Tuan Tran
A20357888
CS 584- Summer 2020

⊛ Numerical Solutions: (NS)

1, Numerical solutions are necessary and beneficial when ~~the opt~~:

+, The optimization problem ~~dose~~ doesn't have an exact closed form solution (ex: non-linear gradient ~~~~)

+, Matrix is too large to invert -> Computationally expensive

2,

- Gradient Descent is the algo where we follow the gradient to descent the ~~loss~~ curve and arrive at the point θ^* where our objective function is minimized.

This is possible since the property of gradient is that the gradient of a function points in the direction of <u>max</u> change

-> negative gradient is the direction of max decrease.

↳ We can update our params:

$$\theta^{(i+1)} \leftarrow \theta^{(i)} - \eta \nabla J(\theta^{(i)}).$$

- Stop Condition is when the value of loss function doesn't change past some small threshold: $(J(\theta^{(i+1)}) - J(\theta^{(i)})) < \epsilon$

3,

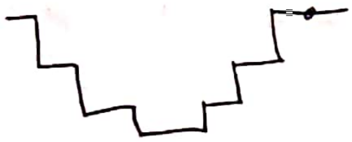
- learning rate is the size of the step ~~we take~~ take along direction of gradient
- learning rate is a hyperparameter -) we have to experiment to set it correctly

- When learning rate is small, we are likely to hit minimum point but ~~Converge~~ learning is slower since we are taking small stepsize

When ~~learn~~ Lr is large, we are taking larger stepsize and thus will descent faster but we might overshoot the minimum point

4,

- GD can't be applied to piecewise constant functions since It's easy to hit a point where grad is 0:



- GD is guaranteed to produce global min if objective function is convex
- We can attempt to avoid local minimum by ~~tt~~ introducing momentum

-> Idea is to use residual velocity from previous descent to move over local minimum.

5,

Loss Func for linear Regression:

$$J(\theta) = \frac{1}{2}(Z\theta - Y)^T(Z\theta - Y)$$

$$\rightarrow \nabla J(\theta) = Z^T(Z\theta - Y)$$

-> Update Equation:

$$\theta^{(i+1)} \leftarrow \theta^{(i)} - \eta Z^T(Z\theta - Y)$$

-> Summation Form:

$$\theta^{(i+1)} \leftarrow \theta^{(i)} - \eta \sum_{j=1}^{m} (z^{(j)T}\theta - y^{(j)}) z^{(j)}$$

6,

Basic GD computes gradient over all training examples

SGD computes gradient over a batch of examples only

↳ Basic GD:

+, Accurate, stable

+, Computationally expensive

SGD:

+, Converge faster but not as accurate and stable

+, less computationally expensive.

7.

$$\theta^{(i+1)} \leftarrow \theta^{(i)} - \eta \left(z^{T}\theta - y\right) z$$

$$\leftarrow \begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix} - \frac{1}{6} \cdot \left(\begin{bmatrix} 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix} - 2 \right) \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

$$\leftarrow \begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix} - \frac{1}{6} (6) \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

$$\leftarrow \begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

$$\rightarrow \theta^{(i+1)} = \begin{bmatrix} 2 \\ 4 \\ 4 \end{bmatrix}$$

8,

Newton's method update equation:

$$\theta^{(i+1)} \leftarrow \theta^{(i)} - H^{-1} \nabla J(\theta^{(i)})$$

H^{-1} = inverse Hessian ~~et~~ is equivalent to the learning rate in GD

- Advantage of Newton's method:
  + Can compute learning rate instead of specifying it as ~~per~~ hyper param.
  + We get different learning rate for different features
- Disadvantage:
  + Hessian computation is expensive and may be noisy
  + Sometimes cannot invert H.

9,

$$J(\theta) = \frac{1}{2}(Z\theta - Y)^T(Z\theta - Y)$$

$$\rightarrow \nabla J(\theta) = Z^T(Z\theta - Y)$$

$$H = \nabla(\nabla J(\theta)) = Z^T Z.$$

$$\rightarrow \boxed{\theta^{(i+1)} \leftarrow \theta^{(i)} - (Z^T Z)^{-1} Z^T (Z\theta - Y)}$$

10,

The line search first ~~first~~ finds the descent direction using GD and the computes a stepsize that determines how ~~much~~ far to move along that direction ~~and the~~ iteratively.

GDA:

1,

$$f(x; \mu; \Sigma) = \frac{1}{(2\pi)^{n/2} |\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)}$$

- ~~Cova~~ Σ gives the variance along ~~d~~ the diagonal and covariance between features ~~et oth~~ otherwise

- when features are uncorrelated, Σ becomes a diagonal matrix since covariance between 2 uncorrelated variables = 0

$$\rightarrow \Sigma = \begin{bmatrix} \sigma_1^2 & & 0 \\ & \ddots & \\ 0 & & \sigma_n^2 \end{bmatrix}$$

2,

- ~~Mahala~~ Euclidean assumes all directions are equally important
  ↳ mahalanobis is basically euclidean normalized by inverse of Σ

  → It gives different weights $\left(\frac{1}{\text{variance}}\right)$ to different direction when computing the distance. It accounts for the variance (uncertainty in different directions.

3,

$$x = \begin{bmatrix} 0 \\ 1 \end{bmatrix} ; \mu = \begin{bmatrix} 2 \\ 3 \end{bmatrix} ; \Sigma = \begin{bmatrix} 4 & 0 \\ 0 & 5 \end{bmatrix} \rightarrow \Sigma^{-1} = \begin{bmatrix} 1/4 & 0 \\ 0 & 1/5 \end{bmatrix}$$

$$\rightarrow \text{Mahalanobis dist} = \left(\begin{bmatrix} 0 \\ 1 \end{bmatrix} - \begin{bmatrix} 2 \\ 3 \end{bmatrix}\right)^T \cdot \begin{bmatrix} 1/4 & 0 \\ 0 & 1/5 \end{bmatrix} \cdot \left(\begin{bmatrix} 0 \\ 1 \end{bmatrix} - \begin{bmatrix} 2 \\ 3 \end{bmatrix}\right)$$

$$= \begin{bmatrix} -2 & -2 \end{bmatrix} \begin{bmatrix} 1/4 & 0 \\ 0 & 1/5 \end{bmatrix} \begin{bmatrix} -2 \\ -2 \end{bmatrix}$$

$$= \begin{bmatrix} -1/2 & -2/5 \end{bmatrix} \begin{bmatrix} -2 \\ -2 \end{bmatrix} = 1 + \frac{4}{5} = \frac{9}{5}$$

4,

In generative learning, we model the feature distribution in each class $P(X|Y)$ and class prior $P(Y)$

→ We can generate more examples given the class wanted

~~tut~~ We have posterior prob for each class:

$$p(y=k|x) = \frac{P(x|y=k)\,P(y=k)}{\sum_i P(x|y=i)\,P(y=i)}$$

→ Since when comparing $P(y=k|X)$ for different values of k, we only compare the numerator ~~finte~~ as denominator is constant

→ We can define discriminant function as log of numerator.

5,

- Bayes rule helps describe ~~find~~ the ~~poste~~ prob of an event based on prior knowledge of conditions that may be ~~rat~~ related to the event.

→ In our case:

$$P(y=j|X) = \frac{P(X|y=j)\,P(y=j)}{\sum_{i=1}^{K} P(x|y=i)\,P(y=i)}$$

where $P(y=j|X)$ = posterior

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

$P(y=j)$ = prior.

- Evidence can be ignored since it is constant ~~for all~~ For $P(y=i|X)$ for all $i=1\ldots k$.

6,

For class k:

$$\mu_k = \frac{1}{m_k} \sum_{i=1}^{m} 1(y^{(i)}=k)\,x^{(i)}$$

$$\Sigma_k = \frac{1}{m_k} \sum_{i=1}^{m} 1(y^{(i)}=k)\,(x^{(i)}-\mu_k)(x^{(i)}-\mu_k)^T$$

$$p(y=k) = \frac{1}{m} \sum_{i=1}^{m} 1(y^{(i)}=k)$$

7,

- ~~we~~ we have for class k:

$$P(y=k|X) = \frac{P(X|y=k)P(y=k)}{\sum_{i\in\alpha} P(X|y=i)P(y=i)}$$

→ denominator doesn't change

→ discriminant func: $g_k(x) = \log\left(P(X|y=k)P(y=k)\right)$

- ~~S+A~~ Since we choose gaussian

$$\to P(X|y=k) = \frac{1}{\sqrt{2\pi}\,\sigma_k} e^{-\frac{1}{2\sigma_k^2}(x-\mu_k)^2}$$

$$\to g_k(x) = \log\left(\frac{1}{\sqrt{2\pi}\,\sigma_k} e^{\frac{-1}{2\sigma_k^2}(x-\mu_k)^2}\right) + \log\left(P(y=k)\right)$$

$$= \underbrace{\log\left(\frac{1}{\sqrt{2\pi}}\right)}_{\text{constant}} - \log(\sigma_k) - \frac{1}{2\sigma_k^2}(x-\mu_k)^2 + \log\left(P(y=k)\right)$$

$$\to g_k(x) = -\log(\sigma_k) - \frac{1}{2\sigma_k^2}(x-\mu_k)^2 + \log P(y=k)$$

- Equal priors and variances

$$\to \left.\begin{array}{l} \sigma_k \to \sigma \\ P(y=k) = \alpha_k \;\forall\; \alpha \end{array}\right\} \to \text{constant}$$

$$\to g_k(x) = -\underbrace{\frac{1}{2\sigma^2}}_{\text{identical scale value for all classes}}(x-\mu_k)^2$$

$$\to g_k(x) = (x-\mu_k)^2$$

→ $\boxed{\text{Simplifies down to choosing class with closest mean}}$

8,

- Model for class k:

$$P(X|y=k) = \frac{1}{(2\pi)^{n/2} |\Sigma_k|^{1/2}} e^{-\frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1} (x-\mu_k)} \quad ; \alpha_k = P(y=k)$$

Similar to before, we have $g_k(x) = \log(P(X|y=k)) + \log \alpha_k$.

$$\rightarrow g_k(x) = \underbrace{-\log(2\pi^{n/2})}_{\text{Constant}} - \frac{1}{2}\log(|\Sigma_k|) - \frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1} (x-\mu_k) + \log \alpha_k$$

$$\rightarrow g_k(x) = -\frac{1}{2}\log(|\Sigma_k|) - \frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1} (x-\mu_k) + \log \alpha_k.$$

- · Equal covariance and priors:
  - → Similar to ~~before~~ question 7, we can simplify down to:

    $g_k(x) = \|x - \mu_k\|^2$

    → euclidean distance to mean.

9,

- IID assumption means that ~~instea~~ instances are independent and follows the same distribution.

- ~~Model for k class:~~

  ~~$p(x|y=k) =$~~

  $L(\theta) = P(x^{(1)} \dots x^{(m)}; \theta) = \prod_{i=1}^{m} P(x^{(i)}; \theta)$

  where $P(x^{(i)}; \theta) = \frac{1}{\sqrt{2\pi}\sigma_k} e^{-\frac{1}{2}\frac{(x-\mu_k)^2}{\sigma^2}}$

  $\rightarrow \ell(\theta) = \sum_{i=1}^{m} \log P(x^{(i)}; \theta)$

$$\rightarrow \ell(\sigma_k) = \sum_{i=1}^{m_k} \log\left(\frac{1}{\sqrt{2\pi}\sigma_k} e^{-\frac{1}{2}\cdot\frac{(x^{(i)}-\mu_k)^2}{\sigma_k^2}}\right)$$

$$\rightarrow \ell(\sigma_k) = m\log\frac{1}{\sqrt{2\pi}} - m\log\sigma_k - \frac{1}{2\sigma_k^2}\sum_{i=1}^{m_k}(x^{(i)}-\mu_k)^2$$

$$\nabla\ell(\sigma_k) = 0 \rightarrow \begin{bmatrix} \frac{\partial \ell}{\partial \mu_k} \\ \frac{\partial \ell}{\partial \sigma_k} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

+, For μ_k:

$$\frac{\partial \ell}{\partial \mu_k} = -\frac{1}{2\sigma_k^2}\sum_{i=1}^{m_k} 2(x^{(i)}-\mu_k)(-1)$$

$$= 0$$

$$\rightarrow \sum_{i=1}^{m_k} (x^{(i)}-\mu_k) = 0$$

$$\rightarrow \sum_{i=1}^{m_k} x^{(i)} = m_k\mu_k$$

$$\rightarrow \mu_k = \frac{1}{m_k}\sum_{i=1}^{m_k} x^{(i)}$$

+, For σ_k:

$$\frac{\partial \ell}{\partial \sigma_k} = -m\frac{1}{\sigma_k} - \frac{1}{2}(-2)\frac{1}{\sigma_k^3}\sum_{i=1}^{m}(x^{(i)}-\mu_k)^2 = 0$$

$$\rightarrow \sigma_k^2 = \frac{1}{m_k}\sum_{i=1}^{m_k}(x^{(i)}-\mu_k)^2.$$

* Naive Bayes:

1,

Naive Bayes assumption is that we assume ~~the~~ features are independent

$$\rightarrow P(X) = P(x_1 \ldots x_n) = \prod_{i=1}^{n} P(x_i)$$

~~This as~~ Motivation is that such assumption leads to a simpler model since it reduces number of parameters to learn

2,

For Bernoulli NB :

$$P(x_j^{(i)} \mid y=1) = \alpha_{j|y=1}^{x_j^{(i)}} (1-\alpha_{j|y=1})^{1-x_j^{(i)}}$$

$\alpha_{j|y=1}$ = prob that word $\underline{\underline{j}}$ is in document $\underline{\underline{i}}$ of class $\underline{\underline{y=1}}$

3, For class ~~k~~ y=1:

$$- \ell(\theta) = \log P(x^1 \dots x^{m_1}; \theta)$$

$$= \log \prod_{i=1}^{m_1} P(x^i; \theta)$$

$$= \log \prod_{i=1}^{m_1} P(x_1^i, x_2^i \dots, x_n^i; \theta)$$

$$= \log \prod_{i=1}^{m_1} \prod_{j=1}^{n} P(x_j^{(i)}; \theta)$$

↳ NB assumpotion

$$\rightarrow \ell(\theta) = \sum_{i=1}^{m_1} \sum_{j=1}^{n} \log P(x_j^{(i)}; \theta)$$

$$= \sum_{i=1}^{m_1} \sum_{j=1}^{n} x_j^{(i)} \log(\alpha_{j|y=1}) + (1-x_j^{(i)}) \log(1-\alpha_{j|y=1})$$

$$- \frac{\partial \ell}{\partial \alpha_{j|y=1}} = \sum_{i=1}^{m_1} x_j^{(i)} \cdot \frac{1}{\alpha_{j|y=1}} + (1-x_j^{(i)}) \cdot \frac{1}{1-\alpha_{j|y=1}} \cdot (-1) = 0$$

$$\rightarrow \frac{1}{\alpha_{j|y=1}} \sum_{i=1}^{m_1} x_j^{(i)} = \frac{1}{1-\alpha_{j|y=1}} \sum_{i=1}^{m_1} (1-x_j^{(i)})$$

$$\rightarrow \boxed{\alpha_{j|y=1} = \frac{1}{m_1} \sum_{i=1}^{m_1} x_j^{(i)}}$$

4, we define ~~dis~~ $g_k(x)$ for class k $= \prod_{j=1}^{n} \alpha_{j|y=k}^{x_j} (1-\alpha_{j|y=k})^{1-x_j}$

$$\rightarrow \hat{y} = \arg\max_k g_k(x)$$

5,

Bernoulli doesn't give us useful info such as frequency of words.

→ with Binomial, we can get each feature as frequency: how many times the word occurs in the document.

6,

~~First we~~

1) Select binomial ~~mode~~ model for $P(X|y)$

2) Use MLE or MAP to find parameters of the binomial model

3) Derive discriminant function

4) $\hat{y} = \arg\max_k g_k(x)$.

7,

- ~~Un For class y = k~~ we have following ~~model~~ log-likelihood:

$$\ell(\theta) = \log \prod_{i=1}^{m} \cancel{P(y^{(i)}, x^{(i)})}\ P(x^{(i)};\theta)$$
$$= \log \prod_{i=1}^{m} \prod_{j=1}^{n} P(x_j^{(i)};\theta)$$
$$= \sum_{i=1}^{m} \sum_{j=1}^{n} \log \binom{d^{(i)}}{x_j^{(i)}} \alpha_{j|y=y^{(i)}}^{x_j^{(i)}} \left(1-\alpha_{j|y=y^{(i)}}\right)^{(d^{(i)}-x_j^{(i)})}$$

$$\rightarrow \frac{\partial \ell}{\partial \alpha_{j|y=k}} = 0$$

$$\rightarrow \cancel{\alpha_{j|y} = \alpha_{j|y=j}} =$$

$$\alpha_{j|y=k} = \frac{\sum_{i=1}^{m} 1(y^{(i)}=k)\, x_j^{(i)} + \epsilon}{\sum_{i=1}^{m} 1(y^{(i)}=k)\, d^{(i)} + k\epsilon}$$ Laplace smoothing

- Laplace smoothing is in place to prevent cases when a word j may not appear in any of the documents → $P(x|y) = 0$

membership func used during classification for class k:

$$g_k(x) = \sum_{j=1}^{n} \left[\log \binom{d}{x_j} + x_j \log(\phi_{j|y=k}) + (d - x_j) \log(1 - \phi_{j|y=k}) \right] + \log \phi_k$$

8,

- ~~Due~~ with gaussian feature vectors:

$$g_k(x) = -\frac{1}{2} \log |\Sigma_k| - \frac{1}{2} (x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k) + \log \phi_k .$$

where $\mu_k = \frac{1}{m_k} \sum_{i=1}^{m} 1(y^{(i)} = k) \, x^{(i)}$

$$\Sigma_k = \frac{1}{m_k} \sum_{i=1}^{m} 1(y^{(i)} = k) \, (x^{(i)} - \mu_k)(x^{(i)} - \mu_k)^T .$$

- Due to NB assumption, Σ_k becomes a diagonal matrix since cov of 2 uncorrelated variables $= 0$.
  → instead of n^2 params in Σ_k, we only have n params.

9,

- Precision $= \frac{\text{True Positive}}{TP + FP}$ = Out of all positive prediction → the ~~pct~~ percent (pct) of actual positive

Recall $= \frac{TP}{TP + FN}$ = Out of all actual positive → the pct of ~~predicte~~ true ~~pr~~ positive prediction.

Sensitivity = True positive rate

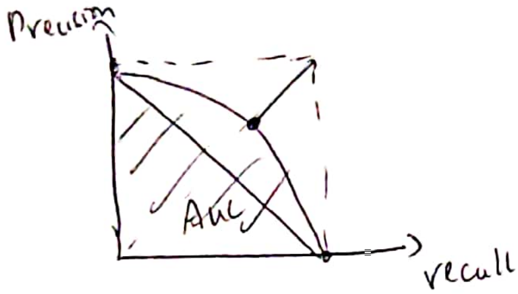
Specificity = False positive rate

Accuracy = ~~percentage~~ number of correct predictions over all instances

$$= \frac{TP + TN}{TP + TN + FP + FN}$$

- The F-measure uses harmonic mean to combine and ~~avg~~ "average" precision and recall to account for their trade off

- The higher the area under ~~the~~ curve in precision - recall curve, the better



- k classes :

$$\text{accuracy} = \frac{\sum C_{ii}}{\sum\sum C_{ij}}$$

$$\text{precision for class } i = \frac{C_{ii}}{\sum_j C_{ij}}$$

$$\text{recall} \;\text{———}\; = \frac{C_{ii}}{\sum_i C_{ij}}$$

$$\rightarrow \text{F-measure} \;\text{———}\; \cancel{\text{ftogs}} = 2\,\frac{P \times R}{P + R}.$$