CPSC340A4

Qinglan Huang n6v9a Liu Yang v4d9

November 20, 2017

1 Multi-Class Logistic

1.1 Softmax Classification

1. Two columns represent the number of the feature, and the row represent the number of the classes that each examples corresponding.

2.$\hat{x}$ =[1 1]
$W_1^T\hat{x} = \begin{pmatrix} 2 & -1 \end{pmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = 1$
$W_2^T\hat{x} = \begin{pmatrix} 2 & 2 \end{pmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = 4$
$W_3^T\hat{x} = \begin{pmatrix} 3 & -1 \end{pmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = 2$
Thus we choose w2 in this case.

1.2 Softmax Loss

$p(y_i|w, x_i) = \frac{exp(wy_i^T x_i)}{\sum_{c'=1}^{k} exp(w_c^T x_i)}$

Loss function:

$-log(p(y_i|w, x_i))$

$= -logexp(wy_i^T x_i) + log \sum_{c'=1}^{k} exp(w_c^T x_i)$

$= -w_{y_i}^T x_i + log \sum_{c'=1}^{k} exp(w_c^T x_i)$

Derivative:
$\frac{d}{dw_{jc}}(-log(p(y_i|w, x_i)))$

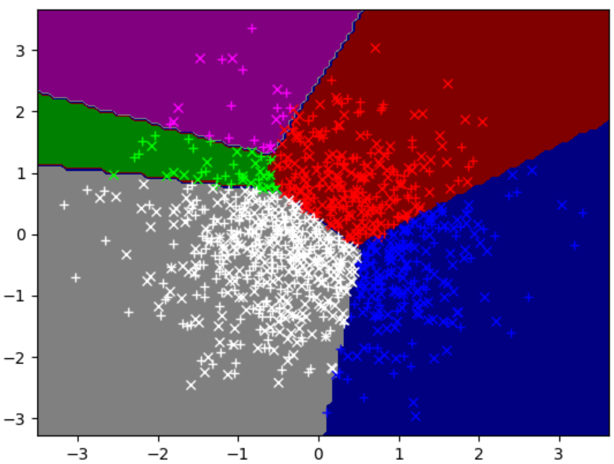
$= \frac{d}{dw_{jc}}(-w_{y_i}^T x_i) + \frac{d}{dw_{jc}}(log \sum_{c'=1}^{k} exp(w_c^T x_i)$

$$= -I(y_i == c)x_i + \frac{exp(w_c^T x_i)x_{ij}}{\sum_{c'=1}^{k} exp(w_c^T x_i)}$$

1.3 Softmax Classifier

```
function softmaxClass(X,y)
    (n,d) = size(X)
    k = maximum(y)

    # Each column of 'w' will be a logistic regression classifier
    W = zeros(d,k)

    # Each binary objective has the same features but different lables
    funObj(w) = SoftObj(w,X,y)


    w = findMin(funObj,reshape(W,d*k),verbose=false, derivativeCheck =true)

    W= reshape(w,d,k)
    # Make linear prediction function
    predict(Xhat) = mapslices(indmax,Xhat * W,2)

    return LinearModel(predict,W)
end

function SoftObj(w,X,y)
    (n,d) = size(X)
    k = maximum(y)


    sumlog0 = 0
    sumlog1 = 0
    f = 0
    W = reshape(w,d,k)

    for i in 1:n
        f += -(W[:,y[i]]'*X[i,:])
        for c in 1: k
            sumlog0 += exp.(W[:,c]'*X[i,:])
        end
        f += log(sumlog0)
        sumlog0  = 0
    end

    g0 = zeros(d,k)
    g1 = zeros(n,d)

    for c in 1: k
        for i in 1: n
            sumlog1 = sum(exp.(W'*X[i,:]))
            g1[i,:] = -X[i,:]*(y[i]==c) + (exp.(W[:,c]'* X[i,:])) * X[i,:]/sumlog1
        end
        g0[:,c] = sum(g1,1)'
    end
```

```
    g0= reshape(g0, d*k)

    return (f,g0)
end

```



The valid error is 0.026.

1.4 Cost of Multinomial Logistic Regression

1. The runtime of $exp(w^T x_i)$ is $O(d^2)$, The runtime of $\sum_{c'=1}^{k} exp(w^T x_i)$ is $O(kd^2)$, the cost of training with training example n the softmax classifier is $O(ntkd^2)$

2. The cost of classifying the test examples is $O(ktd)$

2 MAP Estimation

1.

$$-\sum_{i=1}^{n} log(p(y_i|x_i, w))$$

$$= -\sum_{i=1}^{n} log(\frac{1}{2} exp(-|w^T x_i - y_i|))$$

$$= -(\sum_{i=1}^{n}(log\frac{1}{2} + log(exp(-|w^T x_i - y_i|)))$$

$= -\sum_{i=1}^{n}(log\frac{1}{2})(constant) - \sum_{i=1}^{n}(-|w^T x^i - y_i|)$

$= \sum_{i=1}^{n}(|w^T x^i - y_i|)$

$= ||XW - y||_1$

$-\sum_{j=1}^{n} log(p(w_j))$

$= -\sum_{j=1}^{n} log(exp(-\frac{\lambda(w_j - w_j^0)^2}{2}))$

$= -\sum_{j=1}^{n}(-\frac{\lambda|w_j - wj^0|^2}{2})$

$= \frac{\lambda}{2}|w_j - w_j^0|^2$

$f(w) = ||Xw - y||_1 + \frac{\lambda}{2}\left||w_j - w_j^0|\right|^2$

2.
$-\sum_{i=1}^{n} log(p(y_i|x_i, w))$

$= -\sum_{i=1}^{n}(log(\frac{1}{\sqrt{2\sigma_i^2\pi}}) + log(exp(\frac{-(w^T x_i - y_i)^2}{2\sigma_i^2})))$

$= -\sum_{i=1}^{n}(log(\text{-}\sqrt{2\sigma_i^2\pi}) + \frac{-(w^T x_i - y_i)^2}{2\sigma_i^2})$

$= (constant) + \sum_{i=1}^{n} \frac{(w^T x_i - y_i)^2}{2\sigma_i^2}$

$= \frac{1}{2}(XW - y)^T Z(XW - y)$

*Z has σ_i^2along digonals

$-\sum_{j=1}^{n} log(p(w_j))$

$= -\sum_{j=1}^{n} log(\frac{\lambda}{2}exp(-\lambda\,|w_j|))$

$= -\sum_{j=1}^{n}(log(\frac{\lambda}{2}) + log(exp(-\lambda\,|w_j|)))$

$= constant + \sum_{j=1}^{n}(\lambda\,|w_j|)$

$= (\lambda||W||_1)$

$f(w) = \frac{1}{2}(XW - y)^T Z(XW - y) + \lambda||W||_1$

3.
$-\sum_{i=1}^{n} log(\frac{exp(y_i w^T x_i)exp(-exp(w^T x_i))}{y_i!})$

$= -\sum_{i=1}^{n}(log(exp(y_i w^T x_i)exp(-exp(w^T x_i))) - log(y_i!))$

$= -\sum_{i=1}^{n}(log(exp(y_i w^T x_i)) + log(exp(-exp(w^T x_i))) - log(y_i!))$

$= \sum_{i=1}^{n}((-y_i w^T x_i) + exp(w^T x_i)) + \sum_{i=1}^{n} log(y_i!)$

$= \sum_{i=1}^{n}(exp(w^T x_i) - y_i w^T x_i + log(y_i!)))$

$-\sum_{j=1}^{n}(logp(w_j))$

$= -\sum_{j=1}^{n} log(\frac{1}{\sqrt{2\sigma^2\pi}}exp(\frac{-(w_j-0)^2}{2\sigma^2}))$

$= -\sum_{j=1}^{n}(log(\frac{1}{\sqrt{2\sigma^2\pi}}) + log(exp(\frac{-(w_j-0)^2}{2\sigma^2})))$

$= -(constant) + \sum_{j=1}^{n} \frac{-(w_j-0)^2}{2\sigma^2}$

$= \frac{1}{2\sigma^2}||W||^2$

$f(w) = \sum_{i=1}^{n}(exp(w^T x_i) - y_i w^T x_i + log(y_i!))) + \frac{1}{2\sigma^2}||W||^2$

3 Principal Component Analysis (2016)

3.1 PCA by Hand

1. We only calculate the second column mean, therefore we move first column to second column.

$$\begin{bmatrix} -2 & -2 \\ -1 & -1 \\ 0 & 0 \\ 1 & 1 \\ 2 & 2 \end{bmatrix}$$

Since the slope of the two dimension x1 and x2 is 1, We normalized the w_1= $(1/\sqrt{2}, 1/\sqrt{2})$, and $|w_1| = 1$.

2.Since the data has d =2, the reconstruction error is only interested for k = 1.
Reconstruction error: mean of $\hat{x_2} = 1$ and mean of $\hat{x_1} = 0$

$$z = W^T(\begin{matrix} 3-0 \\ 3-1 \end{matrix})$$

$$= (1/\sqrt{2}, 1/\sqrt{2})(\begin{matrix} 3 \\ 2 \end{matrix}) = 5/\sqrt{2}$$

$x_i = W^T Z = (\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}})\frac{5}{\sqrt{2}} + (\hat{u_1}, \hat{u_2}) = (\frac{5}{2}, \frac{7}{2})$

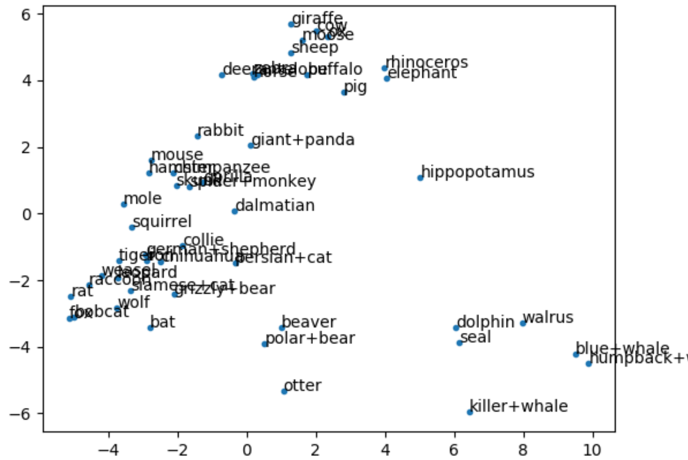
Reconstruction error
$|x_i - \hat{x_i}|$
$= \sqrt{(x_1 - \hat{x_1})^2 + (x_2 - \hat{x_2})^2}$
$= \sqrt{(\frac{5}{2} - 3)^2 + (3.5 - 3)^2}$
$= \frac{1}{\sqrt{2}}$

The reconstruction error is $\frac{1}{\sqrt{2}}$ here.

3. $z = \frac{3-0}{\sqrt{2}} + \frac{4-1}{\sqrt{2}}$
$= \frac{3}{\sqrt{2}} + \frac{3}{\sqrt{2}}$
$= \frac{6}{\sqrt{2}}$

$x = W^T Z = [\frac{1}{\sqrt{2}} \quad \frac{1}{\sqrt{2}}]\frac{6}{\sqrt{2}} + [0 \quad 1]$
$= [\frac{6}{2} \quad \frac{6}{2}] + [0 \quad 1]$
$= [3 \quad 4]$
The error is 0 here.

3.2 Data Visualization



1.

```julia
# Load data
dataTable = readcsv("animals.csv")
X = float(real(dataTable[2:end,2:end]))
(n,d) = size(X)

# Standardize columns
include("misc.jl")
(X,mu,sigma) = standardizeCols(X)
include("PCA.jl")
model = PCA(X, 2)
Z = model.compress(X)

# Plot matrix as image
using PyPlot
figure("PCA")
clf()

plot(Z[:,1], Z[:,2], ".")

for i in 1:n
    annotate(dataTable[i+1, 1], xy = [Z[i,1], Z[i,2]], xycoords = "data")
end
```

2. The furry has the largest influence on the first principal component.
3. The grazer has the largest influence on the second principal component.

3.3 Data Compression

1. When k =2 there is only 30.2 % of the variance represent in the data.
2. k = 5, it explain 50% of the variance in the data.
3. k = 13, it explain 75% of the variance in the data.

4 Very-Short Answer Questions

1. When n value being extreme large and goes infinitely.The classic convergence analysis does not reply on n value in stochastic gradient method.However, gradient descent method will stop collecting data.

2.Stochastic gradient descent make the variance really big, hard to converge to the minimum of a convex function.

3.Multi-class has only one correct label for each object, however, Multi-label classification can have more than one correct labels for each object.

4.In MAP equation and MLE equation that the only difference is inclusion of prior p(θ), which means some likelihood from prior.

5.Not the same line.PCA minimize the orthogonal distance to predict the feature and Linear regression minimize the vertical squared distance for prediction.

6.No, because every vector in direction can minimize the PCA.

7.Non-negative least squared and Linear regression with L1-norm.

8.No, when value of d is greater than 1 we need to use projected gradient algorithm.