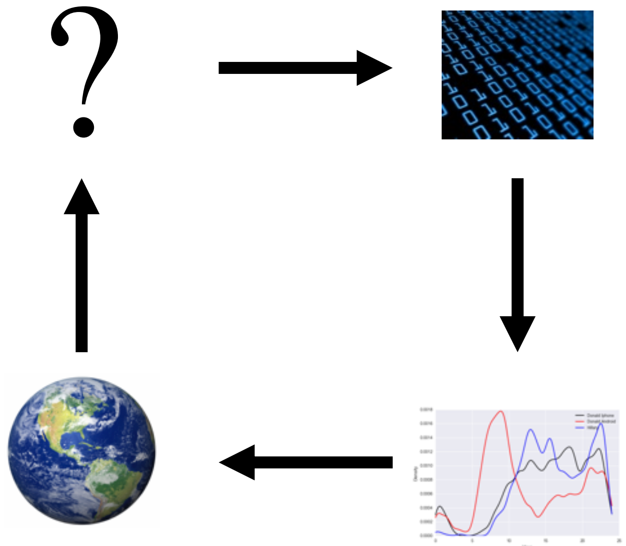
Linear Models & Feature Engineering

Slides by:

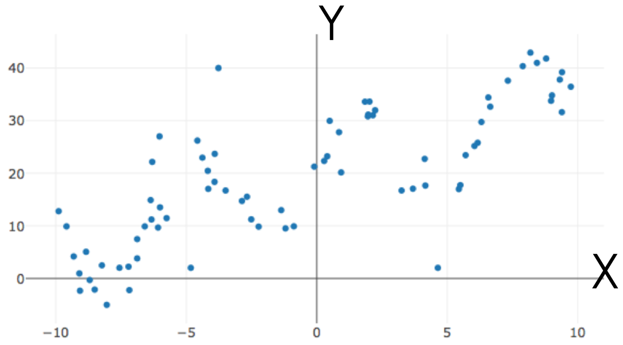
Joseph E. Gonzalez jegonzal@cs.berkeley.edu



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

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

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

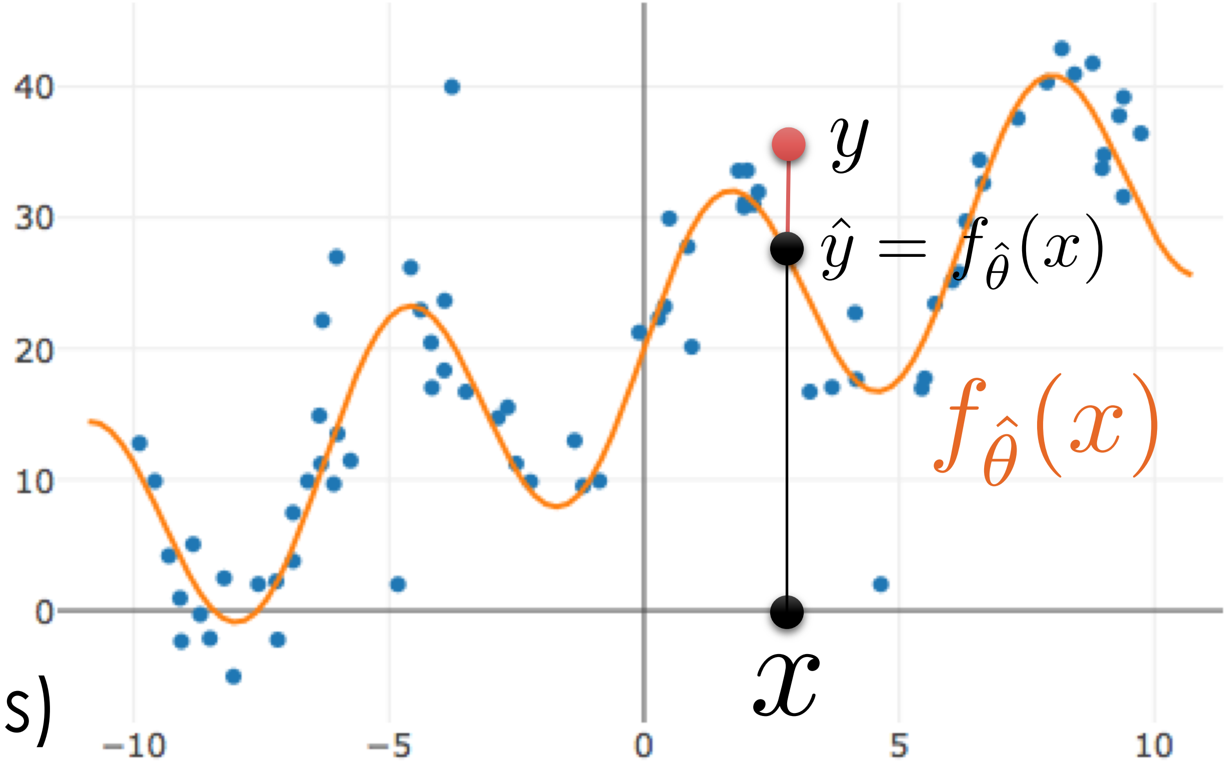
   $$x$$

2. Make prediction using learned model

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

3. Test Error (using squared loss)

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



Training Objective

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

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

Idealized Objective

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

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

Analysis of Squared Error

Quantities in **red** are random variables

Training on a **random sample** of data from the population.

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

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

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

Expectation is taken over all possible Y observations.

Expectation is taken over all possible training datasets

In the last lecture we showed that

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

Obs. Var. + **(Bias)**^{2} + **Mod. Var.**

Other terminology:

“Noise” + **(Bias)**^{2} + **Variance**

Assuming 0 mean observation noise and true function *h*(*x*)

$$Y = h(x) + \epsilon$$

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

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

Obs. Variance

"Noise"

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

(Bias)^2

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

Model Variance

Alternative proof

Courtesy of Allen Shen

Assuming 0 mean observation noise and true function *h(x)*

$$Y = h(x) + \epsilon$$

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

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

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

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

$\mathbf{E}[\epsilon] = 0$, $\mathbf{E}[\epsilon^2] = \sigma^2$ Defn' of ϵ

Bonus study material!

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

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

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

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

$\mathbf{E}[\epsilon] = 0$, $\mathbf{E}[\epsilon^2] = \sigma^2$ Defn' of ϵ

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

Bonus study material!

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

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

Y is independent of θ (only depends on noise)

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

Definition of Y

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

Linearity of expectation

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

Assuming 0 mean observation noise and true function *h(x)*

$$Y = h(x) + \epsilon$$

Bonus study material!

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

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

Definition of Variance

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

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

Rearranging terms

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

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

Bonus study material!

Summary

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

Expectation is taken over all possible Y observations.

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

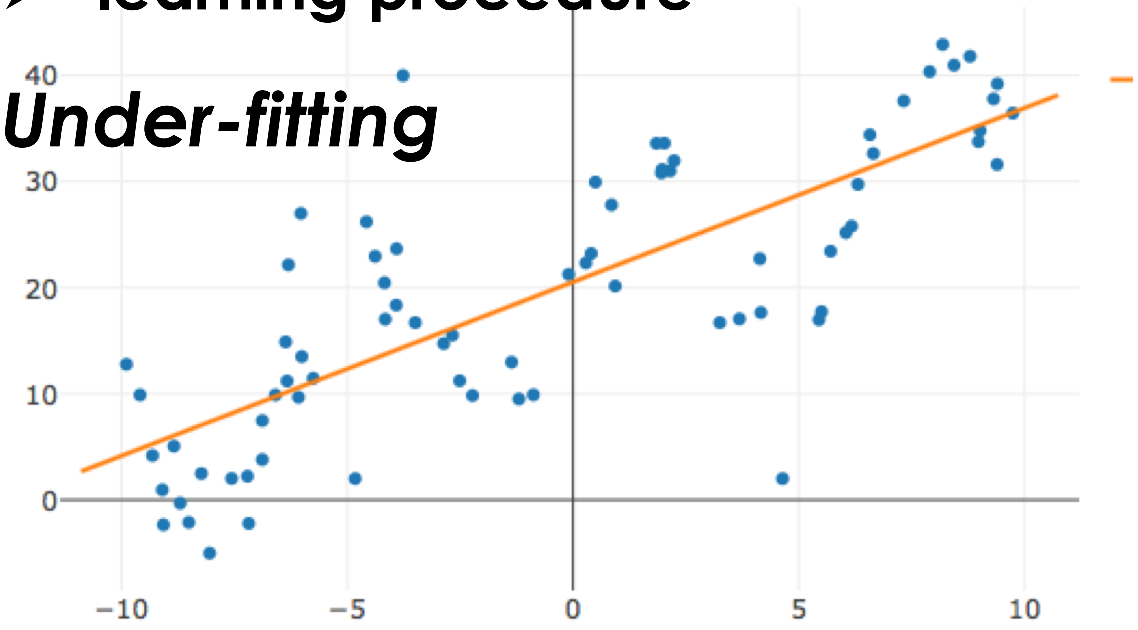
Obs. Var. + (Bias)^2 + Mod. Var.

Expectation is taken over all possible training datasets

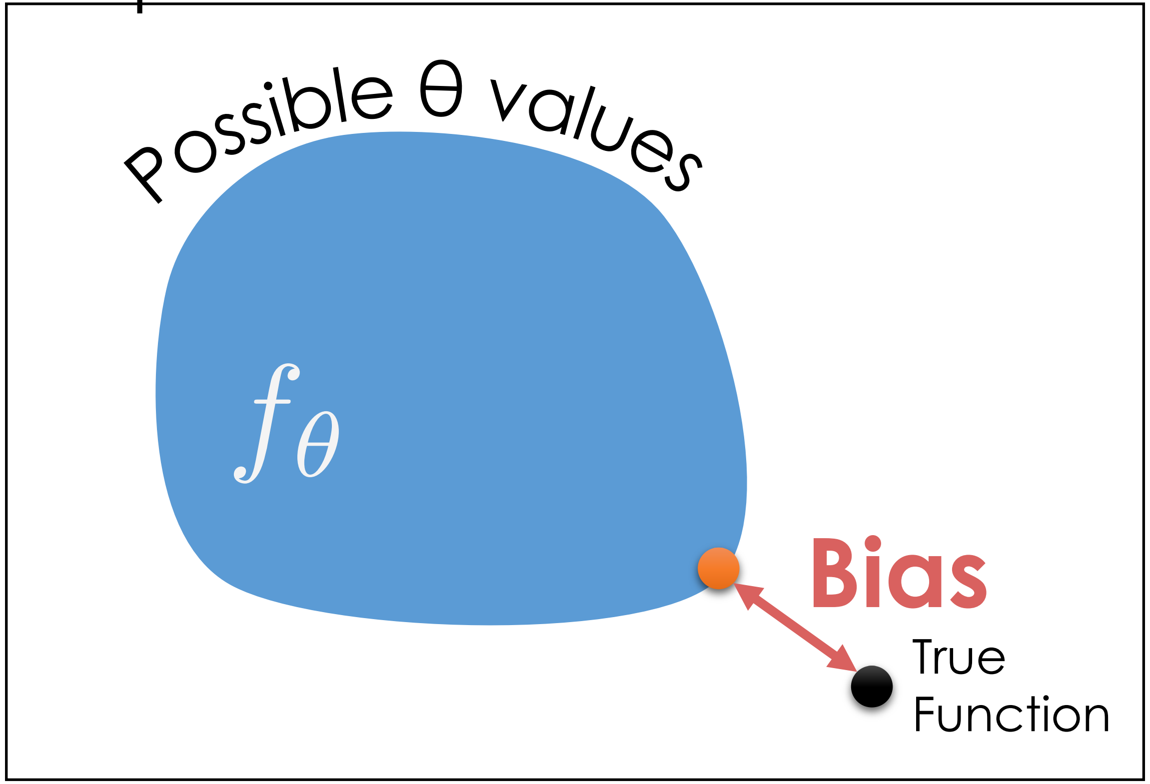
$$\text{Bias} = h(x) - \mathbf{E}\left[f_{\hat{\theta}}(x)\right]$$

The expected deviation between the predicted value and the true value

- Depends on both the:
  - **choice** of f
  - **learning procedure**
- ***Under-fitting***



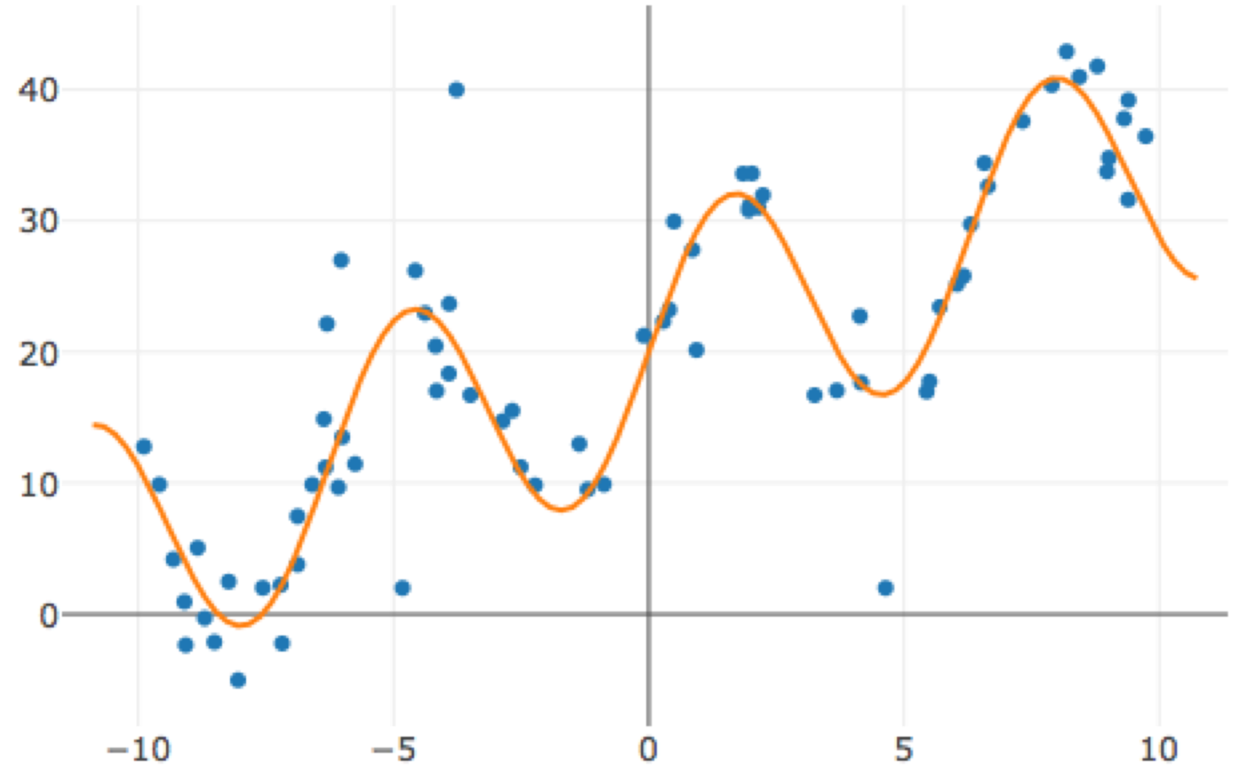
All possible functions



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

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

- measurement variability
- stochasticity
- missing information



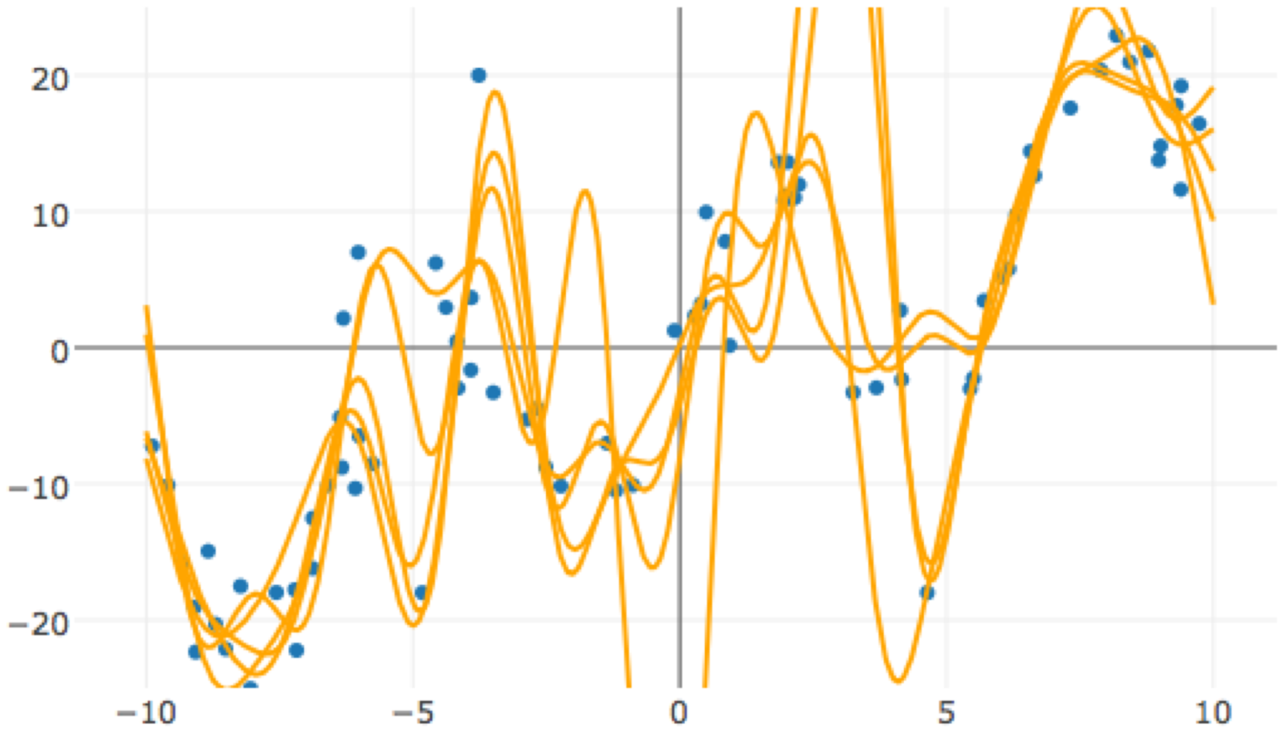
Beyond our control (usually)

Estimated Model Variance =

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

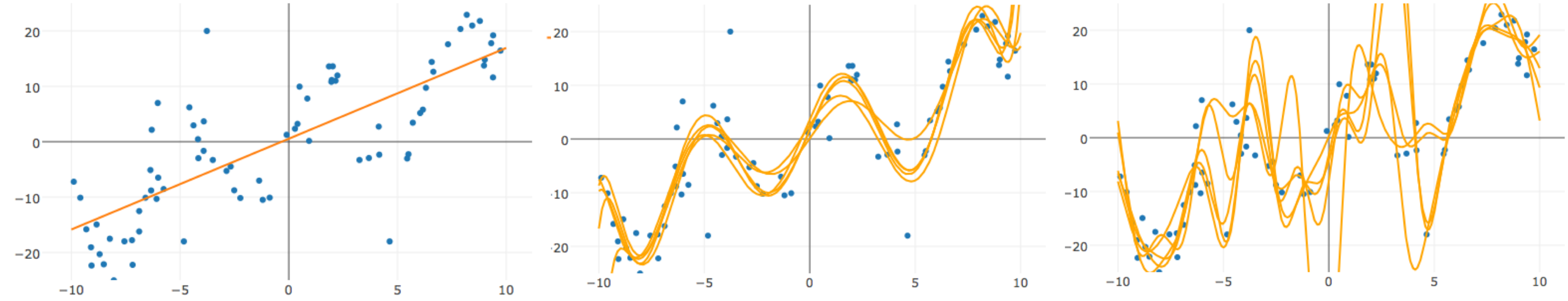
variability in the predicted value across different training datasets

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



The Bias-Variance Tradeoff

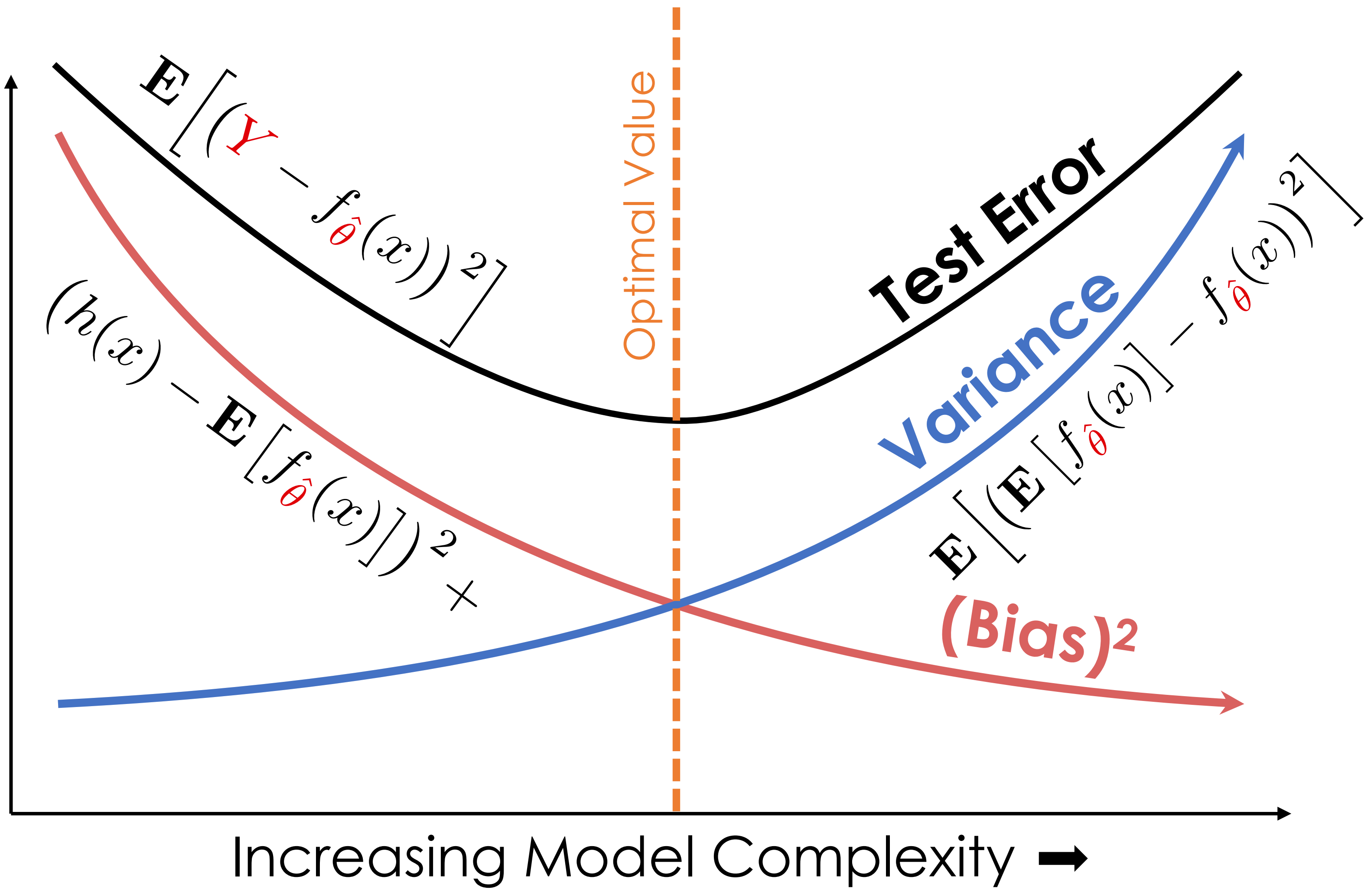
Estimated Model Variance



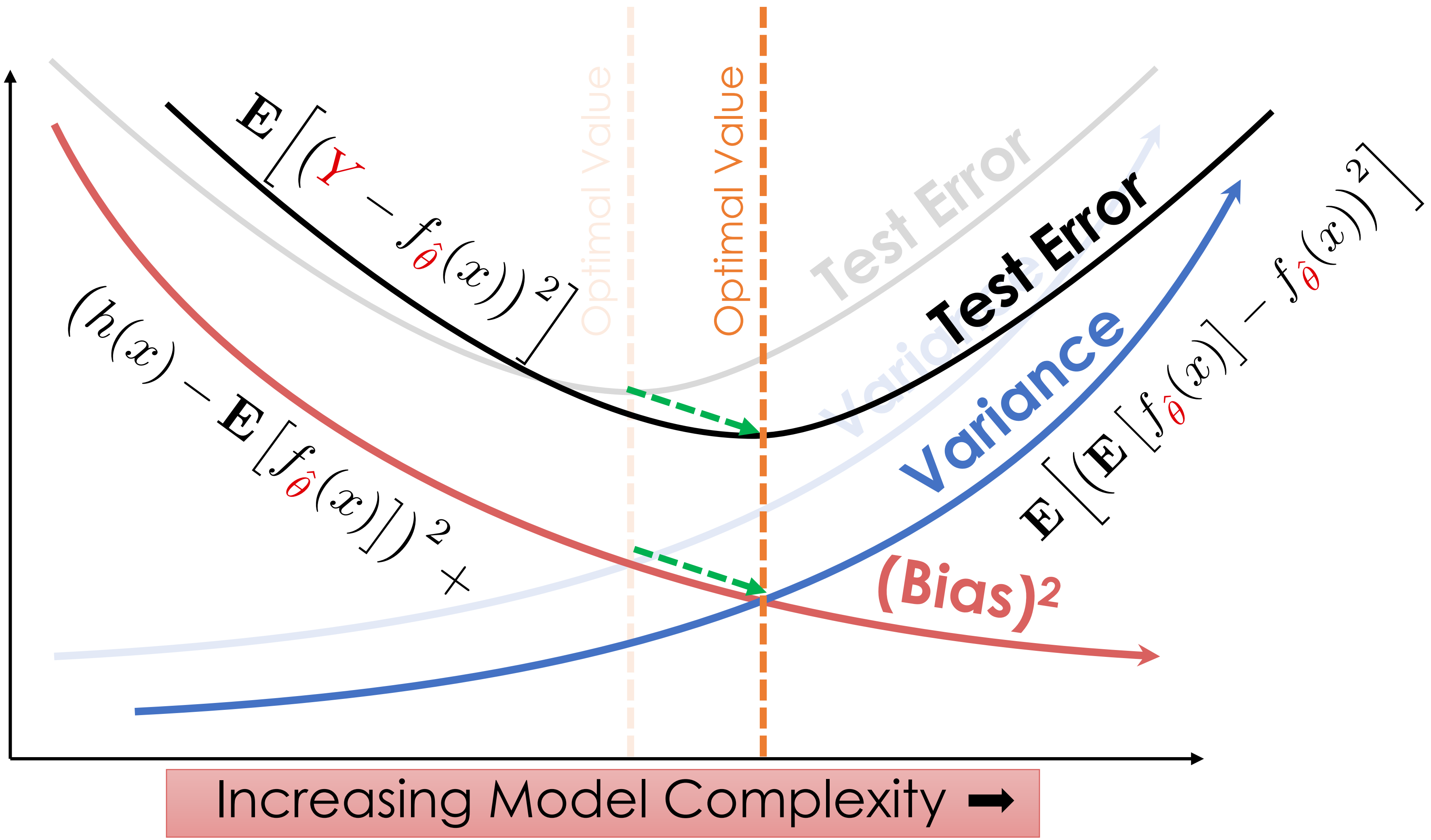
We want to **decrease both bias and variance** but often decreasing one results in an increase in the other.

Bias

Bias Variance Plot



More Data supports **More Complexity**



Model Complexity

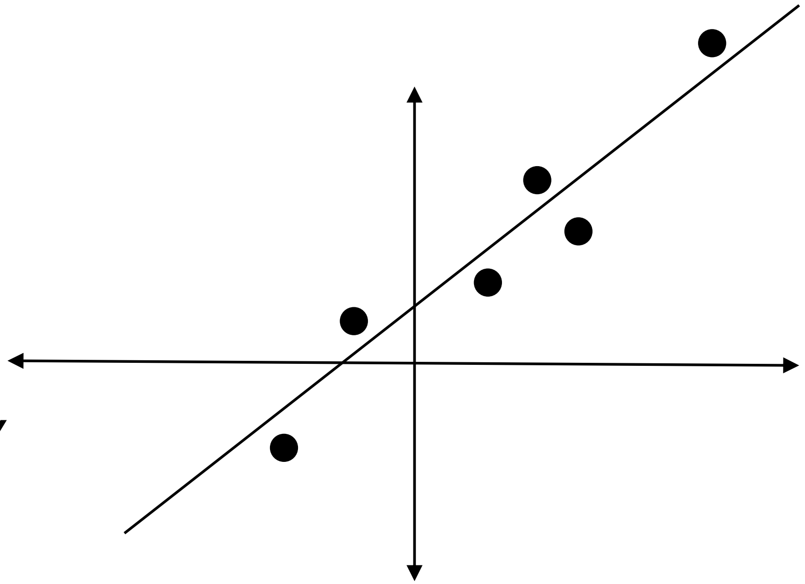
- Roughly: *capacity of the model to fit the data*
- Many different measures and factors
  - Covered in machine learning class
- Dominant factors in **linear models**
  - Number and types of features
  - Regularization

Start with this

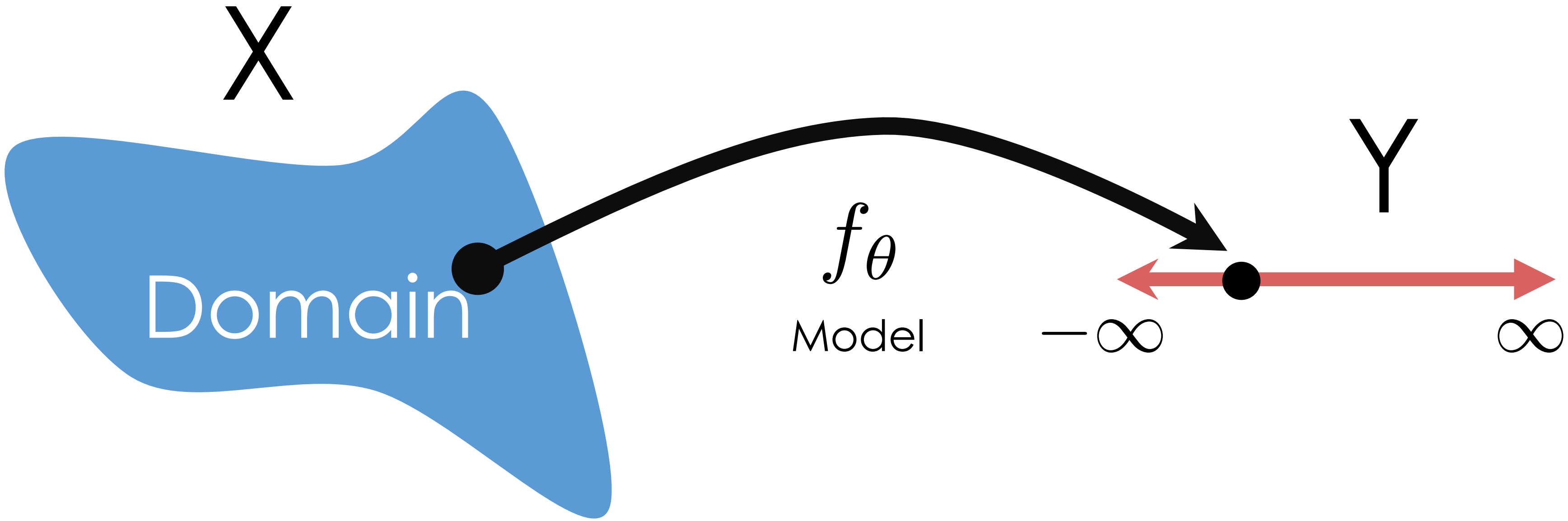
Return to this

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

Linear in the Parameters

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

Feature Functions

Squared Loss

Loss Minimization

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

We will return to
solving this soon!

Linear Models and Feature Functions

Linear in the Parameters

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

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

Linear in the Parameters

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

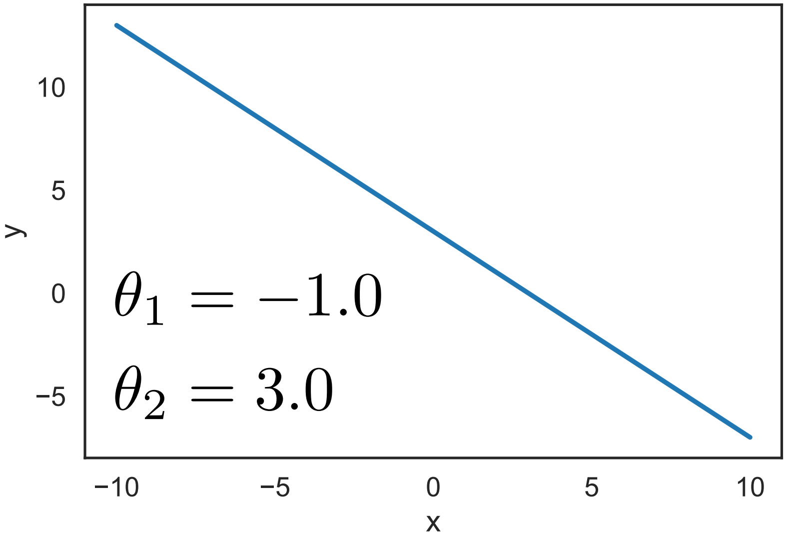
Feature Functions

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

Features:

$$\phi_1(x) = x$$

$$\phi_2(x) = 1$$



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

Linear in the Parameters

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

Feature Functions

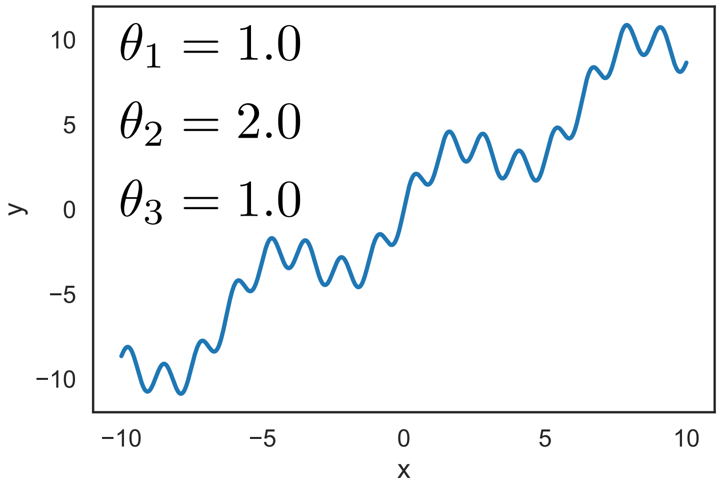
For Example: $x \in \mathbb{R}$ $\quad f_\theta(x) = \theta_1 x + \theta_2 \sin(x) + \theta_3 \sin(5x)$

Features:

$$\phi_1(x) = x$$

$$\phi_2(x) = \sin(x)$$

$$\phi_3(x) = \sin(5x)$$



← This is a linear model!

Linear in the parameters

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

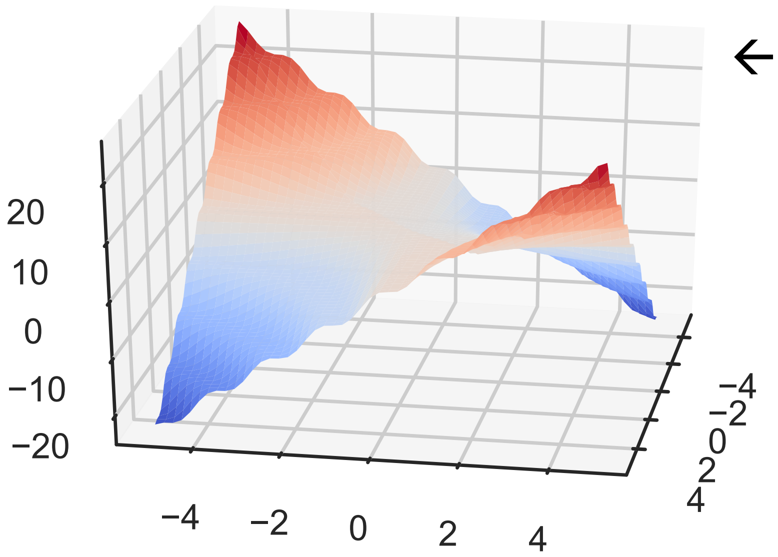
$$f_\theta(x) = \theta_1 x_1 x_2 + \theta_2 \cos(x_2 x_1) + \theta_3 \mathbb{I}\left[x_1 > x_2\right]$$

Features:

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

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

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



← This is a linear model!

Linear in the parameters

Linear Models and Feature Functions

Linear in the Parameters

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

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	[illegible]	[illegible]	[illegible]ked out of [illegible]ox …”	4.5
57	16	[illegible]	[illegible]	[illegible]a tots lit yo …”	4.1

Answer:
Feature engineering

How do we define ϕ?

Feature Engineering

Keeping it *Real*

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

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)
```

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

As a Linear Model?

X=

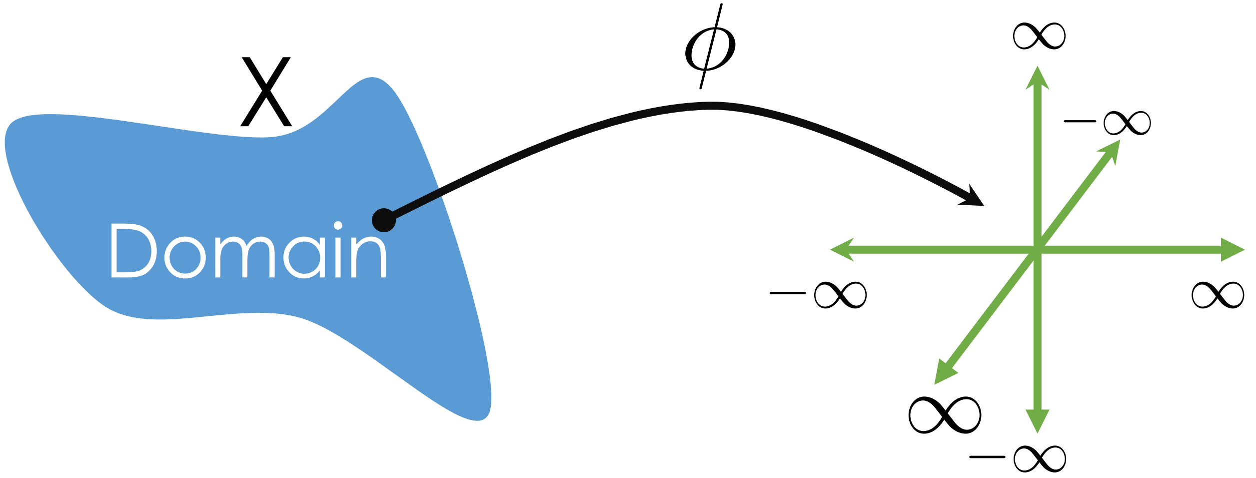
uid	age	state	hasBought	review
0	32	NY	True	"Meh."
42	50	WA	True	"Worked out of the box …"
57	16	CA	NULL	"Hella tots lit yo …"

Y=

rating
2.0
4.5
4.1

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 – mean)/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
NY
WA
CA

➡

AK	…	CA	…	NY	…	WA	…	WY
0	…	0	…	1	…	0	…	0
0	…	0	…	0	…	1	…	0
0	…	1	…	0	…	0	…	0

Corresponding feature functions

$$\phi_1(x) = \mathbb{I}\left[x \text{ is } \texttt{'AK'}\right]$$

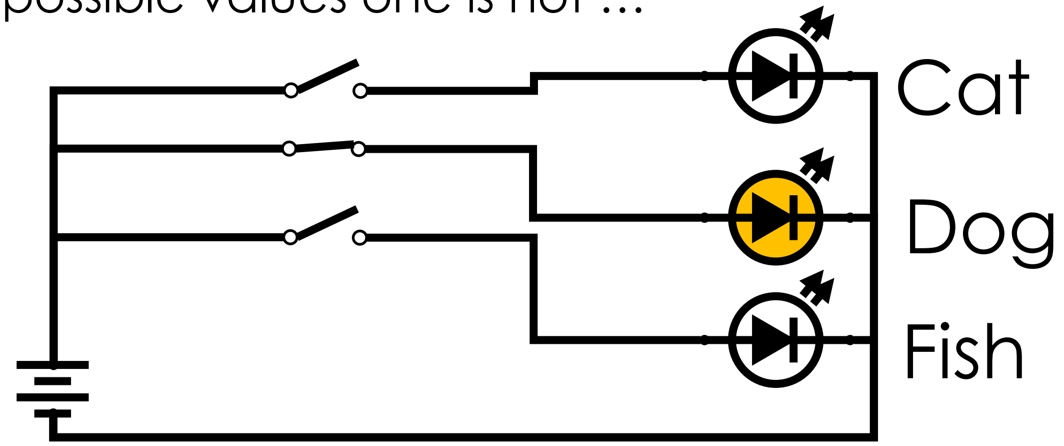
$$\phi_2(x) = \mathbb{I}\left[x \text{ is } \texttt{'AL'}\right]$$

$$\ldots$$

$$\phi_{50}(x) = \mathbb{I}\left[x \text{ is } \texttt{'WY'}\right]$$

See notebook for example code.

Origin of the term: multiple "wires" for possible values one is hot …



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 addition 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
NY
WA
CA

AL	…	CA	…	NY	…	WA	…	WY
0	…	0	…	1	…	0	…	0
0	…	0	…	0	…	1	…	0
0	…	1	…	0	…	0	…	0

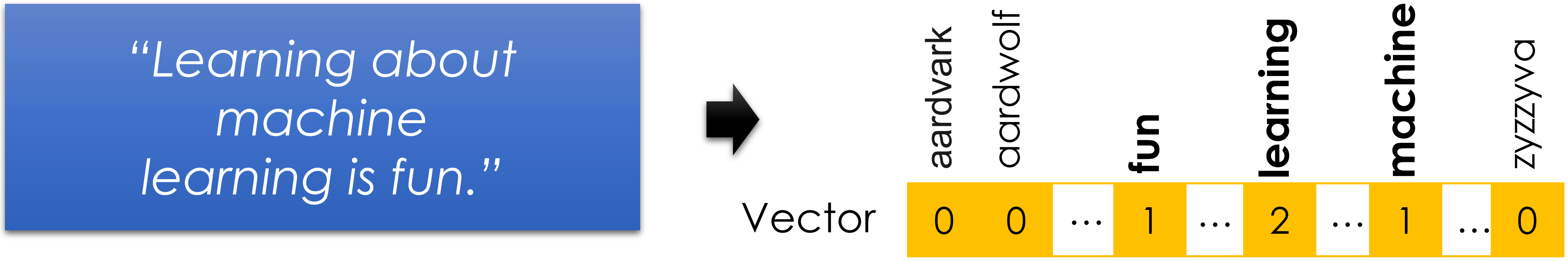
- *Text Data*
  - **Bag-of-words & N-gram models**

"Learning about machine learning is fun."

	aardvark	aardwolf		**fun**		**learning**		**machine**		zyzzyva
Vector	0	0	…	1	…	2	…	1	…	0

Bag-of-words Encoding

- Generalization of one-hot-encoding for a string of text:



- 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?

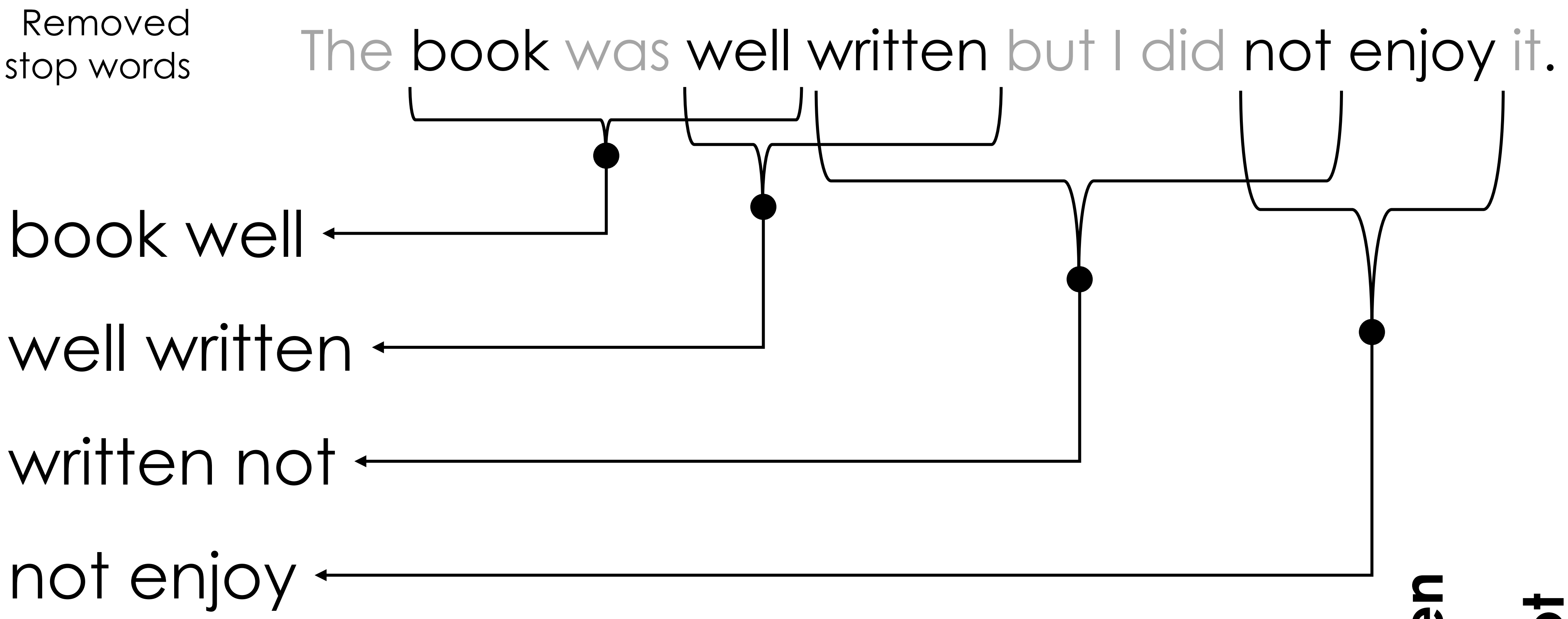
There used to be a dustbin and broom … but the janitors got confused …

N-Gram Encoding

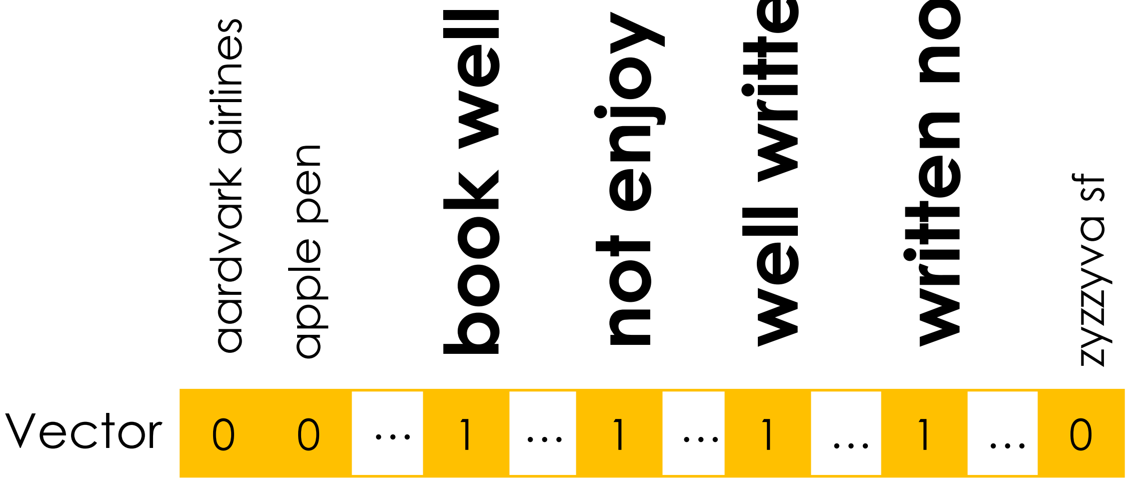
- Sometimes word order matters:

*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*”



2-Gram Encoding



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

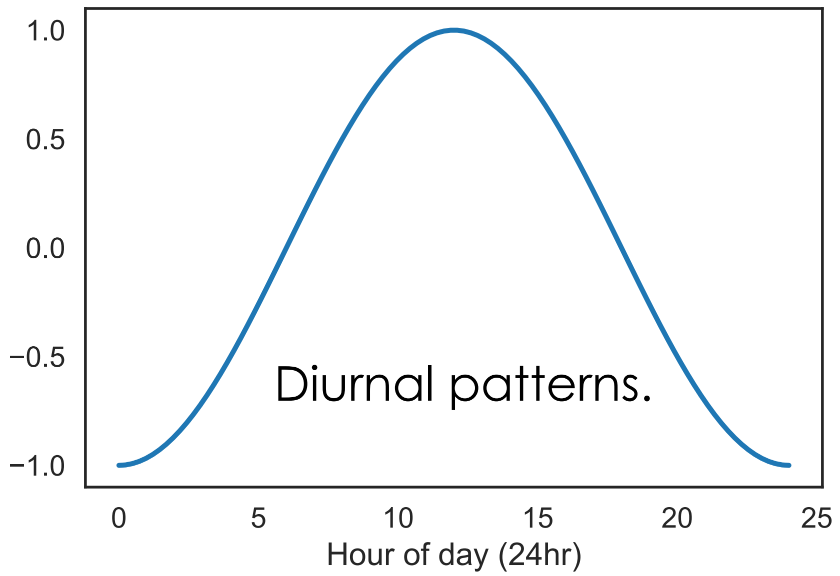
- Feature functions capture domain knowledge by introducing **additional information** from other sources **and/or combining features**

Could do a database lookup

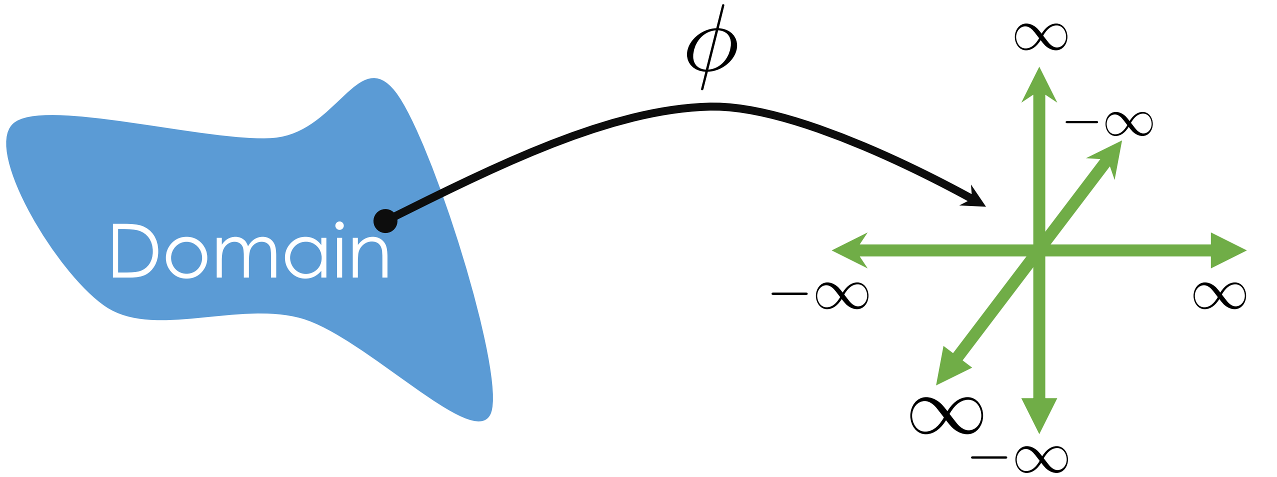
$$\phi_i(x) = \textbf{isWinter}(x_{\text{date}}, x_{\text{location}})$$

- Encoding non-linear patterns

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



The Feature Matrix Φ



X DataFrame

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

ϕ →

$\Phi \in \mathbb{R}^{n \times d}$

AK	…	NY	…	WY	age	hasBought	hasBought missing
0	…	1	…	0	32	1	0
0	…	0	…	0	50	1	0
0	…	0	…	0	16	0	1

Entirely **Quantitative** Values

The Feature Matrix Φ

AK	...	NY	...	WY	age	hasBought	hasBought missing
0	...	1	...	0	32	1	0
0	...	0	...	0	50	1	0
0	...	0	...	0	16	0	1

Entirely **Quantitative** Values

$$\Phi \in \mathbb{R}^{n \times d} = \phi\left[X\right] = \underbrace{\begin{bmatrix} \text{---} & \phi\left(x^{(1)}\right) & \text{---} \\ \text{---} & \phi\left(x^{(2)}\right) & \text{---} \\ & \ldots & \\ \text{---} & \phi\left(x^{(n)}\right) & \text{---} \end{bmatrix}}_{d} \Bigg\} n$$

DataFrame

Rows of the $\mathbf{\Phi}$ matrix correspond to records.

Columns of the $\mathbf{\Phi}$ matrix correspond to features.

Making Predictions

$$\Phi \in \mathbb{R}^{n \times d} = \phi\big[X\big] = \begin{bmatrix} \text{—} \ \phi\left(x^{(1)}\right) \ \text{—} \\ \text{—} \ \phi\left(x^{(2)}\right) \ \text{—} \\ \cdots \\ \text{—} \ \phi\left(x^{(n)}\right) \ \text{—} \end{bmatrix}$$

DataFrame

n

d

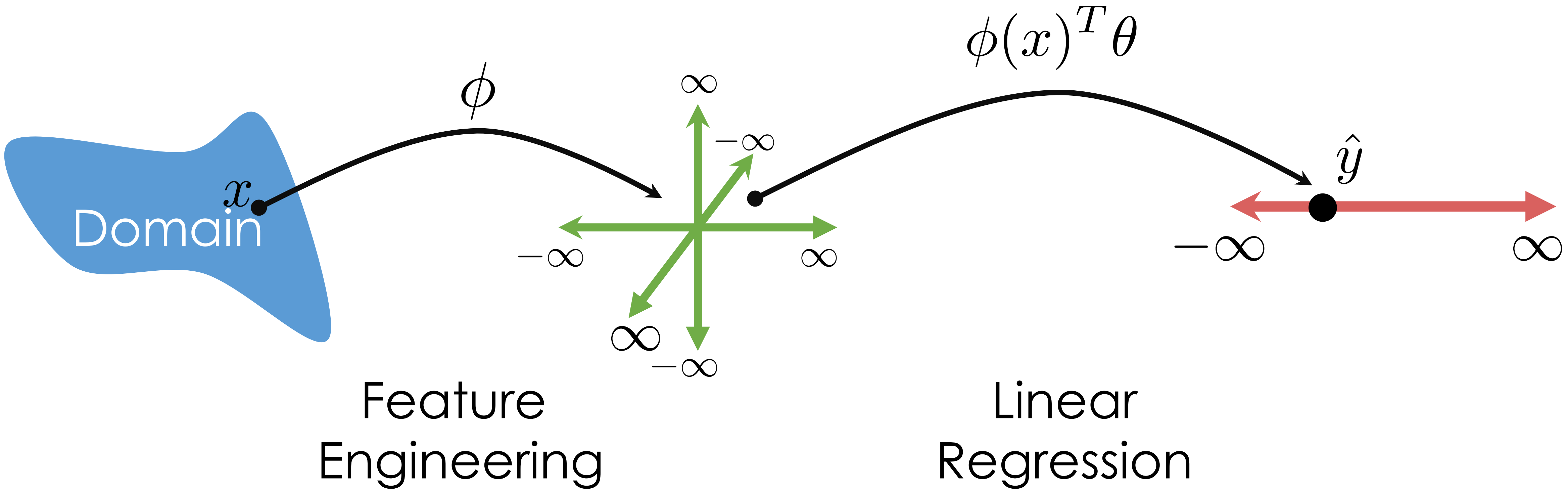
Rows of the $\mathbf{\Phi}$ matrix correspond to records.

Columns of the $\mathbf{\Phi}$ matrix correspond to features.

Prediction

$$\hat{Y} = f_{\hat{\theta}}(X) = \Phi\hat{\theta} = \begin{bmatrix} \text{—} \ \phi\left(x^{(1)}\right) \ \text{—} \\ \text{—} \ \phi\left(x^{(2)}\right) \ \text{—} \\ \cdots \\ \text{—} \ \phi\left(x^{(n)}\right) \ \text{—} \end{bmatrix} \begin{bmatrix} | \\ \hat{\theta} \\ | \end{bmatrix} = \begin{bmatrix} \hat{y}^{(1)} \\ \hat{y}^{(2)} \\ \cdots \\ \hat{y}^{(n)} \end{bmatrix}$$

Summary of Notation



Optimizing the Loss (Bonus Material)

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

$$\begin{bmatrix} y_1 - \hat{y}_1 & y_2 - \hat{y}_2 & \cdots & y_n - \hat{y}_n \end{bmatrix} \begin{bmatrix} y_1 - \hat{y}_1 \\ y_2 - \hat{y}_2 \\ \cdots \\ y_n - \hat{y}_n \end{bmatrix}$$

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

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

Taking the Gradient of the loss

Optimizing the Loss (Bonus Material)

Deriving the Normal Equation

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

Rule 1 Rule 2

Taking the Gradient of the loss

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

Useful Matrix Derivative Rules:

(1) $\nabla_\theta \left(A\theta\right) = A^T$

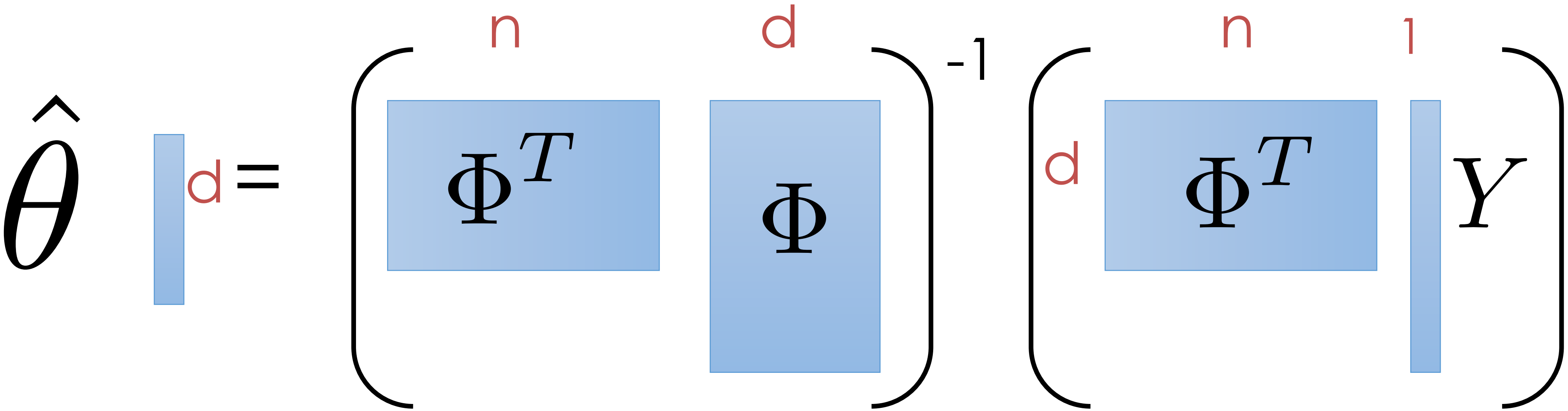
(2) $\nabla_\theta \left(\theta^T A\theta\right) = A\theta + A^T\theta$

Setting the gradient equal to 0 and solving for θ:

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

"Normal Equation"

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



$$\hat{\theta} = \left(\Phi^T \Phi\right)^{-1} \left(\Phi^T Y\right)$$

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)~~

We have decided to make this derivation not bonus material and therefore you should know it!

- Examine the column spaces:

Columns space of $\boldsymbol{\Phi}$

$$\Phi = \underbrace{\begin{pmatrix} | & | & & | \\ \Phi^{(1)}, & \Phi^{(2)}, & \ldots, & \Phi^{(d)} \\ | & | & & | \end{pmatrix}}_{d} \in \mathbb{R}^{n \times d} \qquad (n \text{ rows})$$

$$Y = \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}}_{1} \in \mathbb{R}^{n} \qquad (n \text{ rows})$$

- Linear model → Y is a linear combination of columns $\boldsymbol{\Phi}$

Columns space of $\boldsymbol{\Phi}$

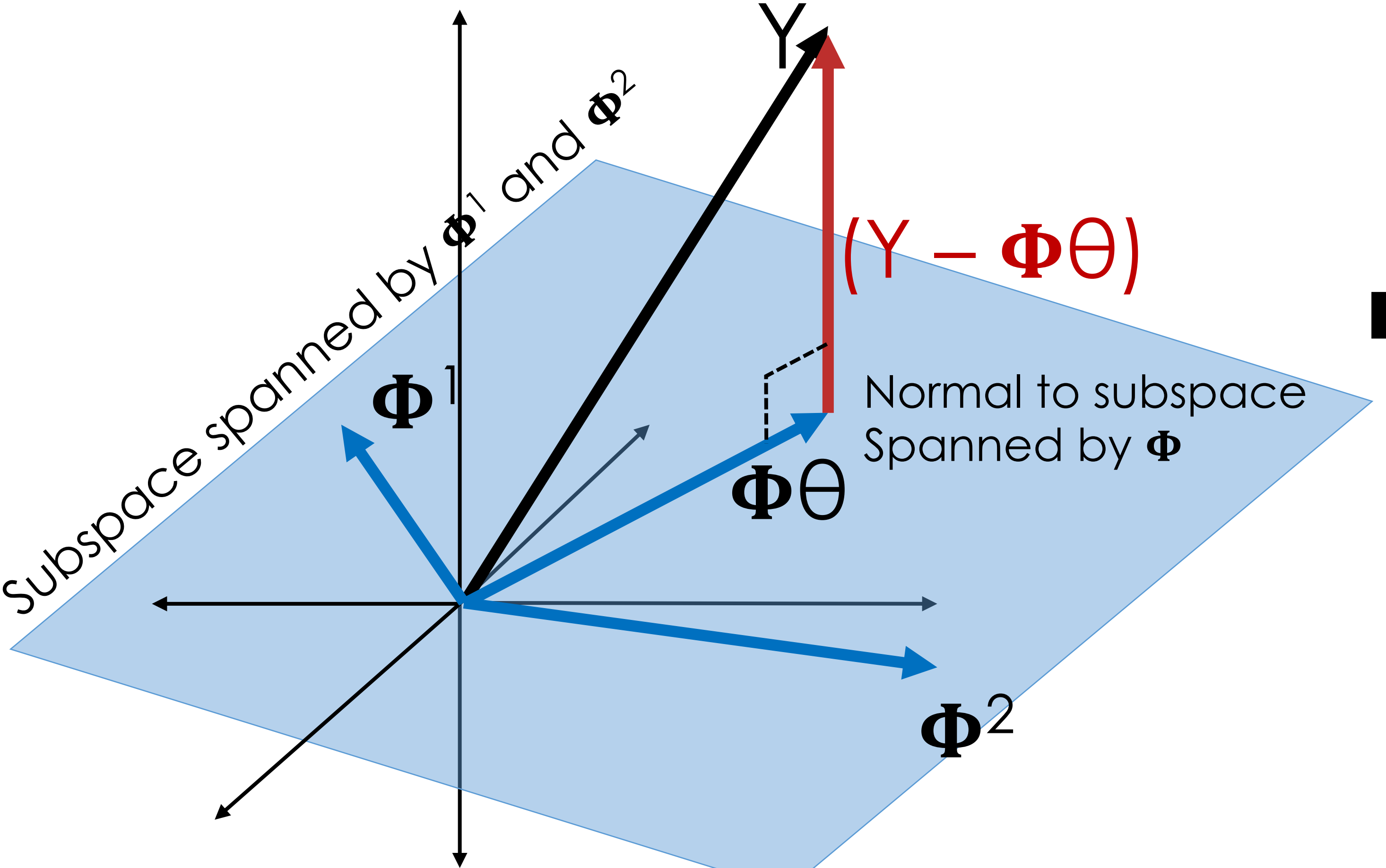
$$\Phi = \underbrace{\left.\begin{pmatrix} | & | & & | \\ \Phi^{(1)}, & \Phi^{(2)}, & \ldots, & \Phi^{(d)} \\ | & | & & | \end{pmatrix}\right\}}_{d} n \in \mathbb{R}^{n \times d} \qquad Y = \underbrace{\left.\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}\right\}}_{1} n \in \mathbb{R}^{n}$$

➢ Linear model → Y is a linear combination of columns $\boldsymbol{\Phi}$

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

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

- $\hat{Y}$ is in the subspace spanned by the columns of $\mathbf{\Phi}$



Definition of orthogonal

$$0 = \Phi^T (Y - \Phi\theta)$$

$$0 = \Phi^T Y - \Phi^T \Phi\theta$$

$$\Phi^T \Phi\theta = \Phi^T Y$$

Normal Equations

$$\theta = \left(\Phi^T \Phi\right)^{-1} \Phi^T Y$$

"Normal Equation"

Lecture ended here

Note you do need to know the final geometric derivation even though I said in lecture that you do not.