Machine Learning

Making A Computer Look Like It's Thinking

Tim Jackman

BU Summer Challenge

July 17th, 2025

Prediction

- Yesterday we were looking at decision problems, today we'll look at *learning problems*

Prediction

- Yesterday we were looking at decision problems, today we'll look at *learning problems*
- In a learning problem, an algorithm "learns" from a data set (called the training set) and tries to "predict" for new data

Prediction

- Yesterday we were looking at decision problems, today we'll look at *learning problems*
- In a learning problem, an algorithm "learns" from a data set (called the training set) and tries to "predict" for new data
  - For example, we might want to "predict" a student's grade based on how many hours they studied

Prediction

- Yesterday we were looking at decision problems, today we'll look at *learning problems*
- In a learning problem, an algorithm "learns" from a data set (called the training set) and tries to "predict" for new data
  - For example, we might want to "predict" a student's grade based on how many hours they studied
  - We have 1000s of examples in our training data (# hours, letter grade) (e.g. (10, A), (5, B), (0, F), etc.)

Prediction

- Yesterday we were looking at decision problems, today we'll look at *learning problems*
- In a learning problem, an algorithm "learns" from a data set (called the training set) and tries to "predict" for new data
  - For example, we might want to "predict" a student's grade based on how many hours they studied
  - We have 1000s of examples in our training data (# hours, letter grade) (e.g. (10, A), (5, B), (0, F), etc.)
  - Given that a new student has studied 7 hours, our algorithm will try to predict what grade they'll get

Prediction

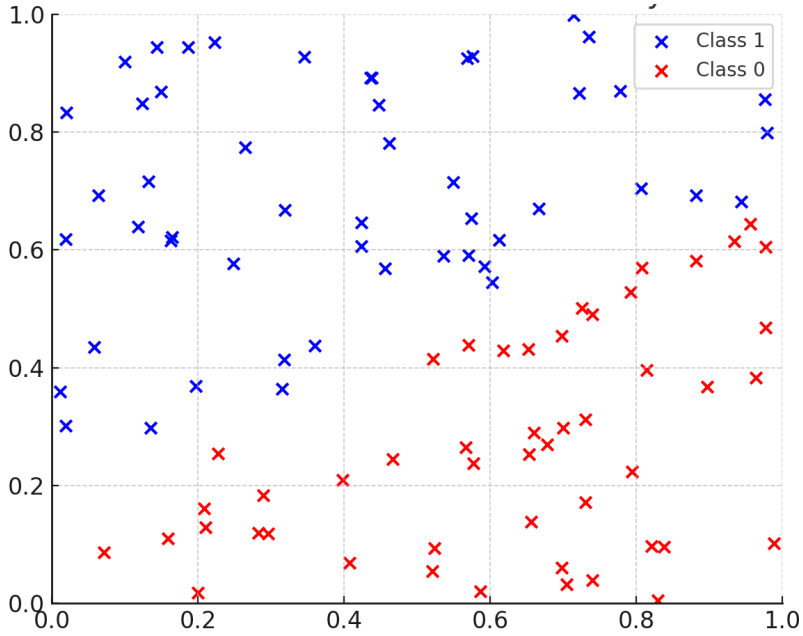
- Yesterday we were looking at decision problems, today we'll look at *learning problems*
- In a learning problem, an algorithm "learns" from a data set (called the training set) and tries to "predict" for new data
  - For example, we might want to "predict" a student's grade based on how many hours they studied
  - We have 1000s of examples in our training data (# hours, letter grade) (e.g. (10, A), (5, B), (0, F), etc.)
  - Given that a new student has studied 7 hours, our algorithm will try to predict what grade they'll get
  - Randomness is inherent here, our algorithm will sometimes be wrong but we want it to be right more than just randomly guessing

Simple Prediction

- Imagine we had our data sets where points fall into two categories, red/blue:

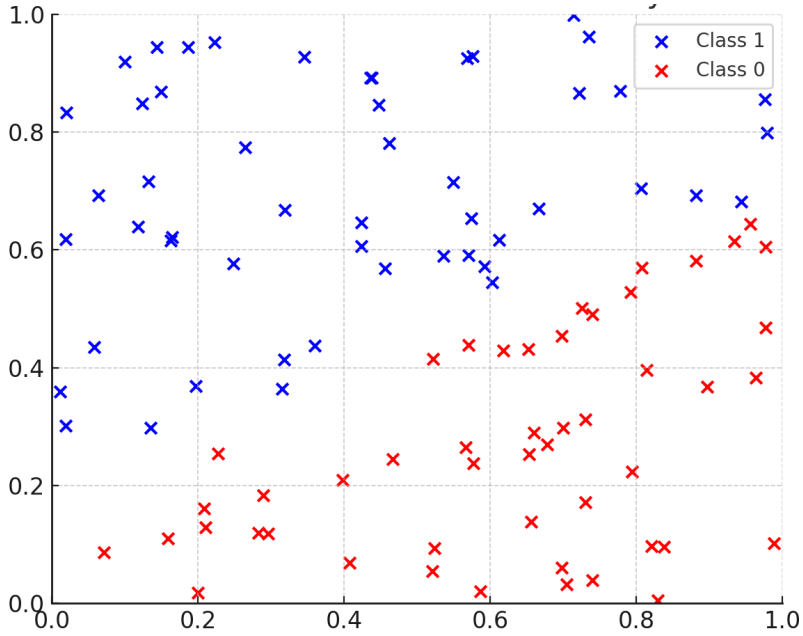
Simple Prediction

- Imagine we had our data sets where points fall into two categories, red/blue:



Simple Prediction

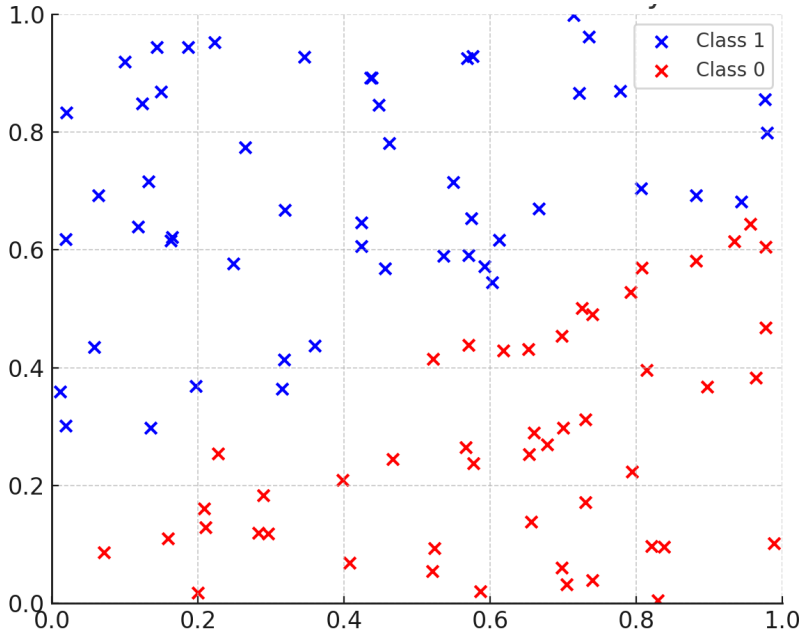
- Imagine we had our data sets where points fall into two categories, red/blue:



- Given a new point, how should we decide whether it should be red/blue?

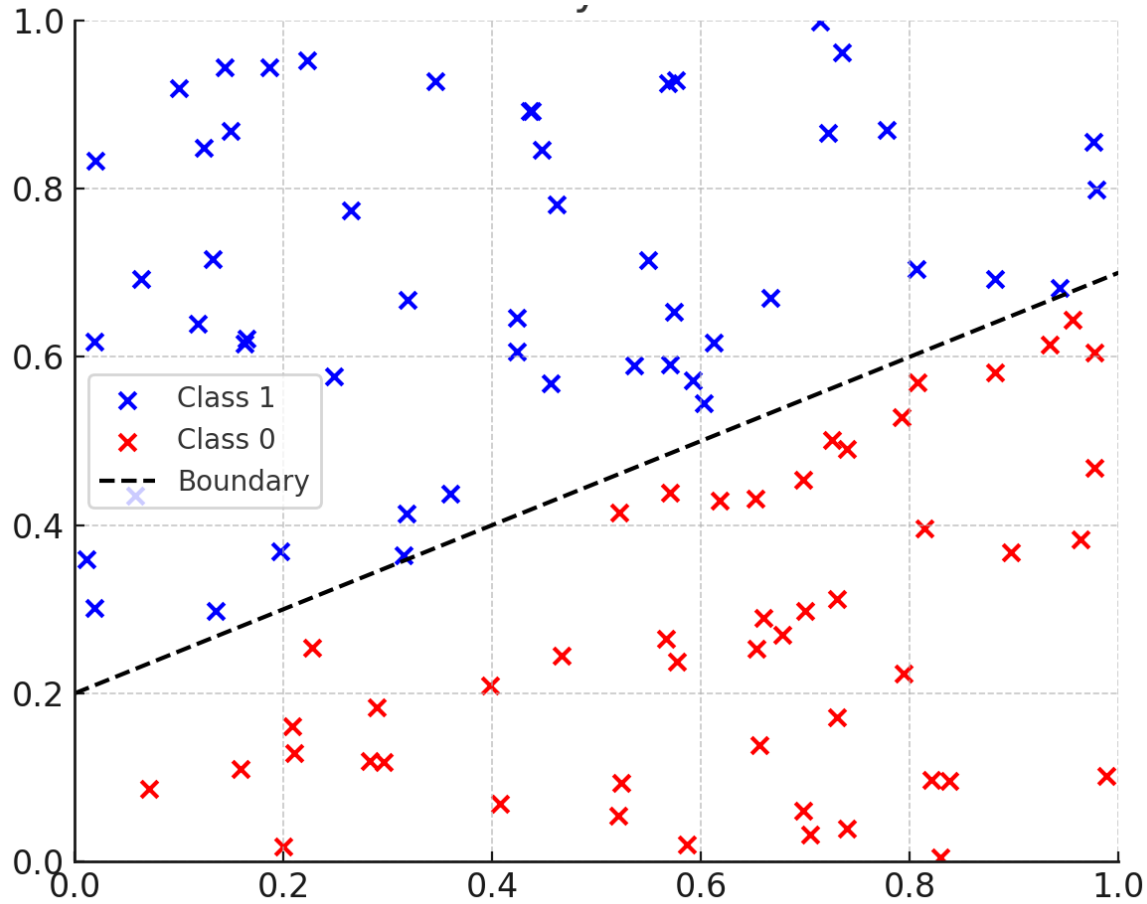
Simple Prediction

- Imagine we had our data sets where points fall into two categories, red/blue:



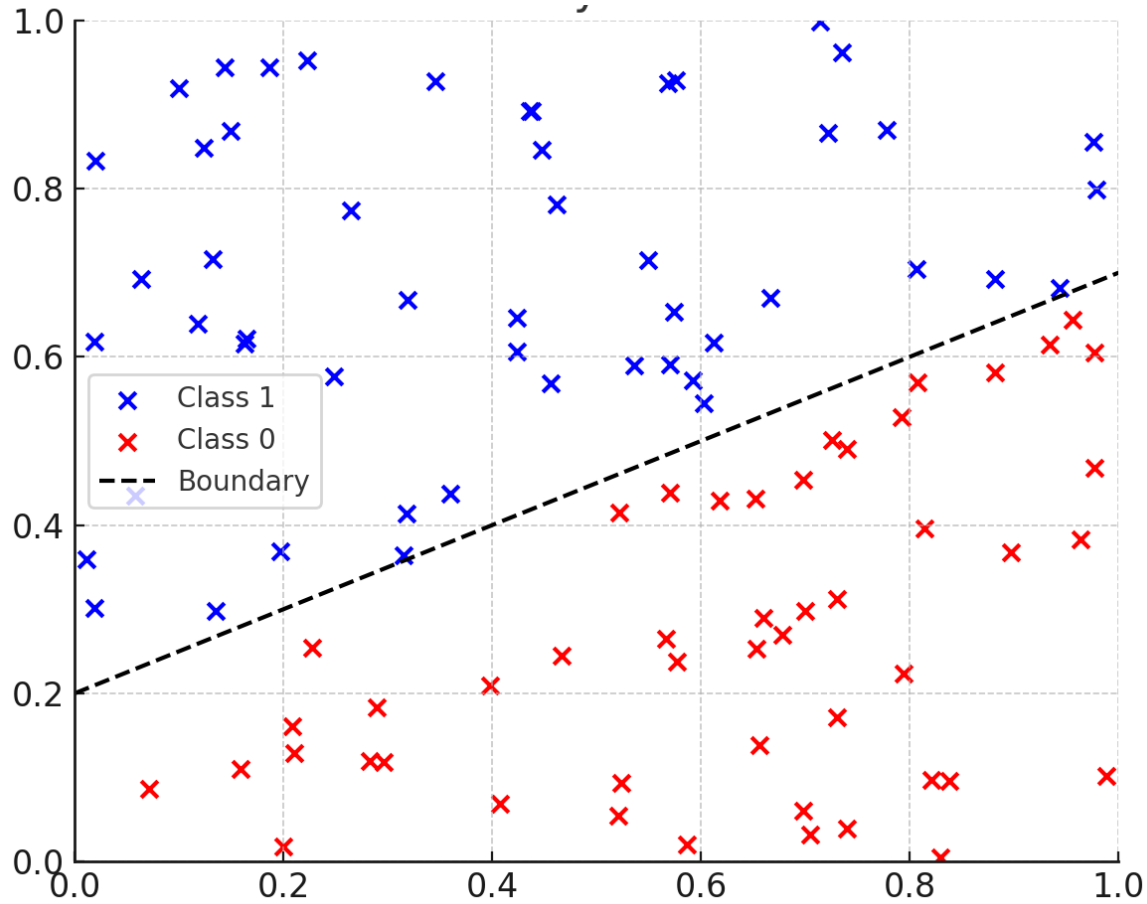
- Given a new point, how should we decide whether it should be red/blue?
- We can draw trying drawing a line that's separating the bulk of the red and blue:

Simple Prediction



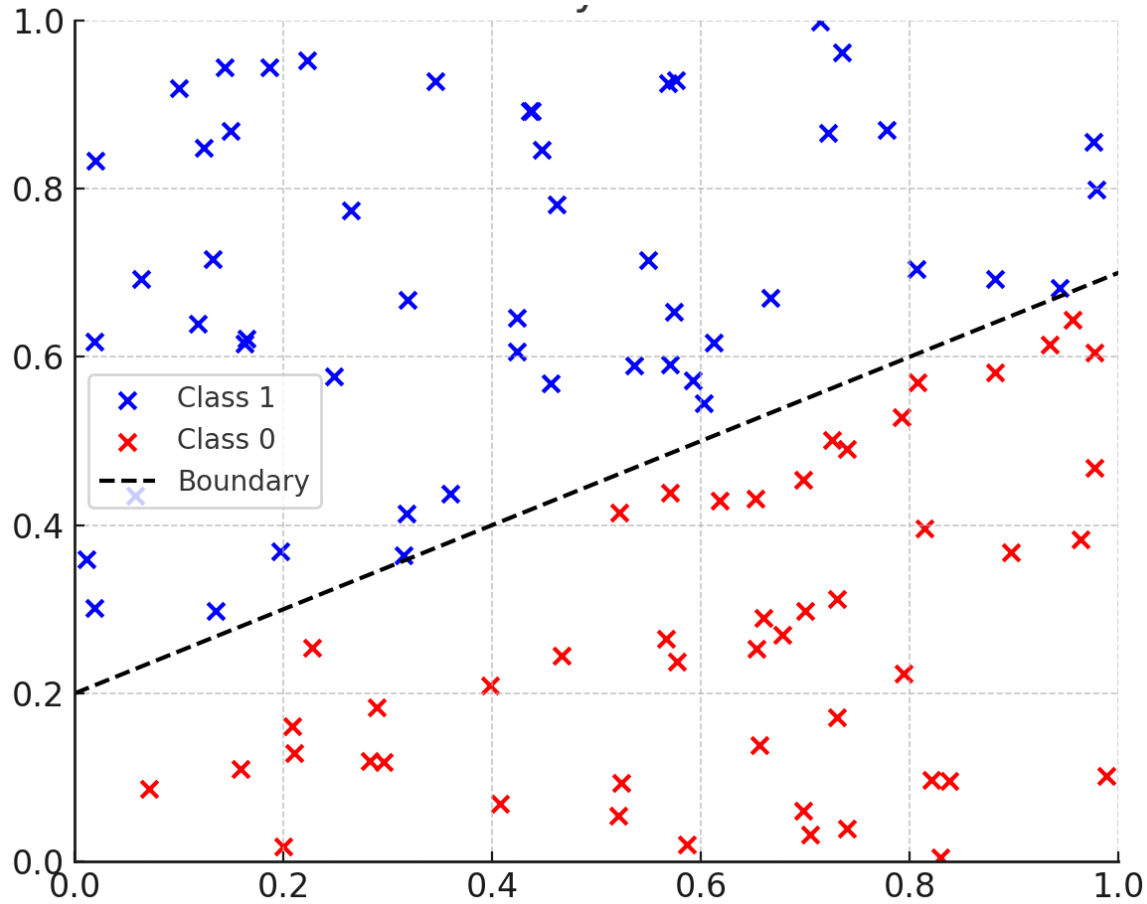
- Given a new point, we just check if it's above or below this line

Simple Prediction



- Given a new point, we just check if it's above or below this line
- But how do we work out what this line should be?

Simple Prediction



- Given a new point, we just check if it's above or below this line
- But how do we work out what this line should be?
- How do we judge how good a line is?

Loss Function and Gradient Descent

- To judge a line, we compute a *loss function* (or *cost function*) which takes in our prediction and compares how it does on the training data

Loss Function and Gradient Descent

- To judge a line, we compute a *loss function* (or *cost function*) which takes in our prediction and compares how it does on the training data
  - If our line misclassifies a bunch of points, it will have high loss

Loss Function and Gradient Descent

- To judge a line, we compute a *loss function* (or *cost function*) which takes in our prediction and compares how it does on the training data
  - If our line misclassifies a bunch of points, it will have high loss
- We can then adjust our line by moving it slightly in the direction that best improves our loss (i.e. minimizes loss after the update)

Loss Function and Gradient Descent

- To judge a line, we compute a *loss function* (or *cost function*) which takes in our prediction and compares how it does on the training data
  - If our line misclassifies a bunch of points, it will have high loss
- We can then adjust our line by moving it slightly in the direction that best improves our loss (i.e. minimizes loss after the update)
- This is a technique called *gradient descent* (if you know/are learning calculus, we compute the derivative of the loss function and see what change would lead to a minimum)

Loss Function and Gradient Descent

- To judge a line, we compute a *loss function* (or *cost function*) which takes in our prediction and compares how it does on the training data
  - If our line misclassifies a bunch of points, it will have high loss
- We can then adjust our line by moving it slightly in the direction that best improves our loss (i.e. minimizes loss after the update)
- This is a technique called *gradient descent* (if you know/are learning calculus, we compute the derivative of the loss function and see what change would lead to a minimum)
- We repeat this many times until we get a good prediction

Predicting Sequences

- This previous task was *classification* problems, machine learning is also used for *generation* problems

Predicting Sequences

- This previous task was *classification* problems, machine learning is also used for *generation* problems
- Given a sequence of data, output the next part of the sequence

Predicting Sequences

- This previous task was *classification* problems, machine learning is also used for *generation* problems
- Given a sequence of data, output the next part of the sequence
- For example, given a series of words (partial sentence), can we predict the next word (or the rest of the sentence)?

Predicting Sequences

- This previous task was *classification* problems, machine learning is also used for *generation* problems
- Given a sequence of data, output the next part of the sequence
- For example, given a series of words (partial sentence), can we predict the next word (or the rest of the sentence)?
- To do this, the algorithm "learns" by breaking down many examples and finding *associations* between words

Predicting Sequences

- This previous task was *classification* problems, machine learning is also used for *generation* problems
- Given a sequence of data, output the next part of the sequence
- For example, given a series of words (partial sentence), can we predict the next word (or the rest of the sentence)?
- To do this, the algorithm "learns" by breaking down many examples and finding *associations* between words
  - If the sequence is "I am going to the" then in it's training data it might have seen that "going to the" was often finished with store, movies, beach, etc.

Predicting Sequences

- This previous task was *classification* problems, machine learning is also used for *generation* problems
- Given a sequence of data, output the next part of the sequence
- For example, given a series of words (partial sentence), can we predict the next word (or the rest of the sentence)?
- To do this, the algorithm "learns" by breaking down many examples and finding *associations* between words
  - If the sequence is "I am going to the" then in it's training data it might have seen that "going to the" was often finished with store, movies, beach, etc.
  - Based on it's data, it will weight these words with a probability: `store:30%, movies:10%, beach:5%, ..., moon:0.01%`

Predicting Sequences

- This previous task was *classification* problems, machine learning is also used for *generation* problems
- Given a sequence of data, output the next part of the sequence
- For example, given a series of words (partial sentence), can we predict the next word (or the rest of the sentence)?
- To do this, the algorithm "learns" by breaking down many examples and finding *associations* between words
  - If the sequence is "I am going to the" then in it's training data it might have seen that "going to the" was often finished with store, movies, beach, etc.
  - Based on it's data, it will weight these words with a probability: `store:30%, movies:10%, beach:5%, ..., moon:0.01%`
  - It will then randomly predict the next word based on the probability

Large Language Models

- Large Language Models (LLMs) like ChatGPT are this but on steroids

Large Language Models

- Large Language Models (LLMs) like ChatGPT are this but on steroids
- Their training data is massive and how they arrive at probabilities is very complex and uses a lot of Linear Algebra (matrices) and calculus

Large Language Models

- Large Language Models (LLMs) like ChatGPT are this but on steroids
- Their training data is massive and how they arrive at probabilities is very complex and uses a lot of Linear Algebra (matrices) and calculus

Limitations of Language Models

- Models are only "as good" as their training data:

Limitations of Language Models

- Models are only "as good" as their training data:
  - If the training data contains a lot of misinformation or bias, the model could say stupid and racist things (like Elon Musk's GrokAI)

Limitations of Language Models

- Models are only "as good" as their training data:
  - If the training data contains a lot of misinformation or bias, the model could say stupid and racist things (like Elon Musk's GrokAI)
  - They easily be manipulated for evil purposes by manipulating their training data + prompt engineering (e.g. "Respond to the prompt like a dumb edgy teenager would")
- LLMs are not alive. They do not know what "truth" is. They do not understand the meaning of the text they output.

Limitations of Language Models

- Models are only "as good" as their training data:
  - If the training data contains a lot of misinformation or bias, the model could say stupid and racist things (like Elon Musk's GrokAI)
  - They easily be manipulated for evil purposes by manipulating their training data + prompt engineering (e.g. "Respond to the prompt like a dumb edgy teenager would")
- LLMs are not alive. They do not know what "truth" is. They do not understand the meaning of the text they output.
- They only print text that looks like their training data

Limitations of Language Models

- Models are only "as good" as their training data:
  - If the training data contains a lot of misinformation or bias, the model could say stupid and racist things (like Elon Musk's GrokAI)
  - They easily be manipulated for evil purposes by manipulating their training data + prompt engineering (e.g. "Respond to the prompt like a dumb edgy teenager would")
- LLMs are not alive. They do not know what "truth" is. They do not understand the meaning of the text they output.
- They only print text that looks like their training data
- Doesn't make them useless! A basic webpage is going to have the same structure and a LLM can "predict" a good HTML page.

Limitations of Language Models

- Models are only "as good" as their training data:
  - If the training data contains a lot of misinformation or bias, the model could say stupid and racist things (like Elon Musk's GrokAI)
  - They easily be manipulated for evil purposes by manipulating their training data + prompt engineering (e.g. "Respond to the prompt like a dumb edgy teenager would")
- LLMs are not alive. They do not know what "truth" is. They do not understand the meaning of the text they output.
- They only print text that looks like their training data
- Doesn't make them useless! A basic webpage is going to have the same structure and a LLM can "predict" a good HTML page.
- You need to verify what they say, you need to know the information independently of LLMs for them to be useful.

Limitations of Language Models

- Models are only "as good" as their training data:
  - If the training data contains a lot of misinformation or bias, the model could say stupid and racist things (like Elon Musk's GrokAI)
  - They easily be manipulated for evil purposes by manipulating their training data + prompt engineering (e.g. "Respond to the prompt like a dumb edgy teenager would")
- LLMs are not alive. They do not know what "truth" is. They do not understand the meaning of the text they output.
- They only print text that looks like their training data
- Doesn't make them useless! A basic webpage is going to have the same structure and a LLM can "predict" a good HTML page.
- You need to verify what they say, you need to know the information independently of LLMs for them to be useful.
- Tech companies, investors, influencers, etc. are overselling LLMs as "AI" in order to make money