Paper CC01-2011

USING SAS TO FIND THE BEST K FOR *K*-NEAREST-NEIGHBOR CLASSIFICATION

Charlie Huang, Oklahoma State University, Stillwater, OK

ABSTRACT

Searching data points" nearest neighbors can serve either supervised learning (K-means clustering) or unsupervised learning (k-Nearest-Neighbor classification). In SAS, a few clustering procedures apply K-means to find centroids and group observations into clusters. k-Nearest-Neighbor (k-NN) rule is a model-free data mining method that determines the categories based on majority vote. A discriminant analysis procedure of SAS, PROC DISCRIM, enables the k-NN classifiers for multivariate data. For some data difficult to be classified, k-NN may outperform least-square based logistic regression. Optimization of the parameter k, which is the number of nearest neighbors, could help minimize the error rate.

INTRODUCTION

The Euclidean or Mahalanobis distances between the data pairs can be measured for multiple purposes. K-means clustering minimizes the sum of squares for the distances between data and finds the corresponding cluster centroids, while k-NN rule assigns the unclassified sample to the class represented by a majority of its k number nearest neighbors in the training set. The k-nearest neighbor (k-NN) concept was first raised by Fix and Hodges [1] and statistically proved by Cover and Hart [2]. Recently Liang Xie discovered that some procedures in SAS/STAT such as PROC DISCRIM and PROC LOESS can be tweaked for k-NN classification [3]. One of the underlying algorithms for the procedure PROC DISCRIM for is the k-NN rule, which makes it an ideal vehicle to realize k-NN based training and scoring.

Two sample datasets shipped with SAS 9.2 or 9.3, SASHELP.IRIS and SASHELP.CARS, were used: the first dataset contains the IRIS data introduced by Sir Ronald Aylmer Fisher as early as 1936; the second describes a number of cars with their model, manufacture country, weight, length, etc. The summary statistics of the variable for this paper were listed in Table 1. Since the IRIS data is a traditional dataset for data mining exercises, we treated it as an „easy" data. SASHELP.CARS is not related to data mining training and its variables are not in typical cause-and-effect relationship, and consequently we regarded it as a "difficult" data.

Table 1. Variables used in the datasets SASHELP.IRIS and SASHELP.CARS

Name	Objective	Variables				
		Species	SepalLength	SepalWidth	PetalLength	PetalWidth
SASHELP.IRIS	To predict Iris's species	Setosa :50	Min. :43.00	Min. :20.00	Min. :10.00	Min. : 1.00
	by the length of sepal,	Versicolor:50	1st Qu.:51.00	1st Qu.:28.00	1st Qu.:16.00	1st Qu.: 3.00
	the width of sepal,	Virginica :50	Median :58.00	Median :30.00	Median :43.50	Median :13.00
	the length of petal and	NA	Mean :58.43	Mean :30.57	Mean :37.58	Mean :11.99
	the width of petal	NA	3rd Qu.:64.00	3rd Qu.:33.00	3rd Qu.:51.00	3rd Qu.:18.00
		NA	Max. :79.00	Max. :44.00	Max. :69.00	Max. :25.00

Name	Objective	Variables			
		Origin	Invoice	Wheelbase	Length
SASHELP.CARS	To predict the manufacture	Asia :158	Min. : 9875	Min. : 89.0	Min. :143.0
	country by price showed on	Europe:123	1st Qu.: 18866	1st Qu.:103.0	1st Qu.:178.0
	invoice, the distance between	USA :147	Median : 25295	Median :107.0	Median :187.0
	the centers of the front	NA	Mean : 30015	Mean :108.2	Mean :186.4
	and rear wheels, and	NA	3rd Qu.: 35710	3rd Qu.:112.0	3rd Qu.:194.0
	the body length	NA	Max. :173560	Max. :144.0	Max. :238.0

K-MEANS CLUSTERING

K-means clustering categorizes ungrouped data into K number of clusters, and this approach has been extensively used by SAS‟s clustering procedures. Method 1 and 2 of all the 6 methods in the MODECLUS procedure are based on K-means [4]. In the PROC statements of this procedure, the option k specifies the number that equals the data point plus the nearest neighbors around it, instead of the number of clusters. PROC MODECLUS decides how many clusters will be formed. For example, if k = 4 for the IRIS data, then 3 closest neighbors for each of the 150 observations were fetched. If some neighbors present identical distances, then they will be counted together. As the result, 3 to 6 neighbors appeared for individual observations of IRIS data, and the aggregated distances among the observations didn’t show significant difference (Figure 1).

```
proc modeclus data = sashelp.iris m = 1 k = 4 out = _test1 neighbor;
var petallength petalwidth sepallength sepalwidth;
ods output neighbor = _test2;
run;
```

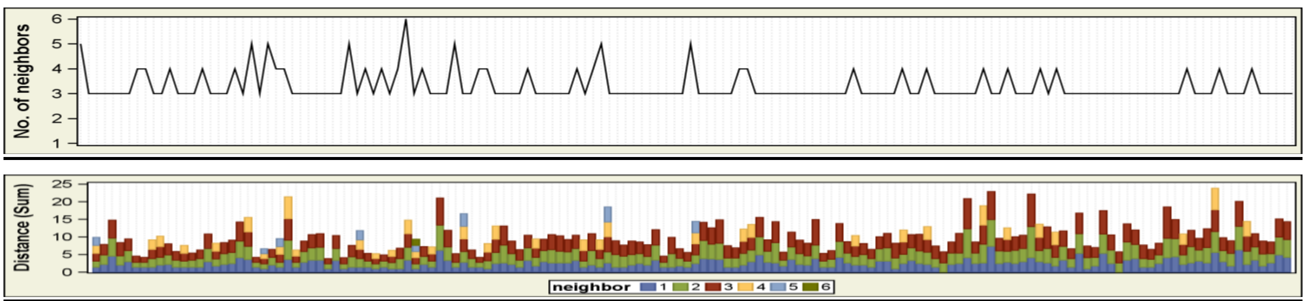

Figure 1. Distances between nearest neighbors by PROC MODELCLUS when K is 4 for IRIS data

In this experiment, the MODECLUS procedure created 14 disjoint clusters by nonparametric density estimation. The scatter plot with the designated cluster numbers as labels showed the affinity of clusters toward certain species levels, which suggests that the space distances may be utilized as a tool to predict classes for unknown data.

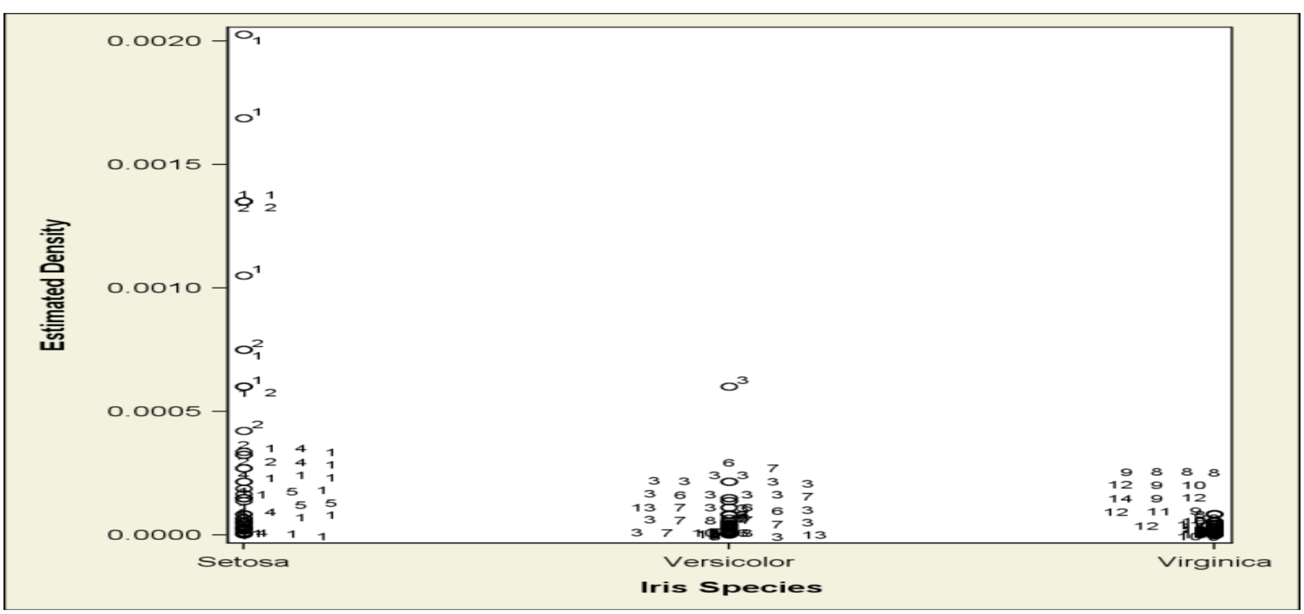

Figure 2. Scatter plot by iris species and estimated densities

K-NEAREST-NEIGHBOR

PROC DISCRIM was used to apply *k*-NN rule. At the beginning, a macro *%partition*() was implemented to randomly separate the raw datasets into two parts and keep their target variable‟s proportions as the raw data. In this paper, two equal-size training and validation sets were created in SAS‟s WORK directory for either IRIS data (75 observations for both IRIS_TRAIN and IRIS_VALIDATE) or CARS data (214 observations for both CARS_TRAIN and CARS_VALIDATE), separately. A user-defined function *knn*() was created through wrapping a complied macro by PROC FCMP. Data visualization was conducted by the statistical graphics procedures. Figures and tables in this paper can be duplicated by running the SAS codes from the appendix in SAS 9.3.

As for the DISCRIM procedure, once METHOD is specified as NPAR and numbers are assigned to either K or R options in the PROC statement, the *k*-NN rule will be activated for the discriminant analysis. While k is set as 5, *k*-NN would easily achieve a decent misclassification rate 1.33% for the IRIS validation set(Figure 3a). On the contrary, for the CARS validation set, although the majority at each level has been classified correctly, the overall misclassification rate 32.24% is not very satisfying (Figure 3b).

```
proc discrim data = cars_train test = cars_validate testout = _score1 method = npar k = 5 testlist;
class species;
var petallength petalwidth sepallength sepalwidth;
run;
proc discrim data = iris_train test = iris_validate testout = _score2 method = npar k = 5 testlist;
class origin;
var invoice wheelbase length;
run;
```

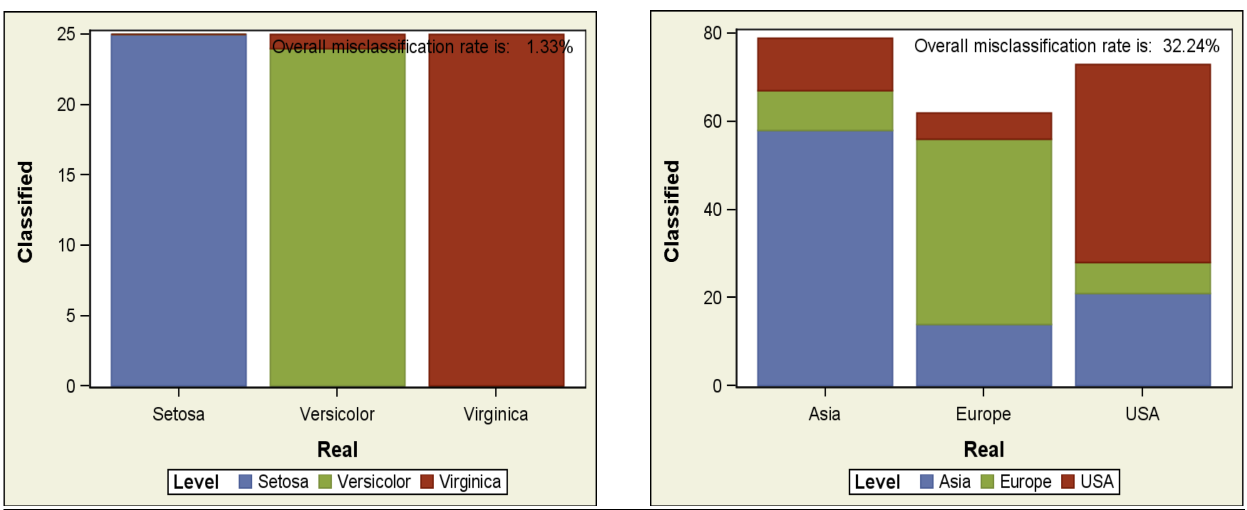

Figure 3. The bar plots showing the classification results for the validation sets of IRIS and CARS data when k is 5

To further compare the real classes and the classified classes for the CARS validation set after classification, we broke down the invoice prices into 4 quartiles to build a 4X2 plot panel. On the scatter plots, we observed that *k*-NN misjudged some European cars as Asian cars in the invoice‟s 2nd quartile, and also mistakenly treated a few Asian cars as US cars in the invoice‟s 3rd quartile (Figure 4).

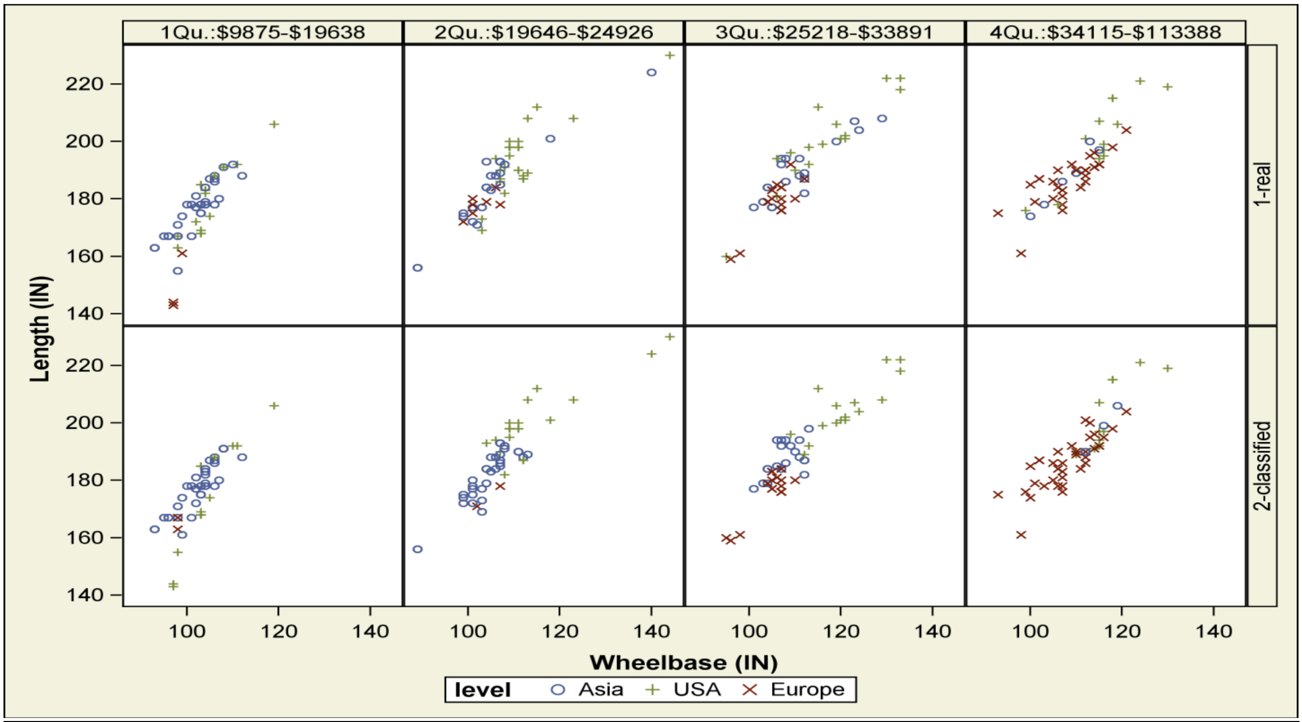

Figure 4. Scatter plots between the real classes and the classified classes in the validation set of the CARS data

Two fundamental prediction approaches are often mentioned for supervised learning: linear model by least-square and *k*-Nearest-Neighbor rule. There is a trade-off for their strength and weakness, because least-square brings low variance but high bias and nearest-neighbor leads into low bias but high variance [5]. Discriminant analysis based on *k*-NN assumes prior knowledge of the classes. The LOGISTIC procedure in SAS/STAT can fit data by generalized logit model if with GLOGIT option. In this experiment, the comparison between the classification results indicated that PROC DISCRIM (k = 5) is slightly better than PROC LOGISTIC for the CARS data (Table 2).

```
proc logistic data = cars_train;
model origin = invoice wheelbase length / link = glogit;
score data = cars_train out = logitscore;
run;
```

Table 2. The classified results comparison between PROC DISCRIM and PROC LOGISTIC

Number of observations and percent classified into the target variable Origin								
	By PROC DISCRIM (k=5)				By PROC LOGISTIC			
From Origin	Asia	Europe	USA	Total	Asia	Europe	USA	Total
Asia	58	9	12	79	54	8	17	79
	73.42%	11.39%	15.19%	100%	68.35%	10.13%	21.52%	100%
Europe	14	42	6	62	18	39	5	62
	22.58%	67.74%	9.68%	100%	29.03%	62.90%	8.06%	100%
USA	21	7	45	73	27	6	40	73
	28.77%	9.59%	61.64%	100%	36.99%	8.22%	54.79%	100%
Total	93	58	63	214	99	53	62	214
	43.46%	27.1%	29.44%	100%	45.45%	26.42%	35.48%	100%
Priors	0.33333	0.33333	0.33333		NA	NA	NA	

FIND THE BEST K

The value of k for *k*-NN is a tuning parameter: increasing k decreases noise but causes less distinctive boundaries. If with very large or infinite N, a larger k usually provides better performance. However, for smaller values of N such as real-world data size, a larger k is not always the best choice. A good k may depend upon cross-validation or other techniques [6].

In this experiment, k for *k*-NN rule for the IRIS validation set ranges from 1 to 20. Except the first two (k=1 or 2), most misclassification rates associated with the varying k values stay at pretty low level. Multiple k values constitute the best k candidates (Figure 5).

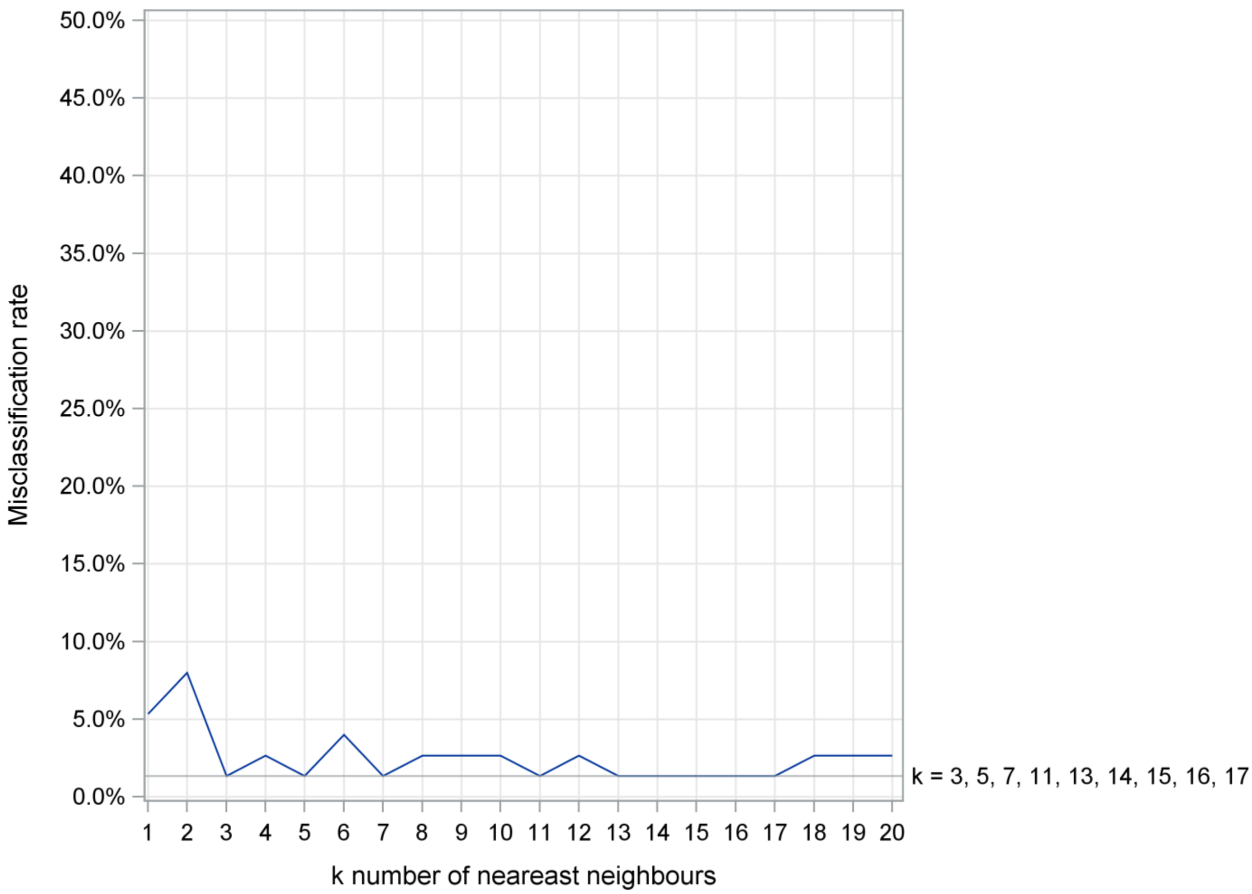

Figure 5. Misclassification rates for the IRIS validation set (k is from 1 to 20)

Classification of the IRIS validation set starts with 1-Nearest-Neighbor and ends with 40-Nearest-Neighbor. Almost all misclassification rates with various k values are well below 37.9% that was resulted by the logistic regression by PROC LOGISTIC (Figure 6). Here 4 is the best value for k. Overall, for this „difficult" CARS data, *k*-NN seems a better choice than the logistic regression.

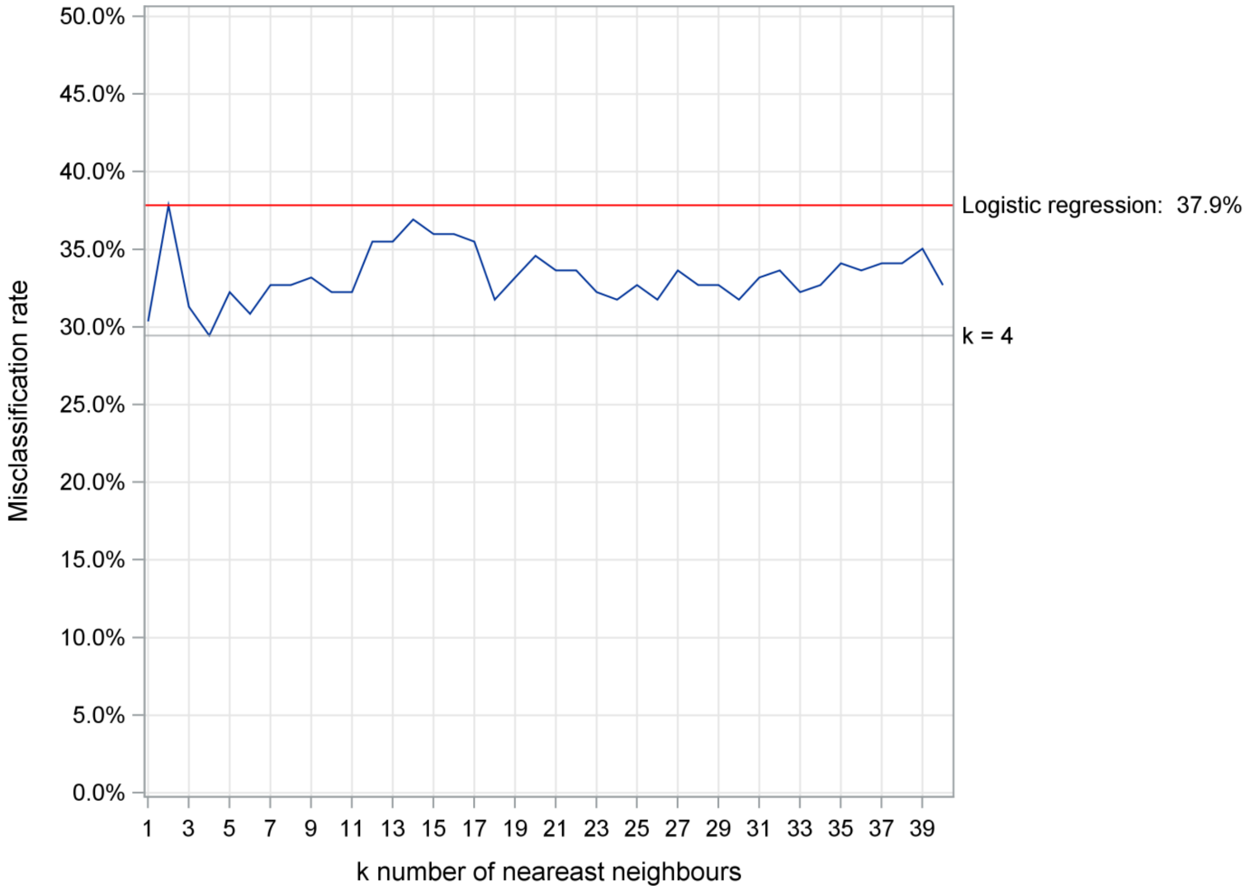

Figure 5. Misclassification rates for the CARS validation set (k is from 1 to 40)

CONCLUSION

In this paper, two sample datasets included in SAS were used to demonstrate the implementation of *k*-Nearest-Neighbor rule. For the „easy" IRIS data, *k*-NN showed excellent prediction power. For the „difficult" CARS data, tuning the parameter k would decrease classification error rate. The nonparametric *k*-NN doesn"t need any stringent assumption, such as normality, which makes it especially useful for complicated or fuzzy data. The result also proves that *k*-NN may be a good alternative to suit those datasets difficult for least-square based generalized linear model to classify.

In conclusion, SAS/STAT provides effective solutions to apply *k*-NN for predicative modeling. The statistical graphics procedures, PROC FCMP, macro facility and other parts of SAS together support SAS to be a productive data mining platform.

REFERENCES

1. E. Fix and J. L. Hodges, "Discriminatory analysis, nonparametric discrimination: Consistency properties". Technical Report 4, USAF School of Aviation Medicine, Randolph Field, TX, 1951.

2. T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification". IEEE Trans. Inform. Theory, vol. IT-13, no. 1, pp. 21–27, 1967.

3. L. Xie, “How to implement K-Nearest-Neighbor in SAS”. http://www.sascommunity.org

4. SAS/STAT 9.3 User‟s Guide, “The MODECLUS Procedure”. pp. 4934-4952. SAS Publishing, 2011

5. T. Hastie, R. Tibshirani, J. H. Friedma, “The elements of statistical learning: data mining, inference, and prediction”. ver. 5th. pp. 459-480, Springer, 2011.

6. S. Theodoridis, K. Koutroumbas, “Pattern recognition”. ver. 5th. pp. 61-63, Elsevier, 2009

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Charlie Huang

Institution research and information management

Oklahoma State University

221 PIO Building

Stillwater, OK. 74075

Phone (405)-7446703

Email: chah@okstate.edu

APPENDIX

Sample Code

```
******(1) USE K-MEANS CLUSTERING TO FIND NEAREAST NEIGHBORS****************;
proc modeclus data = sashelp.iris m = 1 k = 4 out = _test1 neighbor;
   var petallength petalwidth sepallength sepalwidth;
   ods output neighbor = _test2;
run;

ods html style = harvest image_dpi = 400;
proc sgplot data=_test1;
   scatter y = density x = species / datalabel = cluster;
run;

data _test3;
   set _test2; retain _tmpid;
   if missing(id) = 0 then _tmpid = id; else id = _tmpid;
run;

data _test4 _test5;
   set _test3; by id notsorted;
   if first.id then neighbor = 0;
   neighbor + 1; output _test4;
   if last.id then output _test5;
run;

ods graphics / width = 6in height = 1in ;
proc sgplot data = _test4;
   vbar id / response = distance group = neighbor;
   xaxis display = none grid;
run;

proc sgplot data = _test5;
   series x = id y = neighbor;
   xaxis display = none grid;
   yaxis values = (1 to 6) label = 'No. of neighbors';
run;

******(2) PARTITION RAW DATASET INTO TRAINING AND VALIDATION DATASETS******;
%macro partition(data = , target = , smpratio = ,
       seed = , train = , validate = );

   /************************************************************
```

```
* MACRO: partition()
* GOAL: divide to training and validation sets that
* represent original target variable's proportion
* PARAMETERS: data = raw dataset
* target = target variable
* smprate = ratio between training and validation
* set * seed = random seed for sampling
**************************************************************/

ods select none;
ods output variables = _varlist;
proc contents data = &data;
run;

proc sql;
   select variable into: num_var separated by ' '
   from _varlist
   where lowcase(type) = 'num';
quit;

proc sort data = &data out = _tmp1;
   by &target;
run;

proc surveyselect data = _tmp1 samprate = &smpratio
   out = _tmp2 seed = &seed outall;
   strata &target / alloc = prop;
run;

data &train &validate;
   set _tmp2; keep &num_var &target;
   if selected = 0 then output &train;
   else output &validate;
run;

proc datasets nolist;
   delete _:;
quit;
ods select all;
%mend;

%partition(data = sashelp.iris, target = species, smpratio = 0.5,
       seed = 20110901, train = iris_train, validate = iris_validate);
%partition(data = sashelp.cars, target = origin, smpratio = 0.5,
```

```
       seed = 20110901, train = cars_train, validate = cars_validate);

********(3) BUILD A USER-DEFINED FUNCTION TO IMPLEMENT K-NN************************;
option mstored sasmstore = sasuser;
%macro knn_macro / store source;
   %let target = %sysfunc(dequote(&target));
   %let input = %sysfunc(dequote(&input));
   %let train = %sysfunc(dequote(&train));
   %let validate = %sysfunc(dequote(&validate));
   %let error = 0;
   %if %length(&k) = 0 %then %do;
      %put ERROR: Value for K is missing ;
      %let error = 1;
   %end;
   %else %if %eval(&k) le 0 or %sysfunc(anydigit(&k)) = 0 %then %do;
      %put ERROR: Value for K is invalid ;
      %let error = 1;
   %end;
   %if %length(&target) = 0 %then %do;
      %put ERROR: Value for target is missing ;
      %let error = 1;
   %end;
   %if %length(&input) = 0 %then %do;
      %put ERROR: Value for INPUT is missing ;
      %let error = 1;
   %end;
   %if %sysfunc(exist(&train)) = 0 %then %do;
      %put ERROR: Training dataset does not exist ;
      %let error = 1;
   %end;
   %if %sysfunc(exist(&validate)) = 0 %then %do;
      %put ERROR: validation dataset does not exist ;
      %let error = 1;
   %end;
   %if &error = 1 %then %goto finish;

   ods output classifiedtestclass = _classifiedtestclass;
   proc discrim data = &train test = &validate testout = _scored
      method = npar k = &k testlist ;
      class &target;
      var &input;
   run;
   data _null_;
      set _scored nobs = nobs end = eof;
```

```
      retain count;
      if &target ne _into_ then count + 1;
      if eof then do;
         misc = count / nobs;
         call symput('misc', misc);
      end;
   run;
   %finish:;
%mend;

proc fcmp outlib = sasuser.knn.funcs;
   /*********************************************************
   * FUNCTION: knn() * GOAL: apply k-Nearest-Neighbor for classification
   * INPUT: k = number of nearest neighbours
   * train = training dataset
   * validate = validation dataset
   * target = target variable
   * input = input variables
   * OUTPUT: overall misclassification rate
   *********************************************************/
   function knn(k, train $, validate $, target $, input $);
      rc = run_macro('knn_macro', k, train, validate, target, input, misc);
      if rc eq 0 then return(misc);
      else return(.);
   endsub;
run;

******(3) APPLY K-NN FUNCTION TO CLASSIFY IRIS AND CARS DATA****************;
%macro errorchk(train = , validate = , target = , input = , k = );
/*********************************************************
   * MACRO: errorchk() * GOAL: use knn()function and visualize result
   * PARAMETERS: train = training dataset
   * validate = validation dataset
   * target = target variable
   * input = input variables
   * k = number of nearest neighbors
   *********************************************************/
   option cmplib = (sasuser.knn) mstored sasmstore = sasuser;
   data _null_;
      misc_rate = knn(&k, symget('train'), symget('validate'),
           symget('target'), symget('input'));
      call symput('misc_rate', misc_rate);
   run;
   proc sql noprint;
```

```
      select distinct &target into : varlist1 separated by ' '
      from &validate;
      select distinct cats("'", lowcase(&target), "'")
         into: varlist2 separated by ','
      from &validate;
   quit;
   proc transpose data = _classifiedtestclass out = _out1;
      by from&target notsorted;
      var &varlist1;
   run;
   data _out2;
      set _out1;
      where lowcase(from&target) in (&varlist2);
      label _name_ = 'Level';
   run;
   proc sgplot data = _out2;
      vbar from&target / response = col1 group = _name_;
      xaxis label = 'Real';
      yaxis label = 'Classified ';
      inset "Overall misclassification rate is:
           %sysfunc(putn(&misc_rate, percent8.2))" / position = topright;
   run;
%mend;

ods graphics / width= 400px height = 300px ;
%errorchk(train = iris_train, validate = iris_validate, target = species,
       input = petallength petalwidth sepallength sepalwidth, k = 5);
%errorchk(train = cars_train, validate = cars_validate, target = origin,
       input = invoice wheelbase length, k = 5);

******(4) VISUALIZE CLASSIFICATION RESULT FOR CARS DATA********************;
proc rank data = _scored groups = 4 out = _out3;
   var invoice;
   ranks q;
run;
proc sort data = _out3 out = _out3;
   by q invoice;
run;
data _out4;
   set _out3;
   by q ;
   retain fmtname "qvar" start end;
   if first.q then start = invoice;
   if last.q then end = invoice;
```

```
   if last.q; length label $35;
   q + 1;
   label = cat(q,'Qu.:', '$',start,'-','$',end);
run;
proc format cntlin = _out4 fmtlib;
run;
data _out5(keep=level name invoice wheelbase length q qfmt);
   set _out3;
   qfmt = put(invoice, qvar.);
   level = _into_;
   name = '2-classified';
   output; level = origin;
   name = '1-real';
   output;
run;

ods graphics / width = 700px height = 500px ;
proc sgpanel data = _out5;
   panelby qfmt name / layout = lattice onepanel novarname;
   scatter x = Wheelbase y = length / group = level ;
run;

******(5) USE LOGISTIC REGREESION TO CLASSIFY CARS DATA*******************;
proc logistic data = cars_train;
   model origin = invoice wheelbase length / link = glogit;
   score data = cars_validate out = logitscored;
run;
proc freq data = logitscored;
   table f_origin*i_origin / nocol nocum nopercent;
run;

******(6) RUN LOOPS ON K-NN TO FIND THE BEST K VALUE**********************;
%macro findk(train = , validate =, target = , input =, maxk =);
   /*****************************************************
   * MACRO: findk()
   * GOAL: visualize results of k-NNs by loops
   * PARAMETERS: train = training dataset
   * validate = validation dataset
   * target = target variable
   * input = input variables
   * maxk = maximum value of k
   *****************************************************/
   option cmplib = (sasuser.knn) mstored sasmstore = sasuser;
   ods select none;
```

```
   data _tmp3;
      do k = 1 to &maxk ;
         misc_rate = knn(k, symget('train'), symget('validate'),
            symget('target'), symget('input'));
         output;
      end;
   run;
   proc sql;
      select min(misc_rate) into: min_misc
      from _tmp3;
      select k into: bestk separated by ', '
      from _tmp3
      having misc_rate = min(misc_rate);
   quit;
   ods select all;
   proc sgplot data = _tmp3;
      series x = k y = misc_rate;
      xaxis grid values = (1 to &maxk by 1)
          label = 'k number of neareast neighbours';
      yaxis grid values = ( 0 to 0.5 by 0.05)
          label = 'Misclassification rate';
      refline &min_misc / transparency = 0.3
          label = "k = &bestk";
      format misc_rate percent8.1;
   run;
   proc datasets nolist;
      delete _:;
   quit;
%mend;

ods html style = htmlblue;
%findk(train = iris_train, validate = iris_validate, target = species,
       input = petallength petalwidth sepallength sepalwidth, maxk = 20);
%findk(train = cars_train, validate = cars_validate, target = origin,
       input = invoice wheelbase length, maxk = 40);

****** END OF ALL CODING ************************************************;
```