DS4400 Notes

DS4400 Notes 01/24

1. Convex functions:

   A function $f : \mathbb{R}^d \to \mathbb{R}$ is convex iff $\forall \theta_1, \theta_2 \in \mathbb{R}^d$ and $\forall \alpha \in [0,1]$ we have $f(\alpha\theta_1 + (1-\alpha)\theta_2) \leq \alpha f(\theta_1) + (1-\alpha) f(\theta_2)$

   In the special case $(d = 1)$ $f : \mathbb{R} \to \mathbb{R}$, f is convex iff $\forall \theta, f''(\theta) \geq 0$

   When the function is convex, **local min** $\equiv$ **global min**. When the system is not convex, we might find only a **local min** but not a **global min**

2. Dealing with non convex function:

   In gradient descent:

   (a) use larger ρ in the beginning and gradually decrease ρ with interation.

   (b) Run SGD/GD with multiple random initializaitons $\theta_1^{(0)}, \theta_2^{(0)} \ldots$ and keep the best solution.

3. $\arg\min_\theta \sum_{i=1}^N (y_i - \theta^T x_i)^2 \triangleq J(\theta)$

   In linear regression, $J(\theta)$ is convex.

4. Robustness of Regression to outliers:

   (a) Run outlier detection algorithm, remove detected outliers, then run Linear Regression on remaining points.

   (b) Robust Regression cost function.
   $\arg\min_\theta \sum_{i=1}^N e_i^2,\ e_i \triangleq y_i - \theta^T x_i$
   e^2 is extremly unhappy with large errors.
   we might use $|e|$ to replace the function. This might be more tolerance. Then, $\arg\min_\theta \sum_{i=1}^N |y_i - \theta^T x_i|$

5. Exercise: D $= \{(x_1, y_1 = 100) \ldots (x_10, y_10 = 100), (x_{11}, y_{11} = 0), (x_{12}, y_{12} = 0)\}$

   e^2: $10(\theta - 100)^2 + 2\theta^2 \to$
   $\frac{\partial}{\partial\theta} = 20(\theta - 100) + 4\theta = 0 \to$
   $\theta = 83.3$

   $|e| : \min_\theta \sum_{i=1}^{12} |\theta - y_i| = 10|\theta - 100| + 2\theta$
   $(\theta \leq 100) = \min_\theta 10(100 - \theta) + 2\theta$
   $= 1000 - 8\theta \to \theta = 100$
   $(\theta \geq 100) = \min_\theta 10(\theta - 100) + 2\theta$
   $= 12\theta - 1000 \to \theta = 100$

6. How to solve l1-norms cost functions?
    (a) No closed form
    (b) we need to be careful with gradient descent
    (c) We need to use convex programming toolboxs (convex optimizations)

7. Huber loss funct

$$l_\delta(e) = \begin{cases} \frac{1}{2}e^2 & |e| \leq \delta \\ \delta|e| - \frac{\delta^2}{2} & |e| \geq \delta \end{cases}$$

$$\frac{\partial l_\delta(e)}{\partial e} = \begin{cases} e & -\delta \leq ele\delta \\ \delta & e > \delta \\ -\delta & e < \delta \end{cases}$$

   in huber loss function, we don't have closed form solution but we can run gredient descent now.

8. Definition: Overfitting:

   Learning a system from traning data that does very well on training data itself (e.g, very low regression error on traning data), but performs poorly on test data.

9. Definition: Overfitting in Linear Regression

   $\Phi^T\Phi\theta = \Phi^T Y$
   $\Rightarrow \theta^* = (\Phi^T\Phi)^{-1}\Phi^T Y$

   $\text{rank}(\Phi^T\Phi) \leq \min\{rk(\Phi^T), rk(\Phi)\} = rk(\Phi) \leq \min\{N, d\}$

   $\Phi^T\Phi$ is $d \times d$ matrix, then rank is $\leq d$.

   Therefore, when $N < d$ it is not invertible which means we have multiple solutions and results in overfitting.

DS4400 Notes 01/28

1. Definition: Overfitting
   Refers to situation where the learned model does well on traning data and poorly on testing data.
   As d (dimension of system) increases, then training error godes down (can be exactly ZERO for sufficiently large d)

2. In Linear regression:

$$\min \sum_{i=1}^{n} (\theta^T \phi(x_i) - y_i)^2$$

   set the derivative to 0 and we find

$$\Phi^T\Phi\theta = \Phi^T Y$$

   Then $\theta^* = (\Phi^T\Phi)^{-1}\Phi^T Y$

   When is it the case that $\Phi^T\Phi$ is not invertible?

Since $\Phi^T\Phi \in \mathbb{R}^{N\times d}$

$$rk(\Phi^T\Phi) \leq rk(\Phi) \leq min\{N, d\}$$

$\Phi^T\Phi \in \mathbf{R}^{d\times d}$ is invertible when $rk(\Phi^T\Phi) = d$. Therefore, when $N < d, rk(\Phi^T\Phi) = N$, $\Phi^T\Phi$ is not invertible. There will be infinitely many solutions for θ.

Generally, need sufficient # samples

3. Test overfitting.
   If $\Phi^T\Phi$ is not invertible,
   $\exists v \neq 0, \Phi^T\Phi v = 0$
   $\Rightarrow \theta^* + \alpha v$ is also a solution for any $\alpha \in R$
   $\Phi^T\Phi(\theta^* + \alpha v) = \Phi^T\Phi\theta^* + \Phi^T\Phi(\alpha v)$
   $= \Phi^T\Phi\theta^* + \alpha\Phi^T\Phi v$
   $= \Phi^T\Phi\theta^* = \Phi^T Y$

   We can find large α so that θ^* have extremly large entries.

   Generally, if the entries are very large (abs) we might have overfitting

4. Treat overfitting
   We want to change regreession optimization to prevent θ from very large terms.

   then we change the cost function:

$$\min_{\theta} \sum_{i=1}^{N} (\theta^T\phi(x_i) - y_i)^2 + \lambda \sum_{j=1}^{d} \theta_j^2$$

   λ: regularization parameter (> 0)
   $\sum_{j=1}^{d} \theta_j^2$: regularizer.
   $\lambda \to 0$: back to overfitting
   $\lambda \to \infty : \theta^* = 0$, underfitting

   (a) closed-form
       $\frac{\partial J}{\partial \theta}$
       $= 2\Phi^T(\Phi\theta - Y) + \lambda \frac{\partial \sum_{j=1}^{N} \theta_j^2}{\partial \theta}$
       $= 2\Phi^T(\Phi\theta - Y) + 2\lambda\theta$
       Let it be zero:

$$\Phi^T\Phi\theta + \lambda\theta = \Phi^T Y$$

$$(\Phi^T\Phi + \lambda I_d)\theta = \Phi^T Y$$

       Then $\theta^* = (\Phi^T\Phi + \lambda I_d)^{-1}\Phi^T Y$

   (b) Gradient descent
       Find initial $\theta^{(0)}$
       $\theta^t = \theta^{(t-1)} - \rho\frac{\partial J}{\partial \theta}|_{\theta^{(t-1)}}$
       $= \theta^{(t-1)} - 2\Phi^T(\Phi\theta^{(t-1)} - Y) + 2\lambda\theta^{(t-1)}$

5. Hyperparameter Tunning

   GD: set learning rate ρ

Robust Reg: Huber loss δ

overfitting and regularization: λ

ρ, δ, λ = hyperparameters

How to pick hyperparameters?

BAD APPROACH 1:

(a) pick some set of possible $\lambda_i \in \{\lambda_1, \lambda_2 \dots\}$
Run regression with λ_i and find θ_i^*
Measure regression error:

$$\epsilon_{tr}(\lambda) = \sum_{i=1}^{N} ((\theta^*(\lambda))^T x_i - y_i)^2$$

To sum: just find λ for which $\epsilon_{tr}(\lambda)$ is minimum

This approach is setting λ back to 0

Test data needed!!!

(a) We need to Train λ_i on **training set** to minimize the cost function

$$2\Phi^T(\Phi\theta - Y) + 2\lambda\theta$$

to find θ_i^*

(b) Measure regression error on the **hold-out set** D^{ho}

$$\epsilon_{tr} = \sum_{x_i, y_i \in D^{ho}} (y_i - (\theta^*(\lambda))^T x_i)^2$$

DS4400 Notes 01/31

1. Hyperparameter Tunning:

$$\min_{\theta} \sum_{i=1}^{N} (\theta^T \phi(x_i) - y_i)^2 + \lambda \sum_{j=1}^{d} \theta_j^2$$

- For $\lambda \in \{\lambda_1, \lambda_2 \dots, \lambda_p\}$
  - Tran using D^{tr} with $\lambda \to \theta^*(\lambda)$
  - Measure validation error

$$\epsilon^{tr}(\lambda) = \sum_{x_i, y_i \in D^{ho}} (y_i - (\theta^*(\lambda))^T x_i)^2$$

- select λ which minimizes

$$\epsilon^{ho}(\lambda) \to \lambda^* = \min_{\{\lambda_1, \lambda_2 \dots, \lambda_p\}} \epsilon^{ho}(\lambda)$$

2. Problems:

   - Take much longer time since we are training the models multiple times
   - Each training is using a subset of the data set, then each training is amplifing the problem of overfitting.

3. K-fold cross validation

   divide Data set to k equally large sets $\{D_1, D_2, \ldots, D_k\} \in D$

   - For $\lambda \in \{\lambda_1, \lambda_2 \ldots, \lambda_p\}$
     - For $i = 1, 2, \ldots, k$
       * train on $\bigcup\limits_{j \neq i} D^j$ and get $\theta_i^*(\lambda)$
       * compute validate error on $D^i \rightarrow \epsilon_i^{ho}(\lambda)$
     - compute average of $\{\epsilon_i^{ho}(\lambda)\}$: $\epsilon^{ho} = \frac{1}{k}\sum_{i=1}^{k} \epsilon^{ho}(\lambda)$
   - select $\lambda^* = \min\limits_{\{\lambda_1, \lambda_2 \ldots, \lambda_p\}} \epsilon^{ho}(\lambda)$

   Once we find the best λ, train the model on the whole set.

4. PROBABILITY REVIEW

   - Random Variable: a variable that takes values corresponding to outcome of a random phenomenon.
   - Discrete r.v.: descrete values
   - continuous r.v. continus range of values
   - Condition: $P(X|Y) = \frac{P(X \cap Y)}{P(Y)}$

     $$P(X, Y) = P(X|Y)P(Y)$$

     $$P(X, Y) = P(Y|X)P(X)$$

     Chain rule:
     $P(X_1, X_2, \ldots, X_n) = P(X_1)P(X_2|X_2)P(X_3|X_1, X_2)$
     $\ldots P(X_N|X_1, X_2 \ldots X_N)$
   - Marginalization
     $p(x, y)$ known
     $p(x) = \sum\limits_{y} p(x, Y = y) = \int p(x, y) dy$
   - Bayes Rule:
     $P(X|Y) = \frac{P(Y|X)P(X)}{P(Y)} = \frac{P(X|Y)P(Y)}{P(X)}$
   - Independence:
     r.v. are independent $(X \perp\!\!\!\perp Y)$ iff
     $P(X|Y) = p(X), P(Y|X) = p(Y)$
     or $P(x, y) = P(x)p(y)$

- conditional independence example: X = height of person, Y = vocabulary, X is not independent of Y since babies may have less vocabulary and with lower heights. However, X = height, Y = vocab, Z = age. Then $(X \perp\!\!\!\perp Y) \mid Z$

$$P(X, Y|Z) = P(X|Z)P(Y|Z)$$

$$\Rightarrow P(X|Y, Z) = P(X|Z)$$

- Expectation:
  $E(X) = \sum xp(x)$ or $\int xp(x)dx$
  $E(f(X)) = \sum f(x)p(x)$ or $\int f(x)p(x)dx$
  Given $X \perp\!\!\!\perp Y$, $E[XY] = E[X]E[Y]$
  hint: $(E[XY] = E[f(x,y)]$
- IID r.v: independent and identically destributed
  $p(X_1 = x_1, X_2 = x_2, \dots X_n = x_n) = p(X_1 = x_1)p(X_2 = x_2)\dots p(X_n = x_n)$ and each expriment is identical.
  $P(X_1 = \theta) = P(X_2 = \theta) = \cdots = P(X_n = \theta)$

DS4400 Notes 02/04
Maximum Likelihood Estimation

1. Some distributions:
   - Gaussian Dist.
     $P(X = x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$
   - Laplace Dist.
     $P(X = x) = \frac{1}{2\lambda} e^{-\frac{|x-\mu|}{\lambda}}$
   - Bernoulli Dist.
     $P_\theta(x = 1) = \theta, P_\theta(x = 0) = 1 - \theta$
2. Goal: Learn parameters of probability models. (fix the prob. model class)

   In ML, we learn these parameters (θ) using training data, D.

   We want to measure $P_\theta(D)$

   MLE: $\theta^* = \arg\max_\theta P_\theta(D)$ Under such θ^*, the probability of observing the given dataset is maximum.
3. Exercise: Fliping a coin.

   This is a Binomial Dist. (n time Bernoulli Trials)

   model: $p(X = x) = \theta^x(1-\theta)^{1-x}, x = 0, 1$

   $P_\theta(D) = P_\theta(X_1 = x_1, X_2 = x_2 \dots)$

   Assuming that tossing coins are iids:

   $P_\theta(D) = P_\theta(x_1)P_\theta(x_2)\dots$
   $= \theta^{\sum x_i}(1-\theta)^{\sum(1-x_i)}$

   Then likelihood funciton:

$L(\theta) = P_\theta(D)$

Take the logrithm of both sides (simplify product to sum)

$\theta^* = \arg\max_\theta logL(\theta)$

REASON:

1. log is monotonically increasing

2. simplify the powers to scale, the product to sum.

3. Increase the dynamic range (working with small numbers is not accurate on computers and memory consuming)

$$\frac{\partial logL(\theta)}{\partial\theta} = \sum x_i \frac{1}{\theta} + (N - \sum x_i)\frac{-1}{1-\theta}$$

Let the derivative equals to 0.

$$\frac{1}{\theta}\sum x_i = \frac{1}{\theta - 1}(N - \sum x_i)$$
$$\theta = \frac{\sum x_i}{N}$$

4. Exercise: People's height

Use model: normal distribution.

$L(\theta) = P_{\sigma,\mu}(D) = \prod_{i=1}^{N} P_e(x_i)$

$logL(\theta) = -\frac{N}{2}log(2\pi\sigma^2) - \frac{\sum_{i=1}^{N}(x_i-\mu)^2}{2\sigma^2} = -\frac{N}{2}log(2\pi) - \frac{N}{2}log(\sigma^2) - \frac{\sum_{i=1}^{N}(x_i-\mu)^2}{2\sigma^2}$

$$\frac{\partial logL(\theta)}{\partial\mu} = -\sum_{i=1}^{N}(\mu - x_i)/\sigma^2$$

$$\Rightarrow \hat{\mu} = \frac{\sum_{i=1}^{N} x_i}{N}$$

$$\frac{\partial logL(\theta)}{\partial\sigma^2} = -\frac{N}{2}\frac{1}{\sigma^2} - \frac{\sum_{i=1}^{N}(x_i-\mu)^2}{2}\frac{-1}{\sigma^4}$$

$$= \frac{-N}{2\sigma^2} + \frac{\sum_{i=1}^{N}(x_i-\mu)^2}{2\sigma^4}$$

$$\hat{\sigma^2} = \frac{1}{N}\sum_{i=1}^{N}(x_i - \hat{\mu})^2$$

DS4400 Notes 02/07
Classification:
Binary Classification:
input = Email $\rightarrow$ output = 'span' vs 'non-span'
Multiclass Classification:
input = Image $\rightarrow$ output = 'car', 'bike', 'stop-sign', ...

- **Classification Setup**:
  Given a training dataset $D = \{(x_1, y_1) \cdots (x_n, y_n)\}$ where $x_i \in \mathbb{R}^d$ is input feature vector and $y_i \in \{0, 1, 2, \ldots, L-1\}$, Find a mapping $g : \mathbb{R}^d \rightarrow \{0, 1, 2, \ldots, L-1\}$ s.t. $g_w(x_i) = y_i$ for many i's $\in \{1, 2, \ldots, N\}$

- **Assumption:**
  Assume that there is a hyperplane $w^T \phi(x) = 0$ that separates data into two classes.
  Then set: $w^T \phi(x) > 0 \rightarrow y = 1$, set: $w^T \phi(x) < 0 \rightarrow y = 0$

- Logistic Regression Model:
  $P_w(y = 1|x) \propto e^{w^T \phi(x)/2}$
  $P_w(y = 0|x) \propto e^{-w^T \phi(x)/2}$
  Determine Z: $P_w(y = 1|x) + P_w(y = 0|x) = 1$:
  $\frac{1}{z} e^{w^T \phi(x)/2} + \frac{1}{z} e^{-w^T \phi(x)/2} = 1$
  $\rightarrow z = e^{w^T \phi(x)/2} + e^{-w^T \phi(x)/2}$
  $\rightarrow P_w(y = 1|x) = \frac{1}{z} e^{w^T \phi(x)/2} = \frac{1}{1+e^{-w^T \phi(x)}}$
  Model: $P_w(y = 1|x) = \frac{1}{1+e^{-w^T \phi(x)}}$
  Model: $P_w(y = 0|x) = 1 - \frac{1}{1+e^{-w^T \phi(x)}}$

- signoid/logistic function:
  $\sigma(z) = \frac{1}{1+e^{-z}}$

- Logistic regression:
  $P_w(y = 1|x) = \frac{1}{1+e^{-w^T \phi(x)}} = \sigma(w^T \phi(x))$

- Training: Learn w^* given training data D

- Testing: $P_w(y^n = 1|x^n) = \frac{1}{1+e^{-w^{*T} \phi(x^n)}}$

- Assign $P > 0.5 \rightarrow class1$, $P \leq 0.5 \rightarrow class0$

- Training via MLE:
  $\max_w P_w(D) = \max_w P_w(y_1|x_1) \cdots P_w(y_N|x_N)$
  $= \max_w \prod_{i=1}^{N} P_w(y_i|x_i)$
  We can write :
  $P_w(y_i|x_i) = P_w(y_i = 1|x_1)^{y_i} P_w(y_i = 0|x_1)^{1-y_i}$
  Apply natural log:
  $\max_w log P_w(D) = \max_w \sum_{i=1}^{N} log[(\frac{1}{1+e^{-w^T \phi(x_i)}})^{y_i} + (\frac{1}{1+e^{w^T \phi(x_i)}})^{1-y_i}]$
  $= \max_w \sum_{i=1}^{N} (y_i) log(\frac{1}{1+e^{-w^T \phi(x_i)}}) + (1 - y_i) log(\frac{1}{1+e^{w^T \phi(x_i)}})$

$= \max_w \sum_{i=1}^N (y_i)[log(\frac{1}{1+e^{-w^T\phi(x_i)}}) - log(\frac{1}{1+e^{w^T\phi(x_i)}})] + log\frac{1}{1+e^{w^T\phi(x_i)}}$
$= \max_w \sum_{i=1}^N (y_i)[log(\frac{1+e^{w^T\phi(x_i)}}{1+e^{-w^T\phi(x_i)}})] + log\frac{1}{1+e^{w^T\phi(x_i)}}$
$\max_w \sum_{i=1}^N (y_i)[log(e^{w^T\phi(x_i)})] + log\frac{1}{1+e^{w^T\phi(x_i)}}$
$= \max_w \sum_{i=1}^N (y_i w^T\phi(x_i)) - log(1+e^{w^T\phi(x_i)})$
$\equiv \min_w \sum_{i=1}^N -y_i w^T\phi(x_i) + log(1+e^{w^T\phi(x_i)})$
Derivative:
$\frac{\partial J}{\partial w}$

DS4400 Notes 02/11

1. REVIEW: logistic model: $P_w(y=1|x) = \frac{1}{1+e^{-w^T\phi(x)}}$
   MLE to learn w:
   $\ell(x) = logP_w(D) = log\prod_{i=1}^N P_w(y_i|x_i)$
   $= \sum_{i=1}^N [y_i\phi(x_i)^T w - log(1+e^{w^T\phi(x_i)})]$ maximizing $\ell(w) \equiv$ minimizing $-\ell(w)$. Then $\min_w -\ell(w)$
   $= \min_w \sum_{i=1}^N -y_i w^T\phi(x_i) + log(1+e^{w^T\phi(x_i)})$
   $= \min_w -y_i\phi(x_i)^T w + log(1+e^{w^T\phi(x_i)})$
   derivative:
   $\frac{\partial -\ell(w)}{\partial w} = \sum_i -y_i\phi(x_i) + \frac{1}{1+e^{w^T\phi(x_i)}}(e^{w^T\phi(x_i)})(\phi(x_i))$
   $= \sum_i -y_i\phi(x_i) + \frac{e^{w^T\phi(x_i)}\phi(x_i)}{1+e^{w^T\phi(x_i)}}$
   $= \sum_i -y_i\phi(x_i) + \frac{\phi(x_i)}{1+e^{-w^T\phi(x_i)}}$
   $= \sum_i (-y_i + \frac{1}{1+e^{-w^T\phi(x_i)}})\phi(x_i)$
   No closed form solution for = 0.

2. GD of logistic regression
   - Initialize w^0
   - For t = 1, 2, ... (until converge)
     - $w^t = w^{t-1} - \rho\frac{\partial J}{\partial w}|_{t-1}$
       $= w^{t-1} - \rho\sum(-y_i + \frac{1}{1+e^{-w^T\phi(x_i)}})\phi(x_i)|_{t-1}$

3. overfitting.
   overfitting: do well on training but poorly on testing.
   Symptom: w with large entries.

4. Regularized logistic regression:
   $min_w J(w) = -\ell(w) + \frac{\lambda}{2}||w||_2^2$
   GD: $\frac{\partial J + \frac{\lambda}{2}||w||_2^2}{\partial w} = \frac{\partial J}{\partial w} + \lambda w$
   $= \sum_i (-y_i + \frac{1}{1+e^{-w^T\phi(x_i)}})\phi(x_i) + \lambda w^{t-1}$

5. clastering more than 2
   one of the methods: Just creating n models for n type of data. each model is a i vs rest.
   For each model we have:
   $$P(y=i|x) = \sigma(w_i^T x)$$

Then see which have the max probability.

$$y^{test} = \underset{i \in \{0,1,\ldots,N\}}{\arg\max}\ \sigma(w_i^T \phi(x^{test}))$$

6. MAXIMUM a Posteriori (MAP) Estimation:
   Incorporating with prior knowledge with parameters. When the data is not enough and we have some prior knowledge, we do MAP.
   MAP setting:
   - we start with a "prior" model on parameters of systems $\rightarrow P_{prior}(\theta)$
   - we observe a dataset D $\rightarrow P_\theta(D)$
   - Given D, how the prior knowledge on θ changes $\rightarrow P(\theta|D)$

   MAP: $\max_\theta P(\theta|D) \rightarrow \hat{\theta_{MAP}}$

DS4400 Notes 02/14
Maximum A Posteriori (MAP) Estimation:
Incoperate prior knowledge into system(parameter) learning

1. Exercise: bernoulli expriment: $\theta = P(x=1) \rightarrow \hat{\theta}_{MLE}$ can be learned from training set $X_1 = x_1 \ldots$
   Tossing a coin : $\theta = P(X=1) = P('H'), D = H, H \rightarrow \hat{\theta}_{MLE} = \frac{2}{2} = 1$
2. MAP setting:
   - Put a prior distribution on θ (that encodes prior knowledge / domain expertise)
   - Observe a data set $D = \{X_1 = x_1, \ldots, x_N = N\} \rightarrow P(D|\theta) = P_\theta(D)$
   - How much our knowledge about θ changes after seeing data, D: $P(\theta|D) \rightarrow$ posterior dist.
     MAP: $\max\limits_\theta P(\theta|D) \equiv \frac{P(D|\theta)P(\theta)}{P(D)} = \max\limits_\theta P(D|\theta)P(\theta) = \max\limits_\theta L(\theta)P(\theta)$
3. Exercise:
   $X_1 = x_1, X_2 = x_2 \ldots, X_N = x_N, x_i \in \{0,1\}$
   $P(x_i = 1) = \theta, p(x_i = 0) = 1 - \theta$
   Using Beta distribution: $P_{\alpha,\beta}(\theta) \propto \theta^{\alpha-1}(1-\theta)^{\beta-1}$
   MAP: $\max_\theta P(\theta|D) = \max P(x_1, x_2, \cdots, x_N|\theta)P_{\alpha,\beta}(\theta)$
   $= \prod\limits_{i=1}^{N} P(X_i = x_i)P_{\alpha,\beta}(\theta)$
   $= \theta^{\sum x_i}(1-\theta)^{N-\sum x_i}\theta^{\alpha-1}(1-\theta)^{\beta-1}$
   $= \theta^{\sum x_i + \alpha - 1}(1-\theta)^{N-\sum x_i + \beta - 1}$
   Let $\alpha' = \sum x_i + \alpha - 1$, $\beta' = N - \sum x_i + \beta - 1$
   Then $P(\theta|D) \propto P_{\alpha',\beta'}(\theta)$
   $\max\limits_\theta P(\theta|D) = \max\limits_\theta P_{\alpha',\beta'}(\theta) = \frac{\alpha'-1}{\alpha'+\beta'-2}$
   $\hat{\theta}_{MAP} = \frac{\alpha'-1}{\alpha'+\beta'-2} = \frac{\sum x + i + \alpha - 1}{N+\alpha+\beta-2}$
   $= \frac{N}{N+\alpha+\beta-2}\frac{\sum x_1}{N} + \frac{\alpha+\beta-2}{N+\alpha+\beta-2}\frac{\alpha-1}{\alpha+\beta-2}$
   $= \eta\hat{\theta}_{MLE} + \eta\hat{\theta}_{prior\ mode}$

4. Conclusion:
   $N \to \infty \Rightarrow \hat{\theta}_{MAP} \sim \hat{\theta}_{MLE}$
   $N \to 0 \Rightarrow \hat{\theta}_{MAP} \sim \hat{\theta}_{prior\ mode}$
   having prior mode is just adding fake observations generated by prior mode. The bigger the $1-\eta$, the more fake observations we add to dataset

5. MAP on logistic regression:
   $\to \min_\omega J(\omega)$. We might have θ of very large terms.
   Then choose $p(\omega) \propto e^{-\omega^T\omega/2\sigma^2} = e^{-\|\omega\|_2^2/2\sigma^2}$
   $\max\limits_\omega P(\omega|D) \propto \max\limits_\omega P(D|\omega)P(\omega)$
   Then $\max_\omega log(P(\omega|D)) \propto \max\limits_\omega logP(D|\omega) + logP(\omega)$
   $= \max\limits_\omega J(\omega) - \frac{1}{2\sigma^2}\|\omega\|_2^2$
   $\equiv \min\limits_\omega -J(\omega) + \lambda\|\omega\|_2^2$

DS4400 Notes 02/18
Classification

1. Discriminative Modeling

   Find a decisim boundary that seperates data into classes
   e.g logistic regression

   Discriminative approacheds model:
   $P(y|x)$, y is class, x is feature vector.
   e.g. $P(y=1|x) = \sigma(w^T\phi(x)) = \frac{1}{1+e^{-w^T\phi(x)}}$

2. Generative Modeling

   Model distribution of data in each class as well as the distribution of classes themselves.

   $\to P(x|y)$(Feature of class) and $P(y)$(class).

   - Assume we learn $P(x|y), P(y)$ during training
   - How to classify a new test sample x^+?
     $\Rightarrow \underset{j=0,1,\ldots,L-1}{\arg\max}\ P(y=j|x^t) = \underset{j=0,1,\ldots,L-1}{\arg\max}\ \frac{P(x^t|y=j)P(y=j)}{P(x^t)} \equiv \underset{j=0,1,\ldots,L-1}{\arg\max}\ P(x^t|y=j)P(y=j)$

3. Example: email classification: $\{(x^1, y^1), \ldots, (x^N, y^N)\}$
   probable $x: \begin{pmatrix} "CPAS" \\ "Free" \\ "Call\ now" \end{pmatrix} y = \{"non-spam", "spam"\}$
   Parameters to learn are:

$$\theta_0^y \triangleq P(y=0) = P(`non-spam')$$

$$\theta_1^y \triangleq P(y=1) = P(`spam')$$

$$\theta_{\bar{x}|0}^{x|y} \triangleq P(x=\bar{x} \mid y=1) = P(x=\bar{x} \mid `non-spam')$$

$$\theta_{\bar{x}|1}^{x|y} \triangleq P(x=\bar{x} \mid y=1) = P(x=\bar{x} \mid `spam')$$

More generally, $\Theta \triangleq$

$$\begin{cases} \theta_j^y \triangleq P(y=j),\ \forall j = 0,1,\ldots,L-1 \\ \theta_{\bar{x}|j}^{x|y} \triangleq P(x=\bar{x} \mid y=j),\ \forall x=\bar{x}, \forall j = 0,1,\ldots,L-1 \end{cases} \tag{1}$$

4. **Approach: MLE**:
   MLE: $L(\theta) = P_\Theta(x1, y1, \ldots, x^N, y^N) =_{iid} = \prod_j P_\Theta(x^i, y^i) = \prod_j P_\Theta(x^i|y^i)P(y^i)$
   $P(y^i) = P(y^i=0)^{1(y^i=0)} \cdot P(y^i=0)^{1(y^i=0)} \cdot \ldots P(y^i=0)^{1(y^i=0)}$
   $= \theta_0^{1(y^i=0)} \theta_0^{1(y^i=1)} \ldots \theta_0^{1(y^i=L-1)}$
   $L(\theta) = \prod_i P(x^i|y^i) \prod_i \prod_{j=0}^{L-1} P(y^i=j)^{1(j^i=j)}$
   $= \prod_i P(x^i|y^i) \prod_i \prod_{j=0}^{L-1} \theta_j^{y\,1(j^i=j)}$
   $\Rightarrow logL(\theta) = \sum_i logP(x^i|y^i) \sum_i \sum_{j=0}^{L-1} 1(j^i=j) log(\theta_j^y)$
   To estimate
   $\hat{\theta}_j^y \Rightarrow \frac{\partial LogL(\theta)}{\partial \theta_j^y} = 0 \Rightarrow \hat{\theta_j^y} = \frac{\sum_i^N 1(y^i=j)}{N}$
   $\hat{\theta}_{\bar{x}|j}^{x|y} = \frac{\sum_i 1(x^i=\bar{x}, y^i=j)}{\sum_{i=1}^N 1(y^i=j)}$
   This is called **Vanilla Generative Model**.
   $\theta_j^y \Rightarrow L$ estimations
   $\theta_{\bar{x}|j}^{x|y} \Rightarrow Lm^d$ estimations
   (given x has d dimension and each dimension has m values) i.e. $x = \begin{pmatrix} 0,1,2,\ldots,m-1 \\ 0,1,2,\ldots,m-1 \\ \vdots \\ 0,1,2,\ldots,m-1 \end{pmatrix}_{d\times 1}$

5. Problem: Document Classifications:
   length of doc: $|DOC|$, we have possiblly $|DOC|$ features, each feature may have $|DOC|$ of possible values. Then the estimation is $L \cdot |DOC|^{|DOC|}$

6. Naive Bayes Method:
   Generative model where feature are independent for a particular given class.
   $P(x=\bar{x}|y=j)$ where x has d features.
   e.g. spam classifications: $x = \begin{pmatrix} 'free' \\ 'caps' \\ 'call\ now' \end{pmatrix}, y = 0,1$
   $P(x=(1,1,1)^T|y=1) = P('free'=1|y=1)P('caps'=1|y=1)P('call\ now'|y=1)$
   class conditional independence.
   $O(Lmd)$

Generative Modeling

Classification:

- Discriminative: $P(y|x) = \frac{1}{1+e^{-w^t\phi(x_i)}}$
- Generative: $P(x|y), P(y)$ to see which normal distribution generates the data.

Learning to firgure out parameter of $p(x|y), p(y)$:

- $\theta_j^y \triangleq P(y=j)$
- $\theta_{\bar{x}|y}^{x|y} \triangleq P(x=\bar{x}|y=j)$

Estimate data using $D = \{(x_1, y_1), (x_2, y_2)\ldots, (x_L, y_L)\}$
Use MLE:
Training time: $O(Lm^d)$, d is dimension of feature, m is number of possible values of each feature.
Naive Bayes assumption:
features are independent given a class.
$P(x=\bar{x}|y=j) = P(x_1=\overline{x_1}|y=j)P(x_2=\overline{x_2}|y=j)P(x_3=\overline{x_3}|y=j)\ldots P(x=\overline{x_d}|y=j)$
$\equiv \theta_{\bar{x}|y}^{x|y} = \theta_{\overline{x_1}|j}^{x_1|y}\theta_{x_2|j}^{\overline{x_2}|y}\ldots\theta_{x_d|j}^{\overline{x_d}|y}$
$\hat{\theta}_j^y = \sum_{i=1}^L 1(y^i=j)/L$
$\hat{\theta}_{\bar{x}|j}^{x|y} = \frac{\sum\limits_{i=1}^{L} 1(x_i=\bar{x}, y_i=j)}{\sum\limits_{i=1}^{L} 1(y_i=j)}$
Total running time: $O(Lmd)$ However, **Unseen cases is going to lead to 0 probability**
We need to put some "fake data" in the data set:
$\hat{\theta}_{\overline{x_l}|j}^{x_l|y} = \frac{\sum(x_l^i=\overline{x_l}, y_i=)+t}{\sum 1(y_i=j)+tm}$
For each case, we added t fake data, we need to add tm on the denominator since totally we added tm data entry.
Gaussian Naive Bayes
$x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \end{pmatrix}$ Each x_i is a gaussian distribution: (μ_i, σ_i^2)
$\mu_{1,j} = \frac{\sum x_i 1(y_i=j)}{\sum 1(y_i=j)}$
$\mu_{l,j} = \frac{\sum x_l 1(y_i=j)}{\sum 1(y_i=j)}$
$\sigma_{l,j} = \frac{\sum(x_l^i-\mu_{l,j})^2 1(y_i=j)}{\sum 1(y_i=j)}$

Convex Set Definition: : A set $S \subseteq R^d$ is convex iff $\forall x_1, x_2 \in S, \forall \alpha \in [0,1]$, we have $\alpha x_1 + (1-\alpha)x_2 \in S$