Deep Learning and Embeddings

(Remote) Lecture 18

COVID-19 Accommodations

- Classes, assignments, exams, etc. all remote through the rest of the semester
    - For this class, this will mean diligence in working remotely with teammates
    - PC5 (Cooperative Testing) has been moved back another week (now due 4/6)
    - PC6 (Sprint Review 3) will now be delivered as a YouTube video (now also 4/6)
    - PC7 (Final Presentations) will be a **scheduled telecon** with all of your team members, me, and one of the IAs (forthcoming)
        - Look at the Piazza post; you can schedule a 30 minute block on my calendar via the link there
        - Try to have most/all your team members present for that
- Grades now P/NRC with option to uncover letter grade

Recap

Natural Language Processing can be broken into several concepts:

- **Data**: Examples with labels
  - e.g., the tuple (“I want a burger” -> “order_food”) is an intent classification data point
- **Model**: A method for quantifying data
  - Features and Weights can be used
  - Contrived Example: “I want a burger”
    - “want” and “burger” => +2 for *order_food* intent
    - “burger” => +1 ; “want” => -1 for *get_nutrition* intent
  - Metrics like **tf-idf** or **n-gram frequency** can be useful for modeling
- **Inference**: deciding based on output from a model
  - We take concrete action based on numerical outputs
  - e.g., we *infer* the *intent* based on the model’s highest output value (the +2 above)
- **Learning**: revising model based on new data
  - How do you decide rules for the model?

Recap: Applying to Conversational AI

- **Intent Classification**
  - Data: tuples of (utterance, intent class)
  - Model: clustering, SVM, rules;
  - Inference: mapping from model output to intent class label

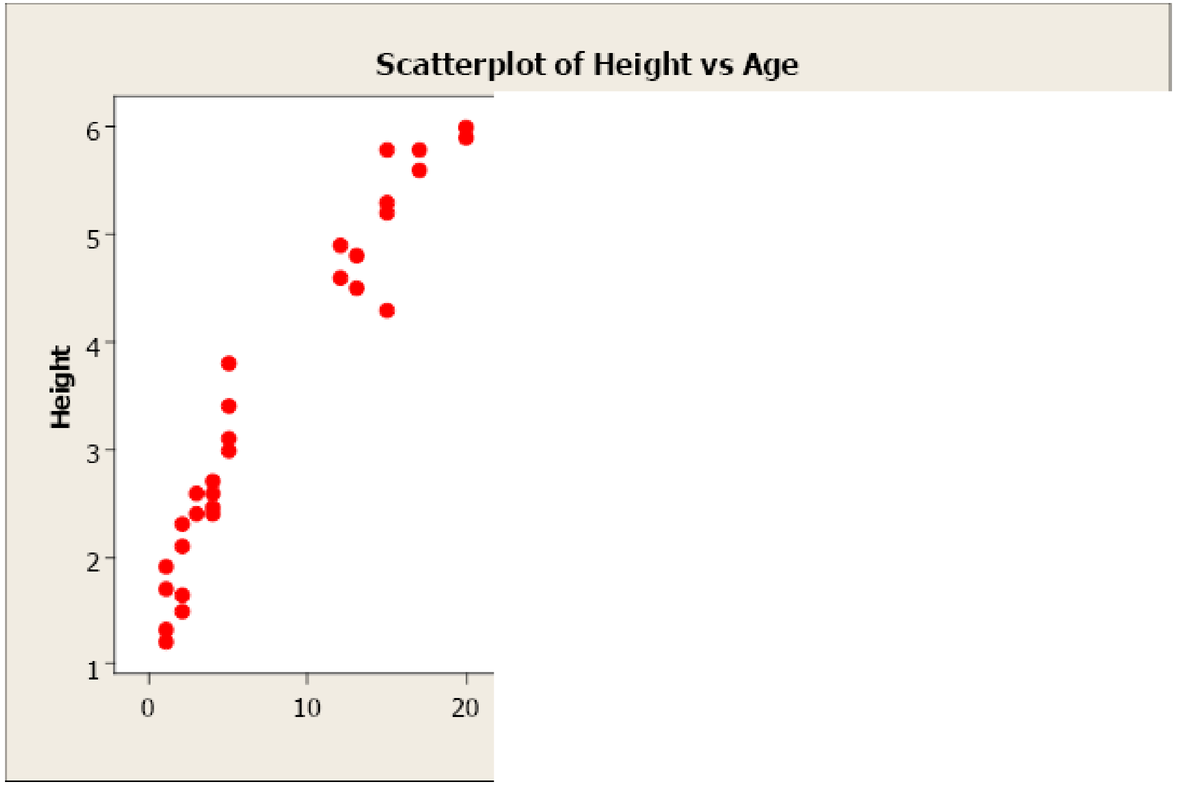
- **Slot Extraction**
  - Data: tuples of (token position, slot label)
  - Model: n-grams, **RNN**
  - Inference: RNN output mapped back to a vocabulary

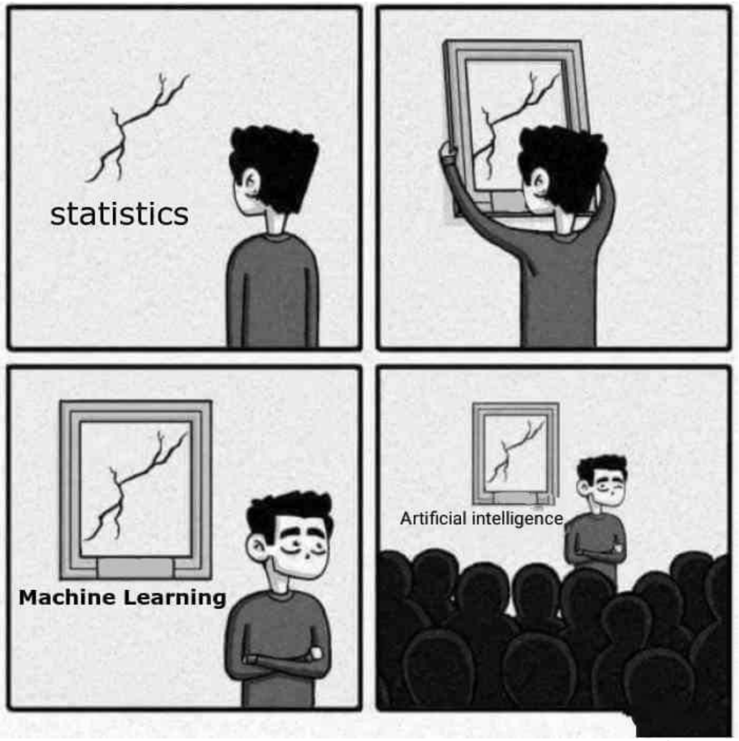
One Slide Summary: Deep Learning and Embeddings

- **Machine Learning** is driven by applied **statistics**
  - Simple linear models are more interpretable (e.g., best-fit line)
  - More complex models yield better accuracy (trading off interpretability)
- **Deep Learning** is used in the NLP space to accurately represent language and classify intents and slots
  - Deep learning allows black-boxing of inputs to eliminate the need to derive costly features or rules
  - In particular, **Recurrent Neural Networks** and derivatives are state-of-the-art for NLU tasks
- **Embeddings** are numerical representations of NLU elements
  - Expressed as **fixed-dimensional vectors**
  - We say that we **embed** a token, sentence, or utterance into a **vector space** called the embedding space

Machine Learning

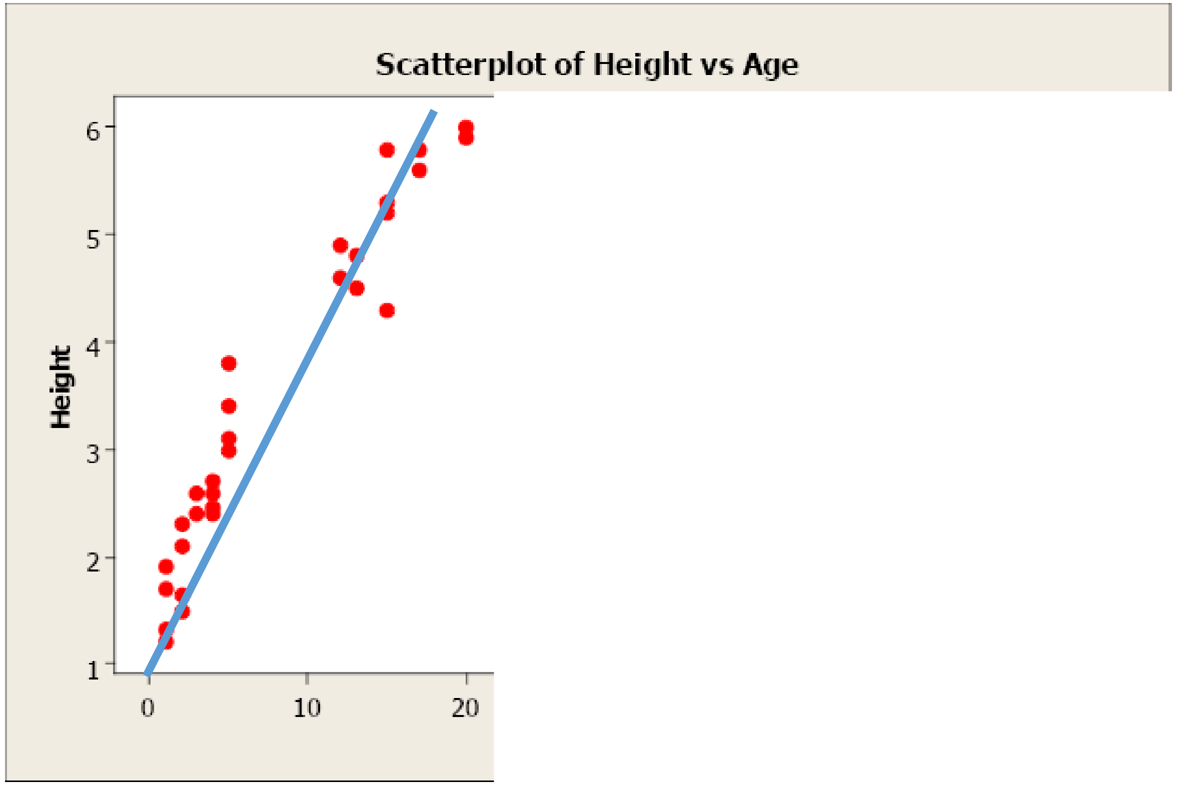
- **AI** is an application of **Machine Learning**
- **ML** is an application of statistics to **make predictions from existing data**

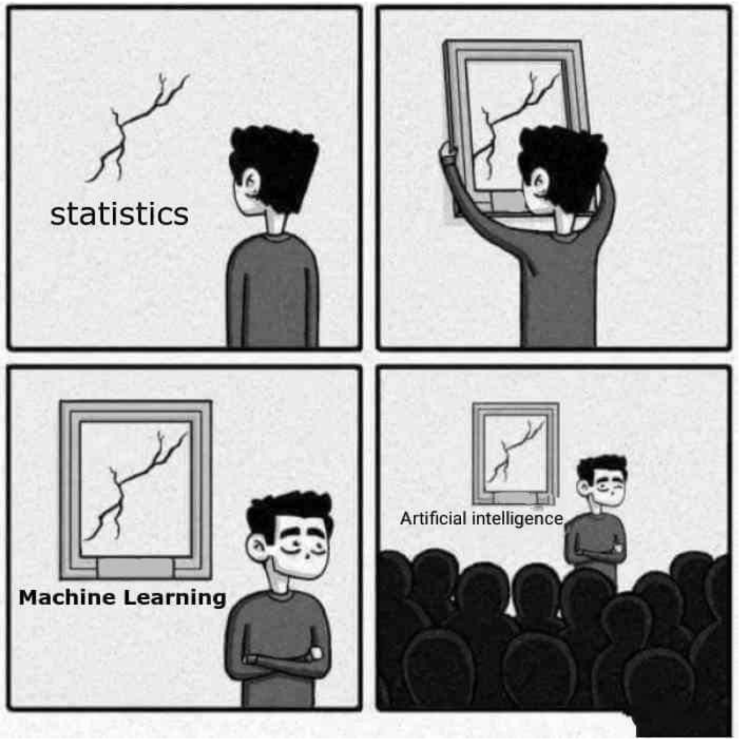




Machine Learning

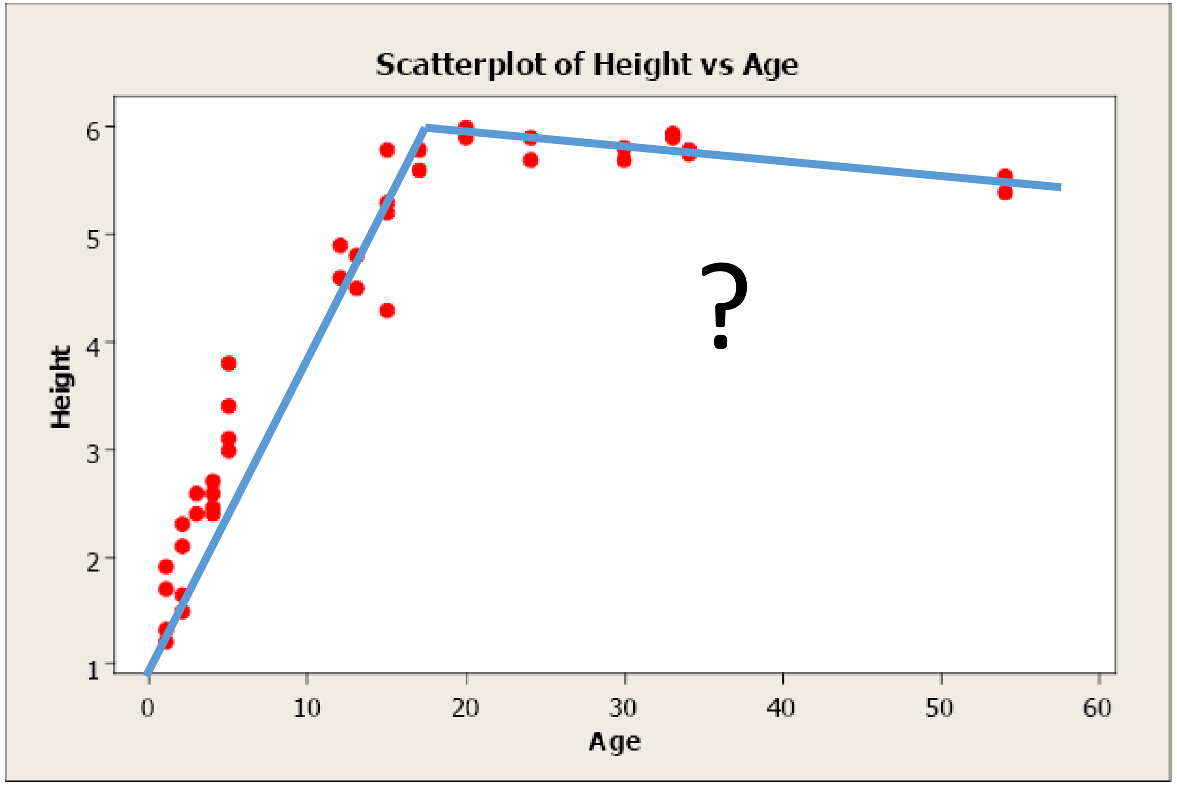
- **AI** is an application of **Machine Learning**
- **ML** is an application of statistics to **make predictions from existing data**

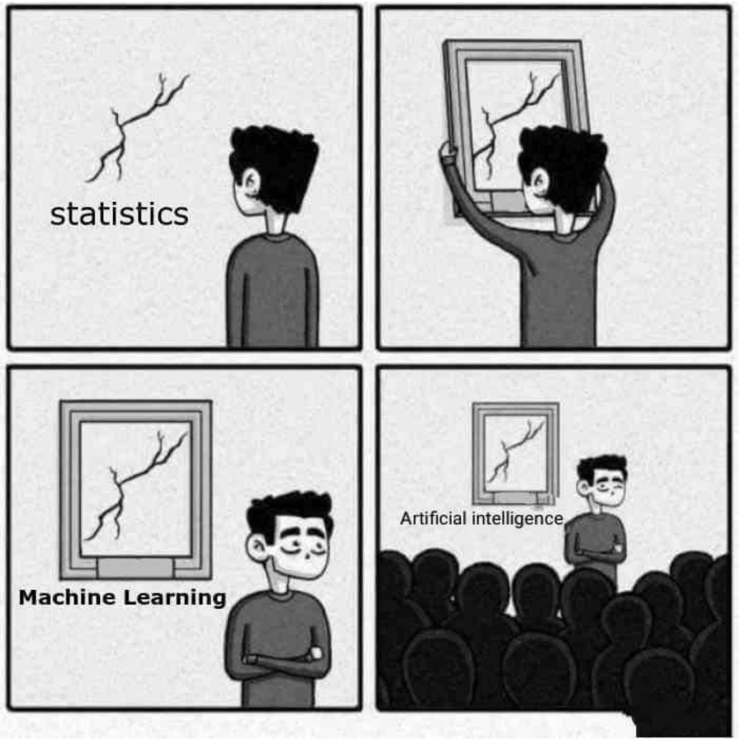




Machine Learning

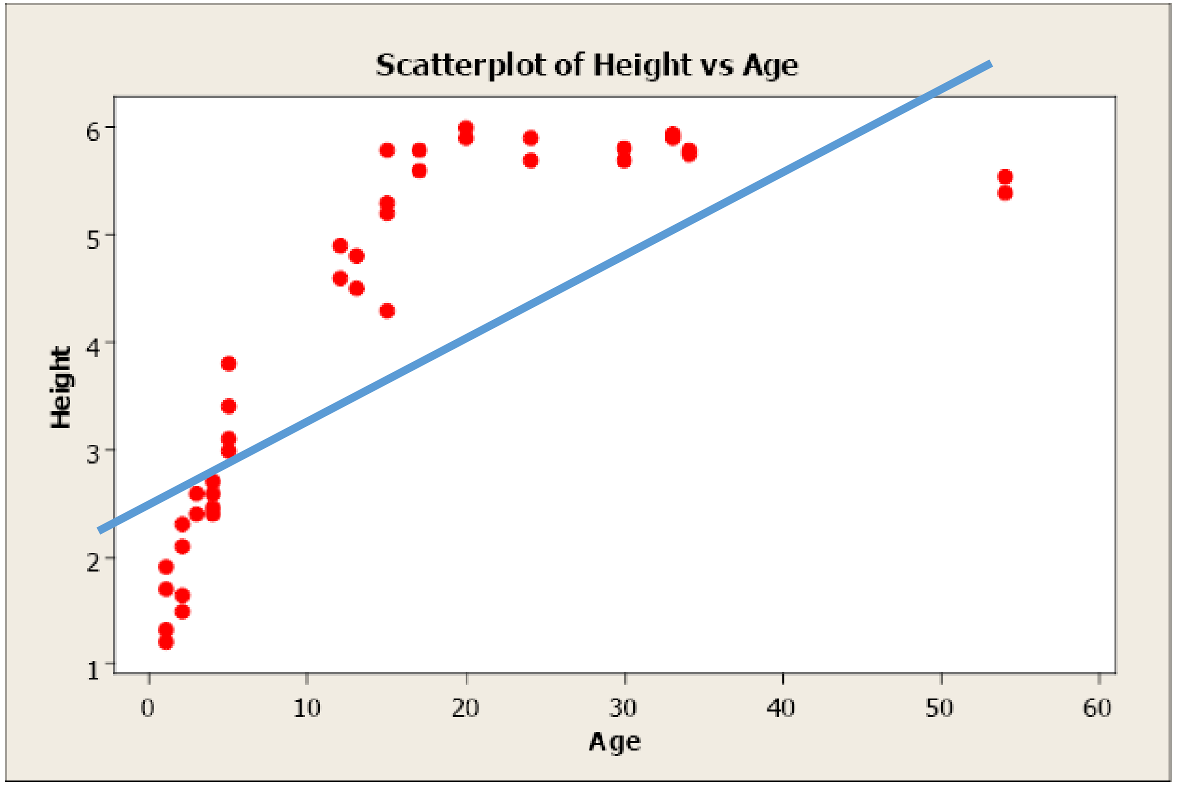
- **AI** is an application of **Machine Learning**
- **ML** is an application of statistics to **make predictions from existing data**





Machine Learning

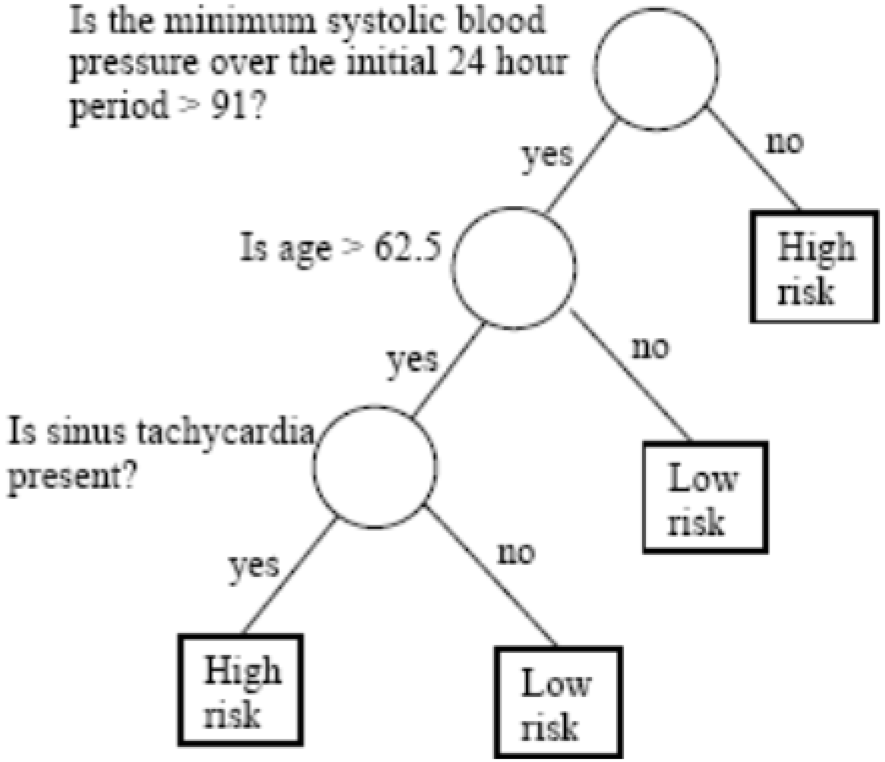
- Must manually
  - Select features (e.g., age)
  - Hypothesize relationship (e.g., linear, piecewise, quadratic…)
- **Time consuming, but interpretable**
- Relies on domain knowledge



Machine Learning

- **Decision Trees** can be used to classify inputs (e.g., tall vs. not tall; high risk vs. low risk)

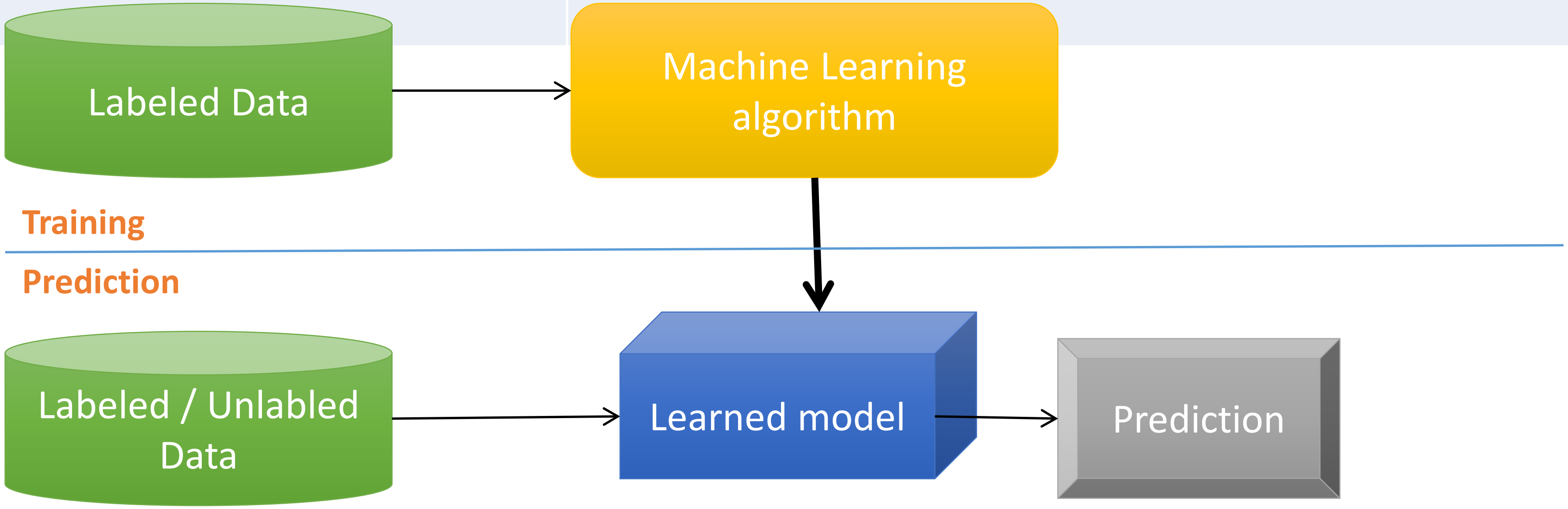
- **Example:** cardiovascular risk
  - Perhaps doctors have access to tons of old medical histories.
  - Might notice clusters in data (i.e., *domain expertise*):
    - Minimum systolic <= 90 -> high risk of death
    - Old with sinus tachycardia rhythm -> high risk


Is the minimum systolic blood pressure over the initial 24 hour period > 91?
yes
no
High risk
Is age > 62.5
yes
no
Low risk
Is sinus tachycardia present?
yes
no
High risk
Low risk

Machine Learning

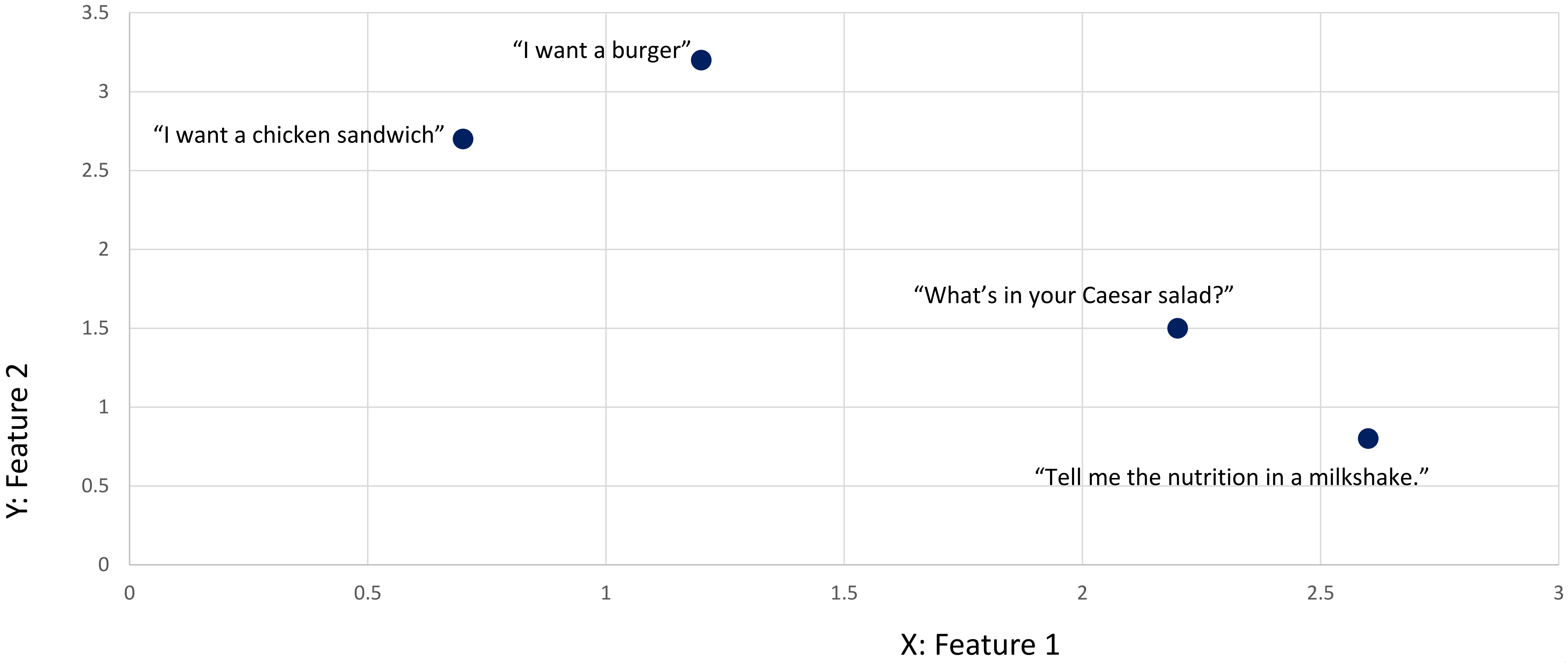
- We use ML to **teach** software to **make predictions**
- Software **learns** from existing data

Supervised learning	Labeled data (e.g., tall vs. short) (20 y.o., 6ft, tall) (20 y.o., 5ft, short)
Unsupervised learning	Unlabeled data (e.g., just points of data) (20 y.o., 6ft) (20 y.o., 5ft)



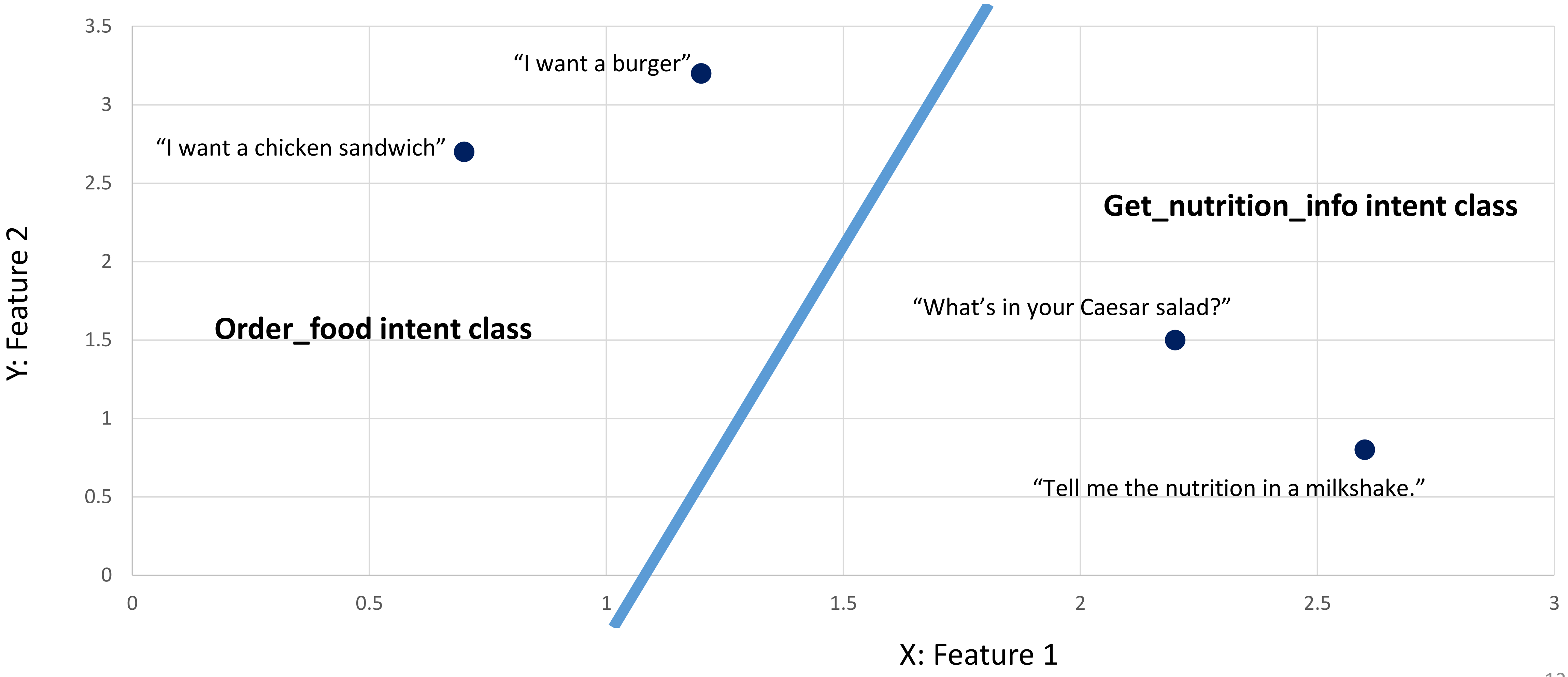
Machine Learning in an NLU Context

A **Model** allows us to quantify utterances. Depending on the specific model, we can visualize data



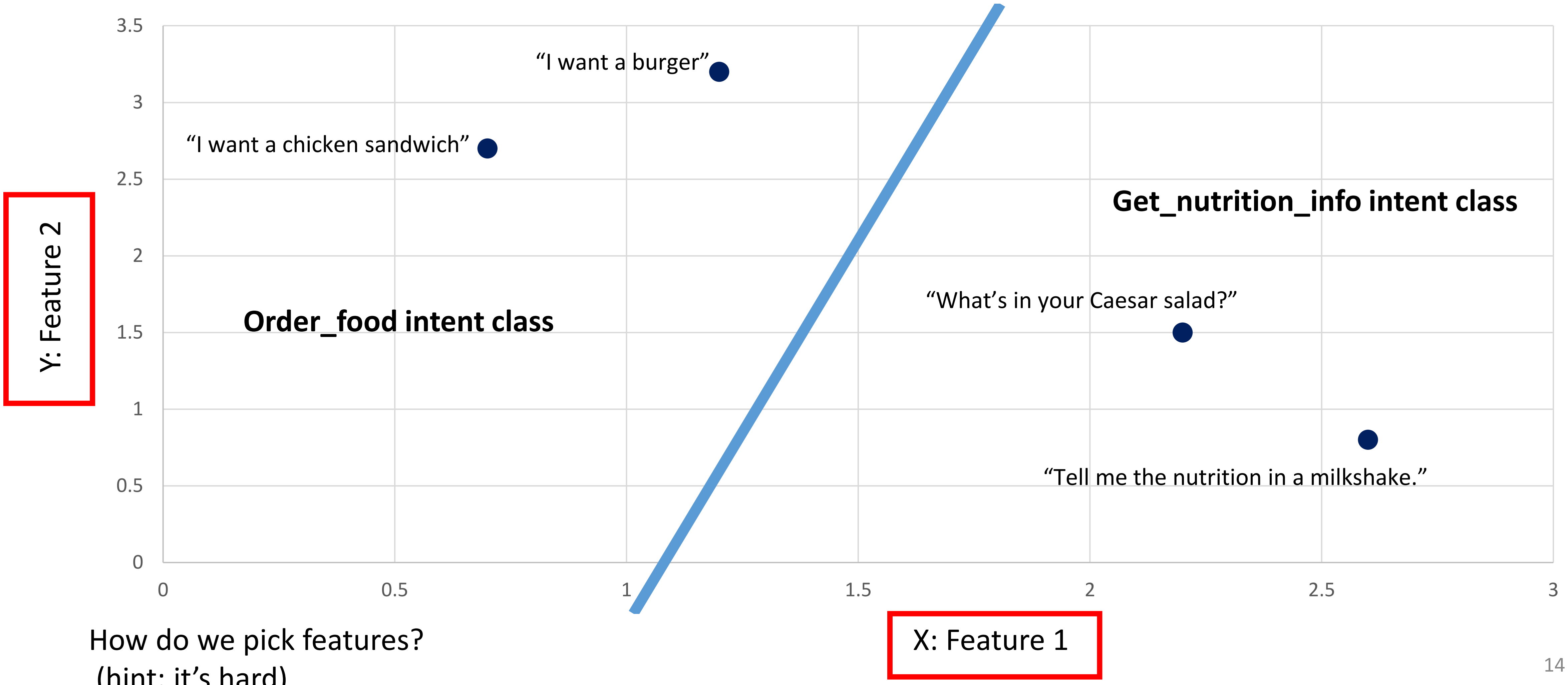
Machine Learning in an NLU Context

A **Model** allows us to quantify utterances. Depending on the specific model, we can visualize data



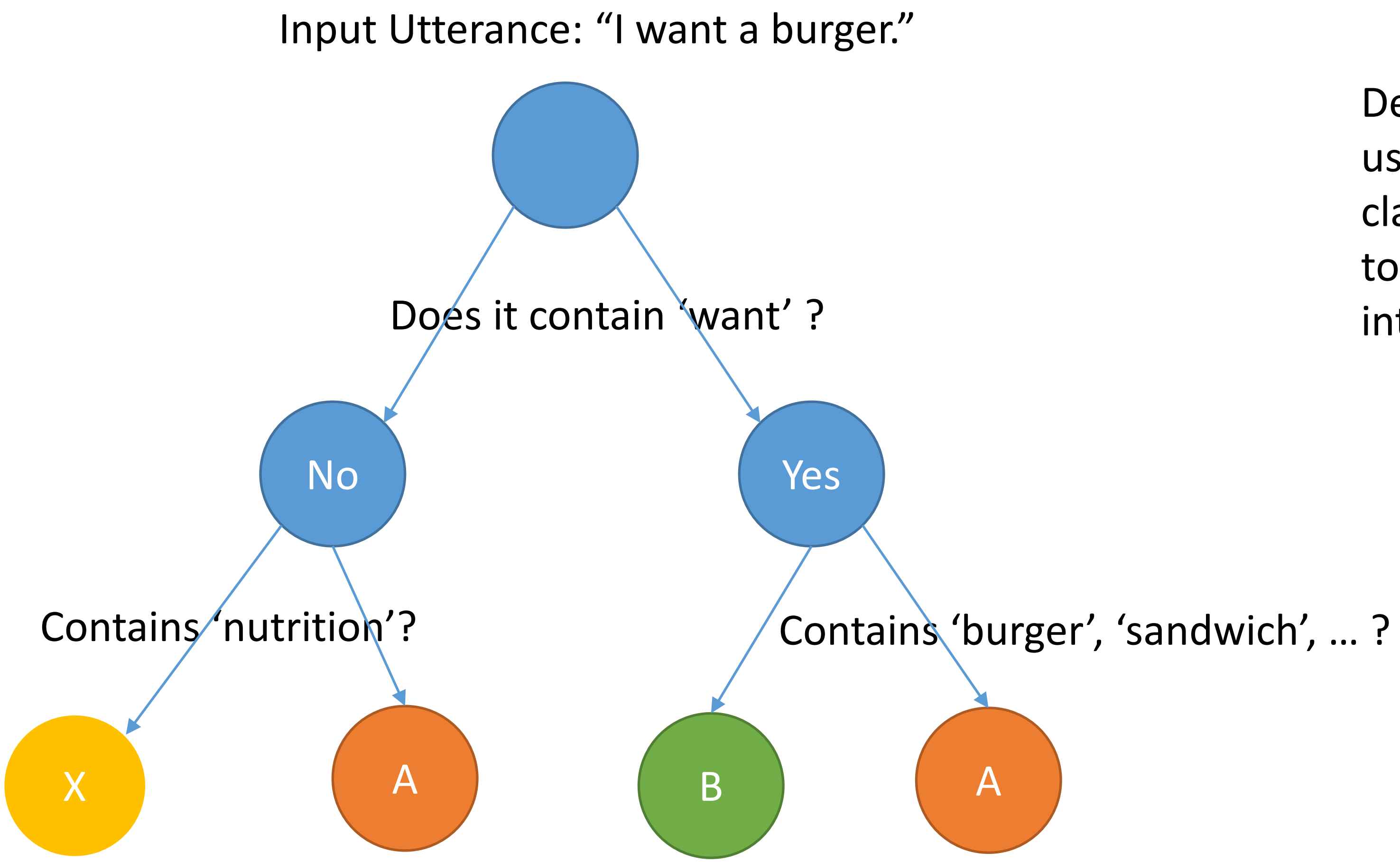
Machine Learning in an NLU Context

A **Model** allows us to quantify utterances. Depending on the specific model, we can visualize data



How do we pick features?
 (hint: it's hard)

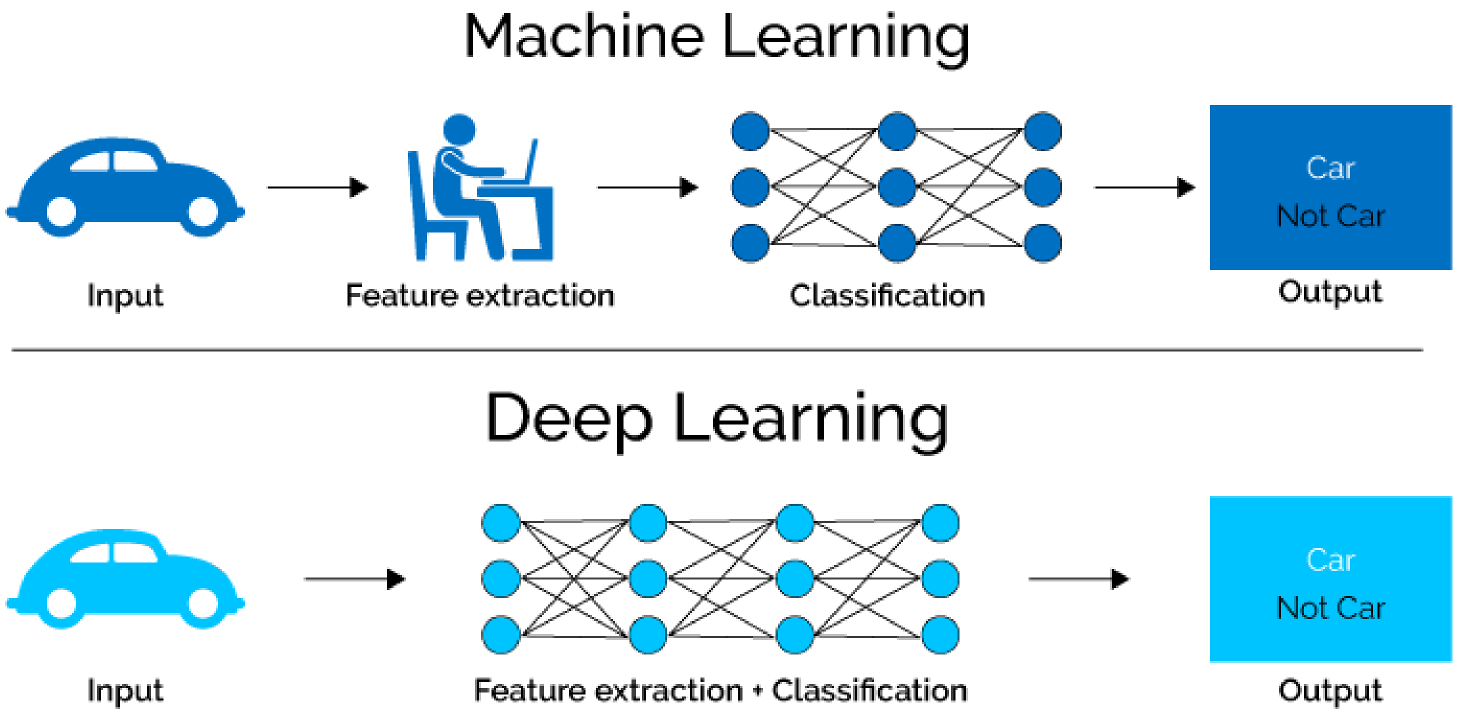
Machine Learning in an NLU Context


Input Utterance: “I want a burger.”
Does it contain ‘want’ ?
No
Yes
Contains ‘nutrition’?
Contains ‘burger’, ‘sandwich’, … ?
X
A
B
A

Decisions Trees can be used for intent classification; but ordinarily too many complex interactions exist.

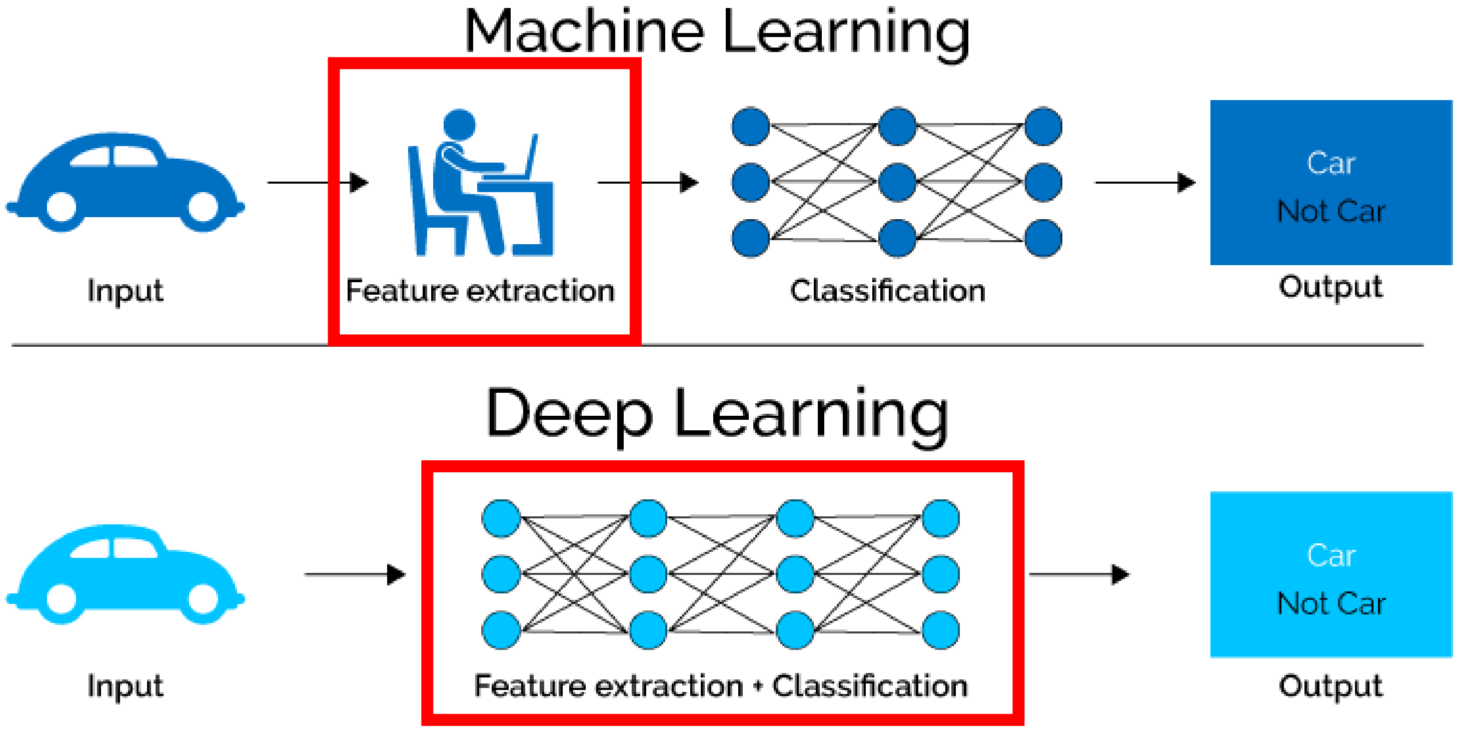
Deep Learning Crash Course

- **Deep Learning** is a catch-all phrase that refers to **Neural Networks** that have multiple layers (c.f. deep pipeline from architecture)



Deep Learning Crash Course

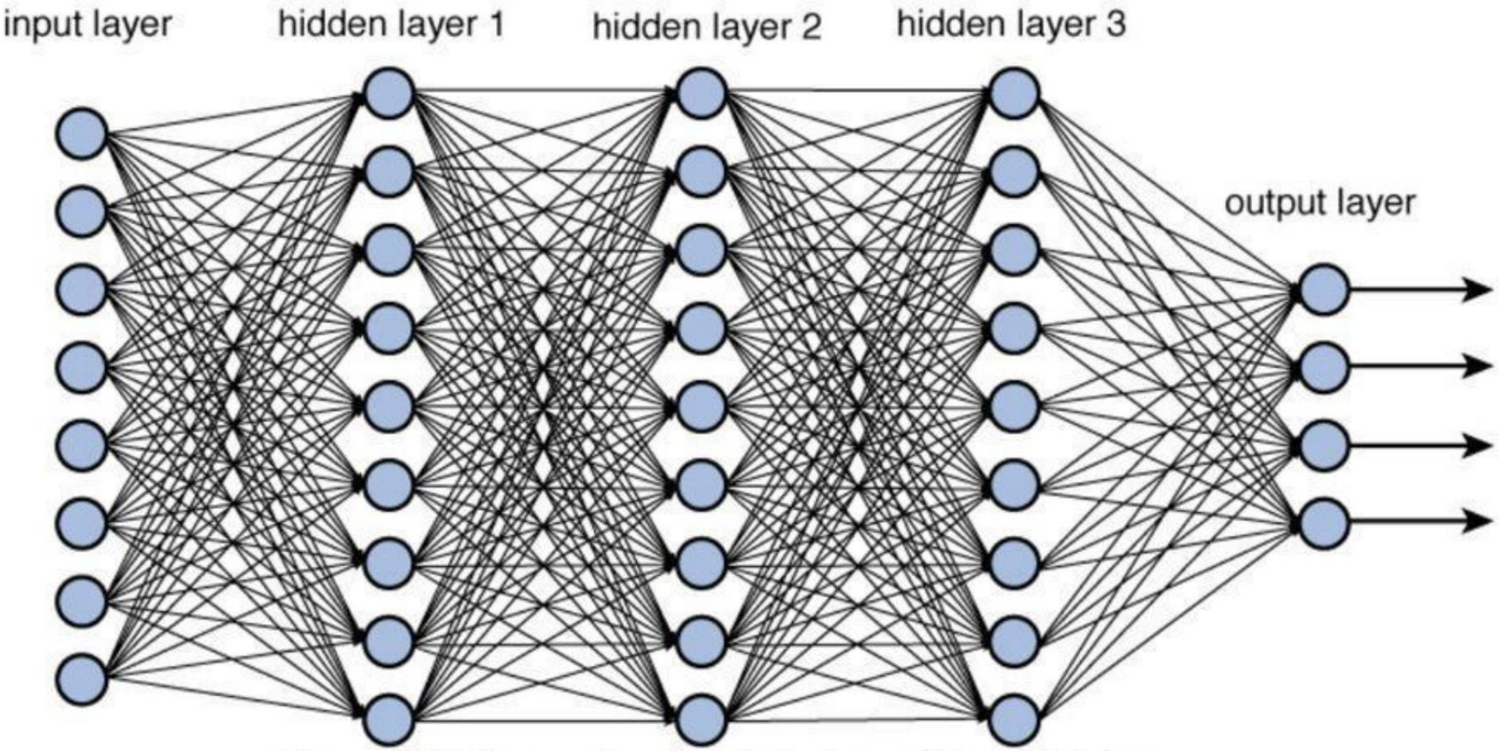
- **Deep Learning** is a catch-all phrase that refers to **Neural Networks** that have multiple layers (c.f. deep pipeline from architecture)



“depth” = more layers

Neural Network

- A **Neural Network** is a structure that feeds **data** through **layers** of simple mathematical operations **(neurons),** producing some set of **numerical outputs** that have some useful interpretation

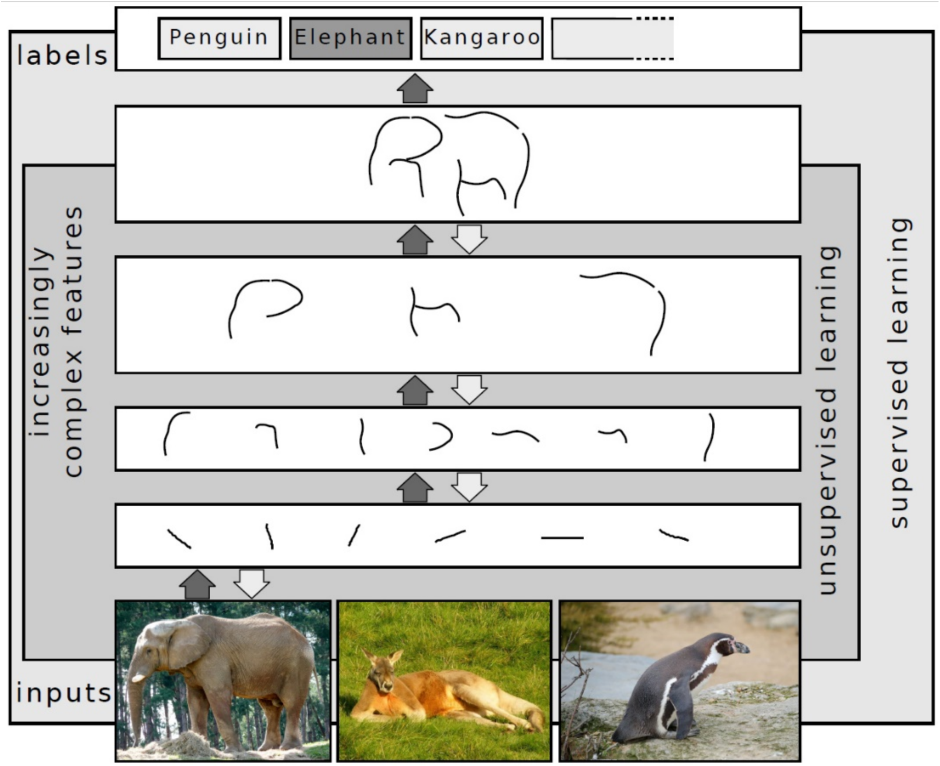


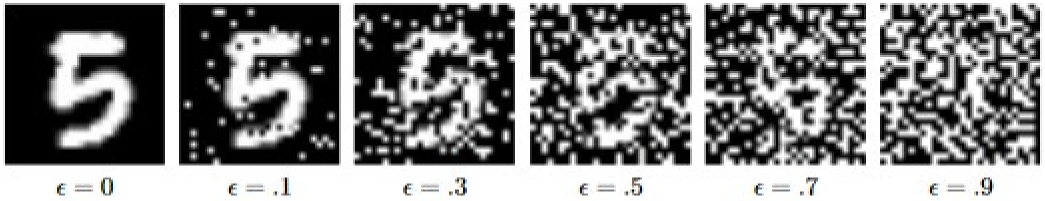
“Things that are lies”

Neural Network

- We use Deep Neural Networks (DNNs) to perform classification of intents, slot mapping, and slot-value pairing
  - DNNs can **learn from** (or “notice”) patterns in data that are not immediately obvious to human domain knowledge experts

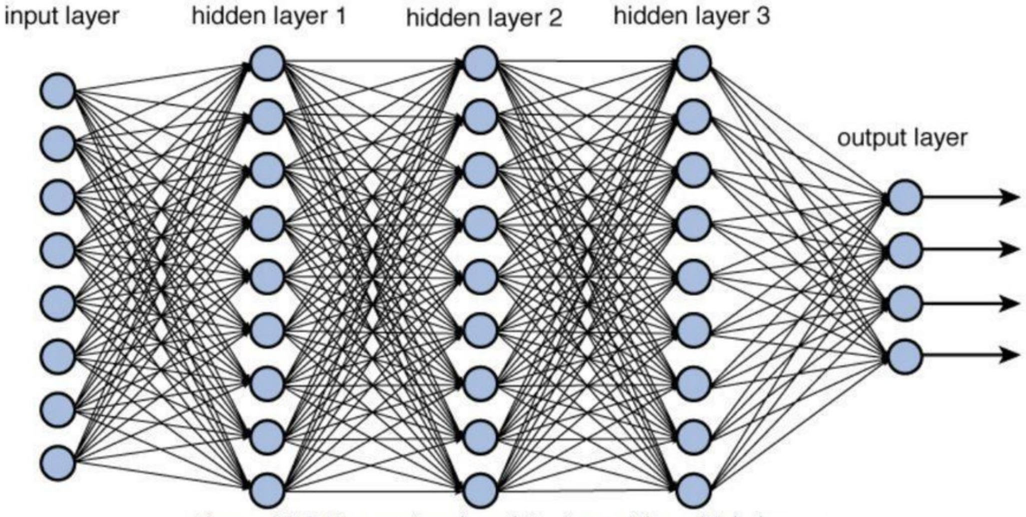
- DNNs benefit from data
  - As long as features are represented, DNNs can learn which ones are important

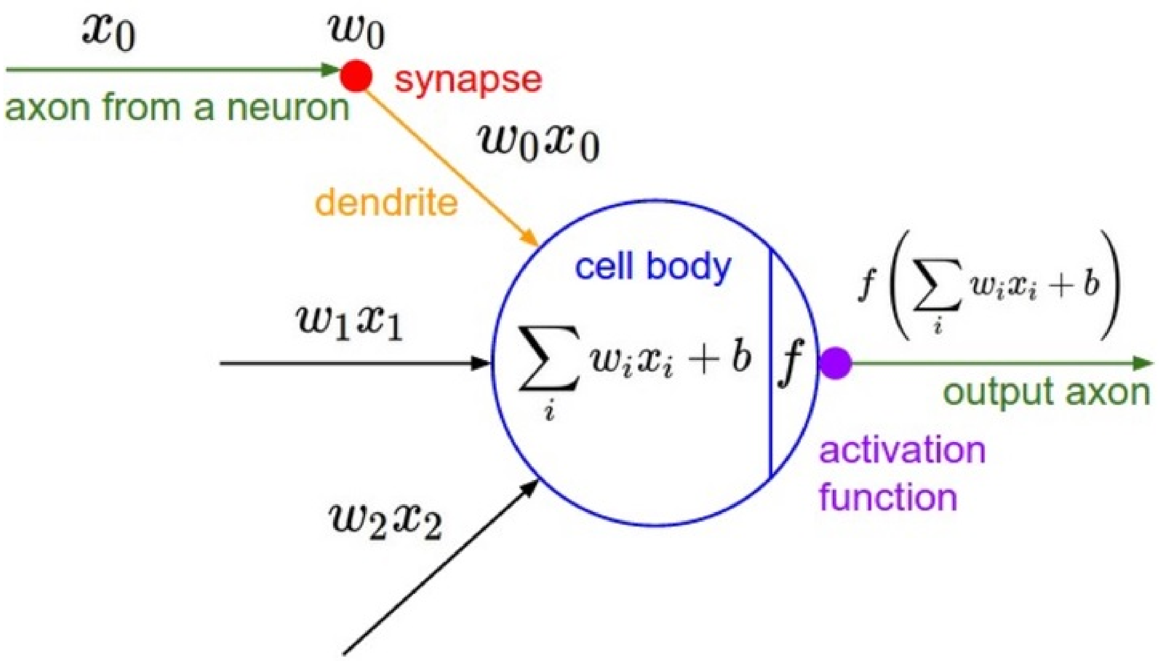




Deeper in NNs

- Each cell in a NN is a simple combination of floating-point inputs

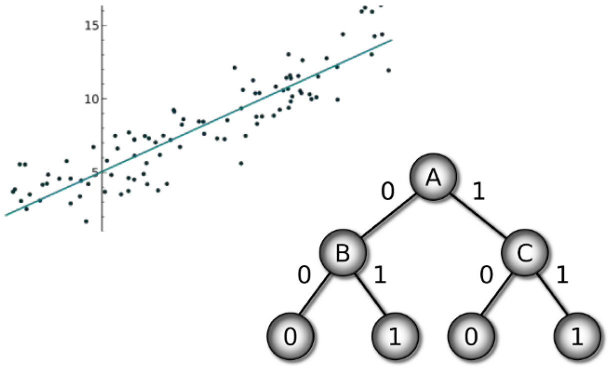




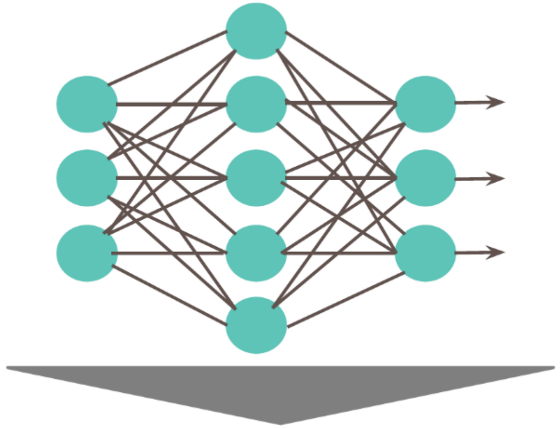
Tradeoffs in ML

- Deep neural networks **perform better,** but are **less explainable**
- Decision trees, linear models are all far easier to **explain**, but lack **expressive power**

Lin. regression / decision trees:
Decision mechanism can be easily explained



Neural networks:
Complex systems that are hard to understand!

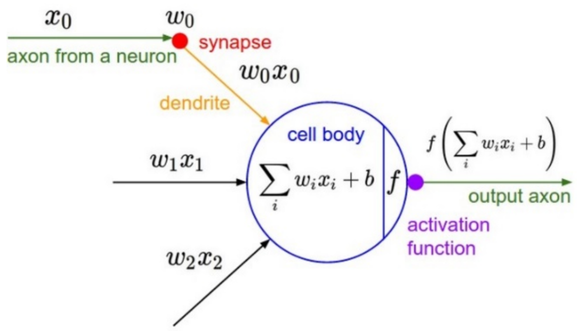


Often **100m+** parameters....

tuning your hyperparameters by hand
doing random sampling in hyperparameter space
using bayesian optimization to find optimal hyperparameters
optimizing the random seed

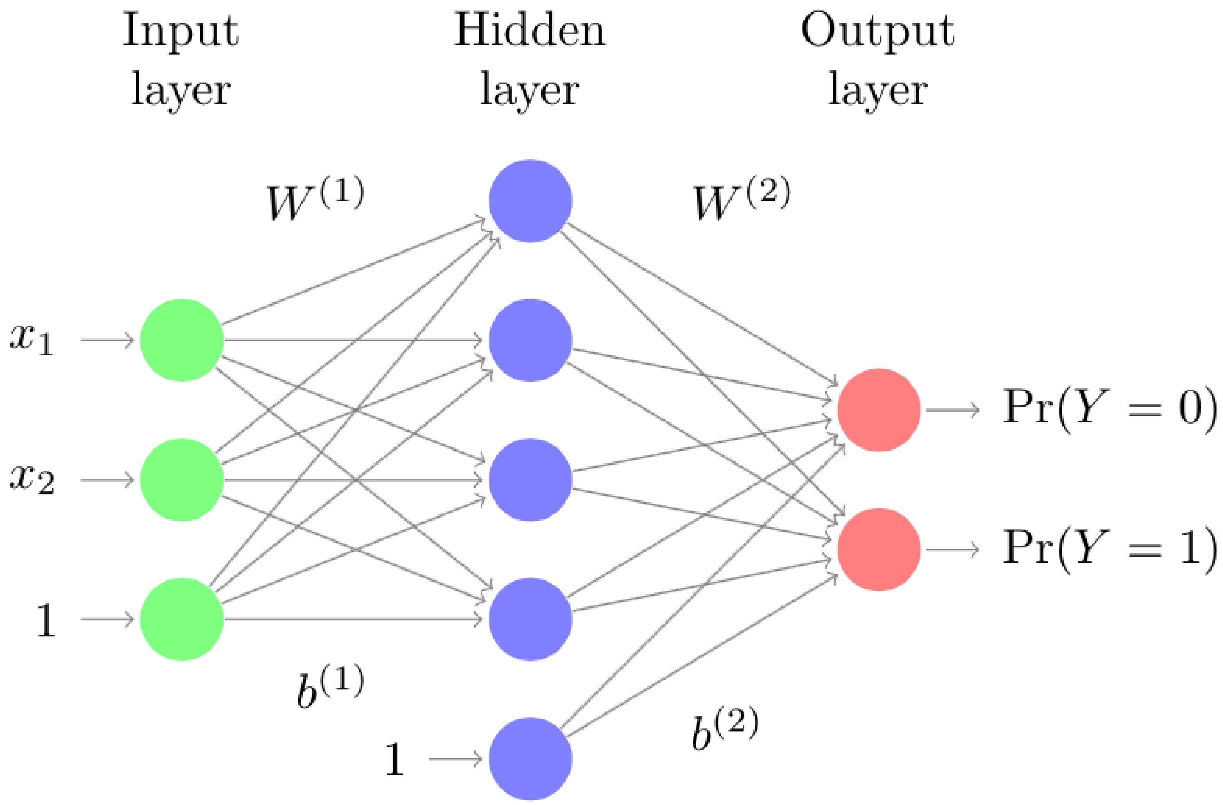
Learning in NNs

- NNs can be characterized by the **weights** of connections from one layer to the next
- We use a **loss function** that captures the difference between known **labels** (i.e., what the output **should be**) and the output produced by the untrained NN
- We **adjust** the weights of the NN based on the **loss function**
  - Over time, the NN learns to capture patterns in the training data



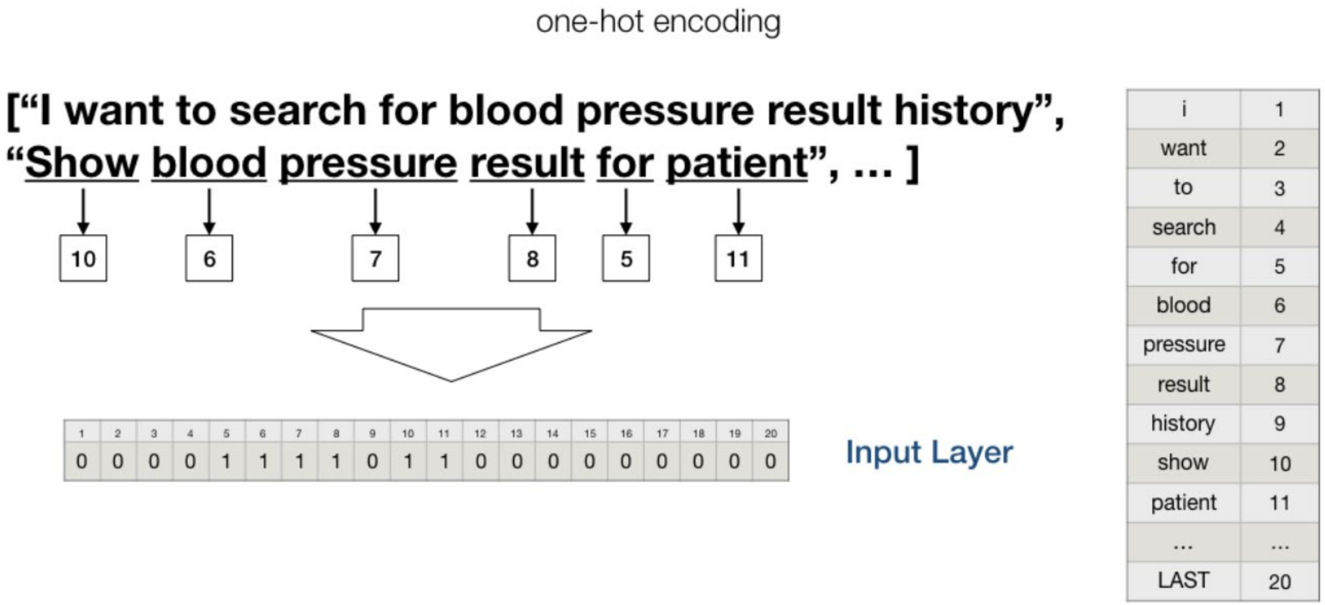
DNNs for Classification

- DNN output layer can be interpreted as some class (e.g., “elephant” vs. “cat” or “noun” vs. “verb”)
- **Key idea:** with enough data, the NN can learn weights that can make future classifications
- Instead of developing complex rules or criteria for a **model**, the weights fall out of the learning process automatically!



DNNs for NLP

- DNNs are continuous mathematical structures
    - Lots of floating point operations
- Natural language is made up of letters
    - We need to **represent** natural language in some **vector space**



one-hot encoding

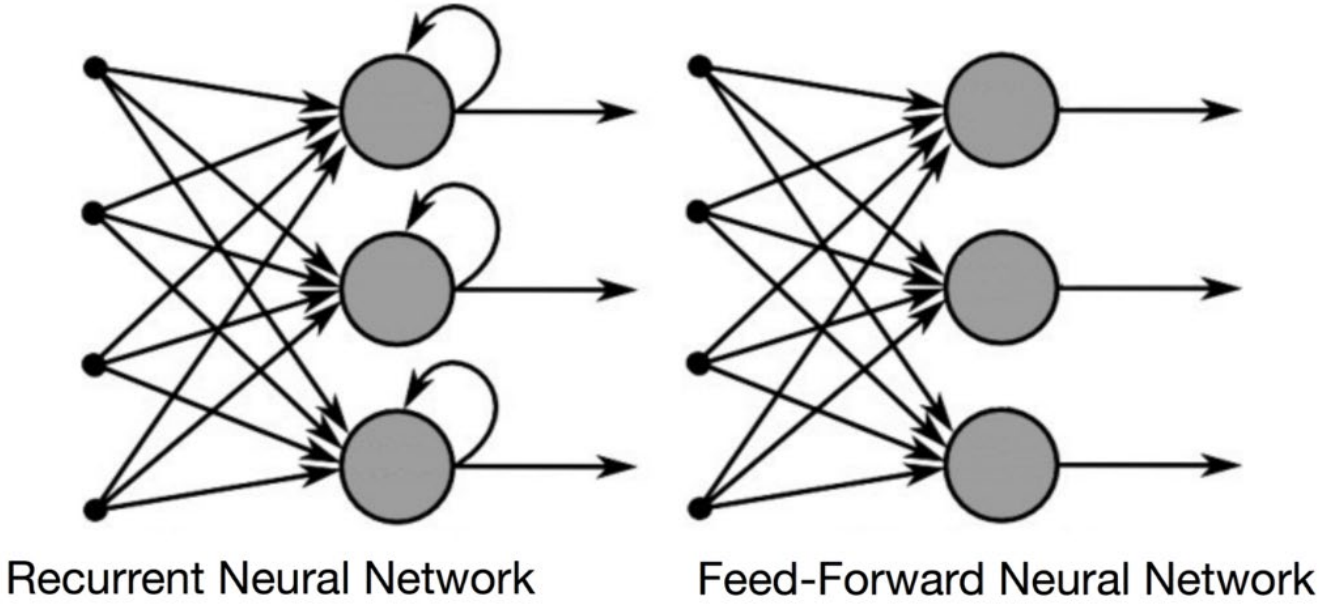
[“I want to search for blood pressure result history”, “Show blood pressure result for patient”, …]

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
0	0	0	0	1	1	1	1	0	1	1	0	0	0	0	0	0	0	0	0

i	1
want	2
to	3
search	4
for	5
blood	6
pressure	7
result	8
history	9
show	10
patient	11
…	…
LAST	20

Recurrent Neural Networks

- RNNs are a type of NN that allow feeding information within a layer
  - (as opposed to feed-forward-only)



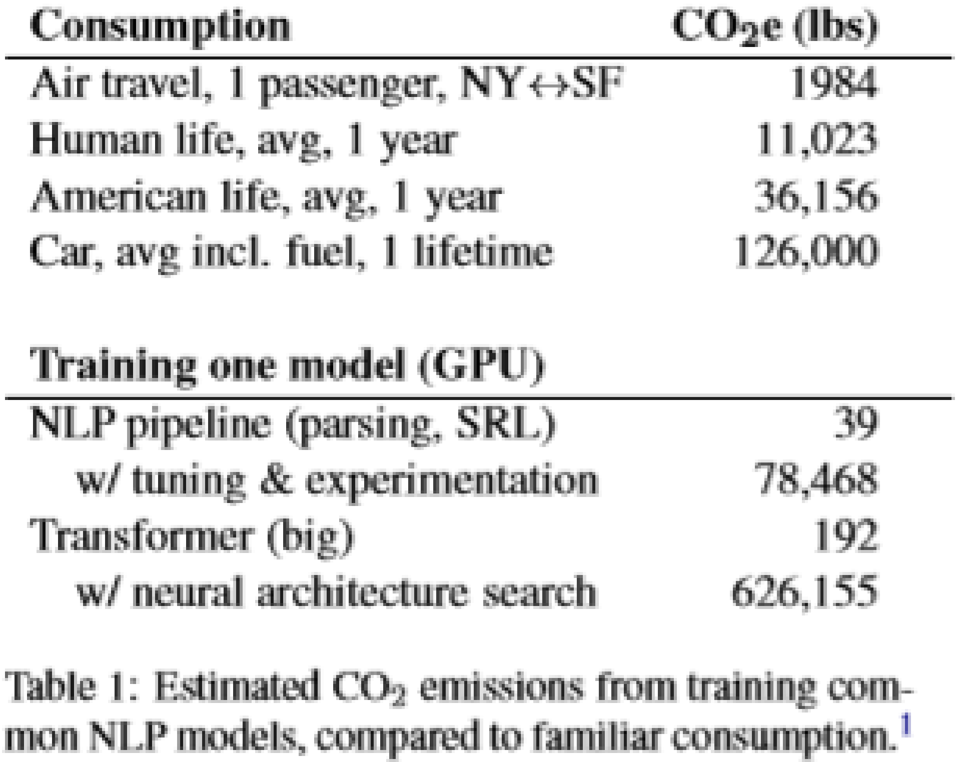
  - Beneficial for **sequential data** (like sequences of tokens)

Why DNNs for NLP?

Model	Slot F1 Score	Intent Accuracy
Bi-model with decoder	96.89	98.99
Stack-Propagation + BERT	96.10	97.50
Stack-Propagation	95.90	96.90
Attention Encoder-Decoder NN	95.87	98.43
SF-ID (BLSTM) network	95.80	97.76
Capsule-NLU	95.20	95.00
Joint GRU model(W)	95.49	98.10
Slot-Gated BLSTM with Attension	95.20	94.10
Joint model with recurrent slot label context	94.64	98.40
Recursive NN	93.96	95.40
Encoder-labeler Deep LSTM	95.66	NA
RNN with Label Sampling	94.89	NA
Hybrid RNN	95.06	NA
RNN-EM	95.25	NA
CNN-CRF	94.35	NA

Neural Networks, Deep Learning, RNNs

- **Neural Networks** underly the majority of **modern AI** techniques
- NNs allow **black-boxing** a lot of **domain-expertise** required in other ML techniques
- **DNNs** are merely **bigger NNs** that have lots of intermediate layers
  - Requirement: need **a lot** of data
- **RNNs** are a type of NN that have a particular property: **loops in the graph**
  - NB: loops imply **statefulness**



Consumption	**CO$_2$e (lbs)**
Air travel, 1 passenger, NY↔SF	1984
Human life, avg, 1 year	11,023
American life, avg, 1 year	36,156
Car, avg incl. fuel, 1 lifetime	126,000

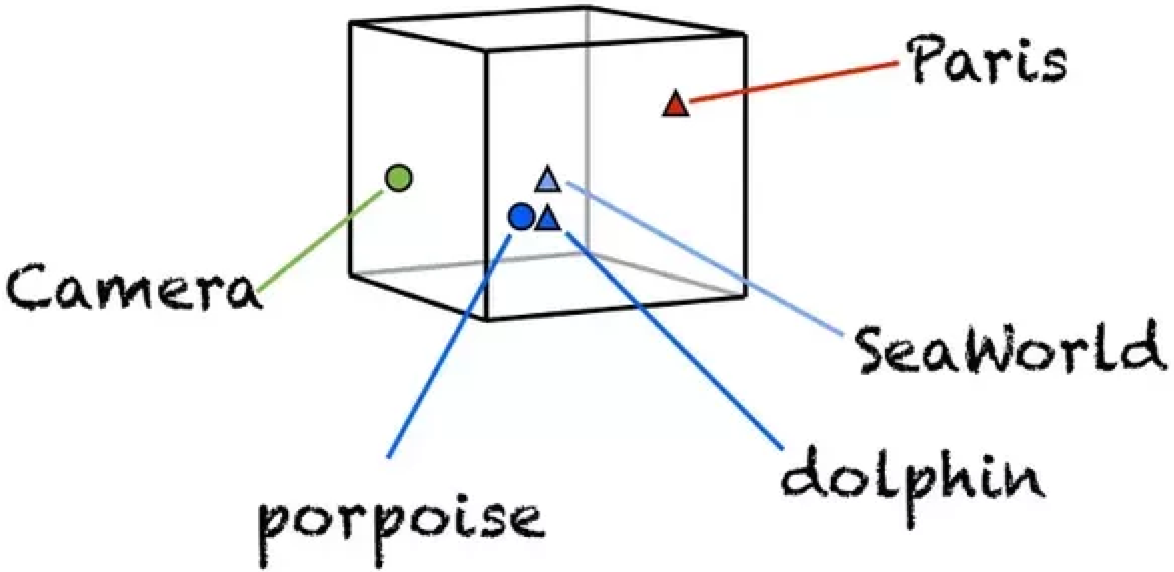
Training one model (GPU)	
NLP pipeline (parsing, SRL)	39
w/ tuning & experimentation	78,468
Transformer (big)	192
w/ neural architecture search	626,155

Table 1: Estimated CO$_2$ emissions from training common NLP models, compared to familiar consumption.[1]

[1] Sources: (1) Air travel and per-capita consumption: https://bit.ly/2Hw0xWc; (2) car lifetime: https://bit.ly/2Qbr0w1.

Embeddings

- A **word embedding** is a way of mapping words to vectors
  - word2vec/GLoVE: unsupervised methods based on corpus statistics
  - Built-in embedding layers: tensorflow/keras support “Embedding layers”



Desired Properties of Embeddings

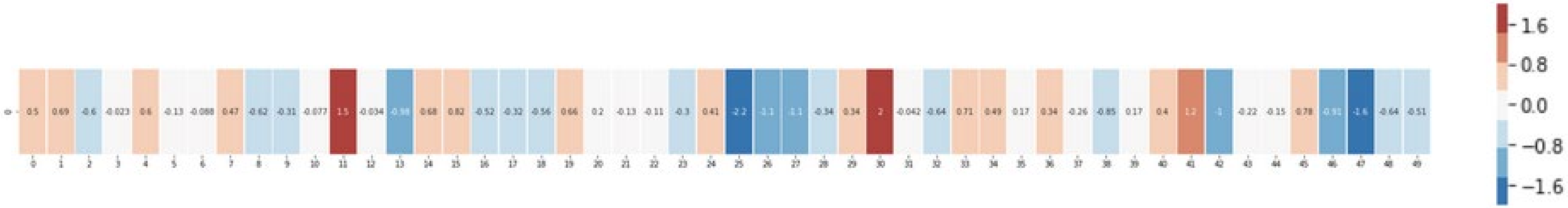
- **Semantic Relationships Represented**
  - Related words should be “Close” in a Euclidean sense
    - “cloud” – “sky” < “cloud” – “steak”
  - Unrelated words should be far away
  - Arithmetic should be possible
    - “cloud” + “sky” – “sun” might yield something near “rain”

- **Compact Representation**
  - We want to do quick math to compute word relationships
  - We need a representation suitable for DNNs

- **Mappable**
  - We need to move to and from word and embedding space quickly
  - A DNN layer may be an embedding… how do we turn it into a word?

Visualizing Word Embeddings

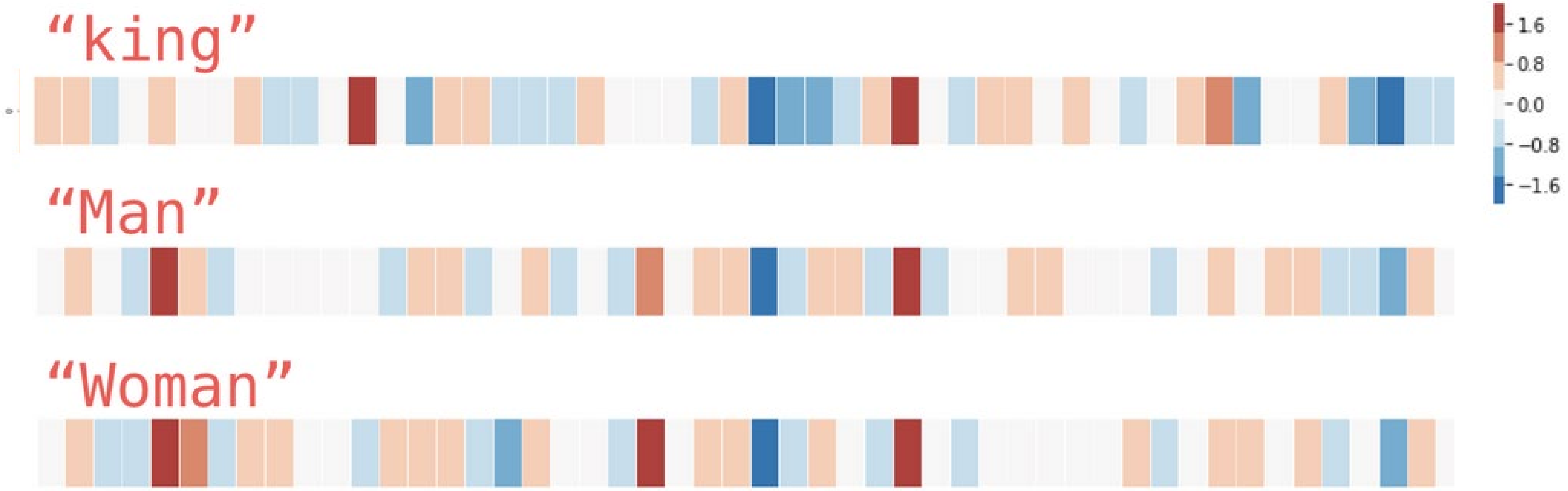
- Let’s embed the word “King”



```
[ 0.50451 , 0.68607 , -0.59517 , -0.022801,  0.60046  , -0.13498 , -0.08813 , 0.47377 , -0.61798 , -0.31012 , -0.076666,  1.493   , -0.034189, -0.98173 ,
  0.68229 , 0.81722 , -0.51874 , -0.31503 , -0.55809 ,   0.66421  , 0.1961  ,-0.13495 , -0.11476 , -0.30344 ,  0.41177 , -2.223   , -1.0756   , -1.0783 ,
 -0.34354 , 0.33505 ,  1.9927  , -0.04234 , -0.64319 ,   0.71125  , 0.49159 , 0.16754 ,  0.34344 , -0.25663 , -0.8523  ,  0.1661 , 0.40102  , 1.1685 ,
 -1.0137  ,-0.21585 , -0.15155 ,  0.78321 , -0.91241 ,  -1.6106 ,  -0.64426 ,-0.51042 ]
```

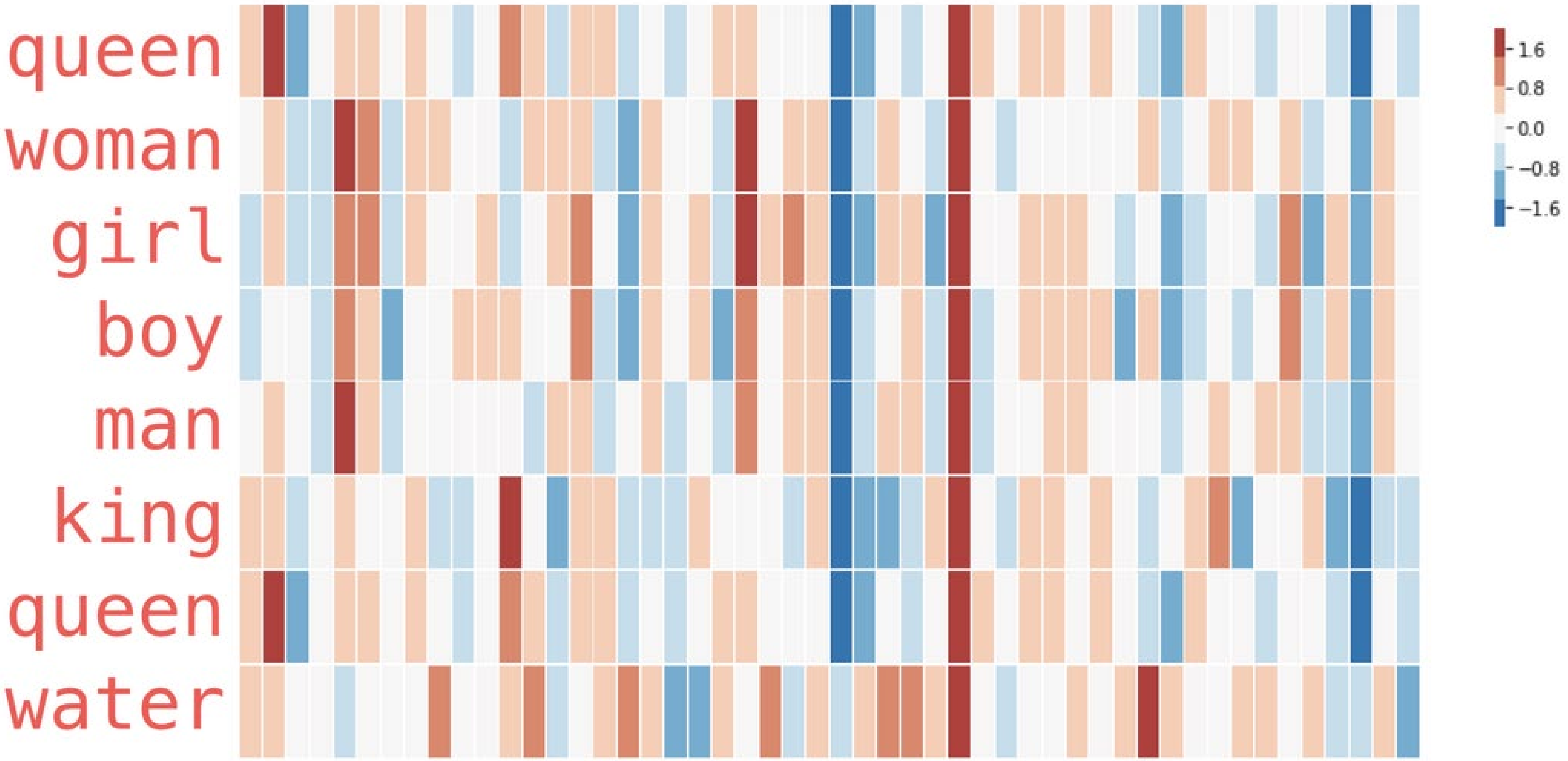
Visualizing Word Embeddings

- Let’s embed the word “King”

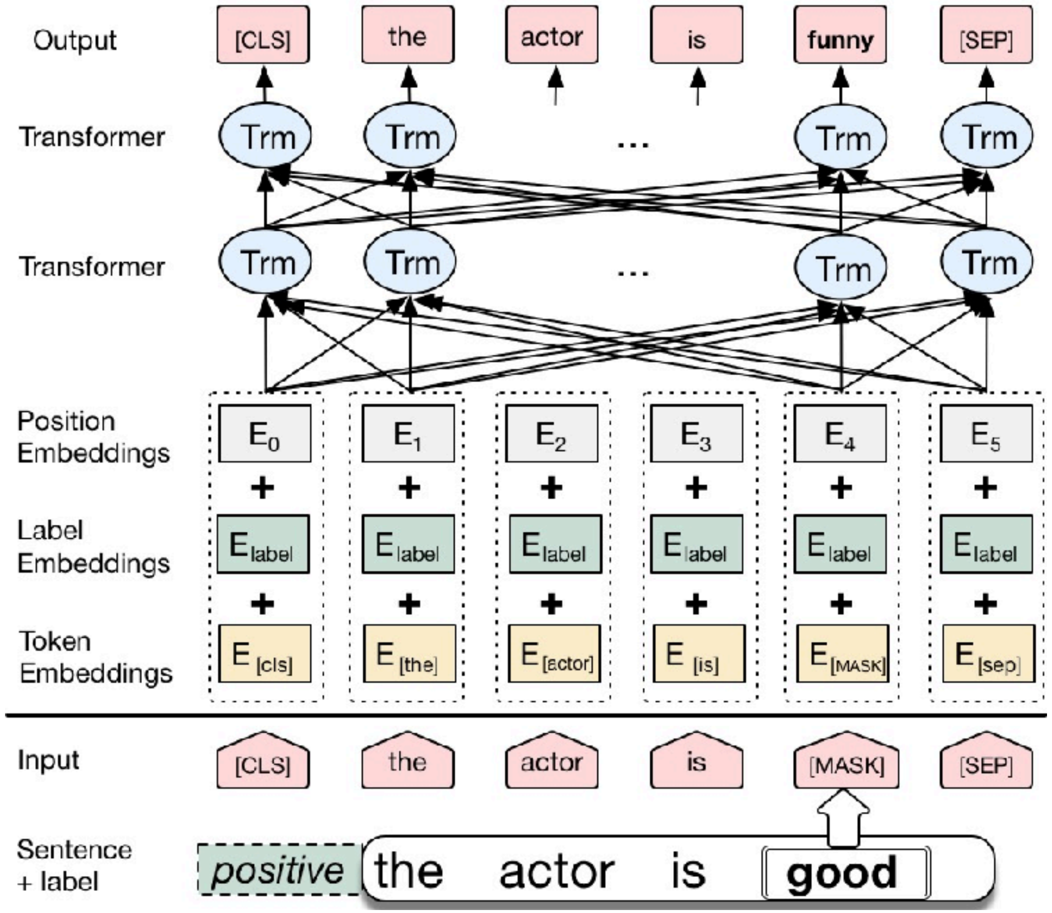


Visualizing Word Embeddings

- Let’s embed the word “King”



BERT: State-of-the-art embeddings



DNNs for NLP

- We can use DNNs for
  - **Classification**: Fixed number of intents (once you build your state graph)
    - Embed utterances; model can learn words (and neighbors in the embedding space!) that distinguish intents
    - The state graph just makes this part simpler (you only need to consider between the intent classes of child nodes in any given state)
  - **Slot Extraction**: Train by labeling portions of utterance
    - Yo fam get me a burger.
    - O B:person O B:person O B:food
    - Model learns a combination of vocabulary and contextual hints