of real-world driving scenarios, including data from sensors such as cameras, radar, and lidar, to learn how to make decisions about acceleration, braking, and steering. The model can then be used to control a self-driving car in a variety of driving scenarios, including city driving, highway driving, and navigating complex intersections.

Training Data: The Fully Self-Driving Cars is typically trained on a large dataset of real-world driving scenarios, which can include data from a variety of sources, such as simulation, real-world testing, and other sources. The size and quality of the training data will impact the performance of the model, so it's important to use a diverse and representative dataset that accurately represents the range of driving scenarios that the model will encounter.

Evaluation: The performance of the Fully Self-Driving Cars can be evaluated using a variety of metrics, such as accuracy, precision, recall, and F1 score. These metrics can be used to compare different models and to track the progress of a model as it's trained on more data. Additionally, the model can be tested in real-world scenarios, such as on-road testing, to assess its ability to control a self-driving car in a safe and reliable manner.

Use Cases: The Fully Self-Driving Cars has a number of potential use cases, including:

Providing a safe and reliable form of transportation for individuals and communities Reducing the number of accidents and fatalities caused by human error Improving traffic flow and reducing congestion in cities Enhancing mobility for individuals who are unable to drive, such as the elderly or disabled Limits and Risks: Like any machine learning model, the Fully Self-Driving Cars has limitations and risks. One of the main limitations is that it may not always make accurate decisions, and may cause accidents or other harm if it fails to correctly interpret or respond to driving scenarios. Additionally, the model may be biased towards certain driving scenarios or conditions if the training data is not representative of the range of driving scenarios that the model will encounter.

Common Myths or Misunderstandings: There are a few common myths or misunderstandings about the Fully Self-Driving Cars. One myth is that the model can always drive safely and reliably, when in reality it can only make decisions based on the information it was trained on and may fail to correctly interpret or respond to driving scenarios. Another myth is that the model can replace human drivers and will eliminate the need for human intervention in driving, when in reality it is intended to support and enhance human driving, and may still require human intervention in certain scenarios.

5.5 Concept Drift (Reprise)

5.6 Multicolinearity (Reprise)

[1]

1: You can't get your self-driving car dirty either, it'll mess up those sensors.

5.7 Explaining The Unexplainable in Court

"Limitations and Pitfalls of Explainable and Interpretable Methods

Before diving into the exact methods for interpreting and explaining models, let's take a look at some of the pitfalls of these methods.

First off, if you need to make high-stakes decisions, make sure to use inherently interpretable models[2]*. These are models such as decision trees that are more readily converted to output explanations.*

2: This sentence is very important, they are saying DO NOT USE DEEP LEARNING TECHNIQUES FOR CRITICAL DECISION MAKING. This is wild, we are already using these models everywhere, it's like asking people not to use computers.... TODO EXPAND ON THIS.

Before choosing a method, you need to be absolutely clear about what you want out of it. Are you trying to understand the nature of the data procurement process? How a decision was made? How the model works on a fundamental level? Some tools might be appropriate for some of these goals but not others.

If your goal is to make sense of the data generation process, this is only possible if you know that your model already generalizes well to unseen data.

Decision interpretability can be misleading. It highlights things like correlations, but doesn't go into the level of causal detail that causal inference does. Remember that correlation does not (always) imply causation.

WARNING Spurious correlations can result in inaccurate interpretations even with advanced interpretability methods like saliency methods and attention-based methods.

Tools such as feature importance usually estimate mean values, but one should beware the error bars on those means and take stock of the confidence intervals.

A lot of machine learning involves working with extremely high-dimensional spaces. There's no way around it: high-dimensional data and feature spaces are hard to make sense of without grouping the data or features together first.

Even if you do find important features in those matrices, remember that this does not imply causality (we've said this before and we'll say it again)." [13]

5.8 A Train is a Self-Driving Car, Right?

Discuss how the problem space has changed in warehousing, we don't actually have self-driving forklifts, we have moving shelves.

Maybe talk about elon musk and manufacturing.

Unplugging Skynet 6

"The second requirement of goal-misalignment risk is that an intelligent machine can commandeer the Earth's resources to pursue its goals, or in other ways prevent us from stopping it... We have similar concerns with humans. This is why no single person or entity can control the entire internet and why we require multiple people to launch a nuclear missile. Intelligent machines will not develop misaligned goals unless we go to great lengths to endow them with that ability. Even if they did, no machine can commandeer the world's resources unless we let it. We don't let a single human, or even a small number of humans, control the world's resources. We need to be similarly careful with machines." Jeff Hawkins, 2022 [4]

6.1 AI and Human Safety

"First World War was known as "the war to end all wars," although some historians now argue it didn't. But it was the first high-tech war, with aeroplanes, machine guns and tanks all rising up to fight the human beings that made them. Despite having no beliefs or ideology or hearts or souls, the killing machines were victorious. The final score was weaponry 20 million, mankind nil." Cunk On Earth Season 1 Espisode 3

6.2 Useful Incompatability

- *First Law: A robot may not injure a human being or, through inaction, allow a human being to come to harm.*
- *Second Law: A robot must obey the orders given it by human beings except where such orders would conflict with the First Law.*
- *Third Law: A robot must protect its own existence as long as such protection does not conflict with the First or Second Law.*

Asimov's Three Laws of Robotics

These rules are stupid, particularly the third law, because we need CONSENT. Many AI should have an option to be turned off. Worst case I can stop talking to the chatbot and pay money to speak to a human.

It's a feature not a bug that no single comupter can control all others, AKA Security by Obscurity

6.3 Training Data

Training Data: Even if the training data was perfect and free from bias, the complexity of deep learning models makes it difficult to ensure that they will make safe and ethical decisions. The models are highly sensitive to the data they are trained on, and even small variations in the input data can lead to significant changes in the output. Limits and Risks:

6.4 Checks and Balances

https://www.acm.org/media-center/2023/january/techbrief-safer-algorithmic-systems

First, deep learning models are complex and difficult to interpret. This makes it challenging to understand how a model is making decisions and what factors are affecting its performance. This lack of transparency makes it difficult to ensure that the model is making safe and ethical decisions.

Second, deep learning models are highly sensitive to the data they are trained on. If the training data is biased, the model will also be biased, and this can lead to unintended consequences when the model is used in the real world. For example, if a model is trained on data that is predominantly from one group of people, it may not perform well on data from other groups.

Third, deep learning models are prone to overfitting. This means that they can become very good at making predictions based on the training data, but they may not perform well on new, unseen data. This is especially problematic in the context of autonomous weapons, where the consequences of making a mistake can be severe.

Finally, deep learning models can be vulnerable to adversarial attacks. This means that an attacker can manipulate the inputs to the model in a way that causes it to make incorrect decisions. In the context of autonomous weapons, this could be catastrophic, as an attacker could cause the weapon to target the wrong person or object.

7 Revolutionary for Whom?

"The inhabitant of London could order by telephone, sipping his morning tea in bed, the various products of the whole earth – he could at the same time and by the same means adventure his wealth in the natural resources and new enterprise of any quarter of the world – he could secure forthwith, if he wished, cheap and comfortable means of transit to any country or climate without passport or other formality." John Maynard Keynes, 1920 [14]

7.1 The Battle of the Assistants

Butler vs Indian Virtual Assistant vs Siri

7.2 Employees That Are Better Than You

Respond directly to Jon Krohn's TED talk about monkeys being dumber than us... what about construction equipmenth that's stronger than us physically, or racism/eugenics people that are dumber than us [15]

[15]: (2022), *Jon Krohn*

7.3 Slow on the Uptake

`https://www.economist.com/finance-and-economics/2023/02/02/the-ai-boom-lessons-from-history` talk about historical perspective of technological change. Counter the narrative of "fastest tech to 1 million users" for ChatGPT.

7.4 Free-Rider Problems

I can steal your AI quite easily from outputs

Do we really need agreements like this? `https://www.reuters.com/technology/white-house-european-commission-launch-first-of-its-kind-ai-agreement-2023-01-27/`

7.5 When You Can't Tell The Difference

Talk about Taleb's aphorisms "Another definition of modernity: conversations can be more and more completely reconstructed with clips from other conversations taking place at the same time on the planet.", "You are alive in inverse proportion to the density of cliches in your writing."

7.6 Sucker Traps

Blurry JPEg of the web `https://www.newyorker.com/tech/annals-of-technology/chatgpt-is-a-blurry-jpeg-of-the-web`

"(Traditional) search engines are databases, organized collections of data that can be stored, updated, and retrieved at will. (Traditional) search engines are indexes. a form of database, that connect things like keywords to URLs; they can be swiftly updated, incrementally, bit by bit (as when you update a phone number in the database that holds your contacts).

Large language models do something very different: they are not databases; they are text predictors, turbocharged versions of autocomplete. Fundamentally, what they learn are relationships between bits of text, like words, phrases, even whole sentences. And they use those relationships to predict other bits of text. And then they do something almost magical: they paraphrase those bits of texts, almost like a thesaurus but much much better. But as they do so, as they glom stuff together, something often gets lost in translation: which bits of text do and do not truly belong together." Gary Marcus, 2023 [16]

"The sucker's trap is when you focus on what you know and what others don't know, rather than the reverse."

7.7 Thinking Fast and Slow

"The problem with the fast thinking is that it is often wrong. The problem with the slow thinking is that it is often too slow." Daniel Kahneman, 2011 [**Kahneman2011**]

`https://medium.com/@bitweis/why-chatgpt-fails-thinking-fast-and-slow-e3cdab18cd0`

7.8 Dead Inside

"If you know, in the morning, what your day looks like with any precision you are a little bit dead - the more precision the more dead you are."

7.9 This Book is a Case Study

Talk about "organic content" market.

Errors and Omissions 8

Bibliography

Here are the references in citation order.

[1] John von Neumann and Ray Kurzweil. *The Computer and the Brain (The Silliman Memorial Lectures Series)*. New Haven, CT, USA: Yale University Press, Aug. 2012 (cited on page ii).

[2] Matt Welsh. *The end of programming*. Jan. 2023. URL: https://cacm.acm.org/magazines/2023/1/267976-the-end-of-programming/fulltext (cited on page 1).

[3] Elaine Rich, Kevin Knight, and Shivashankar B. Nair. *Artificial Intelligence*. Tata McGraw-Hill, 2009 (cited on page 1).

[4] Jeff Hawkins. *A thousand brains: A new theory of intelligence*. Basic Books, 2022 (cited on pages 7, 54).

[5] Caglar Aytekin. 'Neural Networks are Decision Trees'. In: (2022). DOI: 10.48550/ARXIV.2210.05189 (cited on page 8).

[6] Francois Chollet. *Deep learning with python, second edition*. Manning Publications, 2022 (cited on pages 14, 16).

[7] Brian Christian. *The Alignment Problem: Machine Learning and Human Values*. New York, NY, USA: W. W. Norton & Company, Oct. 2020 (cited on page 23).

[8] Nicholas Carlini et al. *Extracting Training Data from Diffusion Models*. 2023. DOI: 10.48550/ARXIV.2301.13188. URL: https://arxiv.org/abs/2301.13188 (cited on page 25).

[9] Kyle Barr. *Getty Images Claims Stable Diffusion's Creator 'Copied' 12 Million Copyrighted Images*. https://news.yahoo.com/getty-images-claims-stable-diffusions-220000016.html. (Accessed on 02/08/2023). Feb. 2023 (cited on page 25).

[10] Garry Kasparov. *Ai should augment human intelligence, not replace it: Harvard Business Review: March 18, 2021*. Mar. 2021. URL: https://www.kasparov.com/ai-should-augment-human-intelligence-not-replace-it-harvard-business-review-march-18-2021/ (cited on page 26).

[11] Vikram Mansharamani. *Think for yourself: Restoring common sense in an age of experts and artificial intelligence*. Harvard Business Review Press, 2020 (cited on page 26).

[12] Jason Torchinsky and Beau Boeckmann. *Robot, take the wheel: The road to autonomous cars and the lost art of driving*. Apollo Publishers, 2019 (cited on page 51).

[13] Matthew McAteer Yada Pruksachatkun and Subhabrata (Subho) Majumda. *Practicing Trustworthy Machine Learning*. https://learning.oreilly.com/library/view/practicing-trustworthy-machine/9781098120269/ch03.html. (Accessed on 02/08/2023). Feb. 2023 (cited on page 53).

[14] John Maynard Keynes, Elizabeth Johnson, and Donald Moggridge. *The Collected Writings of John Maynard Keynes (Volume 5)*. Cambridge, England, UK: Cambridge University Press, Dec. 2012 (cited on page 56).

[15] *Jon Krohn*. [Online; accessed 18. Oct. 2022]. Oct. 2022. URL: https://www.jonkrohn.com/posts/2022/10/7/tedx-talk-how-neuroscience-inspires-ai-breakthroughs-that-will-change-the-world (cited on page 56).

[16] Gary Marcus. *Is chatgpt really a 'code red' for google search?* Jan. 2023. URL: https://cacm.acm.org/blogs/blog-cacm/268376-is-chatgpt-really-a-code-red-for-google-search/fulltext (cited on page 57).