EE-508: Hardware Foundations for Machine Learning

Lecture 2: Quick Review of ML Algorithms

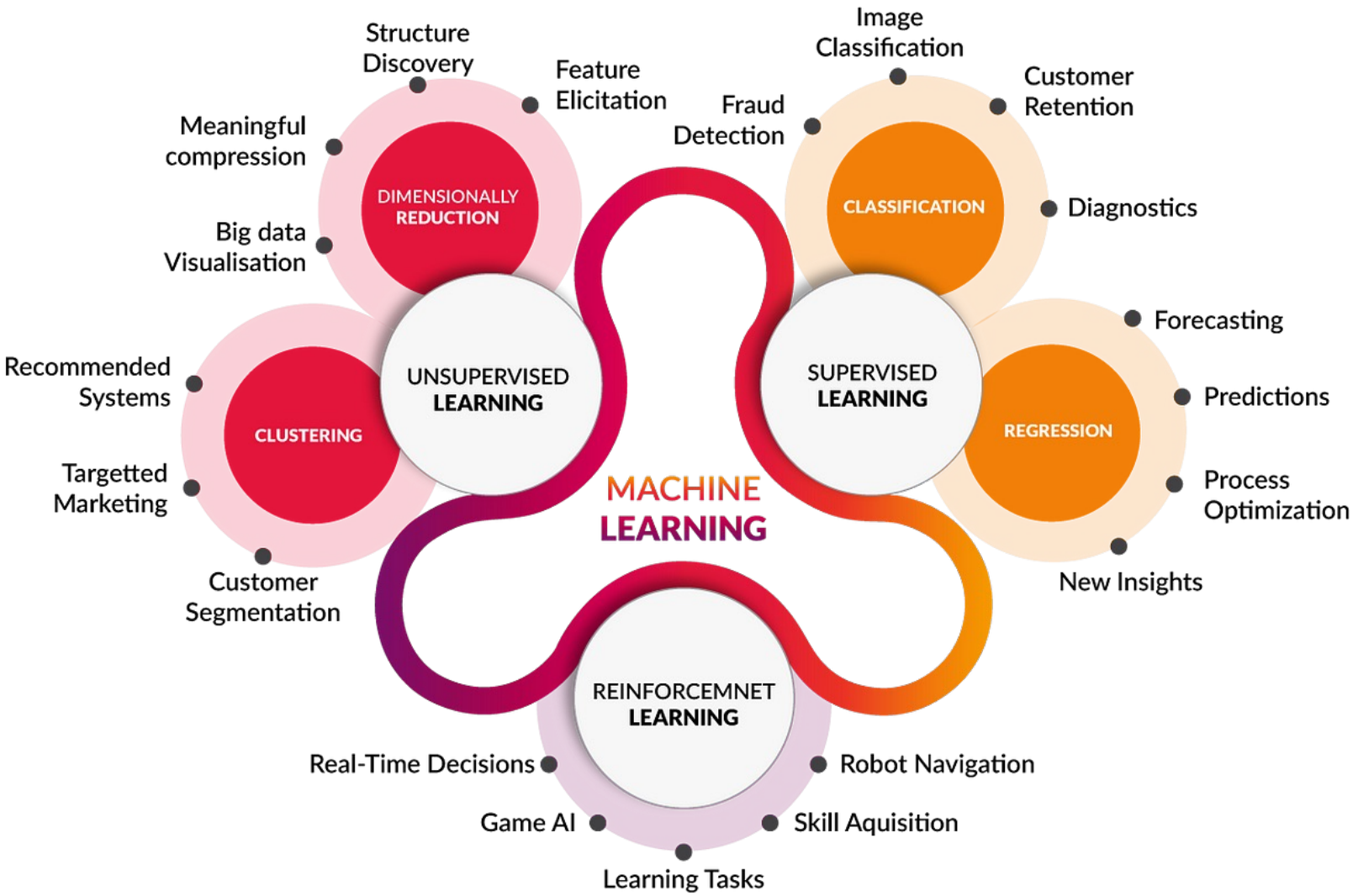
University of Southern California

Ming Hsieh Department of Electrical and Computer Engineering

Instructor:
Arash Saifhashemi

Type of Machine Learning Problems

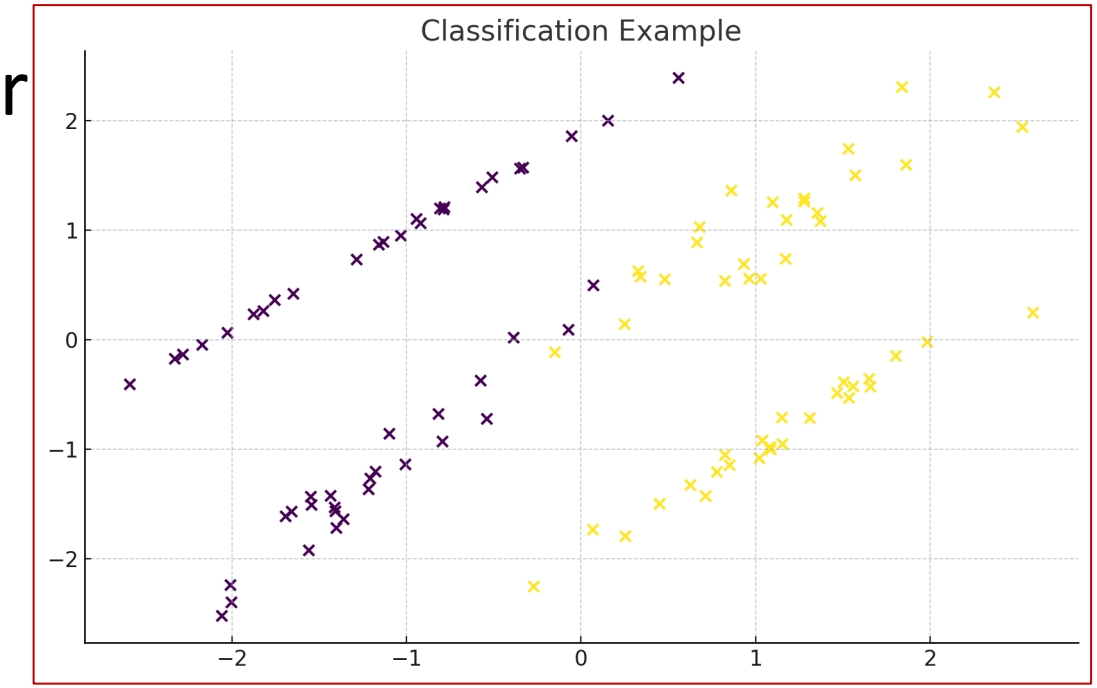
Classic Machine Learning Problem Categories



Source: http://www.cognub.com/index.php/cognitive-platform/

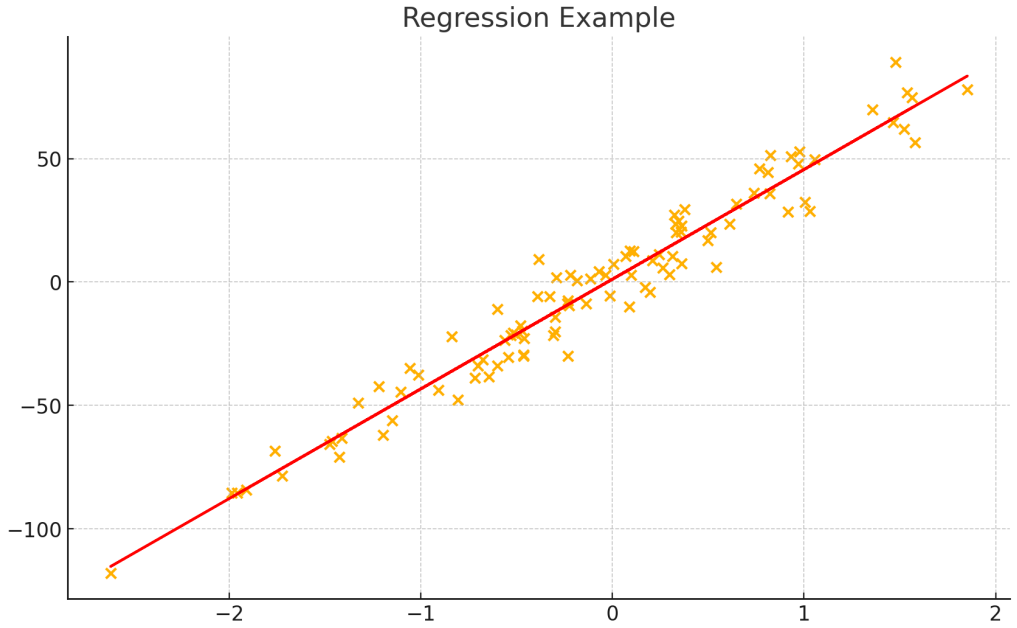
Classification

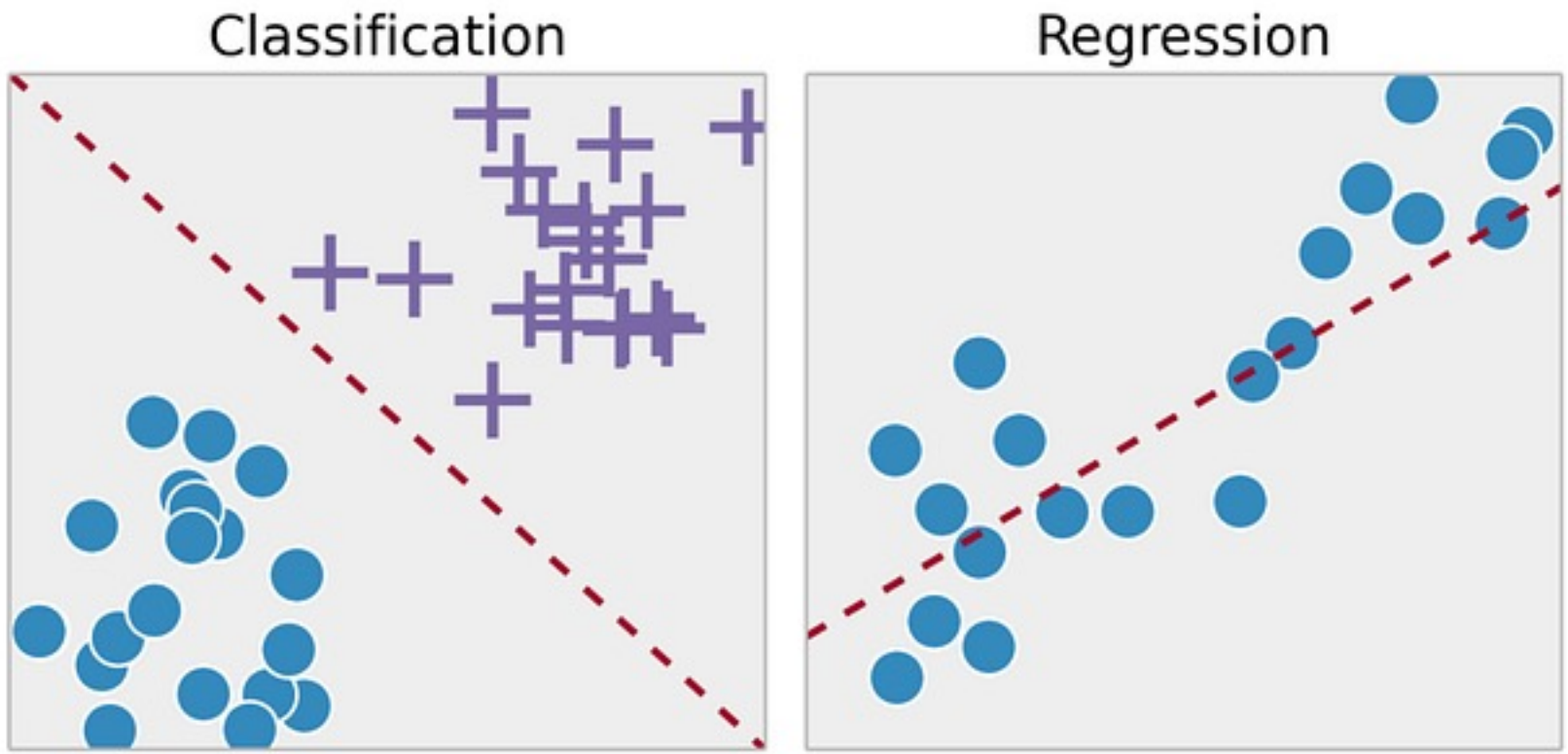
- **Definition**: Predicts a discrete label or category for given input data.
- **Examples**:
  - Email spam detection (Spam/Not Spam).
  - Disease diagnosis (Positive/Negative).
- **Characteristics**:
  - Supervised learning.
  - Output: Finite set of classes.
- **Evaluation Metrics**:
  - Accuracy, Precision, Recall, F1-Score.



Regression

- **Definition**: Predicts a continuous numerical value based on input data.
- **Examples**:
  - House price prediction.
  - Weather forecasting (temperature).
- **Characteristics**:
  - Supervised learning.
  - Output: Real numbers.
- **Evaluation Metrics**:
  - Mean Squared Error (MSE), R-squared.





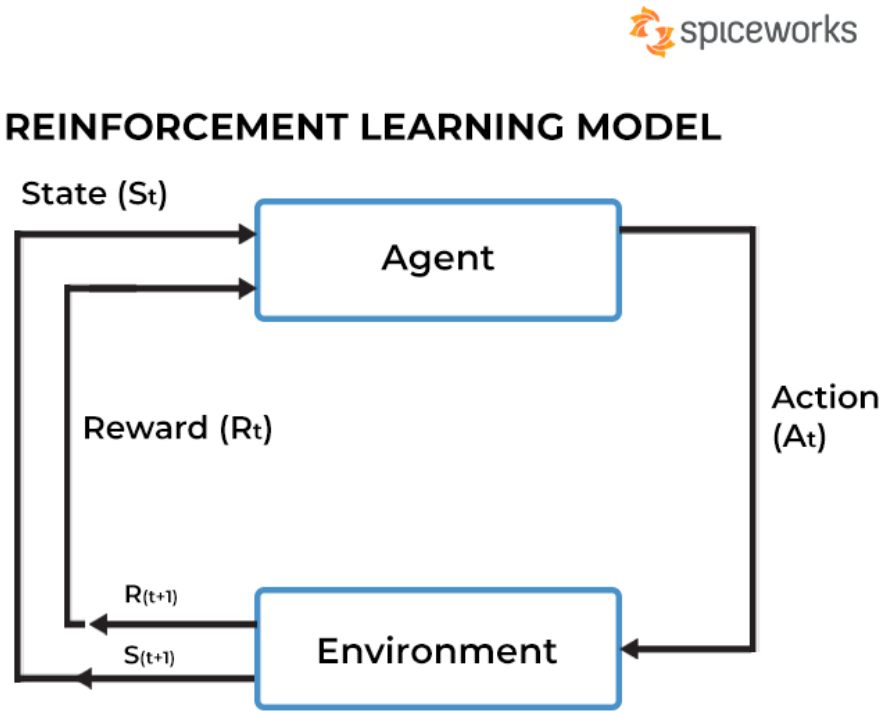
Source: https://medium.com/shecodeafrica/introduction-to-machine-learning-for-beginners-1cce26966b5c

Clustering

- **Definition**: Groups data into clusters based on similarity without predefined labels.
- **Examples**:
  - Customer segmentation.
  - Image compression.
- **Characteristics**:
  - Unsupervised learning.
  - No labeled outputs.
- **Algorithms**:
  - K-Means, Hierarchical Clustering.

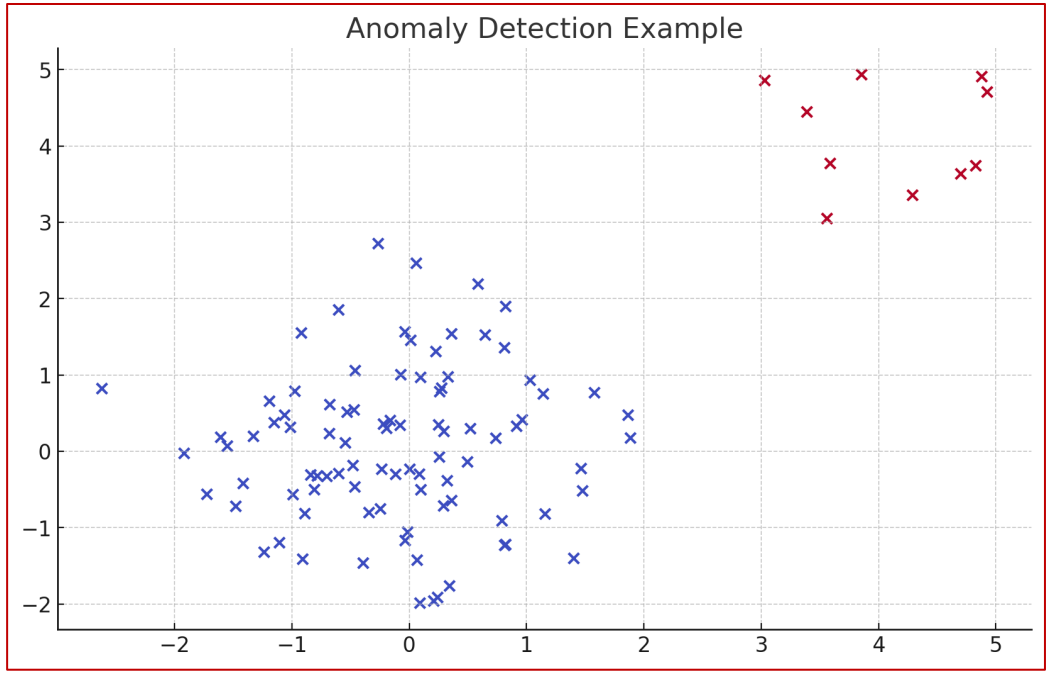
Reinforcement Learning

- **Definition**: Learns to take actions to maximize cumulative reward.
- **Examples**:
  - Game playing (Chess, Go).
  - Robotics navigation.
- **Characteristics**:
  - No explicit supervision; relies on trial and error.
  - Key Concepts: Agent, Environment, Reward.



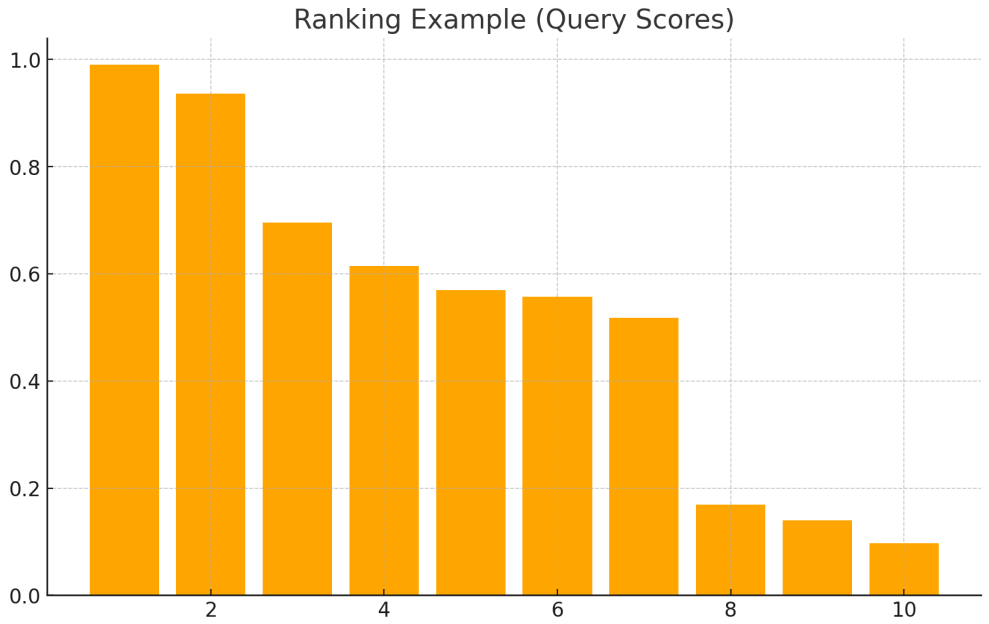
Anomaly Detection

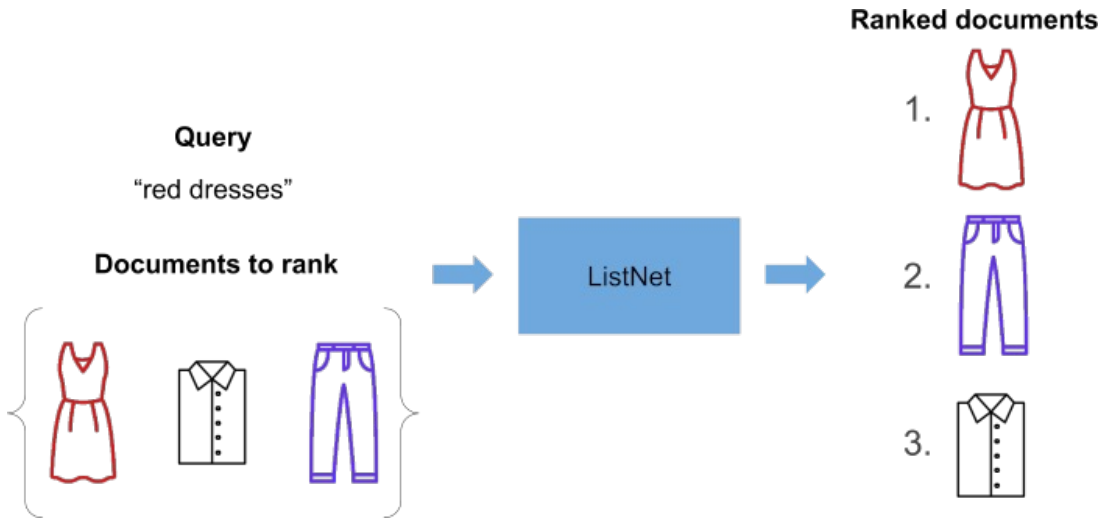
- **Definition**: Identifies rare or unusual patterns that do not conform to expected behavior.
- **Examples**:
  - Fraud detection.
  - Fault detection in manufacturing.
- **Characteristics**:
  - Often unsupervised or semi-supervised.
  - Focuses on outliers.



Ranking

- **Definition**: Assigns a rank or order to items based on relevance or priority.
- **Examples**:
  - Search engine results (ranking web pages).
  - Recommendation systems (ranking products or movies).
- **Characteristics**:
  - Supervised or unsupervised learning.
  - Output: Ordered list or scores.
- **Evaluation Metrics**:
  - Normalized Discounted Cumulative Gain (NDCG), Mean Reciprocal Rank (MRR).

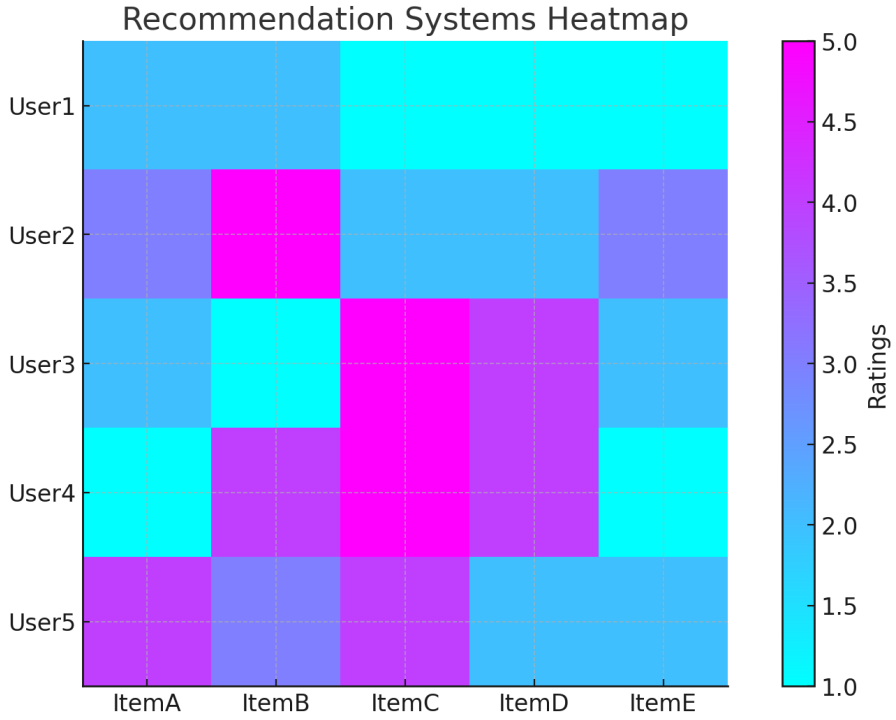


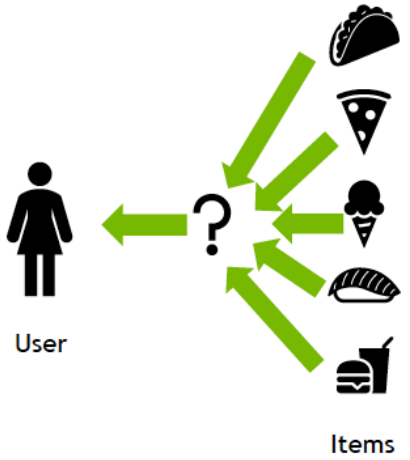


Source: embracingtherandom.com

Recommendation Systems

- **Definition**: Suggests items of interest to users based on preferences, behaviors, or similarity.
- **Examples**:
  - Movie recommendations on Netflix.
  - Product recommendations on Amazon.
- **Characteristics**:
  - Can be collaborative (user-based or item-based) or content-based.
  - Often combines supervised and unsupervised techniques.
- **Evaluation Metrics**:
  - Precision@K, Recall@K, Mean Average Precision (MAP).

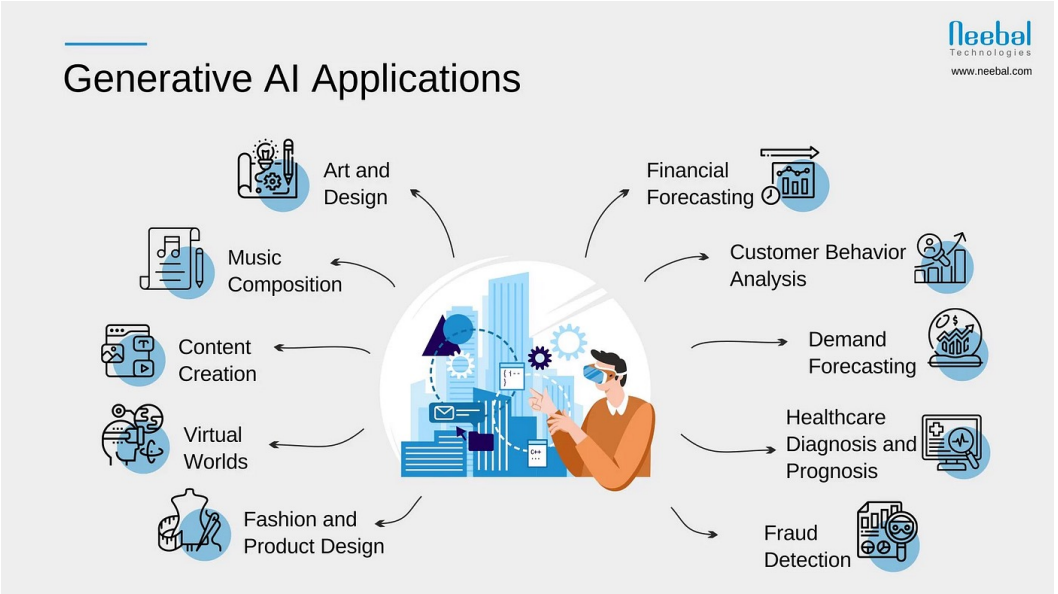




Source: nvidia.com

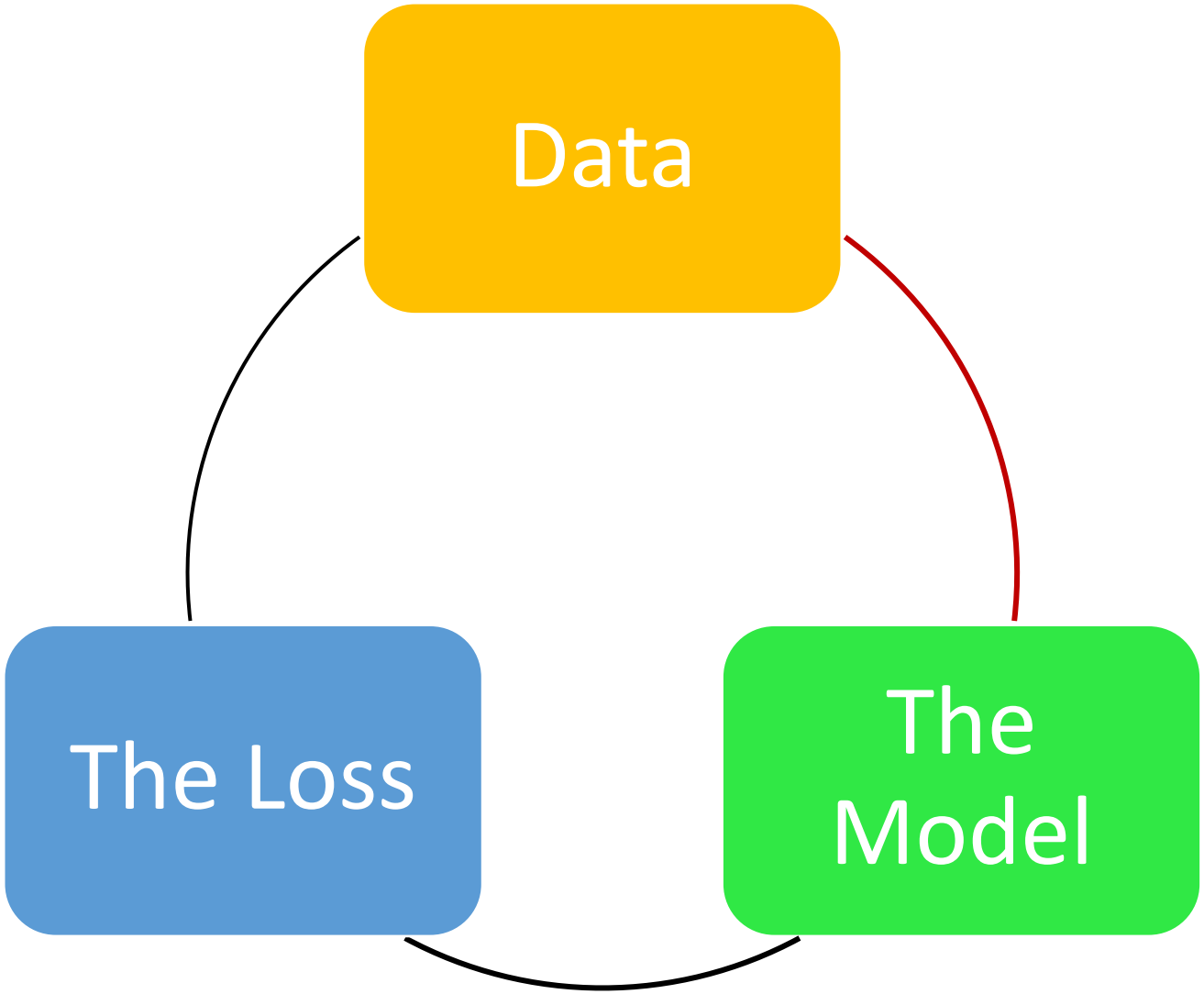
Generative AI

- **Definition**: Generating new data that resembles training data by learning its patterns.
- **Examples**:
  - Chatbots (e.g., ChatGPT).
  - Image generation (e.g., DALL-E).
  - Text summarization and translation.
- **Characteristics**:
  - Often based on transformer architectures or GANs (Generative Adversarial Networks).
  - Uses unsupervised pretraining and supervised fine-tuning or reinforcement learning.
  - Tasks like content creation and augmentation.
- **Evaluation Metrics**:
  - Perplexity, BLEU, ROUGE (text).
  - FID (Frechet Inception Distance) for images



Components of Machine Learning

Machine Learning in a Nutshell



In Machine Learning we fit a model to some data while trying to minimize the loss

Data

- Fundamental units of information in ML
  - Highly flexible in definition (e.g., images, time series, customers, diseases).
  - Quality and quantity of data directly impact model performance.

Data

- Fundamental units of information in ML
  - Highly flexible in definition (e.g., images, time series, customers, diseases).
  - Quality and quantity of data directly impact model performance.
- Dataset Parameters
  - **Sample size (m)**: Number of data points.
  - **Number of features (n)**: Properties used to describe data points.
  - Performance often improves with higher m/n ratios.

Features and Labels

- Features and Labels
  - **Features**: Input variables representing data point properties.
  - **Labels**: Output variables or targets; interchangeable with features depending on the application.

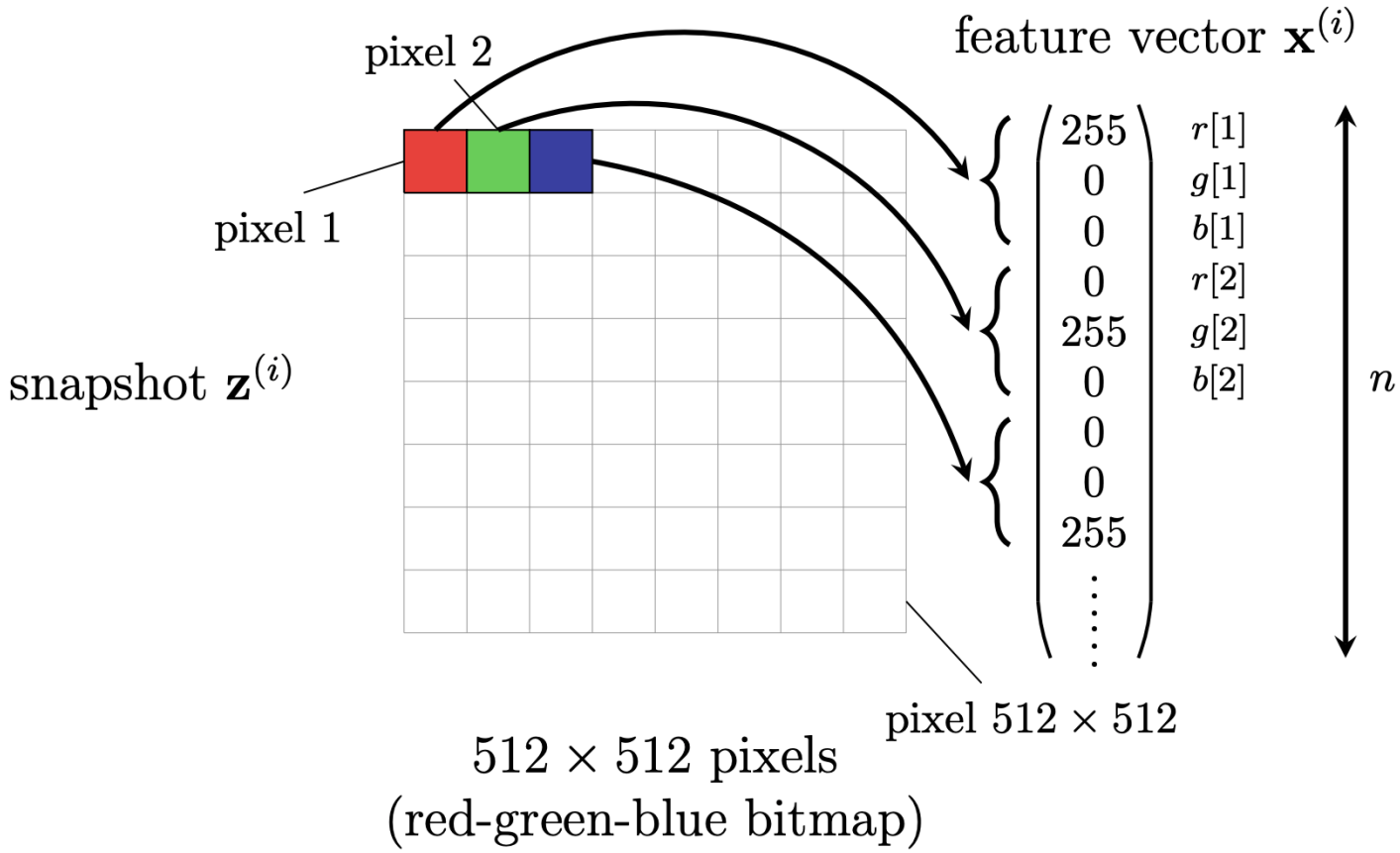
Bedrooms	Bathrooms	Square Footage	Lot Size	Year Built	Property Type	Garage Spaces	Price
3	2	1500	0.25 acres	1985	Single Family	2	$300,000
4	3	2200	0.5 acres	2000	Single Family	2	$450,000
5	4	3000	1 acre	2010	Single Family	3	$600,000

m

Features (n)

Label

Features



Source: Machine Learning: The Basics (Machine Learning: Foundations, Methodologies, and Applications) , A. Jung

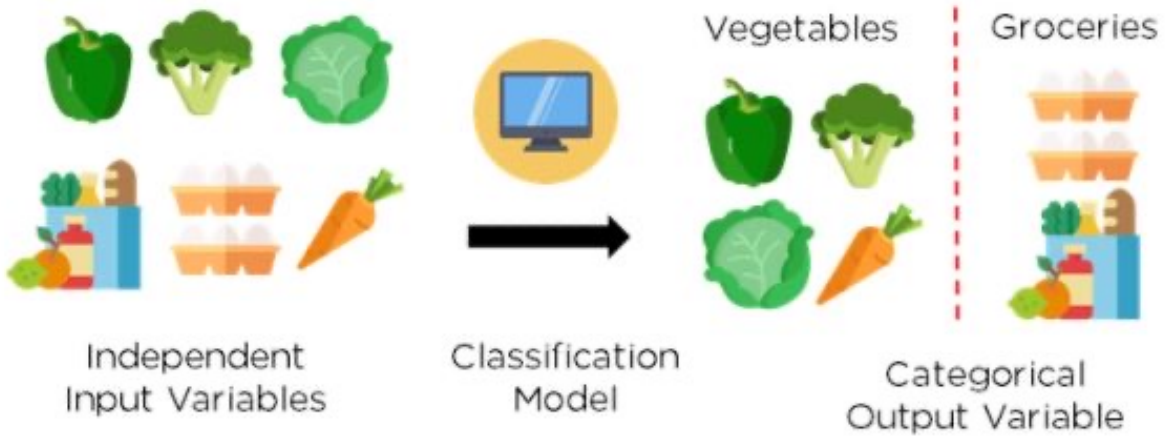
Example for an image snapshot: features $x \in \mathbb{R}^n$: the red-, green- and blue component of each pixel in the snapshot. The length of the feature vector would then be n = 3 · 512 · 512 ≈ 786000

Labels

- Types of Labels
  - **Numeric (Regression):** Real-number labels; used in regression problems (e.g., predicting house prices).

Labels

- Types of Labels
  - **Numeric (Regression):** Real-number labels; used in regression problems (e.g., predicting house prices).
  - **Categorical (Classification):** Labels indicating classes or categories (e.g., benign vs. malignant tumors).
    - **Binary Classification:** Two classes (e.g., {0, 1}).
    - **Multi-Class Classification:** More than two categories (e.g., animal species).
    - **Multi-Label Classification:** Data points can belong to multiple categories simultaneously.



Source: https://www.simplilearn.com/tutorials/machine-learning-tutorial/classification-in-machine-learning

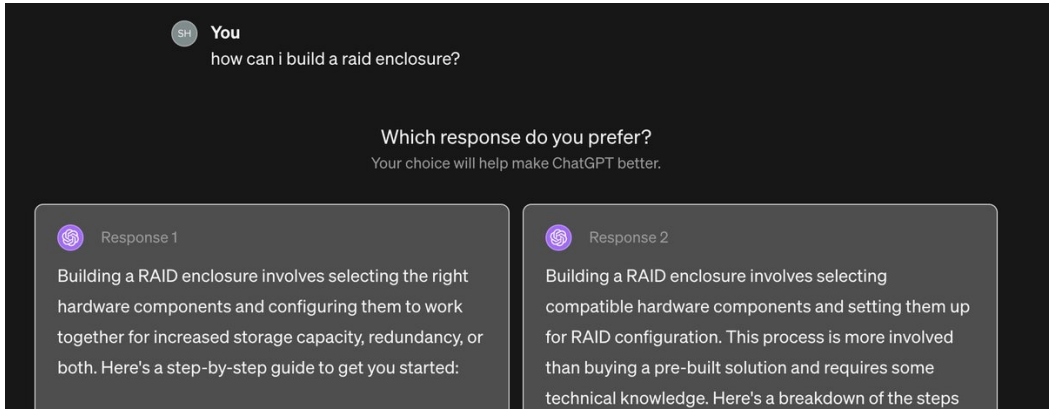
Labels

- Types of Labels
  - **Numeric (Regression):** Real-number labels; used in regression problems (e.g., predicting house prices).
  - **Categorical (Classification):** Labels indicating classes or categories (e.g., benign vs. malignant tumors).
    - **Binary Classification:** Two classes (e.g., {0, 1}).
    - **Multi-Class Classification:** More than two categories (e.g., animal species).
    - **Multi-Label Classification:** Data points can belong to multiple categories simultaneously.
  - **Ordinal:** Labels with an ordered structure (e.g., "low", "medium", "high" forest coverage).

How do We get Labels?

How do We get Labels?

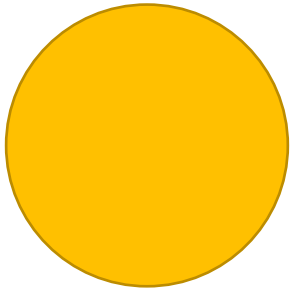
- **Labeled data** is scarce and often requires human effort which can make it expensive
  - Expert annotations
  - Crowdsourced labeling
  - User input
- **Unlabeled data** is common, making unsupervised learning valuable in extracting insights.



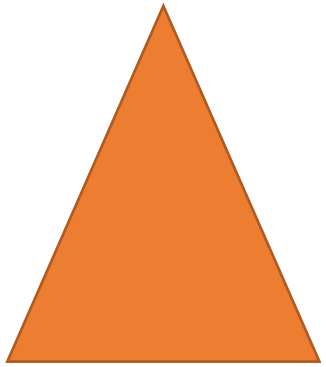
Asking user for labeling

Oval

Rectangle



Circle



Triangle

Example in advertisement

- What are the features?

Example in advertisement

- What are the features?
  - The image

Example in advertisement

- What are the features?
  - The image
  - The text

Example in advertisement

- What are the features?
  - The image
  - The text
  - The publisher's history

Example in advertisement

- What are the features?
  - The image
  - The text
  - The publisher's history
- Labels:

Example in advertisement

- What are the features?
  - The image
  - The text
  - The publisher's history
- Labels:
  - Is this ad offensive? (Yes, No)

Example in advertisement

- What are the features?
  - The image
  - The text
  - The publisher's history
- Labels:
  - Is this ad offensive? (Yes, No)
  - What is it about?

Example in advertisement

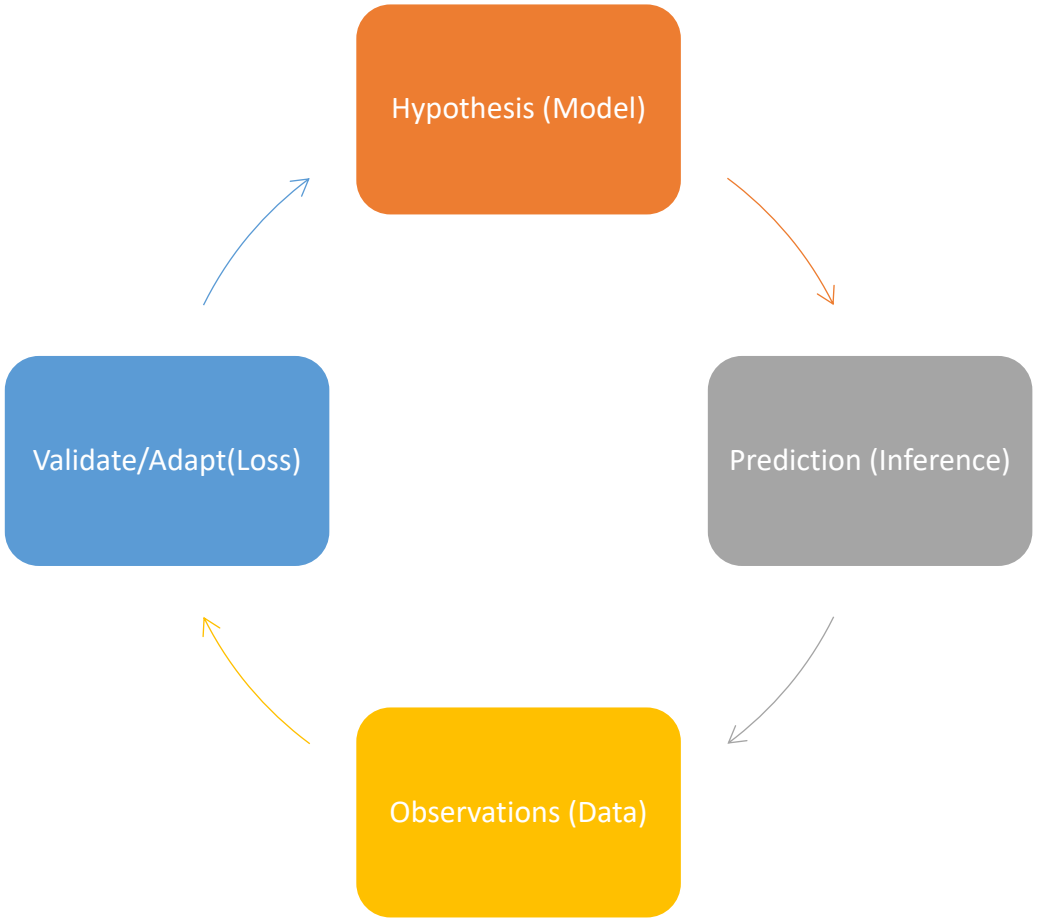
- What are the features?
  - The image
  - The text
  - The publisher's history
- Labels:
  - Is this ad offensive? (Yes, No)
  - What is it about?
    - Crib, baby, family planning, ...

Example in advertisement

- What are the features?
  - The image
  - The text
  - The publisher's history
- Labels:
  - Is this ad offensive? (Yes, No)
  - What is it about?
    - Crib, baby, family planning, ...
  - Does it violate any policy?
    - Violence
    - Sensitive contents
    - Child safety

The Model

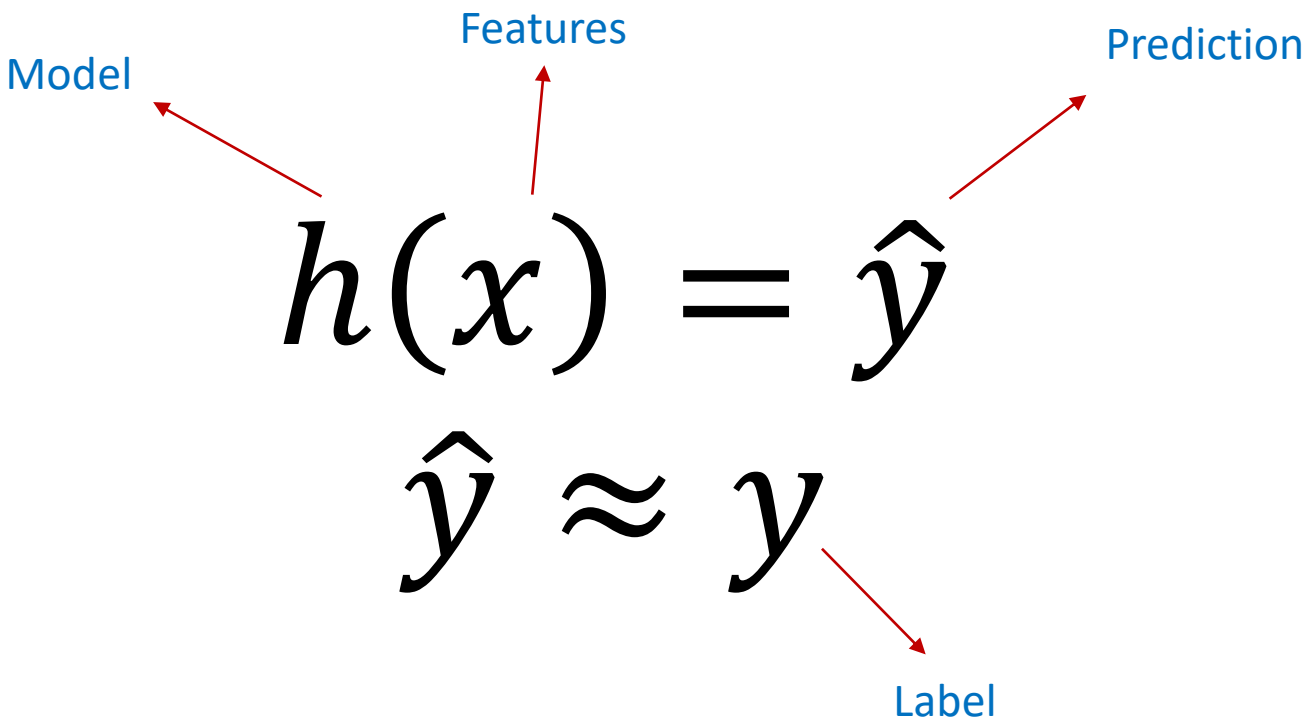
- Machine Learning is based on Trial and Error principle



We want our model prediction to be as close as possible to the actual labels

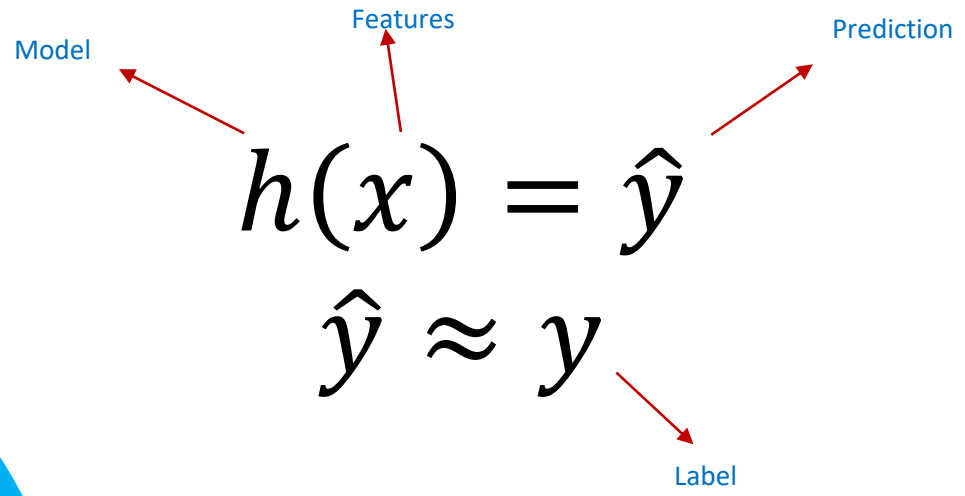
Objective of ML Methods

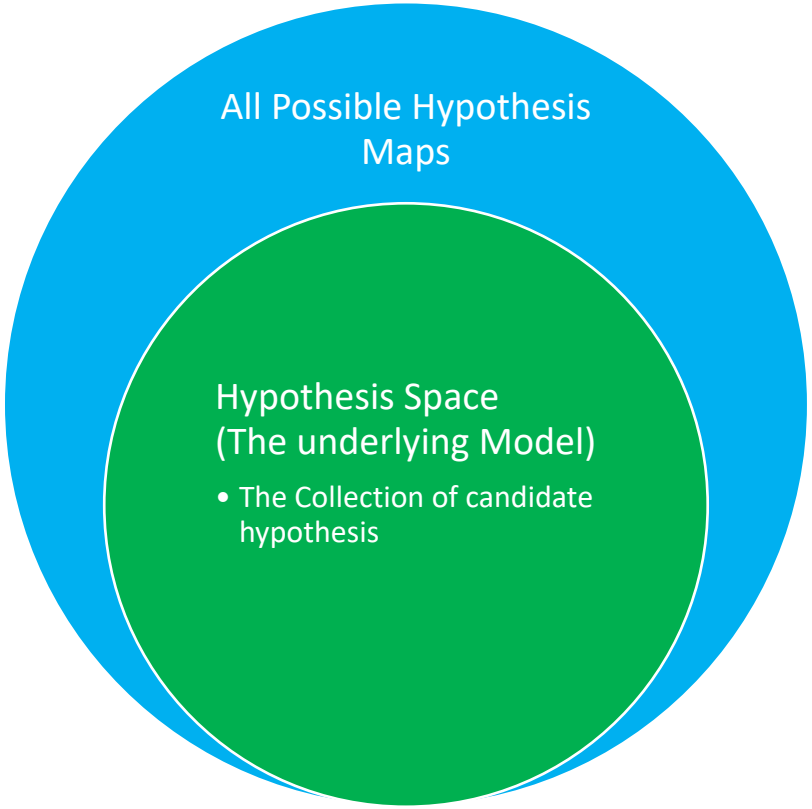
- Build a predictor



We want our model prediction to be as close as possible to the actual labels

Hypothesis Space





Example: In linear regression, the hypothesis space includes all linear functions of the form

$$y=w_1x_1 + w_1x_1+w_1x_1+\cdots+w_1x_1+b$$

Three Flavors of ML

Supervised Learning

- Learning from labeled data (e.g. classification, regression)

Unsupervised Learning

- Discovering patterns on unlabeled data

Reinforcement Learning

- Learning through trial-error interactions with the environment to maximize reward

Other Flavors of ML

Semi-Supervised Learning

- Combining a small amount of labeled data with a large amount of unlabeled data
- E.g. Train on the labeled data, then use the model to label the rest of data. Repeat until convergence.

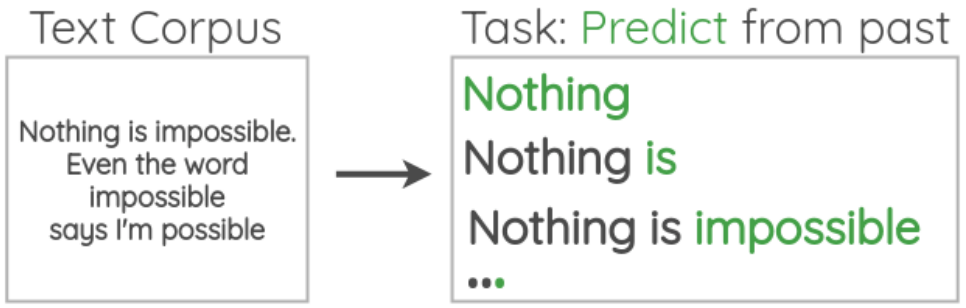
Other Flavors of ML

Semi-Supervised Learning

- Combining a small amount of labeled data with a large amount of unlabeled data
- E.g. Train on the labeled data, then use the model to label the rest of data. Repeat until convergence.

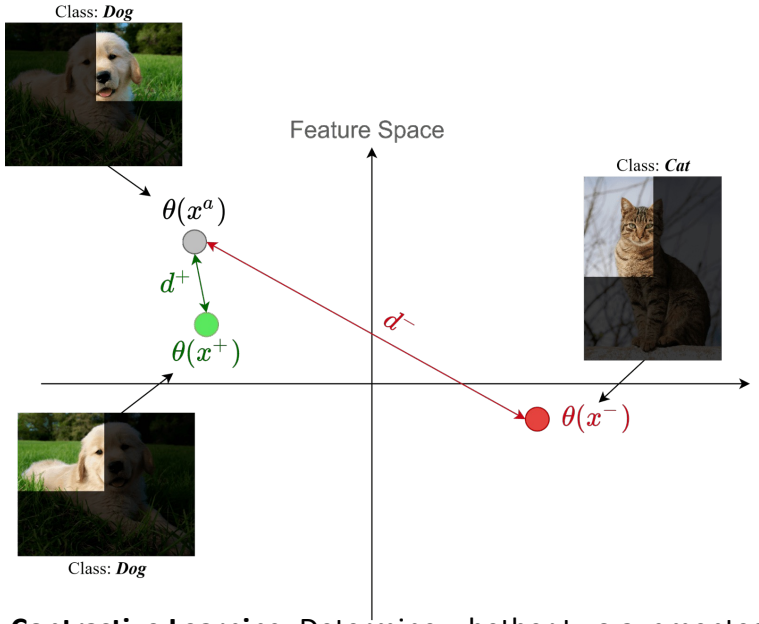
Self-Supervised Learning

- Generating labels from the raw data
- Used in NLP or Vision
- E.g. randomly removing some words and using it as a label.



https://amitness.com/posts/self-supervised-learning-nlp

Using the next word as label



Contrastive Learning: Determine whether two augmented views of the same input belong to the same class.

Other Flavors of ML

Semi-Supervised Learning

- Combining a small amount of labeled data with a large amount of unlabeled data
- E.g. Train on the labeled data, then use the model to label the rest of data. Repeat until convergence.

Self-Supervised Learning

- Generating labels from the raw data
- Used in NLP or Vision
- E.g. randomly removing some words and using it as a label.

Online Learning (Incremental)

- Continuously updating the model in real-time

Other Flavors of ML

Semi-Supervised Learning

- Combining a small amount of labeled data with a large amount of unlabeled data
- E.g. Train on the labeled data, then use the model to label the rest of data. Repeat until convergence.

Self-Supervised Learning

- Generating labels from the raw data
- Used in NLP or Vision
- E.g. randomly removing some words and using it as a label.

Online Learning (Incremental)

- Continuously updating the model in real-time

Transfer Learning

- Using knowledge from one task to enhance learning in a related task

Other Flavors of ML

Semi-Supervised Learning

- Combining a small amount of labeled data with a large amount of unlabeled data
- E.g. Train on the labeled data, then use the model to label the rest of data. Repeat until convergence.

Self-Supervised Learning

- Generating labels from the raw data
- Used in NLP or Vision
- E.g. randomly removing some words and using it as a label.

Online Learning (Incremental)

- Continuously updating the model in real-time

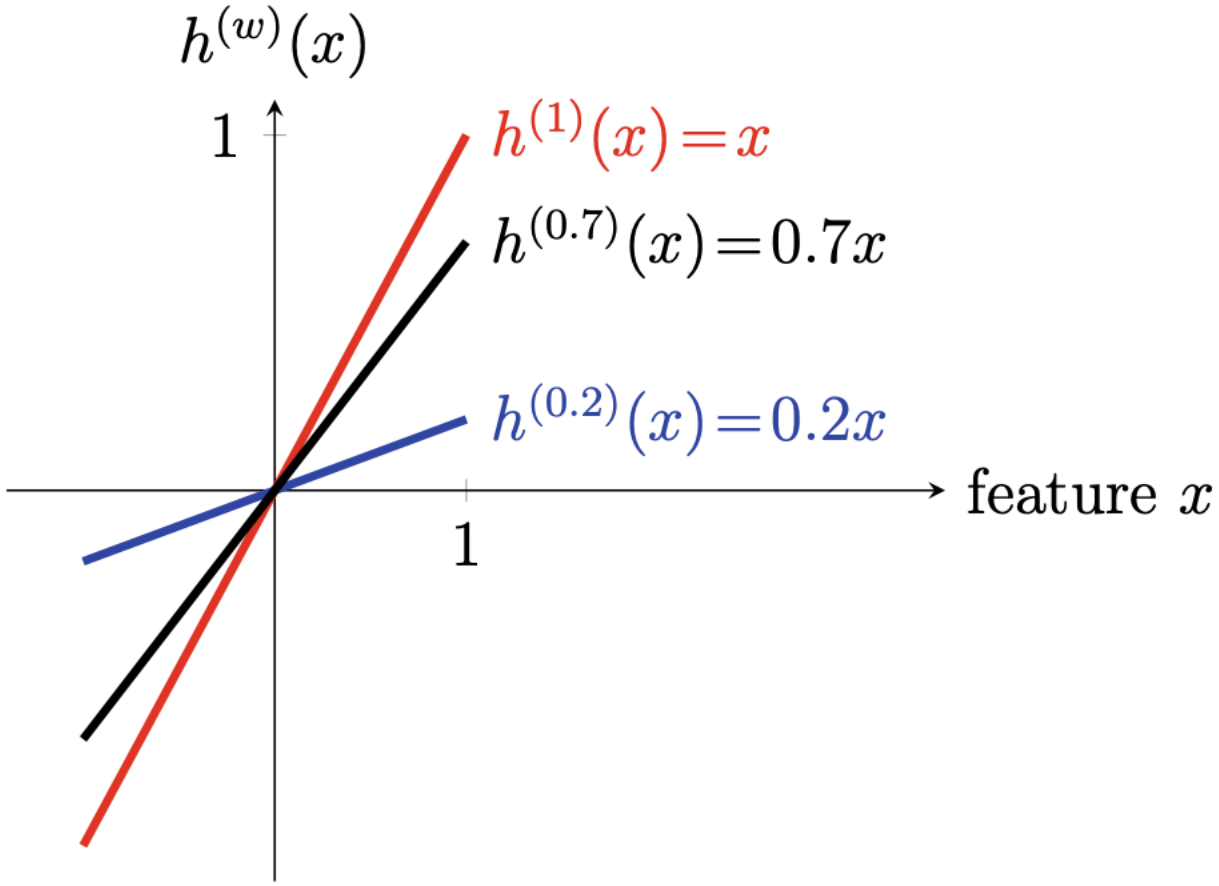
Transfer Learning

- Using knowledge from one task to enhance learning in a related task

Evolutionary Algorithms (Genetic Algorithms)

- Optimization inspired by natural selection

Example of Parameterized Model

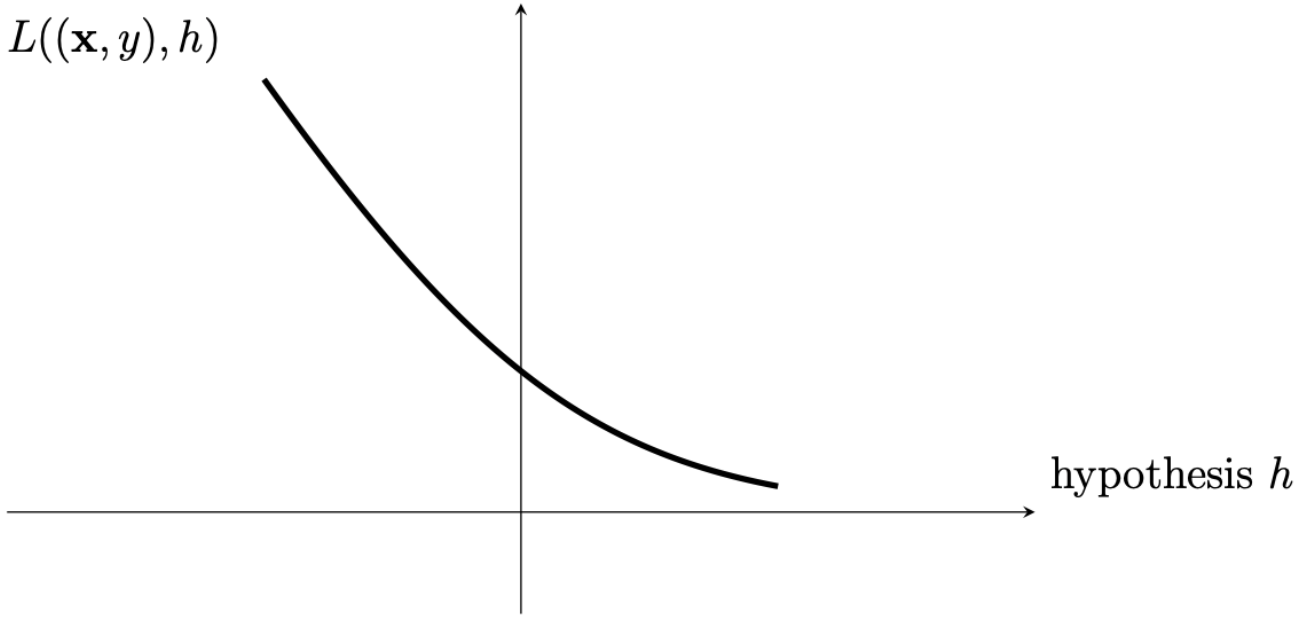


Source: Machine Learning: The Basics (Machine Learning: Foundations, Methodologies, and Applications) , A. Jung

$$\mathcal{H} = \{h^{(w)} : \mathbb{R} \to \mathbb{R}, h^{(w)}(x) = w \cdot x\}$$

Hypothesis Space+

The Loss



Source: Machine Learning: The Basics (Machine Learning: Foundations, Methodologies, and Applications) , A. Jung

$$L : \mathcal{X} \times \mathcal{Y} \times \mathcal{H} \to \mathbb{R}_+ : \big((\mathbf{x}, y), h\big) \mapsto L((\mathbf{x}, y), h)$$

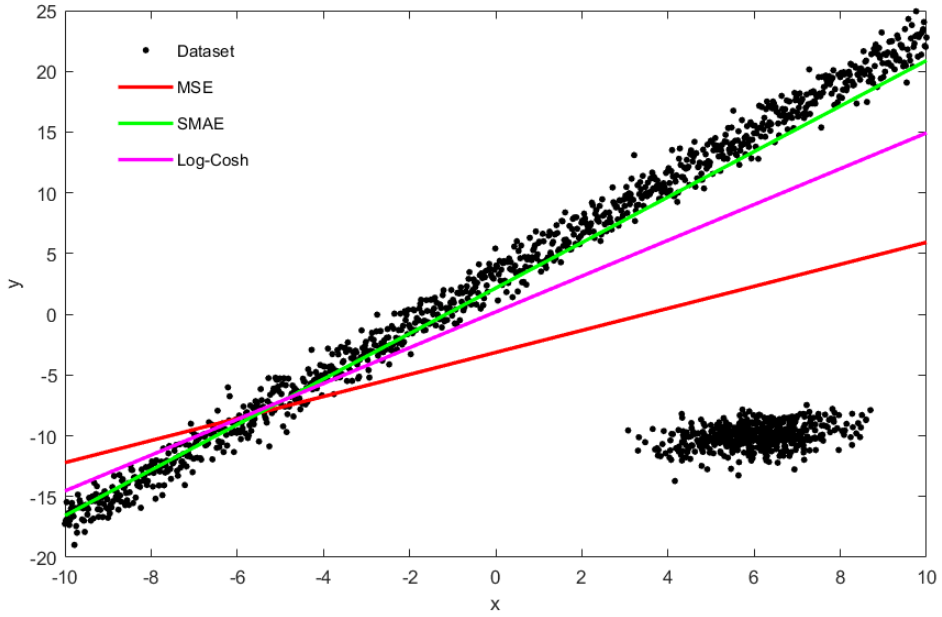
ML methods try to find (learn) a hypothesis that incurs minimum loss

Different Aspects of the Loss Function

- **Computational Aspects:**
  - Convex and differentiable loss functions (e.g. logistic loss) allow efficient optimization using gradient descent.

Different Aspects of the Loss Function

- **Computational Aspects:**
  - Convex and differentiable loss functions (e.g. logistic loss) allow efficient optimization using gradient descent.
- **Statistical Aspects:**
  - Some loss functions handle outliers better.
  - Probabilistic methods can guide loss function design.

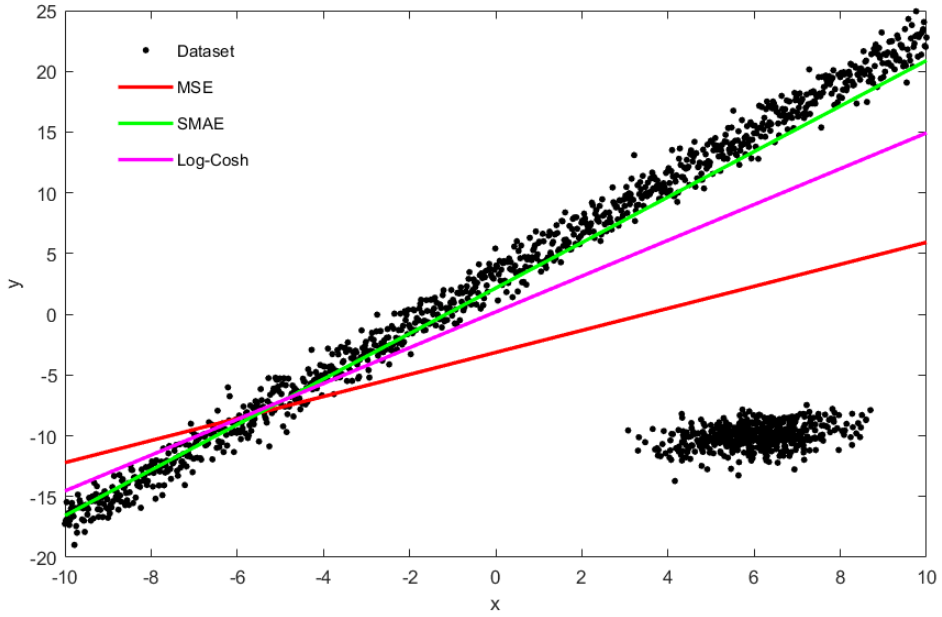


Source: Wikipedia

Effect of using different loss functions, when the data has outliers.
The symmetric mean absolute percentage error (SMAPE or sMAPE) seems to result a better model

Different Aspects of the Loss Function

- **Computational Aspects:**
  - Convex and differentiable loss functions (e.g. logistic loss) allow efficient optimization using gradient descent.
- **Statistical Aspects:**
  - Some loss functions handle outliers better.
  - Probabilistic methods can guide loss function design.
- **Interpretability:**
  - Loss functions like 0/1 loss are easily interpretable.
  - It's not easy to interpret values (e.g. 0.81) of some arbitrary loss function.

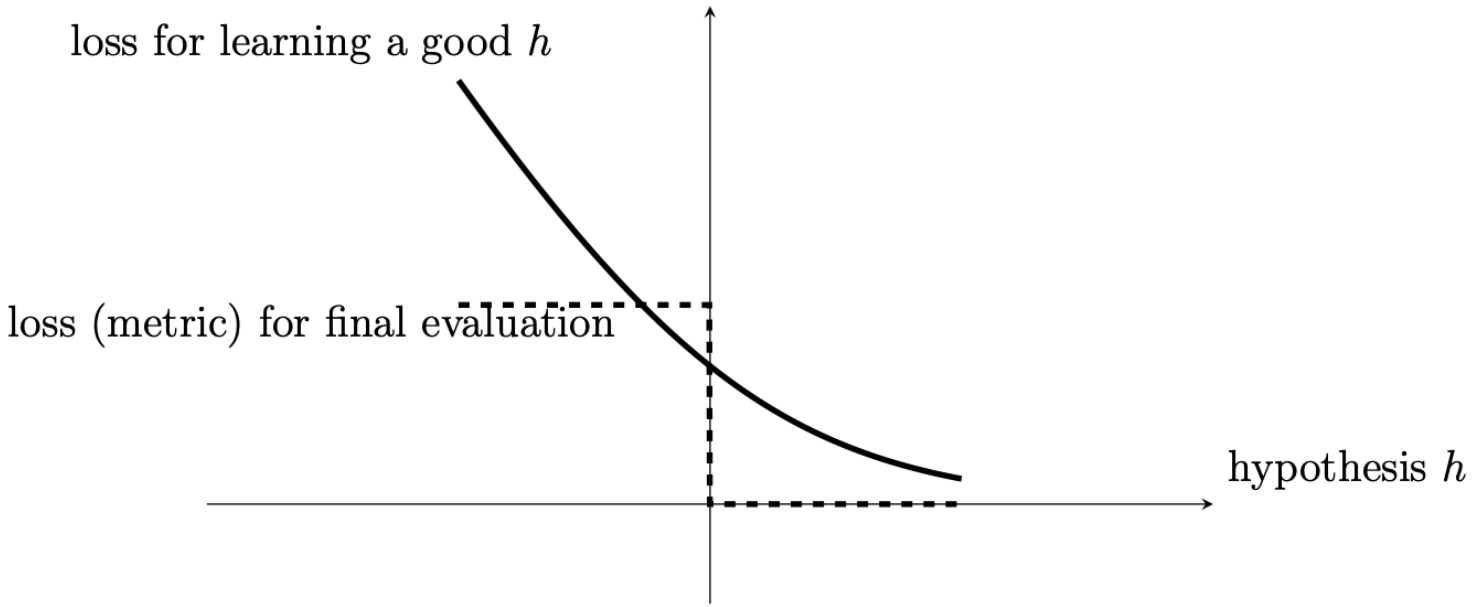


Source: Wikipedia

Effect of using different loss functions, when the data has outliers.
The symmetric mean absolute percentage error (SMAPE or sMAPE) seems to result a better model

The Role of the Loss Function

- Trade-off
  - When selecting the loss function, we are often trading off the above aspects



Source: Machine Learning: The Basics (Machine Learning: Foundations, Methodologies, and Applications) , A. Jung

- Two different loss functions for a given data point and varying hypothesis h.
- One loss function (solid curve) is used to learn a good hypothesis by minimizing the loss.
- Another loss function (dashed curve) is used for the final performance evaluation of the learnt hypothesis.
- The loss function used for the final performance evaluation is referred to as a metric

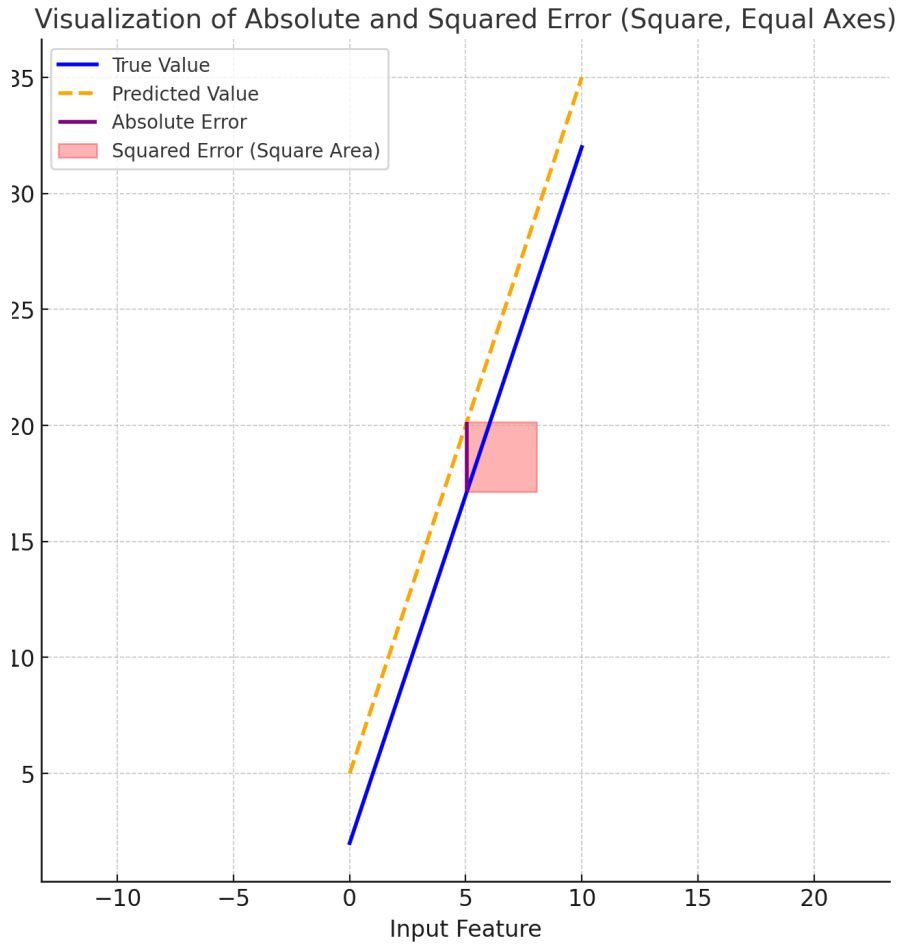
Loss Function For Numerical Labels

Squared Error (SE):

- Measures the square of the difference between the true value (y) and the predicted value ($\hat{y}$).
- Formula: $(y - \hat{y})^2$
- **Key Features**:
  - Penalizes larger errors more heavily due to squaring.
  - Not directly interpretable because it's not in the same units as the original data.

Mean Squared Error (MSE):

- The average of all squared errors across the dataset.
- Formula: $\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$
- **Key Features**:
  - Used in regression tasks to evaluate overall model performance.
  - Sensitive to outliers due to squaring.



Mean Squared Error (MSE): 9.00

Mean Absolute Error (MAE): 3.00

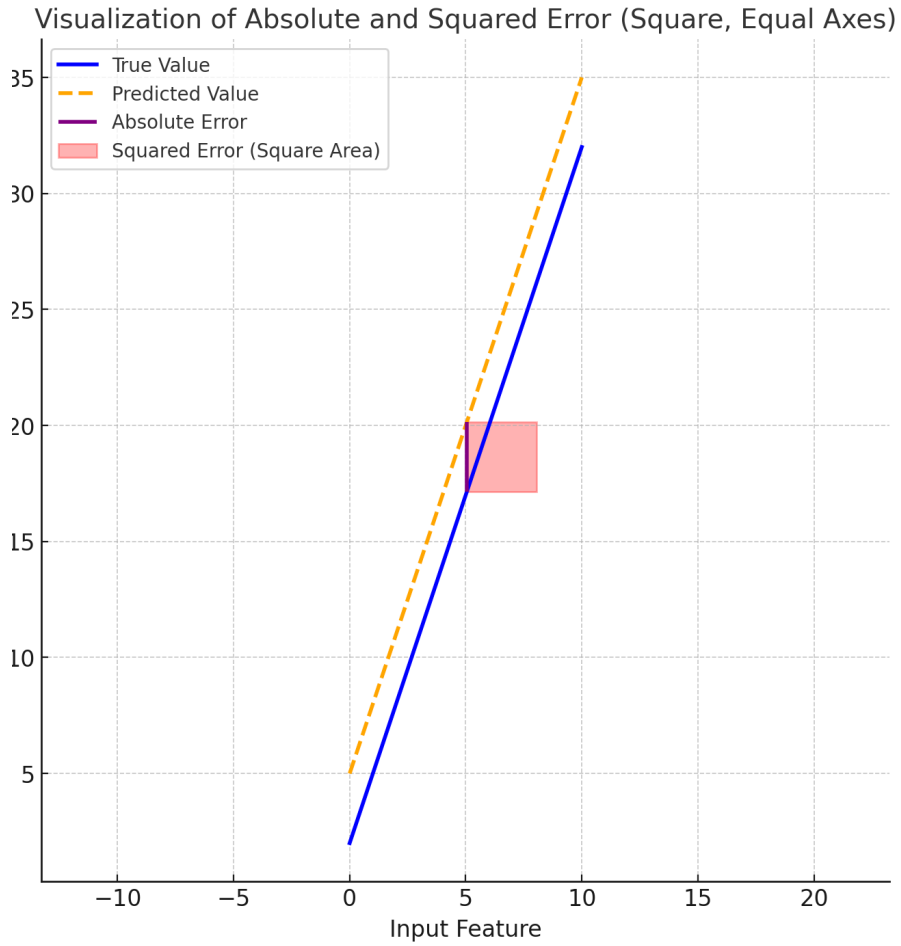
Loss Function For Numerical Labels

Squared Error (SE):

- Measures the square of the difference between the true value (y) and the predicted value ($\hat{y}$).
- Formula: $(y - \hat{y})^2$
- **Key Features**:
  - Penalizes larger errors more heavily due to squaring.
  - Not directly interpretable because it's not in the same units as the original data.

Mean Squared Error (MSE):

- The average of all squared errors across the dataset.
- Formula: $\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$
- **Key Features**:
  - Used in regression tasks to evaluate overall model performance.
  - Sensitive to outliers due to squaring.



- **SE/MSE**: Prioritize penalizing larger errors, commonly used for optimizing models due to differentiability.
- **AE/MAE**: Focus on direct, interpretable average error, especially useful when outliers are less critical.

Loss Function For Categorical Labels

- SE loss minimizes the difference between the predicted values and the true labels (treated as continuous values).
- Classification aims to assign correct **discrete labels**, not minimize numerical differences.

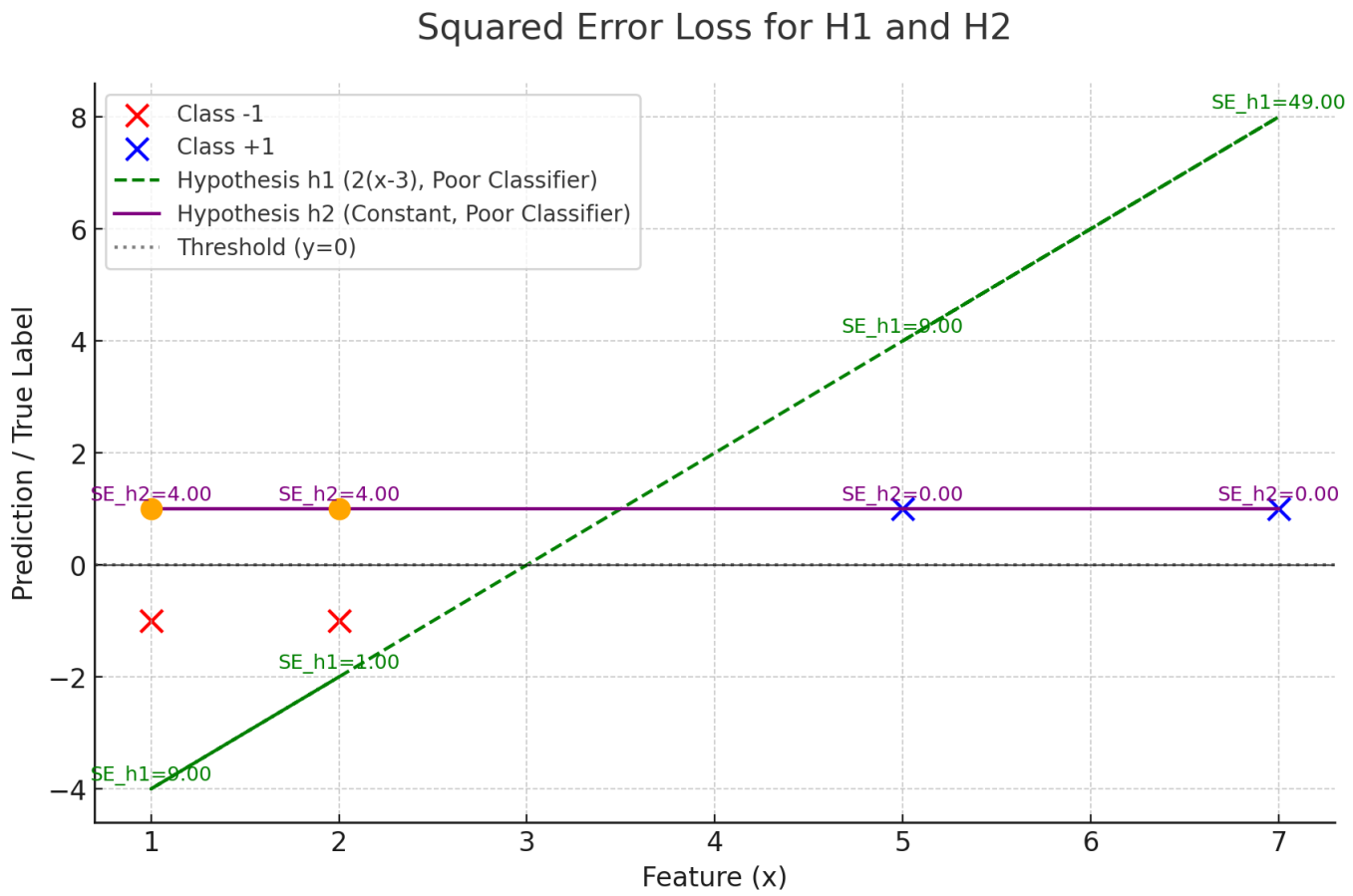
Loss Function For Categorical Labels

- SE loss minimizes the difference between the predicted values and the true labels (treated as continuous values).
- Classification aims to assign correct **discrete labels**, not minimize numerical differences.

Example: consider two classifiers: H1 and H2

x is classified as **1** if $h(x) \geq 0$.

x is classified as **-1** if $h(x) < 0$.



Loss Function For Categorical Labels

- SE loss minimizes the difference between the predicted values and the true labels (treated as continuous values).
- Classification aims to assign correct **discrete labels**, not minimize numerical differences.

Example: consider two classifiers: H1 and H2

x is classified as **1** if $h(x) \geq 0$.

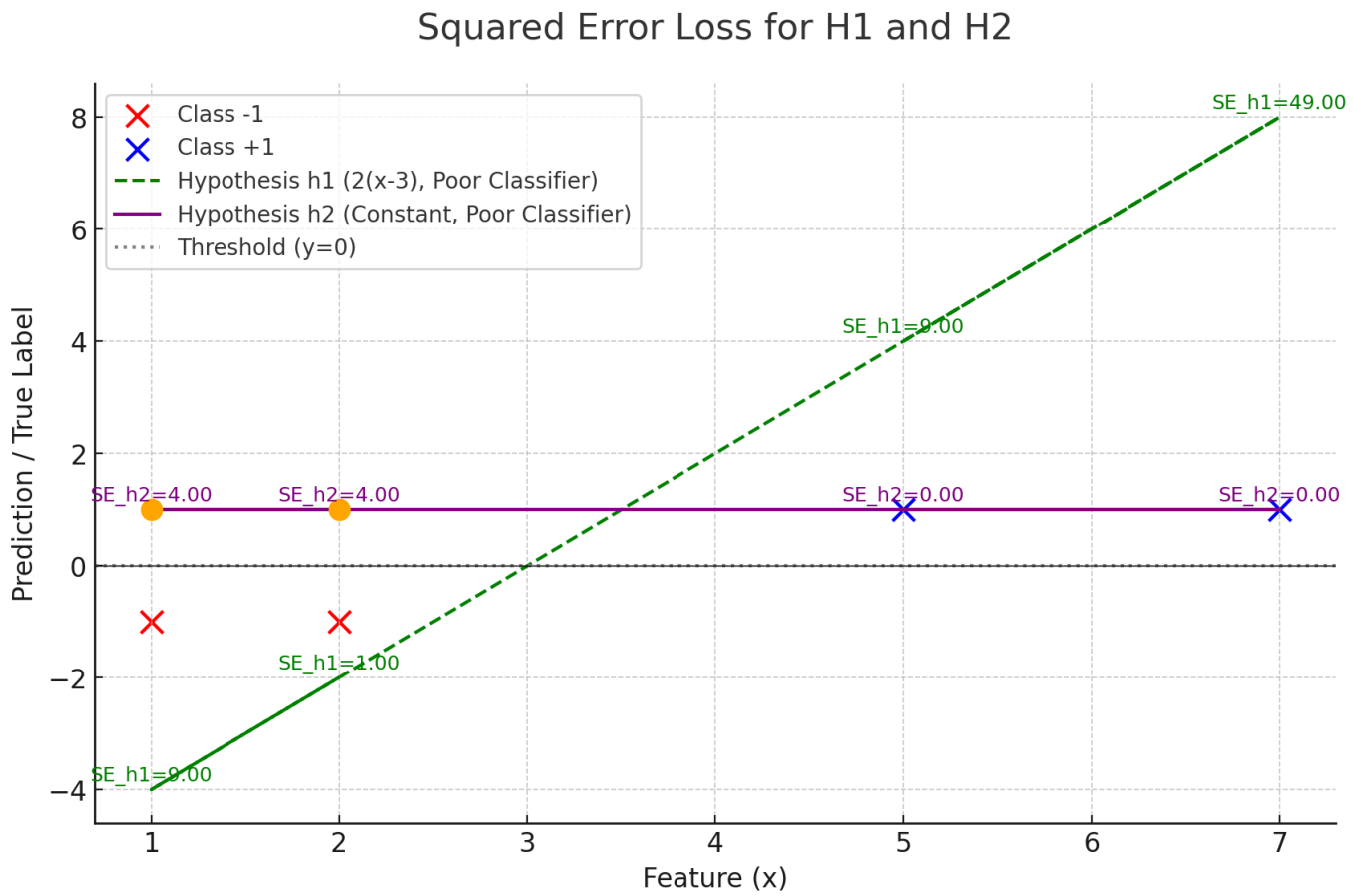
x is classified as **-1** if $h(x) < 0$.

$$\text{MSE}_{h_1} = 17.0$$

$$\text{MSE}_{h_2} = 2.0$$

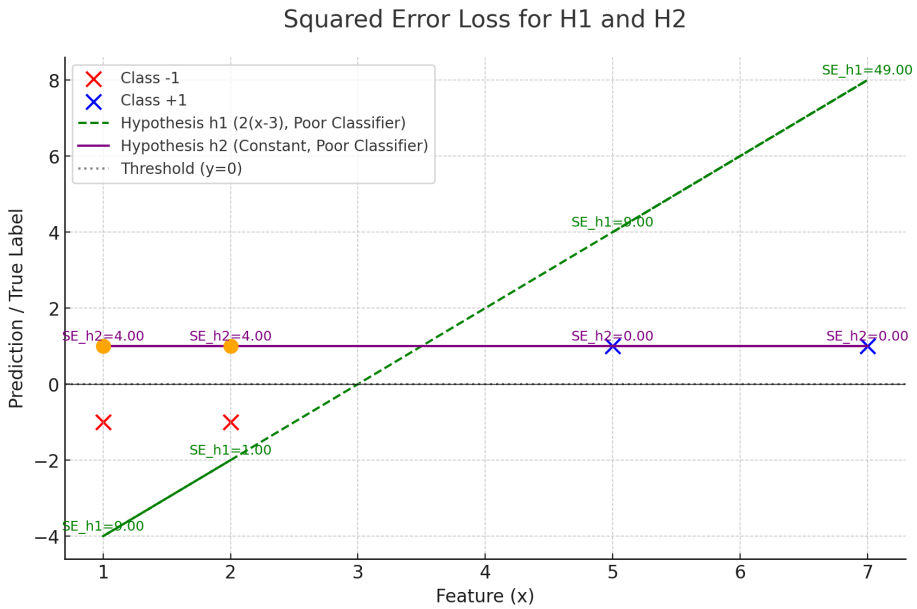
While h1 has a higher MSE, it is a perfect one!

SE is not a good loss function for categorical labels



Loss Function For Categorical Labels

- **Desired Characteristics of a Loss Function**:
  - **Punish Wrong Classifications**: A good loss function should give large values for wrong classifications ($\hat{y} \neq y$), especially when the model is confident ($|h(x)|$ is large).
  - **Reward Correct Classifications**: It should give small values for correct classifications ($\hat{y} = y$), especially when the model is confident ($|h(x)|$ is large).
- **Issue with Squared Loss**:
  - Squared loss penalizes predictions based on the difference between $h(x)$ and y, regardless of correctness.
  - If $|h(x)|$ (model confidence) is large, squared loss produces large penalties, even when the classification is correct ($\hat{y} = y$).
  - It does not differentiate between confidence in correct and incorrect predictions.



The 0/1 Loss

For a single data point:

$$L(y, \hat{y}) = \begin{cases} 0, & \text{if } \hat{y} = y \text{ (correct prediction)} \\ 1, & \text{if } \hat{y} \neq y \text{ (incorrect prediction)} \end{cases}$$

The 0/1 Loss

For a single data point:

$$L(y, \hat{y}) = \begin{cases} 0, & \text{if } \hat{y} = y \text{ (correct prediction)} \\ 1, & \text{if } \hat{y} \neq y \text{ (incorrect prediction)} \end{cases}$$

For a dataset with n samples:

$$\text{Average } 0/1 \text{ Loss} = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}(\hat{y}_i \neq y_i)$$

$\mathbb{I}(\hat{y}_i \neq y_i)$: An indicator function that is 1 if $\hat{y}_i \neq y_i$ (misclassification) and 0 otherwise.

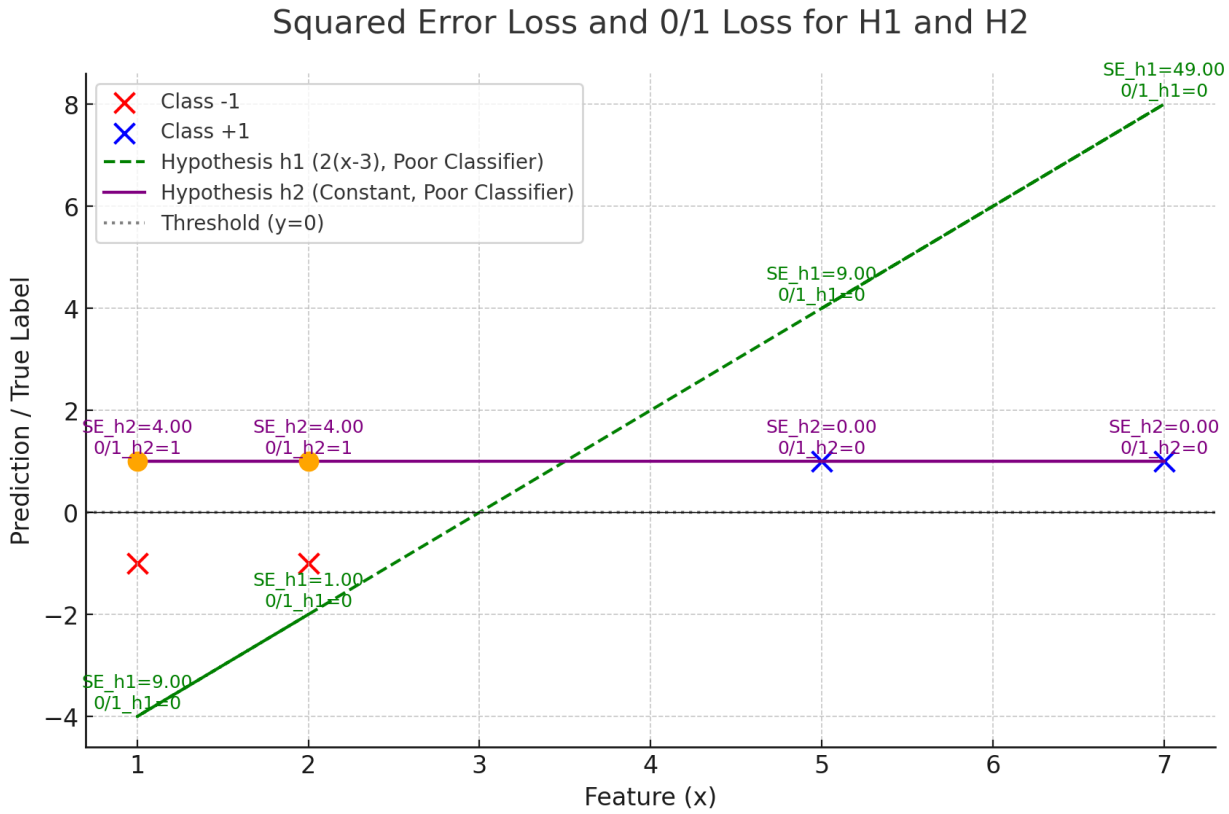
The 0/1 Loss

For a single data point:

$$L(y, \hat{y}) = \begin{cases} 0, & \text{if } \hat{y} = y \text{ (correct prediction)} \\ 1, & \text{if } \hat{y} \neq y \text{ (incorrect prediction)} \end{cases}$$

For a dataset with n samples:

$$\text{Average } 0/1 \text{ Loss} = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}(\hat{y}_i \neq y_i)$$



$\mathbb{I}(\hat{y}_i \neq y_i)$: An indicator function that is 1 if $\hat{y}_i \neq y_i$ (misclassification) and 0 otherwise.

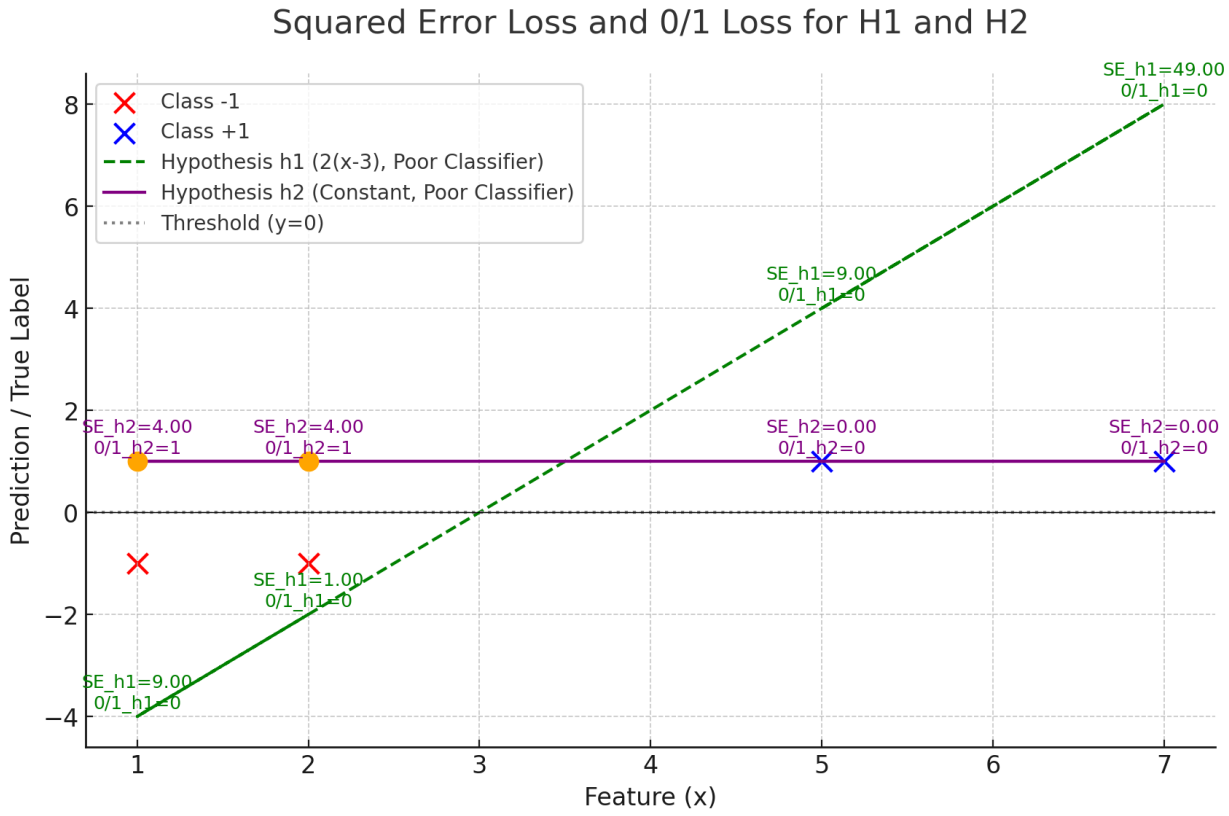
The 0/1 Loss

For a single data point:

$$L(y, \hat{y}) = \begin{cases} 0, & \text{if } \hat{y} = y \text{ (correct prediction)} \\ 1, & \text{if } \hat{y} \neq y \text{ (incorrect prediction)} \end{cases}$$

For a dataset with n samples:

$$\text{Average } 0/1 \text{ Loss} = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}(\hat{y}_i \neq y_i)$$



$\mathbb{I}(\hat{y}_i \neq y_i)$: An indicator function that is 1 if $\hat{y}_i \neq y_i$ (misclassification) and 0 otherwise.

$\text{Average } 0/1 \text{ Loss}_{h_1} = 0.0$ (no misclassifications).

$\text{Average } 0/1 \text{ Loss}_{h_2} = 0.5$ (50% of the points are misclassified).

$\text{MSE}_{h_1} = 17.0$

$\text{MSE}_{h_2} = 2.0$

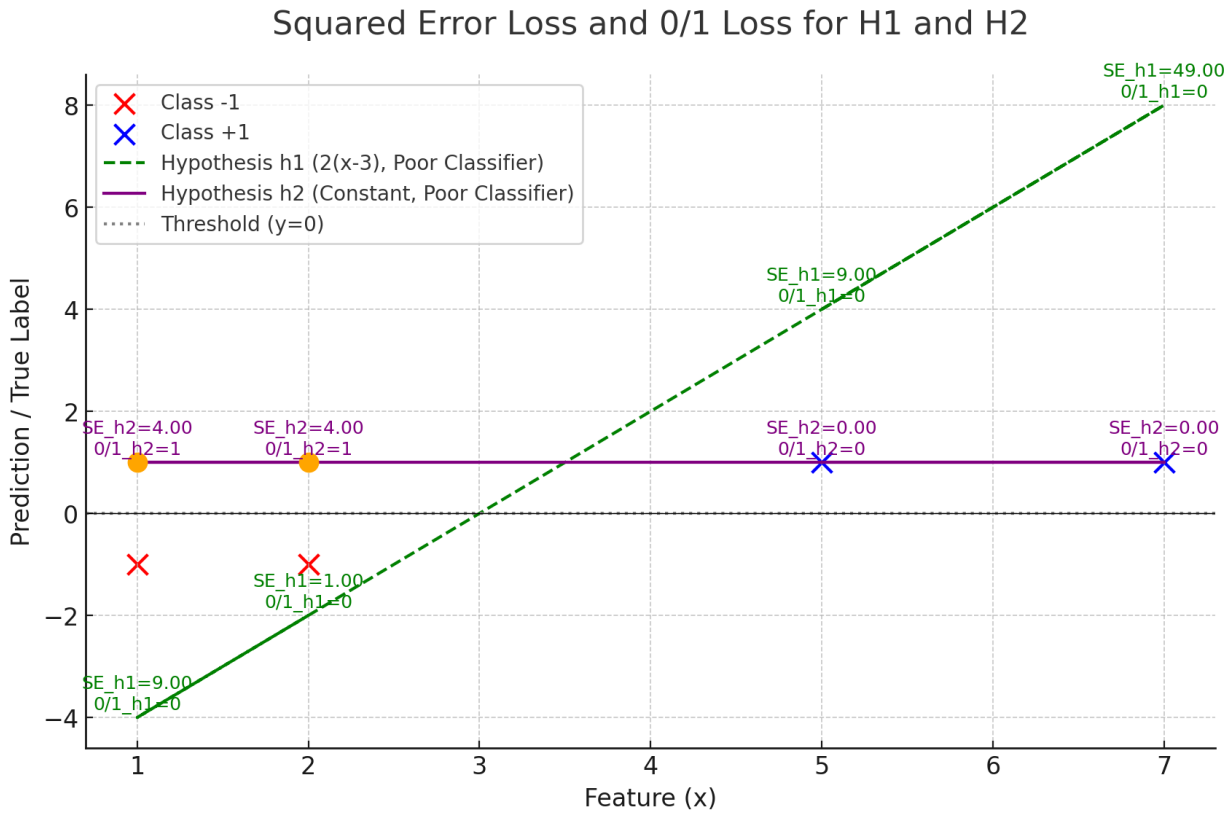
The 0/1 Loss

For a single data point:

$$L(y, \hat{y}) = \begin{cases} 0, & \text{if } \hat{y} = y \text{ (correct prediction)} \\ 1, & \text{if } \hat{y} \neq y \text{ (incorrect prediction)} \end{cases}$$

For a dataset with n samples:

$$\text{Average } 0/1 \text{ Loss} = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}(\hat{y}_i \neq y_i)$$



$\mathbb{I}(\hat{y}_i \neq y_i)$: An indicator function that is 1 if $\hat{y}_i \neq y_i$ (misclassification) and 0 otherwise.

$\text{Average } 0/1 \text{ Loss}_{h_1} = 0.0$ (no misclassifications).

$\text{MSE}_{h_1} = 17.0$

$\text{Average } 0/1 \text{ Loss}_{h_2} = 0.5$ (50% of the points are misclassified).

$\text{MSE}_{h_2} = 2.0$

Limitations

- It It is non-differentiable, making it unsuitable for gradient-based optimization.
- Not sensitive to how confident a model is in its predictions, as it only considers correctness.

Average 0/1 Loss

For a dataset with n samples:

$$\text{Average } 0/1 \text{ Loss} = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}(\hat{y}_i \neq y_i)$$

Where:

- $\hat{y}_i$: The predicted label for the i-th data point.
- y_i: The true label for the i-th data point.
- $\mathbb{I}(\hat{y}_i \neq y_i)$: An indicator function that is 1 if $\hat{y}_i \neq y_i$ (misclassification) and 0 otherwise.

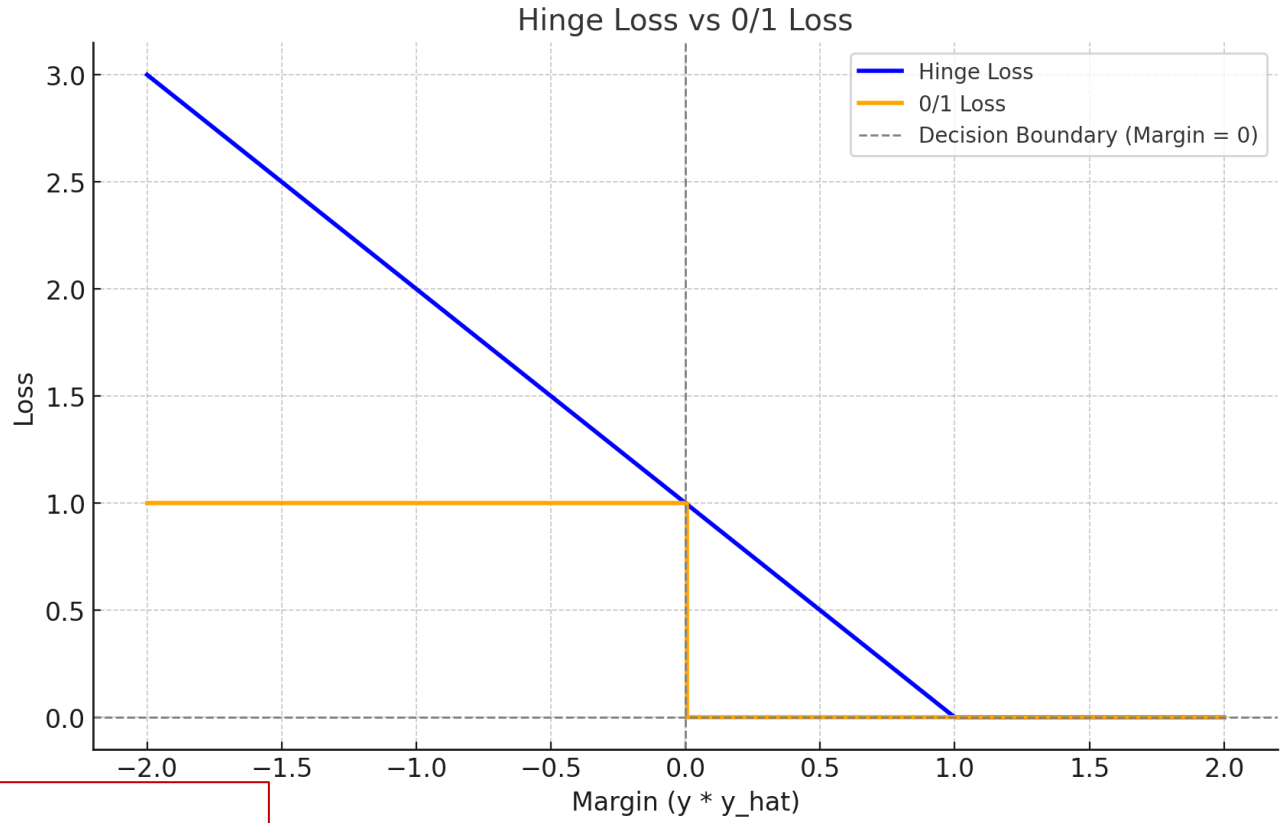
In practice, smoother loss functions (e.g., logistic loss, cross-entropy loss) are often used for training, while 0/1 loss is used for final evaluation.

Hinge Loss

$$L = \max(0, 1 - y \cdot \hat{y})$$

Where:

- y: True label ($+1$ or -1).
- $\hat{y}$: Predicted score/output of the model.



The margin $y \cdot \hat{y}$:

- Measures how aligned the prediction $\hat{y}$ is with the true label y.
- A positive margin ($y \cdot \hat{y} > 0$) indicates a correct prediction, and the larger the margin, the more confident the prediction.
- A negative margin ($y \cdot \hat{y} < 0$) indicates a misclassified data point.

The term $1 - y \cdot \hat{y}$ is used in **hinge loss** specifically:

- If $y \cdot \hat{y} \geq 1$, hinge loss is 0, meaning the margin is sufficient.
- If $y \cdot \hat{y} < 1$, hinge loss penalizes the model with a value $1 - y \cdot \hat{y}$.

Hinge Loss

$$L = \max(0, 1 - y \cdot \hat{y})$$

Where:

- y: True label ($+1$ or -1).
- $\hat{y}$: Predicted score/output of the model.

Punishes Incorrect Predictions:

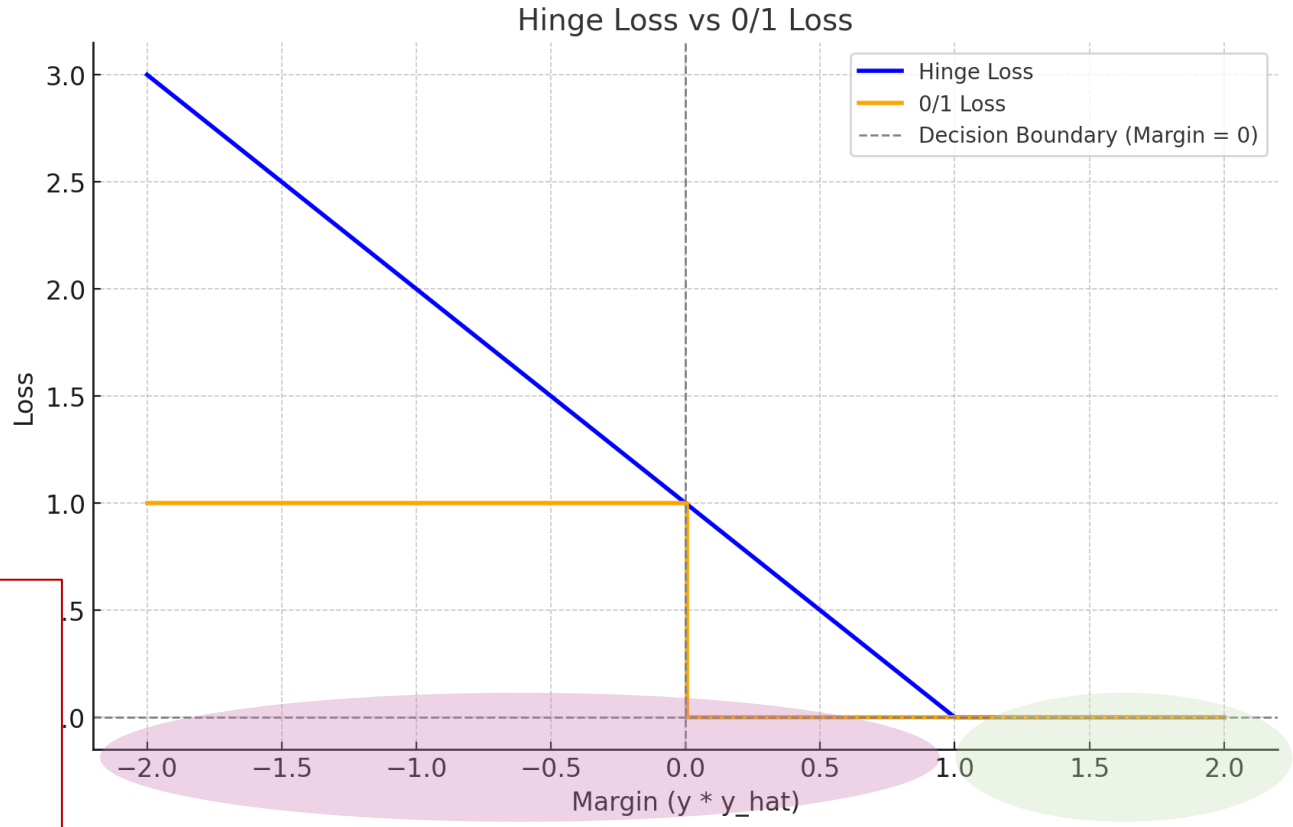
- When $y \cdot \hat{y}(x) < 1$ (low margin or wrong classification):
  - The loss increases as $y \cdot \hat{y}(x)$ moves further from 1.
  - Encourages the model to adjust predictions to increase the margin.

Encourages Confidence in Correct Predictions:

- When $y \cdot \hat{y}(x) \geq 1$ (confident correct classification):
  - The loss is 0, meaning confident, correct predictions are not penalized.
  - Encourages the model to make confident, correct predictions.

Linear Growth for Incorrect Predictions:

- For $y \cdot \hat{y}(x)$ far from 1, the hinge loss grows linearly, penalizing the prediction proportionally to its error.

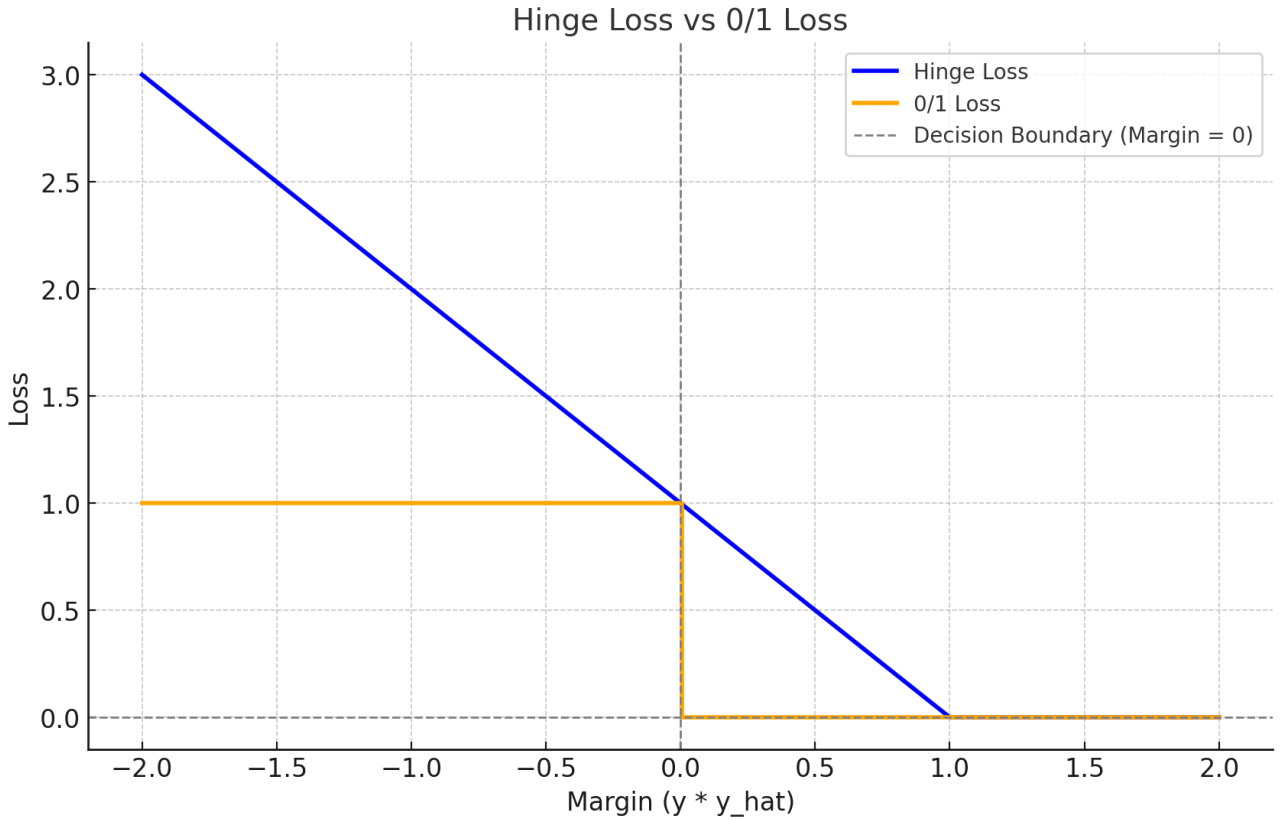


Hinge Loss

$$L = \max(0, 1 - y \cdot \hat{y})$$

Where:

- y: True label ($+1$ or -1).
- $\hat{y}$: Predicted score/output of the model.



- **Hinge Loss** (blue curve):
  - Continuous and linear for margins less than 1.
  - Encourages predictions to be on the correct side of the decision boundary with a margin ≥1\geq 1≥1.
- **0/1 Loss** (orange step):
  - Binary, assigning a loss of 1 for incorrect predictions (margin <0) and 0 for correct predictions (margin ≥0).

Even though it's **non-differentiable** at y·y^=1,
it is **sub-differentiable**, convex and piecewise linear, enabling the use of algorithms like stochastic gradient descent (SGD).

Numerical Example

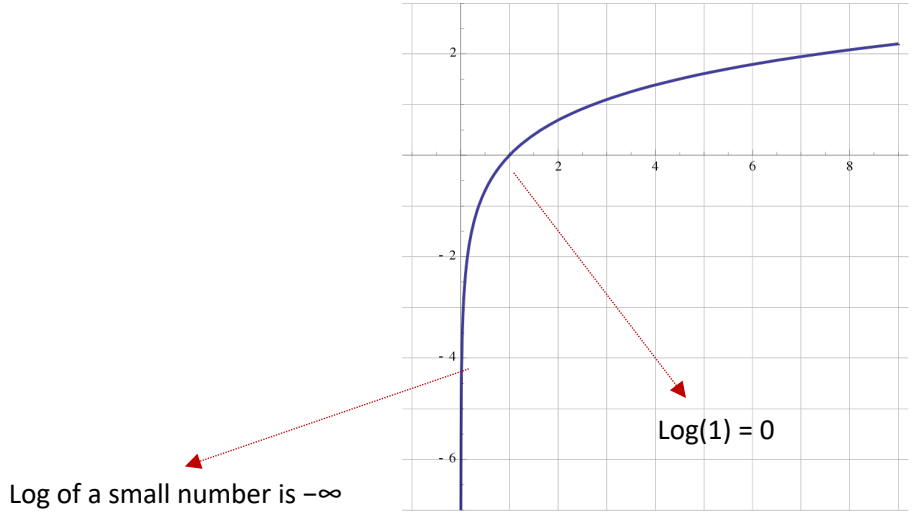
Example Data:

y	$\hat{y}$	Margin ($y \cdot \hat{y}$)	0/1 Loss $L_{0/1}$	Hinge Loss L_{hinge}
$+1$	1.5	$+1.5$	0	$\max(0, 1 - 1.5) = 0$
$+1$	0.8	$+0.8$	0	$\max(0, 1 - 0.8) = 0.2$
-1	-1.2	$+1.2$	0	$\max(0, 1 - 1.2) = 0$
-1	0.5	-0.5	1	$\max(0, 1 - (-0.5)) = 1.5$

Observations:

- For $y = +1$ and $\hat{y} = 1.5$: Both losses are 0, as the prediction is correct and the margin is sufficient.
- For $y = +1$ and $\hat{y} = 0.8$: 0/1 Loss is 0, but Hinge Loss is 0.2, as the margin is less than 1.
- For $y = -1$ and $\hat{y} = -1.2$: Both losses are 0, as the prediction is correct with sufficient margin.
- For $y = -1$ and $\hat{y} = 0.5$: 0/1 Loss is 1 (wrong prediction), and Hinge Loss is 1.5 (penalizes the incorrect prediction heavily).

Logistic Loss



Logistic Loss

- Used in binary classification tasks.
- Based on the **logistic regression model**, which predicts probabilities for binary outcomes.

Definition:

For a single data point (x, y), where $y \in \{0, 1\}$ and $\hat{y}$ is the predicted probability for $y = 1$:

$$L(y, \hat{y}) = -\left[y \cdot \log(\hat{y}) + (1 - y) \cdot \log(1 - \hat{y})\right]$$

$$y_i \log(p_i)$$

Key Points:

- Penalizes predictions that are confident but incorrect.
- Encourages the model to output probabilities close to the true labels (y).
- The loss is low when $\hat{y}$ is close to y (e.g., $\hat{y} = 0.9$ for $y = 1$).

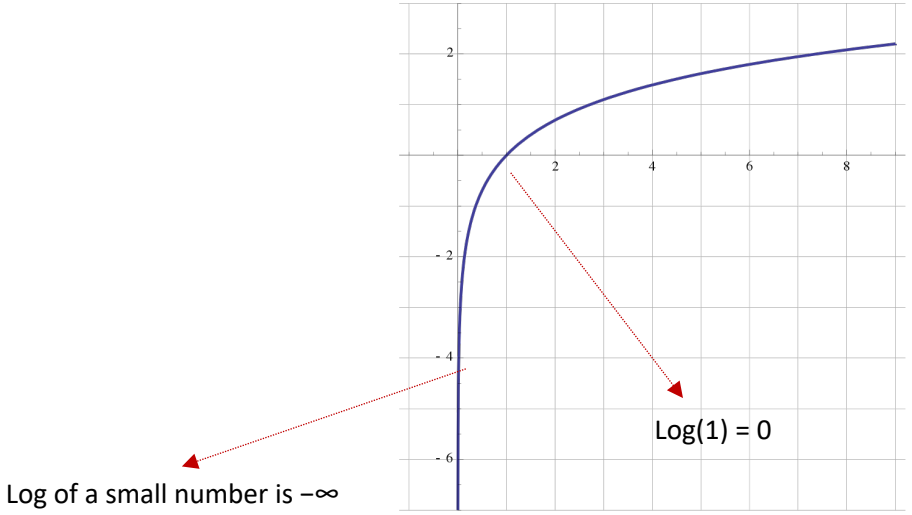
Scenario:

- True label (y): 1 (indicating class 1)
- Predicted probability for class 1 ($\hat{y}$): 0.9

$$\text{Loss} = -\left[1 \cdot \log(0.9) + (1 - 1) \cdot \log(1 - 0.9)\right]$$

$$\text{Loss} = -\log(0.9)$$

Logistic Loss



Logistic Loss

- Used in binary classification tasks.
- Based on the **logistic regression model**, which predicts probabilities for binary outcomes.

Definition:

For a single data point (x, y), where $y \in \{0, 1\}$ and $\hat{y}$ is the predicted probability for $y = 1$:

$$L(y, \hat{y}) = -\big[y \cdot \log(\hat{y}) + (1 - y) \cdot \log(1 - \hat{y})\big]$$

$$y_i \log(p_i)$$

Key Points:

- Penalizes predictions that are confident but incorrect.
- Encourages the model to output probabilities close to the true labels (y).
- The loss is low when $\hat{y}$ is close to y (e.g., $\hat{y} = 0.9$ for $y = 1$).

Note that **Cross-Entropy Loss** A generalization of logistic loss for multi-class classification problems.

Cross Entropy Loss

- **Cross Entropy Loss**:
  - Suppose there are C classes, and the model outputs a probability distribution for a given input:

$$[\hat{y}_1, \hat{y}_1, \ldots, \hat{y}_c]$$

  - True labels: $y_i = 1$ for the correct class and 0 for all other classes.

$$[y_1, y_2, \ldots, y_C]$$

Cross Entropy Loss

- **Cross Entropy Loss**:
  - Suppose there are C classes, and the model outputs a probability distribution for a given input:

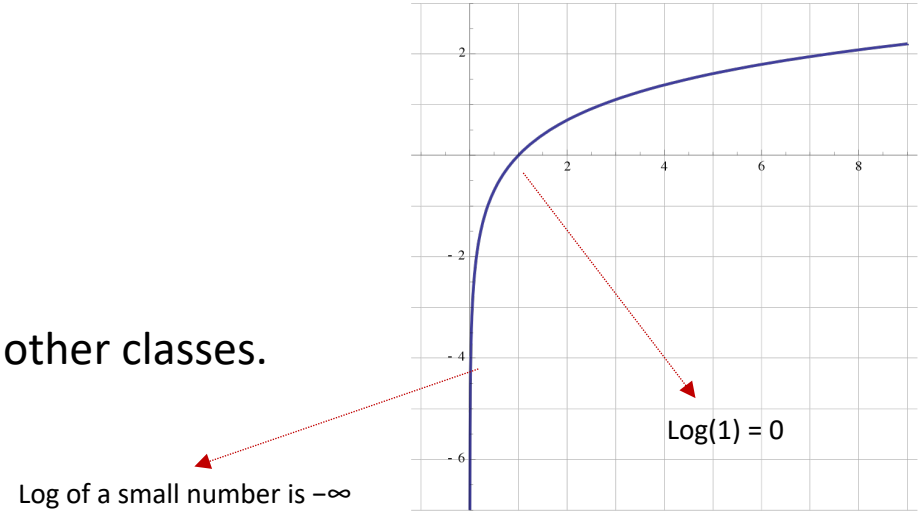
$$[\hat{y}_1, \hat{y}_1, \ldots, \hat{y}_c]$$

  - True labels: $y_i = 1$ for the correct class and 0 for all other classes.

$$[y_1, y_2, \ldots, y_C]$$

  - Cross-Entropy Loss for a Single Example:

$$LOSS = -\sum_{i=1}^{C} y_i \log(\hat{y}_i)$$



Cross Entropy Loss

- **Cross Entropy Loss**:
  - Suppose there are C classes, and the model outputs a probability distribution for a given input:

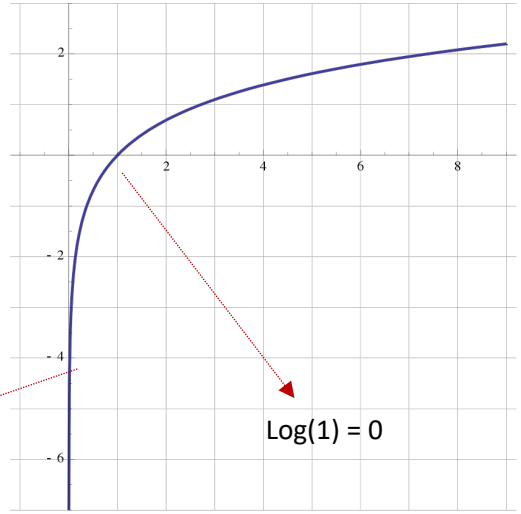
$$[\hat{y}_1, \hat{y}_1, \dots, \hat{y}_c]$$

  - True labels: $y_i = 1$ for the correct class and 0 for all other classes.

$$[y_1, y_2, \dots, y_C]$$

  - Cross-Entropy Loss for a Single Example:

$$LOSS = -\sum_{i=1}^{C} y_i \log(\hat{y}_i)$$



$$y_i \log(p_i)$$

- **High** Confidence, **Wrong** Prediction:
  - For correct y_i, $\log(\hat{y}_i)$ gets close to $-\infty$
  - High contribution to loss
- Encouraging **Correct High Confidence**:
  - For correct y_i, $\log(\hat{y}_i)$ gets close to 0
  - Low contribution to loss

Scenario:

- True label (y): Class 2 $\quad y_{\text{true}} = [0, 1, 0]$
- Predicted probabilities ($\hat{y}$): [0.1, 0.7, 0.2] (for Classes 1, 2, 3)

$$\text{Loss} = -\log(\hat{y}_{\text{true}}) \qquad \text{Loss} = -(-0.3567) = 0.3567$$

$(\hat{y}_{true})$refers to the **predicted probability assigned to the true class** by the model.

Cross Entropy Loss

- **Cross Entropy Loss**:
  - Suppose there are C classes, and the model outputs a probability distribution for a given input:

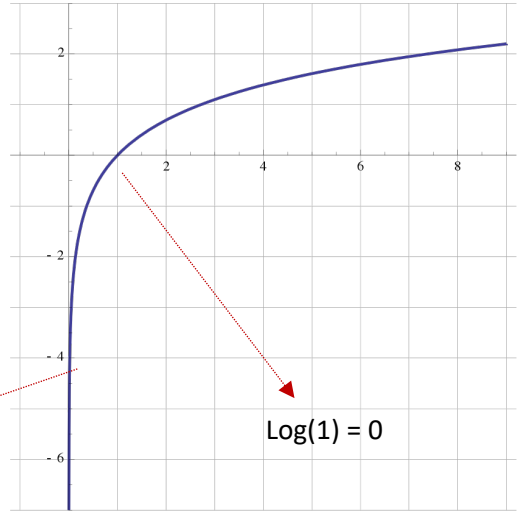
$$[\hat{y}_1, \hat{y}_1, \dots, \hat{y}_c]$$

  - True labels: $y_i = 1$ for the correct class and 0 for all other classes.

$$[y_1, y_2, \dots, y_C]$$

  - Cross-Entropy Loss for a Single Example:

$$LOSS = -\sum_{i=1}^{C} y_i \log(\hat{y}_i)$$



$$y_i \log(p_i)$$

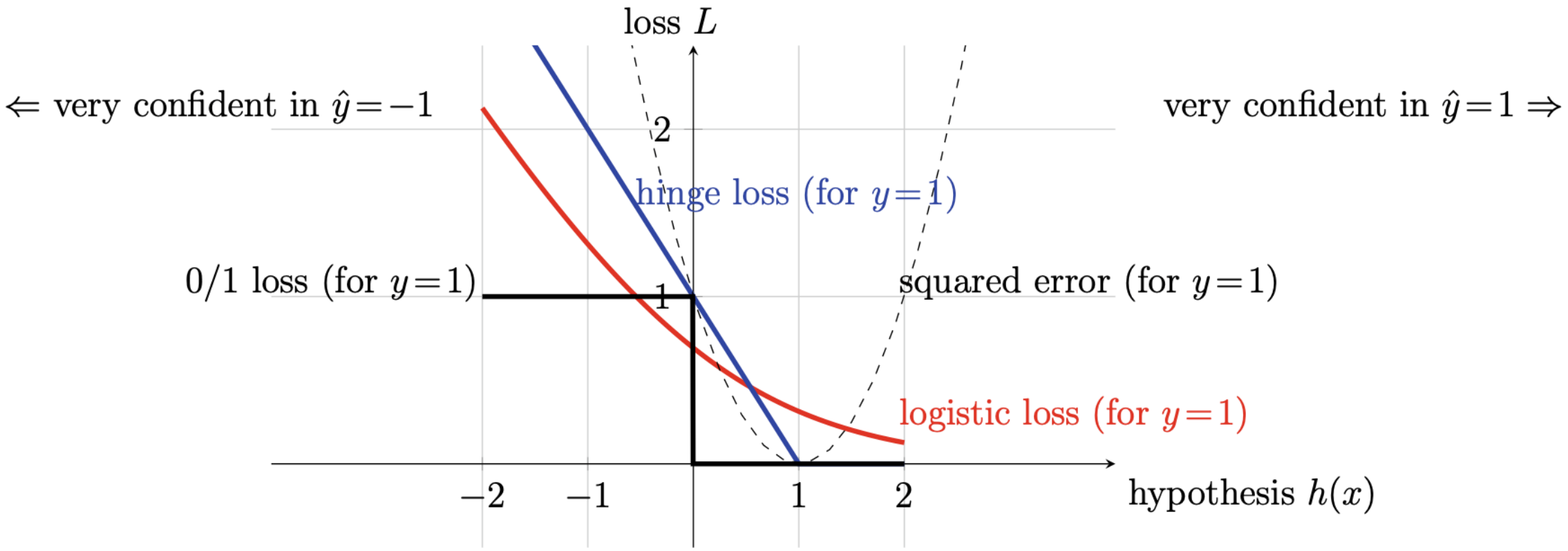
Scenario:

- True class: **Class 2** $y_{\text{true}} = [0, 1, 0]$
- Predicted probabilities ($\hat{y}$): [0.9, 0.05, 0.05] (high confidence in Class 1, which is incorrect).

$$\text{Loss} = -\log(\hat{y}[2]) \quad \text{Loss} = -\log(0.05) = 2.9957$$

- **High** Confidence, **Wrong** Prediction:
  - For correct y_i, $\log(\hat{y}_i)$ gets close to $-\infty$
  - High contribution to loss
- Encouraging **Correct High Confidence**:
  - For correct y_i, $\log(\hat{y}_i)$ gets close to 0
  - Low contribution to loss

Comparison of Loss Functions



Source: Machine Learning: The Basics (Machine Learning: Foundations, Methodologies, and Applications) , A. Jung

The SE loss does not stay at 0 as margin grows.

Expected Loss

Definition

For a hypothesis h, the expected loss is:

$$\mathbb{E}[L] = \mathbb{E}_{(x,y)\sim P(x,y)}[L((x,y),h(x))]$$

Where:

- (x,y): A data point (x: features, y: true label).
- $P(x,y)$: The joint probability distribution of x and y.
- $L((x,y),h(x))$: The loss incurred when the classifier predicts $h(x)$ for x, and the true label is y.

Example with 0/1 Loss:

For the **0/1 loss function** (1 for incorrect predictions, 0 for correct predictions), the expected loss is the **probability of misclassification**:

$$\mathbb{E}[L] = P(y \neq h(x))$$

Example of Expected Loss

- Suppose the problem involves classifying an email as "spam" (y=1) or "not spam" (y=0).

Given:

- For a specific email (x):
  - $P(y = 1 \mid x) = 0.7$ (70% chance it's spam).
  - $P(y = 0 \mid x) = 0.3$ (30% chance it's not spam).
- Classifier $h(x)$:
  - Predicts "spam" ($h(x) = 1$).

0/1 Loss Function:

$$L((x, y), h(x)) = \begin{cases} 0, & \text{if } h(x) = y \\ 1, & \text{if } h(x) \neq y \end{cases}$$

Expected Loss Formula:

$$\mathbb{E}[L] = \sum_{c \in \{0,1\}} P(y = c \mid x) \cdot L((x, c), h(x))$$

Case 1: $y = 1$ **(True label is spam):**

- Classifier predicts $h(x) = 1$, so $L = 0$.
- Contribution to expected loss:

$$P(y = 1 \mid x) \cdot L = 0.7 \cdot 0 = 0$$

Case 2: $y = 0$ **(True label is not spam):**

- Classifier predicts $h(x) = 1$, so $L = 1$.
- Contribution to expected loss:

$$P(y = 0 \mid x) \cdot L = 0.3 \cdot 1 = 0.3$$

Total Expected Loss:

$$\mathbb{E}[L] = 0 + 0.3 = 0.3$$

This means there is a 30% chance of misclassifying the email using this classifier.

Limitations of Expected Loss

Definition

For a hypothesis h, the expected loss is:

$$\mathbb{E}[L] = \mathbb{E}_{(x,y)\sim P(x,y)}[L((x,y),h(x))]$$

Where:

- (x,y): A data point (x: features, y: true label).
- $P(x,y)$: The joint probability distribution of x and y.
- $L((x,y),h(x))$: The loss incurred when the classifier predicts $h(x)$ for x, and the true label is y.

- Computing expected loss requires knowledge of the **true probability distribution** $P(x,y)$, which is rarely available in practice.
- In real-world applications, we often approximate it using empirical data.

Average Loss

For a dataset with N examples $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$, and a hypothesis (model) $h(x)$, the average loss is defined as:

$$\text{Average Loss} = \frac{1}{N} \sum_{i=1}^{N} L((x_i, y_i), h(x_i))$$

Where:

- $L((x_i, y_i), h(x_i))$: The loss incurred by the model $h(x)$ on the i-th data point.
- N: Total number of data points in the dataset.

Average Loss

For a dataset with N examples $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$, and a hypothesis (model) $h(x)$, the average loss is defined as:

$$\text{Average Loss} = \frac{1}{N} \sum_{i=1}^{N} L((x_i, y_i), h(x_i))$$

Where:

- $L((x_i, y_i), h(x_i))$: The loss incurred by the model $h(x)$ on the i-th data point.
- N: Total number of data points in the dataset.

- It approximates the expected loss (theoretical loss over the entire data distribution) using the available dataset.
- Helps assess how well a model generalizes to the dataset.
- Many machine learning algorithms minimize the average loss during training (e.g., gradient descent).

Average Loss

For a dataset with N examples $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$, and a hypothesis (model) $h(x)$, the average loss is defined as:

$$\text{Average Loss} = \frac{1}{N} \sum_{i=1}^{N} L((x_i, y_i), h(x_i))$$

Where:

- $L((x_i, y_i), h(x_i))$: The loss incurred by the model $h(x)$ on the i-th data point.
- N: Total number of data points in the dataset.

- It approximates the expected loss (theoretical loss over the entire data distribution) using the available dataset.
- Helps assess how well a model generalizes to the dataset.
- Many machine learning algorithms minimize the average loss during training (e.g., gradient descent).

Aspect	Average Loss	Expected Loss
Definition	Computed on a finite dataset.	Computed over the entire data distribution $P(x, y)$.
Nature	Empirical and practical measure.	Theoretical and often unobservable.

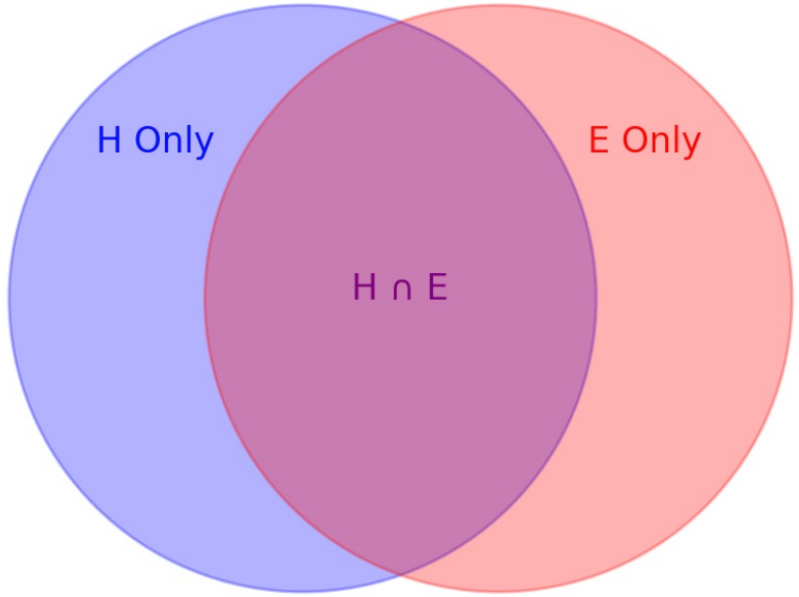
Quick Review of Probability

Conditional Probability

- The conditional probability of H given E is:

$$P(H \mid E) = \frac{P(H \cap E)}{P(E)}$$

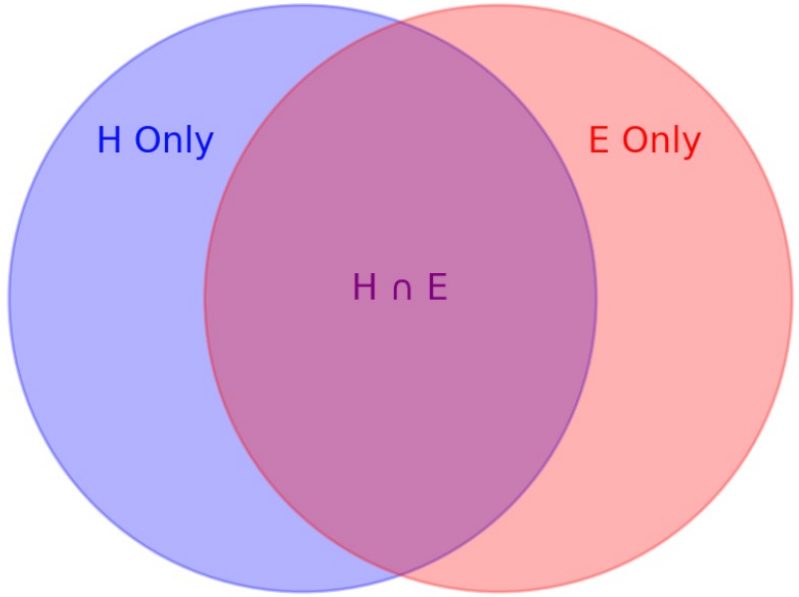
- $P(H \mid E)$: Probability of H occurring given E has occurred.
- $P(H \cap E)$: Probability of both H and E occurring.
- $P(E)$: Probability of E.



Conditional Probability

- The conditional probability of H given E is:

$$P(H \mid E) = \frac{P(H \cap E)}{P(E)}$$



- $P(H \mid E)$: Probability of H occurring given E has occurred.
- $P(H \cap E)$: Probability of both H and E occurring.
- $P(E)$: Probability of E.

Example:

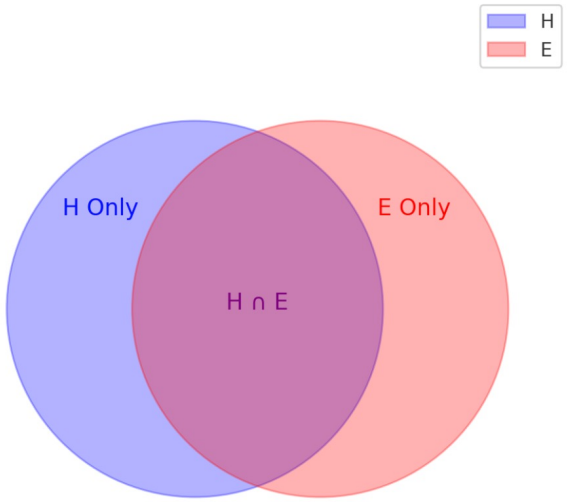
- Suppose $P(H) = 0.4$, $P(E) = 0.6$, and $P(H \cap E) = 0.2$.
- Using the formula:

$$P(H \mid E) = \frac{P(H \cap E)}{P(E)} = \frac{0.2}{0.6} = 0.333$$

- Interpretation: Given E, there is a 33.3% chance that H occurs.

Bayes Theorem

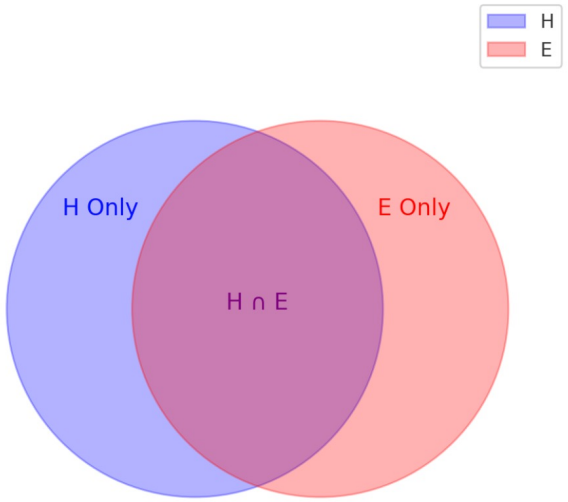
$$P(E \mid H) = \frac{P(H \cap E)}{P(H)}$$



$$P(H \mid E) = \frac{P(H \cap E)}{P(E)}$$

Bayes Theorem

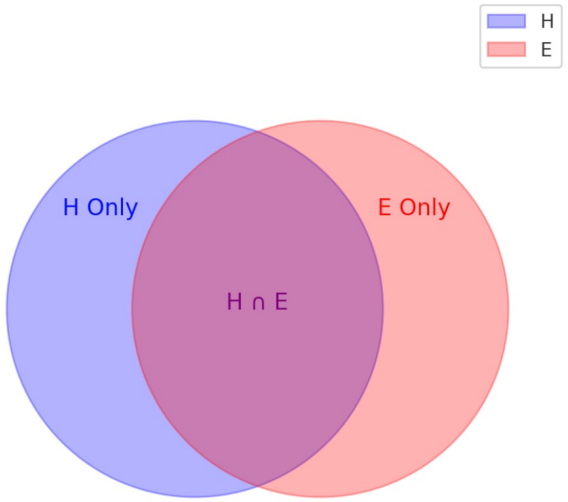
$$P(E \mid H) = \frac{P(H \cap E)}{P(H)}$$



$$P(H \mid E) = \frac{P(H \cap E)}{P(E)}$$

Bayes Theorem

$$P(E \mid H) = \frac{P(H \cap E)}{P(H)}$$

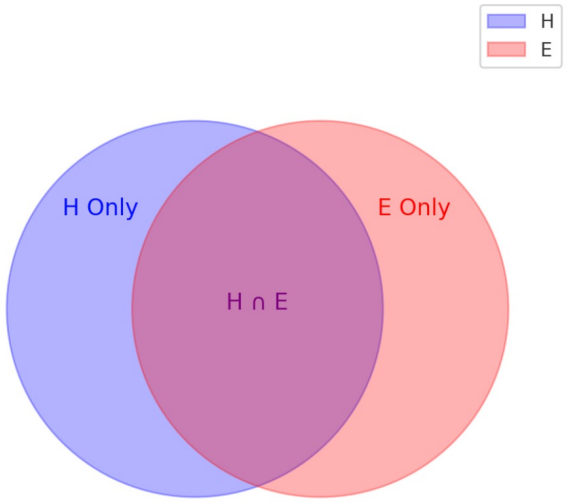


$$P(H \mid E) = \frac{P(H \cap E)}{P(E)}$$

$$P(E \cap H) = P(E \mid H) \cdot P(H) = P(H \mid E) \cdot P(E)$$

Bayes Theorem

$$P(E \mid H) = \frac{P(H \cap E)}{P(H)}$$



$$P(H \mid E) = \frac{P(H \cap E)}{P(E)}$$

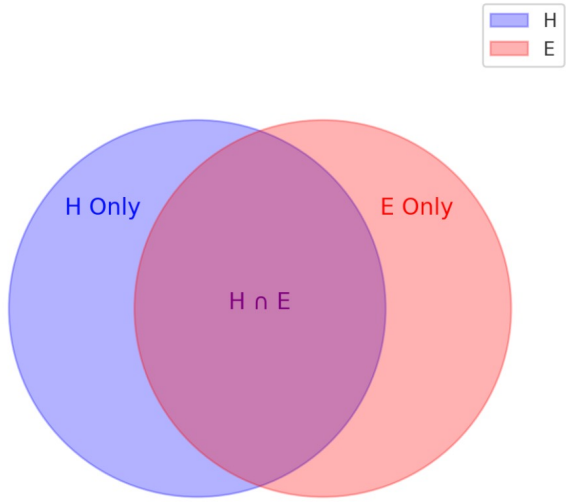
$$P(E \cap H) = P(E \mid H) \cdot P(H) = P(H \mid E) \cdot P(E)$$

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

- It's especially useful in cases where $P(H \cap E)$ is not directly available, but $P(E \mid H)$, $P(H)$, and $P(E)$ are **known or can be estimated**.
- It's a way of flipping E, H: If calculating $P(E \mid H)$ is easy, Bayes theorem gives us the flipped probability: $P(H \mid E)$

Bayes Theorem, Prior, and Posterior Probabilities

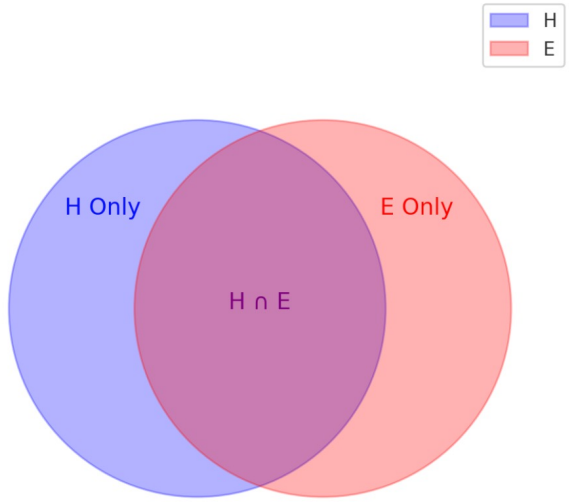
$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$



Bayes Theorem, Prior, and Posterior Probabilities

Likelihood (probability of observing E if H is true).

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

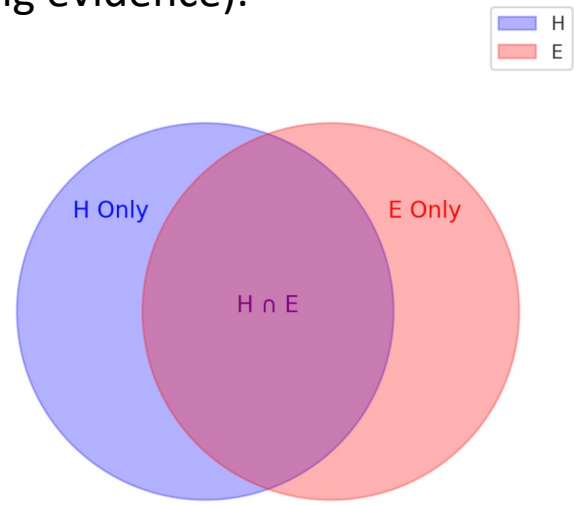


Bayes Theorem, Prior, and Posterior Probabilities

Likelihood (probability of observing E if H is true).

Prior probability (belief about H before seeing evidence).

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$



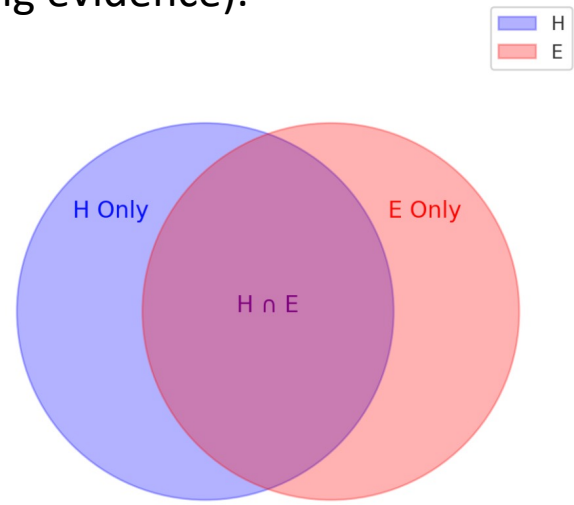
Bayes Theorem, Prior, and Posterior Probabilities

Likelihood (probability of observing E if H is true).

Prior probability (belief about H before seeing evidence).

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

Evidence probability



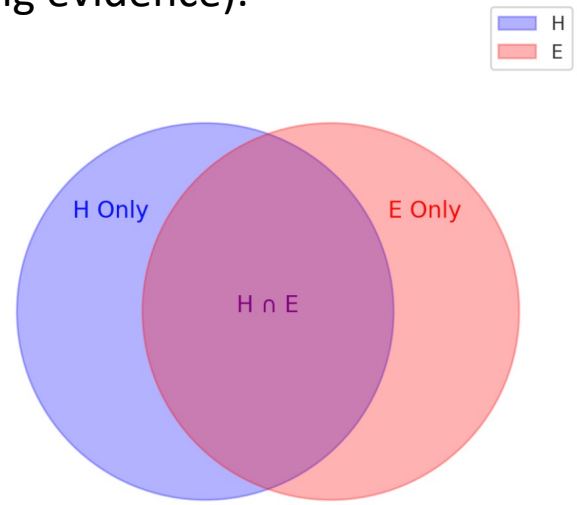
Bayes Theorem, Prior, and Posterior Probabilities

Likelihood (probability of observing E if H is true).

Prior probability (belief about H before seeing evidence).

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

Evidence probability



- $P(H)$: **Prior probability** (belief about H before seeing evidence).
- $P(E \mid H)$: **Likelihood** (probability of observing E if H is true).
- $P(E)$: **Evidence probability** (total probability of E occurring).

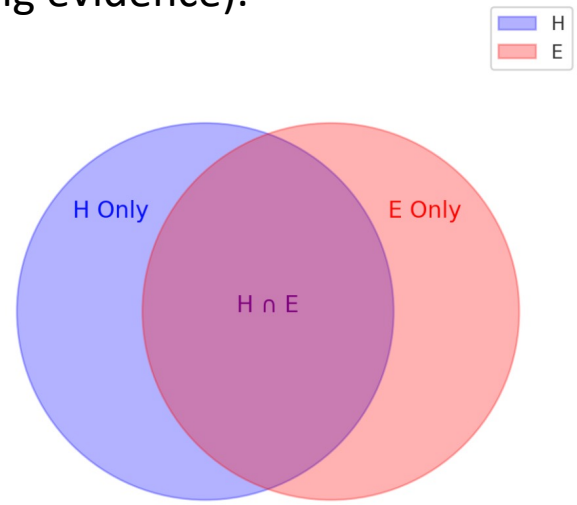
Bayes Theorem, Prior, and Posterior Probabilities

Likelihood (probability of observing E if H is true).

Prior probability (belief about H before seeing evidence).

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

Evidence probability



- $P(H)$: **Prior probability** (belief about H before seeing evidence).
- $P(E \mid H)$: **Likelihood** (probability of observing E if H is true).
- $P(E)$: **Evidence probability** (total probability of E occurring).

If P(E) is not directly known, we often calculate it using H^C, the complement of H:

$$P(E) = P(E \mid H) \cdot P(H) + P(E \mid H^c) \cdot P(H^c)$$

Example

Likelihood (probability of observing E if H is true).

Prior probability (belief about H before seeing evidence).

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

Evidence probability

Example

Likelihood (probability of observing E if H is true).

Prior probability (belief about H before seeing evidence).

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

Evidence probability

- H: A patient has a disease.
- E: A medical test returns positive.
- Known:
  - $P(H) = 0.01$ (1% of the population has the disease).
  - $P(E \mid H) = 0.95$ (test correctly identifies disease 95% of the time).
  - $P(E) = 0.05$ (5% of tests return positive overall).

Example

Likelihood (probability of observing E if H is true).

Prior probability (belief about H before seeing evidence).

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

Evidence probability

- H: A patient has a disease.
- E: A medical test returns positive.
- Known:
  - $P(H) = 0.01$ (1% of the population has the disease).
  - $P(E \mid H) = 0.95$ (test correctly identifies disease 95% of the time).
  - $P(E) = 0.05$ (5% of tests return positive overall).

Question: What is the probability of disease if the test is positive? i.e., calculate $P(H|E)$.

Example

Likelihood (probability of observing E if H is true).

Prior probability (belief about H before seeing evidence).

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

Evidence probability

- H: A patient has a disease.
- E: A medical test returns positive.
- Known:
  - $P(H) = 0.01$ (1% of the population has the disease).
  - $P(E \mid H) = 0.95$ (test correctly identifies disease 95% of the time).
  - $P(E) = 0.05$ (5% of tests return positive overall).

Question: What is the probability of disease if the test is positive? i.e., calculate $P(H|E)$.

Using Bayes' Theorem:

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)} = \frac{0.95 \cdot 0.01}{0.05} = 0.19$$

Example

Likelihood (probability of observing E if H is true).

Prior probability (belief about H before seeing evidence).

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

Evidence probability

- H: A patient has a disease.
- E: A medical test returns positive.
- Known:
  - $P(H) = 0.01$ (1% of the population has the disease).
  - $P(E \mid H) = 0.95$ (test correctly identifies disease 95% of the time).
  - $P(E) = 0.05$ (5% of tests return positive overall).

Question: What is the probability of disease if the test is positive? i.e., calculate $P(H|E)$.

Using Bayes' Theorem:

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)} = \frac{0.95 \cdot 0.01}{0.05} = 0.19$$

Interpretation: Even with a positive test, there's only a 19% chance the patient has the disease due to the low prior probability.
This shows how Bayes' Theorem helps **update probabilities considering evidence**.

Example when $P(E)$ is missing

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

- H: A patient has a disease.
- E: A medical test returns positive.

Known:

- $P(H) = 0.01$ (1% of the population has the disease).
- $P(E \mid H) = 0.95$ (test correctly identifies the disease 95% of the time). → True positive rate
- $P(E \mid \neg H) = 0.05$ (5% of tests are positive even without the disease). → False positive rate

Example when $P(E)$ is missing

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

- H: A patient has a disease.
- E: A medical test returns positive.

Known:

- $P(H) = 0.01$ (1% of the population has the disease).
- $P(E \mid H) = 0.95$ (test correctly identifies the disease 95% of the time). → True positive rate
- $P(E \mid \neg H) = 0.05$ (5% of tests are positive even without the disease). → False positive rate

$$P(E) = P(E \mid H) \cdot P(H) + P(E \mid \neg H) \cdot P(\neg H)$$

Example when $P(E)$ is missing

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

- H: A patient has a disease.
- E: A medical test returns positive.

Known:

- $P(H) = 0.01$ (1% of the population has the disease).
- $P(E \mid H) = 0.95$ (test correctly identifies the disease 95% of the time). → True positive rate
- $P(E \mid \neg H) = 0.05$ (5% of tests are positive even without the disease). → False positive rate

$$P(E) = P(E \mid H) \cdot P(H) + P(E \mid \neg H) \cdot P(\neg H)$$

$$P(E) = (0.95 \cdot 0.01) + (0.05 \cdot 0.99)$$

$$P(E) = 0.0095 + 0.0495 = 0.059$$

Example when $P(E)$ is missing

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

- H: A patient has a disease.
- E: A medical test returns positive.

Known:

- $P(H) = 0.01$ (1% of the population has the disease).
- $P(E \mid H) = 0.95$ (test correctly identifies the disease 95% of the time). → True positive rate
- $P(E \mid \neg H) = 0.05$ (5% of tests are positive even without the disease). → False positive rate

$$P(E) = P(E \mid H) \cdot P(H) + P(E \mid \neg H) \cdot P(\neg H)$$

$$P(E) = (0.95 \cdot 0.01) + (0.05 \cdot 0.99)$$

$$P(E) = 0.0095 + 0.0495 = 0.059$$

$$P(H \mid E) = \frac{0.95 \cdot 0.01}{0.059} \approx 0.161$$

Bayes Classifier

Suppose a data point (i.e. feature vector) can be in either class $H1, H2$, ...

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

Objective:

- Assign a data point to the most probable class H^* based on its features E.

Decision Rule:

- Select the class H that maximizes the posterior probability:

$$H^* = \arg\max_H P(H \mid E)$$

Use this formula for each class

Bayes Classifier

- You want to classify an email as **Spam** $(\boldsymbol{H_1})$ or **Not Spam (H2)** based on whether it contains the word "win" (E).
- Given:

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

Prior probabilities:

- $P(H_1) = 0.3$ (30% of emails are spam).
- $P(H_2) = 0.7$ (70% of emails are not spam).

Likelihoods:

- $P(E \mid H_1) = 0.8$ (80% of spam emails contain "win").
- $P(E \mid H_2) = 0.1$ (10% of non-spam emails contain "win").

Bayes Classifier

- You want to classify an email as **Spam** ($\boldsymbol{H_1}$) or **Not Spam (H2)** based on whether it contains the word "win" (E).
- Given:

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

Prior probabilities:

- $P(H_1) = 0.3$ (30% of emails are spam).
- $P(H_2) = 0.7$ (70% of emails are not spam).

Likelihoods:

- $P(E \mid H_1) = 0.8$ (80% of spam emails contain "win").
- $P(E \mid H_2) = 0.1$ (10% of non-spam emails contain "win").

Evidence:

- Total probability of the word "win" appearing in any email, $P(E)$:

$$P(E) = P(E \mid H_1) \cdot P(H_1) + P(E \mid H_2) \cdot P(H_2)$$

Substituting:

$$P(E) = (0.8 \cdot 0.3) + (0.1 \cdot 0.7) = 0.24 + 0.07 = 0.31$$

Bayes Classifier

Prior probabilities:

- $P(H_1) = 0.3$ (30% of emails are spam).
- $P(H_2) = 0.7$ (70% of emails are not spam).

Likelihoods:

- $P(E \mid H_1) = 0.8$ (80% of spam emails contain "win").
- $P(E \mid H_2) = 0.1$ (10% of non-spam emails contain "win").

Evidence:

- Total probability of the word "win" appearing in any email, $P(E)$:

$$P(E) = P(E \mid H_1) \cdot P(H_1) + P(E \mid H_2) \cdot P(H_2)$$

Substituting:

$$P(E) = (0.8 \cdot 0.3) + (0.1 \cdot 0.7) = 0.24 + 0.07 = 0.31$$

$$P(H_1 \mid E) = \frac{P(E \mid H_1) \cdot P(H_1)}{P(E)}$$

$$P(H_1 \mid E) = \frac{0.8 \cdot 0.3}{0.31} = \frac{0.24}{0.31} \approx 0.774$$

$$P(H_2 \mid E) = \frac{P(E \mid H_2) \cdot P(H_2)}{P(E)}$$

$$P(H_2 \mid E) = \frac{0.1 \cdot 0.7}{0.31} = \frac{0.07}{0.31} \approx 0.226$$

Since $P(H1 \mid E) \approx 0.774 > P(H2 \mid E) \approx 0.226$, classify the email as **Spam ($H1$)**.

Bayes Classifier

Suppose a data point (i.e. feature vector) can be in either class $H1, H2, \ldots$

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

Objective:

- Assign a data point to the most probable class H^* based on its features E.

Decision Rule:

- Select the class H that maximizes the posterior probability:

$$H^* = \arg\max_H P(H \mid E)$$

Use this formula for each class

- **Infeasibility**: In most real-world scenarios, $P(E \mid H)$ and $P(H)$ are unknown or difficult to compute
- **Complexity**: Requires estimating the full joint probability distribution of features for each class.

Bayes Classifier

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

Scenario:

We want to classify an email as either **Spam** or **Not Spam** based on the presence of certain words.

Features:

- X_1: Whether the word "cheap" appears.
- X_2: Whether the word "offer" appears.
- X_3: Whether the word "win" appears.

Classes:

- Y: Email is either Spam ($Y = \text{Spam}$) or Not Spam ($Y = \text{Not Spam}$).

Bayes Classifier:

The Bayes Classifier calculates:

$$P(Y \mid X_1, X_2, X_3) \propto P(Y) \cdot P(X_1, X_2, X_3 \mid Y)$$

Challenge:

- To compute $P(X_1, X_2, X_3 \mid Y)$, we need the **joint probability distribution** of all features given the class.
- With 3 features, this means estimating probabilities for all combinations of X_1, X_2, X_3 (e.g., $[0, 0, 1]$, $[1, 1, 0]$, etc.).
- If there are n features, this grows exponentially (2^n) and requires a large amount of data to estimate reliably.

Naive Bayes Classifier:

Naive Bayes simplifies this by assuming **conditional independence**:

$$P(X_1, X_2, X_3 \mid Y) \approx P(X_1 \mid Y) \cdot P(X_2 \mid Y) \cdot P(X_3 \mid Y)$$

Advantage:

- We only need to estimate $P(X_i \mid Y)$ for each feature X_i, reducing the number of probabilities to estimate.
- For n features, the number of parameters is proportional to n, making it computationally efficient.

Naive Bayes Classifier

- **Assumption**: Features are conditionally independent given the class.

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

$$P(E \mid H) = \prod_i P(E_i \mid H)$$

$$P(E|H) = P(E_1, E_2, \ldots, E_n|H)$$

where E_i are individual features.

Find

$$H^* = \arg\max_H P(H \mid E)$$

Naive Bayes Classifier

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

$$P(E \mid H) = \prod_i P(E_i \mid H)$$

$$P(E|H) = P(E_1, E_2, \ldots, E_n|H)$$

Problem:

Same scenario as before, but now consider **multiple features** (e.g., whether the email contains "win" and "free").

Evidence ($E = \{E_1, E_2\}$):

1. E_1: The email contains "win".
2. E_2: The email contains "free".

Suppose:

- $P(E_1 \mid H_1) = 0.8, P(E_1 \mid H_2) = 0.1$
- $P(E_2 \mid H_1) = 0.7, P(E_2 \mid H_2) = 0.2$
- $P(H_1) = 0.3, P(H_2) = 0.7$

Compute Posterior for H_1: Using the Naive Bayes assumption:

$$P(H_1 \mid E) \propto P(H_1) \cdot P(E_1 \mid H_1) \cdot P(E_2 \mid H_1)$$

Substituting values:

$$P(H_1 \mid E) \propto 0.3 \cdot 0.8 \cdot 0.7 = 0.168$$

Compute Posterior for H_2: Similarly:

$$P(H_2 \mid E) \propto P(H_2) \cdot P(E_1 \mid H_2) \cdot P(E_2 \mid H_2)$$

Substituting values:

$$P(H_2 \mid E) \propto 0.7 \cdot 0.1 \cdot 0.2 = 0.014$$

Normalize: Compute the probabilities:

$$P(H_1 \mid E) = \frac{0.168}{0.168 + 0.014} \approx 0.923$$

$$P(H_2 \mid E) = \frac{0.014}{0.168 + 0.014} \approx 0.077$$

H_1 (Spam) has the higher posterior probability (0.923).

Naive Bayes Classifier

- **Assumption**: Features are conditionally independent given the class.

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

$$P(E \mid H) = \prod_i P(E_i \mid H)$$

$$P(E|H) = P(E_1, E_2, \ldots, E_n|H)$$

where E_i are individual features.

Find

$$H^* = \arg\max_H P(H \mid E)$$

- The **Naive Bayes Classifier** is called "naive" because it makes a **simplifying assumption**:
  - It assumes that all features are **conditionally independent** given the class label.
  - E.g., the words "win" and "free" are likely correlated. Spam emails often have "win a free prize".
  - This assumption is often unrealistic in real-world scenarios, where features are frequently dependent or correlated.
- But it is fast and efficient!