Machine Learning Handbook

Xinhe Liu

2018-2-28

Contents

Part I

High-level Views

Chapter 1

Math Review

1.0.1 Basic Math

- Law of Sine, Law of Cosines

1.1 Calculus

1.1.1 Derivatives and Integration

- Derivatives: product, quotient, chain rule, $x^n, sinx, cosx, tanx, a^x, lnx$
- Limits
$$e^x = \lim_{n\to\infty}(1+\frac{x}{n})^n, \lim_{x\to 0}\frac{sinx}{x} = 1$$
- L'Hospital's Rule
- Maximum and Minimum, second derivative test (in multivariate case, Hessian Matrix is positive definitive indicates local minimum)
- Mean-Value Theorem

  Rolle, Cauchy
- Taylor's Theorem

  Maclaulin Seris: $sinx, cosx, e^x, \frac{1}{1-x}$

- Integration x^n
  Remann Integral
  Substitution Rule, Integration By Parts
- Fundamental Theorem of Calculus
  Lebnitz Rule for Integration
- Series, Indefinite Integral
  Convergence and Divergence
  Comparison Test, Limit Comparison Test, Ratio Test
- Multi-variate calculus
  Jacobian, Hessian Matrix
  Fubini's Rule
  Polar Coordinate ($\Delta A = rdrd\theta$), Spherical Coordinate
- Ordinary Derivative Equations
  Separation of Variable
  First Order O.D.E, integrating factor
  Second Order O.D.E, characteristic root
- Partial Derivative Equations

1.2 Linear Algebra

Concepts:

- scalar, vector, matrix, tensor(n-rank tensor, matrix is a rank 2 tensor)
- Gaussian Elimination, rank
- Invertible, Singular(non invertible, degenerate) matrices
- p-norm

$$|X|_p = (\sum_i |x_i|^p)^{\frac{1}{p}}$$

- inner product$< x_i, y_i >$, outer product
- orthogonaldimension, basis, orthogonal basis
- linear transformation $Ax = y$
- Determinant
- eigenvalue, eigenvector $Ax = \lambda x, det(A - \lambda I) = 0$ (transformation and speed)

  Diagonalizable matrix (dimension of eigen space, linearly independent eigen values)

  Diagonalization
- Positive Semidefinite/definite matrix

  $x^T Ax \geq 0$, eigen values non-negative, all upper left(or lower right) submatrices have positive determinants
- vector space, linear space(with summation, scalar production), inner product space(inner product space)
- QR Decomposition

$$Rx = Q^T b$$

- LU Decomposition and Cholesky Decomposition

$$A = LU$$

$$A = R^T R$$

  for s.p.d matrices
- SVD(Singular Value Decomposition)

$$X = UDV^T$$

1.2.1 Matrix Derivative

- $\frac{\partial y}{\partial x}$, $\frac{\partial \mathbf{y}}{\partial x}$, $\frac{\partial \mathbf{Y}}{\partial x}$ just list normal derivatives by column

$$a = \mathbf{XW}, \frac{\partial A}{\partial W} = \mathbf{X}^T$$

- $\frac{\partial y}{\mathbf{x}}$, $\frac{\partial y}{\mathbf{X}}$ list the result according to denominator
- Jacobian $(x_1, ..., x_n) \to (h_1, ...h_m)$

$$\frac{\partial \mathbf{h}}{\partial \mathbf{x}} = \begin{pmatrix} \frac{\partial h_1}{\partial x} & \cdots & \frac{\partial h_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_m}{\partial x_1} & \cdots & \frac{\partial h_m}{\partial x_n} \end{pmatrix}$$

- For matrix to matrix, vector to matrix, more than one dimension tensors as result (result must include all results and can be used with Chain Rule)
-

$$\frac{\partial \mathbf{a^T x}}{\mathbf{x}} = \frac{\partial \mathbf{xa}}{\mathbf{x}} = \mathbf{a}$$

$$\frac{\partial \mathbf{Ax}}{\mathbf{x}} = \mathbf{A}, \frac{\partial \mathbf{x^T Ax}}{\partial \mathbf{x}} = (\mathbf{A} + \mathbf{A^T})\mathbf{x}$$

$$\frac{\partial \mathbf{x^T Ax}}{\partial \mathbf{x} \partial \mathbf{x}} = 2\mathbf{A}$$

$$\frac{\partial (\mathbf{AX} + b)\mathbf{C}(\mathbf{DX} + e)}{\partial \mathbf{x}} = \mathbf{A^T C}(\mathbf{DX} + e) + \mathbf{D^T C^T}(\mathbf{AX} + b)$$

1.3 Important Inequalities

- Distance Inequality : all "distance measures" should satisfy

$$|\mathbf{x} + \mathbf{y}| \leq |\mathbf{x}| + |\mathbf{y}|$$

- Cauchy-Shwartz

$$|\mathbf{xy}| \leq |\mathbf{x}||\mathbf{y}|$$

- Jensen's Inequality
$$f((E(x)) \leq E(f(x))$$
for convex f
- Infinite Series
$$e^{-x^p} \leq \frac{c}{x^q}, \exists c, p, q, \forall x \geq a$$
-

1.4 Probability

Concepts:

- Classic Probability Model: Frequentist
- Bayesian Probability Theory
- Random variable, continuous RV, discrete RV, probability mass function, probability density function, cumulative density function
- Independence, Correlation
- Permutation, Combination, Binomial Theorem, Inclusion-Exclusion Principles
- expectation, moments, variance, covariance, correlation coefficient

  Consine similarity (correlation in Euclidean space)
$$cos\theta = \frac{\mathbf{A}^T\mathbf{B}}{||\mathbf{A}||\,||\mathbf{B}||}$$
- Moment Generating Functions

Advanced

- Measure based probability: $(\Omega, \mathcal{F}, P)$ Sample Space(collection of outcomes), Sigma-algebra(events), Probability measure (assigns probability to events)

  Probability Measure satisfies

Theorems:

- Law of Total Probability
- Bayes' Rule

$$P(H|D) = \frac{P(D|H)P(H)}{P(D)}$$

  $P(H)$-prior probability, $P(D|H)$-likelihood, $P(H|D)$-posterior probability,

Important Distributions:

1. Bernoulli distribution
2. Uniform Distributiondiscrete)
3. Binomial distribution(n,p)

$$P(X = k) = \binom{N}{k} p^k (1-p)^{(n-k)}$$

$$E(X) = np, Var(X) = np(1-p)$$

4. Geometric Distribution

$$P(x) = (1-p)^{x-1} p$$

$$E(X) = \frac{1}{p}, Var(X) = \frac{1-]}{p^2}$$

5. Poisson distribution

$$P(X = k) = \lambda^k \frac{e^{-\lambda}}{k!}$$

$$E(X) = \lambda, Var(X) = \lambda$$

6. Negative Binomial Distribution
7. Normal Distribution, See next chapter
8. Bernoulli Distribution

9. Uniform Distribution(continuous)
10. Exponential distribution

$$e^{-\frac{x}{\theta}}\theta$$

$$P(x > s + t|X > s) = P(x > t)$$

,

$$E(x) = \frac{1}{\lambda}, Var(X) = \frac{1}{\lambda^2}$$

11. Normal distribution
12. t-distribution
13. Beta Distribution (the conjugate prior probability distribution for the Bernoulli, binomial, negative binomial and geometric distributions.)
14. Gamma Distribution

    (The exponential distribution, Erlang distribution, and chi-squared distribution are special cases of the gamma distribution.)

    (the gamma distribution is the conjugate prior to many likelihood distributions: the Poisson, exponential, normal (with known mean), Pareto, gamma with known shape , inverse gamma with known shape parameter)

Moment Generating Functions:

1.5 Information Theory

Concepts:

- Information

$$h(A) = -log_2 p(A)$$

(bit)

- (Information Source) Entropy

$$H(X) = -\sum_{i=1}^{n} p(a_i) log_2 p(a_i) \leq log_2 n$$

  Maximize under equal probability

- Conditional Entropy

$$H(Y|X) = -\sum_{i=1}^{n} p(x_i) H(Y|X = x_i) = -\sum_{i=1}^{n} p(x_i) \sum_{j=1}^{n} p(y_j|x_i) log_2 p(y_j|x_i)$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{n} p(x_i, y_j) log_2 p(y_j|x_i)$$

- Mutual Information/Information Gain

$$I(X;Y) = H(Y) - H(Y|X)$$

- Kullback-Leibler Divergence (K-L) Divergence

$$D_{KL}(P||Q) = \sum_{i=1}^{n} p(x_i) log_2 \frac{p(x_i)}{q(x_i)} \neq D_{KL}(Q||P)$$

$$D_{KL}(f, \hat{f})) = \int_{-\infty}^{\infty} log(\frac{f_X(x)}{\hat{f(x)}}) f_X(x) dx$$

  K-L Divergence Measures the Distance of two distributions. The optimal encoding of information has the same bits as the entropy. Measures the extra bits if the real distribution is q rather than p. (Using P to approximate Q) K-L divergence plays an important role in both information theory and MLE theory. MLE $\hat{\theta}$ is actually finding the closest K-L Distance approximation of $f(x;\theta)$ to sample distribution.

Theorems:

- The Maximum Entropy Principle. Without extra assumption, max entropy/equal probability has the minimum prediction risk.

1.6 Optimization

1.6.1 Optimization Theory

- Objective function/Evaluation function, constrained/unconstrained optimizationFeasible Set, Optimal Solution, Optimal Value, Binding Constraints, Shadow Price, Infeasible Price, Infeasibility, Unboundedness
- Linear Programming
- Lagrange Multiplier

$$L(x, y, \lambda) = f(x, y) + \lambda \varphi(x, y)$$

- Convex Set, Convex Function $f : S \rightarrow R$ is convex if and only if $\bigtriangledown^2 f(\mathbf{x})$ is positive semidefinite
- Duality - the equivalent problem of the primal problem.

1.6.2 Optimization Methods

- Linear Search Method: Direction First, Step Size second
  - Gradient Descent: Batch Processing(Use all samples) vs Stochastic Gradient Descent(Use one sample)

$$\theta = \theta - \alpha \frac{\partial J}{\partial \theta}$$

  - Newton's Method: Use Curvature Information

$$\boldsymbol{\beta}^{t+1} = \boldsymbol{\beta}^{t} - \left(\frac{\partial^2 Loss(\boldsymbol{\beta})}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^T}\right)^{-1} \frac{\partial Loss(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}}$$

- Trust Region: Step first, direction second. Find optimal direction of second-order approximation. If the descent size is too small, make step size smaller.
- Heuristics Method
  - Genetic Algorithm

- Simulated Annealing
- Partical Swarming/Ant Colony Algorithm

Quadratic programming (QP)

- Sequential Minimal Optimization(SMO)

1.6.3 Optimization Algorithms in Machine Learning

Loss Function Entropy K-L Distance Regularization Methods EM Algorithm Gradient Descent Stochastic Gradient Descent Batch Gradient Descent Momentum AdaGrad Adam Backward Propagation Gradient Checkling

1.6.4 Optimization in Deep Learning

Optimization Algos used in Deep Learning Includes

- Mini-batch gradient descent: Use one batch (subset) of sample to compute the gradient each time. (one epoch) (one batch size =1, it is Stochastic gradient descent)
- Momentum Method: Smooth the gradient series with EWMA (Exponentially weighted averages

$$V_{dw} = \beta_1 V_{dw} + (1-\beta_1)dw, V_{dw}/ = (1-\beta_1^t)$$

$$V_{db} = \beta_1 V_{db} + (1-\beta_1)dw, V_{db}/ = (1-\beta_1^t)$$

- Root-Mean Square Prop (RMSProp)

$$S_{dw} = \beta S_{dw} + (1-\beta)dw^2$$

$$S_{db} = \beta S_{db} + (1-\beta)dw^2$$

$$w := w - \alpha \frac{dw}{\sqrt{sdw}}, b := b - \alpha \frac{db}{\sqrt{sdb}}$$

- Adam(Adaptive Moment Estimation) Algorithm L Combine RMSProp and Momentum

$$V_{dw} = \beta_1 V_{dw} + (1-\beta_1)dw, V_{dw}/ = (1-\beta_1^t)$$

$$V_{db} = \beta_1 V_{db} + (1-\beta_1)dw, V_{db}/ = (1-\beta_1^t)$$

$$S_{dw} = \beta_2 S_{dw} + (1-\beta_2)dw^2$$

$$S_{db} = \beta_2 S_{db} + (1-\beta_2)dw^2$$

$$w := w - \alpha\frac{v_{dw}}{\sqrt{sdw}+\epsilon}, b := b - \alpha\frac{v_{db}}{\sqrt{sdb}+\epsilon}$$

$$w := w - \alpha\frac{dw}{\sqrt{sdb}+\epsilon}, b := b - \alpha\frac{db}{\sqrt{sdb}+\epsilon}$$

- Learning Rate Decay

$$\alpha = \frac{1}{1 + \text{decay rate} \times \text{epoch num}}$$

Beam Search: with hyper-parameter Beam-width(B): keep the top B answers in each training step (a heuristic method that generalized Greedy)

1.6.5 Hyperparameter Tuning Methods

- Grid method

- Batch Normalization

$$z_{norm} = \frac{z^{(i)} - \mu}{\sqrt{\sigma^2 + \epsilon}}$$

$$\tilde{z} = \gamma z_{norm} + \beta$$

Can speed up learning and add some noise to avoid overfitting. (Similar to dropout). In test time, usually use the EWMA across mini-batches on the mean and variance series to normalize the use trained β, γ to transform.

1.7 Formal Logic

Concepts

- Generative Expert System: Rule+Facts+Deduction Engine
- Godel's incompleteness theorems

Chapter 2

Statistics

2.1 Concepts

2.1.1 Basic

- parameter(constant for probability model), statistic (model of sample data), estimator, data, sample, population
- point estimation, interval estimation, Confidence Interval($P(L \leq \theta \leq U)$, notice: θ is not random, L, U is random! (We repeat constructing confidence interval a n times, α percent of the times, it will contain $theta$.

2.1.2 Estimator and Estimation

Method of Moments

Estimate $E(X^k)$ based on Law Of Large Numbers
If We have p parameters, we can use p moments to form a system of equations to solve $\theta_1, ...\theta_p$

$$\sum_{i=1}^{n} X_i^j = E(X^j)$$

, for j = 1,...,p Properties

- Almost surely exist
- Consistent
- Asymptotically Normal (variance decrease at $\frac{1}{\sqrt{n}}$)

Maximum Likelihood Estimation

Multiply p.m.f/p.d.f since every sample is independent. Maximize the likelihood of finding samples.
If $X_1, ...X_n \overset{i.i.d}{\sim} f_x(x, \theta)$,

$$l(\theta) = \prod_{i=1}^{n} f_{X_i}(x_i; \theta), L(\theta) = logl(\theta)$$

$$\hat{\theta}_{MLE} = argmax_{\theta} f_x(x; \theta) = argmax_{\theta} L(\theta)$$

Analytical or Numerically solved.

$$\frac{\partial}{\partial \theta}[logL(\theta)] = 0, \frac{\partial^2}{\partial \theta^2}[logL(\theta)] < 0$$

, for multiple parameters, we need the Hessian matrix to be negative definite $x^t H x < 0, \forall x$

Properties of MLE

1. Invariance $\hat{\theta}$ is MLE of θ, then $g(\hat{\theta})$ is MLE of $g(\theta)$
2. Consistency
$$P(\hat{\theta} - \theta) \to 0$$
as $n \to 0, \forall \epsilon > 0$ Under the conditions
   (a) $X_1, ...X_n \overset{i.i.d}{\sim} f_x(x|\theta)$
   (b) parameters are identifiable, $\theta \neq \theta', f_x(x|\theta) \neq f_x(x|\theta')$
   (c) densities $f_x(x|\theta)$ has common support(set of x with positive density/probability), $f_x(x|\theta)$ is differentiable at θ

(d) parameter space Ω contains open set ω where true θ_0 is an interior point

3. Asymptotic Normality

$$\sqrt{n}(\hat{\theta}_{MLE} - \theta_0) \to N(0, I^{-1}(\theta_0))$$

$$I(\theta_0) = E(-(\frac{\partial}{\partial\theta}[logf(x,\theta)])^2) = E(-\frac{\partial^2}{\partial\theta^2}[logf(x,\theta)])$$

called the **Fisher Information**

$$\hat{\theta}_{MLE} \approx N(\theta_0, \frac{1}{nI(\theta_0)})$$

$$nI(\theta_0) = E(-\frac{\partial^2}{\partial\theta^2}logL(\theta))$$

- So the Variance of MLE($1/E(-\frac{\partial^2}{\partial\theta^2}logL(\theta))$) is the reciprocal of amount of curvature at MLE. Usually, We can just use the *observed Fisher Information* (curvature near $\hat{\theta_{MLE}}$) instead. $(I(\hat{\theta_{MLE}}))$
$\frac{1}{nI(\theta_0)}$ is called Cramer-Rao Lower Bound.
Under Multi-dimensional Case,

$$I(\theta_0)_{ij} == E(-\frac{\partial^2}{\partial\theta_i\partial\theta_j}[logf(x,\theta)])$$

$Hessian \approx nI(\theta_0)$ $Hessian^{-1} \approx nI(\theta_0)$ when we use numerical approach.

- Under the above four conditions plus

  (a) $\forall x \in \chi$, $f_x(x|\theta)$ is three times differentiable with respect to θ, and third derivative is continuous at θ, and $\int f_x(x|\theta)dx$ can be differentiated three times under integral sign

  (b) $\forall\theta \in \Omega, \exists c, M(x)$ (both depends on θ_0) such that

$$\frac{\partial^3}{\partial\theta^3}[logf(x,\theta)] \leq M(x), \forall x \in \chi, \theta_0 - c < \theta < \theta_0 + c, E_{\theta_0}[M(x)] < \infty$$

Δ -Method $g(\hat{\theta}_{MLE})$ is approximately

$$N(g(\theta), (g'(\theta))^2 \frac{1}{nI(\theta)})$$

if asymptotic normality is satisfied.
In Multivariate Case:

$$\hat{\theta} \sim N(\theta, \Sigma/n), \theta, \hat{\theta} \in R^p$$

$$g : R^p \to R^m$$

$$g(\hat{\theta}) \sim N(g(\theta), G\Sigma G^T/n)$$

$$G = \begin{pmatrix} \frac{\partial g_1(\theta)}{\partial \theta_1} & \cdots & \frac{\partial g_1(\theta)}{\partial \theta_p} \\ \vdots & \ddots & \vdots \\ \frac{\partial g_m(\theta)}{\partial \theta_1} & \cdots & \frac{\partial g_m(\theta)}{\partial \theta_p} \end{pmatrix}$$

Estimation criteria

- Unbiased $E(\hat{\theta}) = \theta$
- Minimum Variance (MVUE, minimum variance unbiased estimator) $Var(\hat{\theta}) < Var(\theta')$
- Efficient
- Coherent

2.1.3 Model Selection

AIC - Akaike Information Criterion

K-L Distance

$$D_{KL}(f, \hat{f})) = \int_{-\infty}^{\infty} log(\frac{f_X(x)}{f(\hat{x})}) f_X(x) dx$$

$$= const + \frac{1}{2} \int (-2log\hat{f}(x)) f(x) dx = const + AIC$$

$$A(f, \hat{f}) = -2logL(\theta) + 2p(\frac{n}{n-p+1})$$

2.1.4 Hypothesis Testing

- Basic Logic: a conditional statement is equivalent to its contrapositive statement

$$A \to \neg(\cup B_i) \Leftrightarrow \cup B_i \to \neg A$$

notice,

$$\neg(\cup B_i) = \cap(\neg B_i)$$

- Hypothese, Test Statistic(T), Rejection Region,One-tail Test, Two-tail Test

- Significance α

  Power $1 - \beta$(Pr(Reject H_0 — H_1 is True))

- p-value (the probability to observe as or more extreme result than current evidence under H_0)

- Type-I Error(wrongly reject, false reject,$1 - \alpha$)

  Type-II error(wrongly accept, failed to reject)

Hypothesis Testings

Based on the distribution of $\hat{\theta}$

- Wald Test

$$T = \frac{\hat{\theta} - \theta_0}{Se(\hat{\theta})}$$

$$\hat{\theta}_{MLE} \approx N(\theta_0, \frac{1}{nI(\theta_0)})$$

$$T = \frac{\hat{\theta} - \theta_0}{\sqrt{\frac{1}{nI(\theta_0)})}}$$

- Likelihood Ratio Test

- Score Test

- Rank based tests - used to test mean, mean-like statistics, not as efficient as Computational based test

  Wilcoxon signed-rank test

  Mann-Whitney U test

Multivariate Testing

- Pearson's Chi-Square Test for Independence

$$U = \frac{X_{ij} - E_{ij}}{E_{ij}}$$

$$E_{ij} = \frac{X_i X_j}{n}$$

- Test Discrete Random Variable vs. Continuous Random Variable
  - Test
    $$sup|\hat{F}_1 - \hat{F}_2|$$
    Y is independent of Z if CDF is the same
  - Do with regression (with categorical parameter) - test β

- Wald Test*

Computation-based hypothesis testing approach

- Permutation tests:
  Test $X_1, ..X_n \sim F, Y_1, ...Y_n \sim G, if F = G$. Use
  $T = Mean(X_i) - Mean(Y_i)$, each time scramble X and V labels and should not not change the distributions of vectors $X_1, ...X_n, Y_1..., Y_n$

- Bootstrapping:
  $X_1, ...X_n \sim F$ with $T = T(X_1, ..., X_n)$, to get the distribution of T, **sample with replacement.** The belief is $(\hat{\theta} - \theta)$ should behave the same as $(\theta * -th\hat{e}ta)$. The first quantity can be treated like a pivot. (use $(\theta *_1 - \hat{\theta}_1), ...(\theta *_n - \hat{\theta}_n)$ to test.

2.1.5 Multiple Testing Examples

- Family-wise Error Rate(FWER) the probability of rejecting at least one of at least one null hypothesis
  Under independence, the probability of making mistake when all null are true: P(any type I mistake) = 1-P(no type I mistake for all) = $1-(1-\alpha)^M = \beta$)
- Bonferroni correction, assuming independence
  $$P(\bigcup_{i=1}^{n} \text{typeI mistake}) \leq \sum_{i=1}^{n} P(\text{typeI mistake}) \leq M\alpha$$
  ,control at $\alpha = \frac{\alpha}{M}$
  α being to small will impact power of the individual tests!
- False Discovery Rate(FDR): bound the fraction of type-I errors. R be the total number of hypotheses rejected. V be the number of rejected hypotheses that were actually null. Let FDR = V/max(R,1), control $E(FDR) \leq \alpha$.

2.1.6 A/B Testing

Application of Hypothesis Testing in industry

1. define metrics

   Direct metrics vs Compound Metrics

   High-level (Unified) Metrics vs Low-level metrics: level of aggregation

   Primary Metric vs Secondary Metrics
   (primary metrics - usually explains metric change)
2. Compare metrics
3. Experiment Design

   Experiment meaning you have to modify the user/object

   Experiment group vs control group/ treatment group

   Confounding Variables and control of Confounding Variables

   Randomized experiment

(a) approximation for treatment effect

(b) random sampling -hashing cookie id

Other sampling methods

- Stratified Sampling
- Cluster Sampling

(c) randomization - flipping a coin

(d) novelty effect - the difference between groups are too large

Techniques

- Matching

  one-to-one match on selected segmentations

  set-to-set matching with frequency of selected factors

  propensity score matching(propensity score is estimated any classification model, typically logistic regression)

- Segmentation divide objects to sub-groups

  deal with sub-group effect

  if you suspect the randomization is not perfect

- Advanced control group settings

  Holdout group (group that you don't expose to new environment) to understand the effect of additional environment changes

  Double control group (AAB test, AA test) to detect potential bias of the testing

- Sample Size (7% on one day or 1% over 7 days)
  - correlations between days (autocorrelation)
  - day effects
  - allow to measure long term effects

4. Hypothesis Testing based on Experiment

   Usually need to approximate discontinuous data using continuous distribution hypothesis test(t-test, Z-test). Convert multi-class metrics to binary metrics (ask yes or no questions)

2.2 Theorems

- Law of Large Number
- Central Limit Theorem

  Casual definition of C.L.T

  Regularization Condition of C.L.T. (eg. Cauchy has infinite variance)

- Bias-Variance decomposition (error = bias + variance + noise) under MSE

  $\sigma_x^2 = -\mathbb{E}((Y - \mathbb{E}(Y|X))^2|X), \mathbb{E}(Y|X) = f(X), \mathbb{E}[(Y - f(x))|X] = 0$

  $$\mathbb{E}\, L(\mu(X)) = \mathbb{E}[(Y - \hat{\mu}(X))^2]$$

  $$= \mathbb{E}[(Y - f(x) + f(x) - \hat{\mu}(X))^2]$$

  $$= \mathbb{E}[(Y - f(x))^2] + 2\,\mathbb{E}[(Y - f(x))(f(x) - \hat{\mu}(X))] + \mathbb{E}[(f(x) - \hat{\mu}(X))^2]$$

  $$= \mathbb{E}[(Y - f(x))^2] + \mathbb{E}(f(x) - \hat{\mu}(X))^2 + 2(f(x) - \mathbb{E}(\hat{\mu}(X)))\,\mathbb{E}[(Y - f(x))]$$

  $$= \sigma_x^2 + \mathbb{E}(f(x) - \hat{\mu}(X))^2 + 0$$

  $$= \sigma_x^2 + \mathbb{E}[\mathbb{E}(f(x) - \hat{\mu}(X))^2|X]$$

  (conditional expectation over X)

  $\mathbb{E}((f(x) - \hat{\mu}(X))^2|X) = \mathbb{E}((f(x) - \mathbb{E}(\hat{\mu}(X)) + \mathbb{E}(\hat{\mu}(X)) - \hat{\mu}(X))^2|X)$

  $= \mathbb{E}((f(x) - \mathbb{E}(\hat{\mu}(X)))^2|X) + \mathbb{E}((\mathbb{E}(\hat{\mu}(X) - \hat{\mu}(X))^2|X) + 2((f(x) - \mathbb{E}(\hat{\mu}(X))\,\mathbb{E}((\mathbb{E}(\hat{\mu}(X) -$

  $= (f(x) - \mathbb{E}(\hat{\mu}(X)))^2 + \mathbb{E}((\mathbb{E}(\hat{\mu}(X) - \hat{\mu}(X))^2|X) + 2((f(x) - \mathbb{E}(\hat{\mu}(X)) \times 0$

  $$= (f(x) - \mathbb{E}(\hat{\mu}(X)))^2 + Var(\hat{\mu}(X))$$

  $$\Rightarrow$$

  $$\mathbb{E}\, L(\mu(X)) = \sigma_x^2 + Bias(\hat{\mu}(X))^2 + Var(\hat{\mu}(X))$$

2.3 Important Distributions

1. Normal Distribution, $X_1, ...X_n \sim N(\mu, \sigma^2)$ then
   (a) $\bar{X}$ and s^2 are independent
   (b) $\frac{\bar{X}-\mu}{\sigma/\sqrt{n}} \sim N(0,1)$
   (c) $\frac{(n-1)s^2}{\sigma^2} \sim \chi^2_{n-1}$
   (d) $\frac{\bar{X}-\mu}{s/\sqrt{n}} = \frac{\frac{\bar{X}-\mu}{\sigma/\sqrt{n}}}{\frac{(n-1)s^2}{\sigma^2}\frac{1}{\sqrt{n-1}}} \sim t_{n-1}$

2. Multivariate normal distribution

$$f_x(x) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} exp(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu))$$

   (a) $X_1, ...X_n$ normal $\Leftarrow (X_1, ...X_n)$ is multivariate normal. (Not equivalent)
   (b) $E(X) = \mu, Var(X) = \Sigma$
   (c) Linear transformations $AX + b \sim N(A\mu + b, A\Sigma A^T)$ remain multivariate normal
   (d) Marginals are multivariate normal, each sub-vector is multivariate normal, the parameters are just sub-matrices.
   (e) All conditionals are multivariate normal

3. t-distribution: like normal distribution, but heavier tails
   (a) $Z \sim N(0,1), Y \sim \chi^2_\nu$, Z, Y independent,

$$X = Z/\sqrt{Y/\nu} \sim t_\nu$$

   (b) pdf has polynomial tails (decays much slower than exponential ones)
   (c) $\nu = 1$, it is the **Cauchy Distribution**, with very heavy tails (no expectation)
   (d) The MCF not exist. $E(|X|^k) < \infty$ for $k < \nu$, $E(|X|^k) = \infty$ for $k > \nu$
   (e) $X \sim t_\nu, E(X) = 0, Var(X) = \frac{\nu}{\nu-2}$

$$f_X(x) = \frac{1}{\pi(1+x^2)}$$

4. χ^2 distribution

$$f_x(x) = \frac{1}{(2^{k/2}\Gamma(k/2)}x^{\frac{k}{2}-1}e^{-\frac{x}{2}}, x \in [0,\infty) \sim Gamma(\frac{k}{2},\frac{1}{2})$$

   (a) $E(X) = k, Var(X) = 2k, M_X(t) = (\frac{1}{1-2^t})^{k/2}$
   (b) $X \sim N(0,1) \Rightarrow X^2 \sim \chi^2$, $X_1,...X_n \sim N(0,1)i.i.d \Rightarrow \sum X_i^2 \sim \chi^2$,

$$f_X(x) = \frac{1}{\pi(1+x^2)}$$

5. F-Distribution

More Generalized Distributions

1. Generalized Error Distribution (symmetric)
2. Non-standard t-distribution (shift and scaling, heavy tailed, symmetric)
3. Theodossious skewed t-distribution
4. Theodossious skewed t-distribution plus shift

2.4 Practice/Examples

1. sample mean($\bar{X}$) is unbiased. Sample variance ($\frac{1}{n-1}\sum_{i=1}^n x_i^n$) is unbiased. But sample std is not unbiased. $SE(\bar{X}) = \frac{\sigma^2}{n}$
2. $\hat{Cov}(X.Y) = \frac{1}{n-1}\sum_{i=1}^n (X_i - \bar{X})(X_i - \bar{Y})$ is unbiased
3. Distributions with Expectation not exist? (Cauchy)
4. Common Confidence Intervals:
   μ: $P(-t_{\alpha/2,n-1} \leq \frac{\bar{X}-\mu}{s/\sqrt{n}} \leq t_{\alpha/2,n-1}) = 1-\alpha$,
   σ: $P(a \leq \frac{(n-1)s^2}{\sigma^2} \leq b) = 1-\alpha$
5. Solve MLE/MOM for beta, exponential ($n/\sum X_i$, normal

6. * prove Asymptotic Normality of MLE(hint: using Taylor Expansion for $\theta, \hat{\theta}$)

7. * Use t^{th} quantile to approximate c.d.f, what's the distribution? ($Y_n = \frac{1}{n}\sum I(X_i < x)$, a Bernoulli distribution with $p = F_x(x)$, $\sqrt{n}[Y_n(x) - F_x(x)] \sim N(0, F(x)(1-F(x))$.

8. $X_1, ...X_n \sim Binomial(n, p)$, What's the MLE for p and Fisher Information? ($\hat{p} = \frac{x_i}{n}, I(p) = 1/p(1-p), var(p) = \frac{p(1-p)}{n}$)

9. $(x_i, y_i) \sim N(\mu_i, \sigma^2)$, find MLE for σ ($\frac{1}{4N}\sum(x_i - y_i)$)

10. How can you get N(0,1) random variables from U[0,1]? (Method1: Inverse Transformation, Method2; Use $SumZ_i^2 \sim \chi_k^2, k = 2, F^{-1}(u) = -2log(1-u)$, $R^2 \sim \chi^2, Z_1 = Rcos\theta, Z_2 = Rsin\theta, \theta \in [0, 2\pi]$

11. (Permutation test) how can you test $X_1, ..., X_n \sim F$, how can you test if F is symmetric? (Multiply -1 on all two form two sample groups)

12. Draw a bootstrap sample, what fraction of original data points appear in this sample on average?
Define I be the indicator is it is in the sample.
$E(\frac{1}{n}\sum I_i = E(I_i) = P(\text{ith point shows up}) = 1 - (1 - \frac{1}{n})^n$

13. Explain A/B Testing on 1.) Binary Data (click rate) 2.) Continuous Data (Conversion Rate)

form statistics

$$Z = \frac{\hat{p}_0 - p_0}{\sqrt{p_0(1-p_0)/n}}$$

$$Z = \frac{\hat{p}_1 - \hat{p}_2}{\sqrt{\hat{p}(1-\hat{p})(\frac{1}{n_1} + \frac{1}{n_1})}}, \hat{p} = \frac{n_1\hat{p}_1 + n_2\hat{p}_2}{n_1 + n_2}$$

$$\text{Z or t} = \frac{\hat{\mu} - \mu_0}{\sqrt{s_n/n}}$$

$$\text{Z or t} = \frac{\hat{\mu_1} - \hat{\mu_2}}{\sqrt{\hat{\sigma_1}/n_1 + \hat{\sigma_2}/n_2}}$$

$$(df = \frac{(\hat{\sigma_1}/n_1 + \hat{\sigma_2}/n_2)^2}{\frac{(\hat{\sigma_1}/n_1)^2}{n_1 - 1} + \frac{(\hat{\sigma_2}/n_2)^2}{n_2 - 1}})$$

(t is used only you have normal distribution assumptions)

Chapter 3

Computational Learning Theory

Chapter 4

Model Evaluation and Model Selection

- Hold-out: Separate to training/test(dev) set.
- Sampling Methods: Important for holding out. eg. Stratified Sampling
- Cross-Validation: Leave-One-Out and k-fold
- Bootstrapping: with m sampling with replacement:

$$\lim_{m \to \infty}(1 - \frac{1}{m}) = \frac{1}{e} \approx 0.368$$

  Use $D \setminus D'$ as testing set

- Hypothesis Test for Cross Validation
  - Binomial test generalized error rate for one CV

$$P(\hat{\epsilon}, \epsilon) = \binom{m}{\hat{\epsilon}m} \epsilon^{\hat{\epsilon}m}(1-\epsilon)^{m-\hat{\epsilon}m}$$

  - t test for multiple CVs

$$\mu = \frac{1}{k}\sum \hat{\epsilon}_i, \sigma = \frac{1}{k-1}\sum(\hat{\epsilon}_i - \mu)^2$$

– Paired t-test for two Classifiers A and B (Permutation test or normal t test on $|\frac{\sqrt{k}\mu}{\sigma}|$
– McNemar Test (χ^2 test)
– Friedman Test and Nemenyi Post-Test (On MUltiple Learners)

4.1 Performance Metrics

Supervised Learning-Regression and Classification

- Confusion Matrix

		True condition				
	Total population	Condition positive	Condition negative	Prevalence $= \frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$	Accuracy (ACC) $= \frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$	
Predicted condition	Predicted condition positive	**True positive**, Power	**False positive**, Type I error	Positive predictive value (PPV), Precision $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Predicted condition positive}}$	False discovery rate (FDR) $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Predicted condition positive}}$	
	Predicted condition negative	**False negative**, Type II error	**True negative**	False omission rate (FOR) $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Predicted condition negative}}$	Negative predictive value (NPV) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Predicted condition negative}}$	
		True positive rate (TPR), Recall, Sensitivity, probability of detection $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$	Positive likelihood ratio (LR+) $= \frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) $= \frac{\text{LR+}}{\text{LR−}}$	F_1 score $= 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$
		False negative rate (FNR), Miss rate $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$	Specificity (SPC), Selectivity, True negative rate (TNR) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$	Negative likelihood ratio (LR−) $= \frac{\text{FNR}}{\text{TNR}}$		

– Accuracy = $\int_{x \in D} \mathbb{I}(f(x) \neq y)p(x)dx$
– Precision or Positive Predictive Value(PPV)

$$Precision = \frac{T}{TP + FP}$$

(Analogy to Type I error when Positive means rejecting, predict when you are very certain H_0)
– Recall, Sensitivity, or True Negative Rate(TNR)

$$Recall = \text{True Positive Rate} = \frac{TP}{TP + FN} = 1 - FPR$$

(Analogy to Type II error)

- False Positive Rate or Fall-out

$$FPR = \frac{FP}{FP + TN}$$

- P-R Curve: Precision vs Recall (like Type I error, Type II error trade-off)
- $F - \beta$ Measure

$$\frac{1}{F_\beta} = \frac{1}{1+\beta^2}(\frac{1}{P} + \frac{\beta^2}{R})$$

$$F_\beta = \frac{(1+\beta^2) \times P \times R}{(\beta^2 \times P) + R}$$

- Specificity = 1- FPR
- ROC (Receiver Operating Characteristic) Curve

$$AUC = \int_{-\infty}^{+\infty} TPR(t)(FPR(t))'dt$$

$$= \int_{-\infty}^{+\infty} \int_{t}^{+\infty} f_1(x)dxf_0(t)dt$$

$$= \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \mathbb{1}_{x>T} f_1(x)f_0(t)dxdt$$

$$= \mathbb{P}(S_1 > S_0) = 1 - Loss_{rank}$$

f are densities for 0,1 class data

- Cost-Sensitive Loss: With unequal loss for FP and FN
- Cost Curve : Used to measure Cost-sensitive error rate: Use P(+)cost as horizontal and normalized cost as vertical.

Model Evaluation Performance Metrics A/B Test Bias, variance, Overfitting and Underfitting Hyperparameters Selection

Chapter 5

Feature Engineering

5.1 Data Wrangling

Some examples normalization won't help

- Information Gain

5.1.1 Transformation

Basic Transformations

- Box-Cox power Transformation -useful when response is strictly postitive

$$y = \begin{cases} \frac{y^\lambda - 1}{\lambda}, \text{ if } \lambda \neq 0 \\ log(y), \text{ if } \lambda = 0 \end{cases}$$

  λ could be selected via MLE

- Yeo-Johnson Transformation

$$y = \begin{cases} \frac{(y+1)^{\lambda}-1}{\lambda}, \text{ if } \lambda \neq 0, y \geq 0 \\ log(y+1), \text{ if } \lambda = 0 \text{ if } \lambda = 0, y \geq 0 \\ \frac{(-y+1)^{2-\lambda}-1}{\lambda}, \text{ if } \lambda \neq 2, y < 0 \\ log(-y+1), \text{ if } \lambda = 0 \text{ if } \lambda = 20, y < 0 \end{cases}$$

  Optimal Transformation can be found with the MLE Method

Feature Engineering

5.2 Discretization and Normalization

5.2.1 Normalization

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$
$$X = z - score = X_{norm} = \frac{X - \mu}{\sigma}$$

We Need Normalization because

- Data Interpretation
- Optimization - speed up gradient descent to run on more "round" shapes

5.2.2 Discrete(Categorical) Features

- ordinal encoding: deal with ordinal (can be compared). eg. Linear Regression
- one-hot encoding: code 1 in 1 positions and 0 in others. eg. words. Need to consider

  storage or large sparse matrix

  Dimension reduction and feature selection. eg. "Curse of Dimension" in K-means. Logistic regression

- Binary encoding: Give IDs to categories and use the binary codes. (eg. Blood type)
- Helmert Contrast, Sum Contrast, Polynomial Contrast, Backward Difference Contract

5.3 Feature Combination

Use the combination as a higher level feature. This could cause dimension increases. eg. for m categories in feature x, n categories in feature y. we have $m \times n$ w:

$$Y = sigmoid(\sum_i \sum_j w_{ij} < x_i, x_j >$$

one way to implement is to have k (k ¡¡ m, n) dimension representation of x, y.

We can use learning algorithms to find how to combine features. eg. Use a Tree model:

5.4 Feature Selection

5.5 Text Features

5.5.1 Text Representation

Feature representation used in Natural Language Processing (NLP)

- bag-of-words model: ignore sequence, view articles as a bag of words. (a long vector). weight in the vector can reflect in the importance (to the topic of the article)
- TF-IDF: Term Frequency-Inverse Document Frequency = TF(t,d) $\times$

IDF(t,d). TF(t,d) is the word t frequency in document d.

$$IDF = log\frac{\text{number of articles}}{\text{articles with word t} + 1}$$

as a measure of importance

- N-gram: use n words appearing together as a feature rather than each word
- word-Stemming: convert word to word stems to normalize
- Topic Model: Find theme and theme distributions on documents.

5.5.2 Word Embedding

- One-hot representation(o): 1 entry for the position in dictionary, 0 all other entires
- Word embedding (E) is a matrix representation with columns words, row features. $E \cdot o_j = e_j$
- t-SNE algorithm (a non-linear dimension reduction technique) to visualize word embeddings
- Indeed a transfer learning technique

  Learn word embeddings from large text corpus

  Transfer embeddings to new task with smaller training set

  (optional) continue to finetune the embedding on new training set
- Property :

$$e_{man} - e_{woman} \approx e_{king} - e_w$$

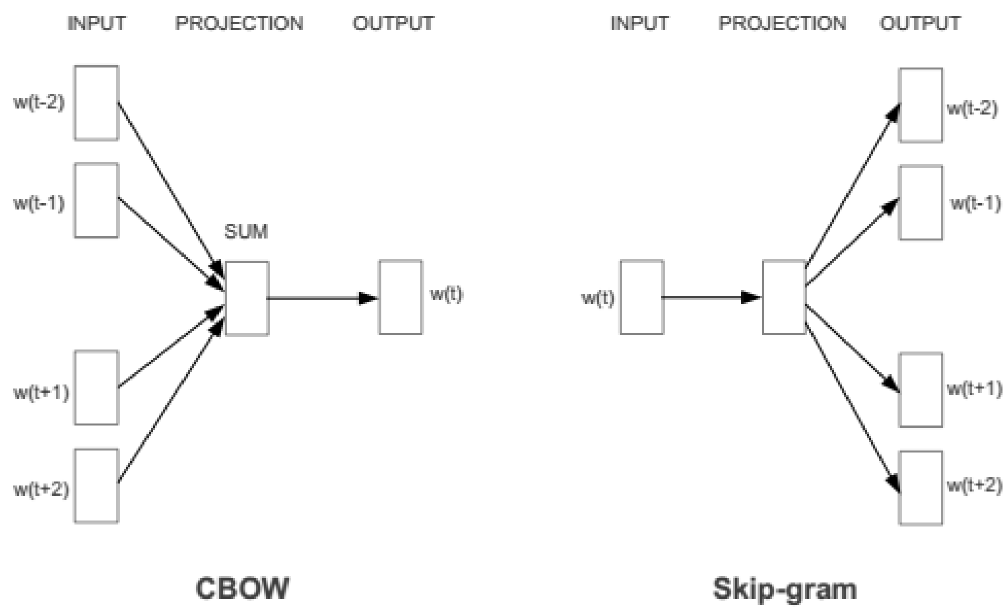
$$\hat{w} = \arg\max_w (e_{man} - e_{king} + e_{woman})$$

- Learning the word embedding: train the previous N words(or, words in the same sentence) embedding E with a neural network (activation + Softmax) ($w^{[1]}, b^{[1]}, w^{[2]}, b^{[2]}$)
- Language model: input the context (words before, after) to learn a word

- Debias word-embeddings: find the bias-direction (eg. gender bias would be the direction from man to woman, father to mother - the natural biased words) and non-bias direction : the midpoint and orthogonal direction to these word. Then neutralize other words should not be biased to that axis.

word2Vec and GloVe

- Two different architectures. CBOW (Continues Bag of Words) - use context to predict current word and Skip-gram - use current word to predict context



it is a generative model to find the weights in the softmax to maximize the likelihood of all words. The embeddings (word vec) is in the hidden layer.

- **Word2Vec** algorithm: for example, skip-gram algorithm. Train a word embedding: one word with fixed distance(eg. 10 words away) as context to predict a target.

$$o_c \rightarrow E \rightarrow e_c \rightarrow softmax \rightarrow \hat{y}$$

$$p(t|c) = \frac{e^{\theta_t e_c}}{\sum e^{\theta_j e_c}}$$

$$L(\hat{y}, y) = -\sum y_i log \hat{y}_i$$

y is one-hot representation (0,0,1,0)

- training word2vec: Could be very slow, because need to loop over all words each time. Need to improve efficiency by **Hierarchical softmax classifier.** Or use **negative sampling** - for each context-target pair, generate k negative examples on purpose. Turn to N (N is the number of such pairs) binary classification problems

  $$p(y = 1|c, t) = \sigma(\theta_t^T e_c)$$

  (Turnning multiclass classification to binary classification) usually use $p(w_i) = \frac{f(w_i)}{\sum f(w_i)}$ as sample probability

- GloVe algorithm (global vectors of word representation):

  $$\sum_i \sum_j f(x_{ij}(\theta_i^T e_j + b_i + b_j - logX_{ij})^2$$

  note, $f0 = 0$, f is the weighting scheme. θ_i, e_j are symmetric

Chapter 6

Sampling

Part II

Supervised Learning

Chapter 7

Regression

7.1 Overview

7.1.1 Type of Models

All Basic Models begins with **Linear Regression** Because

- Linear relationship is the simplest relationship other than constant relationship or "null" model (average)
- It's a global model
- Data Invariance: Simple linear model don't do any pre-processing or transformation on the covariants.
- Very Explainable, limited interpretation power.

So, the alternation/improvements also focuses on these aspects

- Nonlinear features-Introduction of basis function
  - Polynomial Regression
  - Spline Models(eg. Cubic Spline, Smoothing Spline)

- Nonlinear parameters: Parameters Self-adjusting.(activation function is an example of basis function as well)
  - Neutral-Network
- global nonlinear: global nonlinear on both parameters and features achieved by linkage function, extends regression models to classification.
  - Generalized Linear Model
- Change the global model to a local model
  - Local Regression (Regression + KNN)
  - Nonparametric Regression
  - Kernel Function
  - Distance Based Learning
- Data Preprocessing (Transformation) and Dimension Reduction
  - PCA
  - LDA
  - Manifold Learning
- Improve Generalization Capability from outside (not from inside the model)
  - Regularization Methods(eg. Ridge, Lasso)
  - Ensemble Learning(Stacking, Aggregating): Random Forest, Boosting(GBDT), Deep Learning...

7.1.2 The Key Questions

- What assumptions are the model making
- How will we access the validity of those assumptions
- How can we be confident about out-of-sample fitting (overfitting problem)
- How do we make predictions and quantify the uncertainty in models?

7.2 Linear Regression

Common Terms

1. Independent Variable, Features, Covariates, Predictors
2. Dependent Variable, Response, Output (variable)
3. Scaling - transform a variable to have mean zero and variance one

7.2.1 Assumptions

Classic Assumptions for Statistics:

1. Linear Relationship between covariates and dependent variable
2. $E(\varepsilon) = 0$
3. $Var(\varepsilon) = \sigma^2$: Homoscedasticity
4. ε is independent with covariates
5. x is observed without error (and no perfect multicollinearity in multivariate case)
6. (optional, Gauss-Markov Theorem) ε is normal - when it is, OLS and MLE agrees and to be BLUE(Best Linear Unbiased Estimator)

7.2.2 Testing the Assumptions of Linear Regression

- Scatter Plot
  Linear Relationship and Outliers
- Residual Analysis $\hat{\varepsilon} = y - \hat{y}$
  Diagnostic Plots:
  1. Plot of Residuals vs. Fitted Values
  2. Normal Probability Plot
  3. Plot Residuals versus time (see any trend of fit)

- Cook's Distance

$$D_j = \frac{\sum_{i=1}^{n}(\hat{y}_i - \hat{y}_{i(-j)})^2}{(p+1)\hat{\sigma}^2}$$

  Test Against $F_{(p+1),(n-p-1}$ degrees of freedom, over 50th percentile will definitely become a problem

- Detect Multicollinearity (two or more predictors are strongly related to one) - Use **Variance Inflation Factor**

$$VIF_k = \frac{1}{1 - R_k^2}$$

  fit feature k against other predictors. Note VIF does not give any information of specific predictors

7.2.3 Resolutions of Assumption Violations

- Verify the Linear Relationships again. (non-linear regression, generalized linear models)
- Transformations (for outliers, heteroskesticities, etc)
- Use different models on different periods/data
- Weighted **Least Squares regression**, (for outliers, heteroskesticity)
- **Robust Regression** and Huber Loss Function

$$\sum_{i=1}^{n} \rho(\frac{y_i - x_i^T \beta}{\sigma})$$

  Huber Loss Function

$$\rho(x) = \begin{cases} x^2 \text{ ,if } |x| < k \\ k(2(|x| - k), \text{ otherwise} \end{cases}$$

  (default k=1.345) (when k=0, it is an L1-regression, $K \to \infty$, the regression goes back to a linear regression model. It is effective in down-weighting the extreme examples.

Special Situations

- Inputs are discrete variable - Factor Inputs (discrete features): a factor of k levels adds k-1 terms into the regression function.(k-1 different *beta*s)

7.2.4 Interpretation

Under Normal Condition, we have

$$y \sim N(\beta_0 + \beta x_i, \sigma^2)$$

$$L(\theta) = (\frac{1}{\sqrt{2\pi}\sigma})^n exp(-\frac{\sum_{i=1}^n (y_i - (\beta_0 + \beta_1 x_i))^2}{2\sigma^2})$$

Equivalent to minimize

$$RSS(\theta) = \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_i))^2$$

$$\partial_{\beta_i} RSS = 0, i = 0, 1$$

, we get

$$r_{xy} = \frac{s_{xy}}{s_x s_y}, \beta_1 = r_{xy}\frac{s_y}{s_x} = \frac{s_{xy}}{s_x^2}, \beta_0 = \bar{y} - \hat{\beta}\bar{x}$$

In Multi-variate Case:

$$f(x) = \mathbf{w}^T \mathbf{x} = \sum_{i=1}^{n} w_i x_i$$

$$\mathbf{w}* = \arg\min_{\hat{\mathbf{w}}} (\mathbf{y} - \mathbf{X}\hat{\mathbf{w}})^T (\mathbf{y} - \mathbf{X}\hat{\mathbf{w}})$$

$$\frac{\partial E}{\partial \hat{\mathbf{w}}} = 2\mathbf{X}^T (\mathbf{X}\hat{\mathbf{w}} - \mathbf{y})$$

$$\mathbf{w}* = (\mathbf{X^T X})^{-1} \mathbf{X^T y}$$

Assuming noise is normal, maximize

$$p(\mathbf{x_1}, \mathbf{x_2} ... \mathbf{x_n} | \mathbf{w}) = \prod_k \frac{1}{\sqrt{2\pi}\sigma} exp[-\frac{1}{2\sigma^2}(y_k - \mathbf{w_t x_k})^2]$$

Another matrix representation

$$f(\beta) = min(Y - X\beta)^T(Y - X\beta), f'(\beta) = 2X^T(Y - X\hat{\beta}) = 0$$

to solve $\hat{\beta}$

$$min||y_k - \mathbf{w^T x}_k||^2 + \lambda||\mathbf{w}||_1$$

Variance Error In Prediction

$$V(\hat{y^*} - y^*) = \sigma^2 + \sigma^2[\frac{1}{n} + \frac{x^* - \bar{x})^2}{(n-1)s_x^2}]$$
$$= V(E(y^*) - y^*) + V(\hat{y^*} - E(y^*)) + 2cov(\hat{y^*} - y^*, \hat{y^*} - y^*)$$

The cross term is zero, the first term is variance with ε^*, second term is variance in β.

The confidence interval is $\hat{y^*} \pm t_{\alpha/2,n-2}SE(\hat{y^*})$.

R^2, the coefficient of determination: The proportion of the sum of squared response which is accounted by the model relative to the model with no covariance. (take mean of response)

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})}$$

Note that $0 \leq R^2 \leq 1$ It only tells predictive power if the model is a good fit.

Adjusted R^2: R^2 + penalty P

Hat Matrix: The relationship of predicted value and response

$$Y = H\hat{Y}$$
$$H = X(X^TX)^{-1}X^T$$

The Diagonal Entires h_{ii} are the Leverages.

7.2.5 Model Selection

- Exhaustive Search by AIC or BIC (more stable than LOOCV)
- Stepwise Regression/Stepwise Variable Selection (At each step one covariate is added or dropped)
- Cross-Validation
  Leave-one-Out cross Validation of Linear Regression: Prediction Error Sum of Squares

$$PRESS = \frac{\sum(y_i - \hat{y}_{-i})^2}{n}$$

$$y_i - \hat{y}_{-i} = \frac{\hat{\varepsilon}_i}{1 - h_{ii}}$$

  h is the leverage (hat matrix)

7.2.6 Regularization, Ridge, Lasso

Ridge

$$Rss + \lambda \sum_{i=1}^{p} \beta^2$$

λ is the regularization parameter. The result of Ridge is a **Shrinkage** of $\hat{\beta}$ towards zero.

Note

1. No penalty for $\beta + 0$ or b.
2. The predictors should usually be standardized prior to fitting
3. Choose λ by cross-validation

Lasso(Least Absolute Shrinkage and Selection Operator)

(Tibshirani)

$$Rss + \lambda \sum_{i=1}^{p} |\beta|$$

Can be extended to

$$-loglikelihood + \lambda \sum_{i=1}^{p} |\beta|$$

Group Lasso

group predictors together to be either included or excluded.

Elastic Net

$$Rss + \lambda \sum_{i=1}^{p} (\alpha|\beta_j| + (1-\alpha)\beta_j^2)$$

, $0 \leq \alpha \leq 1$

7.3 Nonlinear Regression Models

- Nonparametric Regression: Complexity controlled by the smoothing parameter (bandwidth). model complexity interpreted in **Degrees of Freedom/Effective degrees of freedom/equivalent degrees of freedom**
  Residual Degrees of freedom is n minus model degrees of freedom.
- Local polynomial Regression: only fit a **neighborhood** of a target point. parameter α to control the span-traditionally, 0.5. When weighting the data in the neighborhood, Fit by weighted sum of squares
  $$\sum_{i=1}^{n} w_i(y_i - (\beta_0 + \beta_1 x_i))^2$$
  $$w_i == \begin{cases} (1 - |\frac{x_i - x_0}{\text{max dist}}|^3)^3 & \text{,if } x_i \text{ is in the neighborhood} \\ 0, \text{ otherwise} \end{cases}$$
- Splines

- Penalized (Smoothing) Splines: find twice differentiable x to minimize
$$\sum_{i=1}^{n}(y_i - f(x_i))^2 + \lambda \int [f^{(2)}(x)]^2 dx$$
λ penalty for wiggy function. search of x can be a combination of **basis functions** (n + 4 basis functions, n is the knots)
- Cubic Splines

7.4 Generalized Additive Models

Additive Model : no interactions/cross terms

Chapter 8

Classification and Generalized Linear Model

8.1 Logistic Regression

- Sigmoid/ Log Probability Function (linkage function)

$$\sigma(z) = \frac{1}{1+e^{-z}} = \frac{1}{1+e^{-\mathbf{w}^T\mathbf{x}}}$$

  Assume y is Bernoulli distributed and $P(Y|X) = \sigma(X)$ (conditional probability plot), indeed we have odds

- Interpretation:
  - the odds be represented as a linear relation

$$Log(\frac{p}{1-p}) = \mathbf{w}^T\mathbf{x}$$

    (log odds, logit function)

$$\frac{p}{1-p} = e^{\mathbf{w}^T\mathbf{x}}$$

$$p(y=1|x) = \frac{e^{\mathbf{w}^T\mathbf{x}}}{1+e^{\mathbf{w}^T\mathbf{x}}}$$

$$p(y=0|x) = \frac{1}{1+e^{\mathbf{w}^T\mathbf{x}}}$$

  - Maximum Entropy (See Loss function) of the exponential family

$$2\sum_{i]1}^{N} -y_i log\frac{y_i}{\hat{p}_i} + (1-y_i)log(\frac{1-y_i}{1-\hat{p}_i})$$

- Loss Function from Log MLE (of Bernoulli, assuming i.i.d.) or entropy

$$J(z) = -ylogy + (1-y)log(1-y)$$

MLE of w based on a Bernoulli Distribution

$$l(\mathbf{w}|\mathbf{x}) = L(\mathbf{w}|\mathbf{x}) = \prod_{i=1}^{N}[p(y=1|\mathbf{x},\mathbf{w})]^{y_i}[1-p(y=1|\mathbf{x},\mathbf{w})]^{1-y_i}$$

Log likelihood

$$logL(\mathbf{w}|\mathbf{x}) = \sum_{i=1}^{N} -y_i logp_i + (1-y_i)log(1-p_i)$$

- Model Training:

Train to maximize log likelihood Get Derivatives

$$\frac{\partial l}{\partial \mathbf{w}} = \frac{\partial l}{\partial p_i}\frac{\partial p_i}{\partial z_i}\frac{\partial z_i}{\partial \mathbf{w}} = (\frac{y_i}{p_i} - \frac{1-y_i}{1-p_i})(p_i(1-p_i))(\mathbf{x})$$

*Here we used the derivative of sigmoid

$$\frac{\partial \sigma(z)}{\partial z} = \frac{\partial p}{\partial z} = \frac{-e^{-z}}{(1+e^{-z})^2} = \sigma(z)(1-\sigma(z))$$

Regularization techniques:With L-1 or L-2 norm (Frobenius Norm)

$$J(z) = \frac{1}{m}\sum_{i=1}^{m} L(y_i, \hat{y}_i) + \frac{\lambda}{2m}\|\mathbf{w}\|_F^2$$

Another Intuition about minimizing the loss function is to minimize the **K-L Divergence** with Maximum-Entropy Model

*Connection with Naive Bayesian

- Naive Bayesian assumes $p(x_i|Y = y_k)$ follows a normal distribution. Then the posterior probability is

$$P(Y=0|x) = \frac{P(Y=0)P(X|Y=0)}{P(Y=0)P(X|Y=0)+P(Y=0)P(X|Y=1)}$$
$$= \frac{1}{1+exp(ln\frac{P(Y=0)P(X|Y=1)}{P(Y=0)P(X|Y=0)})}$$
$$= \frac{1}{exp(ln\frac{1-p_0}{p_0}+\sum(\frac{\mu_{i1}-\mu_{i0}}{\sigma_i^2}X_i+\frac{\mu_{i0}^2-\mu_{i1}^2}{2\sigma_i^2}))}$$

- Though the solution follows the exact same pattern, Logistic Regression does not have the assumption of independence. When assumptions differ, the results differ. Generally, logistic regression results less bias, more variance(more flexible)
- The rate of convergence is also different, logistic regression needs more data feeding to perform better.

8.2 Multi-Class Classification

8.2.1 Use binary classifier

- One-vs-One(OvO): $\frac{N(N-1)}{2}$ two-class classifications: Then vote among classification results
- One-vs-Rest(OvR): one as positive, consider all rest negative: combine the results
- Many-vs-Many(MvM): Cut N Samples with M partitions, do classification on M training examples. Use Error Correction Output Codes(EOOC) to get minimum-distance ones as result

8.2.2 softmax

$$P(Y=k|x) = \frac{e^{\mathbf{w}^T\mathbf{x}}}{\sum_{i=1}^{K} e^{\mathbf{w}^T\mathbf{x}}}$$

notice, $\mathbf{w} = (\theta_1, ..., \theta_n)$ has redundancy, when K = 2, it is logistic regression

8.3 Generalized Linear Model

$$y(\mathbf{x}) = g^{-1}(\mathbf{w}^T\mathbf{x} + b)$$

g is the linkage function

8.4 Practice/Examples

1. What is Anscombe's Quartet

Chapter 9

Support Vector Machine

9.0.1 Model and Assumptions

Find an hyperplane can separate all the samples:

$$\begin{cases} \mathbf{w}^T\mathbf{x} + b \geq +1, y = +1 \\ \mathbf{w}^T\mathbf{x} + b \leq -1, y = -1 \end{cases}$$

The vectors make ″=″ are the support vectors.

The margin is

$$\gamma = \frac{2}{\|\mathbf{w}\|}$$

($\frac{\mathbf{w}^T\mathbf{x}+b}{\|\mathbf{w}\|}$ is the point distance to plane)

So the problem is

$$\arg\max_{\mathbf{w},b} \frac{2}{\|\mathbf{w}\|}$$

$$s.t.(\mathbf{w}^T\mathbf{x}_i + b)y_i \geq 1$$

Equivalent to

$$\arg\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2$$

$$s.t.(\mathbf{w}^T\mathbf{x}_i + b)y_i \geq 1$$

Lagrange Multiplier

$$L = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{m} \alpha_i(1 - (\mathbf{w}^T\mathbf{x}_i + b)y_i)$$

With first-order condition for **w** and b we can have

$$\mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i, b = \sum_{i=1}^{m} \alpha_i y_i$$

Then we get the dual problem

$$\arg\max_{\boldsymbol{\alpha}} \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i\alpha_j y_i y_j \mathbf{x}_i^T\mathbf{x}_j$$

$$s.t \sum_{i=1}^{m} \alpha_i y_i = 0$$

$$\alpha_i \geq 0$$

When Satisfy K.K.T (Karush-Kuhn-Tucker) condition

$$\begin{cases} \alpha_i \geq +1, y = +1 \\ y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 \geq 0 \\ \alpha_i(y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1) \geq 0 \end{cases}$$

(See Optimization), could be solved using SMO(Sequential Minimal Optimization)

9.0.2 Kernel Function

For Linear Un-separable problems, we can project to higher-dimensions

$$\arg\max_{\mathbf{w},b} \frac{2}{\|\mathbf{w}\|}$$
$$s.t.(\mathbf{w}^T\phi(\mathbf{x}_i) + b)y_i \geq 1$$

$$\arg\max_{\boldsymbol{\alpha}} \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i\alpha_j y_i y_j \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$$
$$s.t \sum_{i=1}^{m} \alpha_i y_i = 0$$
$$\alpha_i \geq 0$$

Kernel

$$\kappa(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$$

We can find the solution by

$$f(x) = (\mathbf{w}^T\phi(\mathbf{x}_i) + b) = \sum_{i=1}^{m} \alpha_i y_i \kappa(\mathbf{x}, \mathbf{x}_i) + b$$

Theorem:

When a symmetric function has semi-positive definite kernel matrix

$$\begin{pmatrix} \kappa(\mathbf{x}_1, \mathbf{x}_1) & \cdots & \kappa(\mathbf{x}_1, \mathbf{x}_m) \\ \vdots & \ddots & \vdots \\ \kappa(\mathbf{x}_m, \mathbf{x}_1) & \cdots & \kappa(\mathbf{x}_m, \mathbf{x}_m) \end{pmatrix}$$

it can be a kernel function. Common Kernels are

- Linear Kernel
$$\kappa(\mathbf{x}_1, \mathbf{y}) = \mathbf{x}^T\mathbf{y}$$
- Polynomial Kernel
$$\kappa(\mathbf{x}_1, \mathbf{y}) = (\mathbf{x}^T\mathbf{y} + c)^d$$
- Gaussian Kernel
$$\kappa(\mathbf{x}_1, \mathbf{y}) = exp(-\frac{\|\mathbf{x} - \mathbf{y}\|^2}{2\sigma^2})$$
- Laplace Kernel
$$\kappa(\mathbf{x}_1, \mathbf{y}) = exp(-\frac{\|\mathbf{x} - \mathbf{y}\|}{\sigma})$$
- sigmoid Kernel
$$\kappa(\mathbf{x}_1, \mathbf{y}) = tanh(\beta\mathbf{x}^T\mathbf{y} + \theta)$$

9.0.3 Soft Margin, Slack Variable and Regularization

Chapter 10

Tree Models and Ensemble Learning

10.1 Decision Trees

ID3 C4.5 CART

advantage intepratable (e.g. popular in marketing, biomedical)

10.1.1 Bagging and Random Forest

10.1.2 Boosting and GBDT

Chapter 11

Dimension Deduction

Part III

Probabilistic Graphical Models

Chapter 12

Bayesian Probability Theory

12.1 Basic Concepts

- Bayesian Theorem:

$$P(\theta|X) = \frac{P(X|\theta)P(\theta)}{P(X)}$$

$P(\theta|X)$ is posterior probability
$P(\theta)$ is prior
$P(X|\theta)$ is likelihood
$P(X)$ is evidence

$$f_{Y|X}(y|x) = \frac{f_{(X,Y)}(x,y)}{f_X(x)} = \frac{f_{(X,Y)}(x,y)}{\int f(x|y)f(y)dy}$$

$$\pi(\theta|x) = \frac{f(x|\theta)\pi(\theta)}{\int f(x|\theta)\pi(\theta)d\theta}$$

$$\pi(\theta|x) \sim f(x|\theta)\pi(\theta)$$

- Bayes Estimator

$$\hat{\theta}_{Bayes} = E(\theta|X) = \int \theta\pi(\theta|X)d\theta$$

- Marginalization:
$$P(X) = \int p(X,Y)dY$$

- Bayesian network

$$P(X_1, ..., X_n) = \prod_{k=1}^{n} P(X_k|Pa(X_k))$$

  Plate notation

- Analytical Inference

  Maximum a Posterior (MAP) Estimation

- **Conjugate Distribution**: $f(x), \pi$ is called conjugate distributions if model $\pi(\theta|x), \pi(\theta)$ follows the same Distribution

  eg. Bernoulli(θ) and Beta(α, β), ($\pi(\theta|x) \sim$ Beta($\alpha + \sum X_i, \beta + n - \sum X_i$) ($f(x|\theta) = \prod_{i=1}^{n} f_{X_i}(X_i|\theta)$)

$$\hat{\theta}_{Bayes} = E(\theta|X) = \frac{\alpha + \sum X_i}{\alpha + \beta + n}$$

$$= \frac{\sum X_i}{n} \frac{n}{\alpha + \beta + n} + \frac{\alpha}{\alpha + \beta} \frac{\alpha + \beta}{\alpha + \beta + n}$$

  The prior mean (second term) influences less as n grows.

  Poisson(θ) and $Gamma(\alpha + \sum X_i, \beta + n)$

12.2 Bayesian Decision Theory

In State ω we take action $a \in A$, incur Loss $L(\omega, a)$, how to choose a to minimize Risk:

$$R(a|x) = \sum_{j=1}^{k} L(\omega_j, a) P(\omega_j|\mathbf{x})$$

Decision Rule $d \in A$

$$d^*(x) = \arg\min_{a \in A} R(a|\mathbf{x})$$

$d^*(x)$ here is a Bayes optimal classifier here. and $R(d^*(x)|\mathbf{x})$ is the Bayes Risk.

12.3 Latent Variable Models

- Latent Variable Models

  Pros: simpler model(network), few parameters, sometimes has meaningful explanations

  Cons: harder to work with

  handle missing data, model missing data as latent variable (distribution). eg *Probabilistic PCA*

- Expectation - Maximization (EM) Algorithm

  E- Step: calculate latent variable distribution

$$q(t_i) = p(t_i|x_i, \theta)$$

for each data point

M- Step:

$$p(t_i|x_i, \theta)$$

E-M:

$$max_\theta \, \mathbb{E}_q \, logp(X, T|\theta)$$

Example: Training Gaussian Mixture Model

$$logp(X|\theta) = \sum logp(x_i|\theta) = \sum log \sum \frac{q(t_i = c)}{q(t_i = c)} p(x_i, t_i = c|\theta) \geq$$

$$\sum \sum q(t_i = c) log \frac{p(x_i, t_i = c|\theta)}{q(t_i = c)} = \mathcal{L}(\theta, q)$$

E-Step:

$$q_{k+1} = \arg\max_q \mathcal{L}(\theta^k, q)$$

$$logp(X|\theta) - \mathcal{L} = \sum \mathcal{KL}(q(t_i)||p(t_i|x_i, \theta))$$

$$\mathcal{L}(\theta, q) = \sum_i \sum_c q(t_i = c) log \frac{p(x_i, t_i = c|\theta)}{q(t_i = c)}$$

$$= \sum_i \sum_c q(t_i = c) logp(x_i, t_i = c|\theta) - \sum_i \sum_c q(t_i = c) logq(t_i = c)$$

$$= \mathbb{E}_q \, logp(X, T|\theta) + const$$

usually use concave function to optimize M-Step:

$$\theta_{k+1} = \arg\max_{\theta} \mathcal{L}(\theta, q^{k+1})$$

Notice the usage of Jensen's inequality here.

Chapter 13

Naive Bayes

13.1 Model and Assumption

When we have a 0-1 loss

$$L = \begin{cases} 0 \text{ , if i=j} \\ 1, \text{ otherwise} \end{cases}$$

the risk become

$$R(a|x) = 1 - P(\omega_j|\mathbf{x})$$

$$d^*(x) = \arg\max_{a \in A} P(a|\mathbf{x})$$

If we build model around $P(a|\mathbf{x})$ directly, this is a **Discriminative Model**. If We try to model the joint distribution $P(\mathbf{x}, a)$, this is a **Generative Model**. Same as we get the Bayesian estimator, we try to find

$$P(a|\mathbf{x}) = \frac{P(a)P(\mathbf{x}|c)}{P(\mathbf{x})}$$

Naive Bayes made the important **assumption of attribute conditional independence** to write

$$\frac{P(a)P(\mathbf{x}|c)}{P(\mathbf{x})} = \frac{P(a)}{P(\mathbf{x})}\prod_{i=1}^{n} P(x_i|a)$$

We just need to count dataset to get

$$\hat{P}(x_i|a) = \frac{|D_{a,x_i}|}{|D|}$$

$$\hat{P}(a) = \frac{|D_a|}{|D|}$$

For continuous data, we can use probability density function to get the estimates.

In most cases, the smoothing **Laplacian Correction** is needed:

$$\hat{P}(x_i|a) = \frac{|D_{a,x_i} + 1|}{|D| + n}$$

$$\hat{P}(a) = \frac{|D_a| + 1}{|D| + n}$$

It can be proven, When we using the conjugate distribution of multinomial distribution to be the prior distribution and correct the parameter for Dirichlet Distribution to be $N_i + \alpha$ is equivalent for the Laplace Correction.

$$Dir(\mathbf{ff}) = \frac{\Gamma(\sum \alpha_i)}{\prod_{i=1}^{K} \gamma(\alpha_i)} \prod_{i=1}^{K} x_i^{\alpha_i - 1}, \sum_{i=1}^{K} x_i = 1$$

13.1.1 Semi-naive Bayesian Classifier

Assume certain dependencies between attributes. The most common case is "One-Dependent Estimator". Such

- Super-Parent ODE
- Tree Augmented Naive Bayes: Use the Maximum Weighted Spanning Tree. Weighted by mutual information (conditional entropy), Build a complete graph on attributes.
- Average One-Dependent Estimator: Ensemble on the SPODE Models

13.2 Model Benefits and Short-comings

- Robust to Missing Values
-

Chapter 14

Max Entropy Model

Chapter 15

Hidden Markov Model

Chapter 16

Conditional Probabilistic Field

Part IV

Unsupervised Learning

Chapter 17

Clustering

Hierarchical Clustering K-means

Chapter 18

Gaussian Mixture Model

Chapter 19

Topic Model

Latent Dirichlet Analysis(LDA)

Chapter 20

Dimension Reduction

20.1 Principal Component Analysis(PCA)

$$\mathbf{x}_i = (x_{i1}, x_{i2}, x_{ip})^T$$

find u to maximize variance $v_i = \sum_{j=1}^{p} u_j x_{ij}$

subject to

$$\sum_{j=1}^{p} u_j^2 = 1$$

u are like the new axes

$$v_i = \sum_{j=1}^{p} u_j x_{ij}$$

same as finding the eigenvalue $Si\hat{g}ma$

X is samples stacking in columns,

$$\mathbf{X} = \mathbf{X} - \mathbf{1}\boldsymbol{\mu}^T$$

be X shifted to mean zero

Then $\mathbf{v} = \mathbf{X}\mathbf{u}$ is a transformation (new position in axis)

$$\mathbf{v}^T 1 = (\mathbf{X}\mathbf{u})^T \mathbf{1} = \mathbf{u}^T \mathbf{X}^T 1 = \mathbf{u}^T \mathbf{0} = 0$$

the mean of v entries is zero

The variance of entries of v :

$$\frac{1}{n}\mathbf{v}^{\mathbf{t}}\mathbf{v} = \frac{1}{n}\mathbf{u}^T \mathbf{X}^T \mathbf{X}\mathbf{u}$$

equivalent to maximizing

$$\frac{\mathbf{u}^T \mathbf{X}^T \mathbf{X}\mathbf{u}}{\mathbf{u}^T \mathbf{u}}$$

$$\mathbf{X}^T \mathbf{X}$$

is the covariance matrix, this is equivalent to find the first eigenvector of covariance matrix if we scale X to have unit standard deviation, the problem is finding eigenvalues for correlation matrix

20.2 LDA

Part V

Deep Learning

Chapter 21

Deep Learning Model Optimization and Regularization

Characteristics of Deep-Learning

- Advantage of Deep-learning is significant mostly with large data set.
- Traditional bias-variance trade-off can largely be overcome by adding more data (reducing variance) and training a larger network(reducing bias) cycle when data is sufficient.
- Optimization becomes more crucial in the training process. Dataset normalization, gradient checking are needed. Initialization carefully to avoid
  - Gradient Vanishing/Exploding

21.1 Common Regularization Techniques

- L-1 or L-2 regularization (notice: also affects backward propagation-cause **weight decay**.) Reduces variance.

$$J(z) = \frac{1}{m}\sum_{i=1}^{m} L(y_i, \hat{y}_i) + \sum_{l=1}^{L} \frac{\lambda}{2m} \|\mathbf{w}^{[l]}\|_F^2$$

- Dropout: randomly shutting down (with certain probability) nodes during every training. Still predicting with the whole network. Reduces over-reliance on a certain node.

- Data Augmentation: adding transformed data to expand training set.

  : Adjust color (use PCA to analyze RGB values, add noises on each direction)

  : Random rotate, crop, shift, change size, symmetric transform

  : Add noise (Gaussian Noise, Salt-and-Pepper Noise)

  : Change resolution, sharpness, contrast

  : Use Generative Model to generate new samples(GAN)

  : Extract features first, then use up-sampling/SMOTE

- Early Stoping: early stopping during the optimization of parameters to avoid overfit.

21.2 Key Questions and Status Quo

- Scarce Data, Learning on Small Example (eg. Transfer Learning)
- Meta-Learning
- Knowledge from Hand-made features and Neural Network Combined Together

Chapter 22

Feedforward Neural Network

22.1 Multi-layer Perceptron

- MP Neuron
$$y = \phi(\sum_{i=1}^{N} w_i x_i)$$
$$w_i(t+1) = w_i(t) + \eta[d_j - y_j(t)]x_{j,i}$$
can only solve linear-separable problems

- Multilayer Perceptron: Including one hidden layer. The first Feedforward Neural Network. Errors passes by Back Propagation. Its it proven that a single-hidden layer multilayer perceptron can approximate any continuous functions at arbitrary error level.(universal approximation)

22.2 Neural Networks

22.3 Radial Basis Function Network

In the hidden layer, use activation function as activation function Radial Basis Function

$$\rho(\mathbf{x}, \mathbf{w_i}, \sigma) = exp(\frac{-\|\mathbf{x} - \mathbf{w_i}\|^2}{2\sigma^2})$$

As long as a feature's distance to the center vector (here $\mathbf{w_i}$) the same, the function value is the same. $\mathbf{w_i}$ separates different hidden unit band with bandwidth σ

The Gaussian function $exp(-\|\mathbf{x} - \mathbf{u_i}\|^2)$(like Kernel) can help to transform linear inseparable case as-if projecting to a high-dimension space (same as SVM), to a linear separable case.

Alternatively, treat an RBF as a interpolation solution. It tries to data hyperplane. It reduces the noise by interpolation among the data points. The interpolated hyperplane still passes all data points.

Training of RBF

1. Initialization of $\mathbf{w_i}$ by random initialization or **unsupervised learning** like K-means.
   Usually, we have $\sigma = d_max/\sqrt{2K}$, d_max is the maximum distance between centers. (make sure bandwidth is not too small or too big)

2. Training $\mathbf{w_i}$. Use Recursive Least Square

$$\mathbf{R}(n)\hat{\mathbf{w}}(n) = \mathbf{r}(n)$$

   $\mathbf{R}(n)$ is the covariance matrix between hidden layer outputs ($\hat{y}$),
   $\mathbf{r}(n)$ is the covariance vector between hidden layer outputs ($\hat{y}$) and model response.
   Training by solving $\mathbf{R}^{-1}(n)$

3. After training, use Back propagation to train all parameters one more time. (train the whole network after training layers)

Compare with Neural Network: both can achieve universal approximation, while RBF network uses a local approximation approach.

Deep Learning sees most results in supervised Learning.

Feedforward neural network

22.4 Convolutional Neural Network(CNN)

Basic Idea is to use Convolution to engineer (extract and filter out) the features. (the "Edge Detection Problem")

22.4.1 Convolution and Pooling

Convolution: The "Convolution" in DL is really Cross-Correlation. (See "Convolution" on wikipedia)

- Convolute the Image Data with a Filter(Kernel) (eg. Sobel Filter, Scharr Filter)

$$(n.n) * (f, f) \to (n - f + 1, n - f + 1)$$

- Padding: add zeros entries so the output size same as input size $(n + 2p - f + 1, n + 2p - f + 1)$ as result
  - Valid Padding: No Padding out
  - Same Padding: Output the Same Size
  - FULL Padding: Maximum Padding does not result in a convolution on just padded elements(eg. for a filter of size k, k-1)
- Stride:steps to take $(\lfloor \frac{n+2pf}{s} + 1 \rfloor, \lfloor \frac{n+2pf}{s} \rfloor + 1)$ rounding down (dont do when it is out)
- Convolution over volume: Traditionally, use a filter with same number of channels (each 3-dimensional filter result an matrix output)
- Convolution of tensor - still all number multiply then sum together could use m filters to result an m channel output
- 1-layer Convolution Network: Input $\to$ n filters $\to$ ReLu on each output (bias parameter added here) $\to$ Stack the output together result an n channel output
- Why convolution works

  - Parameter Sharing: One feature detector is useful for one part of image probably be useful for all parts(fewer parameters)
  - Sparsity of Connections: In each layer, each output value depends only on a small number of inputs

Pooling:

- Max Pooling - Take max of sub-areas' outputs, result a matrix
- Average Pooling - Take the average of sub-areas, Used less often
- No padding
- Act as **Dimension Reduction** and **Down-sampling**

22.4.2 Convolutional Neural Network(CNN)

Basic Architecture

Image of Different Channels (RGB) $\rightarrow$ Conv Layer $\rightarrow$ Pooling (multiple layers of both) $\rightarrow$ Fully connect layer (flatten all data) $\rightarrow$ Sotfmax $\rightarrow$ Prediction. Use Back propagation to train (Mini-batch gradient descent)

- LeNet-5
  - avg pool
  - shrink each step
  - no padding
  - conv-pool, conv-pool pattern
  - activation:sigmoid, ReLu
- AlexNet
  - much bigger network
  - ReLU
  - Multiple GPU Training
  - Local Response Normalization (normalize over all channels)

- VGG
  - VGG-16
  - VGG-19: even bigger

22.4.3 Deep Residual Network(ResNet)

- Train significant Deeper Networks
- Build by Residual Blocks
  - identity block
  - convolution block
- **Skip Connections/ Short Cut**

$$a^{[l+2]} = g(z^{[l+2]}) + a^{[l]})$$

- helps gradient propagation
- less likely to learn identity functions

22.4.4 Inception Net

Network in Network and 1x1 Convolution

- Adds no linearity to the neural Network
- Shrink the number of Channels

Inception Net

- Uses 1x1 convolution
- Create a "bottleneck layer"
- Reduce the number of computations significantly
- Different (may be one or two layers of) Pooling/Convolutions in one layer, Channel Concatenation
- Identity Blocks connected together

22.4.5 Computer Vision and YOLO Algorithm

Object Detection

- Can have different objects (several objects)(little different than classification with detection)
- Output a vector: $[p_c, b_x, b_y, b_h, b_w, c_1, c_2...c_n]$ for both location(center + bounding box) and class

  use square loss function
- IOU - Intersection under union

  Intersection of two bounding boxes/union of boxes (predicted and actual)

  thredhold usually 0.5

Landmark Detection

Output x y coordinates for important points in the image

Convolutional Implementation of Sliding Windows + YOLO - You Only Look Once Algo(Bounding Box Detection)

- Use convolution to replace two) fully connected layers(two layers) in the network

  pic $\rightarrow$ Deep CNN $\rightarrow$ encoding of dimension(m, grid, grid, $n_{anchorboxes}$, $n_{features}$)

  $n_{features}$ is dimension of $[p_c, b_x, b_y, b_h, b_w, c_1, c_2...c_n]$
- Assign center of the object to the grid
- **Non-max suppression**: detect the object **only once**

  Use the p_c probability, only keep the one with highest p_c intersected rectangles

  discard any box below a p_c thredhold

  once for each output class

- Anchor Boxes

    different shape boxes to deal with overlapping objects

    output becomes $[p_c, b_x, b_y, b_h, b_w, c_1, c_2 ... c_n]$ for two anchor boxes

    each object assigned to the center grid cell and anchor box IOU

Region Proposal

- Segmentation Algorithm to find blob and only run on bounding box on these
- R-CNN and Fast R-CNN algorithm, Faster Algorithm

Face Recognition

- Verification: confirm identity
- recognition: database of K persons, output id if any of the K persons
- One-shot Learning: learn using just one example
- Learn a similarity function: $d(p1, p2) \leq t$ the same person
- **Siamese network**: use neural network as encoding. $|f(x(i)) - f(x(j))|$ is small

    Triplet Loss: encoding of the anchor-positive smaller than anchor-positive.

    $$L(A, P, N) = max((|f(A) - f(P)|^2 - |f(A) - f(N)|)^2 + \alpha, 0)$$

    choose triplet that is hard rather than randomly to improve computational efficiency

    $y_h at = \sigma(\sum w_i |f(xi)k - f(xj)k| + b)$. usually use pre-computed network to make prediction

Neural Style Transfer

- $$J(G) = \alpha J_{content}(c, G) + \beta J_{style}(S, G)$$

- Content cost: use pre-trained Conv Net (eg. VGG Net)

  $$J(C, G) = 1/2|a[l](c) - a[l](G)|^2$$

  (L2-norm as cost)

- Style Cost: correlation should be the measure of closeness in style. Use **Style matrix** - i,j,k is on dimension H,W,C

  $$G^l_{kk} = \sum_i \sum_j a^{[l]}_{ijk} a^{[l]}_{ijk}$$

  the gram matrix

  $$J(S, G) = |G[l](S) - G[l](G)|^2_F$$

  normalized

22.4.6 Convolution in 2D and 1D Data

Filter/Knernel will be 1D and 2D to convolute on the data

22.5 self organizing feature map(SOMNet)

22.6 Restricted Boltzman Machine(RBM)

22.7 Model Optimization/Regularization

Batch Normalization Dropout Activation sigmoid softmax tanh ReLu

Chapter 23

Sequence Model

aa Sequence models deal with sequence data (speech, music, DNA, Sentiment Classification, etc). In sequence data, Inputs, outputs can be different in size and Input might not share features across sequence (eg. text)

23.1 Recurrent Neural Network(RNN)

- Use the output from t-n(eg. t- 1) as an input for t prediction
- the output at each time step is passed by a random sampling with predicted probability
- forward propagation
  - w parameters shared across t
  - $a^{<t>} = g(w_{ax}x_t + w_{aa}a^{<t-1>})$
  - $y^{<t>} = g(w_{ya}a^{<t>} + b_y)$
- Vectorization
  - $[w_{aa}|w_{ax}] = w_a$
  - $[a_{t-1}, x_t]$ stacked vertically
- Backward propagation

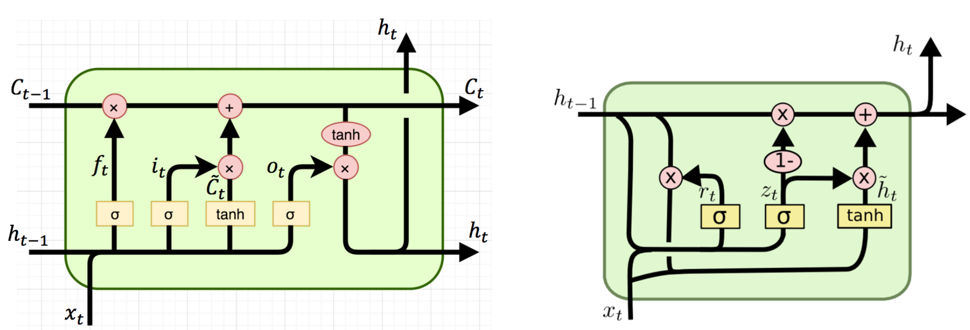
- $L^{<t>}(\hat{y}_t, y_t) = -y^t log \hat{y}_t - (1 - y_t) log(1 - \hat{y}_t)$
- $L(\hat{y}, y) = \sum L(t) L^{<t>}$ across t
- weights propagates back from t to 0

- Different Architecture Types
  - Standard RNN(many-to-many RNN)
  - encoder + decoder(x RNN connects to y RNN in sequence
  - many-to-one RNN (only one output at t)
  - One-to-One (only one time piece)
  - One-to-Many (only one input at time 0)

- Vanishing Gradient
  - basic RNN has many local influences (influence not far)
  - Exploding: solved by **gradient clipping** - rescale gradient when gradient hits some thredhold
  - Vanishing is harder to solve

23.2 Gated Recurrent Unit (GRUand Long Short Term Memory (LSTM) Model

GRU and LSTM can solve the problem of gradient vanishing and creates long-distance dependencies by "peephole connection". Stacking LTSM or GRU units together creates LTSM/GRU network.



(a) Long Short-Term Memory (b) Gated Recurrent Unit

Gradient Recurrent Unit:

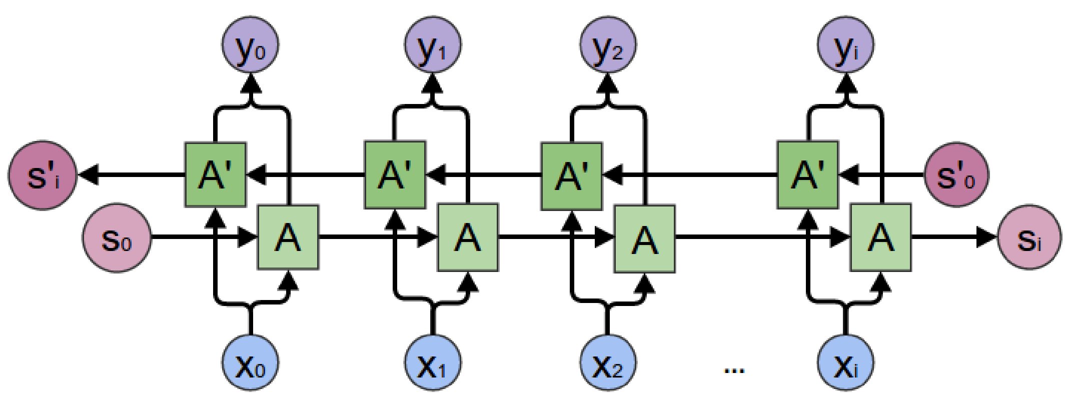
$$\tilde{c}^{<t>} = tanh(W_c[\Gamma_r * c^{<t-1>}, x^{<t>}] + b_c)$$
$$\Gamma_u = \sigma(W_u[c^{<t-1>}, x^{<t>}] + b_u)$$
$$\Gamma_r = \sigma(W_r[c^{<t-1>}, x^{<t>}] + b_r)$$
$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1 - \Gamma_u) * c^{<t-1>}$$
$$a^{<t>} = c^{<t>}$$

Long Short Term Memory:

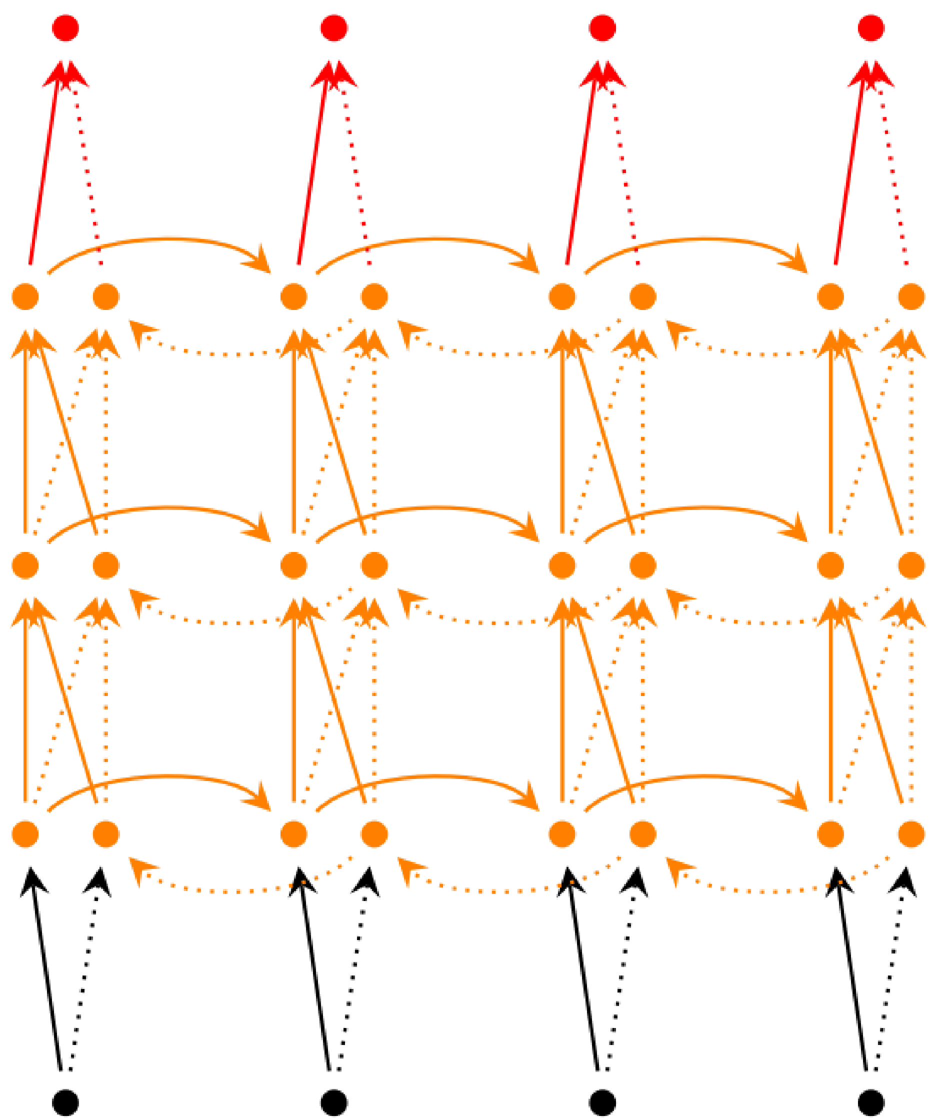
$$\tilde{c}^{<t>} = tanh(W_c[c^{<t-1>}, x^{<t>}] + b_c)$$
$$\Gamma_u = \sigma(W_u[c^{<t-1>}, x^{<t>}] + b_u)$$
$$\Gamma_f = \sigma(W_f[c^{<t-1>}, x^{<t>}] + b_r)$$
$$\Gamma_o = \sigma(W_o[c^{<t-1>}, x^{<t>}] + b_o)$$
$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + \Gamma_f * c^{<t-1>}$$
$$a^{<t>} = \Gamma_o c^{<t>}$$

23.3 Bidirectional RNN and Deep RNN

Bidirectional RNN - Two activations connected in two different directions



Deep RNN - Stacking RNN Layers together. Three layers are already pretty deep.



23.4 Language Model

Language model uses data from a **Corpus**. it models probabilities of words appear(generative model).

The input is a language sequence, prediction is the (conditional) probability of next word appearing. The prediction is a Sampling (according to predicted probability until we hit EOS).

Application examples like

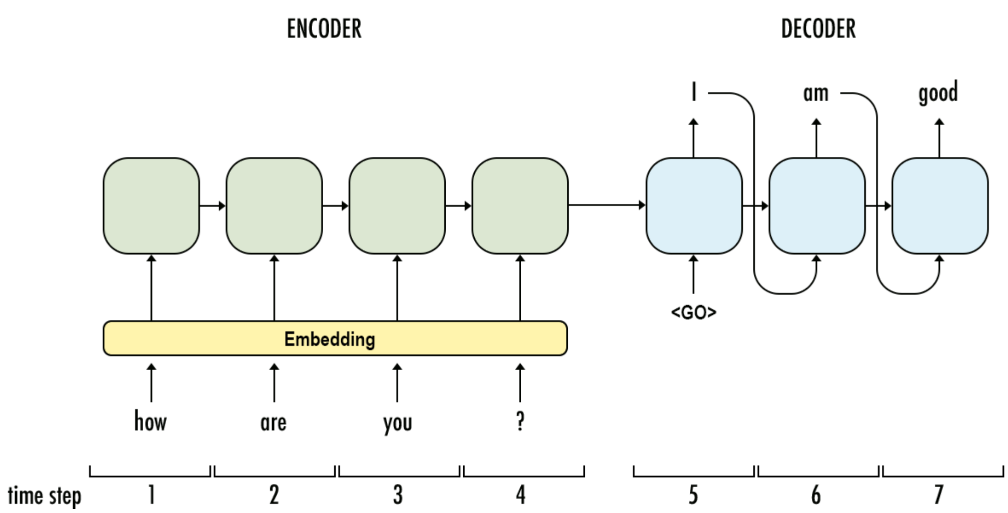
Sentiment Classification:

$$\text{word embedding} \rightarrow Average \rightarrow softmax \rightarrow \hat{y}$$

Or a *many to one* RNN with sentiment as the (only) output at the last step.

23.4.1 Sequence-to-Sequence model

In applications like machine translation, a encoder-decode model is used.



Beam Search for training.

usually refine to

$$\arg\max_{y} \frac{1}{T_y^{\alpha}} \sum_{t=1}^{T_y} logP(y^{<t>}|x, y^{<1>}, ..., y^{<t-1>})$$

to avoid a preference for short sentence (length normalization)

Error Analysis:

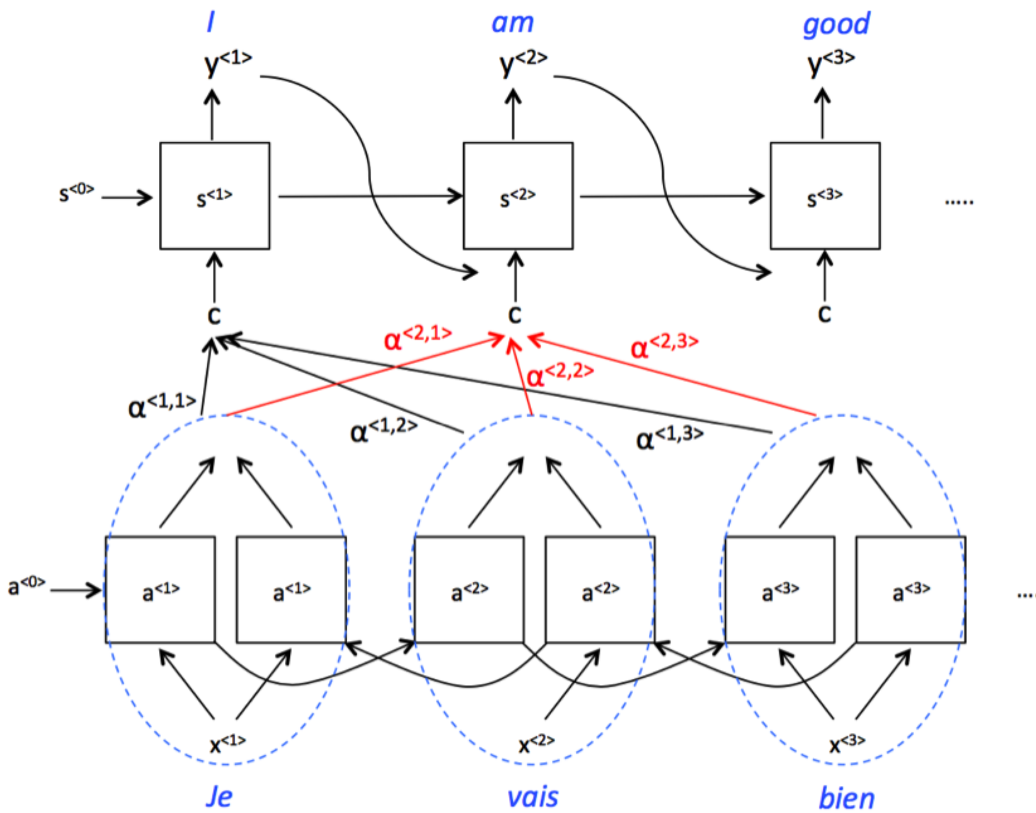
By comparing $P(y^*|x)$ and $P(\hat{y}|x)$ we can attribute the error to beam search or RNN(if searched probability lower than correct probability-Beam Searchs fault)

Translation problem measure - Bleu Score (See paper textitmethod for automatic evaluation of machine translation)

$$p_n = \frac{\sum_{ngram \in \hat{y}} count_{clip}(ngram)}{\sum_{ngram \in \hat{y}} count(ngram)}$$

23.4.2 Attention Model

To Solve the problem sequence-to-sequence model faces in long sentences (information become less useful when too far). We add double hidden state and parameters to represent the attention paid to models.



Attentions are trained by a small-neural network

$$a_{<t,t'>} = \frac{exp(e_{<t,t'>})}{\sum_{t'=1}^{T_x} exp(e_{<t,t'>})}$$

$$f(a_{<t>}, s^{<t-1>}) = e_{<t,t'>}$$

Chapter 24

Generative Adversarial Networks(GAN)

Chapter 25

Reinforcement Learning