11-695: AI Engineering
ML Reivews

LTI/SCS

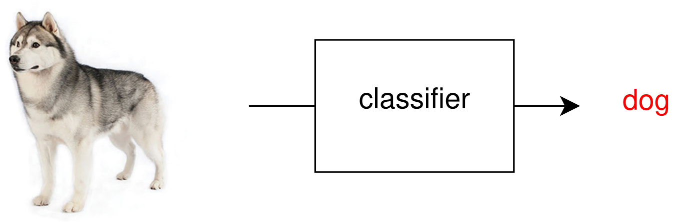
Spring 2020

Outline

1. The Learning Problem
2. Data for Supervised Learning
3. Learning Models
4. Error and Loss Function

Example: Classification

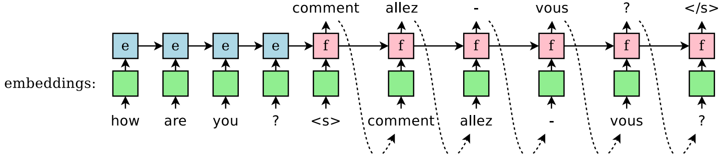
- Find a function $\mathbf{y} = \mathbf{f}(\mathbf{x})$
  - $\mathbf{x}$: an image
  - $\mathbf{y}$: dog, cat, bird, car, etc.



Example: Machine Translation

Carnegie Mellon

- Find a function $\mathbf{y} = \mathbf{f}(\mathbf{x})$
  - $\mathbf{x}$: an English sentence
  - $\mathbf{y}$: an French sentence



How to Find the Function (or Model) f?

Carnegie Mellon

- An optimization problem: find **f** that minimizes the errors towards a defined an objective function
- Learn from data given
- In many cases, with knowledge/prior/bias about the nature of
  - Data: text, videos, images, ...
  - Expert knowledge: NLP, Medicine, ... For example, to guide the models.
  - ...

How to Model f?

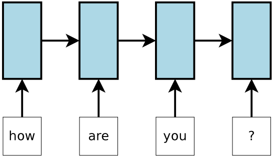
- Parametric:
  - Believe that a fixed param $\mathbf{W} \sim \Omega$ is enough to represent $\mathbf{f}$ in any case
  - The param space Ω is well-defined, *e.g.* $\mathbb{R}^{200\times10}$
  - Choose a learning method, *e.g.* MLE, MAP, ..., to learn $\mathbf{W}$
  - Simple: only estimate params, but we have to inject our bias about data
- Nonparametric:
  - $f \sim \mathbb{F}$, a function space
  - It *has* params, but infinite of them
  - Number of params can change when data change
  - Make no assumption about data, but more complicated because we have to estimate the model, and params of that model
  - More flexible, but a caveat: need to make model assumption

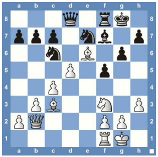
Outline

Carnegie Mellon

Data for Supervised Learning

- Pairs of $(\mathbf{x}, \mathbf{y})$: $\mathbb{D} = \{(\mathbf{x}^{(1)}, \mathbf{y}^{(1)}), (\mathbf{x}^{(2)}, \mathbf{y}^{(2)}), \ldots, (\mathbf{x}^{(n)}, \mathbf{y}^{(n)})\}$
- Each $\mathbf{x}^{(i)}$ is called a *data point*
- Each $\mathbf{y}^{(i)}$ is called a *label*
- $\mathbf{x}^{(i)}$ and $\mathbf{y}^{(i)}$ can be *anything*

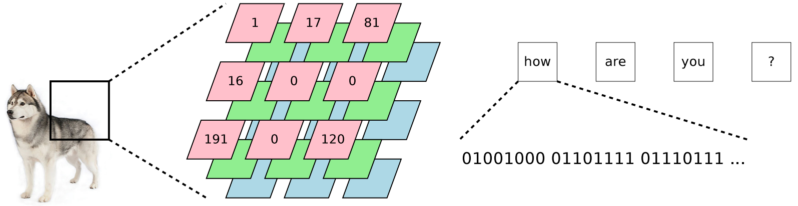




Data Representation

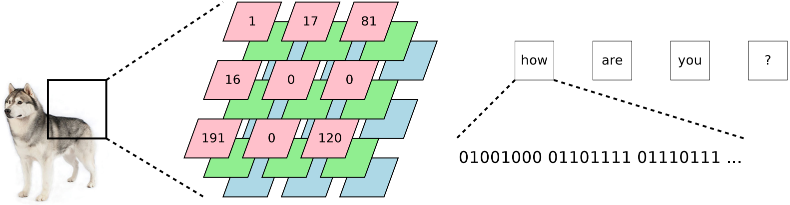
CarnegieMellon

- Computers don't "see" things like we do



Data Representation

- Computers don't "see" things like we do



- ... so it's hard to make them think like we do

Some Types of Labels

- Label $\mathbf{y}$ is in a discrete set $\mathcal{C} = \{1, 2, ..., |\mathcal{C}|\}$
  - *Classification* problems

Some Types of Labels

CarnegieMellon

- Label $\mathbf{y}$ is in a discrete set $\mathcal{C} = \{1, 2, ..., |\mathcal{C}|\}$
  - *Classification* problems
- Label $\mathbf{y}$ is in a "continuous" set, *e.g.* $\mathbf{y} \in [0, 1]$
  - *Regression* problems

Some Types of Labels

Carnegie Mellon

- Label $\mathbf{y}$ is in a discrete set $\mathcal{C} = \{1, 2, ..., |\mathcal{C}|\}$
  - *Classification* problems
- Label $\mathbf{y}$ is in a "continuous" set, *e.g.* $\mathbf{y} \in [0, 1]$
  - *Regression* problems
- Label $\mathbf{y}$ is has some self-dependencies, *e.g.* a sentence in French
  - *Structured prediction* problems

Some Types of Labels

- Label $\mathbf{y}$ is in a discrete set $\mathcal{C} = \{1, 2, ..., |\mathcal{C}|\}$
  - *Classification* problems
- Label $\mathbf{y}$ is in a "continuous" set, *e.g.* $\mathbf{y} \in [0, 1]$
  - *Regression* problems
- Label $\mathbf{y}$ is has some self-dependencies, *e.g.* a sentence in French
  - *Structured prediction* problems
- Why learn these?
  - The types of problems you tackle (loosely) tell you how to design the learning models.

Outline

CarnegieMellon

1. The Learning Problem

2. Data for Supervised Learning

3. Learning Models

4. Error and Loss Function

Parametric Learning Problem

- Data $\mathbb{D} = \{(\mathbf{x}^{(1)}, \mathbf{y}^{(1)}), (\mathbf{x}^{(2)}, \mathbf{y}^{(2)}), \ldots, (\mathbf{x}^{(n)}, \mathbf{y}^{(n)})\}$
- Objective: find parameter $\mathbf{W}$ that best fits data
- Each model has its own definition of "best fits"

Maximum Likelihood Estimation (MLE)

Carnegie Mellon

- Goal: Maximize probability of data given params $P(\mathbb{D} \mid \theta)$, *a.k.a* likelihood of *the params*
- Normally we deal with log of this likelihood

$$\mathcal{L}(\theta) = \log P(\mathbb{D} \mid \theta) = log \prod_{i=1}^{n} P(\mathbf{x}^{(i)} \mid \theta) = \sum_{i=1}^{n} log P(\mathbf{x}^{(i)} \mid \theta)$$

- MLE estimator is:

$$\hat{\theta}_{\mathrm{MLE}} = \underset{\theta}{\operatorname{argmax}}\, \mathcal{L}(\theta) = \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{n} log P(\mathbf{x}^{(i)} \mid \theta)$$

Maximum-A-Posteriori (MAP)

- Goal: Maximize probability of posterior of params given data

$$\underbrace{P(\theta \mid \mathbb{D})}_{\text{posterior}} = \frac{P(\theta) \times P(\mathbb{D} \mid \theta)}{P(\mathbb{D})} \propto \underbrace{P(\theta)}_{\text{prior}} \times \underbrace{P(\mathbb{D} \mid \theta)}_{\text{likelihood}}$$

- Same as MLE, normally we deal with log of this posterior

$$\log P(\theta \mid \mathbb{D}) \propto \log P(\theta) + \log P(\mathbb{D} \mid \theta)$$

- MAP estimator is:

$$\hat{\theta}_{\text{MAP}} = \underset{\theta}{\operatorname{argmax}} \log P(\theta \mid \mathbb{D}) = \underset{\theta}{\operatorname{argmax}} \Big(\log P(\theta) + \underbrace{\log P(\mathbb{D} \mid \theta)}_{\text{loglikelihood}} \Big)$$

A Classic Linear Regression Example

Carnegie Mellon

- Linear Regression model: $\mathbf{y} = \mathbf{f}(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + \epsilon$ where $\epsilon \sim \mathbf{N}(\mathbf{0}, \sigma^2)$
- Each $i.i.d$ sample:

$$\begin{aligned} P(y^{(i)} \mid x^{(i)}, \mathbf{w}) &= \mathbf{N}(y^{(i)} \mid \mathbf{w}^T x^{(i)}, \sigma^2) \\ &= \frac{1}{\sigma\sqrt{2\pi}} \exp\Big(-\frac{(y^{(i)} - \mathbf{w}^T x^{(i)})^2}{2\sigma^2}\Big) \end{aligned}$$

- Log Likelihood

$$\mathcal{L}(\mathbf{w}) = \sum_{i=1}^{n} \Big(-\frac{1}{2}\log(2\pi\sigma^2) - \frac{(y^{(i)} - \mathbf{w}^T x^{(i)})^2}{2\sigma^2}\Big)$$

A Classic Linear Regression Example

Carnegie Mellon

- MLE estimator

$$\hat{\mathbf{w}}_{MLE} = \operatorname*{argmin}_{\mathbf{w}} \frac{1}{2\sigma^2} \sum_{i=1}^{n} (y^{(i)} - \mathbf{w}^T x^{(i)})^2$$

- With prior $\mathbf{w} \sim \mathbf{N}(\mathbf{0}, \lambda^{-1}\mathbf{I}) = \frac{1}{(2\pi)^{D/2}} \exp(-\frac{\lambda}{2}\mathbf{w}^T\mathbf{w})$ then MAP estimator

$$\hat{\mathbf{w}}_{MAP} = \operatorname*{argmin}_{\mathbf{w}} \frac{1}{2\sigma^2} \sum_{i=1}^{n} (y^{(i)} - \mathbf{w}^T x^{(i)})^2 + \frac{\lambda}{2}\mathbf{w}^T\mathbf{w}$$

A Classic Logistic Regression Example

Carnegie Mellon

- Binary classification with labels $\mathbf{y} \in \{-1, 1\}$ and logistic classification function: $\text{sigmoid}(x) = 1/(1 + \exp(-x))$
- Log Likelihood

$$\mathcal{L}(\mathbf{w}) = \sum_{i=1}^{n} - \log \left(1 + \exp(-y^{(i)} \mathbf{w}^T x^{(i)}\right)$$

- MLE estimator

$$\hat{\mathbf{w}}_{MLE} = \operatorname*{argmin}_{\mathbf{w}} \sum_{i=1}^{n} \log \left(1 + \exp(-y^{(i)} \mathbf{w}^T x^{(i)}\right)$$

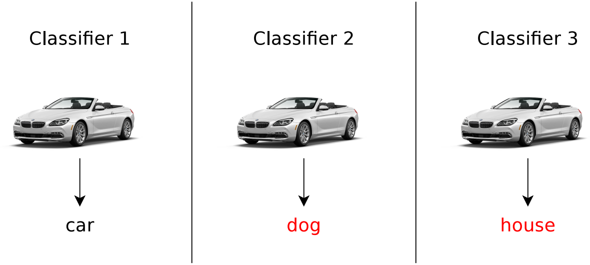
- With the same prior for $\mathbf{w}$, MAP estimator:

$$\hat{\mathbf{w}}_{MAP} = \operatorname*{argmin}_{\mathbf{w}} \sum_{i=1}^{n} \log \left(1 + \exp(-y^{(i)} \mathbf{w}^T x^{(i)}\right) + \frac{\lambda}{2} \mathbf{w}^T \mathbf{w}$$

Outline

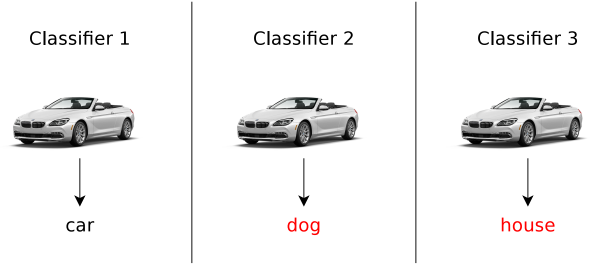
1. The Learning Problem
2. Data for Supervised Learning
3. Learning Models
4. Error and Loss Function

Loss Function



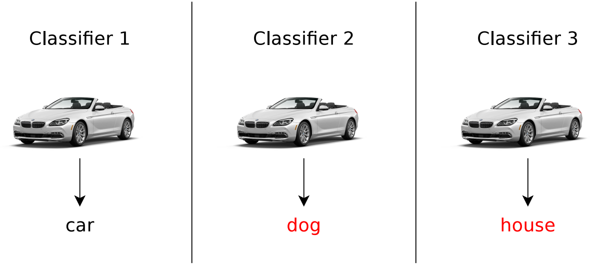
- Notations:
  - $\mathbf{x}$ is your data; $\mathbf{y}$ is your label;
  - $\mathbf{f}$ is your model; θ is your parameter;
  - $\hat{\mathbf{y}} = \mathbf{f}(\mathbf{x}; \theta)$ is your *empirical prediction.*

Loss Function



- Notations:
  - $\mathbf{x}$ is your data; $\mathbf{y}$ is your label;
  - $\mathbf{f}$ is your model; θ is your parameter;
  - $\hat{\mathbf{y}} = \mathbf{f}(\mathbf{x};\theta)$ is your *empirical prediction.*
- Loss function: $\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = \mathcal{L}(\mathbf{y}, \mathbf{f}(\mathbf{x}, \theta))$

Loss Function

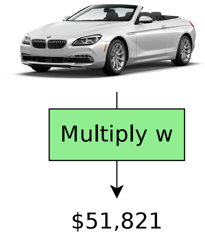


- Notations:
  - $\mathbf{x}$ is your data; $\mathbf{y}$ is your label;
  - $\mathbf{f}$ is your model; θ is your parameter;
  - $\hat{\mathbf{y}} = \mathbf{f}(\mathbf{x}; \theta)$ is your *empirical prediction.*
- Loss function: $\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = \mathcal{L}(\mathbf{y}, \mathbf{f}(\mathbf{x}, \theta))$
  - How "off" is $\mathbf{y}$ from $\hat{\mathbf{y}} = \mathbf{f}(\mathbf{x}; \theta)$

Case Study: Car prize prediction

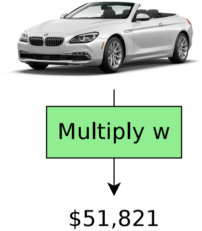
Carnegie Mellon

- Data: image of a car $\mathbf{x} \in \mathbb{R}^{3 \times 512 \times 512}$
- Label: cost of the car $\mathbf{x}$, namely $\mathbf{y} \in \mathbb{R}$
- Linear regression:
  - Parameters: $\theta = \{\mathbf{w} \in \mathbb{R}^{786432 \times 1}\}$, where $786432 = 3 \times 512 \times 512$.
  - $\mathbf{x}_1 = \text{reshape}(\mathbf{x}, [1, 786432])$
  - $\hat{\mathbf{y}} = \mathbf{x}_1 \cdot \mathbf{w}$



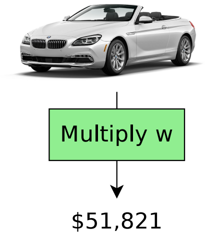
Case Study: Car prize prediction

- Data: image of a car $\mathbf{x} \in \mathbb{R}^{3 \times 512 \times 512}$
- Label: cost of the car $\mathbf{x}$, namely $\mathbf{y} \in \mathbb{R}$
- Linear regression:
  - Parameters: $\theta = \{\mathbf{w} \in \mathbb{R}^{786432 \times 1}\}$, where $786432 = 3 \times 512 \times 512$.
  - $\mathbf{x}_1 = \text{reshape}(\mathbf{x}, [1, 786432])$
  - $\hat{\mathbf{y}} = \mathbf{x}_1 \cdot \mathbf{w}$
  - $\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = (\hat{\mathbf{y}} - \mathbf{y})^2$ is called the ℓ_2-*loss*



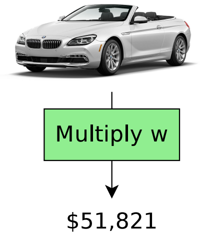
Case Study: Car prize prediction

- Data: image of a car $\mathbf{x} \in \mathbb{R}^{3\times512\times512}$
- Label: cost of the car $\mathbf{x}$, namely $\mathbf{y} \in \mathbb{R}$
- Linear regression:
  - Parameters: $\theta = \{\mathbf{w} \in \mathbb{R}^{786432\times1}\}$, where $786432 = 3 \times 512 \times 512$.
  - $\mathbf{x}_1 = \text{reshape}(\mathbf{x}, [1, 786432])$
  - $\hat{\mathbf{y}} = \mathbf{x}_1 \cdot \mathbf{w}$
  - $\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = (\hat{\mathbf{y}} - \mathbf{y})^2$ is called the ℓ_2-*loss*
  - $\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = |\hat{\mathbf{y}} - \mathbf{y}|$ is called the ℓ_1-*loss*



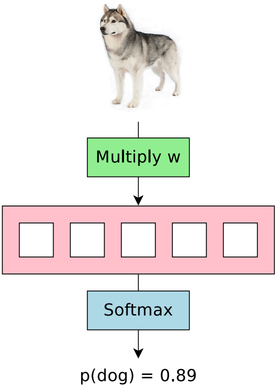
Case Study: Car prize prediction

- Data: image of a car $\mathbf{x} \in \mathbb{R}^{3\times512\times512}$
- Label: cost of the car $\mathbf{x}$, namely $\mathbf{y} \in \mathbb{R}$
- Linear regression:
  - Parameters: $\theta = \{\mathbf{w} \in \mathbb{R}^{786432\times1}\}$, where $786432 = 3 \times 512 \times 512$.
  - $\mathbf{x}_1 = \text{reshape}(\mathbf{x}, [1, 786432])$
  - $\hat{\mathbf{y}} = \mathbf{x}_1 \cdot \mathbf{w}$
  - $\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = (\hat{\mathbf{y}} - \mathbf{y})^2$ is called the ℓ_2-*loss*
  - $\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = |\hat{\mathbf{y}} - \mathbf{y}|$ is called the ℓ_1-*loss*
- Do we have other options of losses?



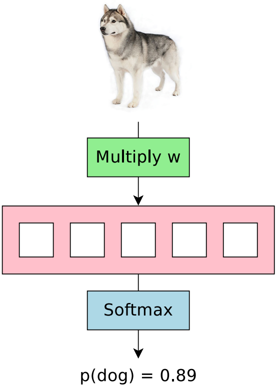
Case Study 2: Image classification

- Data: $\mathbf{x} \in \mathbb{R}^{3\times 32\times 32}$
- Label: $\mathbf{y} \in \{\text{dog}, \text{cat}, \text{house}, \text{car}, \text{flower}\}$
  - For ease: $\mathbf{y} \in \{1, 2, 3, 4, 5\}$
- Softmax classification:
  - Parameters: $\theta = \{\mathbf{w} \in \mathbb{R}^{3072\times 10}\}$.
  - $\mathbf{x}_1 = \text{reshape}(\mathbf{x}, [1, 3072])$
  - $\ell = \mathbf{x}_1 \cdot \mathbf{w}$
  - $\hat{p} = \mathbf{Prob}\left[\mathbf{y} = i\right] = \underbrace{\frac{\exp\{\ell_i\}}{\sum_{j=1}^{5} \exp\{\ell_j\}}}_{\text{soft-max}}$ l



Case Study 2: Image classification

- Data: $\mathbf{x} \in \mathbb{R}^{3\times 32\times 32}$
- Label: $\mathbf{y} \in \{\text{dog}, \text{cat}, \text{house}, \text{car}, \text{flower}\}$
  - For ease: $\mathbf{y} \in \{1, 2, 3, 4, 5\}$
- Softmax classification:
  - Parameters: $\theta = \{\mathbf{w} \in \mathbb{R}^{3072\times 10}\}$.
  - $\mathbf{x}_1 = \text{reshape}(\mathbf{x}, [1, 3072])$
  - $\ell = \mathbf{x}_1 \cdot \mathbf{w}$
  - $\hat{p} = \textbf{Prob}\left[\mathbf{y} = i\right] = \underbrace{\frac{\exp\{\ell_i\}}{\sum_{j=1}^{5} \exp\{\ell_j\}}}_{\text{soft-max}}$
  - Cross-entropy loss: $\mathcal{L}(\hat{p}, \mathbf{y}) = -\mathbf{y} \log \hat{p}_{\mathbf{y}}$.



Review: Information Theory

Carnegie Mellon

- Let $X \sim P(X)$ be a random variable
- (Shannon) information content of an outcome x is $h(x) = -\log_2 P(x)$

Image credit: David Mackay

Review: Information Theory

Carnegie Mellon

- Let $X \sim P(X)$ be a random variable
- (Shannon) information content of an outcome x is $h(x) = -\log_2 P(x)$
- In: bits/nats/shannons/dits/bans/hartleys

Image credit: David Mackay

Review: Information Theory

Carnegie Mellon

- Let $X \sim P(X)$ be a random variable
- (Shannon) information content of an outcome x is $h(x) = -\log_2 P(x)$
- In: bits/nats/shannons/dits/bans/hartleys
- Measure the “uncertainty” of an outcome

Image credit: David Mackay

Review: Information Theory

Carnegie Mellon

- Let $X \sim P(X)$ be a random variable
- (Shannon) information content of an outcome x is $h(x) = -\log_2 P(x)$
- In: bits/nats/shannons/dits/bans/hartleys
- Measure the "uncertainty" of an outcome
- Entropy of a set S is the average information content:

$$\mathrm{H}(X) = \sum_{x \in S} -P(x) \log P(x)$$
$$= -\mathbb{E}_{x \sim P}[logP(x)]$$

i	a_i	p_i	$h(p_i)$
1	a	.0575	4.1
2	b	.0128	6.3
3	c	.0263	5.2
4	d	.0285	5.1
5	e	.0913	3.5
6	f	.0173	5.9
7	g	.0133	6.2
8	h	.0313	5.0
9	i	.0599	4.1
10	j	.0006	10.7
11	k	.0084	6.9
12	l	.0335	4.9
13	m	.0235	5.4
14	n	.0596	4.1
15	o	.0689	3.9
16	p	.0192	5.7
17	q	.0008	10.3
18	r	.0508	4.3
19	s	.0567	4.1
20	t	.0706	3.8
21	u	.0334	4.9
22	v	.0069	7.2
23	w	.0119	6.4
24	x	.0073	7.1
25	y	.0164	5.9
26	z	.0007	10.4
27	-	.1928	2.4
$\sum_i p_i \log_2 \frac{1}{p_i}$			4.1

Image credit: David Mackay

Review: Information Theory

- Joint Entropy of X, Y is

$$\mathrm{H}(X, Y) = -\sum_{x,y} P(x, y) \log P(x, y)$$

- Conditional entropy

$$\mathrm{H}(X|Y) = -\sum_{y} \mathrm{H}(X|Y = y) P(Y = y) = -\sum_{x,y} P(x, y) \log P(x|y)$$

- Chain rule:

$$\mathrm{H}(X|Y) = \mathrm{H}(X, Y) - \mathrm{H}(Y)$$

- Bayes' rule:

$$\mathrm{H}(Y|X) = \mathrm{H}(X|Y) - \mathrm{H}(X) + \mathrm{H}(Y)$$

Review: Information Theory

- Mutual Information (Information gain) of X and Y:

$$\mathrm{I}(X,Y) = \sum_{x,y} P(x,y) \log \frac{P(x,y)}{P(x)P(y)}$$

- Relation to conditional entropy:

$$\begin{aligned}\mathrm{I}(X,Y) &= \mathrm{H}(X) - \mathrm{H}(X|Y) \\ &= \mathrm{H}(Y) - \mathrm{H}(Y|X) \\ &= \mathrm{H}(X) + \mathrm{H}(Y) - \mathrm{H}(X,Y)\end{aligned}$$

- If X, Y are independent, we have:

$$\mathrm{I}(X,Y) = 0$$

$$\mathrm{H}(X|Y) = H(X)$$

$$\mathrm{H}(X,Y) = \mathrm{H}(X) + \mathrm{H}(Y)$$

Review: Information Theory

- Relative Entropy or Kullback-Leibler (KL) divergence of 2 distributions P(x) and Q(x) over the same set is:

$$\mathbf{D}_{KL}(P \,\|\, Q) = \sum_x P(x) \log \frac{P(x)}{Q(x)} = \mathbb{E}_{x \sim P}[P(x)] - \mathbb{E}_{x \sim P}[Q(x)]$$

- Is Asymmetric:

$$\mathbf{D}_{KL}(P \,\|\, Q) \neq \mathbf{D}_{KL}(Q \,\|\, P)$$

- Satisfy Gibbs inequality: $\mathbf{D}_{KL}(P \,\|\, Q) \geq 0$, with equality happens iff $P \equiv Q$

Review: Information Theory

- Cross-Entropy of P and Q over the same set:

$$\mathrm{H}(P, Q) = -\sum_{x \sim P} P(x) \log Q(x) = -\mathbb{E}_{x \sim P}[\log Q(x)]$$

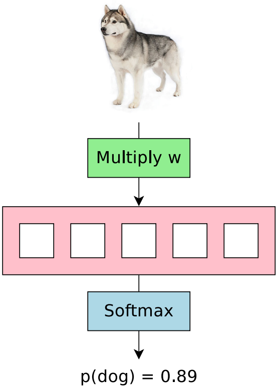
- Relation to relative entropy:

$$\mathrm{H}(P, Q) = \mathrm{H}(P) + \mathbf{D}_{KL}(P \,\|\, Q)$$

- Let P be the ground-truth distribution, and Q be the predicted distribution

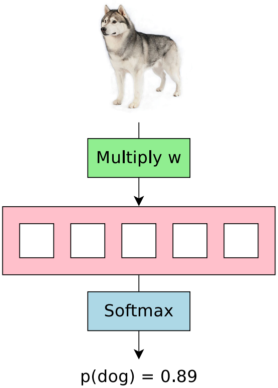
Cross-Entropy Loss

- $\mathcal{L}(\hat{p}, \mathbf{y}) = \mathrm{H}(y, \hat{p}_{\mathbf{y}}) = -\mathbf{y} \log \hat{p}_{\mathbf{y}}$
  - One of the most important loss functions of deep learning, and classification in general.



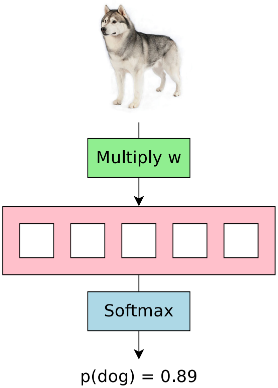
Cross-Entropy Loss

- $\mathcal{L}(\hat{p}, \mathbf{y}) = \mathrm{H}(y, \hat{p}_{\mathbf{y}}) = -\mathbf{y} \log \hat{p}_{\mathbf{y}}$
  - One of the most important loss functions of deep learning, and classification in general.
  - $\mathcal{L}$ is mall when $\hat{p}_{\mathbf{y}}$ is large, i.e. model is more confident



Cross-Entropy Loss

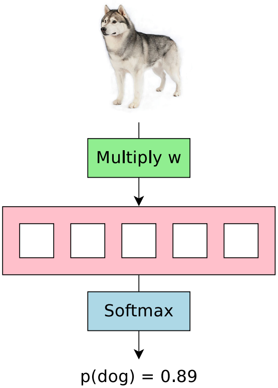
- $\mathcal{L}(\hat{p}, \mathbf{y}) = \mathrm{H}(y, \hat{p}_{\mathbf{y}}) = -\mathbf{y} \log \hat{p}_{\mathbf{y}}$
  - One of the most important loss functions of deep learning, and classification in general.
  - $\mathcal{L}$ is mall when $\hat{p}_{\mathbf{y}}$ is large, i.e. model is more confident
  - But $\mathcal{L}$ is *always positive*



Cross-Entropy Loss

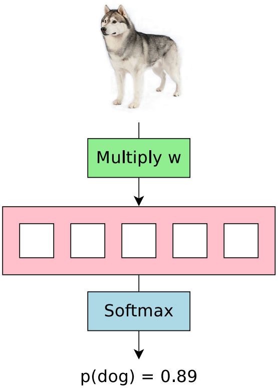
Carnegie Mellon

- $\mathcal{L}(\hat{p}, \mathbf{y}) = \mathrm{H}(y, \hat{p}_{\mathbf{y}}) = -\mathbf{y} \log \hat{p}_{\mathbf{y}}$
    - One of the most important loss functions of deep learning, and classification in general.
    - $\mathcal{L}$ is mall when $\hat{p}_{\mathbf{y}}$ is large, i.e. model is more confident
    - But $\mathcal{L}$ is *always positive*
    - When $\hat{p}_{\mathbf{y}}$ is large, $\hat{p}_{\neq \mathbf{y}}$ are small



Cross-Entropy Loss

- $\mathcal{L}(\hat{p}, \mathbf{y}) = \mathrm{H}(y, \hat{p}_{\mathbf{y}}) = -\mathbf{y} \log \hat{p}_{\mathbf{y}}$
  - One of the most important loss functions of deep learning, and classification in general.
  - $\mathcal{L}$ is mall when $\hat{p}_{\mathbf{y}}$ is large, i.e. model is more confident
  - But $\mathcal{L}$ is *always positive*
  - When $\hat{p}_{\mathbf{y}}$ is large, $\hat{p}_{\neq\mathbf{y}}$ are small
  - Making more probabilistic sense



Cross-Entropy Loss

- $\mathcal{L}(\hat{p}, \mathbf{y}) = \mathrm{H}(y, \hat{p}_{\mathbf{y}}) = -\mathbf{y} \log \hat{p}_{\mathbf{y}}$
  - One of the most important loss functions of deep learning, and classification in general.
  - $\mathcal{L}$ is mall when $\hat{p}_{\mathbf{y}}$ is large, i.e. model is more confident
  - But $\mathcal{L}$ is *always positive*
  - When $\hat{p}_{\mathbf{y}}$ is large, $\hat{p}_{\neq \mathbf{y}}$ are small
  - Making more probabilistic sense
  - Differentiable. Recall $\hat{p}_i = \exp\{\ell_i\} / \sum_j \exp\{\ell_j\}$
    - Important for learning.