PSU IST 557: ML Basics Review

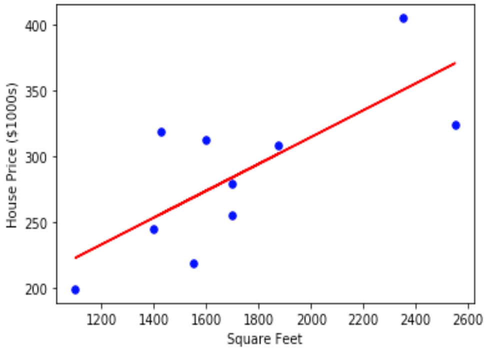
Lu Lin, Ph.D.

College of Information Sciences and Technology
Pennsylvania State University
lulin@psu.edu

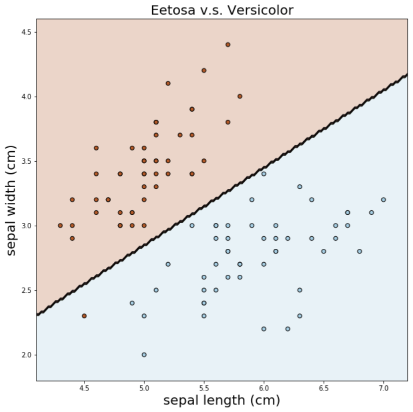
ML Basics Review

Supervised Learning: making prediction
Given data points $\{(X_1, Y_1), \dots, (X_n, Y_n)\}$, learn a model to predict Y_i from X_i

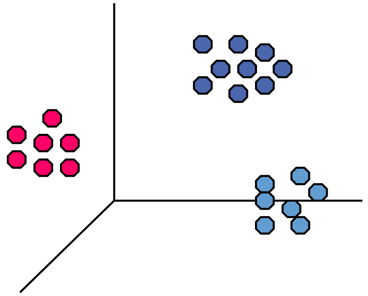
Unsupervised Learning: discovering structure
Given data points $\{X_1, \dots, X_n\}$, learn the underlying structure within data



Regression

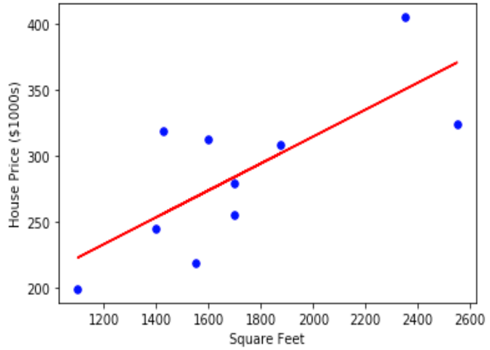


Classification



Clustering

Linear Regression



- Supervision signal is continuous
  - Given: a training set $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$, where each $\mathbf{x}_i$ is a set of predictor variables and y_i is the corresponding value of the continuous target variable
  - Regression task: learn a target function $f(\mathbf{x}; \mathbf{w})$ to predict the continuous value of y for any given input $\mathbf{x}$
    - $\mathbf{w}$ is the model parameter

Linear Regression

- Supervision signal is continuous
    - Step 1: hypothesis -- define a set of (linear) functions

ID	Feature
x_{living}	Living room size
x_{bed}	Number of bedrooms
x_{bath}	Number of bathrooms
x_{base}	Basement size
…	…

$f($

$x) =$

y

Hypothesis: $f(\mathbf{x};\mathbf{w}) = \mathbf{w}^T\mathbf{x} = w_0 + w_1 x_{living} + w_2 x_{bed} + \cdots$

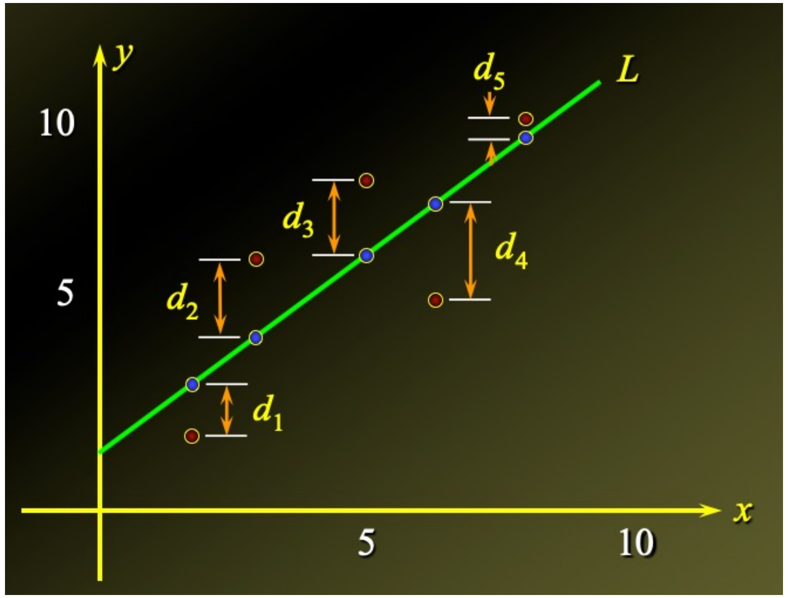
$\mathbf{w}^T = [w_0, w_1, w_2, \dots, w_d]$ are parameters

Linear Regression

- Supervision signal is continuous
  - Step 2: loss function -- estimate goodness of functions
  - Hypothesis: $f(\mathbf{x}; \mathbf{w}) = \mathbf{w}^T\mathbf{x}$

The distance d_1, d_2, d_3, d_4 and d_5 are the errors the linear function makes in fitting these points:

$$d_i = y_i - f(\mathbf{x}_i; \mathbf{w}) = y_i - \mathbf{w}^T\mathbf{x}_i$$



Linear Regression

- Supervision signal is continuous
  - Step 2: loss function -- estimate goodness of functions

  - Least squares principle: the linear function that fits the data best should result in minimal sum of the squares of errors

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \cdots \\ y_n \end{bmatrix} \quad \mathbf{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1d} \\ 1 & x_{21} & x_{22} & \cdots & x_{2d} \\ & & \cdots & & \\ 1 & x_{n1} & x_{n2} & \cdots & x_{nd} \end{bmatrix} \quad \mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ \cdots \\ w_d \end{bmatrix}$$

    - Square loss:

$$L(w,b) = \sum_{i=1}^{n} \left(y_i - f(\mathbf{x}_i; \mathbf{w})\right)^2 = ||\mathbf{y} - \mathbf{X}\mathbf{w}||^2$$

Linear Regression

- Supervision signal is continuous
  - Step 3: optimization -- pick the "best" function

  - Find the function that minimizes the loss function

$$\mathbf{w}^* = \text{argmin}_{\mathbf{w}}\, L(\mathbf{w}) = \text{argmin}_{\mathbf{w}} ||\mathbf{y} - \mathbf{X}\mathbf{w}||^2$$

Testing:
Given a test feature $\mathbf{x}_{\text{test}}$:
$\hat{y}_{\text{test}} = f(\mathbf{x}_{\text{test}};\ \mathbf{w}^*)$

Analytical solution: $\nabla_{\mathbf{w}} ||\mathbf{y} - \mathbf{X}\mathbf{w}||^2 = -2\mathbf{X}^T\mathbf{y} + 2\mathbf{X}^T\mathbf{X}\mathbf{w} = 0 \Leftrightarrow \mathbf{w}^* = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$

Gradient descent: $\mathbf{w}^{t+1} \leftarrow \mathbf{w}^t - \eta \frac{\partial L(\mathbf{w})}{\partial \mathbf{w}} |_{\mathbf{w}=\mathbf{w}^t}$

Probabilistic Perspective

- Given:
  - n training examples $\{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$, features: $\mathbf{x}_i \in \mathbb{R}^d$, target: $y_i \in \mathbb{R}$
- Probabilistic view: targets are generated via a probabilistic model
  - Assume a "noisy" linear model with parameters $\mathbf{w} \in \mathbb{R}^d$

$$y_i = h(\mathbf{x}_i; \mathbf{w}) + \epsilon_i \qquad h(\mathbf{x}_i; \mathbf{w}) = \mathbf{w}^T \mathbf{x}_i$$

  - Assume Gaussian noise: $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$, thus the target follows

$$y_i \sim \mathcal{N}(h(\mathbf{x}_i; \mathbf{w}), \sigma^2)$$

  - Goal: learn model parameter vector $\mathbf{w}$ to predict y^* for a new x^*

Probabilistic Perspective

- For target following Gaussian distribution:

$$y_i \sim \mathcal{N}(h(\mathbf{x}_i;\mathbf{w}), \sigma^2)$$

$$\Rightarrow P(y_i|\mathbf{x}_i;\mathbf{w}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{\left(y_i - h(\mathbf{x}_i;\mathbf{w})\right)^2}{2\sigma^2})$$

- The probability or likelihood of the data (assuming i.i.d. target y)

$$P(\mathbf{y}|\mathbf{X};w) = P(y_1,\cdots,y_n|\mathbf{X};w) = \prod_{i=1}^{n} P(y_i|\mathbf{x}_i;w) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{\left(y_i - h(\mathbf{x}_i;\mathbf{w})\right)^2}{2\sigma^2})$$

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

- Greater likelihood indicates higher probability of observing the given data

Maximum Likelihood Estimation (MLE)

- Maximizing the likelihood is the same as minimizing the negative of maximum likelihood (e.g., Sum of Squared Errors)

$$\mathbf{w}^* = \operatorname{argmax}_{\mathbf{w}} \log\big(P(y_1, \cdots, y_n | \mathbf{X}; w)\big) = \operatorname{argmax}_{\mathbf{w}} \sum_{i=1}^{n} \log\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) + \sum_{i=1}^{n} -\frac{\big(y_i - h(\mathbf{x}_i; \mathbf{w})\big)^2}{2\sigma^2}$$

$$\mathbf{w}^* = \operatorname{argmin}_{\mathbf{w}} \; -\log\big(P(y_1, \cdots, y_n | \mathbf{X}; w)\big) = \operatorname{argmin}_{\mathbf{w}} \overbrace{\boxed{\sum_{i=1}^{n} -\log\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)}}^{c} + \overbrace{\boxed{\sum_{i=1}^{n} \frac{\big(y_i - h(\mathbf{x}_i; \mathbf{w})\big)^2}{2\sigma^2}}}^{\sigma^2 = \frac{1}{2}}$$

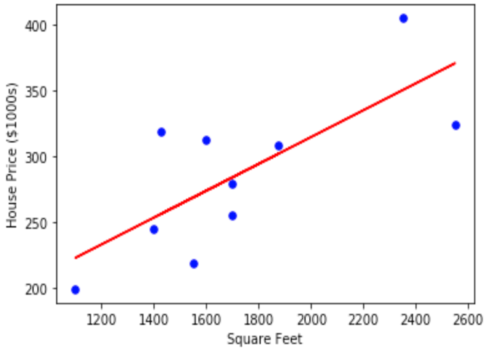
$$= \operatorname{argmin}_{\mathbf{w}} \; c + \sum_{i=1}^{n} \big(y_i - h(\mathbf{x}_i; \mathbf{w})\big)^2$$

SSE loss function

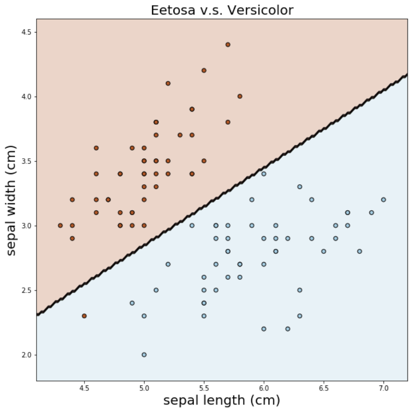
ML Basics Review

Supervised Learning: making prediction

Given data points $\{(X_1, Y_1), \ldots, (X_n, Y_n)\}$, learn a model to predict Y_i from X_i



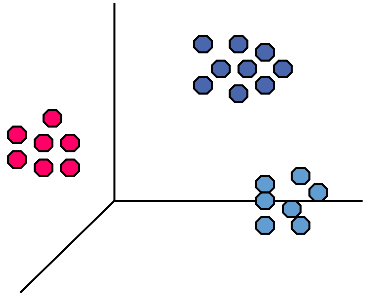
Regression



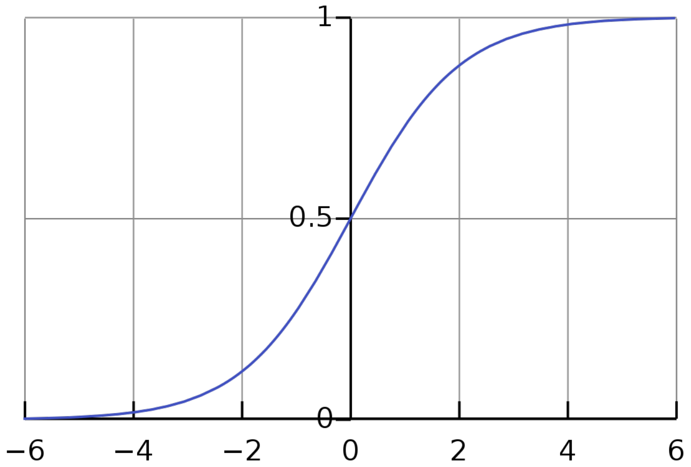
Classification

Unsupervised Learning: discovering structure

Given data points $\{X_1, \ldots, X_n\}$, learn the underlying structure within data



Clustering



Logistic Regression

- Resulting model reflects class probability:

$$h(\mathbf{x};\mathbf{w}) = P(Class = 1|\underbrace{\mathbf{x} = [x_1, \dots, x_d]}_{\text{a data point with } d \text{ features}}) = \frac{1}{1 + e^{-(\mathbf{w}^T\mathbf{x})}}$$

$$\mathbf{w}^T\mathbf{x} = w_0 + w_1 x_1 + \cdots + w_d x_d$$

- If $P(Class = 1|\mathbf{x}) \geq 0.5$, then Class label is 1;
- If $P(Class = 1|\mathbf{x}) < 0.5$, then Class label is 0.

Logistic Regression

$$h(\mathbf{x};\mathbf{w}) = P(Class = 1|\mathbf{x} = [x_1, \ldots, x_d]) = \frac{1}{1 + e^{-(\mathbf{w}^T\mathbf{x})}}$$

- Assume the class target follows a Bernoulli distribution

$$\boxed{\begin{aligned} P(y = 1|\mathbf{x};\mathbf{w}) &= h(\mathbf{x};\mathbf{w}) = \frac{1}{1 + e^{-(\mathbf{w}^T\mathbf{x})}} \\ P(y = 0|\mathbf{x};\mathbf{w}) &= 1 - h(\mathbf{x};\mathbf{w}) \end{aligned}}$$

$$\Downarrow$$

$$p(y|\mathbf{x};\mathbf{w}) = (h(\mathbf{x};\mathbf{w}))^y(1 - h(\mathbf{x};\mathbf{w}))^{1-y}$$

Logistic Regression

$$\mathbf{w}^* = \operatorname{argmax}_{\mathbf{w}} \log p(\mathbf{y}|\mathbf{X};\mathbf{w}) = \operatorname{argmax}_{\mathbf{w}} \sum_{i=1}^{n} \log(p(y_i|\mathbf{x}_i;\mathbf{w}))$$

$$= \operatorname{argmax}_{\mathbf{w}} \sum_{i=1}^{n} [y_i \log\big(h(\mathbf{x}_i;\mathbf{w})\big) + (1-y_i)\log(1-h(\mathbf{x}_i;\mathbf{w}))]$$

Maximum Likelihood Estimation (MLE)

$$\mathbf{w}^* = \operatorname{argmin}_{\mathbf{w}} -\log p(\mathbf{y}|\mathbf{X};\mathbf{w})$$

Negative log-likelihood is a **cross entropy loss function** $L(\mathbf{w}) = -\log p(\mathbf{y}|\mathbf{X};\mathbf{w})$

Gradient Descent

For iteration t:

$$w_j^{(t+1)} = w_j^{(t)} - \eta \frac{\partial}{\partial w_j}(L(\mathbf{w})) = w_j^{(t)} + \eta \frac{\partial}{\partial w_j} \log p(\mathbf{y}|\mathbf{X};\mathbf{w})$$

Bayesian classifiers

- Compute the posterior probability $P(C_j|A_1, A_2, \cdots, A_d)$ for all class C_j using the Bayes rule

$$P\left(C_j \middle| A_1, A_2, \cdots, A_d\right) = \frac{P\left(A_1, A_2, \cdots, A_d \middle| C_j\right) P(C_j)}{P(A_1, A_2, \cdots, A_d)}$$

- Choose class C_j that maximizes $P(C_j|A_1, A_2, \cdots, A_d)$

- Equivalent to choosing class C_j that maximizes

$$P\left(C_j \middle| A_1, A_2, \cdots, A_d\right) \propto P\left(A_1, A_2, \cdots, A_d \middle| C_j\right) P(C_j)$$

Naïve Bayes classifier

- Assume independence among features A_i when class is given:

$$P(A_1, A_2, \cdots, A_d | C_j) = P(A_1 | C_j) P(A_2 | C_j) \cdots P(A_d | C_j) = \prod_{i=1}^{d} P(A_i | C_j)$$

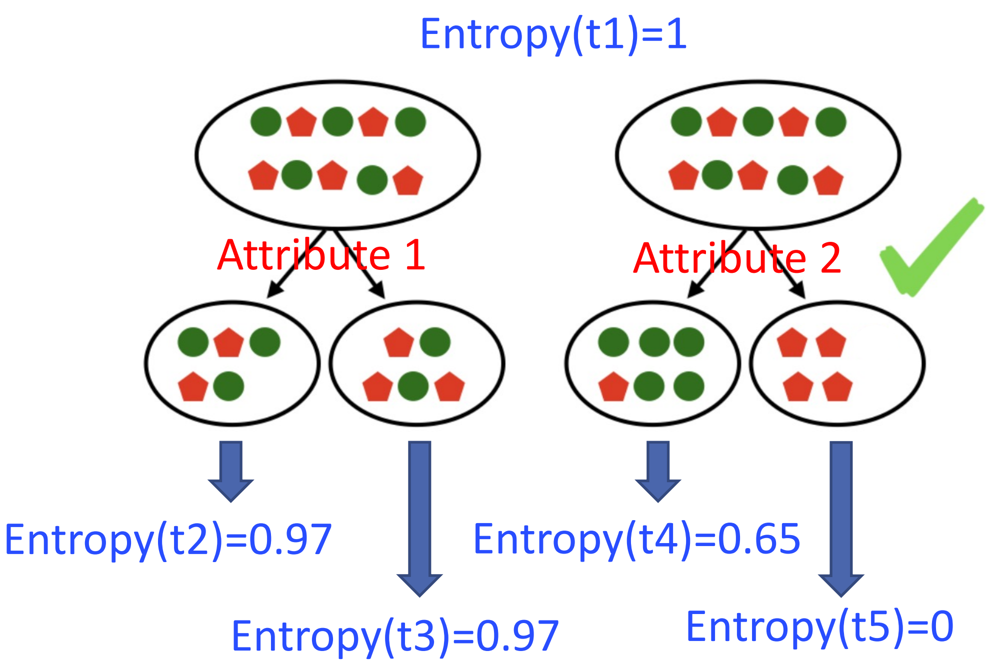
- A new sample is classified to C_j that maximizes

$$P(C_j | A_1, A_2, \cdots, A_d) \propto P(C_j) \prod_{i=1}^{d} P(A_i | C_j)$$

- Assume a distribution on feature (categorical/continuous)
- Estimate these probabilities

Decision Tree

- Key: select the best feature to split
  - Well separate the records, such that the resulting two subsets are purer
  - Impurity: GINI impurity, entropy
  - Choose the split that achieves the most impurity reduction (maximizes GAIN)
  - Reduction in impurity measures the gain of information due to the split



$$GAIN_{split} = Impurity(t_0) - \left(\sum_{i=1}^{k} \frac{n_i}{n} Impurity(t_i) \right)$$

Classification Evaluation

	PREDICTED CLASS	
ACTUAL CLASS	Class=Yes	Class=No
Class=Yes	a (TP)	b (FN)
Class=No	c (FP)	d (TN)

$$\text{Accuracy} = \frac{\#\,correctly\ classified\ records}{\#\,records}$$

$$\text{Precision (p)} = \frac{a}{a + c}$$

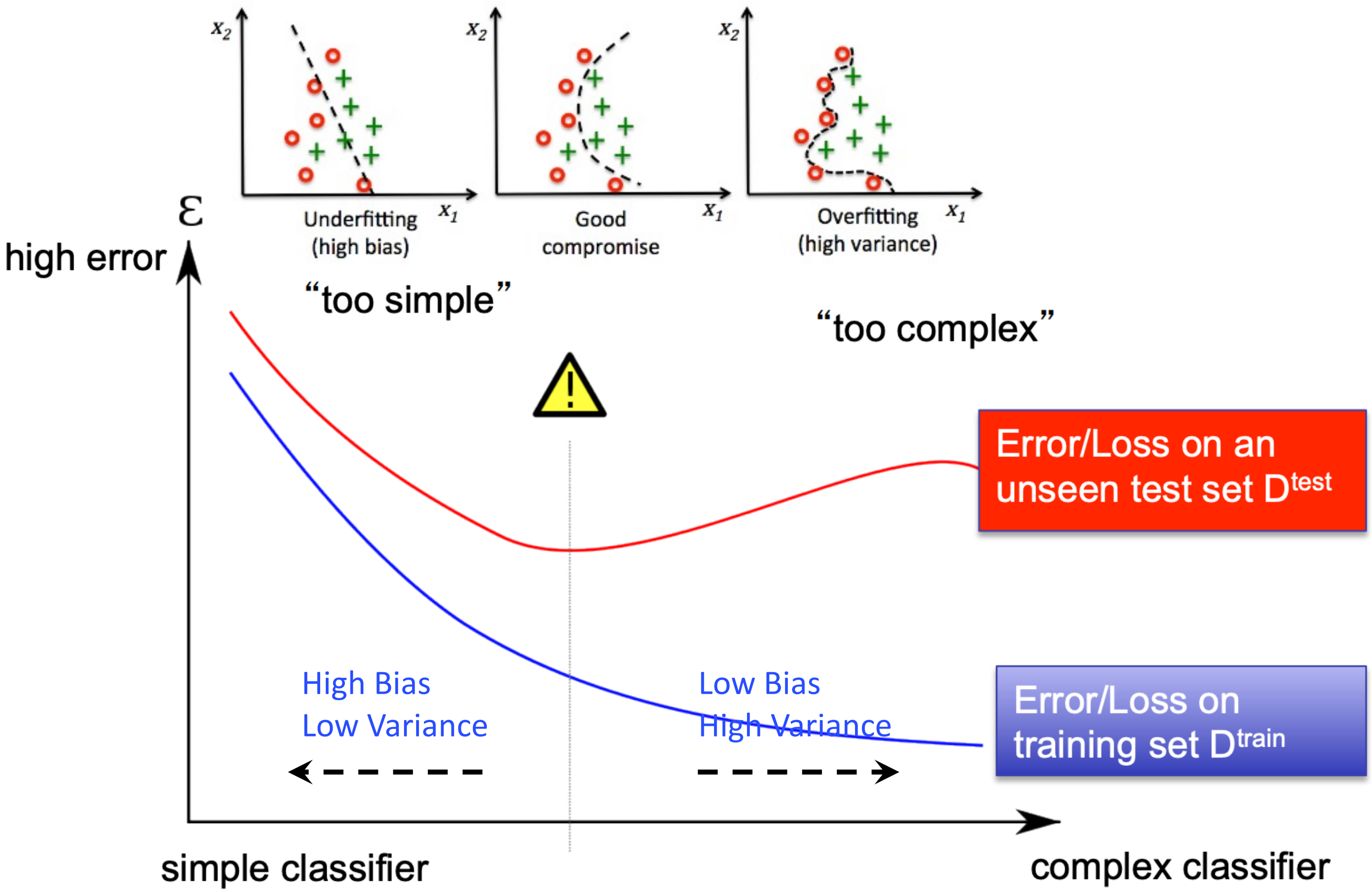
$$\text{Recall (r)} = \frac{a}{a + b}$$

$$\text{F-measure (F)} = \frac{2rp}{r + p} = \frac{2a}{2a + b + c}$$

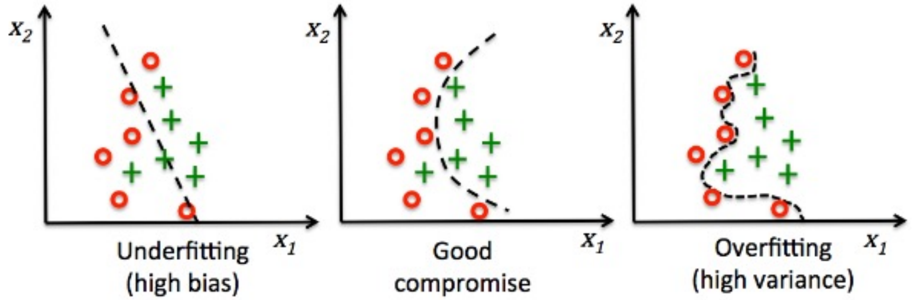
Q: which metric should be used to evaluate a classifier for detecting cancer?

Recall, because FN could lead to a big risk

Evaluation on training data is not enough

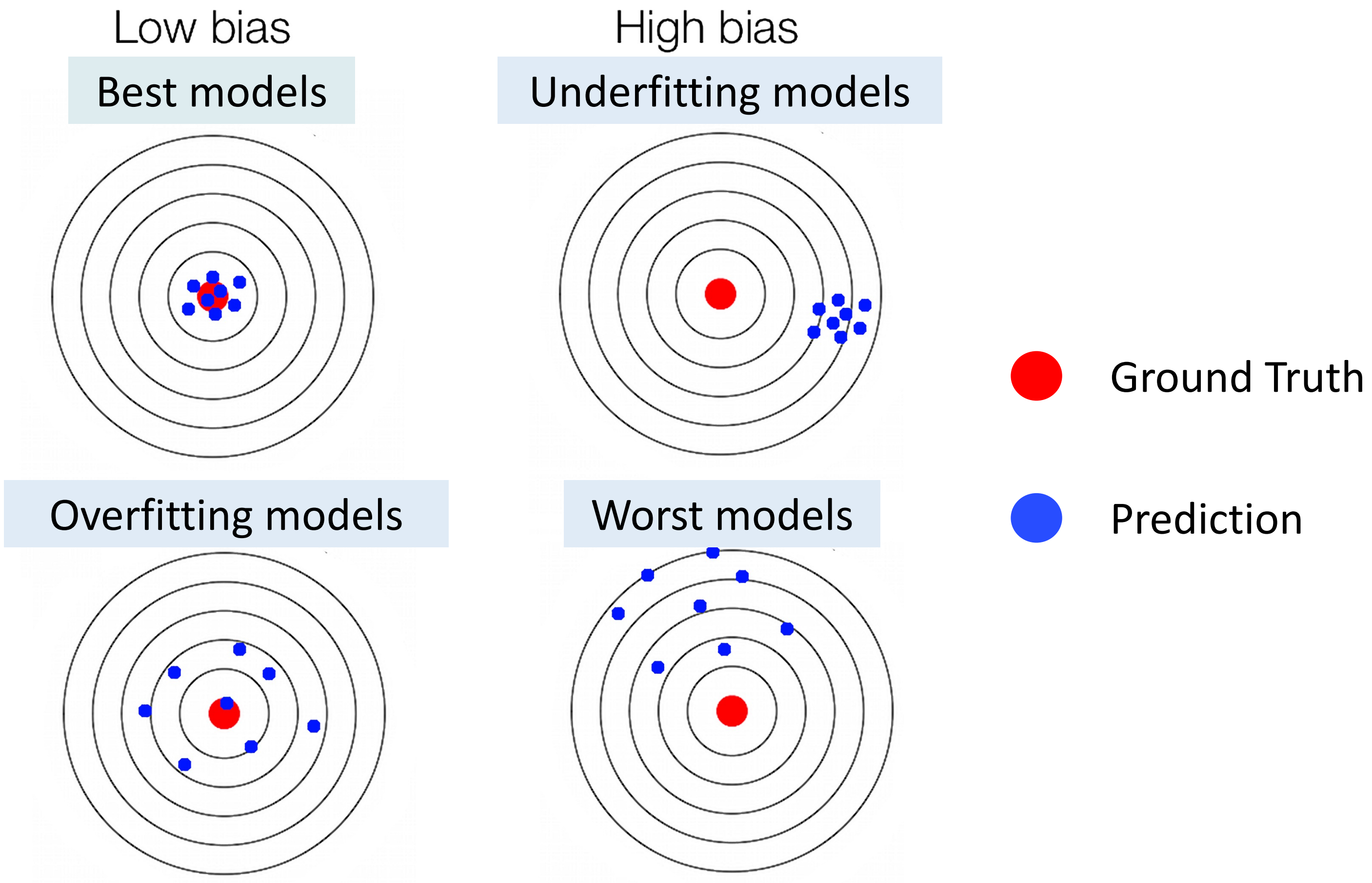


Bias-variance trade-off



Stable Models → Low variance

Unstable Models → High variance



Ensemble method

- Advantage
  - Often reduces variance by using simple base models on diverse data
  - Often reduces bias by improving predictive performance
- Type
  - Bagging
  - Boosting
  - Random forest (bagging + random feature set)

Bagging (Bootstrap Aggregation)

This approach would be helpful when your model is complex, easy to overfit, such as decision tree.

- Bootstrap: generating new datasets of size *n* by sampling from the original dataset *with replacement*
  - One sampled dataset is called bootstrap sample
- Aggregation: combining predictions by voting/averaging
  - Training model on each bootstrap sample
  - Each model receives equal weight
  - Can be applied to regression (AVG) and classification (VOTING)

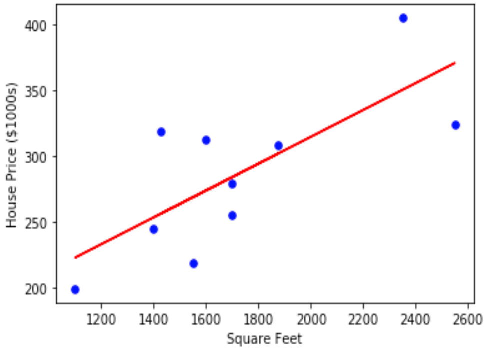
Boosting

- Train classifiers in a sequence
- A new classifier should focus on those cases which were misclassified in the previous round — hard examples
- Combine the classifiers on the final prediction (like bagging)
  - But each base classifier has a weight
- Iterative: new models are influenced by performance of previously built ones
  - Encourage new model to become an “expert” for instances misclassified by earlier models
  - Intuitive justification: models should be experts that complement each other

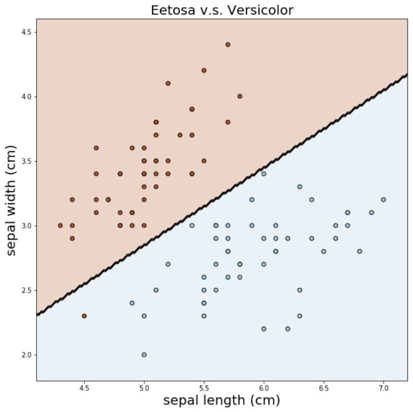
ML Basics Review

Supervised Learning: making prediction

Given data points $\{(X_1, Y_1), \dots, (X_n, Y_n)\}$, learn a model to predict Y_i from X_i



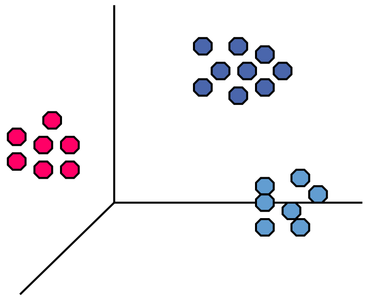
Regression



Classification

Unsupervised Learning: discovering structure

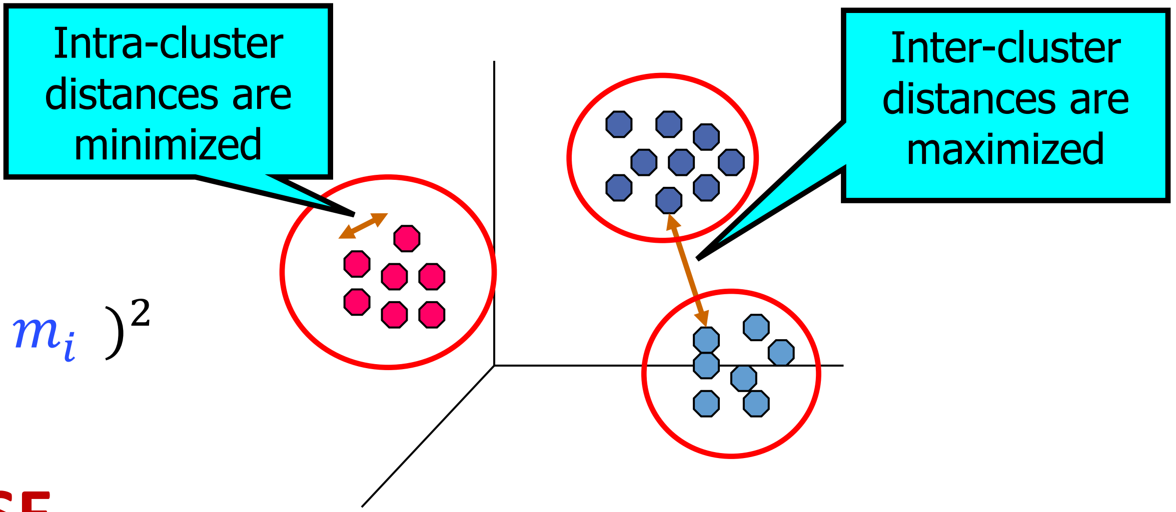
Given data points $\{X_1, \dots, X_n\}$, learn the underlying structure within data



Clustering

K-means Clustering

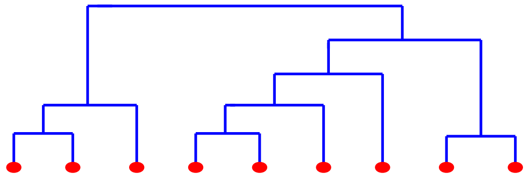
$$\min \sum_{i} \sum_{x \in C_i} (x - m_i)^2$$



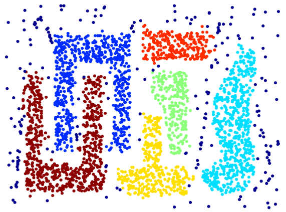
- **Two sets of variables to minimize SSE**
  - Each instance *x* belongs to which cluster?
  - What's the cluster centroid m_i ?
- **Iterative update**
  - Fix the cluster centroid -- find cluster assignment that minimizes the current error
  - Fix the cluster assignment -- compute the cluster centroids that minimize the current error

K-means Clustering

- **Strength**
  - Efficient: O($n*k*t$), where n is #objects, k is #clusters, and t is # iterations
  - Easy to implement



- **Issues**
  - Need to preset K, the number of clusters ← Hierarchical clustering
  - Local minimum– sensitive to initialization ← Multiple rounds
  - Could be sensitive to outliers ← K-medoids
  - May not well handle clusters with different densities or irregular shapes



← Density-based clustering