

Explainable Machine Learning

# Interpretable Models

Shim Jaewoong

*jaewoong@seoultech.ac.kr*

# Machine Learning

---

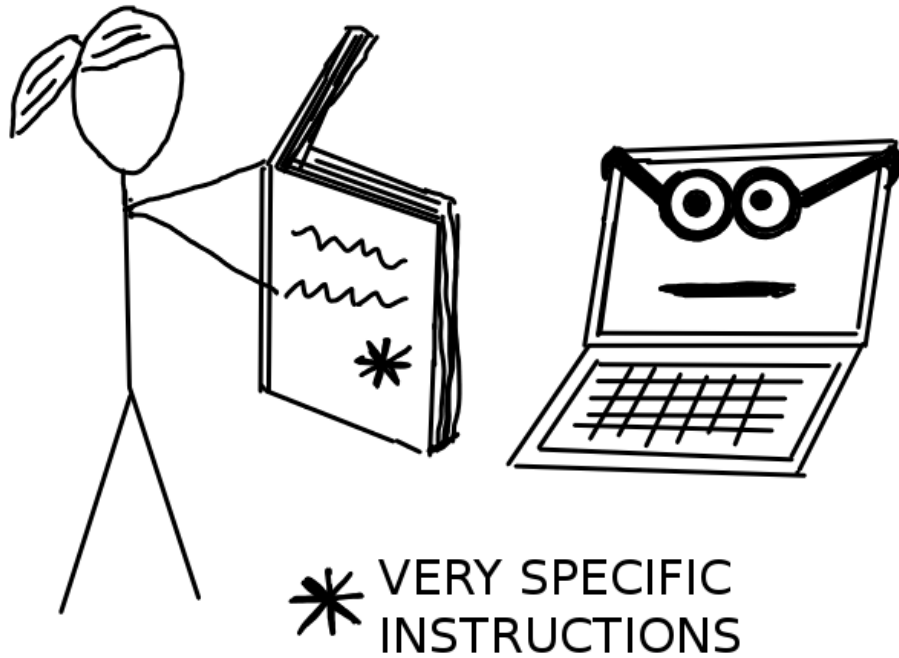
# Machine Learning

- **Tom Mitchell (Professor at CMU)'s definition of "learning"**
  - A computer program is said to **learn** from **experience E**
  - with respect to some class of **tasks T**
  - and **performance measure P**,
  - if its performance at tasks in T, as measured by P,
  - improves with E.
- A "**machine learning algorithm**" is an algorithm that is able to **learn from data**
  - Machine Learning = Learning from **E** in form of **Data**

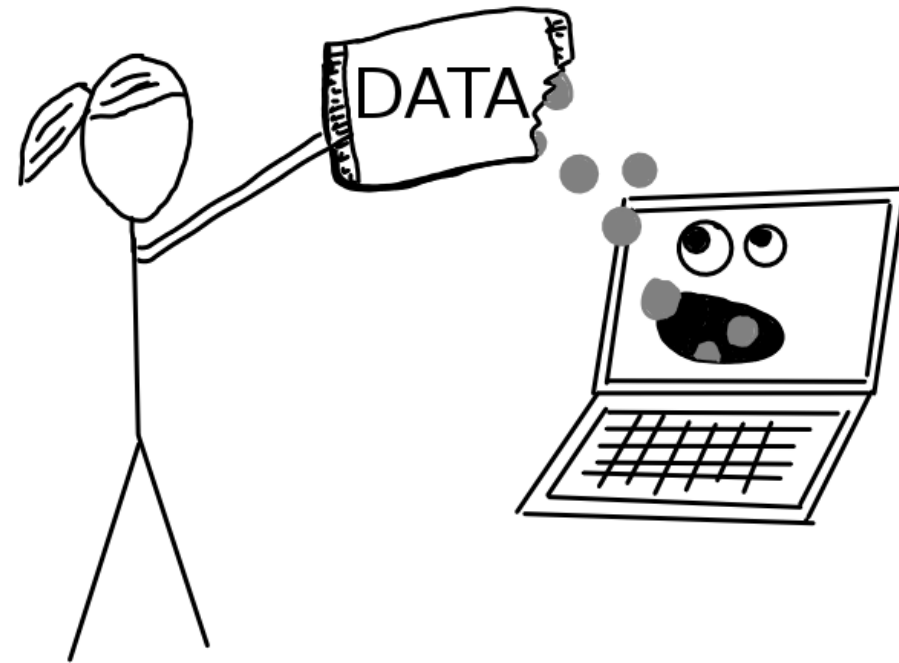


# Machine Learning

## Without Machine Learning

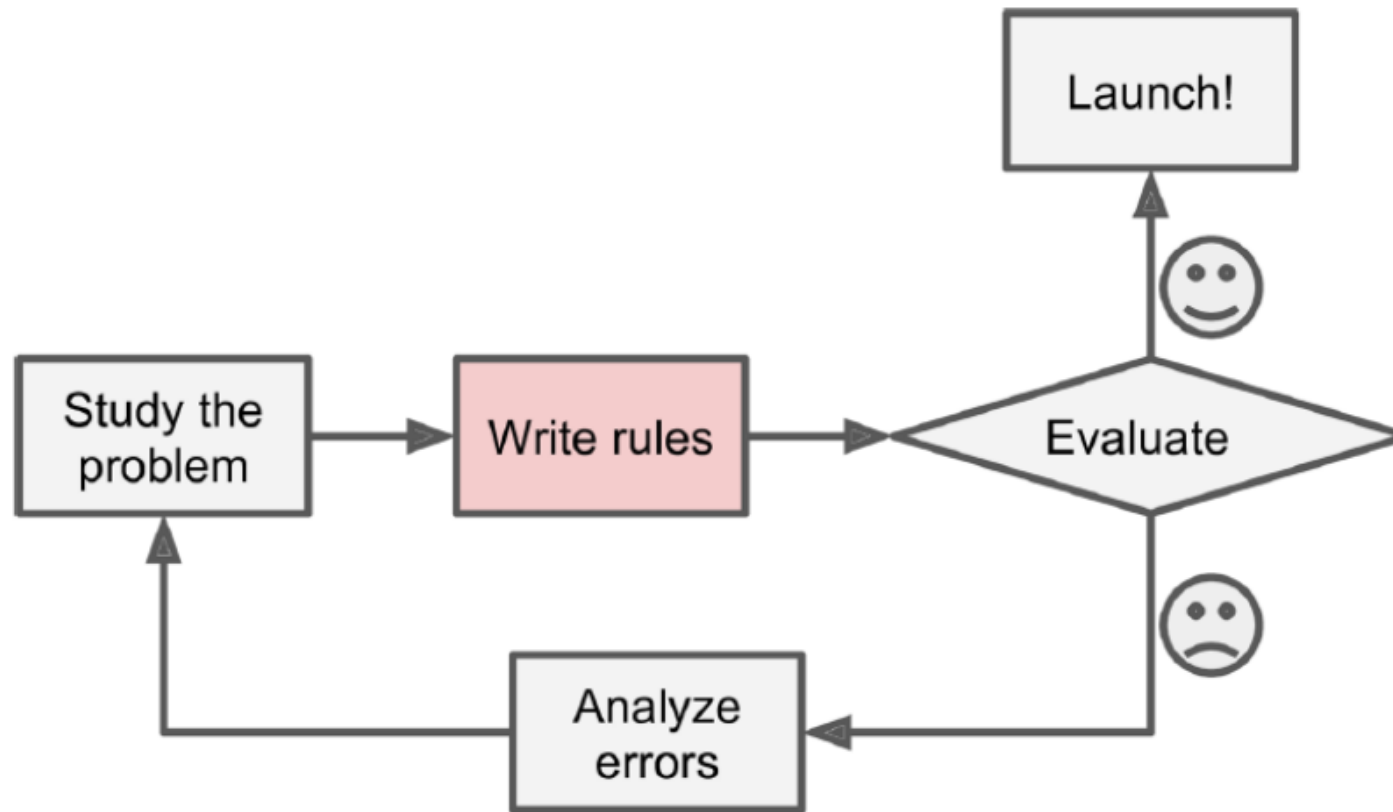


## With Machine Learning



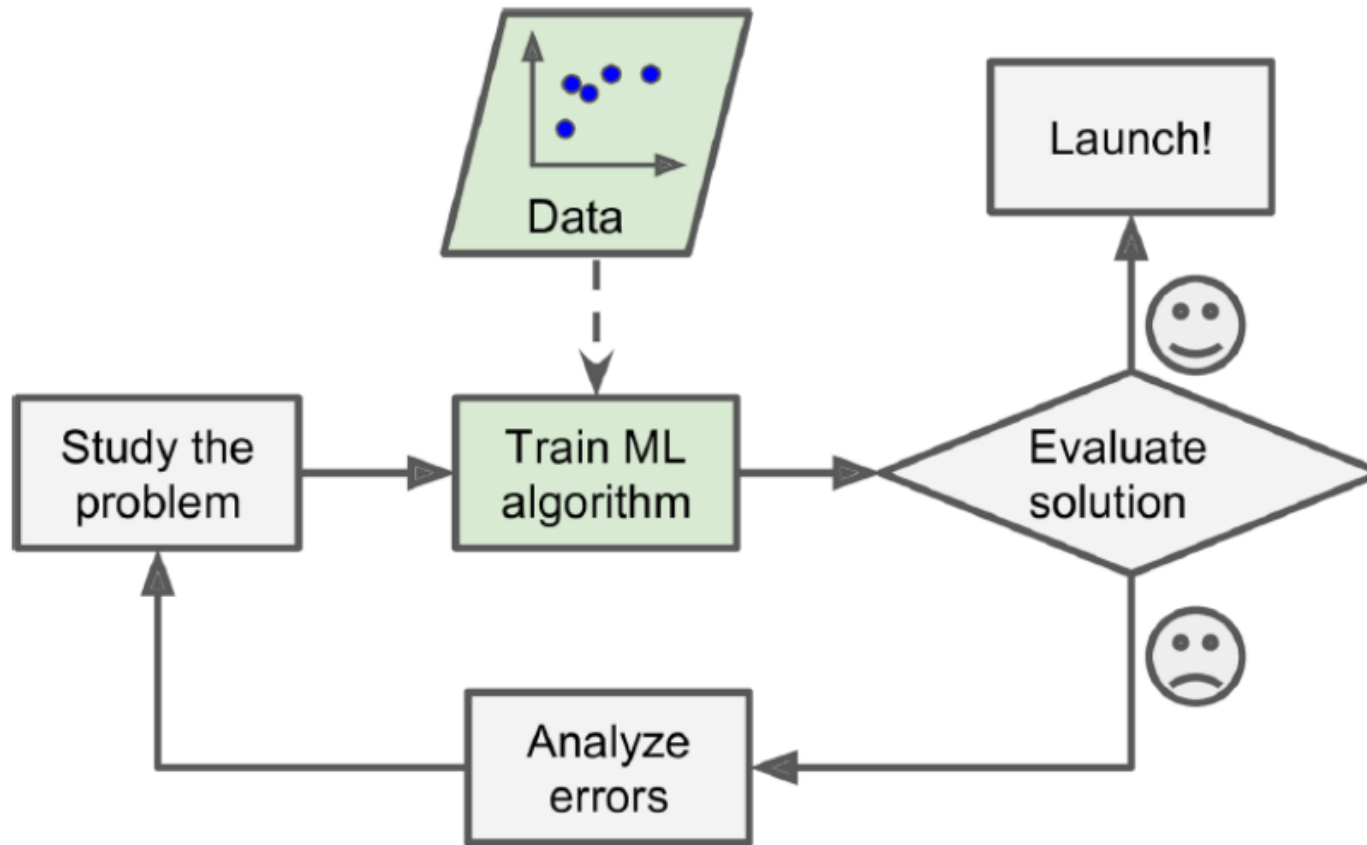
# Traditional Learning

- Expert (Rule-Based) Systems – Explicit knowledge, “Human-driven”
  - Since the problem is not trivial, your program will likely become a long list of complex rules, thus pretty hard to maintain



# Machine Learning

- Machine Learns from Data – Implicit knowledge, “Data-driven”
- The program is much shorter, easier to maintain, most likely more accurate, and applicable to problems that either are too complex for traditional approaches or have no known algorithm.



# Machine Learning

- **Supervised Learning**

- *(in general)* **Labeled dataset**  $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_n, y_n)\}$ ,  
where each  $\mathbf{x}_i \in R^d$  contains  $d$  features  $\mathbf{x}_i = (x_{i1}, \dots, x_{id})$  and is associated with a label  $y_i$ .
- To find the functional relationship between input and output  $\hat{y} = f(\mathbf{x})$  from the dataset
  - Input:  $\mathbf{x} = (x_1, \dots, x_d)$  (vector of  $d$  features)
  - Output:  $y$  (label)
- To use feature values to predict an unknown label.
- *e.g.*, classification (categorical label), regression (continuous-valued label), etc...

- **Unsupervised Learning**

- *(in general)* **Unlabeled dataset**  $D = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ ,  
where each  $\mathbf{x}_i \in R^d$  contains  $d$  features  $\mathbf{x}_i = (x_{i1}, \dots, x_{id})$
- To find useful properties/patterns of the structures of the dataset
- *e.g.*, clustering, density estimation, one-class classification, association analysis, etc...

# Machine Learning

- Supervised Learning – Labeled Dataset

- *important to have a representation of data that a computer can understand*

**Column:** variable, feature, attribute, ...

**Input**

**Output**

**Row:**

data point,  
instance,  
example,  
record,  
pattern,  
object,  
...

id	$X_1$	$X_2$	$X_3$	...	$X_d$	$Y$
1	$x_{11}$	$x_{12}$	$x_{13}$	...	$x_{1d}$	$y_1$
2	$x_{21}$	$x_{22}$	$x_{23}$	...	$x_{2d}$	$y_2$
3	$x_{31}$	$x_{32}$	$x_{33}$	...	$x_{3d}$	$y_3$
4	$x_{41}$	$x_{42}$	$x_{43}$	...	$x_{4d}$	$y_4$
5	$x_{51}$	$x_{52}$	$x_{53}$	...	$x_{5d}$	$y_5$
6	$x_{61}$	$x_{62}$	$x_{63}$	...	$x_{6d}$	$y_6$
7	$x_{71}$	$x_{72}$	$x_{73}$	...	$x_{7d}$	$y_7$
...	...	...	...	...	...	...



# Machine Learning

- **Unsupervised Learning – Unlabeled Dataset**

- *important to have a representation of data that a computer can understand*

**Column:** variable, feature, attribute, ...

**Input**

**Output**

**Row:**

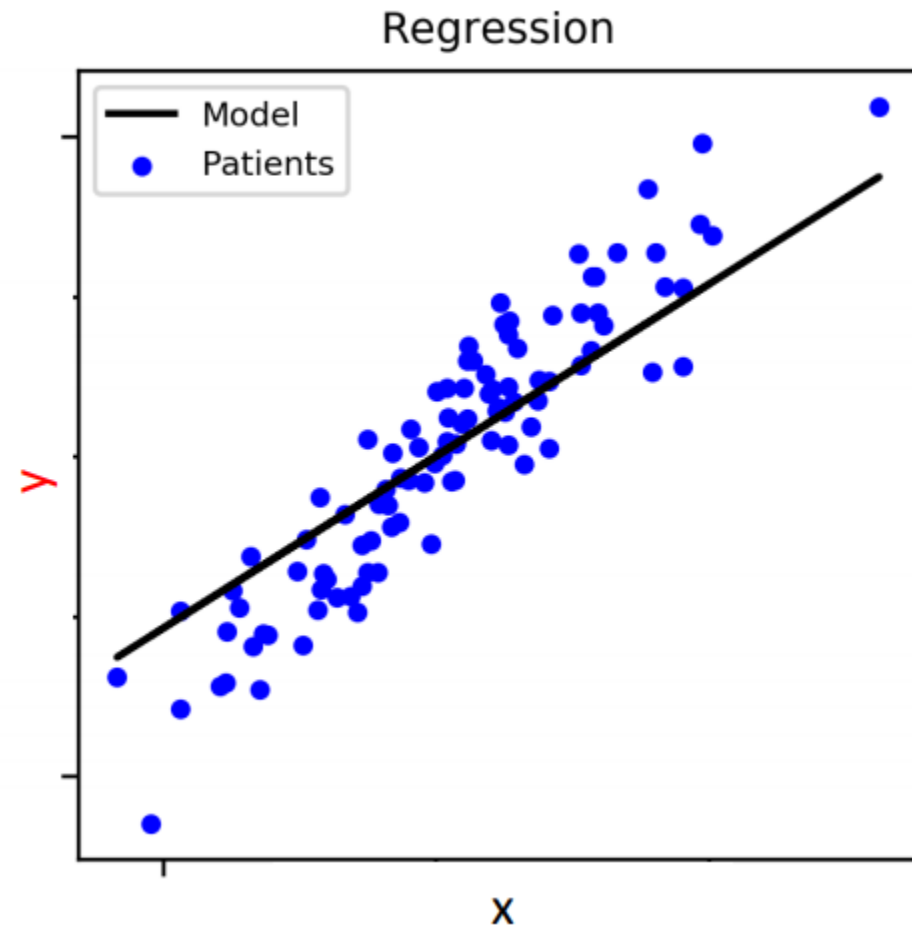
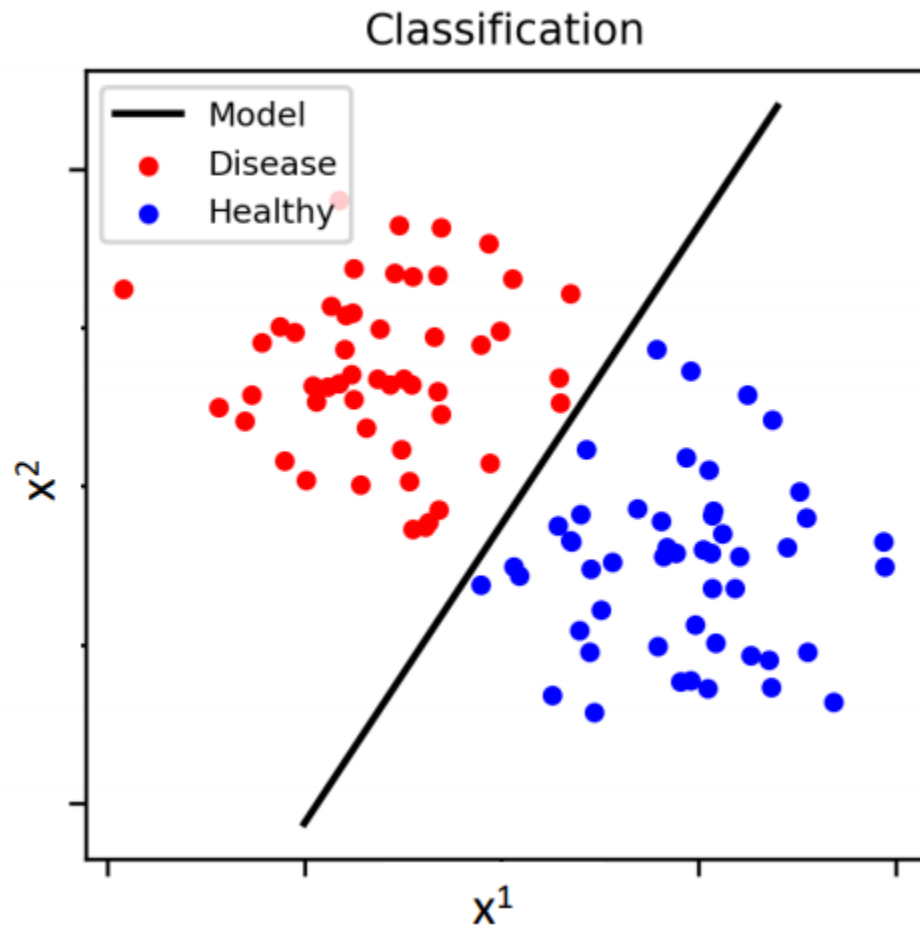
data point,  
instance,  
example,  
record,  
pattern,  
object,  
...

id	$X_1$	$X_2$	$X_3$	...	$X_d$
1	$x_{11}$	$x_{12}$	$x_{13}$	...	$x_{1d}$
2	$x_{21}$	$x_{22}$	$x_{23}$	...	$x_{2d}$
3	$x_{31}$	$x_{32}$	$x_{33}$	...	$x_{3d}$
4	$x_{41}$	$x_{42}$	$x_{43}$	...	$x_{4d}$
5	$x_{51}$	$x_{52}$	$x_{53}$	...	$x_{5d}$
6	$x_{61}$	$x_{62}$	$x_{63}$	...	$x_{6d}$
7	$x_{71}$	$x_{72}$	$x_{73}$	...	$x_{7d}$
...	...	...	...	...	...

**X**

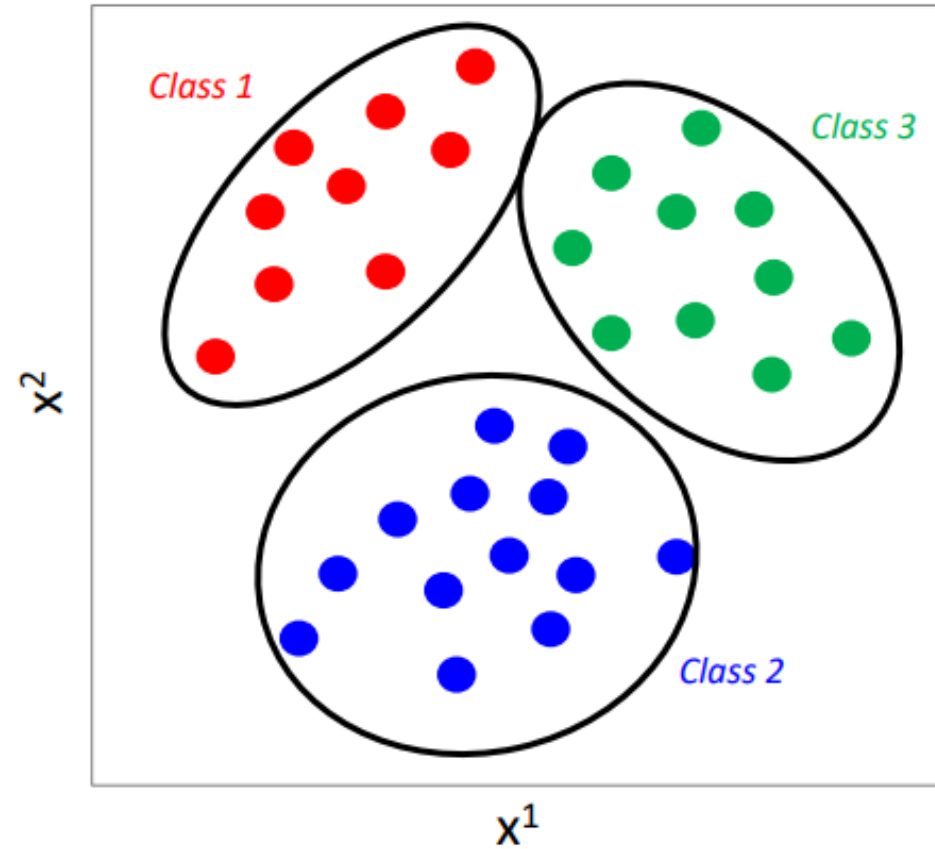
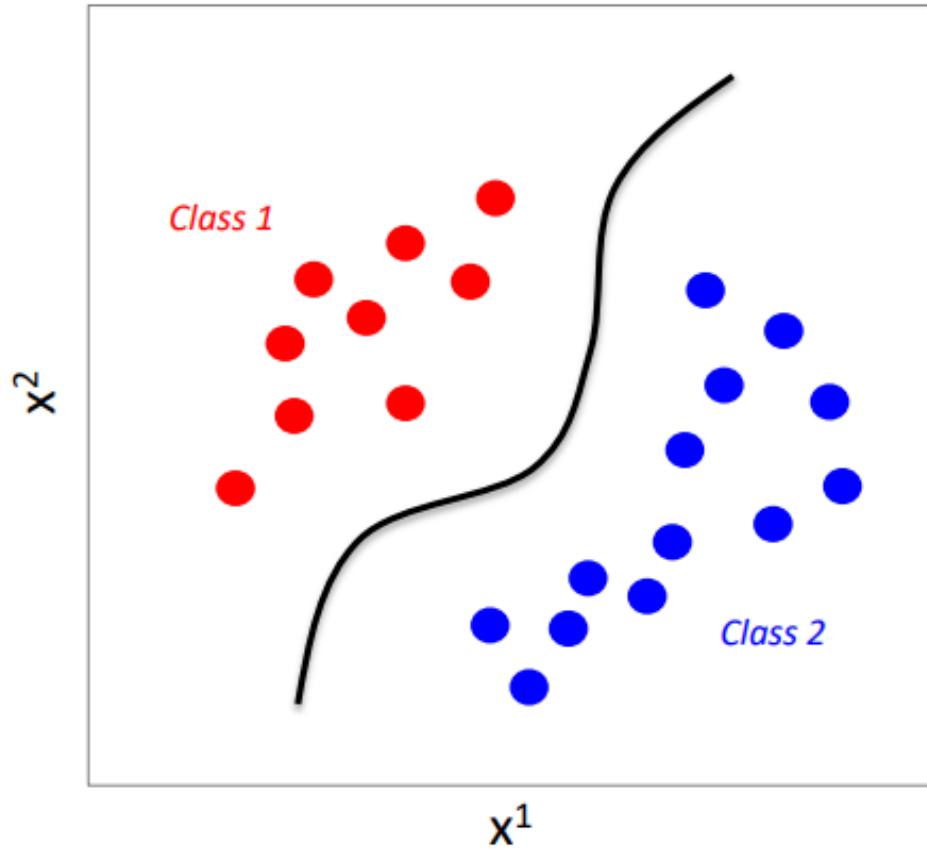
# Supervised Learning

- Classification vs Regression



# Supervised Learning

- Binary vs Multi-class Classification



# Supervised Learning

- Multi-label Classification



**“Dog”**



**“Cat”**



**?**

# Supervised Learning

## ■ Multi-label Classification

### Binary Classification



- Spam
- Not spam

### Multiclass Classification



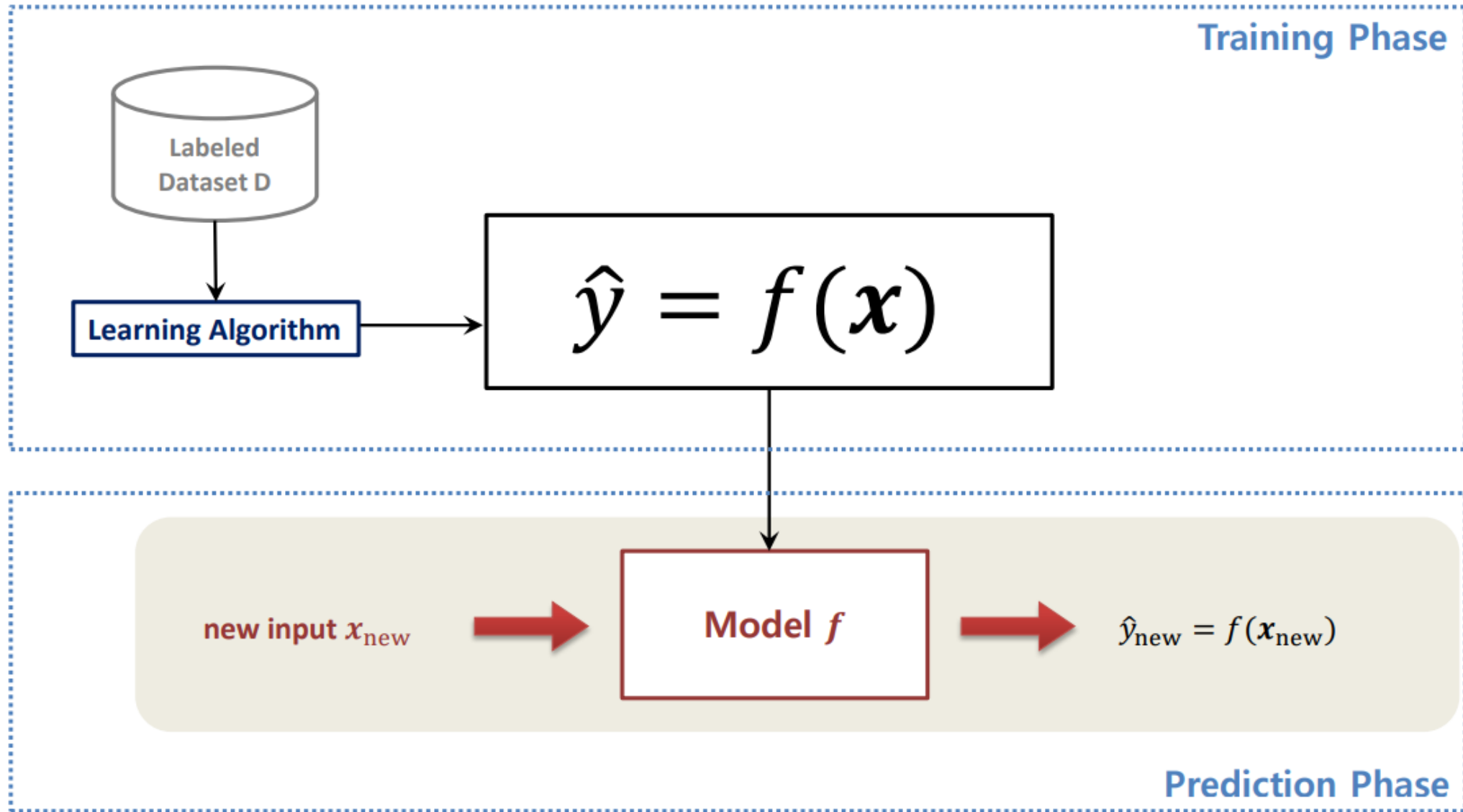
- Dog
- Cat
- Horse
- Fish
- Bird
- ...

### Multi-label Classification

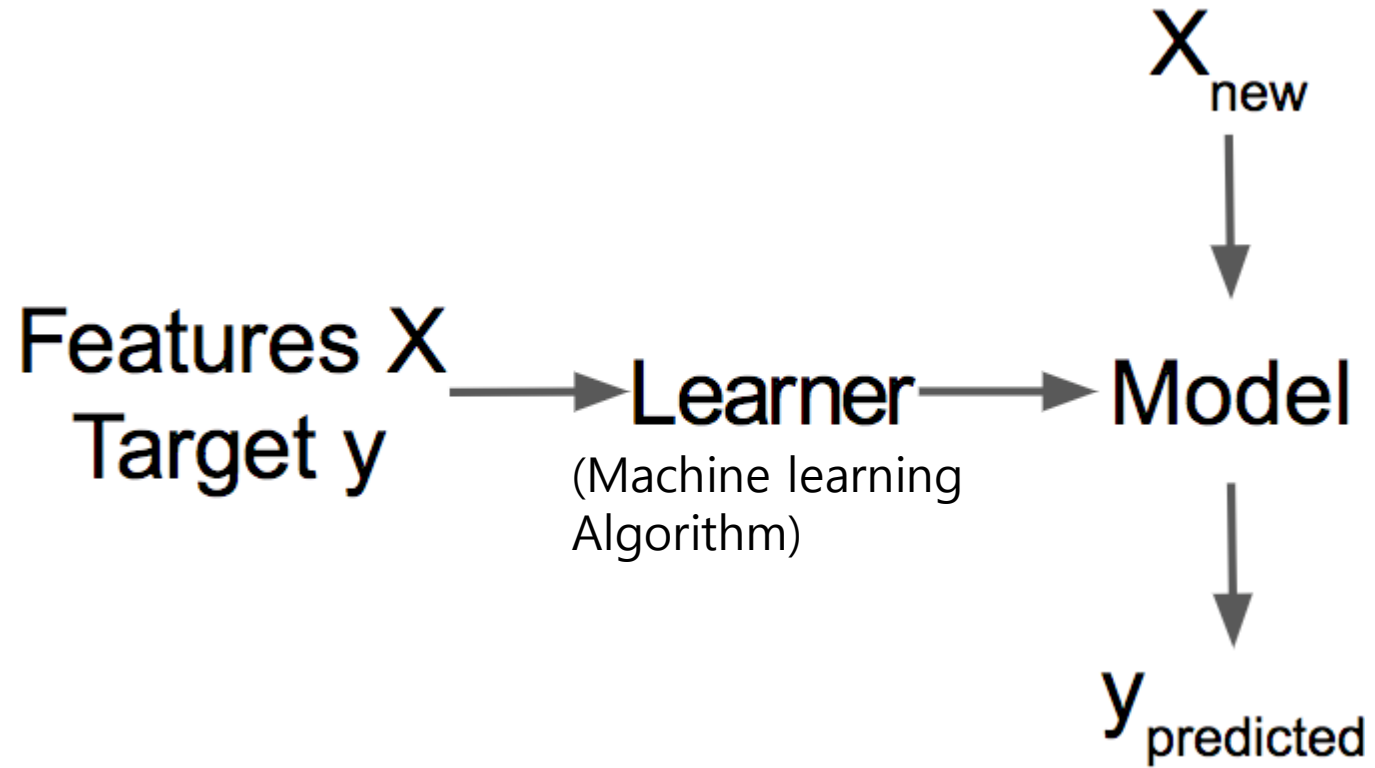


- Dog
- Cat
- Horse
- Fish
- Bird
- ...

# Supervised Learning



# Supervised Learning



Neural Network?

Backpropagation?

Stochastic gradient decent?

# Supervised Learning Algorithms

---

## ▪ Regression

- Linear Regression
- K-Nearest Neighbors
- Decision Tree
- Neural Network
- Support Vector Regression
- ...

## ▪ Classification

- Logistic Regression
- K-Nearest Neighbors
- Decision Tree
- Neural Network
- Support Vector Machine
- ...



# Interpretability

---

# Interpretability

**explanation** | ɛksplə'neɪʃ(ə)n |

noun

a statement or account that makes something clear: *the birth rate is central to any explanation of population trends.*

Oxford Dictionary of  
English

**interpret** | ɪn'tɜːprɪt |

verb (**interprets, interpreting, interpreted**) [*with object*]

1 explain the meaning of (information or actions): *the evidence is difficult to interpret.*

Interpretable machine learning is a useful umbrella term that captures the “extraction of relevant knowledge from a machine-learning model concerning relationships either contained in data or learned by the model”

# Importance of Interpretability

- Need for Interpretation
  - **Human curiosity and learning**
    - For most people it is okay that they do not understand how a computer works. **Unexpected events makes us curious.** For example: Why is my computer shutting down unexpectedly?
    - If a machine learning model rejects a loan application, this may be completely unexpected for the applicants. They can only reconcile this **inconsistency between expectation and reality** with some kind of explanation.
    - I always think about why certain products or movies have been algorithmically recommended to me.
  - **Gain more knowledge**
    - **The model itself becomes the source of knowledge instead of the data.** Interpretability makes it possible to extract this additional knowledge captured by the model.
  - Safety
  - Debugging for detecting bias
  - Social acceptance
  - ....



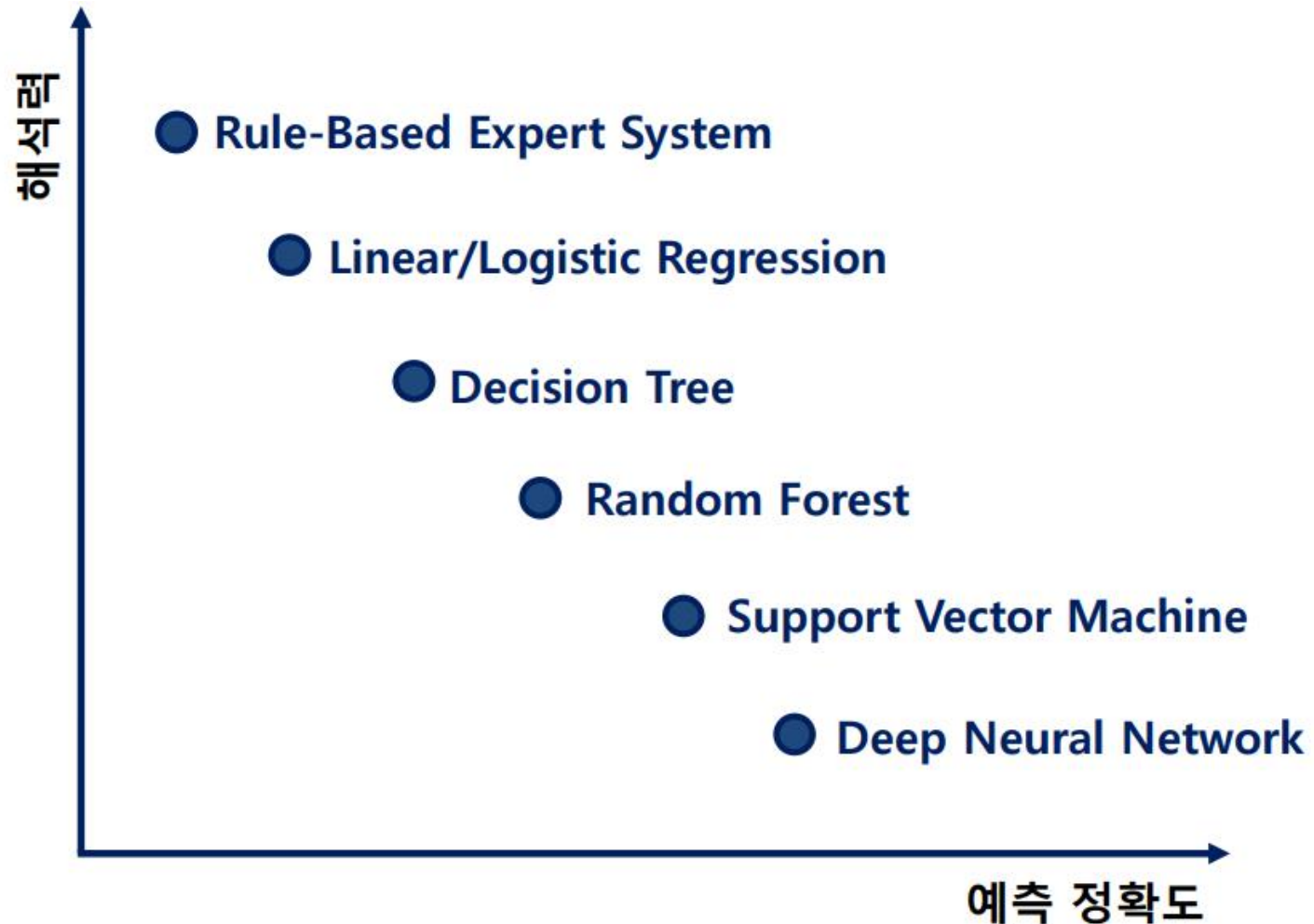
- Fairness
- Privacy
- Reliability or Robustness
- Causality
- Trust

# Scope of Interpretability

- **Algorithm Transparency**
  - *How does the algorithm create the model?*
- **Global, Holistic Model Interpretability**
  - *How does the trained model make predictions?*
- **Global Model Interpretability on a Modular Level**
  - *How do parts of the model affect predictions?*
- **Local Interpretability for a Single Prediction**
  - *Why did the model make a certain prediction for an instance?*
- **Local Interpretability for a Group of Predictions**
  - *Why did the model make specific predictions for a group of instances?*

# Interpretability vs Accuracy

- 일반적으로, 복잡한 구조의 모델이 높은 예측 정확도 달성이 가능하나, 해석력이 낮음



〈 주요 인공지능 모델의 예측 정확도와 해석력 비교 〉

# Interpretable Models: **Linear Regression**

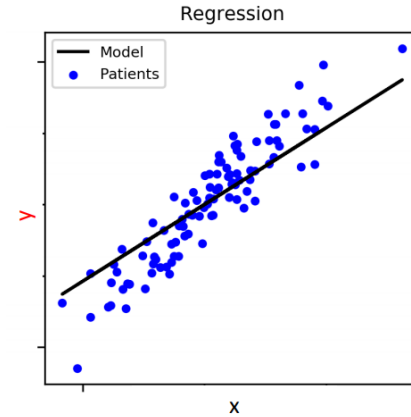
---

# Linear Models

- Linear models make a prediction using a linear function of the input features

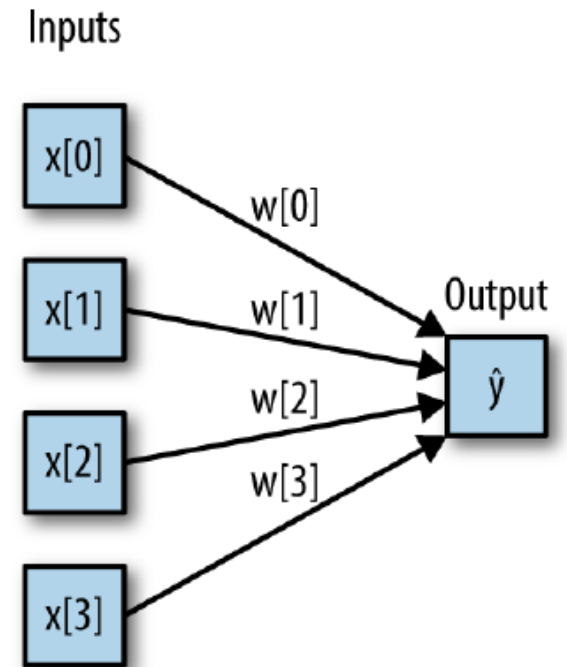
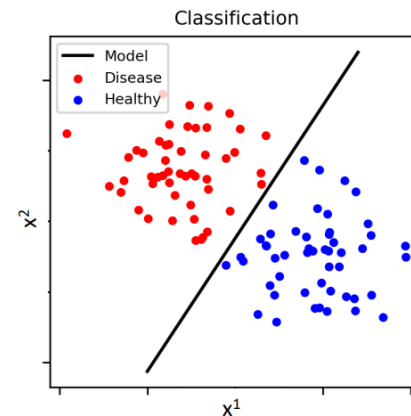
- Learning algorithms for regression**

- Linear Regression
- Ridge Regression
- Lasso Regression
- Elastic Net Regression
- Principal Component Regression
- Partial Least Squares Regression
- (Linear) Support Vector Regression
- ...



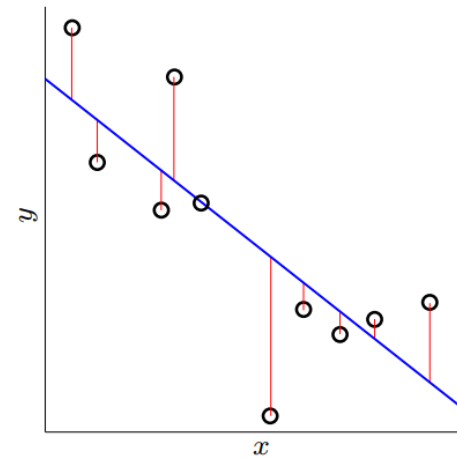
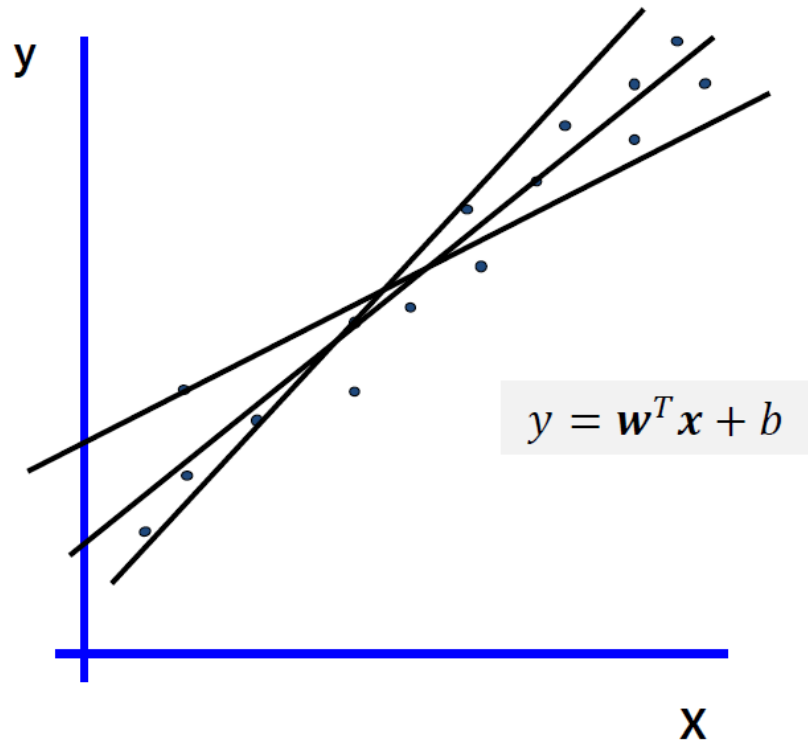
- Learning algorithms for binary classification**

- Logistic Regression
- (Linear) Support Vector Machine
- Linear Discriminant Analysis
- ...

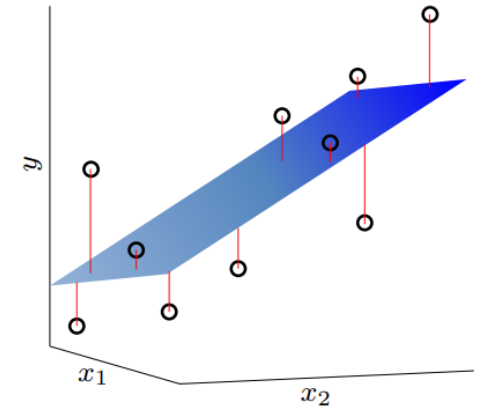


# Linear Regression

- For linear models for **regression**, the prediction  $\hat{y}$  is a **linear** function of input features.



(a) 1D (line)

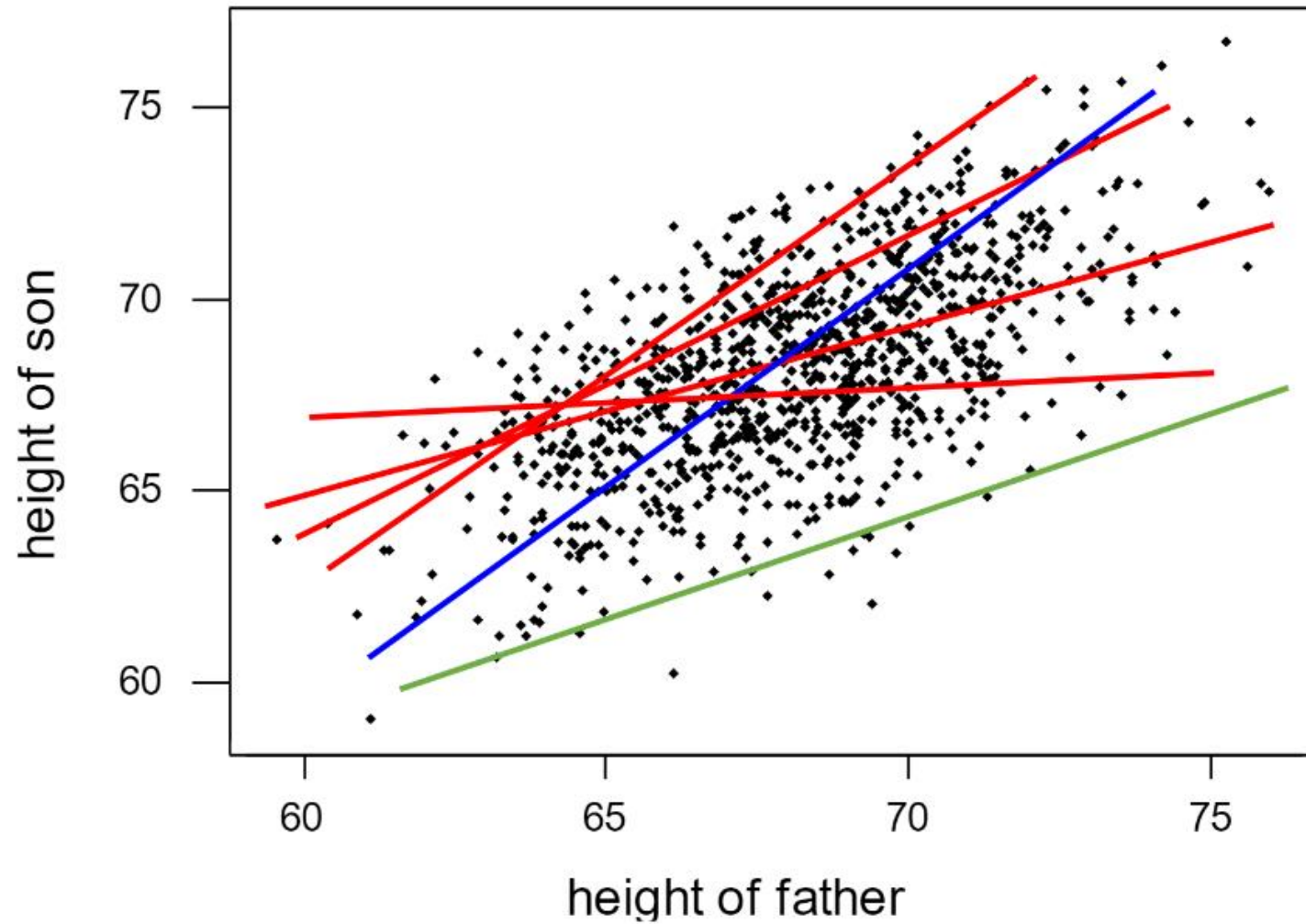


(b) 2D (hyperplane)



# Linear Regression

- Which line(hyperplane) is optimal?



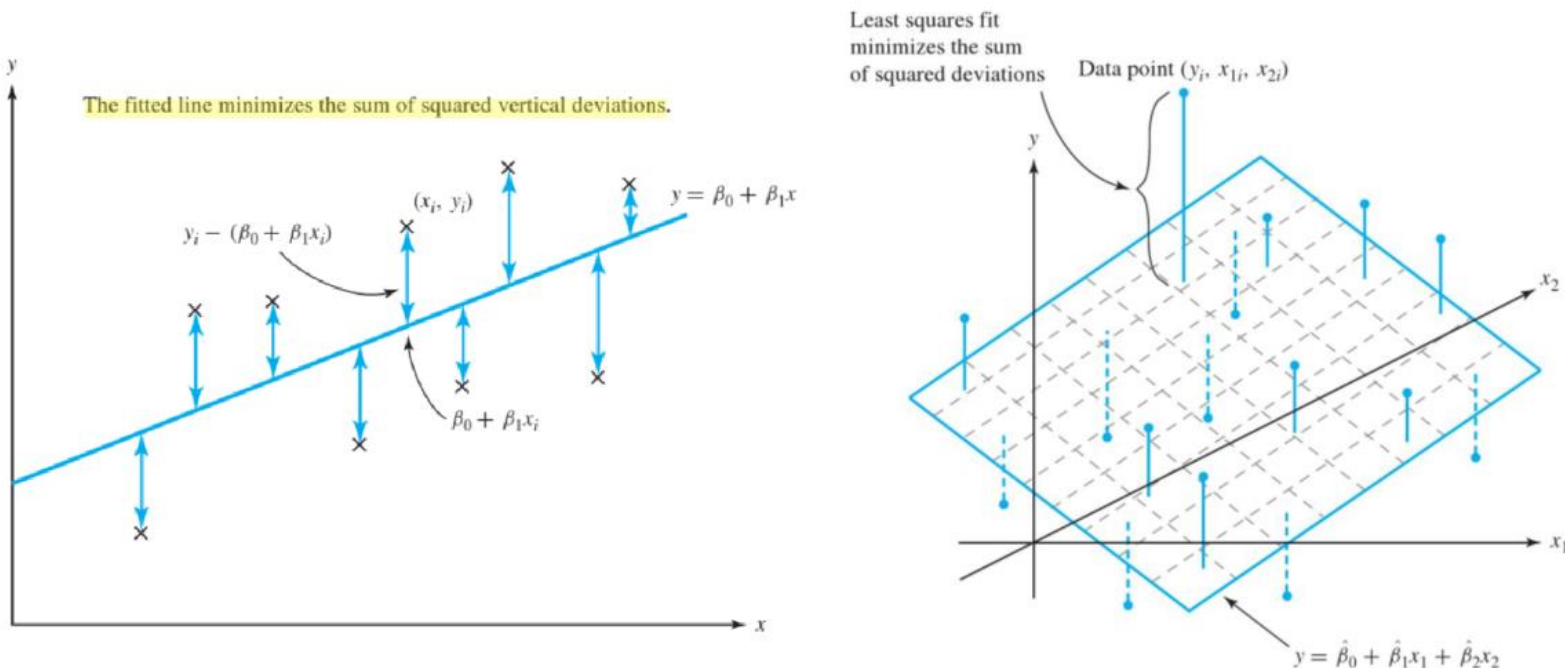
# Ordinary Least Squares

- **Linear Regression (ordinary least squares (OLS))**

- Linear regression finds the parameters  $\mathbf{w}$  and  $b$  that minimize the *mean squared error* between predictions and the true regression targets on the training set.

$$\hat{y} = \mathbf{w}^T \mathbf{x} + b = w_1 x_1 + \dots + w_d x_d + b$$
$$\mathbf{x} = (x_1, \dots, x_d) \in \mathbb{R}^d, \quad y, \hat{y} \in \mathbb{R}$$

- Linear regression has no hyperparameters, thus has no way to control model complexity.



# Ordinary Least Squares

- Given a (training) dataset  $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_n, y_n)\}$  such that  $\mathbf{x}_i = (\mathbf{1}, x_{i1}, \dots, x_{id}) \in \mathbb{R}^{d+1}$  is the  $i$ -th input vector of  $d$  features and  $y_i \in \mathbb{R}$  is the corresponding target label.

- the first entry is always set to "1"

- The output of model  $f$  (prediction of  $y$ )  
:  $\hat{y} = f(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$ , where  $\mathbf{w} = (w_0, w_1, \dots, w_d)$  is a vector of parameters.

-  $w_1, \dots, w_d$  are called "coefficients" or "weights"  
-  $w_0$  is called "intercept" or "bias" (same as  $b$ )

# Ordinary Least Squares

- **Training:** To find the optimal parameter  $\mathbf{w}^*$  that minimizes the training error (cost function)

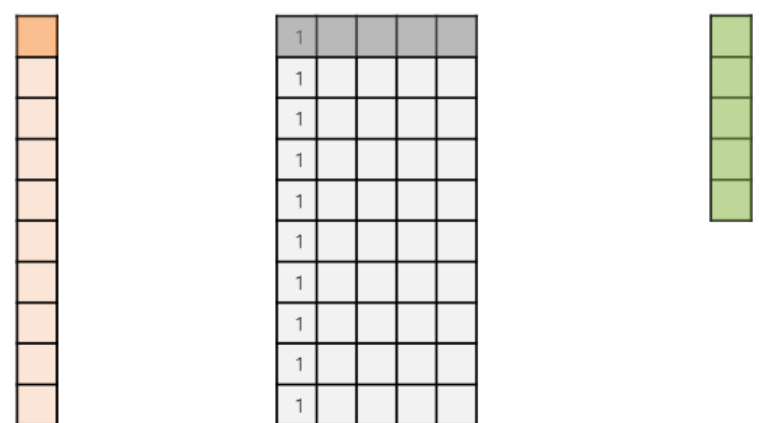
Here we use “squared error” loss  $L(y, \hat{y}) = (\hat{y} - y)^2$ , then  $J(\mathbf{w}) = \text{MSE}_{\text{train}}$

$$J(\mathbf{w}) = \frac{1}{n} \sum_{(x_i, y_i) \in D} L(y_i, \hat{y}_i) = \frac{1}{n} \sum_{(x_i, y_i) \in D} (\hat{y}_i - y_i)^2 = \frac{1}{n} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2$$

-  $\mathbf{X}$ ,  $\mathbf{y}$  are matrix representation of  $D$

$\mathbf{X} : n \times (d + 1)$  matrix,  $\mathbf{y} : n \times 1$  vector

$\mathbf{w} : (d + 1) \times 1$  vector

$$\hat{\mathbf{y}} = \mathbf{X} \times \mathbf{w}$$


# Ordinary Least Squares

- **Training:** To find the optimal parameter  $\mathbf{w}^*$  that minimizes the training error  
→ an optimization problem

$$\text{MSE}_{\text{train}} = \frac{1}{n} \sum_{(x_i, y_i) \in D} (\hat{y}_i - y_i)^2 = \frac{1}{n} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2$$

► how? set the gradient to 0 → a closed-form solution (normal equation)

$$\nabla_{\mathbf{w}} \text{MSE}_{\text{train}} = \frac{1}{n} \nabla_{\mathbf{w}} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2 = 0$$

...

...

...

$$\mathbf{w}^* = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$$

- The trained model  $f(\mathbf{x}) = \mathbf{w}^{*T} \mathbf{x}$

# Ordinary Least Squares

- Recall : Matrix derivative

$$\begin{aligned}\nabla_{\mathbf{w}}(\mathbf{w}^T A \mathbf{w}) &= (A + A^T) \mathbf{w} \\ \nabla_{\mathbf{w}}(\mathbf{w}^T \mathbf{b}) &= \mathbf{b}\end{aligned}$$

★ scalar  $w$

$$\begin{aligned}E_{\text{in}}(w) &= aw^2 - 2bw + c \\ \nabla E_{\text{in}}(w) &= 2aw - 2b\end{aligned}$$

★ vector  $\mathbf{w}$

$$\begin{aligned}E_{\text{in}}(\mathbf{w}) &= \mathbf{w}^T A \mathbf{w} - 2\mathbf{w}^T \mathbf{b} + c \\ \nabla E_{\text{in}}(\mathbf{w}) &= (A + A^T) \mathbf{w} - 2\mathbf{b}\end{aligned}$$

# Ordinary Least Squares

$$\frac{1}{n} \nabla_w \|Xw - y\|^2 = 0$$

$$\frac{1}{n} \nabla_w (w^T X^T X w - 2w^T X^T y + y^T y) = 0$$

$$\frac{2}{n} (X^T X w - X^T y) = 0$$

$$X^T X w = X^T y$$

$$w^* = (X^T X)^{-1} X^T y$$

$$w^* = \begin{pmatrix} X^T & X \end{pmatrix}^{-1} X^T y$$

The diagram illustrates the matrix dimensions for the OLS equation. The vector  $w^*$  is a 5x1 green column vector. The matrix  $X^T$  is a 5x10 grid where the first row contains 10 ones. The matrix  $X$  is a 10x5 grid where the first column contains 10 ones. The matrix  $X^T y$  is a 5x1 orange column vector. The vector  $y$  is a 10x1 orange column vector.

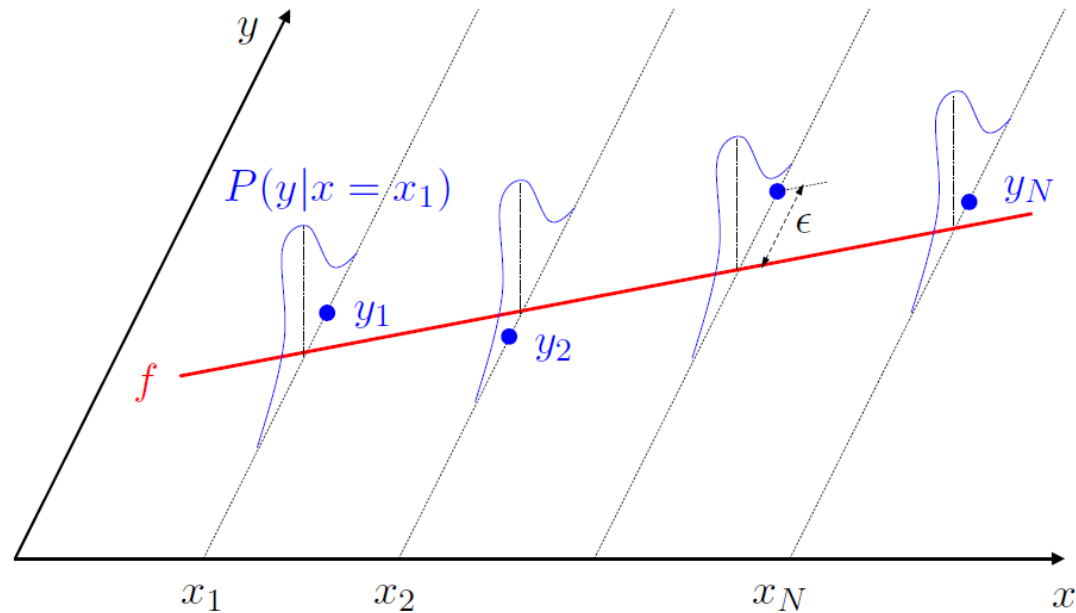
# Linear Regression

- Probabilistic Interpretation of Linear Regression

- Assume  $y \sim \mathcal{N}(\hat{y}, \sigma^2)$ ,  $\hat{y} = \mathbf{w}^T \mathbf{x}$

$$p(y|\mathbf{x}; \mathbf{w}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y - \mathbf{w}^T \mathbf{x})^2}{2\sigma^2}\right)$$

p.d.f. of  $\mathcal{N}(\hat{y}, \sigma^2)$





# Linear Regression

- **Maximum Likelihood Estimation** (with respect to  $\hat{y}$ )

$$\begin{aligned}\mathbf{w}^* &= \operatorname{argmax}_{\mathbf{w}} \prod_{(x_i, y_i) \in D} p(y_i | x_i; \mathbf{w}) = \operatorname{argmax}_{\mathbf{w}} \sum_{(x_i, y_i) \in D} \log p(y_i | x_i; \mathbf{w}) \quad \text{log-likelihood} \\ &= \operatorname{argmax}_{\mathbf{w}} \left[ -\frac{n}{2} \log 2\pi\sigma^2 - \frac{1}{2\sigma^2} \sum_{(x_i, y_i) \in D} (y_i - \mathbf{w}^T x_i)^2 \right]\end{aligned}$$

# Interpretation

## ■ R-squared

- how much of the total variance of your target outcome is explained by the model

$$R^2 = 1 - SSE/SST$$

$$SSE = \sum_{i=1}^n (y^{(i)} - \hat{y}^{(i)})^2$$

$$SST = \sum_{i=1}^n (y^{(i)} - \bar{y})^2$$

- usually ranges between 0 for models where the model does not explain the data at all and 1 for models that explain all of the variance in your data

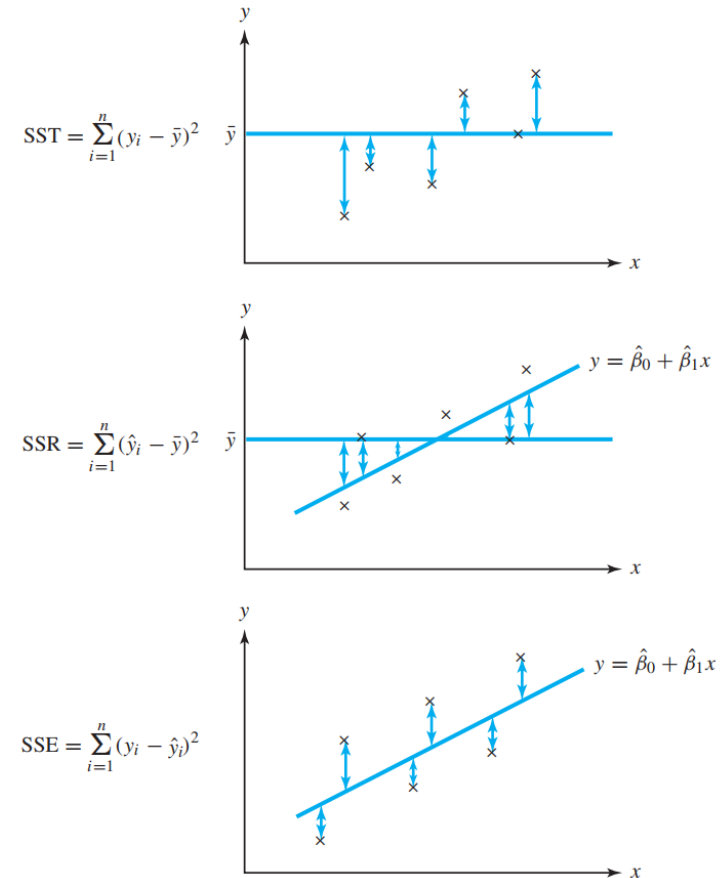
## ■ Adjusted R-squared

- R-squared increases with the number of features in the model
- accounts for the number of features used in the model

$$\bar{R}^2 = 1 - (1 - R^2) \frac{n - 1}{n - p - 1}$$

p: the number of features

n: the number of instances



# Interpretation

## ■ Example

- the linear regression model to predict the number of rented bikes on a particular day, given weather and calendar information

**categorical feature:** The estimated number of bicycles is -1901.5 lower when it is raining, snowing or stormy, compared to good weather –assuming that all other features do not change. When the weather is misty, the predicted number of bicycles is -379.4 lower compared to good weather, given all other features remain the same.

**numerical feature:** An increase of the temperature by 1 degree Celsius increases the predicted number of bicycles by 110.7, when all other features remain fixed

	Weight	SE	t
(Intercept)	2399.4	238.3	10.1
seasonSPRING	899.3	122.3	7.4
seasonSUMMER	138.2	161.7	0.9
seasonFALL	425.6	110.8	3.8
holidayHOLIDAY	-686.1	203.3	3.4
workingdayWORKING DAY	124.9	73.3	1.7
weathersitMISTY	-379.4	87.6	4.3
weathersitRAIN/SNOW/STORM	-1901.5	223.6	8.5
temp	110.7	7.0	15.7
hum	-17.4	3.2	5.5
windspeed	-42.5	6.9	6.2
days_since_2011	4.9	0.2	28.5

# Visual Interpretation

- Weight Plot

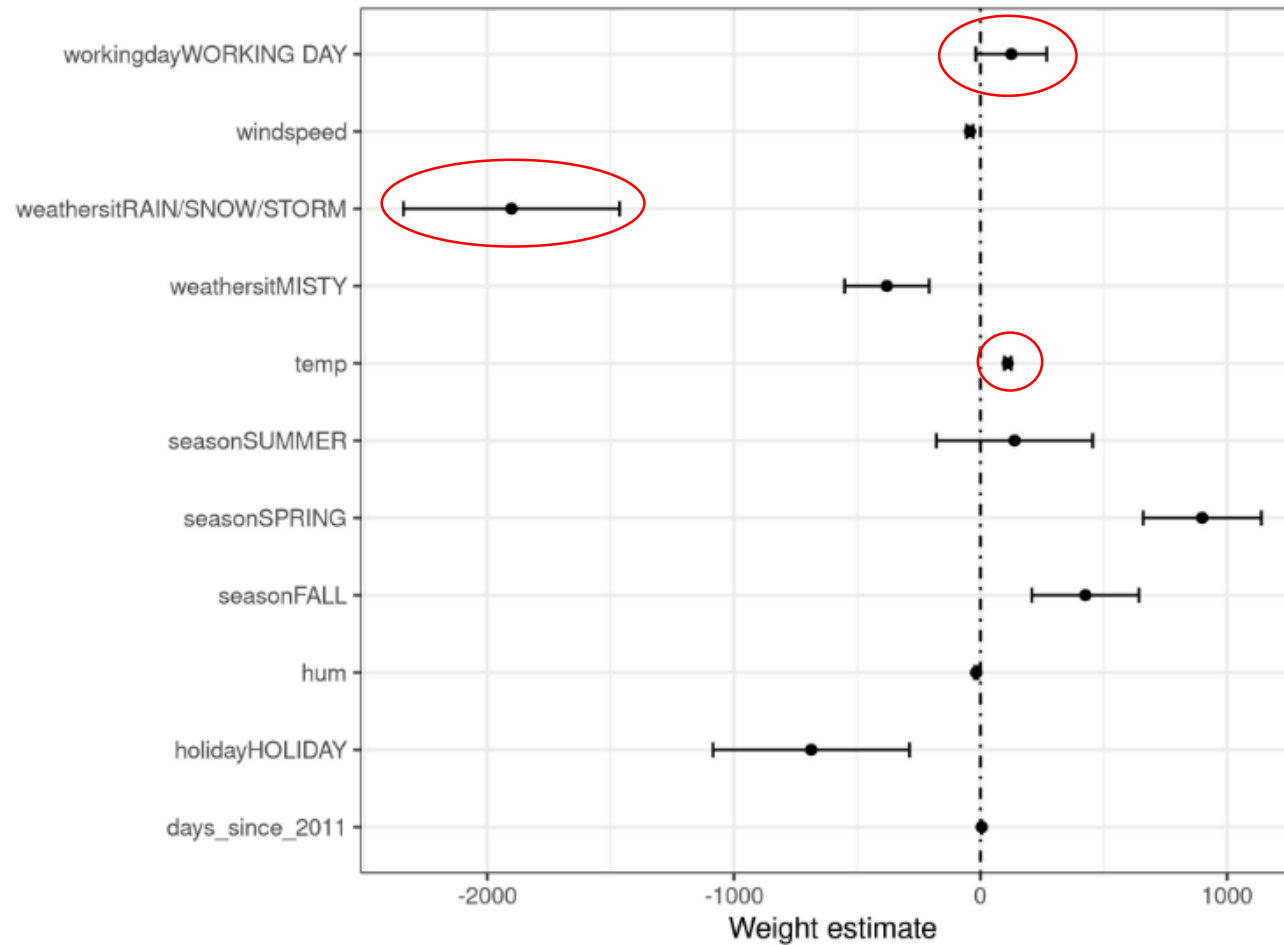


FIGURE 5.1: Weights are displayed as points and the 95% confidence intervals as lines.

Scale??

# Interpretable Models: **Sparse Linear Models**

---

# Generalization

- Using “Linear Regression”, we trained the model  $f$  by minimizing the **training error** (the error on the training set  $D$ )
- The model  $f$  must perform well on new, previously unseen instances, which is called “**Generalization**”.
- To evaluate generalization performance, we use a **test set**  $D^{(\text{test})}$  consisting of  $n^{(\text{test})}$  instances that were collected separately from the training set  $D$ .
- We want the **generalization error**, *a.k.a.* **test error** (the error on the test set) to be low as well.

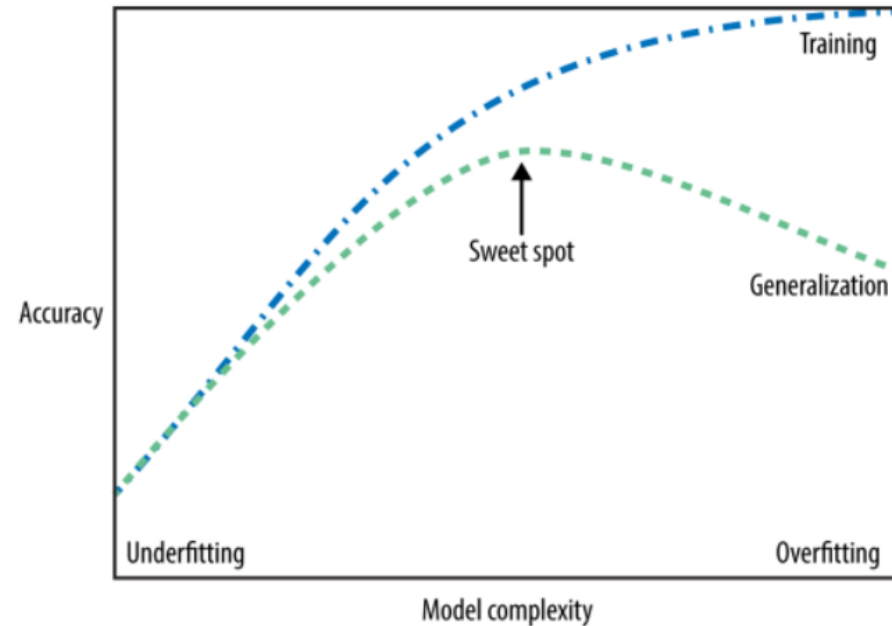
# Generalization

- The factors determining how well a machine learning algorithm will perform are its ability to:
  1. Make the training error small.
  2. Make the gap between training and test error small
- **Overfitting:** the gap between the training error and test error is too large.
- **Underfitting:** the model is not able to obtain a sufficiently low error value on the training set.
- We can control whether a model is more likely to **overfit or underfit** by altering the **capacity**. (complexity)

# Regularization

- **Regularization:** Any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error
  - **Example:** adding a penalty called a **regularizer**  $\Omega(\mathbf{w})$  to the cost function

$$J(\mathbf{w}) = \text{MSE}_{\text{train}} + \lambda\Omega(\mathbf{w})$$





# Regularized Linear Regression

- **Linear Regression:**  $\hat{y} = \mathbf{w}^T \mathbf{x}$

Find the optimal parameter  $\mathbf{w}^*$  that minimizes the training error (cost function)

$$J(\mathbf{w}) = \text{MSE}_{\text{train}} = \frac{1}{n} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2$$

- **Ridge Regression:**  $\hat{y} = \mathbf{w}^T \mathbf{x}$ , **L2 regularization** for linear regression

Add an L2 regularization term  $\alpha \|\mathbf{w}\|_2^2$  to the cost function

$$\tilde{J}(\mathbf{w}) = \text{MSE}_{\text{train}} + \alpha \|\mathbf{w}\|_2^2 = \frac{1}{n} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2 + \alpha \|\mathbf{w}\|_2^2$$

- **Lasso Regression:**  $\hat{y} = \mathbf{w}^T \mathbf{x}$ , **L1 regularization** for linear regression

Add an L1 regularization term  $\alpha \|\mathbf{w}\|_1$  to the cost function

$$\tilde{J}(\mathbf{w}) = \text{MSE}_{\text{train}} + \alpha \|\mathbf{w}\|_1 = \frac{1}{n} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2 + \alpha \|\mathbf{w}\|_1$$

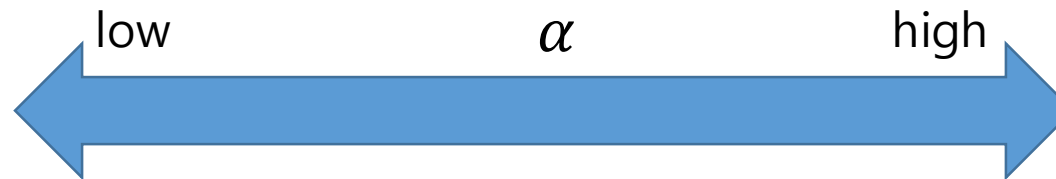
→ Feature selection

Shrinkage  
method

# Regularized Linear Regression

- Adding regularization (explicitly restricting a model to avoid *overfitting*) forces the learning algorithm to not only fit the data but also keep the magnitude of the model parameters as small as possible.
- The hyperparameter  $\alpha$  controls how much you want to regularize the model.
  - If  $\alpha = 0$  then Regularized Linear Regression (Ridge and Lasso) is just Linear Regression.
  - If  $\alpha$  is very large, then all parameters end up very close to zero and the result is a flat line going through the mean of the labels in the training set.

- Complex model
- Low training error
- Prone to overfitting



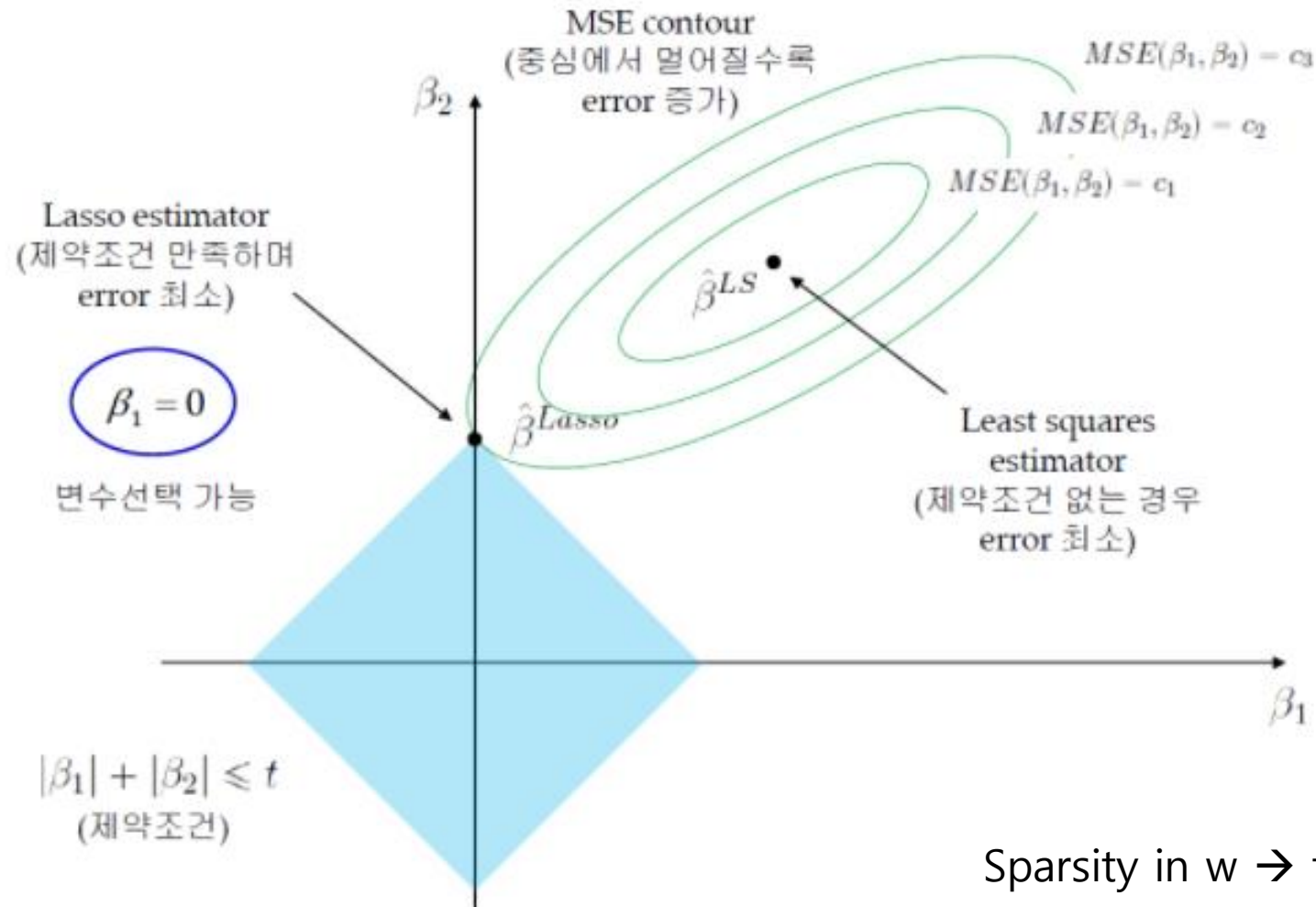
- Simple model
- High training error
- Prone to underfitting

**Better interpretability !**

# Regularized Linear Regression

- Intuitive understanding
  - Lasso (Least **A**bsolute **S**hrinkage and **S**election **O**perator)

$$\tilde{J}(\mathbf{w}) = \text{MSE}_{\text{train}} + \alpha \|\mathbf{w}\|_1 = \frac{1}{n} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2 + \alpha \|\mathbf{w}\|_1$$



Sparsity in  $\mathbf{w} \rightarrow$  feature selection

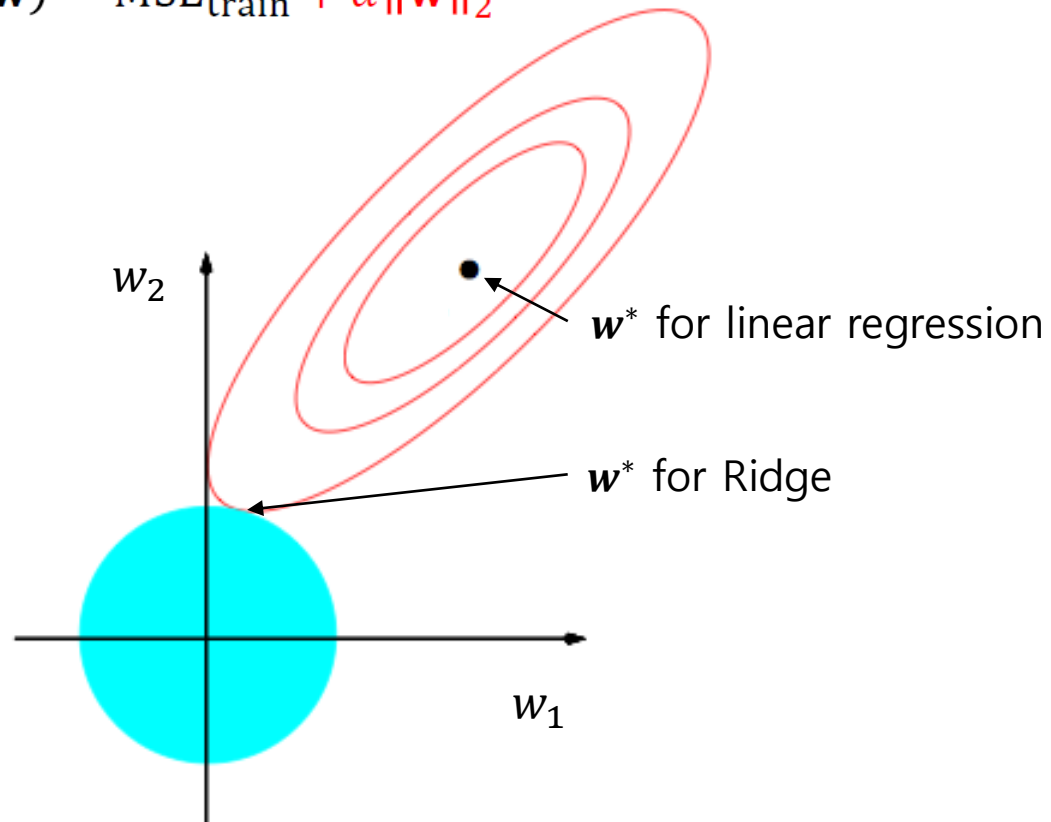
# Regularized Linear Regression

- Intuitive understanding

- The solid blue areas are the constraint regions, while the red ellipses are the contours of the least squares error function (MSE).

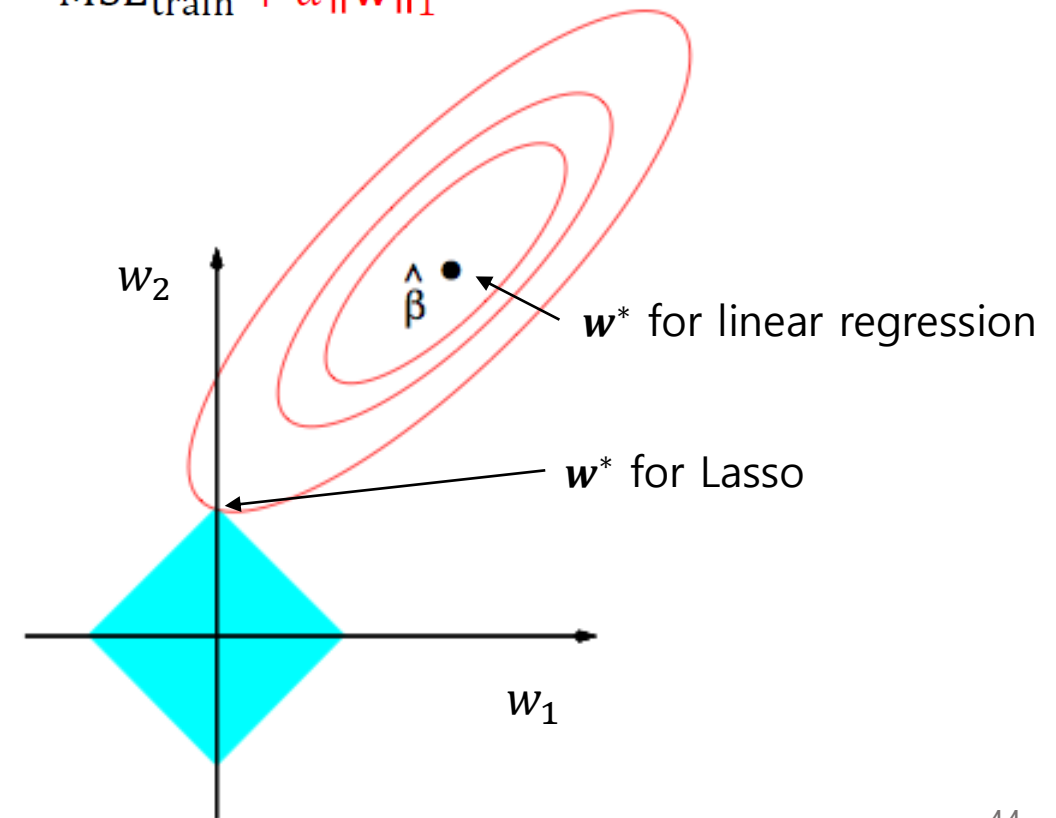
## Ridge

$$\tilde{J}(\mathbf{w}) = \text{MSE}_{\text{train}} + \alpha \|\mathbf{w}\|_2^2$$



## Lasso

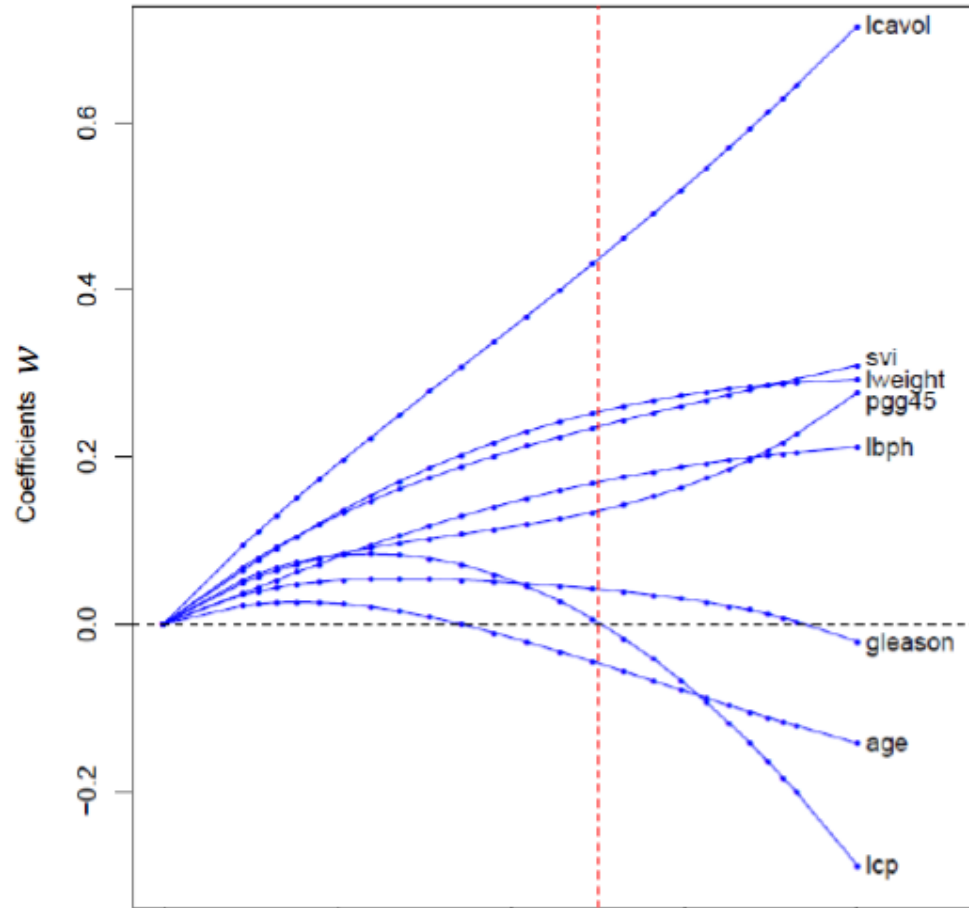
$$\tilde{J}(\mathbf{w}) = \text{MSE}_{\text{train}} + \alpha \|\mathbf{w}\|_1$$



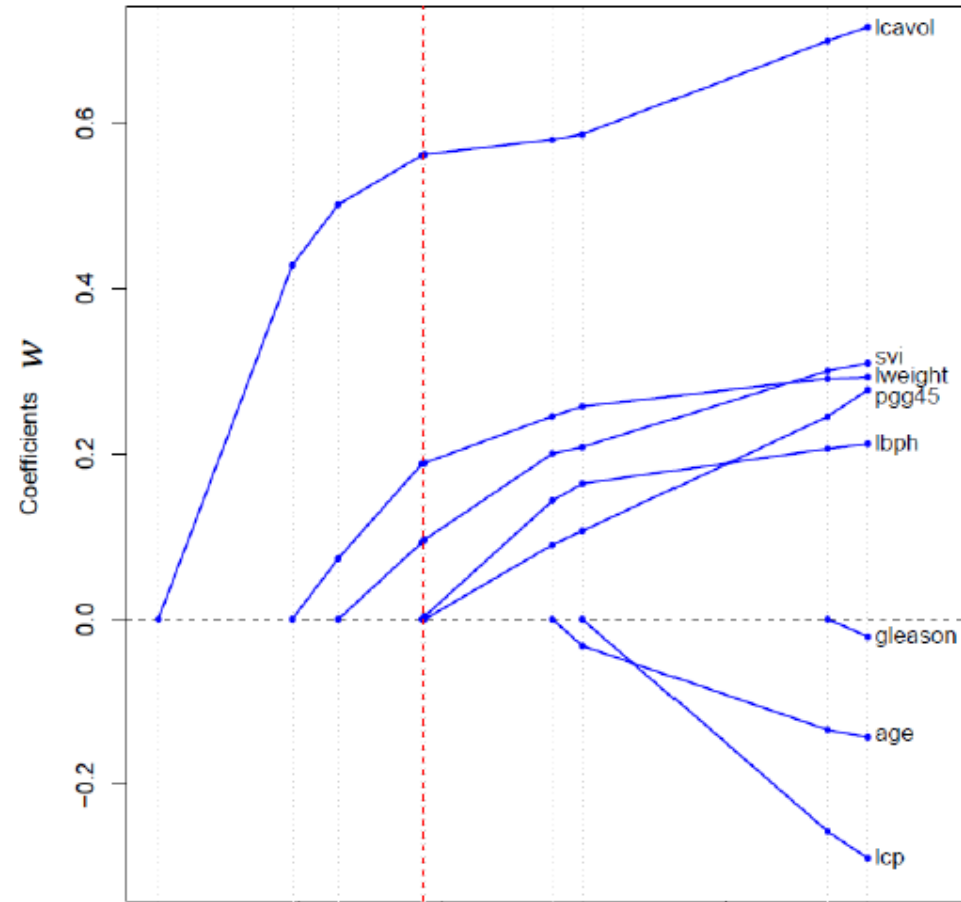
# Regularized Linear Regression

- Example:** Ridge Regression vs Lasso Regression

- $y = w_0 + w_1x_1 + w_2x_2 + \dots + w_dx_d$



constant function ← larger  $\lambda$  → smaller  $\lambda$  standard linear regression



constant function ← larger  $\lambda$  → smaller  $\lambda$  standard linear regression

## Other methods for sparsity in linear models (feature selection methods)

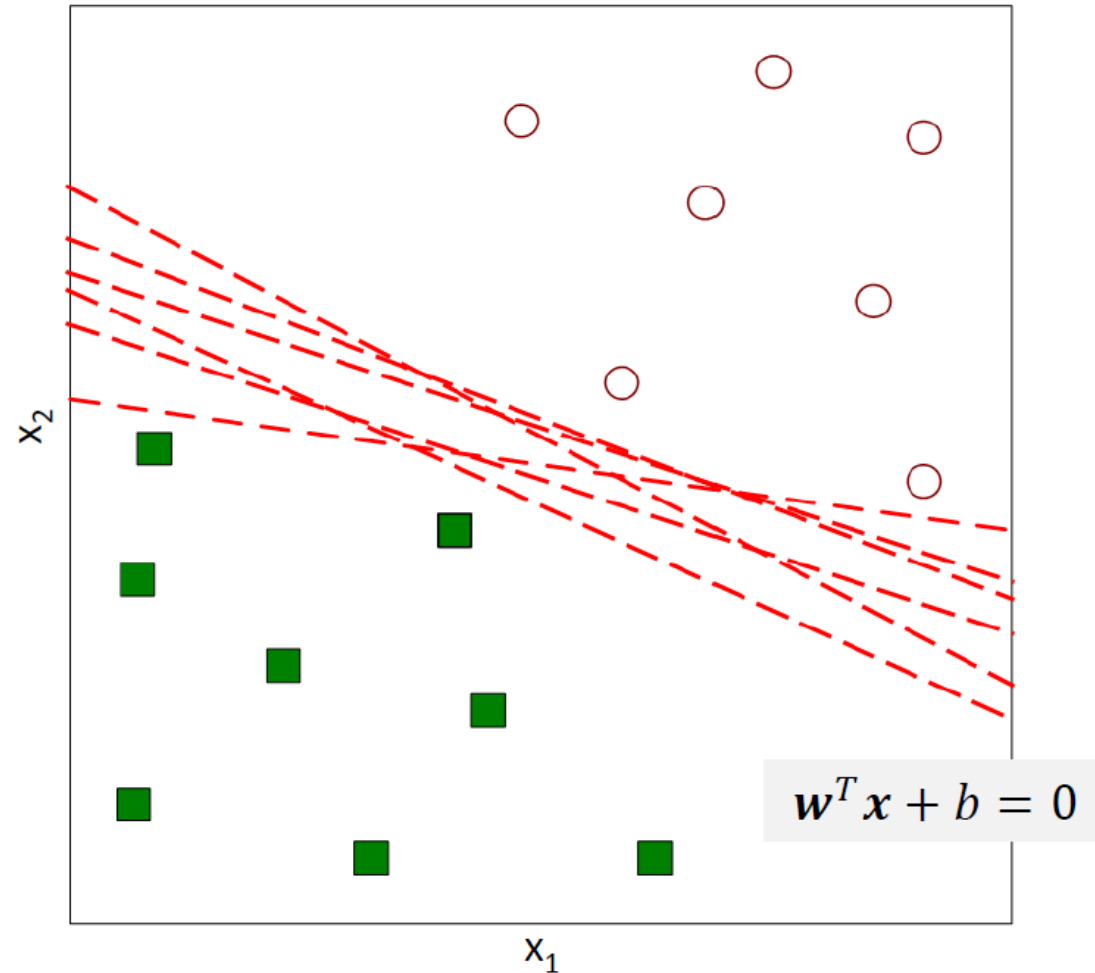
- Pre-processing methods ([Filter](#)):
  - **Manually selected features:** You can always use expert knowledge to select or discard some features.
  - **Univariate selection:** An example is the correlation coefficient. You only consider features that exceed a certain threshold of correlation between the feature and the target.
- Step-wise methods ([Wrapper](#)):
  - **Forward selection:** Fit the linear model with one feature. Do this with each feature. Select the model that works best (e.g. highest R-squared). Now again, for the remaining features, fit different versions of your model by adding each feature to your current best model. Select the one that performs best. Continue until some criterion is reached, such as the maximum number of features in the model.
  - **Backward selection:** Similar to forward selection. But instead of adding features, start with the model that contains all features and try out which feature you have to remove to get the highest performance increase.

# Interpretable Models: **Logistic Regression**

---

# Logistic Regression

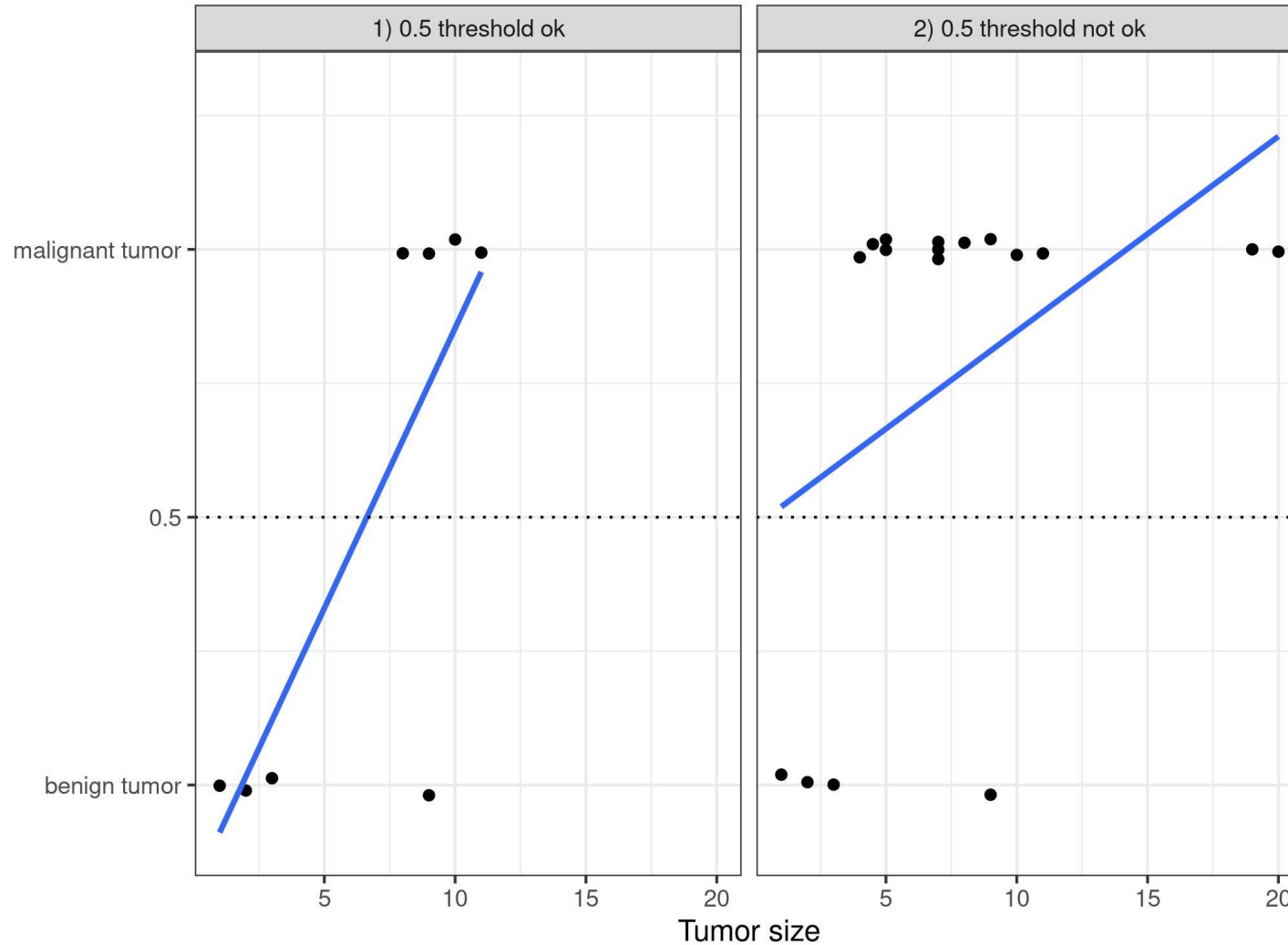
- For linear models for binary classification, the **decision boundary (hyperplane)** that separates two classes is a **linear function** of input features.





# Logistic Regression

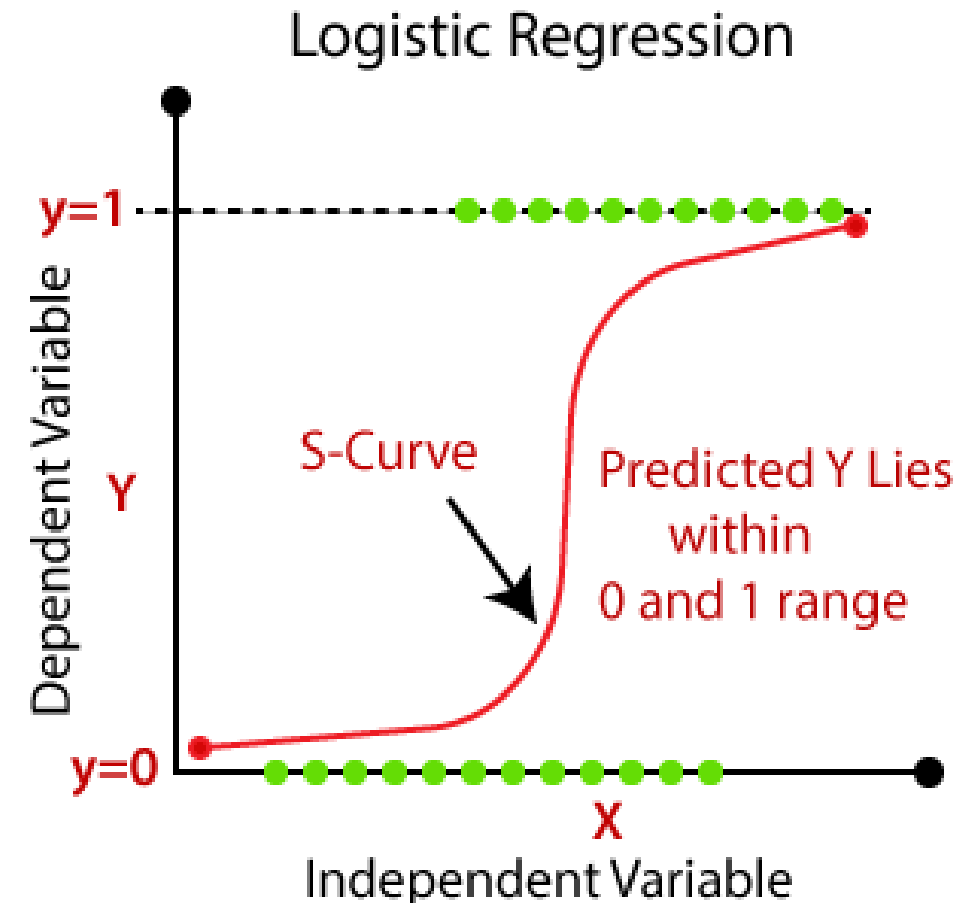
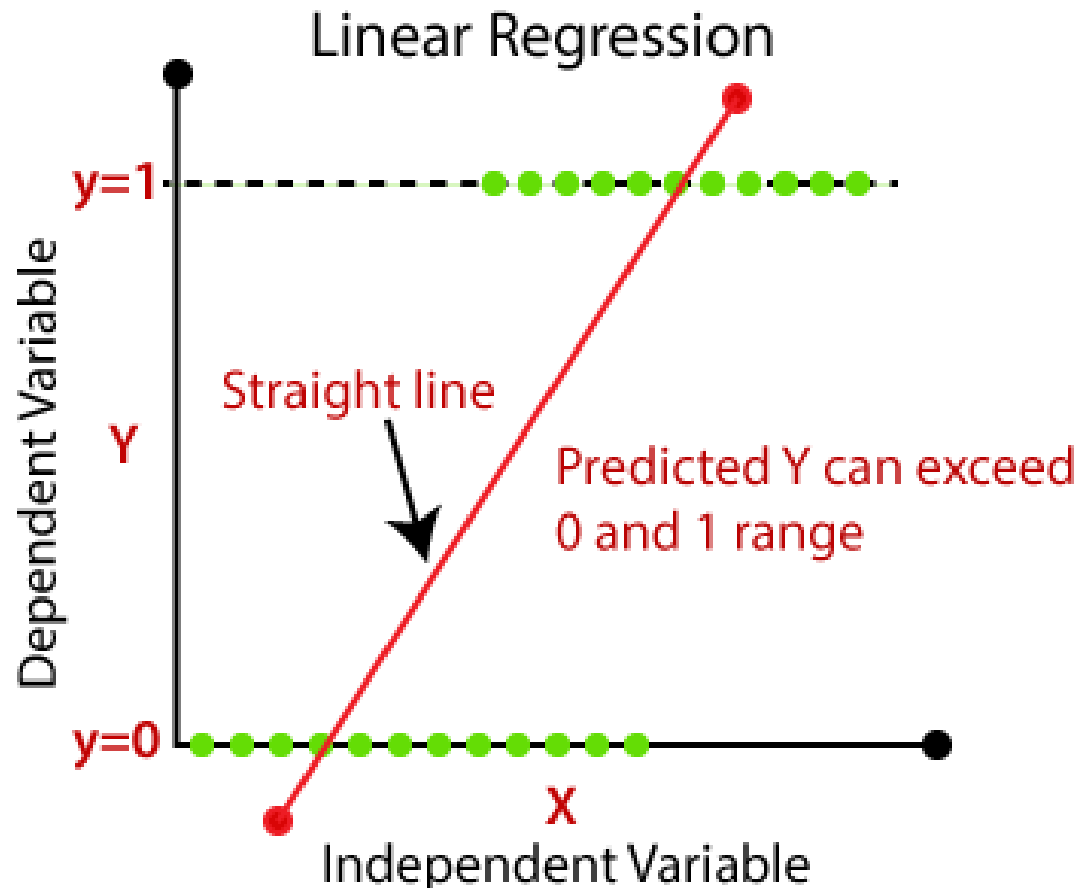
- Using linear regression for binary classification problem



After introducing a few more malignant tumor cases, the regression line shifts and a threshold of 0.5 no longer separates the classes.

# Logistic Regression

- Using linear regression for binary classification problem



# Logistic Regression

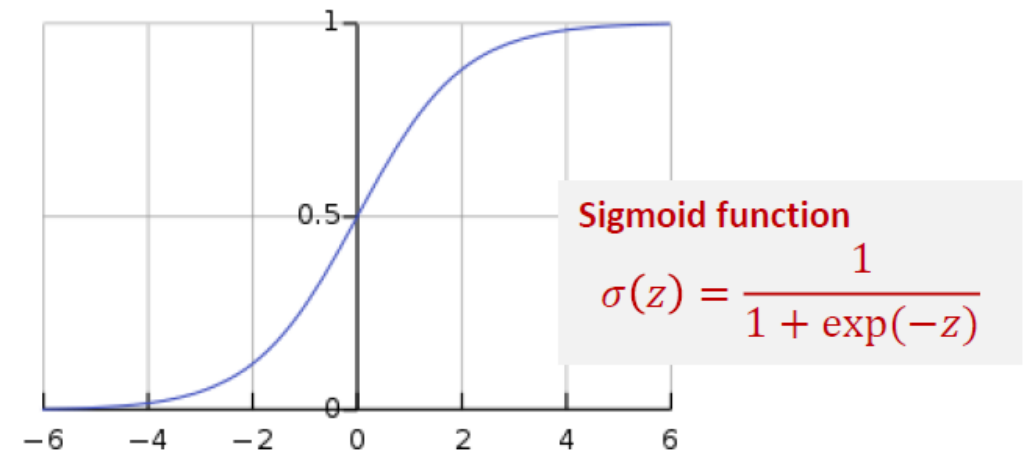
- **Logistic Regression**

- Extends the idea of linear regression to situation where the label is binary  
( $y = 0$  or  $1$ )

$$\hat{y} = \sigma(\mathbf{w}^T \mathbf{x} + b) = \frac{1}{1 + \exp(-\mathbf{w}^T \mathbf{x} - b)}$$
$$\mathbf{x} = (x_1, \dots, x_d) \in \mathbb{R}^d, \quad y \in \{0, 1\}, \quad \hat{y} \in [0, 1]$$

the sigmoid function to squeeze the output of a linear equation between 0 and 1

- If  $\hat{y} > 0.5$ , classify as “1”, If  $\hat{y} < 0.5$ , classify as “0”



# Interpretation

- Interpretation of the weights

↙ **Odds:** probability of event divided by probability of no event

$$\ln \left( \frac{P(y=1)}{1 - P(y=1)} \right) = \log \left( \frac{P(y=1)}{P(y=0)} \right) = \beta_0 + \beta_1 x_1 + \dots + \beta_p x_p$$

→ the logistic regression model is a linear model for the **log odds**!

$$\frac{P(y=1)}{1 - P(y=1)} = odds = \exp(\beta_0 + \beta_1 x_1 + \dots + \beta_p x_p)$$

increase one of the feature values by 1, and check the ratio of the two predictions

$$\begin{aligned} \frac{odds_{x_j+1}}{odds_{x_j}} &= \frac{\exp(\beta_0 + \beta_1 x_1 + \dots + \beta_j(x_j + 1) + \dots + \beta_p x_p)}{\exp(\beta_0 + \beta_1 x_1 + \dots + \beta_j x_j + \dots + \beta_p x_p)} \\ &= \exp(\beta_j(x_j + 1) - \beta_j x_j) = \exp(\beta_j) \end{aligned}$$

A change in a feature by one unit changes the odds ratio by a factor of  $\exp(\beta_j)$

# Interpretation

## ■ Example

TABLE 5.2: The results of fitting a logistic regression model on the cervical cancer dataset. Shown are the features used in the model, their estimated weights and corresponding odds ratios, and the standard errors of the estimated weights.

	Weight	Odds ratio	Std. Error
Intercept	-2.91	0.05	0.32
Hormonal contraceptives y/n	-0.12	0.89	0.30
Smokes y/n	0.26	1.30	0.37
Num. of pregnancies	0.04	1.04	0.10
Num. of diagnosed STDs	0.82	2.27	0.33
Intrauterine device y/n	0.62	1.86	0.40

**numerical feature** : An increase in the number of diagnosed STDs (sexually transmitted diseases) increases the odds by a factor of 2.27, when all other features remain the same.

**categorical feature** : For women using hormonal contraceptives, the odds for cancer vs. no cancer are by a factor of 0.89 lower

# Advantages and Disadvantages

## ■ Disadvantages

- restrictive expressiveness (e.g. interactions must be added manually) so could lead to low performance.
- the interpretation is more difficult than linear regression
  - the interpretation of the weights is multiplicative and not additive.
- **Complete separation:**
  - If there is a feature that would perfectly separate the two classes, the logistic regression model can no longer be trained. This is because the weight for that feature would not converge, because the optimal weight would be infinite.
  - Actually in this case, you do not need machine learning...
  - Could be solved by introducing penalization of the weights or defining a prior probability distribution of weights (**Regularization**).

## ■ Advantages

- the logistic regression model is not only a classification model, but also **gives you probabilities**.
  - Knowing that an instance has a 99% probability for a class compared to 51% makes a big difference.
- Logistic regression can be extended to multi-class classification: **Multinomial Regression (= softmax regression)**.

# Logistic Regression: Training

- Given a (training) dataset  $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_n, y_n)\}$  such that  $\mathbf{x}_i = (\mathbf{1}, x_{i1}, \dots, x_{id}) \in \mathbb{R}^{d+1}$  is the  $i$ -th input vector of  $d$  features and  $y_i \in \{0, 1\}$  is the corresponding target label.

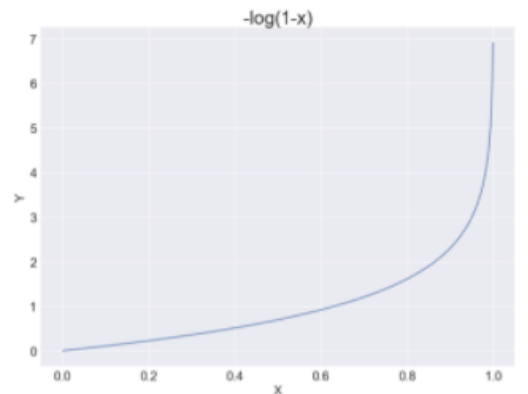
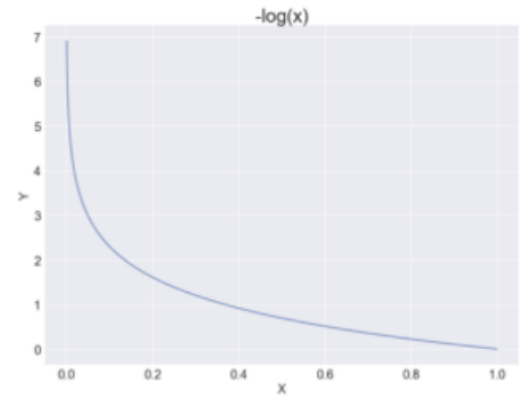
- the first entry is always set to "1"

- The output of model  $f$  (prediction of  $y$ )  
:  $\hat{y} = f(\mathbf{x}) = \sigma(\mathbf{w}^T \mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{w}^T \mathbf{x})}, \hat{y} \in [0, 1]$

- Training:** To find the optimal parameter  $\mathbf{w}^*$  that minimizes the training error (cost function) → here we use "binary cross-entropy" loss

$$J(\mathbf{w}) = \frac{1}{n} \sum_{(\mathbf{x}_i, y_i) \in D} L(y_i, \hat{y}_i) = \frac{1}{n} \sum_{(\mathbf{x}_i, y_i) \in D} [-y_i \log \hat{y}_i - (1 - y_i) \log(1 - \hat{y}_i)]$$

$$BCE = \begin{cases} -\log(\hat{y}), & \text{where } y = 1 \\ -\log(1 - \hat{y}), & \text{where } y = 0 \end{cases}$$



# Logistic Regression: Training

- **Training:** To find the optimal parameter  $\mathbf{w}^*$  that minimizes the training error (cost function) → here we use “binary cross-entropy” loss

$$J(\mathbf{w}) = \frac{1}{n} \sum_{(x_i, y_i) \in D} L(y_i, \hat{y}_i) = \frac{1}{n} \sum_{(x_i, y_i) \in D} [-y_i \log \hat{y}_i - (1 - y_i) \log(1 - \hat{y}_i)]$$

what if  $y_i = 0$ ?  $y_i = 1$ ?

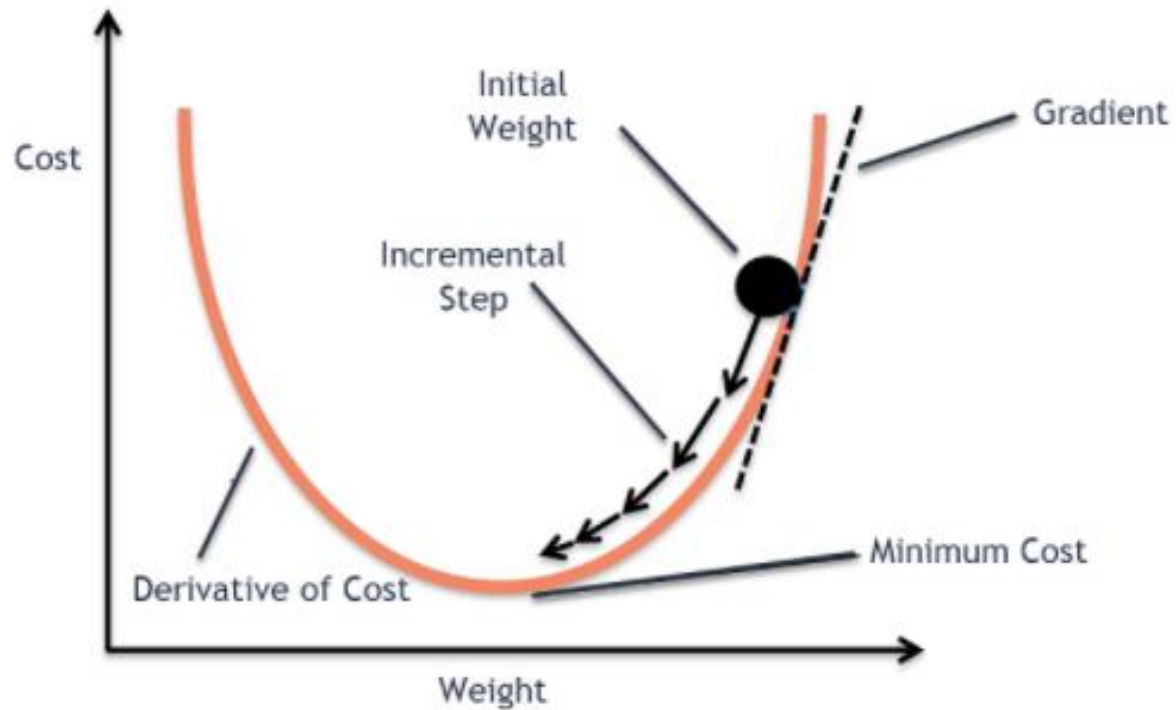
→ No closed-form solution.  
Let's consider simple gradient descent.

*\* Shrinkage methods can also be applied to Logistic Regression*



# Logistic Regression: Training

- Simple gradient descent algorithm

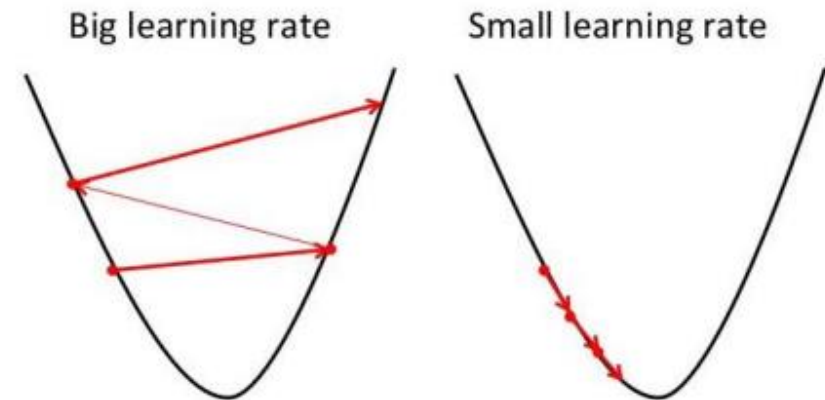


Repeat the following until convergence

$$\mathbf{w} := \mathbf{w} - \epsilon \nabla_{\mathbf{w}} J(\mathbf{w})$$

$$\rightarrow w_j := w_j - \epsilon \frac{\partial}{\partial w_j} J(\mathbf{w}), \forall w_j \in \mathbf{w}$$

$\epsilon$  is the learning rate



- The trained model  $f(\mathbf{x}) = \sigma(\mathbf{w}^{*T} \mathbf{x})$

# Why are linear models so important ?

- (logistic) regression에서 사용되는 방법들은 Neural Network에서 유사하게 활용됨
  - Gradient decent algorithm
  - Regularization (Weight decay)
  - Activation function
  - ...
- Neural Network 차이점?
  - Non-linear 관계를 표현하기 위해 여러 층을 쌓는 구조
  - learning algorithm: Backpropagation을 활용한 gradient decent

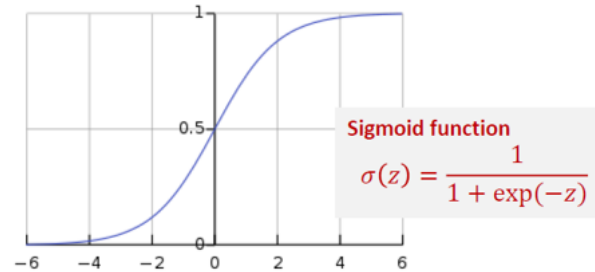
# Appendix: Logistic Regression: Training

$$L(y, \hat{y}) = -y \log \sigma(z) - (1 - y) \log(1 - \sigma(z)),$$

$$\text{where } \hat{y} = \sigma(z), z = \mathbf{w}^T \mathbf{x} = w_0 + w_1 x_1 + \dots + w_d x_d$$

Chain rule

$$\frac{\partial L(\mathbf{w})}{\partial w_j} = \frac{\partial L(\mathbf{w})}{\partial z} \frac{\partial z}{\partial w_j} = (\hat{y} - y) x_j$$



$$\begin{aligned} \frac{\partial L(\mathbf{w})}{\partial z} &= -y \frac{\partial \log \sigma(z)}{\partial z} - (1 - y) \frac{\partial \log(1 - \sigma(z))}{\partial z} \\ &= -y \frac{1}{\sigma(z)} \frac{\partial \sigma(z)}{\partial z} - (1 - y) \frac{-1}{1 - \sigma(z)} \frac{\partial \sigma(z)}{\partial z} \\ &= -y \frac{1}{\sigma(z)} \sigma(z)(1 - \sigma(z)) - (1 - y) \frac{-1}{1 - \sigma(z)} \sigma(z)(1 - \sigma(z)) \\ &= -y + \sigma(z) = \hat{y} - y \end{aligned}$$

$$\begin{aligned} \frac{\partial \sigma(z)}{\partial z} &= \frac{\exp(-z)}{(1 + \exp(-z))^2} \\ &= \frac{1}{1 + \exp(-z)} \cdot \frac{\exp(-z)}{1 + \exp(-z)} = \sigma(z)(1 - \sigma(z)) \end{aligned}$$

$$\frac{\partial z}{\partial w_j} = \frac{\partial (w_0 + w_1 x_1 + \dots + w_d x_d)}{\partial w_j} = x_j$$

---

- You must read!

- Text book
  - 2. Introduction
  - 3. Interpretability
  - 5. Interpretable Models
    - 5.1 Linear Regression
    - 5.2 Logistic Regression