Lecture 002

Model accuracy and selection

Edward Rubin

Admin

Admin

Today

In-class

- Model accuracy
- Loss for regression and classification
- The bias-variance tradeoff
- The Bayes classifier
- KNN

Admin

Upcoming

Readings

- *Today*
  - Finish *ISL* Ch2
  - Prediction Policy Problems by Kleinberg *et al.* (2015)
- *Next*
  - *ISL* Ch. 3–4

Model accuracy

Model accuracy

Review: Supervised learning

1. Using **training data** $(\mathbf{y}, \mathbf{X})$, we train $\hat{f}$, estimating $\mathbf{y} = f(\mathbf{X}) + \varepsilon$.

Model accuracy

Review: Supervised learning

1. Using **training data** ($\mathbf{y}$, $\mathbf{X}$), we train $\hat{f}$, estimating $\mathbf{y} = f(\mathbf{X}) + \varepsilon$.

2. Using this estimated model $\hat{f}$, we *can* calculate **training MSE**

$$\text{MSE}_{\text{train}} = \frac{1}{n} \sum_{1}^{n} \underbrace{\left[\mathbf{y}_i - \hat{f}\left(x_i\right)\right]^2}_{\text{Squared error}} = \frac{1}{n} \sum_{1}^{n} \left[\mathbf{y}_i - \hat{\mathbf{y}}\right]^2$$

Note: Assuming $\mathbf{y}$ is numeric (regression problem).

Model accuracy

Review: Supervised learning

1. Using **training data** ($\mathbf{y}$, $\mathbf{X}$), we train $\hat{f}$, estimating $\mathbf{y} = f(\mathbf{X}) + \varepsilon$.

2. Using this estimated model $\hat{f}$, we *can* calculate **training MSE**

$$\text{MSE}_{\text{train}} = \frac{1}{n} \sum_{1}^{n} \underbrace{\left[\mathbf{y}_i - \hat{f}\left(x_i\right)\right]^2}_{\text{Squared error}} = \frac{1}{n} \sum_{1}^{n} \left[\mathbf{y}_i - \hat{\mathbf{y}}\right]^2$$

Note: Assuming $\mathbf{y}$ is numeric (regression problem).

3. We want the model to accurately predict previously unseen (**test**) data. This goal is sometimes call generalization or external validity.

Average $\left[y_0 - \hat{f}\left(x_0\right)\right]^2$ for obs. (y_0, x_0) in our **test data**.

Model accuracy

Errors

The item at the center of our focus is the (test-sample) **prediction error**

$$\mathbf{y}_i - \hat{f}\left(x_i\right) = \mathbf{y}_i - \hat{\mathbf{y}}_i$$

the difference between the label ($\mathbf{y}$) and its prediction ($\hat{\mathbf{y}}$).

The distance (*i.e.*, non-negative value) between a true value and its prediction is often called **loss**.

Model accuracy

Loss functions

Loss functions aggregate and quantify loss.

L1 loss function: $\sum_i \left| y_i - \hat{y}_i \right|$

Mean abs. error: $\dfrac{1}{n} \sum_i \left| y_i - \hat{y}_i \right|$

L2 loss function: $\sum_i \left(y_i - \hat{y}_i \right)^2$

Mean squared error: $\dfrac{1}{n} \sum_i \left(y_i - \hat{y}_i \right)^2$

Model accuracy

Loss functions

Loss functions aggregate and quantify loss.

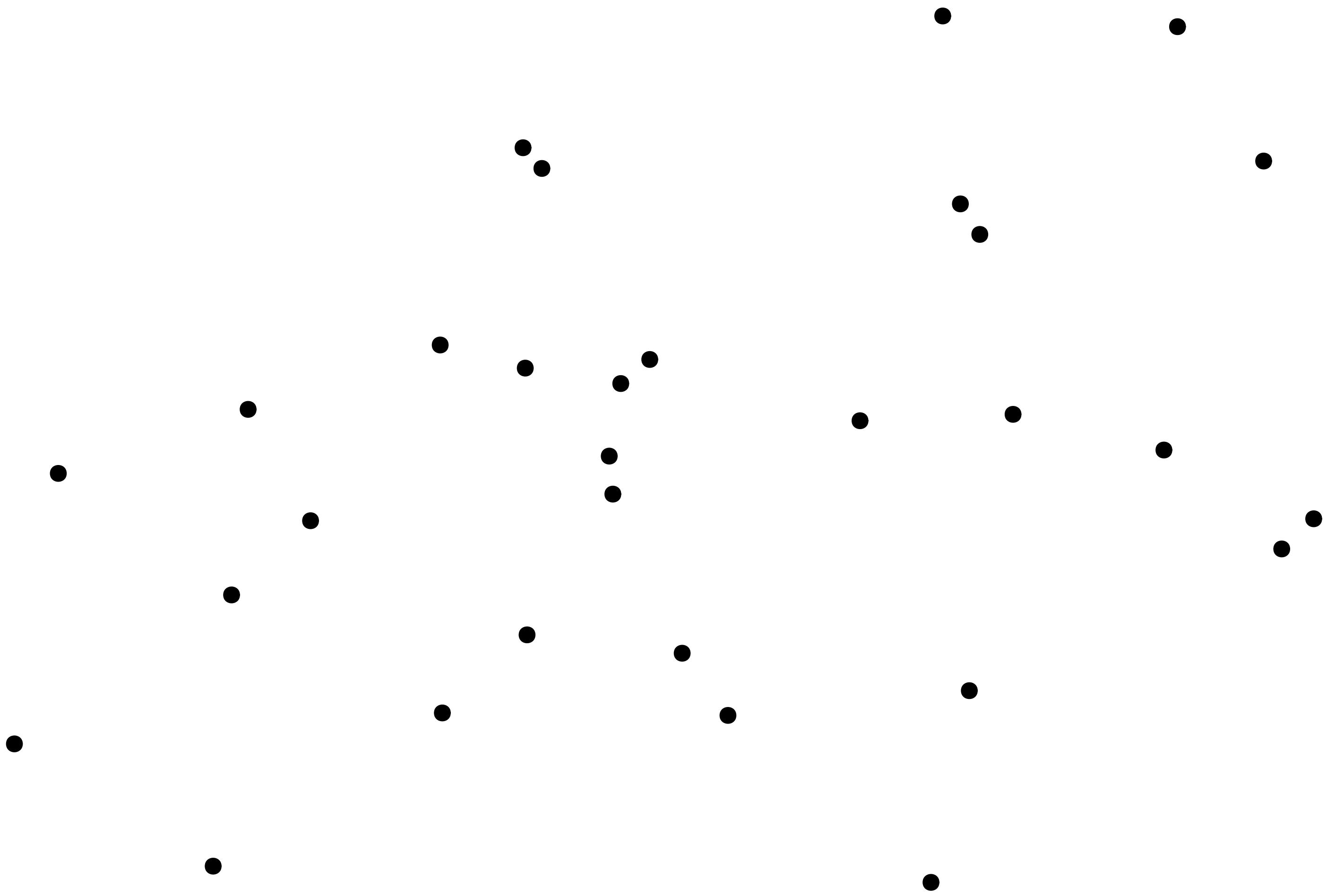
L1 loss function: $\sum_i \left| y_i - \hat{y}_i \right|$ **Mean abs. error**: $\dfrac{1}{n} \sum_i \left| y_i - \hat{y}_i \right|$

L2 loss function: $\sum_i \left(y_i - \hat{y}_i \right)^2$ **Mean squared error**: $\dfrac{1}{n} \sum_i \left(y_i - \hat{y}_i \right)^2$

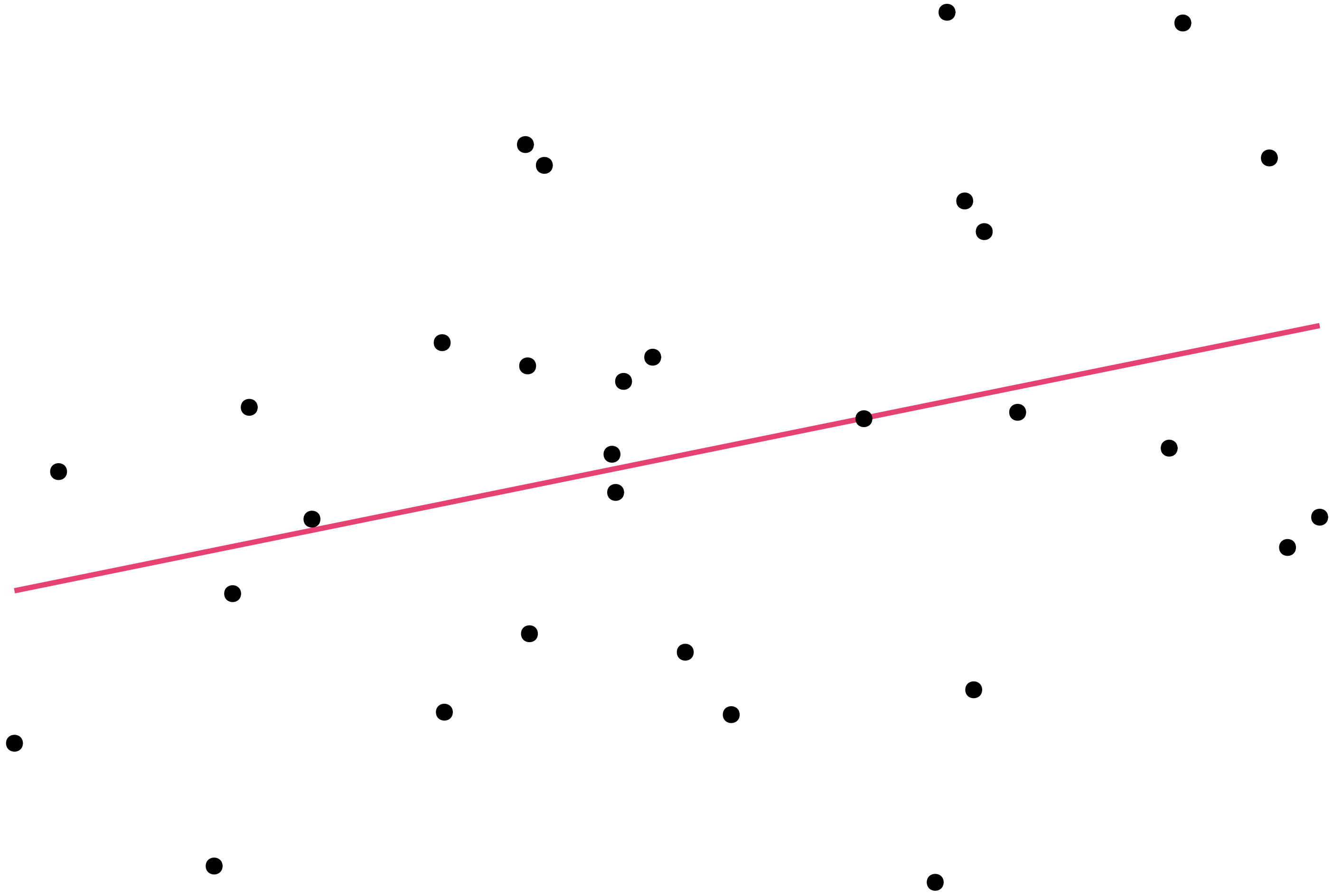
Notice that **both functions impose assumptions**.

1. Both assume **overestimating** is equally bad as **underestimating**.
2. Both assume errors are similarly bad for **all individuals** (i).
3. They differ in their assumptions about the **magnitude of errors**.
   - **L1** an additional unit of error is **equally bad** everywhere.
   - **L2** an additional unit of error is **worse** when the error is already big.

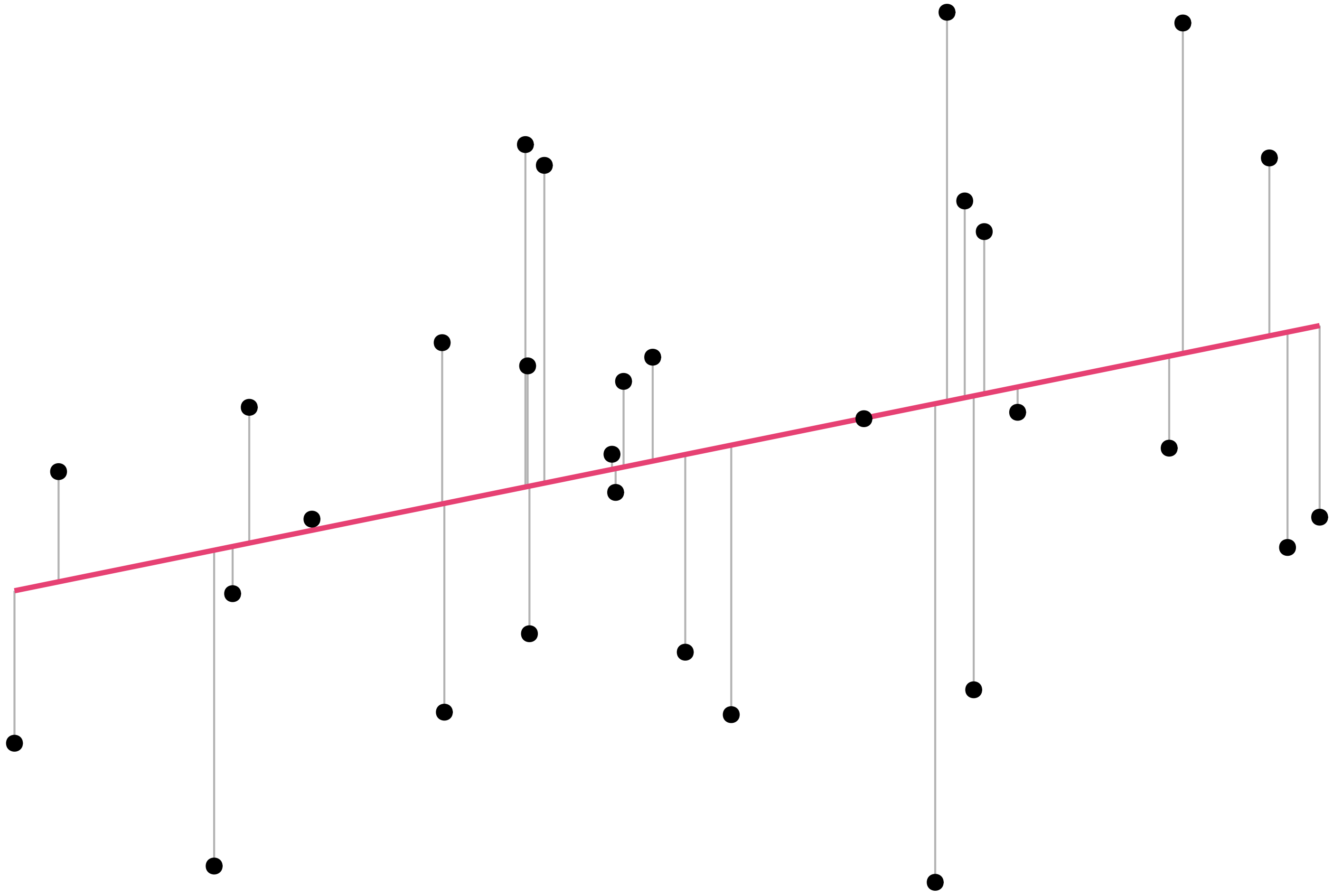
A very simple, univariate dataset $(\mathbf{y}, \mathbf{x})$



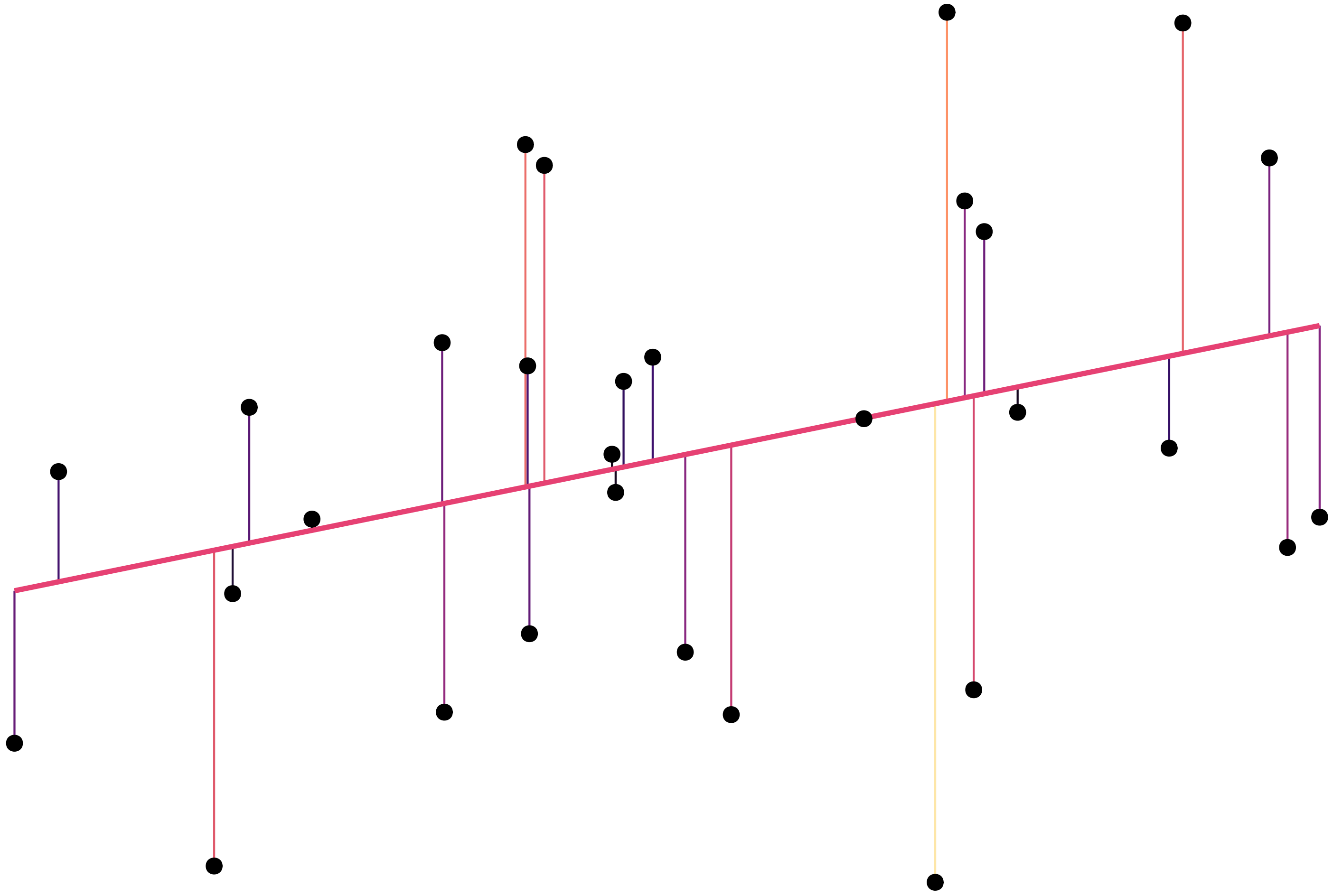
… on which we run a simple linear regression.



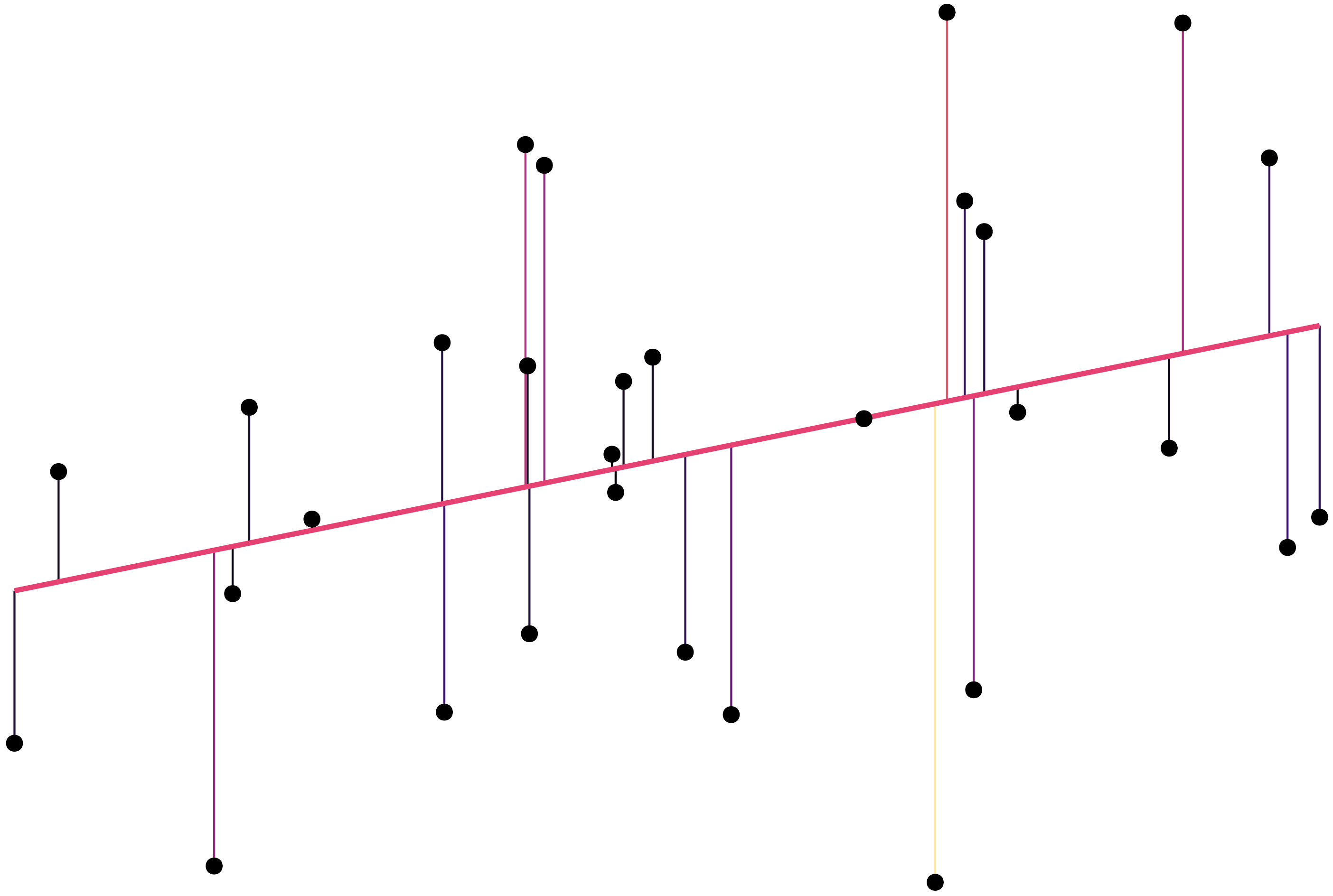
Each point $(y_i,\, x_i)$ has an associated loss (error).



The L1 loss function weights all errors equally: $\sum_i |y_i - \hat{y}_i|$



The L2 loss function *upweights* large weights: $\sum_i (y_i - \hat{y}_i)^2$



Model accuracy

Overfitting

So what's the big deal? (**Hint:** Look up.)

Model accuracy

Overfitting

So what's the big deal? (**Hint:** Look up.)

We're facing a tradeoff—increasing model flexibility

- offers potential to better fit complex systems
- risks overfitting our model to the training data

Model accuracy

Overfitting

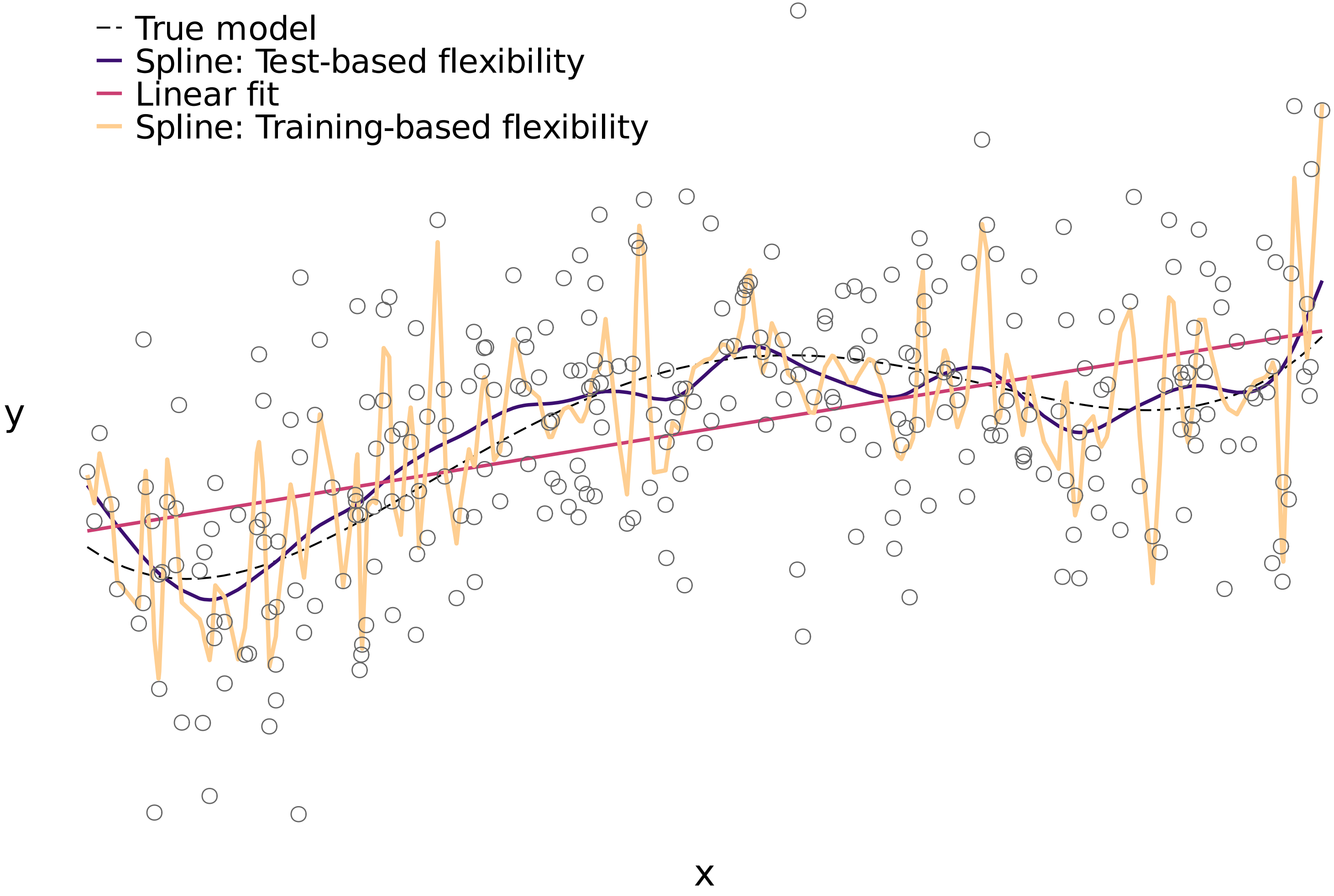
So what's the big deal? (**Hint:** Look up.)

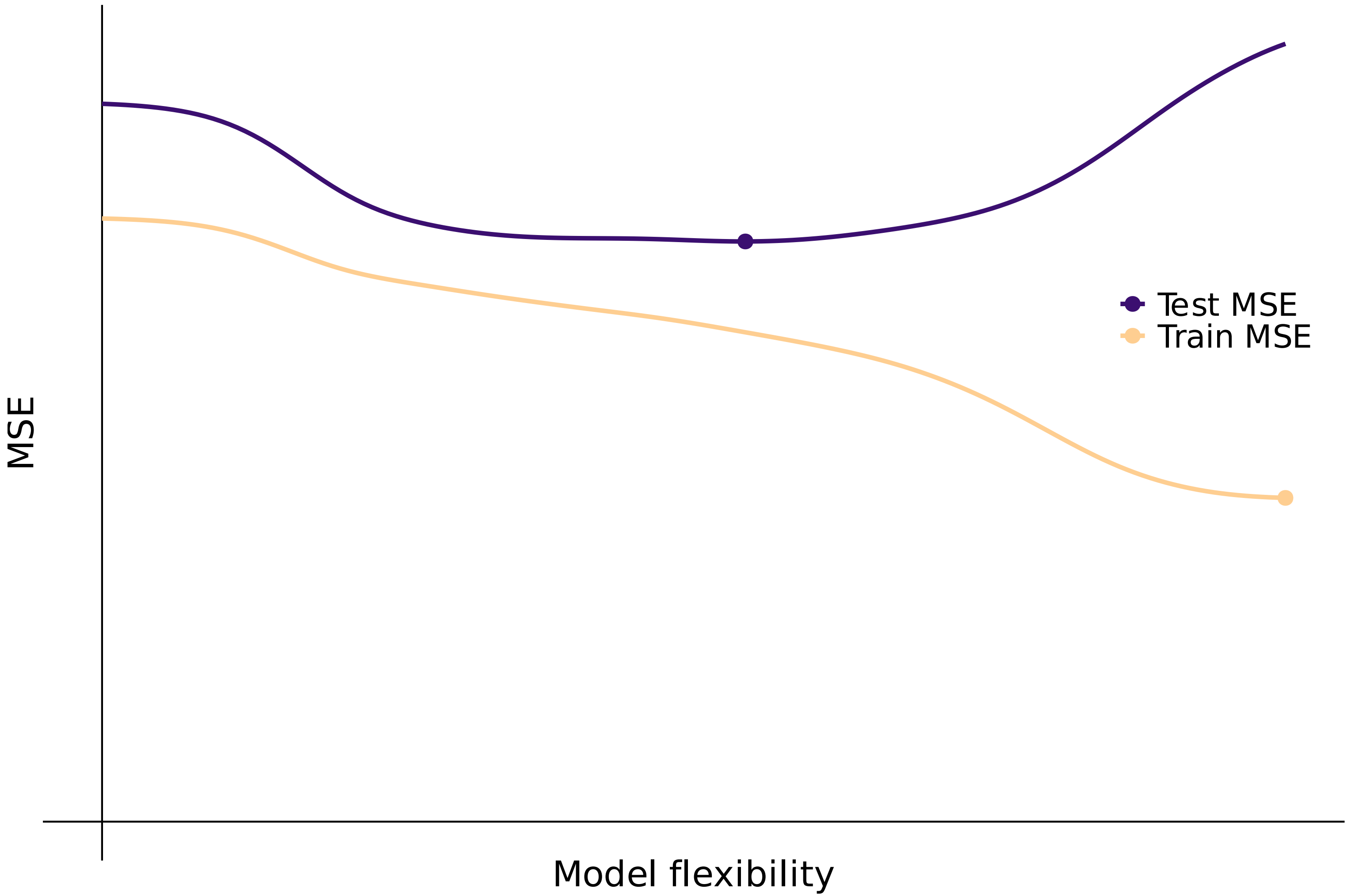
We're facing a tradeoff—increasing model flexibility

- offers potential to better fit complex systems
- risks overfitting our model to the training data

We can see these tradeoffs in our **test MSE** (but not the **training MSE**).

Training data and example models (splines)

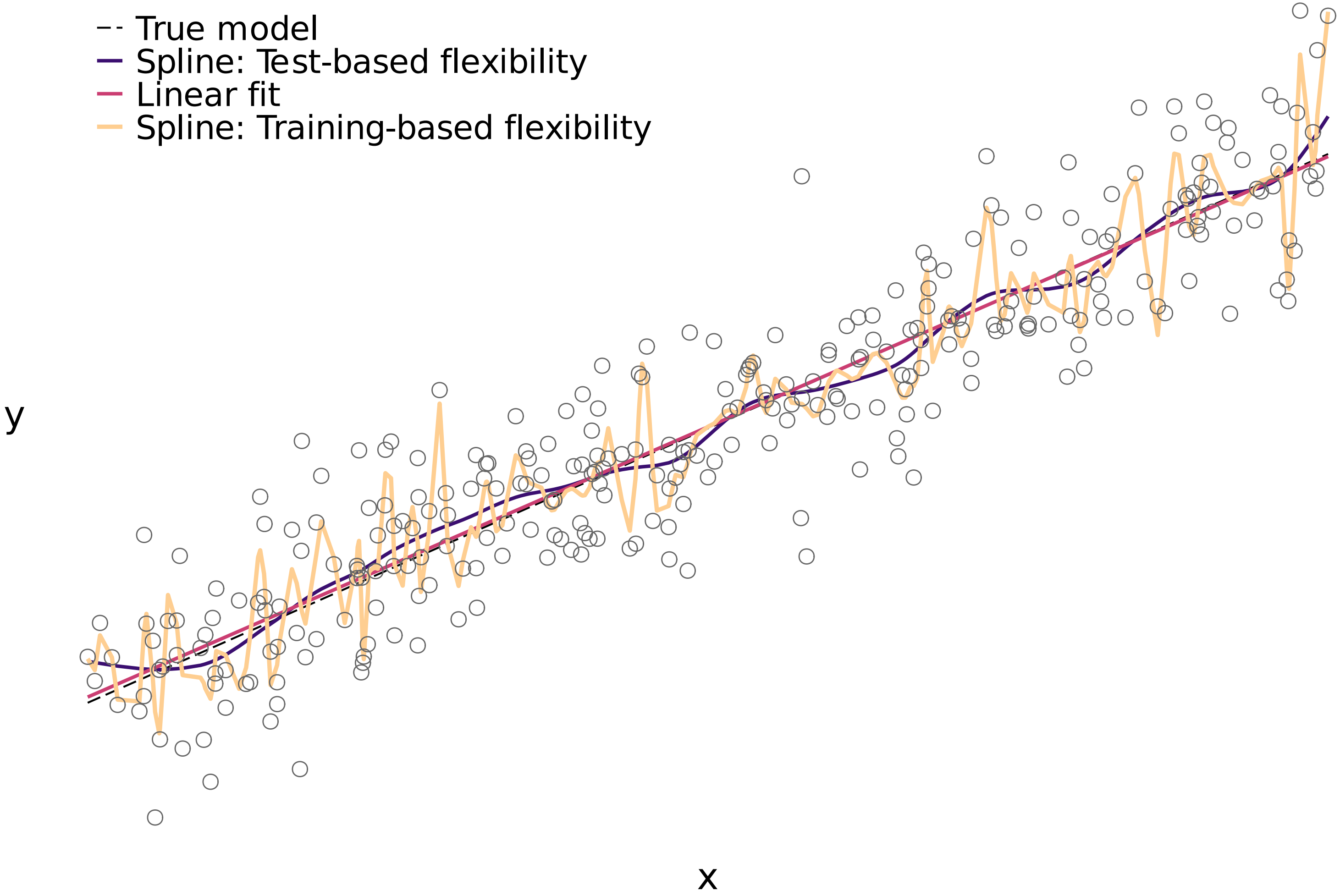


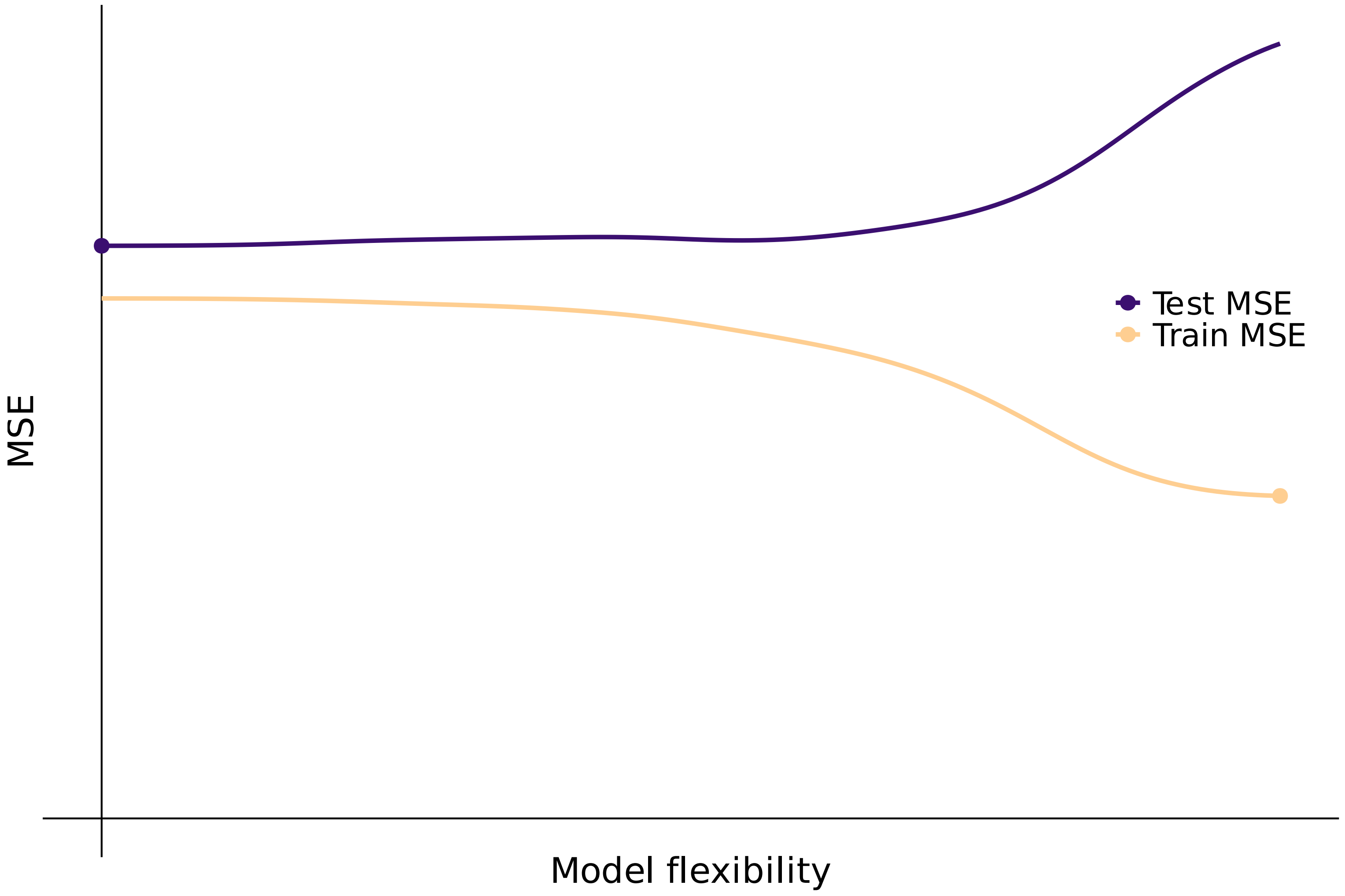

MSE
Model flexibility
Test MSE
Train MSE

The previous example has a pretty nonlinear relationship.

Q What happens when truth is actually linear?

Training data and example models (splines)




Test MSE
Train MSE
MSE
Model flexibility

Model accuracy

Solutions?

Clearly we don't want to overfit our **training data**.

It loos like our **testing data** can help.

Q How about the following routine?

1. train a model $\hat{f}$ on the **training data**
2. use the **test data** to "tune" the model's flexibility
3. repeat steps 1–2 until we find the optimal level of flexibility

Model accuracy

Solutions?

Clearly we don't want to overfit our **training data**.
It loos like our **testing data** can help.

Q How about the following routine?

1. train a model $\hat{f}$ on the **training data**
2. use the **test data** to "tune" the model's flexibility
3. repeat steps 1–2 until we find the optimal level of flexibility

A **No**

Model accuracy

Solutions?

Clearly we don't want to overfit our **training data**.

It loos like our **testing data** can help.

Q How about the following routine?

1. train a model $\hat{f}$ on the **training data**
2. use the **test data** to "tune" the model's flexibility
3. repeat steps 1–2 until we find the optimal level of flexibility

A **No!**

Model accuracy

Solutions?

Clearly we don't want to overfit our **training data**.
It loos like our **testing data** can help.

Q How about the following routine?

1. train a model $\hat{f}$ on the **training data**
2. use the **test data** to "tune" the model's flexibility
3. repeat steps 1–2 until we find the optimal level of flexibility

A **No!!!**

Model accuracy

Solutions?

Clearly we don't want to overfit our **training data**.
It loos like our **testing data** can help.

Q How about the following routine?

1. train a model $\hat{f}$ on the **training data**
2. use the **test data** to "tune" the model's flexibility
3. repeat steps 1–2 until we find the optimal level of flexibility

A **No!!!** This is an algorithm for **overfitting your test data**.

Okay... so maybe that was on overreaction, but we need to be careful.

Model accuracy

This tradeoff that we keep coming back to has an official name: **bias-variance tradeoff**. (or the variance-bias tradeoff)

Model accuracy

This tradeoff that we keep coming back to has an official name: **bias-variance tradeoff**. (or the variance-bias tradeoff)

Variance The amount $\hat{f}$ would change with a different **training sample**

- If new **training sets** drastically change $\hat{f}$, then we have a lot of uncertainty about f (and, in general, $\hat{f} \not\approx f$).
- More flexible models generally add variance to f.

Model accuracy

This tradeoff that we keep coming back to has an official name: **bias-variance tradeoff**. (or the variance-bias tradeoff)

Variance The amount $\hat{f}$ would change with a different **training sample**

- If new **training sets** drastically change $\hat{f}$, then we have a lot of uncertainty about f (and, in general, $\hat{f} \not\approx f$).
- More flexible models generally add variance to f.

Bias The error that comes from inaccurately estimating f.

- More flexible models are better equipped to recover complex relationships (f), reducing bias. (Real life is seldom linear.)
- Simpler (less flexible) models typically increase bias.

Model accuracy

The bias-variance tradeoff, formally

The expected value[†] of the **test MSE** can be written

$$E\left[\left(\mathbf{y}_0 - \hat{f}(\mathbf{X}_0)\right)^2\right] = \underbrace{\mathrm{Var}\left(\hat{f}(\mathbf{X}_0)\right)}_{(1)} + \underbrace{\left[\mathrm{Bias}\left(\hat{f}(\mathbf{X}_0)\right)\right]^2}_{(2)} + \underbrace{\mathrm{Var}(\varepsilon)}_{(3)}$$

[†] Think: *mean or tendency*

The bias-variance tradeoff, formally

The expected value[†] of the **test MSE** can be written

$$E\left[\left(\mathbf{y}_0 - \hat{f}(\mathbf{X}_0)\right)^2\right] = \underbrace{\text{Var}\left(\hat{f}(\mathbf{X}_0)\right)}_{(1)} + \underbrace{\left[\text{Bias}\left(\hat{f}(\mathbf{X}_0)\right)\right]^2}_{(2)} + \underbrace{\text{Var}(\varepsilon)}_{(3)}$$

Q_1 What does this formula tell us? (Think intuition/interpretation.)

Q_2 How does model flexibility feed into this formula?

Q_3 What does this formula say about minimizing **test MSE**?

[†] Think: *mean or tendency*

Model accuracy

The bias-variance tradeoff, formally

The expected value[†] of the **test MSE** can be written

$$E\left[\left(\mathbf{y}_0 - \hat{f}(\mathbf{X}_0)\right)^2\right] = \underbrace{\text{Var}\left(\hat{f}(\mathbf{X}_0)\right)}_{(1)} + \underbrace{\left[\text{Bias}\left(\hat{f}(\mathbf{X}_0)\right)\right]^2}_{(2)} + \underbrace{\text{Var}(\varepsilon)}_{(3)}$$

Q_1 What does this formula tell us? (Think intuition/interpretation.)

Q_2 How does model flexibility feed into this formula?

Q_3 What does this formula say about minimizing **test MSE**?

A_2 In general, model flexibility increases (1) and decreases (2).

[†] Think: *mean or tendency*

Model accuracy

The bias-variance tradeoff, formally

The expected value[†] of the **test MSE** can be written

$$E\left[\left(\mathbf{y}_0 - \hat{f}(\mathbf{X}_0)\right)^2\right] = \underbrace{\mathrm{Var}\left(\hat{f}(\mathbf{X}_0)\right)}_{(1)} + \underbrace{\left[\mathrm{Bias}\left(\hat{f}(\mathbf{X}_0)\right)\right]^2}_{(2)} + \underbrace{\mathrm{Var}(\varepsilon)}_{(3)}$$

Q_1 What does this formula tell us? (Think intuition/interpretation.)

Q_2 How does model flexibility feed into this formula?

Q_3 What does this formula say about minimizing **test MSE**?

A_2 In general, model flexibility increases (1) and decreases (2).

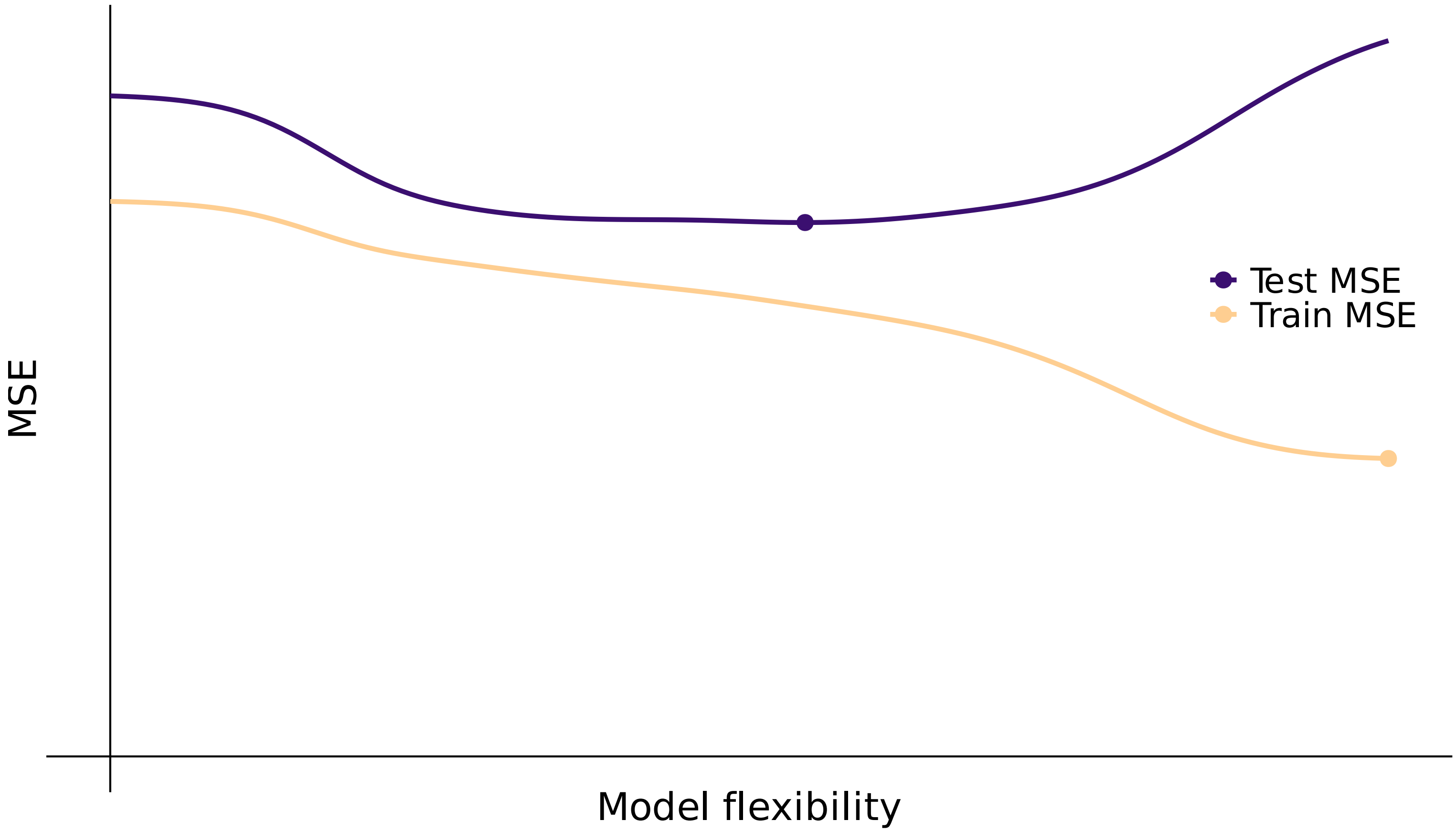
A_3 Rates of change for variance and bias will lead to optimal flexibility.

We often see U-shape curves of **test MSE** w.r.t. to model flexibility.

[†] Think: *mean* or *tendency*

U-shaped test MSE w.r.t. model flexibility

Increases in variance eventually overcome reductions in (squared) bias.



Model accuracy

Bias-variance tradeoff

The bias-variance tradeoff key to understanding many ML concepts.

- Loss functions and model performance
- Overfitting and model flexibility
- Training and testing (and cross validating)

Spend some time thinking about it and building intution.

It's time well spent.

So far we've focused on regression problems; what about classification?

Model accuracy

Classification problems

Recall We're still supervised, but now we're predicting categorical labels.

With categorical variables, MSE doesn't work—*e.g.*,

$$\mathbf{y} - \hat{\mathbf{y}} = \text{(Chihuahua) - (Blueberry muffin)} = \text{not math (does not compute)}$$

Clearly we need a different way to define model performance.

Model accuracy

Classification problems

The most common approach is exactly what you'd guess...

Training error rate The share of training predictions that we get wrong.

$$\frac{1}{n}\sum_{i=1}^{n}\mathbb{I}(y_i \neq \hat{y}_i)$$

where $\mathbb{I}(y_i \neq \hat{y}_i)$ is an indicator function that equals 1 whenever our prediction is wrong.

Model accuracy

Classification problems

The most common approach is exactly what you'd guess...

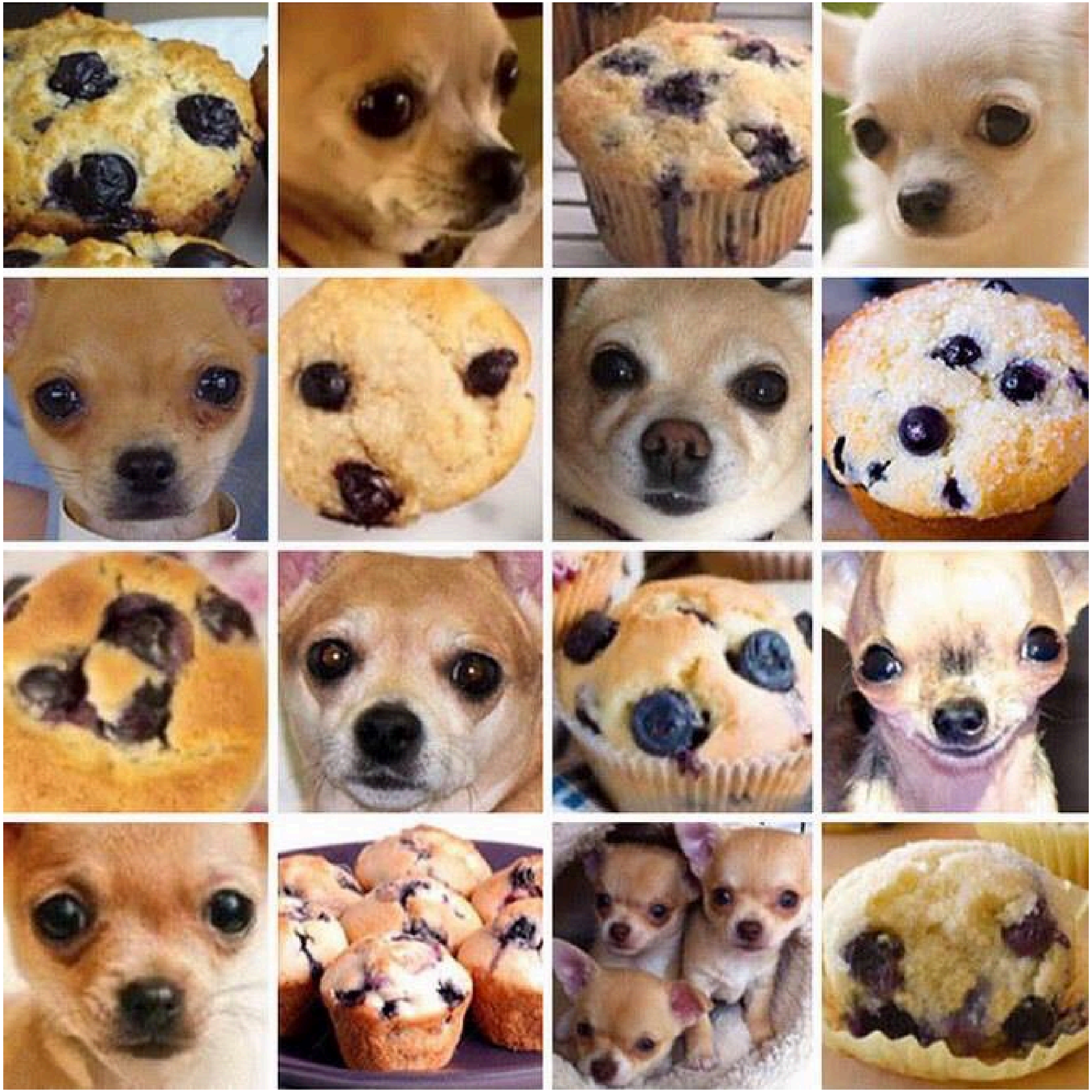
Training error rate The share of training predictions that we get wrong.

$$\frac{1}{n}\sum_{i=1}^{n}\mathbb{I}(y_i \neq \hat{y}_i)$$

where $\mathbb{I}(y_i \neq \hat{y}_i)$ is an indicator function that equals 1 whenever our prediction is wrong.

Test error rate The share of test predictions that we get wrong.

Average $\mathbb{I}(y_0 \neq \hat{y}_0)$ in our **test data**



Model accuracy

The Bayes classifier

Recall **Test error rate** is the share of test predictions that we get wrong.

Average $\mathbb{I}(y_0 \neq \hat{y}_0)$ in our **test data**

The **Bayes classifier** as the classifier that assigns an observation to its most probable groups, given the values of its predictors, *i.e.*,

Assign obs. i to the class j for which $\mathbf{Pr}(\mathbf{y} = j | \mathbf{X} = \mathbf{x}_0)$ is the largest

The **Bayes classifier** minimizes the **test error rate**.

Model accuracy

The Bayes classifier

Recall **Test error rate** is the share of test predictions that we get wrong.

Average $\mathbb{I}(y_0 \neq \hat{y}_0)$ in our **test data**

The **Bayes classifier** as the classifier that assigns an observation to its most probable groups, given the values of its predictors, *i.e.*,

Assign obs. i to the class j for which $\mathbf{Pr}(\mathbf{y} = j|\mathbf{X} = \mathbf{x}_0)$ is the largest

The **Bayes classifier** minimizes the **test error rate**.

Note $\mathbf{Pr}(\mathbf{y} = j|\mathbf{X} = x_0)$ is the probability that random variable $\mathbf{y}$ equals j, given[†] the variable(s) $\mathbf{X} = x_0$.

[†] The "given" is also read as "conditional on". Think of it as subsetting to where $X = x_0$.

Model accuracy

The Bayes classifier

Example

- Pr(y = "chihuahua" | X = "orange and purple") = 0.3
- Pr(y = "blueberry muffin" | X = "orange and purple") = 0.4
- Pr(y = "squirrel" | X = "orange and purple") = 0.2
- Pr(y = "other" | X = "orange and purple") = 0.1

Then the Bayes classifier says we should predict "blueberry muffin".

Model accuracy

The Bayes classifier

More notes on the Bayes classifier

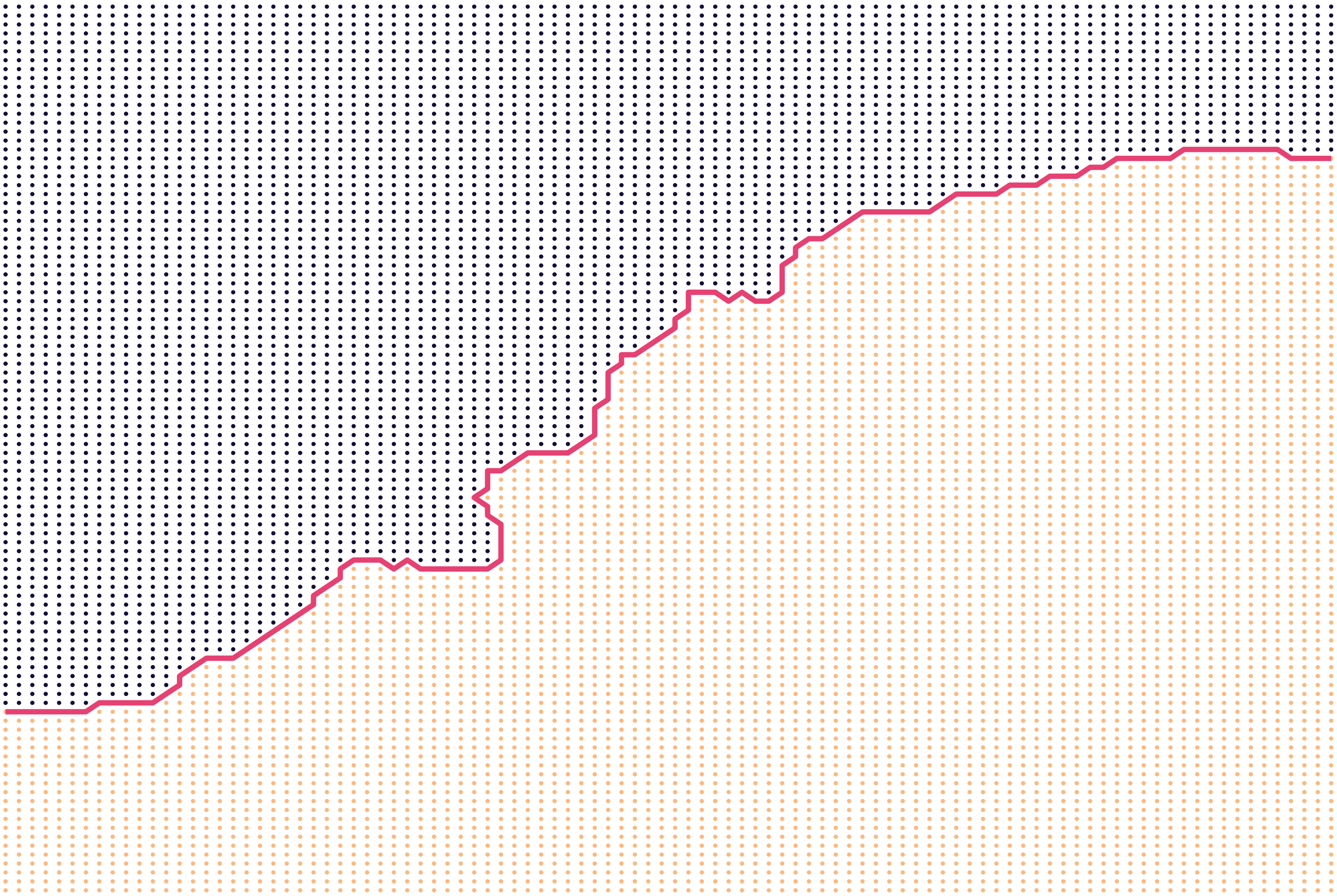
1. In the **two-class case**, we're basically looking for $\Pr(\mathbf{y} = j|\mathbf{X} = x_0) > 0.5$ for one class.

2. The **Bayes decision boundary** is the point where the probability is equal between the most likely groups (*i.e.*, exactly 50% for two groups).

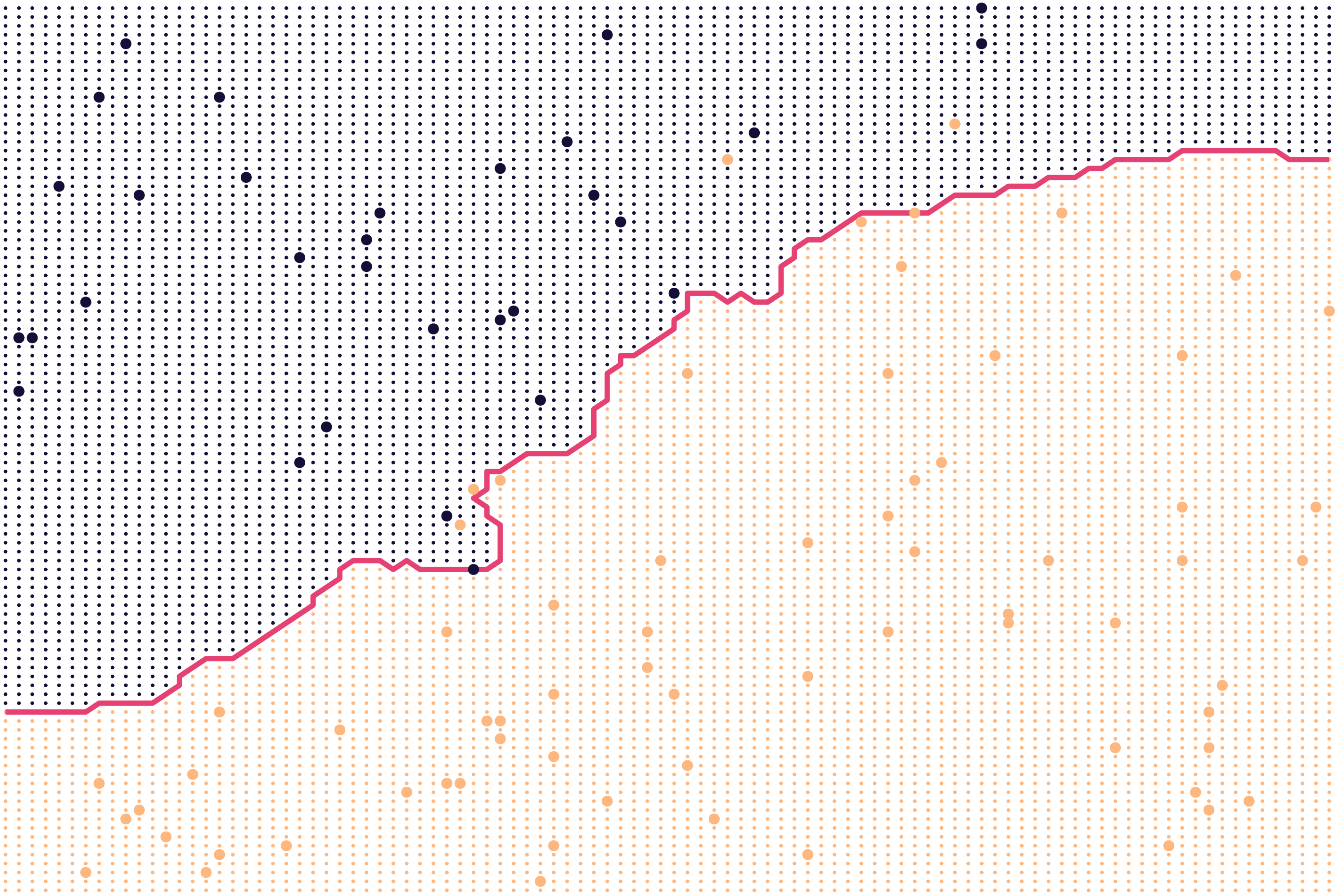
3. The Bayes classifier produces the lowest possible **test error rate**, which is called the **Bayes error rate**.

4. Just as with f, the probabilities $\Pr(\mathbf{y} = j|\mathbf{X} = x_o)$ that the Bayes classifier relies upon are **unknown**. We have to estimate.

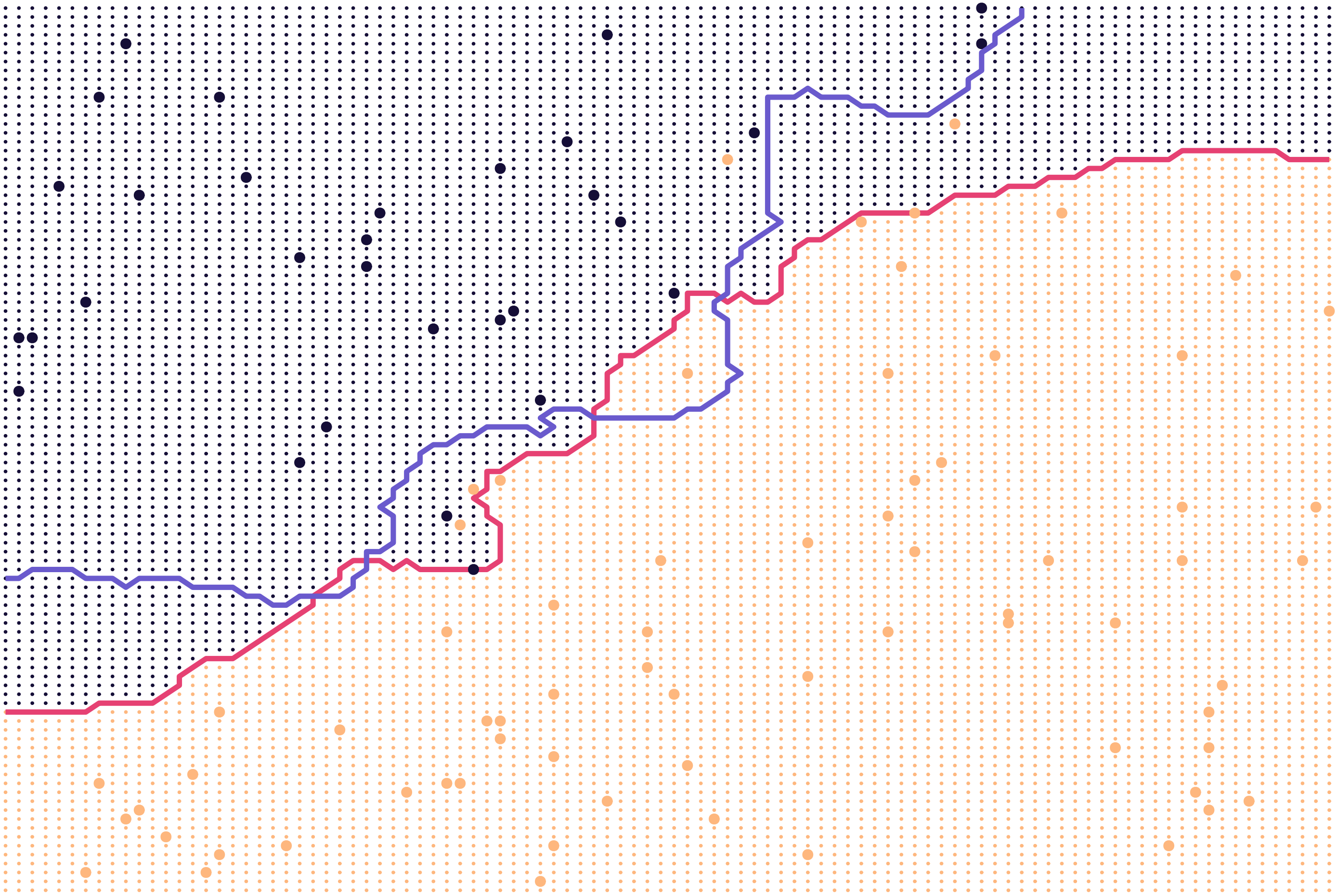
The **Bayes decision boundary** between classes A and B



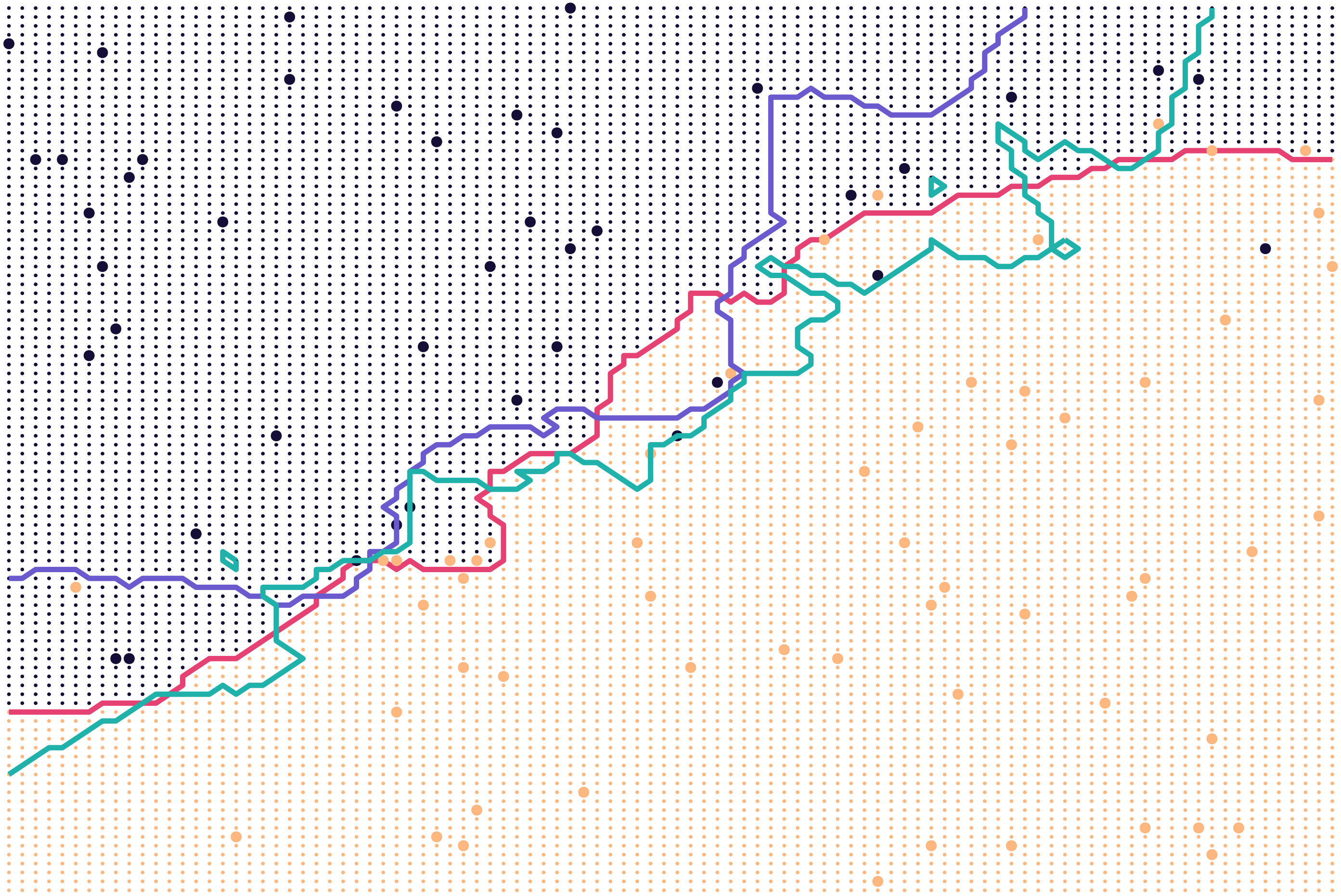
Now we sample...



… and our sample gives us an **estimated decision boundary**.



And a new sample gives us another **estimated decision boundary**.



One non-parametric way to estimate these unknown conditional probabilities: K-nearest neighbors (KNN).

K-nearest neighbors

Setup

K-nearest neighbors (KNN) simply assigns a category based upon the nearest K neighbors votes (their values).

K-nearest neighbors

Setup

K-nearest neighbors (KNN) simply assigns a category based upon the nearest K neighbors votes (their values).

More formally: Using the K closest neighbors[†] to test observation $\mathbf{x_0}$, we calculate the share of the observations whose class equals j,

$$\hat{\Pr}\left(\mathbf{y} = j|\mathbf{X} = \mathbf{x_0}\right) = \frac{1}{K}\sum_{i\in\mathcal{N}_0}\mathbb{I}(\mathbf{y}_i = j)$$

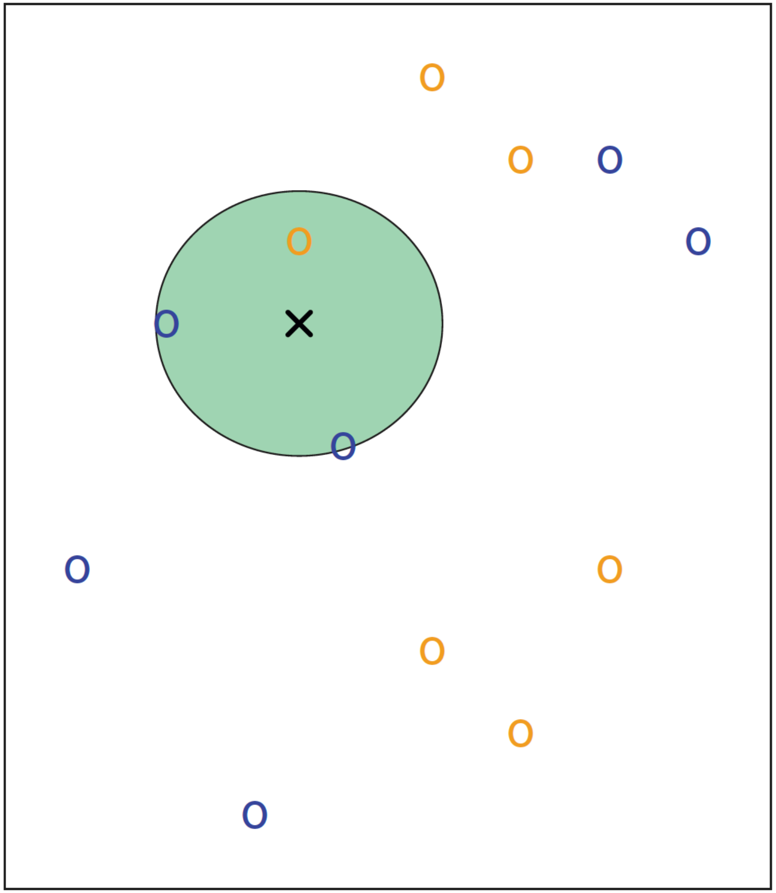
These shares are our estimates for the unknown conditional probabilities.

We then assign observation $\mathbf{x_0}$ to the class with the highest probability.

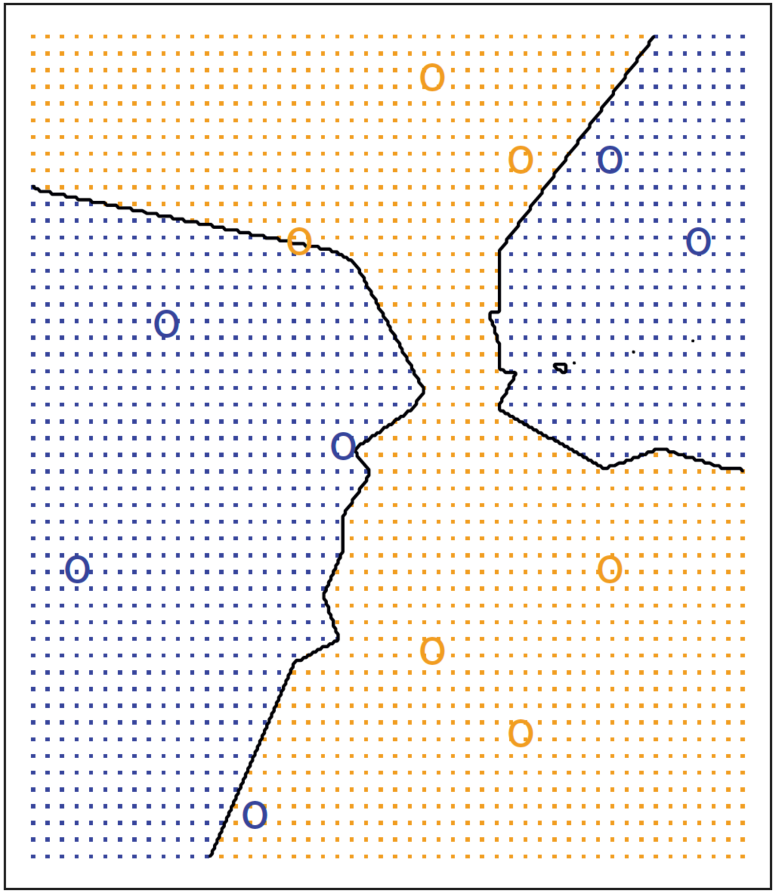
[†] In $\mathbf{X}$ space.

KNN in action

Left: K=3 estimation for "x".



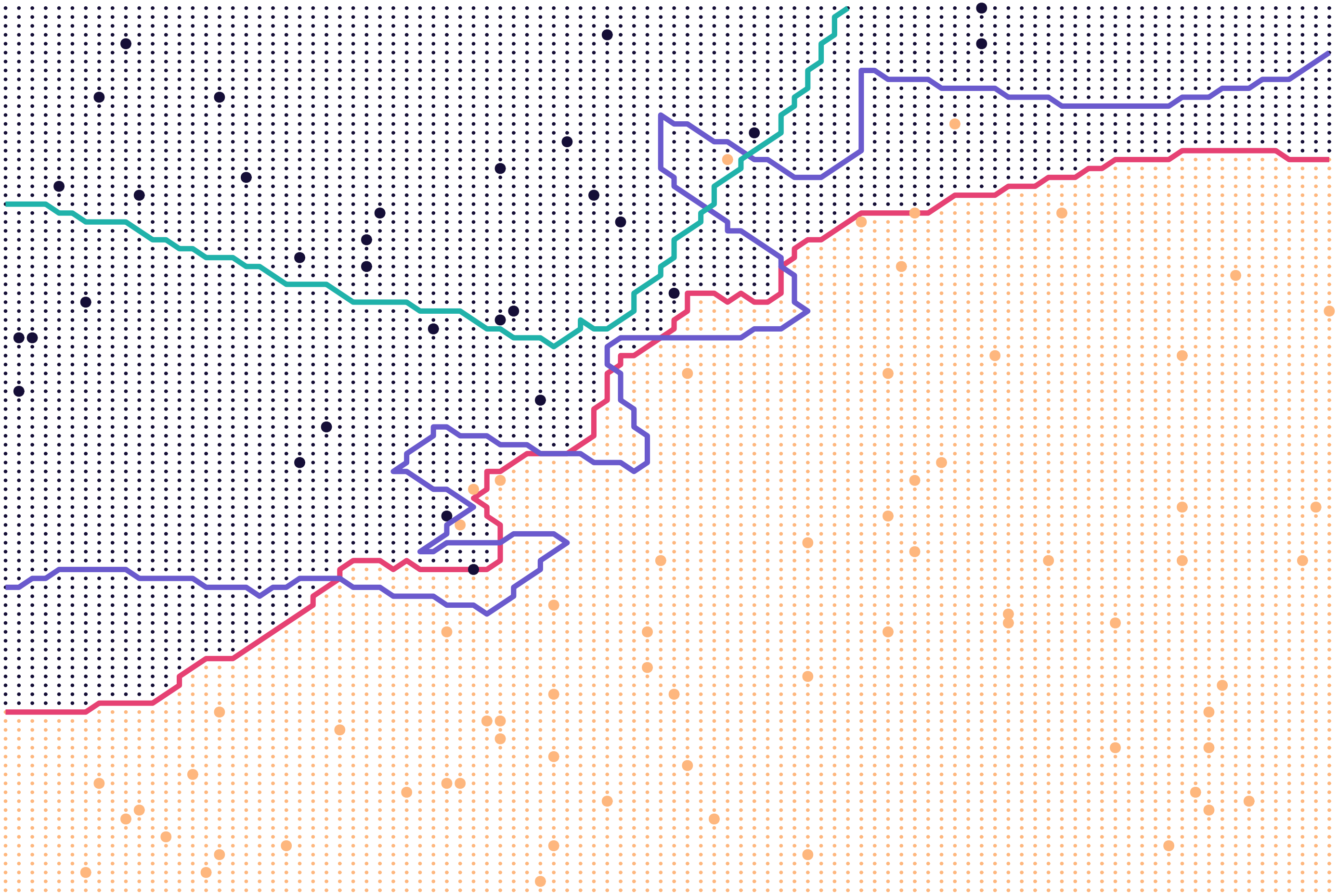
Right: KNN decision boundaries.



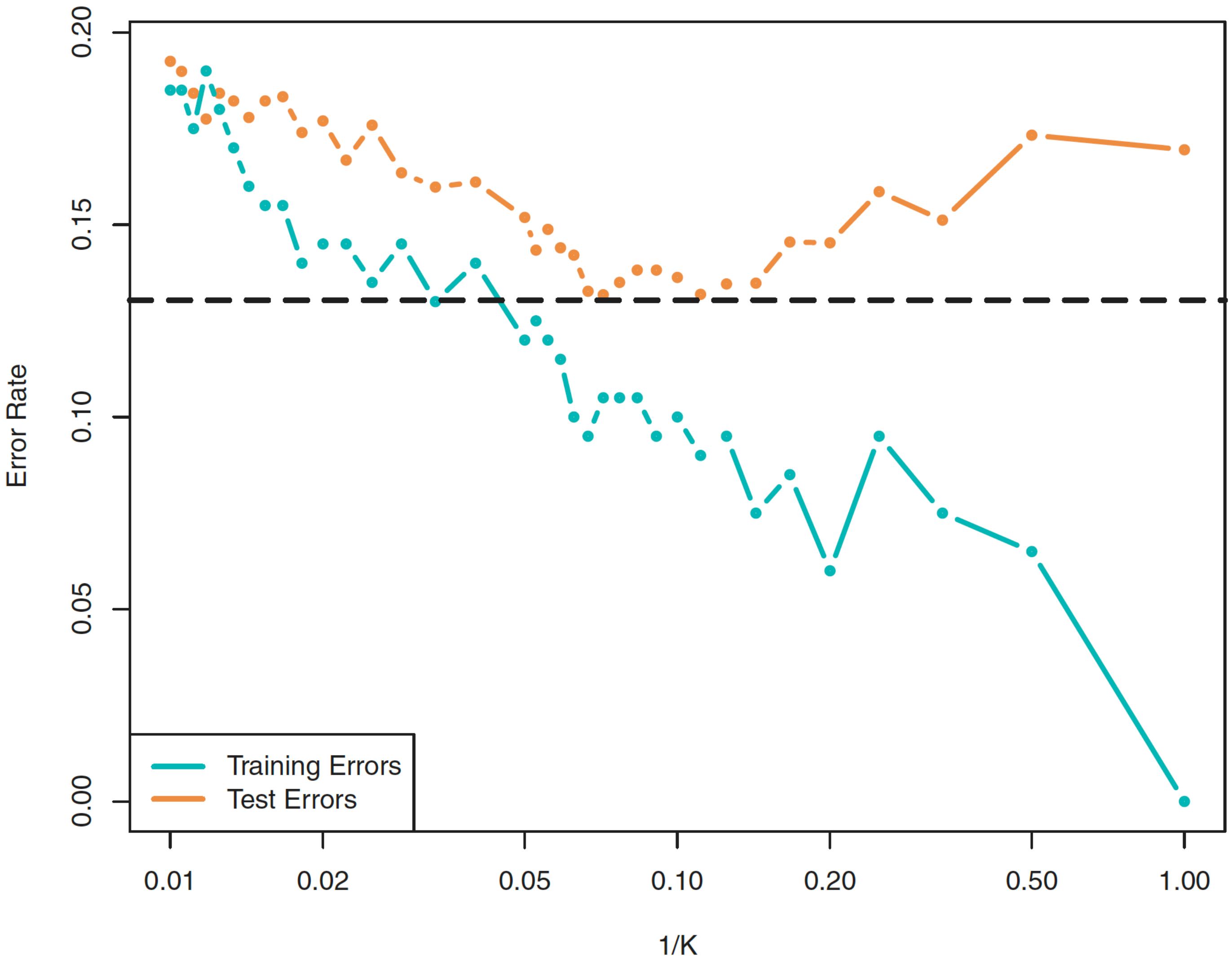
Source: ISL

The choice of K is very important—ranging from super flexible to inflexible.

Decision boundaries: **Bayes**, **K=1**, and **K=60**



KNN error rates, as K increases



Source: ISL

Model accuracy

Summary

The bias-variance tradeoff is central to quality prediction.

- Relevant for classification and regression settings
- Benefits and costs of increasing model flexibility
- U-shaped test error curves
- Avoid overfitting—including in test data

Sources

These notes draw upon

- An Introduction to Statistical Learning (*ISL*)
  James, Witten, Hastie, and Tibshirani
- Python Data Science Handbook
  Jake VanderPlas
- 'Chihuahua or Muffin' is from Twitter

Table of contents

Admin

Model accuracy: Regression

Model accuracy: Classification

KNN

Examples

Other