ILLINOIS
Gies College of Business

Rules, Rules and More Rules

“From Data to Decisions”
Classification Methods

Sridhar Seshadri

Contents

Rules

This customer will buy something from that rack of clothes

This student looks like the ones who did well in the past

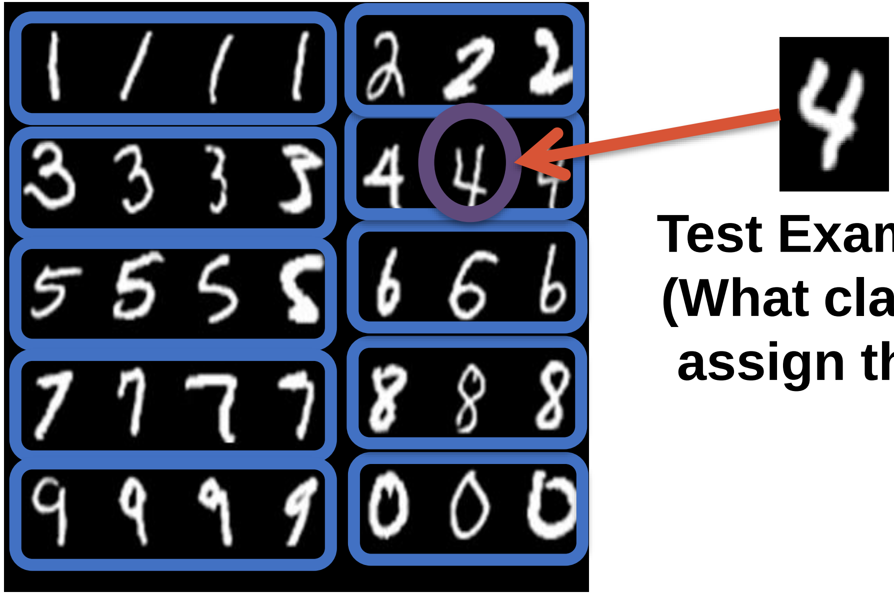
This situation calls for an emergency clampdown on credit limits!

K-Nearest Neighbor

Non-Parametric Classifier

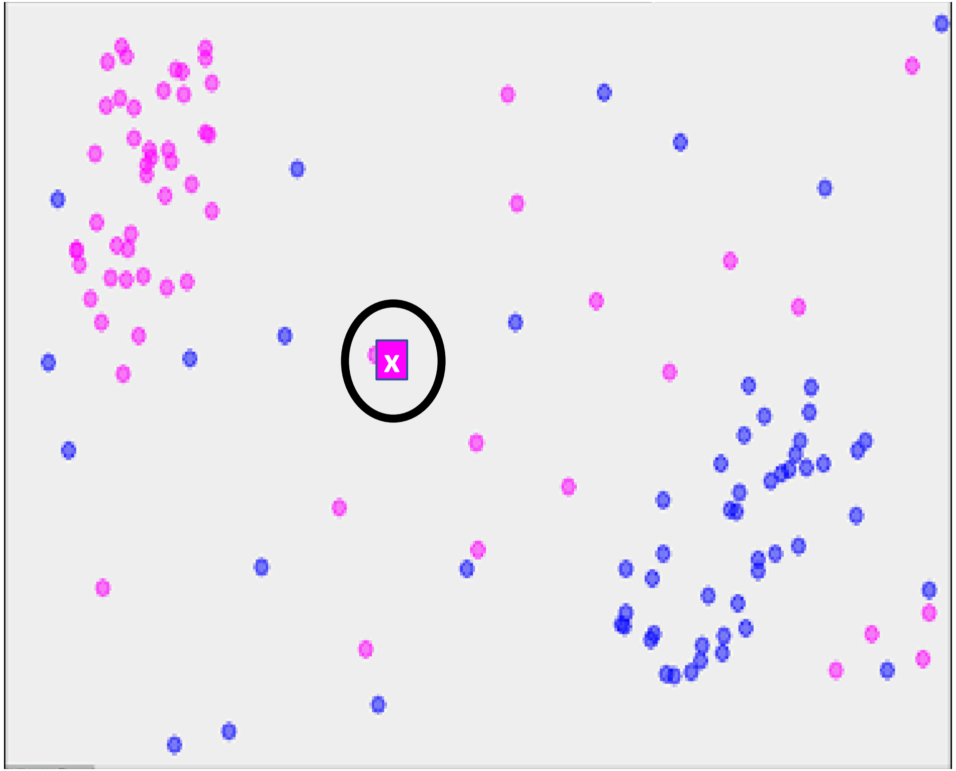
1-Nearest Neighbor Classifier

Training Examples (Instances)
Some for each CLASS

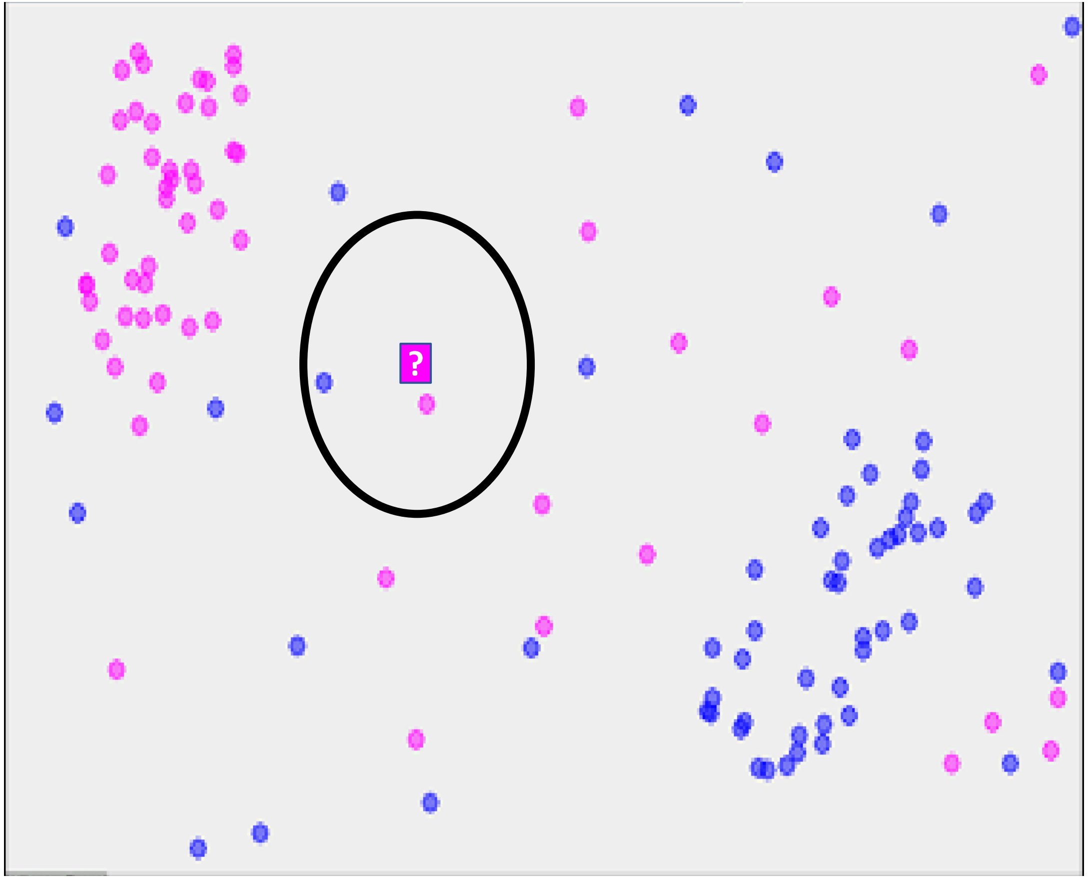


Source: LeCun et al. / http://yann.lecun.com/exdb/mnist/index.html

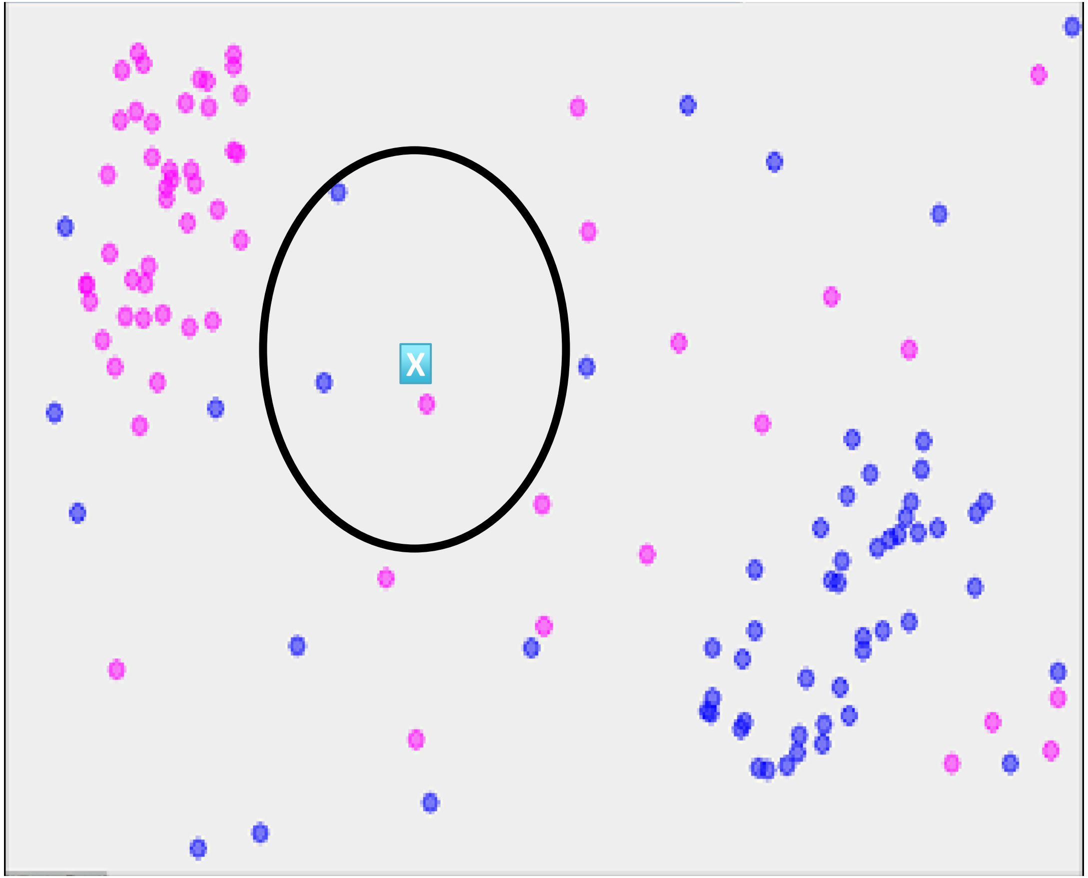
1-Nearest Neighbor



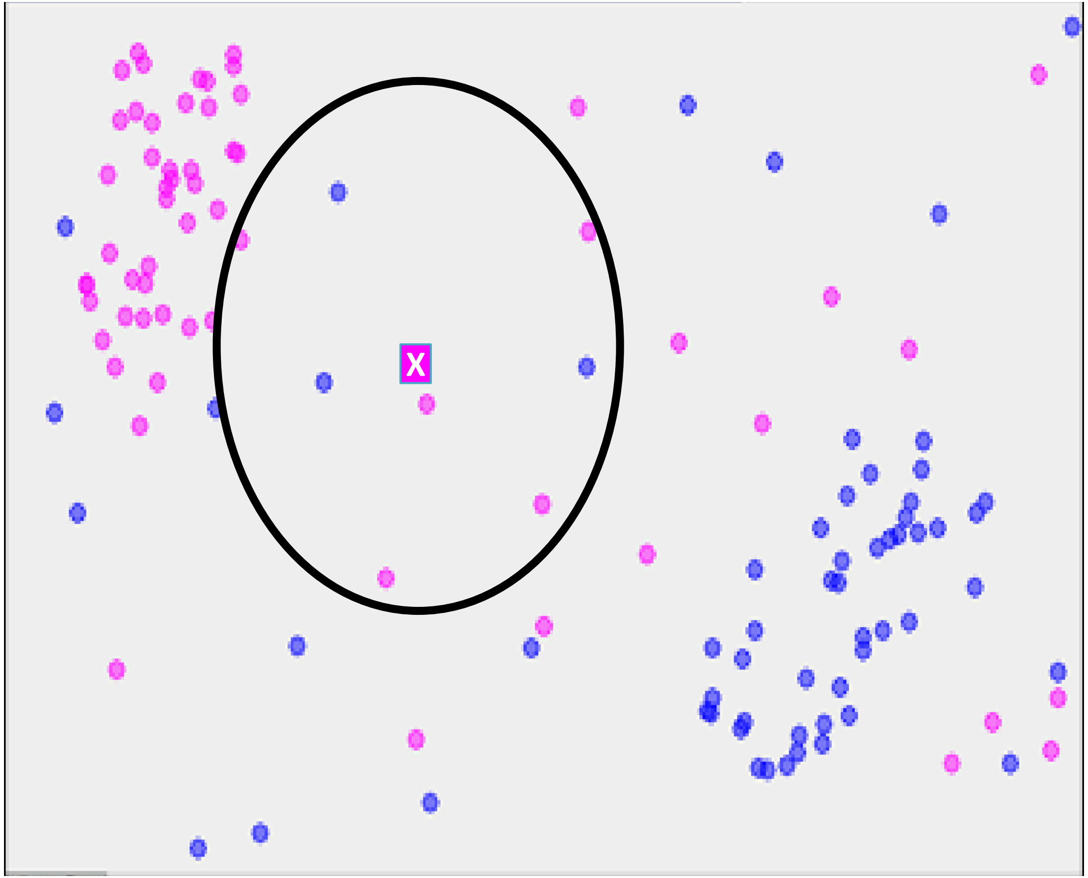
2-Nearest Neighbor



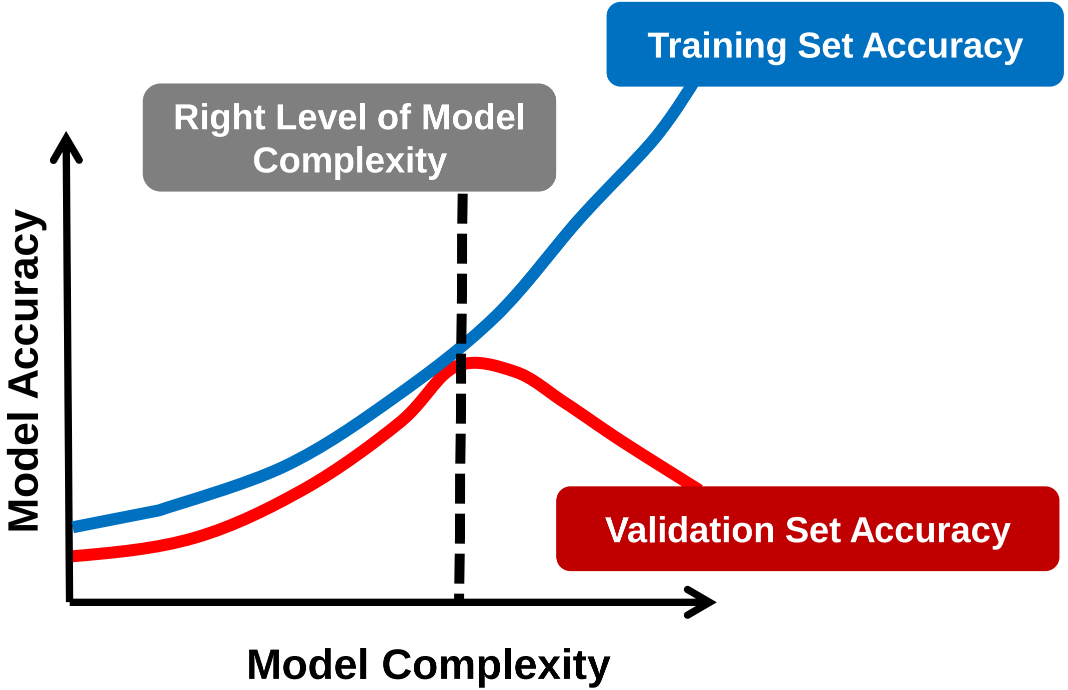
3-Nearest Neighbor



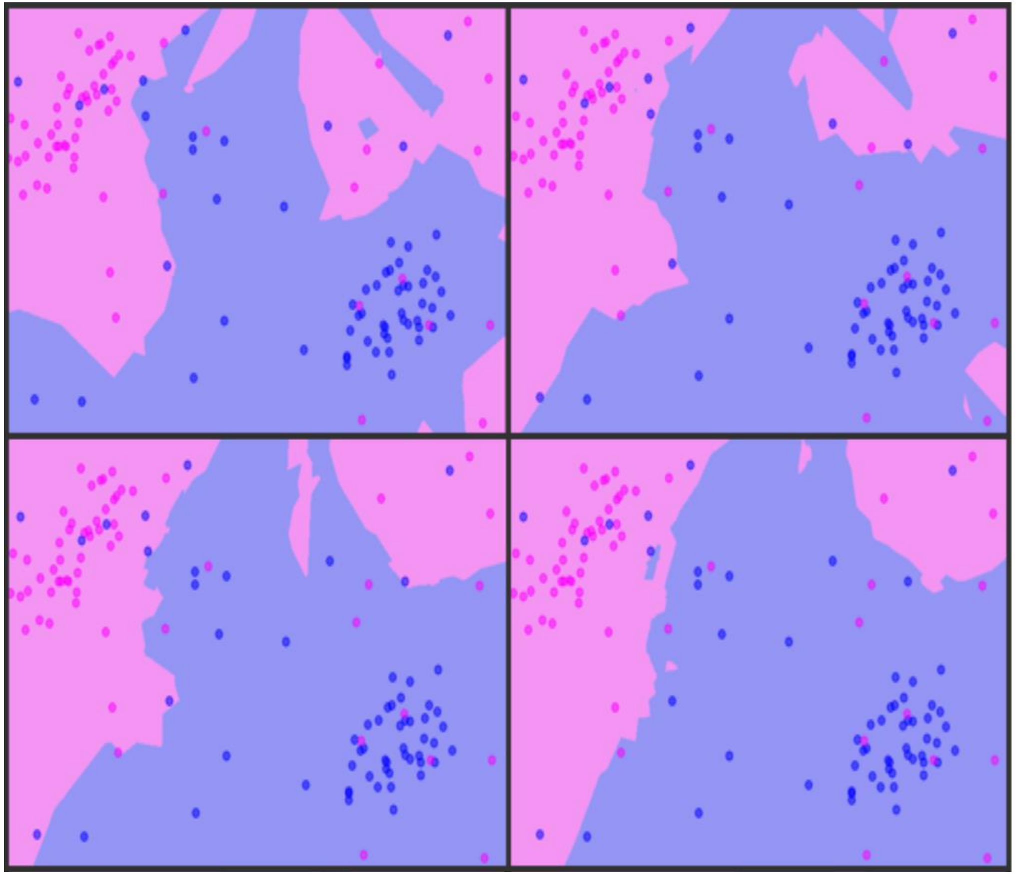
8-Nearest Neighbor



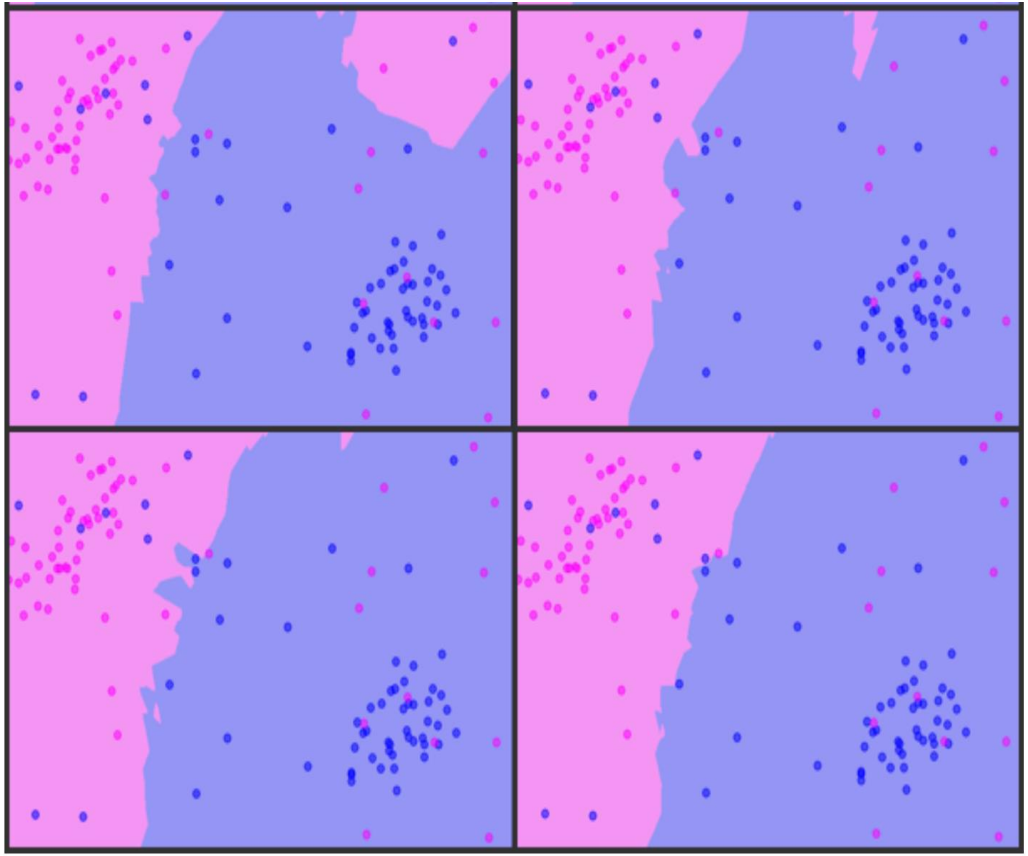
Controlling COMPLEXITY in k-NN



K = 3, 5, 7, 9



K = 11,13,15,17



K-Nearest Neighbor Classifier

What are the **PARAMETERS**?

What’s the **TRAINING TIME**?

What’s the **SCORING COMPLEXITY**?

Distance / Similarity function is the key

K-Nearest Neighbor Classifier

Choosing *K*

Even value of *K* could lead to confusion – breaking the tie

K = 1, 3, 5, 7, 9, …

Higher the Noise in the data, more *K* is better!

Better Lookup times using KD-Trees and other tricks!

Brittle in presence of noisy data

Loosing the actual distance value

Note that if training time involves search for k, then for each point have to find nearest neighbors etc. Scoring complexity depends on finding the class with the maximum number of points – which is relatively easy to do.

Data – Real Estate

Objective- predict the house price class (low, medium, high) based on the eight input variables.

Variable	Description
full_sq	TotalArea (in square feet)
life_sq	Living Area (in square feet)
floor	Floor House (on which floor the house is built)
max_floor	Max Floors (what is the maximum floors inthat building)
material	Type of Material used (labeled as 1,2,4,5,6)
build_year	Build Year (Year in 19xx in which the house isbuilt)
num_room	Number of Rooms
kitch_sq	Kitchen Area(in square feet)
priceClass	House price: low, med, high

realEstate.csv

Top five rows

id	full_sq	life_sq	floor	max_floor	material	build_year	num_room	kitch_sq	priceClass
1	187	187	2	10	2	143	1	204	Low
2	765	493	1	8	1	78	2	102	Low
3	646	408	5	7	2	85	2	85	Low
4	663	323	4	10	1	65	1	136	Low
5	1020	714	3	8	1	72	3	102	Low
6	1122	782	3	5	5	67	3	119	Low

Slight change in train/test split in R version 3.6.3

Preprocess and Prepare Data - New

KNN- session 6-April-10.R

```
install.packages("pacman")
library(pacman)
p_load("rsample","dplyr", "caTools","caret", "e1071", "FNN")
# Read the data file
real_es <- read.csv("realEstate.csv")
dim(real_es)  #  11995 rows and   10 columns

# Define input variables
X = real_es[,2:9]
# Define target variable
y = real_es[,10]

# Normalize the inputs
norm.values <- preProcess(X, method=c("center", "scale"))
X.norm <- predict(norm.values, X) # Normalized input

# train test split
sample = sample.split(real_es, SplitRatio = 0.80) # select a random sample of 80%
X_train = subset(X.norm, sample==TRUE) # input for training
X_test = subset(X.norm, sample==FALSE) # input for prediction accuracy
dim(X_train) # 9596 rows and   8 columns
dim(X_test) # 2399 rows and 8 columns
y_train = subset(y,sample==TRUE) # labels for training
y_test = subset(y, sample==FALSE) # labels for prediction accuracy
```

Use head(X.norm) to get the first six rows normalized input

Normalized input

full_sq	life_sq	floor	max_floor	material	build_year	num_room	kitch_sq
-1.9699778	-1.1495138	-0.8196683	-0.6481221	0.05410562	0.009119031	-1.10858437	0.16279274
-0.3870199	-0.2422583	-0.9211151	-0.7874589	-0.60477450	0.008763976	0.05826975	-0.06192231
-0.7129230	-0.4942738	-0.5153278	-0.8571273	0.05410562	0.008802212	0.05826975	-0.09937482
-0.6663654	-0.7462892	-0.6167747	-0.6481221	-0.60477450	0.008692965	-1.10858437	0.01298270
0.3113439	0.4129817	-0.7182215	-0.7874589	-0.60477450	0.008731201	1.22512386	-0.06192231
0.5906894	0.6145940	-0.7182215	-0.9964640	2.03074597	0.008703889	1.22512386	-0.02446981

Due to rounding off, the number of train and test can be 9595 and 2400

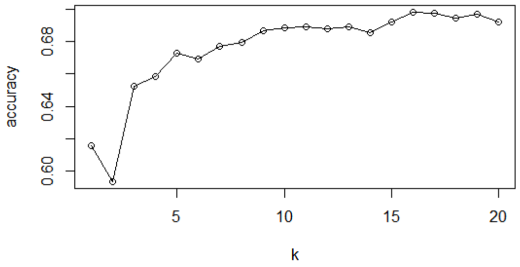
Selecting the Best K

```
nn_model <- knn(train = X_train[,1:3], test=X_test[,1:3], cl = y_train, k=5)
summary(nn_model)
confusionMatrix(nn, y_test)$overall[1]# accuracy of prediction on test (validation) data
# Accuracy -  0.67

###############################################################################

# define a dataframe in which we will save accuracy for different values of K
accuracy.df <- data.frame(k = seq(1, 20, 1), accuracy = rep(0, 20))
accuracy.df # right now we have filled accuracy to be 0 for all values of K

# compute knn for different k on validation by loopi
for(i in 1:20) { # we will loop through K= 1 to 20
  knn_model <- knn(train = X_train[1:3], test=X_test[1:3], cl = y_train, k = i)
  accuracy.df[i, 2] <- confusionMatrix(knn_model, y_test)$overall[1]
}
plot(accuracy.df) # plot accuracy for different values of K
lines(accuracy.df)
which.max(accuracy.df$accuracy) # optimal K = 16
```

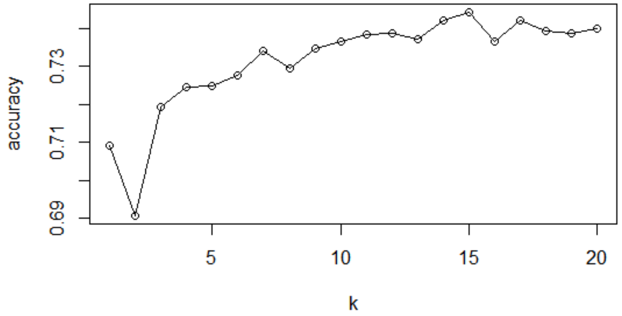


K	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Accuracy	0.6154	0.5933	0.6525	0.6588	0.6729	0.6696	0.6775	0.6796	0.6867	0.6888	0.6892	0.6879	0.6892	0.6858	0.6925	0.6983	0.6979	0.6946	0.6971	0.6921

Model Improvement - Add More Inputs

We will use all inputs.

```
# compute knn for different k on validation for all inputs
for(i in 1:20) {
  knn_model<- knn(train = X_train, test=X_test, cl = y_train, k = i)
  accuracy.df[i, 2] <- confusionMatrix(knn_model y_test)$overall[1]
}
plot(accuracy.df)
lines(accuracy.df)
which.max(accuracy.df$accuracy) # k=15
# accuracy 0.74
```



We see around 4 % improvement in accuracy

K	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Accuracy	0.7091667	0.6908333	0.7191667	0.7245833	0.725	0.7275	0.7341667	0.7295833	0.7345833	0.7366667	0.7383333	0.73875	0.7370833	0.7420833	0.7441667	0.7366667	0.7420833	0.7391667	0.73875	0.74

Summary

Small to moderate datasets

Not too many features

Interesting extension to categorical data

Handy and Good benchmark method

Bayes Rule

Prior probability

New data

Update

Example

DH Inc. offers service contracts for home appliances. It conducts several promotional campaigns in a year. Deepak, the VP for Marketing, has been reviewing the data from past campaigns and wants to fine tune the next one using previously gathered data.

Example - continued

Hema, his Data Science consultant suggests using a Bayesian approach to determine which prospects are more likely to accept the plan. To illustrate her approach, she says, “Let’s take an example marketing dataset with just 20 rows and 5 features / variables:

Data

customerID	Job	Marital_status	Education	Default_loan	own_house	BoughtProduct
1	Unknown	Single	Secondary	No	yes	yes
2	blue-collar	Single	Secondary	No	yes	Yes
3	blue-collar	Married	Secondary	No	No	Yes
4	Retired	Divorced	Tertiary	No	yes	Yes
5	blue-collar	Single	Tertiary	No	No	Yes
6	Retired	Married	Secondary	No	No	Yes
10	Student	Single	Secondary	No	No	Yes
11	blue-collar	Single	Secondary	Yes	yes	No
12	blue-collar	Married	Secondary	No	yes	Yes
13	Unknown	Married	Tertiary	No	No	Yes
14	Student	Single	Secondary	No	No	Yes
15	Student	Single	Tertiary	No	No	Yes
16	Student	Single	Secondary	No	No	Yes
17	Retired	Divorced	Primary	No	No	Yes
18	Retired	Married	Secondary	No	No	Yes
19	blue-collar	Married	Secondary	No	No	No
20	Unknown	Married	Secondary	No	No	No

Initial Musings

Suppose the new requirements of a new buyer are as shown below and we want to know whether he will buy the product or not:

job	Marital_status	Education	Default_loan	own_house
blue-collar	Single	tertiary	No	No

One way is to look up the table and see if any such person bought the product. The answer is Yes! One person of that type purchased the product (row 5).

P(Buy and data) = 1 person. P(Data) = 1 person.

P(Buy|Data) = P(Buy and Data)/P(Data)

Drawbacks and a Remedy

Doing this type of look-up might be difficult for many reasons, such as, having a huge data and numerous features. Therefore, the Naïve Bayes method comes in handy.

The logic for the Naïve Bayes algorithm is we count the occurrences of each level of a variable so that we can calculate the probability of purchase under each possible scenario using the independence assumption.

Drawbacks and a Remedy

In the table below, each variable has different number of levels. The data summary shows how many did not buy + how many did buy and the total. For example, of the 5 who own houses (level = 2), four bought.

Summary by Feature

Levels	Job	Marital_status	education	Default_loan	Own_house	boughtProduct
1	retired 2+4=6	divorced 0+2=2	Primary 2+1=3	No 4+14=18	No 5+10=15	No 6
2	unknown 2+2=4	Married 5+5=10	Secondary 4+9=13	Yes 2+0=2	Yes 1+4=5	Yes 14
3	Blue-collar: 2+4=6	Single 1+7=8	Tertiary 0+4=4			
4	student: 0+4=4					

Priors

P (buy = yes) = 14/20 = 7/10 =0.7.

P (buy = no) = 6/20 = 3/10 = 0.3.

P (job = retired | buy = no) = 2/6 = 0.333.

P (job = unknown | buy = no) = 2/6 = 0.333.

P (job = blue-collar | buy = no) = 2/6 = 0.334.

P (job = student| buy = no) = 0/6 = 0.

Priors

P (job = retired | buy = yes) = 4/14 = 0.285.

P (job = unknown | buy = yes) = 2/14 = 0.143.

P (job = blue-collar | buy = yes) = 4/14 = 0.285.

P (job = student| buy = yes) =4 /14 = 0.285.

Similarly – Remaining Priors

P (marital = divorced | buy = no) = 0/6 = 0.

P (marital = Married | buy = no) = 5/6 = 0.833.

P (marital = Single | buy = no) = 1/6 = 0.167

P (marital = divorced | buy = yes) = 2/14 = 0.143.

P (marital = Married | buy = yes) = 5/14 = 0.357.

P (marital = Single | buy = yes) = 7/14 = 0.5.

P (education = primary | buy = no) = 2/6 = 0.333.

P (education = Secondary | buy = no) = 4/6 = 0.667.

P (education = Tertiary | buy = no) = 0/6 = 0.

P (education = primary | buy = yes) = 1/14 = 0.071.

P (education = Secondary | buy = yes) = 9/14 = 0.643.

P (education = Tertiary | buy = yes) = 4/14 = 0.285.

Similarly – Remaining Priors

P (default = no | buy = no) = 4/6 = 0.667.

P (default = yes | buy = no) = 2/6 = 0.333.

P (default = no | buy = yes) = 14/14 = 1.

P (default = yes | buy = yes) = 0/14 = 0.

P (housing = no | buy = no) = 5/6 = 0.833.

P (housing = yes | buy = no) = 1/6 = 0.167.

P (housing = no | buy = yes) = 10/14 = 0.714.

P (housing = yes | buy = yes) = 4/14 = 0.285.

To Apply Bayes Method

P (buy=yes | Marital_status = single and education = tertiary and job = blueCollar and no default and no-house)

= P (buy=yes and Marital_status = single and education = tertiary and job = blueCollar and no default and no-house)

divided by

P (Marital_status = single and education = tertiary and job = blueCollar and no default and no-house)

The Naïve Bayes Trick

P (buy=yes and Marital_status = single and education = tertiary and job = blueCollar and no default and no-house)

= (approximately) P (buy=yes) * P (marital_status = Single | buy = yes) * P (education = tertiary | buy = yes) * P (job = blue-collar | buy = yes) * P (default = no | buy = yes) * P (housing = no | buy = yes)

= 0.7 * 0.5 * 0.285 * 0.285 * 1 * 0.714

=0.02029813.

Similarly – Naïve Bayes Trick

P (buy=No and marital status = single and education level = tertiary and job = blueCollar and no default and no house)

= (approximately) P (buy=no) * P (marital_status l = Single | buy = no) * P (education = tertiary | buy = no) * P (job = blue-collar | buy = no) * P (default = no | buy = no) * P (housing = no | buy = no)

= 0.3 * 0.1667 * 0 * 0.334 * 0.667 * 0.8334 = 0

Posterior Calculations

The sum of these two probabilities gives (approximately):

P (buy= yes and Marital_status = single and education = tertiary and job = blueCollar and no default and no-house)
+ P (buy=No and Marital_status = single and education = tertiary and job = blueCollar and no default and no-house)

= P (Marital status = single, education level = tertiary, job = blueCollar, no default, and no house) = 0.02029813.

Posterior Calculations

Thus, the model would predict the prospect will buy the product, (P(Buy| Data) = P(Buy;Data)/P(Data)

P (buy=yes and marital status = single, education level = tertiary, job = blueCollar, no default, and no house) / P (marital status = single, education level = tertiary, job = blueCollar, no default, and no house) = 1!

Another Calculation Example

As another illustration, we can ask what is the probability buyer will buy if we know the prospect is single and has secondary education? This will be given by:

P (Buy=yes)*P (Single|buy=yes)*P (Secondary|Buy=yes)/

(P (Buy=yes)*P (Single|buy=yes)*P (Secondary|Buy=yes) + P (Buy=No)*P (Single|Buy=No)*P (Secondary|Buy=no))=

(0.7)*(0.5)*(0.643)/ ((0.7)*(0.5)*(0.643) + (0.3)*(0.167)*(0.667)) = 0.871.

Another Calculation Example

In the data there were six customers of the given type, five out of them bought the service. The estimated probability from using the Bayesian (as against the naïve bayes) method is 0.833. This is quite close to the naïve estimate.

Data - Employee Attrition

Objective - Predict whether an employee would leave the organization

Variable Name	Description
ATTRITION	Employee leaving the company (0=no, 1=yes)
JOB LEVEL	Numerical Value - LEVEL OF JOB
BUSINESS TRAVEL	(1=No Travel, 2=Travel Frequently, 3=Tavel Rarely)
DEPARTMENT	(1=HR, 2=R&D, 3=Sales)
EDUCATION	Numerical Value
EDUCATION FIELD	(1=HR, 2=LIFE SCIENCES, 3=MARKETING, 4=MEDICAL SCIENCES, 5=OTHERS, 6= TEHCNICAL)
MARITAL STATUS	(1=DIVORCED, 2=MARRIED, 3=SINGLE)
TRAINING TIMES LAST YEAR	Numerical Value - HOURS SPENT TRAINING
STOCK OPTIONS LEVEL	Numerical Value - STOCK OPTIONS

Source: “rsample” package R

```
Attrition JobLevel      BusinessTravel           Department     Education EducationField MaritalStatus TrainingTimesLastYear StockOptionLevel
      Yes        2       Travel_Rarely                Sales       College  Life_Sciences        Single                     0                0
       No        2 Travel_Frequently Research_Development Below_College  Life_Sciences       Married                     3                1
      Yes        1       Travel_Rarely Research_Development       College          Other        Single                     3                0
       No        1 Travel_Frequently Research_Development        Master  Life_Sciences       Married                     3                0
       No        1       Travel_Rarely Research_Development Below_College        Medical       Married                     3                1
       No        1 Travel_Frequently Research_Development       College  Life_Sciences        Single                     2                0
```

Preprocessing - New

NaiveBayes - session_7-April-10.R

```
install.packages("pacman") # Package manager
library(pacman)
p_load("rsample","dplyr", "caTools","caret", "e1071")

dim(attrition)# There are 31 variable, we will use only 9 variables, 1 target and 8 input
# we will select a few variables to predict attrition
# BusinessTravel, Department, Education, EducationField,  Marital Status, TrainingTimesLastYear, StockOptionLevel

# Subsetted dataframe
attrition.df <- attrition[,c("Attrition","JobLevel","BusinessTravel", "Department", "Education", "EducationField",  "MaritalStatus",
"TrainingTimesLastYear", "StockOptionLevel")]

set.seed(101)
Attrition.df <- attrition.df %>% # Convert numeric variables into categorical/ factor so Naive Bayes takes as categorical input
  mutate( JobLevel = factor(JobLevel),
    StockOptionLevel = factor(StockOptionLevel),
    TrainingTimesLastYear = factor(TrainingTimesLastYear)
)

# train test split
sample = sample.split(attrition.df, SplitRatio = 0.80) # seelect a random sample of 80%
train <- subset(attrition.df, sample==TRUE) # training
test <- subset(attrition.df, sample=FALSE) # test

# distribution of Attrition rates across train & test set
table(train$Attrition) %>% prop.table()
table(test$Attrition) %>% prop.table()
```

Naïve Bayes to Predict Attrition

On the basis of "JobLevel“, "Department", "Education", and "EducationField"

```
# We perform Naive Bayes on "JobLevel" + "Department", "Education", "EducationField"
Naive_Bayes_Model_4=naiveBayes(train$Attrition ~ JobLevel + Department + Education + EducationField, data=train)
Naive_Bayes_Model_4$tables  #Produces the prior tables

# Accuracy on training set -  here we set the cutoff probability to “0.2” instead of “0.5” (which is default). We do so  because the data is unbalanced with very high proportion of “No” than “Yes”.
train_predictions_4 = as.data.frame(predict(Naive_Bayes_Model_4, train[,-1], type="raw")) # removing target variable from dataframe for prediction
confusionMatrix(as.factor(ifelse(train_predictions_4$Yes> 0.2, 'Yes', 'No')), train$Attrition)
# Accuracy - 0.6859

# Accuracy on the test set
test_predictions_4 = as.data.frame(predict(Naive_Bayes_Model_4, test[,-1], type="raw"))
confusionMatrix(as.factor(ifelse(test_predictions_4$Yes>0.2, 'Yes', 'No')), test$Attrition)
# Accuracy - 0.682 – The model does 68% accurate prediction.
```

Can we improve the model performance on Test Data by adding more information (variables)?

Model (Prediction) Improvement – Adding More Variables

```
# Naive Bayes with all 8 input
Naive_Bayes_Model_8=naiveBayes(train$Attrition ~., data=train)

# Accuracy on training set
train_predictions_8 = as.data.frame(predict(Naive_Bayes_Model_8, train[,-1], type="raw")) # removing target variable from dataframe for prediction
confusionMatrix(as.factor(ifelse(train_predictions_8$Yes>0.2, 'Yes', 'No')), train$Attrition)
# Accuracy - 0.727

# Accuracy on the test set
test_predictions_8 = as.data.frame(predict(Naive_Bayes_Model_8, test[,-1], type="raw"))
confusionMatrix(as.factor(ifelse(test_predictions_8$Yes>0.2, 'Yes', 'No')), test$Attrition)
# Accuracy - 0.715

# We see that the model performance improves by adding more information.
```

Exercise

Can you use Real_estate data and perform Naïve Bayes to predict house price class?

If yes, do you need to transform your input variable? Why?

Perform Naïve Bayes to predict house price class, suitably transforming the input data.

Exercise

Use Employee attrition data to predict whether an employee would leave using KNN method.

What transformation would you have to perform on the input data to before running KNN on the employee attrition data?

Summary

Simple rules are often quite powerful

Seek similarity based on data or past experience

Predict for new data

Too many features create problems

Further Readings

Naïve Bayes

Decision Tree and Naïve Bayes Algorithm for Classification and Generation of Actionable Knowledge for Direct Marketing
http://file.scirp.org/pdf/JSEA_2013042913162682.pdf

Business Applications of Data Mining
http://www.inf.uni-konstanz.de/dbis/teaching/ws0607/busintelligence/papers/BA_DM.pdf

KNN

Foreign Exchange Market Prediction with Multiple Classifiers
https://onlinelibrary.wiley.com/doi/pdf/10.1002/for.1124

Managerial Applications of Neural Networks: The Case of Bank Failure Predictions
(This is not open access)
https://pubsonline.informs.org/doi/pdf/10.1287/mnsc.38.7.926

Preprocess and Prepare Data

```
library(caret)
library(e1071)
library(caTools)
install.packages("FNN") # for KNN
library(FNN)

# Read the data file
real_es <- read.csv("realEstate.csv")
dim(real_es)  #  11995 rows and   10 columns

# Define input variables
X = real_es[,2:9]
# Define target variable
y = real_es[,10]

# Normalize the inputs
norm.values <- preProcess(X, method=c("center", "scale"))
X.norm <- predict(norm.values, X) # Normalized input

# train test split
sample = sample.split(real_es, SplitRatio = 0.80) # select a random
sample of 80%
X_train = subset(X.norm, sample==TRUE) # input for training
X_test = subset(X.norm, sample==FALSE) # input for prediction
accuracy
dim(X_train) # 9596 rows and   8 columns
dim(X_test) # 2399 rows and 8 columns
y_train = subset(y,sample==TRUE) # labels for training
y_test = subset(y, sample==FALSE) # labels for prediction accuracy
```

Normalized input

full_sq	life_sq	floor	max_floor	material	build_year	num_room	kitch_sq
-1.9699778	-1.1495138	-0.8196683	-0.6481221	0.05410562	0.009119031	-1.10858437	0.16279274
-0.3870199	-0.2422583	-0.9211151	-0.7874589	-0.60477450	0.008763976	0.05826975	-0.06192231
-0.7129230	-0.4942738	-0.5153278	-0.8571273	0.05410562	0.008802212	0.05826975	-0.09937482
-0.6663654	-0.7462892	-0.6167747	-0.6481221	-0.60477450	0.008692965	-1.10858437	0.01298270
0.3113439	0.4129817	-0.7182215	-0.7874589	-0.60477450	0.008731201	1.22512386	-0.06192231
0.5906894	0.6145940	-0.7182215	-0.9964640	2.03074597	0.008703889	1.22512386	-0.02446981

Preprocessing

```
install.packages("rsample")
library(rsample) # to read dataset on Attrition
library(dplyr)
library(caTools)
library(caret)

dim(attrition)# there are 31 variable, we will use only 9 variables, 1 target and 8 input
# we will select a few variables to predict attrition
# BusinessTravel, Department, Education, EducationField,  Marital Status, TrainingTimesLastYear,
StockOptionLevel , …

# Subsetted dataframe
attrition.df <- attrition[,c("Attrition","JobLevel","BusinessTravel", "Department", "Education", "EducationField",
"MaritalStatus", "TrainingTimesLastYear", "StockOptionLevel")]

set.seed(101)
Attrition.df <- attrition.df %>% # Convert numeric variables into categorical/ factor so Naive Bayes takes as
categorical input
  mutate( JobLevel = factor(JobLevel),
    StockOptionLevel = factor(StockOptionLevel),
    TrainingTimesLastYear = factor(TrainingTimesLastYear)
)

# train test split
sample = sample.split(attrition.df, SplitRatio = 0.80) # select a random sample of 80%
train <- subset(attrition.df, sample==TRUE) # training
test <- subset(attrition.df, sample=FALSE) # test

# distribution of Attrition rates across train & test set
table(train$Attrition) %>% prop.table()
table(test$Attrition) %>% prop.table()
```

References

LeCun, Y., Cortes, C., Burges, C.J.C. (n.d.). *The MNIST Database of Handwritten Digits.* Retrieved May 22, 2019, from http://yann.lecun.com/exdb/mnist/index.html

Mirkes, E. (2011). *KNN and Potential Energy (Applet).* Retrieved May 22, 2019 from University of Leicester: http://www.math.le.ac.uk/people/ag153/homepage/KNN/KNN3.html