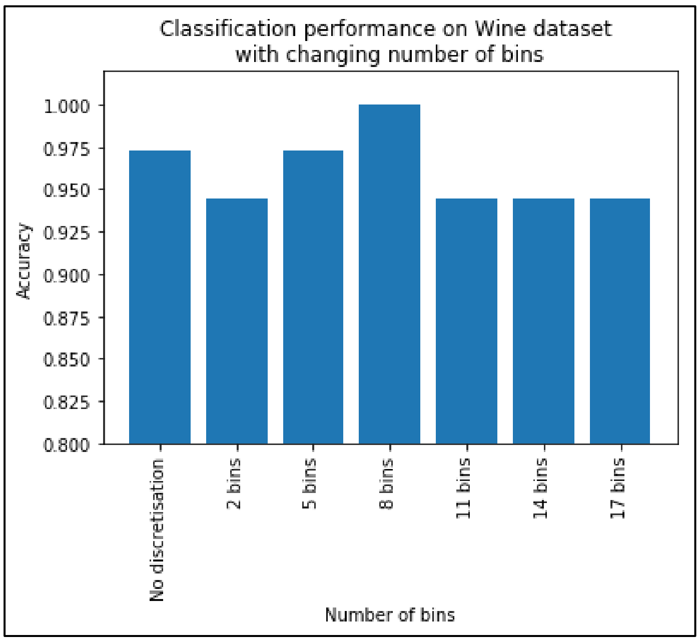
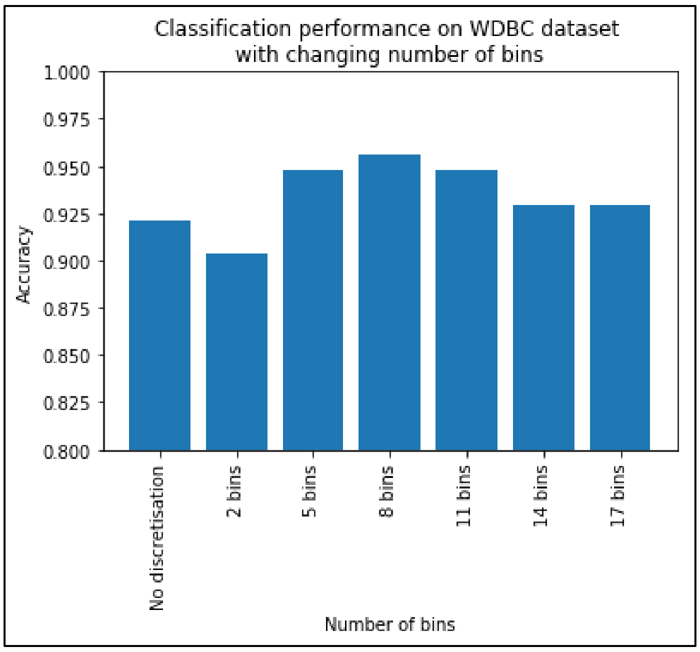
Question 1. "Does discretizing the variables improve classification performance, compared to the Gaussian naïve Bayes approach? Why or why not?"

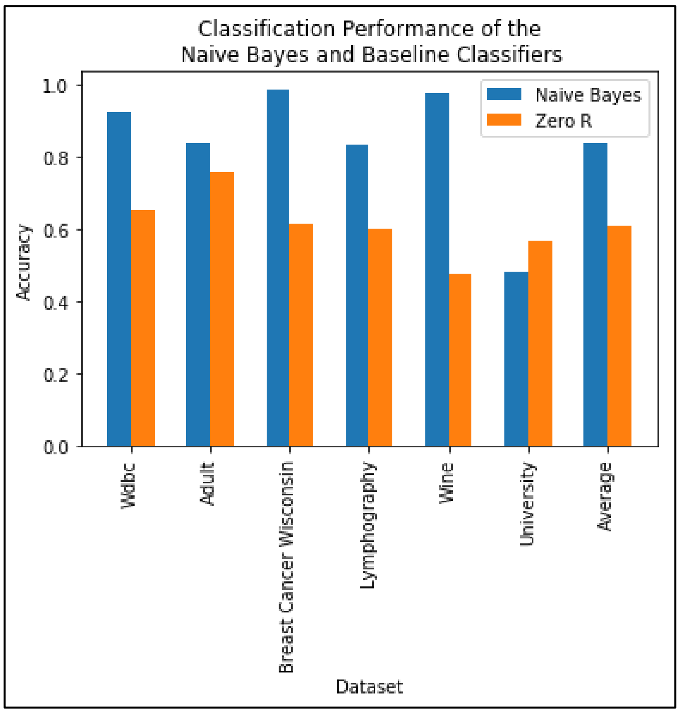




We used equal-frequency discretisation with various number of bins on the continuous datasets, WDBC and Wine. Equal-frequency method is chosen for its robustness against outliers, and because it is reliably reproducible (unlike k-means which uses random processes). Both datasets show a similar pattern, where using too few bins (e.g. 2) reduces performance, as significant loss of information occurs. However, with a suitable number of bins (close to 8 in both datasets), discretising can improve performance, by allowing us to avoid the Gaussian assumption which may not hold. Furthermore, estimations of the Gaussian distribution's parameters are sensitive to outliers, as outliers may incorrectly distort the estimated mean and standard deviation away from their true values. Meanwhile, equal-frequency discretisation is robust to outliers, as outliers are simply added to the highest (or lowest) bin and will not significantly affect the model.

With too many bins, there may be very few instances in a bin. So, it is sensitive to noise and might overfit, especially with small training sets. For example, even if the true probability of a bin given a class is large, the bin may not contain training instances from that class, due to randomness. Thus, the model would incorrectly think the relevant conditional probability is very small. This is likely what led to lower performance when we chose 11-17 bins. However, in the datasets tested, the number of bins (or not discretising) does not impact performance significantly, as they all achieve over 90% accuracy on datasets with reasonable class balance.

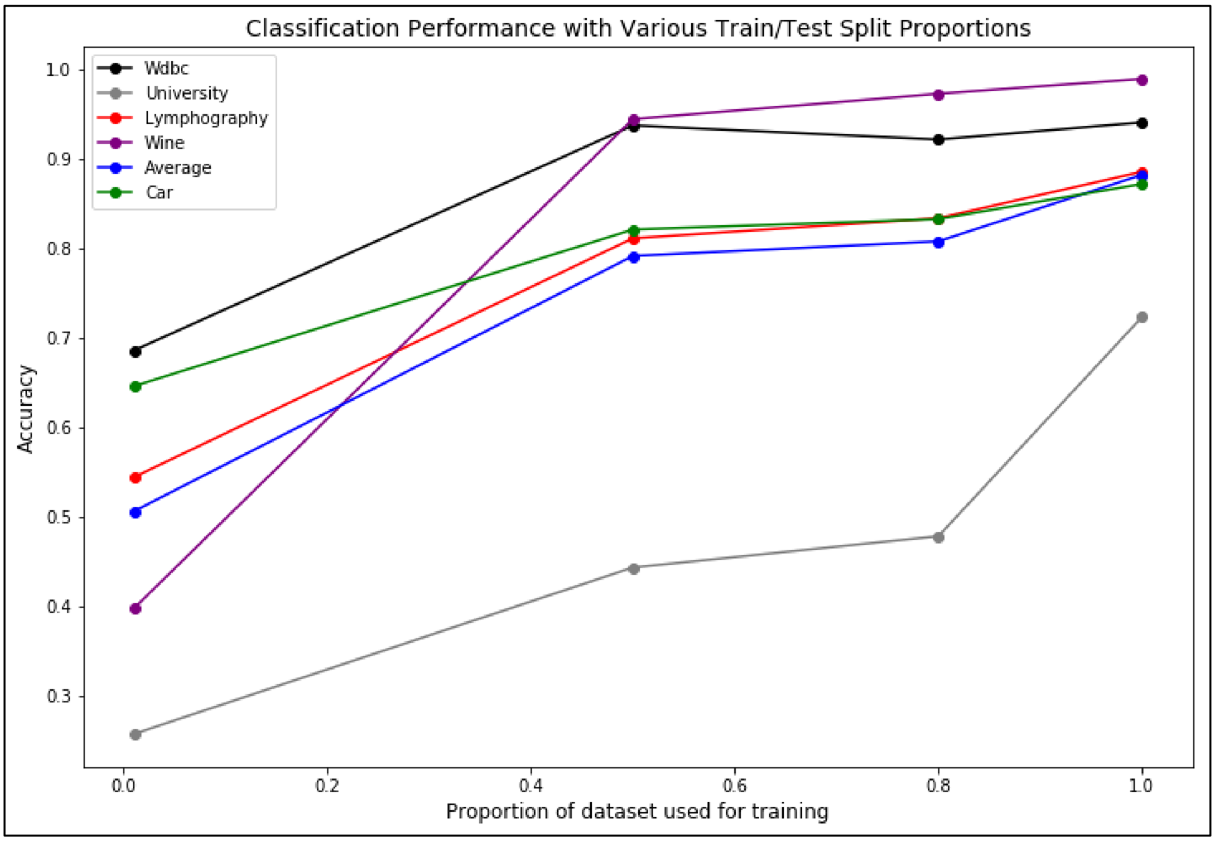
Question 2. "Discuss why the baseline performance varies across datasets, and to what extent the naïve Bayes classifier improves on the baseline performance."



Throughout this report, we use accuracy to measure classifier performance, as it produces a single-number result for datasets with any number of classes, thus simplifying comparisons across datasets. With that said, the accuracy of the zero-R classifier varies significantly between datasets for several reasons. First, these datasets have differing numbers of classes; the first three (in the graph) have only 2 classes, while others have more. With more classes, the accuracy of the zero-R classifier is likely to be lower, because it is less likely for a high proportion of instances to be concentrated in a single class. Also, its accuracy depends on the dataset's class balance. In datasets with extreme class imbalance, for example where one class has 99% of the instances, the zero-R classifier will have greater accuracy than in datasets where class labels are equally distributed.

For most datasets, the naïve Bayes classifier is significantly better than the zero-R classifier, as it considers features and their conditional probabilities in addition to the prior distribution. This provides it with more information, leading to better predictions. However, for certain datasets, the naïve Bayes classifier is only slightly better or even worse; this is likely due to these datasets not fitting the assumptions of the naïve Bayes model. For example, their features may be heavily dependent on each other conditional on the class, or (for continuous attributes) the Gaussian distribution may not be appropriate, thus preventing the naïve Bayes model from significantly improving on the zero-R classifier.

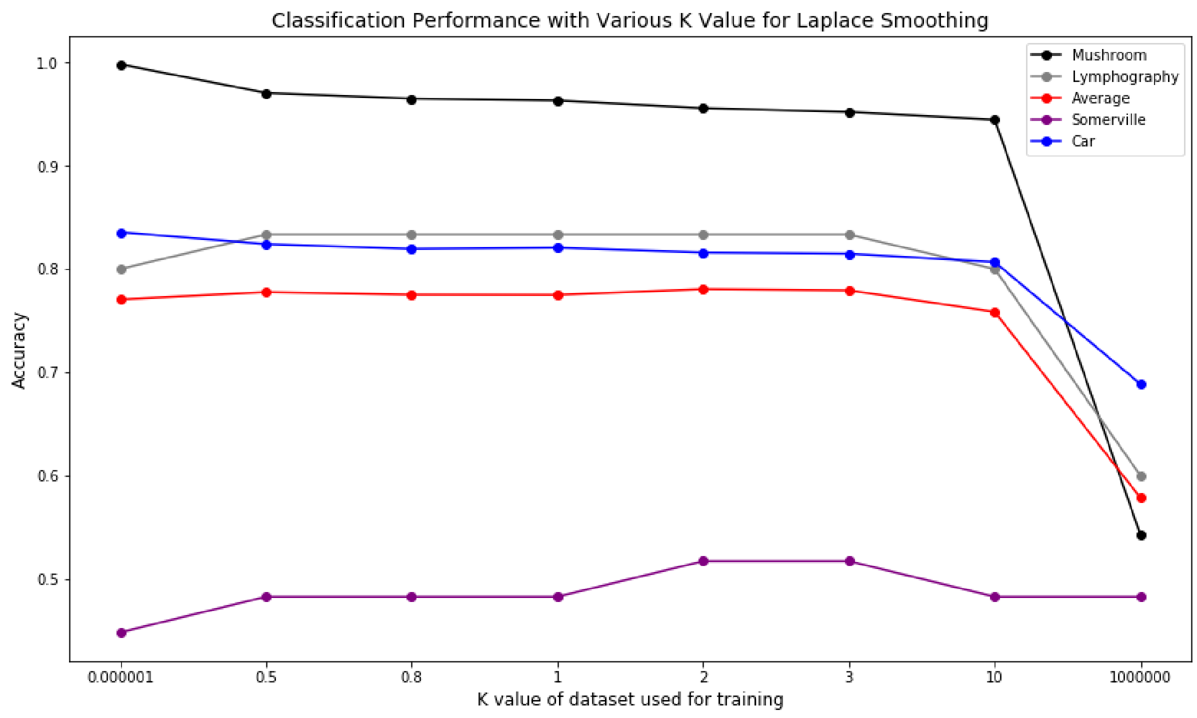
Question 4. "Implement a hold–out or cross–validation evaluation strategy… How does your estimate of effectiveness change, compared to testing on the training data? Explain why."



We ran the Naïve Bayes classifier with varying proportions of the dataset used for training: 0.01, 0.5, 0.8, and 1. Then, we evaluated its performance on a test set. When the train proportion is not 1, the remaining proportion becomes the test set, and when the train proportion is 1, we tested on the training set. The graph above shows that in general, higher train proportions lead to better performance. This is because when the train proportion is high, there are more instances for the model to learn from, resulting in more accurate estimates of the probabilities involved. In addition, this reduces overfitting and allows the model to generalise. The minor exception to this is the WDBC dataset, where the model performed worse with a train proportion of 0.8 than with a train proportion of 0.5. This is likely due to the randomness involved in creating train and test sets. As there are less test instances with higher train proportions, there is a greater chance that the test instances are (on average) more difficult than normal to predict, due to sheer randomness, especially with small datasets such as WDBC. Consequently, the classifier has worse accuracy.

It is also evident that the classifier performs best when we test on the training set, as it does not need to generalise. In contrast, it performs worst with a train proportion of only 0.01, because with very few training instances, it does not have enough data to sufficiently learn the patterns present.

Question 5. “Does changing the smoothing regime (or indeed, not smoothing at all) affect the effectiveness of the naïve Bayes classifier? Explain why, or why not.”



According to the graph above, for large datasets (Mushroom, Car, and Lymphography), the classifier’s accuracy is almost constant as we increase k (the Laplace smoothing constant) from 0.5 to 3. With many instances, counts of most attribute levels are much greater than these values, so these values of k would not significantly affect many conditional probabilities (aside from unseen attribute levels). However, for Somerville (a smaller dataset), accuracy peaks when k is around 2-3. With fewer instances, the classifier’s conditional probabilities (and thus performance) are more noticeably affected by k. Evidently, for Somerville, 2-3 are the suitable k-values which accurately represents the likelihood of observing unseen events.

However, large datasets are also impacted when k is extremely large, which makes the classifier resemble the zero-R classifier. The conditional probability of an attribute value given any class becomes virtually identical, as k is the dominant term. Therefore, when multiplying priors with conditional probabilities, the prior determines the result, and the class with the highest prior probability is chosen. We can clearly see this occurring in the graph above when $k = 1{,}000{,}000$.

Choosing very small values of k also noticeably affects the classifier’s performance. Comparing the results with $k = 0.000001$ (i.e. essentially no smoothing) and $k = 1$, the classifier without smoothing did worse on Somerville and Lymphography, but better on Car and Mushroom. This reflects the likelihood of observing unseen events in those datasets; assuming unseen events will almost never occur is accurate for Car and Mushroom, but not for Lymphography and Somerville.