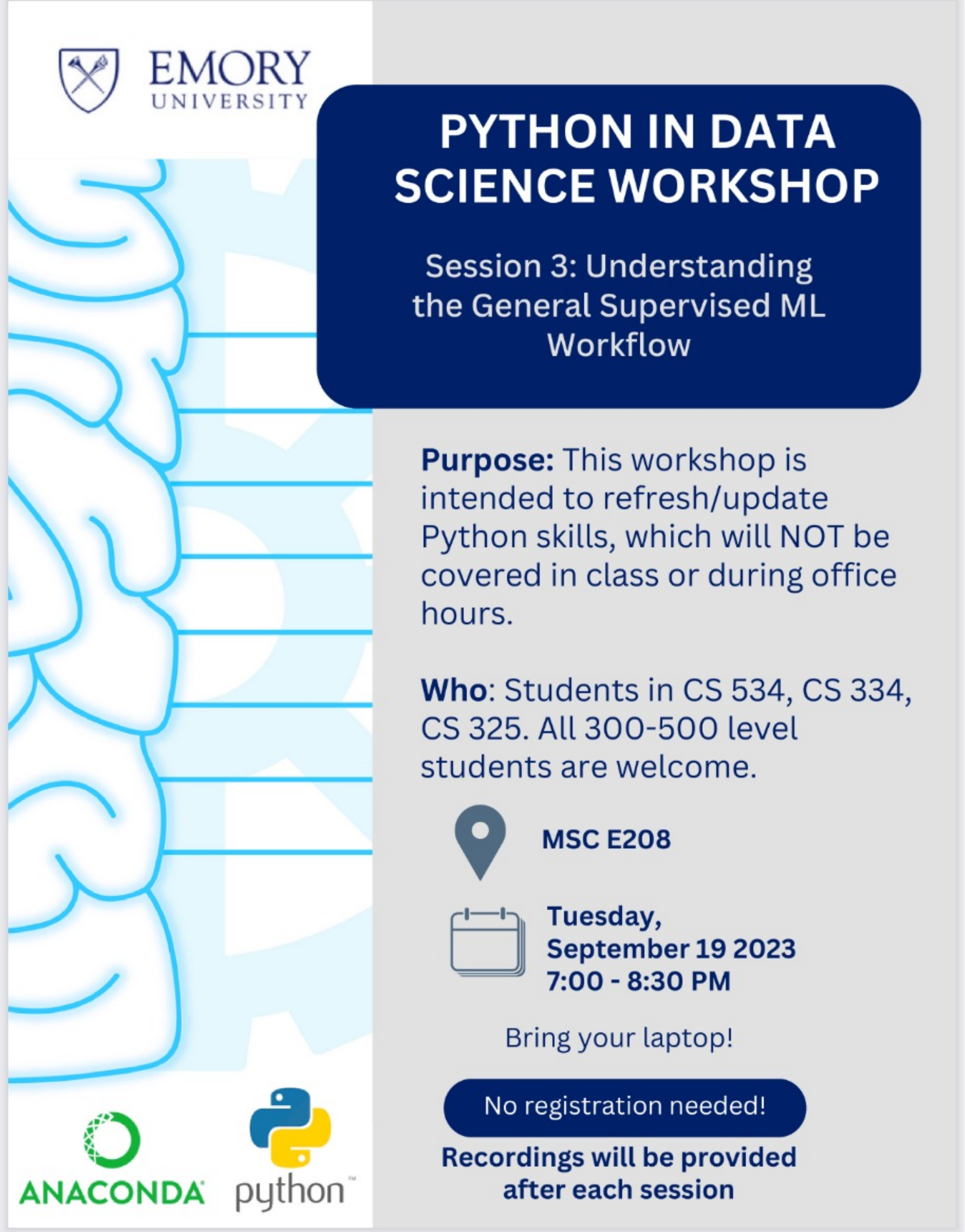
PYTHON WORKSHOP SESSION 3

- Session 3: Supervised ML workflow
  - Scikit-learn
  - Data preprocessing
  - Model assessment/selection
  - Linear regression model



HOMEWORK #2

- Out 9/13, Due 9/29 @ 11:59 PM ET
- 3 questions
  - Q1: Decision tree implementation
  - Q2: Model assessment
  - Q3: Model selection and robustness of k-nn and decision tree

DECISION TREE IMPLEMENTATION FAQ

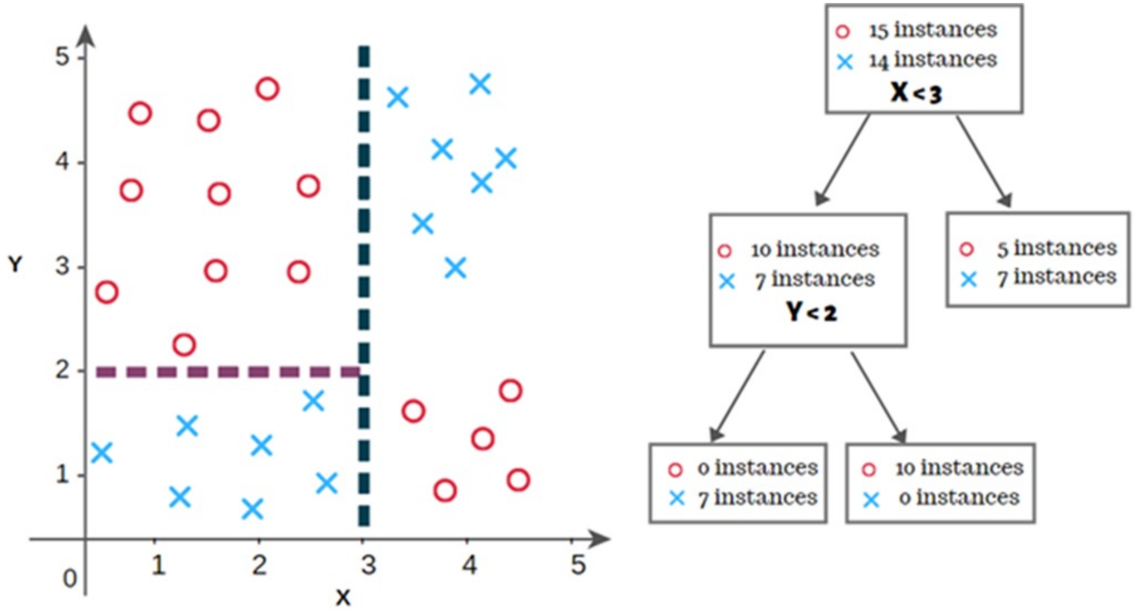
- Tree implementation in Python
- Stopping criteria – maximum depth
- Stopping criteria – minimum leaf samples
- Entropy vs. information gain

DECISION TREE: TRAINING (C4.5 ALGORITHM)

Algorithm 1.1 C4.5(D)

Input: an attribute-valued dataset D

Tree = {}
if D is "pure" OR other stopping criteria met **then**
    terminate
end if
for all attribute $a \in D$ **do**
    Compute information-theoretic criteria if we split on a
end for
a_{best} = Best attribute according to above computed criteria
Tree = Create a decision node that tests a_{best} in the root
D_v = Induced sub-datasets from D based on a_{best}
for all D_v **do**
    $\text{Tree}_v = \text{C4.5}(D_v)$
    Attach Tree_v to the corresponding branch of Tree
end for
return Tree

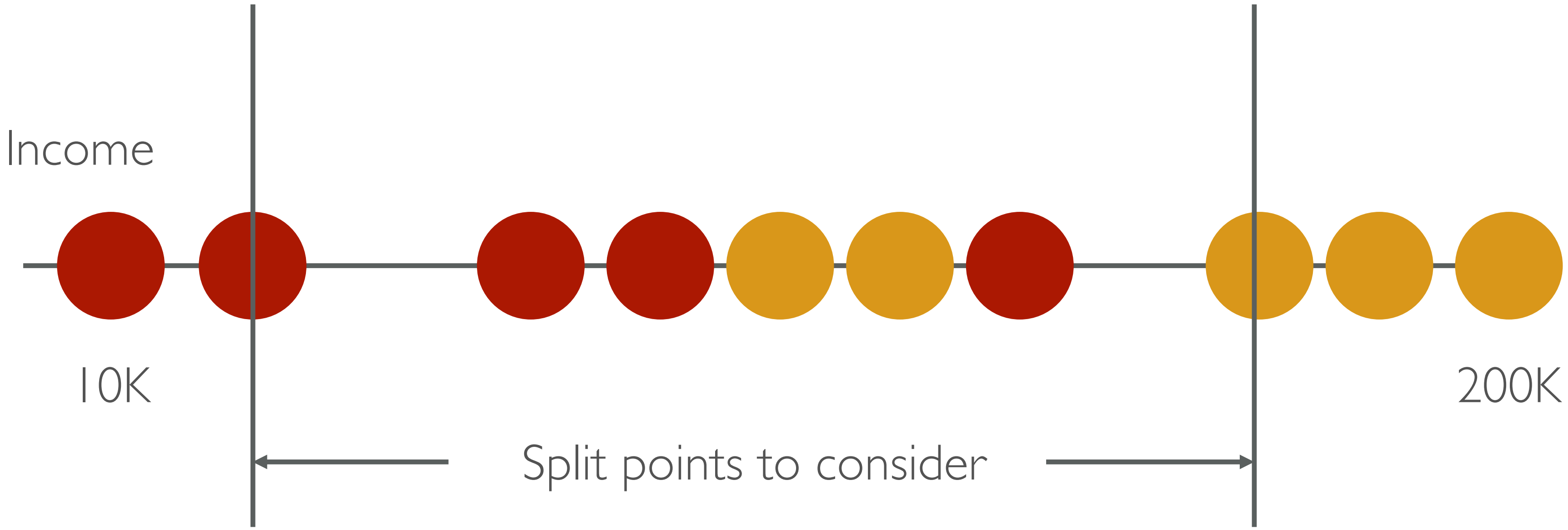


TREE IMPLEMENTATION IN PYTHON

```
class DecisionTree(object):

    # define some variable to hold the tree model

    def decision_tree(self, xFeat, y, depth):

        # Check stopping criteria (e.g. maximum depth), if it is met, return majority class of y

        # Find the split: enumerate all possible splits (for each feature and each split value), compute the score
        (entropy or gini) for each split, find the best split feature and split value

        # Partition data using the split feature and split value into two sets: xFeatL, xFeatR, yL, yR

        # Recursive call of decision_tree()

        return {"left": self.decision_tree (xFeatL, yL, depth+1),
                "right": self.decision_tree(xFeatR, yR,  depth+1),
                #other key information including split variable and split value}
```

MINIMUM LEAF SAMPLES: IMPLEMENTATION

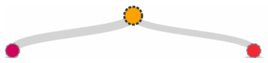
Income <= t*   Minimum leaf samples = 2

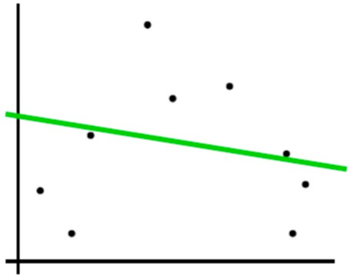


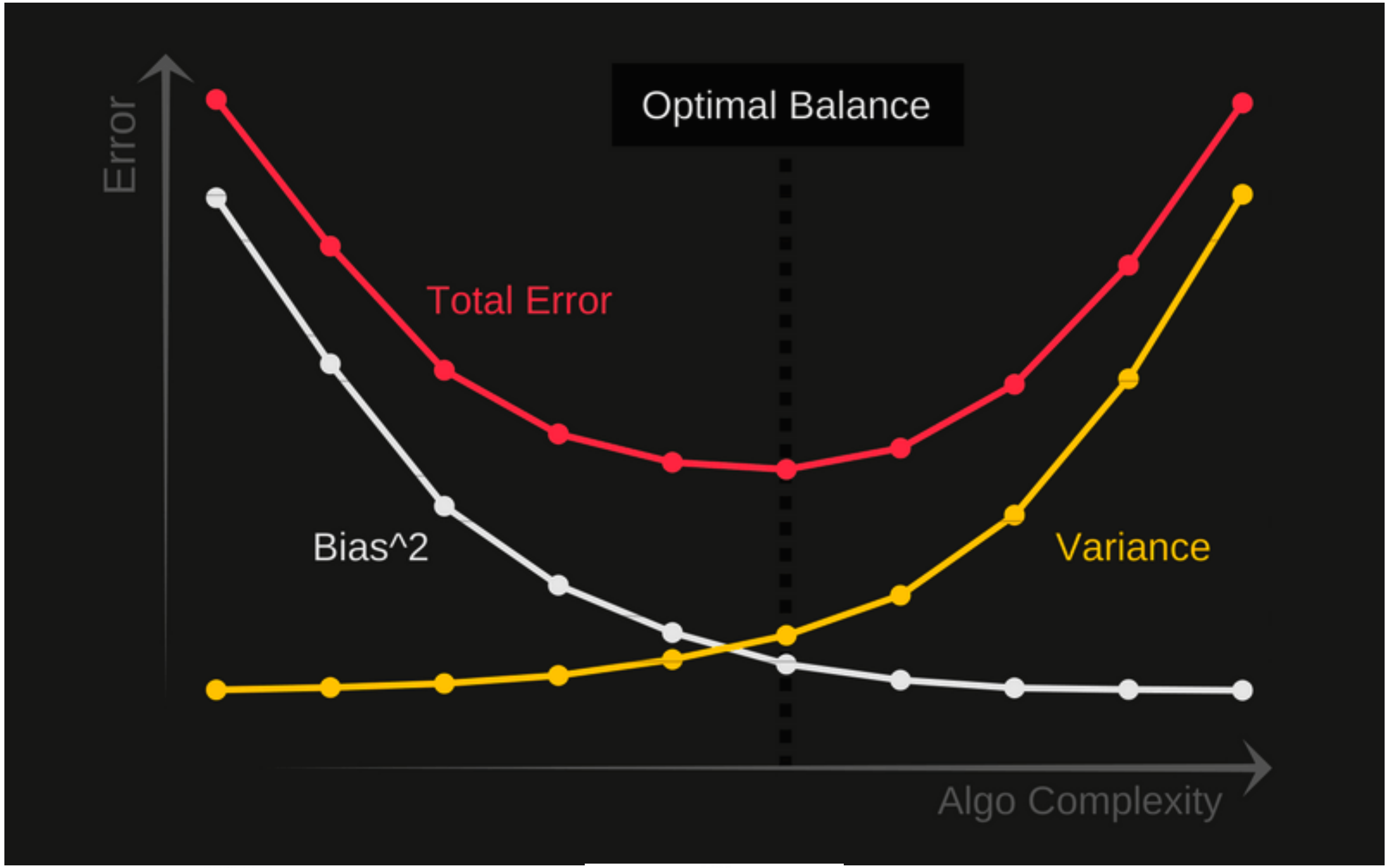
BIAS AND VARIANCE TRADEOFF (CONT.)

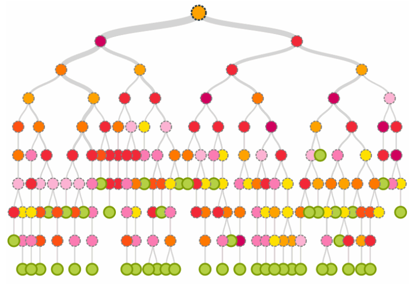
CS 334: Machine Learning

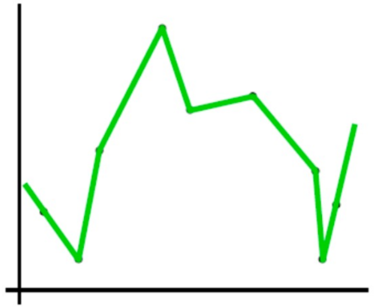
Review: Total Error = Bias^2 + Variance + Irreducible Error

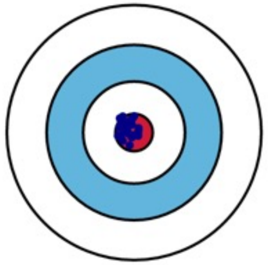


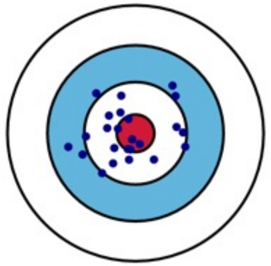












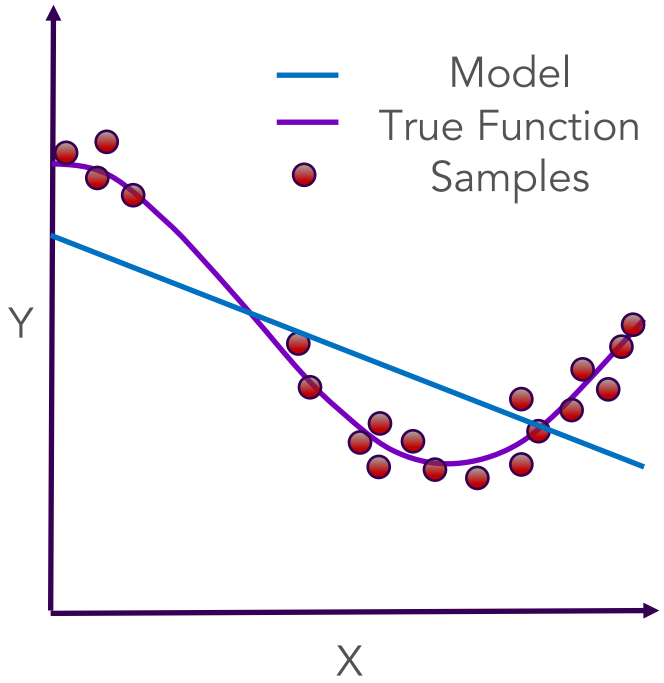
://elitedatascience.com/bias-variance-tradeoff

GENERALIZATION & OVERFITTING

- Generalization — model performance of a model on independent / future unseen data (data not used in training)
- Underfitting – model is unable to capture the relationship between the input and output variables accurately; high error on both training and test data
- Overfitting — model is specific to the training set and is learning the noise from the data instead of generalizable rule; low error on training but high error on test data

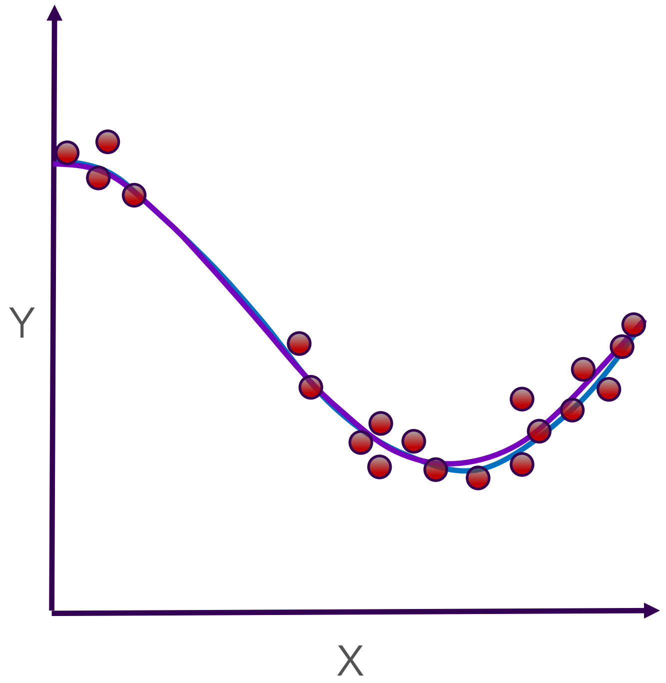
MODEL GENERALIZATION

Polynomial Degree = 1



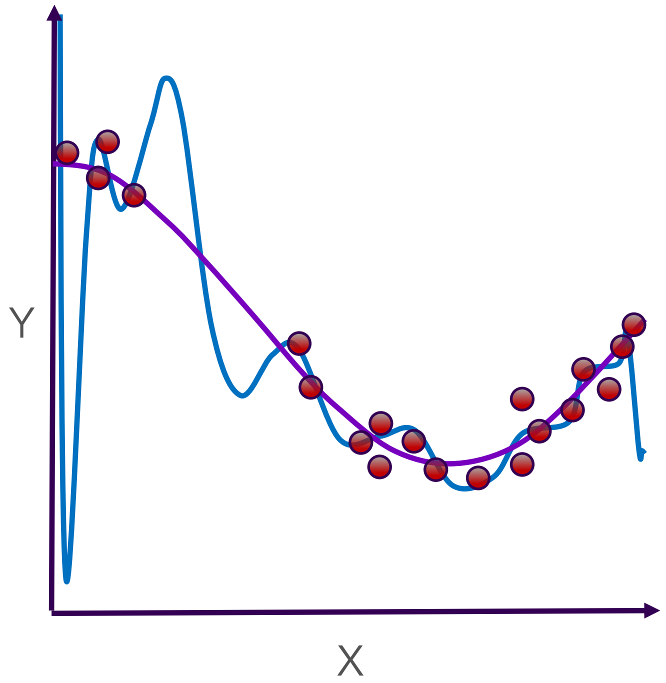
Poor on Training Set
Poor at Predicting

Polynomial Degree = 4



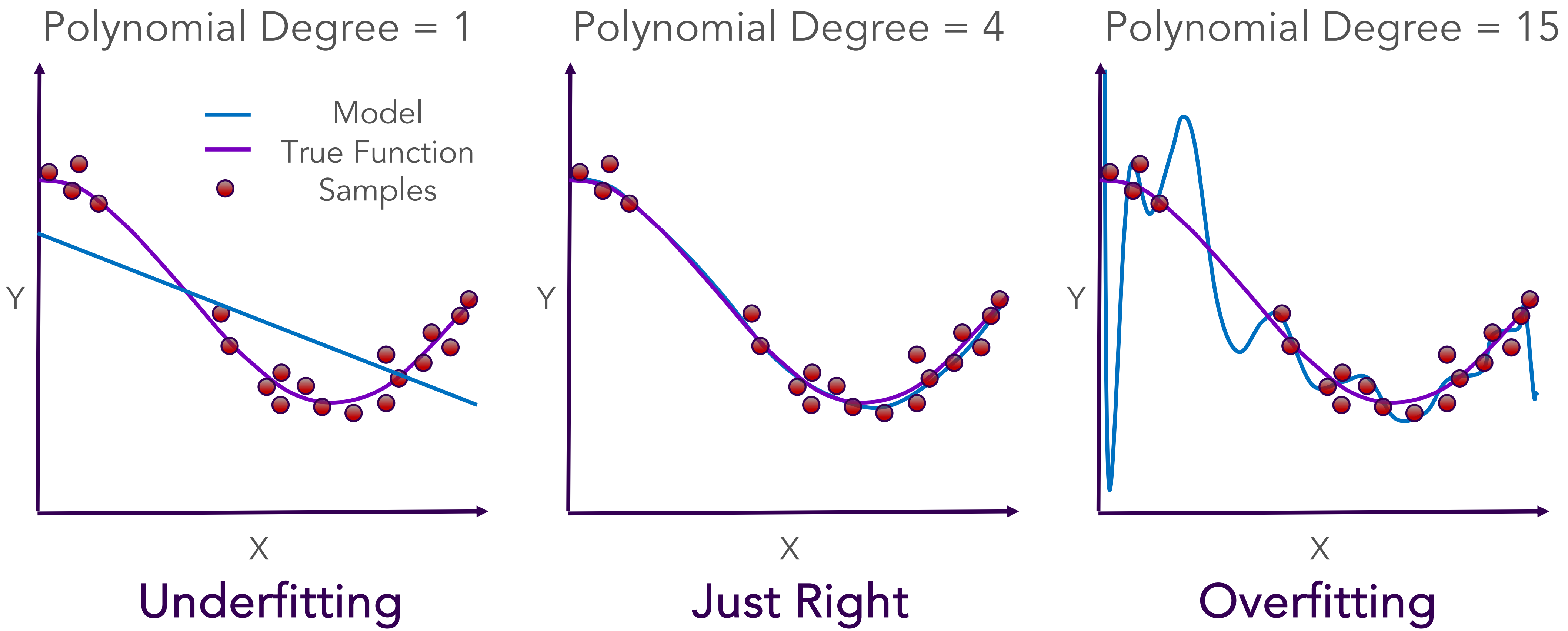
Generalizable

Polynomial Degree = 15

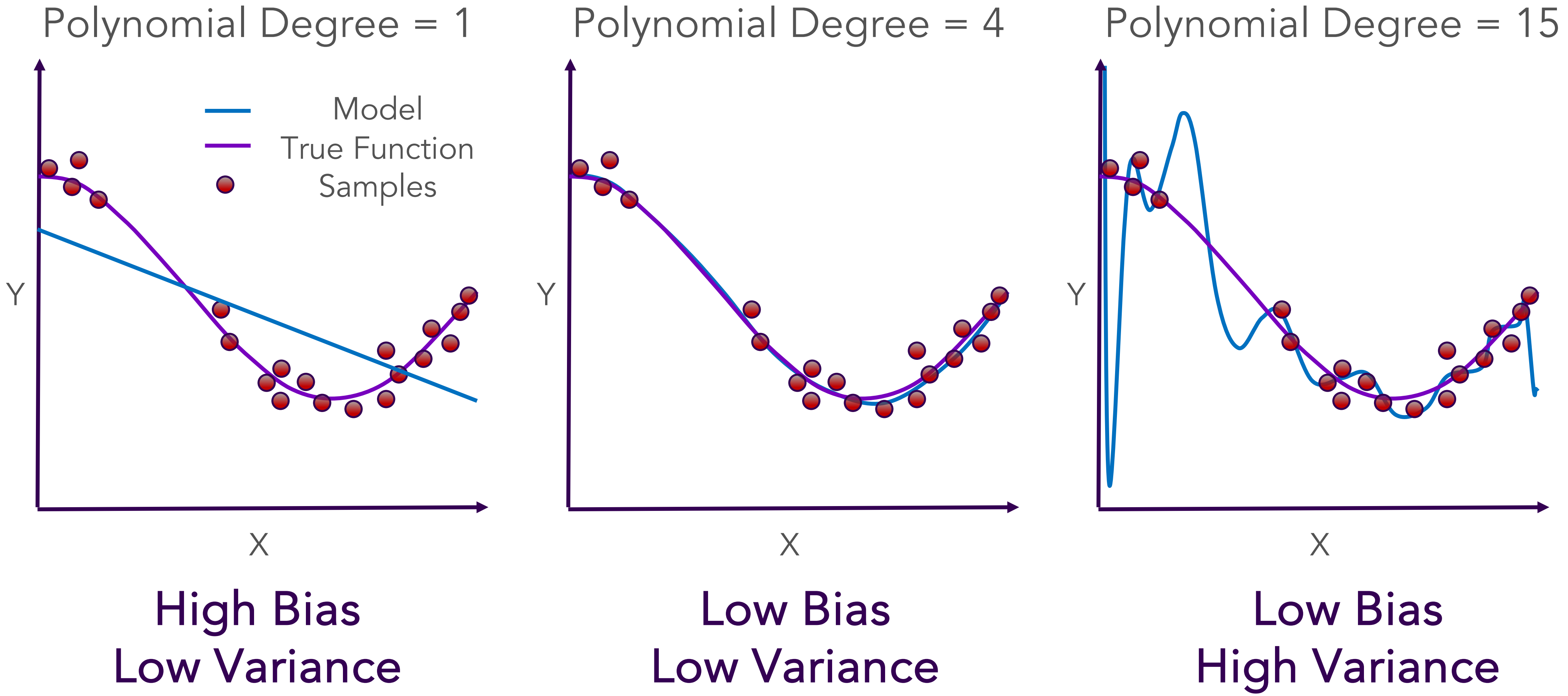


Very Good on Training Set
Poor at Predicting

UNDERFITTING VS OVERFITTING



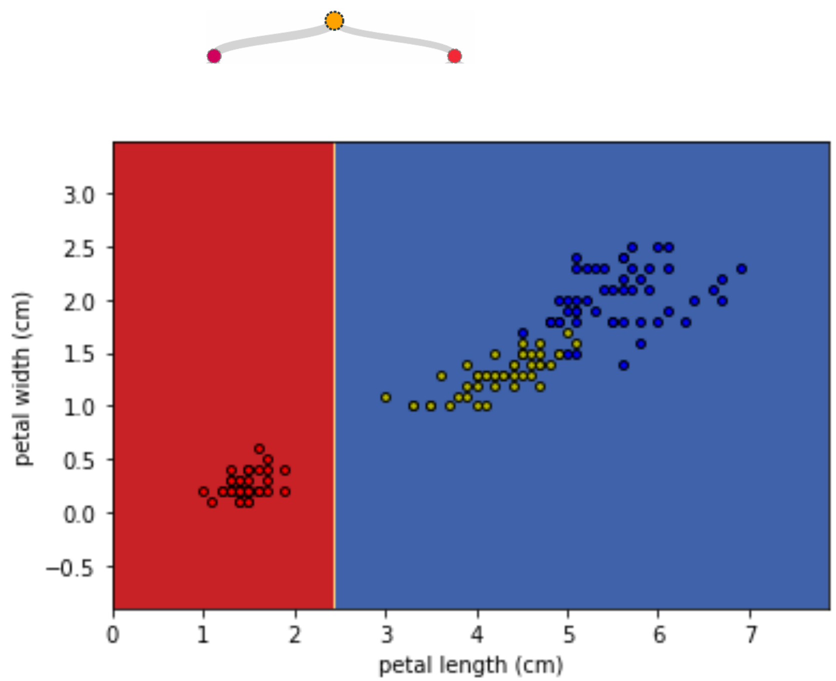
BIAS-VARIANCE TRADE-OFF



BIAS ANALYSIS: SOURCES

- Inability to represent certain decision boundaries

- Classifiers are "too global" (e.g., single linear separator)



Decision tree (max depth = 1)

High bias —> underfitting

How to reduce bias?

BIAS ANALYSIS: REDUCTION

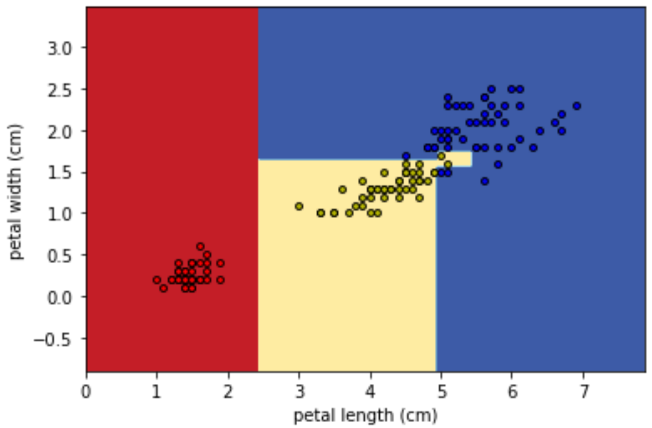
- More complex models
- More features

VARIANCE ANALYSIS: SOURCES

- Noise in labels or features
- Training data too small
- "Too local" algorithms that easily fit data
- Randomness in learning algorithm (i.e., non-convex algorithms)

High variance —> overfitting

How to reduce variance?

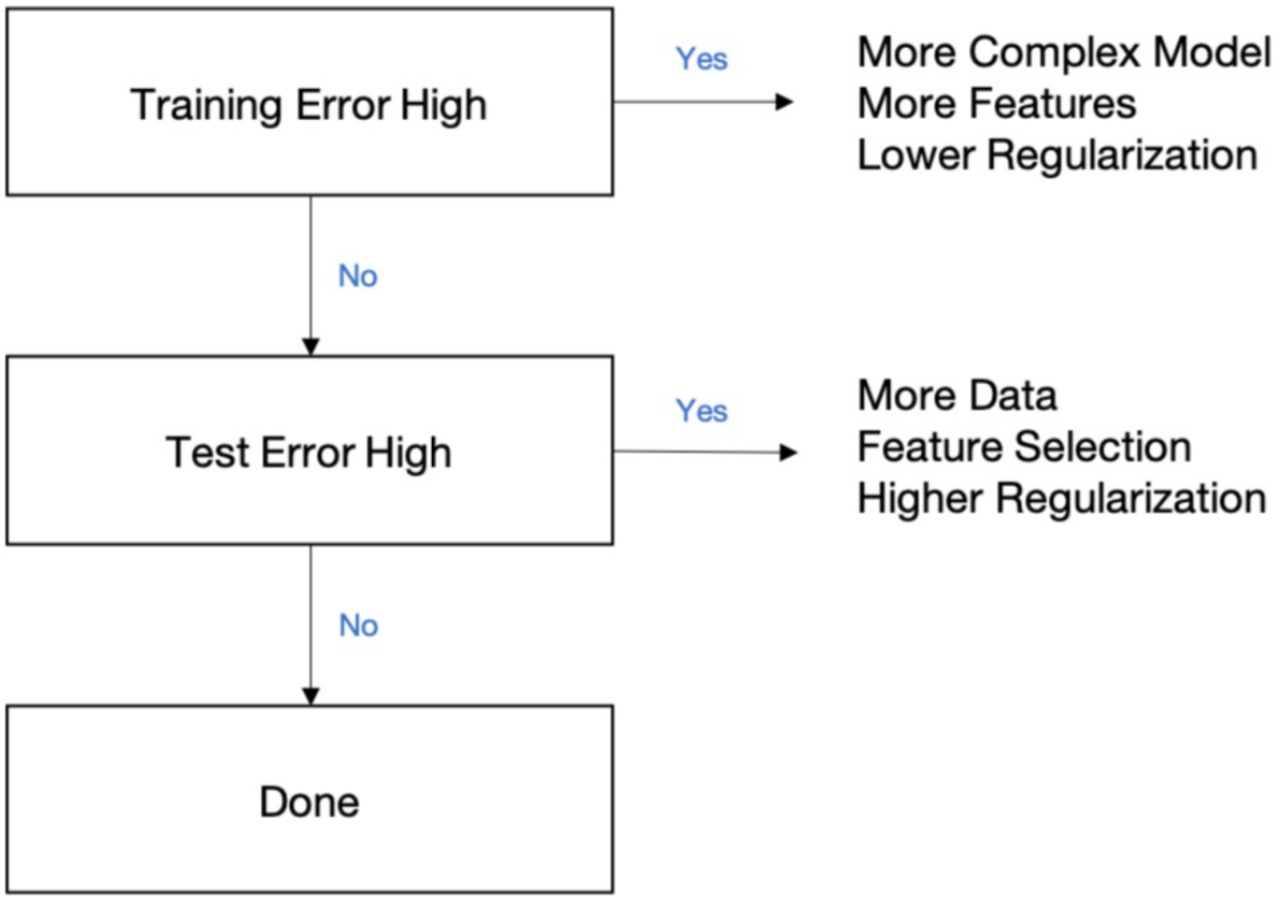


Decision tree (max depth = 3)

VARIANCE ANALYSIS: REDUCTION

- Use more data (increase size of training data)
- Less complex models
- Fewer features (feature selection)

HOW TO USE BIAS-VARIANCE



GROUP ACTIVITY

EXERCISE: BIAS AND VARIANCE TRADEOFF

What happens to bias and variance when we

1. Increase k for kNN classifier
2. Only consider a subset of features in kNN classifier
3. Increase maximum tree depth for learning decision tree
4. Increase minimum leaf samples for learning decision tree
5. Consider only a (random) subset of features at each node for learning decision tree
6. Increase alpha for post-pruning decision tree

$$C_\alpha(T) = \sum_{j=1}^{|T|} [1 - \hat{p}_{g_j}(R_j)] + \alpha|T|$$

LINEAR REGRESSION

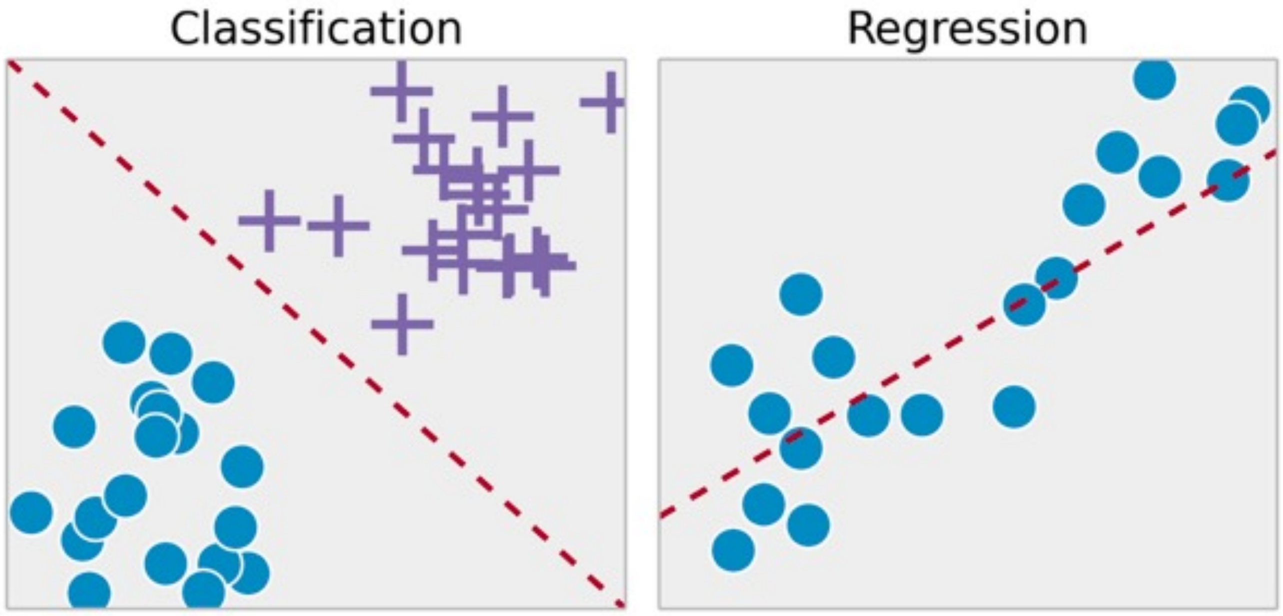
CS 334: Machine Learning

Slides adapted from Joyce Ho, Lee Cooper, Joydeep Ghosh, Carlos Carvalho, and Ryan Tibshirani

REVIEW: PREDICTION TASKS

- **Classification**: Predicting *qualitative* targets (values in a finite set)

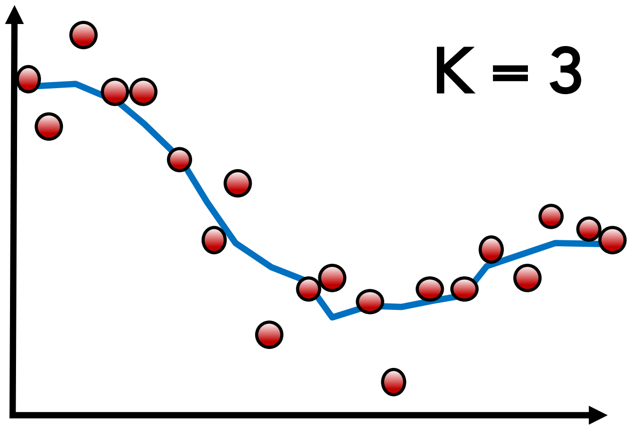
- **Regression**: Predicting *quantitative* responses (continuous valued, natural ordering)

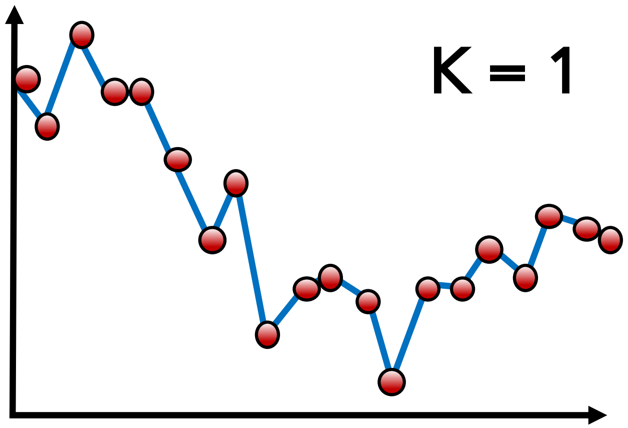


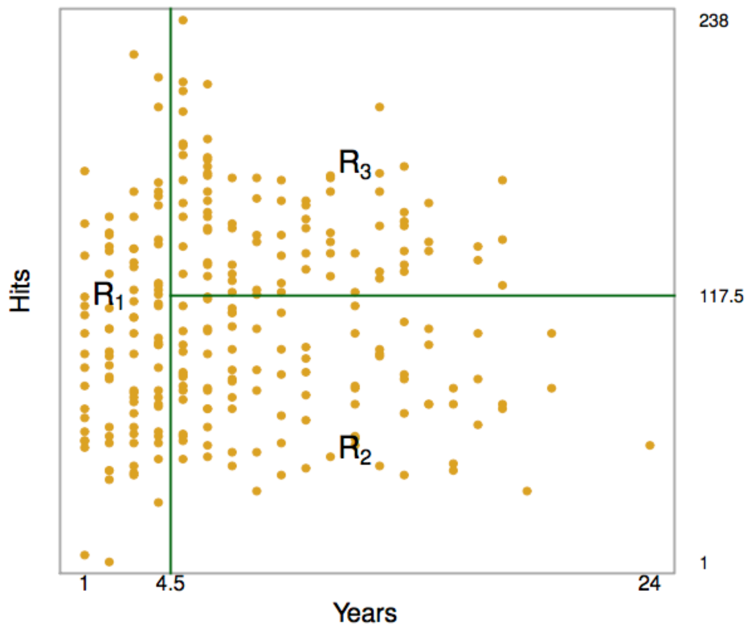
REGRESSION: EXAMPLES

- Straight prediction questions:
  - How many games will the Atlanta United win?
  - Will you like Star Wars: The Last Jedi?
- Explanation & understanding:
  - What is the impact of an MBA on income?
  - Does Walmart pay women less in salary?

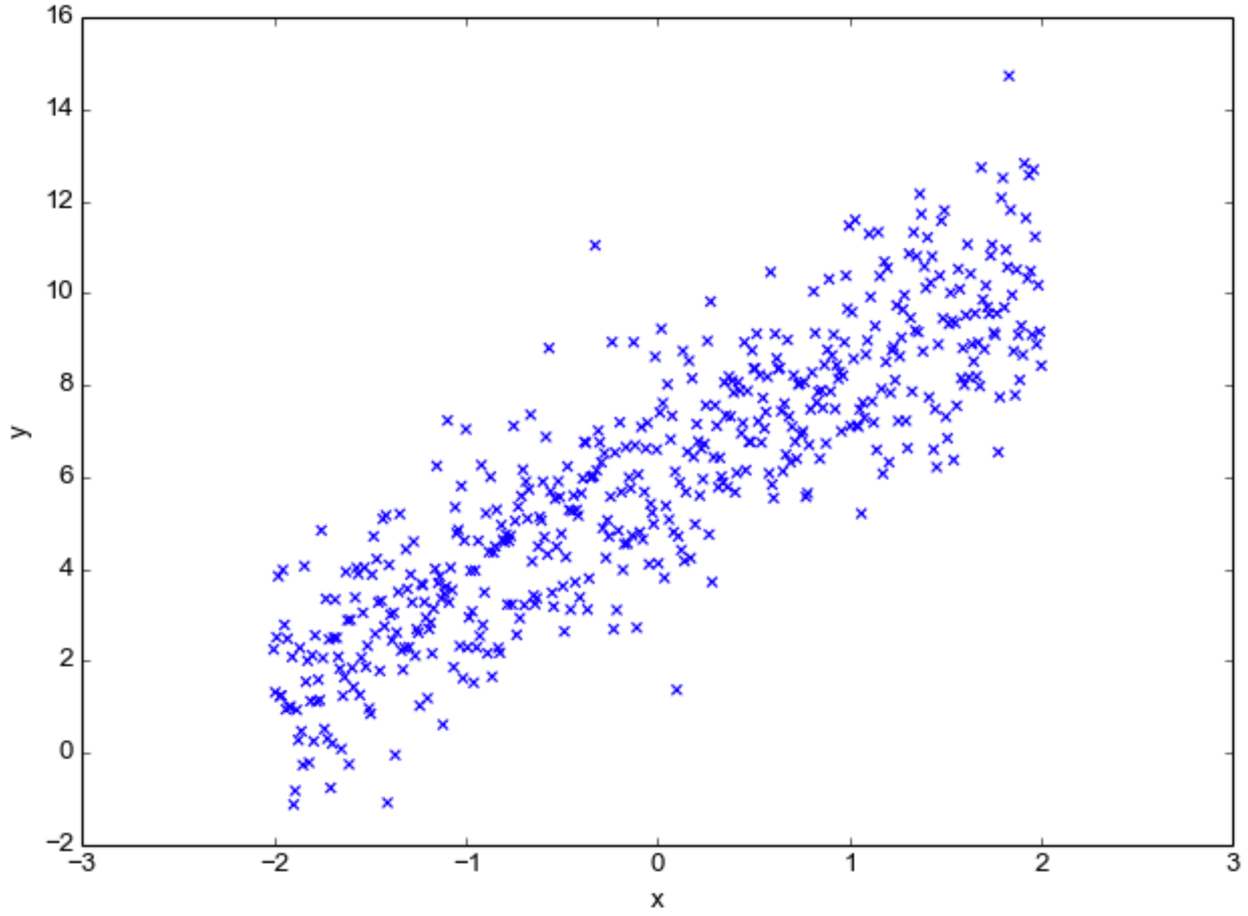
REVIEW: REGRESSION W/ KNN AND DECISION TREE



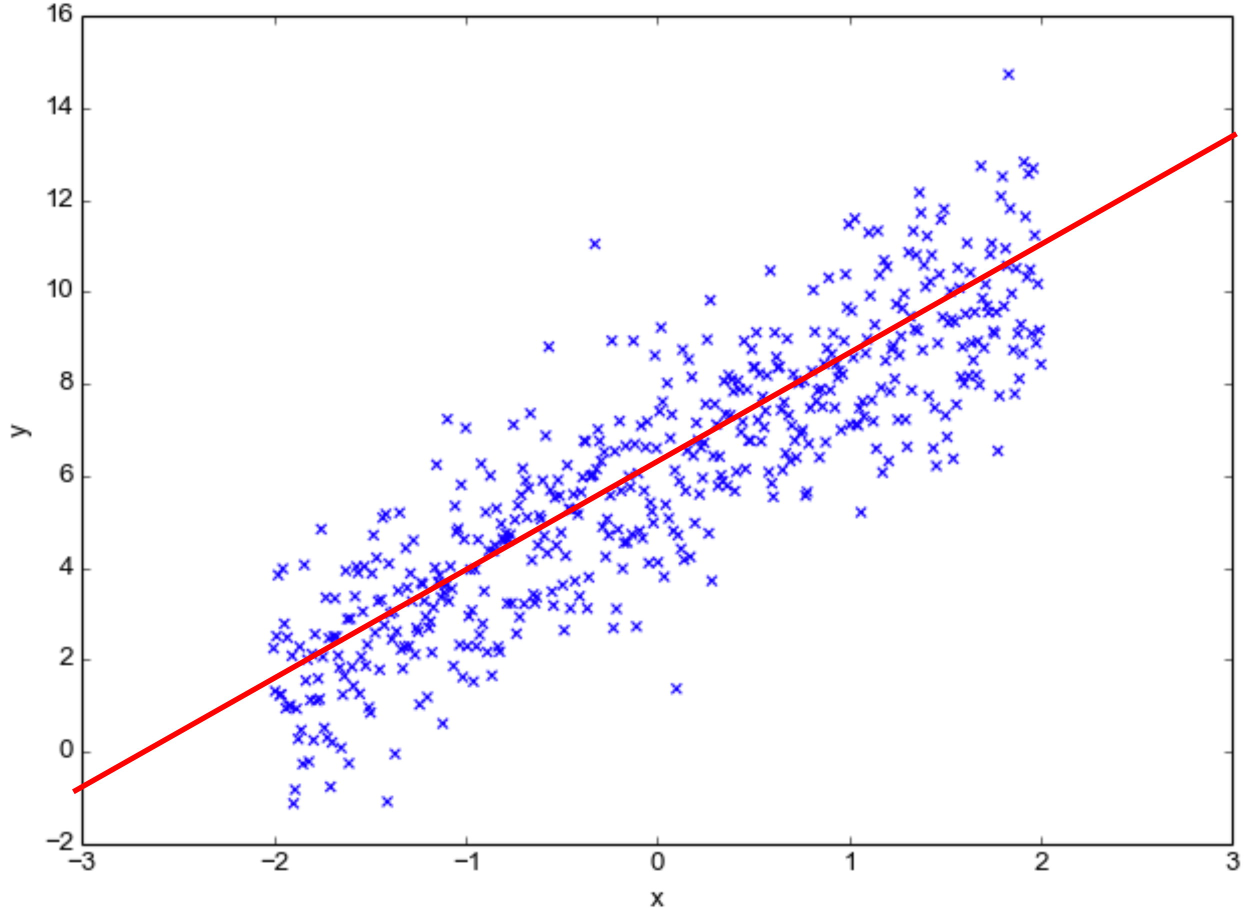




HOW TO PREDICT Y BASED ON X?



HOW TO PREDICT Y BASED ON X?



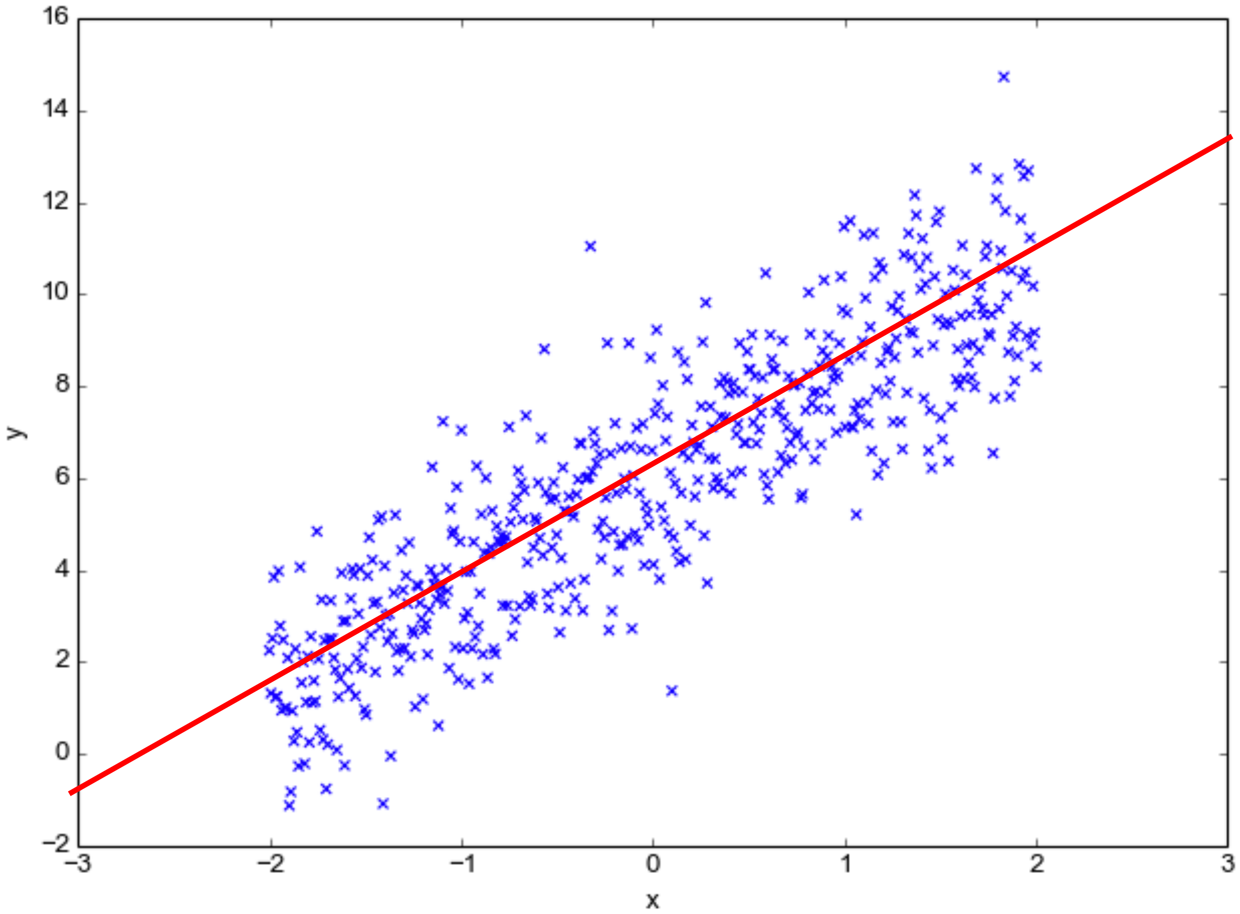
LINEAR REGRESSION: OVERVIEW

- Assumes there is approximately a linear relationship between the predictor variables and the outcome of interest
- Models the linear relationship in form of mathematical equation (parametric)
- Most widely used statistical tool (“workhorse”) for understanding relationships amongst variables

SIMPLE LINEAR REGRESSION

- Use a linear function to model the relationship between a dependent (target) variable Y and predictor variable X

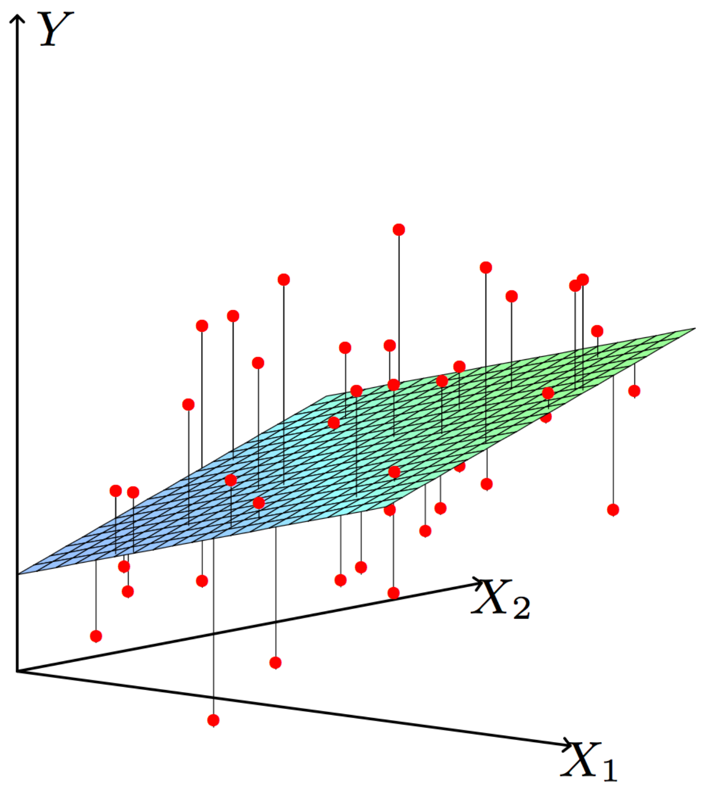
$$Y \approx \beta_0 + \beta_1 X$$



MULTIPLE LINEAR REGRESSION (MLR)

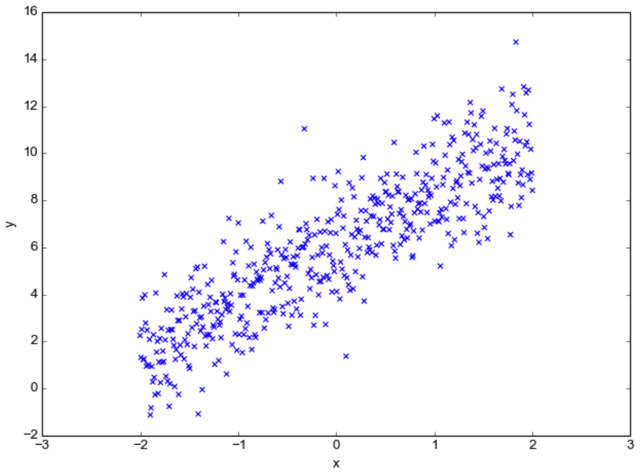
- Use a linear function to model the relationship between a dependent (target) variable Y and a vector of multiple predictor variables $\mathbf{x}^T = (x_1, x_2, \ldots, x_p)$

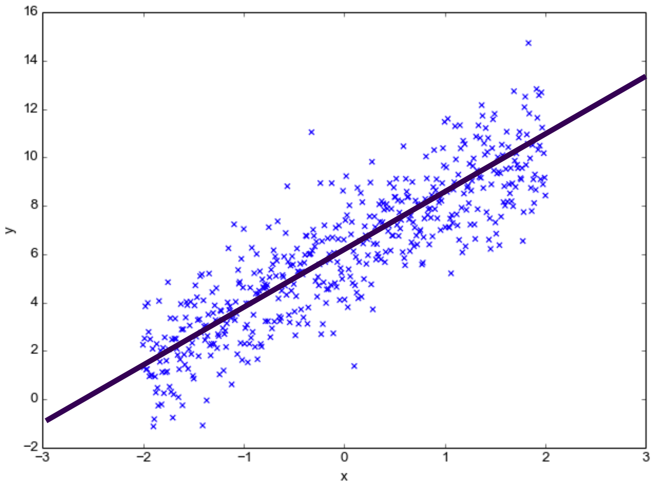
$$f(\mathbf{x}) = \beta_0 + \sum_{i=1}^{p} x_i \beta_i$$



LINEAR REGRESSION

- Training:

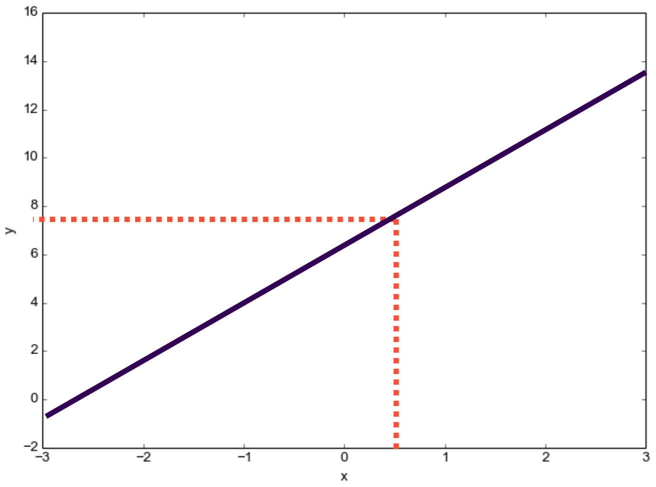




$$Y \approx \beta_0 + \beta_1 X$$

estimates $\hat{\beta}_0$ and $\hat{\beta}_1$

- Prediction: x

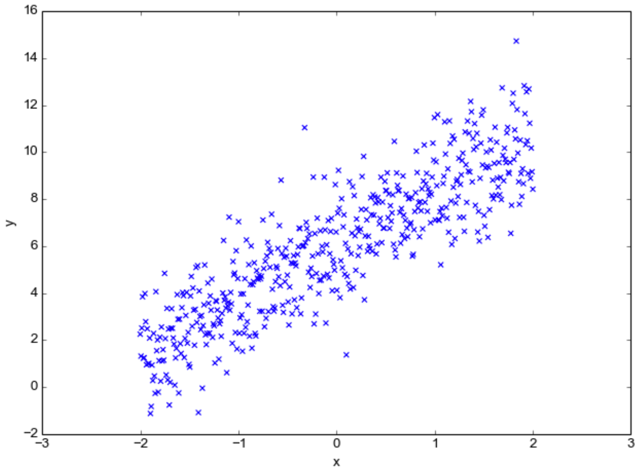


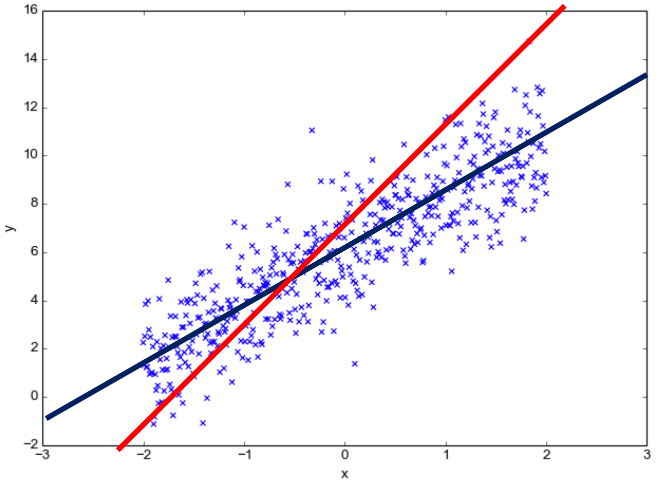
$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$$

GROUP ACTIVITY

LINEAR REGRESSION: TRAINING

- Training:





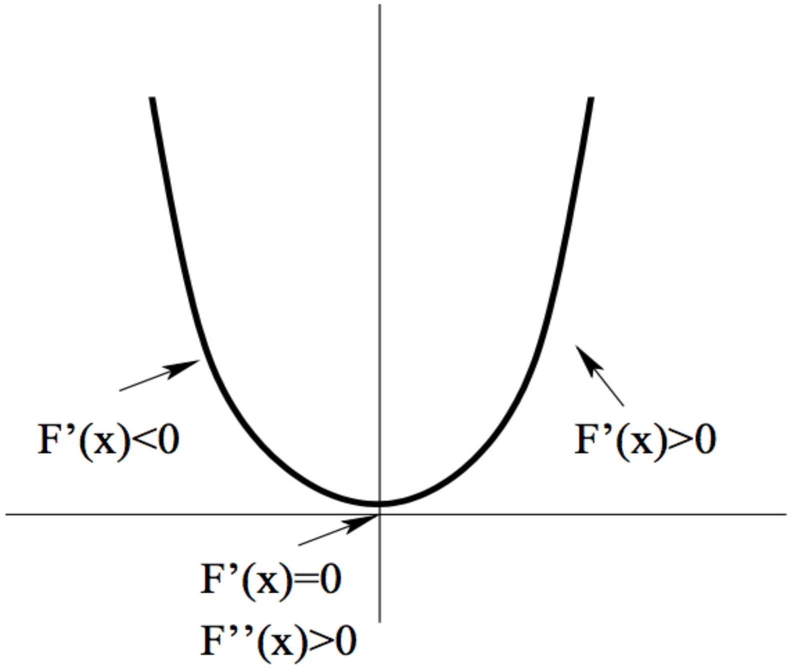
$$Y \approx \beta_0 + \beta_1 X$$

estimates $\hat{\beta}_0$ and $\hat{\beta}_1$

Which one is better? How to choose the "best" coefficients?

LEARNING THE PARAMETERS

- Closed form (direct solution): set partial derivatives to zero and solve parameters
- Iterative algorithms: Gradient descent (GD) and Stochastic gradient descent (SGD) (later)



Partial derivatives: https://www.khanacademy.org/math/multivariable-calculus

REVIEW: DERIVATIVE RULES

Common Functions	Function	Derivative
Constant	c	0
Line	x	1
	ax	a
Square	x^2	2x
Square Root	√x	$(½)x^{-½}$
Exponential	e^x	e^x
	a^x	$\ln(a)\ a^x$
Logarithms	ln(x)	1/x
	$\log_a(x)$	1 / (x ln(a))
Trigonometry (x is in radians)	sin(x)	cos(x)
	cos(x)	−sin(x)
	tan(x)	$\sec^2(x)$
Inverse Trigonometry	$\sin^{-1}(x)$	$1/\sqrt{(1-x^2)}$
	$\cos^{-1}(x)$	$-1/\sqrt{(1-x^2)}$
	$\tan^{-1}(x)$	$1/(1+x^2)$

Rules	Function	Derivative
Multiplication by constant	cf	cf′
Power Rule	x^n	nx^{n-1}
Sum Rule	f + g	f′ + g′
Difference Rule	f - g	f′ − g′
Product Rule	fg	f g′ + f′ g
Quotient Rule	f/g	$(f' g - g' f)/g^2$
Reciprocal Rule	1/f	$-f'/f^2$
Chain Rule (as "Composition of Functions")	f º g	(f′ º g) × g′
Chain Rule (using ′)	f(g(x))	f′(g(x))g′(x)
Chain Rule (using $\frac{d}{dx}$)	$\frac{dy}{dx} = \frac{dy}{du}\frac{du}{dx}$	

DIRECTION SOLUTION: SIMPLE LINEAR REGRESSION

- Find β_0 and β_1 that minimizes squared residual sum of residuals (SSR)

$$
\begin{aligned}
SSR &= \sum_{i=1}^{n}(y_i - (\beta_0 + \beta_1 x_i))^2 \\
&= \sum_{i=1}^{n}\left(y_i^2 - 2y_i(\beta_0 + \beta_1 x_i) + \beta_0^2 + 2\beta_0\beta_1 x_i + \beta_1^2 x_i^2\right)
\end{aligned}
$$

- Solve β_0 by setting partial derivative with respect to β_0 to 0

$$
\begin{aligned}
\frac{\partial SSR}{\partial \beta_0} &= \sum_{i=1}^{n}(-2y_i + 2\beta_0 + 2\beta_1 x_i) \\
0 &= \sum_{i=1}^{n}\left(-y_i + \hat{\beta}_0 + \hat{\beta}_1 x_i\right) \\
0 &= -n\bar{y} + n\hat{\beta}_0 + \hat{\beta}_1 n\bar{x} \\
\hat{\beta}_0 &= \bar{y} - \hat{\beta}_1\bar{x}
\end{aligned}
$$

DIRECT SOLUTION: SIMPLE LINEAR REGRESSION

- Solve β_1 by setting partial derivative with respect to β_1 to 0

$$
\begin{aligned}
SSR &= \sum_{i=1}^{n}(y_i - (\beta_0 + \beta_1 x_i))^2 \\
&= \sum_{i=1}^{n}\left(y_i^2 - 2y_i(\beta_0 + \beta_1 x_i) + \beta_0^2 + 2\beta_0\beta_1 x_i + \beta_1^2 x_i^2\right)
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial SSR}{\partial \beta_1} &= \sum_{i=1}^{n}\left(-2x_i y_i + 2\beta_0 x_i + 2\beta_1 x_i^2\right) \\
0 &= -\sum_{i=1}^{n} x_i y_i + \hat{\beta}_0 \sum_{i=1}^{n} x_i + \hat{\beta}_1 \sum_{i=1}^{n} x_i^2 \\
0 &= -\sum_{i=1}^{n} x_i y_i + (\bar{y} - \hat{\beta}_1 \bar{x}) \sum_{i=1}^{n} x_i + \hat{\beta}_1 \sum_{i=1}^{n} x_i^2 \\
\hat{\beta}_1 &= \frac{\sum_{i=1}^{n} x_i(y_i - \bar{y})}{\sum_{i=1}^{n} x_i(x_i - \bar{x})}
\end{aligned}
$$

EXAMPLE

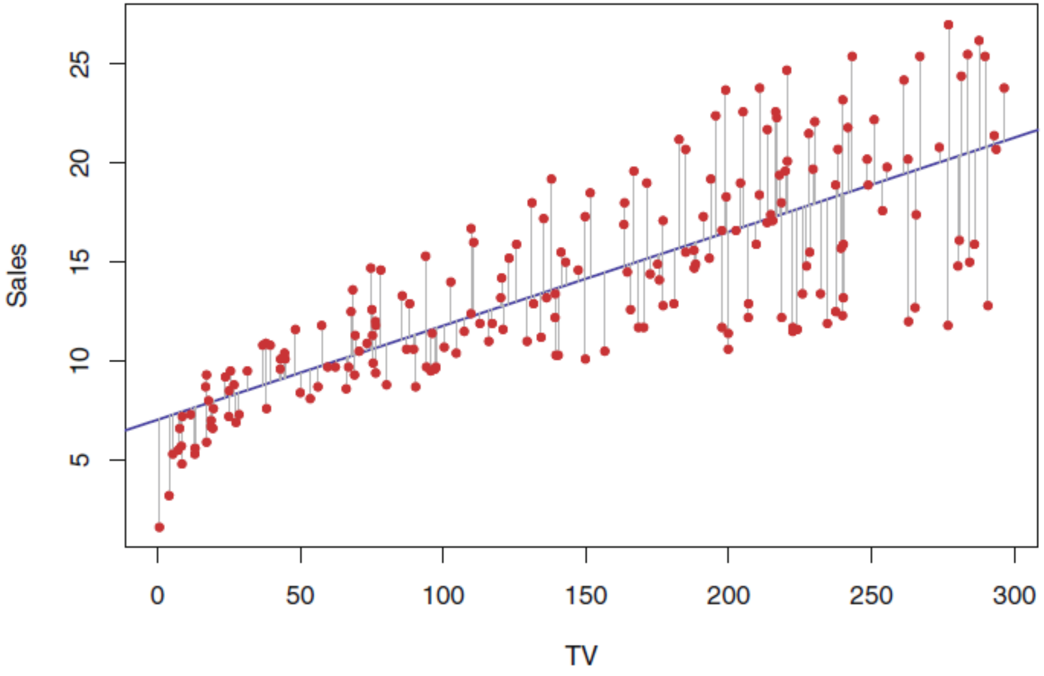


FIGURE 3.1. *For the* `Advertising` *data, the least squares fit for the regression*

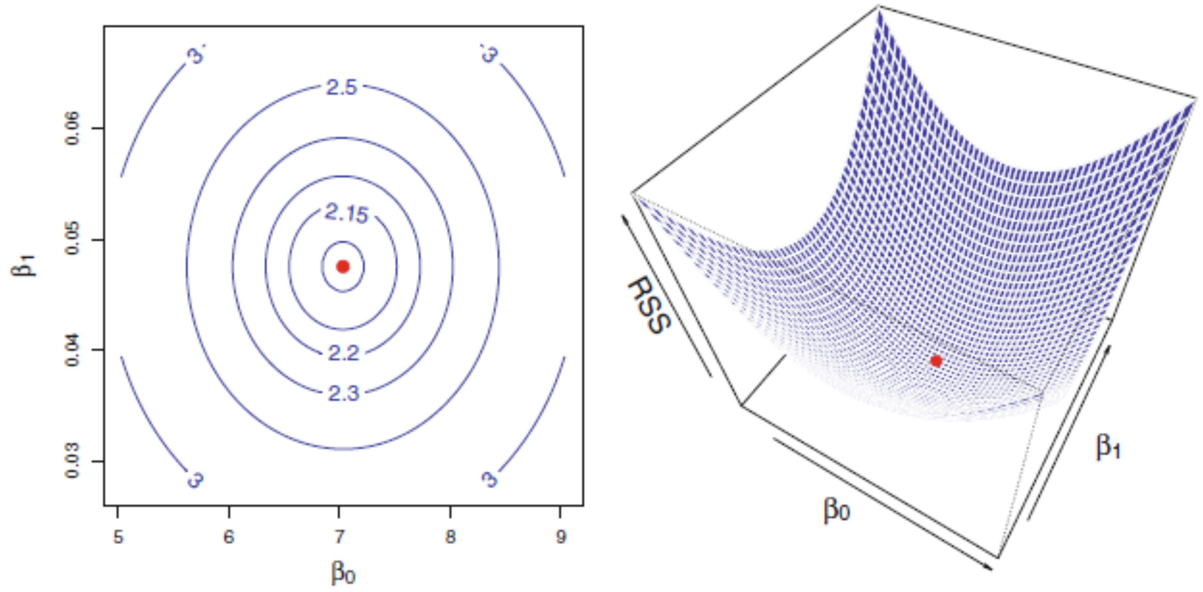


FIGURE 3.2. *Contour and three-dimensional plots of the RSS on the* `Advertising` *data, using* `sales` *as the response and* `TV` *as the predictor. The red dots correspond to the least squares estimates $\hat{\beta}_0$ and $\hat{\beta}_1$, given by (3.4).*

DIRECT SOLUTION: SIMPLE LINEAR REGRESSION

- Elementwise representation can be cumbersome
- Many features/coefficients in practice

$$
\begin{aligned}
SSR &= \sum_{i=1}^{n}(y_i - (\beta_0 + \beta_1 x_i))^2 \\
&= \sum_{i=1}^{n}\left(y_i^2 - 2y_i(\beta_0 + \beta_1 x_i) + \beta_0^2 + 2\beta_0\beta_1 x_i + \beta_1^2 x_i^2\right)
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial SSR}{\partial \beta_1} &= \sum_{i=1}^{n}\left(-2x_i y_i + 2\beta_0 x_i + 2\beta_1 x_i^2\right) \\
0 &= -\sum_{i=1}^{n} x_i y_i + \hat{\beta}_0 \sum_{i=1}^{n} x_i + \hat{\beta}_1 \sum_{i=1}^{n} x_i^2 \\
0 &= -\sum_{i=1}^{n} x_i y_i + (\bar{y} - \hat{\beta}_1 \bar{x}) \sum_{i=1}^{n} x_i + \hat{\beta}_1 \sum_{i=1}^{n} x_i^2 \\
\boxed{\hat{\beta}_1 = \frac{\sum_{i=1}^{n} x_i(y_i - \bar{y})}{\sum_{i=1}^{n} x_i(x_i - \bar{x})}}
\end{aligned}
$$

(elementwise representation)

VECTORIZATION

- Rewrite the linear regression model and solution methods in matrices and vectors
- Simpler and more compact
- Utilize linear algebra libraries for faster computations

Linear algebra: https://www.khanacademy.org/math/linear-algebra

REVIEW: NOTATION

- Vector: $\mathbf{x} \in \mathbb{R}^n$

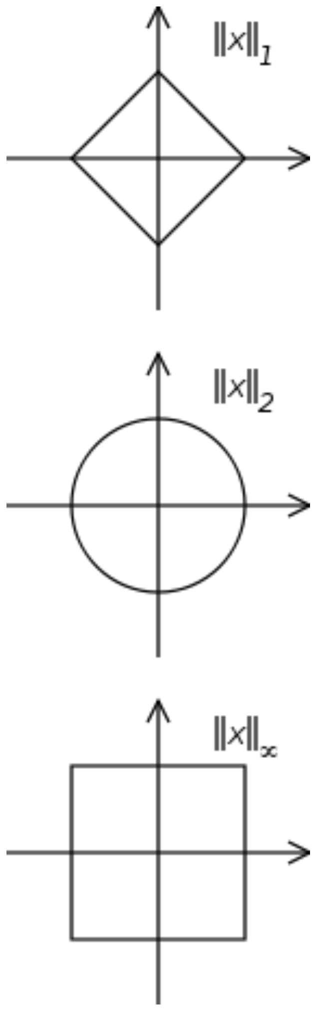
$$\mathbf{x} = X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

- Matrix: $\mathbf{A} \in \mathbb{R}^{m \times n}$

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

REVIEW: COMMON VECTOR NORMS

Norm	Formula
Euclidean	$\|\|\mathbf{x}\|\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2}$
Taxicab (Manhattan)	$\|\|\mathbf{x}\|\|_1 = \sum_{i=1}^{n} \|x_i\|$
Maximum (infinity)	$\|\|\mathbf{x}\|\|_\infty = \max_{x_i} \|x_i\|$
p-norm	$\|\|\mathbf{x}\|\|_p = \left(\sum_{i=1}^{n} \|x_i\|^p\right)^{1/p}$



REVIEW: RANK

- Column rank: size of largest subset of columns of A such that constitute a linearly independent set
- Row rank: largest number of rows of A that constitute a linearly independent set
- For any matrix in real space, column rank = row rank

REVIEW: MATRIX INVERSE

- Unique matrix such that

$$\mathbf{A}^{-1}\mathbf{A} = \mathbf{I} = \mathbf{A}\mathbf{A}^{-1}$$

- A is invertible and non-singular if inverse exists
- A is singular if not invertible
- A must be full rank to have an inverse

$$\underbrace{\begin{bmatrix} -3 & 1 \\ 5 & 0 \end{bmatrix}}_{\text{matrix A}} \underbrace{\begin{bmatrix} 0 & \frac{1}{5} \\ 1 & \frac{3}{5} \end{bmatrix}}_{\text{matrix } A^{-1}} = \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}}_{\substack{\text{2 x 2} \\ \text{identity matrix}}}$$

https://deepai.org/machine-learning-glossary-and-terms/invertible-matrix

REVIEW: MATRIX/VECTOR MANIPULATION

Rule	Comments
$(\mathbf{AB})^T = \mathbf{B}^T\mathbf{A}^T$	order is reversed, everything is transposed
$(\mathbf{a}^T\mathbf{Bc})^T = \mathbf{c}^T\mathbf{B}^T\mathbf{a}$	as above
$\mathbf{a}^T\mathbf{b} = \mathbf{b}^T\mathbf{a}$	(the result is a scalar, and the transpose of a scalar is itself)
$(\mathbf{A}+\mathbf{B})\mathbf{C} = \mathbf{AC}+\mathbf{BC}$	multiplication is distributive
$(\mathbf{a}+\mathbf{b})^T\mathbf{C} = \mathbf{a}^T\mathbf{C}+\mathbf{b}^T\mathbf{C}$	as above, with vectors
$\mathbf{AB} \neq \mathbf{BA}$	multiplication is **not** commutative

http://www.gatsby.ucl.ac.uk/teaching/courses/sntn/sntn-2017/resources/Matrix_derivatives_cribsheet.pdf

REVIEW: GRADIENTS

- Generalize derivatives to several variables
- Gradient of function f:

$$\nabla f(\mathbf{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \dots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}$$

REVIEW: VECTOR DERIVATIVES

Scalar derivative $f(x) \rightarrow \frac{\mathrm{d}f}{\mathrm{d}x}$	Vector derivative $f(\mathbf{x}) \rightarrow \frac{\mathrm{d}f}{\mathrm{d}\mathbf{x}}$
$bx \rightarrow b$	$\mathbf{x}^T\mathbf{B} \rightarrow \mathbf{B}$
$bx \rightarrow b$	$\mathbf{x}^T\mathbf{b} \rightarrow \mathbf{b}$
$x^2 \rightarrow 2x$	$\mathbf{x}^T\mathbf{x} \rightarrow 2\mathbf{x}$
$bx^2 \rightarrow 2bx$	$\mathbf{x}^T\mathbf{B}\mathbf{x} \rightarrow 2\mathbf{B}\mathbf{x}$

http://www.gatsby.ucl.ac.uk/teaching/courses/sntn/sntn-2017/resources/Matrix_derivatives_cribsheet.pdf

LINEAR REGRESSION: MATRIX REPRESENTATION

- Outcome variables

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

- Predictor variables
  n x (p+1)

$$\mathbf{x} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}$$

- Coefficients

$$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix}$$

LINEAR REGRESSION: MATRIX REPRESENTATION

- Outcome variables $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$

- Predictor variables
  n x (p+1)
  $\mathbf{x} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}$

- Coefficients $\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix}$

- Prediction $\mathbf{x}\beta = \begin{bmatrix} \beta_0 + \beta_1 x_1 \\ \beta_0 + \beta_1 x_2 \\ \vdots \\ \beta_0 + \beta_1 x_n \end{bmatrix}$

- Residual $\mathbf{e}(\beta) = \mathbf{y} - \mathbf{x}\beta$

- MSE

$$\begin{aligned} MSE(\beta) &= \frac{1}{n}\mathbf{e}^T\mathbf{e} \\ &= \frac{1}{n}(\mathbf{y} - \mathbf{x}\beta)^T(\mathbf{y} - \mathbf{x}\beta) \end{aligned}$$

DIRECT SOLUTION: MATRIX FORM

- Goal: find coefficient vector β that minimizes MSE

$$
\begin{aligned}
MSE(\beta) &= \frac{1}{n}\mathbf{e}^T\mathbf{e} \\
&= \frac{1}{n}(\mathbf{y}-\mathbf{x}\beta)^T(\mathbf{y}-\mathbf{x}\beta) \\
&= \frac{1}{n}(\mathbf{y}^T-\beta^T\mathbf{x}^T)(\mathbf{y}-\mathbf{x}\beta) \\
&= \frac{1}{n}\left(\mathbf{y}^T\mathbf{y}-\mathbf{y}^T\mathbf{x}\beta-\beta^T\mathbf{x}^T\mathbf{y}+\beta^T\mathbf{x}^T\mathbf{x}\beta\right)
\end{aligned}
$$

- Computer the gradient of the MSE with respect to β

$$
\begin{aligned}
\nabla MSE(\beta) &= \frac{1}{n}\left(\nabla\mathbf{y}^T\mathbf{y}-2\nabla\beta^T\mathbf{x}^T\mathbf{y}+\nabla\beta^T\mathbf{x}^T\mathbf{x}\beta\right) \\
&= \frac{1}{n}\left(0-2\mathbf{x}^T\mathbf{y}+2\mathbf{x}^T\mathbf{x}\beta\right) \\
&= \frac{2}{n}\left(\mathbf{x}^T\mathbf{x}\beta-\mathbf{x}^T\mathbf{y}\right)
\end{aligned}
$$

- Set the gradient to 0, solve β

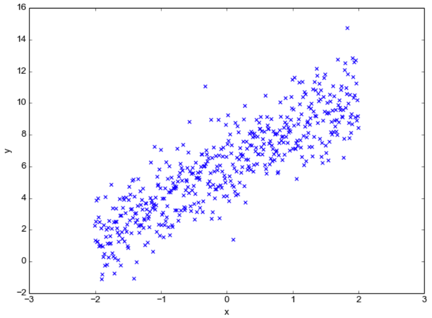
$$\mathbf{x}^T\mathbf{x}\widehat{\beta}-\mathbf{x}^T\mathbf{y}=0$$

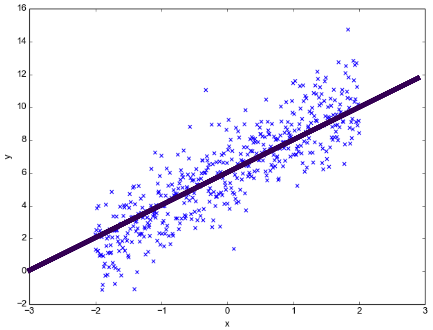
$$\boxed{\widehat{\beta}=(\mathbf{x}^T\mathbf{x})^{-1}\mathbf{x}^T\mathbf{y}}$$

http://www.stat.cmu.edu/~cshalizi/mreg/15/lectures/13/lecture-13.pdf

LINEAR REGRESSION

- Training:





$$\mathbf{x} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix} \qquad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

$$\widehat{\beta} = (\mathbf{x}^T\mathbf{x})^{-1}\mathbf{x}^T\mathbf{y}$$

- Prediction: $\quad \mathrm{X} \longrightarrow \hat{\mathbf{y}} = \mathbf{X}\hat{\beta}$

GEOMETRY OF LS SOLUTION

- Outcome vector is orthogonally projected onto hyperplane spanned by input features

$$\hat{\mathbf{y}} = \mathbf{X}\hat{\boldsymbol{\beta}} = \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}$$

- The "hat" matrix or projection matrix

$$\mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top$$

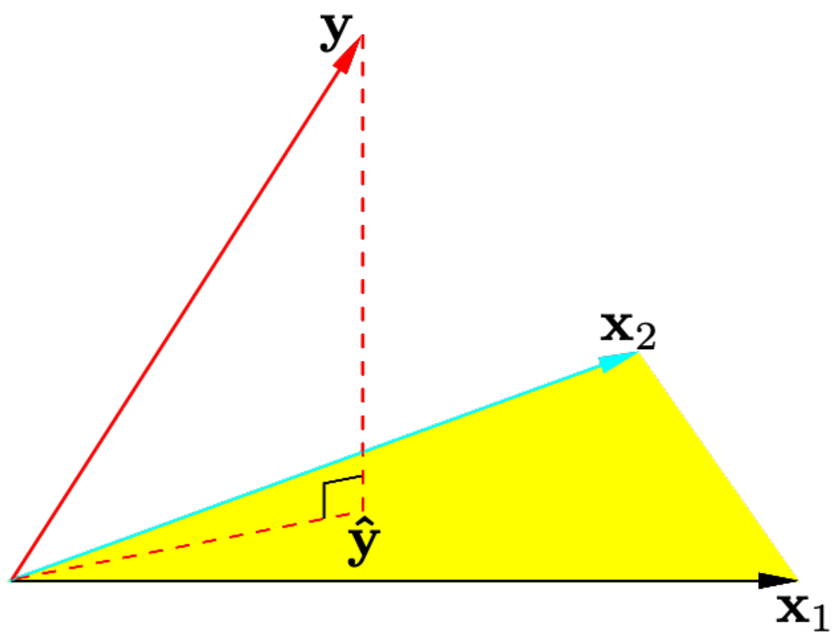


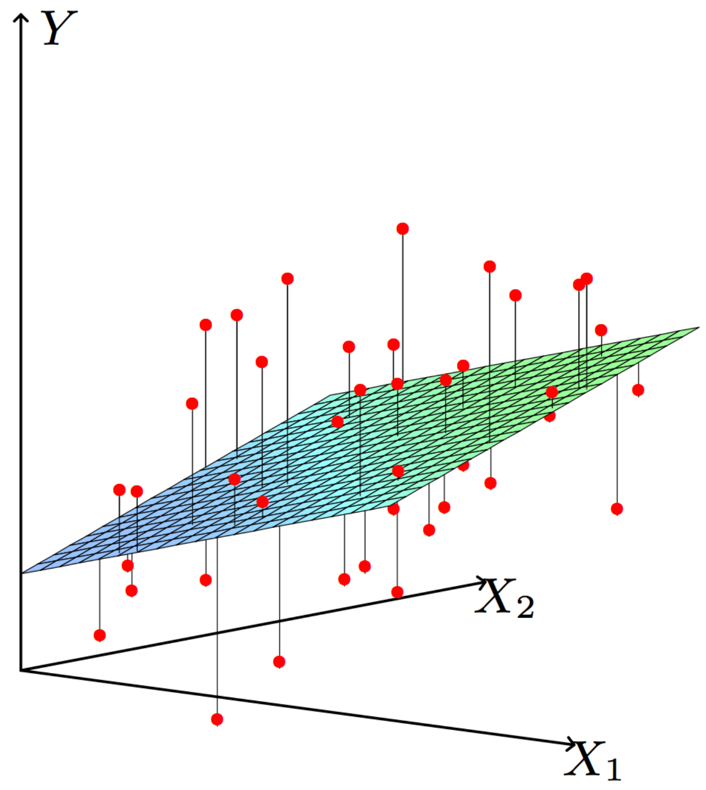
Figure 3.2 (Hastie et al.)

LINEAR ALGEBRA: PYTHON (HINT FOR HW3)

- Create an array of ones: numpy.ones
- Concatenation: numpy.concatenate
- Multiplication: numpy.matmul
- Transpose numpy.transpose
- Inverse: numpy.linalg.inv

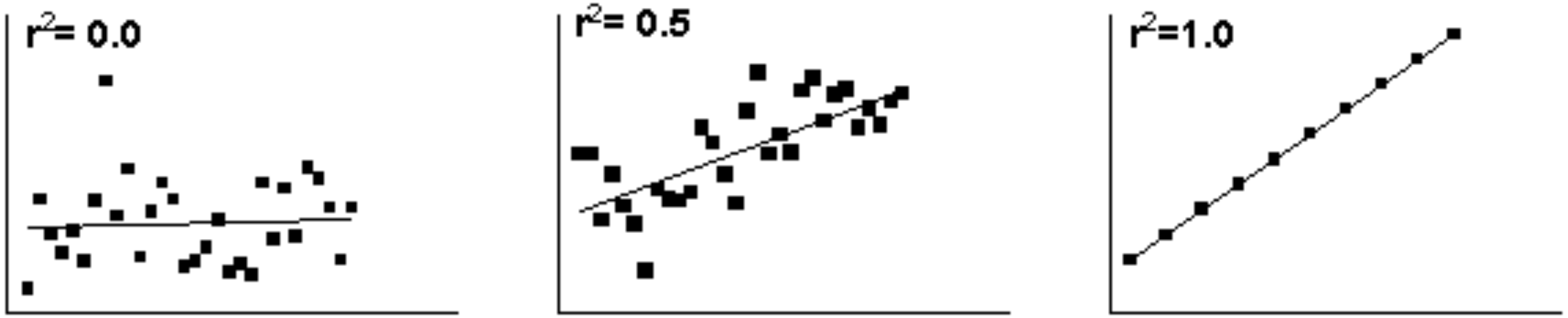
ASSESSING THE ACCURACY OF THE MODEL

- Residual error
- R^2 statistic



MEASURE OF FIT: R^2

- "Goodness" of fit measure $$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$$
- Interpretation: The proportion of variability in y explained by the model
- Always lies between 0 and 1



https://socratic.org/questions/why-doesn-t-an-r-squared-value-indicate-anything-about-causation

STANDARD LINEAR REGRESSION: RECAP

- Objective function: Minimize RSS
- Coefficients have a nice interpretation
- Closed-form solution $\hat{\boldsymbol{\beta}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$

UNDERSTANDING MLR

- Extremely hard to find “causal” relationships between features and outcome
- Any correlation (association) could be caused by other variables in the background — correlation is NOT causation
- Multivariate regression allows us to control for all important variables by including them in the regression

CORRELATION DOES NOT IMPLY CAUSALITY