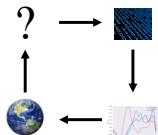


Linear Models & Feature Engineering

Slides by:

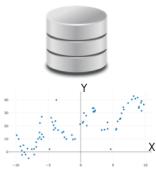
Joseph E. Gonzalez jegonzal@cs.berkeley.edu



Recap

Machine Modeling and Estimation (Learning)

Training Data



1. Define the model

$$\hat{y} = f_{\theta}(x) = \theta_0 + \theta_1 x$$

2. Choose a loss

$$L(\theta) = \frac{1}{n} \sum_{i=1}^n (y_i - f_{\theta}(x_i))^2$$

3. Minimize the loss

$$\hat{\theta} = \arg \min_{\theta} L(\theta)$$



Prediction (Testing)

Sometimes also called inference and scoring

1. Receive a new query point

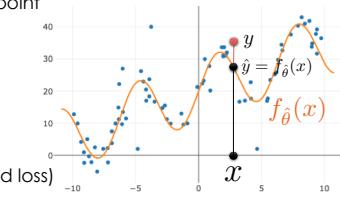
x

2. Make prediction using learned model

$$\hat{y} = f_{\hat{\theta}}(x)$$

3. Test Error (using squared loss)

$$(y - f_{\hat{\theta}}(x))^2 = (y - \hat{y})^2$$



Training Objective

$$\arg \min_{\theta} \frac{1}{n} \sum_{i=1}^n (y_i - f_{\theta}(x_i))^2$$

- Minimize error on training data
 - sample of data from the world
 - estimate of the expected error
- We can compute this directly

Idealized Objective

$$\arg \min_{\theta} \mathbf{E} [(y - f_{\theta}(x))^2]$$

- Minimize our expected prediction error over all possible test points
- Ideal Goal
 - Can't be computed ... ☺
- But we can analyze it!

Quantities in red are random variables

Training on a random sample of data from the population.

$$(\mathbf{X}_i, \mathbf{Y}_i) \sim \mathbf{P}(x, y) \rightarrow \hat{\theta} = \arg \min_{\theta} \frac{1}{n} \sum_{i=1}^n (\mathbf{Y}_i - f_{\theta}(\mathbf{X}_i))^2$$

Testing at a given query point x and computing expected squared error

$$\mathbf{E} [(\mathbf{Y} - f_{\hat{\theta}}(x))^2]$$

Expectation is taken over all possible Y observations.

Expectation is taken over all possible training datasets.

In the last lecture we showed that

$$\mathbf{E} \left[(\mathbf{Y} - f_{\hat{\theta}}(x))^2 \right] =$$

Obs. Var. + **(Bias)**² + **Mod. Var.**

Other terminology:

"Noise" + **(Bias)**² + **Variance**

$$\mathbf{E} \left[(\mathbf{Y} - f_{\hat{\theta}}(x))^2 \right] =$$

Assuming 0 mean observation
noise and true function $h(x)$
 $\mathbf{Y} = h(x) + \epsilon$

$$\mathbf{E} \left[(\mathbf{Y} - h(x))^2 \right] +$$

Obs. Variance
"Noise"

$$(h(x) - \mathbf{E} [f_{\hat{\theta}}(x)])^2 +$$

(Bias)²

$$\mathbf{E} \left[(\mathbf{E} [f_{\hat{\theta}}(x)] - f_{\hat{\theta}}(x))^2 \right] \quad \textbf{Model Variance}$$

Alternative proof

Courtesy of Allen Shen

Assuming 0 mean observation
noise and true function $h(x)$
 $\mathbf{Y} = h(x) + \epsilon$

$$\mathbf{E} \left[(\mathbf{Y} - f_{\hat{\theta}}(x))^2 \right] = \mathbf{E} \left[\mathbf{Y}^2 - 2f_{\hat{\theta}}(x)\mathbf{Y} + f_{\hat{\theta}}^2(x) \right]$$

$$\text{Linearity of Expectation} = \mathbf{E} [\mathbf{Y}^2] - \mathbf{E} [2f_{\hat{\theta}}(x)\mathbf{Y}] + \mathbf{E} [f_{\hat{\theta}}^2(x)]$$

$$\text{Definition of } \mathbf{Y} = \mathbf{E} [(h(x) - \epsilon)^2] - \mathbf{E} [2f_{\hat{\theta}}(x)\mathbf{Y}] + \mathbf{E} [f_{\hat{\theta}}^2(x)]$$

$$\mathbf{E} [(h(x) - \epsilon)^2] = h^2(x) - 2h(x)\mathbf{E} [\epsilon] + \mathbf{E} [\epsilon^2]$$

$\Downarrow \circ$ $\Downarrow \circ$ Defn' of ϵ

Bonus study material!

$$\mathbf{E} \left[(\mathbf{Y} - f_{\hat{\theta}}(x))^2 \right] = \mathbf{E} \left[\mathbf{Y}^2 - 2f_{\hat{\theta}}(x)\mathbf{Y} + f_{\hat{\theta}}^2(x) \right]$$

$$\text{Linearity of Expectation} = \mathbf{E} [\mathbf{Y}^2] - \mathbf{E} [2f_{\hat{\theta}}(x)\mathbf{Y}] + \mathbf{E} [f_{\hat{\theta}}^2(x)]$$

$$\text{Definition of } \mathbf{Y} = \mathbf{E} [(h(x) - \epsilon)^2] - \mathbf{E} [2f_{\hat{\theta}}(x)\mathbf{Y}] + \mathbf{E} [f_{\hat{\theta}}^2(x)]$$

$$\mathbf{E} [(h(x) - \epsilon)^2] = h^2(x) - 2h(x)\mathbf{E} [\epsilon] + \mathbf{E} [\epsilon^2]$$

$\Downarrow \circ$ $\Downarrow \circ$ Defn' of ϵ

$$= h(x)^2 + \sigma^2 - \mathbf{E} [2f_{\hat{\theta}}(x)\mathbf{Y}] + \mathbf{E} [f_{\hat{\theta}}^2(x)]$$

Bonus study material!

$$\mathbf{E} \left[(\mathbf{Y} - f_{\hat{\theta}}(x))^2 \right] = \mathbf{E} \left[\mathbf{Y}^2 - 2f_{\hat{\theta}}(x)\mathbf{Y} + f_{\hat{\theta}}^2(x) \right]$$

$$= h(x)^2 + \sigma^2 - \mathbf{E} [2f_{\hat{\theta}}(x)\mathbf{Y}] + \mathbf{E} [f_{\hat{\theta}}^2(x)]$$

$$\text{Y is independent of } \theta \text{ (only depends on noise)} = h(x)^2 + \sigma^2 - 2\mathbf{E} [f_{\hat{\theta}}(x)] \mathbf{E} [\mathbf{Y}] + \mathbf{E} [f_{\hat{\theta}}^2(x)]$$

$$= h(x)^2 + \sigma^2 - 2\mathbf{E} [f_{\hat{\theta}}(x)] \mathbf{E} [h(x) + \epsilon] + \mathbf{E} [f_{\hat{\theta}}^2(x)]$$

$$= h(x)^2 + \sigma^2 - 2\mathbf{E} [f_{\hat{\theta}}(x)] h(x) + \mathbf{E} [f_{\hat{\theta}}^2(x)]$$

Assuming 0 mean observation
noise and true function $h(x)$
 $\mathbf{Y} = h(x) + \epsilon$

Bonus study material!

$$\mathbf{E} \left[(\mathbf{Y} - f_{\hat{\theta}}(x))^2 \right] = \mathbf{E} \left[\mathbf{Y}^2 - 2f_{\hat{\theta}}(x)\mathbf{Y} + f_{\hat{\theta}}^2(x) \right]$$

$$= h(x)^2 + \sigma^2 - 2\mathbf{E} [f_{\hat{\theta}}(x)] h(x) + \mathbf{E} [f_{\hat{\theta}}^2(x)]$$

Definition of Variance

$$\mathbf{Var} [f_{\hat{\theta}}] = \mathbf{E} [f_{\hat{\theta}}^2(x)] - \mathbf{E} [f_{\hat{\theta}}(x)]^2$$

$$= h(x)^2 + \sigma^2 - 2\mathbf{E} [f_{\hat{\theta}}(x)] h(x) + \mathbf{E} [f_{\hat{\theta}}(x)]^2 + \mathbf{Var} [f_{\hat{\theta}}(x)]$$

$$\text{Rearranging terms} = \sigma^2 + h(x)^2 - 2\mathbf{E} [f_{\hat{\theta}}(x)] h(x) + \mathbf{E} [f_{\hat{\theta}}(x)]^2 + \mathbf{Var} [f_{\hat{\theta}}(x)]$$

$$= \sigma^2 + (h(x) - \mathbf{E} [f_{\hat{\theta}}(x)])^2 + \mathbf{Var} [f_{\hat{\theta}}(x)]$$

Bonus study material!

Summary

$$(\mathbf{X}_i, Y_i) \sim P(x, y) \Rightarrow \hat{\theta} = \arg \min_{\theta} \frac{1}{n} \sum_{i=1}^n (Y_i - f_{\theta}(x))^2$$

Expectation is taken over all possible Y observations.

$$E[(Y - f_{\hat{\theta}}(x))^2] = \sigma^2 + (h(x) - E[f_{\hat{\theta}}(x)])^2 + \text{Var}[f_{\hat{\theta}}(x)]$$

Obs. Var. + (Bias)² + Mod. Var.

Expectation is taken over all possible training datasets

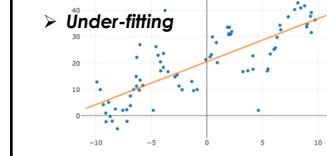
$$\text{Bias} = h(x) - E[f_{\hat{\theta}}(x)]$$

The expected deviation between the predicted value and the true value

Depends on both the:

- choice of f
- learning procedure

➤ Under-fitting



All possible functions

Possible θ values

f_{θ}

Bias

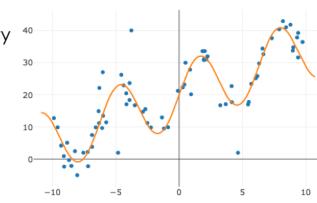
True Function

$$\text{Observation Variance} = E[(Y - h(x))^2] = \sigma^2$$

the variability of the random noise in the process we are trying to model

- measurement variability
- stochasticity
- missing information

Beyond our control (usually)

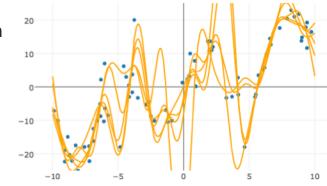


$$\text{Estimated Model Variance} =$$

$$\text{Var}[f_{\hat{\theta}}(x)] = E[(f_{\hat{\theta}}(x) - E[f_{\hat{\theta}}(x)])^2]$$

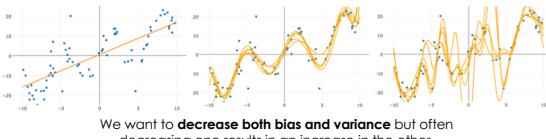
variability in the predicted value across different training datasets

- Sensitivity to variation in the training data
- Poor generalization
- Overfitting



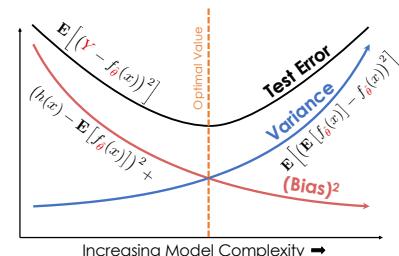
The Bias-Variance Tradeoff

Estimated Model Variance →

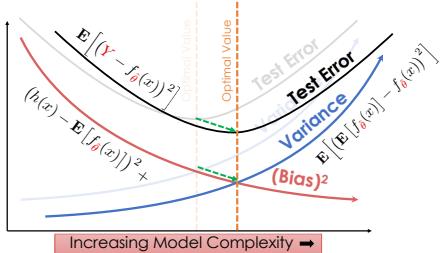


← Bias

Bias Variance Plot



More Data supports More Complexity



Model Complexity

- Roughly: capacity of the model to fit the data
- Many different measures and factors
 - Covered in machine learning class
- Dominant factors in **linear models**
 - Number and types of features
 - Regularization

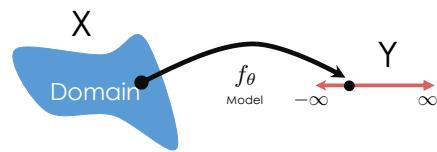
[Return to this](#)

[Start with this](#)

Regression and Linear Models

Regression

- Estimating relationship between X and Y
- Y is a quantitative value
- We will soon see X can be almost anything ...



Least Squares Linear Regression

One of the most widely used tools in machine learning and data science

Model

$$\hat{y} = f_{\theta}(x) = \sum_{j=1}^d \theta_j \phi_j(x)$$

Linear in the Parameters
Feature Functions

Loss Minimization

$$\hat{\theta} = \arg \min \frac{1}{n} \sum_{i=1}^n \left(y_i - \sum_{j=1}^d \theta_j \phi_j(x_i) \right)^2$$

Squared Loss
We will return to solving this soon!

Linear Models and Feature Functions

$$\hat{y} = f_{\theta}(x) = \sum_{j=1}^d \theta_j \phi_j(x)$$

Linear in the Parameters
Feature Functions

Designing the feature functions is a big part of machine learning and data science.

Feature Functions

- capture domain knowledge
- substantial contribute to expressivity (and complexity)

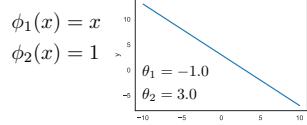
Linear Models and Feature Functions

$$\hat{y} = f_{\theta}(x) = \sum_{j=1}^d \theta_j \phi_j(x)$$

Linear in the Parameters
Feature Functions

For Example: Domain: $x \in \mathbb{R}$ Model: $f_{\theta}(x) = \theta_1 x + \theta_2$

Features:



Adding a "constant" feature function $\phi_2(x) = 1$

is a common method to introduce an offset (also sometimes called bias) term.

Linear Models and Feature Functions

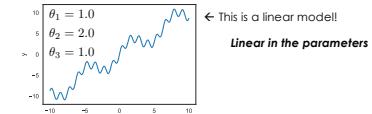
$$\hat{y} = f_{\theta}(x) = \sum_{j=1}^d \theta_j \phi_j(x)$$

Linear in the Parameters
Feature Functions

For Example: $x \in \mathbb{R}$ Model: $f_{\theta}(x) = \theta_1 x + \theta_2 \sin(x) + \theta_3 \sin(5x)$

Features:

$$\begin{aligned}\phi_1(x) &= x \\ \phi_2(x) &= \sin(x) \\ \phi_3(x) &= \sin(5x)\end{aligned}$$



Linear Models and Feature Functions

$$\hat{y} = f_{\theta}(x) = \sum_{j=1}^d \theta_j \phi_j(x)$$

Linear in the Parameters
Feature Functions

For Example: $x \in \mathbb{R}^2$

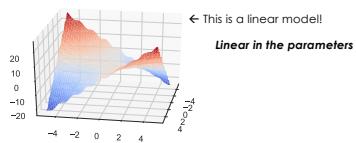
$$f_{\theta}(x) = \theta_1 x_1 x_2 + \theta_2 \cos(x_2 x_1) + \theta_3 \mathbb{I}[x_1 > x_2]$$

Features:

$$\phi_1(x) = x_1 x_2$$

$$\phi_2(x) = \cos(x_2 x_1)$$

$$\phi_3(x) = \mathbb{I}[x_1 > x_2]$$



Linear Models and Feature Functions

$$\hat{y} = f_{\theta}(x) = \sum_{j=1}^d \theta_j \phi_j(x)$$

Linear in the Parameters
Feature Functions

What if x is a record with numbers, text, booleans, etc...

	X					Y
uid	age	state	hasBought	review	rating	
0	32	NY	True	"Meh."	2.0	
42	50	VA	False	5 stars out of 5! "box..."	4.5	
57	16	CA	True	Feature engineering is tons lit yo ..."	4.1	

Feature Engineering

- The process of transforming the inputs to a model to improve prediction accuracy.
- A key focus in many applications of data science
- Feature Engineering enables you to:
 - **capture domain knowledge** (e.g., periodicity or relationships between features)
 - **encode non-numeric features** to be used as inputs to models
 - **express non-linear relationships** using linear models

How do we define ϕ ?

Feature Engineering

Keeping it *Real*

Predict rating from review information

uid	age	state	hasBought	review	rating
0	32	NY	True	"Meh."	2.0
42	50	WA	True	"Worked out of the box ..."	4.5
57	16	CA	NULL	"Hella tots lit yo ..."	4.1

Schema:

RatingsData(uid INTEGER, age FLOAT,
state STRING, hasBought BOOLEAN,
review STRING, rating FLOAT)

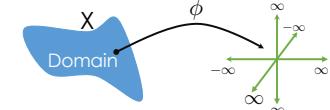
As a Linear Model?

$$X = \begin{pmatrix} uid & age & state & hasBought & review \end{pmatrix}$$

$$Y = \begin{pmatrix} rating \end{pmatrix}$$

Can I use X and Y directly in a linear model?

- No! Why?
- Text, Categorical data, Missing values...



Basic Transformations

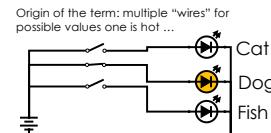
- Uninformative features: (e.g., UID)
 - Is this informative (probably not?)
 - Transformation: remove uninformative features (why?)
 - Could increase model variance ...
- Quantitative Features (e.g., Age)
 - Transformation: May apply non-linear transformations (e.g., log)
 - Transformation: Normalize/standardize (more on this later ...)
 - Example: $(x - \text{mean})/\text{stdev}$
- Categorical Features (e.g., State)
 - How do we convert State into meaningful numbers?
 - Alabama = 1, ..., Utah = 50 ?
 - Implies order/magnitude means something ... we don't want that ...
 - Transformation: One-hot-Encode

One Hot Encoding (dummy encoding)

- Transform categorical feature into many binary features:

state	AK	...	CA	...	NY	...	WA	...	WY
NY	0	...	0	...	1	...	0	...	0
WA	0	...	0	...	0	...	1	...	0
CA	0	...	1	...	0	...	0	...	0

$\phi_1(x) = \mathbb{I}[x \text{ is 'AK'}]$
Corresponding feature functions
 $\phi_2(x) = \mathbb{I}[x \text{ is 'AL'}]$
...
See notebook for example
 $\phi_{50}(x) = \mathbb{I}[x \text{ is 'WY'}]$



Encoding Missing Values

- Missing values in **Quantitative Data**
 - Try to impute (estimate) missing values... (tricky)
 - Substitute the sample mean
 - Try more sophisticated algorithms to predict the missing value ...
 - Add a binary field called "missing_col_name". (why?)
 - Sometimes missing data is signal!
- Missing values in **Categorical Data**
 - Add an additional category called "missing_col_name"
 - Some Boolean values can be converted into
 - True => +1, False => -1, Missing => 0

Encoding categorical data

- Categorical Data → **One-hot encoding:**

state	AL	...	CA	...	NY	...	WA	...	WY
NY	0	...	0	...	1	...	0	...	0
WA	0	...	0	...	0	...	1	...	0
CA	0	...	1	...	0	...	0	...	0

- Text Data
- Bag-of-words & N-gram models



Bag-of-words Encoding

- Generalization of one-hot-encoding for a string of text:
- "Learning about
machine
learning is fun."

Vector

0	0	...	1	...	2	...	1	...	0
aardwolf	ardvark		fun		learning		machine		zzyyya
- Encode text as a long vector of word counts (Issues?)
 - Long = millions of columns → typically high dimensional and very sparse
 - Word order information is lost... (is this an issue?)
 - New unseen words at prediction (test) time → drop them ...
 - A **bag** is another term for a **multiset**: an unordered collection which may contain multiple instances of each element.
 - **Stop words**: words that do not contain significant information
 - Examples: the, in, at, or, on, a, an, and ...
 - Typically removed

I made this art piece in graduate school

Do you see the stop word?

There used to be a dustbin and broom ... but the janitors got confused ...



N-Gram Encoding

- Sometimes word order matters:
- The book was not well
written but I did enjoy it.

The book was well written but I did not enjoy it.
- How do we capture word order in a "vector" model?
 - N-Gram: "Bag-of-sequences-of-words"

Removed stop words

The book was well written but I did not enjoy it.
 book well
 well written
 written not
 not enjoy

2-Gram Encoding

Vector

0	0	...	1	...	1	...	1	...	0
aardwolf	apple pen		book well		not enjoy		well written		written not

N-Gram Encoding

- Sometimes word order matters:
- The book was not well
written but I did enjoy it.

The book was well written but I did not enjoy it.
- How do we capture word order in a "vector" model?
 - N-Gram: "Bag-of-sequences-of-words"
 - Issues:
 - Can be very sparse (many combinations occur only once)
 - Many combinations will only occur at prediction time → drop ..
 - Often use hashing approximation:
 - Increment counter at `hash("not enjoy")` collisions are okay

Feature Transformations to Capture Domain Knowledge

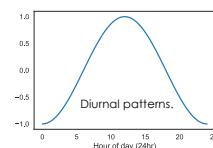
- Feature functions capture domain knowledge by introducing **additional information** from other sources **and/or combining features**

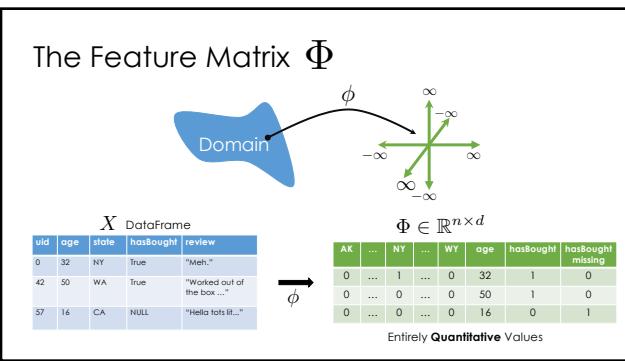
Could do a database lookup

$$\phi_i(x) = \text{isWinter}(x_{\text{date}}, x_{\text{location}})$$

- Encoding non-linear patterns

$$\phi_i(x) = \cos\left(\frac{x_{\text{hour}}}{12}\pi + \pi\right)$$



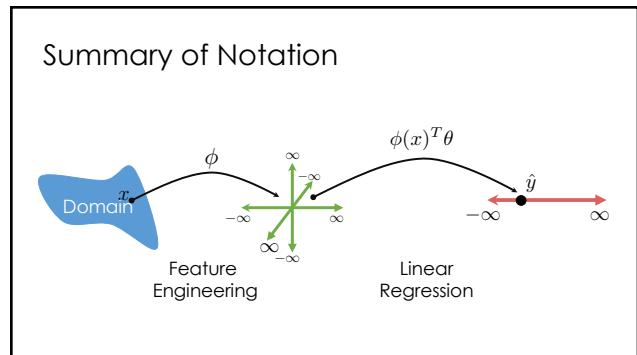
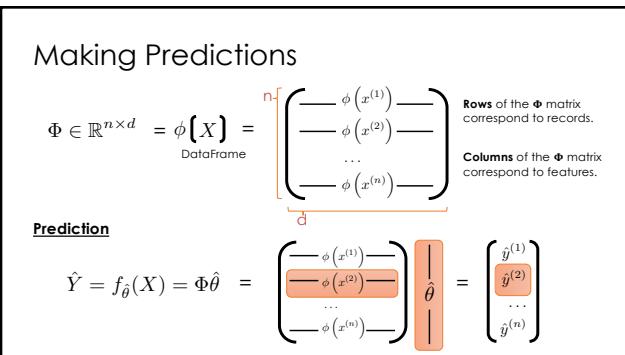


The Feature Matrix Φ

A table showing the Feature Matrix Φ with columns: AK, ..., NY, ..., WY, age, hasBought, hasBought missing. The table has three rows with values 0, 0, 0, 0, 32, 1, 0; 0, 0, 0, 0, 50, 1, 0; and 0, 0, 0, 0, 16, 0, 1. A red bracket labeled n indicates there are n rows. A red bracket labeled d indicates there are d columns.

$\Phi \in \mathbb{R}^{n \times d} = \phi[X]$ DataFrame

Rows of the Φ matrix correspond to records.
Columns of the Φ matrix correspond to features.



Optimizing the Loss (Bonus Material)

$L(\theta) = \frac{1}{n} \sum_{i=1}^n \left(y_i - \sum_{j=1}^d \theta_j \phi_j(x_i) \right)^2 = (Y - \hat{Y})^T (Y - \hat{Y})$

$$= \frac{1}{n} (Y - \Phi\theta)^T (Y - \Phi\theta)$$

$$= \frac{1}{n} (Y^T Y - 2Y^T \Phi\theta + \theta^T \Phi^T \Phi\theta)$$

Taking the Gradient of the loss

Optimizing the Loss (Bonus Material)

Deriving the Normal Equation

$$L(\theta) = \frac{1}{n} (Y^T Y - 2Y^T \Phi\theta + \theta^T \Phi^T \Phi\theta)$$

Taking the Gradient of the loss

$$\nabla_{\theta} L(\theta) = -\frac{2}{n} \Phi^T Y + \frac{2}{n} \Phi^T \Phi\theta$$

Rule 1 Rule 2

Setting the gradient equal to 0 and solving for θ :

$$0 = -\frac{2}{n} \Phi^T Y + \frac{2}{n} \Phi^T \Phi\theta \implies \hat{\theta} = (\Phi^T \Phi)^{-1} \Phi^T Y$$

"Normal Equation"

Useful Matrix Derivative Rules:

- (1) $\nabla_{\theta} (A\theta) = A^T$
- (2) $\nabla_{\theta} (\theta^T A\theta) = A\theta + A^T\theta$

The Normal Equation $\hat{\theta} = (\Phi^T \Phi)^{-1} \Phi^T Y$

$$\hat{\theta} \mid \text{d} = \begin{pmatrix} n & d \\ \Phi^T & \Phi \end{pmatrix}^{-1} \begin{pmatrix} n & 1 \\ \Phi^T & | & Y \end{pmatrix}$$

Note: For inverse to exist Φ needs to be full column rank.

→ cannot have co-linear features

This can be addressed by adding regularization ...

In practice we will use regression software
(e.g., scikit-learn) to estimate θ

Geometric Derivation (Bonus Material)

- Examine the column spaces:

$$\Phi = \begin{pmatrix} n & & \\ & \Phi^{(1)}, \Phi^{(2)}, \dots, \Phi^{(d)} & \\ n & & \\ d & & \end{pmatrix} \in \mathbb{R}^{n \times d} \quad Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \in \mathbb{R}^n$$

Linear model → Y is a linear combination of columns Φ

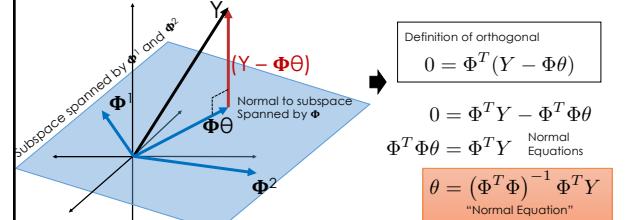
$$\Phi = \begin{pmatrix} n & & \\ & \Phi^{(1)}, \Phi^{(2)}, \dots, \Phi^{(d)} & \\ n & & \\ d & & \end{pmatrix} \in \mathbb{R}^{n \times d} \quad Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \in \mathbb{R}^n$$

Linear model → Y is a linear combination of columns Φ

$$Y \approx \hat{Y} = \Phi \hat{\theta} \Rightarrow \begin{pmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_n \end{pmatrix} \approx \begin{pmatrix} 1 & 1 & \dots & 1 \\ \Phi^{(1)}, \Phi^{(2)}, \dots, \Phi^{(d)} & | & | & | \\ 1 & 1 & \dots & 1 \end{pmatrix} \mid \hat{\theta}$$

$$Y \approx \hat{Y} = \Phi \hat{\theta} \Rightarrow \begin{pmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_n \end{pmatrix} \approx \begin{pmatrix} 1 & 1 & \dots & 1 \\ \Phi^{(1)}, \Phi^{(2)}, \dots, \Phi^{(d)} & | & | & | \\ 1 & 1 & \dots & 1 \end{pmatrix} \mid \hat{\theta}$$

➤ \hat{Y} is in the subspace spanned by the columns of Φ



Least Squares Regression in Practice

- Use optimized software packages
 - Address numerical issues with matrix inversion
- Incorporate some form of regularization
 - Address issues of collinearity
 - Produce more robust models
- We will be using scikit-learn:
 - http://scikit-learn.org/stable/modules/linear_model.html
 - See Homework 6 for details!

Daily Quiz

<http://bit.ly/ds100-sp18-lin>

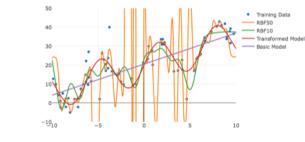
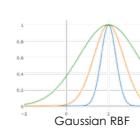
- Which of the following are **not** linear models:

- (A) $f_\theta(x) = \theta_1 x^2 + \frac{\theta_2}{\exp(x)}$
- (B) $f_\theta(x) = \theta_1 + \theta_2 \sin(\theta_3 x)$
- (C) $f_\theta(x) = \theta_1^2 + \theta_2(\theta_3 - x)$
- (D) $f_\theta(x) = \theta_1 \exp\left(\frac{(x - \mu)^2}{\sigma^2}\right) + \theta_2$

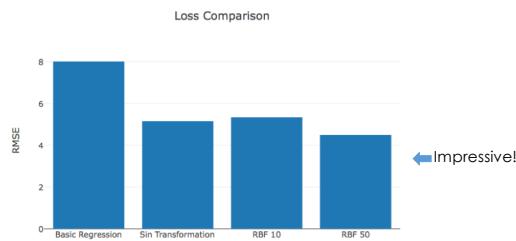
Notebook Demo

- **Generic Features:** increase model expressivity
- **Gaussian Radial Basis Functions:**

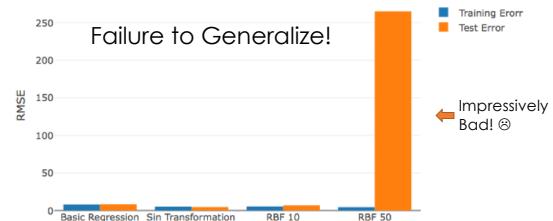
$$\phi_{\lambda_i, \mu_i}(x) = \exp\left(-\frac{\|x - \mu_i\|_2^2}{\lambda_i}\right)$$



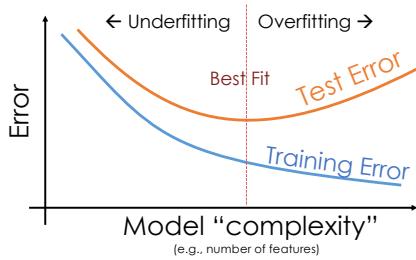
Training Error



Training vs Test Error

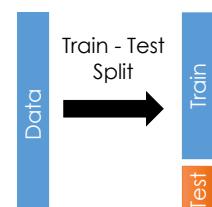


Training vs Test Error



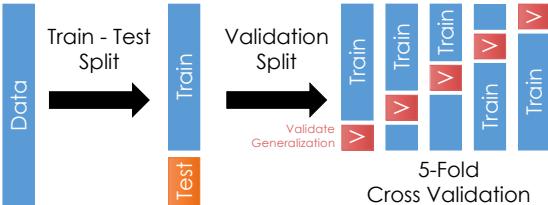
Generalization: The Train-Test Split

- **Training Data:** used to fit model
- **Test Data:** check generalization error
- How to split?
 - Randomly, Temporally, Geo...
 - Depends on application (usually randomly)
- What size? (90%-10%)
 - Larger training set → more complex models
 - Larger test set → better estimate of generalization error
 - Typically between 75%-25% and 90%-10%



You can only use the test dataset once after deciding on the model.

Generalization: Validation Split



Cross validation simulates multiple train test-splits on the training data.

Returning to Regularization

Regularization

Parametrically Controlling the Model Complexity

- Tradeoff:
 - Increase bias
 - Decrease variance



Basic Idea of Regularization

$$\hat{\theta} = \arg \min_{\theta} \frac{1}{n} \sum_{i=1}^n \text{Loss}(y_i, f_{\theta}(x_i)) + \lambda R(\theta)$$

- How should we define $R(\theta)$?
- How do we determine λ ?

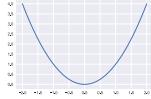
Fit the Data

Penalize Complex Models

Regularization Parameter

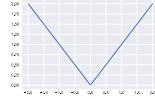
Common Regularization Functions

Ridge Regression (L2-Reg) $R_{\text{Ridge}}(\theta) = \sum_{i=1}^d \theta_i^2$



- Distributes weight across related features (robust)
- Analytic solution (easy to compute)
- Does not encourage sparsity → small but non-zero weights.

LASSO (L1-Reg) $R_{\text{Lasso}}(\theta) = \sum_{i=1}^d |\theta_i|$



- Encourages sparsity by setting weights = 0
 - Used to select informative features
 - Does not have an analytic solution → numerical methods

Regularization and Norm Balls

