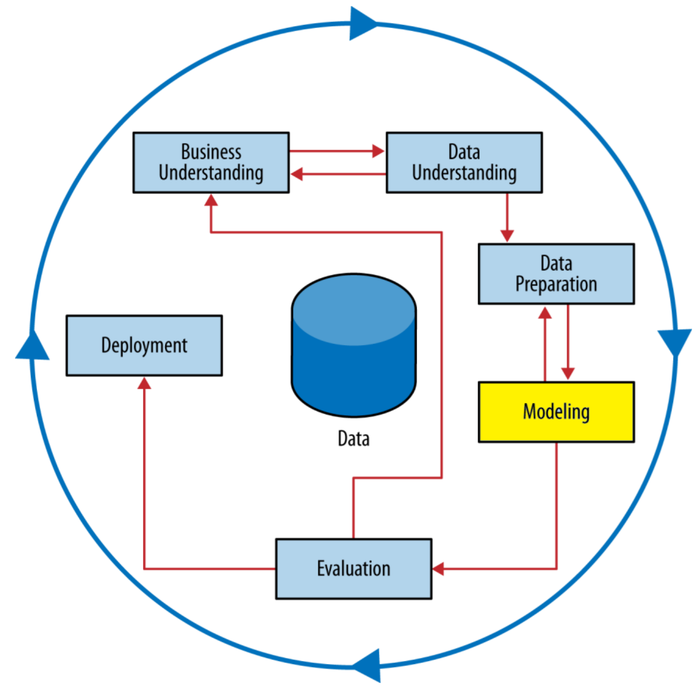
Fit a Model to Data:
Logistic Regression and SVM

PF4

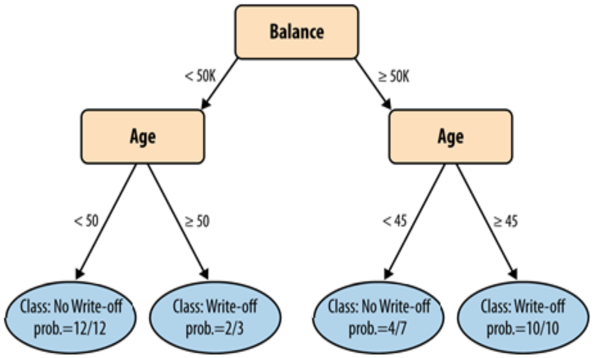
Learning Goals

- Fit a model to the data
  - Learn the **optimal** parameters from the training data
- **Objective functions**
  - Is it the best fit with the data and the data mining goal?
- Supervised classification
  - Logistic regression
  - Support vector machine

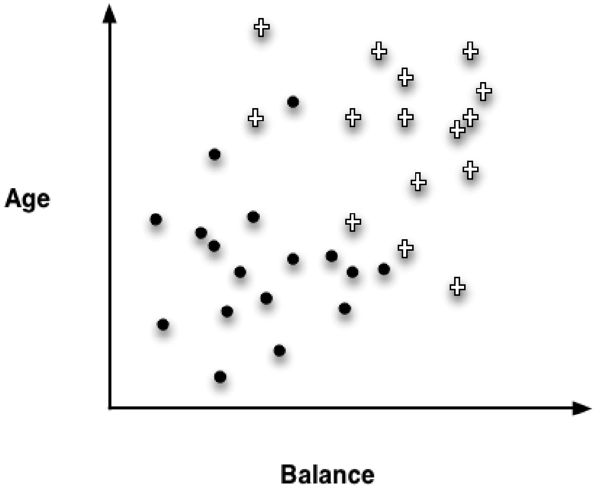


Recap: Decision Tree

- Decision Tree
  - Segment instances into sub-groups, so that members in the same group have similar target value.
  - Each group is pure to some extent.
    - Degree of (im)purity measured by entropy



- The **decision boundary** in tree models is perpendicular to the axes.
  - Can we segment the customer data differently?



Supervised Classification

Non-parametric approach
(e.g., Decision Tree)

- Decision boundaries are perpendicular to the axes
- Classify instances recursively using divide-and-conquer approach
  - The model is a set of rules.

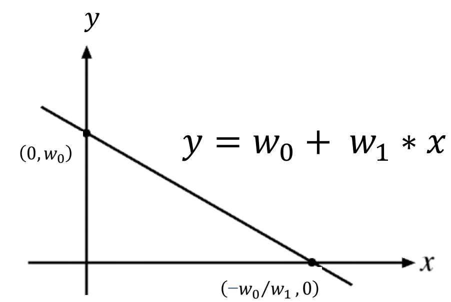
Parametric approach
(e.g., Logistic Regression, SVM)

- Linear classifiers can be in any direction
- Classify instances using mathematical functions
  - $f(x)$ is a weighted sum of various features x

Linear Classifier

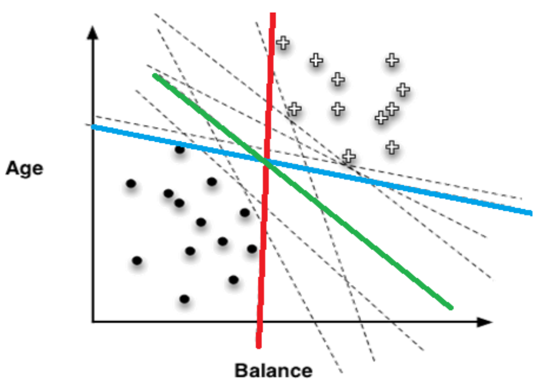
- A **Linear classifier** is weighted sum of various features x.
  - Weights refer to the coefficients.

- Recall the **mathematical function** for a simple linear regression?
  - $y = w_0 + w_1\, x$
    - w_0 is the intercept (bias)
    - w_1 is the slope (coefficient)

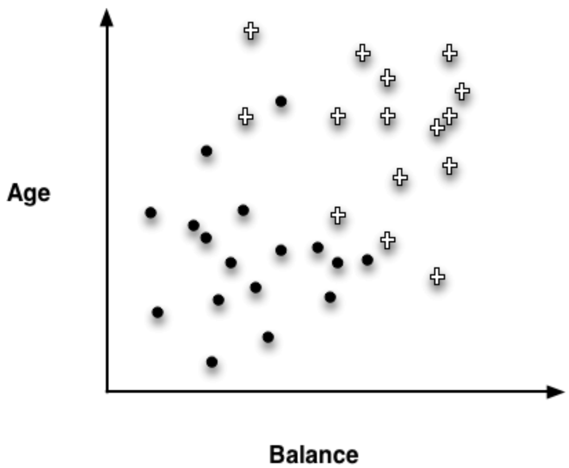


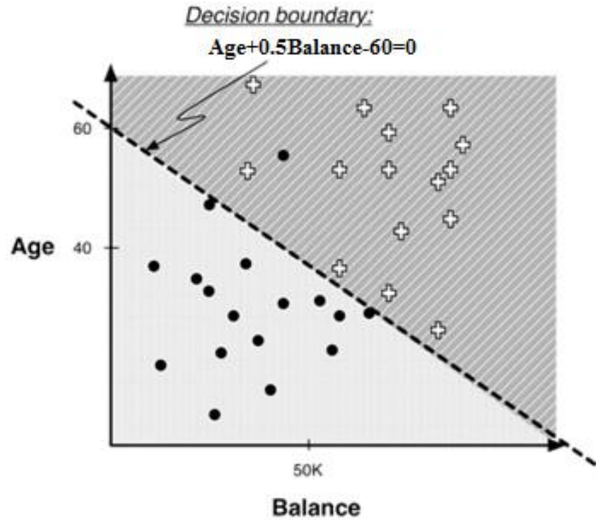
- We can also use a **straight line** to segment the customer data.
  - Each line is a linear mathematical function for *age* and *balance*.

$$Age = w_0 + w_1 \times Balance$$



Linear Classifier





- **Mathematical Function** (Decision Boundary)

$$Age = -0.5 \times Balance + 60$$

$$\boldsymbol{Age + 0.5 \times Balance - 60 = 0}$$

- **Class Prediction**

$$class = \begin{cases} \bullet \text{ if } Age + 0.5 \times Balance - 60 \le 0 \\ + \text{ if } Age + 0.5 \times Balance - 60 > 0 \end{cases}$$

Linear Classifier

$$f(X) = 2X_1 + 3X_2 + 1$$

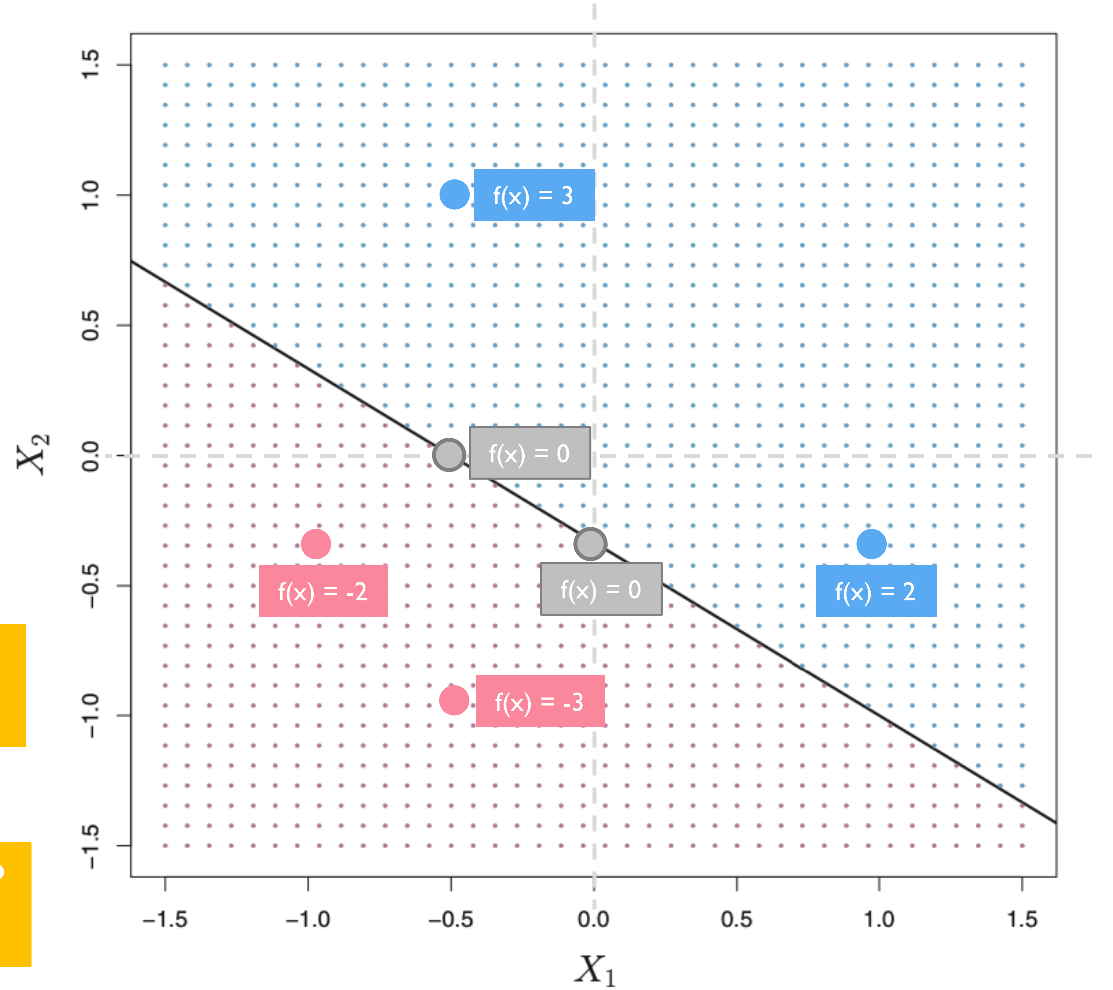
- The decision boundary:
  - $f(X) = 2X_1 + 3X_2 + 1 = 0$
  - or $X_2 = -\frac{2}{3}X_1 - \frac{1}{3}$

- To have $f(X)$ = 0:
  - When X_1= - $\frac{1}{2}$, X_2 = 0

Caculate $f(X)$ value if X_2 = 1 or -1? No change for X_1.

  - When X_1 = 0, X_2 = - $\frac{1}{3}$

Calculate $f(X)$ value if X_1 = -1 or 1? No change for X_2.

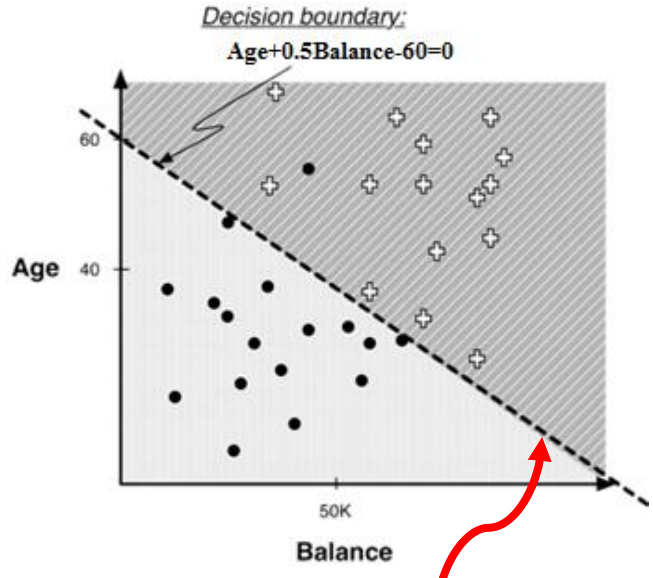


Linear Classifier

- A general linear model

$$\boldsymbol{f}(\boldsymbol{x}) = w_0 + \boldsymbol{w_1 x_1} + \cdots + \boldsymbol{w_p x_p}$$

- $|\boldsymbol{f}(\boldsymbol{x})|$ measures the distance between an instance with features x and the decision boundary.
  - $|\boldsymbol{f}(\boldsymbol{x})|$ is NOT the perpendicular distance to decision boundary, but proportional to it.



$\boldsymbol{f}(\boldsymbol{x}) = \mathbf{0}$: **most uncertain**

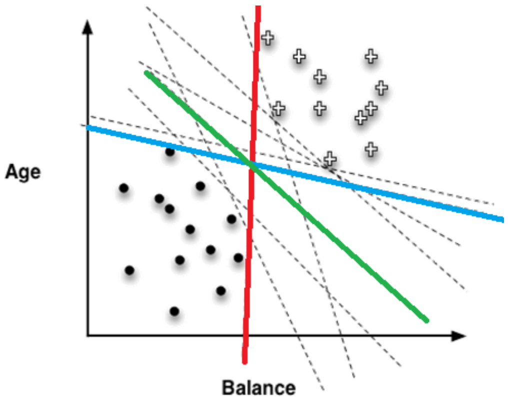
- $|\boldsymbol{f}(\boldsymbol{x})|$ reflects the degree of **certainty** that an instance belongs to a class.
  - Example: loan write-off

    + write-off ● not write-off
  - $|f(x)|$ increases, certainty increases

Perpendicular distance of an instance to decison boundary: $D = |f(x)| \Big/ \sqrt{\sum_{j=1}^{p} w_j^2}$

Objective Function: Find the Best Line

- The optimal parameter w_p depends on the **objective function**.
  - An objective function represents what we want to achieve in a data mining task.
  - It usually tries to reduce the error/loss for all training instances.
    - There are various error/loss functions to choose.



- Like linear regression, both logistic regression and SVM optimize the parameters by **fitting a (linear) model to data**, the difference is in their objective functions.
  - recap: linear regression minimizes sum of squared errors (i.e., RSS).

$$min \sum_{i=1}^{n} (y_i - \hat{y_i})^2$$

What is our goal in a classification task then?

Logistic Regression: Probability Estimate

- Logistic regression applies a **linear mathematical function** to estimate the **class probability**.
  - Binary classification: $f(x)$ is used to estimate $P(y = 1|\, x)$.

$$f(x) = w_0 + w_1 x_1 + \cdots + w_p x_p$$

  - What is the range of the linear output $f(x)$?

[-∞, ∞]

  - What is the range of a class probability $P(y = 1|x)$?

$\mathbf{[0, 1]}$

- How can we convert $f(x)$ into $P(y = 1|x)$?

Step 1: Odds and Probability

- The **odds (ratio)** of an event is the ratio of the probability of the event *occurring* vs. the probability of the event not *occurring*.

- Assume p is the probability of instance x_i belonging to the positive class, the **odds** of instance x_i being positive is

$$\frac{p}{1-p} = \frac{P(y=1 \mid x)}{P(y=0 \mid x)}$$

Odds is ranged from [0, ∞]

Probability (p)	**Odds ($\frac{p}{1-p}$)**
0.5	50:50 (or 1)
0.9	90:10 (or 9)
0.999	999:1 (or 999)
0.01	1:99 (or 0.0101)
0.001	1:999 (or 0.001001)

Step 2: Odds and Log-odds

- **The natural logarithm of odds** is ranged from **[-∞, ∞]**.
  - Natural logarithm takes base e, a mathematical constant 2.71…

Probability (p)	**Odds ($\frac{p}{1-p}$)**	**Log-odds ($ln(\frac{p}{1-p})$)**
0.5	50:50 (or 1)	0
0.9	90:10 (or 9)	2.19
0.999	999:1 (or 999)	6.9
0.01	1:99 (or 0.0101)	-4.6
0.001	1:999 (or 0.001001)	-6.9

- The linear value $f(x)$ estimates the log-odds that instance x belong to the positive class:

$$f(x) = w_0 + w_1 x_1 + \cdots + w_p x_p = ln\left(\frac{p}{1-p}\right)$$

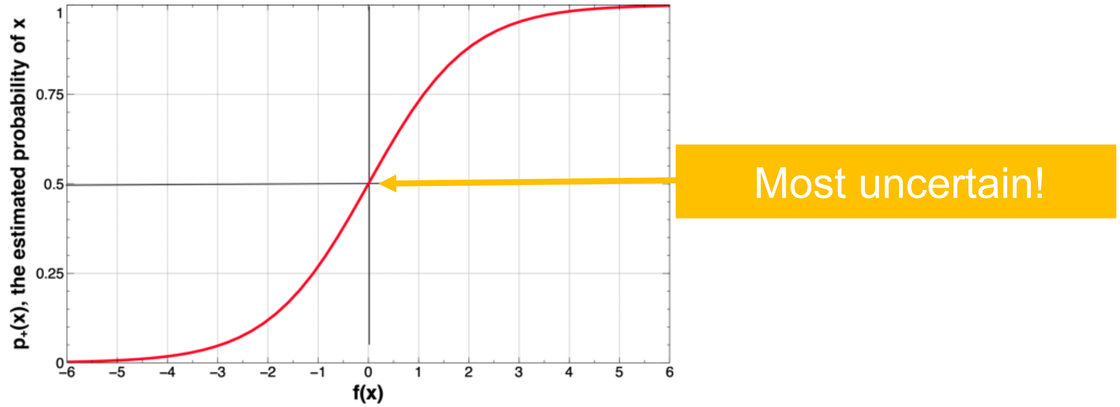
Both $ln(x)$ or $\log(x)$ represent natural logarithm of x

Logistic Regression: Probability Estimate

- **Logistic Function**, also named sigmoid function, transforms log-odds to class probability.
  - Class probability is a logistic function of log-odds (i.e., $f(x)$).

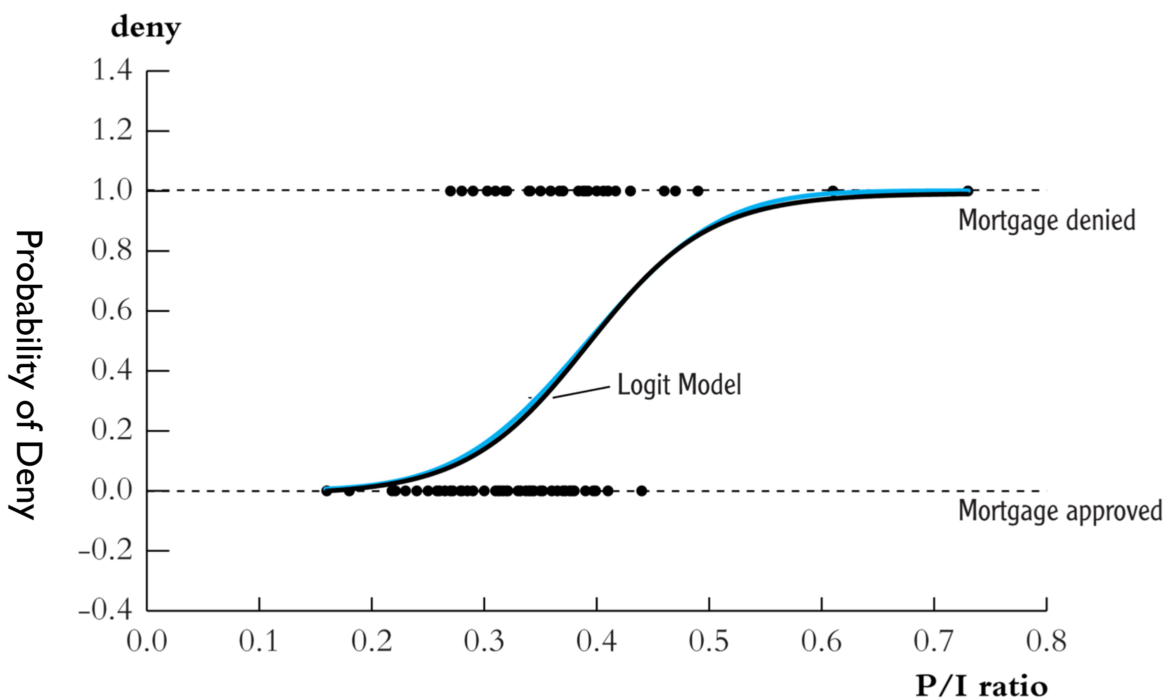
$$P(y=1|f(x)) = \frac{e^{f(x)}}{1+e^{f(x)}} \text{ or } \frac{1}{1+e^{-f(x)}}$$

- When $f(x) = 0$, what is $P(y=1)$?
  - How shall we interpret it?



Example: Mortgage Application

- Bank's decision: deny a mortgage application?
  - **Target**: Deny = reject (1) or approve (0) the application.
  - **Feature**: *PI ratio* measures the ratio of debt to income for a person.



Example: Mortgage Application

- The logistic regression model estimates w_0= -1 and w_1 = 2.5

$$f(x) = -1 + 2.5 \times PI\ ratio$$

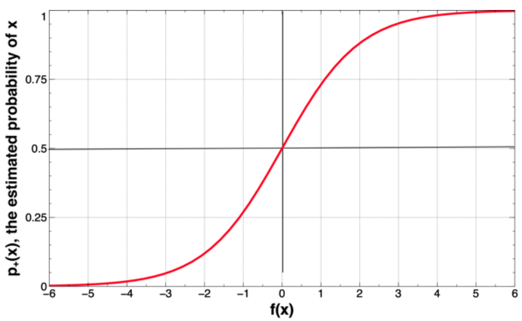
$$P(deny = 1|PI\ ratio) = \frac{1}{1 + e^{-f(x)}}$$

- **Interpret the coefficient (w_1)**: 1 unit increase in PI ratio is associated with 2.5 unit increase in log-odds of loan rejection.

- When P/I ratio = 0.3:
  - The log-odds $f(x) = -1 + 2.5 \times 0.3 = -0.25$
  - The probability $P(deny = 1|PI\ ratio) = \frac{1}{1+e^{0.25}} \approx 0.44$
    - There is 44% chance that the application will be rejected.

Objective Function: Logistic Regression

- What is our goal in a binary classification task?
  - For positive instances (i.e., actual class y_i = 1), we expect
    - $\hat{y}_i$ = $p(+)$ is as close to 1 (i.e., y_i) as possible.
  - For negative instances (i.e., actual class y_i = 0), we expect
    - $\hat{y}_i$ = $p(+)$ is as close to 0 (i.e., y_i) as possible. This means, $p(-)$ or 1-$\hat{y}_i$ is as close to 100% as possible.



Maximum Likelihood Estimation: maximize the estimated probability of each instance's actual class.

- **Objective Function**:

$$\min_{w} \sum_{i=1}^{n} \boxed{-y_i \, ln(\hat{y}_i) - (1 - y_i) \, ln(1 - \hat{y}_i)}$$

Log-loss or Binary Cross-entropy

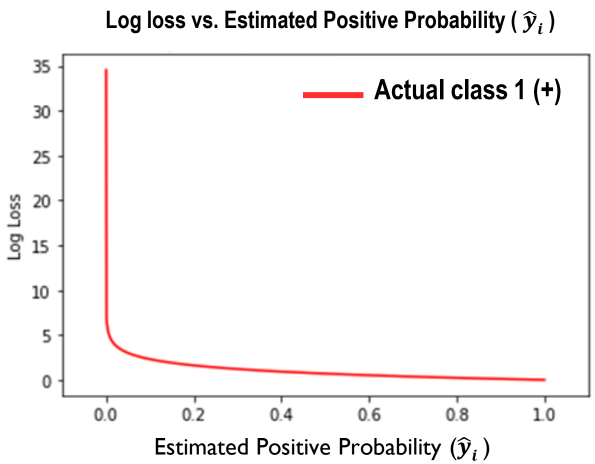
Log-loss = - $y_i \, ln(\hat{y}_i) - (1 - y_i) \, ln(1 - \hat{y}_i)$

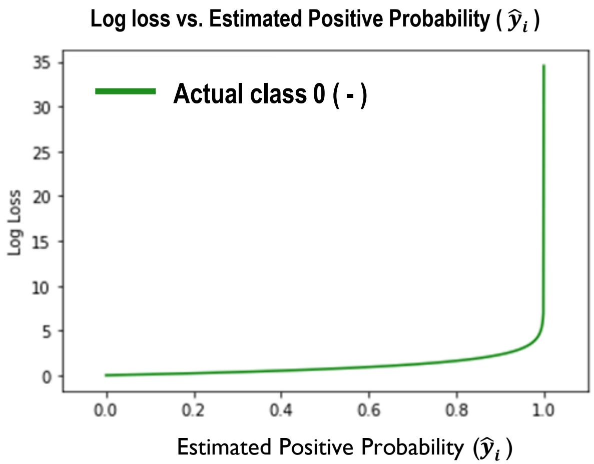
Objective Function: Logistic Regression

- For any instance, if ***p(+)*** (i.e., $\hat{y}_i$) goes farther away from its actual class label (i.e., y_i), its **log loss** is greater.
  - For a positive instance (i.e., y_i = 1): the farther away $\hat{y}_i$ is from 1 (actual class y_i), the greater its log loss is.

    Log-loss = - $ln(\hat{y}_i)$
  - For a negative instance (i.e., y_i = 0): the farther away $\hat{y}_i$ is from 0 (actual class y_i), the greater its log loss is.

    Log-loss = - $ln(1-\hat{y}_i)$





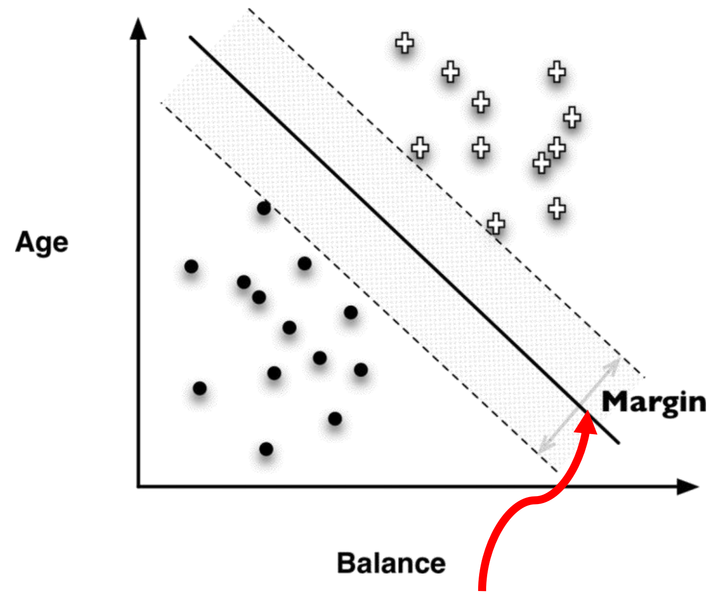
- By **minimizing the sum of log loss** for all training instances, logistic regression maximizes the estimated probability of actual class for all.

Support Vector Machine

- **Support vector machines** are also linear classifiers as well.

$$f(x) = w_0 + w_1 x_1 + \cdots + w_p x_p = 0$$

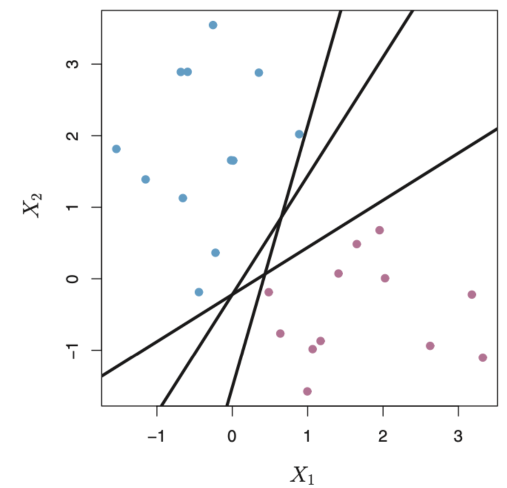
- Instead of separating data with a line directly, SVM also fit the **widest bar** (i.e., margin) between the two classes.
  - Why the bar is necessary?



$f(x) = 0$: **most uncertain**

Why a Margin is Necessary?

- Given a 2D feature space (X_1, X_2), find a **separating hyperplane (i.e., decision boundary)** to separate two classes $y_i = \{Yes, No\}$.
  - The decision boundary is $f(x) = w_0 + w_1X_1 + w_2X_2 = 0$
    - $f(x) > 0$, predict Yes
    - $f(x) < 0$, predict No

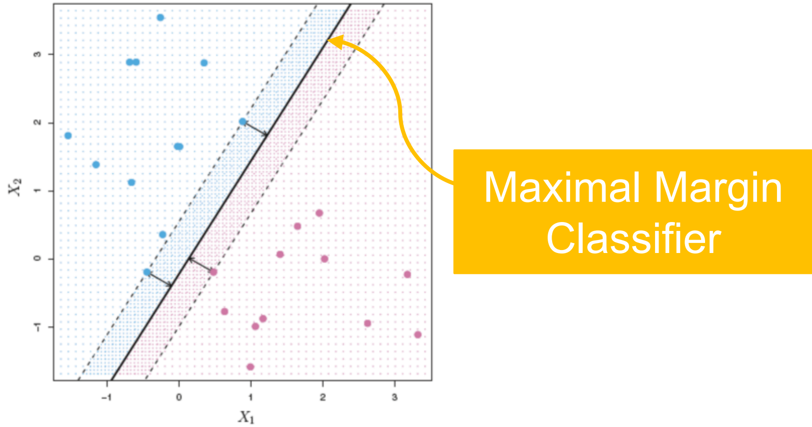


- However, there can be multiple separating hyperplanes for a dataset.
  - in the right graph, all lines can be shifted a tiny bit, without touching any instance.

- With training instances at least **certain distance away** (i.e., margin) from the hyperplane, we can avoid the **uncertain area**.
  - The margin prevents the hyperplane from being overly influenced by noises and therefore avoid overfitting.

Road to SVM: Maximal Margin Classifier

- **Hard margin**: no training instance shall violate the margin lines.
  - All instances are separable – no misclassification.
  - Instances are at least certain distance away from their margin line.
- To find the best hyperplane (i.e., the model):
  - For each hyperplane, find the **margin width**: i.e., the shortest perpendicular distance between instances and the hyperplane.
    - Need to compare the perpendicular distance for all instances (to hyperplane).
  - The best hyperplane is the one with the **maximized margin**.
- **Support Vectors**: instances on the margin lines (i.e., dashed lines).
  - Equidistant to the hyperplane.
  - Only movement of support vectors affects the model.
    - Movement of other instances does not.



Objective Function: Maximal Margin Classifier

$$\max_{w_1 \dots w_p, M} \quad M \qquad \text{subject to} \qquad \sum_{j=1}^{p} w_j^2 = 1$$

Here y_i takes either 1 or -1, and $\hat{y}_i = f(x)$

- For positive instances ($y_i = +1$): $\hat{y}_i = w_0 + w_1 x_{1i} + \cdots + w_p x_{pi} > 0$
- For negative instances ($y_i = -1$): $\hat{y}_i = w_0 + w_1 x_{1i} + \cdots + w_p x_{pi} < 0$

For separable training instances x_i: $\quad y_i \, \hat{y}_i \geq \boxed{M}$

Margin Width (always > 0)

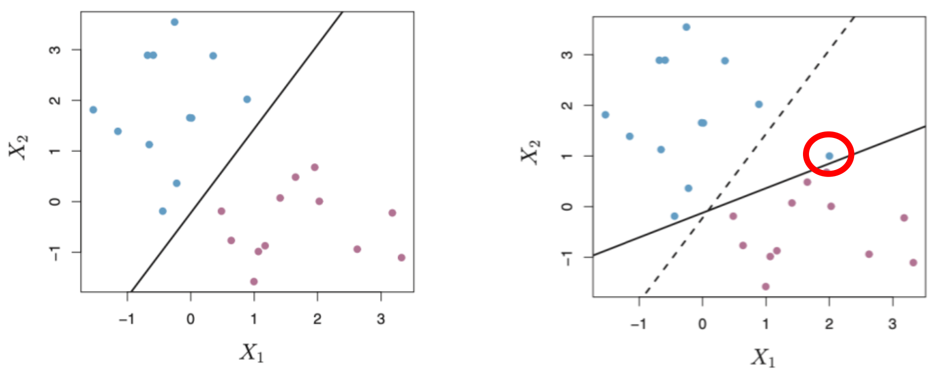
Note: Maximal Margin Classifier assumes all instances are separable.

$\boldsymbol{y_i \hat{y}_i}$ **=** $|\boldsymbol{\hat{y}_i}|$: the *perpendicular distance* between instance x_i to the hyperplane, given $\sum_{j=1}^{p} w_j^2 = 1$

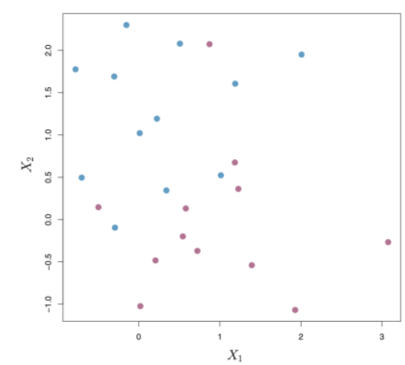
Perpendicular distance of x_i to the hyperplane: $D_i = |\hat{y}_i| \Big/ \sqrt{\sum_{j=1}^{p} w_j^2}$

Drawbacks of Maximal Margin Classifier

- Maximal Margin Classifier is **extremely sensitive** to small changes in training data and tend to **overfit**.
  - According to its objective function, each instance should not only be on the correct side of the hyperplane, but also the correct side of its margin line.

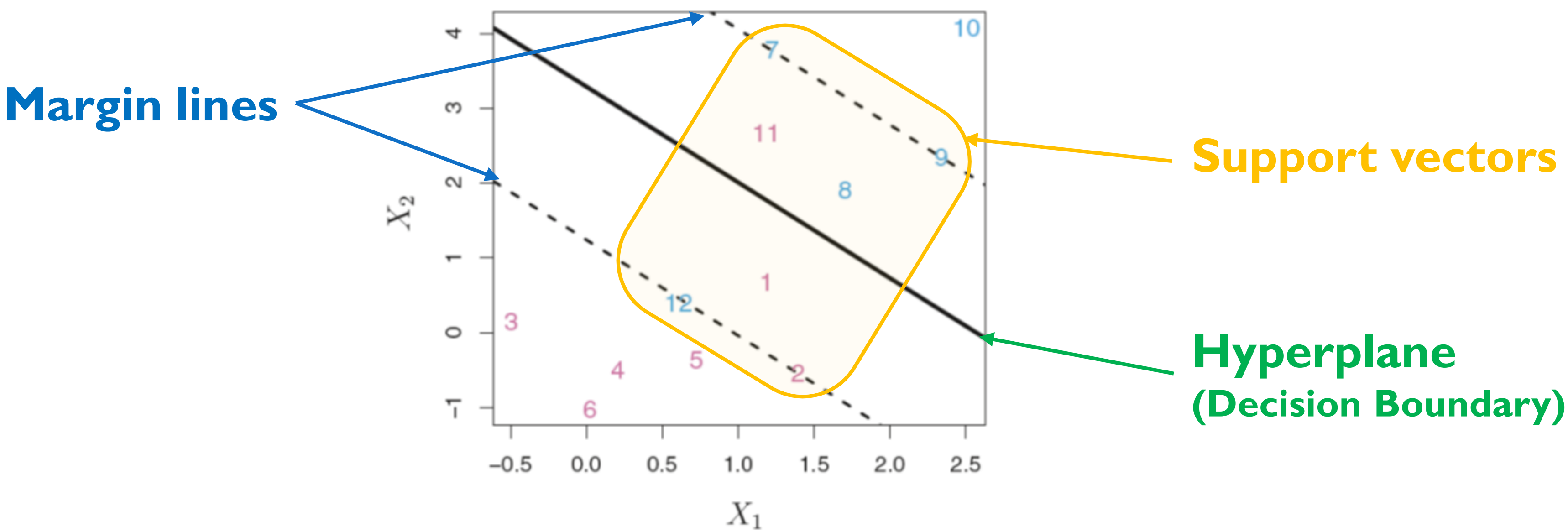


- The separating hyperplane **may not exist**.
  - When training instances are non-separable, there is NO maximal margin classifier with margin width > 0.



Support Vector Classifier

- **Support vector classifier** relies on **soft margin**, which allows some training instances to be the incorrect side of its margin line.
  - **Support vectors** are training instances violating its margin line.
    - Support vectors bring errors to objective function and affect the hyperplane.
  - Only instances violating the hyperplane will be misclassified.
    - Instances that violating its margin line but not the hyperplane bring errors to objective function but will NOT be misclassified.



Objective Function: SVC

$$\max_{w_1 \ldots w_p,\, \xi_i,\, M} M \quad \text{subject to} \quad \textstyle\sum_{j=1}^{p} w_j^2 = 1$$

Here y_i takes either 1 or -1, and $\hat{y}_i = f(x)$
- $\hat{y}_i = w_0 + w_1 x_{1i} + \cdots + w_p x_{pi}$ can be any value

$$y_i\, \hat{y}_i \geq M(1 - \xi_i) \quad \text{where} \quad \xi_i \geq 0$$

Hinge loss allows instance x_i to be at distance ξ_i from its margin line.
- $\xi_i = 0$: in the correct side of its margin line.
- $0 < \xi_i < 1$: violate its margin line only (but not hyperplane)
- $\xi_i > 1$: violate the hyperplane (**misclassification**)

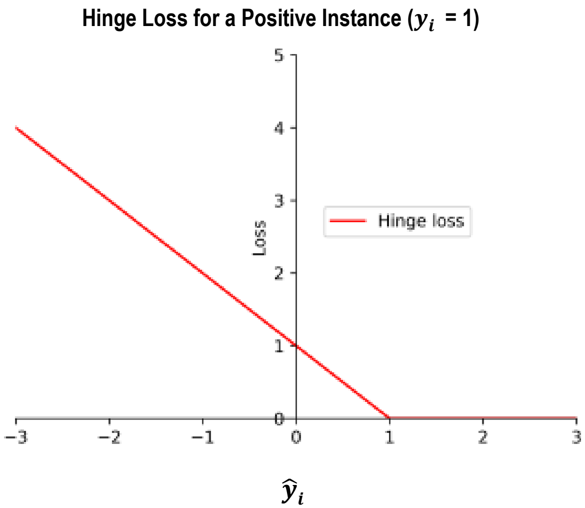
- Rewritten as: $$\min_{w} \sum_{i=1}^{n} \xi_i \quad \text{where} \quad \xi_i = \max_i\, [0, 1 - y_i \hat{y}_i]$$

If $\sum_{j=1}^{p} w_j^2 = 1$, margin width **M = 1/** $\sqrt{\sum_{j=1}^{p} w_j^2} = 1$

Hinge Loss

- **Hinge loss** (ξ) is the error for instances violating its own margin line.
  - Hinge loss is proportional to the distance between the instance and its margin line.

$$\boldsymbol{\xi_i = \max_i \, [0, 1 - \, y_i \hat{y}_i]}$$



- For a positive instance (y_i = 1):
  - $\hat{y}_i > 1$: in the positive side and outside of positive margin - no loss.
    - $1 - \, y_i \hat{y}_i < 0$ ➔ $\xi_i = 0$
  - $0 < \hat{y}_i < 1$: still in the positive side, violates positive margin only - loss occurs.
    - $0 < 1 - y_i \hat{y}_i < 1$ ➔ $\xi_i = 1 - y_i \hat{y}_i$ (ranged between [0,1])
  - $\hat{y}_i < 0$: in the negative side, misclassification - greater loss.
    - $1 - \, y_i \hat{y}_i > 1$ ➔ $\xi_i = 1 - y_i \hat{y}_i$ (greater than 1)

Hinge loss for a negative instance (y_i = -1) is symmetrical to that of a positive instance.

Support Vector Machine

- **Support vector machine** is an extension of support vector classifier. It enlarges the original feature space by mapping them to higher dimensions, using *kernels*.

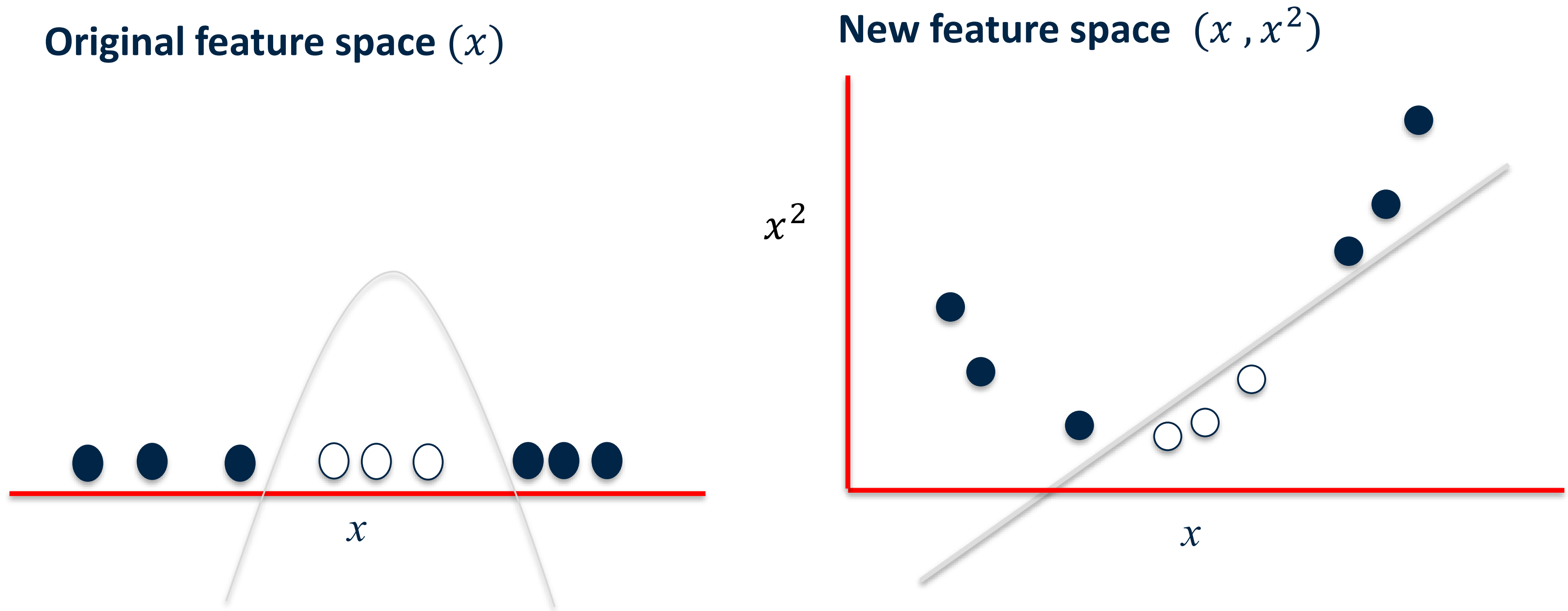
  - Linear kernel ➔ Support vector classifier
    - No exponential or interaction terms.

  - Polynomial kernel with *degree*
    - e.g., a polynomial kernel maps the original 2D feature space (X_1, X_2) to a new feature space $(X_1, X_2, X_1^2, X_2^2, X_1X_2)$.

  - Other kernels
    - Options includes "*rbf*", "*sigmoid*", "*precomputated*".

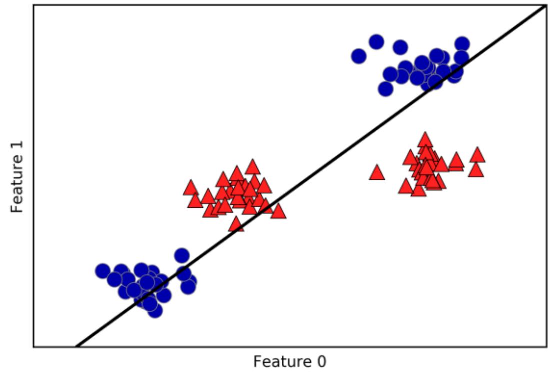
Accommodating Nonlinearity

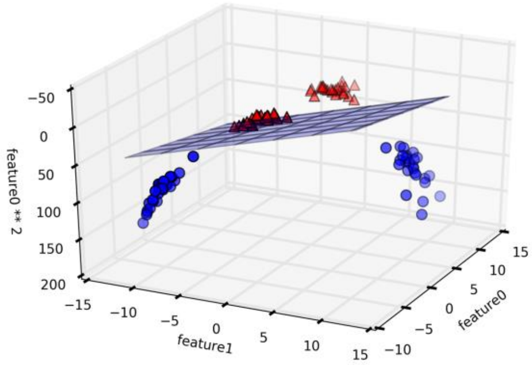
- Inseparable instances in 1D space (left) become linearly separable in a 2D feature space (right).
  - Hou would the hyperplane in 2D feature space look like in the original 1D feature space?



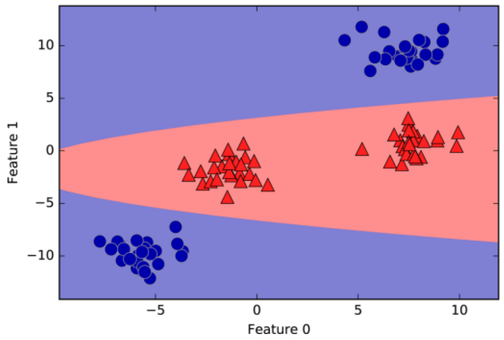
Accommodating Nonlinearity

- Inseparable instances in 2D space becomes linearly separable if instances are are mapped in a feature space of higher dimension (3D).





- What does the linear hyperplane in 3D look like in 2D? It looks non-linear.



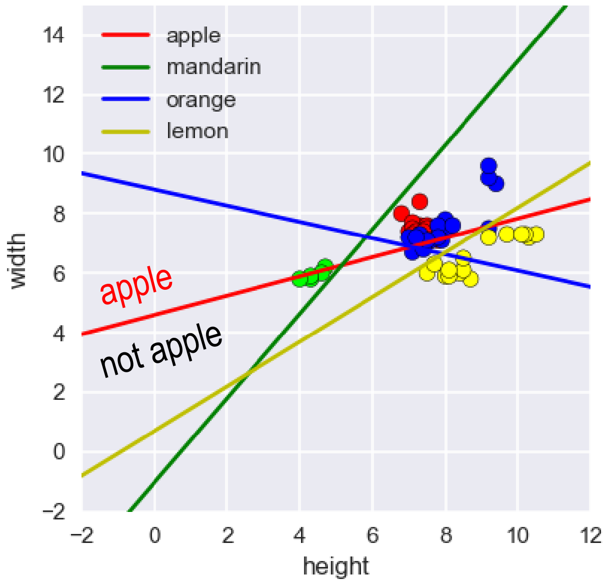
Linear Classifiers: Multi-Class Classification

- **One-vs-rest** classifier for each class in multi-class task.
  - Each set of parameters corresponds to a class.

```
clf = LinearSVC()
clf.fit(X_train, y_train)

clf.coef_
[[-0.23401135  0.72246132]
 [-1.63231901  1.15222281]
 [ 0.0849835   0.31186707]
 [ 1.26189663 -1.68097   ]]

clf.intercept_
[-3.31753728  1.19645936 -2.7468353  1.16107418]
```



$y(\text{apple}) = -3.31753728 - 0.23401135 * \text{height} + 0.72246132 * \text{width}$

- **height=2, width=6: $\hat{y}(\text{apple}) = 0.549$ (>= 0: predicted as apple)**
- **height=2, width=2: $\hat{y}(\text{apple}) = -2.340$ (< 0: predicted as others)**