STA 141A – Fundamentals of Statistical Data Science

Department of Statistics; University of California, Davis

Instructor: Dr. Akira Horiguchi (ahoriguchi@ucdavis.edu)
A01 TA: Zhentao Li (ztlli@ucdavis.edu)
A02 TA: Zijie Tian (zijtian@ucdavis.edu)
A03 TA: Lingyou Pang (lyopang@ucdavis.edu)

Section 5: Overview of Statistical Learning

Spring 2025 (Mar 31 – Jun 05), MWF, 01:10 PM – 02:00 PM, Young 198

Comments

Based on Chapters 1 and 2 of ISL book James et al. (2021).

Section 3: Main concepts of statistical learning

- Statistical learning
- Supervised learning
- Assessing model accuracy
- Unsupervised learning

A note on generating "random" numbers

How do `sample()` and `rnorm()`, for instance, really work?

- Are the generated values really random? Technically, NO.
- The generated values are based on a certain *seed* (a positive integer). The seed is the 'starting point' or an initial value. It is plugged into a certain function/generator leading to our generated value.
- **Setting a seed (e.g. `set.seed(23)` in R) at the beginning of your code will lead to the same sequence of "random" numbers. This helps to make your "random" results more reproducible, which aids with debugging, science replicability, etc. It is recommended that you do this.**
  - If you want a different sequence of "random" numbers, replace 23 with your favorite positive integer.
- The generated values are called *pseudo-random*, so "almost random", as they appear to be random.
- If the seed is not specified, it is usually based on milliseconds of the computer's current time. This helps make the numbers "more random."

Further detail is outside the scope of this class.

Section 3: Main concepts of statistical learning

Statistical learning

How does advertising affect sales? (supervised learning)

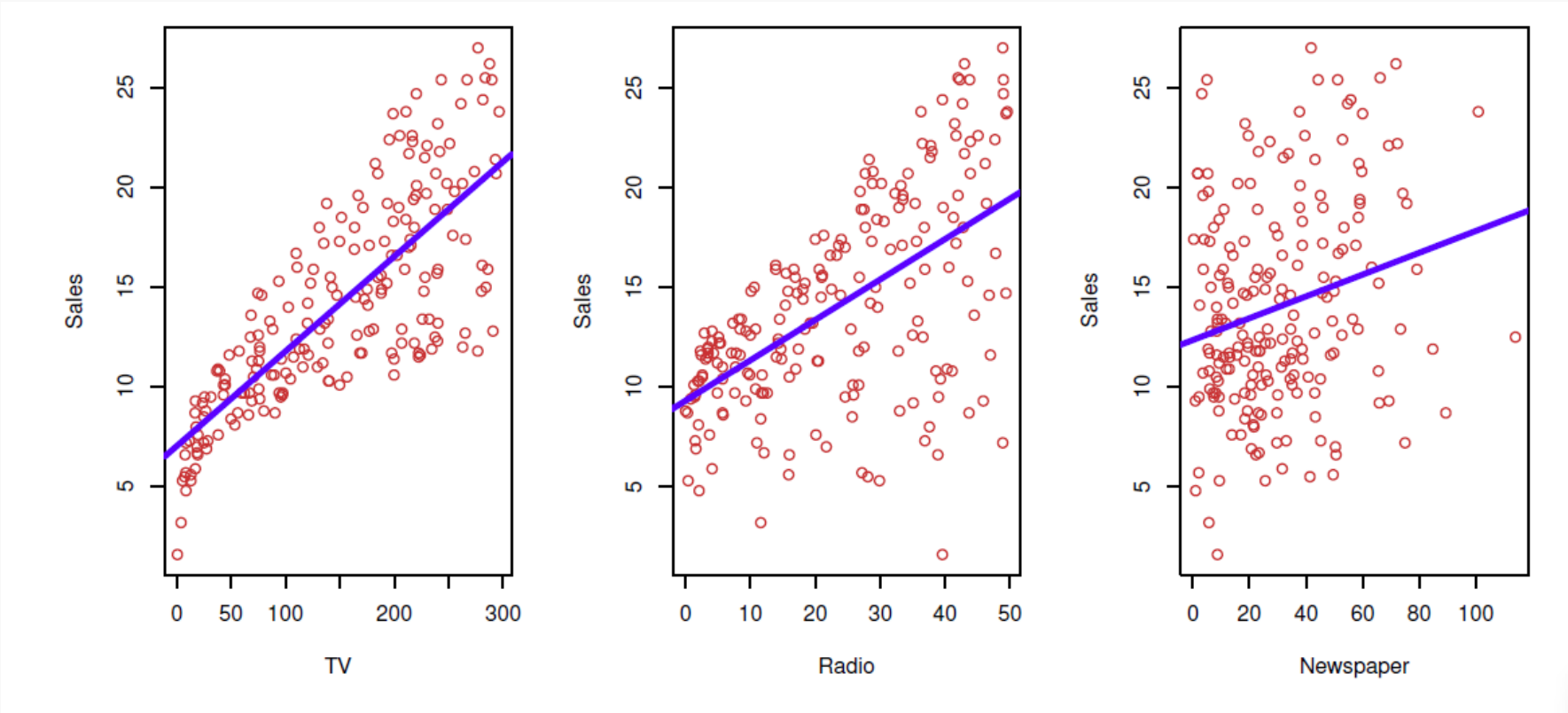

Figure 1: Image by James et al. (2021), based on the `Advertising` data set in R. The plot diplays `sales` in thousands of units depending on the input TV, `radio` and `newspaper` (advertising) budgets, in thousand dollars, for 200 different marktes.

Who will default on credit card payment? (supervised learning)

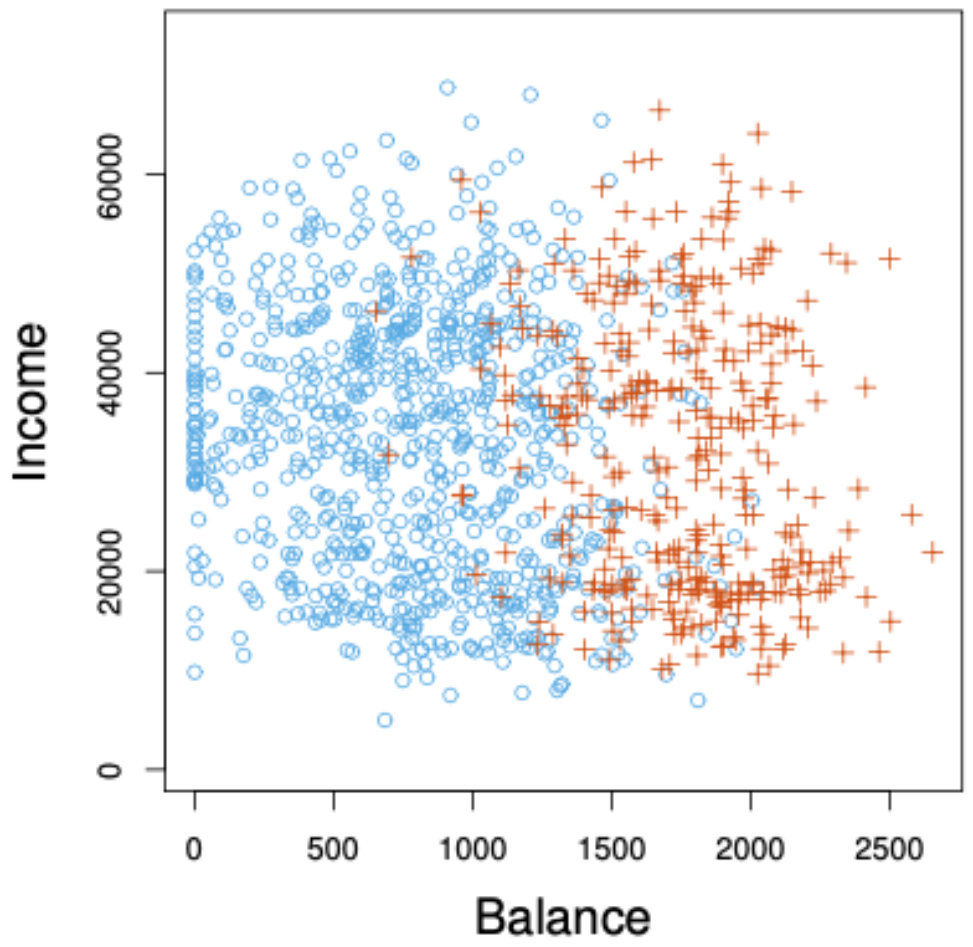

Figure 2: Image by James et al. (2021). The `Default` data set. The annual incomes and monthly credit card balances of a number of individuals. Orange $+$s indicate individuals who defaulted on their credit card payments; blue circles indicate individuals who did not default.

Flow cytometry (unsupervised learning)

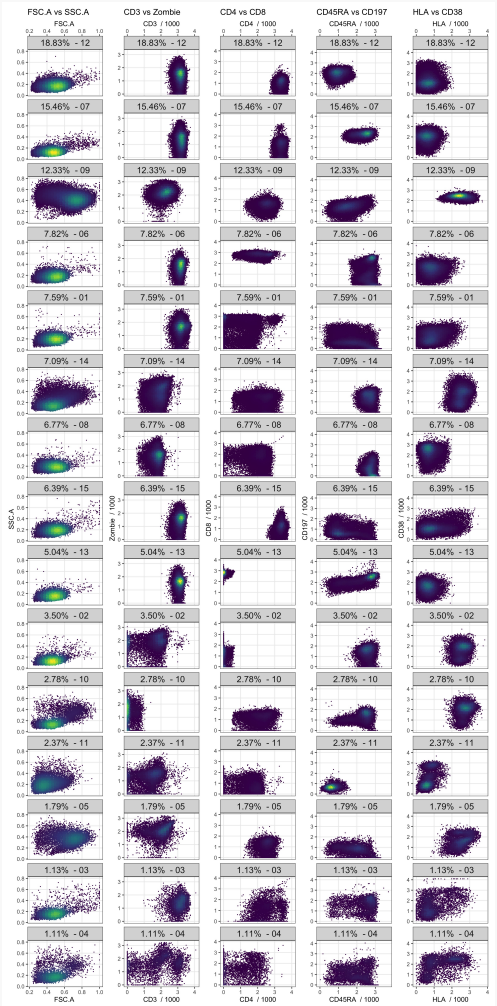

Figure 3: Image by Horiguchi et al. (2024) – https://projecteuclid.org/journals/bayesian-analysis/advance-publication/A-Tree-Perspective-on-Stick-Breaking-Models-in-Covariate-Dependent/10.1214/24-BA1462.full

Statistical learning: supervised vs unsupervised

Statistical learning refers to a vast set of tools for understanding data.

- These tools can be classified as *supervised* or *unsupervised*.
- *Supervised* statistical learning: predict or estimate an output based on one or more inputs. (STA 142A)
- *Unsupervised* statistical learning: learn relationship or structure among observations. (STA 142B)
- (Are there outputs to "supervise" the learning task?)

Section 3: Main concepts of statistical learning

Supervised learning

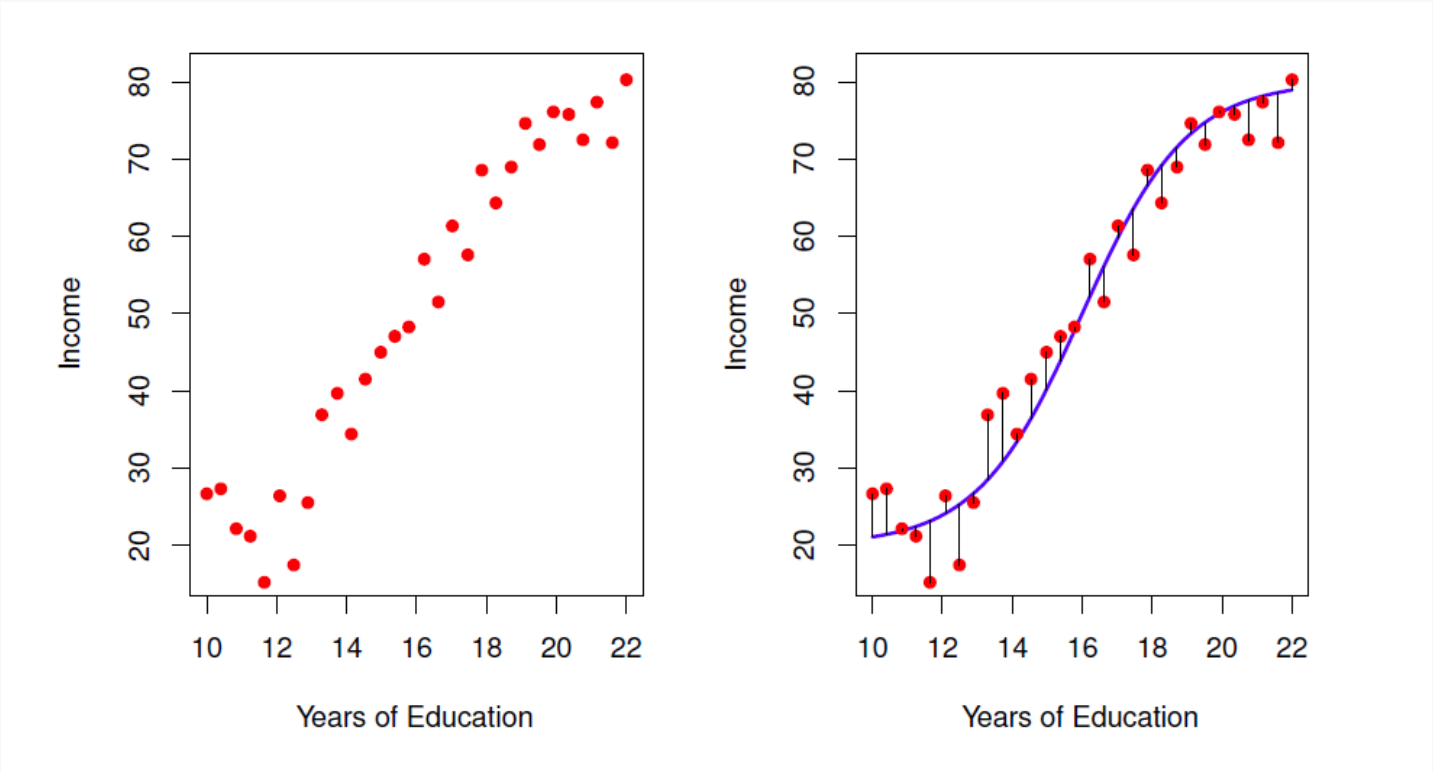

Figure 4: Image by James et al. (2021), based on the `Income` data set in R. The red dots are the observed values of `income` in tens of thousand dollars and `years of education` for 30 individuals.

A three-dimensional plot

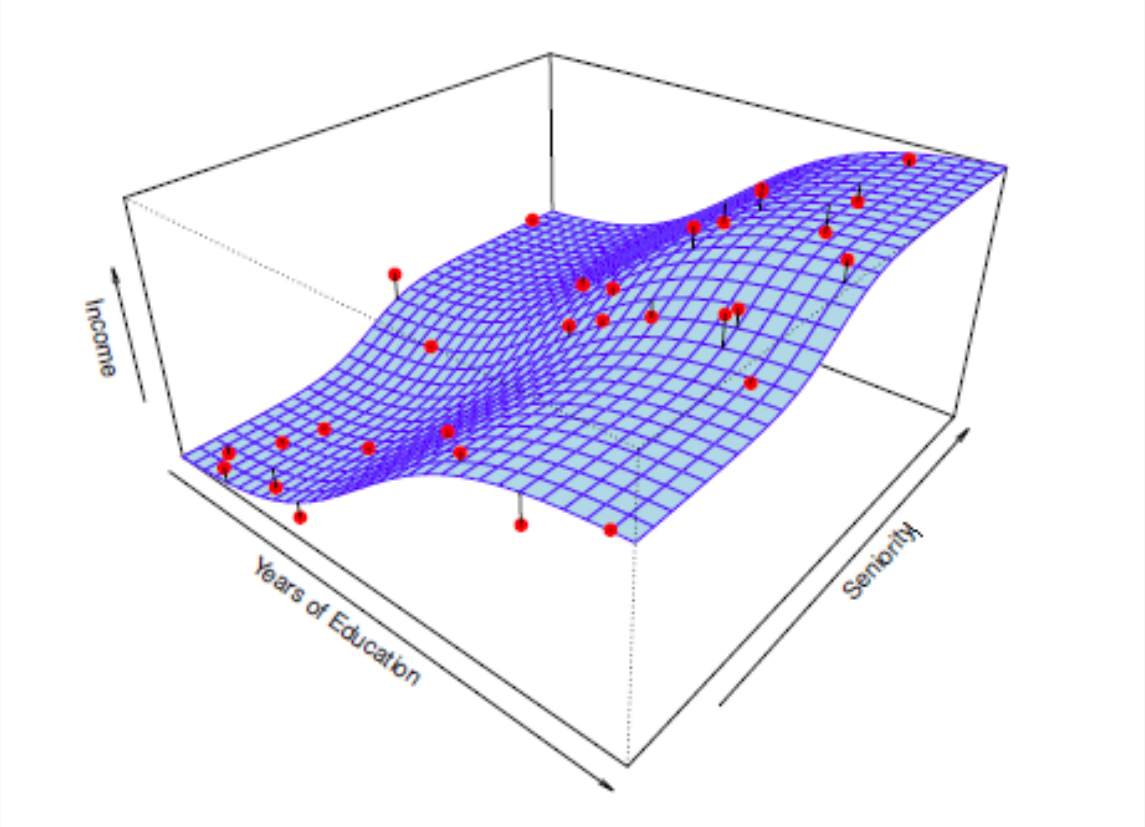

Figure 5: Image by James et al. (2021), based on the `Income` data set in R. The `income` is displayed as a function of `years of education` and `seniority`.

Combined plots

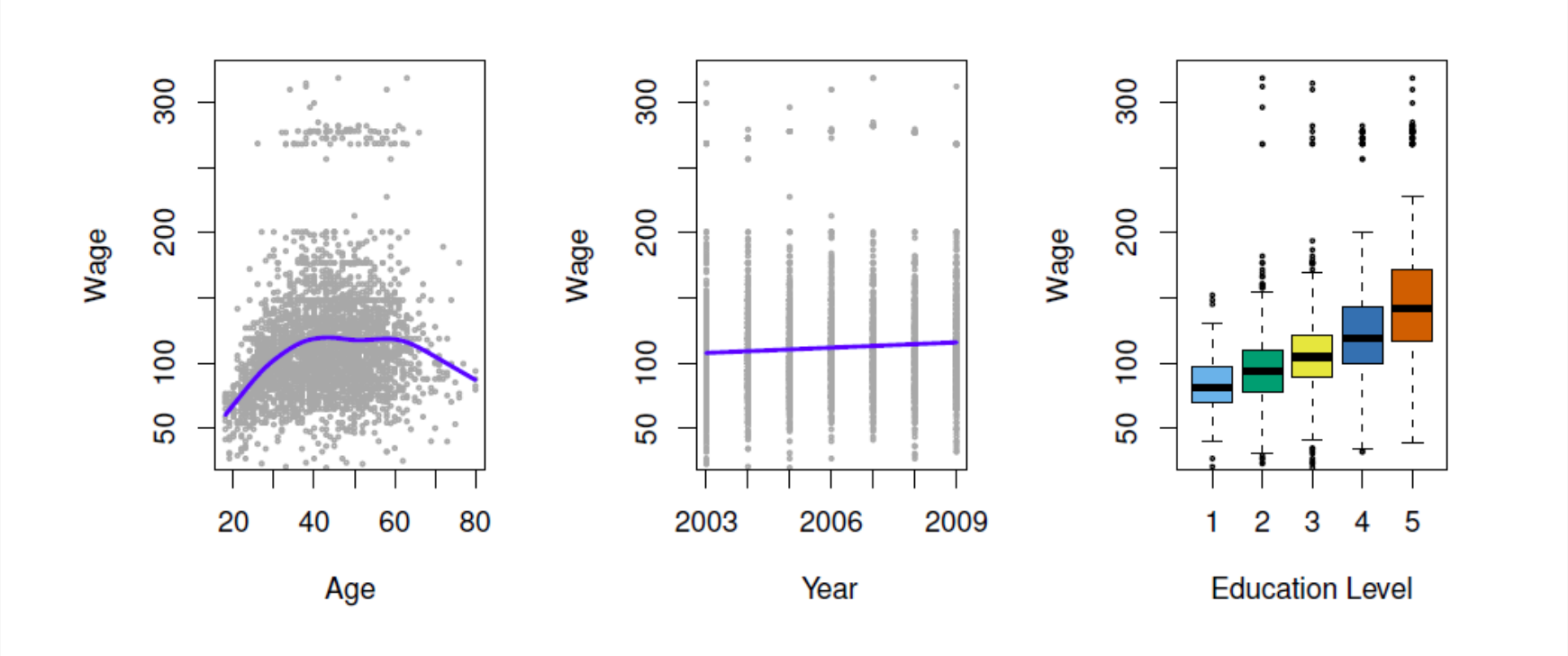

Figure 6: Image by James et al. (2021), based on the `Wage` data set in R. The `wage` is displayed as a function of `age`, `year` and `education`.

Some notation

Recall: predict or estimate an output based on one or more inputs.

- *Input variables* are called *predictors*, *independent variables*, or *features*; denoted by X, and X_1, X_2, X_3 etc. if there is more than one.
- *Output variable* is called *response* or *dependent variable*; denoted by Y.

Example: In Figure 1, the predictors are TV, `radio`, `newspaper`, denoted by X_1, X_2, X_3, respectively, and the response is `sales`, denoted by Y.

Suppose we observe a quantitative (i.e., numeric) response Y, and predictors $X_1, \ldots, X_p$ for some $p \in \mathbb{N}$.

- We assume that there is some relationship between the response Y and the vector of predictors $X = (X_1, \ldots, X_p)$, namely

$$Y = f(X) + \varepsilon. \tag{1}$$

  - f denotes a fixed but unknown function of $X_1, \ldots, X_p$.
  - ε is a random *error term*, which is independent of X, with $E(\varepsilon) = 0$.

- Two main reasons to estimate f: *prediction* and *inference*.

Goal: predict response Y at a set of inputs X.

- If X is available, because the error term averages to zero, we can predict Y using

$$\hat{Y} = \hat{f}(X), \tag{2}$$

  where $\hat{f}$ denotes the estimate for f, and $\hat{Y}$ the resulting prediction for Y.
- For prediction tasks, $\hat{f}$ is often treated as a *black box* – does it accurately predict Y?

Example: The blue surface in Figure 5 is an estimate $\hat{f}$ for the unknown function f describing the relationship of the predictors `years of education` and `seniority` to the response `income`:

$$\widehat{\texttt{income}} = \hat{f}(\texttt{years of education}, \texttt{seniority}).$$

The accuracy of $\hat{Y}$ as a prediction for Y depends on the *reducible error* and the *irreducible error*.

- If we interpret both the estimator $\hat{f}$ and X as fixed so that the only variability comes from ε, we get

$$\begin{aligned} E\left(Y-\hat{Y}\right)^2 &= E\left(f(X)-\hat{f}(X)+\varepsilon\right)^2 \\ &= E\left(f(X)-\hat{f}(X)\right)^2 - 2E\left((f(X)-\hat{f}(X))\varepsilon\right) + E\left(\varepsilon^2\right) \\ &= \underbrace{\left(f(X)-\hat{f}(X)\right)^2}_{\text{reducible error}} + \underbrace{Var(\varepsilon)}_{\text{irreducible error}}. \end{aligned}$$

- The *reducible error* can be potentially reduced by using a more appropriate learning technique to estimate f.
- The *irreducible error* comes entirely from ε, and does not depend on how we estimate f.
- Even if we would estimate f perfectly, i.e. $\hat{Y} = f(X)$, there is still some *irreducible* prediction error from ε, since $Y = f(X) + \varepsilon$.

Goal: learn relationship between response Y and inputs $X_1, \ldots, X_p$.

- One may want to answer:
  - Which predictors are associated with the response?
  - What is the relationship between the response and each predictor?
  - Can we use a linear equation to describe the relationship between $X_1, \ldots, X_p$ to Y, or is there a more complex relationship?
- Knowing more about f allows us to ask questions about Y, such as:
  - What value of $(X_1, \ldots, X_p)$ maximizes Y?
  - How much is Y affected by each predictor X_i?
    E.g., in Figure 1 we might have
    - 60% of $Var(\texttt{sales})$ can be explained by TV budget,
    - 30% of $Var(\texttt{sales})$ can be explained by `Radio` budget,
    - 8% of $Var(\texttt{sales})$ can be explained by `Newspaper` budget,
    - remaining 2% can be explained by $X_4, X_5, \ldots, X_p$
  - These questions can be difficult to answer if f is highly non-linear!
  - `https://www.climateinteractive.org/en-roads/`

How to estimate f?

Supervised learning: estimate unknown function f using n observed data points

$$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$$

where $x_i = (x_{i1}, x_{i2}, \ldots, x_{ip})^\top \in \mathbb{R}^p$.

- We index the observations by $i = 1, \ldots, n$.
- We index the inputs/predictors by $j = 1, \ldots, p$.
- Let $x_{ij} \in \mathbb{R}$ represent the value of the jth predictor for the ith observation.
- Let $y_i \in \mathbb{R}$ represent the response variable for the ith observation.
- The n observed data points are called the *training data* because they will train our method on how to estimate f.

A statistical learning method will use the training data to estimate unknown f.

- I.e., compute a function $\hat{f}$ such that $Y \approx \hat{f}(X)$ for any observed (X, Y).
- What kind of function is our function estimate $\hat{f}$ allowed to be?
  Dictated by our chosen learning method (*parametric* vs *non-parametric*).

Parametric methods involve two steps:

1. Define what estimates $\hat{f}$ of the function f are allowed to look like. E.g., allow $\hat{f}$ to be a linear function of $X = (X_1, \ldots, X_p)$:

$$\hat{f}(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p. \tag{3}$$

2. Use the *training data* to *fit*/*train* the model. For the linear model (3), we have to estimate the parameters $\beta_0, \beta_1, \ldots, \beta_p$ so that $Y \approx \hat{f}(X)$.

Called *parametric* because $\hat{f}$ is determined by finitely many parameters.

- Advantage: Reduces the problem of estimating an arbitrary p-dimensional function f to the easier problem of estimating only $p + 1$ parameters.
- Disadvantage: The allowed form of $\hat{f}$ likely will not match true form of f. One could try to use more *flexible* models (however, this could lead to *overfitting* the data, i.e. the models follows the errors too closely).
- The left plot in Figure 1 estimates the function f by

$$\hat{f}(\texttt{TV}, \texttt{Radio}, \texttt{Newspaper}) = \hat{f}(\texttt{TV}) \approx 6 + \frac{1}{20} X_1. \tag{4}$$

(Language: “make an assumption about the functional form of f”)

Non-parametric (or *nonparametric*) methods

Non-parametric methods: $\hat{f}$ is determined by infinitely many parameters.

- Misnomer: non-parametric $\neq$ no parameters!
- Advantage: more flexibility in what $\hat{f}$ is allowed to be $\Rightarrow$ more likely that $\hat{f}$ better estimates f.
- Disadvantage: requires a large number of observations to accurately estimate f.
- Disadvantage: inference is often more difficult.

Example: (next slide)

Example – non-parametric

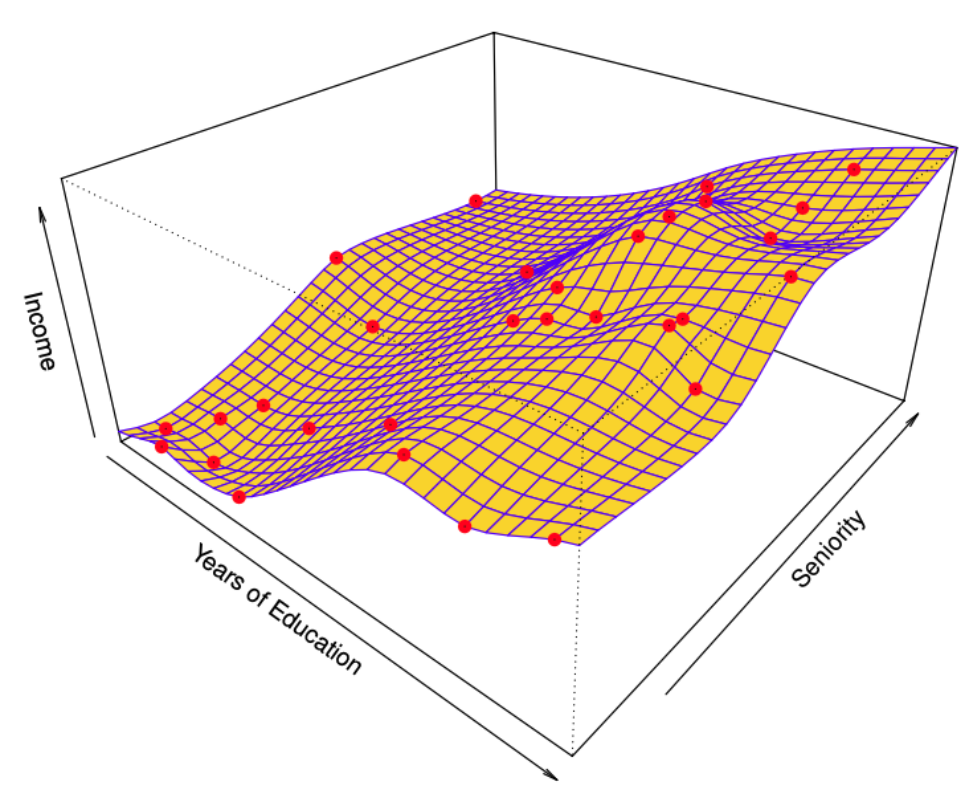

Figure 7: Image by James et al. (2021), based on the `Income` data set in R. "A rough thin-plate spline fit to the Income data from Figure 2.3. This fit makes zero errors on the training data." Here the yellow surface need only be continuous.

- Interpolation theorem: Any n bivariate data points (x_i, y_i) (where x_is are distinct) can be interpolated by a polynomial of order at most $n - 1$. `https://en.wikipedia.org/wiki/Polynomial_interpolation`

Choosing a model: interpretability vs flexibility

The choice of the model (e.g., model (1)) depends on whether we are interested in prediction, inference, or a combination of both.

- Simpler models typically are easier to interpret and make inference on.
- More flexible models might more accurately predict Y but are often less interpretable.
- Example: Simple linear regression allows an intercept and a slope (see Figure 1) compared to a quadratic/cubic/polynomial regression where even more parameters can be chosen (see Figure 5).

When choosing a statistical model for the data, there is usually a trade-off between interpretability and flexibility.

- One has to choose where to be on this spectrum (see next slide).

Choosing a model: interpretability vs flexibility

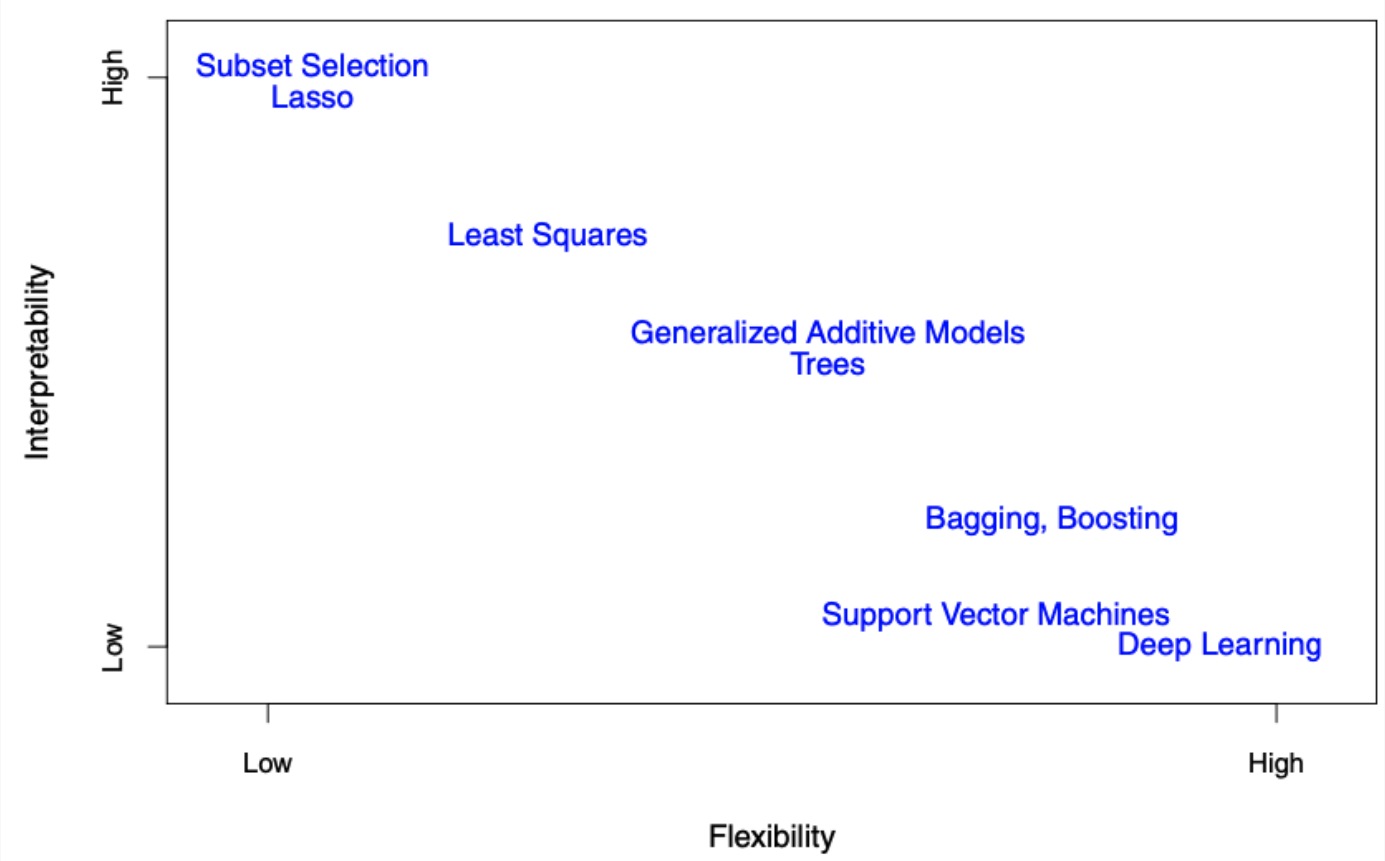

Figure 8: Image by James et al. (2021). "A representation of the tradeoff between flexibility and interpretability, using different statistical learning methods."

Regression vs. classification – Reasoning

Variables can be treated as:

- *quantitative* (numeric). E.g., age, height, income, house value, stock price.
- *qualitative/categorical* ("discrete" – value is one of K classes). E.g., marital status (married or not), brand of product purchased (brand A, B, or C).

One tends to select the statistical learning methods on the basis whether the response Y is quantitative or qualitative.

- Whether the predictors are quantitative or qualitative is less important.
- Problems with a quantitative response are usually referred to as *regression* problems.
- Problems with a qualitative response are usually referred to as *classification* problems.
  - Techniques include *logistic regression* and *K-nearest neighbors* (see Figure 9).
- The distinction is not always crisp; logistic regression can be thought of as either classification or regression.

Example of classification – K-nearest neighbors

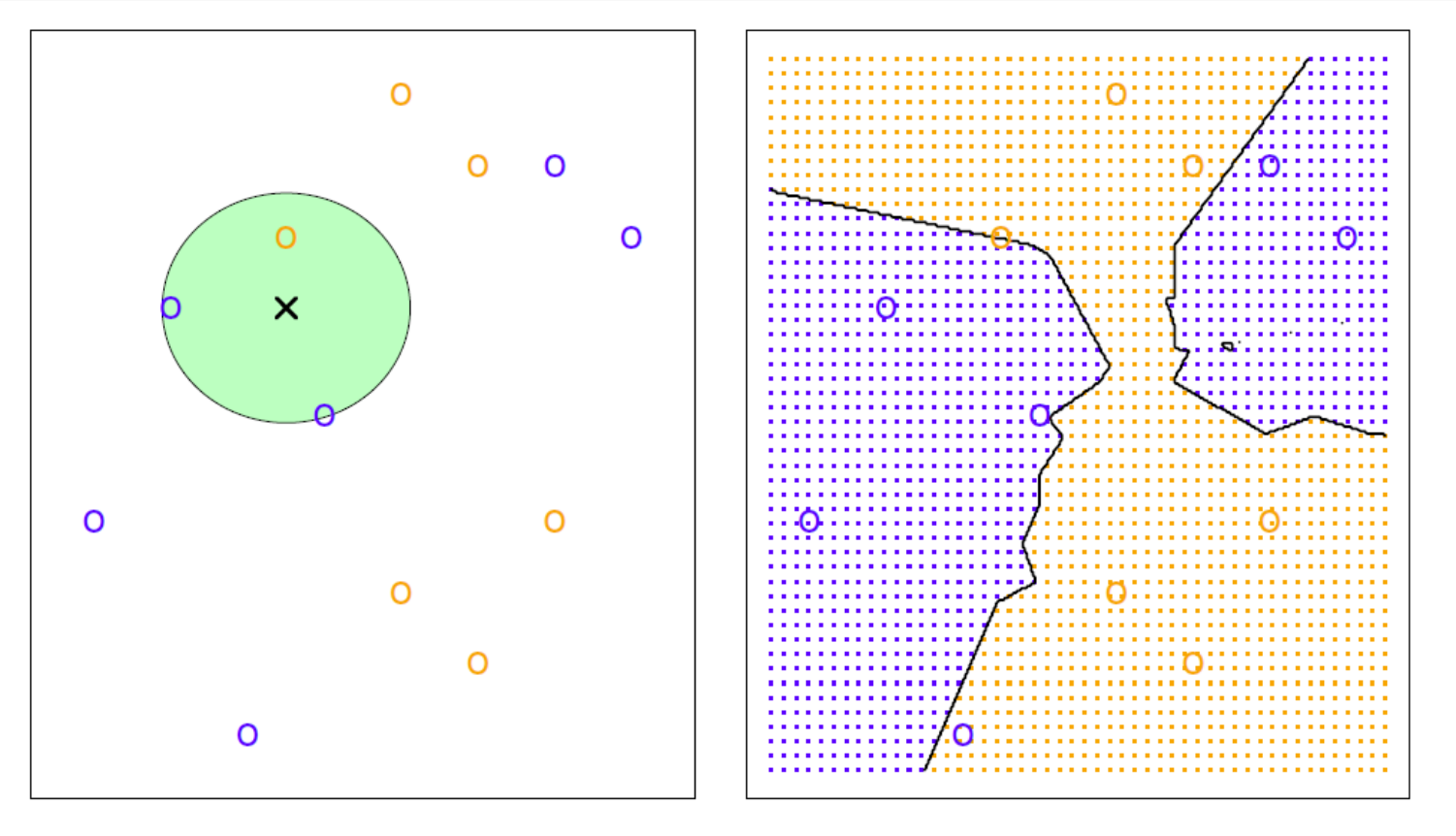

Figure 9: Image by James et al. (2021). Classification using the K-nearest neighbor approach with $K = 3$. Left: We assign a new observation (this is the "x") to the class for which most of three neighbors of "x" belong to. Right: A decision line/region how we would assign a new element for $K = 3$ if they would fall in a certain indicated region.

Section 3: Main concepts of statistical learning

Assessing model accuracy

General comments

- No method dominates *all* other methods – some methods work well on one particular data setting, but worse on another one.
- (Some methods perform poorly across all data settings, so we don't spend time discussing them.)
- In order to find an appropriate model, we have to analyze its quality of fit, establish a comparable measure (if possible), and choose the model which gives us the smallest approximation errors (in some sense).
- Also need to distinguish between regression and classification.

In regression, most commonly used measure is the *mean squared error* (MSE):

$$MSE = \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \hat{f}(x_i) \right)^2, \tag{5}$$

where $(x_1, y_1), \ldots, (x_n, y_n)$, with $n \in \mathbb{N}$, are data points.

- If MSE is computed on the *training data*, we call it the *training MSE*, denoted by MSE_{train}.
- We'd like our estimator $\hat{f}$ to have a low MSE on data it wasn't trained on:

$$MSE_{test} = \frac{1}{m} \sum_{i=1}^{m} \left(y_{n+i} - \hat{f}(x_{n+i}) \right)^2, \tag{6}$$

where $(x_{n+1}, y_{n+1}), \ldots, (x_{n+m}, y_{n+m})$, with $m \in \mathbb{N}$, are data points on which we can test our estimator $\hat{f}$. We call (6) the *test MSE*.

Mean squared error (MSE) – ii) Procedure

Conceptually, we can summarize the procedure to find a good estimator $\hat{f}$ by:

1. Find the estimator $\hat{f}$ of the assumed model (for instance linear model) by minimizing MSE_{train}.
2. i) If we don't have test data, select the model for which $\hat{f}$ gives the smallest MSE_{train}. (In the case that there is no test data, the procedure ends here).
   ii) If we have test data, calculate MSE_{test} by plugging in the derived estimator $\hat{f}$ minimizing MSE_{train} from each model assumption.
3. Select the model (and thus the related $\hat{f}$) leading to the smallest MSE_{test}.

Mean squared error (MSE) – iii) A sketch

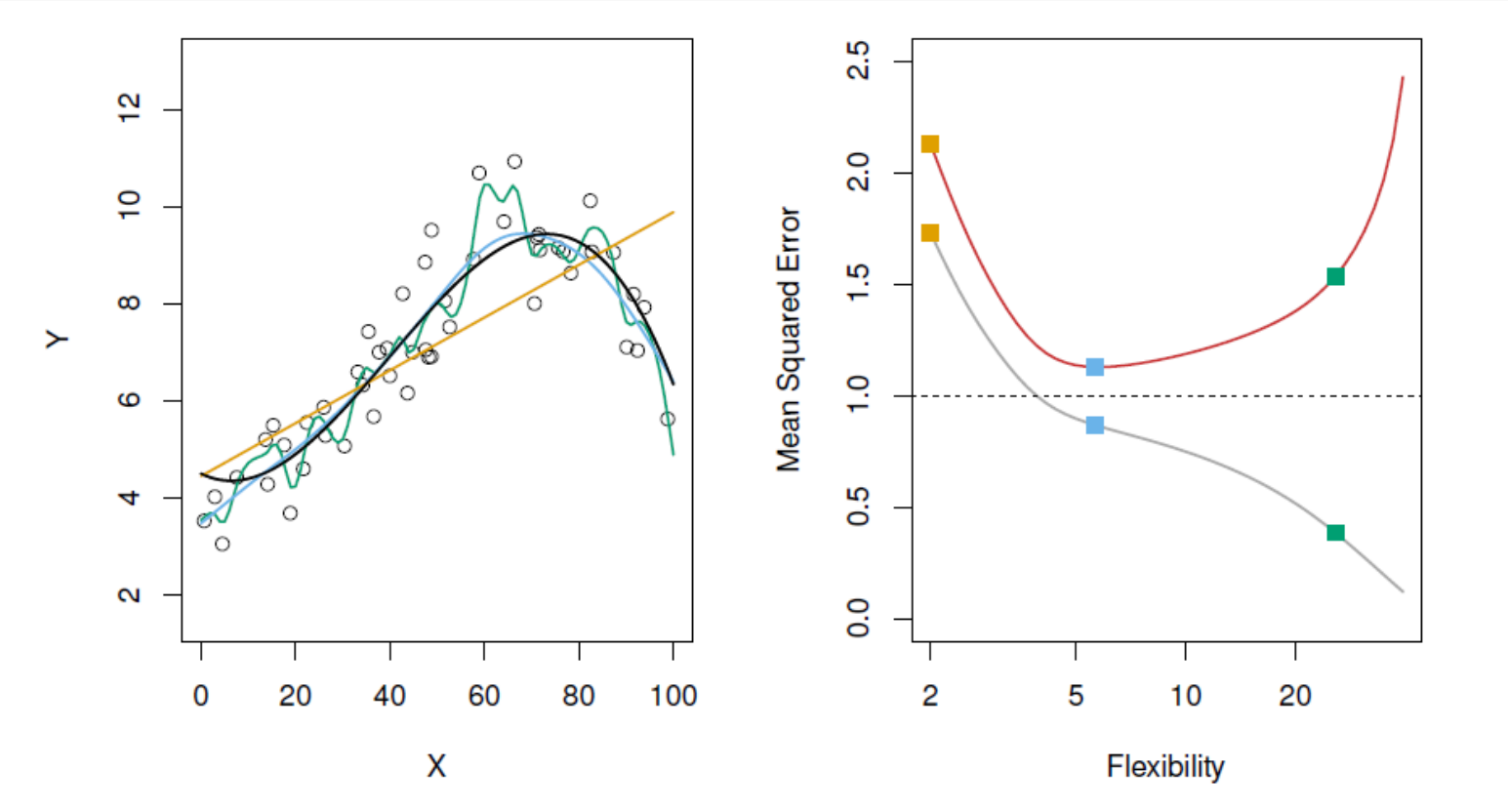

Figure 10: Image by James et al. (2021). Left: Data simulated from f (in black). Three estimates for f: Linear regression (in orange), and two smoothing splines (in blue and green). Right: Training MSE (in grey), test MSE (in red), and minimum possible test MSE over all methods (dashed line). The squares represent the three fits from the left panel.

Mean squared error (MSE) – iv) Comments on the sketch

- The dashed line in Figure 8 indicates the irreducible error (this is $Var(\varepsilon)$) which is the lowest achievable test MSE among all possible methods.
- Training MSE *always* decreases as we increase model flexibility.
- The U-shape in the test MSE in Figure 8 indicates that it decreases up to a certain amount of flexibility, but gets worse afterwards.
- When a given method yields a small training MSE, but a large test MSE, we say that the data points are *overfitted*.

Mean squared error (MSE) – v) Another sketch

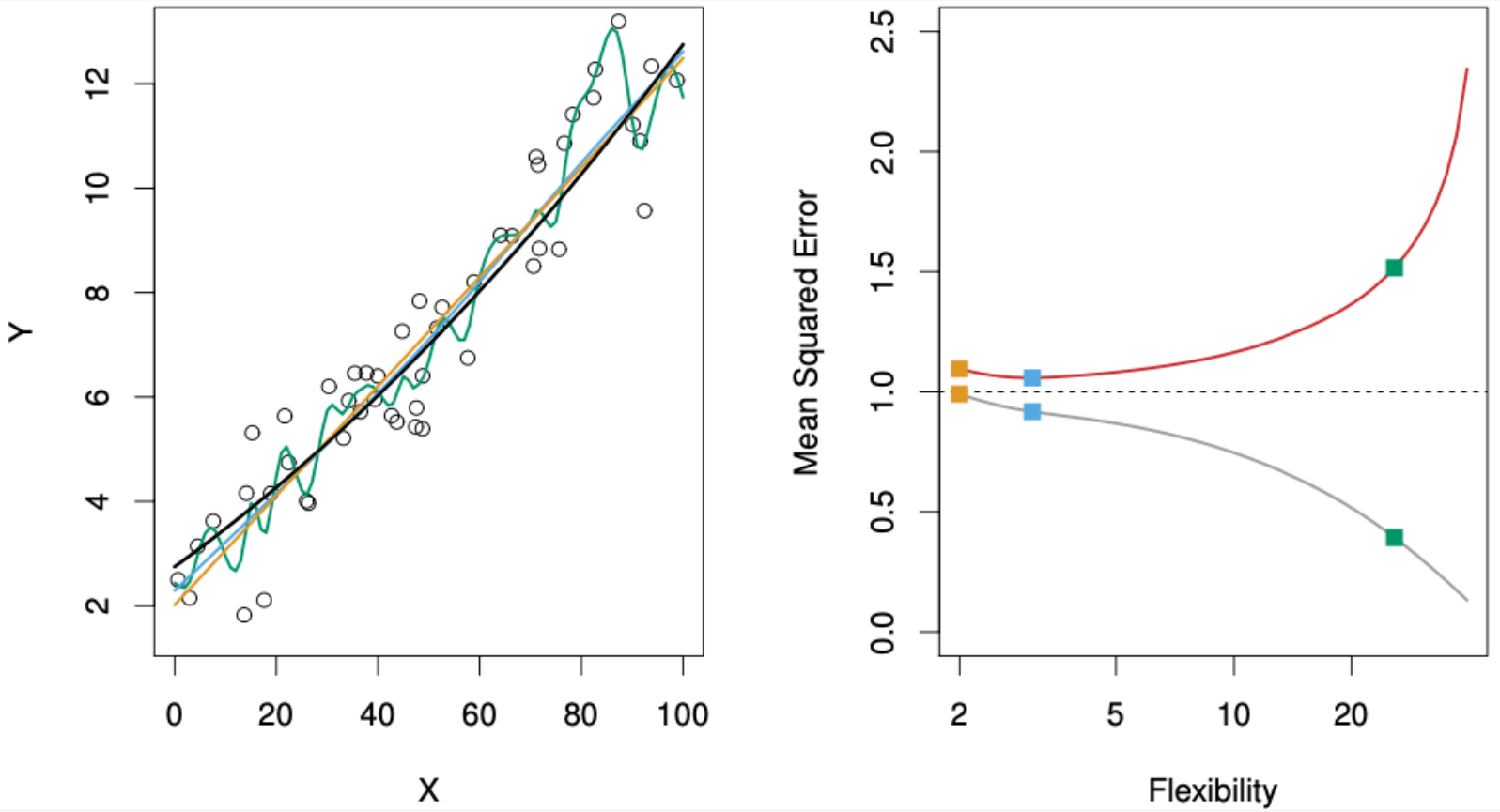

Figure 11: Image by James et al. (2021). Details are same as in previous figure, but using a different true f that is much closer to linear. In this setting, linear regression provides a very good fit to the data. U-shape is barely noticeable.

The U-shape observed in the test MSE curves turns out to be the result of two competing properties of statistical learning methods.

- Consider the *expected test MSE* at a new test data point (x, y). This is the average test MSE obtained if we repeatedly estimate f over infinitely many training data sets. (Training data is drawn from some underlying data distribution.)
- The *expected test MSE* at (x, y) can always be decomposed into the sum:

$$E\left(y - \hat{f}(x)\right)^2 = Var\left(\hat{f}(x)\right) + \left[Bias\left(\hat{f}(x)\right)\right]^2 + Var(\varepsilon). \tag{7}$$

  - $Var(\hat{f}(x))$ is the amount $\hat{f}$ would change by using different training data sets.
  - $\left[Bias(\hat{f}(x))\right]^2$ is the squared *bias* of $\hat{f}(x)$, where $Bias(\hat{f}(x)) := f(x) - E(\hat{f}(x))$.
    - Squared bias can be interpreted as the error introduced by approximating f using the given model assumptions (which often do not capture the full complexity of f).
  - $Var(\varepsilon)$ is the variance of the error term ε.
- The expected test MSE can never be below $Var(\varepsilon)$, because the variance and the squared bias are non-negative.

(7) is a *bias-variance trade-off*. As a general rule, as model flexibility increases, the variance will increase and the (squared) bias will decrease.

In classification, we quantify accuracy of $\hat{f}$ using the *training error rate*,

$$Err_{train} = \frac{1}{n} \sum_{i=1}^{n} I\left(y_i \neq \hat{f}(x_i)\right), \tag{8}$$

where $(x_1, y_1), \ldots, (x_n, y_n)$, with $n \in \mathbb{N}$, are training data.

- $I\left(y_i \neq \hat{f}(x_i)\right)$ is the *indicator variable* that equals 1 if $y_i \neq \hat{f}(x_i)$ and equals 0 if $y_i = \hat{f}(x_i)$.
- As $\hat{f}(x_i)$ is the predicted class given the observation x_i, the training error rate counts the average number of wrong classifications.
- A good classifier, however, produces a small *test error rate*

$$Err_{test} = \frac{1}{m} \sum_{i=1}^{m} I\left(y_{n+i} \neq \hat{f}(x_{n+i})\right), \tag{9}$$

  where $(x_{n+1}, y_{n+1}), \ldots, (x_{n+m}, y_{n+m})$, with $m \in \mathbb{N}$, are test data.

Bias-variance trade-off also appears in classification, as we will see.

The test error rate is on average minimized, by a classifier that assigns each observation to the most likely class given its predictor.

- In other words, such a classifier assigns a test observation x to the class

$$\arg\max_j P(Y = j|X = x). \tag{10}$$

- This classifier is called the *Bayes classifier*.
- Special case: if Y must belong to one of only two classes, we predict class 1 if $P(Y = 1|X = x) > 0.5$, and class 2 otherwise.
- The Bayes classifier produces the lowest possible test error rate, called the *Bayes error rate* — analogous to the irreducible error discussed earlier.

KNN – i) Idea

- In theory, we would always like to predict using the Bayes classifier, but in practice we don't know the conditional distribution of Y given X.
- Many approaches attempt to estimate the conditional probabilities.
- The *K-nearest neighbor* (KNN) classifier estimates the Bayes classifier by counting the $K \in \mathbb{N}$ closest values of x in a neighborhood $\mathcal{N}$

$$\widehat{P(Y = j|X = x)} = \frac{1}{K} \sum_{i \in \mathcal{N}} I(y_i = j), \tag{11}$$

  and assigns x to the class j with the highest probability.

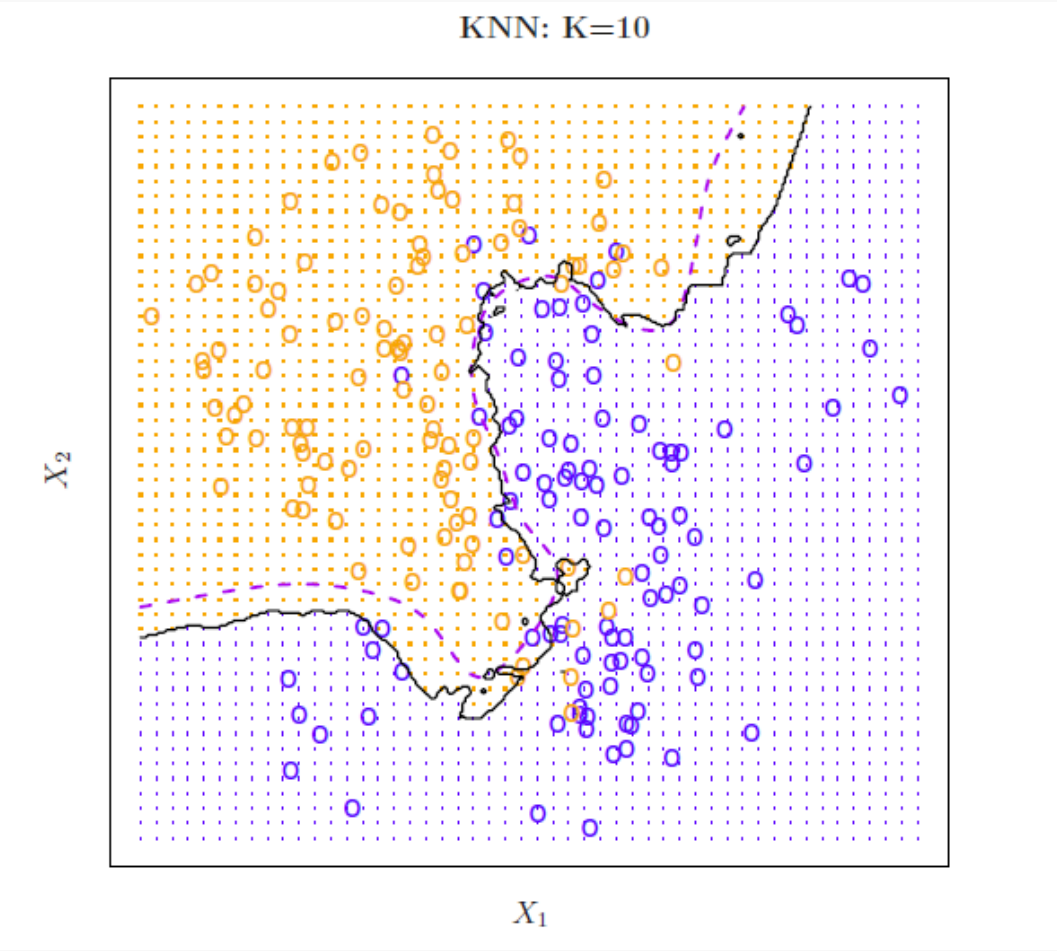

Figure 12: Image by James et al. (2021). The KNN decision boundary for $K = 10$ (in black), and the Bayes decision boundary (in purple).

KNN – iii) Different K

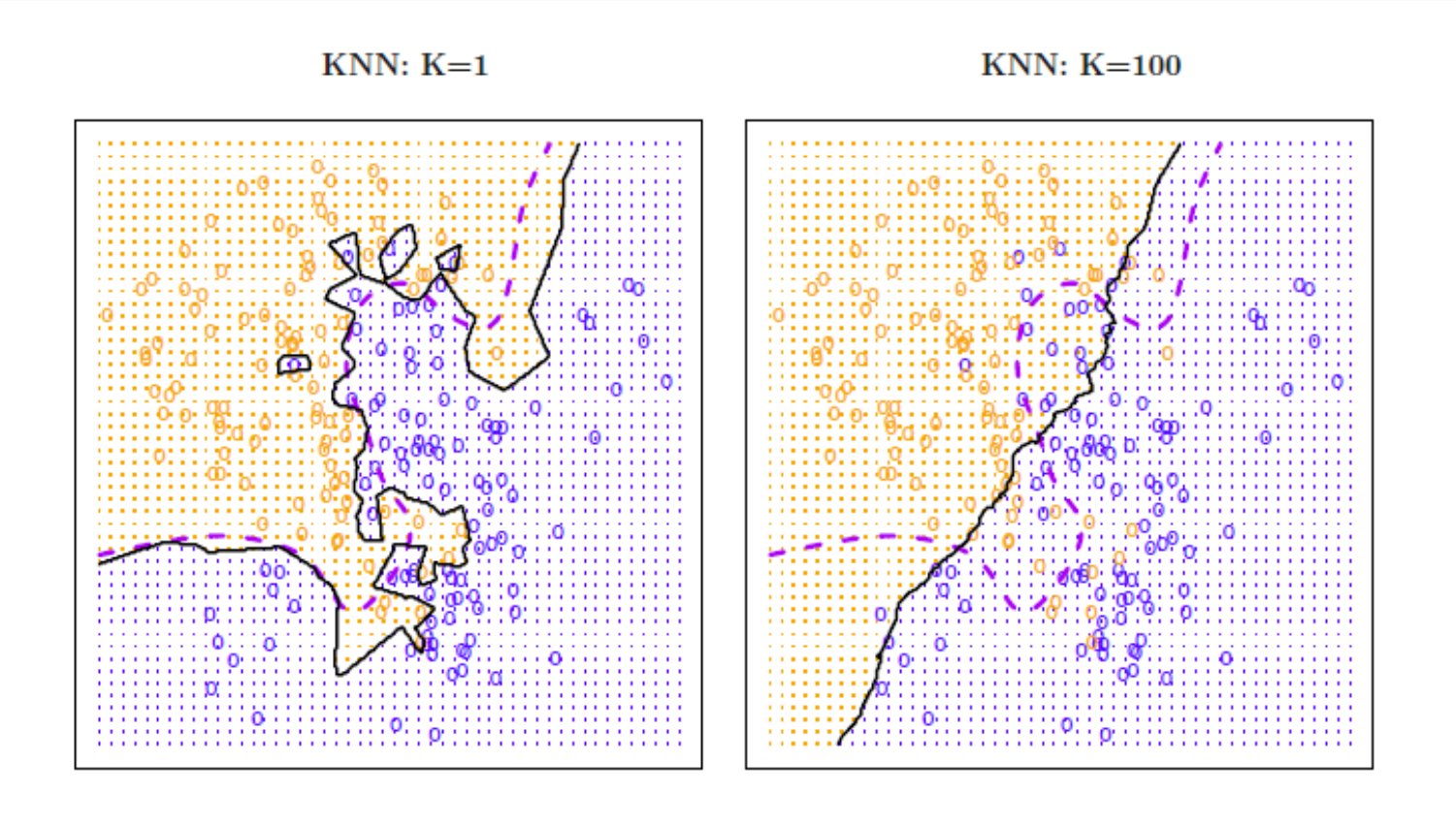

Figure 13: Image by James et al. (2021). KNN decision boundaries (in black) for $K = 1$ and $K = 100$, and the Bayes decision boundary (in purple). With $K = 1$, the decision boundary is overly flexible, while with $K = 100$ it is not sufficiently flexible.

KNN – iv) Bias-variance trade-off

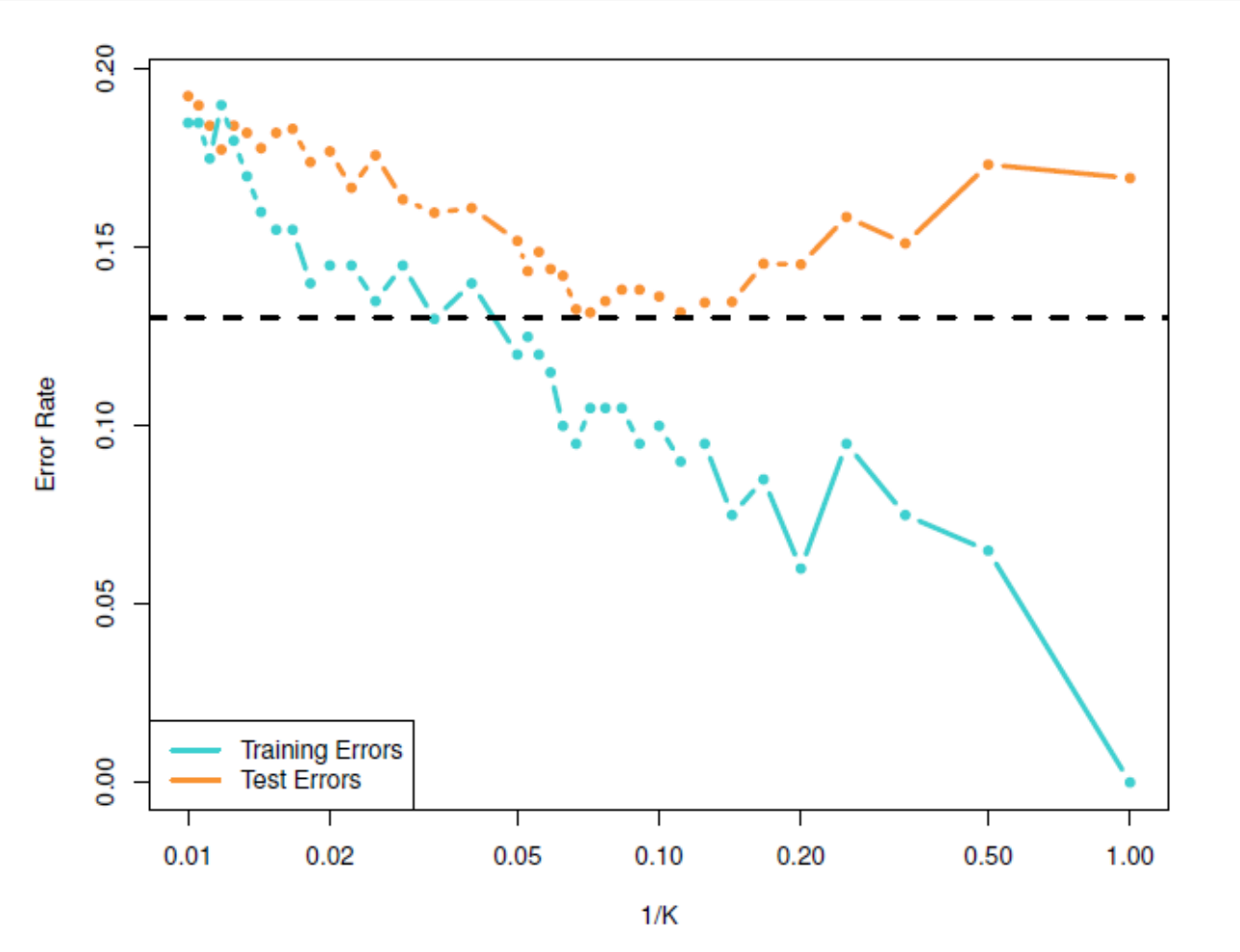

Figure 14: Image by James et al. (2021). The KNN training error rate (in blue) and test error rate (in orange) and the Bayes error rate (in black). The jumpiness of the curves is due to the small size of the training data set.

Section 3: Main concepts of statistical learning

Unsupervised learning

Recall: learn relationship or structure among observations. Example tasks:

- *Dimension reduction*: derive a low-dimensional set of features from higher-dimensional observations $X_1, \ldots, X_n$.
  - Uses: plotting 2-d representations of higher-dimensional data, regression.
  - *Principal components analysis* is a popular approach.
- *Cluster analysis*: partition observations $X_1, \ldots, X_n$ into distinct groups.

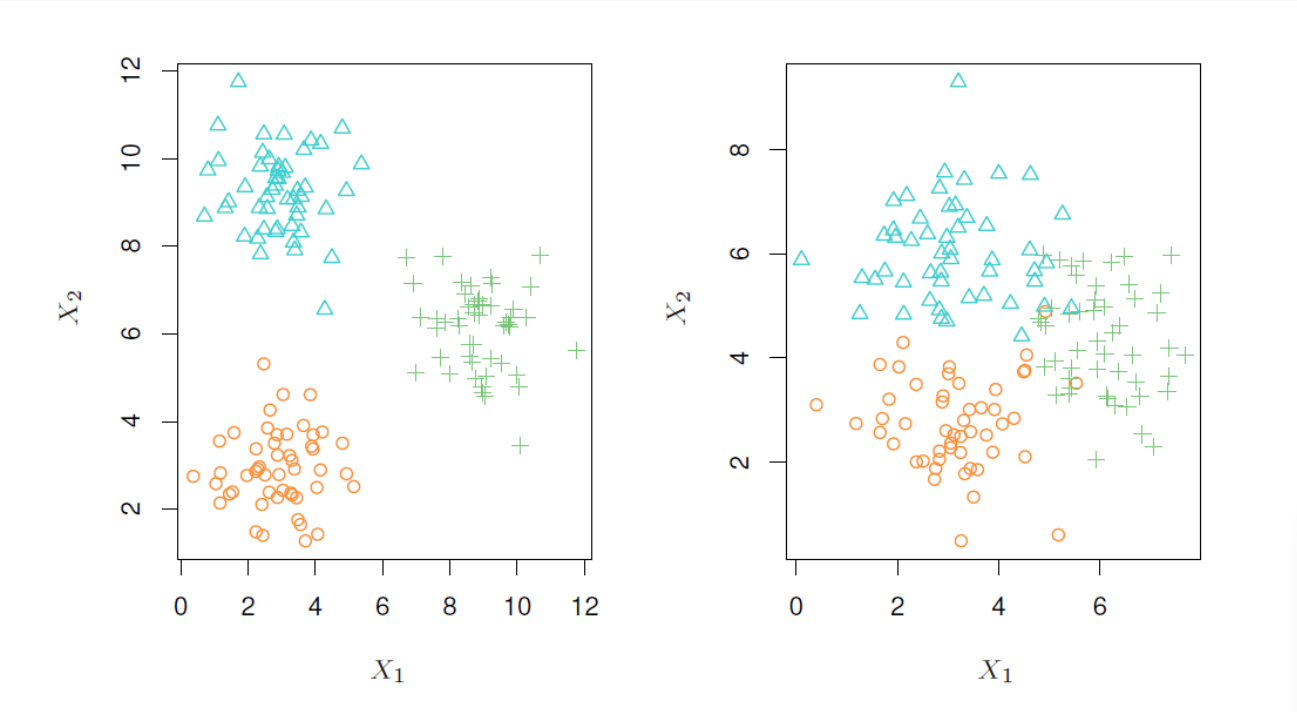

Figure 15: Image by James et al. (2021). Clustering in a data set involving three groups.