hw2

Bill Chambers :: StudentID:25912237

October 10, 2015

Contents

1 Problem 1

1.1 Problem 1 Solution

CDF = $\int$ f(x) dx
$= \int \frac{2}{\pi(1+x^2)} dx$
$= \frac{2}{\pi}\int \frac{2}{1+x^2} dx$
$= \frac{2}{\pi} tan^{-1}(x)$

Now we just need to evaluate this for each interval. Keep in mind that each of these need to be multiplied by $2/\pi$.

$\tan^{-1}(0) = 0$
$\tan^{-1}(1/\sqrt{3}) = \pi/6$
$\tan^{-1}(1) = \pi/4$
$\tan^{-1}(\sqrt{3}) = \pi/3$

We know that $\int_{ab}$ f(x)dx = F(b) - F(a). So we can use that to get our probabilities.

For the interval from 0 -> $1/\sqrt{3}$
$2/\pi$ * $\pi/6$ - 0 = 2/6

For the interval from $1/\sqrt{3} -> 1$
$2/\pi$ * $\pi/4$ - $2/\pi$ * $\pi/6$ = 1/6

For the interval from 1 -> $\sqrt{3}$
$2/\pi$ * $\pi/3$ - $2/\pi$ * $\pi/4$ = 1/6

Now that we have those values we can go about calculating the expectation which is simply the probabilities of each of those happening (that we calculated above) multiplied by the point values in order to get the expected value.

4/6 + 3/6 + 2/6 = $\frac{13}{6}$

2 Problem 2

2.1 Problem 2 Solution

All sums/products are over N.

$$P(x_i|\theta) = \theta e^{-\theta x}$$

$$lik(\theta) = \prod^{n} p(x_i|\theta)$$

$$= \sum_{i=1}^{n} log(p(x_i|\theta))$$

$$= \sum_{i=1}^{n} log(\theta) - \theta x_i$$

$$= nlog(\theta) - \sum_{i=1}^{n} \theta x_i$$

Now we can set the derivative equal to 0 to get the maximum likelihood.
max. lik(θ) = ∂ y / ∂ θ = 0

$$\frac{\partial nlog(\theta) - \sum_{i=1}^{n} \theta x_i}{\partial \theta} = 0$$

$$n/\theta - \sum_{i=1}^{n} x_i = 0$$

$$\frac{n}{\theta} = \sum_{i=1}^{n} x_i$$

$$n/\sum_{i=1}^{n} x_i = \theta$$

$$\frac{5}{5.7} = \theta$$

3 Problem 3

3.1 Problem 3

3.1.1 Part a

$k(x,y) = (\sum_{i=1}^{n} x_i\ y_i + c\ )^2$
$= ((x_1\ y_1 + c) + \ldots\ + (x_n\ y_n + c))^2$
$= x_1^2\ y_1^2 + 2\ x_1\ y_1\ x_2\ y_2 + x_2^2 + x_2^2 + 2c\ x_1\ y_1 + \ldots\ + c^2$
or expressed as a dot product of the vectors
$= (x_1^2, \ldots., x_n^2, y_1^2, \ldots., y_n^2, \sqrt{2}x_{n\text{-}2}\ x_{n\text{-}1}, \ldots, \sqrt{2c}x_n, c)$

3.1.2 Part b

Cross validation and leveraging that to tune hyperparameters. We want to get something that generalizes well and doesn't just minimize the empirical risk but rather the generalized risk.

4 Problem 4

4.1 Problem 4

4.1.1 a

i and j are equal to the rows and columns of the vectors respectively.

$$x^T Ax = \sum_{j=1}^{n} \sum_{i=1}^{n} a_{j,i} x_i x_j$$

4.1.2 b

Let x be all vectors such that we have all vectors that have a 1 in one specific place and zero in all the others. For example,

$\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$ $\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$ $\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$

But continued for the length of N so that we can perform x^T A x. These vectors exist so that they can extract the diagonal values of matrix A. Given that x^T A x is necessarily greater than 0 for all non-zero vectors x, the diagonal elements of A have to be positive or else we've invalidated the fact that x^T A x > 0.

The only way that x^T A x can be negative or zero, is if A diagonal values are negative or zero. This proves if A diagonals are positive, A is positive definite.

5 Problem 5

5.1 Problem 5

From the previous problem we proved that if B is a positive definite matrix, then all the diagonal values have to be positive. If we repeat that proof along with the assertion that for p.s.d. $x^T Ax \geq 0$ we prove that all diagonal values in the p.s.d. matrix are greater than or equal to 0.

Using that, we can see that if we extract the minimum value from the diagonal of the lowest p.s.d. matrix B, then we will extract a 0 value (and no less) we will call that β.

$$\beta = 0$$

Secondly if we extract the minimum value from the smallest possible matrix of γ I, that has to necessarily be positive if $\gamma > 0$. We'll call that γ^* given that $\gamma^* > 0$.

$$\gamma* > 0$$

From these definitions,

$$\beta + \gamma* > 0$$

Since there are the minimum values along the diagonals, we know that all other values are greater than or equal to the above values. We have therefore proved that all values along the diagonal in the resulting matrix C are positive.

Now we can apply the proof from the previous problem, that for all non-zero x, x^T Cx must be greater than 0. Our newly created matrix (C) satisfies this requirement because the only way that X^T Cx would be ≤ 0 would be if one of the diagonal elements was 0 or negative which we just proved is impossible.

6 Problem 6

6.1 Problem 6

6.1.1 Part a

$$x^T a = \sum_{i=1}^{n} x_i * a_i$$

$$\frac{\partial \sum_{i=1}^{n} x_i * a_i}{\partial [x_1 .... x_n]}$$

Derive it component by component and you get the vector:

$$a$$

6.1.2 Part b

i = columns, j = rows

$$x^T Ax = \sum_{j=1}^{n} \sum_{i=1}^{n} a_{ji} x_i^T x_j$$

$$\frac{\partial \sum_{j=1}^{n} \sum_{i=1}^{n} a_{ji} x_i^T x_j}{\partial [x_1 .... x_n]}$$

This can be expressed as a summation as a sort of dot product like we see in the summation above

$$a_{11}x_1^2 + ... + a_{nn}x_n^2 + (a_{12} + a_{21})x_1 x_2 + ... + (a_{n-1,n} + a_{n,n-1})x_{n-1}x_n$$

Which is all product combinations.

When you derive it component by component you end up with the derivatives of x on the columns added to the derivatives of x on the rows of matrix A.(since x^T affects columns and x affects rows). You get the answer

$$Ax + A^T x$$

or

$$(A + A^T)x$$

6.1.3 Part c

i = columns, j = rows

$$\frac{\partial Trace(XA)}{\partial X}$$

$$Trace(XA) == \sum_{i=1}^{n} (XA)_{ii} == \sum_{i=1}^{n} (\sum_{j=1}^{n} x_{ji} a_{ij})$$

Now when we look at the derivative w.r.t. X. . . .

$$\frac{\partial \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ji} a_{ij}}{\partial \begin{bmatrix} x_{1,1} & ... & x_{1,n} \\ ... & ... & ... \\ x_{n,1} & ... & x_{n,n} \end{bmatrix}}$$

We can see that we're ending up with the values of A, however they're transposed because of the trace summation identity shown above.

So the answer is

$$A^T$$

6.1.4 Part d

i = columns, j = rows

$$a^T X b = \sum_{j=1}^{m} \sum_{i=1}^{n} X_{ji} * a_i * b_j$$

$$\frac{\partial \sum_{j=1}^{m} \sum_{i=1}^{n} X_{ji} * a_i * b_j}{\partial \begin{bmatrix} x_{1,1} & ... & x_{1,n} \\ ... & ... & ... \\ x_{m,1} & ... & x_{m,n} \end{bmatrix}}$$

Which, when we derive by components brings us to...

$$\begin{bmatrix} a_1 b_1 & ... & a_1 b_n \\ ... & ... & ... \\ a_m b_1 & ... & a_m b_n \end{bmatrix}$$

Leaving us with

$$ab^T$$

6.1.5 Part e

$$||x||_2^2 = \sum_{i=1}^{n} x_i^2$$

$$||x||_1^2 = (\sum_{i=1}^{n} |x_i|)^2$$

$$||x||_1^2 = [x_1 + x_2 + x_3, ..., x_n]^2$$

$$||x||_1^2 = x_1^2 + x_2^2 + 2x_1 x_2 .... 2x_n x_{n-1}$$

$$||x||_1^2 = \sum_{i=1}^{n} x_i^2 + 2 \sum_{i=1}^{n} \sum_{j=1}^{n} x_i x_j, i \neq j$$

Therefore

$$\sqrt{\sum_{i=1}^{n} x_i^2} \leq \sqrt{\sum_{i=1}^{n} x_i^2 + 2\sum_{i=1}^{n}\sum_{j=1}^{n} x_i x_j, i \neq j}$$

$$\sum_{i=1}^{n} x_i^2 + 2\sum_{i=1}^{n}\sum_{j=1}^{n} x_i x_j, i \neq j \leq n\sum_{i=1}^{n} x_i^2$$

$$2\sum_{i=1}^{n}\sum_{j=1}^{n} x_i x_j, i \neq j \leq n\sum_{i=1}^{n} x_i^2 - \sum_{i=1}^{n} x_i^2$$

$$2\sum_{i=1}^{n}\sum_{j=1}^{n} x_i x_j, i \neq j \leq (n-1)\sum_{i=1}^{n} x_i^2$$

or alternatively

$$\frac{\sum_{i=1}^{n} x_i^2}{n} + \frac{2\sum_{i=1}^{n}\sum_{j=1}^{n} x_i x_j, i \neq j}{n} \leq \sum_{i=1}^{n} x_i^2$$

Which is necessarily greater.

7 Problem 7

7.1 Problem 7

In this problem, ' == ^T, the ' is equal to the transpose. Typing all this up, I fatigued on having to write it in brackets for the transpose.

7.1.1 Part a

$$R[w] = \Lambda(Xw - Y)^2$$

$$R[w] = (Xw - Y)\Lambda(Xw - Y)$$

If the above does suffice, please see below where I'm sure you'll see the answer you're looking for.

7.1.2 Part b

$$R[w] = \Lambda(Xw - y)^2$$

$$= (Xw - y)^T\Lambda(Xw - y)$$

$$= w'X'\Lambda XW - y'\Lambda Xw - y'\Lambda y - w'X'\Lambda y$$

we can transpose the last term to get the middle term

$$= w'X'\Lambda XW - 2y'\Lambda Xw - y'\Lambda y$$

Now we can take the derivative w.r.t the weights of the above statement and set it to 0 to minimize the risk.

$$0 = \frac{\partial(w'X'\Lambda XW - 2y'\Lambda Xw - y'\Lambda y)}{\partial w}$$

$$0 = \frac{\partial w'X'\Lambda Xw}{\partial w} - 2\frac{\partial y'\Lambda Xw}{\partial w} - \frac{\partial y'\Lambda y}{\partial w}$$

remove the last term...

$$0 = \frac{\partial w'X'\Lambda Xw}{\partial w} - 2\frac{\partial y'\Lambda Xw}{\partial w}$$

now set β = X' Λ X and and α = y' Λ X

$$0 = \frac{\partial w'\beta w}{\partial w} - 2\frac{\partial \alpha w}{\partial w}$$

from our identities and the previous problem we can derive these easily.

$$0 = (\beta + \beta^T)w - 2\alpha^T$$

now we can fill back in the β and α

$$0 = 2X'\Lambda Xw - 2X'\Lambda y$$

$$X'\Lambda y = X'\Lambda X w$$

$$w = (X'\Lambda X)^{-1} X'\Lambda y$$

7.1.3 Part c

$$R[w] = \Lambda(Xw - y)^2 + \gamma w'w$$

Now we just need to add in another term to the end of our derivation.

$$0 = 2X'\Lambda X w - 2X'\Lambda y + \frac{\partial \gamma w'w}{\partial w}$$

$$0 = 2X'\Lambda X w - 2X'\Lambda y + 2\gamma w$$

remove all 2's

$$X'\Lambda y = X'\Lambda X w + \gamma w$$

we can bring in the identity matrix because $AI = A$

$$X'\Lambda y = (X'\Lambda X + \gamma I) w$$

$$w = (X'\Lambda X + \gamma I)^{-1} X'\Lambda y$$

7.1.4 part d

Shrinkage is a tool to prevent overfitting of the training set. Quite simply if we minimize our training error, we're likely going to fit the training data but not generalize well to unseen data. Shrinkage punishes large weights by shrinking them towards 0 or the null model. It's a way of penalizing models with extreme parameter values.

8 Problem 8

8.1 Problem 8

8.1.1 Part a

We basically have 3 requirements

$$P(w_i|x) \geq P(w_j|x) \; for \; all \; j$$

$$P(w_i|x) \geq 1 - \lambda_r/\lambda_s$$

$$\lambda_s otherwise$$

Now we know if we get a right answer, our cost will be zero. So if our model predicts that x is of class i and y is actually of class j and i=j, then our loss/cost is 0. Therefore, if we are confident above a certain threshold that we have this class, we should choose that class. However that's obviously not without risk...

$$R[choice_i|x] = \lambda_s(1 - P(w_i|x))$$

Here we can see that when the probability that we know the output class goes up than the risk of misclassification decreases. However if we're not confident, the risk of misclassification increases, as we can see in the first equation above. This proves our first requirement, assuming we think we have the right class (above a certain threshold) - we should choose that one to minimize our loss.

But choosing doubt has its own risk.

$$R[choice_{i+1}|x] = \lambda_r$$

Now re-arranging these risk functions allows us to prove our second requirement.

Logically, we're creating an inequality in that if it holds, we do one thing and if it doesn't, we do another. The risk of choosing doubt will be a function of the cost of misclassification along with the probability of making such a misclassification. If the cost of doubt is greater than the cost of making a guess (our confidence multiplied by the cost of misclassification) then we should guess, if not we shouldn't. Or mathematically,

$$r[choice_{i+1}|x] \geq R[choice_i|x]$$

$$\lambda_r \geq \lambda_s(1 - P(w_i|x))$$

Now what we can do from here is re-arrange this inequality to

$$P(w_i|x) \geq 1 - \lambda_s/\lambda_r$$

8.1.2 Part b

If there is no loss when choosing doubt, the risk minimization that takes place will mean that the lowest risk choice is always doubt - making our algorithm useless. If the doubt cost is greater than the mis-classification cost then our algorithm will always take a guess at classifying the example and will never go with doubt. This is a way of forcing our algorithm to take a guess and what class something should belong to.