Predictive Model Validation Theory

Dr Liew How Hui

Jan 2021

Motivation

Two major hazard in predictive modelling —"bias" and "lack of generalisation".

Theoretical limit of "bias": Bias-Variance Decomposition of "Expected Test Error".

Ways to estimate "expected test error" & "Performance" — Resampling methods:

- cross-validation
- bootstrap

The right generalisation — training error & testing error balance.

Outline

Theory on the "Bias"-ness of Models

Model: $h_D(\mathrm{x})$; x: is usually fixed but can be random.

Random: D; $y \sim$ some distribution

Bias-Variance Decomposition

$$\underbrace{\mathbb{E}_{\mathrm{x},y,D}\left[\left(h_D(\mathrm{x})-y\right)^2\right]}_{\text{Expected Test Error}} = \underbrace{\mathbb{E}_{\mathrm{x},D}\left[\left(h_D(\mathrm{x})-\overline{h}(\mathrm{x})\right)^2\right]}_{\text{Variance}} + \underbrace{\mathbb{E}_{\mathrm{x}}\left[\left(\overline{h}(\mathrm{x})-\overline{y}(\mathrm{x})\right)^2\right]}_{\text{Bias}^2} + \underbrace{\mathbb{E}_{\mathrm{x},y}\left[\left(\overline{y}(\mathrm{x})-y\right)^2\right]}_{\text{Noise}}. \quad (1)$$

Theory (cont)

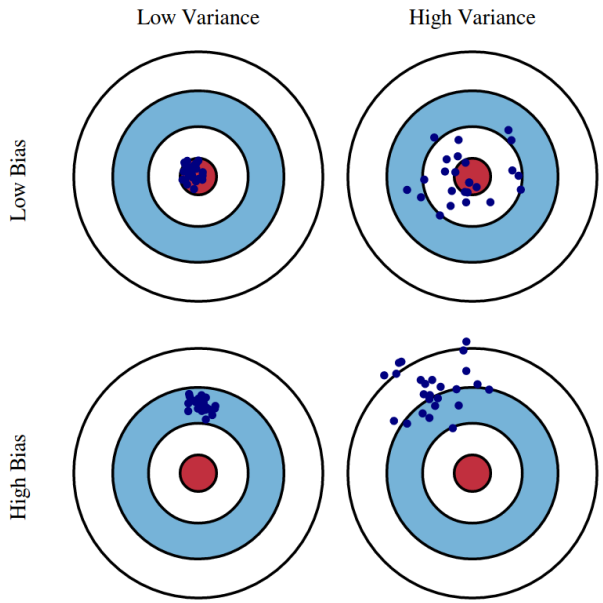
The "Variance" captures how much the classifier changes if we train on a different training set. How "over-specialised" is the classifier to a particular training set (i.e. is it overfitting)?

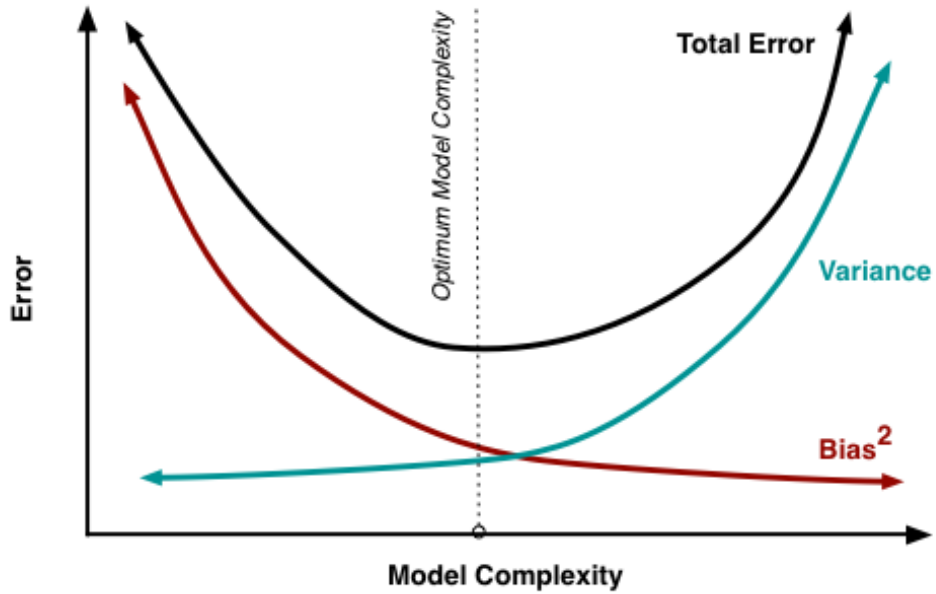
What is the inherent error that we obtain from the classifier even with infinite training data? This is due to the classifier being "biased" to a particular kind of solution (e.g. linear classifier).

How big is the data-intrinsic "noise"? This error measures ambiguity due to our data distribution and feature representation. We can never beat this, it is an aspect of the data.

Theory (cont)

(a) Graphical illustration of bias and variance; (b) The variation of Bias and Variance with the model complexity. This is similar to the concept of overfitting and underfitting. More complex models overfit while the simplest models underfit.

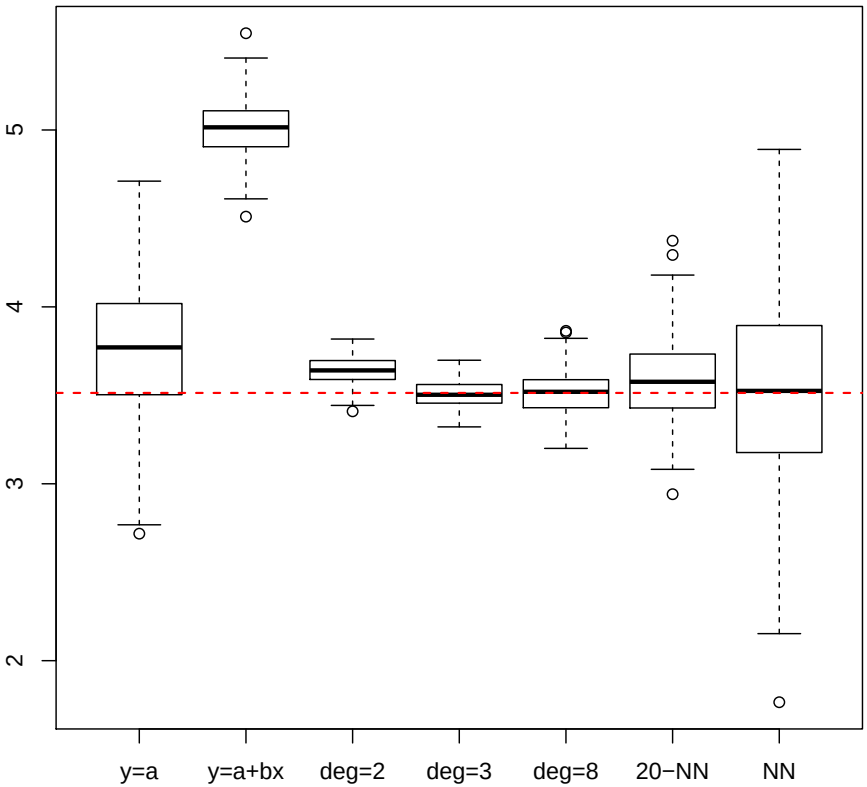


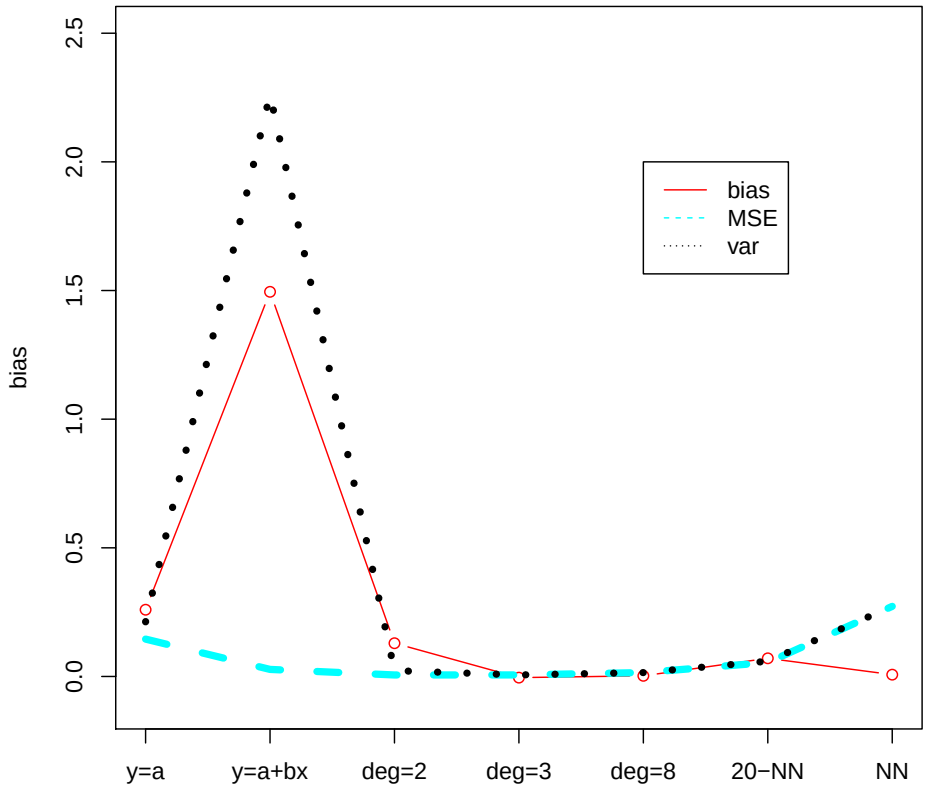


Example: Tutorial 04

Theory (cont)

Using Tutorial 04 technique with $f(x) = x^2 + 1.2x^3$, $0 \leq x \leq 2$, $\sigma = 0.5$ and the prediction at $x = 1.2$ for various polynomial regression and kNN regression models, we obtain:





Outline

Cross-Validation

In this topic, we assumed that we are given a dataset

$$D = \{(\mathrm{x}_1, y_1), \ldots, (\mathrm{x}_n, y_n)\} \sim P(X, Y)$$

drawn i.i.d. from some (unknown) distribution $P(X, Y)$ and the responses $y_i \in \mathbb{R}$.

Suppose the data D is used to train a supervised learning model $\widehat{h}_D$ and suppose that we have a new test data

$$(\mathrm{x}_1^{new}, y_1^{new}), \cdots, (\mathrm{x}_L^{new}, y_L^{new}) \sim P. \tag{2}$$

Cross-Validation (cont)

Loss function = "error measurement"

$$\frac{1}{L}\sum_{i=1}^{L}\ell(y_i^{new}, \widehat{h}_D(\mathrm{x}_i^{new})) \to \mathbb{E}[\ell(Y^{new}, \widehat{h}_D(X^{new}))] \quad (3)$$

where the right is the *theoretical test set error* for a predictive model training on a particular training set $\widehat{h}_D$. The *theoretical generalisation error* is defined as the average of theoretical test set error over all possible training sets, i.e.

$$\mathbb{E}_D\left[\mathbb{E}_{(X^{new}, Y^{new})}[\ell(Y^{new}, \widehat{h}_D(X^{new}))]\right]. \quad (4)$$

Cross-Validation (cont)

For classification problems, the usual form of test error (3) and ℓ is

$$\frac{1}{L}\sum_{i=1}^{L} I(y_i^{new} \neq \widehat{h}_D(\mathrm{x}_i^{new})).$$

For regression problems, the usual form of test error (3) and ℓ is

$$\frac{1}{L}\sum_{i=1}^{L} (y_i^{new} - \widehat{h}_D(\mathrm{x}_i^{new}))^2.$$

Cross-Validation (cont)

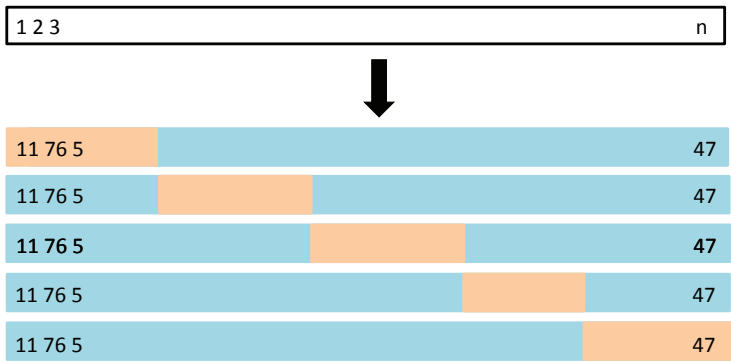
A true test data (2) is not available at the time we want to run and tune our supervised learning algorithm. Therefore, we need artificial ways to construct training and testing datasets:

1. Split-test validation (Topic 1);
2. Leave-one-out cross-validation (LOOCV);
3. K-fold cross-validation.

Cross-Validation (cont)

k-fold cross validation (CV)

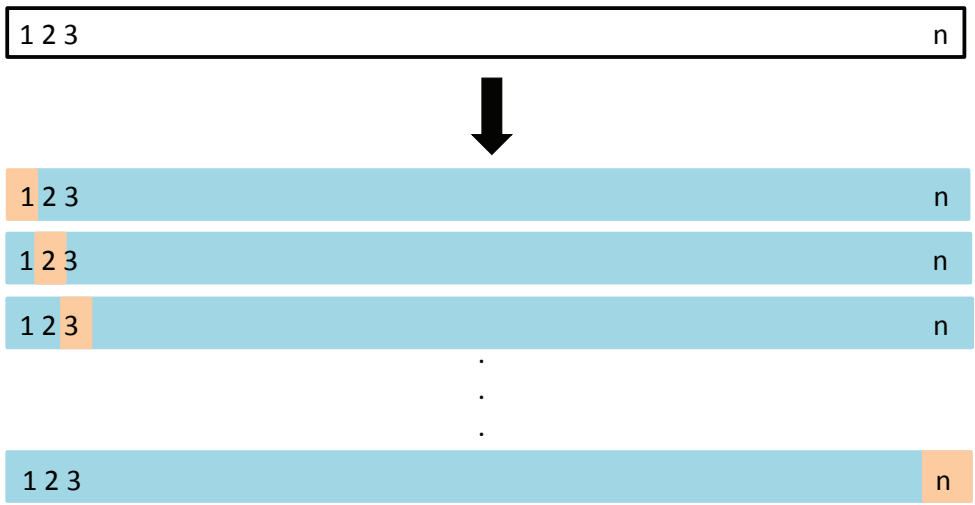
- Randomly divides the set of observations into k "equal" "folds";
- First fold = validation set & remaining $k-1$ folds = training set.
- Second fold = validation set & remaining $k-1$ folds = training set.
- etc.



5-fold example:

Cross-Validation (cont)

Special case $k = n \Rightarrow$ **leave-one-out cross validation (LOOCV)**



Since training sample is close to n, the bias is small. The variance is usually high because n training sets are similar to each other due to strong correlation.

Cross-Validation (cont)

Software support:

R: `caret::createDataPartition`, `createResamples` (for bootstrapping), `createFolds` (for k-fold cross validation), `createMultiFolds` (for repeated cross-validation), `createTimeSlices` (for dealing with time-series).

Python: `train_test_split` (simple random split), `KFold` or `StratifiedKFold` (k-fold CV), `LeaveOneOut` (LOOCV) from `sklearn.cross_validation`.

Cross-Validation (cont)

https://stats.stackexchange.com/questions/61090/how-to-split-a-data-set-to-do-10-fold-cross-val

```
#Randomly shuffle the data & create 10 fold
d.f = d.f[sample(nrow(d.f)),]
folds = cut(seq(1,nrow(d.f)),breaks=10,labels=FALSE)
#Perform 10 fold cross validation
for(i in 1:10){
    #Segement your data by fold using which()
    testIndexes = which(folds==i,arr.ind=TRUE)
    testData  = d.f[testIndexes, ]
    trainData = d.f[-testIndexes, ]
    ...
}
```

Cross-Validation (cont)

Using caret library (not my cup of tea)

```
set.seed(123)
train.control <- trainControl(method="cv", number=5)
model <- train(y ~., data=train.data, method="lm",
               trControl=train.control)
print(model)
```

Cross-Validation (cont)

Final Exam May 2019, Q3

(a) Supervised learning includes classification and regression.
   (i) State the difference between classification and regression in term of response variable. (1 mark)
   (ii) Explain the sampling methods used in splitting data for classification and regression respectively. (4 marks)

(b) (i) State an issue that comes along with split validation, which can be overcome by using cross validation. (1 mark)
   (ii) Describe the process of a 5-fold cross validation. (4 marks)

Cross-Validation (cont)

Final Exam May 2019, Q3 (cont)

(c) A sample of 500 males and 800 females had been collected to test on a model of gender prediction. The model resulted that 380 males and 510 females were predicted correctly.

(i) Assume male as positive class and female as negative class, calculate the count of true positive (TP), true negative (TN), false positive (FP) and false negative (FN) for the model's result. (2 marks)

(ii) Construct the confusion matrix for the model. State the classification error, specificity and sensitivity of the model. (4 marks)

(iii) Compare the recall and precision for both male and female. Interpret your results. (4 marks)

Bootstrap

$$\mathbb{E}_D\left[\mathbb{E}_{(X^{new},Y^{new})}[\ell(Y^{new},\widehat{h}^{(-B_k)}_{n-|B_k|}(X^{new}))]\right]$$

- K-fold CV: $\approx \frac{1}{K}\sum_{i=1}^{K}\frac{1}{|B_k|}\sum_{i\in B_k}\ell(y_i^{new},\widehat{h}^{(-B_k)}_{n-|B_k|}(\mathrm{x}_i^{new}))$
- LOOCV: $\approx \frac{1}{n}\sum_{i=1}^{n}\ell(y_i^{new},\widehat{h}^{(-i)}_{n-1}(\mathrm{x}_i^{new}))$
- Bootstrap: Choose a number of bootstrap samples, B, a sample size n, allow repeated sampling
  - Jacknife bootstrap: Similar to LOOCV?
  - Block/Spatial bootstrap: Similar to K-fold?

Outline

Training vs Testing Errors

Suppose the data is split into "training" D_{train} and "testing" D_{test}.

For D_{train}, the regression error is

$$RSME_{train} = \sqrt{\frac{1}{|D_{train}|} \sum_{(x,y) \in D_{train}} (y - h_{D_{train}}(x))^2}$$

For D_{test}, the regression error is

$$RSME_{test} = \sqrt{\frac{1}{|D_{test}|} \sum_{(\tilde{x},\tilde{y}) \in D_{test}} (\tilde{y} - h_{D_{train}}(\tilde{x}))^2}$$

Training vs Testing Errors (cont)

Theoretically, the more data the better the "trained" model, we expect.

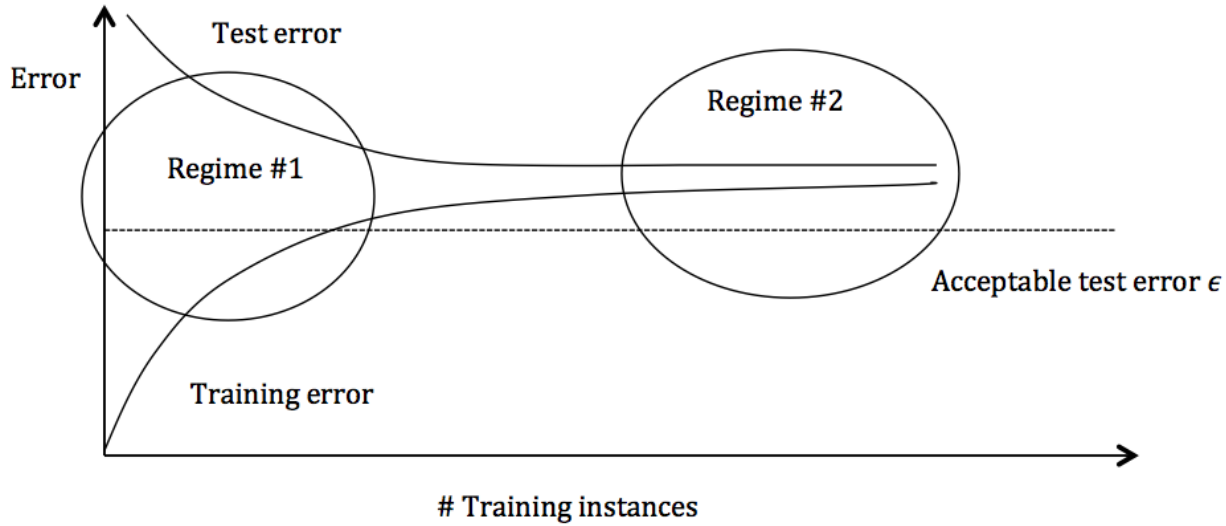


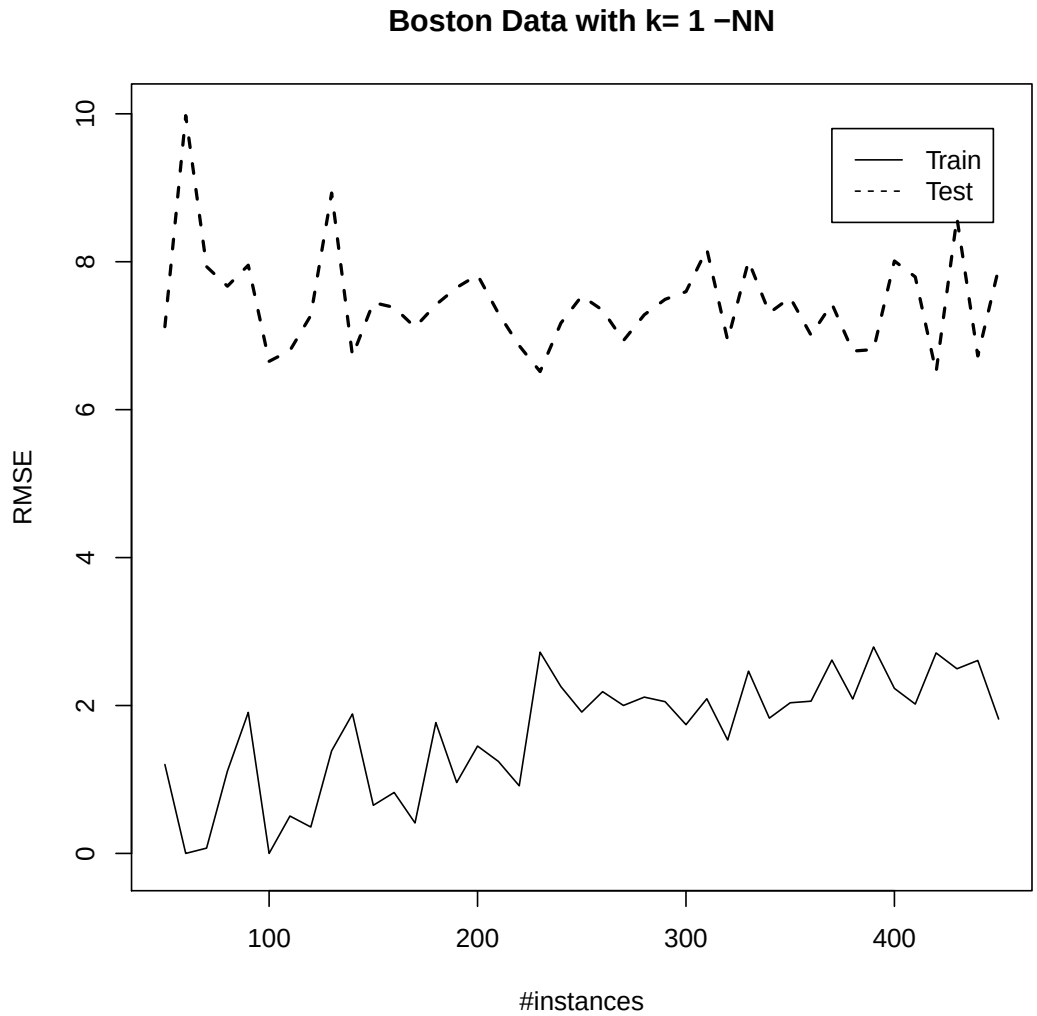
Figure: Test and training error as the number of training instances ↑. Region 1: High variance; Region 2: High bias

Remedy 1: Add more data / reduce the model complexity

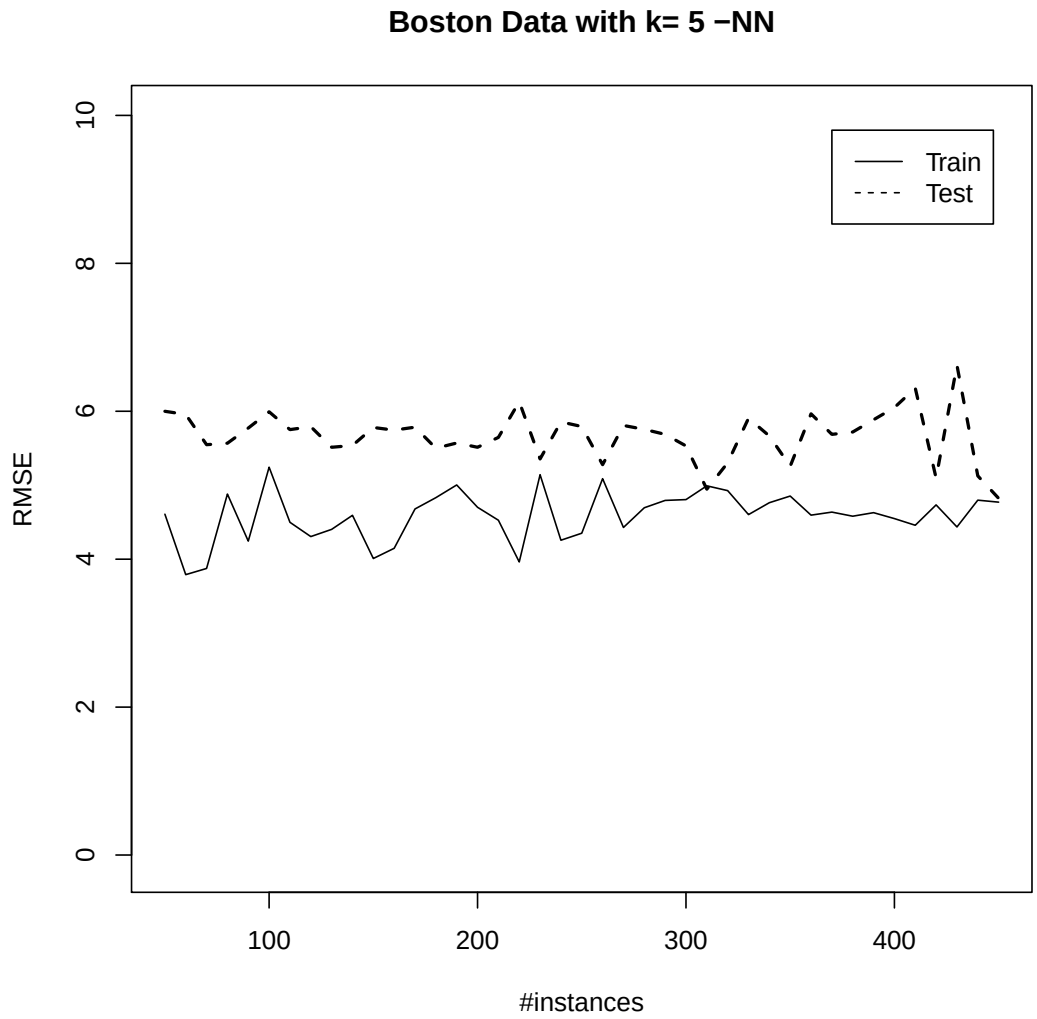
Remedy 2: Use a more complex model

Training vs Testing Errors (cont)

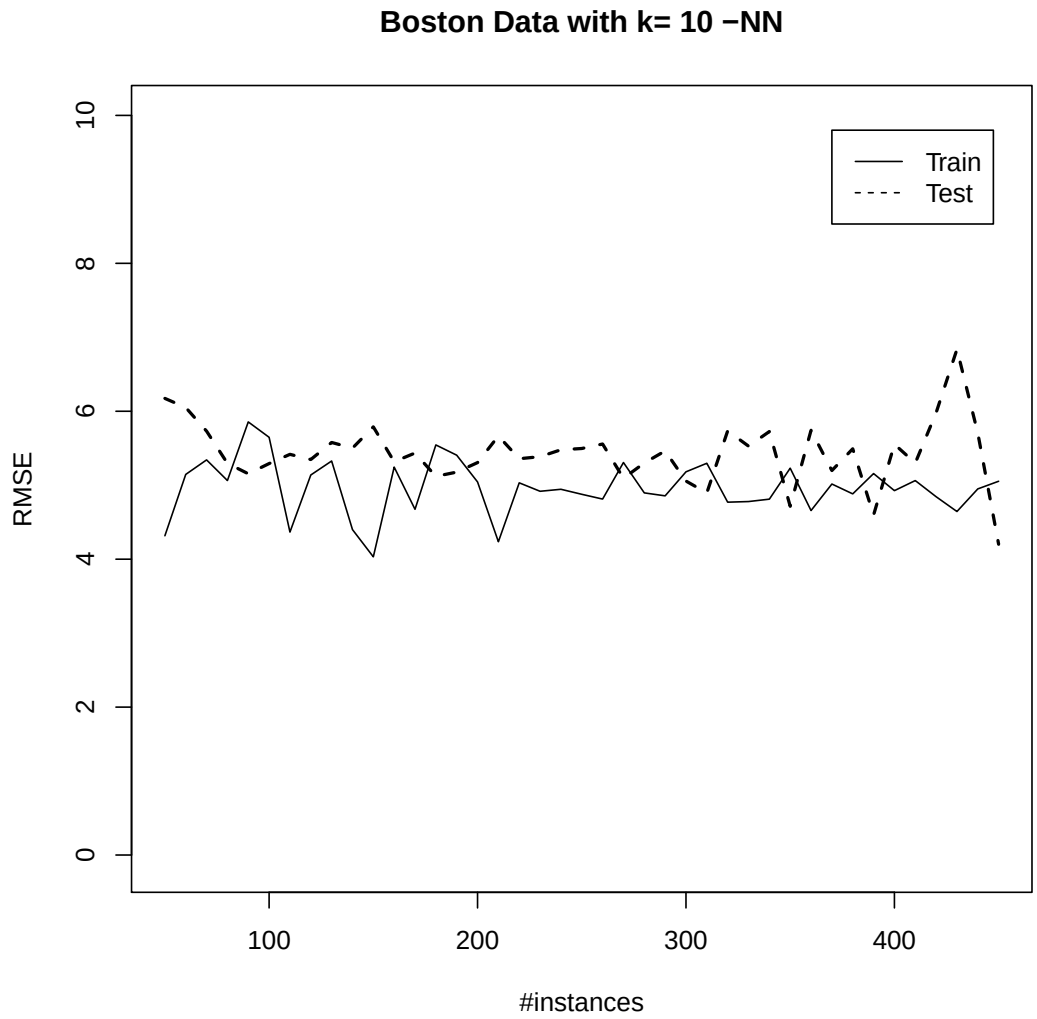
In reality, things are "sometimes different":



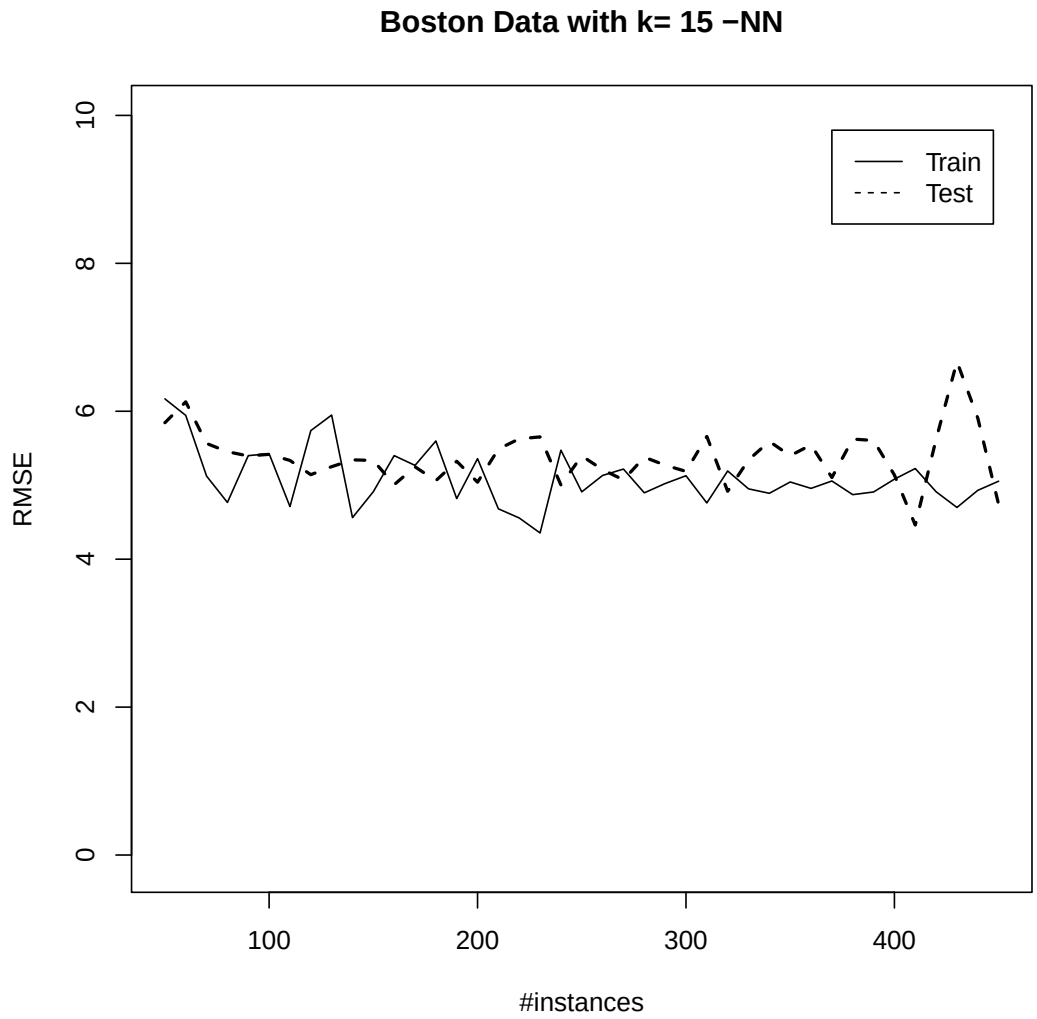
Training vs Testing Errors (cont)



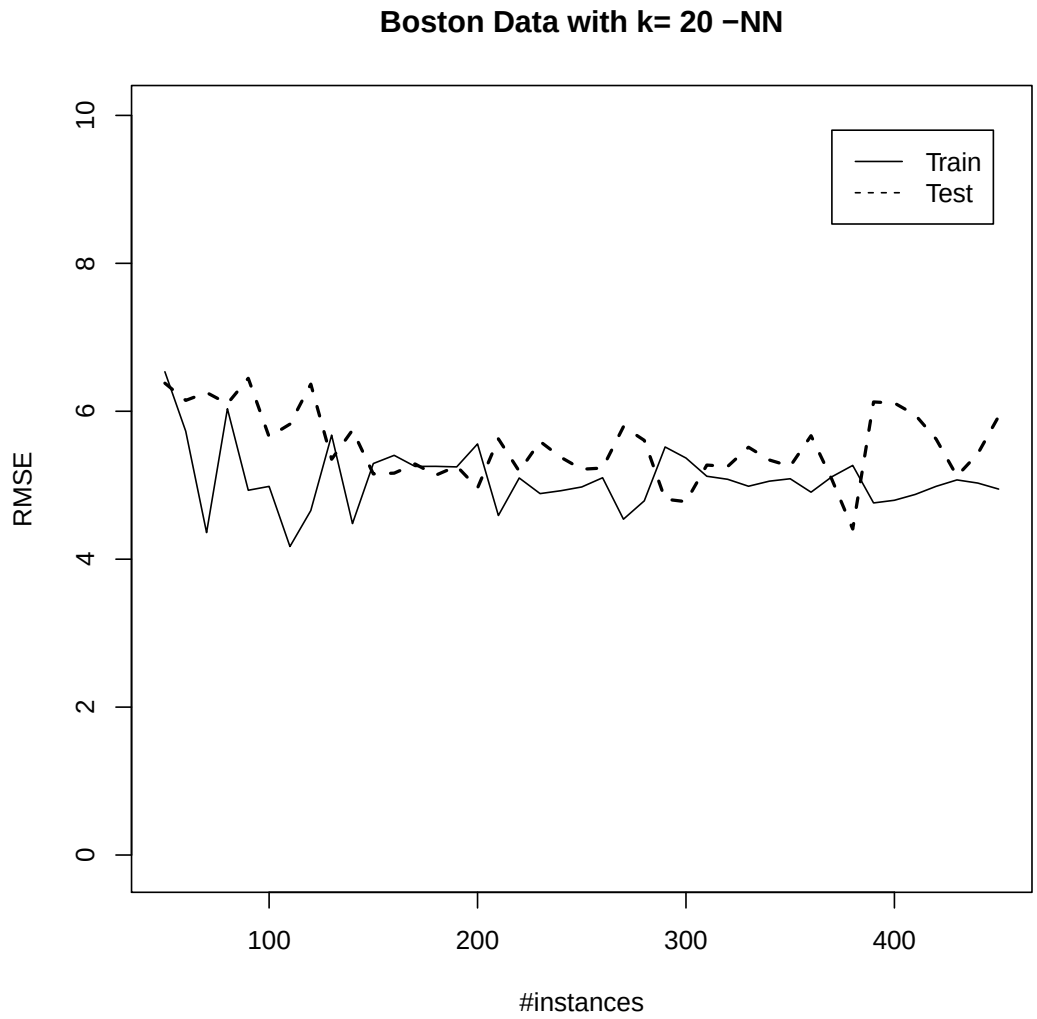
Training vs Testing Errors (cont)



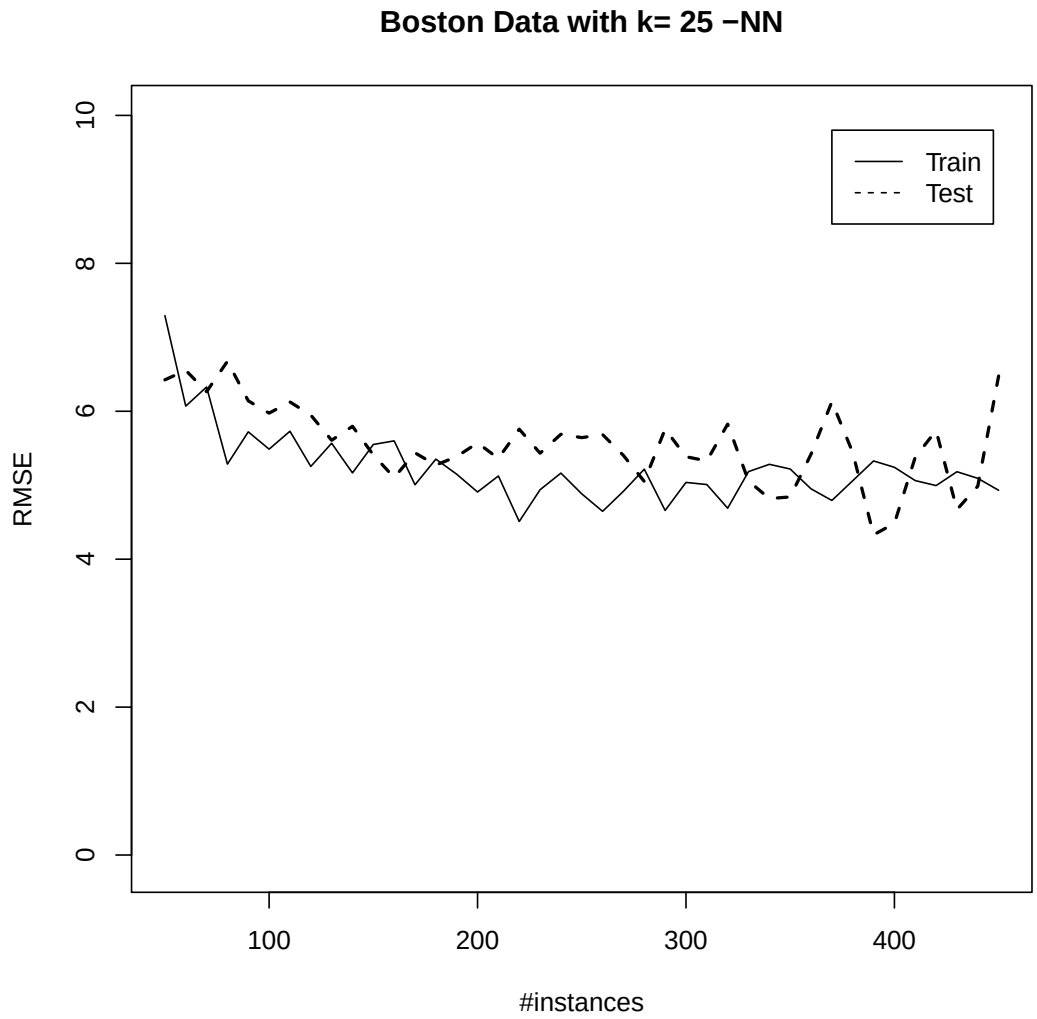
Training vs Testing Errors (cont)



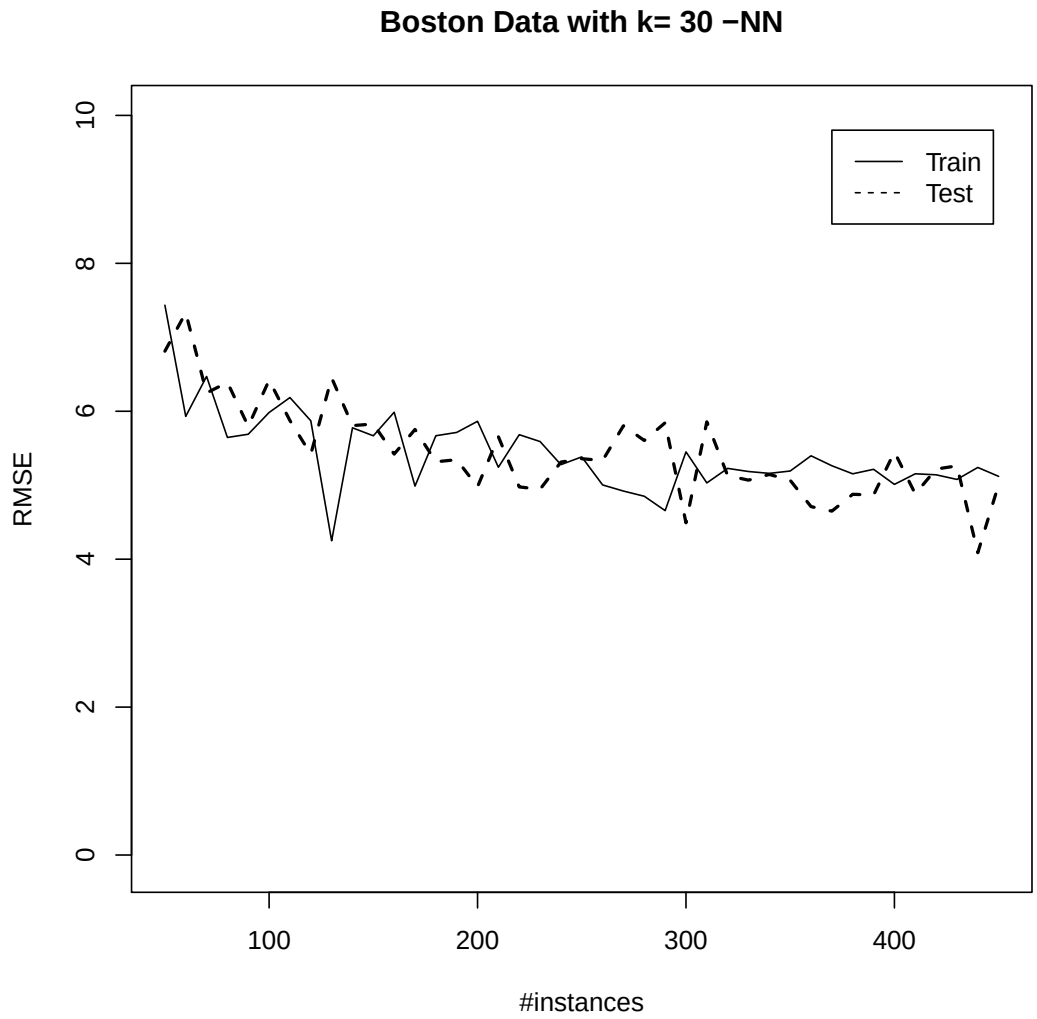
Training vs Testing Errors (cont)



Training vs Testing Errors (cont)

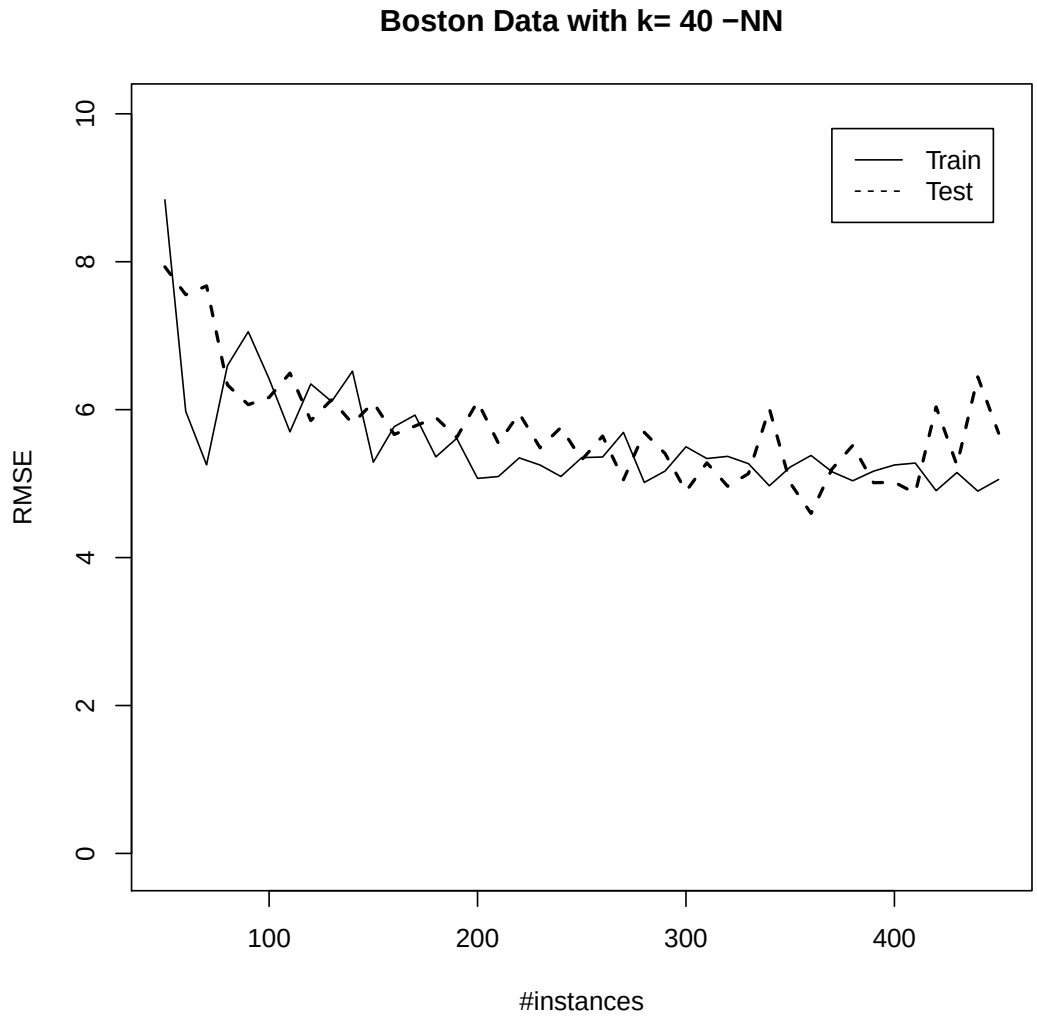


Training vs Testing Errors (cont)



Training vs Testing Errors (cont)

These are based on an adaption from Practical 2.



Training vs Testing Errors (cont)

The main use of "Training vs Testing Errors" is to capture the "best" hyperparameters as illustrated in the last part of Practical 2.

$k = 1$: Training error low, Testing error high $\rightarrow$ Overfitting

$k = 5$: Training error higher, Testing error lower $\rightarrow$ improving

$k = 10$: Training error higher, Testing error lower $\rightarrow$ OK?

$k = 15$: Training error higher, Testing error lower $\rightarrow$ OK?

$k = 20$: Training error higher, Testing error lower $\rightarrow$ OK?

$k = 25$: Training error higher, Testing error lower $\rightarrow$ OK?

$k = 30$: Training error higher, Testing error lower $\rightarrow$ OK?

$k = 40$: Training error higher, Testing error higher $\rightarrow$ Underfitting?