CS5483 Group Project Report

Yu SU 55323707
Xinpin Xu 55362327
Peiyu Li 55327269
Zhonghao Yin 55308083

April 2019

1 Project Motivation

We aim to get some important knowledge from a dataset related to a real world problem, then we choose a mobile price classification dataset from kaggle, which consists of 21 attributes and 2000 data.

The website of the dataset is: https://www.kaggle.com/iabhishekofficial/mobile-price-classification, and we plan to use weka and python to analyze and visualize. In addition to get the relation between the mobile price range and parameters, we also want to compare the performance between different classifiers, and then draw the right conclusions.

2 Data Preprocessing

2.1 Data Description

First we check the missing value of each attribute using Python, and found that there is no missing value in this dataset. Then we describe those data with statistics which could show a more intuitionistic information for our data. The introduction of each attribute and the description of data corresponding to each attribute shows blow.

Attribute	Description	Type	Mean	Std.deviation	min	max
battery_power	Total energy a battery can store in one time measured in mAh	Numeric	1.24k	439	501	2k
blue	Has bluetooth or not	Boolean	0.49	0.5	0	1
clock_speed	speed at which microprocessor executes instructions	Numeric	1.52	0.82	0.5	3
dual_sim	Has dual sim support or not	Boolean	0.51	0.5	0	1
fc	Front Camera mega pixels	Numeric	4.31	4.34	0	19
four_g	Has 4G or not	Boolean	0.52	0.5	0	1
int_memory	Internal Memory in Gigabytes	Numeric	32	18.1	2	64
m_dep	Mobile Depth in cm	Numeric	0.5	0.29	0.1	1
mobile_wt	Weight of mobile phone	Numeric	140	35.4	80	200
n_cores	Number of cores of processor	Numeric	4.52	2.29	1	8
pc	Primary Camera mega pixels	Numeric	9.92	6.06	0	20
px_height	Pixel Resolution Height	Numeric	645	444	0	1.96k
px_width	Pixel Resolution Width	Numeric	1.25k	432	500	2k
ram	Random Access Memory in Mega Bytes	Numeric	2.12k	1.08k	256	4k
sc_h	Screen Height of mobile in cm	Numeric	12.3	4.21	5	19
sc_w	Screen Width of mobile in cm	Numeric	5.77	4.36	0	18
talk_time	longest time that a single battery charge will last when you are	Numeric	11	5.46	2	20
three_g	Has 3G or not	Boolean	0.76	0.43	0	1
touch_screen	Has touch screen or not	Boolean	0.5	0.5	0	1
wifi	Has wifi or not	Boolean	0.51	0.5	0	1
price_range	This is the target variable with value of 0(low cost), 1(medium cost), 2(high cost) and 3(very high cost).	Numeric	1.5	1.12	0	3

Figure 1: Data Description

We visualize each attribute to better understanding of dataset show in figure 2 and 3, and try to find whether there are some abnormal data here. And we also could find some information from those plots, for instance, most of mobiles can support 3G and have a low speed of which microprocessor executes instructions.

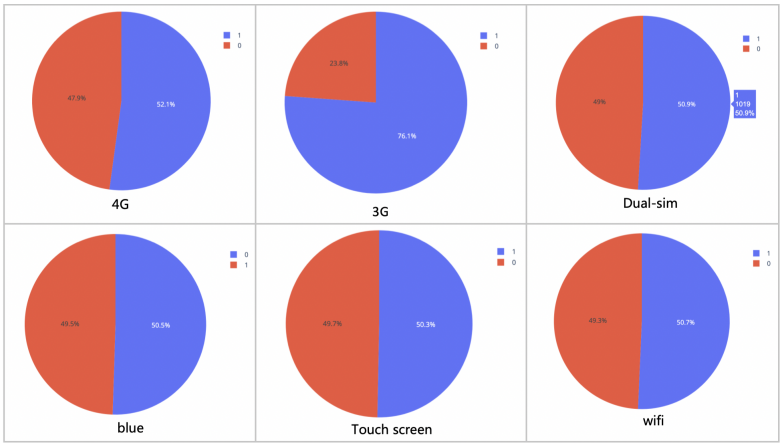

Figure 2: boolean type

2.2 Noise Analysis

From the visualization of weka in figure 4, which shows the discrete point of each corresponding attribute, we can find that all the attribute pairs are smooth. We selected the

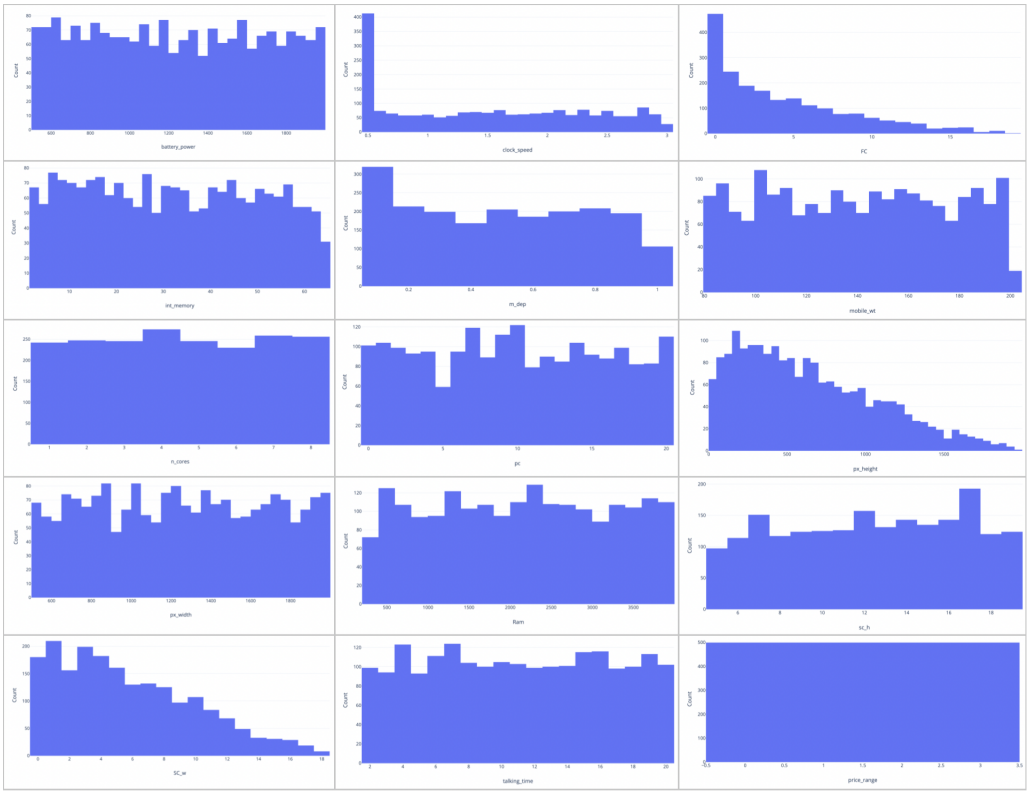

Figure 3: numeric type

price range as the colour to distinguish the influence between each attribute pair. And we delete the noise points. In detail, we delete the instance whose screen width <1 cm and delete the instance whose resolution pixel height <50.

2.3 Relationship between Attributes

We use a heatmap(figure 5) which is computed by correlation coefficient of each attribute to visualize the relationship between them, and found that the attribute of pc and fc, four g and three g are correlated to each other, which means the front camera pixels of most of mobiles are related to its primary camera pixels, and mobiles with 4g may always have 3g. Another distinct correlation is between ram and price range, so the random access memory may have a huge influence of the price range.

3 Association Rules

3.1 Data Preprocessing

At first, the data is numeric, and some of the data is continuous values, such as the camera pixels and the screen sizes, which are not comparable for association rules, because this will cause their support is very low and finally the algorithm will ignore the continuous attribute.

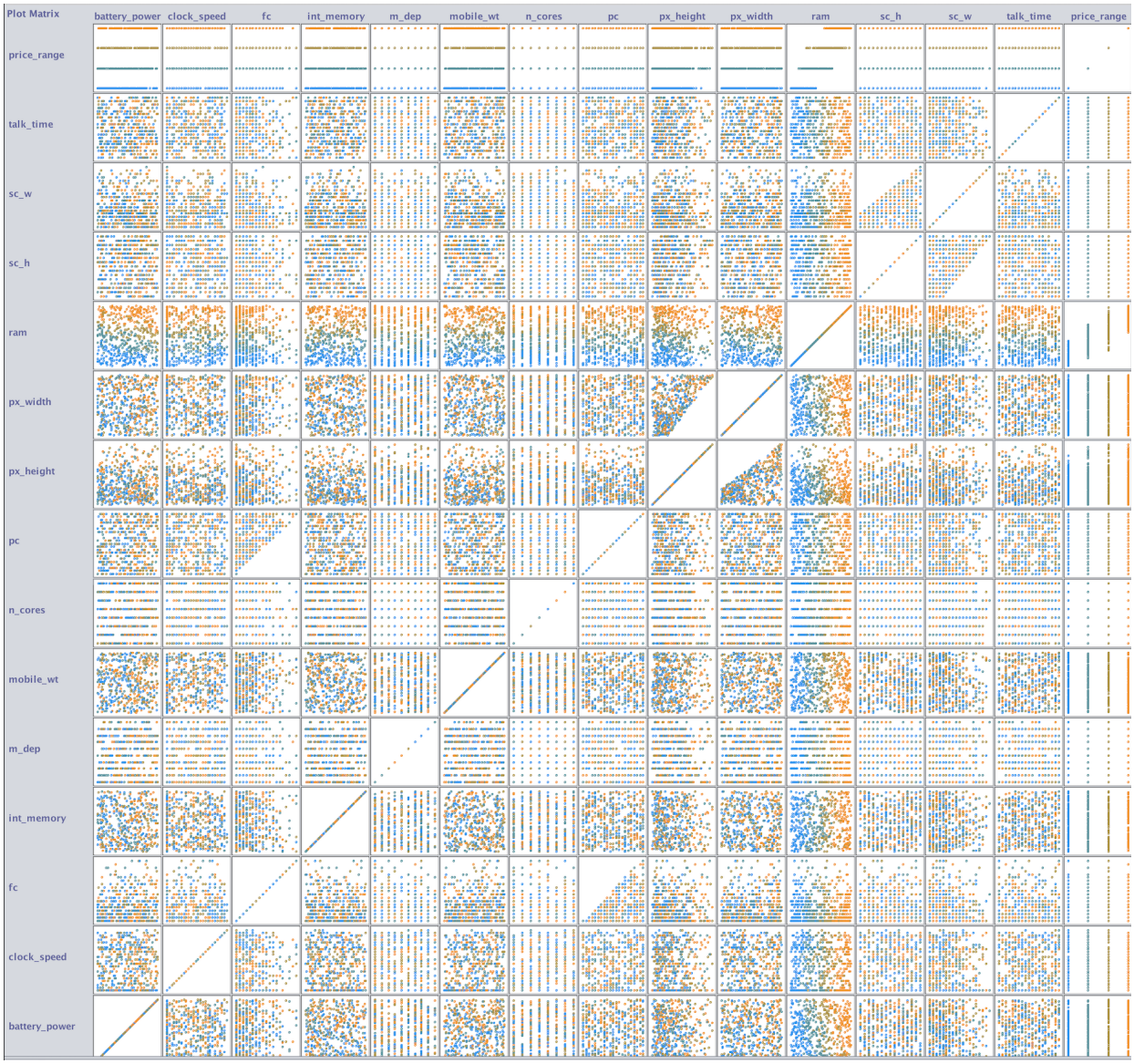

Figure 4: Noise Analysis

So the first thing to do is to discretize the data with equal frequency to make is more fair when do association rule.

Because we want to use two algorithms to extract association rules and the data requirement of FP-Growth is binary, we convert the data into binary data to control the variable.

3.2 Parameter setting

We use three dimension evaluation to make the association rules result better. Using lift as the threshold to find the really meaningful rules. Using support as the threshold to avoid the coincidence. Using convinction to evaluate the pairs.

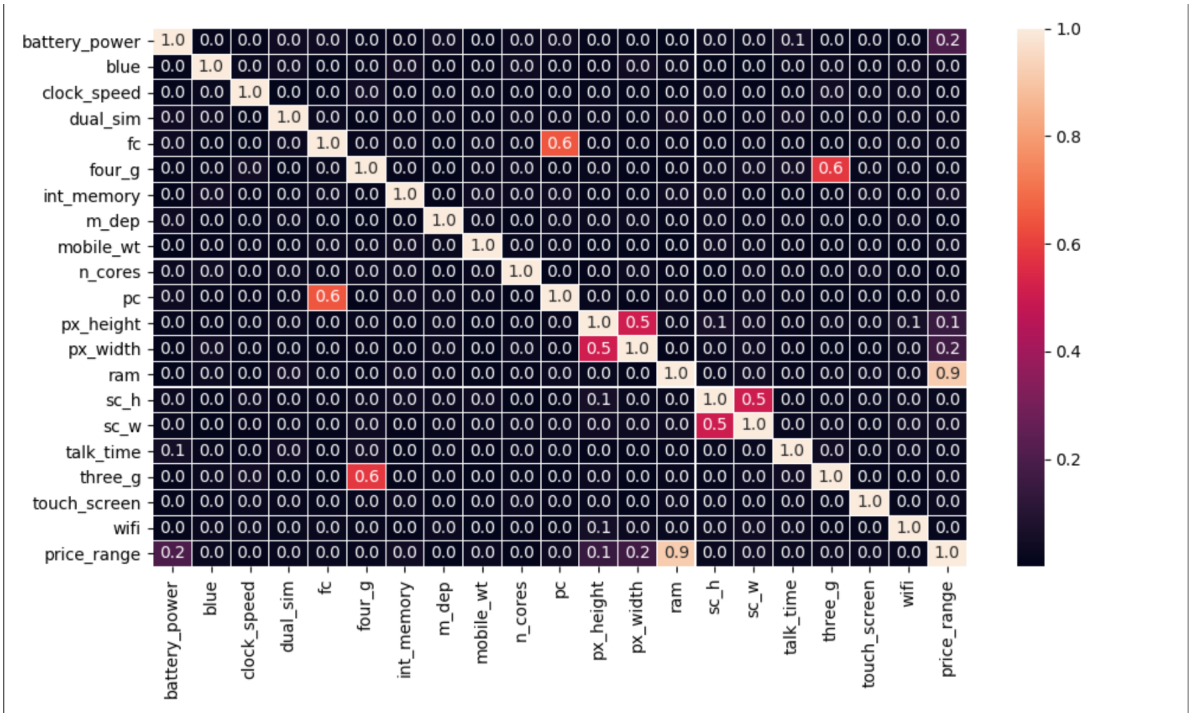

Figure 5: Heatmap for Attribute relationship

3.3 Two Algorithms

3.3.1 Apriori

parameter: support>0.3, lift>1.2
The figure 6 illustrates the result:

- Front camera mega pixel - Primary camera mega pixel
  1. fc='(3.5-inf)'==>pc='(9.5-inf)' 2. pc='(9.5-inf)'==>fc='(3.5-inf)'
  3. fc='(inf-3.5)' ==>pc='(inf-9.5)' 4. pc='(inf-9.5)'==>fc='(inf-3.5)'

- Pixel Resolution Height - Pixel Resolution Width
  5. px_width='(1247.5-inf)'==>px_height='(565.5-inf)'
  6. px_height='(565.5-inf)'==>px_width='(1247.5-inf)'
  9. px_width='(inf-1247.5)' ==>px_height='(inf-565.5)'
  10. px_height='(inf-565.5)' ==>px_width='(inf-1247.5)'

- Screen Height of mobile in cm - Screen Width of mobile in cm
  7. sc_w='(inf-4.5)'==>sc_h='(inf-12.5)'
  8. sc_h='(inf-12.5)' ==>sc_w='(inf-4.5)'
  13. sc_h='(12.5-inf)'==>sc_w='(4.5-inf)'
  14. sc_w='(4.5-inf)'==>sc_h='(12.5-inf)'

- Four g - Three g
  11. four_g=1 ==>three_g=1 12. three_g=1 ==>four_g=1

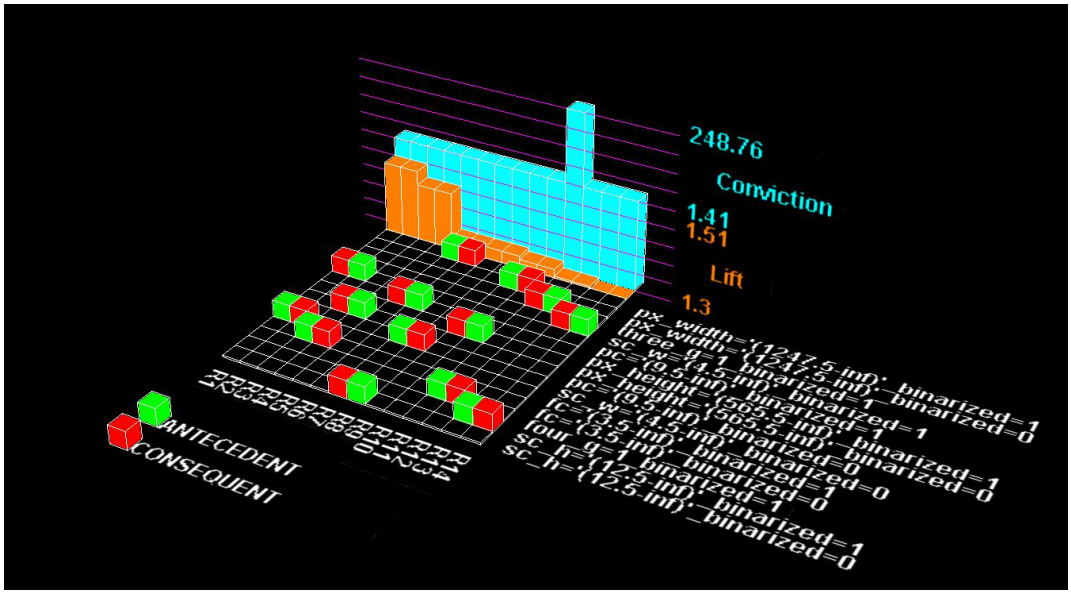

Figure 6: rules for Apriori

3.3.2 FP-growth

To compare with the Apriori, we use the same parameter for FP-growth. parameter: support>0.3, l ift>1.2 *set positiveindex equal to 1 and 2 respectively, because if the value is 2, the 0 value will be treated as missing and if the value is 1, the 1 value will be treated as missing, both of which are not what we want in this situation for missing lots of information.

The figure 7 illustrates the result:

- Front camera mega pixel - Primary camera mega pixel
  1. pc='(9.5-inf)'==>fc='(3.5-inf)' 2. fc='(3.5-inf)'==>pc='(9.5-inf)'

- Pixel Resolution Height - Pixel Resolution Width
  3. px_width='(1247.5-inf)'==>px_height='(565.5-inf)'
  4. px_height='(565.5-inf)'==>px_width='(1247.5-inf)'

- Screen Height of mobile in cm - Screen Width of mobile in cm
  7. sc_w='(4.5-inf) ==>sc_h='(12.5-inf)'
  8. sc_h='(12.5-inf)' ==>sc_w='(4.5-inf)'

- Four g - Three g
  5. three_g=1 ==>four_g=1 6. four_g=1 ==>three_g=1

3.4 comparison

Apriori:
The disadvantage of Apriori is the efficiency, because it needs to generate lots of candidate items. The overhead of IO and the number of scam increase fast with the increase of data.

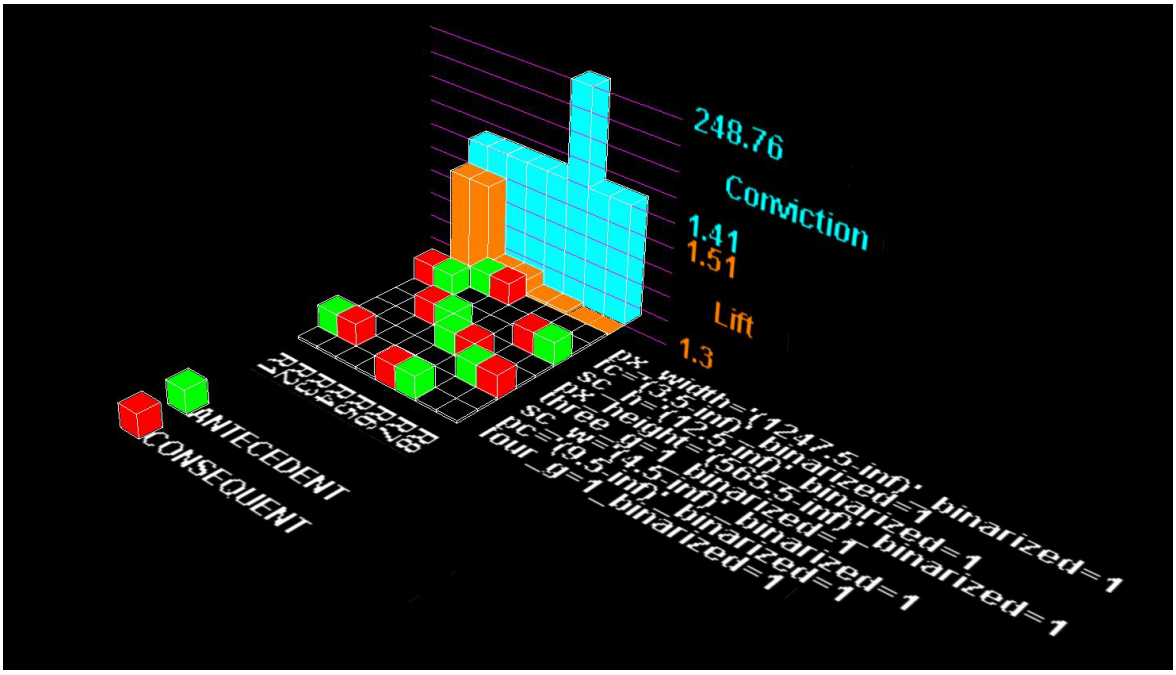

Figure 7: rules for FP-growth

FP-Growth:
FP-Growth solve the problem of Apriori, which is fast by generating FP-tree. In this way, the times of scan are only two. While the disadvantage for FP-Growth is it will regard one side of data as missing, which will lose much information. To avoid missing information, we need to do the algorithm twice in different postiveindex to get the complete rules.

3.5 Conclusion

Finally, we get two 2 types of rules. One is the completely strong rules, like the Front camera mega pixel – Primary camera mega pixel, Pixel Resolution Height – Pixel Resolution Width and Screen Height of mobile in cm – Screen Width of mobile in cm. This kind of rules means the two items are completely related and if the data volume is very large, one of the attribute could be deleted. The other one is the one-way rules, like Four g – Three g, which could only be gotten in one direction or one side and cannot be deleted.

4 Classification

4.1 IBK Classification

By comparing several different classification methods, we can roughly judge different classifiers. The first one is the IBK classification algorithm, which compares the N instances closest to the instance to get the class of the instance. A smaller value means that only training instances that are closer to the input instance will work for the prediction, but it is prone to overfitting; if the N value is large, the advantage is that the estimation error of

learning can be reduced, but the disadvantage is the approximation of learning. The error increases, and the training instance that is farther away from the input instance will also contribute to the prediction, causing the prediction to be erroneous. First, we use the default setting (N=1), and the obtained Correctly Classification rate is 38.3787%.

By changing the value of N, the correlation coefficient changes. When N=25, Correctly Classification rate is 48.8662%.Kappa coefficient is 0.3184, ROC area coefficient is 0.741. The number of instances of a single class is about 500, taking N=50 (10% of instances), Correctly Classification rate, is 51.4739%, Kappa coefficient is 0.3184, ROC area coefficient is 0.754.

The bad performance of IBK is because the instance has more attributes, of which some have no contribute to the classification but could generate interference. The redundant and/or irrelevant attributes can make instances in different classes very close in the sample space, which results in less accurate model. Taking a larger N value can reduce the affects of noisy points, but the results are still ideal. After Attribute Selection, those redundant and/or irrelevant attributes are eliminated, the correctly classified rate improved a lot.

```
Correctly Classified Instances         908               51.4739 %
Incorrectly Classified Instances       856               48.5261 %
Kappa statistic                          0.3531
Mean absolute error                      0.3428
Root mean squared error                  0.4037
Relative absolute error                 91.4327 %
Root relative squared error             93.2324 %
Total Number of Instances             1764

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
               0.665     0.128     0.626       0.665   0.645      0.526    0.922     0.776     0
               0.384     0.206     0.381       0.384   0.383      0.178    0.585     0.288     1
               0.414     0.215     0.395       0.414   0.404      0.196    0.588     0.300     2
               0.598     0.098     0.676       0.598   0.634      0.521    0.922     0.808     3
Weighted Avg.  0.515     0.162     0.520       0.515   0.516      0.355    0.754     0.543

=== Confusion Matrix ===

   a   b   c   d   <-- classified as
 286 115  27   2 |   a = 0
 138 168 107  24 |   b = 1
  30 129 155 103 |   c = 2
   3  29 149 269 |   d = 3
```

Figure 8: Result of IBK50

4.2 Methods of Bayes

Bayes Net:The Bayesian network is a directed acyclic graph with probabilistic annotations. Each node in the graph represents a random variable. If there is an arc between two nodes in the graph, it means that the two nodes correspond to each other. The random variables are probability dependent, and vice versa, the two random variables are conditionally independent.

Naive Bayes:The NBC model requires few parameters to estimate, is less sensitive to missing data, and the algorithm is simpler. The NBC model assumes that the attributes are independent of each other that is not in fact.When the number of attributes is large or the correlation between attributes is large, the classification efficiency of the NBC model is inferior to that of the decision tree model. The performance of the NBC model is the best when the attribute correlation is small.

The BayesNet classification algorithm has a Correctly Classification rate of 76.4172%, a Kappa coefficient of 0.6856, and a ROC area coefficient of 0.935. Another Bayesian algorithm is the NavieBayes algorithm, the Correctly Classification rate is 79.1383%, the Kappa coefficient is 0.7218, and the ROC area coefficient is 0.949. The classification algorithm is more accurate than BayesNet.

```
Correctly Classified Instances        1348               76.4172 %
Incorrectly Classified Instances       416               23.5828 %
Kappa statistic                          0.6856
Mean absolute error                      0.1694
Root mean squared error                  0.2858
Relative absolute error                 45.1715 %
Root relative squared error             66.0144 %
Total Number of Instances             1764

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0.833    0.035    0.884      0.833    0.857      0.814    0.983     0.948     0
               0.760    0.130    0.657      0.760    0.705      0.601    0.900     0.670     1
               0.629    0.099    0.682      0.629    0.654      0.544    0.881     0.681     2
               0.838    0.049    0.853      0.838    0.845      0.793    0.976     0.924     3
Weighted Avg.  0.764    0.079    0.769      0.764    0.765      0.687    0.935     0.805

=== Confusion Matrix ===

   a   b   c   d   <-- classified as
 358  72   0   0 |   a = 0
  47 332  58   0 |   b = 1
   0 101 281  65 |   c = 2
   0   0  73 377 |   d = 3
```

Figure 9: Result of Bayes Net

```
Correctly Classified Instances        1396               79.1383 %
Incorrectly Classified Instances       368               20.8617 %
Kappa statistic                          0.7218
Mean absolute error                      0.1515
Root mean squared error                  0.2675
Relative absolute error                 40.4136 %
Root relative squared error             61.7809 %
Total Number of Instances             1764

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0.886    0.033    0.896      0.886    0.891      0.856    0.989     0.969     0
               0.691    0.099    0.697      0.691    0.694      0.594    0.917     0.698     1
               0.711    0.111    0.685      0.711    0.698      0.593    0.906     0.698     2
               0.878    0.036    0.894      0.878    0.886      0.847    0.985     0.962     3
Weighted Avg.  0.791    0.070    0.793      0.791    0.792      0.722    0.949     0.831

=== Confusion Matrix ===

   a   b   c   d   <-- classified as
 381  48   1   0 |   a = 0
  44 302  91   0 |   b = 1
   0  82 318  47 |   c = 2
   0   1  54 395 |   d = 3
```

Figure 10: Result of Naive Bayes

4.3 Decision Tree

J48:The decision tree is also a classification algorithm. In weka we use the J48 algorithm. The default minObjnum is 2, the Correctly Classification rate is 84.4671%, the Kappa coefficient is 0.7929, and the ROC area coefficient is 0.918.Change the number of minnumObj, the corresponding coefficient is rising, indicating that the classification result is better, and the size of tree also decreases. But the best result is when minnumObj is 3.

```
Correctly Classified Instances        1497               84.8639 %
Incorrectly Classified Instances       267               15.1361 %
Kappa statistic                          0.7982
Mean absolute error                      0.0853
Root mean squared error                  0.2649
Relative absolute error                 22.743  %
Root relative squared error             61.1747 %
Total Number of Instances             1764

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0.916    0.034    0.895      0.916    0.906      0.875    0.965     0.888     0
               0.801    0.059    0.818      0.801    0.809      0.747    0.905     0.752     1
               0.783    0.068    0.797      0.783    0.790      0.720    0.890     0.720     2
               0.896    0.041    0.882      0.896    0.889      0.850    0.955     0.873     3
Weighted Avg.  0.849    0.051    0.848      0.849    0.848      0.798    0.929     0.808

=== Confusion Matrix ===

   a   b   c   d   <-- classified as
 394  36   0   0 |   a = 0
  45 350  42   0 |   b = 1
   1  42 350  54 |   c = 2
   0   0  47 403 |   d = 3
```

Random Forest:Random forest is a classifier that contains multiple decision trees, and the output category is determined by the mode of the category of the individual tree output.The Correctly Classification rate is 88.8322%, the Kappa coefficient is 0.8511, and the ROC area

coefficient is 0.984.

```
Correctly Classified Instances        1567               88.8322 %
Incorrectly Classified Instances       197               11.1678 %
Kappa statistic                          0.8511
Mean absolute error                      0.1693
Root mean squared error                  0.2405
Relative absolute error                 45.1431 %
Root relative squared error             55.5431 %
Total Number of Instances             1764

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0.960    0.025    0.924      0.960    0.942      0.923    0.996     0.985     0
               0.835    0.048    0.851      0.835    0.843      0.792    0.975     0.910     1
               0.823    0.051    0.846      0.823    0.834      0.779    0.969     0.904     2
               0.936    0.024    0.929      0.936    0.932      0.909    0.995     0.987     3
Weighted Avg.  0.888    0.037    0.887      0.888    0.888      0.851    0.984     0.947

=== Confusion Matrix ===

   a   b   c   d   <-- classified as
 413  17   0   0 |   a = 0
  34 365  38   0 |   b = 1
   0  47 368  32 |   c = 2
   0   0  29 421 |   d = 3
```

Figure 12: Result of Random Forest

4.4 Attributes Selection

We use the select attributes function of weka to generate the best attributes subset. We combine CfsSubsetEval with BestFirst and select four attributes which are battery_power, px_height, ram and touch_screen. And we use the new dataset for classification. In weka experiment environment, we select five different types of representative classification algorithms.

4.5 Performance in Weka Experiment environment

We can clearly see that the performance of the IBK algorithm is greatly improved after the attribute selection. The improvement of the two decision tree algorithms makes you not obvious. The performance of the two Bayesian algorithms even decreases slightly.

```
Dataset                   (1) lazy.IBk '-K 1 | (2) lazy.IBk '- (3) lazy.IBk '- (4) bayes.Bayes (5) bayes.Naive (6) trees.J48 ' (7) trees.J48 ' (8) trees.Rando
-------------------------------------------------------------------------------------------------------------------------------------------------------------
'train-weka.filters.unsup(100)   38.20(3.65) |   48.13(2.92) v   50.80(3.60) v   76.58(2.92) v   79.62(2.51) v   83.75(2.85) v   84.06(2.94) v   89.10(2.35) v
-------------------------------------------------------------------------------------------------------------------------------------------------------------
                                        (v/ /*) |         (1/0/0)         (1/0/0)         (1/0/0)         (1/0/0)         (1/0/0)         (1/0/0)         (1/0/0)
```

Figure 13: Result before Attribute Selection

```
Dataset                   (1) lazy.IBk '-K 1 | (2) lazy.IBk '- (3) lazy.IBk '- (4) bayes.Bayes (5) bayes.Naive (6) trees.J48 ' (7) trees.J48 ' (8) trees.Rando
-------------------------------------------------------------------------------------------------------------------------------------------------------------
'train-weka.filters.unsup(100)   83.53(2.71) |   87.25(2.54) v   85.69(2.80) v   75.53(2.77) *   78.56(2.50) *   84.85(2.69)     84.96(2.64)     87.60(2.68) v
-------------------------------------------------------------------------------------------------------------------------------------------------------------
                                        (v/ /*) |         (1/0/0)         (1/0/0)         (0/0/1)         (0/0/1)         (0/1/0)         (0/1/0)         (1/0/0)
```

Figure 14: Result after Attribute Selection

4.6 Comparison

By comparison, the best classifier for this data set is the Random Forest decision tree, followed by J48 Decision Tree. Then Naive Bayes and Bayes Net, And the performance of these two Bayes Methods are close. and the worst classifier is IBk. The classification results

obtained have no reference value. The accuracy of classification increases. Kappa coefficient and ROC will rise, and they are positively related.

Methods	Correctly Classified rate	Kappa	TP	FP	Precision	Recall	ROC	Root mean squared error
IBK1	38.3787 %	0.1781	0.384	0.206	0.399	0.384	0.596	0.5544
IBK25	48.8662 %	0.3184	0.489	0.170	0.498	0.489	0.741	0.4019
IBK50	51.4739 %	0.3531	0.515	0.162	0.520	0.515	0.754	0.4037
Bayes Net	76.4172 %	0.6856	0.764	0.079	0.769	0.764	0.935	0.2858
Naive Bayes	79.1383 %	0.7218	0.791	0.070	0.793	0.791	0.949	0.2675
J48-2	84.4671 %	0.7929	0.845	0.052	0.844	0.845	0.918	0.271
J48-3	84.8639 %	0.7982	0.849	0.051	0.848	0.849	0.929	0.2649
J48-4	84.4671 %	0.7929	0.845	0.052	0.843	0.845	0.941	0.2613
Random Forest	88.8322 %	0.8511	0.888	0.037	0.887	0.888	0.984	0.2405

Figure 15: Comparison of Each Method

5 Regression

5.1 Methods Selection

The data in our data set is discrete rather than continuous, so we cannot use linear regression. We choose two logistic regression model in Weka: Logistic and SimpleLogistic.

SimpleLogistic is a symmetric model whereas Logistic is not, and both of them fit multinomial logistic regression problem. SimpleLogistic fits a multinomial logistic regression model using the LogitBoost algorithm which uses a symmetric model. In each iteration, it adds one SimpleLinearRegression model per class into the logistic regression model. And if the number of iterations is large, the results of this two methods maybe the same.

In addition, SimpleLogistic has a built-in attribute selection, but Logistic aims to fit a full multinomial logistic regression model based on all attributes.

5.2 SimpleLogistic

We use Cross-validation with 10 Folds for both of these two regression methods. The result is shown below:

- Class 0: 66.67 + battery_power * -0.02 + int_memory * -0.01 + mobile_wt * 0.02 + n_cores * -0.04 + px_height * -0.01+ px_width * -0.01 + ram * -0.03 + support_wifi * 0.53

- Class 1: 23.43 + battery_power * -0.01 + support_bluetooth * -0.06 + clock_speed * -0.09 + support_dual_sim * -0.07 + m_dep * 0.5 + mobile_wt * 0.01 + n_cores * -0.05 + pc * -0.01 + ram * -0.01 + sc_w * -0.03 + talk_time * -0.01 + support_three_g * -0.22 + support_wifi * 0.29

- Class 2: -25.51 + support_bluetooth * 0.08 + clock_speed * -0.08 + support_dual_sim * -0.18 + support_four_g * -0.28 + n_cores * -0.02 + ram * 0.01 + sc_h * -0.02 + support_three_g * 0.08 + has_touch_screen * -0.19 + support_wifi * -0.24

- Class 3: -95.79 + battery_power * 0.01 + int_memory * 0.03 + mobile_wt * -0.03 + n_cores * 0.05 + px_height * 0.01 + px_width * 0.01 + ram * 0.02 + sc_h * 0.07 + support_wifi * -0.24

From the above formulas, we can see that the SimpleLogistic model selects several attributes for each class. Wifi has the biggest impact for Class0 and it also has a big impact for Class1. The mobile depth has biggest impact for Class1 whereas it has less effect for others. Touch screen has a big influence on Class2. The formulas of Class 0 and Class 2 show a association rule in Section 3 that the weight of px_height and px_width is the same.

In conclusion, the network condition(wifi,4g,3g) is closely related to the price of mobile phone. Although the weight the phone weight, the number of cpu cores, the battery power and the storage space are small, they have a wide range of influence. The accuracy and the confusion matrix of this model is shown in Figure 16.

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        1711               96.9955 %
Incorrectly Classified Instances        53                3.0045 %
Kappa statistic                          0.9599
Mean absolute error                      0.0459
Root mean squared error                  0.1209
Relative absolute error                 12.2435 %
Root relative squared error             27.9108 %
Total Number of Instances             1764

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0.984    0.006    0.981      0.984    0.983      0.977    1.000     0.999     0
               0.959    0.014    0.957      0.959    0.958      0.944    0.998     0.995     1
               0.953    0.013    0.962      0.953    0.957      0.943    0.998     0.995     2
               0.984    0.007    0.980      0.984    0.982      0.976    1.000     0.999     3
Weighted Avg.  0.970    0.010    0.970      0.970    0.970      0.960    0.999     0.997

=== Confusion Matrix ===

   a   b   c   d   <-- classified as
 423   7   0   0 |   a = 0
   8 419  10   0 |   b = 1
   0  12 426   9 |   c = 2
   0   0   7 443 |   d = 3
```

Figure 16: The accuracy and the confusion matrix of SimpleLogistic

5.3 Logistic

Unlike SimpleLogistic, Logistic give coefficients and odds ratios of all the attributes which represent some knowledge not shown by SimpleLogistic. The odds ratio is a simple and straightforward way to tell us how much each attribute affects a certain type of forecast.

From the Figure 17 and Figure 18, we could see that all the classes care about the network condition, bluetooth and touch screen. Class 0 and 1 concern the depth of mobile phone. It is interesting that the number of cores of processor almost has no impact for classification, and only Class 0 cares a lot about the speed at which microprocessor executes instructions.

Also, the figures show the relation of px_height and px_width since the coefficients and odds ratios of these two attributes are nearly the same for each class. The accuracy and the

confusion matrix of this model is shown in Figure 19.

Logistic Regression with ridge parameter of 1.0E-8
Coefficients...

Variable	Class 0	1	2
battery_power	-0.6381	-0.4481	-0.2189
blue_binarized=1	4.0574	4.9536	6.7712
clock_speed	2.96	-1.197	0.4752
dual_sim_binarized=1	-8.2155	-5.6549	-4.2615
fc	2.3974	1.5851	0.8109
four_g_binarized=1	10.3064	9.1885	-2.686
int_memory	-0.9971	-0.7948	-0.5439
m_dep	9.5241	14.3588	0.5859
mobile_wt	1.4435	0.9964	0.6408
n_cores	-5.7929	-3.9829	-2.5874
pc	-2.0194	-1.8595	-0.9785
px_height	-0.3978	-0.2823	-0.1508
px_width	-0.376	-0.2691	-0.1335
ram	-1.041	-0.724	-0.3629
sc_h	-1.3911	-1.6925	-2.1359
sc_w	-0.3262	-0.7092	-0.046
talk_time	-0.3413	-0.6692	-0.2235
three_g_binarized=1	-3.343	-9.2184	1.3217
touch_screen_binarized=1	20.7101	17.5331	6.1995
wifi_binarized=1	31.9614	28.3227	11.8209
Intercept	3611.2807	2866.6572	1604.9682

Figure 17: The coefficients of attributes

Odds Ratios...

Variable	Class 0	1	2
battery_power	0.5283	0.6389	0.8034
blue_binarized=1	57.8228	141.68	872.3612
clock_speed	19.2986	0.3021	1.6084
dual_sim_binarized=1	0.0003	0.0035	0.0141
fc	10.995	4.88	2.25
four_g_binarized=1	29923.7741	9783.8646	0.0682
int_memory	0.3689	0.4517	0.5805
m_dep	13685.3638	1721604.0636	1.7965
mobile_wt	4.2354	2.7086	1.898
n_cores	0.003	0.0186	0.0752
pc	0.1327	0.1558	0.3759
px_height	0.6718	0.754	0.86
px_width	0.6866	0.764	0.8751
ram	0.3531	0.4848	0.6956
sc_h	0.2488	0.1841	0.1181
sc_w	0.7217	0.4921	0.9551
talk_time	0.7108	0.5121	0.7997
three_g_binarized=1	0.0353	0.0001	3.7498
touch_screen_binarized=1	986950222.0699	41164535.5694	492.5232
wifi_binarized=1	7.597153807486972E13	1.99716468766442E12	136071.1987

Figure 18: The odds ratios of attributes

5.4 Comparison

It is clearly that the Logistic has better performance than SimpleLogistic. Specifically in the following aspects:

- The accuracy of Logistic is a bit higher.

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        1713               97.1088 %
Incorrectly Classified Instances        51                2.8912 %
Kappa statistic                          0.9614
Mean absolute error                      0.0143
Root mean squared error                  0.1168
Relative absolute error                  3.8081 %
Root relative squared error             26.9755 %
Total Number of Instances             1764

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
               0.977     0.002     0.993      0.977     0.985      0.980    1.000     0.999     0
               0.975     0.013     0.962      0.975     0.968      0.958    0.999     0.996     1
               0.960     0.015     0.955      0.960     0.958      0.943    0.998     0.994     2
               0.973     0.008     0.976      0.973     0.974      0.966    0.999     0.998     3
Weighted Avg.  0.971     0.010     0.971      0.971     0.971      0.961    0.999     0.997

=== Confusion Matrix ===

   a   b   c   d   <-- classified as
 420  10   0   0 |   a = 0
   3 426   8   0 |   b = 1
   0   7 429  11 |   c = 2
   0   0  12 438 |   d = 3
```

Figure 19: The accuracy and the confusion matrix of Logistic

- For SimpleLogistic model, the relative absolute error and mean absolute error is higher, so the stability of it is worse than Logistic.
- SimpleLogistic model has a built-in attribute selection which may let the model to ignore some special attributes such as the clock_speed for Class 0.
- The Margin Curve of Logistic model is better than SimpleLogistic(shown in Figure 20).

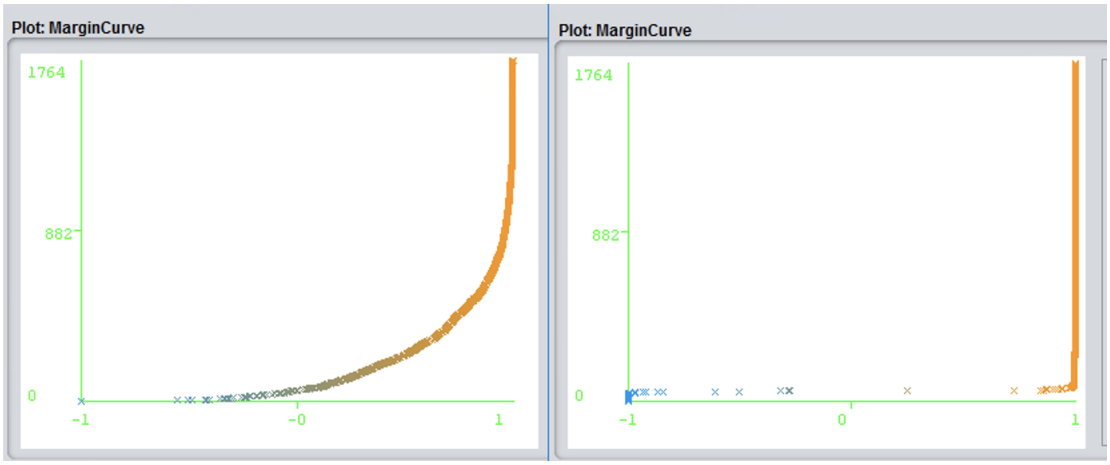

Figure 20: The margin curve of two models(left:SimpleLogistic,right:Logistic)

6 Prediction

We collect information about several mobile phones as the test set to predict the mobile price according to their performance, and selected the best performance with highest accuracy from section 4 and section 5 to be the training model, which is Random Forest and Logistic respectively. The result is shown in the figure 21. Through the result combined with the real world, we could find the practical prediction accuracy of classification model is higher than the regression model, which just verified that the logistic regression does not fit the data with many attributes.

id	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	pc	px_height	px_width	ram	sc_h	sc_w	talk_time	three_g	touch_screen	wifi
iphone 8 p	2691	1	2.4	0	7	1	256	0.8	202	6	24	1080	1920	3000	16	8	21	1	1	1
iphone 7	1960	1	2.34	0	7	1	128	0.7	138	4	12	750	1334	2000	14	7	14	1	1	1
iphone 6s	1715	1	1.84	0	5	1	64	0.7	143	2	12	750	1334	2000	14	7	14	1	1	1
redmi 3p	4100	1	1.5	1	5	1	32	0.8	144	4	13	720	1280	3000	14	7	20	1	1	1
mi 5s	3200	1	2.15	1	4	1	64	0.8	145	4	12	1080	1920	3000	15	7	18	1	1	1
redmi 4	4100	1	1.4	1	5	1	16	0.9	156	8	13	720	1080	2000	14	7	20	1	1	1
huawei mate 9	4000	1	2.4	1	8	1	64	0.8	190	8	20	1080	1920	4000	17	8	32	1	1	1
huawei mate 8	4000	1	2.3	1	8	1	64	0.8	185	8	16	1080	1920	3000	16	8	28	1	1	1
huawei honor 8	3000	1	2.3	1	8	1	64	0.8	153	8	12	1080	1920	4000	15	7	19	1	1	1

iphone 8 puls	iphone 7	iphone 6	redmi 3p	mi 5s	redmi 4	Hawei mate 9	Hawei mate 8	Honor 8
3	2	2	3	3	1	3	3	3
3	2	2	3	3	3	3	3	3

Figure 21: predicted mobile price