CSCI 5541: Natural Language Processing

Lecture 3: Text Classification

Dongyeop Kang (DK), University of Minnesota

dongyeop@umn.edu | twitter.com/dongyeopkang | dykang.github.io

computer science & engineering

MINNESOTA NLP EST. 2021

UNIVERSITY OF MINNESOTA
Driven to Discover®

Outline

- Applications of text classification
- Why is sentiment analysis difficult?
- How can we build a sentiment classifier?
- Tutorial on building text classifier using Scikit-Learn and PyTorch (Shirley)

Movie review

Eternals is far from perfect, but it pushes the MCU into promising new territory....it feels like an amalgam of what Marvel does best - splendidly chaotic fight scenes, dazzling special effects, and stories that speak to who we are as human beings.

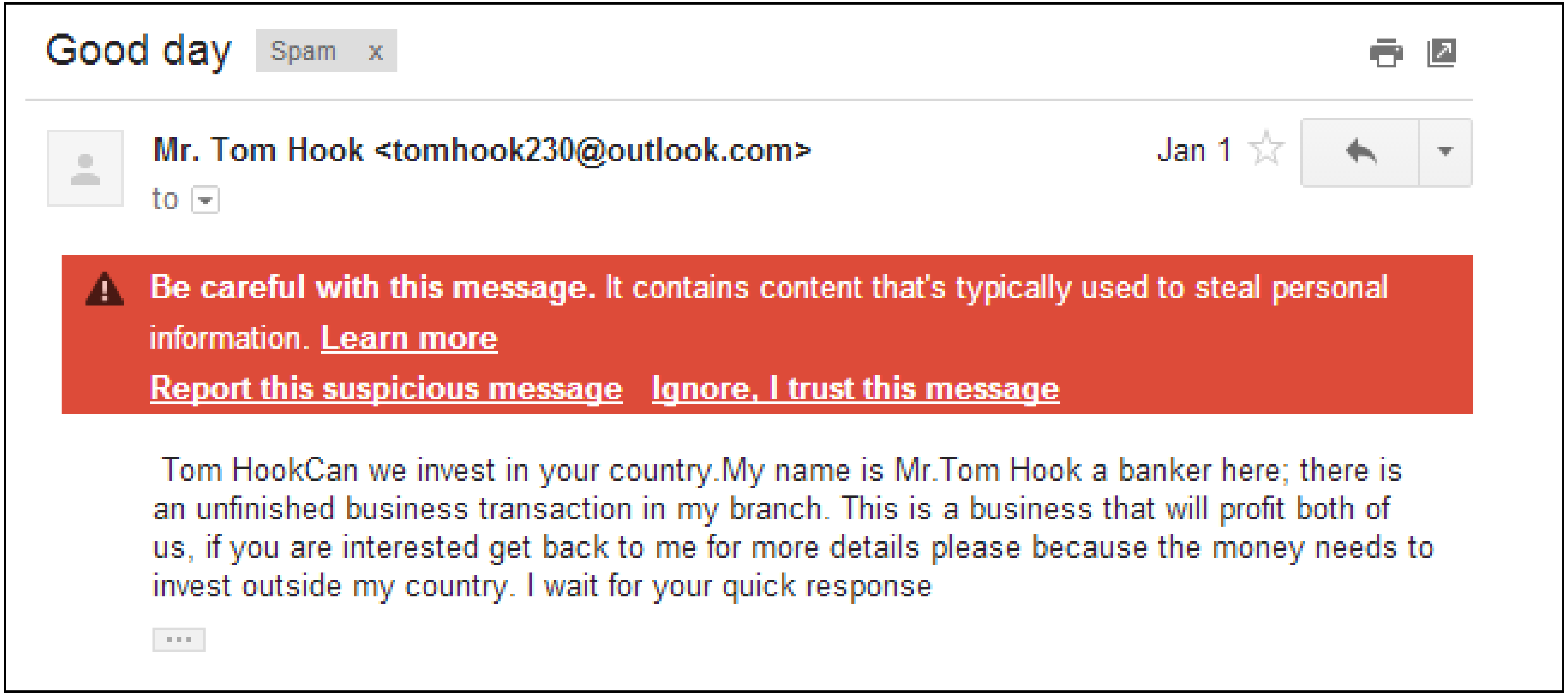
December 17, 2021 | Full Review...

Michael Blackmon
BuzzFeed News
TOP CRITIC

Erick V

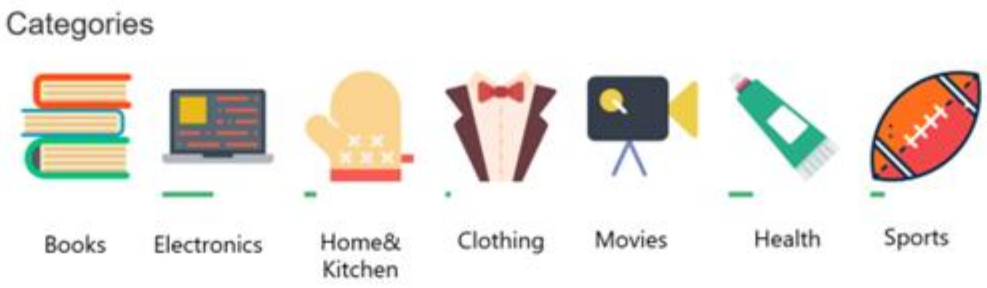
Really bad story, uninteresting and boring.

Spam detection



Language Identification

Authorship Identification



Topic/Genre Assignment

Why is sentiment analysis difficult?

There was an earthquake in California

The team failed to complete physical challenge. (We win/lose!)

They said it would be great.

They said it would be great, and they were great.

They said it would be great, and they were wrong.

Oh, you're terrible!

Long-suffering fans, bittersweet memories, hilariously embarrassing moments

Examples from Chris Potts

Scherer Typology of Affective States

- **Emotion**: brief organically synchronized … evaluation of a major event
  - angry, sad, joyful, fearful, ashamed, proud, elated
- **Mood**: diffuse non-caused low-intensity long-duration change in subjective feeling
  - cheerful, gloomy, irritable, listless, depressed, buoyant
- **Attitudes**: enduring, affectively colored beliefs, dispositions towards objects or persons
  - liking, loving, hating, valuing, desiring
- **Interpersonal stances**: affective stance toward another person in a specific interaction
  - friendly, flirtatious, distant, cold, warm, supportive, contemptuous
- **Personality traits**: stable personality dispositions and typical behavior tendencies
  - nervous, anxious, reckless, morose, hostile, jealous

Difficulty of task

- Simplest task:
  - Is the attitude of this text positive or negative (or neutral)?
- More complex:
  - Rank the attitude of this text from 1 to 5
- Advanced:
  - Detect the target (stance detection)
  - Detect source
  - ..

What makes reviews hard to classify?

Subtlety

“If you are reading this because it is your darling fragrance, please wear it at home exclusively, and tape the windows shut.”

Perfume review in *Perfumes: the Guide*

What makes reviews hard to classify?

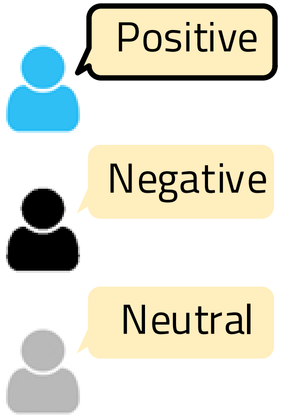
Thwarted expectations and ordering effects

"This film should be brilliant. It sounds like a great plot, the actors are first grade, and the supporting cast is good as well, and Stallone is attempting to deliver a good performance. However, it can't hold up.."

"Well as usual Keanu Reeves is nothing special, but surprisingly, the very talented Laurence Fishbourne is not so good either, I was surprised."

What makes reviews hard to classify?

Subjectivity and degree of sentiment



A: I got 3 veggies and a side of fries for over a 11 dollars if you like homecooked food

B: She listened to my ideas, asked questions to get a better idea about my style, and was excellent at offering advice as if I were a total pleb.

Annotation Records

Extractive

A is preferably more positive than B. (A > B)

Crowd Workers

Subjective

B is preferably more positive than A. (A < B)

"Prefer to Classify: Improving Text Classifiers via Auxiliary Preference Learning", ICML 2023

Why is sentiment analysis hard ?

- Sentiment is a measure of a speaker's private state, which is *unobservable*.
- Sentiment is contextual;
  - Words are a good indicator of sentiment (love, hate, terrible); but many times it requires deep world + contextual knowledge

> "*Valentine's Day* is being marketed as a Date Movie. I think it's more of a First-Date Movie. If your date likes it, do not date that person again. And if you like it, there may not be a second date."
>
> Roger Ebert, Valentine's Day

- Deep understanding of language behaviors (e.g., politeness)

Related Tasks

- Subjectivity (Pang & Lee 2008)
- Stance (Anand et al., 2011)
- Hate-speech (Nobata et al., 2016)
- Sarcasm (Khodak et al., 2017)
- Deception and betrayal (Niculae et al., 2015)
- Online trolls (Cheng et al., 2017)
- Politeness (Danescu-Niculescu-Mizil et al., 2013)
- …

How can we build a sentiment classifier?

Supervised Learning

- Given training data in the form of <x, y> pairs, learn **f(x)**

X	Y
I loved it!	Positive
Terrible movie.	Negative
Not too shabby	Positive
Such a lovely movie!	Positive

Learning f(x)

Two components:

- The formal structure of the learning method:
  - f: how x and y are mapped
  - Logistic regression, Naïve Bayes, RNN, CNN, etc
- The representation of the data (x)

Representation of data (x)

- Only positive/negative words in sentiment dictionaries
- Only words in isolation
- Conjunctions of words
- Linguistic structures
- ..

Sentiment Dictionaries

- General Inquirer (1996)
- MPQA subjectivity lexicon (Wilson et al., 2005)
  - http://mpqa.cs.pitt.edu/lexicons/subj_lexicon/
- LIWC (Pennebaker 2015)
- AFINN (Nielsen 2011)
- NRC Word-Emotion Association Lexicon (EmoLex) (Mohmmad and Turney, 2013)

Positive	Negative
unlimited	lag
prudent	contortions
superb	fright
closeness	lonely
impeccably	tenuously
fast-paced	plebeian
treat	mortification
destined	outrage
blessing	allegations

Dictionary Counting

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!

Positive	Negative
unlimited	lag
prudent	contortions
superb	fright
closeness	lonely
impeccably	tenuously
fast-paced	plebeian
treat	mortification
destined	outrage
blessing	allegations

happy	1
love	2
recommend	2

lonely	0
outrage	0
not	2

Limitation?

f (

happy	1
love	2
recommend	2
lonely	0
outrage	0
not	2

) = y

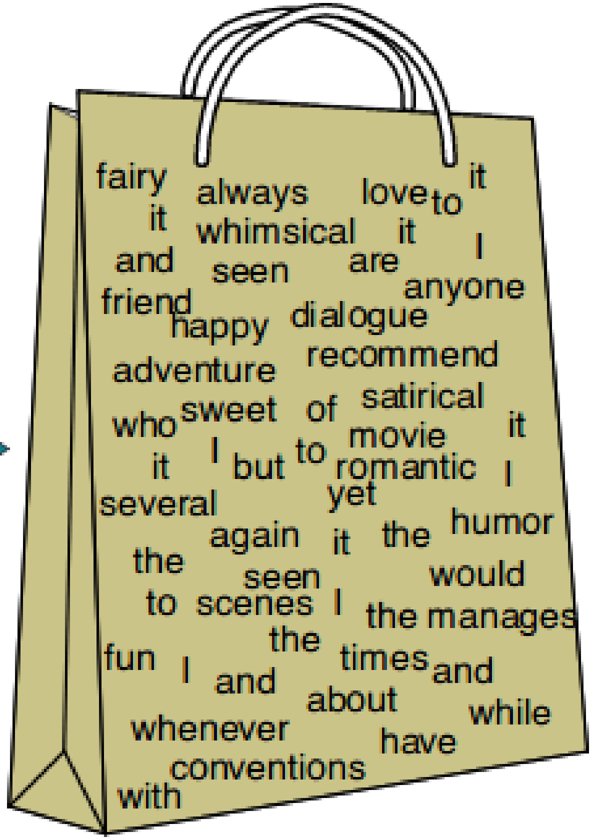
Representation of data (x)

- Only positive/negative words in sentiment dictionaries
- Only words in isolation (bag-of-words)
  - E.g., good, bad
- Conjunctions of words (sequential, high-order n-grams, skip n-grams, etc)
  - E.g., “not good”, “not bad”
- Linguistic structures (Part-of-speech, etc)
- ..

Bag of words

Representation of text only as the counts of words that it contains

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!



it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
...	...

$f($

it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
...	...

$) = y$

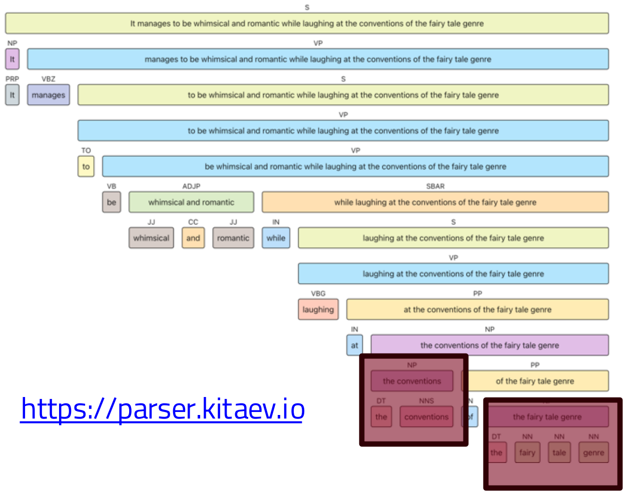
Representation of data (x)

- Only positive/negative words in sentiment dictionaries
- Only words in isolation (bag-of-words)
- Conjunctions of words (sequential, high-order n-grams, skip n-grams, etc)
- Linguistic structures (Part-of-speech, etc)
- ..

Linguistic Structures

Count the number of part-of-Speech, depth of constituency parses, etc

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!



https://parser.kitaev.io

Parse depth	5

NP	5
VP	2
..	..

f (

NP	5
VP	2
Parse depth	5

) = y

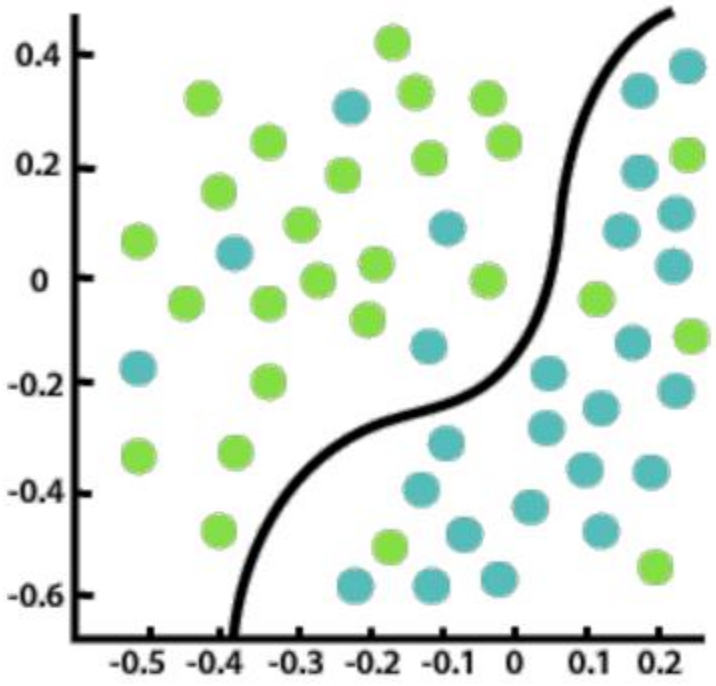
How to implement f(x)=y using Python?

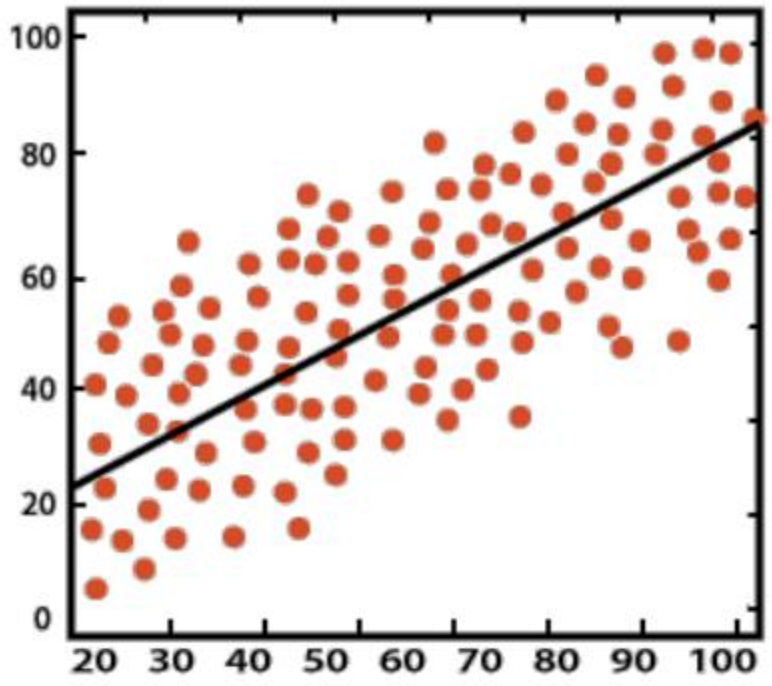
Two components:

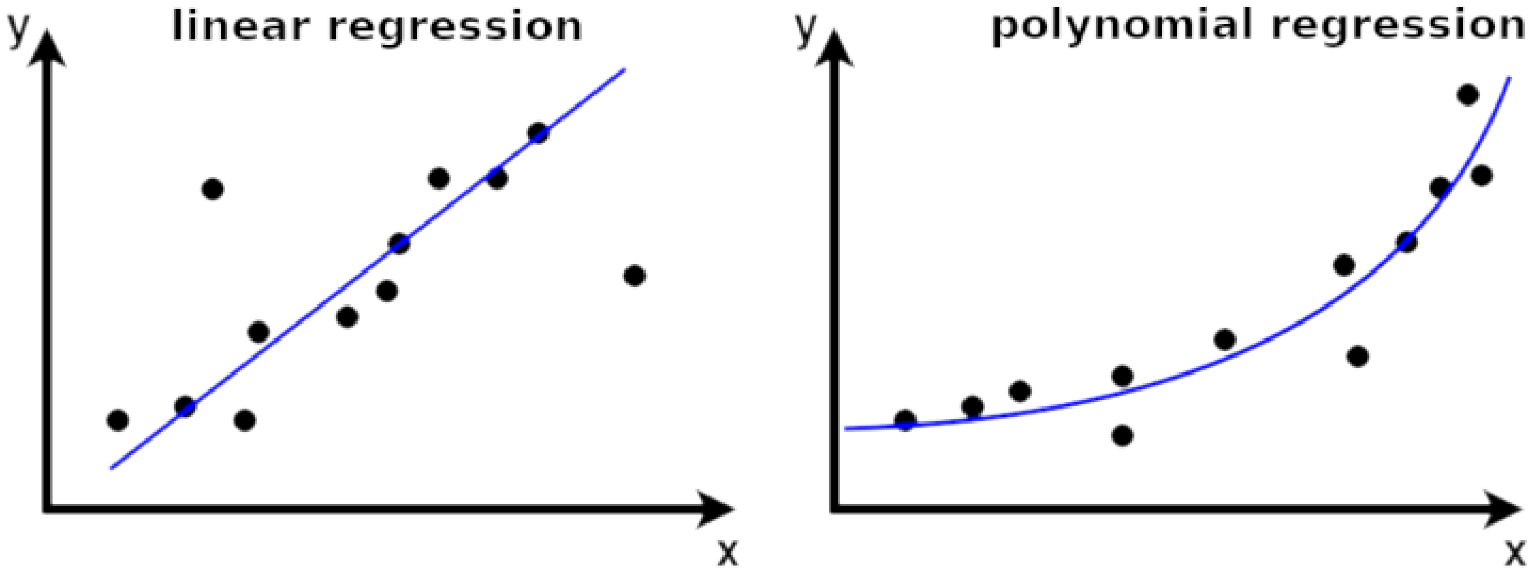
- **The formal structure of the learning method:**
  - f: how x and y are mapped
  - Logistic regression, Naïve Bayes, RNN, CNN, etc
- ~~The representation of the data (x)~~

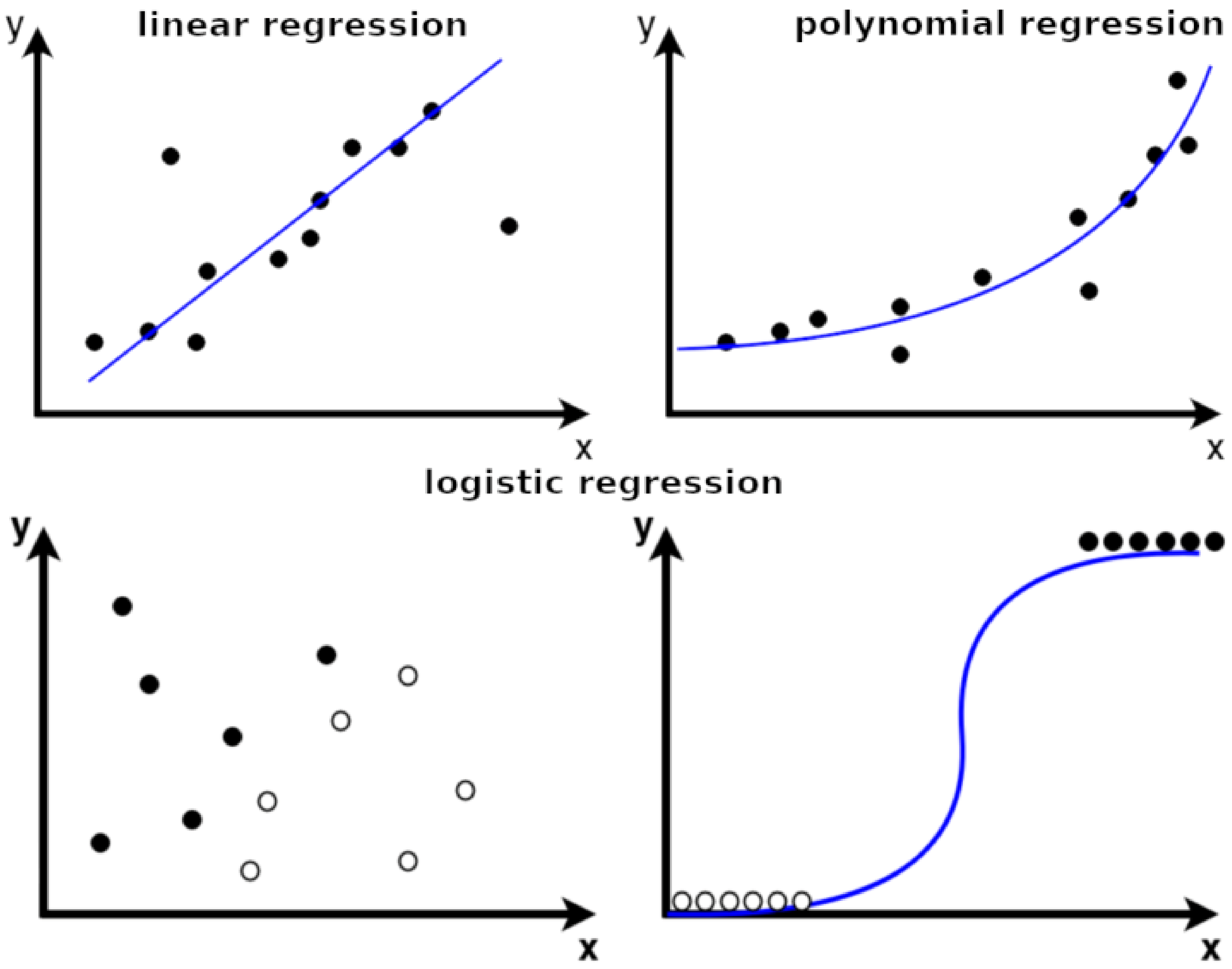
Tutorial on building text classifier using Scikit-Learn and PyTorch

Classification vs Regression

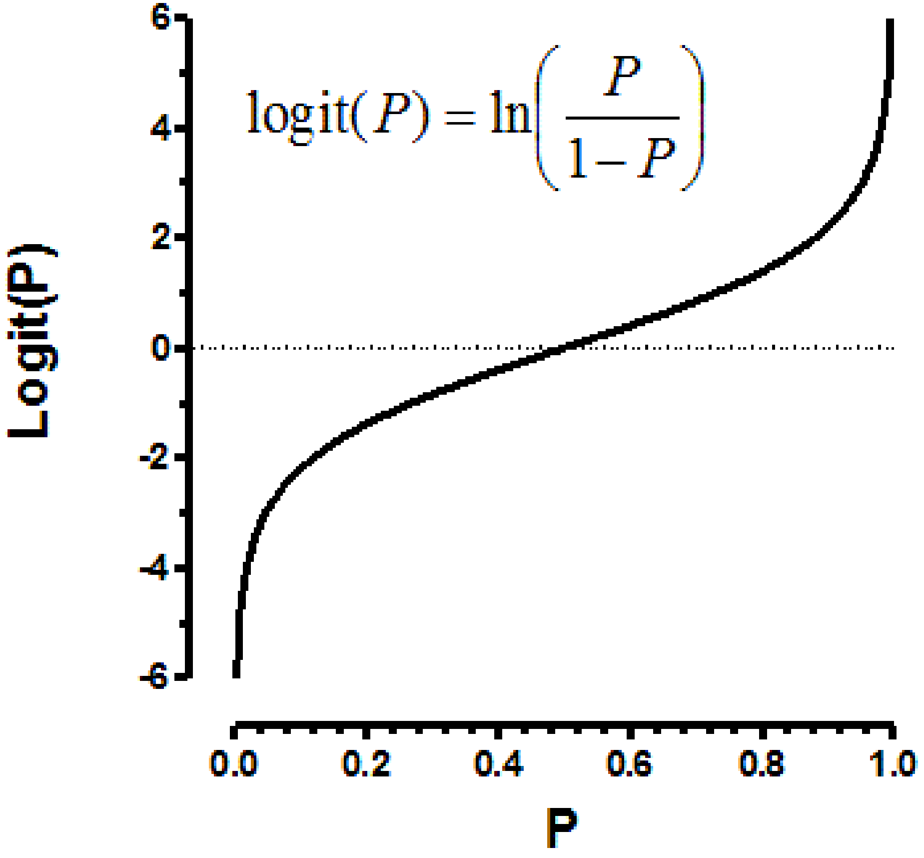


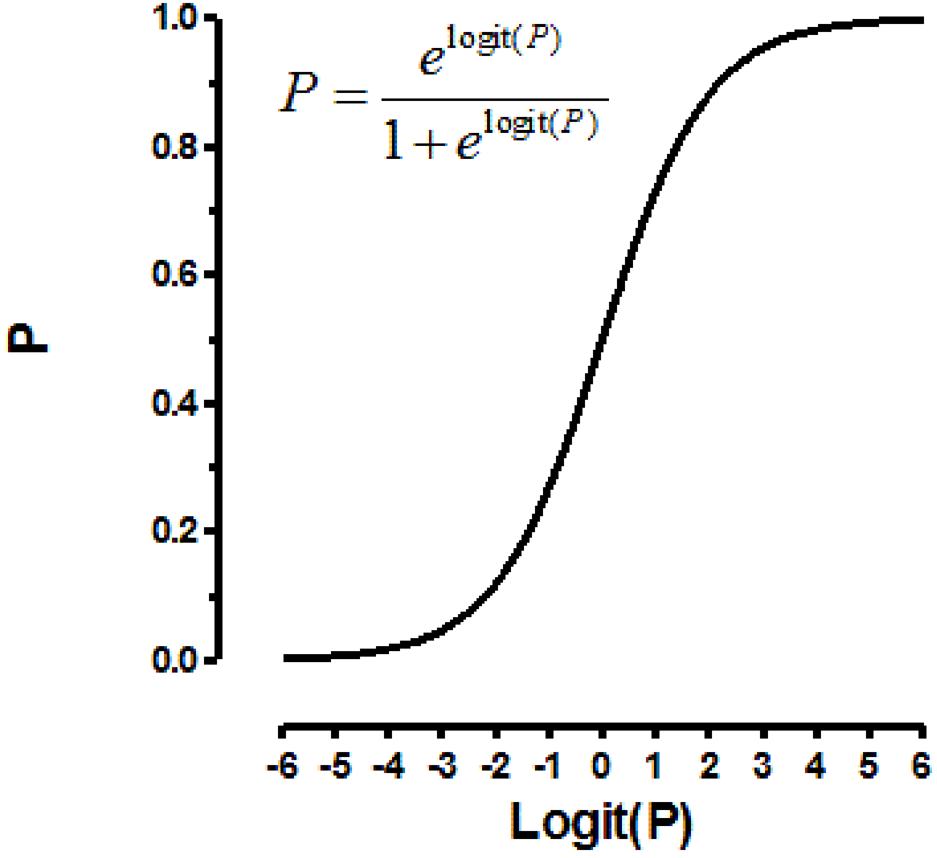



linear regression
y
x
polynomial regression
y
x


linear regression
polynomial regression
logistic regression
y
x
y
x
y
x
y
x

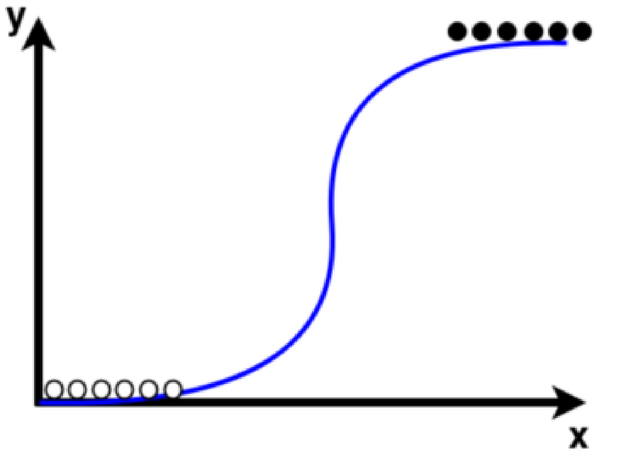
Probability 101: Logit(P) and Logistic Regression

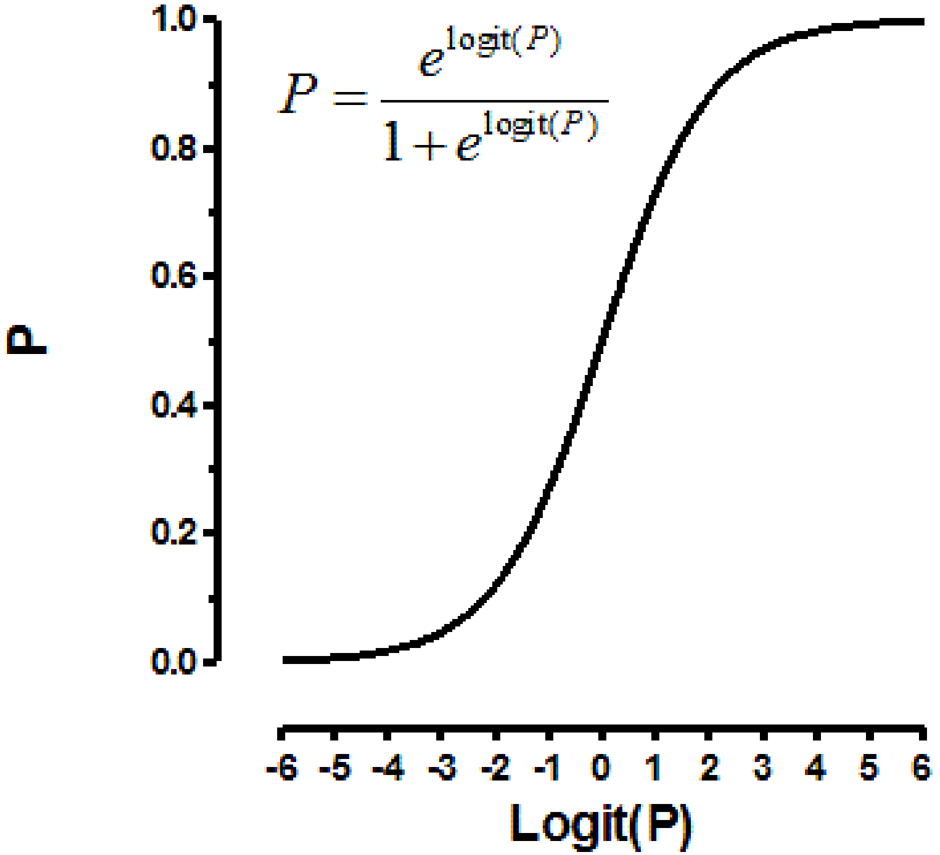




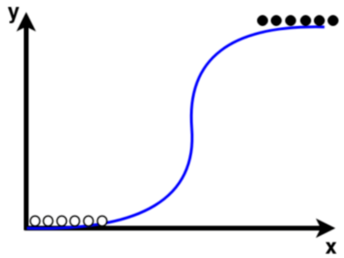
Logistic regression

$$p = \frac{1}{1 + e^{-(b_0 + b_1 x)}}$$





Binary logistic regression

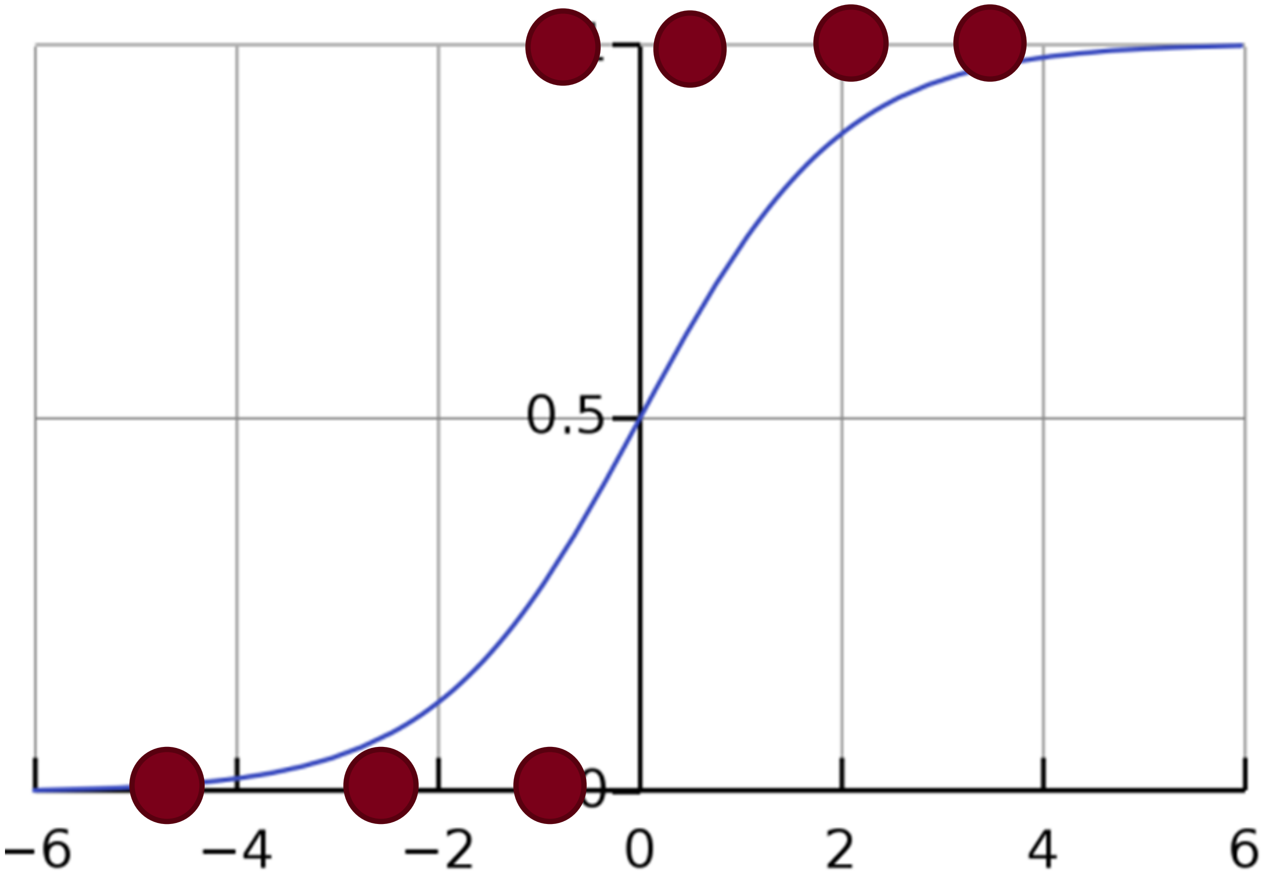


Model parameters to learn

$$P(y = 1 \mid x, \beta) = \frac{1}{1 + \exp\left(-\sum_{i=1}^{F} x_i \beta_i\right)}$$

output space $\mathcal{Y} = \{0, 1\}$

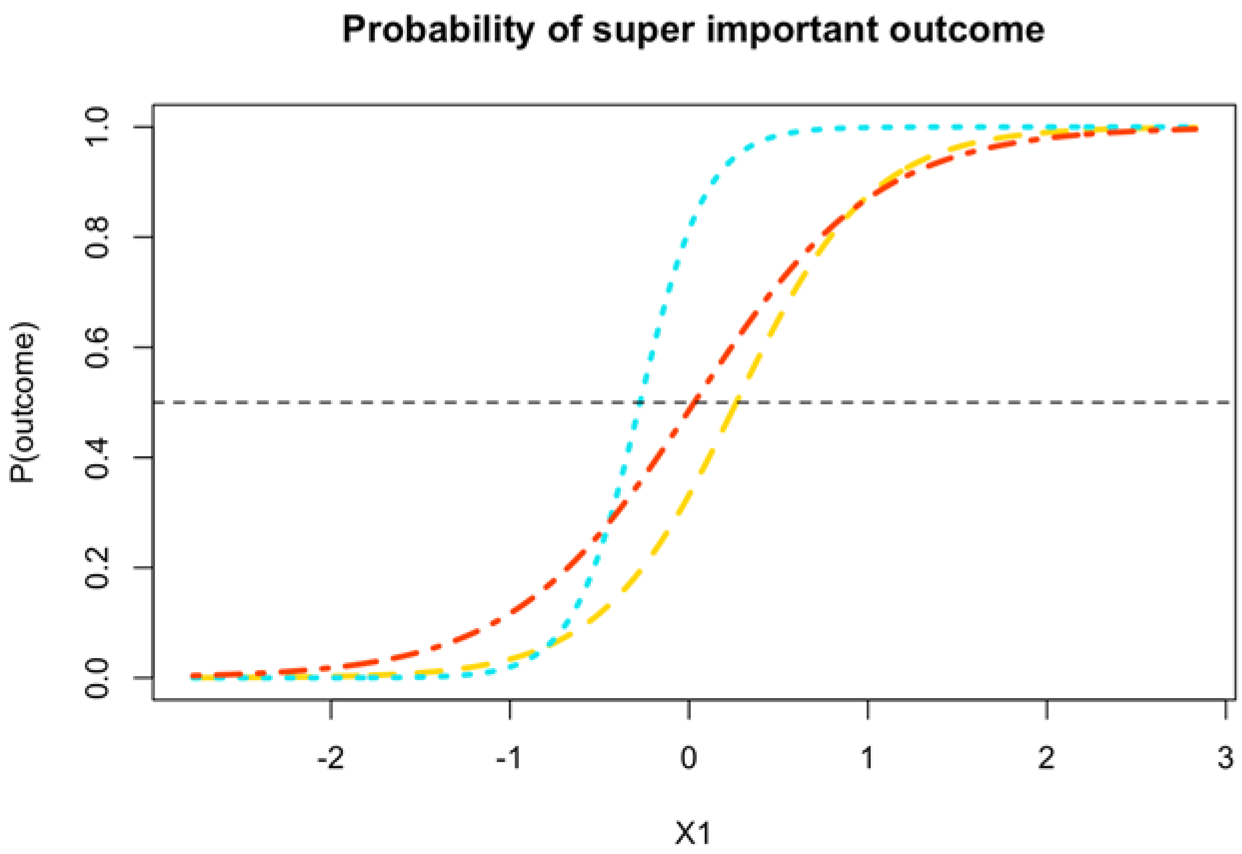
Binary logistic regression



f (x) = y

it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1

Importance of your features: β



	x	β
not	1	-0.5
bad	1	-1.7
movie	0	0.3

Logistic regression

$$P(y = 1 \mid x, \beta) = \frac{1}{1 + \exp\left(-\sum_{i=1}^{F} x_i \beta_i\right)}$$

- We want to find the value of β that leads to the highest value of the conditional log likelihood:

$$\ell(\beta) = \sum_{i=1}^{N} \log P(y_i \mid x_i, \beta)$$

$$\begin{aligned}
\nabla_\beta l(\beta; y, X) &= \nabla_\beta \left(\sum_{i=1}^{N} [-\ln(1 + \exp(x_i\beta)) + y_i x_i \beta] \right) \\
&= \sum_{i=1}^{N} (\nabla_\beta [-\ln(1 + \exp(x_i\beta)) + y_i x_i \beta]) \\
&= \sum_{i=1}^{N} \left(-\frac{\exp(x_i\beta)}{1 + \exp(x_i\beta)} x_i + y_i x_i \right) \\
&= \sum_{i=1}^{N} \left(y_i - \frac{\exp(x_i\beta)}{1 + \exp(x_i\beta)} \right) x_i \\
&= \sum_{i=1}^{N} \left(y_i - \frac{\exp(x_i\beta)}{1 + \exp(x_i\beta)} \frac{\exp(-x_i\beta)}{\exp(-x_i\beta)} \right) x_i \\
&= \sum_{i=1}^{N} \left(y_i - \frac{1}{1 + \exp(-x_i\beta)} \right) x_i \\
&= \sum_{i=1}^{N} [y_i - S(x_i\beta)] x_i
\end{aligned}$$

Logistic regression

- We want to find the value of β that leads to the highest value of the conditional log likelihood:

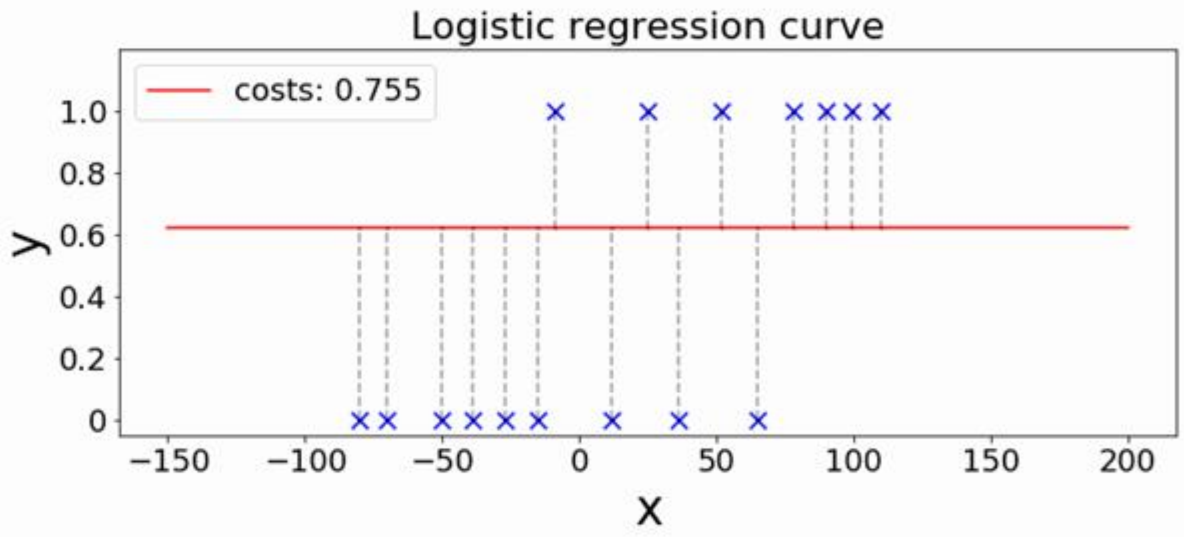
$$\ell(\beta) = \sum_{i=1}^{N} \log P(y_i \mid x_i, \beta)$$

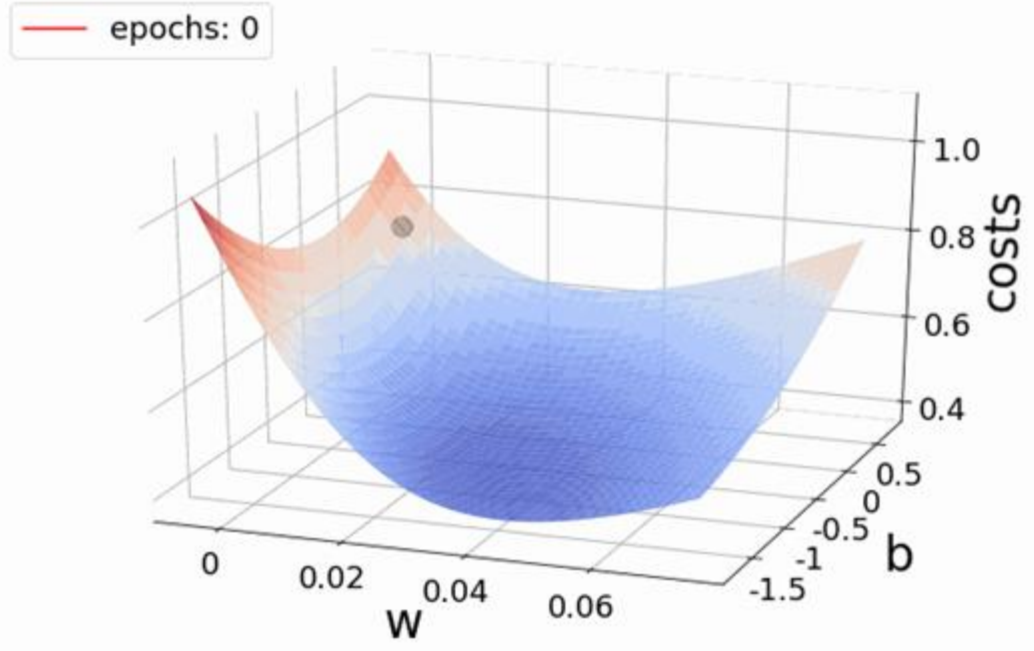
- Train it with stochastic gradient descent

Algorithm 2 Logistic regression stochastic gradient descent

Data: training data $x \in \mathbb{R}^F, y \in \{0,1\}$
$\beta = 0^F$
while not converged **do**
  for $i = 1$ to N **do**
    $\boxed{\beta_{t+1} = \beta_t + \alpha\left(y_i - \hat{p}(x_i)\right) x_i}$
  end for
end while

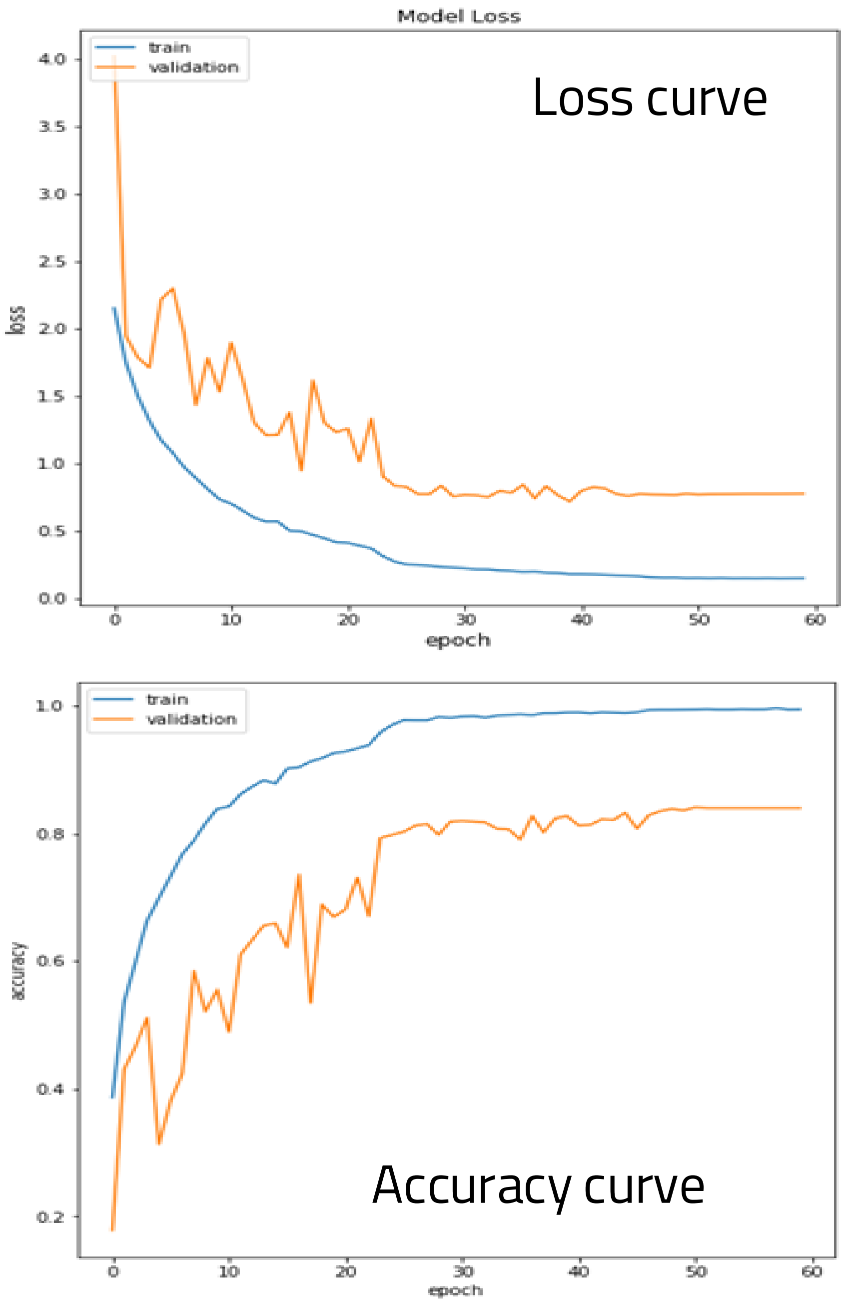
$$\begin{aligned}
\nabla_\beta l(\beta; y, X) &= \nabla_\beta \left(\sum_{i=1}^{N} [-\ln(1+\exp(x_i\beta)) + y_i x_i \beta] \right) \\
&= \sum_{i=1}^{N} (\nabla_\beta [-\ln(1+\exp(x_i\beta)) + y_i x_i \beta]) \\
&= \sum_{i=1}^{N} \left(-\frac{\exp(x_i\beta)}{1+\exp(x_i\beta)} x_i + y_i x_i \right) \\
&= \sum_{i=1}^{N} \left(y_i - \frac{\exp(x_i\beta)}{1+\exp(x_i\beta)} \right) x_i \\
&= \sum_{i=1}^{N} \left(y_i - \frac{\exp(x_i\beta)}{1+\exp(x_i\beta)} \frac{\exp(-x_i\beta)}{\exp(-x_i\beta)} \right) x_i \\
&= \sum_{i=1}^{N} \left(y_i - \frac{1}{1+\exp(-x_i\beta)} \right) x_i \\
&= \sum_{i=1}^{N} [y_i - S(x_i\beta)] x_i
\end{aligned}$$

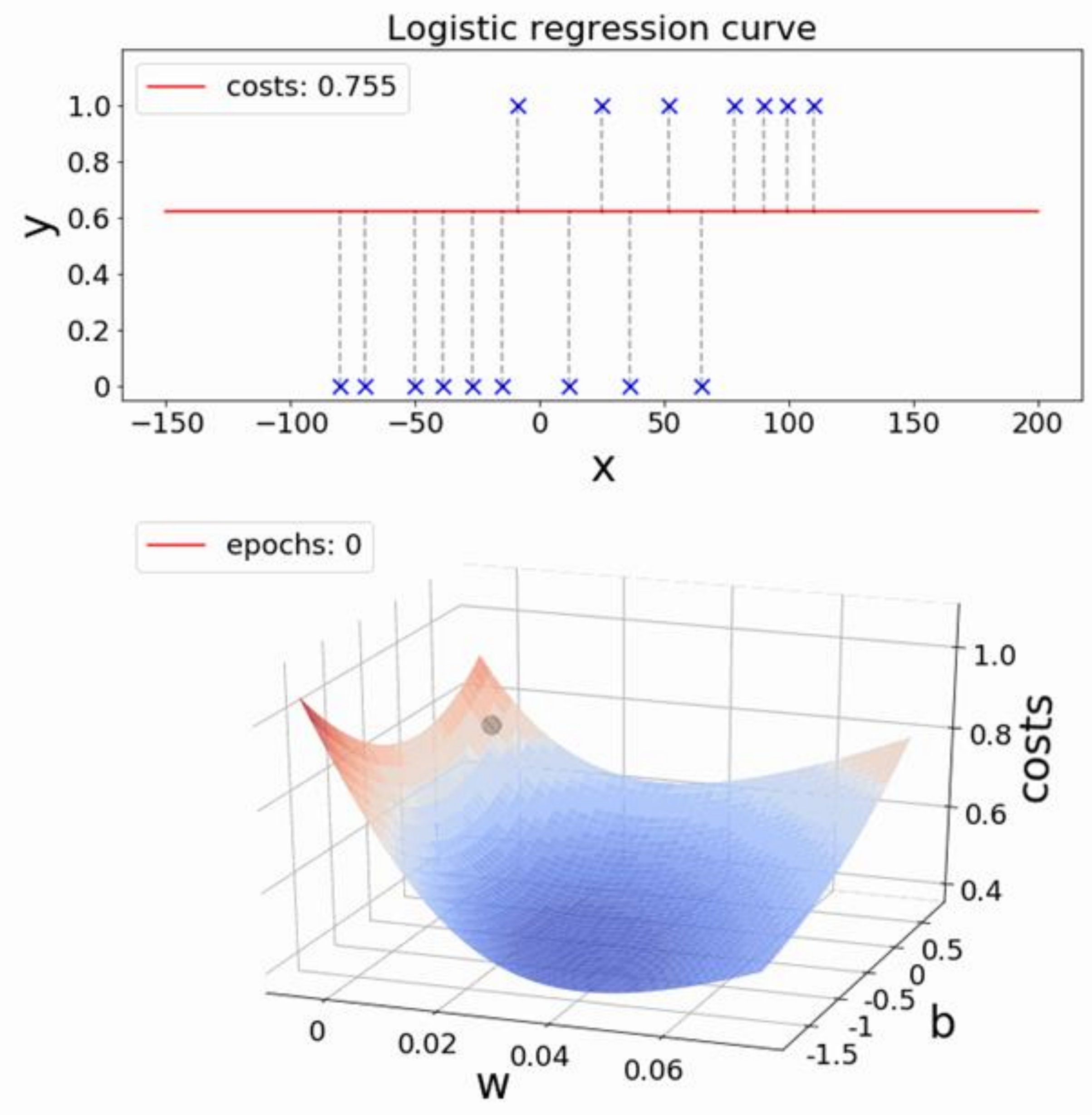




Algorithm 2 Logistic regression stochastic gradient descent

1: Data: training data $x \in \mathbb{R}^F, y \in \{0, 1\}$
2: $\beta = 0^F$
3: **while** not converged **do**
4:     **for** $i = 1$ to N **do**
5:         $\beta_{t+1} = \beta_t + \alpha \left(y_i - \hat{p}(x_i)\right) x_i$
6:     **end for**
7: **end while**


Model Loss
train
validation
Loss curve
loss
epoch
train
validation
accuracy
epoch
Accuracy curve


Logistic regression curve
costs: 0.755
y
x
epochs: 0
costs
w
b

Representation of data (x)

- As a discriminative classifier, logistic regression doesn't assume features are independent like Naive Bayes does.
- Its power partly comes in the ability to create richly expressive features without the burden of independence.
- We can represent text through features that are not just the identities of individual words, but any feature that is scoped over the entirety of the input.

Features
Unigrams ("like")
Bigrams ("not like"), trigrams, etc
Prefixes (word that start with "un-"
Words that appear in the positive/negative dictionary
Reviews begin with "I love"
At least 3 mentions of positive verbs (like, love, etc)

Representation of data (x)

f (

it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
…	…

I want	1
have you	2
how many	2
isn't it	1
…	

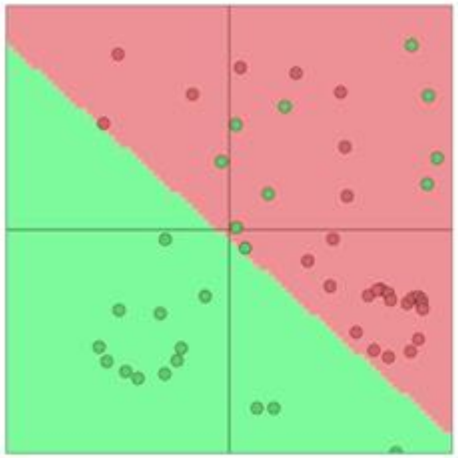
happy	1
love	2
recommend	2

lonely	0
outrage	0
not	2

NP	5
VP	2
Parse depth	5

)=y

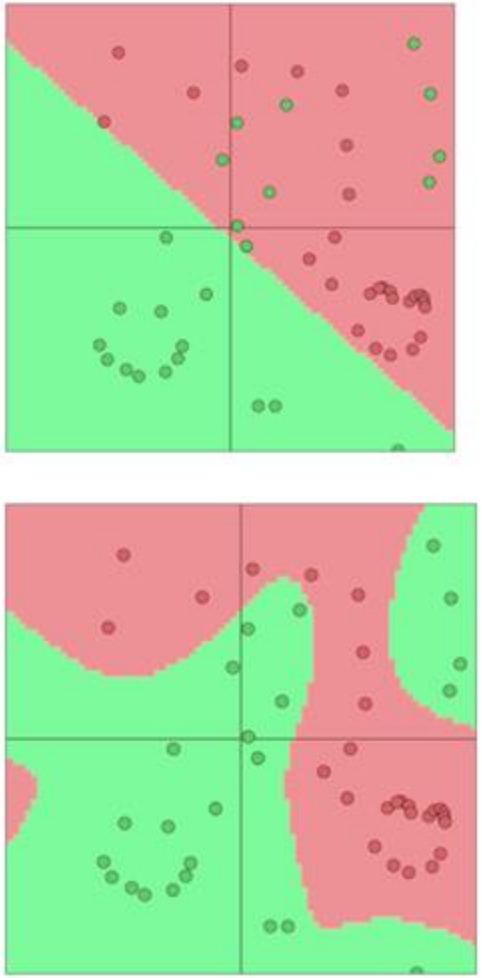
Features
Unigrams ("like")
Bigrams ("not like"), trigrams, etc
Prefixes (word that start with "un-"
Words that appear in the positive/negative dictionary
Reviews begin with "I love"
At least 3 mentions of positive verbs (like, love, etc)



What if your input representation is *complex* and cannot be modeled by simple *linear projection*?

Neural Networks

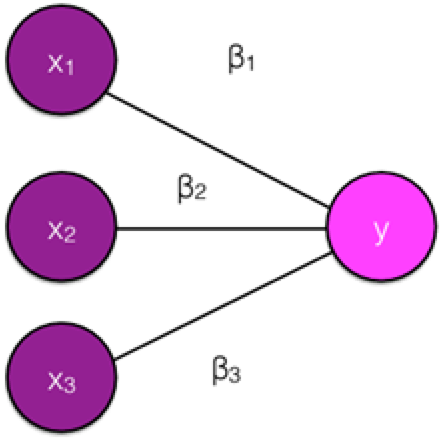
- Discrete, high-dimensional representation of inputs (one-hot vectors) => low-dimensional "distributed" representations.
  - Distributional semantics and word vectors (To be covered)
- Static representations -> contextual representations, where representations of words are sensitive to local context.
  - Contextualized Word Embeddings (To be covered)
- Multiple layers to capture hierarchical structure



Recap: Logistic regression

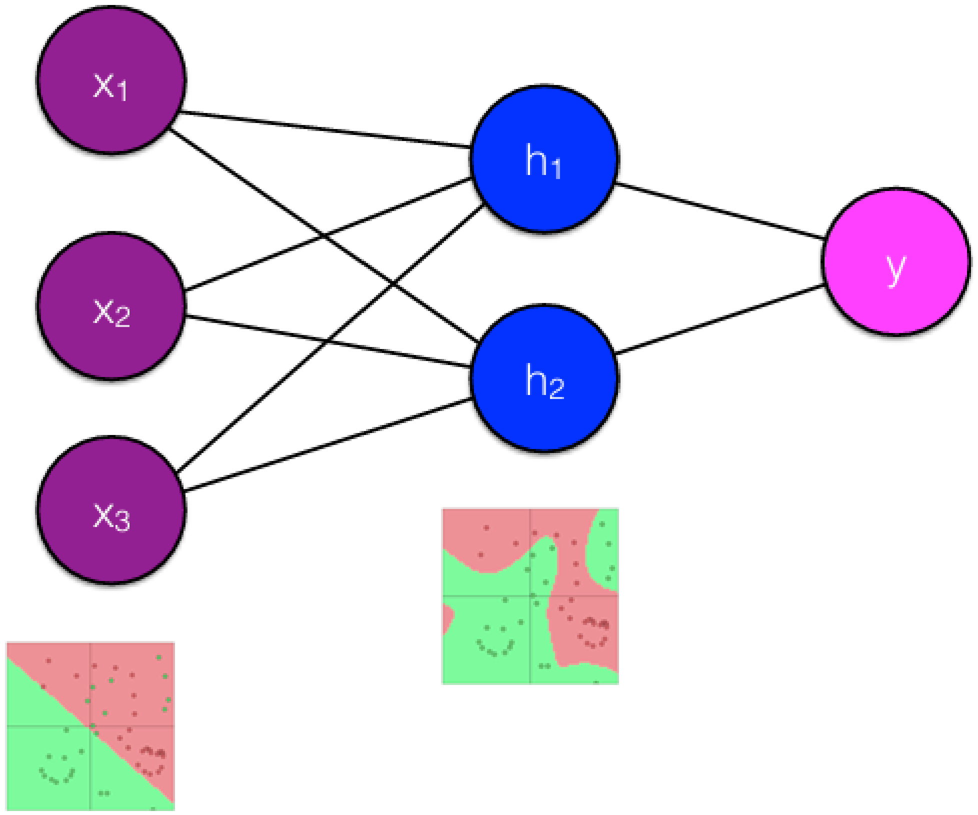
$$P(\hat{y} = 1) = \frac{1}{1 + \exp\left(-\sum_{i=1}^{F} x_i \beta_i\right)}$$

	x	β
not	1	-0.5
bad	1	-1.7
movie	0	0.3

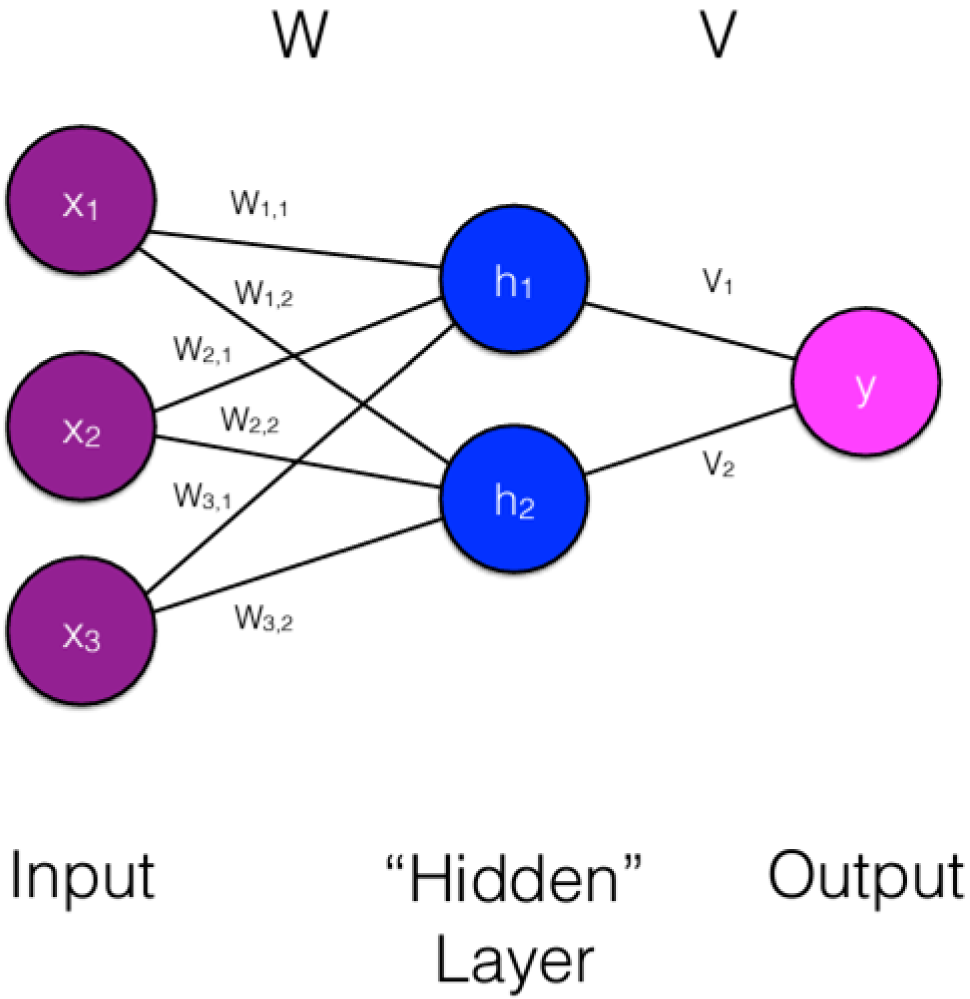


Feedforward neural network

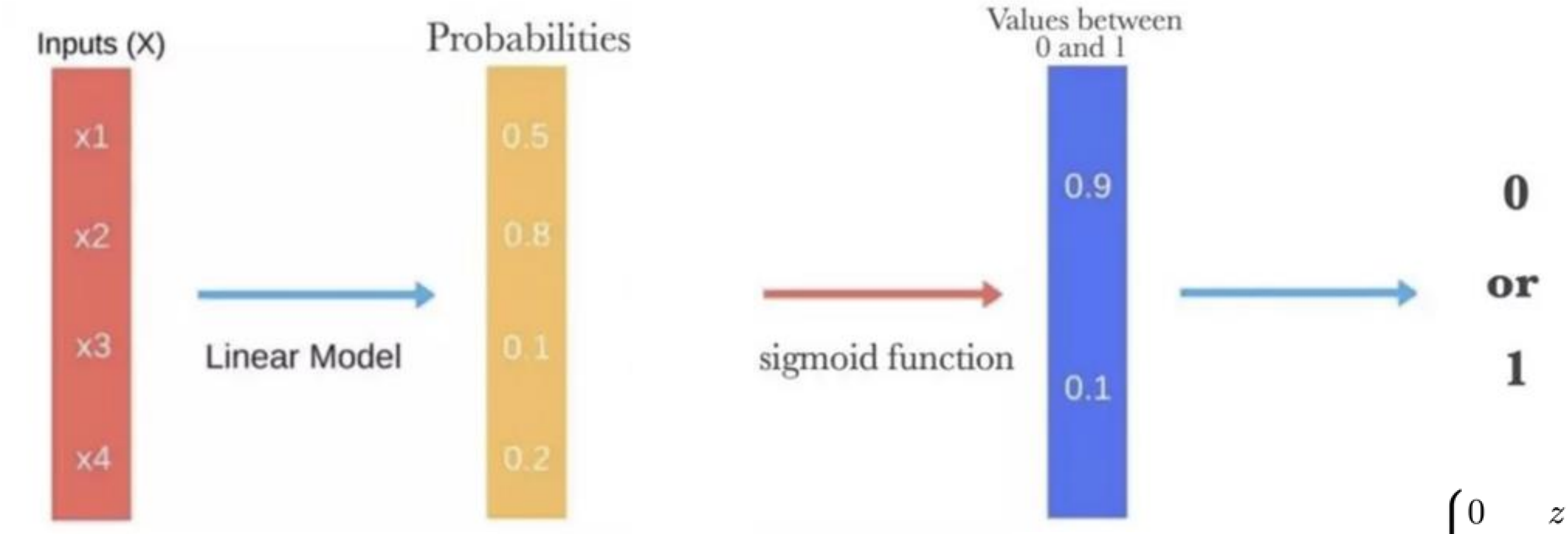
- Input and output are mediated by at least one hidden layer.



*For simplicity, we're leaving out the bias term, but assume most layers have them as well.



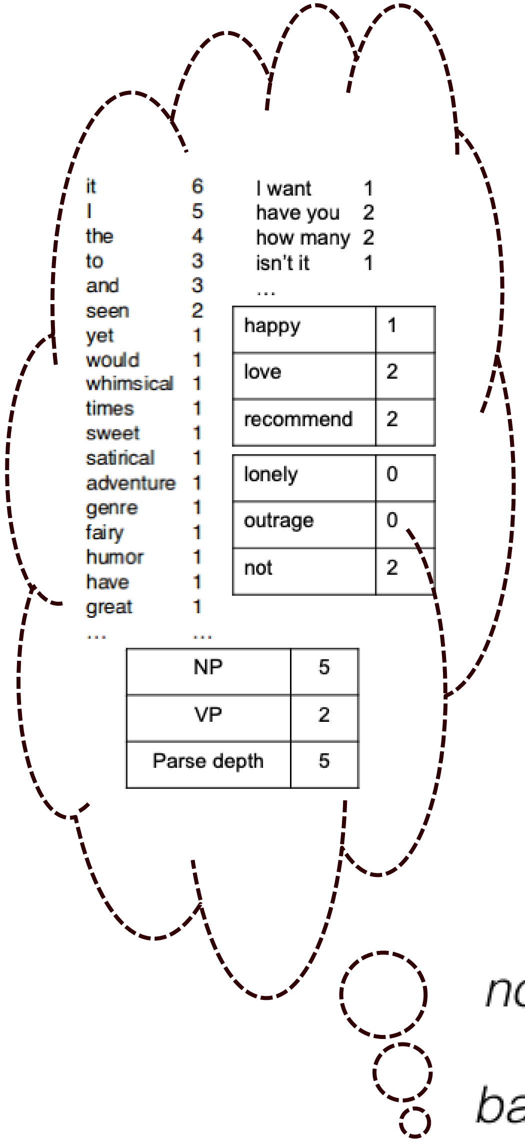
Relations with logistic regression

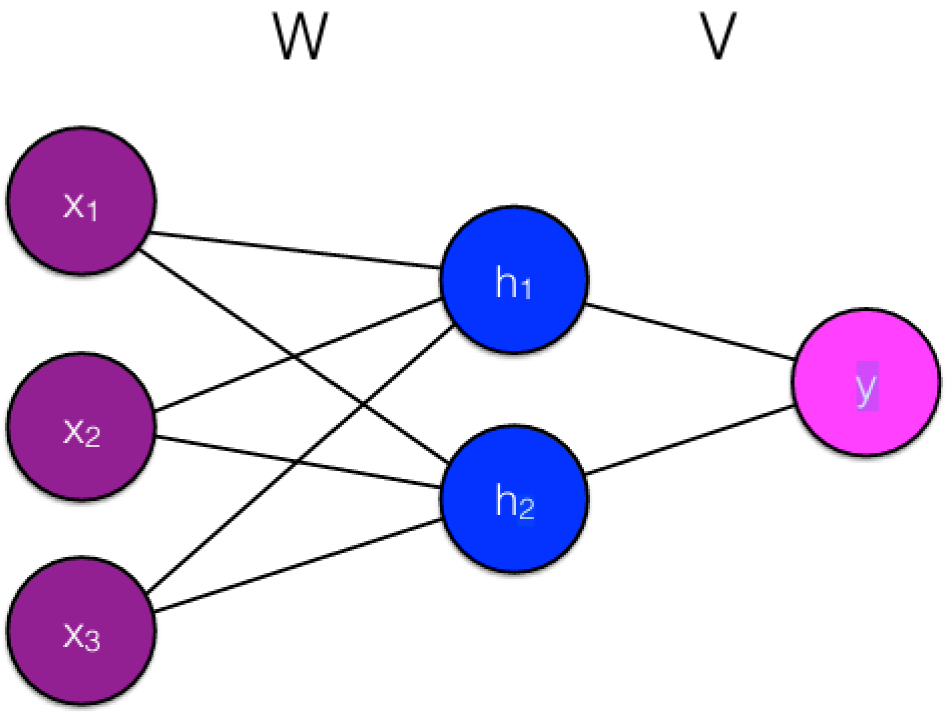


$$(z = \mathbf{w}^{T}\boldsymbol{x} + b)$$

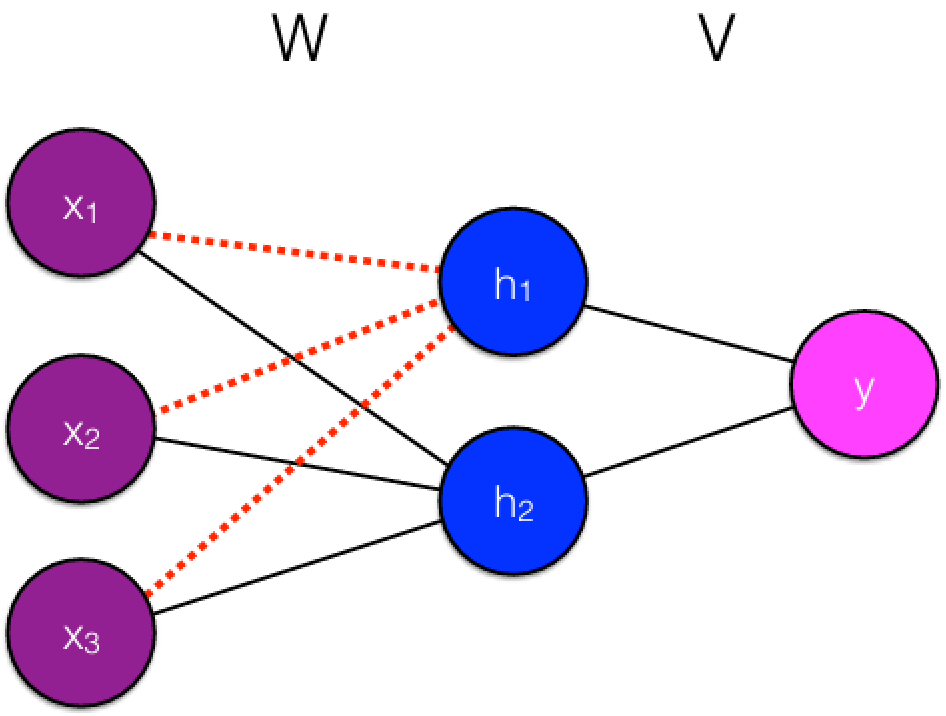
$$y = \frac{1}{1+e^{-z}} = \frac{1}{1+e^{-\mathbf{w}^{T}\boldsymbol{x}+b}}$$

$$y = \begin{cases} 0 & z < 0 \\ 0.5 & z = 0 \\ 1 & z > 0 \end{cases}$$





	x	W		V	y
not	1	-0.5	1.3	4.1	1
bad	1	0.4	0.08	-0.9	
movie	0	1.7	3.1		



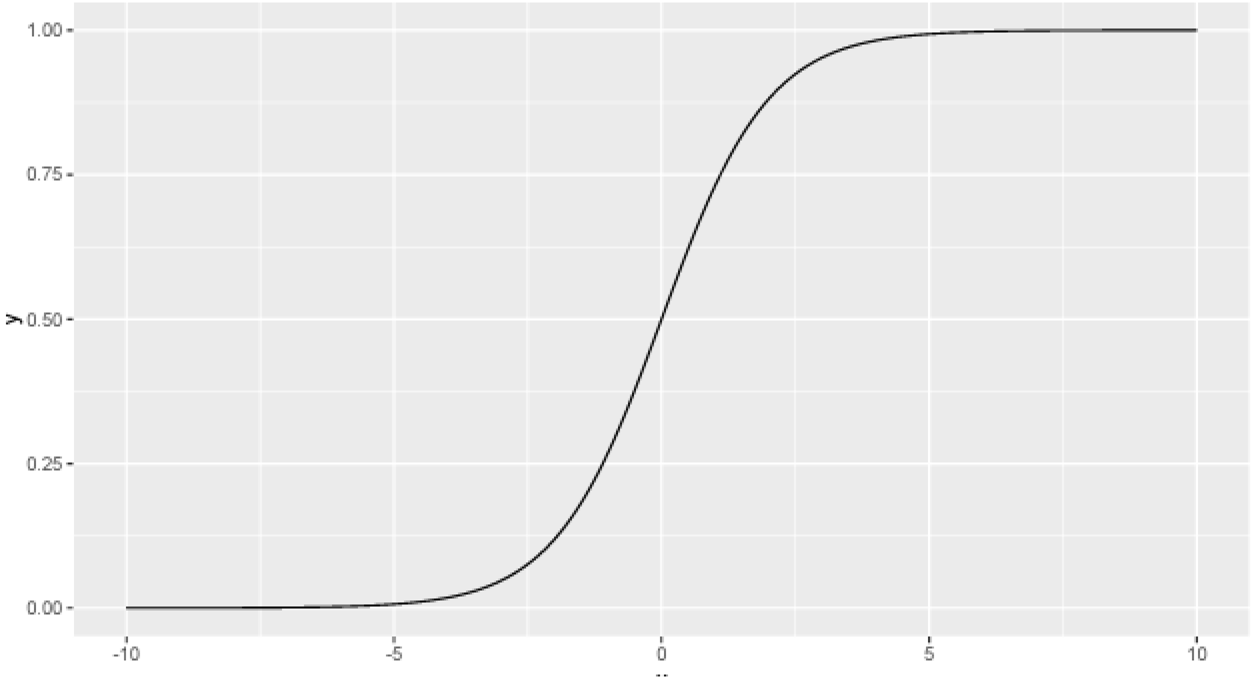
$$h_j = f\left(\sum_{i=1}^{F} x_i W_{i,j}\right)$$

the hidden nodes are completely determined by the input and weights

Activation functions

Squeezing outputs between 0 and 1

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

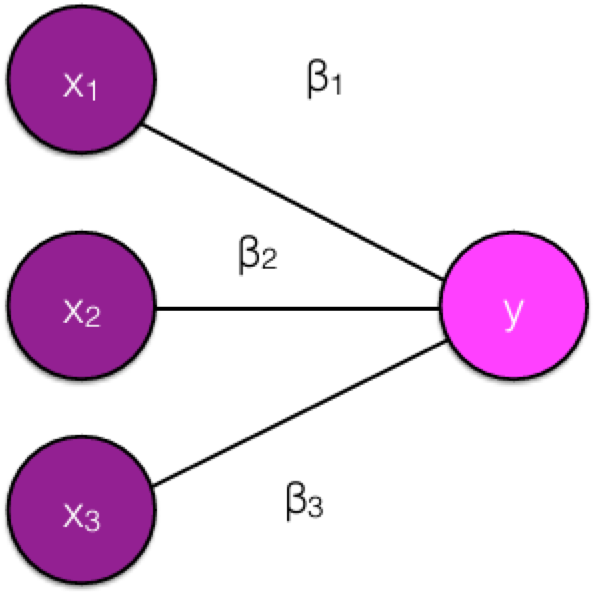


Activation functions

Squeezing outputs between 0 and 1

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

$$P(\hat{y} = 1) = \sigma\left(\sum_{i=1}^{F} x_i \beta_i\right)$$



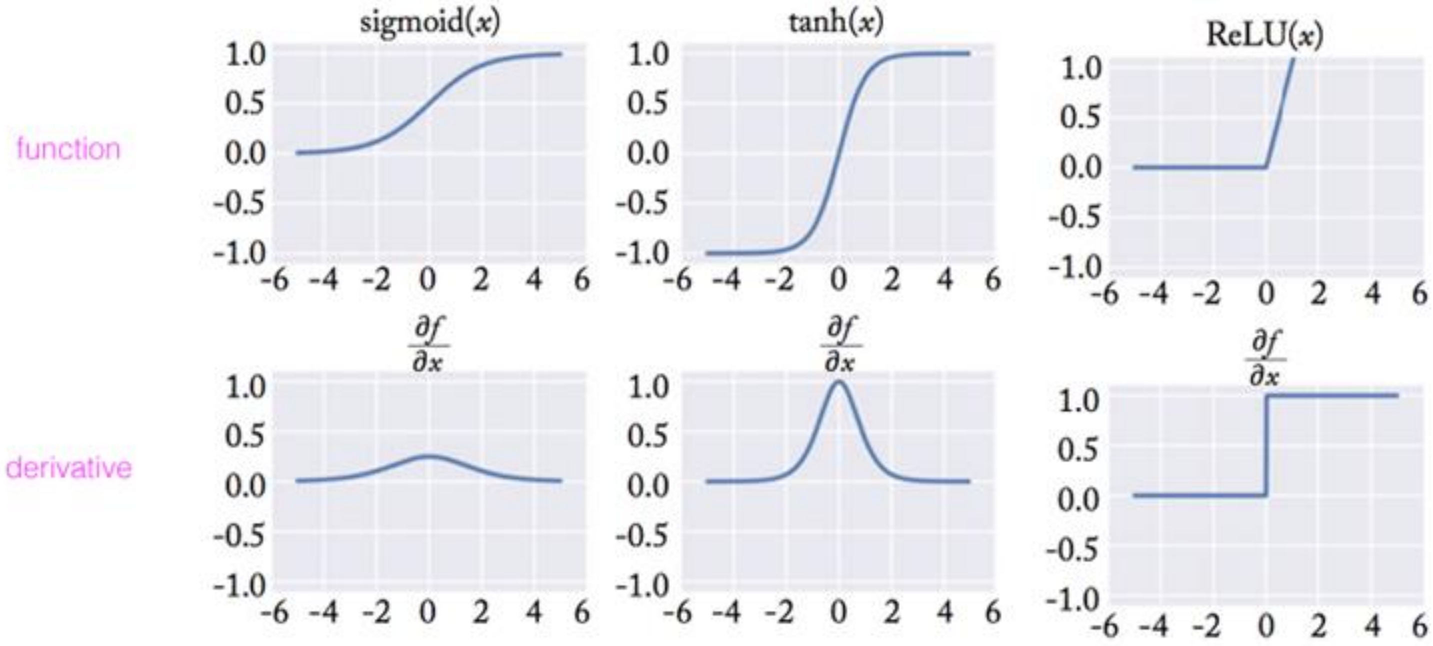
$$P(\hat{y} = 1) = \frac{1}{1 + \exp\left(-\sum_{i=1}^{F} x_i \beta_i\right)}$$

We can think about logistic regression as a neural network with no hidden layers

Activation functions

Squeezing outputs between 0 and 1

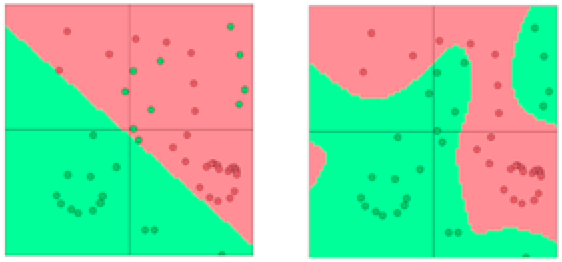
- Sigmoid is useful for final layer to scale output between 0 and 1, but is not often used in intermediate layers.
- ReLU and tanh are both used extensively in modern systems.
  - Check out the derivative

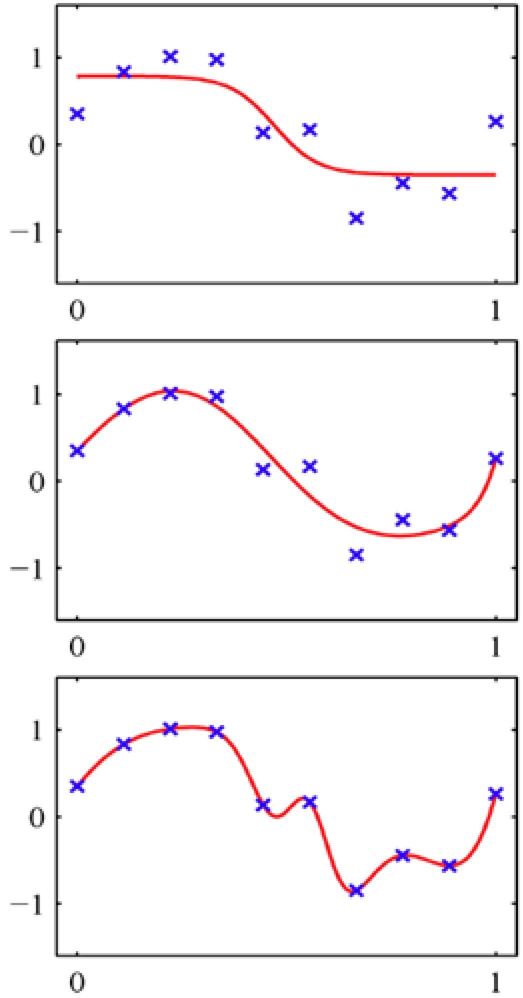


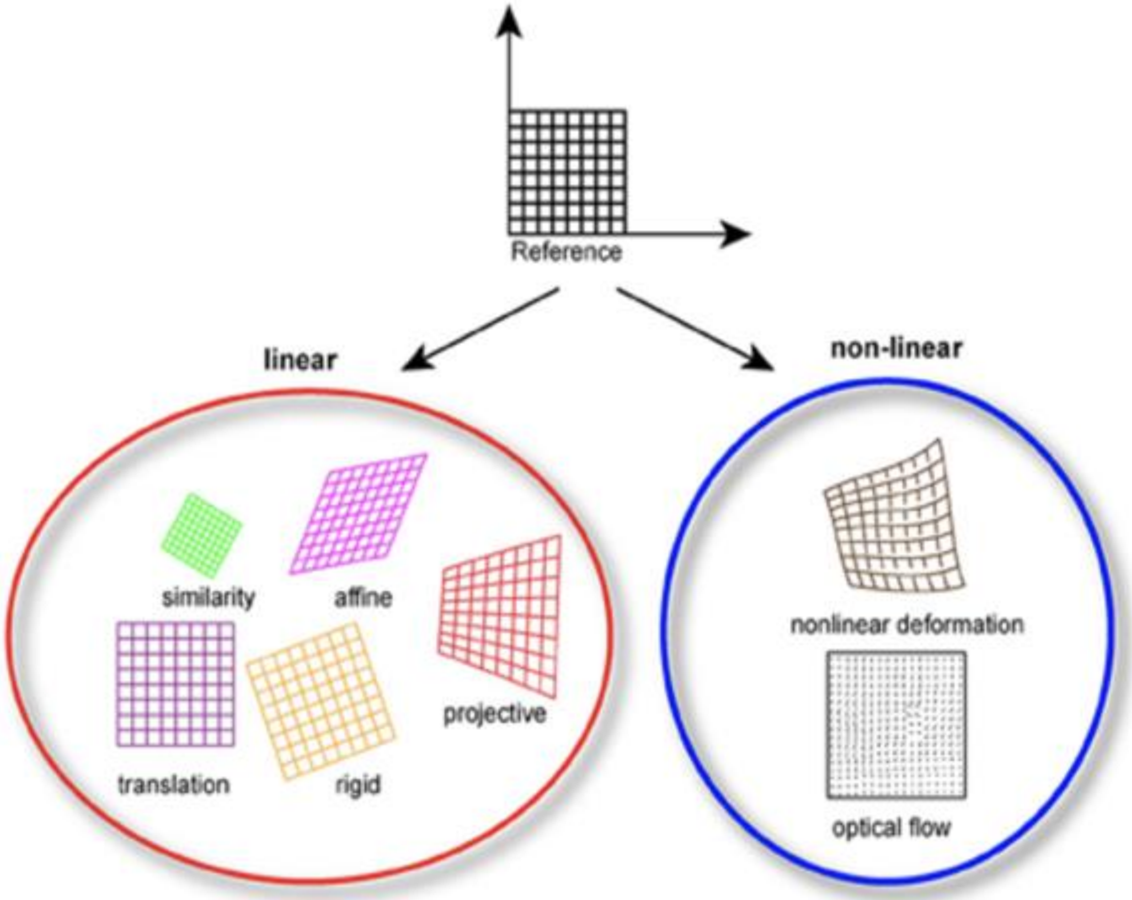
Goldberg 46

Non-linearities (i.e., f): why they're needed?

- Neural nets do function approximation
  - E.g., regression or classification
  - Without non-linearities, deep neural nets can't do anything more than a linear transform.
  - Extra layers could just be complied down into a single linear transform: $W_1 W_2 x = W x$
  - But, with more layers that include non-linearities, they can approximate more complex functions

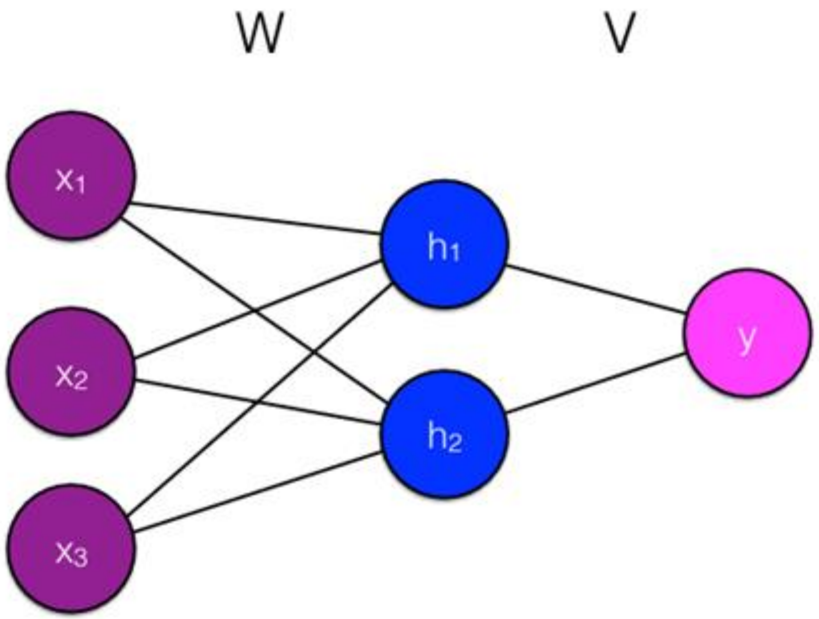


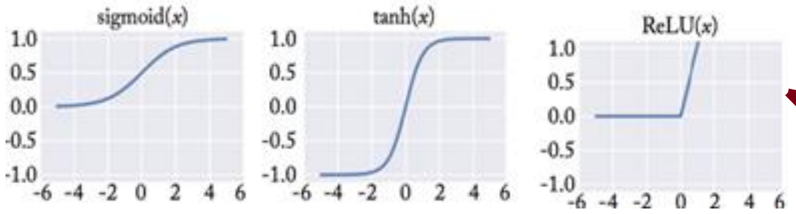




Linear models include translation, rigid (translation + rotation), similarity (translation + rotation + scale), affine and projective transformations. Nonlinear models, which consider non-linear transformations allow for more complex deformations.

New techniques for motion-artifact-free in vivo cardiac microscopy

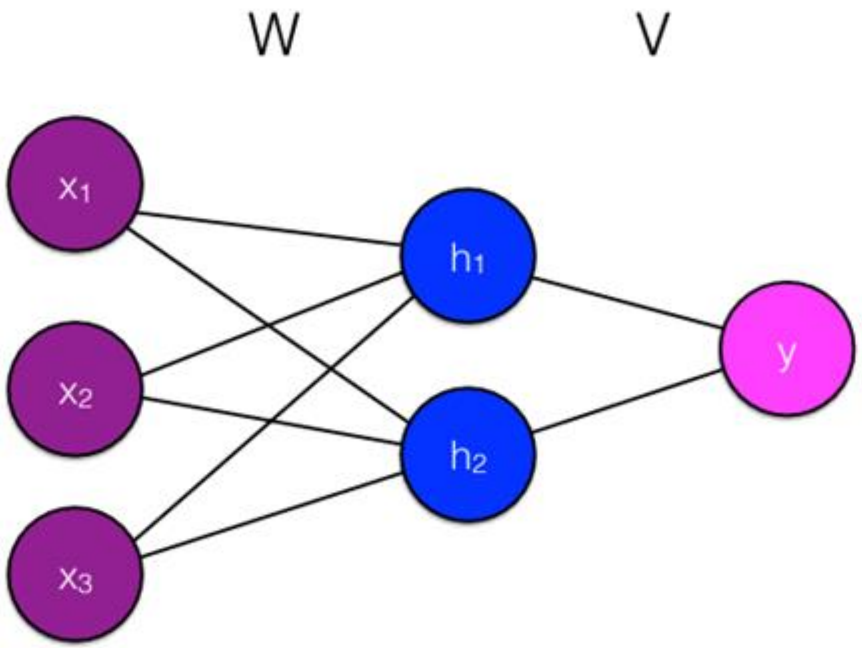




$$h_1 = \sigma\left(\sum_{i=1}^{F} x_i W_{i,1}\right)$$

$$\hat{y} = \sigma\left[V_1 h_1 + V_2 h_2\right]$$

$$h_2 = \sigma\left(\sum_{i=1}^{F} x_i W_{i,2}\right)$$

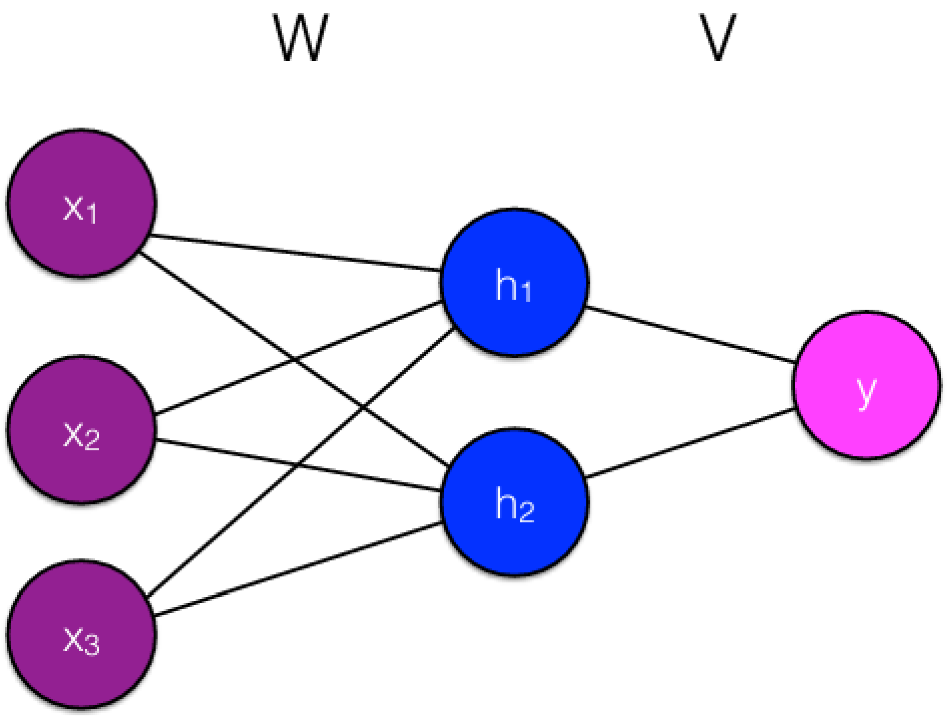


$$\hat{y} = \sigma \left[V_1 \left(\sigma \left(\sum_{i}^{F} x_i W_{i,1} \right) \right) + V_2 \left(\sigma \left(\sum_{i}^{F} x_i W_{i,2} \right) \right) \right]$$

$$\hat{y} = \sigma \left[V_1 \underbrace{\left(\sigma \left(\sum_{i}^{F} x_i W_{i,1} \right) \right)}_{h_1} + V_2 \underbrace{\left(\sigma \left(\sum_{i}^{F} x_i W_{i,2} \right) \right)}_{h_2} \right]$$

This is differentiable via backpropagation

Backpropagation: Given training samples of <x,y> pairs, we can use stochastic gradient descent to find the values of W and V that minimize the loss.



Neural networks are a series of functions chained together

$$xW \rightarrow \sigma(xW) \rightarrow \sigma(xW)V \rightarrow \sigma(\sigma(xW)V)$$

The loss is another function chained on top

$$\log(\sigma(\sigma(xW)V))$$

Chain rule

$$\frac{\partial}{\partial V}\log\left(\sigma\left(\sigma\left(xW\right)V\right)\right)$$

$$= \frac{\partial \log\left(\sigma\left(\sigma\left(xW\right)V\right)\right)}{\partial \sigma\left(\sigma\left(xW\right)V\right)} \frac{\partial \sigma\left(\sigma\left(xW\right)V\right)}{\partial \sigma\left(xW\right)V} \frac{\partial \sigma\left(xW\right)V}{\partial V}$$

$$= \overbrace{\frac{\partial \log\left(\sigma\left(hV\right)\right)}{\partial \sigma\left(hV\right)}}^{A} \overbrace{\frac{\partial \sigma\left(hV\right)}{\partial hV}}^{B} \overbrace{\frac{\partial hV}{\partial V}}^{C}$$

$$= \overbrace{\frac{1}{\sigma\left(hV\right)}}^{A} \times \overbrace{\sigma\left(hV\right) \times \left(1-\sigma\left(hV\right)\right)}^{B} \times \overbrace{h}^{C}$$

$$= \left(1-\sigma\left(hV\right)\right)h$$

$$= \left(1-\hat{y}\right)h$$

Algorithm 2 Logistic regression stochastic gradient descent

Data: training data $x \in \mathbb{R}^F, y \in \{0,1\}$
$\beta = 0^F$
while not converged **do**
  for $i = 1$ to N **do**
    $\boxed{\beta_{t+1} = \beta_t + \alpha\left(y_i - \hat{p}(x_i)\right)x_i}$
  end for
end while

Backpropagation

- Forward and backward propagation
  - Compute value/gradient of each node with respect to previous nodes
- Good news is that modern automatic differentiation tools do this all for you!
- Deep learning nowadays is like modular programming

```
class Feedforward(torch.nn.Module):
        def __init__(self, input_size, hidden_size):
            super(Feedforward, self).__init__()
            self.input_size = input_size
            self.hidden_size  = hidden_size
            self.fc1 = torch.nn.Linear(self.input_size,
self.hidden_size)
            self.relu = torch.nn.ReLU()
            self.fc2 = torch.nn.Linear(self.hidden_size, 1)
            self.sigmoid = torch.nn.Sigmoid()


        def forward(self, x):
            hidden = self.fc1(x)
            relu = self.relu(hidden)
            output = self.fc2(relu)
            output = self.sigmoid(output)
            return output
```

xW → $\sigma(xW)$ → $\sigma(xW)V$ → $\sigma(\sigma(xW)V)$

```python
class Feedforward(torch.nn.Module):
        def __init__(self, input_size, hidden_size):
            super(Feedforward, self).__init__()
            self.input_size = input_size
            self.hidden_size  = hidden_size
            self.fc1 = torch.nn.Linear(self.input_size, 
self.hidden_size)
            self.relu = torch.nn.ReLU()
            self.fc2 = torch.nn.Linear(self.hidden_size, 1)
            self.sigmoid = torch.nn.Sigmoid()


        def forward(self, x):
            hidden = self.fc1(x)
            relu = self.relu(hidden)
            output = self.fc2(relu)
            output = self.sigmoid(output)
            return output
```

```python
model = Feedforward(2, 10)
criterion = torch.nn.BCELoss()
optimizer = torch.optim.SGD(model.parameters(), lr = 0.01)
```

```
model.eval()
y_pred = model(x_test)
before_train = criterion(y_pred.squeeze(), y_test)
print('Test loss before training' , before_train.item())

model.train()
epoch = 20

for epoch in range(epoch):

    optimizer.zero_grad()

    # Forward pass
    y_pred = model(x_train)

    # Compute Loss
    loss = criterion(y_pred.squeeze(), y_train)

    print('Epoch {}: train loss: {}'.format(epoch, loss.item()))

    # Backward pass
    loss.backward()
    optimizer.step()
```

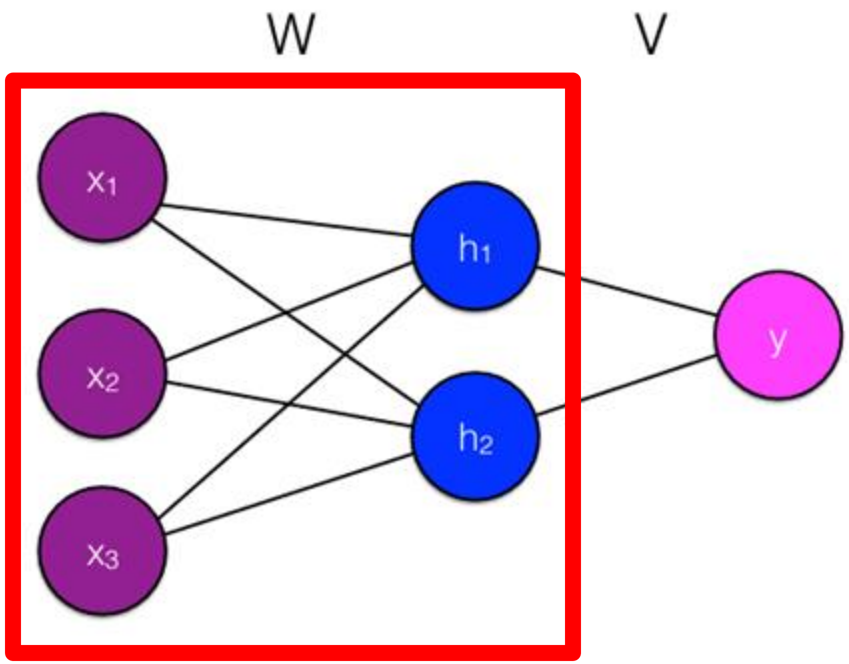
```
model.eval()
y_pred = model(x_test)
after_train = criterion(y_pred.squeeze(), y_test)
print('Test loss after Training' , after_train.item())
```

$$\log\left(\sigma\left(\sigma\left(xW\right)V\right)\right)$$

Other tricks in neural network training

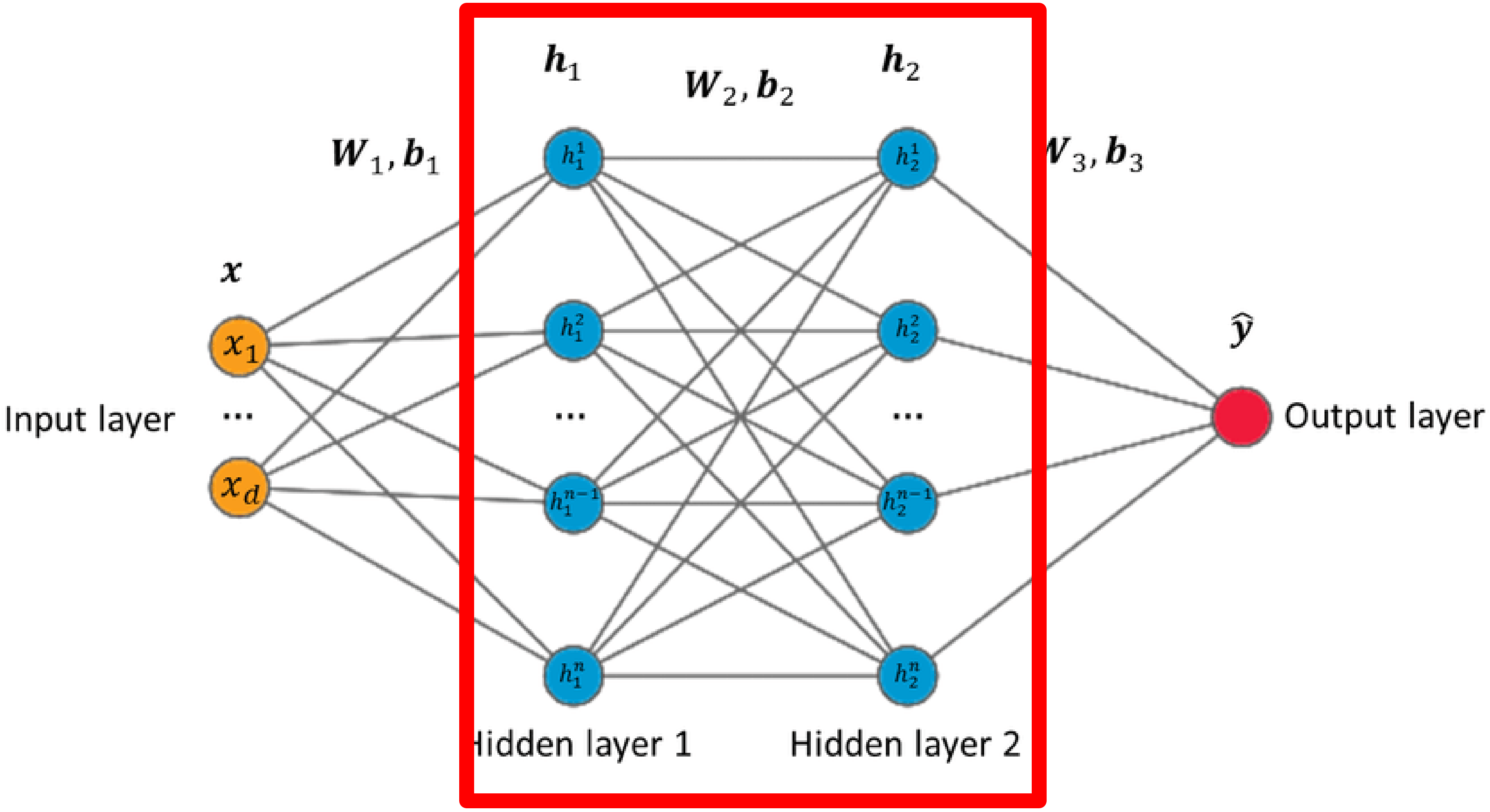
- Avoid overfitting with dropout
- Average/max/min pooling
- Smart initialization
- Adaptive learning rates than SGD
- Gradient clipping
- Early stopping with validation set
- Hyper-parameter tuning
- …

Feedforward Neural Network (i.e., Single-layer Perceptron)

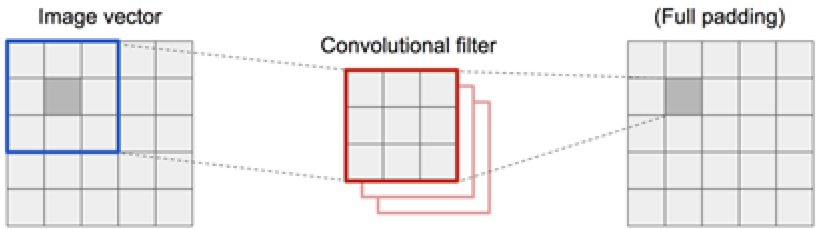


Feedforward Neural Network
(i.e., Two-layer Perceptron)

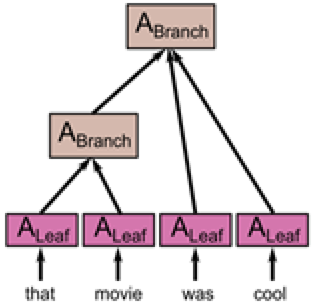
HW0



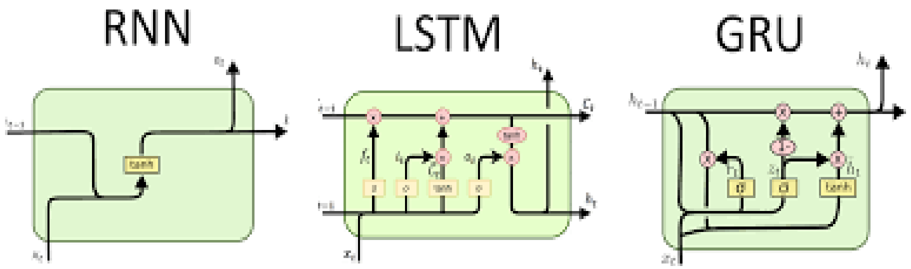
Other neural network models



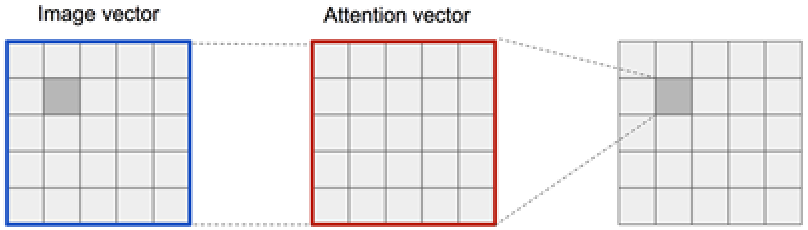
Convolution NN



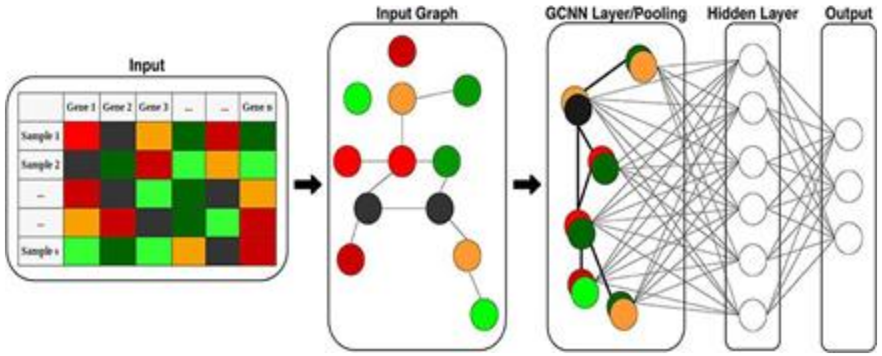
Recursive NN



Recurrent-NN/LSTM/GRU

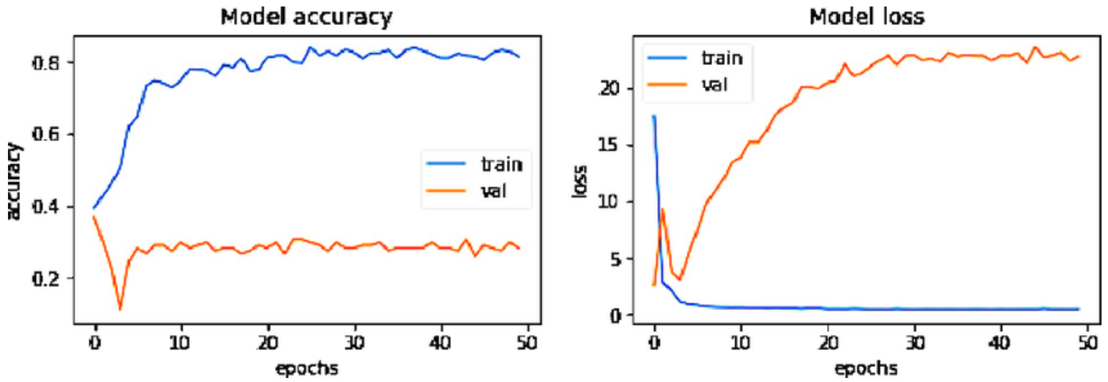


Self-attention / Transformers

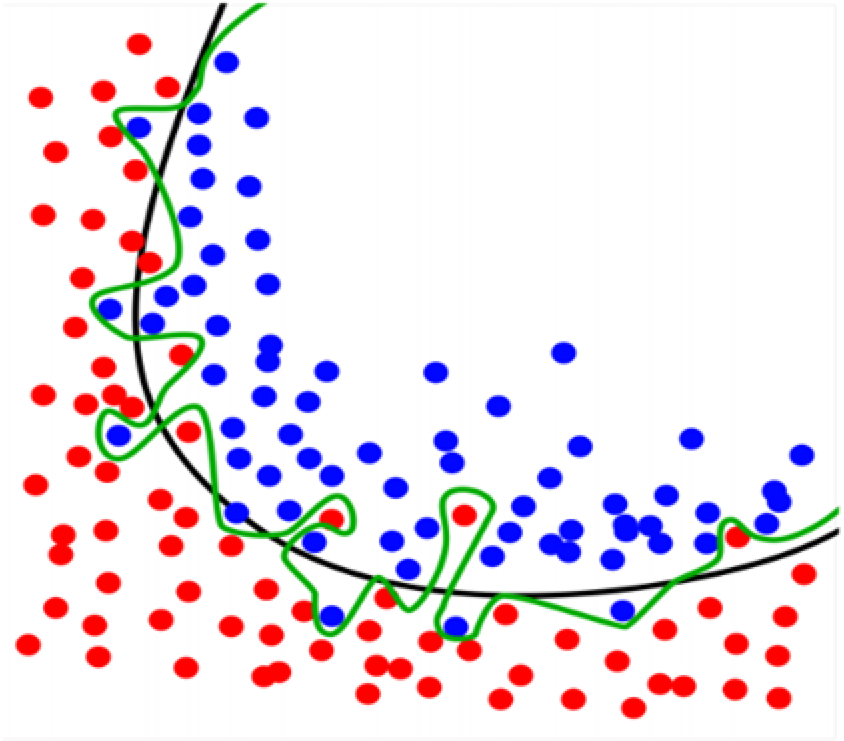


Graph Convolutional / Neural Network

Overfitting

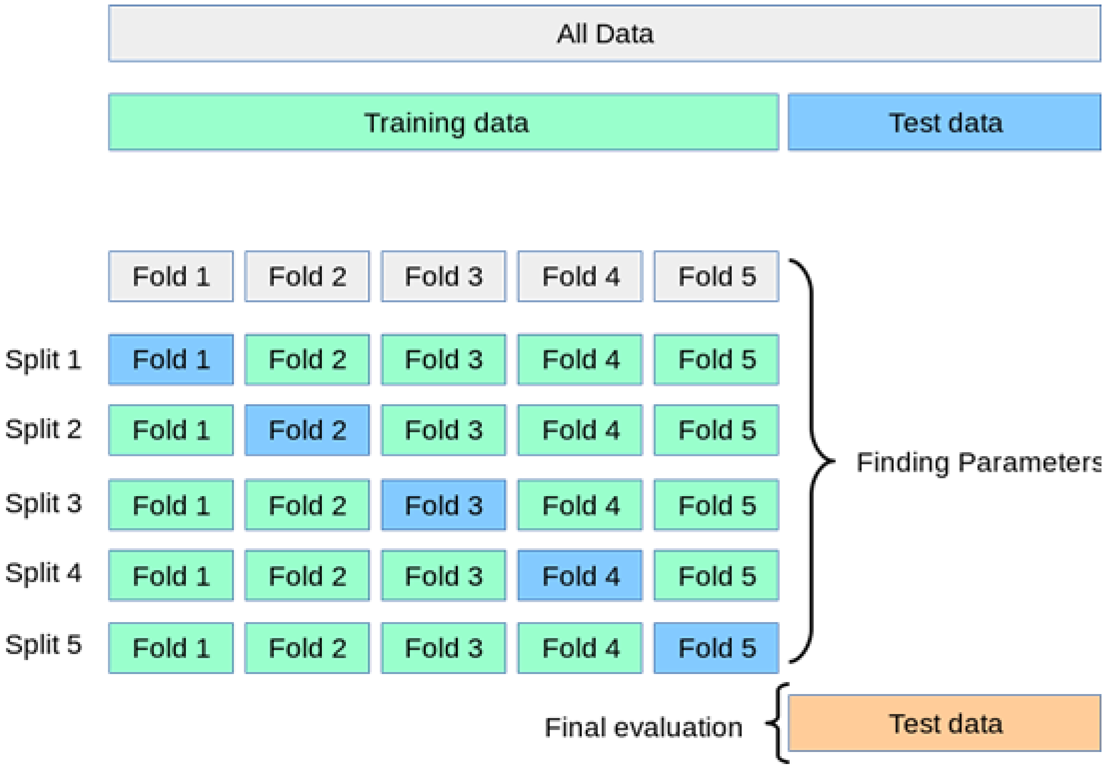


- A model that perfectly match the training data that has a problem
- It will also **overfit** to the data, modeling noise
  - A random word that perfectly predicts y (it happens to only occur in one class) will get a very high weight.
  - Failing to generalize to a test set without this word.
- A good model should be able to generalize



Cross validation

- Break up “training” data into 5 folds
- For each fold
  - Choose the fold as a temporary test set
  - Train on 5 folds, compute performance on test fold
- Report average performance of the 5 runs
- Find the best parameters



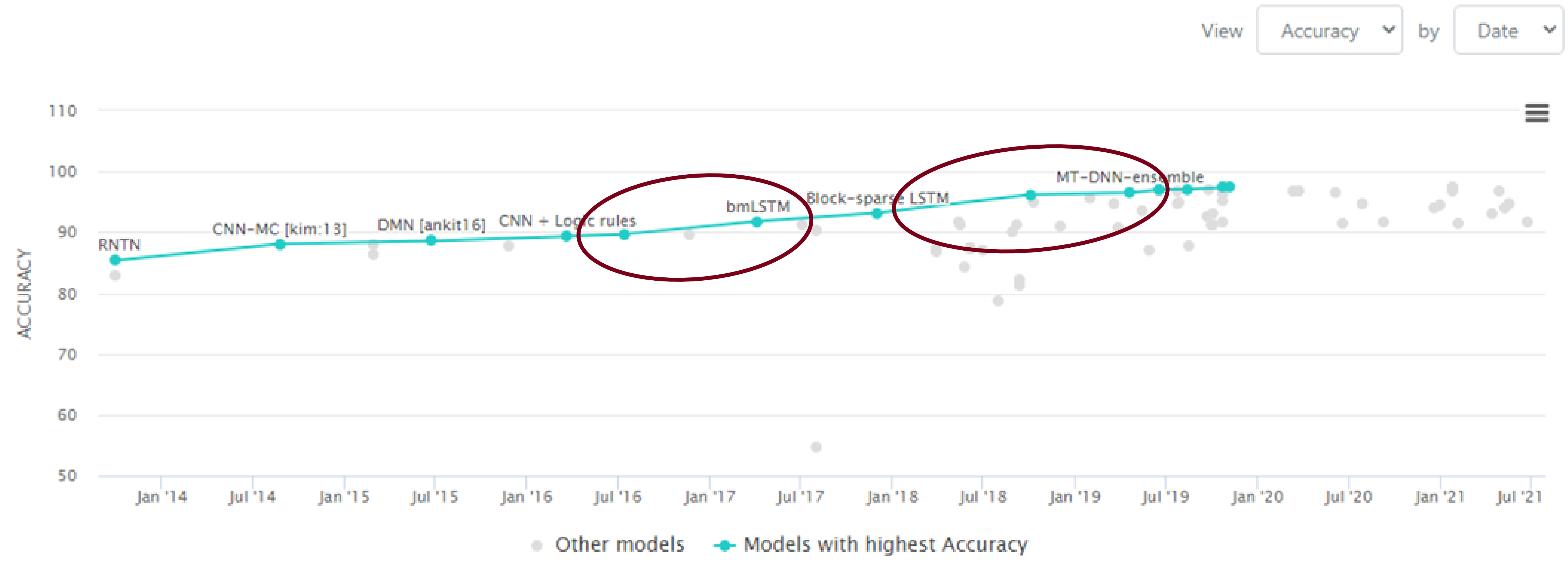
https://scikit-learn.org/stable/modules/cross_validation.html

State of the Art

https://paperswithcode.com/

Sentiment Analysis on SST-2 Binary classification

Leaderboard Dataset



Rank	Model	Accuracy	Paper	Code	Result	Year	Tags
1	**SMART-RoBERTa Large**	97.5	SMART: Robust and Efficient Fine-Tuning for Pre-trained Natural Language Models through Principled Regularized Optimization			2019	Transformer
2	**T5-3B**	97.4	Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer			2019	Transformer
3	**MUPPET Roberta Large**	97.4	Muppet: Massive Multi-task Representations with Pre-Finetuning			2021	
4	**ALBERT**	97.1	ALBERT: A Lite BERT for Self-supervised Learning of Language Representations			2019	Transformer
5	**T5-11B**	97.1	Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer			2019	Transformer
6	**StructBERTRoBERTa ensemble**	97.1	StructBERT: Incorporating Language Structures into Pre-training for Deep Language Understanding			2019	Transformer
7	**XLNet** (single model)	97	XLNet: Generalized Autoregressive Pretraining for Language Understanding			2019	Transformer
8	**ELECTRA**	96.9	ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators			2020	
9	**EFL**	96.9	Entailment as Few-Shot Learner			2021	Transformer
10	**XLNet-Large** (ensemble)	96.8	XLNet: Generalized Autoregressive Pretraining for Language Understanding			2019	Transformer
11	**RoBERTa**	96.7	RoBERTa: A Robustly Optimized BERT Pretraining Approach			2019	Transformer

Robustness of Neural Classifiers

Test case	Expected	Predicted	Pass?
A Testing **Negation** with ***MFT*** Labels: negative, positive, neutral			
`Template: I {NEGATION} {POS_VERB} the {THING}.`			
I can't say I recommend the food.	neg	pos	X
I didn't love the flight.	neg	neutral	X
...			
Failure rate = 76.4%			

Ribeiro et al., Beyond Accuracy: Behavioral Testing of NLP Models with CheckList, ACL 2020

Robustness of Neural Classifiers

Test case	Expected	Predicted	Pass?
B Testing **NER** with ***INV*** Same pred. (inv) after removals / additions			
@AmericanAir thank you we got on a different flight to [Chicago → Dallas].	inv	pos neutral	X
@VirginAmerica I can't lose my luggage, moving to [Brazil → Turkey] soon, ugh.	inv	neutral neg	X
...			
Failure rate = 20.8%			

Ribeiro et al., Beyond Accuracy: Behavioral Testing of NLP Models with CheckList, ACL 2020

Robustness of Neural Classifiers

Test case	Expected	Predicted	Pass?
Testing **Vocabulary** with ***DIR***	Sentiment monotonic decreasing (↓)		
@AmericanAir service wasn't great. You are lame.	↓	neg → neutral	X
@JetBlue why won't YOU help them?! Ugh. I dread you.	↓	neg → neutral	X
...			
		Failure rate = 34.6%	

Ribeiro et al., Beyond Accuracy: Behavioral Testing of NLP Models with CheckList, ACL 2020

Rank	Model	Accuracy	Paper	Code	Result	Year	Tags
1	**SMART-RoBERTa Large**	97.5	SMART: Robust and Efficient Fine-Tuning for Pre-trained Natural Language Models through Principled Regularized Optimization			2019	Transformer
2	**T5-3B**	97.4	Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer			2019	Transformer
3	**MUPPET Roberta Large**	97.4	Muppet: Massive Multi-task Representations with Pre-Finetuning			2021	
4	**ALBERT**	97.1	ALBERT: A Lite BERT for Self-supervised Learning of Language Representations			2019	Transformer
5	**T5-11B**	97.1	Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer			2019	Transformer
6	**StructBERTRoBERTa ensemble**	97.1	StructBERT: Incorporating Language Structures into Pre-training for Deep Language Understanding			2019	Transformer
7	**XLNet** (single model)	97	XLNet: Generalized Autoregressive Pretraining for Language Understanding			2019	Transformer
8	**ELECTRA**	96.9	ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators			2020	
9	**EFL**	96.9	Entailment as Few-Shot Learner			2021	Transformer
10	**XLNet-Large** (ensemble)	96.8	XLNet: Generalized Autoregressive Pretraining for Language Understanding			2019	Transformer
11	**RoBERTa**	96.7	RoBERTa: A Robustly Optimized BERT Pretraining Approach			2019	Transformer

Interpretability: why? learning dataset, not task

Human: Polite

BERT: Polite

I will understand if you decline, but would very much like you to accept. May I nominate you?

Human

BERT

Both

Hayati et al., Does BERT Learn as Humans Perceive? Understanding Linguistic Styles through Lexica

Dataset Characterization

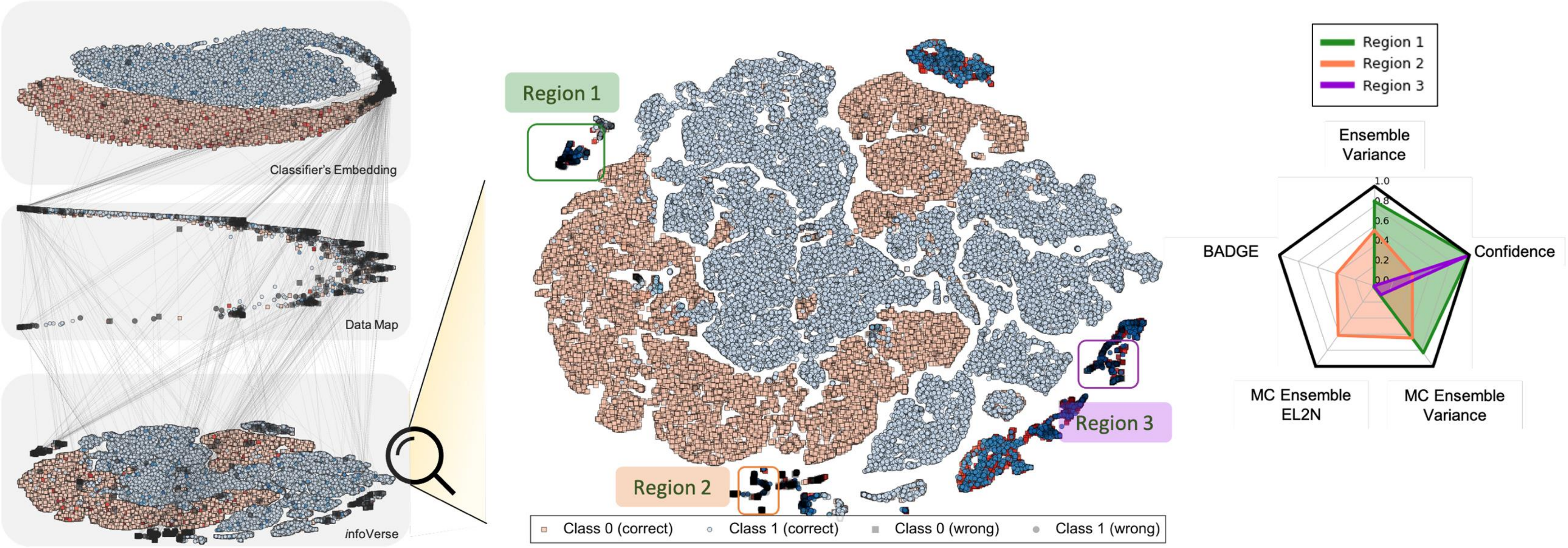
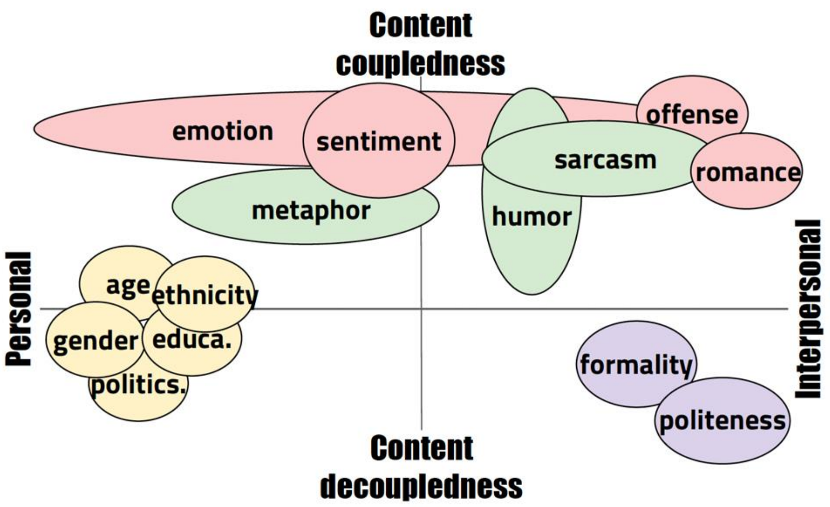


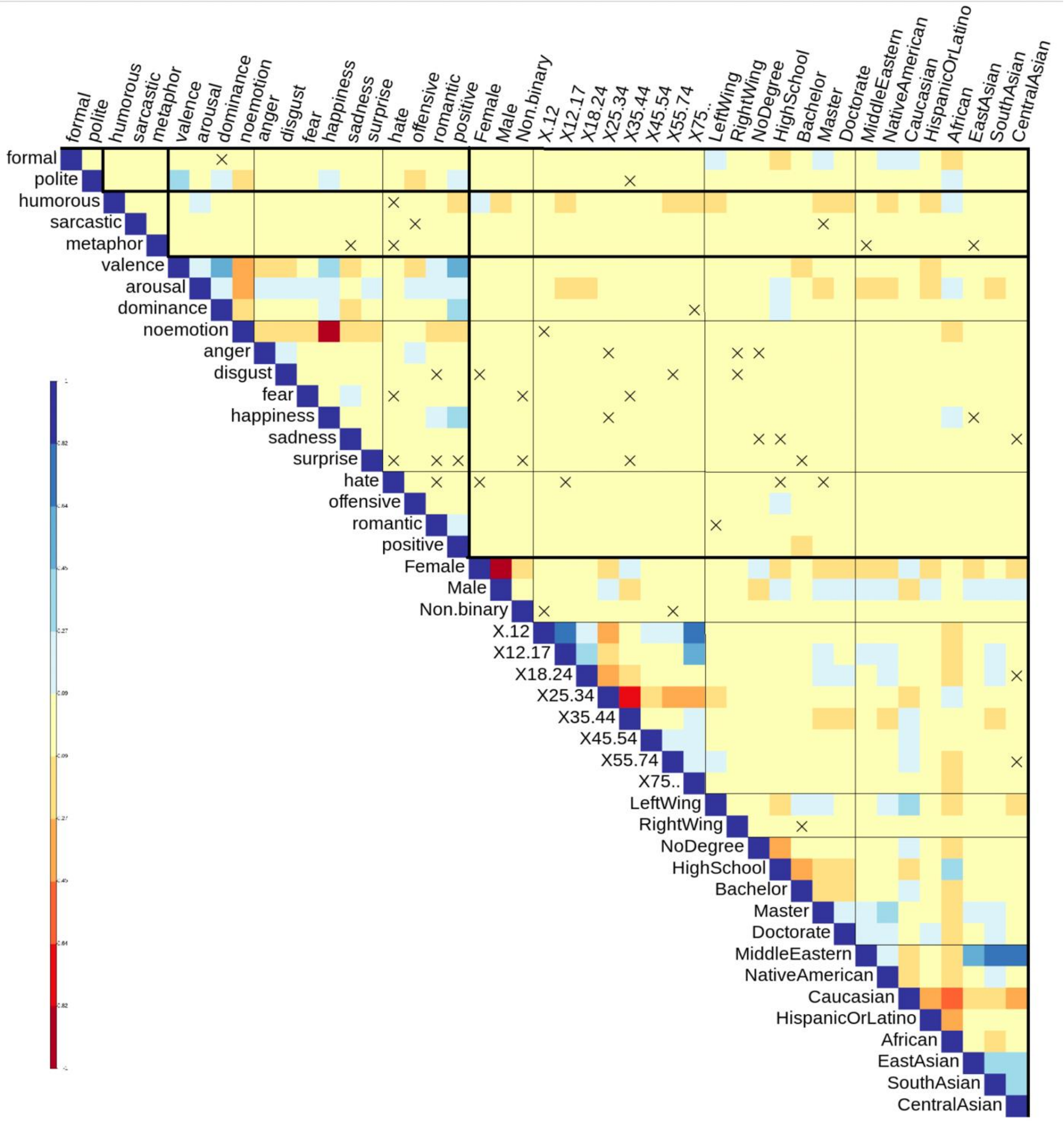
Figure 20: infoVerse (bottom left) on SST-2 along with other feature spaces: classifier embedding (top left) and *data map* (Swayamdipta et al., 2020) (middle left). (middle) Zoomed version of infoVerse is presented. (right) Score distribution of each wrong region characterized by infoVerse.

Kim et al., infoVerse: A Universal Framework for Dataset Characterization with Multidimensional Meta-information, ACL 2023

Cross-style Analysis

Groups	Styles
INTERPERSONAL	Formality, Politeness
FIGURATIVE	Humor, Sarcasm, Metaphor
AFFECTIVE	Emotion, Offense, Romance, Sentiment
PERSONAL	Age, Ethnicity, Gender, Education level, Country, Political view





Kang & Hovy , "Style is NOT a single variable: Case Studies for Cross-Stylistic Language Understanding", ACL 2021 (Oral)

Run yourself

https://huggingface.co/datasets/sst2

Summary

- Various applications using sentiment analysis in political and social sciences, stock market prediction, advertising, etc.
- Sentiment of text is reflection of the speaker's private state, which is hardly *observable*.
- Lexicon dictionaries have limitations, because sentiment is *contextual*
- Sentiment + X
- Modern deep representations perform better but are hard to *interpret*, and easy to be *biased* to the dataset
- 97.5 accuracy on SST2, but poor *robustness* in practice

Questions

- Is there any way to take advantages from both the classical dictionary based method and modern neural model?
- How can we evaluate and improve robustness of the model? How can we collect even more challenging samples that the current best model can’t predict well?
- How can we make black-box deep learning models to be more interpretable?
- Is benchmarking/leader-boarding a good practice for evaluation? If not, what is the solution?