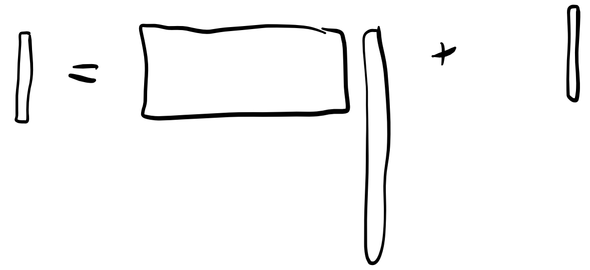
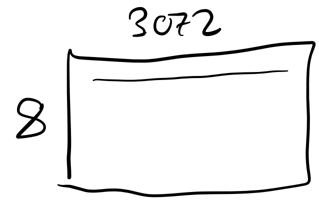
ML3 - Practicalities

(class scores) (image)

$$\hat{\vec{y}} = W \vec{x} + b$$

10×1 $\quad 10 \times 3072$ $\quad 3072 \times 1$ $\quad 10 \times 1$

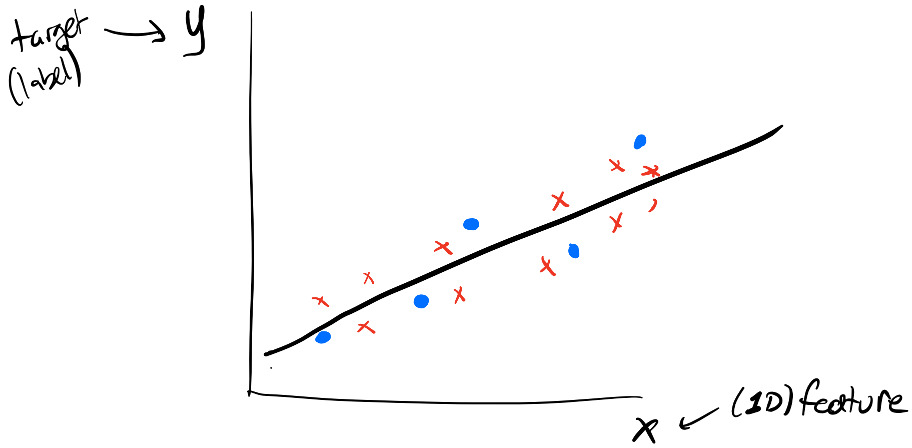




30730

ML3: Generalization

Example problem setting: Regression



Big Assumption: 1. Data was drawn from some distribution $P(x, y)$

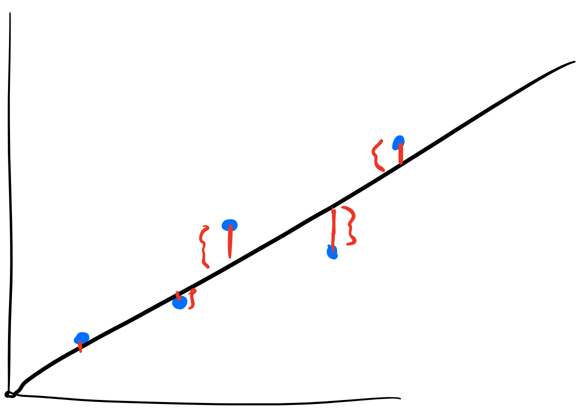
2. unseen data is drawn from the same distribution!

In other words, correlation doesn't imply causation, but "past" correlations are indicative of "future" correlations.

Risk (loss, cost, ...)

"fit

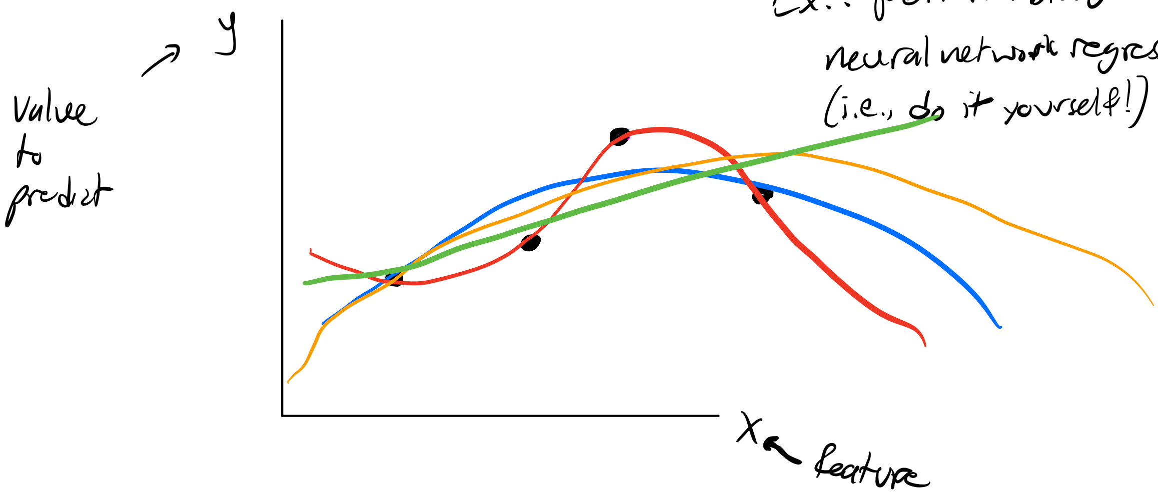
measure of model badness"

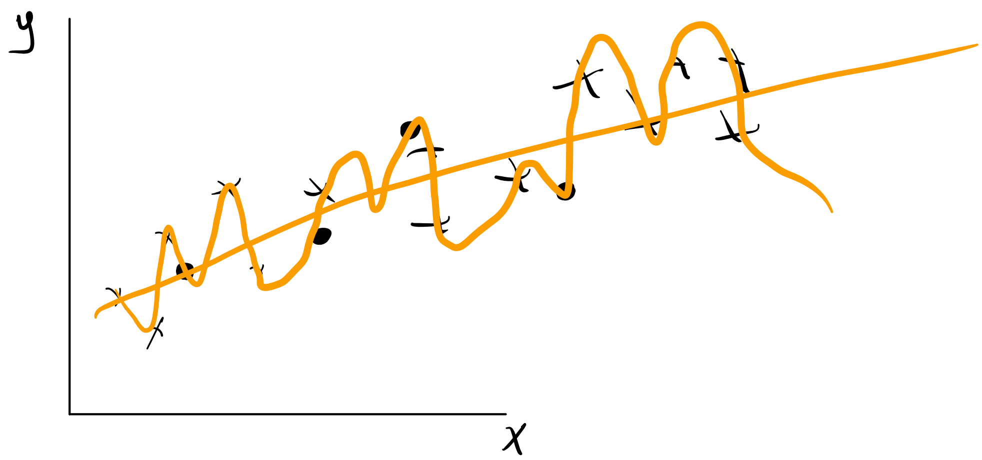


Occam's Razor: Use the simplest possible explanation for the data.

Task: regression (not necessarily linear)

Ex.: perform biological neural network regression (i.e., do it yourself!)





Overfitting

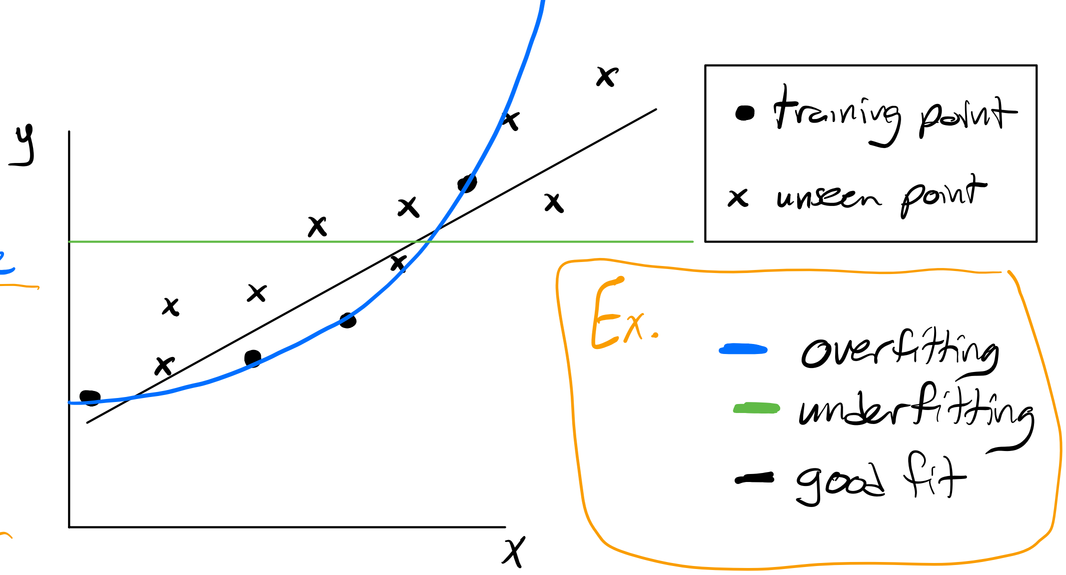
Goal: make predictions for unseen data.

overfitting:

model mistakes noise for signal

underfitting

model does not fit fit signal

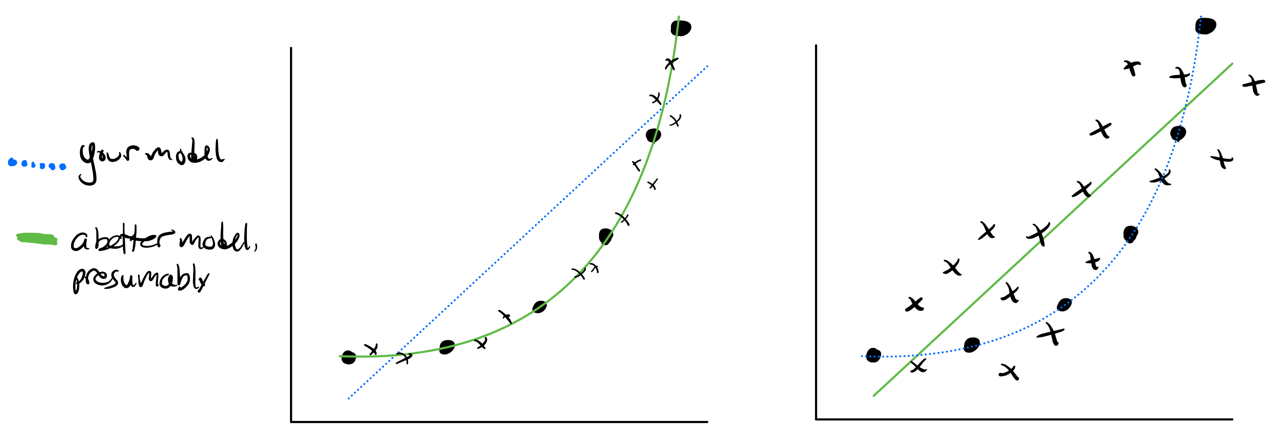


"Simplest possible explanation for the data"
needs to take into account noise/sampling error

Bias vs. Variance

Bias: modeling error - your model can't fit the underlying phenomenon

Variance: sampling error - your model mistakes noise for the underlying phenomenon



Ex: In each plot, is "your model" bad mainly because of bias, or variance?

Tools in the fight against overfitting

How can we fit a model and convince ourselves it isn't overfitting?

1. Withhold data!
2. Use a simple model regardless

1. Data Splits

Idea: Hold back some training data

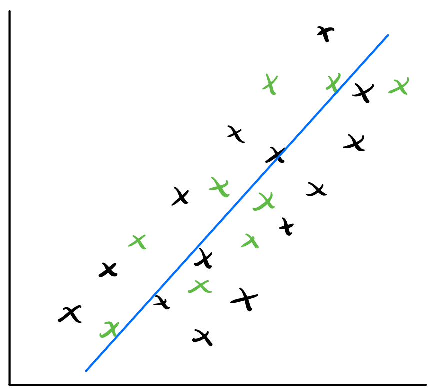
Train on ⊗, validate on ⊗

x training set

x validation set

Scenario: accuracy is measured as distance from x to

All available training data:



Accuracy on training set	Accuracy on validation set: Bad	Accuracy on validation set: Good
Bad	underfit	lucky cheating
Good	overfit	

Pseudocode for MLBot 1.0:

→ Make modeling assumptions
While True:
    train
    validate   not unseen!
    if train ^^ val == good
        break
    else:
        tweak modeling assumptions

Problem?

Solution?

Data split best practice:

Labeled data	Train	Val	Test

What %? Depends on size of data and amount of noise.

Smaller → higher variance

larger → less data for other splits

For small data, take full advantage of as much data as possible:

Val$_1$	Val$_2$	Val$_3$	...	Val$_k$	Test

"k-fold cross-validation":

train on each subset of k-1 chunks, val on the last

avg val accuracy across all k trials

+ better training, lower variance val accuracy

- need to train k times

"leave-one-out cross-validation":

$k = n$

Tools in the fight against overfitting

2. Regularization: build a "simplicity prior" into your model.

Ex: "weight decay":

$$c(w,b,x,y) = (Wx+b-\hat{y}) + \underbrace{\lambda W^T W}_{\text{regularization term}}$$