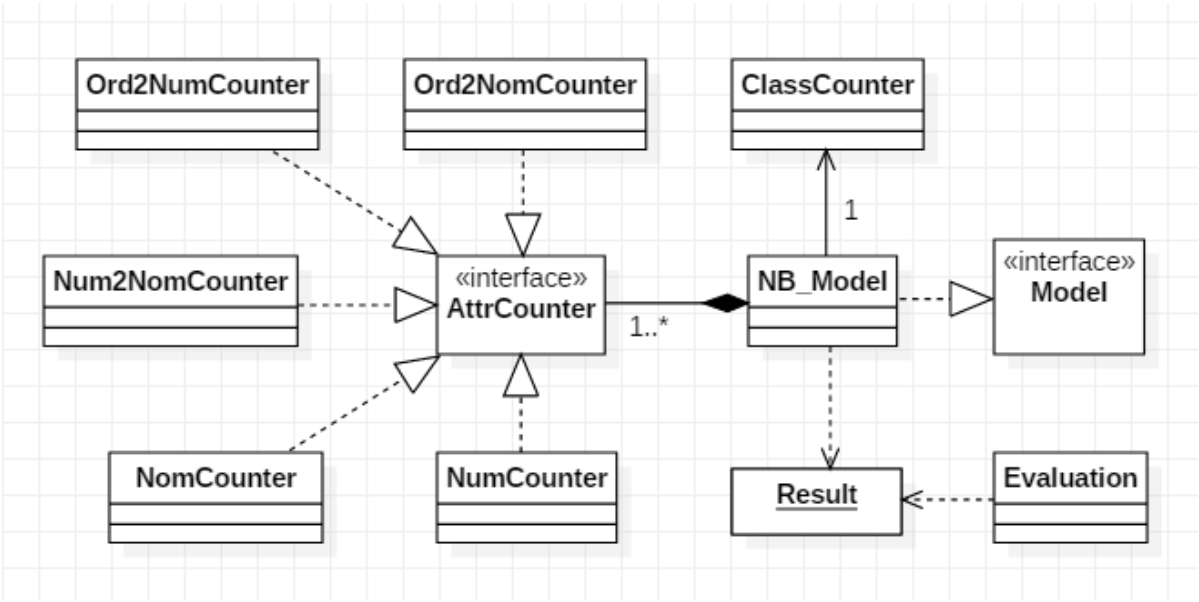
COMP30027 Machine Learning, 2020 Semester 1

Assignment 1: Naive Bayes Classifiers

Student Names: Yue Peng; Tian Qiu
Student IDs: 958289; 988121

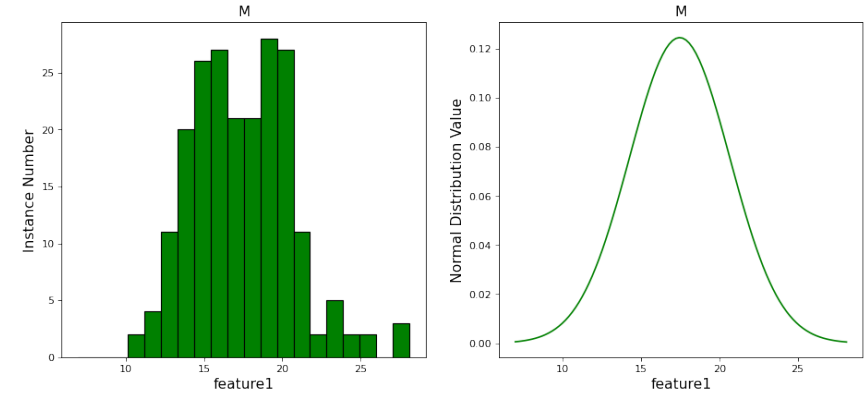
1. Baisc Implementation

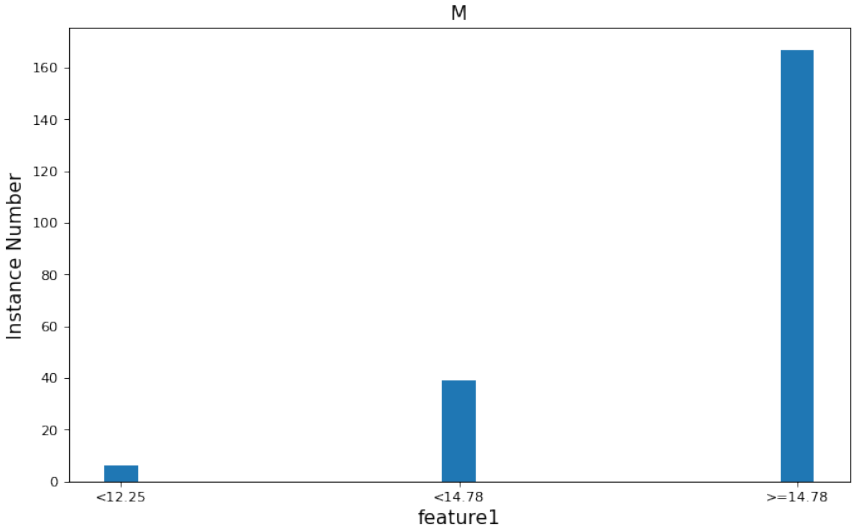


We construct our program using an object oriented way. The Naive Bayes model implements the Model interface where we define the preprocess(), train() and predict() function. We leave the evaluate() function to an evaluation class which can also draw graphs and charts other than simply output statistics. The evaluation class takes a Result object(e.g. [(truth value, predict value)...]) generated from the predict() function of the model class, processes the result and tells us how the model performs.
A NB_Model contains a single ClassCounter and several AttrCounters. They are the basic calculation units to read and counter classes/attributes in the preprocess stage, calculate the probability of classes/attributes in the train stage, and finally return the probability for a specific value in the predict stage. Attribute counters consist of:

- NomCounter - for nominal attributes
- NumCounter - for numeric attributes
- Num2NomCounter - to convert numeric attributes to nominal attributes (Discretisation)
- Ord2NomCounter - to convert ordinal attributes to nominal attributes
- Ord2NumCounter - to convert ordinal attributes to numeric attributes

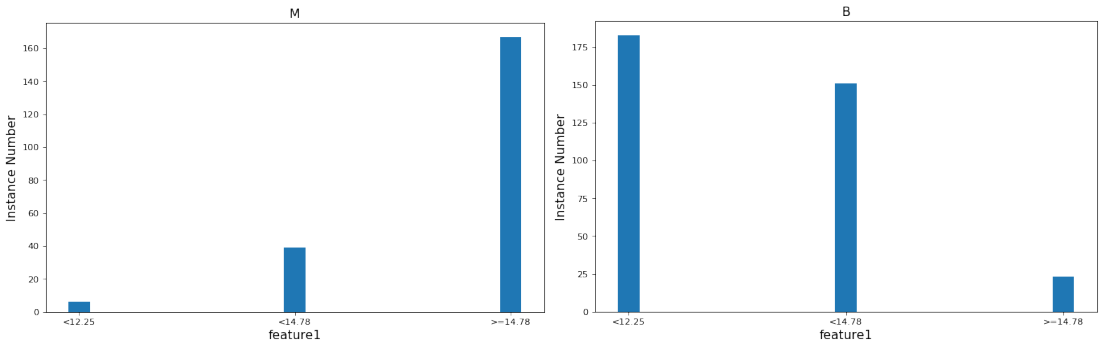
2. Discretizing the Numeric Attributes



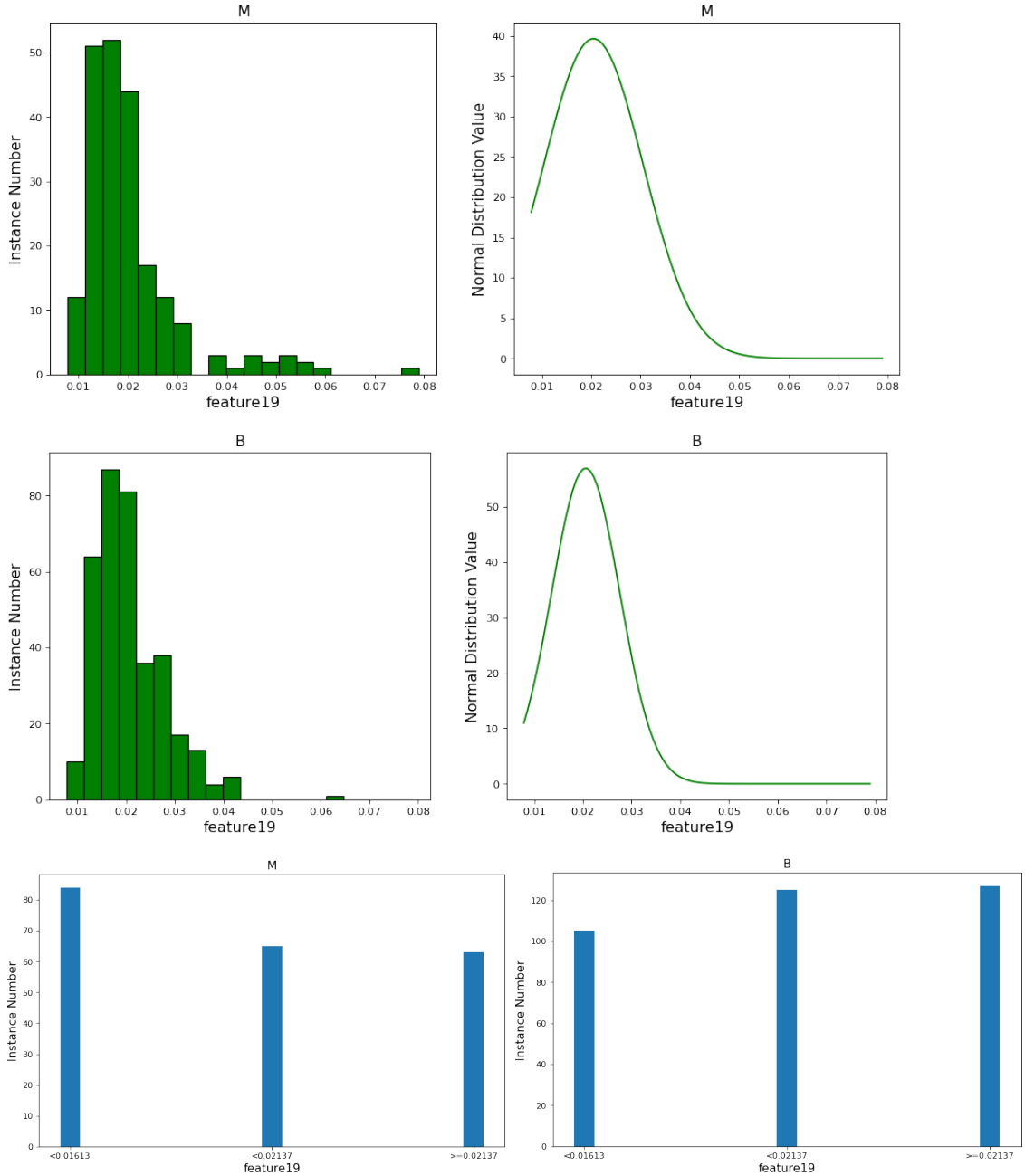


We use the equal frequency method to discretise numeric attributes. These graphs above illustrate the feature1(the first feature)'s distribution among "M" class in the wdbc dataset. The bar chart on the top shows the raw data distribution, with a normal distribution constructed on the data presented by the graph on the right . The bottom chart illustrates the data equally splitted into 3 distributed over class "M".

The normal distribution describes the raw data distribution better in terms of the discretization. Whereas, the 3-fold distribution only informs us how many instances in each interval, and it can not depict the raw distribution. Thus, the entropy in the 3-fold distribution model is low because it knows little about the data, and as a result the discretization would actually worsen the performance of the model.

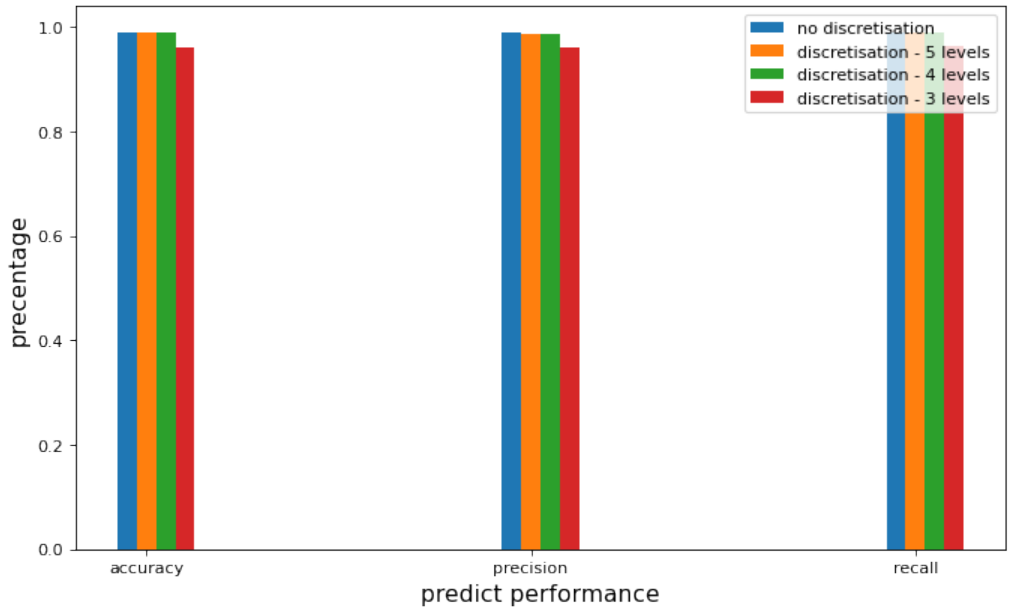


But in fact, discretization does not impose much negative impact on the performance of the model. We get 0.940246 accuracy in no discretisation result and 0.938489 accuray in discretisation result, and the recall/precision is also very close. Why are the accuracy so close? Actually, the model already gets enough information from the 3 folds such that they can tell which class the instance belongs to for most attributes. For example, we can clearly tell the distribution of feature 1 among class “M” and class “B” is different from the following graph. If we find feature 1 is less than 12.25 we can be very sure that the instance belongs to class “B”, or if it is greater than 14.78 then it belongs to class “M”.

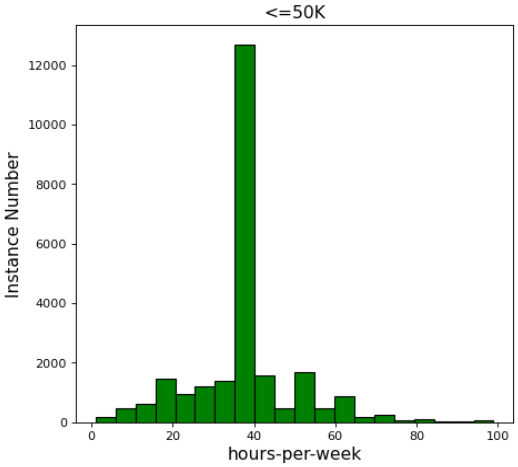


The 3-fold distribution does show its inaccuracy sometimes. In feature 19, the distribution of the 3 folds is quite similar, in other words, we have less confidence to determine which class the instance belongs to for each interval. More specifically, if we found a feature 19 value at 0.05, the instance would probably belong to class ‘M’ , and the normal distribution supports this.

However, 0.05 falls in the interval of >=0.02137 where we get class ‘M’ probability less than class ‘B’. So our 3-fold distribution tells us a wrong result by just looking at feature 19.



As discussed before, the discretisation method gives correct judges between two classes for most attributes, so the performance does not decrease performance much. Moreover, we also test the wine dataset and the performance differences are shown below. As the number of levels increases, the amount of information increases and the performance also goes up.



The equal width method might worsen the performance more because it is very sensitive to outliers. But sometimes the equal frequency method does not work. For example, the feature “hours-per-week” in the adult dataset has a dominant value of “40” and when we apply an equal frequency method on it, we would probably get a meaningless interval (40, 40).

3. Implement a Baseline Model

(Before analysis)Decisions I made:

1). I choose zero_R method to implement the baseline model.

Because this is the most commonly-used baseline in machine learning, and it is pretty easy to implement

2). Also make all the instances as both training set and test set

I tried both using no test data(use all data for both training and evaluating) and splitting data in to 80% training and 20% testing. The results are quit similar, and using no test data can see the performance of the model more clearly.

```
test for Nursery Dataset (no test data)

################## nursery dataset result ##################
accuracy is: 0.903009
error rate is: 0.096991

----------------the macro averaging way------------------
Marco averaging precision is:  0.7248820267369591
Marco averaging recall is: 0.5664257996860103
----------------the micro averaging way------------------
Mirco averaging precision is: 0.9030092592592592
Mirco averaging recall is: 0.903009259259259
----------------the weight averaging way-----------------
weight averaging precision is: 0.905644265528916
weight averaging recall is: 0.9030092592592592
```

```
test for Nursery Dataset (split data into 80% for training and 20% for test)

################## nursery dataset - splited result ##################
accuracy is: 0.940246
error rate is: 0.059754

----------------the macro averaging way------------------
Marco averaging precision is:  0.3757480526457158
Marco averaging recall is: 0.3729057660800169
----------------the micro averaging way------------------
Mirco averaging precision is: 0.9402460456942003
Mirco averaging recall is: 0.9402460456942003
----------------the weight averaging way-----------------
weight averaging precision is: 0.9401589983322217
weight averaging recall is: 0.9402460456942003
```

(Evaluation result of nursery.data)

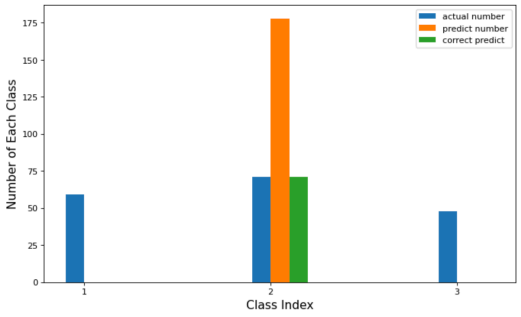
Analysis:

First Half Question(baseline part)

This baseline model(0R) classify all instances as the most common class in the training data.

We can see the how the actual number for each class, and how predict one work on the right. (this dataset choose class 2 as predict class)

Because how this model works, the performance of the model is totally depends on how classes distributed.



Factor 1

The most intuitive reason that affect the performance is the proportion of the highest frequency class in the entire dataset.

If the percentage is high, The accuracy is high.

Theoretically, (if the test distribution is the same to the training distribution)

$$\text{accuracy} = \frac{\text{number of highest frequency class}}{\text{total number of instance}}$$

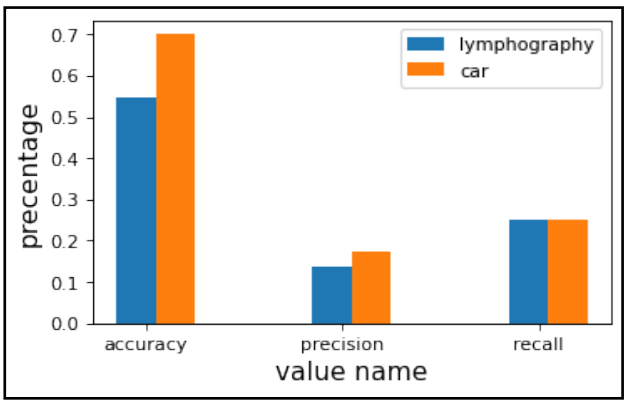
In practice, If I use no test data to evaluate the data, the accuracy is exactly equal to the the proportion of the highest frequency class. (the results for splitting data in 8:2 are also pretty similar). Example:

```
accuracy is: 0.700231
error rate is: 0.299769

----------------the macro averaging way------------------
Marco averaging precision is:  0.17505787037037038
Marco averaging recall is: 0.25
----------------the micro averaging way------------------
Mirco averaging precision is: 0.7002314814814815
Mirco averaging recall is: 0.7002314814814815
----------------the weight averaging way-----------------
weight averaging precision is: 0.4903241276577504
weight averaging recall is: 0.7002314814814815

Class Distribution
number:
[1210, 384, 69, 65]
percentage:
[0.7, 0.222, 0.04, 0.038]
```

The evaluation result of car.data

```
accuracy is: 0.547297
error rate is: 0.452703

----------------the macro averaging way------------------
Marco averaging precision is:  0.13682432432432431
Marco averaging recall is: 0.25
----------------the micro averaging way------------------
Mirco averaging precision is: 0.5472972972972973
Mirco averaging recall is: 0.5472972972972973
----------------the weight averaging way-----------------
weight averaging precision is: 0.2995343316289262
weight averaging recall is: 0.5472972972972973

Class Distribution
number:
[2, 81, 61, 4]
percentage:
[0.014, 0.547, 0.412, 0.027]
```

The evaluation result of lymphography.data



Compare evaluation value between lymphography.data and car.data

In the evaluation result of car.data, there are 4 type of class unacc, acc, good, v-good. The number of instance that the class is unacc is much larger than other 3 class, The percentage is higher, so that the accuracy is higher.

In the evaluation result of lymphography.data, there are also 4 type of class (1)normal find,(2)metastases, (2)malign lymph, (4)fibrosis. The metastases takes up 55% of entire dataset. It is lower than unacc 70% in car.data.So the accuracy of lymphography is lower than car. (which is also shows in the graph)

If the percentage is high, The precision is high.

The formula of the precision is

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \qquad Precision_M = \frac{\sum_{i=1}^{c} Precision(i)}{c}$$

The number of the highest frequency class is the only TP has value in this case. As it increase the precision increase.

In the lymphography.data and car.data example, we can see this is quite obvious (*the precision is in the lined with blue*)

This is also make sense because precision is the value that shows how often is the model correct, when it predicts a positive case. We make the same prediction for the whole dataset with EXACT ONE class, More this class occur in the raw dataset, more accurate the model preform.

If the percentage is high, The recall is also high

The formula of the recall is

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \qquad Recall_\mu = \frac{\sum_{i=1}^{c} TP_i}{\sum_{i=1}^{c} TP_i + FN_i}$$

Similar to the precision, according to the formula, The number of the highest frequency class is the only TP has value in this case, As it increase the recall increase.

We can we can see this is for Micro Averaging and Weight Averaging recall. *(In the purple line)*

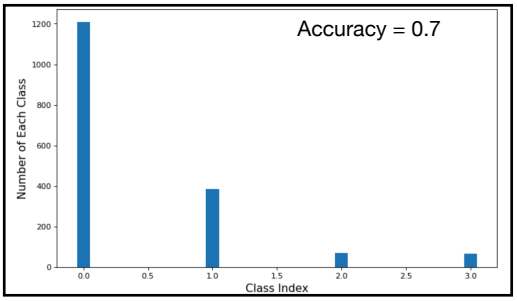
$$Recall_M = \frac{\sum_{i=1}^{c} Recall\ (i)}{c}$$

But in this case, Macro Averaging has some issues, cause except all the predict class column, all the other column will be zero. It will cause Number_of_class recall are all 1 (cause there are no False Negative).
So the proportion of the highest frequency class has no effect to Macro averaging precision in 0R baseline model.
(in the pink square)

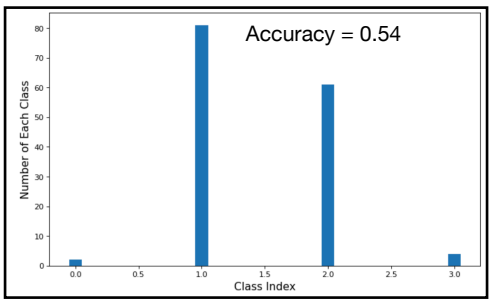
		Predicted		
Actual	Pedestrian	Road	Sidewalk	...
Pedestrian	TP	0 FN	0 FN	...
Road	FP	0 TN	0 TN	...
Sidewalk	FP	0 TN	0 TN	...
...	...	...	...	...

Factor 2

The distribution of number of classes



Car.data class distribution



lymphography.data class distribution

The proportion of the highest frequency class is the key factor that effect the performance of the model, while the distribution of number of classes is the factor to effect the proportion.
In the worst condition, that all classes are uniformly distributed, we have to randomly choose a class as our predict class, the accuracy is 1/number of class type.
Like the lymphography.data is more uniform than the car.data, the accuracy is lower

Factor 3

As going further about the proportion of the highest frequency classes, we can observe that number of class also effect the performance

---

If the number of classes is big, accuracy might be lower

Theoretically,
As we see in the factor2, in the worst case, when all classes are uniformly distributed, the accuracy is 1/ number of class type.
Like the worst accuracy for 2 type of class is 50%
The worst accuracy for 3 type of class is 33.33%
The worst accuracy for 4 type of class is 20%
……
So the conclusion would be: when the number of different class type is relatively uniformly distributed, the number of class increase, the accuracy decrease.

In the practise, the result can also prove that:

```
accuracy is: 0.538462
error rate is: 0.461538

----------------the macro averaging way------------------
Marco averaging precision is:  0.2692307692307692
Marco averaging recall is: 0.5
----------------the micro averaging way------------------
Mirco averaging precision is: 0.5384615384615384
Mirco averaging recall is: 0.5384615384615384
----------------the weight averaging way-----------------
weight averaging precision is: 0.28994082840236685
weight averaging recall is: 0.5384615384615384

Class Distribution
number:
[66, 77]
percentage:
[0.462, 0.538]
```

Evaluation result for somerville.data

```
accuracy is: 0.398876
error rate is: 0.601124

----------------the macro averaging way------------------
Marco averaging precision is:  0.13295880149812733
Marco averaging recall is: 0.3333333333333333
----------------the micro averaging way------------------
Mirco averaging precision is: 0.398876404494382
Mirco averaging recall is: 0.398876404494382
----------------the weight averaging way-----------------
weight averaging precision is: 0.15910238606236585
weight averaging recall is: 0.398876404494382

Class Distribution
number:
[59, 71, 48]
percentage:
[0.331, 0.399, 0.27]
```

Evaluation result for wine.datda

Both someville.data and wine.data has 4 type of class and Both number of different class type are relatively uniformly distributed, but the accuracy of Somerville.data is nearly 15% higher than the wine.data.

```
accuracy is: 0.333333
error rate is: 0.666667

----------------the macro averaging way------------------
Marco averaging precision is:  0.06666666666666667
Marco averaging recall is: 0.2
----------------the micro averaging way------------------
Mirco averaging precision is: 0.3333333333333333
Mirco averaging recall is: 0.3333333333333333
----------------the weight averaging way-----------------
weight averaging precision is: 0.1111111111111111
weight averaging recall is: 0.3333333333333333

Class Distribution
number:
[4320, 2, 328, 4266, 4044]
percentage:
[0.333, 0.0, 0.025, 0.329, 0.312]
```

We can also see that the in the nursery.data, there are 5 class type, The distribution are not uniformly like the wine.data, but it accuracy is 30% lower than the wine.data.

So, the number of class is not a direct cause that effect the accuracy,

but it can potentially effect it.

Evaluation result for nursery.data

If the number of classes effect the Macro Averaging recall

$$Recall_M = \frac{\sum_{i=1}^{c} Recall\ (i)}{c}$$

As we analysis in **Factor 1**. Cause except the predict class column, all the other column will be zero. It will cause Number_of_class recall are all 1 (cause there are no False Neg on ative). So the

$$Recall_M = 1/number\_of\_class$$

According to this, Macro Averaging is not a good way to evaluate recall for 0R baseline.

If the number of classes is big, precision and recall might be lower

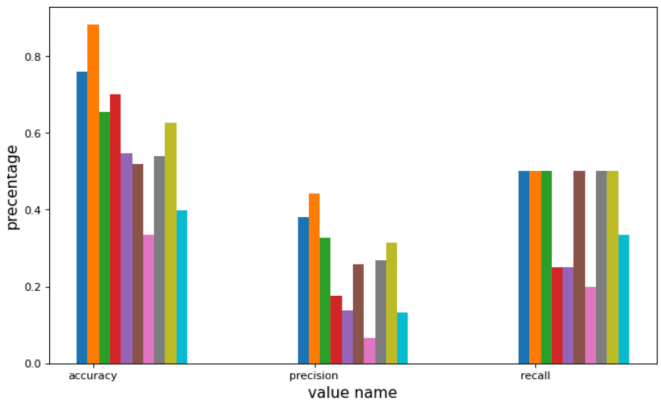
Because the number of class type will potentially effect the accuracy. So, it will potentially effect the number of TP, then the precision and recall will change .
If the number of classes is big, precision and recall might be lower together

Second Half Question(Naive Bayes improves baseline performance)

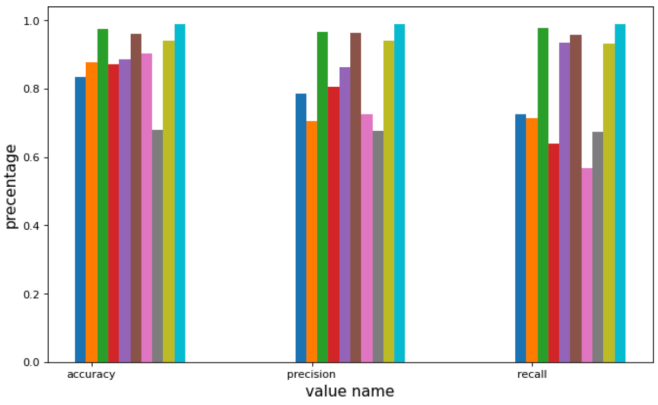
I used the 0R baseline predict the class with the 10 datasets
At the same time, I trained 10 dataset with Naive Bayes classifier, and evaluate the result.
By comparing the results, I can see 3 Improvement points that Naive Bayes classifier has.

Improvement point 1:

Naive Bayes classifier accuracy is more stable than the baseline.



All the evaluation values that implement 0R baseline on 10 datasets



All the evaluation values that implement NB classifier on 10 datasets

The graph on the left is all the evaluation values that implement 0R baseline.

The graph on the right is all the evaluation values implement Naive Bayes classifier.
As we can observe

For the 0R graph, that the range of accuracy is from 0.333 - 0.883

For the Naive babes classifier, the range of accuracy is from 0.678 - 0.989
Also, for the grey bar in the NB graphs is the somerville.data evaluation, This is a dateset with only 143 instance and only 6 attributes. Not enough number of instance and attributes will seriously affect Naie Bayes Classifier accuracy. So the grey bar can be treated as an outlier.
Then the range of the accuracy are all above the 0.8

So Naive Bayes classifier accuracy is more stable than the baseline.

It is not hard to understand. The 0R baseline predict a class only depends on the proportion of the highest frequency class in the entire dataset. So that when proportion of the highest frequency class is high, then the accuracy is high. When the proportion of the highest frequency class is low, the the accuracy is low. The proportion of the highest frequency class varies from dataset to dataset, so that the accuracy of the 0R are not stable.

Improvement point 2:
Baseline does not use the attribute to predict the result, so the accuracy is lower than the Naive Bayes.

The formula of the 0R Baseline:

$$\hat{c} = \arg\max_{c_j \in C} P(c_j)$$

The formula of the Naive Bayes Classifier:

$$\hat{c} = \arg\max_{c_j \in C} P(c_j | x_1, x_2, \dots, x_n)$$

$$= \arg\max_{c_j \in C} P(x_1, x_2, \dots, x_n | c_j) P(c_j)$$

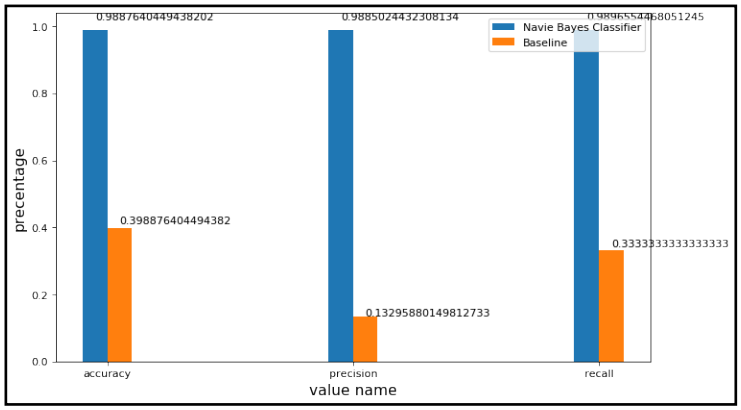
As we can see from the formula. The predict class of 0R is only base on the proportion of the highest frequency class. But the predict class of NB base on both attribute distribution and the class distribution.So when dataset class is uniformly distributed, NB Classifier would be more accurate.
Take the wine.data as example:

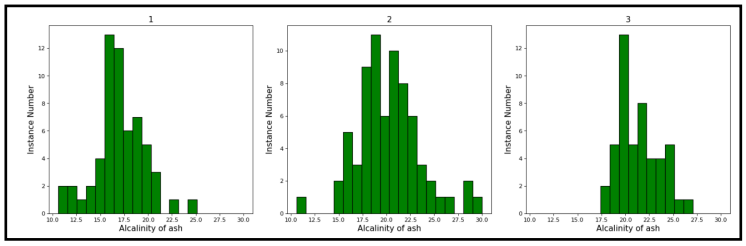
```
accuracy is: 0.988764
error rate is: 0.011236

----------------the macro averaging way------------------
Marco averaging precision is:  0.9885024432308134
Marco averaging recall is: 0.9896554468051245
----------------the micro averaging way------------------
Mirco averaging precision is: 0.9887640449438202
Mirco averaging recall is: 0.9887640449438202
----------------the weight averaging way------------------
weight averaging precision is: 0.9888786975464341
weight averaging recall is: 0.9887640449438202
```

the evaluation result by implement 0R on wine.data

```
accuracy is: 0.398876
error rate is: 0.601124

----------------the macro averaging way------------------
Marco averaging precision is:  0.13295880149812733
Marco averaging recall is: 0.3333333333333333
----------------the micro averaging way------------------
Mirco averaging precision is: 0.398876404494382
Mirco averaging recall is: 0.398876404494382
----------------the weight averaging way------------------
weight averaging precision is: 0.15910238606236585
weight averaging recall is: 0.398876404494382
```

the evaluation result by implement NB classifier on wine.data





An attribute distribution of the different class in wine.data

Class Distribution

number:	percentage:
[59, 71, 48]	[0.331, 0.399, 0.27]

There are three class in dataset. And they are relatively uniformly distributed. So the accuracy of the 0R is very low.

But when we read deeper in the dataset. We can see that nearly all the attribute are normal distribution (An example attribute distribution of the different class in wine.data on the left), which is a very useful information to predict class.

Naive Bayes classifier used this information, make accuracy much higher than 0R baseline model.

The Naive Bayes Classifier use attribute distribution make a more accurate evaluation than the 0R baseline

---

Improvement point 3:

The recall and precision of 0R are actually meaningless, Naive Bayes Classifier's recall and precision give us more information about the dataset and prediction

Precision shows how often is the model correct, when it predicts a positive case
Recall shows what proportion of the true positive cases in the dataset was the model able to detect.

As I explained on the first part of analysis. We know 0R baseline only predict one class (most frequent class) for all dataset. So both precision and recall are meaningless, Because the only information it deliver are still the distribution of the class or even just number of class type.

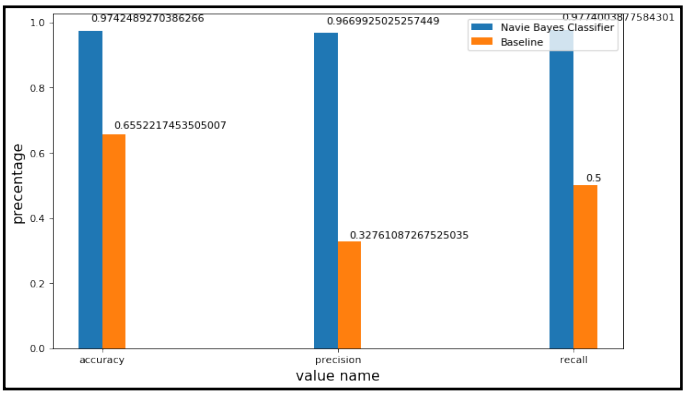
While in the Naive Bayes classifier. It gives us more information about how the model work.
Sometime these two values are very important in some case.
For example, In the breast-cancer-wisconsin.data

```
accuracy is: 0.974249
error rate is: 0.025751

----------------the macro averaging way------------------
Marco averaging precision is:  0.9669925025257449
Marco averaging recall is: 0.9774003877584301
----------------the micro averaging way------------------
Mirco averaging precision is: 0.9742489270386266
Mirco averaging recall is: 0.9742489270386266
----------------the weight averaging way------------------
weight averaging precision is: 0.9751512803459279
weight averaging recall is: 0.9742489270386266
```

Evaluation result of NB classifier

```
accuracy is: 0.655222
error rate is: 0.344778

----------------the macro averaging way------------------
Marco averaging precision is:  0.32761087267525035
Marco averaging recall is: 0.5
----------------the micro averaging way------------------
Mirco averaging precision is: 0.6552217453505007
Mirco averaging recall is: 0.6552217453505007
----------------the weight averaging way------------------
weight averaging precision is: 0.4293155355801564
weight averaging recall is: 0.6552217453505007
```

Evaluation result of 0R baseline




```
Class Distribution
number:
[458, 241]
percentage:
[0.655, 0.345]
```

This dataset breast-cancer-wisconsin.data's class is to evaluate whether the sample cancer are benign or malignant.
Precision shows how often is the model benign, when it predicts a benign case

And the Recall shows what proportion of the true benign cases in the dataset was the model able to detect.
These two values are key to tell people whether they should improve the model or increase the size of dataset, so that it do not give up any one malignant.

In the Naive Bayes Classifier, these two values give use this information
While in 0R baseline, precision only shows the how these two class benign or malignant distributed. Precision = (num_of_ benign/num_of_instance)/2. And the recall only show us how many classes type are there.

Conclusion

The baseline performance varies across datasets, because the proportion of the highest frequency class, the number of class type, and the distribution of number of classes
Naive Bayes classifier improves on the baseline performance,
It make accuracy more stable than the baseline
It use the attribute to predict the result, make accuracy is higher than the baseline.
It makes recall and precision has more information about the dataset and prediction than the baseline

4. Handle Ordinal Data

(Before analysis)Decisions I made:

1). I make all the instances as both training set and test set

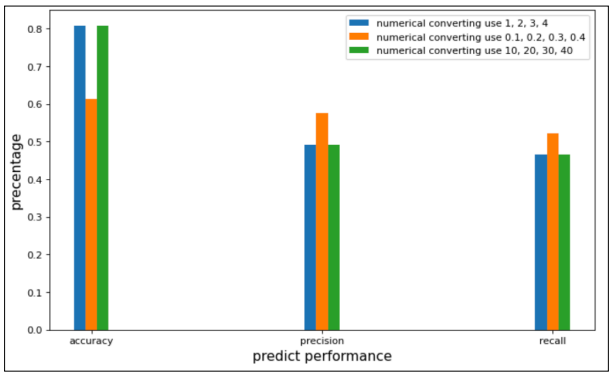
I tried both using no test data(use all data for both training and evaluating) and splitting data in to 80% training and 20% testing. The results are quit similar, and using no test data can see the performance of the model more clearly.

```
test for Nursery Dataset (no test data)

accuracy is: 0.903009
error rate is: 0.096991

----------------the macro averaging way------------------
Marco averaging precision is:  0.7248691953671452
Marco averaging recall is: 0.566428373344821
----------------the micro averaging way------------------
Mirco averaging precision is: 0.9030092592592592
Mirco averaging recall is: 0.9030092592592592
----------------the weight averaging way------------------
weight averaging precision is: 0.9056265890564255
weight averaging recall is: 0.9030092592592593
```

```
test for Nursery Dataset (split data into 80% for training and 20% for test)

accuracy is: 0.896605
error rate is: 0.103395

----------------the macro averaging way------------------
Marco averaging precision is:  0.740087597409457
Marco averaging recall is: 0.5652675395545248
----------------the micro averaging way------------------
Mirco averaging precision is: 0.8966049382716049
Mirco averaging recall is: 0.8966049382716049
----------------the weight averaging way------------------
weight averaging precision is: 0.9029365999980138
weight averaging recall is: 0.896604938271605
```

(Evaluation result of nursery.data)

2). I map the original data to integer 1,2,3,4 according to its order



Result of different converting numeric value of car.data

Cause all the ordinal data has its own order information, the value it mapping should be according to its order.

At the same time, in the ordinal datasets in our assignment. The value that each attribute has are few. So using integer is good.

Also I compare the result to map value to float number, single digit integer, Two-digit integer.The result show the integer gives the highest accuracy. So using integer with gap 1 is the best choice

For example, in car.data, attribute buying, I mapped “v_high”->3, “high"->2, “med"->1, “low”->0

Analysis:

First Half Question (which approach has higher classification accuracy)

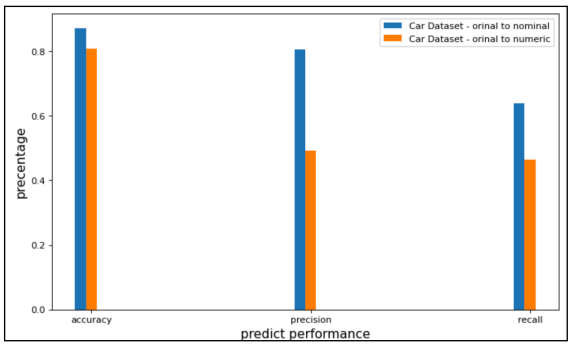
For all the datasets that this assignment provide, there are three ordinal attributes only datasets, which is car.data, nursery.data, somerville.data. (There also some datasets with mix type attribute with ordinal data inside,In order to make the effect of ordinal most obvious, I didn’t use them).

I convert these three datasets values into both nominal variables and numeric variables, and run the Navie Bayes classification on each of them.

By comparing the result, we can say that converting ordinal variables into the nominal variables is the better choice.

The result of car.data

```
################ Car Dataset - orinal to nominal result ################
accuracy is: 0.871528
error rate is: 0.128472

----------------the macro averaging way------------------
Marco averaging precision is:  0.8040352387146505
Marco averaging recall is: 0.6378405556688042
----------------the micro averaging way------------------
Mirco averaging precision is: 0.8715277777777778
Mirco averaging recall is: 0.8715277777777778
----------------the weight averaging way------------------
weight averaging precision is: 0.8688062641518524
weight averaging recall is: 0.8715277777777779
```

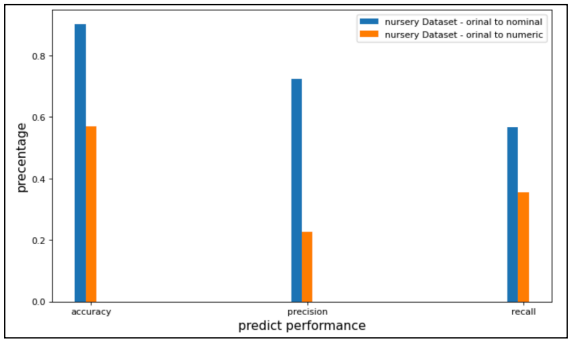



```
################ Car Dataset - orinal to numeric result ################
accuracy is: 0.808449
error rate is: 0.191551

----------------the macro averaging way------------------
Marco averaging precision is:  0.4911436895877529
Marco averaging recall is: 0.46454782196969696
----------------the micro averaging way------------------
Mirco averaging precision is: 0.8084490740740741
Mirco averaging recall is: 0.8084490740740741
----------------the weight averaging way------------------
weight averaging precision is: 0.7668494144356257
weight averaging recall is: 0.8084490740740741
```

The accuracy of nominal one is higher than the numerical one

The result of nursery.data

```
################ nursery Dataset - orinal to nominal result ################
accuracy is: 0.903009
error rate is: 0.096991

----------------the macro averaging way------------------
Marco averaging precision is:  0.7248820267369591
Marco averaging recall is: 0.5664257996860103
----------------the micro averaging way------------------
Mirco averaging precision is: 0.9030092592592592
Mirco averaging recall is: 0.9030092592592592
----------------the weight averaging way------------------
weight averaging precision is: 0.905644265528916
weight averaging recall is: 0.9030092592592592
```

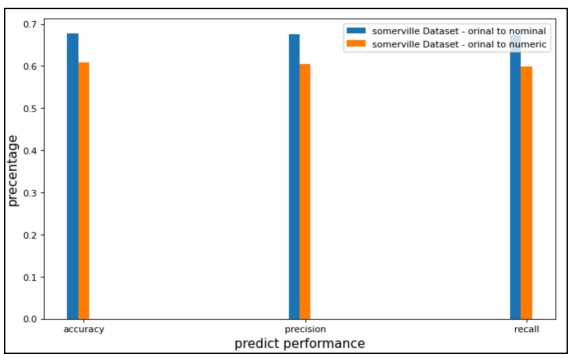



```
################ somerville Dataset - orinal to numeric result ################
accuracy is: 0.608392
error rate is: 0.391608

----------------the macro averaging way------------------
Marco averaging precision is:  0.6052850603412401
Marco averaging recall is: 0.5995670995670996
----------------the micro averaging way------------------
Mirco averaging precision is: 0.6083916083916084
Mirco averaging recall is: 0.6083916083916084
----------------the weight averaging way------------------
weight averaging precision is: 0.6062614040142129
weight averaging recall is: 0.6083916083916083
```

The accuracy of nominal one is higher than the numerical one. And It is much higher, nearly 30%

The result of somerville.data

```
################ somerville Dataset - orinal to nominal result ################
accuracy is: 0.678322
error rate is: 0.321678

----------------the macro averaging way------------------
Marco averaging precision is:  0.6763241736360015
Marco averaging recall is: 0.6742424242424243
----------------the micro averaging way------------------
Mirco averaging precision is: 0.6783216783216783
Mirco averaging recall is: 0.6783216783216783
----------------the weight averaging way------------------
weight averaging precision is: 0.6774806237171829
weight averaging recall is: 0.6783216783216783
```



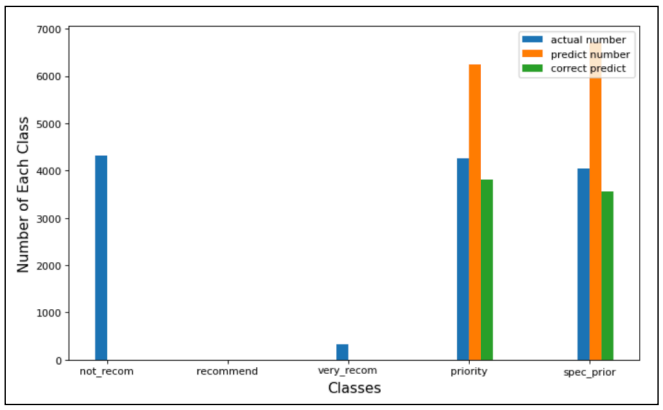

```
################ somerville Dataset - orinal to numeric result ################
accuracy is: 0.608392
error rate is: 0.391608

----------------the macro averaging way------------------
Marco averaging precision is:  0.6052850603412401
Marco averaging recall is: 0.5995670995670996
----------------the micro averaging way------------------
Mirco averaging precision is: 0.6083916083916084
Mirco averaging recall is: 0.6083916083916084
----------------the weight averaging way------------------
weight averaging precision is: 0.6062614040142129
weight averaging recall is: 0.6083916083916083
```

The accuracy of nominal one is still higher than the numerical one.

According to these three comparison pair, We can see that the accuracy of converting to nominal is always higher than converting to numeric. And it is higher about 7% to 30%. So treating the ordinal data as the nominal data is the approach has the higher classification accuracy.

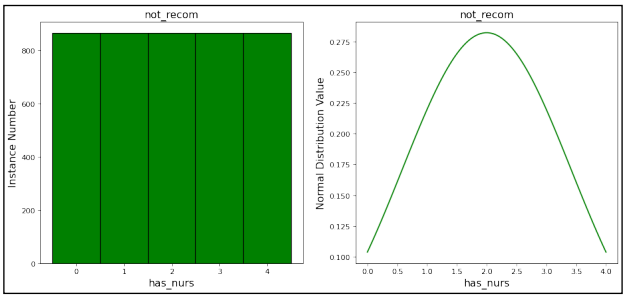
Second Half Question (why)

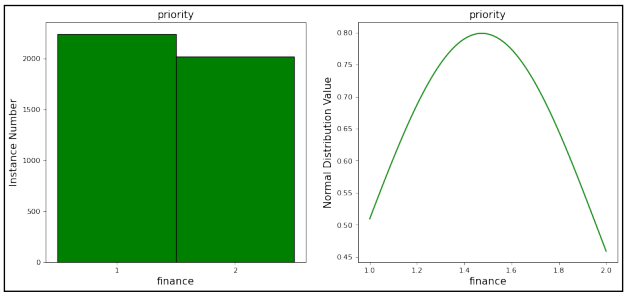
Because, in the nursery.data's, the difference between the two way is the biggest. So I read into the training and testing process.

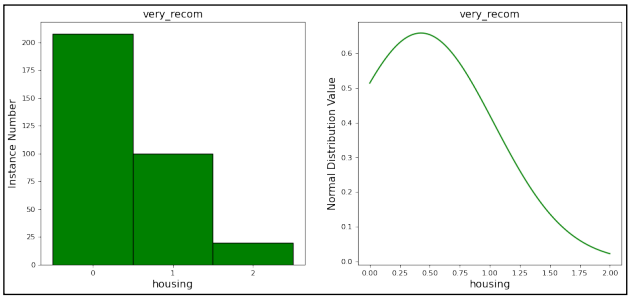


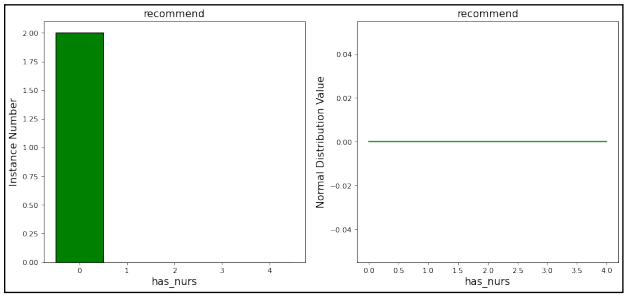
When we print out the number of different class(the actual number, predict number, and the correctly predict number). We can see that for class not_recom, recommend and very_recom, our model predicts none of them.

And when we print out the distribution of the attribute, we can find out the reason







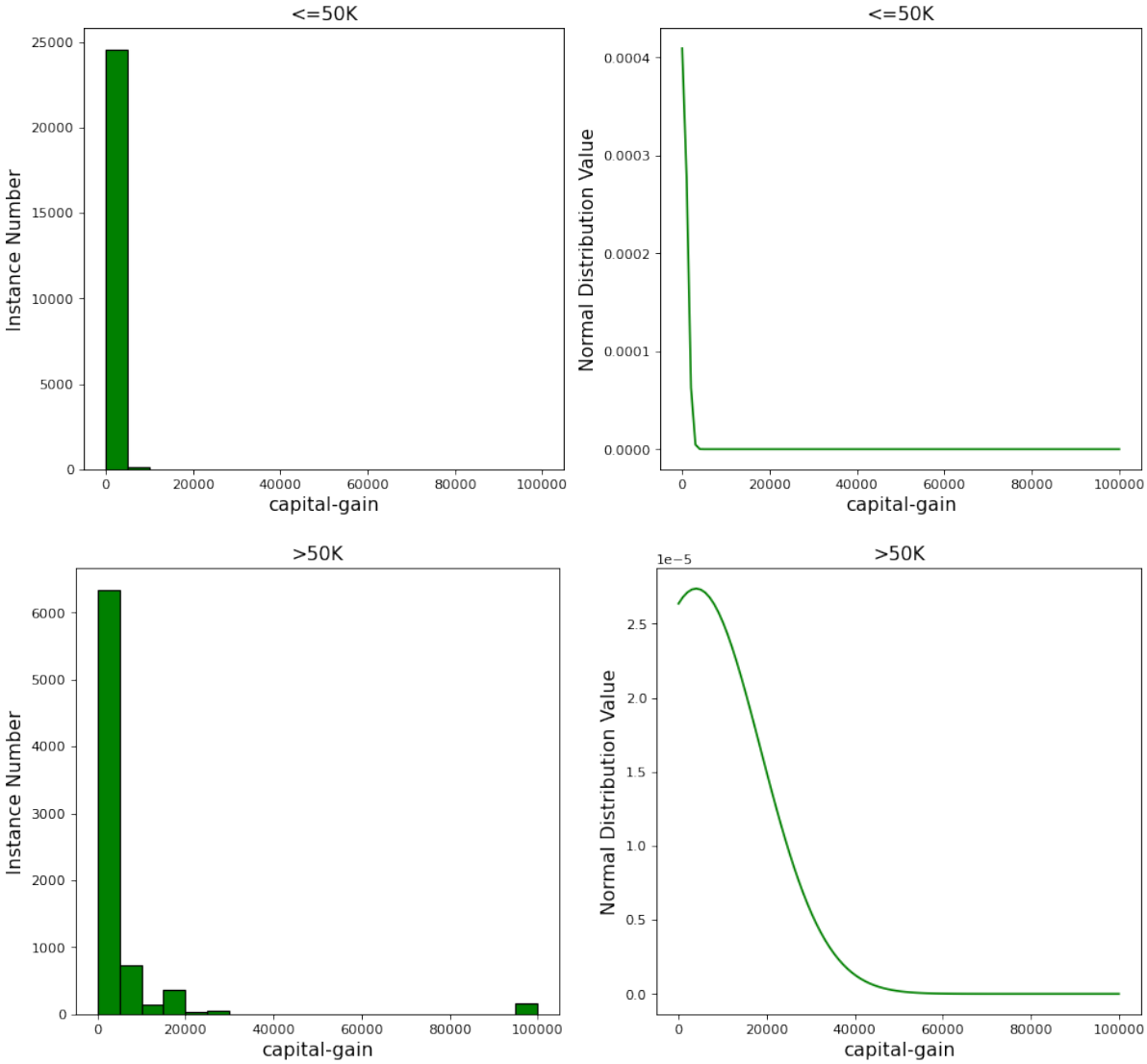


As we can see the the distribution of the attributes are not normal distributed.
Like the has_nurs attribute under not_recom class, and finance attribute under the priority class are the uniform distribution.
Even the housing attribute under very_recom class, it has kind of normal distribution shape, but there are not enough type of value.
And the has_nurs under recommend class, only have only value.

So when we use the normal distribution method to calculate its probability, the result will be not accurate.

Because the ordinal attribute has few values, when it converts to numerical value, it will not match the normal distribution very good. To that extend, The accuracy will be lower than converting to the nominal values.

5. violated Gaussian Assumption



The above distribution is from the feature capital-gain of adult dataset. In both “<=50k” class and “>50k” class almost 90 percent of the instances have 0 capital-gain and a very tiny percentage of them have some capital-gain. Apparently it is inappropriate to apply a Gaussian assumption on it. For most values other than 0, the assumption overestimates the probability.

There is a more dangerous effect. Actually a bunch of rich people have near 100000 capital-gain. If we test the model with such people, the normal distribution function will return 0 for that capital-gain value because it is too extreme, and multiplying by 0 will ruin our Naive Bayes method. So in my implementation, I assign a very small number to replace 0.

In conclusion, if any attribute in the dataset is not naturally distributed and has some extreme outliers, Gaussian assumption could be violated.