Before midterm:

* Decision Tree
* KNN
* Evaluation
* Naïve Bayes
* Linear Regression/ Linear Logistic Regression
* Neural Network – back prop
* CNN

Outline (After midterm):

* RNN/LSTM/GRU/Attention/Autoencoder
* GAN
* Learning Theory
* Reinforcement learning
* Graphical model
* SVM
* Ensemble learning
* Feature selection
* PCA
* Active learning**：** <http://www.cs.columbia.edu/~djhsu/papers/hier.pdf>
* Semi-supervised learning
* Adversarial learning

**LSTM/GRU**

A picture containing text

Description automatically generatedA screenshot of a cell phone

Description automatically generated

**Encoder-decoder/Attention**

**Autoencoder & PCA**

Undercomplete linear autoencoder 🡺 PCA

* Undercomplete linear autoencoder: Minimizes (x’ - x)^2
* PCA: Maximizes variance

According to 勾股定理, these two objective functions **are equivalent.**

A close up of text on a white background

Description automatically generated

**Learning Theory:**

Rn(g): Empirical risk

G\_hat: empirical risk minimizer – g\_hat = argmin(Rn(g))

R(g): Risk, the risk of model g in real data distribution (But usually we don’t the real distribution, so we use empirical risk instead)

R(g\_hat): risk of best model from hypothesis space with given dataset

R(g\*): risk of best model from hypothesis space

R(g\_bayes): risk of the best model (may not be in the current hypothesis space) in the real data distribution

Estimation error: R(g\_hat) – R(g\*)

Approximation error: R(g\*) – R(g\_bayes)

We want to find the bound of estimation error – using PAC analysis

Hoeffding bound:

A close up of text on a whiteboard

Description automatically generated

**Shattering**

**VC – dimension**

**Graphical Model**

Assumes conditional independence

P(V) = , where V consists of many Vi

**Parameter learning**: using MLE to calculate all probabilities to make probability tables

Naïve Bayes – also conditional independence 🡪 graph will be from Y to all Xi

**Structure learning: (Chow-Liu Algorithm)**

Calculate MI:

A picture containing text, whiteboard

Description automatically generated

Find maximum weight spanning tree using greedy algorithm (edge weights are MI)

Arbitrarily pick a root and assign direction from root and update all directions on cascade

**Support Vector Machine**

RBF kernel

Mercer’s condition for kernels

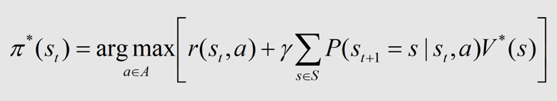
Soft-margin SVM:

Equivalent to using hinge loss

**Reinforcement learning:**

Bellman’s equation/ Using value iteration to find best strategy:

A close up of a logo

Description automatically generated

Value iteration for learning V\* (assumes we have a model of the world: i.e. know P(st | st-1, at-1))

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

Q-learning: (deterministic VS. non-deterministic)

* Exploitation: in order to learn about better alternatives, we shouldn’t always follow the current policy
* A picture containing object

  Description automatically generatedExploration: sometimes, we should select random actions

where c > 0 is a constant that determines how strongly selection favors actions with higher Q values

Using deep learning net to approximate Q-table:

Loss = pred\_Q(s,a) – [r+ gamma\*max(Q(s’,a’))] 🡺 prediction – calculation

**Ensemble methods:**

General Case/ Random Forest:

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

**Feature Selection:**

1. Select features that has greatest Mutual Info with y

2. Using LASSO regression – induces sparsity

3. greedy wrapper:

Initialize I as empty set

For j in all features:

Using features: [I union j] to produce a learner f(x)

Predict validation set with f(x) 🡺 get a risk

Select best j (the one that reduces risk the most), add j to I

Repeat until I reaches the threshold or the reduction of risk is too small

**A close up of a piece of paper

Description automatically generatedA close up of a logo

Description automatically generated**

**Evaluation:**

**Learning curve**: x – training set sample size, y – error on *test set*

**Confusion Matrix:** X – predicted**;** Y – actual

**Confusion Matrix in 2-class situation:**

TP rate (Recall) = TP/actual positive

FP rate = FP/actual negative

Precision = TP/predicted positive = TP/(TP+FP)

**ROC Curve:** X – FP rate, Y – TP rate

**Precision/Recall curve:** X – Recall, Y – precision

**A close up of a map

Description automatically generatedA close up of text on a white background

Description automatically generated**

A picture containing text

Description automatically generatedA screenshot of a cell phone

Description automatically generated**A screenshot of a cell phone

Description automatically generated**

**RBF kernel/ Mercer’s condition**

A close up of a person

Description automatically generatedA screenshot of a cell phone

Description automatically generated