ECE521 Lecture2

Fundamentals of Machine Learning

UNIVERSITY OF TORONTO

Outline

- **Review of probability**
- Data representation in ML
- Problem formulation
- Type of loss functions
- Learning algorithms

Notations

- A sample space $\mathcal{S}$ is a set of possible outcomes in a random experiment

e.g. a dice roll: $\mathcal{S} = \{1, 2, 3, 4, 5, 6\}$, Stock prices: $\mathcal{S} = \mathbb{R}^+ = (0, \infty)$

- An event $E \subseteq \mathcal{S}$, e.g. a dice roll obtaining a face greater than 3

- A random variable (RV) X is a function/mapping: $X : \mathcal{S} \to \mathbb{R}$

e.g. the sum of two dice roll is 6, $X = 6$,
which is an event set for all s such that $X(s) = 6$

- Allocation of probability to events / random variables, $P(\mathcal{S}) = 1$

$P(E) \geq 0$

Simplified notation for functions: $P(X = x) \equiv P(x)$

$f_X(x) \equiv P(x)$

Discrete Random Variables

The sample space $\mathcal{S}$ is discrete and countable, we can measure the probabilities using probability mass function: $0 \leq P(x) \leq 1, \forall x \in \mathcal{S}$

Note that sum of the probabilities is always one $\sum_{x \in \mathcal{S}} P(x) = 1$

e.g. fair dice roll $\mathcal{S} = \{1, 2, 3, 4, 5, 6\}$, $P(x = 6) = \frac{1}{6}$

fair coin flip $\mathcal{S} = \{head, tail\}$, $P(x = head) = 1/2$

image classification:

$\mathcal{S} = \{chest, honeycomb, French\_loaf, stole, thimble, velvet, \cdots\} P(x = chest) = 0.3942$

Continuous Random Variables

The sample space $\mathcal{S}$ is continuous, we use probability density function to represent a point measurement: $f_X(x) \equiv P(x) \geq 0, \forall x \in \mathcal{S}$

Note that sum of the probabilities is always one $\int_{\mathcal{S}} P(x)dx = 1$

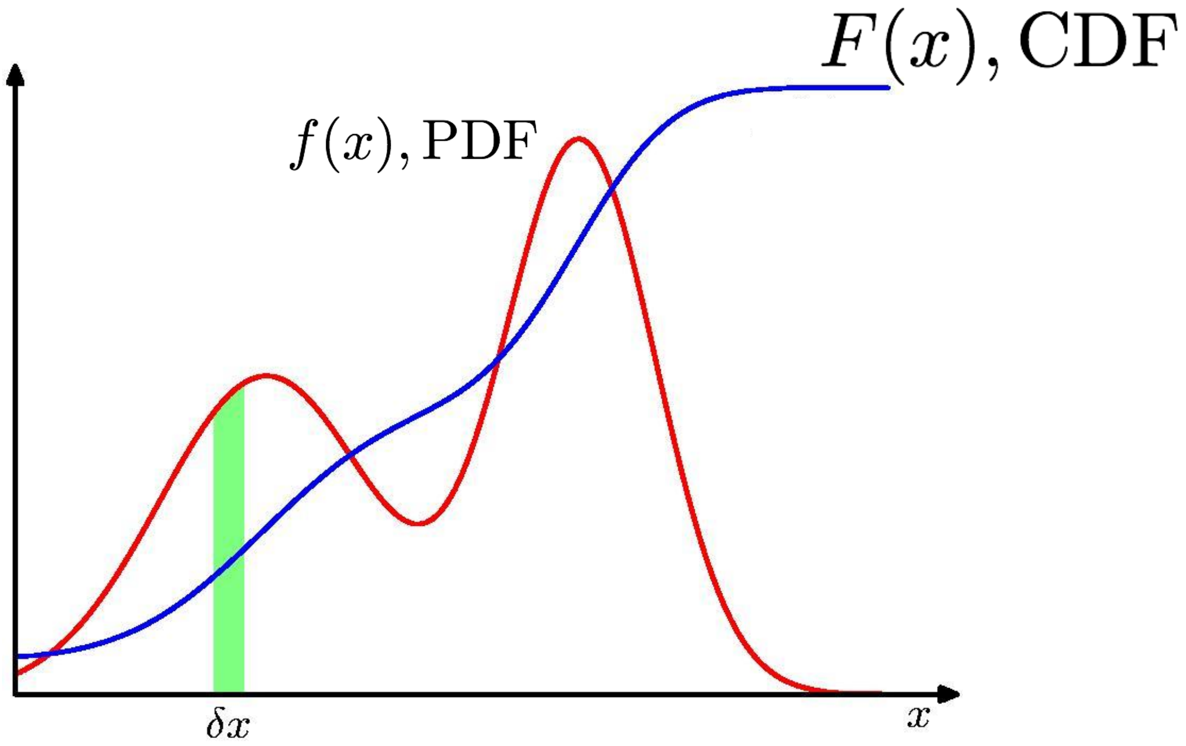
e.g. model all real numbers: $\mathcal{S} = \mathbb{R}$, Gaussian dist. : $P(x) = \mathcal{N}(x; \mu, \sigma^2)$

where $\mathcal{N}(x; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$

model an interval: $\mathcal{S} = [0, 1]$, Beta dist. : $P(x) = \mathrm{Beta}(x; \alpha, \beta)$

where, $\mathrm{Beta}(x; \alpha, \beta) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{B(\alpha, \beta)}$

Continuous Random Variables



Statistics of Random Variables

Expected value of a random variable x:

Expectation measures the average outcome

Discrete: $$\mathbb{E}[x] = \sum_{x \in \mathcal{S}} xP(x) \,,$$

$$\mathbb{E}[\text{fair dice roll}] = 1/6 + 2/6 + 3/6 + 4/6 + 5/6 + 6/6 = 3.5$$

Continuous: $$\mathbb{E}[x] = \int_{\mathcal{S}} xP(x)dx$$ if x has Gaussian dist. $\mathbb{E}[x] = \mu$

Statistics of Random Variables

Variance of a random variable x:

Discrete: $$Var[x] = \sum_{x \in \mathcal{S}} (x - \mathbb{E}[x])^2 P(x)$$

$$\begin{aligned} Var[\text{fair dice outcome}] &= (1-3.5)^2/6 + (2-3.5)^2/6 + (3-3.5)^2/6 + (4-3.5)^2/6 + (5-3.5)^2/6 + (6-3. \\ &= 2.9 \end{aligned}$$

Continuous: $$Var[x] = \int_{\mathcal{S}} (x - \mathbb{E}[x])^2 P(x) dx$$

if x has Gaussian dist. $Var[x] = \sigma^2$

Additive Rule: Marginal Probability

Given two random variables x,y

Joint probability: $P(x, y) \equiv P(x \cap y) \equiv P(x \wedge y)$

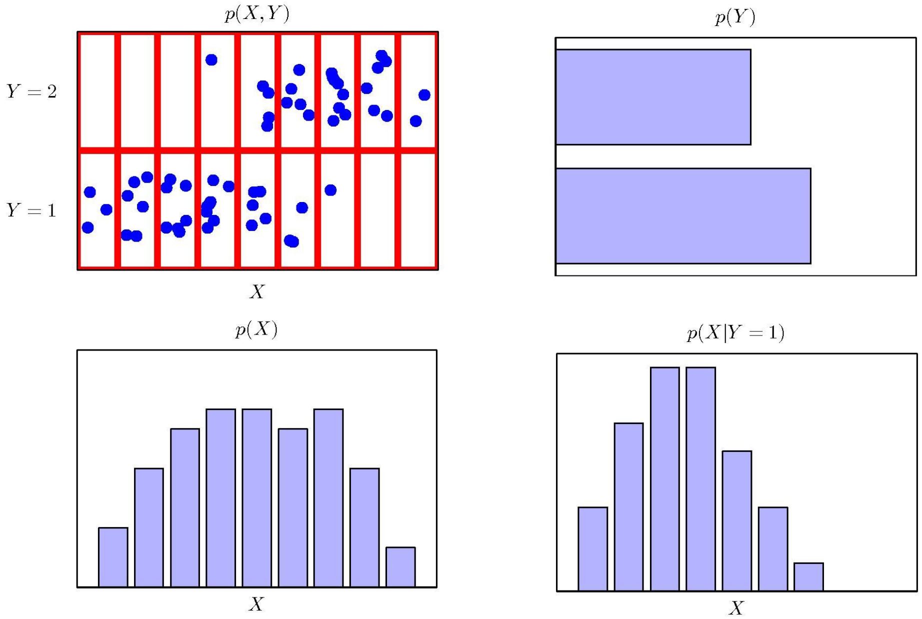
Marginal probability: $P(x)$ and $P(y)$

Marginalization: discrete $$P(y) = \sum_{x \in \mathcal{S}_x} P(x, y)$$

continuous $$P(y) = \int_{\mathcal{S}_x} P(x, y)\, dx$$

Example

Distribution over two variables: X takes on 9 possible values, and Y takes on 2 possible values.



Multiplicative Rule: Conditional Probability

Given two random variables x,y

Conditional probability: $P(x|y)$ and $P(y|x)$

Joint probability: $P(x,y)$

$$=P(y|x)P(x)$$

$$=P(x|y)P(y)$$

e.g. P(snowing and school cancel)

=P(school cancel | snowing) P(snowing)

=> $P(x|y) = \frac{P(x,y)}{P(y)}$

Always true for both discrete and continuous

Total Probability Theorem

We can rewrite the marginalization rule using conditional probability:

Total probability theorem:

discrete:

$$P(x) = \sum_{y \in \mathcal{S}_y} P(y)P(x|y)$$

Continuous:

$$P(y) = \int_{\mathcal{S}_x} P(y)P(x|y)\,dx$$

Bayes' Rule

Using conditional probability definition:

$$P(x,y) = P(y|x)P(x) = P(x|y)P(y)$$

Bayes' rule: $$P(y|x) = \frac{P(x|y)P(y)}{P(x)}$$

Apply total probability theorem: $$P(y|x) = \frac{P(x|y)P(y)}{\sum_{y\in\mathcal{S}_y} P(x|y)P(y)}$$

Summary

- Notations for Random Variables (RVs)
- Discrete and Continuous RVs: pmf, pdf, cdf
- Statistics of RVs: expectation, variance
- Most important tools in probability:
    a. marginalization
    b. conditioning
    c. total probability theorem
    d. Bayes' rule

Outline

- Review of probability
- **Data representation in ML**
- Problem formulation
- Type of loss functions
- Learning algorithms

How do we represent data in ML?

- So we finally collected an amazing image dataset of two classes:

source

How do we represent data in ML?

- We can represent the images as a vector pixels intensities:
    - Here we will have a vector length of H x W x C

$$\longrightarrow \begin{bmatrix} 25 \\ 32 \\ 33 \\ 38 \\ 43 \\ 45 \\ 41 \end{bmatrix} \\ \vdots$$

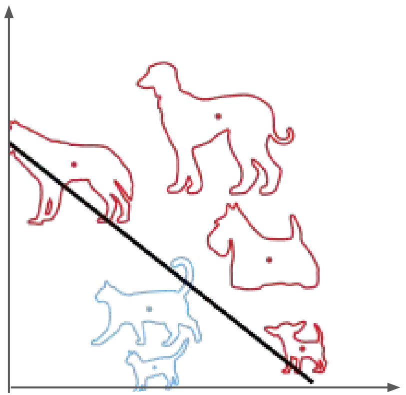
How do we represent data in ML?

- In general, we will use vector to represent a data point of n-dimension
  - The elements in the vector can be either integers or continuous real values

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \cdot \\ \cdot \\ \cdot \\ x_n \end{bmatrix}$$

How do we represent data in ML?

- We can represent the input data points now in a coordinate space
  - A classification task is about learning a decision boundary that separate out the classes in the dataset.



source

How do we represent data in ML?

- We often encounter discrete categorical data:
    - Labels of classification tasks, e.g. {cat, bat, dog, frog}
    - Discrete actions for a robot, e.g. {accelerate, brake, turn left, turn right, …}
    - English lexicon, e.g. {the, he, …, king, boy, …, stringyfy, …}
- One useful representation is to encode the K categories into a one-of-K representation
    - cat -> [1, 0, 0, 0], bat ->[0, 1, 0, 0], …
    - The ordering can be arbitrary
    - More sophisticated coding scheme can be used to optimize the encoding length, e.g. Hoffman coding

How do we represent data in ML?

- Sequential data is very common in speech processing, time series prediction, natural language processing (NLP), machine translation:
  - Any sequential data can be represented as a sequence of vectors
  - $(\mathbf{x}_1, \ldots, \mathbf{x}_T), \quad \text{where } \mathbf{x}_t \in \mathcal{X}$

Outline

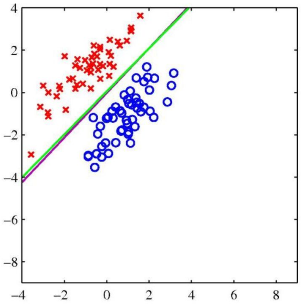
- Review of probability
- Data representation in ML
- **Problem formulation**
- Type of loss functions
- Learning algorithms

What is a machine learning problem?

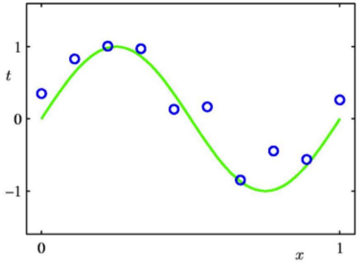
- Under the supervised learning:
  - Given a training dataset of some input $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(M)}\}$ and output $\{\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}\}$
  - Learn a function f , such that $\hat{\mathbf{y}} = f(\mathbf{x})$ relationship closely mimic the training dataset

What is a machine learning problem?

- For example, there are classification tasks in which targets are categorical class labels. f here is a classifier.

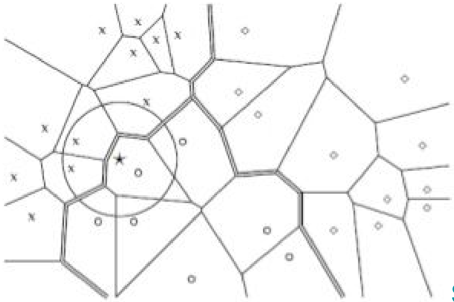


- Regression tasks are to predict targets of continuous values.



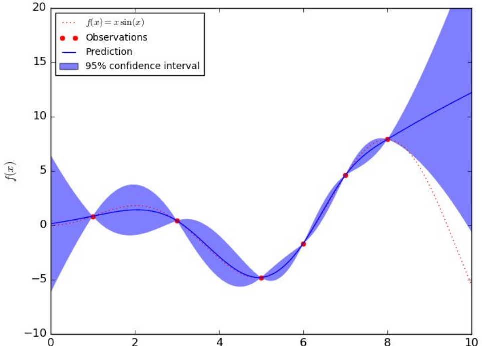
Examples of prediction models

- Non-parametric models:
  - K Nearest Neighbour (K-NN):



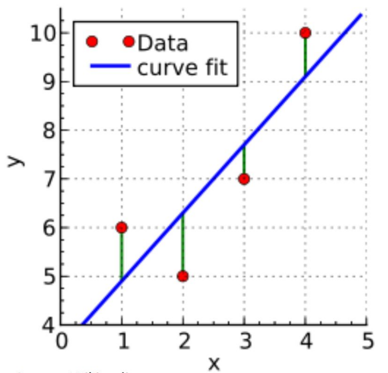
source

  - Gaussian processes:



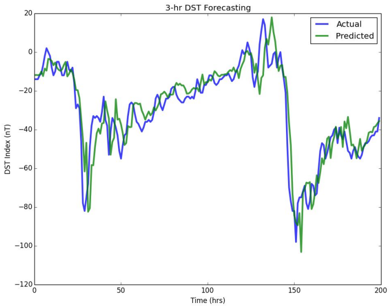
Examples of prediction models

- Parametric models:
  - Linear regression



Source: Wikipedia

  - Neural networks



The art of model selection

- The actual function that generates the training dataset is unknown
- Solving machine learning problem requires us to find a hypothesis to explain the dataset
- Formally, we would like to find a function f in the hypothesis space $\mathcal{H}$ that approximates the underlying function that generated the training dataset
- The set of all possible hypothesis is the hypothesis space $\mathcal{H}$

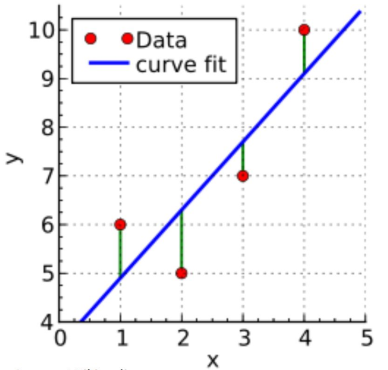
The art of model selection

- Hypothesis space in the simplest case can be the family of the prediction function f
    - For example, in K-NN, $\mathcal{H}$ is all the K from 1 to the number of data points
    - In linear regression, $\mathcal{H}$ is the all the possible weight configuration in $\mathbb{R}^n$
- It is an art to select which family of function to use. It is also possible to consider a hypothesis space that include two different family of functions.
- A more rigorous treatment of model selection will be covered later in the course

How good are the predictions?

- The learnt function f should closely mimic the training examples

- We need to measure how close the predictions are to our training data examples



Source: Wikipedia

Outline

- Review of probability
- Data representation in ML
- Problem formulation
- **Type of loss functions**
- Learning algorithms

Type of loss functions

- The simplest and most intuitive loss function is the 0-1 loss:

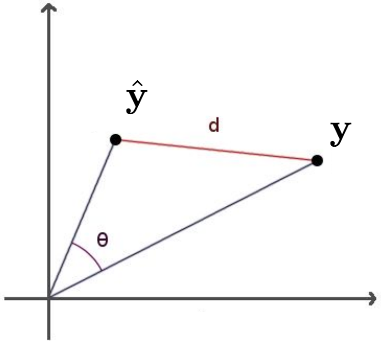
$$I(\mathbf{y} \neq \hat{\mathbf{y}}) = \begin{matrix} 0, & \mathbf{y} = \hat{\mathbf{y}} \\ 1, & \mathbf{y} \neq \hat{\mathbf{y}} \end{matrix}$$

- It is “hard” and not continuous
- It does not have an implicit “neighbourhood”

Type of loss functions

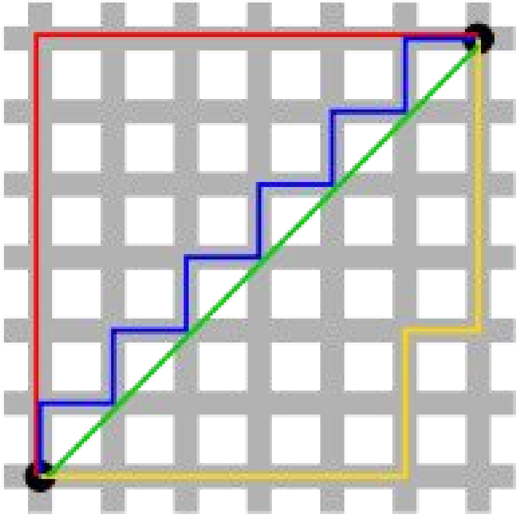
- Squared error or “squared Euclidean distance” or squared L2 distance is a natural measure of distance in our world:

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_2^2 = \sum_{i=1}^{N} (\hat{y}_i - y_i)^2$$



Type of loss functions

- Consider measure distances between two city blocks as an Uber driver:

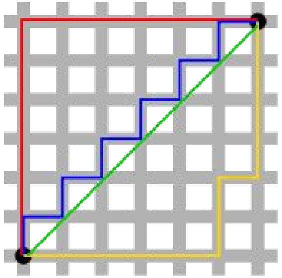


source

Type of loss functions

- The Taxicab geometry is also known as L1 distance

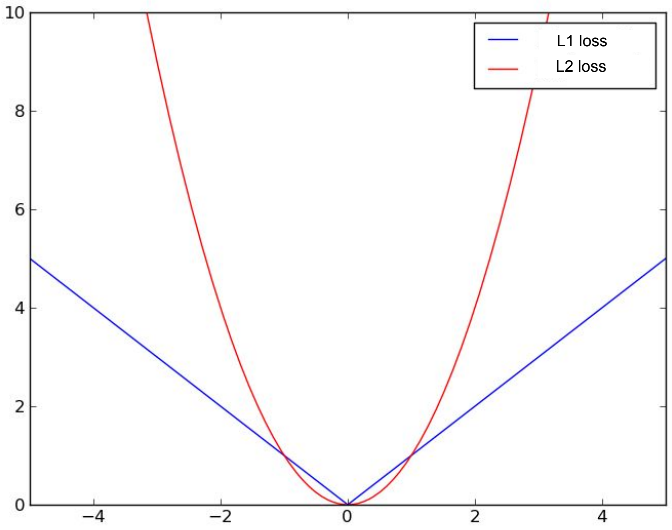
$$\|\hat{\mathbf{y}} - \mathbf{y}\|_1 = \sum_{i=1}^{N} |\hat{y}_i - y_i|$$



source

Type of loss functions

- Visualizing squared error vs L1 loss



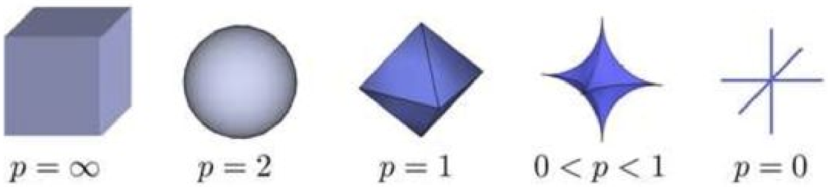
Type of loss functions

- We can measure the distance in general Lp space

$$\|\hat{\mathbf{y}} - \mathbf{y}\|_p^p = \sum_{i=1}^{N} (|\hat{y}_i - y_i|)^p$$

Type of loss functions

- Visualizing Lp distances



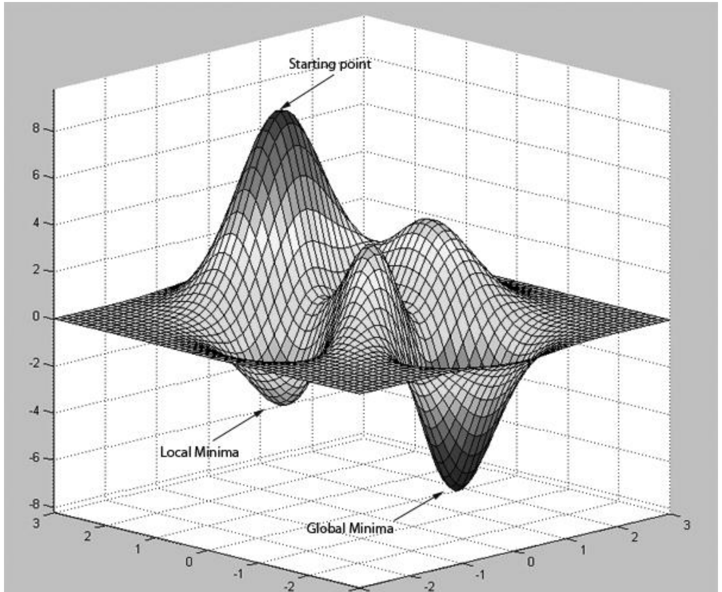
source

Outline

- Review of probability
- Data representation in ML
- Problem formulation
- Type of loss functions
- **Learning algorithms**

Learning algorithms

- Searching for a model in the hypothesis space can be visualized as moving around on the error surface defined by the loss function:
  - We would like our model to minimize the loss function



source

Learning algorithms

- A typical machine learning problem can be casted as a double nested for-loop:

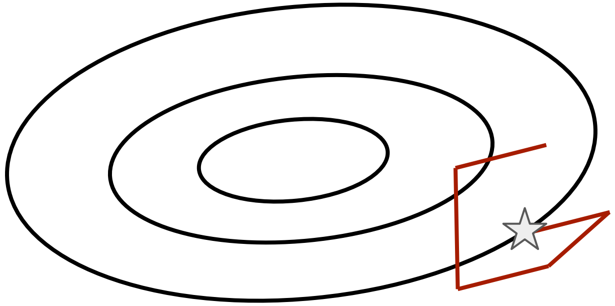
```
for some passes over the dataset (epochs):
    for some iterations:
        find/refine a search_direction from data minimizing our loss function
    model.update(search_direction)
```

Learning algorithms

- How we can learn our models efficiently?

Learning algorithms

- Let us look at inefficient learning first:
    - Random search



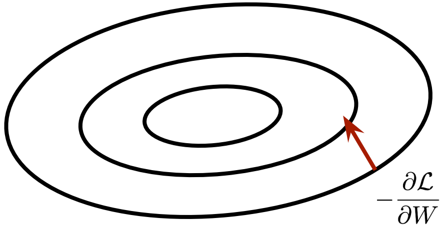
random search after 5 epochs

Learning algorithms

- Let us look at inefficient learning first:
  - hill climbing(/sliding): find a local descent direction within a local neighbourhood and make incremental improvements.
  - A local descent direction is a direction that decreases the loss function locally.
  - For example, binary search is a form of hill climbing algorithm

Learning algorithms

- Steepest descent:
  - Instead of a random search direction, we can find the direction of the most rapid increase of the loss function $\mathcal{L}$ using calculus: $\frac{\partial \mathcal{L}}{\partial W}$
  - The partial derivative of the loss function $\mathcal{L}$ with respect to the model parameters W is the gradient of the loss function
  - If we follow the negative gradient direction closely with a small enough increments to our model parameters, we can find a minimum eventually.



$-\frac{\partial \mathcal{L}}{\partial W}$ negative gradient direction

Learning algorithms

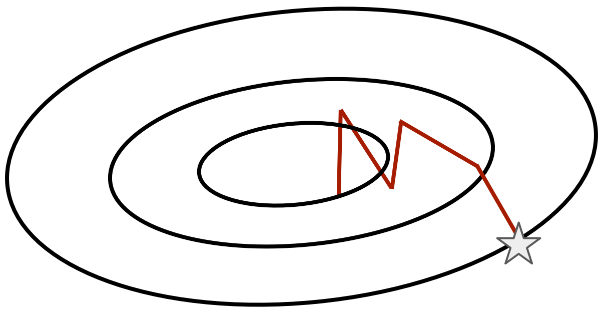
- Steepest descent:
  - The steepest descent or the gradient descent algorithm is one of the simplest optimization algorithms.
  - The “inner-loop” of the steepest descent accumulates the gradient direction from each data points in the loss function
  - The model is updated as:

$$W \leftarrow W - \eta \frac{\partial \mathcal{L}}{\partial W}$$

  - η is the learning rate

Learning algorithms

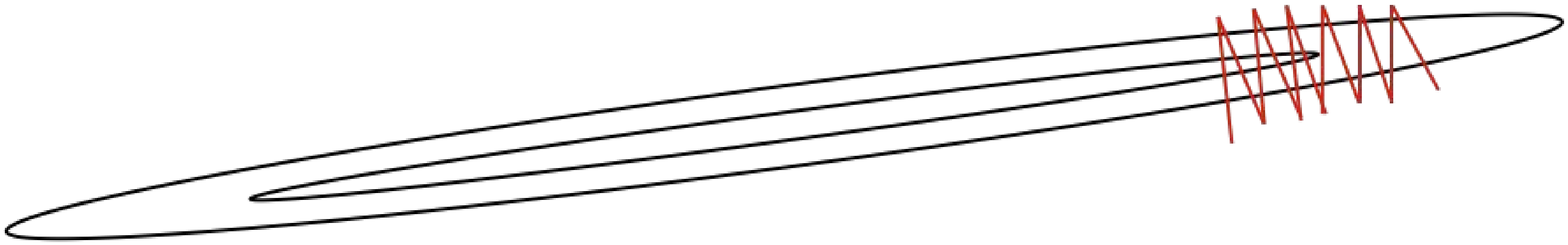
- Steepest descent:



Gradient descent after 5 epochs

Learning algorithms

- Steepest descent:
  - Sometimes, gradient descent can be very inefficient for the outer loop



Ill-conditioned loss function, that is
elongated along some direction