Chronic Kidney Disease Prediction Using Machine Learning Techniques

Wenting Li

Overview

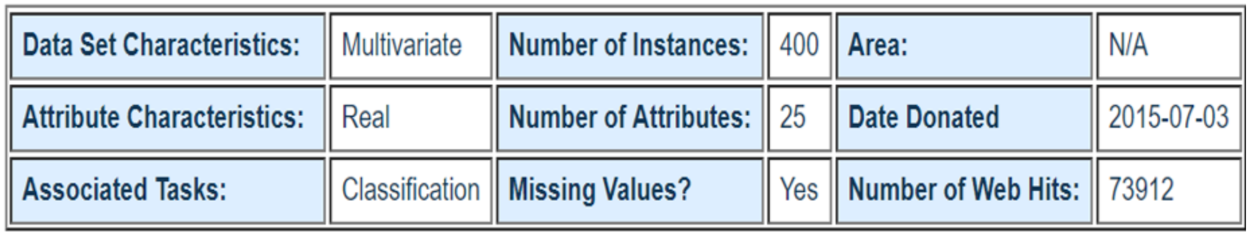
Chronic kidney disease is also called chronic kidney failure. It describes the gradual loss of kidney function. Chronic kidney disease reduces the kidney's ability to function. If it gets worse, wastes in blood will build up to high levels leading to sickness of the patient. Patients may develop symptoms like anemia, high blood pressure, poor nutritional health, weak bones, etc. As kidney disease progresses, it may lead to kidney failure eventually. Then dialysis or a kidney transplant is required to maintain life.

There are some indexes that doctors can rely on to determine whether the patient has chronic kidney disease. Some of the indexes include: glomerular filtration rate, protein in urine, blood pressure, etc. It is important for doctors to give accurate prediction based on these. Early prediction and detection can help prevent the deterioration of kidney disease, thus helping save the patient's life.

In this project, machine learning techniques are explored to improve accuracy in classification prediction. We tried various preprocessing methods and prediction techniques on WEKA. Several classification algorithms have been used to build a model that predicts whether a patient has chronic kidney disease or not. The prediction results and performance of each algorithm is compared.

Dataset Overview and Visualization

The data set "Chronic_Kidney_Disease" has 400 instances in total, which include 250 cases of ckd (chronic kidney disease) and 150 cases of notckd (not chronic kidney disease). Apparently, the class is binary. The data is quite balanced. The data set has 24 attributes, including 11 numerical ones and 13 nominal ones. There are missing values in this dataset. The information of the data is summarized in Table 1.



Data Set Characteristics:	Multivariate	**Number of Instances:**	400	**Area:**	N/A
Attribute Characteristics:	Real	**Number of Attributes:**	25	**Date Donated**	2015-07-03
Associated Tasks:	Classification	**Missing Values?**	Yes	**Number of Web Hits:**	73912

Table 1. Chronic_Kidney_Disease Data Set Summary

We visualized the data in two ways. Firstly, each attribute was plotted separately. The individual plot was plotted in Figure 1. In this plot, red color represents for the class “ckd”, while blue represents for “notckd”. From this plot, we can see that the data set is multivariate, containing both numerical and nominal features. Certain attributes separate the classes quite neatly, like hemoglobin and packed cell volume.

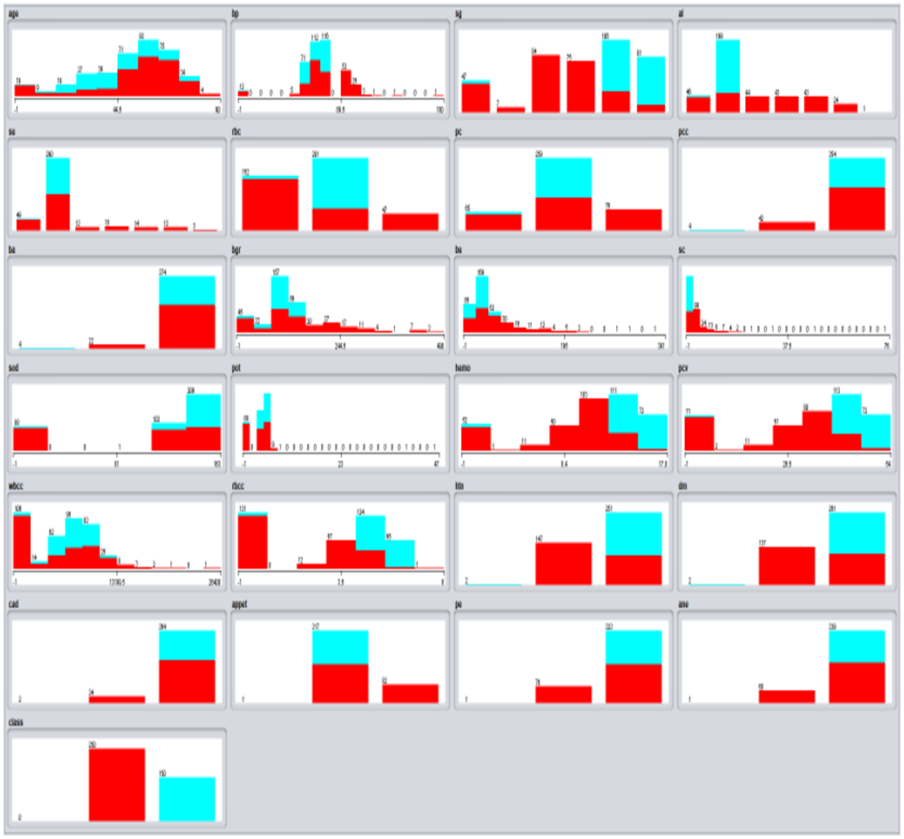


Figure 1. individual plots of 24 attributes and 1 class in the dataset

Next, we plotted the matrix plot of the attributes. Figure 2 shows a part of the matrix plot. From this figure, some attributes are highly corrected, like packed cell volume and hemoglobin, while others are not correlated, like appetite and blood urea. The information that we have obtained can be used as a guide for the following data preprocessing.

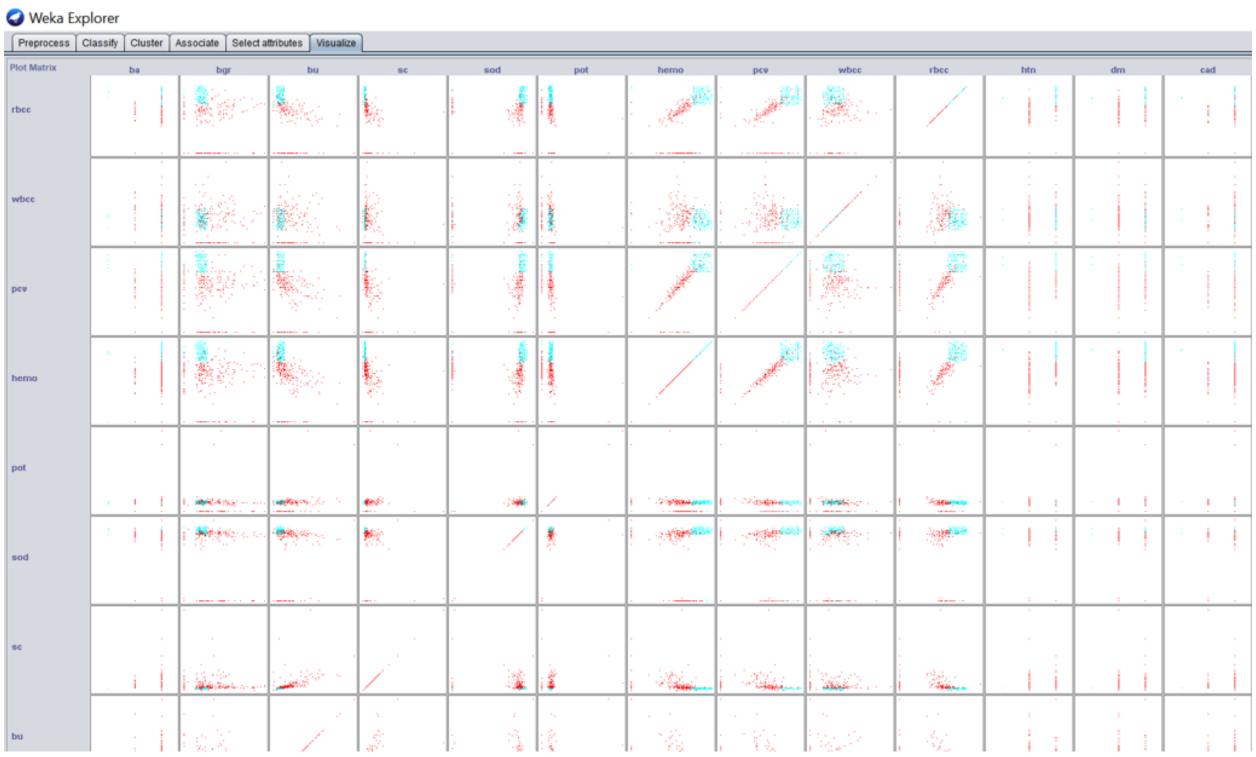
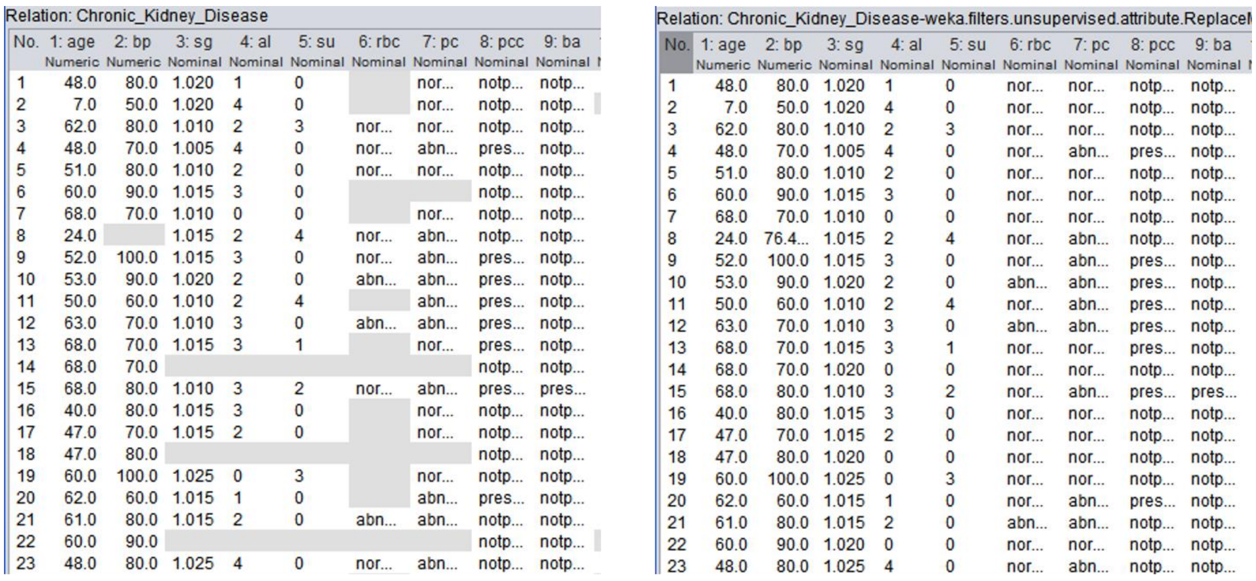


Figure 2. part of matrix plot of attributes

Data Preprocessing

Missing values

As mentioned in data description, there are some missing values in this data set. This can also be verified on weka explorer. Once an attribute is selected, the number of missing values in this attribute will be displayed. Instead of removing the missing values, I imputed the missing values with the mean of the attributes. This is carried out by using “ReplaceMissingValues” filter, which is under unsupervised.attribute.ReplaceMissingValues. Now the attribute values that were marked missing are set to the mean value of the distribution. Part of the data set before and after missing value replacement is shown in Figure 3.



Relation: Chronic_Kidney_Disease

No.	1: age	2: bp	3: sg	4: al	5: su	6: rbc	7: pc	8: pcc	9: ba
	Numeric	Numeric	Nominal	Nominal	Nominal	Nominal	Nominal	Nominal	Nominal
1	48.0	80.0	1.020	1	0		nor...	notp...	notp...
2	7.0	50.0	1.020	4	0		nor...	notp...	notp...
3	62.0	80.0	1.010	2	3	nor...	nor...	notp...	notp...
4	48.0	70.0	1.005	4	0	nor...	abn...	pres...	notp...
5	51.0	80.0	1.010	2	0	nor...	nor...	notp...	notp...
6	60.0	90.0	1.015	3	0			notp...	notp...
7	68.0	70.0	1.010	0	0		nor...	notp...	notp...
8	24.0		1.015	2	4	nor...	abn...	notp...	notp...
9	52.0	100.0	1.015	3	0	nor...	abn...	pres...	notp...
10	53.0	90.0	1.020	2	0	abn...	abn...	pres...	notp...
11	50.0	60.0	1.010	2	4		abn...	pres...	notp...
12	63.0	70.0	1.010	3	0	abn...	abn...	pres...	notp...
13	68.0	70.0	1.015	3	1		nor...	pres...	notp...
14	68.0	70.0						notp...	notp...
15	68.0	80.0	1.010	3	2	nor...	abn...	pres...	pres...
16	40.0	80.0	1.015	3	0		nor...	notp...	notp...
17	47.0	70.0	1.015	2	0		nor...	notp...	notp...
18	47.0	80.0						notp...	notp...
19	60.0	100.0	1.025	0	3		nor...	notp...	notp...
20	62.0	60.0	1.015	1	0		abn...	pres...	notp...
21	61.0	80.0	1.015	2	0	abn...	abn...	notp...	notp...
22	60.0	90.0						notp...	notp...
23	48.0	80.0	1.025	4	0	nor...	abn...	notp...	notp...

A

Relation: Chronic_Kidney_Disease-weka.filters.unsupervised.attribute.Replace

No.	1: age	2: bp	3: sg	4: al	5: su	6: rbc	7: pc	8: pcc	9: ba
	Numeric	Numeric	Nominal	Nominal	Nominal	Nominal	Nominal	Nominal	Nominal
1	48.0	80.0	1.020	1	0	nor...	nor...	notp...	notp...
2	7.0	50.0	1.020	4	0	nor...	nor...	notp...	notp...
3	62.0	80.0	1.010	2	3	nor...	nor...	notp...	notp...
4	48.0	70.0	1.005	4	0	nor...	abn...	pres...	notp...
5	51.0	80.0	1.010	2	0	nor...	nor...	notp...	notp...
6	60.0	90.0	1.015	3	0	nor...	nor...	notp...	notp...
7	68.0	70.0	1.010	0	0	nor...	nor...	notp...	notp...
8	24.0	76.4...	1.015	2	4	nor...	abn...	notp...	notp...
9	52.0	100.0	1.015	3	0	nor...	abn...	pres...	notp...
10	53.0	90.0	1.020	2	0	abn...	abn...	pres...	notp...
11	50.0	60.0	1.010	2	4	nor...	abn...	pres...	notp...
12	63.0	70.0	1.010	3	0	abn...	abn...	pres...	notp...
13	68.0	70.0	1.015	3	1	nor...	nor...	pres...	notp...
14	68.0	70.0	1.020	0	0	nor...	nor...	notp...	notp...
15	68.0	80.0	1.010	3	2	nor...	abn...	pres...	pres...
16	40.0	80.0	1.015	3	0	nor...	nor...	notp...	notp...
17	47.0	70.0	1.015	2	0	nor...	nor...	notp...	notp...
18	47.0	80.0	1.020	0	0	nor...	nor...	notp...	notp...
19	60.0	100.0	1.025	0	3	nor...	nor...	notp...	notp...
20	62.0	60.0	1.015	1	0	nor...	abn...	pres...	notp...
21	61.0	80.0	1.015	2	0	abn...	abn...	notp...	notp...
22	60.0	90.0	1.020	0	0	nor...	nor...	notp...	notp...
23	48.0	80.0	1.025	4	0	nor...	abn...	notp...	notp...

B

Figure 3. part of attribute values before (A) and (B) after missing values replaced

Feature selection

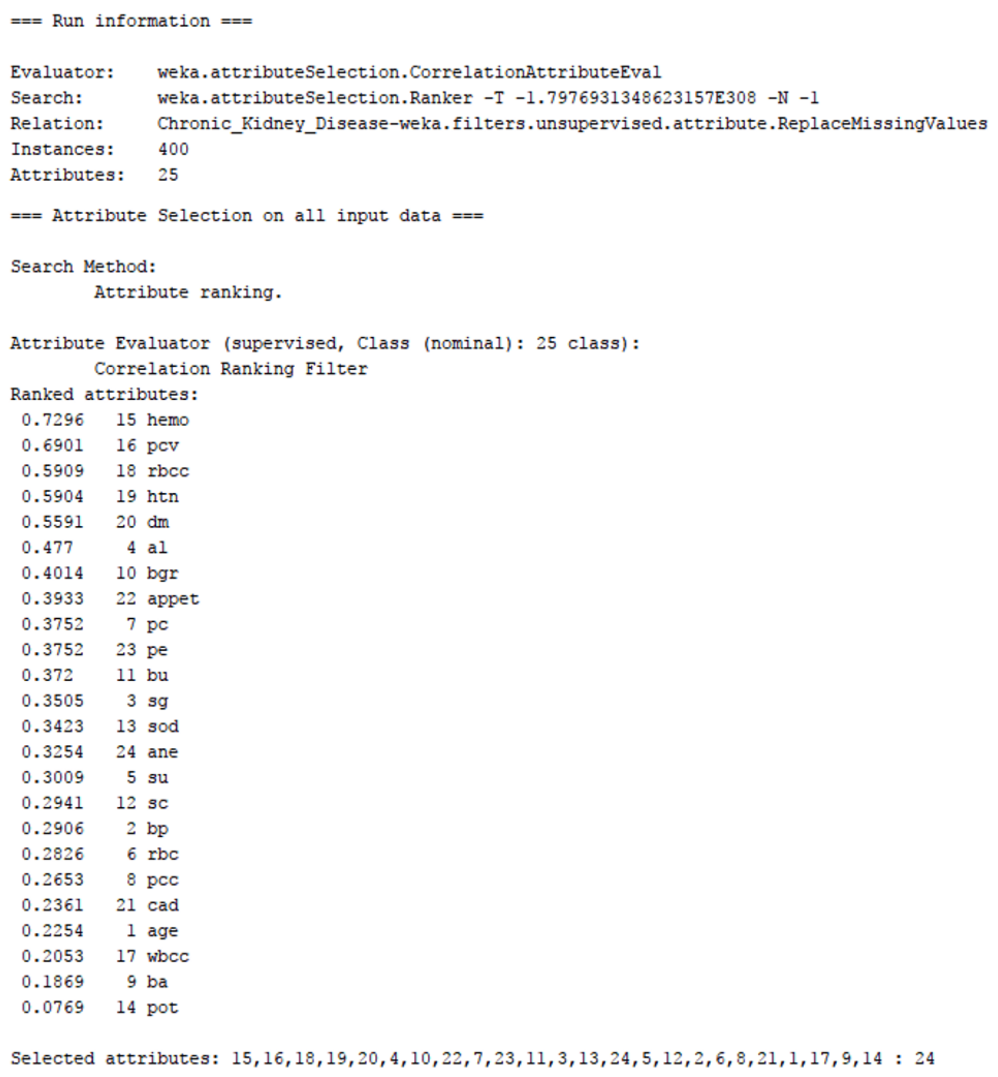
As we know, not all attributes are equal. Whether the data is provided to us, or the data is gathered by ourselves, it is critically important to do the attributes selection. It's an important step as it can totally determine whether we can successfully model the problem or not.

Before evaluation of algorithms, redundant and irrelevant attributes from the dataset should be deleted. Feature selection is a process in which we select for the optimal feature subset. Usually the notion of "optimal" means the highest accuracy. As we don't know which feature selection techniques on the dataset will produce the best models, I tried different feature selection methods. This allow me to have different views of the data.

Correlation Based Feature Selection

First, I used correlation to select the most relevant attributes. The correlation between each attribute and the output variable was calculated. Then, I selected those attributes that have a moderate-to-high correlation, while dropping those with a low correlation.

Weka supports this feature selection with "CorrelationAttributeEval" function which has to be combined with a Ranker search method. Running this on our dataset suggests that attribute "hemo" has the highest correlation (0.7296) with the output class. It is followed by the attributes "pcv", "rbcc", "htn", etc. The results of correlation based feature selection are shown on Figure 4. By using 0.3 as the cut-off for relevant attributes, we keep the attributes above 0.3, while the remaining attributes were removed.



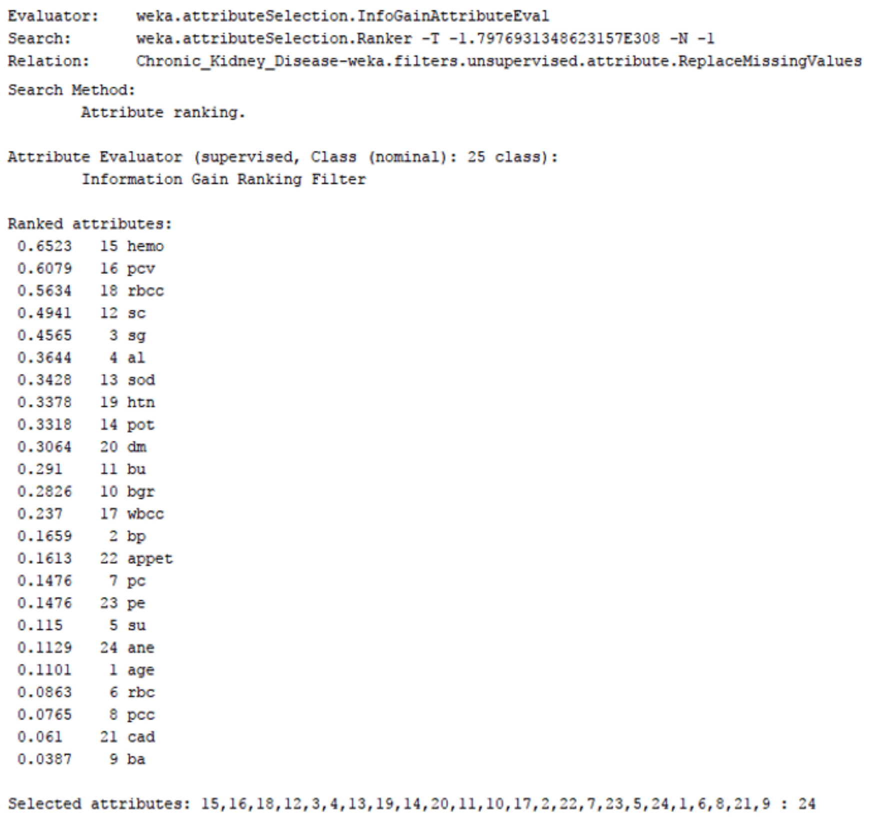
```
=== Run information ===

Evaluator:    weka.attributeSelection.CorrelationAttributeEval
Search:       weka.attributeSelection.Ranker -T -1.7976931348623157E308 -N -1
Relation:     Chronic_Kidney_Disease-weka.filters.unsupervised.attribute.ReplaceMissingValues
Instances:    400
Attributes:   25

=== Attribute Selection on all input data ===

Search Method:
        Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 25 class):
        Correlation Ranking Filter
Ranked attributes:
 0.7296   15 hemo
 0.6901   16 pcv
 0.5909   18 rbcc
 0.5904   19 htn
 0.5591   20 dm
 0.477     4 al
 0.4014   10 bgr
 0.3933   22 appet
 0.3752    7 pc
 0.3752   23 pe
 0.372    11 bu
 0.3505    3 sg
 0.3423   13 sod
 0.3254   24 ane
 0.3009    5 su
 0.2941   12 sc
 0.2906    2 bp
 0.2826    6 rbc
 0.2653    8 pcc
 0.2361   21 cad
 0.2254    1 age
 0.2053   17 wbcc
 0.1869    9 ba
 0.0769   14 pot

Selected attributes: 15,16,18,19,20,4,10,22,7,23,11,3,13,24,5,12,2,6,8,21,1,17,9,14 : 24
```

Figure 4. correlation based feature selection results on Weka

Information Gain Based Feature Selection

Another popular feature selection method is to calculate the information gain. Information gain is also known as the “entropy”. It can be calculated for each attribute for the output variable. Feature selection based on information gain is supported by “InfoGainAttributeEval” in Weka. Like the correlation method, it also must use Ranker as the search method.

Running it on our data, we can see that like in correlation technique, the attribute “hemo” still ranks highest. It contributes more information than all the other attributes (0.6523). It is followed by the attributes “pcv”, “rbcc”, “sc”, “sg”, etc. By using 0.2 as the cut-off for relevant attributes, we keep the attributes above 0.2, while removing the rest.

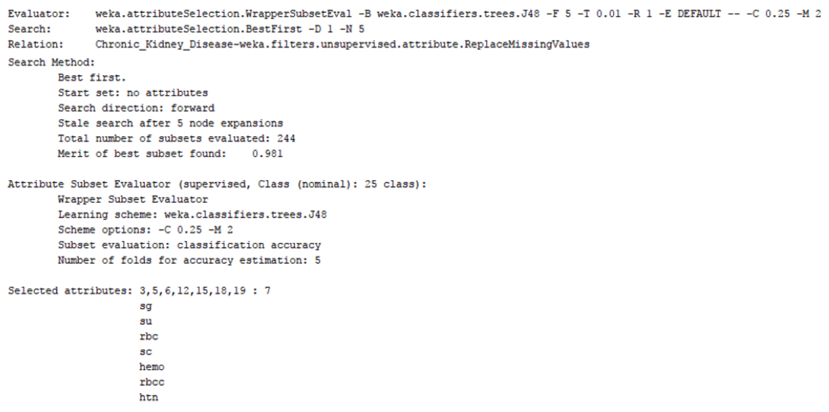


```
Evaluator:    weka.attributeSelection.InfoGainAttributeEval
Search:       weka.attributeSelection.Ranker -T -1.7976931348623157E308 -N -1
Relation:     Chronic_Kidney_Disease-weka.filters.unsupervised.attribute.ReplaceMissingValues
Search Method:
	Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 25 class):
	Information Gain Ranking Filter

Ranked attributes:
 0.6523   15 hemo
 0.6079   16 pcv
 0.5634   18 rbcc
 0.4941   12 sc
 0.4565    3 sg
 0.3644    4 al
 0.3428   13 sod
 0.3378   19 htn
 0.3318   14 pot
 0.3064   20 dm
 0.291    11 bu
 0.2826   10 bgr
 0.237    17 wbcc
 0.1659    2 bp
 0.1613   22 appet
 0.1476    7 pc
 0.1476   23 pe
 0.115     5 su
 0.1129   24 ane
 0.1101    1 age
 0.0863    6 rbc
 0.0765    8 pcc
 0.061    21 cad
 0.0387    9 ba

Selected attributes: 15,16,18,12,3,4,13,19,14,20,11,10,17,2,22,7,23,5,24,1,6,8,21,9 : 24
```

Figure 5. information gain based feature selection results on Weka

Learner Based Feature Selection

It is also a popular feature selection technique. The idea is to use a generic and powerful learning algorithm and evaluate the performance of the algorithm on the dataset with different subsets of attributes selected. Then we select the subset resulting in the best performance.
In Weka, learner based feature selection is supported by “WrapperSubsetEval” technique. I used J48 as the evaluation algorithm and BestFirst as the search method. Running it on the dataset selected 7 out of 24 input variables: sg, su, rbc, sc, hemo, rbcc, and htn.



```
Evaluator:    weka.attributeSelection.WrapperSubsetEval -B weka.classifiers.trees.J48 -F 5 -T 0.01 -R 1 -E DEFAULT -- -C 0.25 -M 2
Search:       weka.attributeSelection.BestFirst -D 1 -N 5
Relation:     Chronic_Kidney_Disease-weka.filters.unsupervised.attribute.ReplaceMissingValues
Search Method:
	Best first.
	Start set: no attributes
	Search direction: forward
	Stale search after 5 node expansions
	Total number of subsets evaluated: 244
	Merit of best subset found:    0.981

Attribute Subset Evaluator (supervised, Class (nominal): 25 class):
	Wrapper Subset Evaluator
	Learning scheme: weka.classifiers.trees.J48
	Scheme options: -C 0.25 -M 2
	Subset evaluation: classification accuracy
	Number of folds for accuracy estimation: 5

Selected attributes: 3,5,6,12,15,18,19 : 7
                     sg
                     su
                     rbc
                     sc
                     hemo
                     rbcc
                     htn
```

Figure 6. learner based feature selection results on Weka

Algorithm Selection

The classification algorithms used in this project include: Logistic Regression (LR), Naïve Bayes, K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Decision Trees. All of these algorithms are implemented on Weka. And 10-fold cross validation is used.

Naïve Bayes

Naïve Bayes is a conditional probability model. A problem instance to be classified is represented by a vector representing features [1]. Naïve Bayes support both nominal variables and numerical variables. The assumption of Naïve Bayes is that the prior probability for each attribute is independent of each other. Although this is an unrealistic assumption, as the variables are expected to interact and dependent, Naïve Bayes has been proved to be an effective classification algorithm. This assumption makes the algorithm calculate fast. In a binary classification, Naïve Bayes calculates the posterior probability for each class and makes a prediction for the class with the higher probability.

Logistic Regression (LR)

Logistic regression measures the relationship between categorical dependent variable with one or more independent variables by estimating probabilities using cumulative logistic function [2]. It is a binary classification algorithm. It assumes the attribute variables are numeric and have a Gaussian distribution. It can still achieve good prediction if the data is not Gaussian. This algorithm learns a coefficient for each input, which will linearly be combined into a regression function an then transformed using a logistic function. It is a simple and fast technique. It only supports binary classification problems.

Support Vector Machine (SVM)

SVM performs classification by constructing a hyper plane to optimally separate data into two categories [3]. It works by finding a line which best separates the data into two groups. This is achieved by using an optimization process which only considers the instances closest to the line that best separate the classes. The instances are called support vectors, hence the name of the algorithm.

In some problems, a straight line cannot separate the classes neatly. In this situation, a margin is usually added around the line to relax the restraints. This allows some cases to be misclassified by achieving a better result overall.

K-Nearest Neighbors (KNN)

KNN algorithm is a non-parametric method used for classification and regression [4]. It works by storing the training dataset and querying the dataset to locate the k closest training cases when trying to make a prediction. It is also known as a lazy method, as it doesn't have the learning process. KNN is a simple algorithm, and there is almost no assumption about the problem except the distance between the instances. It often achieves good performance when making predictions.

Decision Tree

Decision Tree is a decision support tool. It uses a tree-like model to make decisions and possible consequences, including resource costs, chance event outcomes, and utility [5]. It starts at the root of the tree, and then move down to the leaves until a prediction is made. It greedily selects the best split point to make predictions. This process repeats until the tree has a fixed depth. As the tree grows, it is pruned so as to improve the model's generality.

Analysis Results and Comparison

After trying various preprocessing method, now we have five type of data sets: original data, missing values replaced with feature mean, data after correlation based feature selection, data after information gain based feature selection, data after learner based feature selection. For convenience, these five data sets will be addressed as 0, 1, 2, 3, 4 in the following paragraphs. And we have five baseline machine learning techniques to try. Therefore, we will have a 5x5 combinations. To evaluate the performance of each machine learning model, Accuracy, Precision, Recall, F-measure, and ROC Area are used as evaluation. As F-measure takes account of both precision and accuracy, we use F-measure as the main scoring functions. The prediction results are summarized in Table 2. In “preprocess” column, 0 to 4 represent for the five types before and after different preprocessing methods.

Algorithm	Preprocess	Accuracy	Precision	Recall	F-measure	ROC Area
Naïve Bayes	0	95%	0.956	0.950	0.951	1
Naïve Bayes	1	94.5%	0.951	0.945	0.946	0.998
Naïve Bayes	2	96.25%	0.966	0.963	0.963	1.000
Naïve Bayes	3	93.75%	0.944	0.938	0.938	0.997
Naïve Bayes	4	94.25%	0.945	0.943	0.943	0.993
Logistic Regression	0	97.5%	0.976	0.975	0.975	0.994
Logistic Regression	1	97.25%	0.978	0.978	0.978	0.994
Logistic Regression	2	96.25%	0.964	0.963	0.963	0.988
Logistic Regression	3	97%	0.970	0.970	0.970	0.994
Logistic Regression	4	97.5%	0.976	0.975	0.975	0.994
SVM	0	97.75%	0.979	0.978	0.978	0.982
SVM	1	97.75%	0.979	0.978	0.978	0.982
SVM	2	98.25%	0.983	0.983	0.983	0.986
SVM	3	97.75%	0.979	0.978	0.978	0.982
SVM	4	95%	0.956	0.950	0.951	0.960
KNN	0	95.75%	0.962	0.958	0.958	0.966

KNN	1	98.5%	0.985	0.985	0.985	0.987
KNN	2	99%	0.990	0.990	0.990	0.992
KNN	3	98%	0.980	0.980	0.980	0.979
KNN	4	98.75%	0.988	0.988	0.988	0.984
Decision Tree	0	99%	0.990	0.990	0.990	0.999
Decision Tree	1	96.75%	0.967	0.968	0.967	0.976
Decision Tree	2	96.5%	0.965	0.965	0.965	0.963
Decision Tree	3	95.75%	0.958	0.958	0.958	0.969
Decision Tree	4	98%	0.981	0.980	0.980	0.972

Table 2. Chronic_Kidney_Disease Data Set Summary

To make the comparison of different prediction results clearer, I plotted F-measure of each prediction versus data sets (Figure 7). As in Table 2, “0” represents for original data, “1” represents for data set in which missing values replaced with feature mean, “2” represents for data set 1 after correlation based feature selection, “3” represents for data set 1 after information gain based feature selection, and “4” represents for data set 1 after learner based feature selection. Plots of different colors represent for different classification algorithms. The five algorithms all achieve fairly good results, with all of accuracy and most F-measure values larger than 0,95. From this plot, we can conclude that KNN perform best overall, while Naïve Bayes perform worst.

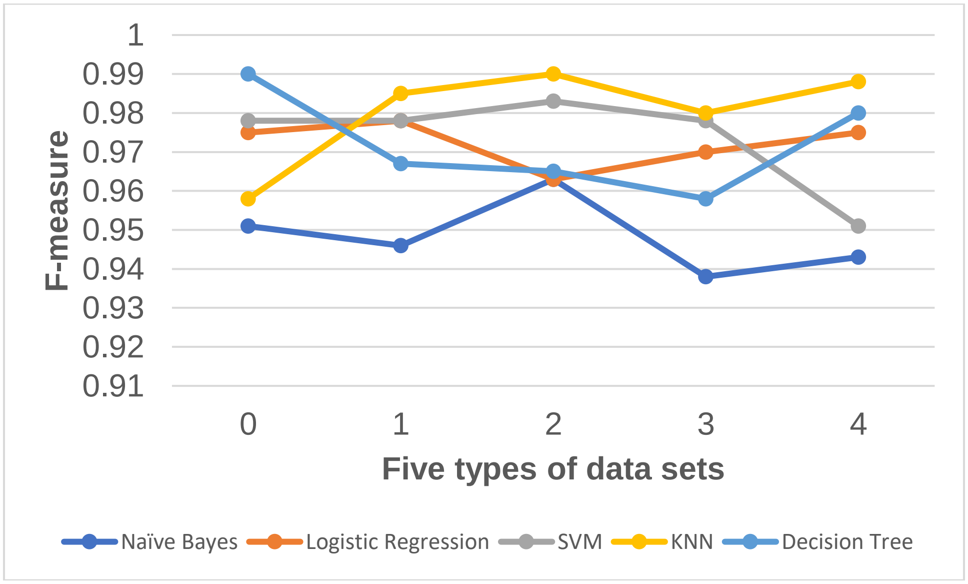


Figure 7. Comparison of different classification methods on five data sets

If we take a look at the five plots individually, we can see: For Decision Tree, F-measurement is largest on dataset 0, which means that the original data without any preprocessing results in the best prediction; For logistic regression, F-measure has the largest value on dataset 1, suggesting that correlation based feature selection helps this algorithm most; For KNN, Naïve Bayes and SVM, the best performance is on dataset 2, indicating information gain based feature selection is more suitable to these algorithms. Another finding is KNN benefits from preprocessing most, with F-measure increasing from 0.958 to 0.990 after preprocessing. If both ROC Area and F-measure are taken into consideration, Decision Tree is the most accurate classifier in this project, giving ROC Area of 0.999 and F-measure of 0.990. The decision tree constructed is shown in Figure 8.

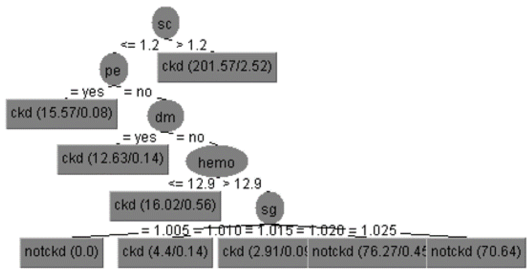


Figure 8. Visualization of decision tree producing F-measure of 0.99

Conclusion

In this project, various preprocessing methods and classification techniques are explored. Decision Tree and KNN algorithms outperformed the other three baseline classification techniques and give best forecasting accuracy. From the prediction results, we can also conclude that preprocessing is very useful in machine learning classification. However, whether the data needs preprocessing, and if it needs, how to preprocess can be a difficult decision. From the attempts in this project, various machine learning techniques should be performed on datasets after different preprocessing methods. Different classification algorithms can have different preferences on attribute variables type and preprocessing methods. In addition, the data set of this project is small and relatively simple, all the classification techniques used in this project have potential to achieve good prediction performance.

References

[1] Murphy, Kevin P, “Naïve Bayes Classifiers,” University of British Columbia

[2] Kurt, Imran, MevlutTure, and A. TurhanKurum, “Comparing performances of logistic regression, classification and regression tree, and neural networks for predicting coronary.”

[3] Alfaro, Rene, et al. “Forests for the New Millennium-Making Forest Work for People and Nature,” Selected Books 1 (2005).

[4] Alman, N.S. “An introduction to kernel and nearest-neighbor nonparametric regression,” The American Statistician. 46. 3: pp. 175-185, 1992

[5] Utgoff, P.E. “Increment induction of decision trees”, Machine learning, 4(2), 161-186, 1989.