MLAP Open Assessment

Y6186228

14th May 2014

1 Linear regression and Logistic regression

1.1 Task 1

In order to experiment with linear regression I have chosen to use the following 8 features, referred to in equations as a through h for brevity:

- a stock volume of previous day
- b difference between the previous two days' stock volumes
- c mean of stock volumes from previous ten days
- d standard deviation of stock volumes from previous ten days
- e stock price of previous day
- f difference between the previous two days' stock prices
- g mean of stock prices from previous ten days
- h standard deviation of stock prices from previous ten days

Further, the elements of a vector θ represent the coefficients of a regression function, with θ_0 always representing the constant term. Hence, a regression function might look as follows:

$$f(\theta) = \theta_0 + \theta_1 a + \theta_2 a^2 + \theta_3 b$$

a is chosen because, combined with b, it gives an indication of the general direction and magnitude of the stock volumes in the past two days. However, it is only a very small amount of data, and likely to give erroneous predictions. c is included as it could give an indication of whether or not a is anomalously low or high in the context of recent data. d allows an insight into the volatility of stock volumes over recent days, which may assist in determining the likelihood of a particularly low or high estimate for the next day. These features are expected to be indirectly useful as they suggest the stock volume, which in turn helps to suggest a stock price (assuming that stock volume and stock price have some form of correlation). Features e, f, g and h are analogous to features a, b, c and d but they are applied to the stock price data rather than the stock volume data.

Figure 1 shows the Mean Squared Errors (MSEs) obtained in my initial phase of experimentation with the chosen features, shown to three significant figures. In this phase of experimentation, I evaluated the performance of each feature used on its own in first- and second-order polynomials. In order to improve the reliability of the results (since the data folds are chosen at random) each evaluation is performed three times. The average of the three results is presented in this report. All evaluations in section 1 of this report are performed and presented in this manner.

It is clear from these results that the best performances come when using the features that relate to stock price as opposed to stock value (e – h). However, feature f, the difference between the last two days' stock prices does not appear to perform very well. Features e and g perform exceptionally well both as first- and second-order polynomials.

My next phase of experimentation is to take the high-performing features and try using them on their own in third-order polynomial functions. Following this, I will experiment with combining the better performing features to see what improvements can be made. The results of the initial third-order polynomial experiments are shown in figure 2. Out of interest, I have chosen to try feature c as a third-order polynomial as it was the best performing of the stock volume-related features.

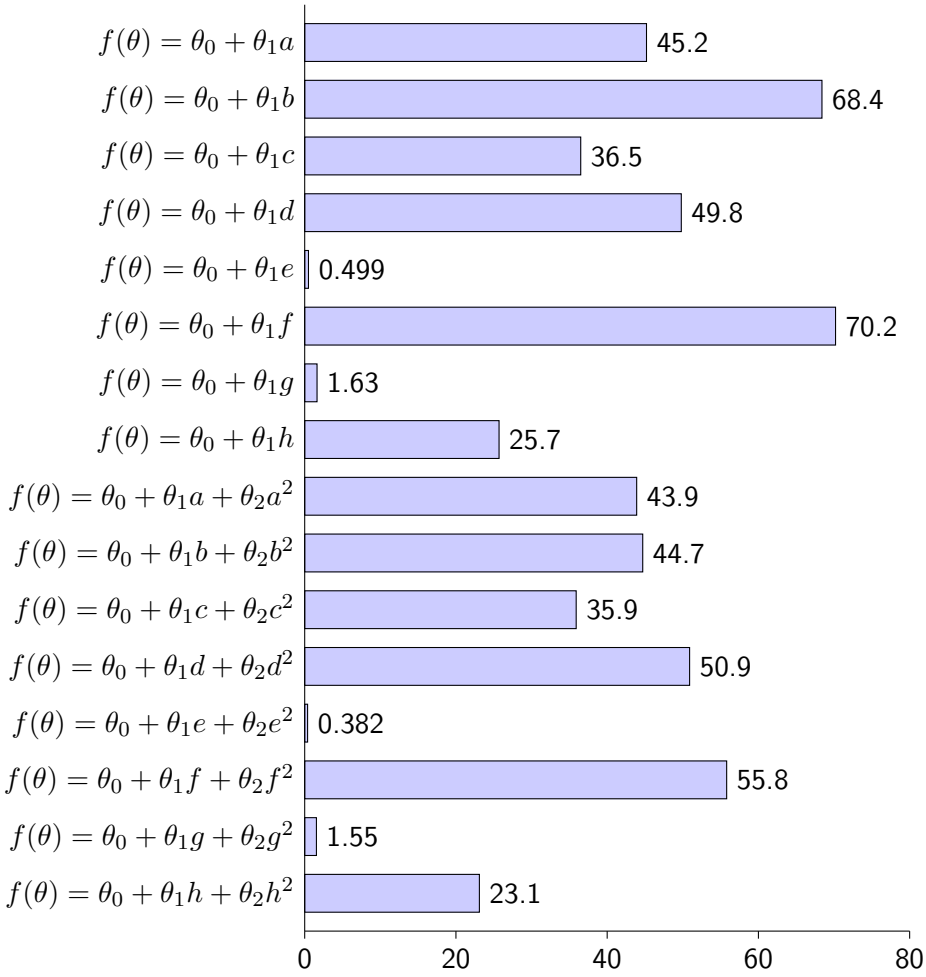

Figure 1: MSEs obtained using each of the features on their own in first- and second-order polynomials

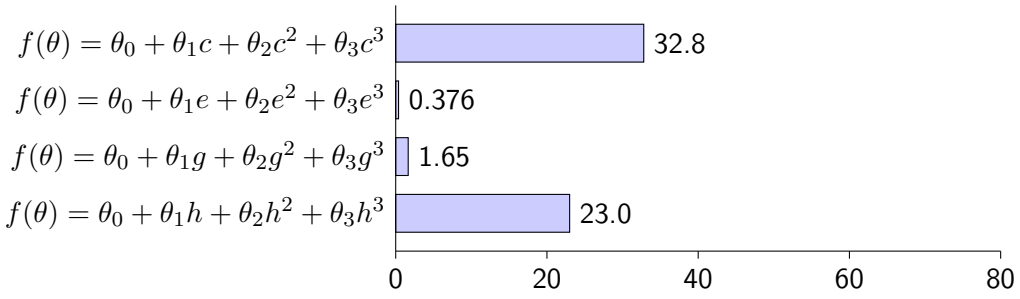

Figure 2: MSEs obtained using selected features on their own in three-order polynomials

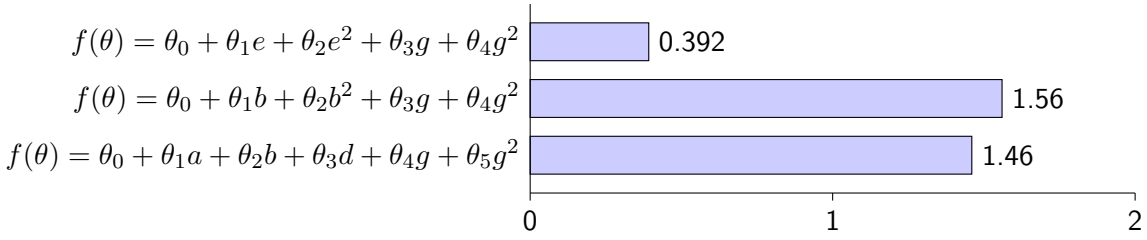

Figure 3: MSEs obtained when combining features into more complex polynomials

As seen by the results, each feature tested in third-order polynomials had very similar results to the second-order polynomial tests. Going forward, I have chosen to try a function combining e and g both as second-order polynomials to see if they perform well as a pair. I am also interested to to see if the recent change in stock volume (b) combined with the mean of the last ten days' stock prices (g) gives an indication of the next stock price. Further, combining a, b, d and g all together may give good results. The MSEs obtained for these tests are shown in figure 3 (note the change of scale for this chart).

From these results we can see that combining e and g does not really improve on the performance achieved when using e alone. Also, combining g variously with a, b and d does not improve on the performance of using g alone.

1.2 Task 2

The same features, represented as a through h, are used in this task. For simplicity, I will simply list the features and powers used in each test (for example, a, a^2, b). The constant term is implicitly used in every case.

Again, I started by evaluating the use of each feature on its own in first- and second-order polynomials.

Figure 4 shows the percentage accuracy achieved by each of the tests using each feature individually in first- and second-order polynomials. As can be seen, the high performing features (f, h) did not benefit from being included in a squared form. Some of the features were better used in their second-order polynomials than in their first-order ones (e.g. a) while others were significantly worse (e.g. b).

I now will test the performance of using different combinations of features. One such combination is f, h, to see if combining them improves on their individual successes. Also, the pairs e, h and g, h, since each pair can give an idea for the general spread of recent stock price data in terms of percentage. The effect of relationship between c, g is also interesting to test, as well as the quadruple of a, c, e, g, since it includes information about the relationship between the last day's data and the mean data from the last 10 days. I will also see what happens if I combine all four of the features that achieved scores in the region of 86% individually (a, d, f, h). In each test, each feature will be raised to the power at which it performed best from the previous tests.

As with task 1, it is seen from the results that combining features (figure 5) did not really improve the performance. Firstly, c, c^2, d, f, h gave worse results than all of its constituents' individual results. As well as this, f and h perform as well individually as when combined together. The e, h result is worse than the independent results of h, though better than both independent results for e. The g, h result is worse than the best independent performances of g and h. c, c^2, g performed poorly. a, a^2, c, c^2, e, g pretty much matches but does not outperform the best performances of 86% from the previous tests, and this is also true for a, a^2, c, c^2, g.

It is interesting to see from these results that the accuracies seem to max out at about 86%. From looking at the actual classifications given by the higher-performing functions I have found that this actually occurs when every row of data is predicted as being in class 0. Some of the functions giving lower accuracies give classification distributions that appear more realistic, but since the evaluated accuracies are lower we must assume that it is the wrong data rows which are being predicted as classes other than 0. Therefore, I am lead to conclude that the best policy found from these experiments is to predict all data rows as being in class 0. These experiments have not been totally comprehensive, however, so it is possible that a function might exist that can achieve a better accuracy (either by including higher polynomials or different combinations of features, or both).

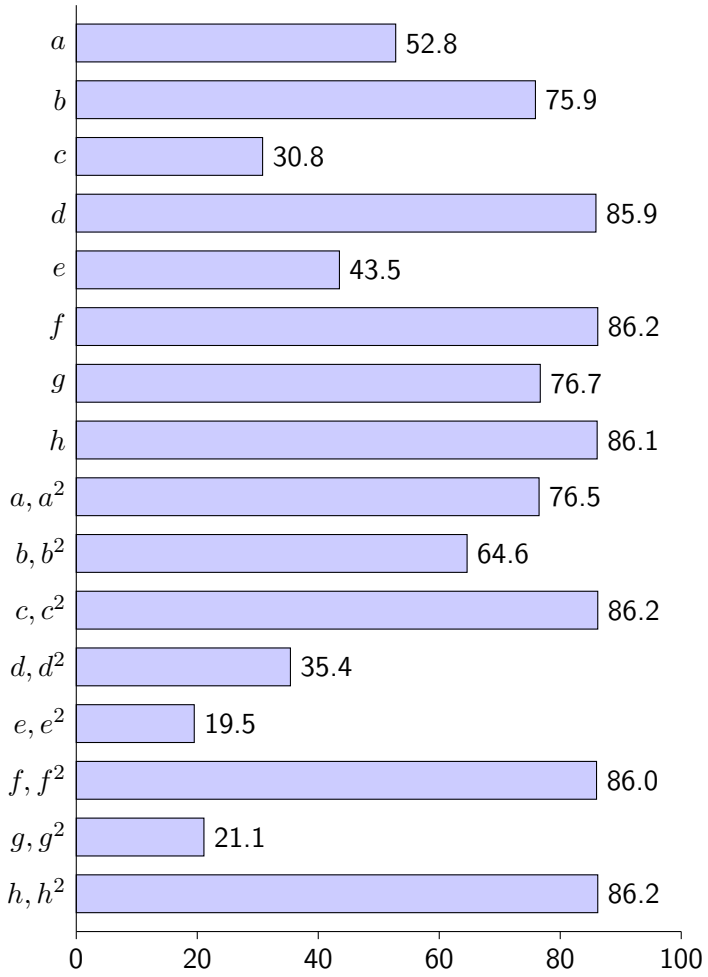

Figure 4: % accuracies obtained using each of the features on their own in first- and second-order polynomials

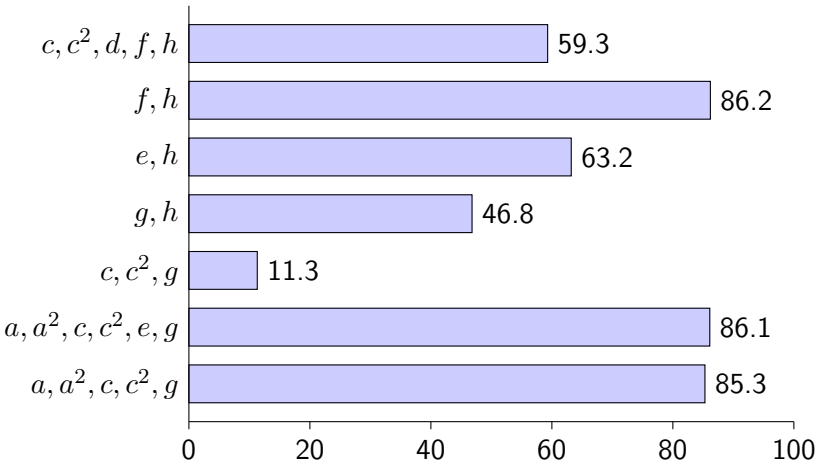

Figure 5: % accuracies obtained when combining features into more complex polynomials

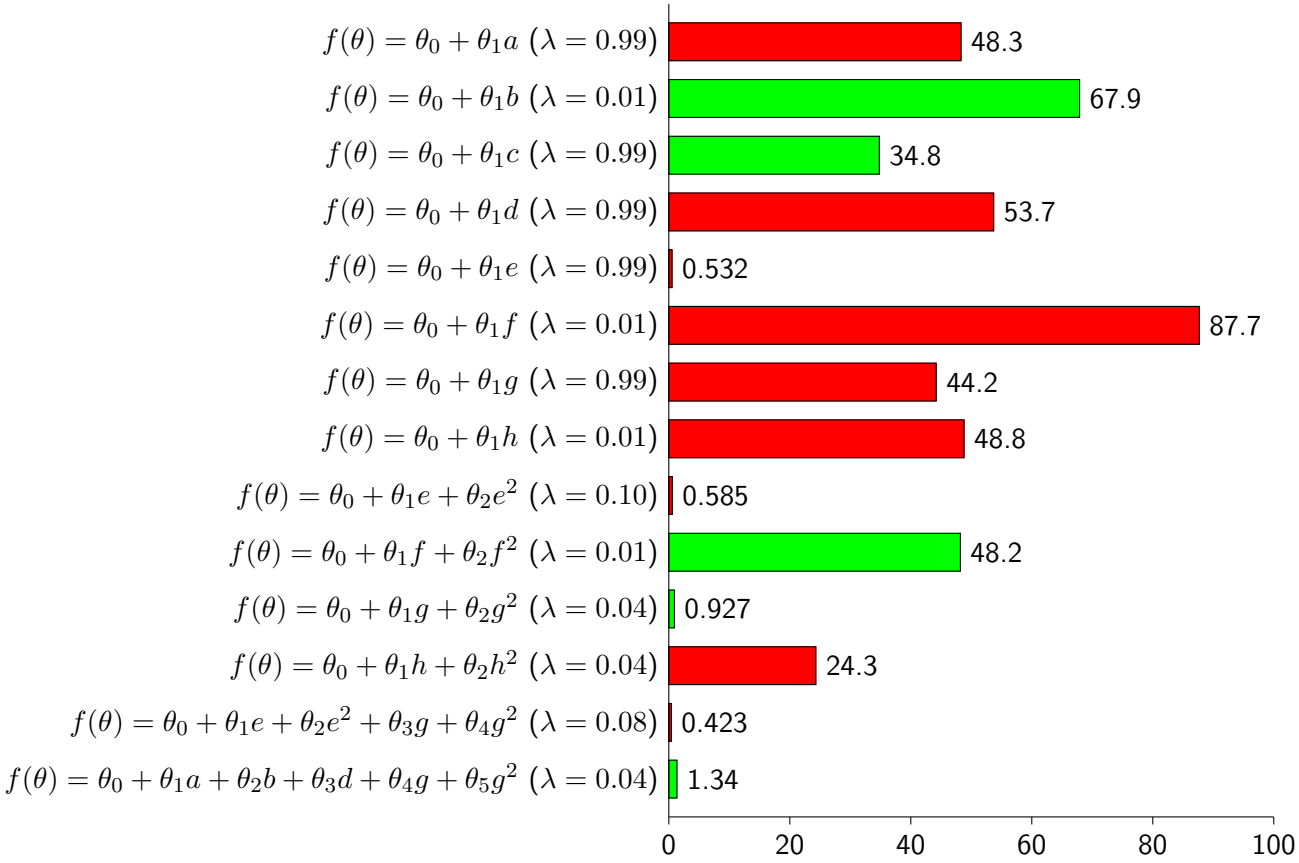

Figure 6: MSEs obtained when using *regularised* linear regression. Green bars show improvements over unregularised linear regression, and red bars show deterioration.

1.3 Task 3

1.3.1 Regularised Linear Regression

For this section I have chosen to evaluate the performance of some of the same functions when regularisation is used (results in figure 6). In each case, I show the λ that was found to be best in training, and the MSE obtained on the test data. Green bars represent functions that performed better under regularised linear regression and red bars represent those that performed worse. The values of λ tested were in the range $[0, 1]$, at increments of 0.01.

From the results we can see that the regularised linear regression was not generally worse or better than the unregularised version. In some cases, there were vastly different results – for example, $f(\theta) = \theta_0 + \theta_1 g$ performed far worse under the regularised version. It is unclear why such a big difference should be found when the λ used is 0.99. It is possible that these differences are only caused by the statistical variation which comes about due to the randomly split data folds. However, an attempt was made to minimise this effect by performing each evaluation three times and reporting the average (as discussed in subsection 1.1). Since regularised linear regression did not improve the results, it may be sensible to suspect that the functions chosen are not on the boundary of overfitting. In that case, it might have been sensible to consider some higher-order polynomials and/or larger combinations of features in order to find functions with lower errors.

1.3.2 Regularised Logistic Regression

Again, I evaluate the performance of some of the combinations of features when using regularised logistic regression (results in figure 7). Again, green bars are those which show improvements when compared to the results for the same features from task 2. This time, the values of λ used were only at increments of 0.1, as the time taken to perform logistic regression is much more significant than the time taken for linear regression.

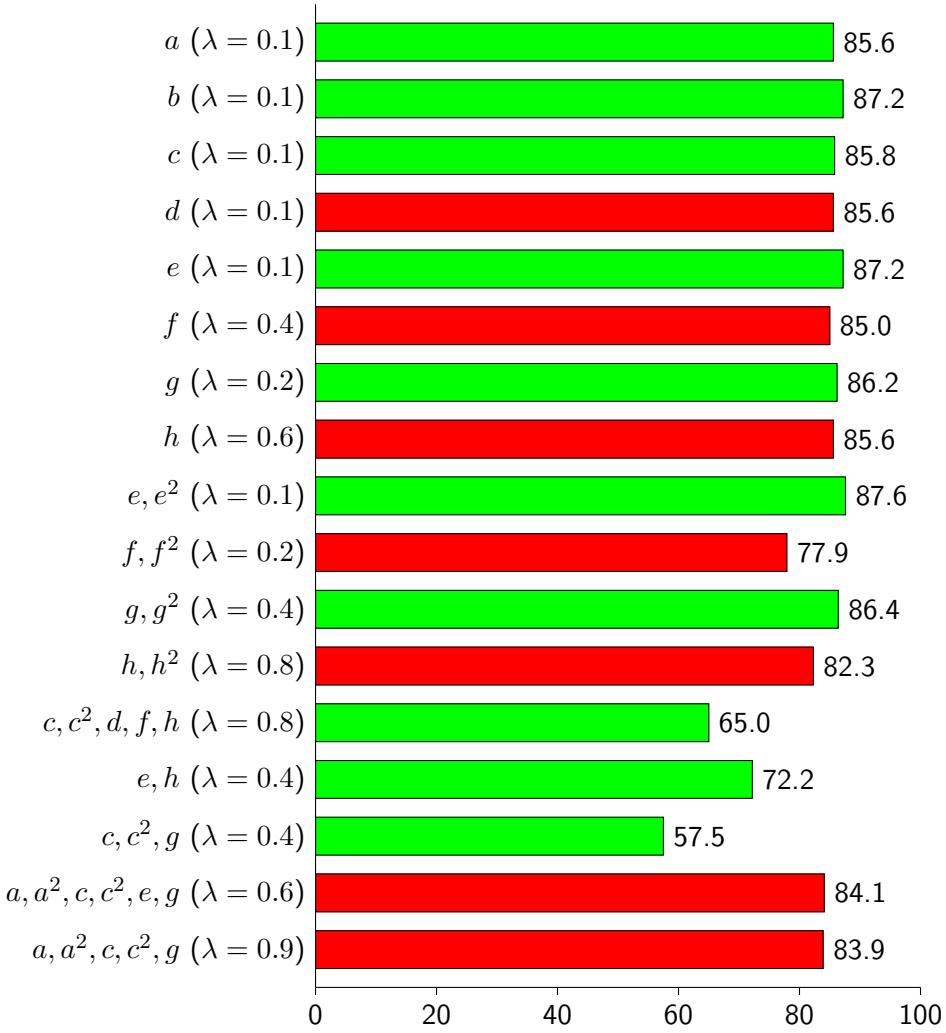

Figure 7: % accuracies obtained when using *regularised* logistic regression. Green bars show improvements over unregularised logistic regression, and red bars show deterioration.

From the results we can see that regularised logistic regression gives very high accuracies. Almost all of the features give scores of around 87% when used independently. The exact value of the 87% scores is not totally consistent. This is likely to be because the folds do not always contain exactly the same data points, and so the accuracy obtained by always classifying rows as class 0 is not totally consistent. It is also possible that on some occasions the theta values obtained from the regression do not quite assign everything to class 0, which potentially explains the lower scores for f, f^2 and h, h^2. The results for c, c^2, d, f, h were notably low when compared to the other regularised results, as were e, h and c, c^2, g. However, these all gave improvements over the non-regularised versions, particularly c, c^2, g. In some of the cases where λ was trained to be quite low (e.g. e; e, e^2; c, c^2, g) we can see vast improvements in the results, particularly for e, e^2 whose accuracy rose from 19.5% to 87.6% when regularisation was used. This is probably because the original non-regularised tests resulted in overfitting, which the regularisation served to reduce. However, there are some notable cases where λ came out very low, but the accuracies increased (e.g. d). The reason for this is unknown.

2 Bayesian networks

2.1 Task 4

The following list gives the means of the posterior distributions for each probability (3sf):

$$
\begin{aligned}
P(0=1|1=0) &= 0.479 \\
P(0=1|1=1) &= 0.936 \\
P(1=1) &= 0.0562 \\
P(2=1|0=0) &= 0.304 \\
P(2=1|0=1) &= 0.610 \\
P(3=1) &= 0.00930 \\
P(4=1|3=0) &= 0.0112 \\
P(4=1|3=1) &= 0.0745 \\
P(5=1|1=0 \wedge 4=1) &= 0.991 \\
P(5=1|1=1 \wedge 4=0) &= 0.998 \\
P(5=1|1=0 \wedge 4=0) &= 0.000107 \\
P(5=1|1=1 \wedge 4=1) &= 0.833 \\
P(6=1|5=0) &= 0.951 \\
P(6=1|5=1) &= 0.0207 \\
P(7=1|2=0 \wedge 5=1) &= 0.701 \\
P(7=1|2=1 \wedge 5=0) &= 0.793 \\
P(7=1|2=0 \wedge 5=0) &= 0.104 \\
P(7=1|2=1 \wedge 5=1) &= 0.922
\end{aligned}
$$

2.2 Task 5

2.2.1 Task 5 Question 1

If the conditional probability to be estimated is one of the ones that is a direct part of the bayesian network (i.e. the probability of a node being true given the values of its parent nodes), then it could be estimated by simply running the sample through the bnbayesfit function, using the same graph structure file but using the sample as the data file. However, this does not allow you directly to calculate any of the following:

- the probability of *multiple* nodes being in some state, i.e. $P(3=1 \wedge 5=0|conditions)$
- the probability of a node where the conditions do not include all of the parents
- the probability of a node where the conditions include a node which is *not* one of the parents
- any probability with a disjunction either in the subject or the condition, i.e. $P(3=1 \vee 5=0|conditions)$ or $P(3=1|cond1 \vee cond2)$

In order to estimate these probabilities the bnbayesfit method is not sufficient. Therefore we must use another method, which I describe below.

A conditional probability $P(A|B)$ could be estimated by selecting all of the instantiations that satisfy the condition B and calculating the proportion of those for which A is true.

For example, if we want to estimate $P(7 = 1|6 = 1)$ (The probability that node 7 is set to 1 given that node 6 is set to 1) then we produce a sample and select from it all of the instantiations in which node 6 is set to 1. Figure 8 shows a full sample of 10 instantiations, with the selected instantiations marked using an asterisk (*).

```
[1, 0, 0, 0, 0, 0, 1, 1] *
[0, 0, 1, 0, 0, 0, 1, 1] *
[1, 0, 1, 0, 0, 0, 1, 1] *
[0, 0, 0, 0, 0, 0, 1, 0] *
[1, 0, 0, 0, 0, 0, 1, 0] *
[1, 0, 1, 0, 0, 0, 1, 0] *
[0, 0, 1, 0, 0, 0, 1, 1] *
[0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 1, 0] *
[1, 0, 1, 0, 0, 0, 1, 1] *
```

Figure 8: A sample of size 10, with the instantiations where node 6 is set to 1 marked by asterisks

From the selection, we find out how many instantiations also have node 7 set to 1. Figure 9 shows just the 9 selected rows, and marks the rows in which node 7 is set to 1 with a plus sign (+).

```
[1, 0, 0, 0, 0, 0, 1, 1] +
[0, 0, 1, 0, 0, 0, 1, 1] +
[1, 0, 1, 0, 0, 0, 1, 1] +
[0, 0, 0, 0, 0, 0, 1, 0]
[1, 0, 0, 0, 0, 0, 1, 0]
[1, 0, 1, 0, 0, 0, 1, 0]
[0, 0, 1, 0, 0, 0, 1, 1] +
[0, 0, 0, 0, 0, 0, 1, 0]
[1, 0, 1, 0, 0, 0, 1, 1] +
```

Figure 9: The selected rows from figure 8, with the instantiations where node 7 is set to 1 marked by plus signs

We can see that, of the 9 instantiations that satisfy $6 = 1$, 5 also satisfy $7 = 1$. We therefore estimate that $P(7 = 1|6 = 1) = \frac{5}{9}$.

2.2.2 Task 5 Question 2

This procedure is most efficient when the size of the sample is just large enough to give a reliable estimate. Too large and the extra arithmetic is not worthwhile. A probability with a rare conditional part requires a larger sample.

2.2.3 Task 5 Question 3

I will use the same example in this question as in question 1 (find $P(7 = 1|6 = 1)$). By counting the satisfying rows in the original data provided in bndata.csv, we know that the probability is around $3688/8887 = 0.415$ (3dp). Using the method described above, I will estimate the probability based on 100, 1000, 5000 and 10,000 samples and record them in the table below. The final column shows the percentage error of the estimation.

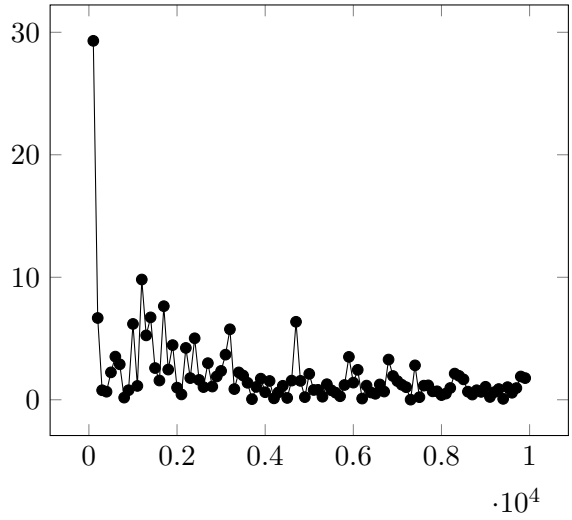

Figure 10: Scatter graph demonstrating the decrease in percentage error based on trials of between 100 and 10,000 samples

Samples	Count $6 = 1$	Count $6 = 1$ and $7 = 1$	Estimated probability (5dp)	Error (3sf)
100	87	28	0.32184	22.4
1000	885	360	0.40678	0.0198
5000	4450	1834	0.41213	0.00692
10,000	8888	3691	0.41528	0.000675

Here, the accuracy of the estimation is shown to get gradually better and better as the number of samples increases from 100 to 10,000.

In addition to the figures given above, in figure 10 I present a scatter graph of the percentage errors obtained when performing estimates of the same probabilities based on varying sample sizes. The samples range from 100 to 10,000 (at intervals of 100) and show a clear trend whereby errors decrease as sample size increases. Note that this data was sampled at a different time to those in the table above, so (due to the randomness involved) the figures do not necessarily match exactly.