ECE521 lecture 3:
16 January 2017

kNN,
convexity in optimization

Overview

- **kNN**
- Optimization

Example: Nearest Neighbours

- Given a training set of M training examples:

$$\{(\mathbf{x}^{(m)}, \mathbf{t}^{(m)})\}, \quad \text{where} \quad \mathbf{x}^{(m)} \in \mathbb{R}^N$$

- The idea is to estimate the target function from the value(s) of the *nearest* (in Euclidean space) training example(s)
- Distance is

$$\text{squared error} = \|\mathbf{x}^{(i)} - (\mathbf{x})^{(j)}\|_2^2 = \sum_{n=1}^{N} (x_n^{(i)} - x_n^{(j)})^2$$

Algorithm:

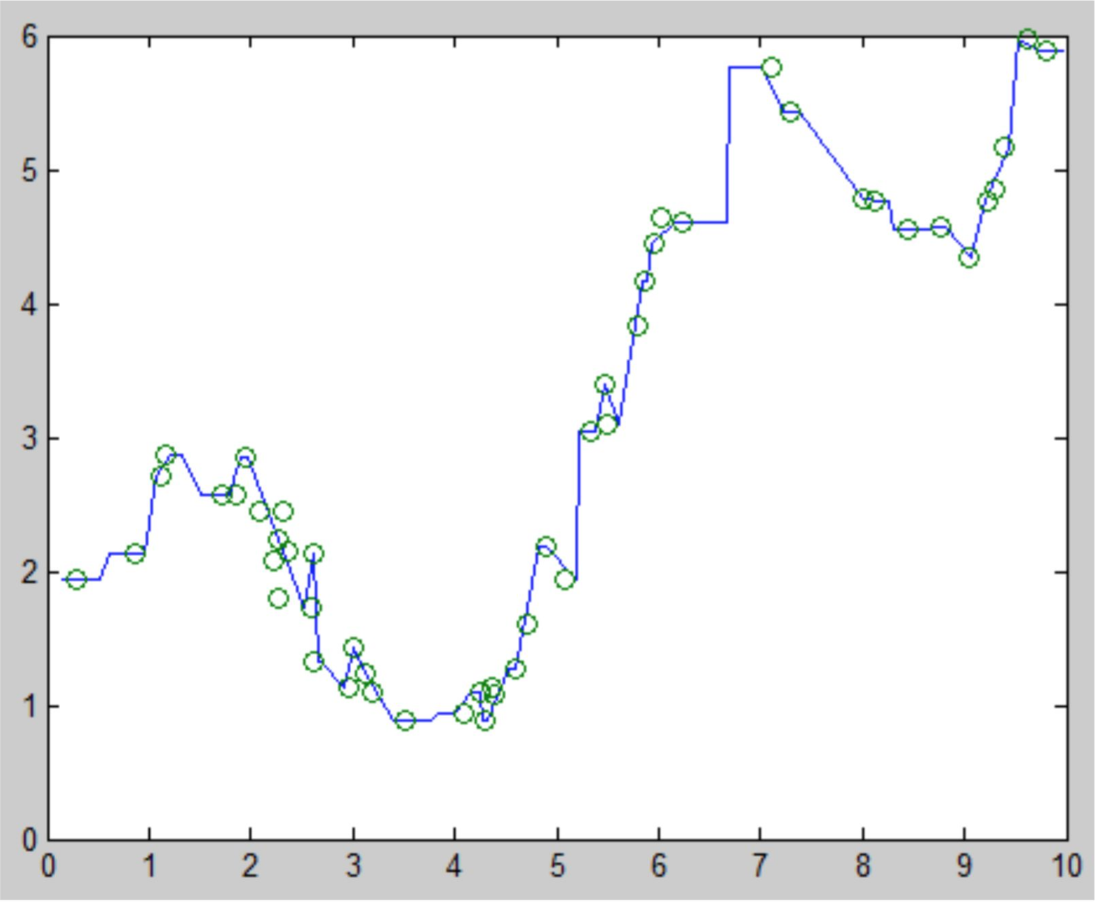
1. Find example $(\mathbf{x}^*, t^*)$ (from the stored training set) closest to the test instance $\mathbf{x}$. That is:

$$\mathbf{x}^* = \underset{\mathbf{x}^{(i)} \in \text{train. set}}{\operatorname{argmin}} \; \text{distance}(\mathbf{x}^{(i)}, \mathbf{x})$$

2. Output $y = t^*$

Example: k Nearest Neighbours (kNN)

- k-NN as a regression model:

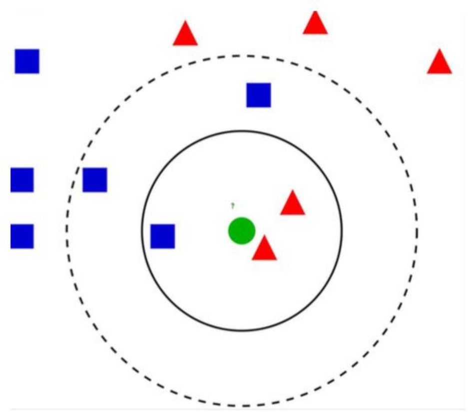


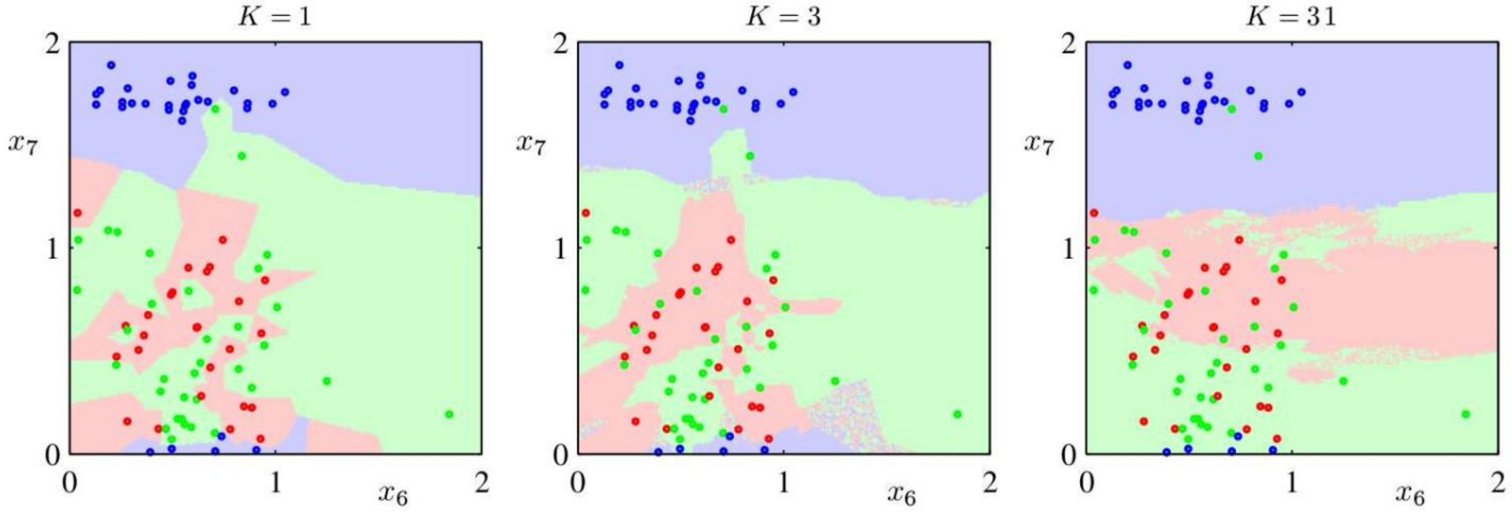
Example: k Nearest Neighbours (kNN)

- Instead of finding a the closest training example, search can be extended to k nearest points.
  - k is hyper-parameter (i.e. a parameter that encodes our prior belief about the solution space of a problem)
  - As k increases, the learnt target function becomes smoother

Example: k Nearest Neighbours (kNN)

- Visualize decision boundaries in K-NN classifiers:





Example: k Nearest Neighbours (kNN)

- K-NN in its standard form:
  - There is no parameter
  - There is one hyper-parameters, K

- Consider quantize the whole input space so it can be represented as a table. Our training set only occupies a tiny amount of entries in this table. The nearest neighbour assumption tells us to fill in the missing entries by their neighbouring values.
  - In other words, K-NN interpolates/extrapolates data points using a constant function assumption.

Example: k Nearest Neighbours (kNN)

- Quiz time:
  - Consider a binary classification task using a training set of 100 examples and equal split of two classes and uniformly distributed in the input space. We decided to use K-NN to solve this task. What is the classification accuracy on the **training set** when K=1 ?

Example: k Nearest Neighbours (kNN)

- Quiz time:
  - Consider a binary classification task using a training set of 100 examples and equal split of two classes and uniformly distributed in the input space. We decided to use K-NN to solve this task. What is the classification accuracy on the **training set** when K=1 ?

- answer: accuracy is 100%

Example: k Nearest Neighbours (kNN)

- Quiz time:
  - Consider a binary classification task using a training set of 100 examples and equal split of two classes and uniformly distributed in the input space. We decided to use K-NN to solve this task. What is the classification accuracy on the **training set** when K=3 ?

Example: k Nearest Neighbours (kNN)

- Quiz time:
  - Consider a binary classification task using a training set of 100 examples and equal split of two classes and uniformly distributed in the input space. We decided to use K-NN to solve this task. What is the classification accuracy on the **training set** when K=3 ?

- answer: accuracy is 1-P(two or more neighbours are from the other class) = 1 - 0.5^2 = 75%

Example: k Nearest Neighbours (kNN)

- Quiz time:
  - Consider a binary classification task using a training set of 100 examples and equal split of two classes and uniformly distributed in the input space. We decided to use K-NN to solve this task. What is the classification accuracy on the **training set** when K=100 ?

Example: k Nearest Neighbours (kNN)

- Quiz time:
  - Consider a binary classification task using a training set of 100 examples and equal split of two classes and uniformly distributed in the input space. We decided to use K-NN to solve this task. What is the classification accuracy on the **training set** when K=100 ?

- answer: accuracy is random guessing 50%

Example: k Nearest Neighbours (kNN)

- Quiz time:
  - Does the performance of K-NNs always get better as K increases?

Example: k Nearest Neighbours (kNN)

- Quiz time:
  - Does the performance of K-NNs always get better as K increases?

- answer: NO! Homework question: Think about a construction in 1-D such that classification accuracy is a periodic function of K.

Overview

- kNN
- **Optimization**

Optimization

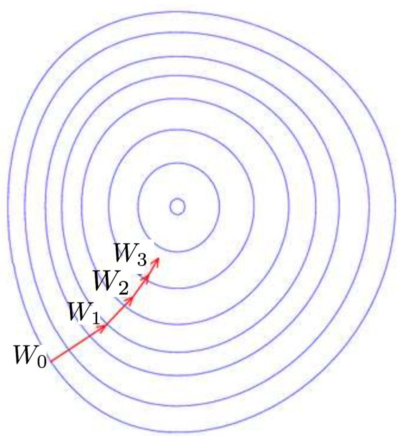
At the heart of any machine learning system, there is an optimization algorithm.

Optimization

- We covered the Steepest Descent / Gradient Descent algorithm last week.
  - The algorithm adjust model parameters to decrease a loss function by following the gradient direction.

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

  - It is an iterative local search algorithm for models that are continuous and differentiable.



Optimization

- Formally, gradient descent is an optimization algorithm that solves the following problem:

$$
\begin{aligned}
\min_{W} \quad & \mathcal{L}(W) \\
s.t. \quad & c(W)
\end{aligned}
$$

  - Find some weights/ model parameters that minimizes the loss function L subject to some constraint c(W)
  - Minimization is equivalent to maximization of the negative loss

Optimization

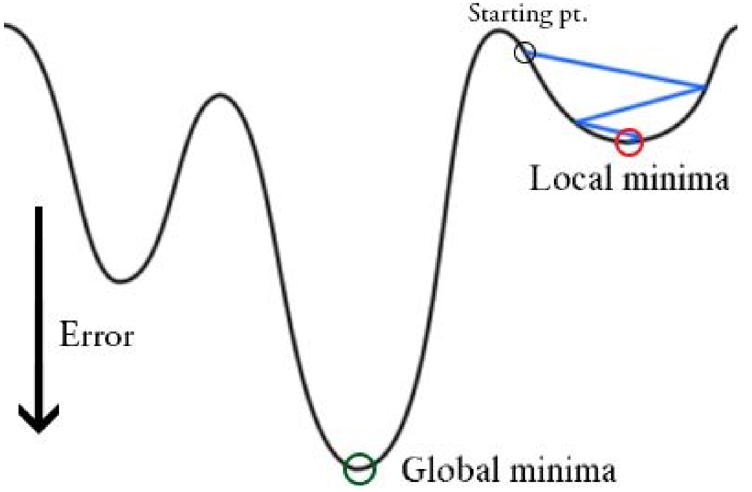
- Formally, gradient descent is an optimization algorithm that solves the following problem:

$$
\begin{aligned}
\min_{W} \quad & \mathcal{L}(W) \\
s.t. \quad & c(W)
\end{aligned}
$$

  - Find some weights/ model parameters that minimizes the loss function L subject to some constraint c(W)
  - Minimization is equivalent to maximization of the negative loss

Optimization

- We have talked about local optimal and global optimal
- Find the global optimal in general for any loss function is NP-hard
- The problem of finding global optimal is provably easy for a certain class of loss functions, e.g. convex functions

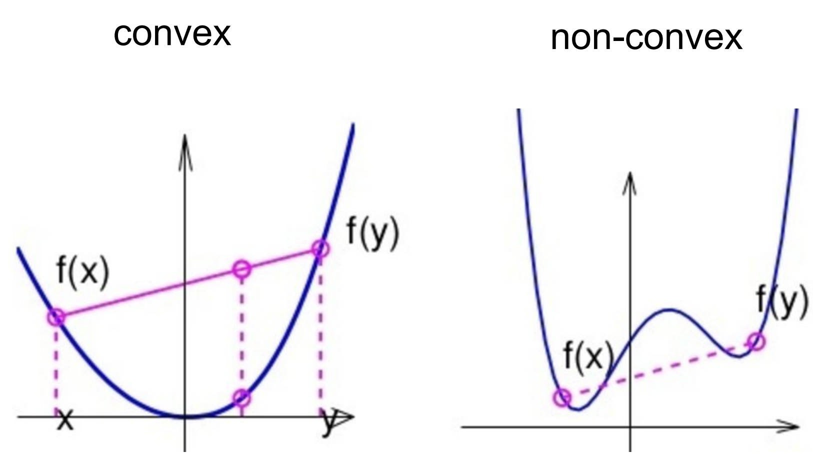


Optimization

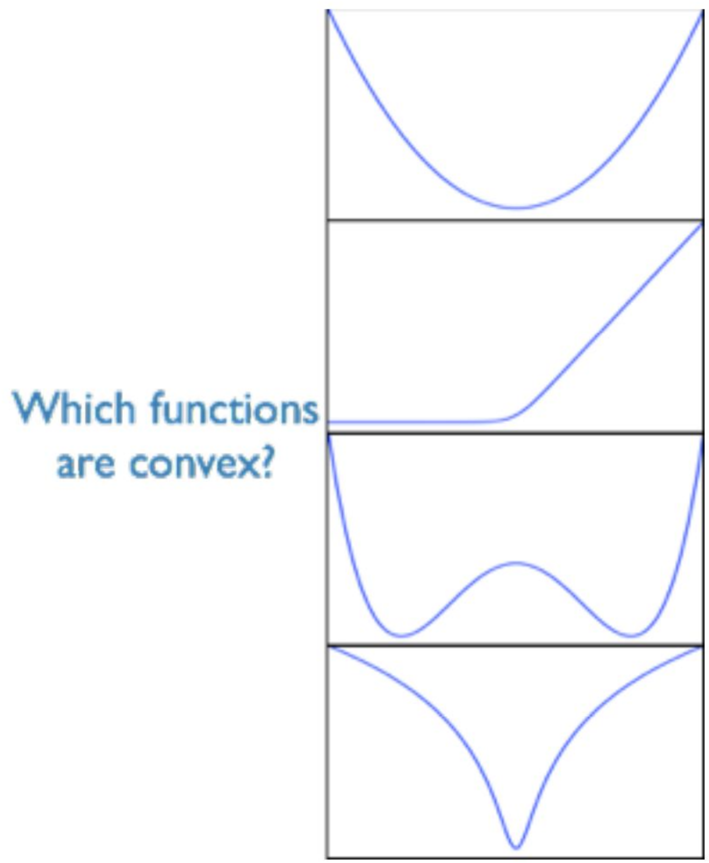
- Convex function:
  - The function f is convex iff:

$$\forall \alpha \in [0,1]$$

$$f(\alpha W_1 + (1-\alpha) W_2) \leq \alpha f(W_1) + (1-\alpha) f(W_2)$$



Optimization



Optimization

- In a convex function, any local minimum is automatically a global minimum

- Namely, we can start the optimization algorithm anywhere in the parameter space and converge to the same and optimal solution.

Example: linear regression

-
    - $\mathbf{X}$ is a 100 by 5 matrix (100 sets of 5-dimensional input data points)
    - $\mathbf{Y}$ is a 1 by 100 target vector (100 target labels for each x)
    - W is a 5 by 1 weights vector
    - b is a scalar
    - Model:

$$\hat{\mathbf{y}} = W^T\mathbf{x} + b = \sum_{i=1}^{5} W_i x_i + b$$

    - Given X and Y,
    - optimize for W and b under mean squared error over the 70 training examples:

$$\min_{W,b} \quad \frac{1}{70}\sum_{m=1}^{70} \|\hat{\mathbf{y}} - \mathbf{y}\|_2^2$$

- Example code at: http://www.psi.toronto.edu/~jimmy/ece521/mult.py
- Also download the two dataset files
  http://www.psi.toronto.edu/~jimmy/ece521/x.npy
  http://www.psi.toronto.edu/~jimmy/ece521/t2.npy

Example: linear regression

$$\hat{\mathbf{y}} = W^T\mathbf{x} + b = \sum_{i=1}^{5} W_i x_i + b$$

$$\min_{W,b} \quad \frac{1}{70}\sum_{m=1}^{70} \|\hat{\mathbf{y}} - \mathbf{y}\|_2^2$$

- The gradient of the parameters W_{ij} w.r.t. the loss function is: ??

Example: linear regression

$$\hat{\mathbf{y}} = W^T\mathbf{x} + b = \sum_{i=1}^{5} W_i x_i + b$$

$$\min_{W,b} \quad \frac{1}{70}\sum_{m=1}^{70} \|\hat{\mathbf{y}} - \mathbf{y}\|_2^2$$

- The gradient of the parameters W w.r.t. the loss function is:

$$\frac{\partial \mathcal{L}}{\partial W_{ij}} = \sum_m \sum_i (\hat{y}_j^{(m)} - y_j^{(m)}) x_i$$

Example: linear regression

$$\hat{\mathbf{y}} = W^T\mathbf{x} + b = \sum_{i=1}^{5} W_i x_i + b$$

$$\min_{W,b} \quad \frac{1}{70}\sum_{m=1}^{70} \|\hat{\mathbf{y}} - \mathbf{y}\|_2^2$$

- The gradient of the parameters W w.r.t. the loss function is:

$$\frac{\partial \mathcal{L}}{\partial W} = \sum_m (\hat{\mathbf{y}}^{(m)} - \mathbf{y}^{(m)})\mathbf{x}^T$$