Do not distribute course material

You may not and may not allow others to reproduce or distribute lecture notes and course materials publicly whether or not a fee is charged.

Topic 3 Model Selection

PROF. LINDA SELLIE

Thanks to:

- Some of the material is from Prof. Sundeep Rangan
  - This includes some slides and the motivating examples
- Some slides (the slides with the green background) are from Yaser Abu-Mostafa

Finding Parameters via Optimization
A general ML recipe

General ML problem

- Get data
- Pick a model with parameters
- Pick a loss function
  - Measures goodness of fit model to data
  - Function of the parameters
- Find parameters that minimizes loss

Multiple linear regression

1) Finding a way to have a more complex hypothesis class
2) If we have more than one hypothesis class to choose from - how do we select which one to use?

Loss function: $\text{RSS}(\mathbf{w}) = \sum_{i=1}^{N} (y^{(i)} - \hat{y}^{(i)})^2$

Select $\mathbf{w} = [w_0, w_1, w_2, \ldots, w_d]^T$ to minimize $\text{RSS}(\mathbf{w})$

- In learning, our goal is to find a hypothesis that minimizes $E_{\text{out}}[g(\mathbf{x})]$ (not just $E_{\text{in}}[g(\mathbf{x})]$).
- In this lecture, we observe that choosing the model with the smallest training error doesn't work.
- Next we explore the different types of errors we make.
- We have to find a way to compare models.

Generalization

Training Error

$$E_{\text{in}}(w_0, w_1) = \underbrace{\frac{1}{N}\sum_{i=1}^{N}}_{\text{Average error on the N training examples}} \overbrace{\text{error}}^{\text{Cost (loss) for prediction not being the same as true label}}(y^{(i)}, \underbrace{g(\mathbf{x}^{(i)})}_{\text{Prediction on input } \mathbf{x}^{(i)}}) = \frac{1}{N}\sum_{i=1}^{N}\left(y^{(i)} - \underbrace{(w_0 + w_1\mathbf{x}^{(i)})}_{\text{Prediction on input } \mathbf{x}^{(i)}}\right)^2$$

MSE over the training data is called the "in sample" error.

Generalization Error

$$E_{\text{out}}(w_0, w_1) = E_{\mathbf{x},y}\left[\text{error}(y, g(\mathbf{x}))\right]$$

Expected error when the model is used on new examples. It is called the "out of sample error".

We cannot compute this value.

Expectation taken over all possible input/labels and the probability that input/label is seen

Assumption is training data is from the same distribution as the hypothesis will be used

Outline

- Motivating example: What polynomial degree should a [...]e?
- Polynomial transformation
- Underfitting and overfitting
- Understanding error: Bias and varian[...]
- Learning curves
- validation and model selection
- Model selection (with limit[...]
- K-fold cross validation
- Regularization

Yea!

Uh oh….

How to create a more complex hypothesis

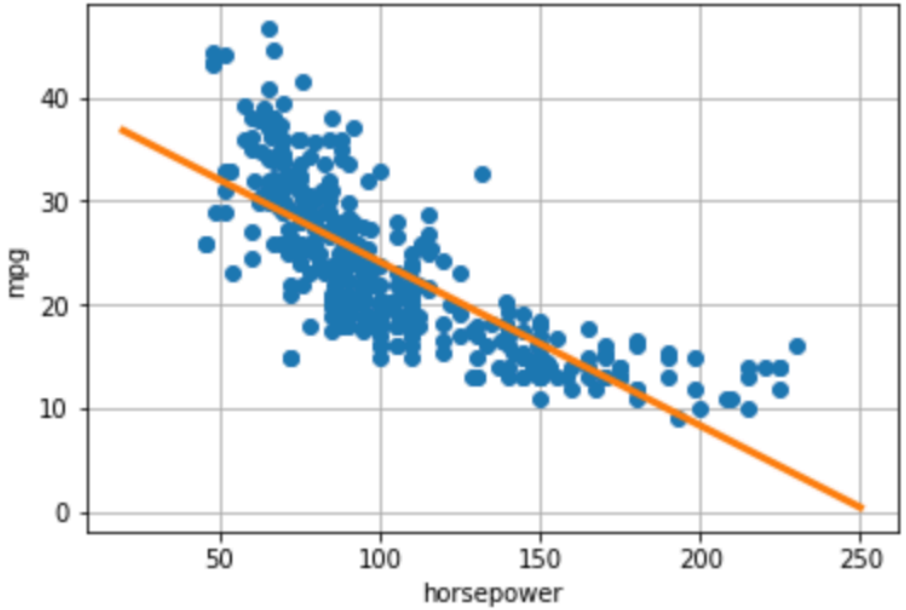
Understanding what went wrong

Understanding where the error comes from, and how to estimate $E_{\text{out}}[g(\mathbf{x})]$

Our strategy

If we have many different hypothesis classes to choose from - how can we choose wisely? And how can we estimate $E_{\text{out}}[g(\mathbf{x})]$?

Estimating Automobile MPG



- ❑ Found best line/hyperplane

$$\hat{y} = \mathbf{w}^T \mathbf{x}$$

- ❑ Shape appear to be nonlinear…
- ❑ To reduce E_{in} (RSS) we need something non-linear…

How can we get a non-linear hypothesis ***easily***?

Outline

- ❑Motivating example: What polynomial degree should a model use?
- ❑Polynomial transformation
- ❑Underfitting and overfitting
- ❑Understanding error: Bias and variance and noise
- ❑Learning curves
- ❑validation
- ❑validation and model selection
- ❑Model selection (with limited data)

How to create a more complex hypothesis

Understanding where the error comes from and how to estimate $E_{\text{out}}[g(\mathbf{x})]$

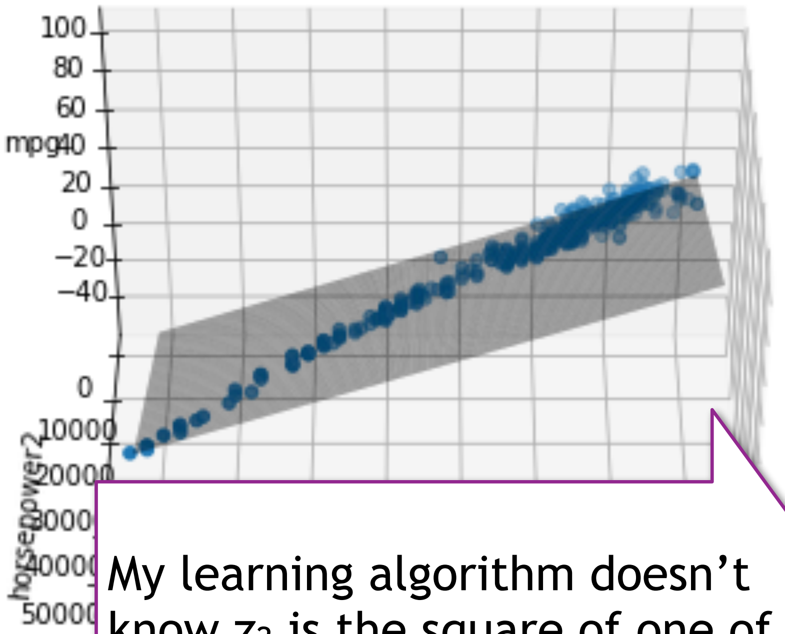
$g(\mathbf{x})$ is our hypothesis

If we have many different hypothesis classes to choose from - how can we choose wisely? And what is the error of the hypothesis we chose?

- ❑Regularization

A better hypothesis:

```
The R² value is  0.69 which is
better our previous R² value 0.53
```



My learning algorithm doesn't know z_2 is the square of one of my original features (horsepower²). The learning algorithm only sees feature z_2

z_1 = horsepower
z_2 = horsepower^2

Trained my linear model on these features: z_1 and z_2
(aka horsepower and horsepower²)

$$\mathbf{x} \to \mathbf{z} = \Phi(\mathbf{x}) = \begin{bmatrix} 1 \\ \phi_1(\mathbf{x}) \\ \phi_3(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix} = \begin{bmatrix} 1 \\ z_1 \\ z_2 \end{bmatrix}$$

Learn in **z** space with $\tilde{\mathbf{w}} = [\tilde{w}_0, \tilde{w}_1, \tilde{w}_2]$

Predict in **z** space $\hat{y} = g(\mathbf{x}) = \tilde{g}(\Phi(\mathbf{x})) = \tilde{g}(\mathbf{z}) = \tilde{\mathbf{w}}^T\mathbf{z}$

$$\hat{y} = \underbrace{56.9}_{\tilde{w}_0} \cdot \underbrace{1}_{z_0} + \underbrace{(-0.466)}_{\tilde{w}_1} \cdot \underbrace{x}_{z_1} + \underbrace{0.00123}_{\tilde{w}_2} \cdot \underbrace{x^2}_{z_2}$$

$$= 56.9 \cdot 1 + (-0.466) \cdot \phi_1(\mathbf{x}) + 0.00123 \cdot \phi_2(\mathbf{x})$$

What is the feature vector in z-space of a car whose horsepower is 170 ?

$[170]$

$[1, 170]^T$

$[1, 170, 170^2]^T$

None of the above

$$X = \begin{array}{c} \begin{array}{cc} & \text{Horsepower} \end{array} \\ \begin{bmatrix} 1 & 130.0 \\ 1 & 165.0 \\ 1 & 150.0 \\ 1 & 150.0 \\ 1 & 140.0 \\ 1 & 198.0 \\ 1 & 220.0 \\ 1 & 215.0 \\ \ldots & \ldots \end{bmatrix} \end{array} \qquad \mathbf{y} = \begin{bmatrix} 18.0 \\ 15.0 \\ 18.0 \\ 16.0 \\ 17.0 \\ 15.0 \\ 14.0 \\ 14.0 \\ \ldots \end{bmatrix}$$

$$X = \begin{array}{c} \begin{array}{ccc} & \text{Horsepower} & \text{Horsepower2} \end{array} \\ \begin{bmatrix} 1 & 130.0 & 16900.0 \\ 1 & 165.0 & 27225.0 \\ 1 & 150.0 & 22500.0 \\ 1 & 150.0 & 22500.0 \\ 1 & 140.0 & 19600.0 \\ 1 & 198.0 & 39204.0 \\ 1 & 220.0 & 48400.0 \\ 1 & 215.0 & 46225.0 \\ \ldots & \ldots & \ldots \end{bmatrix} \end{array} \qquad \mathbf{y} = \begin{bmatrix} 18.0 \\ 15.0 \\ 18.0 \\ 16.0 \\ 17.0 \\ 15.0 \\ 14.0 \\ 14.0 \\ \ldots \end{bmatrix}$$

Using our closed form solution we calculate $\tilde{\mathbf{w}} = \begin{bmatrix} 56.9 \\ -0.466 \\ 0.00123 \end{bmatrix}$

To predict a new $\mathbf{x}$

- transform $\mathbf{x}$ to $\Phi(\mathbf{x})$=$\mathbf{z}$
- predict with $\tilde{\mathbf{w}}$ in $\mathbf{z}$-space

$$\hat{y} = \tilde{g}(\mathbf{z}) = \tilde{\mathbf{w}}^T\mathbf{z} = \tilde{\mathbf{w}}^T\Phi(\mathbf{x})$$

Estimated value of a car with horsepower = 170?

$$\tilde{\mathbf{w}}^T\Phi(\mathbf{x}) = \tilde{\mathbf{w}}^T \begin{bmatrix} 1 \\ 170 \\ 28900 \end{bmatrix} = [56.9 \quad -0.466 \quad 0.00123] \begin{bmatrix} 1 \\ 170 \\ 28900 \end{bmatrix}$$

$$X = \begin{array}{cc} & \text{Horsepower} \\ \left[\begin{array}{c} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ \dots \end{array}\right. & \left.\begin{array}{c} 130.0 \\ 165.0 \\ 150.0 \\ 150.0 \\ 140.0 \\ 198.0 \\ 220.0 \\ 215.0 \\ \dots \end{array}\right] \end{array} \qquad \mathbf{y} = \begin{bmatrix} 18.0 \\ \vdots \\ 14.0 \\ \dots \end{bmatrix}$$

My learning algorithm doesn't know the features are a transformation of my original features

$$X = \begin{array}{ccc} & \text{Horsepower} & \text{Horsepower2} \\ \left[\begin{array}{c} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ \dots \end{array}\right. & \begin{array}{c} 130.0 \\ 165.0 \\ 150.0 \\ 150.0 \\ 140.0 \\ 198.0 \\ 220.0 \\ 215.0 \\ \dots \end{array} & \left.\begin{array}{c} 16900.0 \\ 27225.0 \\ 22500.0 \\ 22500.0 \\ 19600.0 \\ 39204.0 \\ 48400.0 \\ 46225.0 \\ \dots \end{array}\right] \end{array} \qquad \mathbf{y} = \begin{bmatrix} 18.0 \\ 15.0 \\ 18.0 \\ 16.0 \\ 17.0 \\ 15.0 \\ 14.0 \\ 14.0 \\ \dots \end{bmatrix}$$

Using our closed form solution we calculate $\tilde{\mathbf{w}} = \begin{bmatrix} 56.9 \\ -0.466 \\ 0.00123 \end{bmatrix}$

To predict a new **x**

- transform **x** to $\Phi(\mathbf{x})=\mathbf{z}$
- predict with $\tilde{\mathbf{w}}$ in **z**-space

$$\hat{y} = \tilde{g}(\mathbf{z}) = \tilde{\mathbf{w}}^T\mathbf{z} = \tilde{\mathbf{w}}^T\Phi(\mathbf{x})$$

Estimated value of a car with horsepower = 170?

$$\tilde{\mathbf{w}}^T\Phi(\mathbf{x}) = \tilde{\mathbf{w}}^T \begin{bmatrix} 1 \\ 170 \\ 28900 \end{bmatrix} = [56.9 \quad -0.466 \quad 0.00123] \begin{bmatrix} 1 \\ 170 \\ 28900 \end{bmatrix}$$

The General Polynomial Transform Φ_k

Polynomial basis function
polynomial features

Example: The degree-k polynomial transform over two features $\mathbf{x} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix}$

k is a hyperparameter (i.e. not one of the decision variables being optimized when fitting the data)

$$\mathbf{z} = \Phi_1(\mathbf{x}) = \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ z_1 \\ z_2 \end{bmatrix} \quad \Phi_2(\mathbf{x}) = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ x_1^2 \\ x_1 x_2 \\ x_2^2 \end{bmatrix} = \begin{bmatrix} 1 \\ z_1 \\ z_2 \\ z_3 \\ z_4 \\ z_5 \end{bmatrix} \quad \Phi_3(\mathbf{x}) = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ x_1^2 \\ x_1 x_2 \\ x_2^2 \\ x_1^3 \\ x_1^2 x_2 \\ x_1 x_2^2 \\ x_2^3 \end{bmatrix} = \begin{bmatrix} 1 \\ z_1 \\ z_2 \\ z_3 \\ z_4 \\ z_5 \\ z_6 \\ z_7 \\ z_8 \\ z_9 \end{bmatrix} \quad \Phi_4(\mathbf{x}) = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ x_1^2 \\ x_1 x_2 \\ x_2^2 \\ x_1^3 \\ x_1^2 x_2 \\ x_1 x_2^2 \\ x_2^3 \\ x_1^4 \\ x_1^3 x_2 \\ x_1^2 x_2^2 \\ x_1 x_2^3 \\ x_2^4 \end{bmatrix} = \begin{bmatrix} 1 \\ z_1 \\ z_2 \\ z_3 \\ z_4 \\ z_5 \\ z_6 \\ z_7 \\ z_8 \\ z_9 \\ z_{10} \\ z_{11} \\ z_{12} \\ z_{13} \\ z_{14} \end{bmatrix} \quad \ldots$$

Square of x_2

And so on

Dimensionality of the features space increases rapidly

No weights in this space

Slide inspired by (modified from) Malik Magdon-Ismail's slide

General Feature Transform

$\Phi : \mathbb{R}^d \to \mathbb{R}^{\tilde{d}}$ is also called a **feature map**

Any function of the original features could be used

$\mathcal{X}$ – space is $\mathbb{R}^d$

Z – space is $\mathbb{R}^{\tilde{d}}$

$$\mathbf{x}^{(i)} = \begin{bmatrix} 1 \\ x_1^{(i)} \\ x_2^{(i)} \\ \cdots \\ x_d^{(i)} \end{bmatrix}$$

$$\Phi(\mathbf{x}^{(i)}) = \mathbf{z}^{(i)} = \begin{bmatrix} 1 \\ \phi_1(\mathbf{x}^{(i)}) \\ \phi_2(\mathbf{x}^{(i)}) \\ \vdots \\ \phi_{\tilde{d}}(\mathbf{x}^{(i)}) \end{bmatrix} = \begin{bmatrix} 1 \\ z_1^{(i)} \\ z_2^{(i)} \\ \vdots \\ z_{\tilde{d}}^{(i)} \end{bmatrix}$$

Training data : $(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \ldots, (\mathbf{x}^{(N)}, y^{(N)})$

$(\mathbf{z}^{(1)}, y^{(1)}), (\mathbf{z}^{(2)}, y^{(2)}), \ldots, (\mathbf{z}^{(N)}, y^{(N)})$

No weights in original space

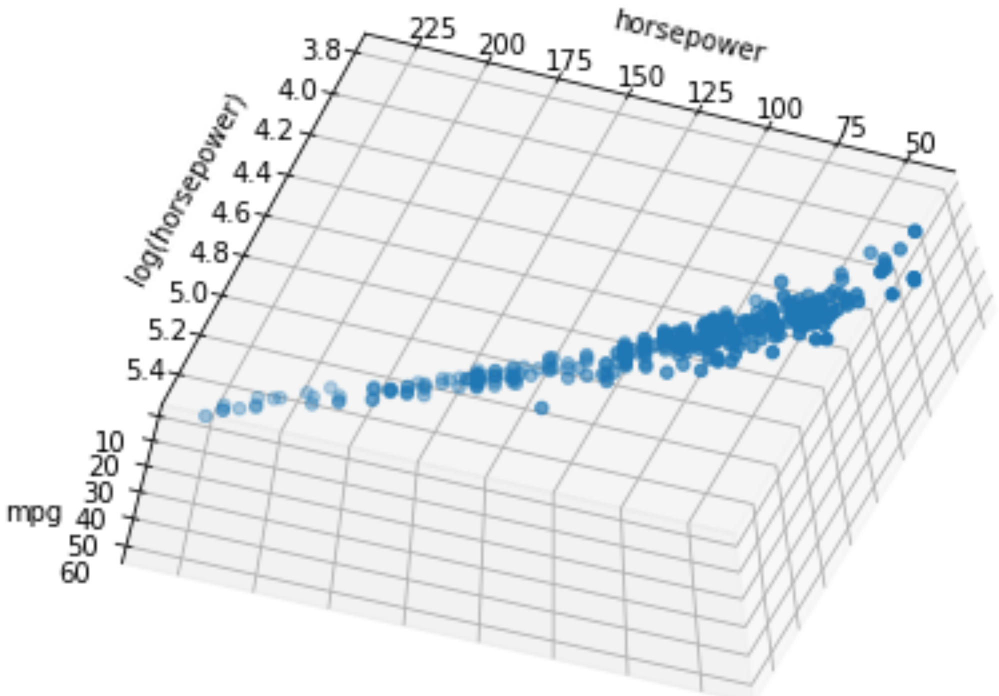
$$\hat{y} = \tilde{g}(\mathbf{z}) = \tilde{\mathbf{w}}^T\mathbf{z} \;= \tilde{\mathbf{w}}^T\Phi(\mathbf{x})$$

$$\tilde{\mathbf{w}} = \begin{bmatrix} \tilde{w}_0 \\ \tilde{w}_1 \\ \vdots \\ \tilde{w}_{\tilde{d}} \end{bmatrix}$$

We form a linear combination of the ϕ_j thus they are called **basis functions**

Many nonlinear features may work

$$\mathbf{x}^{(i)} \to \mathbf{z}^{(i)} = \Phi(\mathbf{x}^{(i)}) = \begin{bmatrix} 1 \\ \phi_1(\mathbf{x}^{(i)}) \\ \phi_2(\mathbf{x}^{(i)}) \end{bmatrix} = \begin{bmatrix} 1 \\ z_1^{(i)} = x_1^{(i)} \\ z_2^{(i)} = \log(x_1^{(i)}) \end{bmatrix}$$



$$\tilde{\mathbf{w}} = [156.04, 0.12, -31.60]^T$$

The R^2 value is 0.68

Polynomial Regression

- Models the relationship between the response and features as an d^th order polynomial $y = w_0 + w_1 x + w_2 x^2 + w_3 x^3 + \cdots + w_d x^d + \epsilon$

Example is using only one feature (monomial)

- Observation: the higher the order of the polynomial, the more shapes you can fit!
  - costs:
    - computational complexity grows as the number of parameters grows
    - chance that you *model the noise* and not the underlying parameters increases - overfitting - lose generalization
- Warning! It is always possible to perfectly fit N points with a model of order (N-1). It is unlikely that such a model will provide knowledge of the unknown function or be able to predict as well on unseen data as a lower order polynomial

How can we choose which (if any) transformation to use?

Outline

- Motivating example: What polynomial degree should a [occluded]e?
- Polynomial transformation
- Underfitting and overfitting
- Understanding error: Bias and varian[occluded]
- Learning curves
- validation and model selection
- Model selection (with limit[occluded]
- K-fold cross validation

Yea!

Uh oh….

How to create a more complex hypothesis

Understanding what went wrong

Understanding where the error comes from, and how to estimate $E_{\text{out}}[g(\mathbf{x})]$

Our strategy

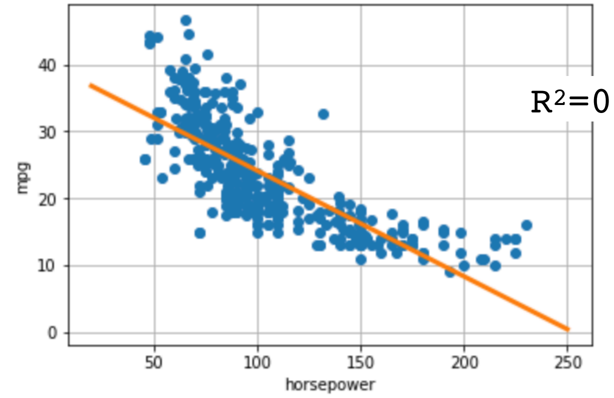
If we have many different hypothesis classes to choose from - how can we choose wisely? And how can we estimate $E_{\text{out}}[g(\mathbf{x})]$?

- Regularization

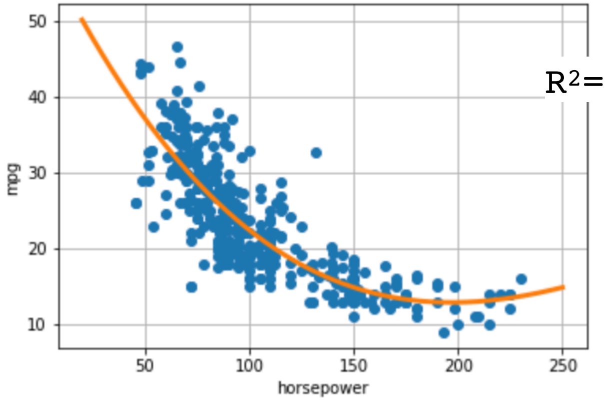
Automobile MPG

- As we increase the degree of the polynomial, do we improve the fit of the model to the data?

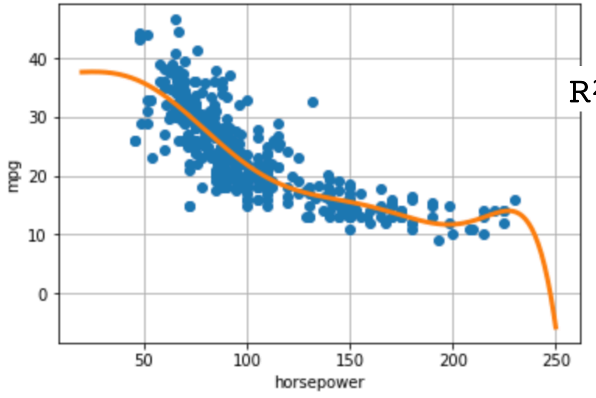
$$y = w_0 + w_1 x + w_2 x^2 + w_3 x^3 + \cdots + w_d x^d + \epsilon$$



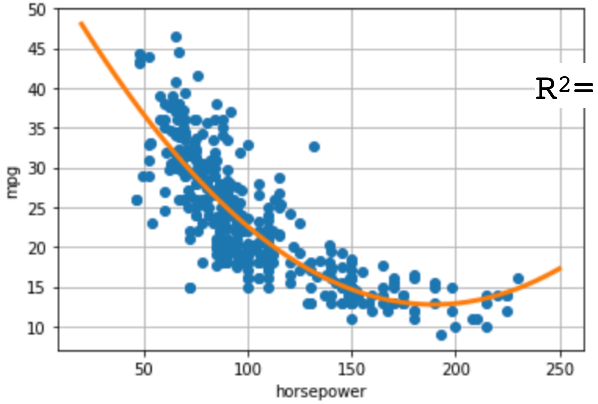
R^2=0.53



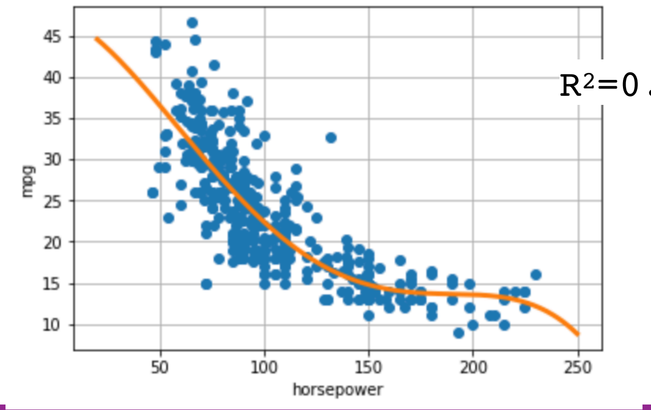
R^2=0.688214



R^2=0.694398



R^2=0.687559

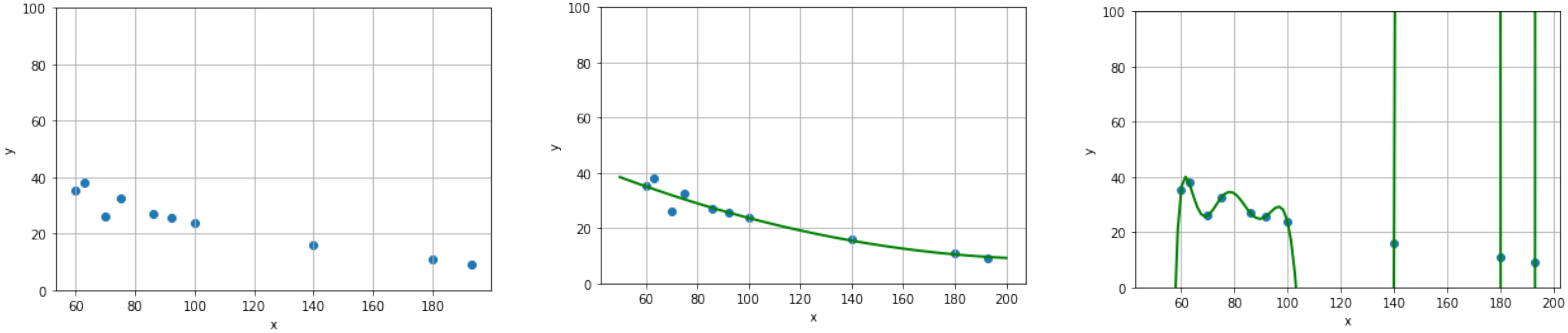


R^2=0.689343

We can keep improving our wrt our training data - but does that mean we would do better on examples not in the training data?

Example

```
mean = 23.4710191083
```



$E_{\text{in}} \approx 0; \quad E_{\text{out}} \gg 0$

Overfitting: Complex hypothesis that fits the training data too well.
It predicts well on patterns found in training data that won't be found in the the future data

What is the goal of machine learning?

Find a hypothesis, $g(\mathbf{x})$ that predicts the correct value of y for new values of $\mathbf{x}$ (i.e. not in our training set).

Let $E_{out}(g)$ be the expected error our hypothesis will incur in the future. For linear regression we incurred a loss of $(g(\mathbf{x}) - y)^2$ Thus for linear regression the expected error over the input space is $E_{out}(g) = E\left[\left(g(\mathbf{x}) - y)\right)^2\right]$

We cannot compute $E_{out}(g)$.

We have been focused on minimizing $E_{in}(g)$, the cost of our training set.

For linear regression $E_{in}(g) = \frac{1}{N}\sum_{i=1}^{N}\left(g(\mathbf{x}^{(i)}) - y^{(i)}\right)^2 = \frac{1}{N}\sum_{i=1}^{N}\left(\mathbf{w}^T\mathbf{x}^{(i)} - y^{(i)}\right)^2.$

What is the goal of machine learning?

Find a hypothesis, $g(\mathbf{x})$ that predicts the correct value of y for new values of $\mathbf{x}$ (i.e. not in our training set).

Also called risk of a hypothesis (model)

Let $E_{out}(g)$ b pothesis will incur in the future. For linear regressio $(\mathbf{x}) - y)^2$ Th

Expectation taken over all possible input/labels and the probability that input/label is seen

expected error over the input space is $E_{out}(g) =$

We cannot compute $E_{out}(g)$.

We have been focused on minimizing $E_{in}(g)$, the cost of our training set.

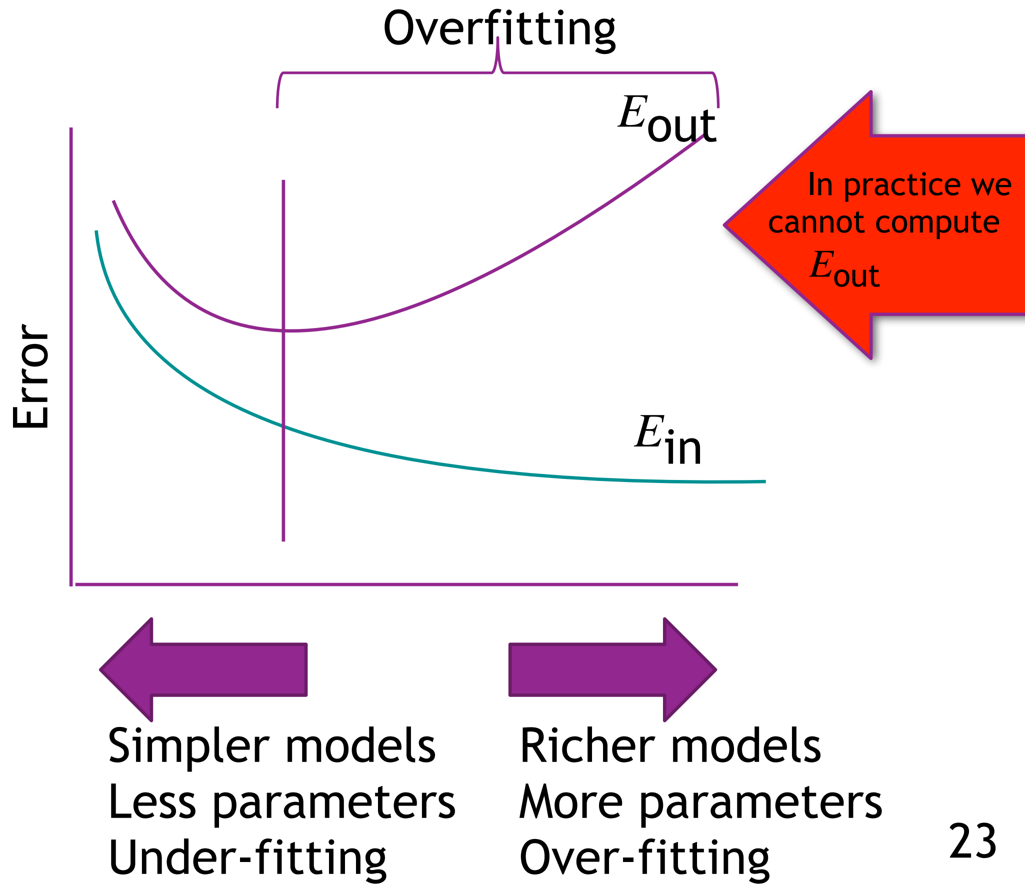
For linear regression $E_{in}(g) = \dfrac{1}{N}\sum_{i=1}^{N}\left(g(\mathbf{x}^{(i)}) - y^{(i)}\right)^2 = \dfrac{1}{N}\sum_{i=1}^{N}\left(\mathbf{w}^T\mathbf{x}^{(i)} - y^{(i)}\right)^2.$

What can go wrong with choosing the hypothesis which has the smallest lost/cost

1. Limited Hypothesis class (model class). No function in our hypothesis class can model the data well - biased solution
2. Limited Data. We might model the noise and not the true pattern. Small changes to the data causes the hypothesis (model) to change - high variance solution

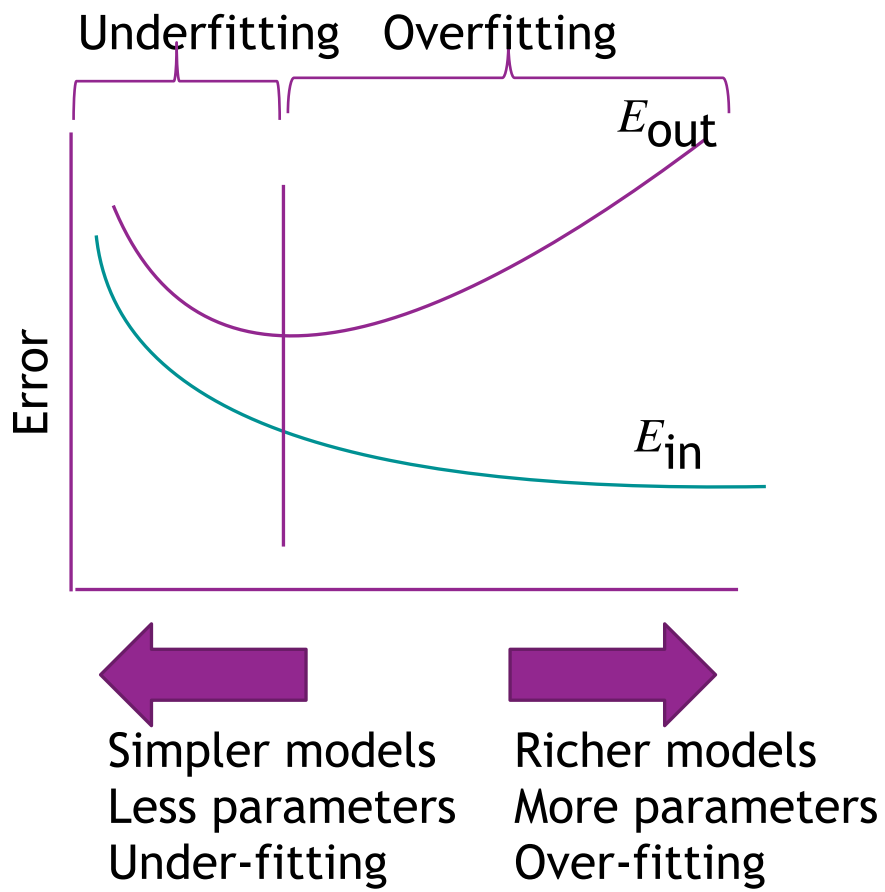
Overfitting

- If we allow a very flexible model by using complicated features or model, our model can be *too* complicated. The regression model becomes tailored to fit the noise of the training data and does not generalize well (i.e. predict well on data not in the training set, and does not accurately describe the relationship between the parameters and the outcome)
- A too complicated model will not generally well.
- ***overfitting:*** The model performs worse on unseen data than a different model from the same class despite performing better on the training data
- Example: Using a degree d=N-1 polynomial transformation
- Training RSS (or MSE), R^2 is not a good indicator of test RSS (or MSE), R^2



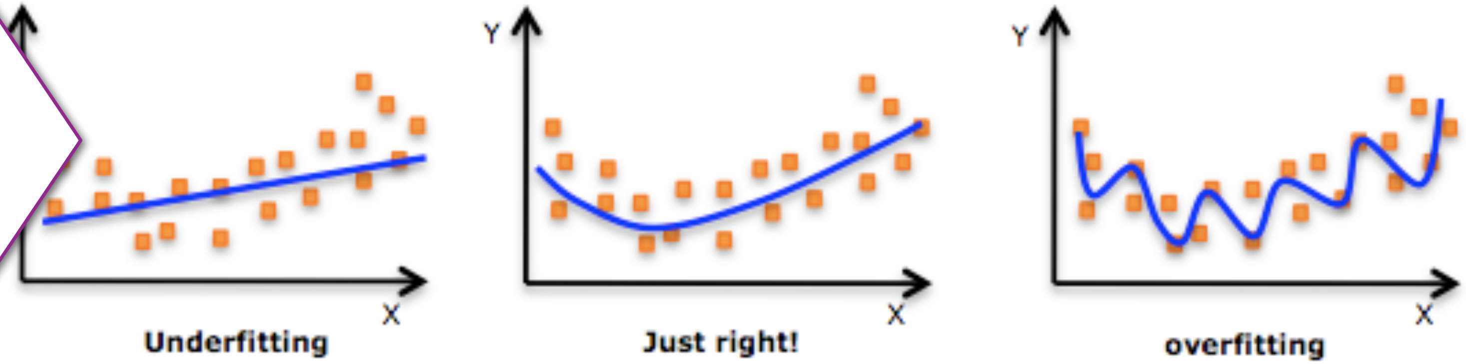
Underfitting

- The model learned does not do well on the training data and does not do well on unseen examples
- A too simple model is called ***underfitting***
- Example: predicting mean of target



How Can You Tell from Data?

Model is too inflexible, it is not fitting the pattern



Model is too flexible, it is fitting the noise

- Is there a way to tell what is the correct model order to use?
- Must use the data. Do not have access to the true d?
- What happens if we guess:
  - d too big?
  - d too small?

Question

For the examples below, might we encounter a problem with the model we chose?

- Examples:
  - True function $f(x) = 2 + 3x$ model class $w_0 + w_1 x + w_2 x^2$
  - True function $f(x) = 2 + 3x + 4x^2$ model class $w_0 + w_1 x$

Outline

- Motivating example: What polynomial degree should a ...e?
- Polynomial transformation
- Underfitting and overfitting
- Understanding error: Bias and varianc...
- Learning curves
- validation and model selection
- Model selection (with limit...
- K-fold cross validation
- Regularization

Yea!

Uh oh....

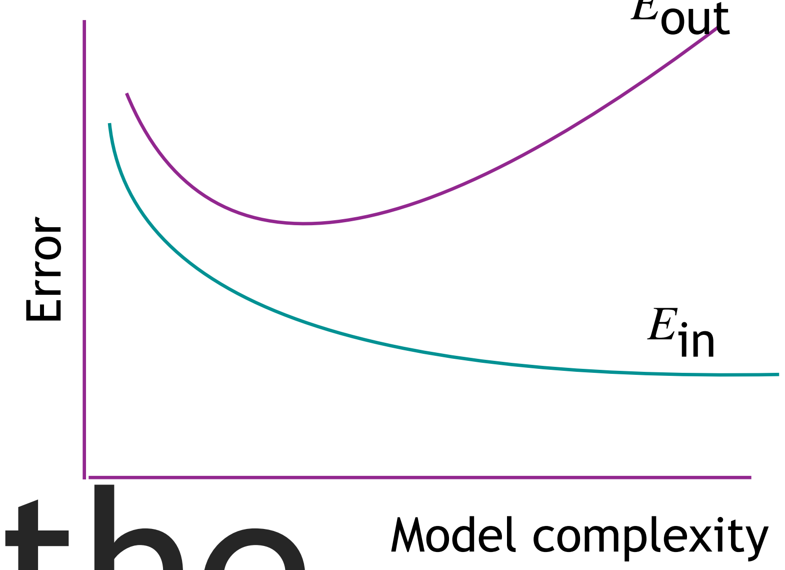
How to create a more complex hypothesis

Understanding what went wrong

Understanding where the error comes from, and how to estimate $E_{\text{out}}[g(\mathbf{x})]$

Our strategy

If we have many different hypothesis classes to choose from - how can we choose wisely? And how can we estimate $E_{\text{out}}[g(\mathbf{x})]$?



Understanding the errors

THEORETICAL METRIC

How do we evaluate our model? Or choose among models (e.g. the which polynomial transformation should we choose?)

Open discussion

- We can evaluate how well it works by looking at its errors
- We would like the error to be zero on all future data. However:
  - The unseen variables means the true model has non-zero error (i.e. the world is a messy place) — Noise
  - Our hypothesis probably doesn't contain the underlying true model — Bias
  - We don't get enough data to perfectly estimate our model. We only get a finite sample of the data. The more data we receive, the more our sample is representative of underlying data and our estimates should converge — Variance

We focus on Algorithm Bias

Understanding Error $E_{out}(g^{(D)}(\mathbf{x})) = E_{\mathbf{x},y}[(y - g^{(D)}(\mathbf{x}))^2]$

Bias-Variance-Noise Decomposition

In predictions there are three sources of error

1. noise - irreducible error
2. bias - error of average hypothesis (estimated from N examples) from the true function f($\mathbf{x}$)+ϵ
3. variance - how much would the prediction for an example change if the hypothesis was fit on a different set of N points

Proof from:https://ocw.mit.edu/courses/sloan-school-of-management/15-097-prediction-machine-learning-and-statistics-spring-2012/lecture-notes/MIT15_097S12_lec04.pdf

We focus on Algorithm Bias

Understanding Error

$$E_{out}(g^{(D)}(\mathbf{x})) = E_{\mathbf{x},y}[(y - g^{(D)}(\mathbf{x}))^2]$$

Our definitions will be for th
squared loss function
You can think of how to
substitute other loss functions

Bias-Variance-Noise Decomposition

This cannot be computed in practice
because we do not have access to the target
function or the probability distribution

In predictions there are three sources of error

1. noise - irreducible error
2. bias - error of average hypothesis (estimated from N examples) from the tru function $f(\mathbf{x})+\epsilon$
3. variance - how much would the prediction for an example change if the hypothesis was fit on a different set of N points

Proof from:https://ocw.mit.edu/courses/sloan-school-of-management/15-097-prediction-machine-learning-and-statistics-spring-2012/lecture-notes/MIT15_097S12_lec04.pdf

Outline

- ❑Motivating example: What polynomial degree should a ?
- ❑Polynomial transformation
- ❑Underfitting and overfitting
- ❑Understanding error: Bias and variance and noise
  - Bias
  - Variance
  - Bias and variance and noise
- ❑Learning curves
- ❑validation
- ❑Model selection
- ❑Cross validation
- ❑Regularization

Yea!

Uh oh….

How to create a more complex hypothesis

Understanding where the error comes from and how to estimate $E_{\text{out}}[g(\mathbf{x})]$

Understanding what went wrong

Our strategy

If we have many different hypothesis classes to choose from - how can we choose wisely? And how to estimate $E_{\text{out}}[g(\mathbf{x})]$?

Average Hypothesis

- Given: N training examples $D = \{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \ldots, (x^{(N)}, y^{(N)})\}$
- Learn: If I had a different set of N training examples, I would get a different hypothesis (models) $g^{(D)}(\mathbf{x})$
- Expected prediction (averaged over hypothesis): $\bar{g}(\mathbf{x}) = E_D[g^{(D)}(\mathbf{x})]$

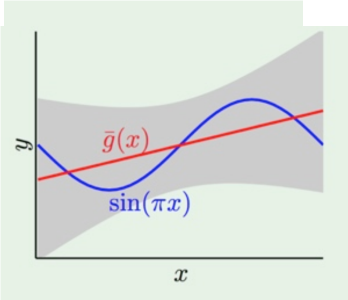
Mean prediction of the algorithm for $\mathbf{x}$

Intuitive approximation:

$$\bar{g}(\mathbf{x}) \approx \frac{1}{k}\sum_{i=1}^{k} g_i^{(D_i)}(\mathbf{x}) \quad D_1, D_2, \ldots, D_k$$

Bias

Bias of the hypothesis class (not an individual hypothesis from the class)



- bias($\mathbf{x}$) = $\left(f(\mathbf{x}) - \bar{g}(\mathbf{x})\right)^2$

Conceptually: squared difference from "average prediction" for $\mathbf{x}$, and expected label $f(\mathbf{x})$

- bias $= E_{\mathbf{x}}\left[(\bar{g}(\mathbf{x}) - f(\mathbf{x}))^2\right]$

$$\approx \frac{1}{N}\sum_{i=1}^{N}(\bar{g}(\mathbf{x}^{(i)}) - f(\mathbf{x}^{(i)}))^2$$

Bias of the hypothesis class

less flexible model then more bias

Occasionally this is called bias^2

When using this model class, measures how well you expect the "average prediction" to represent the true solution
We expect the bias to decreases with a more complex model

Outline

- ❑Motivating example: What polynomial degree should a ?
- ❑Polynomial transformation
- ❑Underfitting and overfitting
- ❑Understanding error: Bias and variance and noise
  - Bias
  - Variance
  - Bias and variance and noise
- ❑Learning curves
- ❑validation
- ❑Model selection
- ❑Cross validation
- ❑Regularization

Yea!

Uh oh….

How to create a more complex hypothesis

Understanding what went wrong

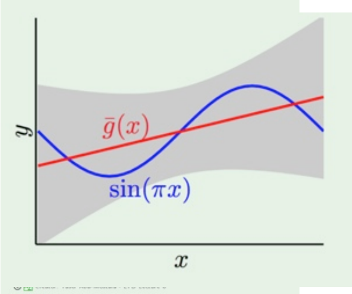
Understanding where the error comes from and how to estimate $E_{\text{out}}[g(\mathbf{x})]$

Our strategy

If we have many different hypothesis classes to choose from - how can we choose wisely? And how to estimate $E_{\text{out}}[g(\mathbf{x})]$?

Variance

Variance of a hypothesis class (model class)



- Variance: difference between the expected prediction and the prediction from a particular dataset
- $\text{var}(\mathbf{x}) = E_D[(g^{(D)}(\mathbf{x}) - \bar{g}(\mathbf{x}))^2] \approx \frac{1}{L}\sum_{\ell=1}^{L}(\bar{g}(\mathbf{x}) - g_\ell^{(D_\ell)}(\mathbf{x}))^2$

Conceptually: variance of a prediction for $\mathbf{x}$ from the mean prediction

$$\text{var} = E_{\mathbf{x}}\left[E_D\left[(g^{(D)}(\mathbf{x}) - \bar{g}(\mathbf{x}))^2\right]\right] \approx \frac{1}{N}\sum_{i=1}^{N}\frac{1}{L}\sum_{\ell=1}^{L}(\bar{g}(\mathbf{x}^{(i)}) - g_\ell^{(D_\ell)}(\mathbf{x}^{(i)}))^2$$

less flexible model then less variance

Measures how sensitive a hypothesis class (model class) is to a specific dataset
Variance typically decreases with simpler models

Outline

- ❑Motivating example: What polynomial degree should a ?
- ❑Polynomial transformation
- ❑Underfitting and overfitting
- ❑Understanding error: Bias and variance and noise
  - •Bias
  - •Variance
  - •Bias and variance and noise
- ❑Learning curves
- ❑validation
- ❑Model selection
- ❑Cross validation
- ❑Regularization

Yea!

Uh oh....

How to create a more complex hypothesis

Understanding where the error comes from and how to estimate $E_{\text{out}}[g(\mathbf{x})]$

Understanding what went wrong

Our strategy

If we have many different hypothesis classes to choose from - how can we choose wisely? And how to estimate $E_{\text{out}}[g(\mathbf{x})]$?

Where did the prediction error in our hypothesis come from?

- Regression example: $y = f(\mathbf{x}) + \epsilon$

Deterministic

Noise $\sim N(0,\sigma)$
We are assuming the noise has mean 0 and variance σ^2

This means $E_{\mathbf{x},y}[f(\mathbf{x}) - y] = 0$ and $E_{\mathbf{x},y}[(f(\mathbf{x}) - y)^2] = E_{\mathbf{x}}(\epsilon^2) = \sigma^2$

- Goal is to understand why our *expected* hypothesis (model) does not have zero error

$$E_D\left[E_{\text{out}}(g^{(D)})\right] = E_D\left[E_{\mathbf{x},y}[(g^{(D)}(\mathbf{x}) - y)^2]\right] \neq 0$$

$E_{\text{out}}(g^{(D)})$

$E_{\mathbf{x},y}[(g^{(D)}(\mathbf{x}) - y)^2]$
expected error for they hypothesis $g^{(D)}(\mathbf{x})$

The expected error of the hypothesis fit using the data set D on any future example

Understanding Error
Bias-Variance Decomposition (noise free)

$$\text{bias}(\mathbf{x}) = \left(f(\mathbf{x}) - \bar{g}(\mathbf{x})\right)^2$$

$$\text{var}(\mathbf{x}) = E_D[\,(g^{(D)}(\mathbf{x}) - \bar{g}(\mathbf{x}))^2]$$

$$\bar{g}(\mathbf{x}) = E_D[g^{(D)}(\mathbf{x})]$$

For any constant c,
E[ac]=cE[a]
E[a+c]=E[a]+c

The linearity of expectation:
E[a + b] = E[a] + E[b]

$$E_D\left[E_{\mathbf{x}}[\left(g^{(D)}(\mathbf{x}) - f(\mathbf{x})\right)^2]\right]$$

$$E_{\mathbf{x}}\left[E_D[\left(g^{(D)}(\mathbf{x}) - f(\mathbf{x})\right)^2]\right] = E_{\mathbf{x}}\left[E_D[(\underbrace{g^{(D)}(\mathbf{x}) - \bar{g}(\mathbf{x})}_{A} + \underbrace{\bar{g}(\mathbf{x}) - f(\mathbf{x})}_{B})^2]\right]$$

$$= E_{\mathbf{x}}\left[E_D[\left(g^{(D)}(\mathbf{x}) - \bar{g}(\mathbf{x})\right)^2 + 2\left(g^{(D)}(\mathbf{x}) - \bar{g}(\mathbf{x})\right)\left(\bar{g}(\mathbf{x}) - f(\mathbf{x})\right) + \left(\bar{g}(\mathbf{x}) - f(\mathbf{x})\right)^2]\right]$$

$$= E_{\mathbf{x}}\left[E_D[\left(g^{(D)}(\mathbf{x}) - \bar{g}(\mathbf{x})\right)^2] + 2E_D[\left(g^{(D)}(\mathbf{x}) - \bar{g}(\mathbf{x})\right)\left(\bar{g}(\mathbf{x}) - f(\mathbf{x})\right)] + E_D[\left(\bar{g}(\mathbf{x}) - f(\mathbf{x})\right)^2]\right]$$

$$= E_{\mathbf{x}}\left[\underbrace{E_D[\left(g^{(D)}(\mathbf{x}) - \bar{g}(\mathbf{x})\right)^2]}_{\text{variance}(\mathbf{x})} + 2E_D[\underbrace{\left(g^{(D)}(\mathbf{x}) - \bar{g}(\mathbf{x})\right)]}_{0}\left(\bar{g}(\mathbf{x}) - f(\mathbf{x})\right) + \underbrace{E_D[\left(\bar{g}(\mathbf{x}) - f(\mathbf{x})\right)^2]}_{\text{bias}(\mathbf{x})}\right]$$

$$= E_{\mathbf{x}}[\text{bias}] + E_{\mathbf{x}}[\text{variance}]$$

$$= \text{bias} + \text{variance}$$

Notice that

$$E_D[\left(g^{(D)}(\mathbf{x}) - \bar{g}(\mathbf{x})\right)]$$

$$= E_D[g^{(D)}(\mathbf{x})] - \bar{g}(\mathbf{x})$$

Proof from:https://ocw.mit.edu/courses/sloan-school-of-management/15-097-prediction-machine-learning-and-statistics-spring-2012/lecture-notes/MIT15_097S12_lec04.pdf

Understanding Error
Bias-Variance-Noise Decomposition

The expected error of the hypothesis fit on a randomly chosen set of N training examples

$$E_{\text{out}}(g) \;=\; \text{bias} + \text{variance} + \text{noise}$$

Noise in the training set contributes to variance

Noise in the test set contributes to irreducible error

Based on averages over what is expected for a training set D
- can we lower variance without increasing too much the bias?
- can we lower bias without increasing too much the variance?