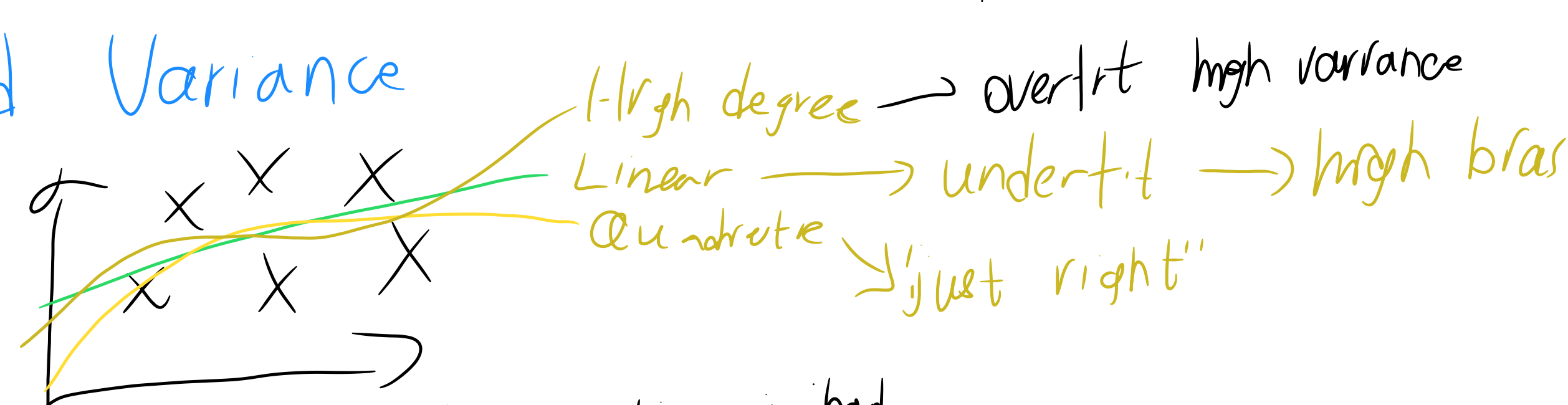


L 8

- Bias and variance
- Regularization
- Train/dev/test splits
- Model selection / cross-validation

Bias and Variance



Bias: original assumption is bad
High Variance: a slightly different train set would create completely different model

Too many features \rightarrow overfit
Too few features \rightarrow not enough

Regularization prevent overfitting (with support vector machine)

$$\min_{\theta} \frac{1}{2} \sum_{i=1}^m \|y^{(i)} - \theta^T x^{(i)}\|^2 + \frac{\lambda}{2} \|\theta\|^2 \rightarrow \text{for linear regression}$$

choose λ to prevent overfitting
 \rightarrow prevent high variability in general

$$\arg \max_{\theta} \sum_{i=1}^n \log p(y^{(i)} | x^{(i)}, \theta) - \lambda \|\theta\|^2 \rightarrow \text{for classification}$$

Why doesn't SVM overfit?

$$\text{somehow } \min \|\theta\|^2 \rightarrow \lambda \|\theta\|^2$$

have effect of regularization

Logistic regression with regularization better than naive bayes in text classification

Note preprocess data to make x_i all on same scale

$$\text{So } \lambda \|\theta\|^2 \text{ is justified by } \frac{x_i - \mu_i}{\sigma_i}$$

SB training set = $\{x^{(i)}, y^{(i)}\}$

$$P(\theta | S) = \frac{P(S | \theta) P(\theta)}{P(S)}$$

$$\arg \max_{\theta} P(\theta | S) = \arg \max_{\theta} P(S | \theta) P(\theta)$$

$$= \arg \max_{\theta} \left(\prod_{i=1}^m p(y^{(i)} | x^{(i)}, \theta) \right) P(\theta)$$

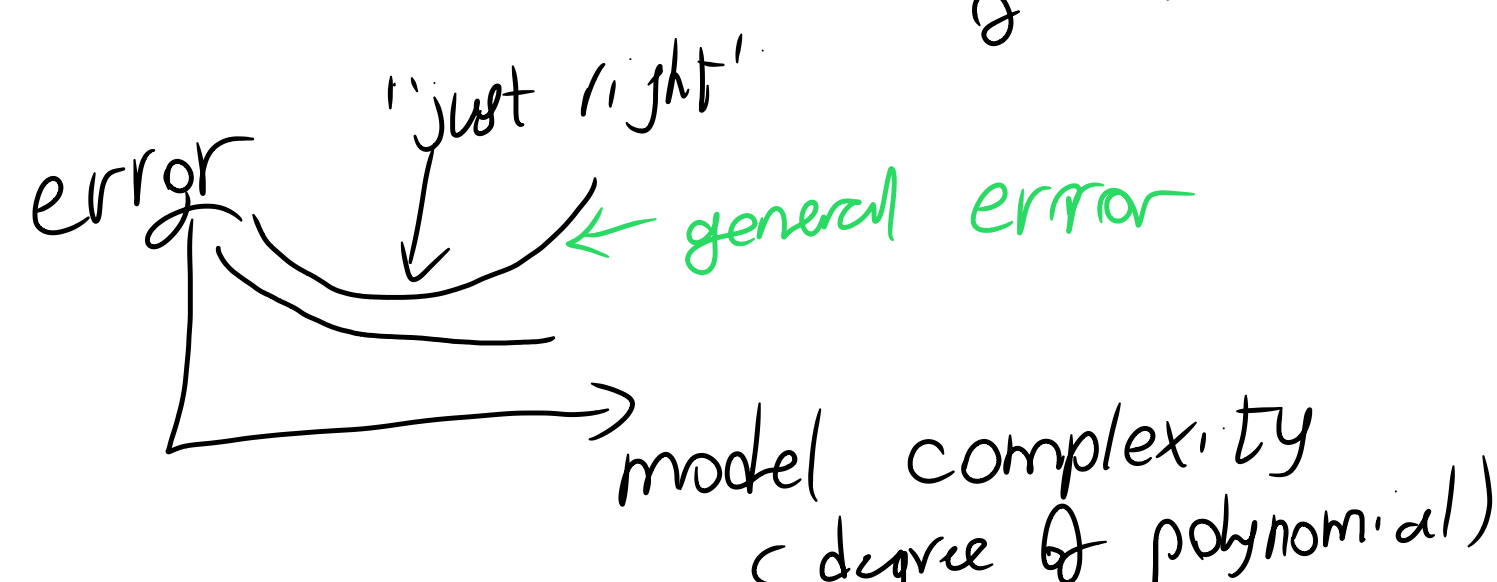
Assume $P(\theta) = \theta \sim N(0, I)$

$$P(\theta) = \frac{1}{(2\pi)^{1/2}} e^{-\frac{1}{2} \theta^T \theta}$$

Frequentist $P(S | \theta)$ - MLE origin of regularization

Bayesian prior - $P(\theta)$ assumption

prior-distribution $\max_{\theta} P(\theta | S)$ - map \rightarrow adding regularization



Train/dev/test set

eg 10,000 examples

order of polynomial

or choose λ

or choose λ

or C

simple cross-validation set

- 1 Split $S \rightarrow S_{train}, S_{dev}, S_{test}$
- 2 Train each model (optimal for degree of polynomial) on S_{train}
- 3 Measure error on S_{dev} , Pick best performance
- 4 Optional evaluate algorithm on S_{test}
 \rightarrow mainly for publication

Historical S if S small

70% train, 30% test
60% train, 20% dev, 20% test

Modern perspective

$|S| = 10^8$ very large

train 6×10^7

dev 2×10^7

test 2×10^7

\rightarrow way too much unless to measure performance increase very small
eg. 90% vs 90.1%, then need more train data
but normally 90% train

Improve on the dev step, by optimizing on dev set, can use feedback development

Do not incorporate test set performance into model

Suppose small set

$m = 100$

70 S_{train} , 30 S_{dev}

Seems S_{train} too small

k-fold cross-validation (only S small)

split into k sets, eg $100 = 5$

For degree $d = 1 \dots 5$ each set 20

For $k = 1 \dots 10$ degree of polynomial

Train (all k -portions) on $k-1$ pieces

Average of the error Test on remaining 1 piece

Optional after choosing degree Refit 100% of data

Better than simple CV since has more data

What if even smaller?

$m = 20$, then choose $k = m$

ml/earning.org

Feature selection

find smaller subset of features to avoid overfitting

start with $f = \emptyset$

- (1) Repeat add each feature v to f , and see which single feature add most important on dev set performance
- (2) Add the feature to f

eg x_1, x_5

Forward selection (\emptyset) $h(x) = \theta_0$

1 $\begin{bmatrix} \phi & x_1 \\ \phi & x_5 \end{bmatrix} \rightarrow f = \{x_1\}$

2 $\begin{bmatrix} x_1 & x_5 \\ x_2 & x_5 \end{bmatrix} \rightarrow f = \{x_1, x_5\}$

3 $\begin{bmatrix} x_1 & x_5 \\ x_2 & x_5 \end{bmatrix} \rightarrow f = \{x_1, x_5\}$

repeat to complete good features

also have backward selection which removes most irrelevant features