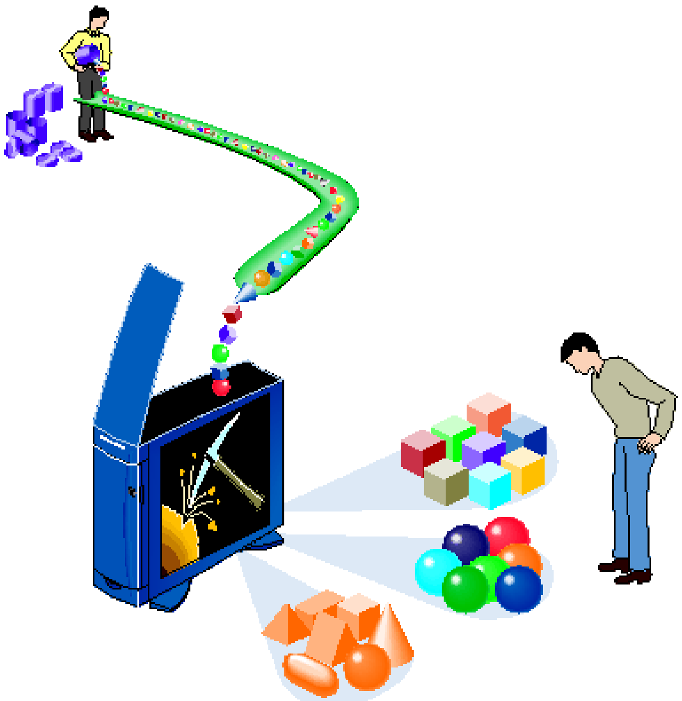
IE 582 Statistical Learning for Data Mining



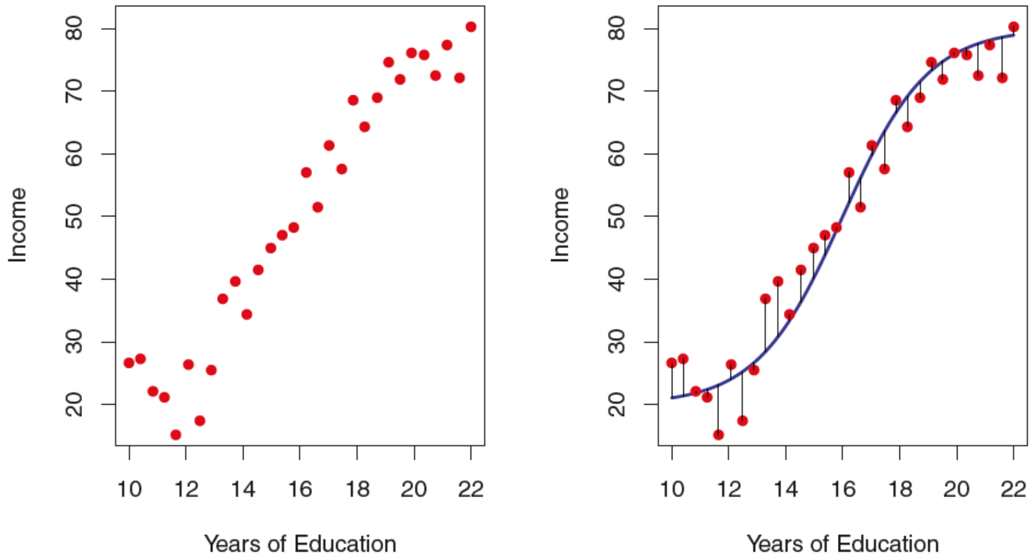
Instructor: Mustafa Gökçe Baydoğan
Office: M4082

mustafa.baydogan@boun.edu.tr
www.mustafabaydogan.com
blog.mustafabaydogan.com

Statistical inference

- A quantitative response (Y) and p predictors (X_1, X_1,..., X_p)

$$Y = f(X) + \epsilon.$$



Estimation of f

- Two main reasons
  - Prediction: estimate $\hat{Y}$ given predictors

$$\hat{Y} = \hat{f}(X)$$

$$\begin{aligned} E(Y-\hat{Y})^2 &= E[f(X)+\epsilon-\hat{f}(X)]^2 \\ &= \underbrace{[f(X)-\hat{f}(X)]^2}_{\text{Reducible}} + \underbrace{\mathrm{Var}(\epsilon)}_{\text{Irreducible}} \end{aligned}$$

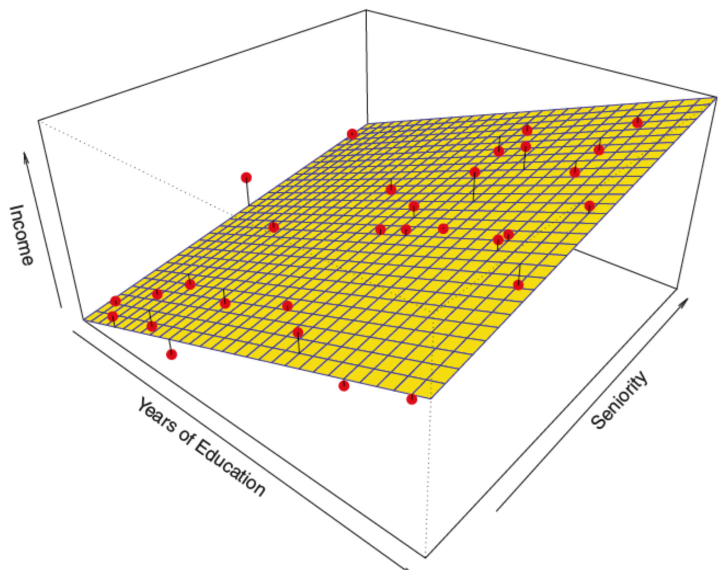
  - Inference: understand how Y is affected with the change in predictors
    - Which predictors are associated with the response?
    - What is the relationship between the response and each predictor?
    - Can the relationship between Y and each predictor be adequately summarized using a linear equation, or is the relationship more complicated?

How to estimate f

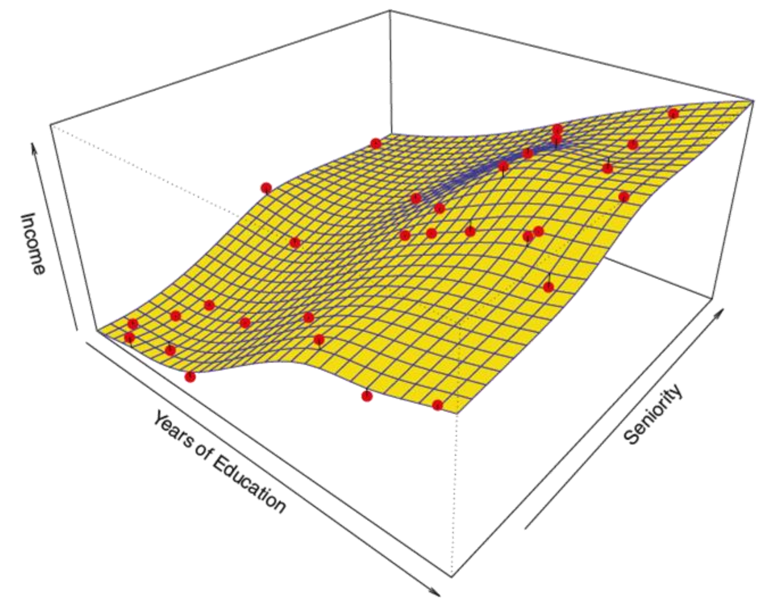
- Parametric (model-based) approaches
  - i.e. linear regression

$$f(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p$$

  - Estimation of the parameters
    - Also referred to as *fitting* or *training* the model



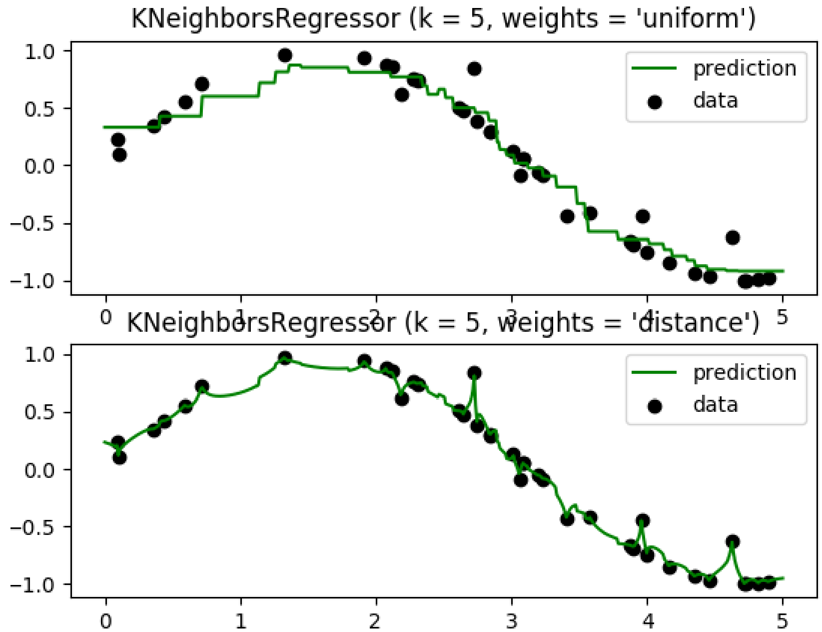
or?



$$\texttt{income} \approx \beta_0 + \beta_1 \times \texttt{education} + \beta_2 \times \texttt{seniority}$$

How to estimate *f*

- Non-parametric approaches
  - No assumption of a particular functional form for *f*
  - Do not reduce the problem of estimating *f* to a small number of parameters
    - Large number of observations is required to obtain an accurate estimate for *f*.

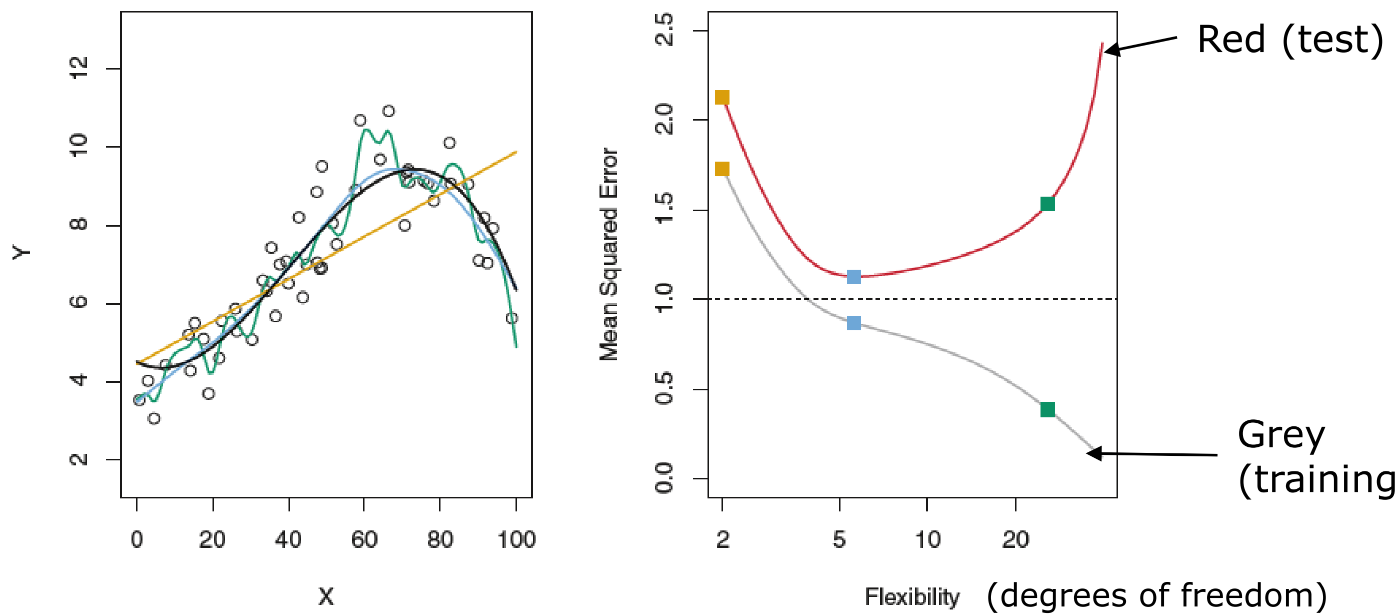


How to evaluate quality of the fit

- Accuracy

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

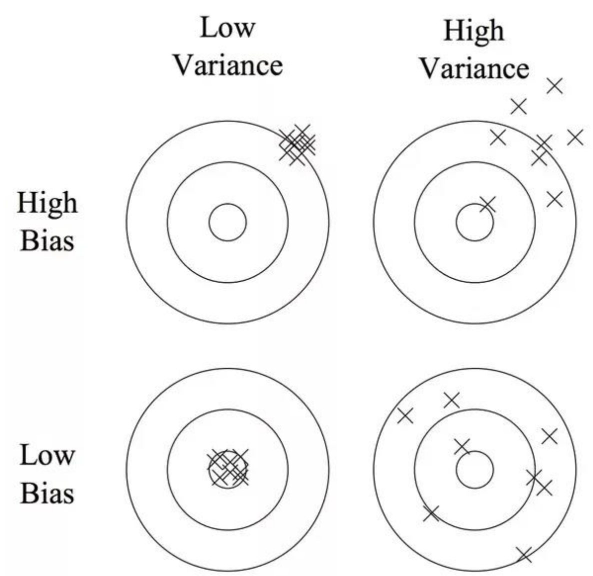
- Training and test data
  - Interested in the accuracy of the predictions on previously unseen test data

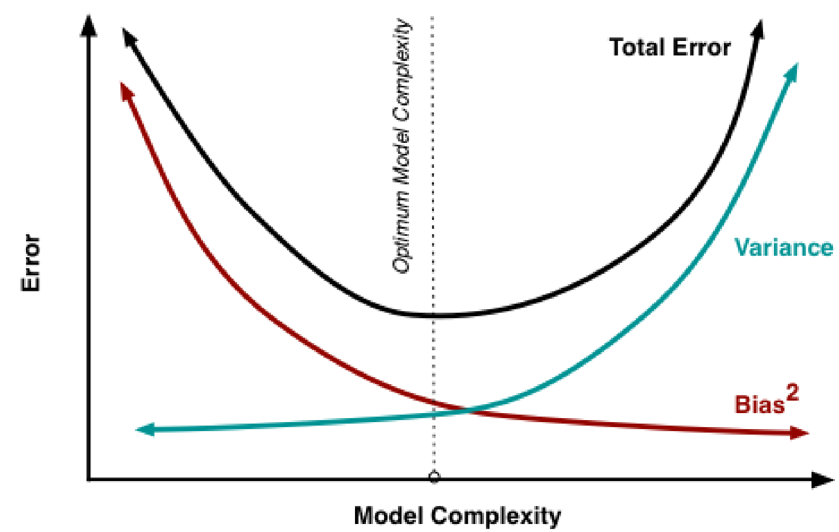


Two competing objectives
Bias and variance of estimators

- Expected test MSE
  - average test MSE if we repeatedly estimated *f* using a large number of training sets, and tested each at observation x_0

$$E\left(y_0 - \hat{f}(x_0)\right)^2 = \mathrm{Var}(\hat{f}(x_0)) + [\mathrm{Bias}(\hat{f}(x_0))]^2 + \mathrm{Var}(\epsilon)$$





Classification and regression

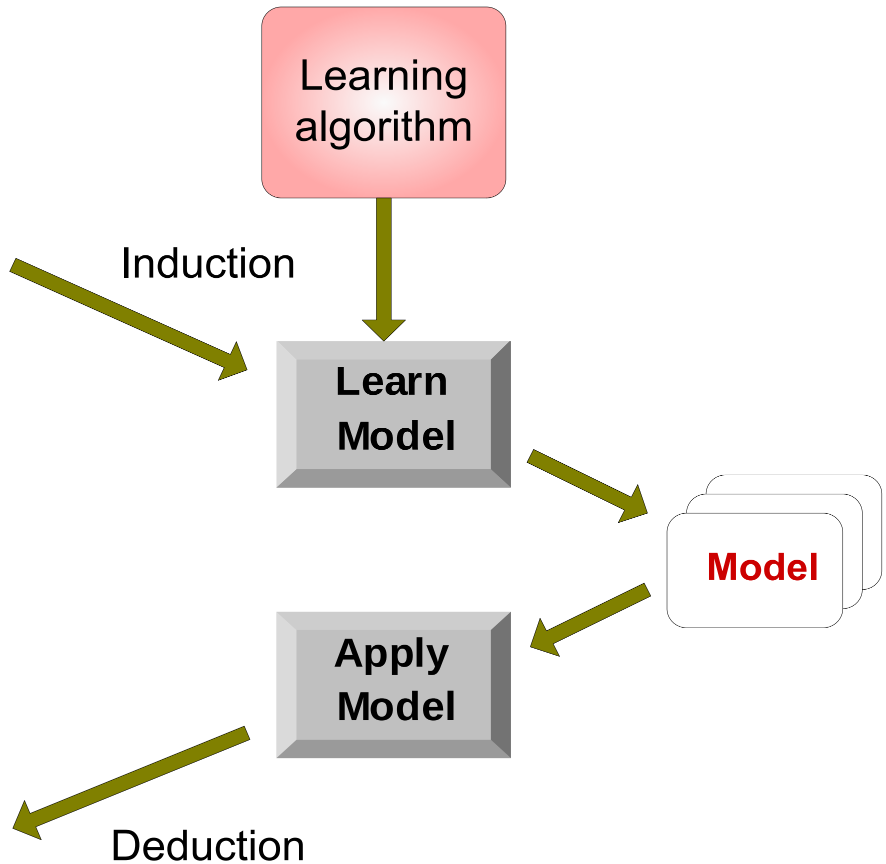
- Distribution of *Y*
  - Regression: i.e. gaussian, exponential, ...
    - Forecasting daily demand
    - Predicting annual income
  - Classification: i.e. bernouilli (or binomial), multinomial
    - Predicting churn
    - Predicting credit default
- Function *f* approximates population (distribution) parameters

$$f(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p$$

Classification

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

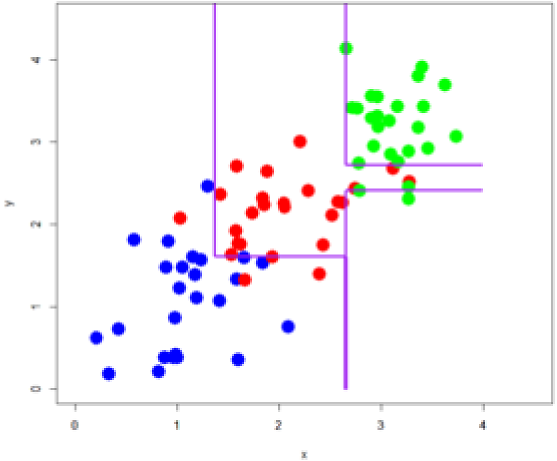


Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

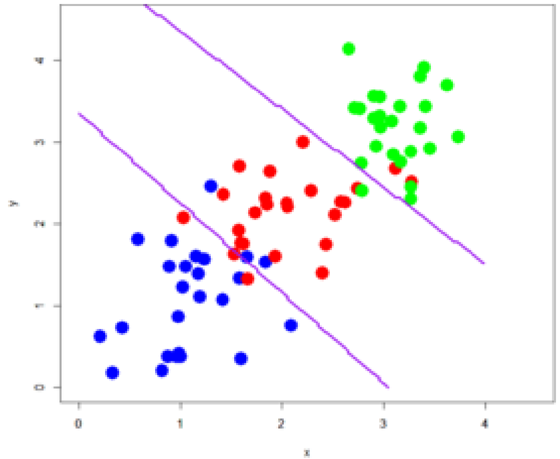
Test Set

Regression vs Classification

- Basic difference
  - In classification, we have dependent variables that are <u>categorical and unordered</u>.
  - In regression, we have dependent variables that are <u>continuous values or ordered whole values</u>.
- All regression approaches can be used to solve the classification problem. **How?**
- From my viewpoint, the classification problem is all about drawing the "right" decision boundary.



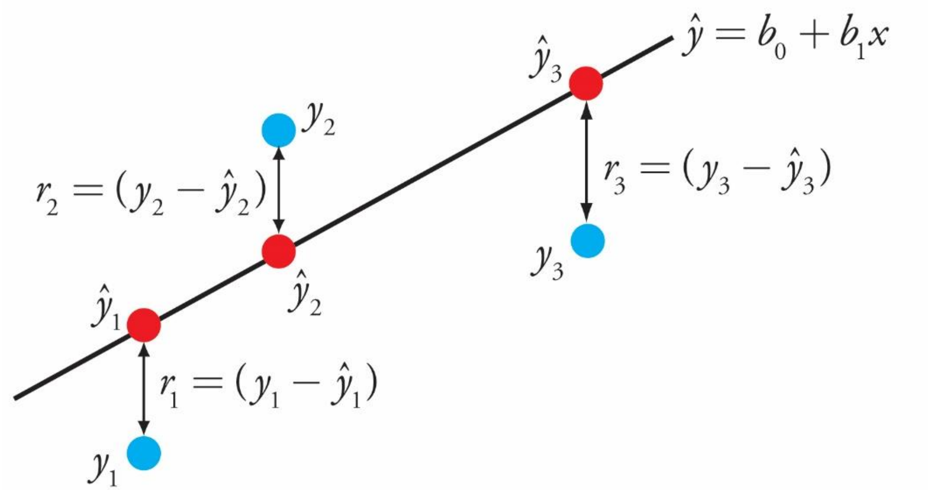
or



<u>Discriminative</u>
versus
Generative

Classification
Using linear regression for classification

- Minimizing sum of square error

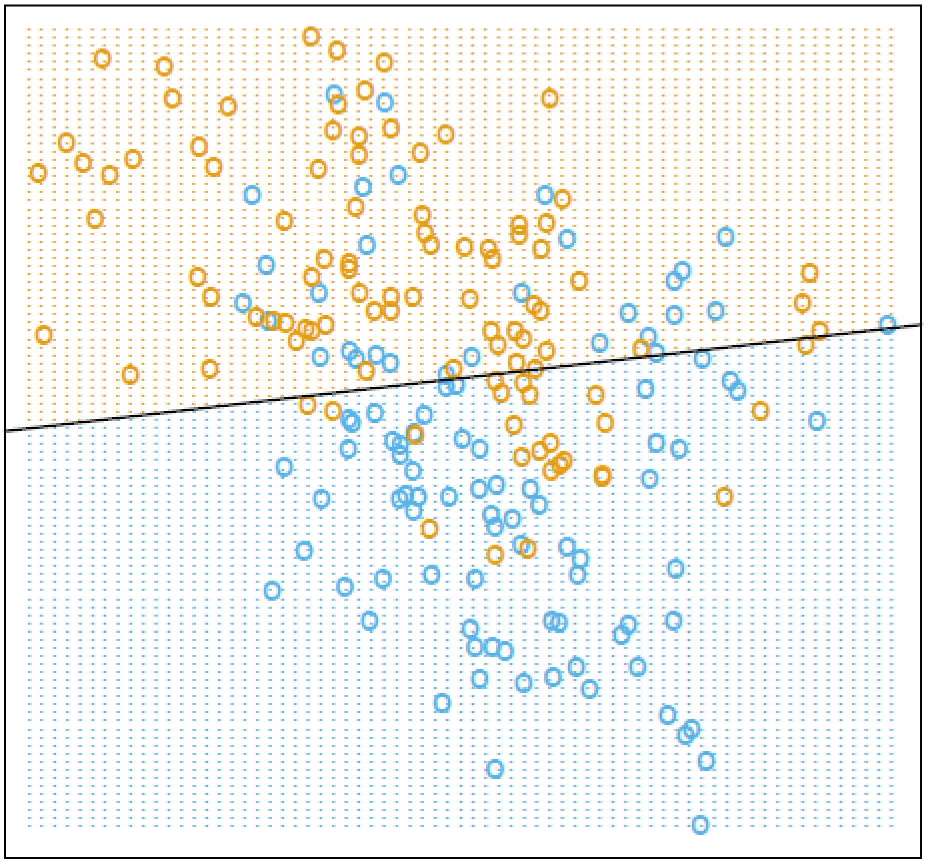


- Suppose we have the following two-class classification problem

X1	X2	Class
7.40	1.91	BLUE
3.92	0.24	ORANGE
2.15	1.08	ORANGE
-2.36	0.70	BLUE
.	.	.
.	.	.
.	.	.
0.09	-1.75	ORANGE
0.71	0.67	BLUE

Classification
Using linear regression for classification

Linear Regression of 0/1 Response



$(\text{BLUE} = 0,\ \text{ORANGE} = 1)$,

ORANGE

$\{x : x^T\hat{\beta} > 0.5\}$

Decision boundary

$\{x : x^T\hat{\beta} = 0.5\}$

X1	X2	Class
7.40	1.91	0
3.92	0.24	1
2.15	1.08	1
-2.36	0.70	0
.	.	.
.	.	.
.	.	.
0.09	-1.75	1
0.71	0.67	0

Classification
Using linear regression for classification

- Potential problems?
  - Assumption of linear regression
    - i.e. Normally distributed residuals
  - Effects of multicollinearity (i.e. correlated predictors) -> unstable regression coefficients
  - Works only for 2-class classification
    - Requires extension for multi-class cases
  - Categorical and ordinal predictors?
    - Requires binary representation (i.e. introducing "dummy" variables)
  - Nonlinear representation
    - Addition of polynomial terms (i.e. X^2)
    - Addition of interaction terms (i.e. XY)
  - Requires the setting of the threshold in practice

Classification
Logistic regression

- Why preferable to linear regression?
  - *e* is not normally distributed because *Y* takes on only two values

$$\hat{Y} = \hat{\beta}_0 + \sum_{j=1}^{p} X_j \hat{\beta}_j \text{ + error } (e)$$

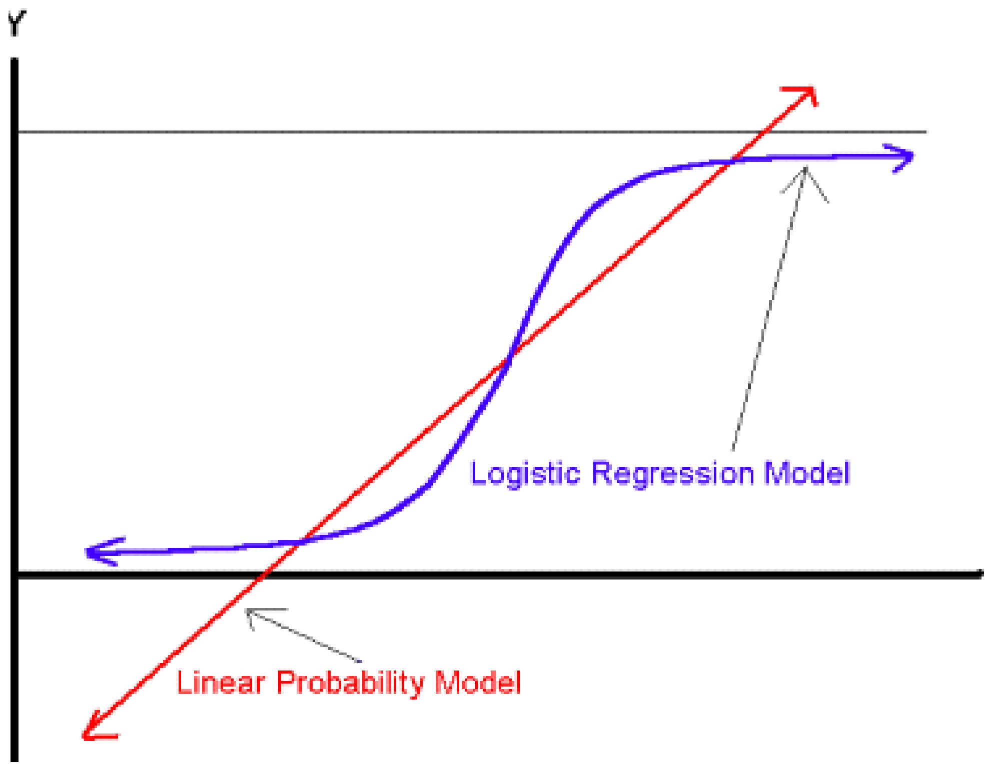
  - The predicted probabilities can be greater than 1 or less than 0
- Logistic regression result is in the range [0,1]

$$\Pr(G = 1|X = x) = \frac{\exp(\beta_0 + \beta^T x)}{1 + \exp(\beta_0 + \beta^T x)},$$

$$\Pr(G = 2|X = x) = \frac{1}{1 + \exp(\beta_0 + \beta^T x)}.$$

Classification
Logistic regression

- Logistic regression versus linear regression



linear regression

$$\hat{Y} = \hat{\beta}_0 + \sum_{j=1}^{p} X_j \hat{\beta}_j.$$

logistic regression

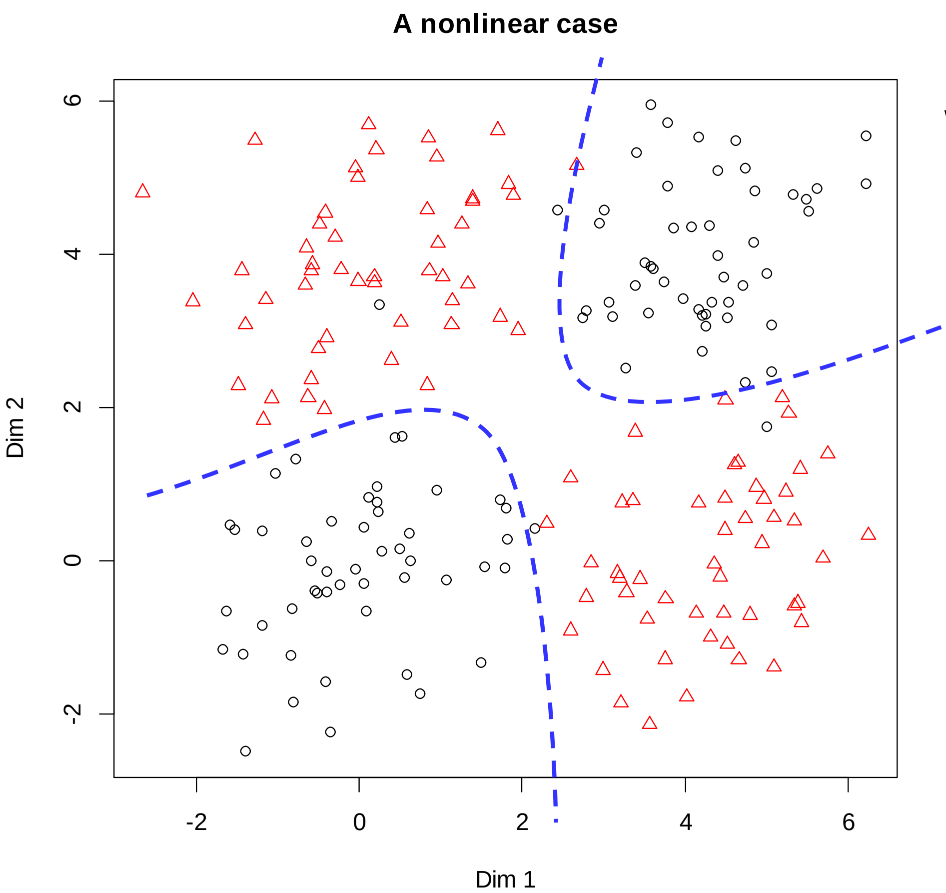
$$\Pr(G = 1|X = x) = \frac{\exp(\beta_0 + \beta^T x)}{1 + \exp(\beta_0 + \beta^T x)}$$

$$\Pr(G = 2|X = x) = \frac{1}{1 + \exp(\beta_0 + \beta^T x)}$$

Classification
Logistic vs Linear illustration

- R codes on Moodle

Classification
Nonlinear cases

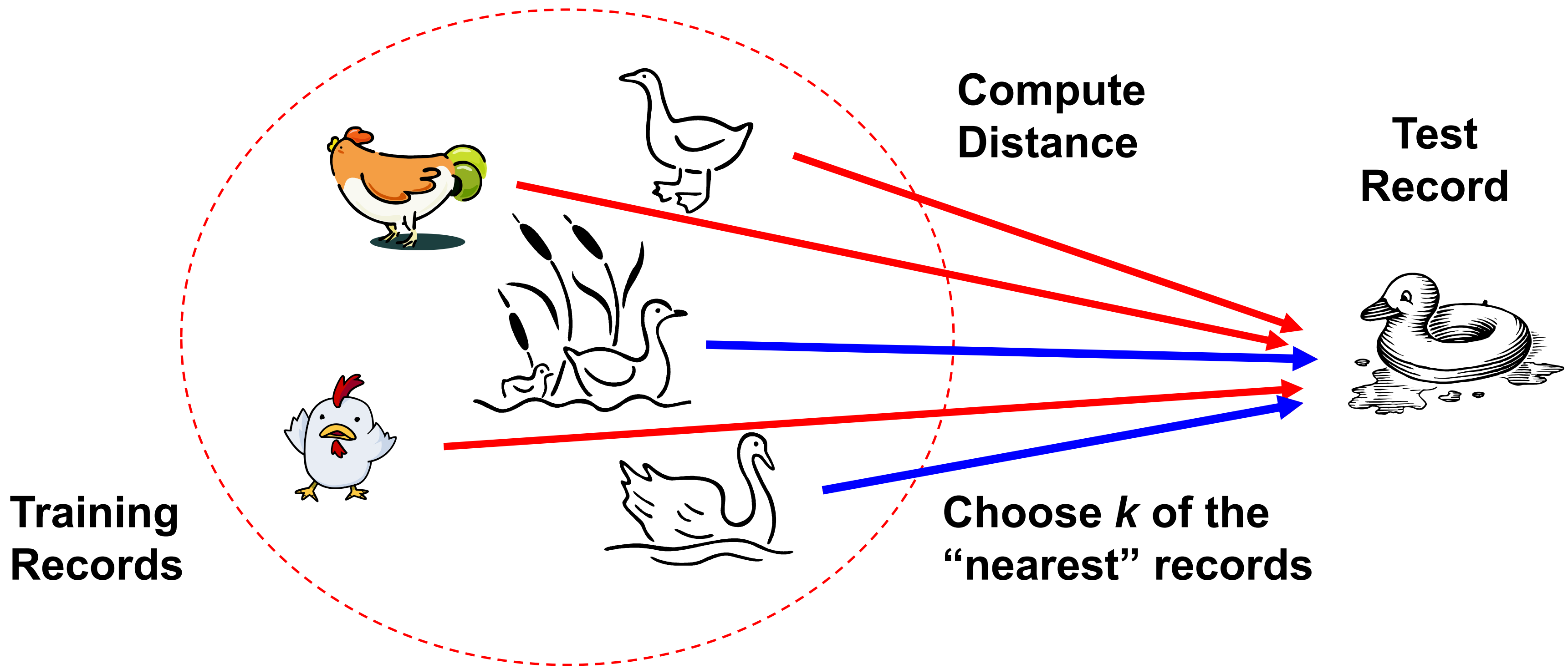


What if a linear boundary does not work?

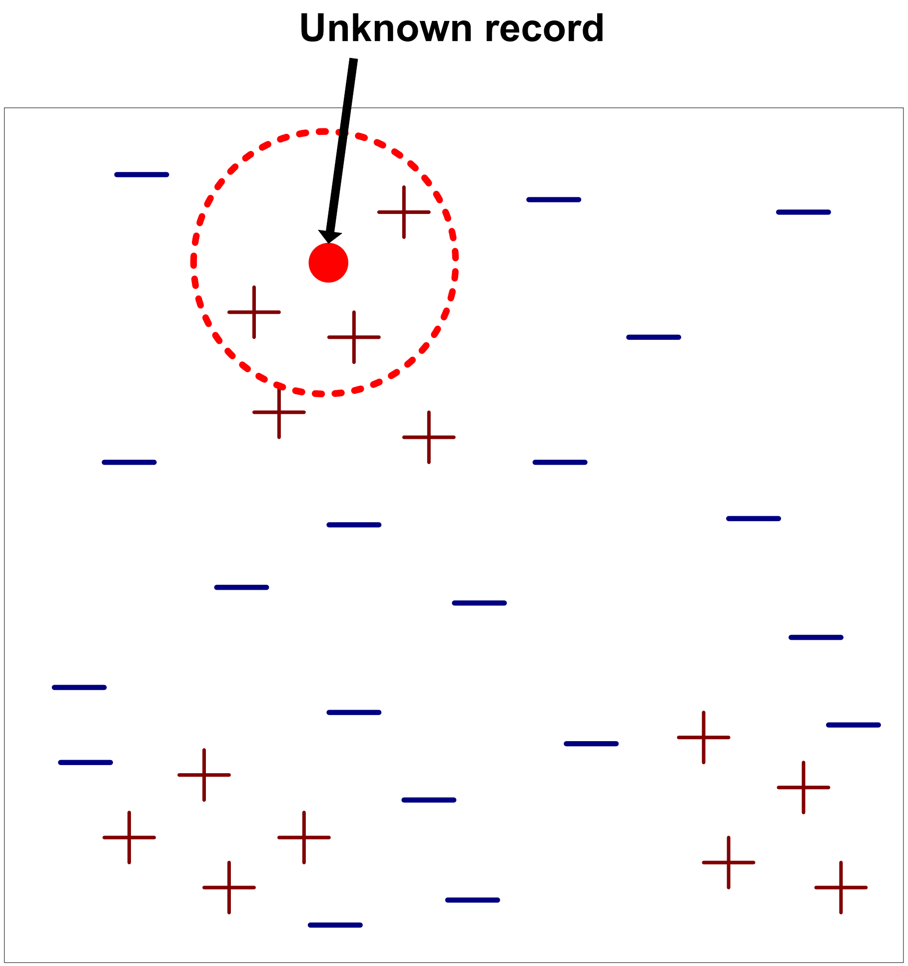
- Introduction of nonlinear terms
  - What are the possibilities?
- Methods that can handle nonlinear relations
  - There are many of them
  - Let’s start with Nearest Neighbor (NN) classifier

Classification
Nearest-Neighbor

- Basic idea:
  - If it walks like a duck, quacks like a duck, then it's probably a duck



Classification
Nearest-Neighbor



- Requires three things
  - The set of stored records
  - Similarity measure to compute distance between records
  - The value of *k*, the number of nearest neighbors to retrieve

- To classify an unknown record:
  - Compute distance to other training records
  - Identify *k* nearest neighbors
  - Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)

Classification
Nearest-Neighbor

- The ***k***-nearest neighbor fit is

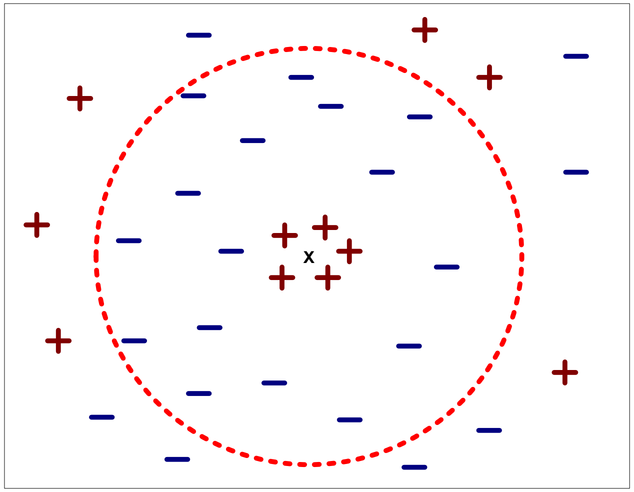
$$\hat{Y}(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i,$$

$\boldsymbol{N_k(x)}$ is the neighborhood of the instance ***x*** defined by the ***k*** closest points (instances) in the training data

- Equation is the average of the outputs of the closest points
  - A solution to regression
- What to do for classification?
  - Mode?
- What about the weighted average?
  - How?

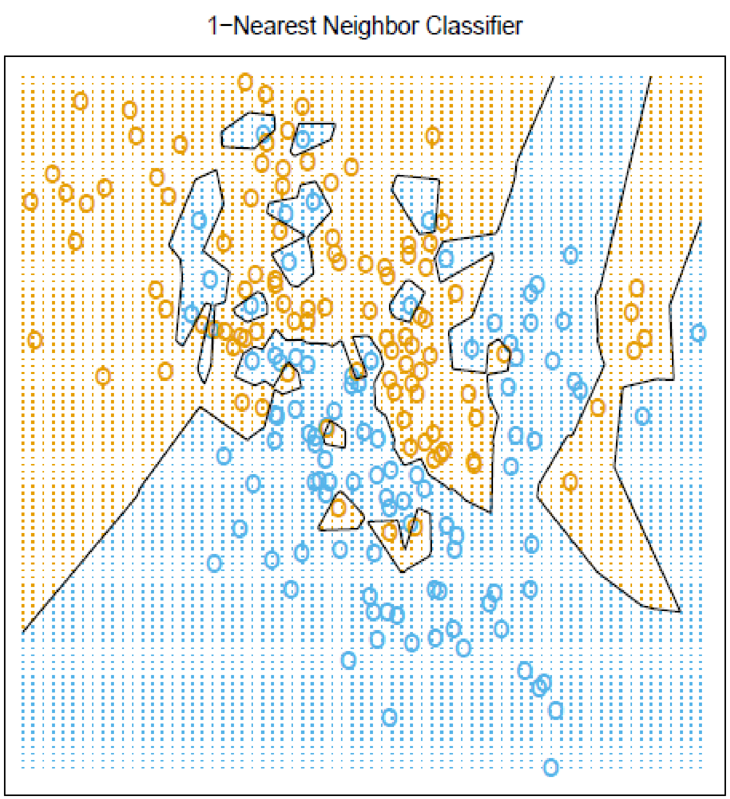
Classification
Nearest-Neighbor

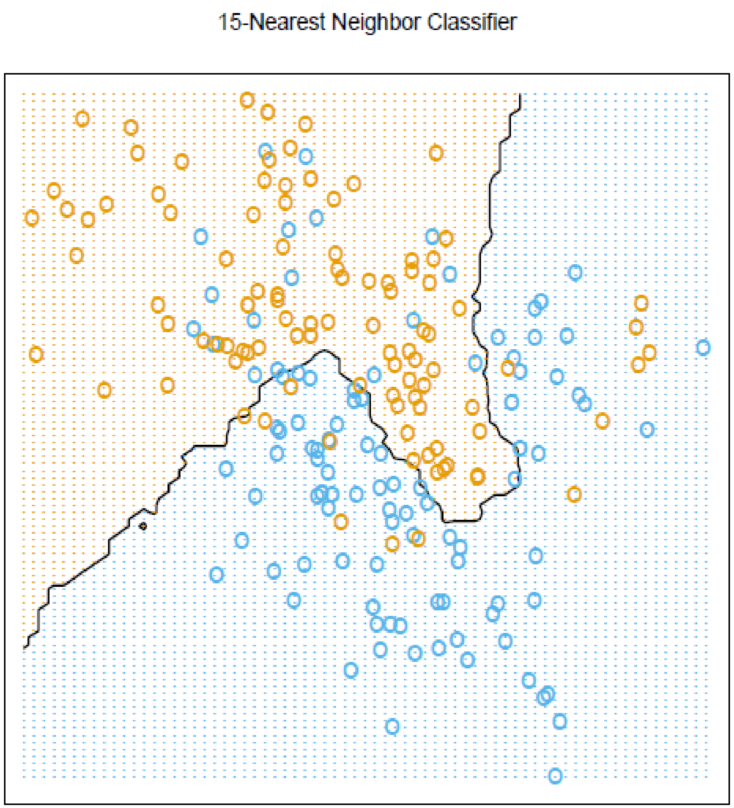
- How to select *k*?
  - We cannot use sum-of-squared errors on the training, why?
  - If *k* is too small, sensitive to noise points
  - If *k* is too large, neighborhood may include points from other classes



Classification
Nearest-Neighbor

- Sample decision boundaries for orange-blue classification problem





Classification
Nearest-Neighbor

- Lazy learner
  - There is no model
    - Not interpretable
  - For each test data, similarity computation to each training data point is required
    - Problematic for <u>real-time</u> applications
      - Especially if the training data size is large
    - Also referred to as <u>instance-based</u> approach (see supplementary slides at the end)
    - <u>Not memory efficient</u>
      - Requires storage of the training data
- Requires a <u>similarity measure</u>
  - Problematic when the number of features is large (i.e. curse of dimensionality)
- Handles <u>nonlinear</u> decision boundaries

Classification Nearest-Neighbor

- Scaling issues
  - Features may have to be scaled to prevent similarity measures from being dominated by one of the features
  - Example:
    - height of a person may vary from 1.5m to 1.8m
    - weight of a person may vary from 40kg to 120kg
    - income of a person may vary from $10K to $1M
- A big problem for the approaches that uses the notion of similarity

Classification
Nearest-Neighbor

- Example: NN Classification on time series
  - R codes on Moodle
  - ECG dataset from
    - http://www.cs.ucr.edu/~eamonn/time_series_data/
    - 2-class (binary) classification problem to distinguish patients with Cardiac dysrhythmia (also known as **arrhythmia** or **irregular heartbeat**) based on their Electrocardiography records
    - 100 training instances with 96 observations
    - 100 test instances