

# CS-GY 6923: Lecture 3

## Model Selection + Regularization + Bayesian Perspective

---

NYU Tandon School of Engineering, Prof. Christopher Musco

- Homework 1 due tonight.
- New lab will be released tonight, due next Thursday.
- Next problem set will be due a week after that.

## LOSS MINIMIZATION

Basic machine learning problem:

- Given model  $f_{\theta}$  and loss function  $L(f_{\theta})$ .
- Choose  $\theta^*$  to minimize  $L(f_{\theta})$ .

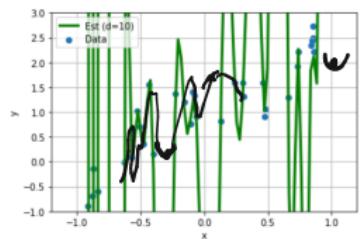
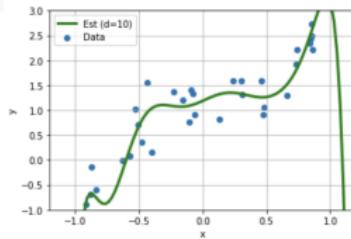
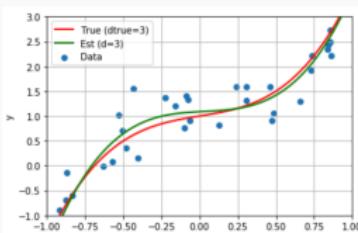
## MODEL SELECTION

Model selection problem:

- Given choice of many models  $f_{\theta_1}^{(1)}, f_{\theta_2}^{(2)}, \dots, f_{\theta_q}^{(q)}$ .
  - Choose  $\theta_1^*, \dots, \theta_q^*$  to minimize  $L(f_{\theta_1}^{(1)}), \dots, L(f_{\theta_q}^{(q)})$ .
  - Then choose the “best” model for our data.
-

## MODEL SELECTION EXAMPLE

Polynomial regression models with different degree. See `demo_polyfit.ipynb`.

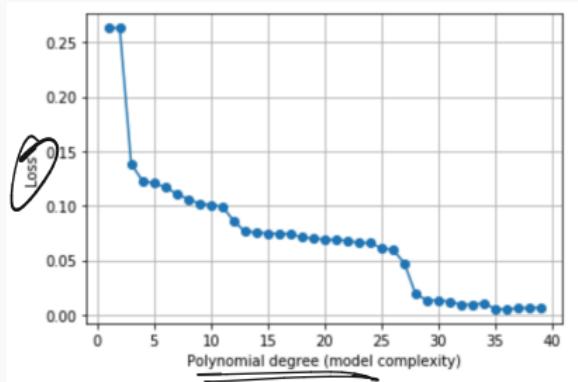


1 2 3

- Model  $f_{\theta_1}^{(1)}$ : all linear functions.
- Model  $f_{\theta_2}^{(2)}$ : all quadratic functions.
- Model  $f_{\theta_3}^{(3)}$ : all cubic functions.
- ...

## MODEL SELECTION

The more **complex** our model class (e.g., the higher degree we allow in polynomial regression) the better our loss:



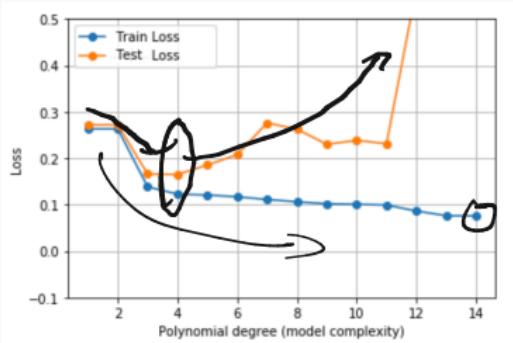
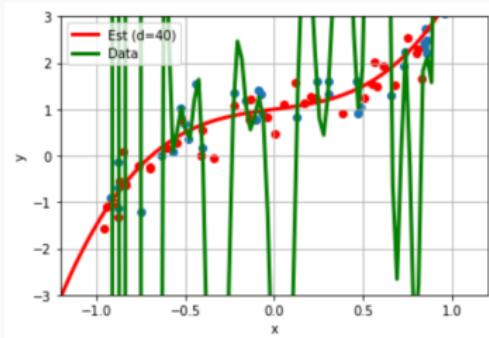
Training loss alone is not usually a good metric for model selection. Small loss does not imply generalization to new data.

**Main approach:** Evaluate model on fresh test data which was not used during training.

**Test/train split:**

- Given data set  $(X, y)$ , split into two sets  $(X_{\text{train}}, y_{\text{train}})$  and  $(X_{\text{test}}, y_{\text{test}})$ .
- Train  $q$  models  $f^{(1)}, \dots, f^{(q)}$  by finding parameters which minimize the loss on  $(X_{\text{train}}, y_{\text{train}})$ .
- Evaluate loss of each trained model on  $(X_{\text{test}}, y_{\text{test}})$ .

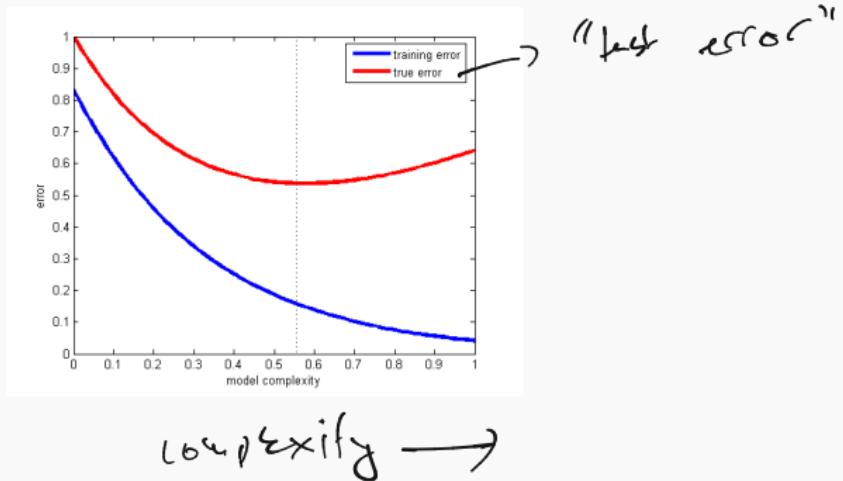
# GENERALIZATION



While train error always decreases, we eventually see test error increase with increasing model complexity.

# THE FUNDAMENTAL CURVE OF ML

The above trend is fairly representative of what we tend to see across the board:



## TRAIN-TEST INTUITION

Is “test error” the end goal though? Don’t we care about  
 “future” error?

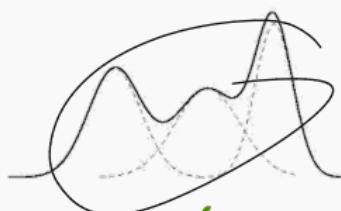
**Intuition:** Models which perform better on the test set will  
generalize better to future data.

**Goal:** Introduce a little bit of formalism to better understand  
what this means. What is “future” data?

# STATISTICAL LEARNING MODEL

## Statistical Learning Model:

- Assume each data example is randomly drawn from some distribution  $(x, y) \sim D$ .



E.g.  $x_1, \dots, x_d$  are Gaussian random variables with parameters

$$\mu_1, \sigma_1, \dots, \mu_d, \sigma_d.$$

This is not a simplifying assumption! The distribution could be arbitrarily complicated.

## Statistical Learning Model:

- Assume each data example is randomly drawn from some distribution  $(\underline{x}, \underline{y}) \sim \mathcal{D}$
- Define the **Risk** of a model/parameters:

$$\textcircled{R(f, \theta)} = \mathbb{E}_{(\underline{x}, \underline{y}) \sim \mathcal{D}} [L(f(\underline{x}, \underline{\theta}), \underline{y})]$$

here  $L$  is our loss function (e.g.  $L(z, y) = |z - y|$  or  $L(z, y) = (\underline{z} - \underline{y})^2$ ).

**Goal:** Find model  $\underline{f} \in \{f^{(1)}, \dots, f^{(q)}\}$  and parameter vector  $\underline{\theta}$  to minimize the  $\underline{R(f, \theta)}$ .

(Population) Risk:

$$R(f, \theta) = \mathbb{E}_{(x,y) \sim \mathcal{D}} [L(f(x, \theta), y)]$$

- Empirical Risk: Draw  $(x_1, y_1), \dots, (x_n, y_n) \sim \mathcal{D}$

$$\underline{R_E}(f, \theta) = \underbrace{\left( \frac{1}{n} \sum_{i=1}^n L(f(x_i, \theta), y_i) \right)}_{\text{test}}$$

$$(x_1, y_1), \dots, (x_n, y_n) \sim \mathcal{D}$$

 test

## EMPIRICAL RISK

$$X \sim \{1, \dots, 6\}$$

$$\underline{Y_1 \ Y_2 \ 0 \ Y_6 \ 0 \ 0}$$

$$\mathbb{E}[X] = \sum_{i=1}^6 \underbrace{\Pr[X=i]}_{\text{Probability}} \cdot i$$

For any fixed model  $f$  and parameters  $\theta$ ,

$$\mathbb{E}[R_E(f, \theta)] = \underline{R(f, \theta)}.$$

Only true if  $f$  and  $\theta$  are chosen *without looking at the data used to compute the empirical risk*.

$$\begin{aligned} & \mathbb{E}\left[\frac{1}{n} \sum_{i=1}^n L(f(x_i, \theta), y_i)\right] \\ &= \frac{1}{n} \sum_{i=1}^n \underbrace{\mathbb{E}[L(f(x_i, \theta), y_i)]}_{R_E(f, \theta)} = \frac{1}{n} \sum_{i=1}^n R(f, \theta) \\ &= R(f, \theta) \end{aligned}$$

## MODEL SELECTION

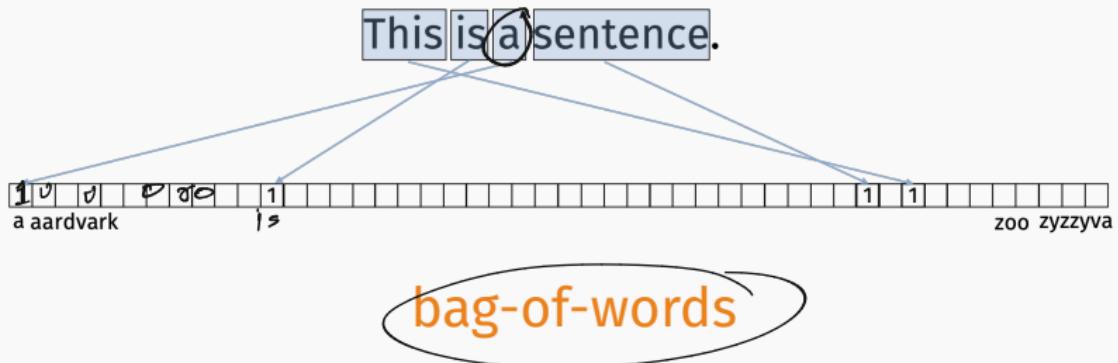
- Train  $q$  models  $(f^{(1)}, \theta_1^*), \dots, (f^{(q)}, \theta_q^*)$ .
- For each model, compute empirical risk  $R_E(f^{(i)}, \theta_i^*)$  using test data.
- Since we assume our original dataset was drawn independently from  $\mathcal{D}$ , so is the random test subset.

No matter how our models were trained or how complex they are,  $R_E(f^{(i)}, \theta_i^*)$  is an unbiased estimate of the true risk  $R(f^{(i)}, \theta_i^*)$  for every  $i$ . Can use it to distinguish between models.

## MODEL SELECTION EXAMPLE

## bag-of-words models and n-grams

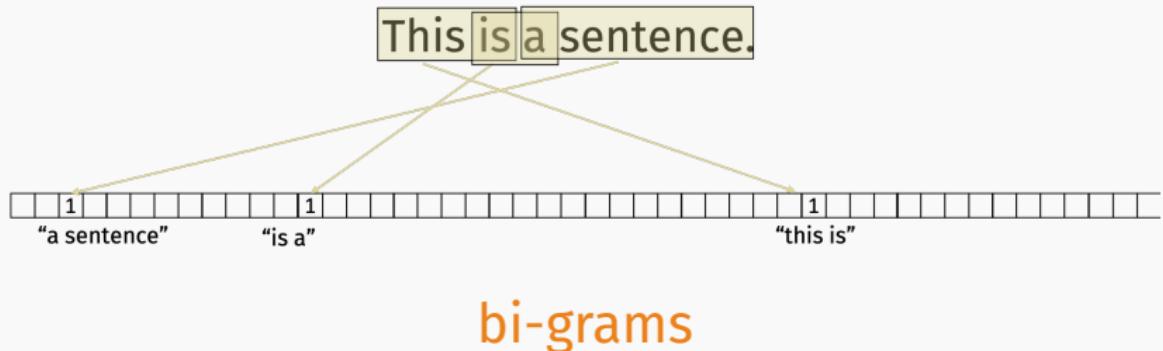
Common way to represent documents (emails, webpages, books) as numerical data. The ultimate example of 1-hot encoding.



## MODEL SELECTION EXAMPLE

### bag-of-words models and n-grams

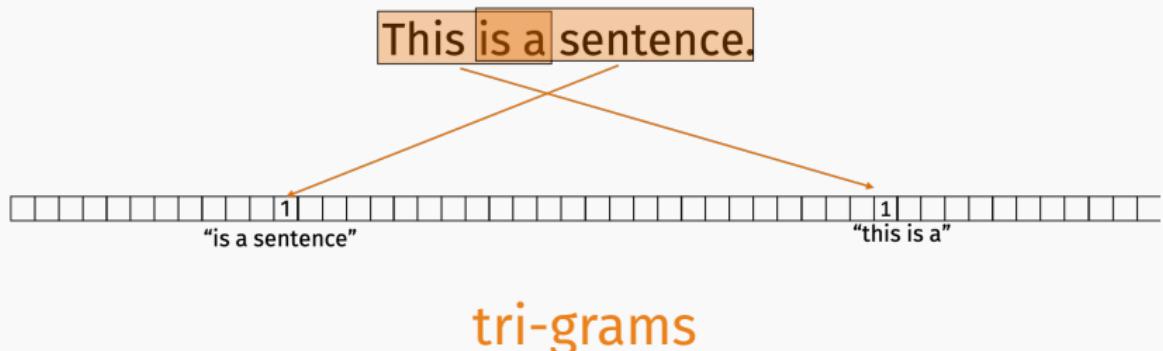
Common way to represent documents (emails, webpages, books) as numerical data. The ultimate example of 1-hot encoding.



## MODEL SELECTION EXAMPLE

### bag-of-words models and n-grams

Common way to represent documents (emails, webpages, books) as numerical data. The ultimate example of 1-hot encoding.



## MODEL SELECTION EXAMPLE

Models of increasing order:

- Model  $f_{\theta_1}^{(1)}$ : spam filter that looks at **single words**.
- Model  $f_{\theta_2}^{(2)}$ : spam filter that looks at **bi-grams**.
- Model  $f_{\theta_3}^{(3)}$ : spam filter that looks at **tri-grams**.
- ...

“interest”



“low interest”



“low interest loan”

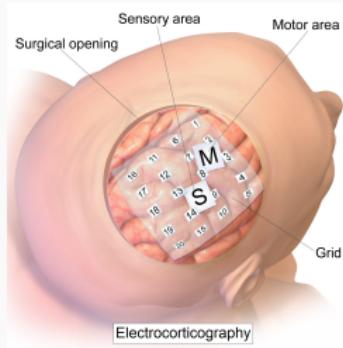


Increased length of **n-gram** means more expressive power.

## MODEL SELECTION EXAMPLE

### Electrocorticography ECoG (upcoming lab):

- Implant grid of electrodes on surface of the brain to measure electrical activity in different regions.

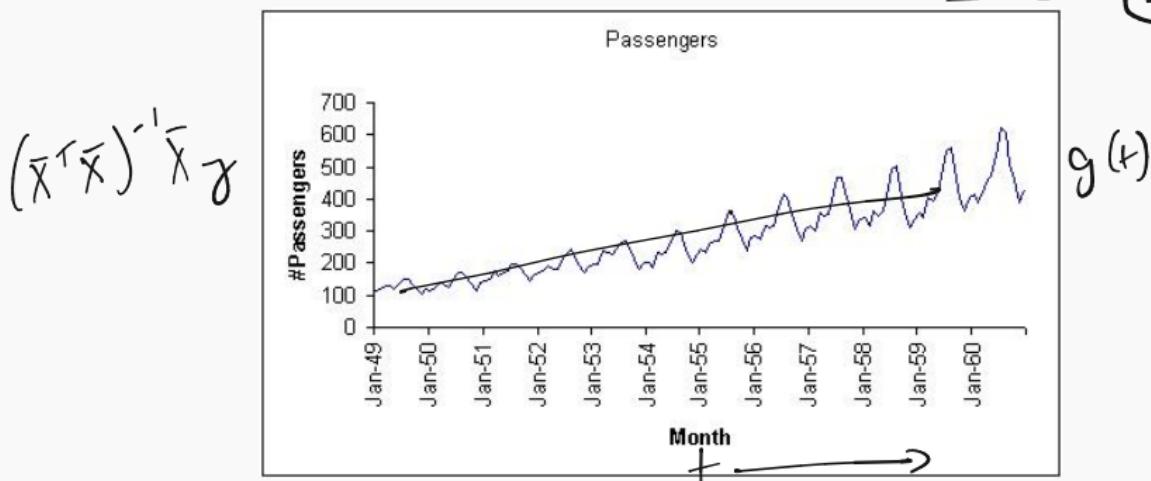


- Predict hand motion based on ECoG measurements.
- Model order:** predict movement at time  $t$  using brain signals at time  $\underline{t}, \underline{t-1}, \dots, \underline{t-q}$  for varying values of  $q$ .

## AUTOREGRESSIVE MODEL

Predicting time  $t$  based on a linear function of the signals at time  $t-1, \dots, t-d$  is not the same as fitting a line to the time series. It's much more expressive.

$$y = b + 10t + 1$$

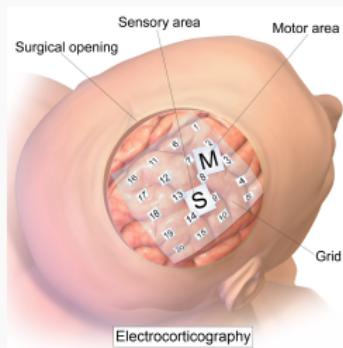


Predecessor of modern “recurrent neural networks”.

Electrocorticography ECoG lab:

Return:

12:20



First lab where computation actually matters (solving regression problems with  $\sim 40k$  examples,  $\sim 1500$  features)

Makes sense to test and debug code using a subset of the data.

Slight caveat: This is typically not how machine learning or scientific discovery works in practice!

Typical workflow:

- Train a class of models.
- Test.
- Adjust class of models.
- Test.
- Adjust class of models.
- Cont...

Final model implicitly depends on test set because performance on the test set guided how we changed our model.

Popularity of ML benchmarks and competitions leads to adaptivity at a massive scale.

11 Active Competitions

	<b>#DFDC</b> Deepfake Detection Challenge Identify videos with facial or voice manipulations <small>Featured · Code Competition · 2 months to go · video data, online video</small>	\$1,000,000 1,595 teams
	<b>Google QUEST Q&amp;A Labeling</b> Improving automated understanding of complex question answer content <small>Featured · Code Competition · 19 hours to go · text data, nlp</small>	\$25,000 1,559 teams
	<b>Real or Not? NLP with Disaster Tweets</b> Predict which Tweets are about real disasters and which ones are not <small>Getting Started · Ongoing · text data, binary classification</small>	\$10,000 2,657 teams
	<b>Bengali.AI Handwritten Grapheme Classification</b> Classify the components of handwritten Bengali <small>Research · Code Competition · a month to go · multiclass classification, image data</small>	\$10,000 1,194 teams

Kaggle (various competitions)



14,197,122 images, 21841 synsets indexed

[Explore](#) [Download](#) [Challenges](#) [Publications](#) [Updates](#) [About](#)

Not logged in. [Login](#) | [Signup](#)

Imagenet (image classification and categorization)

Is adaptivity a problem? Does it lead to over-fitting? How much? How can we prevent it? All current research.

REPORT

## The reusable holdout: Preserving validity in adaptive data analysis

Cynthia Dwork<sup>1,\*</sup>, Vitaly Feldman<sup>2,\*</sup>, Moritz Hardt<sup>3,\*</sup>, Toniann Pitassi<sup>4,\*</sup>, Omer Reingold<sup>5,\*</sup>, Aaron Roth<sup>6,\*</sup>

\* See all authors and affiliations

Science 07 Aug 2015:  
Vol. 349, Issue 6248, pp. 636-638  
DOI: 10.1126/science.aaa9375

### Do ImageNet Classifiers Generalize to ImageNet?

Benjamin Recht\*  
UC Berkeley

Rebecca Roelofs  
UC Berkeley

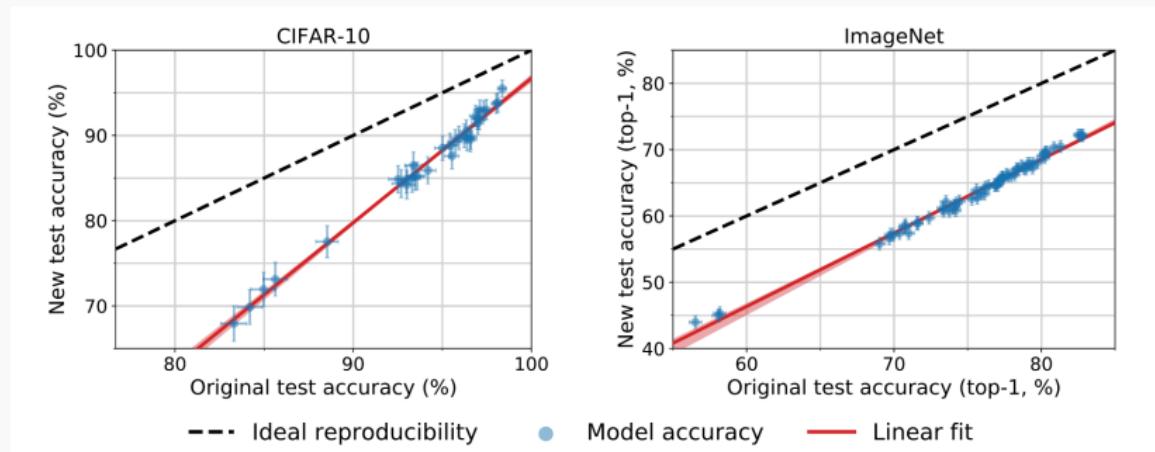
Ludwig Schmidt  
UC Berkeley

Vaishaal Shankar  
UC Berkeley

#### Abstract

We build new test sets for the CIFAR-10 and ImageNet datasets. Both benchmarks have been the focus of intense research for almost a decade, raising the danger of overfitting to excessively re-used test sets. By closely following the original dataset creation processes, we test to what extent current classification models generalize to new data. We evaluate a broad range of models and find accuracy drops of 3% – 15% on CIFAR-10 and 11% – 14% on ImageNet. However, accuracy gains on the original test sets translate to larger gains on the new test sets. Our results suggest that the accuracy drops are not caused by adaptivity, but by the models' inability to generalize to slightly "harder" images than those found in the original test sets.

## Do ImageNet Classifiers Generalized to ImageNet?



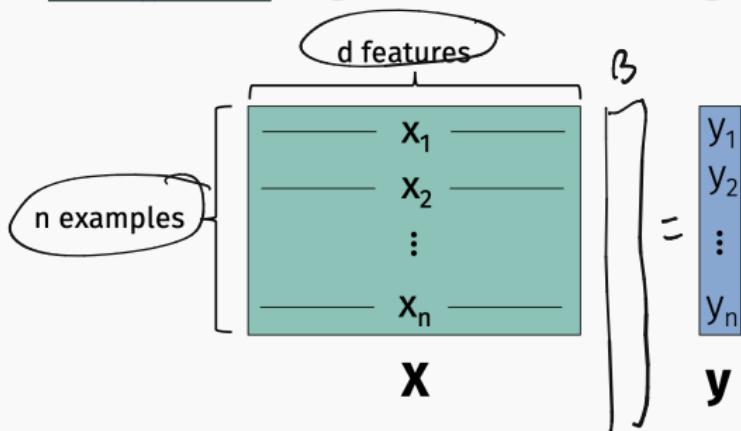
Interestingly, when comparing popular vision models on “fresh” data, while performance dropped across the board, the relative rank of model performance did not change significantly.

# REGULARIZATION

## OVER-PARAMETERIZED MODELS

In all the model selection examples we discussed we had full control over the complexity of the model: could range from underfitting to overfitting.

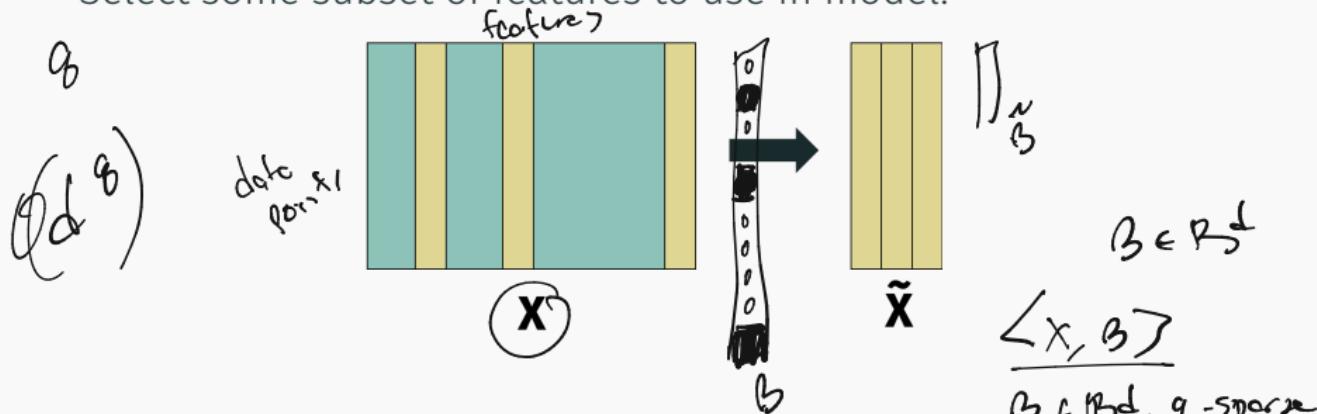
In practice, you often don't have this freedom. Even the most basic model might lead to overfitting.



**Example:** Linear regression model where  $d \geq n$ . Can always find  $\beta$  so that  $X\beta = y$  exactly.

## FEATURE SELECTION

Select some subset of features to use in model:



Filter method: Compute some metric for each feature, and select features with highest score.

- Example: compute loss or  $R^2$  value when each feature in  $\mathbf{X}$  is used in single variate regression.

Any potential limitations of this approach?

Exhaustive approach: Pick best subset of  $q$  features.

Faster approach: Greedily select  $q$  features.

**Stepwise Regression:**

- **Forward:** Step 1: pick single feature that gives lowest loss.  
Step  $k$ : pick feature that when combined with previous  $k - 1$  chosen features gives lowest loss.
- **Backward:** Start with all of the features. Greedily eliminate those which have least impact on model performance.

Feature selection deserves more than two slides, but we won't go into too much more detail!

## ALTERNATIVE APPROACH

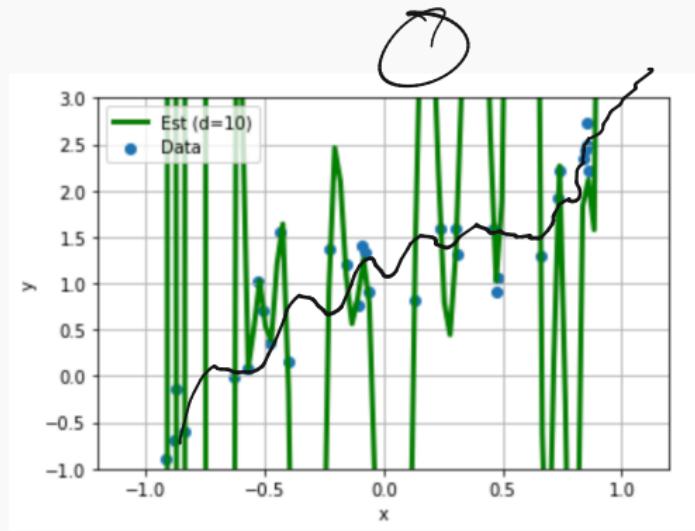
**Regularization:** Explicitly discourage overfitting by adding a regularization penalty to the loss minimization problem.

$$\min_{\theta} [L(\theta) + \underbrace{\text{Reg}(\theta)}_{+\text{Reg}(\beta)}].$$

**Example:** Least squares regression.  $L(\beta) = \|\underline{X\beta - y}\|_2^2 + \lambda \|\beta\|_2^2$

- Ridge regression ( $\ell_2$ ):  $\text{Reg}(\beta) = \underbrace{\lambda \|\beta\|_2^2}_{\lambda > 0}$
- LASSO (least absolute shrinkage and selection operator) ( $\ell_1$ ):  $\text{Reg}(\beta) = \underline{\lambda \|\beta\|_1}$
- Elastic net:  $\text{Reg}(\beta) = \underbrace{\lambda_1 \|\beta\|_1}_{\lambda_1} + \underbrace{\lambda_2 \|\beta\|_2^2}_{\lambda_2}$

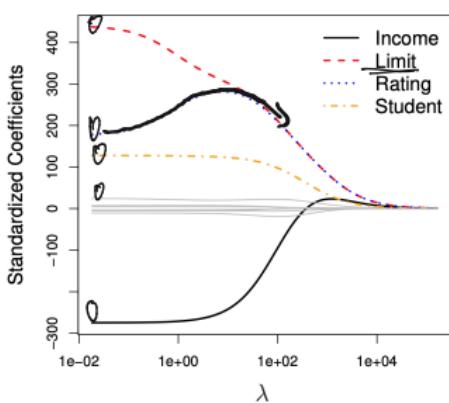
# REGULARIZATION



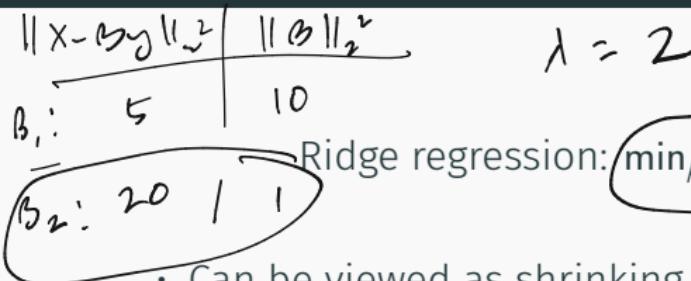
## RIDGE REGULARIZATION

Ridge regression:  $\min_{\beta} \|\mathbf{X}\beta - \mathbf{y}\|_2^2 + \lambda \|\beta\|_2^2$ .

- As  $\lambda \rightarrow \infty$ , we expect  $\|\beta\|_2^2 \rightarrow 0$  and  $\|\mathbf{X}\beta - \mathbf{y}\|_2^2 \rightarrow \|\mathbf{y}\|_2^2$ .
- Feature selection methods attempt to set many coordinates in  $\beta$  to 0. Ridge regularization encourages coordinates to be small.



## DUALITY WITH CONSTRAINED REGRESSION



Ridge regression:  $\min_{\beta} \|X\beta - y\|_2^2 + \lambda \|\beta\|_2^2$ .

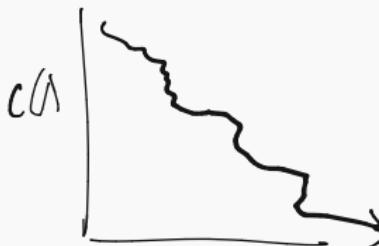
- Can be viewed as shrinking the size of our model class.

Relaxed version of  $\min_{\beta: \|\beta\|_2^2 < c} \|X\beta - y\|_2^2$ .

Claim: For any  $\underline{\lambda}$ , let  $\underline{\beta}^* = \arg \min_{\beta} \|X\beta - y\|_2^2 + \lambda \|\beta\|_2^2$ . Then there is some  $c(\lambda)$  such that:

$$\underline{\beta}_\lambda^* = \arg \min_{\beta: \|\beta\|_2^2 < c(\lambda)} \|X\beta - y\|_2^2.$$

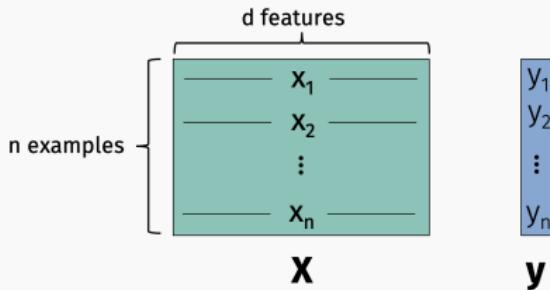
Moreover, we have the for  $\underline{\lambda}' > \underline{\lambda}$ ,  $c(\lambda') < c(\lambda)$ .



## RIDGE REGULARIZATION

Ridge regression:  $\min_{\beta} \|X\beta - y\|_2^2 + \lambda\|\beta\|_2^2$ .

- $\min_{\beta: \|\beta\|_2^2 < c} \|X\beta - y\|_2^2$  won't have a solution at zero for all  $y$ , even when over-parameterized.



- Regularization methods are not invariant to data scaling. Typically when using regularization we mean center and scale columns to have unit variance.

## RIDGE REGULARIZATION

How do we minimize:  $L_R(\beta) = \|\underline{X\beta} - \underline{y}\|_2^2 + \lambda \|\beta\|_2^2$ ?

$$(X^T X)^{-1} X^T y$$

$$\nabla L_R(\beta) = 0$$

$$\nabla L_R(\beta) = \nabla (\|X\beta - y\|_2^2) + \nabla (\lambda \|\beta\|_2^2)$$

$\underbrace{\hspace{10em}}$

$$2X^T(X\beta - y) + 2\lambda \beta = 0$$

$$2\underline{X^T X \beta} - 2\underline{X^T y} + 2\underline{\lambda I \beta} = 0$$

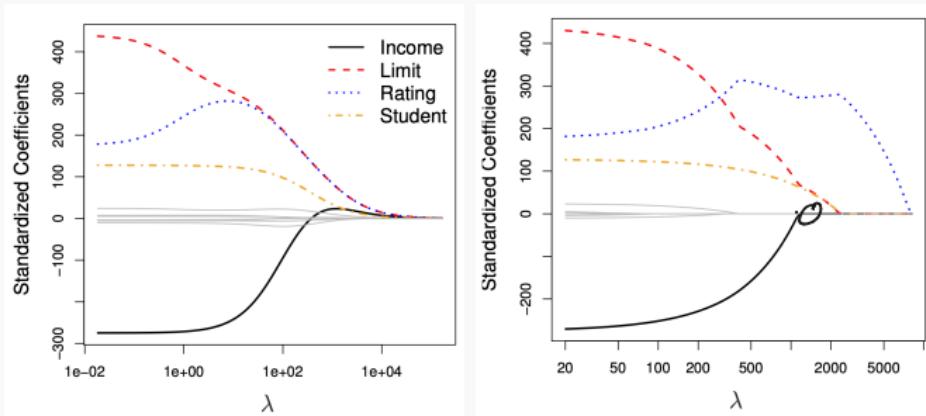
$$-(X^T X + \lambda I)\beta - X^T y = 0$$

$$\beta = (X^T X + \lambda I)^{-1} X^T y$$

# LASSO REGULARIZATION

$$\text{Lasso regularization: } \min_{\beta} \|X\beta - y\|_2^2 + \underline{\lambda \|\beta\|_1}$$

- As  $\lambda \rightarrow \infty$ , we expect  $\|\beta\|_1 \rightarrow 0$  and  $\|X\beta - y\|_2^2 \rightarrow \|y\|_2^2$ .
- Typically encourages subset of  $\beta_i$ 's to go to zero, in contrast to ridge regularization.



### Pros:

- Simpler, more interpretable model.
- More intuitive reduction in model order.

### Cons:

- No closed form solution because  $\|\beta\|_1$  is not differentiable.
- Can be solved with iterative methods, but generally not as quickly as ridge regression.

## REGULARIZATION

$2 \{0, 1, \dots, 10\}$

$\underbrace{2 \ 4 \ 6 \ 8 \ 10 \ \dots \ 1024}_{2^k}$

$2 \ 4 \ \textcircled{8} \ 16 \ \dots \ 1024$

Notes:

- Model selection/cross validation used to choose optimal scaling  $\lambda$  on  $\lambda \|\beta\|_2^2$  or  $\lambda \|\beta\|_1$ .
- Often grid search for best parameters is performed in “log space”. E.g. consider  $[\lambda_1, \dots, \lambda_q] = \underbrace{1.5[-4, -3, -2, -1, -0.1, 0.2, 0.3, 0.4]}_{2^k}$ .

$\underline{2} \quad \underline{102} \quad 202 \quad 1024$

$0(1) \quad 0(10) \quad 0(100)$

## THE BAYESIAN/PROBABILISTIC MODELING PERSPECTIVE

## CLASSIFICATION SETUP

- Data Examples:  $\underline{x_1}, \dots, \underline{x_n} \in \mathbb{R}^d$
- Target:  $\underline{y_1}, \dots, \underline{y_n} \in \{0, 2, \dots, q - 1\}$  when there are  $q$  classes.
  - Binary Classification:  $q = 2$ , so each  $y_i \in \{0, 1\}$ .
  - Multi-class Classification:  $q > 2$ . <sup>1</sup>

---

<sup>1</sup>Note that there is also multi-label classification where each data example maybe belong to more than one class.

## CLASSIFICATION EXAMPLES

- Medical diagnosis from MRI: 2 classes.
- MNIST digits: 10 classes.
- Full Optical Character Recognition: 100s of classes.
- ImageNet challenge: 21,000 classes.

Running example today: Email Spam Classification.

### Noise Proges (loss) free

Classification can (and often is) solved using the same **loss-minimization framework** we saw for regression.

We won't see that today! We're going to use classification as a window into another way of thinking about machine learning.

Will give new and interesting justifications for tools like regularization.

Today: ML from a Probabilistic Modeling/Bayesian Perspective.

## PROBABILISTIC MODELING

In a Bayesian or Probabilistic approach to machine learning we always start by conjecturing a

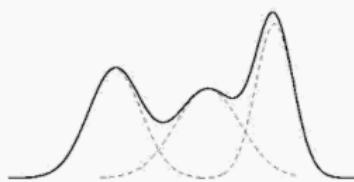
probabilistic model

that plausibly could have generated our data.

- The model guides how we make predictions.
- The model typically has unknown parameters  $\vec{\theta}$  and we try to find the most reasonable parameters based on observed data (more on this later in lecture).

# PROBABILISTIC MODELING

Typically we try to keep things simple!



**Exercise:** Come up with a probabilistic model for any one of the following data sets  $(x_1, y_1), \dots, (x_n, y_n)$ .

1. For  $n$  **people**: each  $x_i \in \{0, 1\}$  with zero indicating male, one indicating female. Each  $y_i$  is the height of the person in inches.
2. For  $n$  **NYC apartments**: each  $x_i$  is the size of the apartment in square feet. Each  $y_i$  is the monthly rent in dollars.
3. For  $n$  **students**: each  $x_i \in \{\text{Fresh.}, \text{Soph.}, \text{Jun.}, \text{Sen.}\}$  indicating class year. Each  $y_i \in \{0, 1\}$  with zero indicating the student has not taken machine learning, one indicating they have.

What are the unknown parameters of your model. What would be a guess for their values? How would you confirm or refine this guess using data?

## PROBABILISTIC MODELING

Dataset:  $(x_1, y_1), \dots, (x_n, y_n)$

Description: For  $n$  people: each  $x_i \in \{0, 1\}$  with zero indicating male, one indicating female. Each  $y_i$  is the height of the person in inches.

Model:  $y_i = \begin{cases} \text{If } x_i = 0, & \mathcal{N}(\mu_m, \sigma_m^2) \\ \text{If } x_i = 1, & \mathcal{N}(\mu_f, \sigma_f^2) \end{cases}$

$\underline{x_i \sim \text{Unif}\{0, 1\}}$ .

$$\mu_m > \mu_f$$

## PROBABILISTIC MODELING

Dataset:  $(x_1, y_1), \dots, (x_n, y_n)$

Description: For  $n$  NYC apartments: each  $x_i$  is the size of the apartment in square feet. Each  $y_i$  is the monthly rent in dollars.

Model:

$$y_i = \underbrace{x_i \cdot \underbrace{(\beta_1 N(0, \sigma^2) + \beta_0)}_{\text{line}}}$$

$$x_i \sim \text{Unif}(1200, 4000)$$

**Dataset:**  $(x_1, y_1), \dots, (x_n, y_n)$

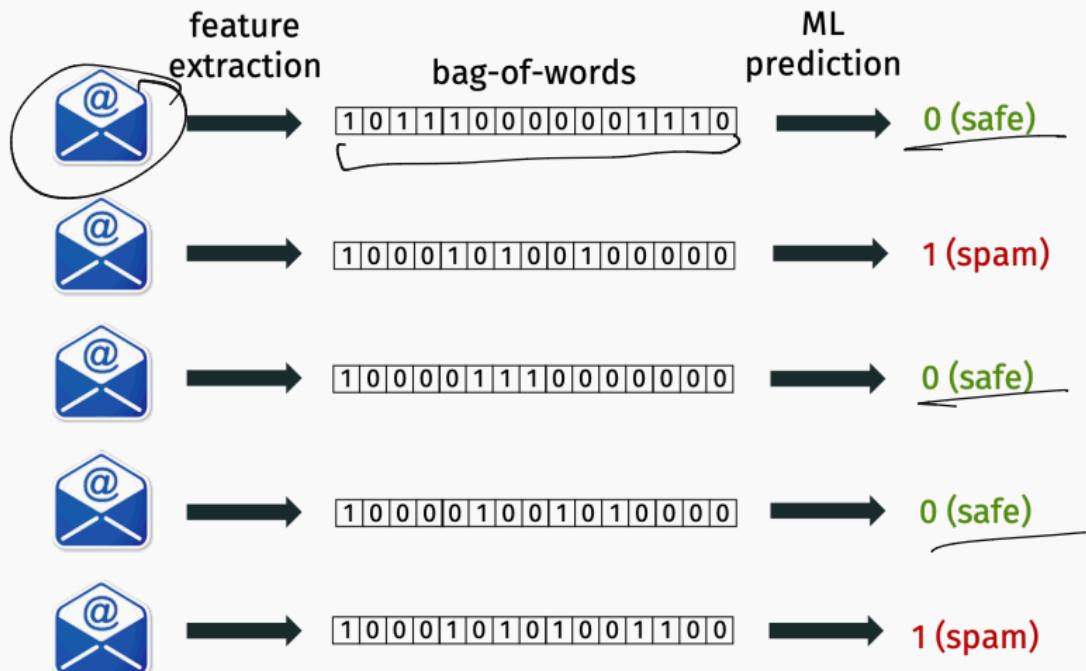
**Description:** For  $n$  students: each  $x_i \in \{\text{Fresh., Soph., Jun., Sen.}\}$  indicating class year. Each  $y_1 \in \{0, 1\}$  with zero indicating the student has not taken machine learning, one indicating they have.

**Model:**

### Goal:

- Build a probabilistic model for a binary classification problem.
- Estimate parameters of the model.
- From the model derive a classification rule for future predictions (the Naive Bayes Classifier).

## SPAM PREDICTION



Both target labels and data vectors are binary.

## PROBABILISTIC MODEL FOR EMAIL

Probabilistic model for (bag-of-words, label) pair  $(x, \underline{y})$ :

- Set  $\underline{y} = 0$  with probability  $p_0$ ,  $\underline{y} = 1$  with probability  $p_1 = 1 - p_0$ .
  - $p_0$  is probability an email is not spam (e.g. 99%).
  - $p_1$  is probability an email is spam (e.g. 1%).
- If  $\underline{y} = 0$ , for each  $i$ , set  $\underline{x_i} = 1$  with prob.  $\underline{p_{i0}}$ .
- If  $\underline{y} = 1$ , for each  $i$ , set  $\underline{x_i} = 1$  with prob.  $\underline{p_{i1}}$ .

$$P_0 = 99\% / 100$$
$$P_1 = 1\% / 100$$

Unknown model parameters:

- $p_0, p_1$ ,
- $p_{10}, p_{20}, \dots, p_{n0}$ , one for each of the  $n$  vocabulary words.
- $p_{11}, p_{21}, \dots, p_{n1}$ , one for each of the  $n$  vocabulary words.

How would you estimate these parameters?

---

Reasonable way to set parameters:

- Set  $p_0$  and  $p_1$  to the empirical fraction of not spam/spam emails.
- For each word  $i$ , set  $p_{i0}$  to the empirical probability word  $i$  appears in a non-spam email.
- For each word  $i$ , set  $p_{i1}$  to the empirical probability word  $i$  appears in a spam email.

Estimating these parameters from previous data examples is the only “training” we will do.

DONE WITH MODELING  
ON TO PREDICTION

## PROBABILITY REVIEW

- **Probability:**  $p(x)$  – the probability event  $x$  happens.
- **Joint probability:**  $p(x,y)$  – the probability that event  $x$  and event  $y$  happen.
- **Conditional Probability**  $p(x | y)$  – the probability  $x$  happens given that  $y$  happens.

$$p(x|y) =$$

## BAYES THEOREM/RULE

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)}$$

**Proof:**

## CLASSIFICATION RULE

Given unlabeled input  $(\mathbf{x}, \underline{\hspace{2cm}})$ , choose the label  $y \in \{0, 1\}$  which is most likely given the data. Recall  $\mathbf{x} = [0, 0, 1, \dots, 1, 0]$ .

Classification rule: maximum a posterior prob. (MAP) estimate.

Step 1. Compute:

- $p(y = 0 | \mathbf{x})$ : prob.  $y = 0$  given observed data vector  $\mathbf{x}$ .
- $p(y = 1 | \mathbf{x})$ : prob.  $y = 1$  given observed data vector  $\mathbf{x}$ .

Step 2. Output: 0 or 1 depending on which probability is larger.

$p(y = 0 | \mathbf{x})$  and  $p(y = 1 | \mathbf{x})$  are called **posterior** probabilities.

## EVALUATING THE POSTERIOR

How to compute the posterior? **Bayes rule!**

$$p(y = 0 | \mathbf{x}) = \frac{p(\mathbf{x} | y = 0)p(y = 0)}{p(\mathbf{x})} \quad (1)$$

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{evidence}} \quad (2)$$

- **Prior:** Probability in class 0 prior to seeing any data.
- **Posterior:** Probability in class 0 after seeing the data.

## EVALUATING THE POSTERIOR

Goal is to determine which is larger:

$$p(y = 0 | x) = \frac{p(x | y = 0)p(y = 0)}{p(x)} \quad \text{vs.}$$

$$p(y = 1 | x) = \frac{p(x | y = 1)p(y = 1)}{p(x)}$$

How to compute posteriors:

- Ignore evidence  $p(x)$  since it is the same for both sides.
- $p(y = 0)$  and  $p(y = 1)$  already known (computed from training data).
- $p(x | y = 0) = ?$   $p(x | y = 1) = ?$

**“Naive” Bayes Rule:** Compute  $p(\mathbf{x} | y = 0)$  by assuming independence:

$$p(\mathbf{x} | y = 0) = p(x_1 | y = 0) \cdot p(x_2 | y = 0) \cdot \dots \cdot p(x_n | y = 0)$$

- $p(x_i | y = 0)$  is the probability you observe  $x_i$  given that an email is not spam.<sup>2</sup>

A more complicated method might take dependencies into account.

---

<sup>2</sup>Recall,  $x_i$  is either 0 when word  $i$  is not present, or 1 when word  $i$  is present.

## Final Naive Bayes Classifier

**Training/Modeling:** Use existing data to compute:

- $p(y = 0), p(y = 1)$
- For all  $i$ :
  - Compute  $p(0 \text{ at position } i | y = 0), p(1 \text{ at position } i | y_0)$
  - Compute  $p(0 \text{ at position } i | y = 1), p(1 \text{ at position } i | y = 1)$

**Prediction:**

- For all  $i$ :
  - Compute  $p(\mathbf{x} | y = 0) = \prod_i p(x_i | y = 0)$
  - Compute  $p(\mathbf{x} | y = 1) = \prod_i p(x_i | y = 1)$
- Return

$$\arg \max [p(\mathbf{x} | y = 0) \cdot p(y = 0), p(\mathbf{x} | y = 1) \cdot p(y = 1)].$$