Lab1 - TDDE01

Anton Gefvert

11/18/2018

Assignment 1

Task 2 - p(Y = 1|X) > 0.5

Confusion matrix for training set:

Table 1: Confusion matrix for training set using threshold 0.5

	0	1
FALSE	913	294
TRUE	32	131

Confusion matrix for test set:

Table 2: Confusion matrix for test set using threshold 0.5

	0	1
FALSE	887	343
TRUE	50	90

As we can see in Table 1 and Table 2, the results using the formula

$$\hat{Y} = \text{if } p(Y = 1|X) > 0.5, \text{ otherwise } \hat{Y} = 0$$

gives the following statisticts:

- Train
  - 32 false positives
  - 294 false negatives
  - total we have 326 out of 1370 wrongful classification
  - missclassification: 326/1370 = 0.238 = 23.8%
- Test
  - 50 false positives
  - 343 false negatives
  - total we have 393 out of 1370 wrongful classification
  - missclassification: 393/1370 = 0.287 = 28.7%

Task 3 - p(Y = 1|X) > 0.9

Confusion matrix for training set:

Table 3: Confusion matrix for train set using threshold 0.9

	0	1
FALSE	945	424
TRUE	0	1

Confusion matrix for test set:

Table 4: Confusion matrix for test set using threshold 0.9

	0	1
FALSE	936	433
TRUE	1	0

What we see from Table 3 and Table 4, is that when we increase the constraint for marking an email as spam, we get way less false positives (1 instead of 50 in test set), but the false negatives are increased.

Missclassifications: * Train: 424/1370 = 0.309 = 30.9% * Test: 434/1370 = 0.317 = 31.7%

As we can see, the missclassification is increased, but you could argue that in this case, marking a non-spam email as spam, is way worse than letting through spam. It's easier for the recipient to filter away less spam, than to check through the spam mail to fins non-spam emails among them. In this case though, there are barely any true positives, so this "spam-filter" will pretty much do nothing.

Task 4 - knn, k = 30

Confusion matrix for train set

Table 5: Confusion matrix for train set using knn with k = 30

	0	1
FALSE	807	98
TRUE	138	327

Confusion matrix for test set

Table 6: Confusion matrix for test set using knn with k = 30

	0	1
FALSE	672	187
TRUE	265	246

As we can see from Table 5 and Table 6 we get the following missclassification rates: * Train: 236/1370 = 0.172 = 17.2% * Test: 452/1370 = 0.329 = 32.9%

What we see here is that missclassification rate for train set is way lower than in exercise 2, which is reasonable since those will be the nodes that can be neighbors. We see that the missclassification rate for the test set has gone up compared to exercise 2. In both cases, the false positive rate has gone up, which is problematic for this spam classification.

Task 5 - knn, k = 1

Confusion matrix for train set

Table 7: Confusion matrix for train set using knn with k = 1

	0	1
FALSE	945	0
TRUE	0	425

Confusion matrix for test set

Table 8: Confusion matrix for test set using knn with k = 1

	0	1
FALSE	640	177
TRUE	297	256

As we can see from Table 7 and Table 8 we get the following missclassification rates: * Train: 0/1370 = 0% * Test: 474/1370 = 0.346 = 34.6%

As can be seen in Table 7, using k = 1, makes this model perfect. This is simply because we are using the model to classify the same dataset that is datapoints, meaning the closest neighbor will be the node itself and since there is only one neighbor, we will perfectly match each node.

The test set gets slightly worse than in exercise 4.

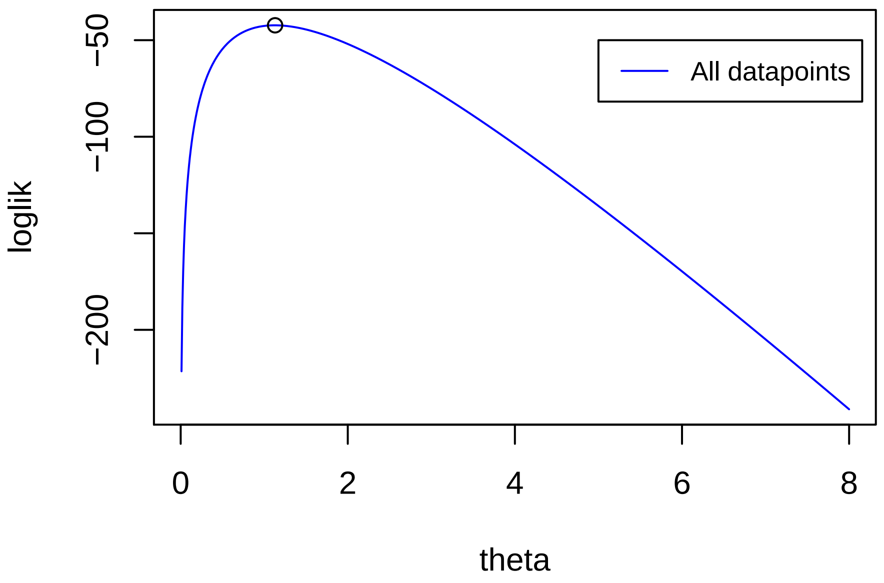


Figure 1: log likelyhood for task 2

Assignment 2

Task 2

The distribution type of $p(x|\theta)$ is a exponential distribution, a distribution that models the time between events in a poisson distribution.

As seen in Figure 1, the maximum loglikelyhood is given for $\theta = 1.13$ and gives the value $loglik \approx -42.3$

Task 3

As seen in Figure 2, the maximum log likelyhood with just the first six variables from x (red line), is given for $\theta = 1.79$ and gives the value $loglik \approx -2.52$.

What we can see from these figures is that the loglikelyhood has a way pointier graf in the case where all values of x is used, meaning it's more reliable (which is reasonable, since we have more datapoints). Whereas in the case with only the first sic values of x, the graph is much more flat top and with more values close to the maximum loglikelyhood.

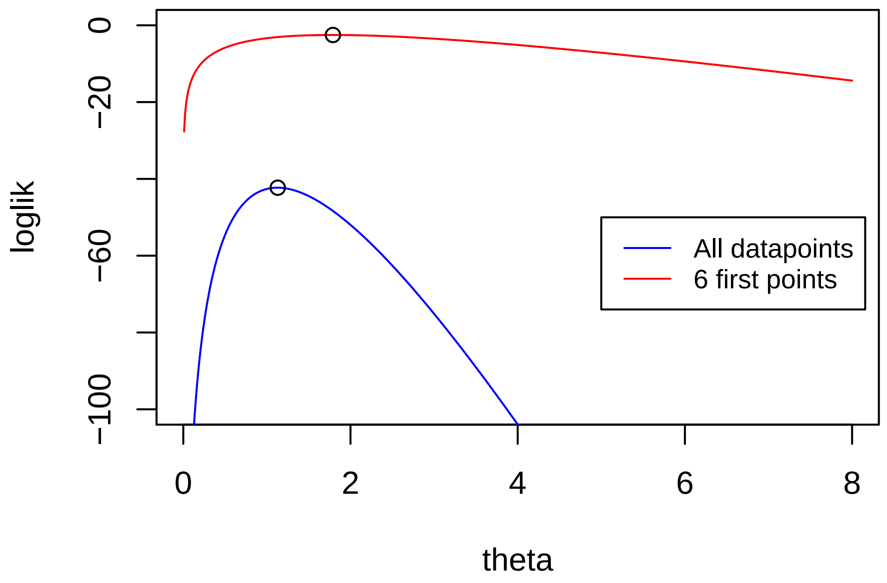


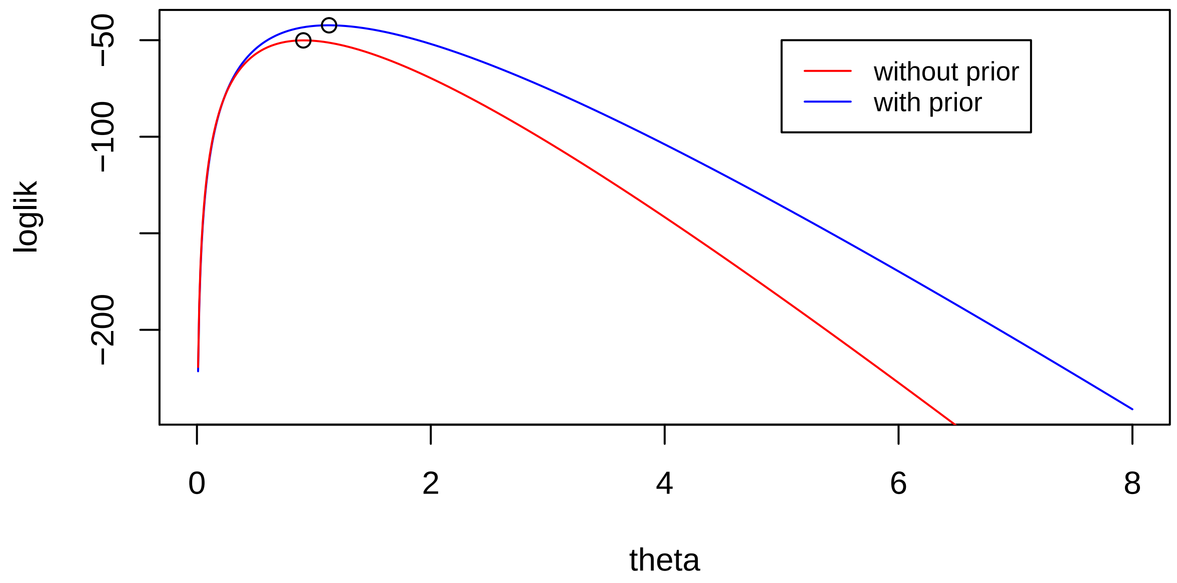
Figure 2: log likelyhood for task 3

Task 4

$$L(\theta) \propto p(\theta|x) \propto p(x|\theta)p(\theta) \propto \theta \left(\prod_{i=1}^{n} e^{-\theta x_i} \right) 10e^{-10\theta}$$

Gives us

$$l(\theta) \propto ln \left[\left(\prod_{i=1}^{n} e^{-\theta x_i} \right) 10e^{-10\theta} \right]$$



The new loglikelihood based on a prior is maximized at $\theta = 1.79$ and gives $loglik \approx -2.51$, if we compare this to the result given when not using a prior, we get about the same loglikelihood and a similar θ, albeit a bit smaller. If we look at the graph in Figure **??**, we see that the one using a prior has a smaller peak, this shows it has a higher certainty (given the conditions) of what θ is.

Task 5

“‘{r hist, fig.width=5, fig.height=4, fig.cap=“\label{fig:fig4} Distribution histograms”} print(data2[,1])

random_x = rexp(50, maxtheta) data2[,1] hist(data2, probability=TRUE) hist(random_x, probability=TRUE) “‘