Machine Learning and Financial Applications

Lecture 4

Regularization and generalization

Liu Peng

liupeng@smu.edu.sg

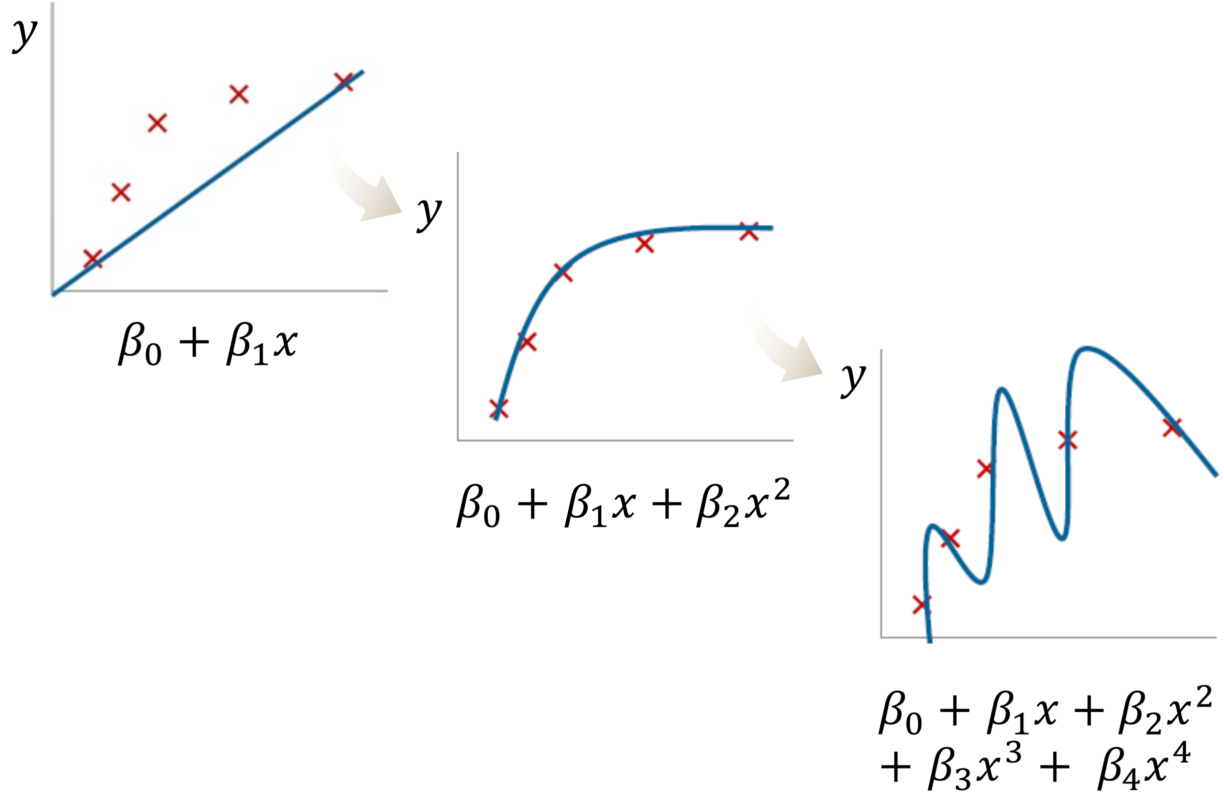
Video tutorial

- https://youtu.be/OyHfB_ZtI3w

Linear regression gone wild

Single input variable

- Polynomial with exponentials



Multiple input variables

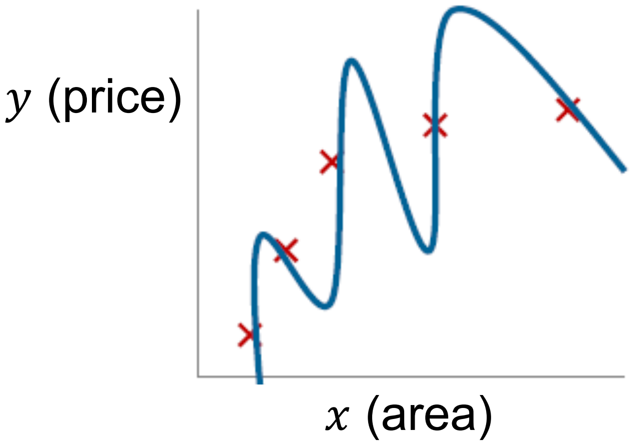
- Polynomial with second-order terms and interaction terms
- 3 input variables x_1, x_2, x_3
  - $x_1^2, x_2^2, x_3^2, x_1x_2, x_1x_3, x_2x_3$
  - 6 second-order terms
- 16 input variables
  - 136 second-order terms
  - No. of terms in the equation: 16 + 136 = 152
  - Do we have enough observations in the training data set?

Property price prediction: Bias vs. Variance

Bias: inherent error from the model even with infinite training data; "biased" to a particular kind of solution (e.g., simple linear regression below)

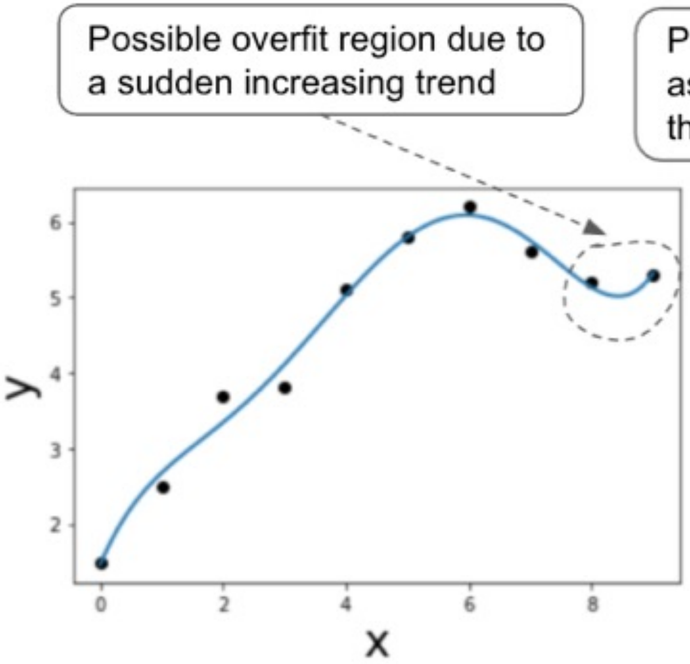
Variance: how much the model would change if a different training data set

- $y = \beta_0 + \beta_1 x$
- **High bias**: Perfect fit impossible even with infinite training data
- **Low variance**: if one row in the training data set changes, model does not change much
- **Underfitting**: fail to capture important characteristics in the data set



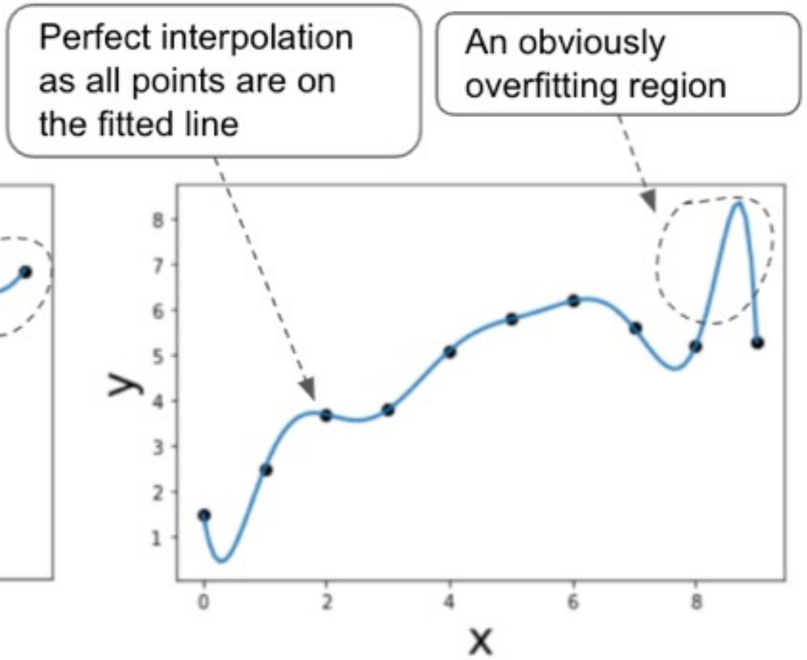
- $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 x^4$
- **Low bias**: Perfect fit is possible for training data
- **High variance**: if one row in the training data set changes, model changes a lot
- **Overfitting**: fails to generalize the solution for new data, e.g., testing data set

Increasing model complexity leads to overfitting



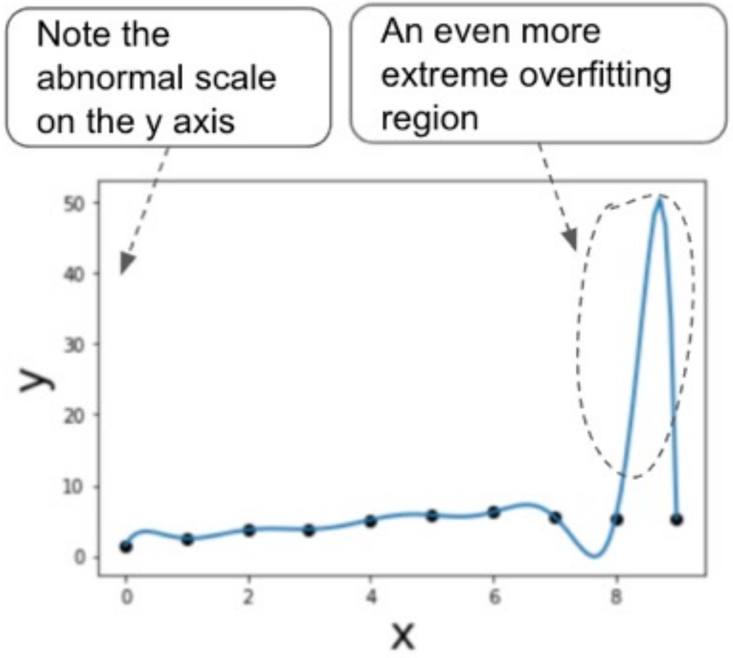
5th degree polynomial

$$Q(\mathbf{w}) = 0.3$$



10th degree polynomial

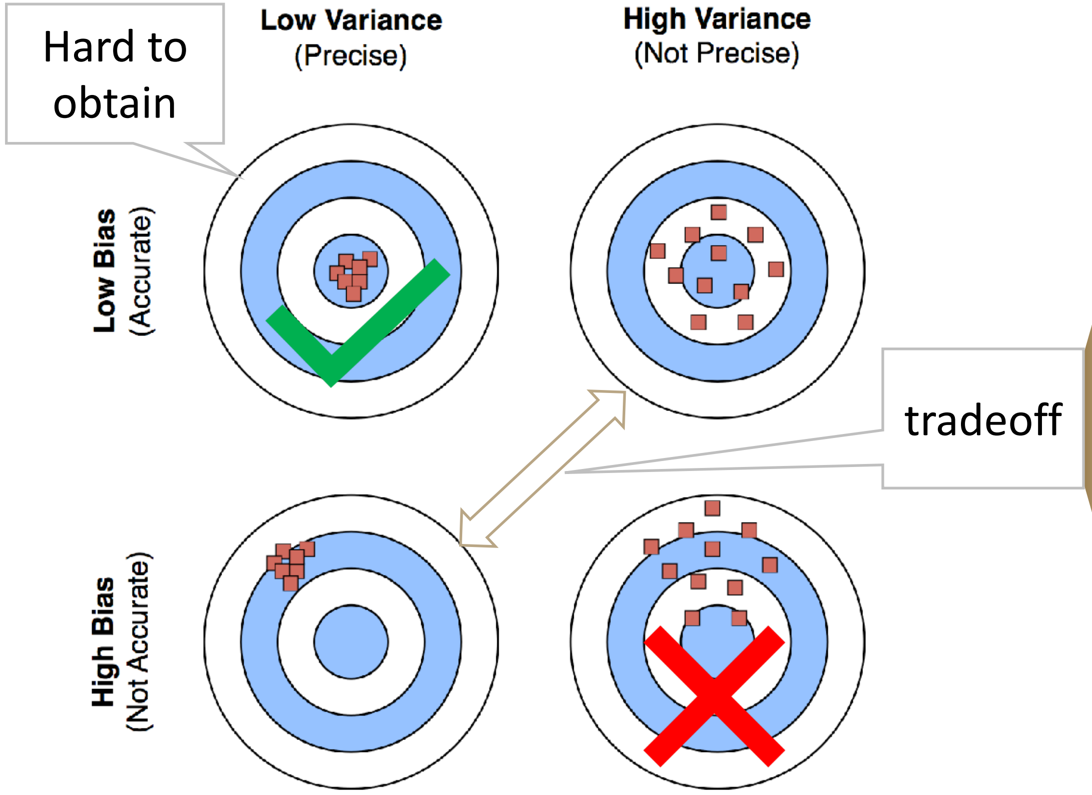
$$Q(\mathbf{w}) = 1.67 \times 10^{-9}$$

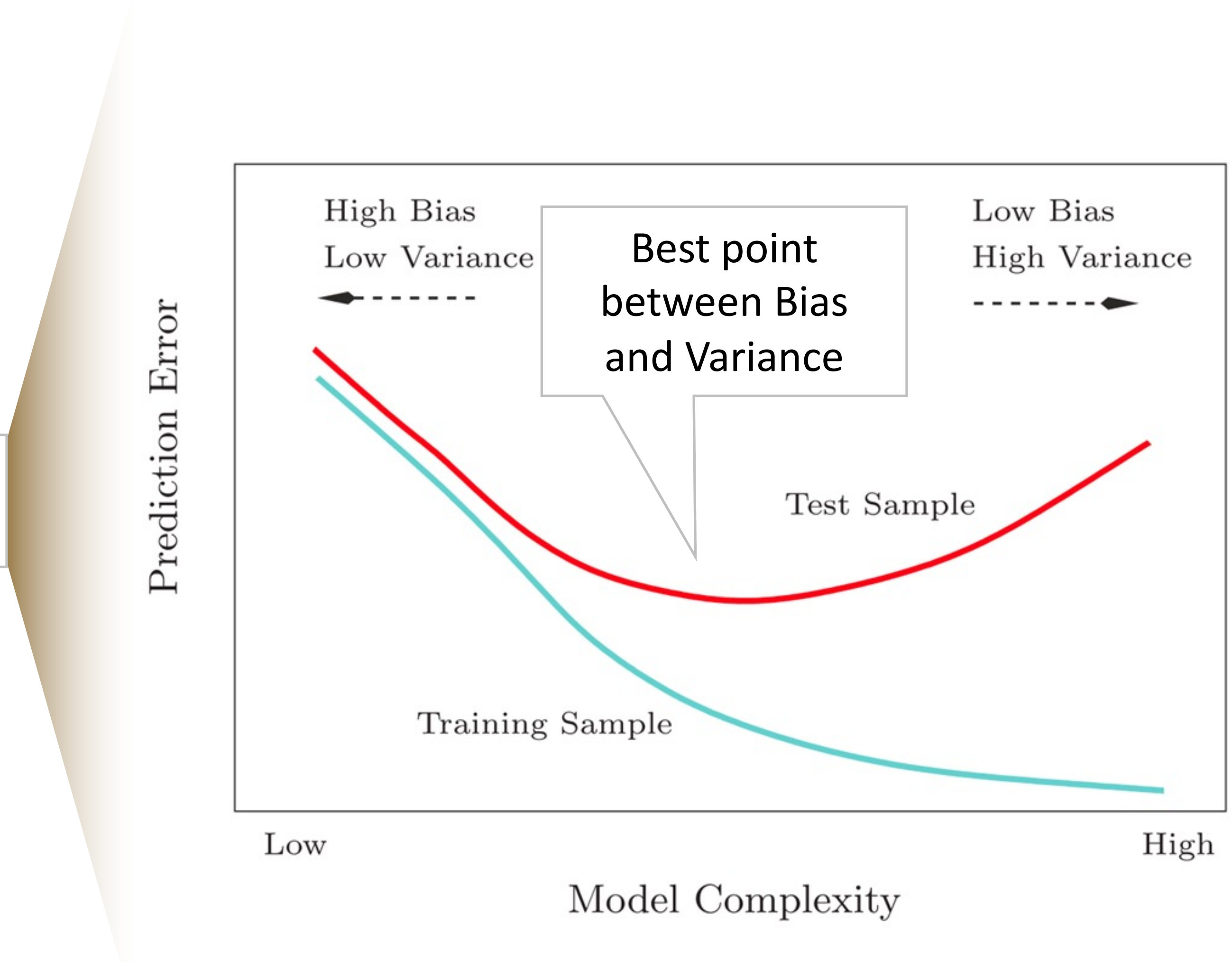


15th degree polynomial

$$Q(\mathbf{w}) = 3.02 \times 10^{-12}$$

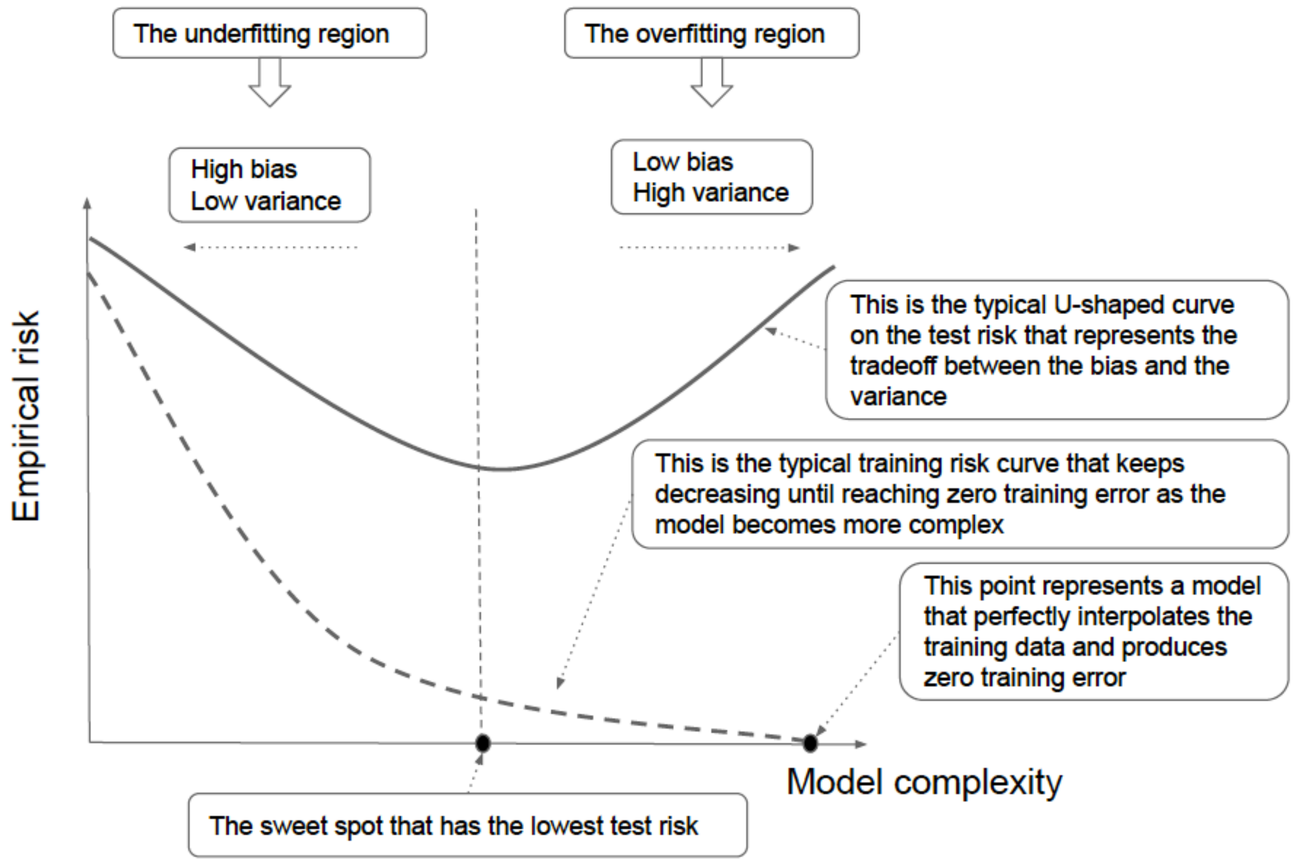
Bias-variance tradeoff



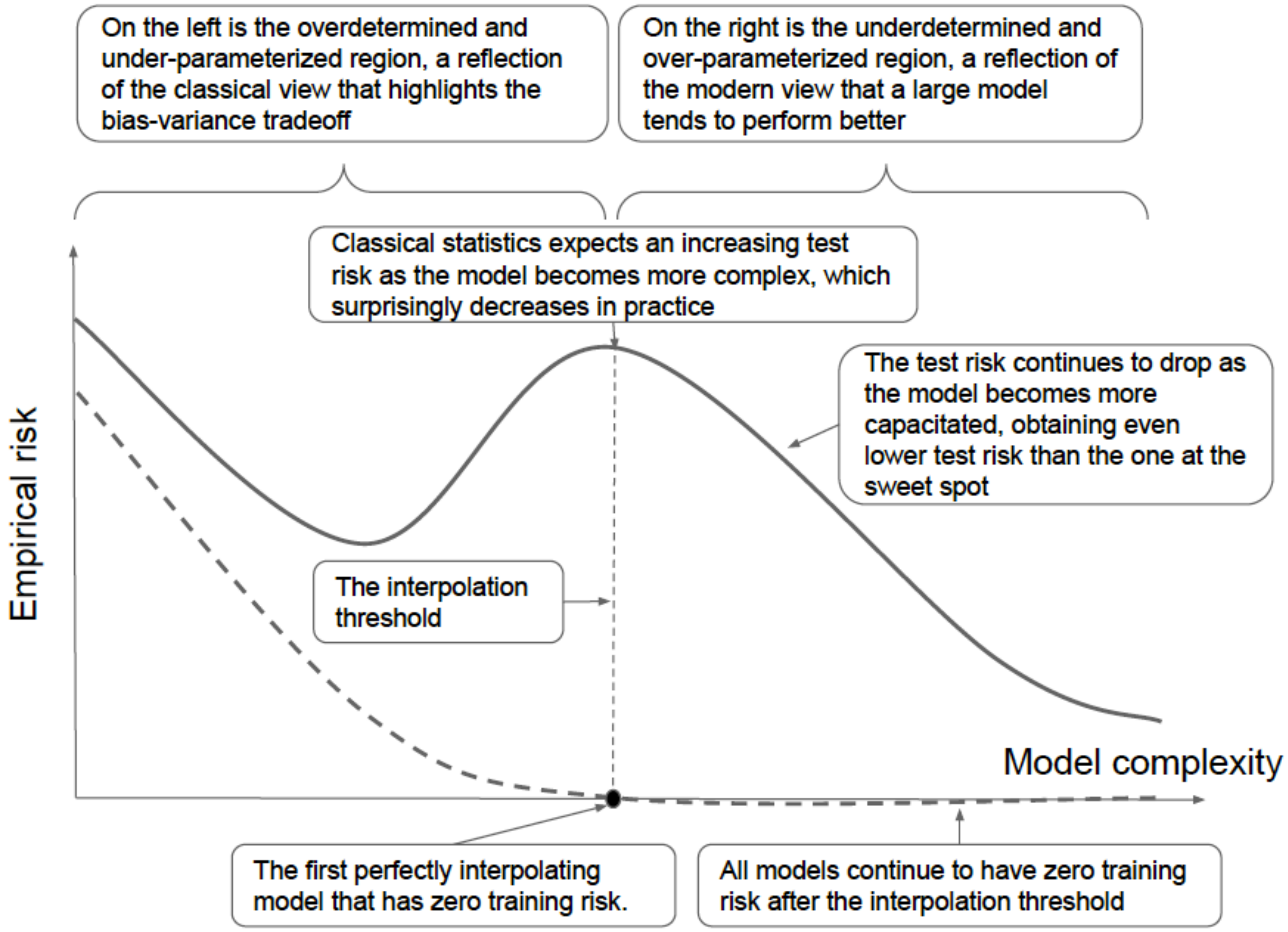


Exercise: check the difference between simple vs. complex model

More on bias-variance tradeoff



A modern view on double descent



Addressing overfitting

- Option 1: Reduce number of terms in the linear regression equation to **keep the model simple; principle of**
  - 1.1: Manually select input variables based on domain knowledge, e.g., area is essential for the prediction of property price, while no_of_convenience_stores is less important
  - 1.2: Model selection algorithm (more on next page)

- Option 2: **Regularization**
  - Keep all the input variables but **reduce magnitude of coefficients $\boldsymbol{\beta_i}$**; suitable for a data set with many input variables, all of which might contribute to predicting y

- Option 3: Dimension reduction (more in future)

Rule-of-thumb for linear regression

$$no.\ of\ terms \leq no.\ of\ observations/10$$

Option 1.2: Model selection algorithm to reduce number of terms

Statistical algorithm to select best subset of terms

- If no. of observations is 200, what is the max no. of terms for the regression equation?
- Choose 20 from 152 potential terms (first and second order) such that sum of squared residuals **(SSR) is minimized**
- Optimal selection is extremely hard to find
- Possible for medium size problem; not scalable to large problems

Approximation algorithm (forward/backward selection[1])

- Forward: starting from $\beta_0 = \bar{y}(mean\ of\ y)$, $\beta_1 = 0$, $\beta_2 = 0$ ..., we add one term at a time
- Backward: starting from OLS with all 152 potential terms, we delete one term at a time
- Certain stopping criterion to **approximately reach minimum SSR** (not exactly minimum SSR)
- For student's own exploration: scikit-learn compatible `mlxtend` package

Source: https://datascience.stackexchange.com/questions/24405/how-to-do-stepwise-regression-using-sklearn/24447#24447

Option 2: Regularization to minimize loss function with a penalty

- Restrict coefficients $\beta_1, \beta_2, \ldots, \beta_k$ to small values
- L1 norm: $|*|_1 = \sum_1^k |\beta_i| = |\beta_1| + |\beta_2| + \cdots + |\beta_k|$
- L2 norm: $|*|_2 = \sqrt{\sum_1^k \beta_i^2} = \sqrt{\beta_1^2 + \beta_2^2 + \cdots + \beta_k^2}$
- Penalty weight: α
- L1 regularization: LASSO (Least Absolute Shrinkage and Selection Operator)

$$min \underbrace{\frac{1}{2n}\sum_1^n (y_i - \hat{y}_i)^2}_{\text{Loss function}} + \underbrace{\alpha|*|_1}_{\text{Regularizer}}$$

- L2 regularization: Ridge (note the **squared** L2 norm for simplicity)

$$min \underbrace{\frac{1}{2n}\sum_1^k (y_i - \hat{y}_i)^2}_{\text{Loss function}} + \underbrace{\frac{1}{2}\alpha|*|_2^{2}}_{\text{Regularizer}}$$

More on penalty weight and intercept

- Penalty for being away from origin is given a weight α, forming the **regularizer**
- No penalty if $\beta_1, \beta_2, \ldots, \beta_k$ is all 0
- Large $\alpha \Rightarrow$ large penalty for being away from origin $\Rightarrow \beta_j$ are restricted to a small space
- Small $\alpha \Rightarrow$ small penalty for being away from origin $\Rightarrow \beta_j$ are restricted to a larger space

---

- Why no restriction on intercept β_0?
  - By not restricting β_0, when $\beta_1, \beta_2, \ldots, \beta_k$ is all 0, we recover

$$\min_{\beta_0} \frac{1}{2n} \sum_{1}^{n} (y_i - \beta_0)^2$$

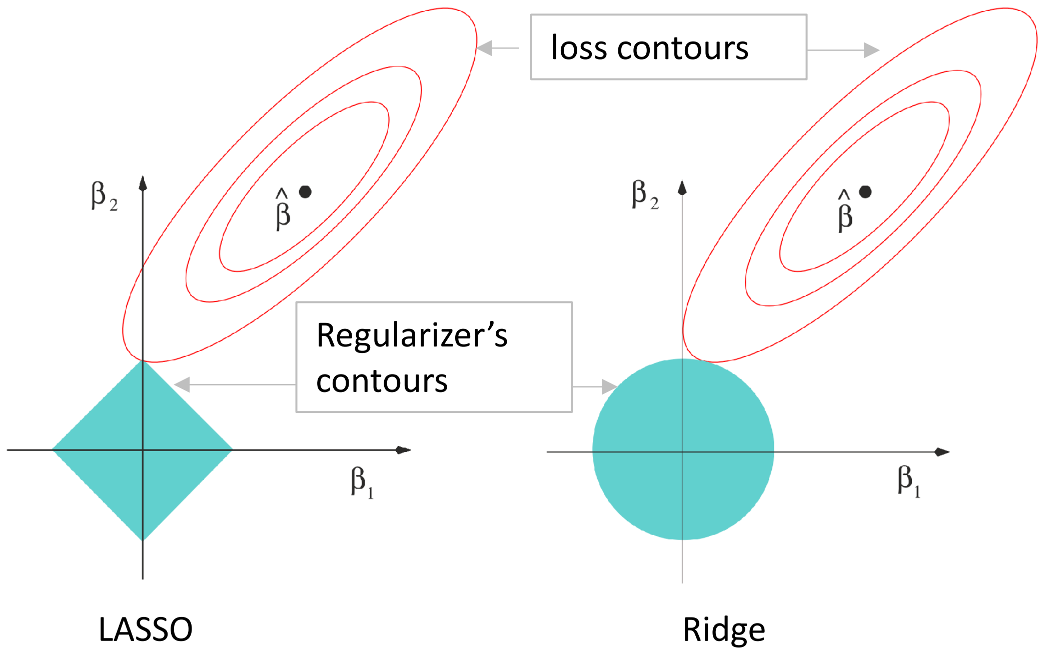
  - Minimizing cost function using one number is the simplest model: $\beta_0 =$ mean of y

LASSO vs. Ridge: which generates sparse solution?

- Ridge
  - Final model includes **all or none** of the input variables
  - Coefficients of least important input variables shrink close to zero; but never exactly zero
  - Major advantage of ridge regression is **coefficient shrinkage** and reducing model complexity
- Lasso (Least Absolute Shrinkage and **Selection** Operator)
  - Along with shrinking coefficients, lasso performs **selection of input variables** as well
  - Some coefficients would become exactly zero, which is equivalent to the particular input variable being excluded
- Popular interview question
  - Dense: all β_i are non-zeros. Which regularization? $\boxed{}$
  - Sparse: non-zero β_i are sparse. Which regularization? $\boxed{}$

Exercise: check the difference between LASSO and Ridge for OLS

Why does LASSO generate sparse solution?



- In 2D, minimum where loss contours is tangent to regularizer's contours
- For LASSO: minima occur at "corners"; hence one coefficient is zero
- For Ridge: minima occurs at any point of the blue circle; hence all coefficients are non-zero

Logistic regression with regularization

- Similarly, LASSO regularization

$$\min(loss\ function + \alpha|*|_1)$$

- Ridge regularization

$$\min(loss\ function + \alpha|*|_2^2)$$

- LogisticRegression class from scikit-learn library
  - LASSO and Ridge are both implemented
  - Note that parameter **C** is the **inverse** of regularization strength
- logit() from statsmodels library
  - **Only LASSO** is implemented
  - Parameter alpha is the penalty weight

Exercise: check the difference between LASSO and Ridge for Logistic Regression

How to choose α, i.e., the balance of Bias-Variance tradeoff

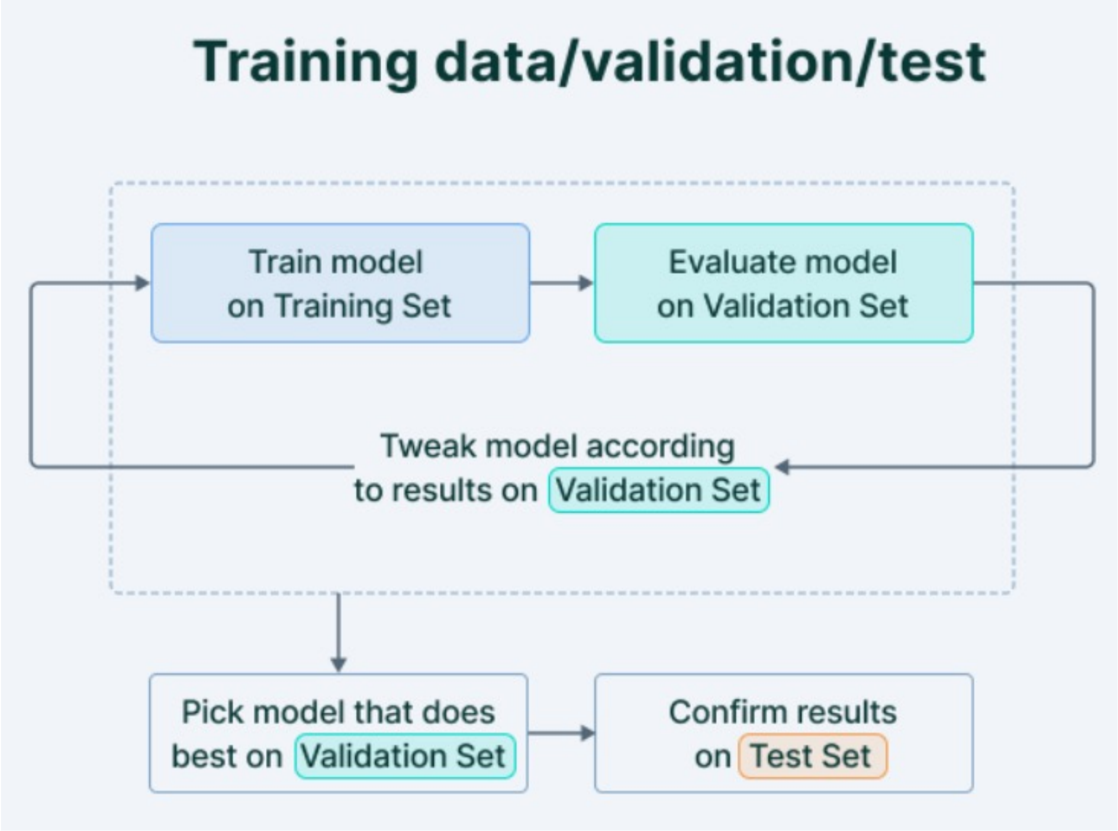
The whole dataset

Training	Validation	Testing

- Used to fine-tune the model hyper-parameters, **including α (penalty weight)**
- model occasionally sees this subset, but never gets trained by this subset
- Can be used more than once

- Well curated containing carefully sampled data that covers all types of rows in the data set
- Used **only once**

More on train vs. validation vs. test split

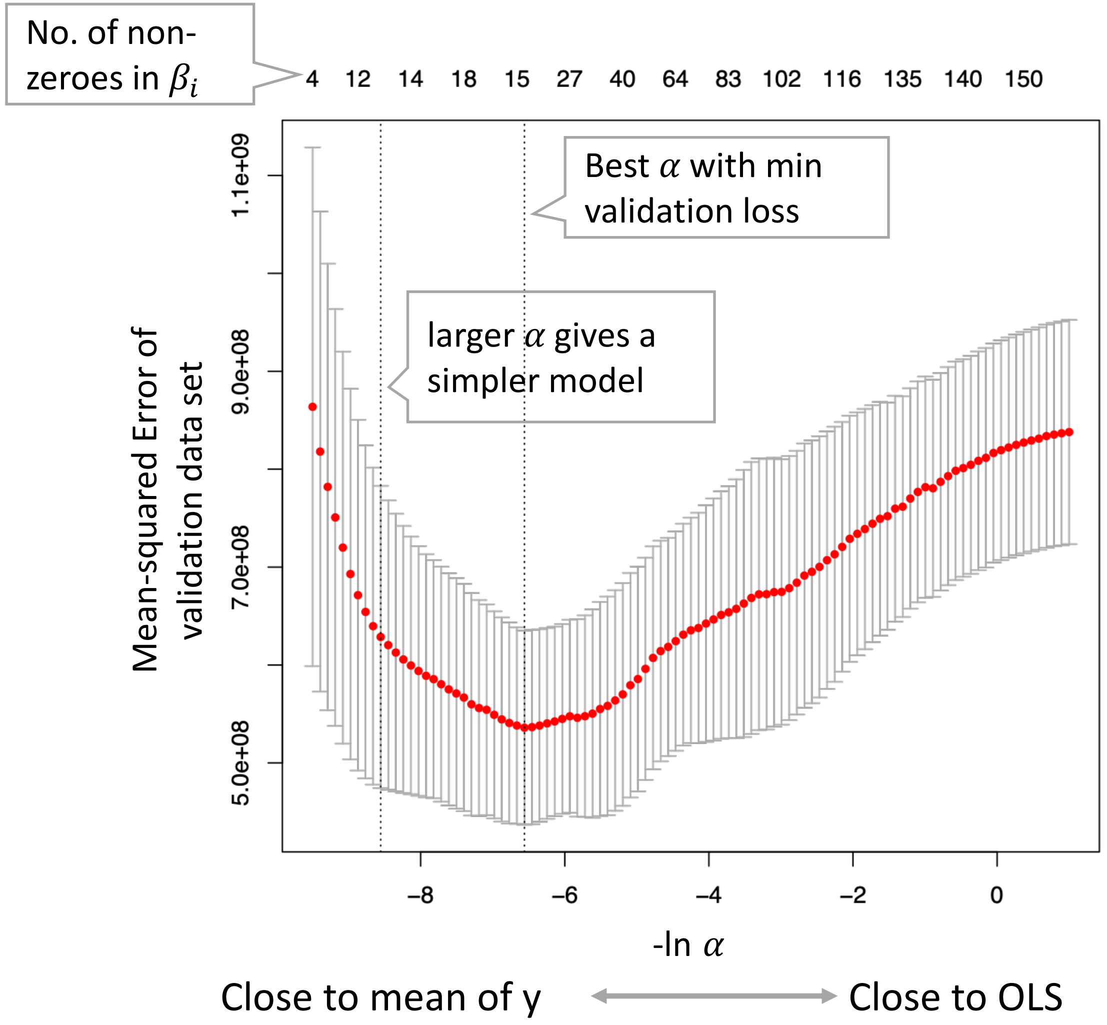


Many competitions on Kaggle release the validation set initially along with the training set. The actual testing set is only released before the competition closes.

The performance of the model on the testing data set determines the winner.

Exercise: split original data set Social_Network_Ads.csv into train/validation/test

Choosing the best α penalty weight for a model with 152 potential terms

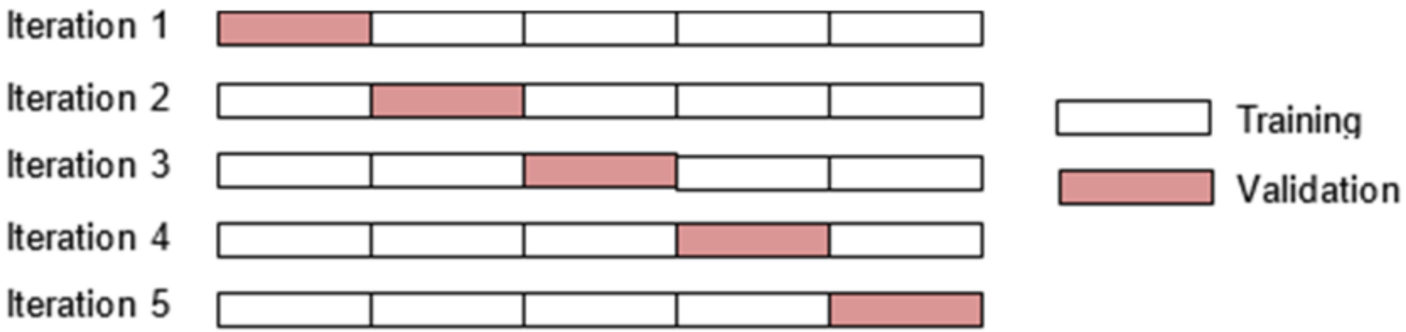


- Sometimes we might want to choose larger α to get a simpler model
- As α increases, the model complexity reduces and gets closer to the simplest model (mean of y)
- High α reduces overfitting, but can cause underfitting
- α should be chosen wisely based on validation loss and model complexity

What if the data set is small and we need to split as 50% train, 25% validation, 25% test? Note that 25% validation cannot be used to train model

k-fold Cross-Validation to use validation data set for training

- Divide a dataset D into k equal-sized subsets
- Suppose k=5, the subsets are labelled as D1, D2, …, D5
- Select 4 of the subsets as training set and the remaining one as validation set
- Rotate to the next subset as validation
- In k-fold cross validation, every subset is used to train the model as well as validate the model



Documentation: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.cross_val_score.html
Example: https://scikit-learn.org/stable/modules/cross_validation.html

Extension of standard k-fold Cross-Validation

- Stratified k-fold cross-validation
  - The training data set is divided in such a way so that the mean of y is approximately equal in all the k subsets
  - Reduce the selection bias caused by random division, such as all the same type of observations are placed into one single subset

- Leave-one-out
  - At each step, one observation is randomly taken out as validation
  - Good for super small datasets

Exercise: perform k-fold cross-validation on previous logistic regression with Ridge

Documentation: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.cross_val_score.html
Example: https://scikit-learn.org/stable/modules/cross_validation.html

One more twist on choosing the best α penalty weight

- Based on validation performance, we already choose LASSO with α = 250
- However, we find out that the loss for test data set when α = 250 is higher than loss when α =100
- Shall we choose α = 100?
- Why? Because then test data set becomes part of training+validation data set, and the model might be tuned to fit test data set and cannot generalize to future data set

The effect of units

- $y = \beta_0 + x_1\beta_1 + x_2\beta_2 + \epsilon$
- Unit of x_1 : 1 square foot → 0.093 square meter
- OLS solution with no regularization remains the same. Why?
  - First order condition (FOC)
  - $(\beta_0^\star, \beta_1^\star, \beta_2^\star)$ is the optimal solution to $\min_{(\beta_0,\beta_1,\beta_2)} f((y - (\beta_0 + x_1\beta_1 + x_2\beta_2))$
  - $(\beta_0^\star, 10\beta_1^\star, \beta_2^\star)$ is the optimal solution to $\min_{(\beta_0,\beta_3,\beta_4)} f\left((y - (\beta_0 + \frac{x_1}{10}\beta_3 + x_2\beta_4)\right)$
- Solution with regularization remains the same?
  - Scaling's effect on regularization
  - Changes in the magnitude of x_1 affects the magnitude of β_1
  - Regularization is to control β_i to a small space

Illustration - scaling's effect on regularization

If x_1 becomes $x_1^{up} = 1{,}000{,}000{,}000 \times x_1$

- new coefficient β_1^{up} is likely to be very small, since y is much smaller than x_1^{up}
- β_1^{up} will likely satisfy the requirement of small space imposed by regularization
- Small or no penalty on β_1^{up}

If x_1 becomes $x_1^{down} = \frac{1}{1{,}000{,}000{,}000} \times x_1$

- new coefficient β_1^{down} is likely to be very large since y is much larger than x_1^{down}
- β_1^{down} will likely **not** satisfy the small space requirement
- Large penalty on β_1^{down}

Change in unit of input variables in LASSO/Ridge changes the solution!
Scale up $x_i \rightarrow$ less penalty on β_i
Scale down $x_i \rightarrow$ more penalty on β_i

Dealing with units of input variables

Remove solution's dependency on units of input variables

- Standardization before LASSO/Ridge regularization
- $x \to \frac{x-\bar{x}}{s}$
- $\bar{x}$ is sample mean of input variable, s is sample standard deviation of input variable
- Does it have unit anymore?

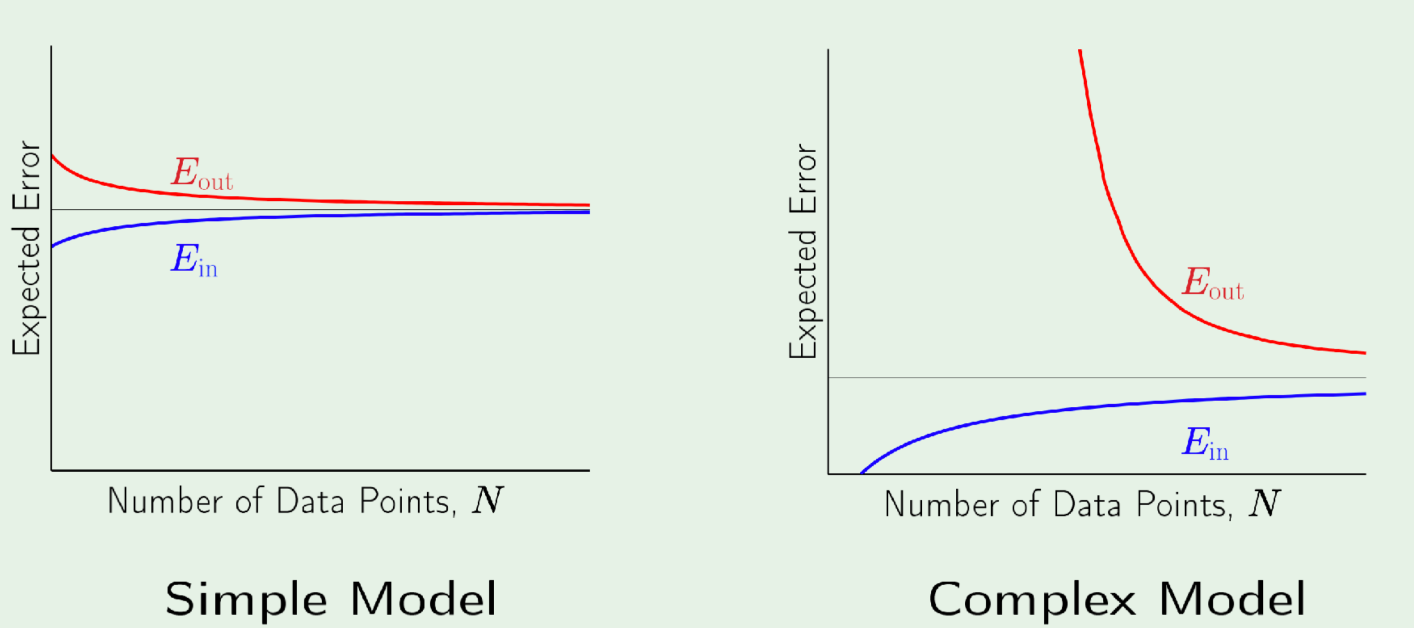
Pick a suitable unit for input variable based on business knowledge

- Input variables have different impact on y, e.g., area vs. floor on property price
- Unit of square feet for area might be essential for the model to work
- Unit of square meter might not work well

Exercise: perform standardization

Learning curve beyond Bias-Variancae Tradeoff

Previous Bias-Variance tradeoff is for fixed N in a given data set; What if $N \rightarrow \infty$?



N: number of samples in the training set

E_{in}: in-sample (i.e. train) error (loss)

E_{out}: out-of-sample (i.e. test) error (loss)

Observations

- Which model has lower E_{in}? Complex model
- Why does E_{in} increase as N increases?
  - Larger training set makes it harder to find a perfect fit
- Which one has lower E_{out}?
  - Small N => simple model
  - Big N => complex model

Source: https://home.work.caltech.edu/slides/slides08.pdf (Caltech's Machine Learning Course)

Recap: Precision vs. Recall

- A large telecom company wants to reduce customer churn by providing discount offers to customers who are predicted to leave soon
- Positive: stay; Negative: leave
- Should accuracy rate be the only metric?
- What are the two kind of errors?

Recall (True positive rate TPR) = 1 – False Negative Rate (FNR)

Precision =
True Positive / Positive Predictions

False Positive Rate =
False Positive / Negative cases

Type	Nature of error	Reality	Pred	What do we lose?
1	False +ve	Leave	Stay	
2	False -ve	Stay	Leave	

- How bad is the false negative?
- Optimize for Precision or Recall?

Alternative solution: Weighting in the loss function

- Standard loss function: all observations are treated equal

$$\sum_i (y_i - \widehat{y_i})^2$$

- **Weighted** loss: residuals of important observations should have high w_i

$$\sum_i w_i (y_i - \widehat{y_i})^2$$

- Observations with type 1 error (leave in reality; predicted to stay) should be given higher weight

$$w_{type\ 1} > w_{type\ 2}$$

Another example on weighting

- A consumer goods company uses demand prediction to prepare stock of all goods
- Their data science team implemented new model and reduce 1/3 of test loss
- Will the profit go up? Not necessarily
  - Different goods have different margins
  - If the improved model performance comes at a cost of high-margin demand prediction, new model might lead to a lower profit

	High Margin Product A	**Low Margin Product B**	**Standard test Loss**	**Drop from max profit**
Profit per product	100	1		
Real demand in test set	1	50		
Old Pred for test set	1	30	$(1-1)^2 + (50-30)^2=400$	(50-30)*1=20
New Pred for test set	0	40	$(1-0)^2 + (50-40)^2=101$	1*100+ (50-40)*1=110

Add weighting in the loss function

- Standard loss function: all observations are treated equal

$$\sum_i (y_i - \widehat{y_i})^2$$

- Weighted loss: residuals of high margin products should have high w_i

$$\sum_i w_i (y_i - \widehat{y_i})^2$$

- Products higher margin should be given higher weight

$$w_{high-margin} > w_{low-margin}$$

- Extension: If problem has a time component

$$w_{yesterday} > w_{last-month}$$

A success story

During the COVID-19 pandemic, e-commerce sellers might not get sufficient delivery support. An online seller weighed different products in their demand forecast system.

They prioritized the model for high-margin products that were not frequently bought (e.g., gems). As a result, their profit did not take much hit.

Is ML helping the business world?

According to a survey of 2,500 executives in 2019, among the 90% respondents who have invested in AI, less than 40% had seen business gains from AI in the past three years.

Sloan management review and Boston Consulting Group

Solution

- Ensure business insights are aligned the loss function to be minimized
- Check out the article on Harvard Business Review - “Why You Aren’t Getting More from Your Marketing AI”

Source: https://sloanreview.mit.edu/projects/winning-with-ai/
https://hbr.org/2021/07/why-you-arent-getting-more-from-your-marketing-ai

Recap and next lecture

Recap

- Simple vs. complex model
- Bias-variance tradeoff
- Reduce number of terms
- Regularization (LASSO vs. Ridge)
- Choosing penalty weight α
- k-fold Cross-Validation
- Effects on units on regularization
- Learning curve
- Weighting in the loss function
- Align business insights to loss function

Next lecture

- Decision Tree

Lab session

Homework

- Watch video tutorial for week 1 lecture (if you have not done so)
  - https://youtu.be/W0IFrZDRP3M
  - https://youtu.be/KNdPqBaUDLc
  - https://youtu.be/ngPjj93B5kE
- Watch video tutorial for week 2 lecture
- Post learning reflections and questions if any
- Complete group assignment and submit as a group before week 2 lecture