Data Mining Methods

Data Pre-Processing

Prof. Dr. Christina Andersson

High Integrity Systems
Frankfurt University of Applied Sciences

Booklet to sprint 2

1 Getting Started

Garbage in, garbage out is a well-known principle, which should be kept in mind when dealing with data mining. The data preparation process often needs much effort, but is absolutely indispensable.

2 Some Useful Resources

- Very basics about pre-processing:
  https://medium.com/easyread/basics-of-data-preprocessing-71c314bc7188
- Very good, a comprehensive book with a lot of ideas:
  Data Preparation for Analytics Using SAS, 2006, Gerhard Svolba, ISBN: 978-1599940472
- Really good, but of course very much (= don't panic!):
  Data preparation for data mining, Dorian Pyle

3 R

3.1 Packages

- **DataExplorer**
- **mlr**

3.2 Some Useful Commands

is.na	**sum**	**na.omit**	**dim**	**plot_missing**	**impute**
read.csv	**set.seed**	**nrow**	**sample**	**length**	**unique**
hist	**log**	**data.frame**			

4 Exercises

1. Make sure that the dataset *airquality* in the package *datasets* is available in your R session (otherwise: Load it).
   (a) Calculate the mean of the variable *Ozone* in the dataset *airquality*. Do you encounter any problems?
   (b) Use the function *is.na* to determine how many missing numbers there are in the variable *Ozone*. What value is returned by the function *is.na*?
   (c) How many missing numbers are there totally in the dataset *airquality*?

(d) What does the option *na.rm* in the function *mean* do? Use this to calculate the mean of *Ozone*.

(e) Apply the function *na.omit* to the dataset *airquality* to construct a new dataset without any missing values. Compare the number of observations in the new dataset and in the original dataset.

(f) Apply the function *complete.cases* to the variable *Ozone*. What does this function do?

(g) Use the function *which* together with the argument *arr.ind* to determine the indices (i.e. column and row numbers) for the missing observations in the dataset *airquality*.

2. In this exercise, we're going to use the function *impute* in the package *mlr* to compare some different methods for replacing missing values (= impute) in the variable *Ozone* in the dataset *airquality*.

(a) Load the library *DataExplorer*.

(b) Use the function *plot_missing* to get an overview over the missing values in the dataset *airquality*.

(c) Load the library *mlr*.

(d) Replace the missing values in the variable *Ozone* in the original data with the constant 100. What's the difference if you in the function *imputation* use the constant 100 or the constant "100"?

(e) Replace the missing values in the variable *Ozone* in the original data with the median of the original variable *Ozone*.

(f) Replace the missing values in the variable *Ozone* in the original data with the constant 0.

(g) Replace the missing values in the variable *Ozone* in the original data with random numbers drawn from an appropriate normal distribution.

(h) Replace the missing values in the variable *Ozone* in the original data with random numbers drawn from the actual distribution of the data.

(i) Use one graphics window to plot six histograms for the original data and the resulting data from the five imputation methods used above (Function *par(mfrow=c(yourvalue, yourvalue))* to plot more than one histogram in the same window).

Each histogram should consist of 25 groups (bins). All histograms shall have the same y-axis scaling.

(j) Calculate the mean value for the original data as well as the five means for the resulting data from the five imputation methods used above (one mean for each imputation method).

(k) Compare the five distributions in e) and also compare the five mean values in f).

Which imputation method would you prefer? Why?

3. **Case Study:**
Direct mail marketing campaign for clothing

The data in this real-world case study are based on a direct mail marketing campaign conducted last year. Our final goal will be to develop classification models for this year's marketing campaign (so that we can predict if a customer is going to respond to the markeing campaign or not).

The dataset contains many different variables with customer information, useful or not, dirty or not ...

The way to reach our final goal is long and we've got to struggle a bit. In this assignment, we take a first glance at the data and do some initial data preparation. The case study will be continued in other assigments (also using the results of this assignment! Don't forget to save your results!).

(a) Read the dataset *Clothing_Store* into R. This dataset is described below.

(b) What are the variable names in this dataset?

(c) How many observations and variables are there in the dataset?

(d) Use the function *sample* (in the base package) to randomly select 70% of the dataset to be in the training dataset and the remaining 30% of the dataset to be in the validation dataset.

Hint:
Actually, you are first just going to randomly select 70% of the total **number** of observations in the dataset *Clothing_Store* and the corresponding 30% as you choose the validation data. As a second step, you then select the training data as the data rows in the dataset *clothing* having the row numbers for training data . The other rows in *clothing* are the validation data.

Another hint:
The functions *set_len* and *setdiff* can be useful.

Why is the partitioning into training and validation data important for the modeling process?
Should sampling be with replacement or not?
How many observations are there in the training data set? How many observations are there in the validation data set?

(e) Find a variable with two unique levels!
Find a variable with 15217 unique levels!
Determine the number of unique levels for some of the other variables!

Hint:
You can use the function *unique* to determine the different levels of a variable.

(f) Distribution of the target variable
The target variable is the variable *RESP*. Construct a histogram to see the distribution of this variable. What are the different levels? What is the proportion of responders? What about symmetry/skewness?

(g) Variables to reject
Let us first discuss if there are any variables that we can reject in the analysis. The first would be the variable containing the customer id *HHKEY*. Why shouldn't we use this variable as an input variable in the analysis?

Another variable that we are going to reject here is the variable *ZIP_CODE*. Could you think of any potential information that this variable could contain?

Create a new training dataset and a new validation dataset (to be used in the following tasks) without the variables *HHKEY* and *ZIP_CODE*.

(h) Skewness
Many of the numeric variables are right skewed. Unfortunately many standard data mining methods require that the variables are symmetric. But often is it possible to transform the original data to obtain symmetry.

Construct a histogram for the variable describing product uniformity, called *HI*. This variable takes large values for customers purchasing only from a few different classes of clothes (e.g. blouses, pants, ...) and small values for customers purchasing from many different cloth classes. This variable is clearly right skewed.

Try some different transformations (e.g. $t^{-2}, t^{-1}, t^{-1/2}, \ln(t), \sqrt{t}, t^1, t^2$) and use histograms to see if you get symmetry. Choose one of the transformations, apply this to *HI* in both the training and validation data and use these datasets in the following.

(i) Derivation of new variables
Usually the variables "delivered" to us in the database are not the most appropriate variables for the analysis. So, often we have to derive new variables, based on some of the original ones. In our case study, let's look at the following three variables: *Amount spent by customer in the last month0* (*OMONSPEND*), *Amount spent by customer in the last three months* (*TMONSPEND*), *Amount spent by customer in the six last months* (*SMONSPEND*).

Would it be good to use all these three variables as input variables at the same time? Why/why not? Which variables would you derive? Do this! (Both for the training data and for the validation data.)

(j) So, now this exercise is finished! Note that all changes to variables (transformations, derivations of new variables, etc.) should be done for both the training data and the validation data. These datasets are going to be used in later assignments!!!

To save your data you can use the command *write.table* (if you have your data as a dataframe).

5 Solutions to Exercises

1. Make sure that the dataset *airquality* in the package *datasets* is available in your R session (otherwise: Load it).

   Solution:

   ```
   > library(datasets)
   ```

   (a) Calculate the mean of the variable *Ozone* in the dataset *airquality*. Do you encounter any problems?

   Solution:

   ```
   > attach(airquality)
   > mean(Ozone)
   [1] NA
   ```

   Missing values in Ozone.

   (b) Use the function *is.na* to determine how many missing numbers there are in the variable *ozone*. What value is returned by the function *is.na*?

   Solution:

   ```
   > is.na(Ozone)
     [1] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
    [13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
    [25]  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE
    [37]  TRUE FALSE  TRUE FALSE FALSE  TRUE  TRUE FALSE  TRUE  TRUE FALSE FALSE
    [49] FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
    [61]  TRUE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE
    [73] FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE
    [85] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
    [97] FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE  TRUE FALSE
   [109] FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE
   [121] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
   [133] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
   [145] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE
   > sum(is.na(ozone))
   [1] 37
   ```

   (c) How many missing numbers are there totally in the dataset *airquality*?

   Solution:

   ```
   > sum(is.na(airquality))
   [1] 44
   ```

   (d) What does the option *na.rm* in the function *mean* do? Use this to calculate the mean of *Ozone*.

   Solution:

```
> mean(Ozone,na.rm=TRUE)
[1] 42.12931
```

(e) Apply the function *na.omit* to the dataset *airquality* to construct a new dataset without any missing values. Compare the number of observations in the new dataset and in the original dataset.

Solution:

```
> airquality_without_missing=na.omit(airquality)
> dim(airquality_without_missing)
[1] 111   6
> dim(airquality)
[1] 153   6
```

(f) Apply the function *complete.cases* to the variable *Ozone*. What does this function do?

Solution:

```
> missing_ind=complete.cases(Ozone)
> missing_ind
  [1]  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE
 [13]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
 [25] FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE
 [37] FALSE  TRUE FALSE  TRUE  TRUE FALSE FALSE  TRUE FALSE FALSE  TRUE  TRUE
 [49]  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [61] FALSE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE
 [73]  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE
 [85]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
 [97]  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE  TRUE  TRUE  TRUE FALSE  TRUE
[109]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE FALSE  TRUE
[121]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
[133]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
[145]  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE
```

Creates a missing indicator for the variable ozone.

(g) Use the function *which* together with the argument *arr.ind* to determine the indices (i.e. column and row numbers) for the missing observations in the dataset *airquality*.

Solution:

```
> d=which(is.na(airquality),arr.ind=TRUE)
> dim(d)
[1] 44  2
```

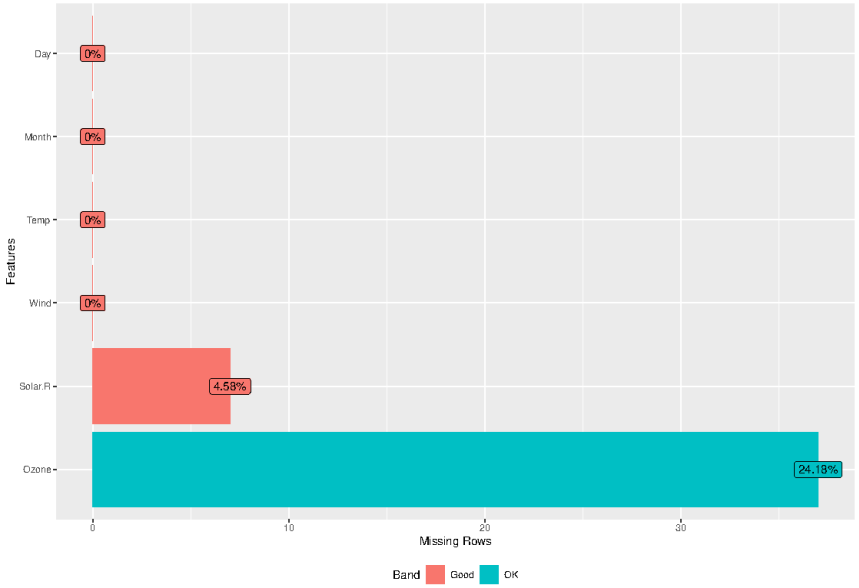
2. In this exercise, we're going to use the function *impute* in the package *mlr* to compare some different methods for replacing missing values (= impute) in the variable *Ozone* in the dataset *airquality*.

(a) Load the library DataExplorer.

```
> library(DataExplorer)
```

(b) Use the function *plot_missing* to get an overview over the missing values in the dataset *airquality*.

```
> plot_missing(airquality)
```



(c) Load the library *mlr*.

```
> library(mlr)
```

(d) Replace the missing values in the variable *Ozone* in the original data with the constant 100. What's the difference if you in the function *imputation* use the constant 100 or the constant "100"? **Solution:**

```
> imputed = impute(airquality, cols = list(Ozone = 100))
> imp_ozone=imputed$data[,1]
```

(e) Replace the missing values in the variable *Ozone* in the original data with the median of the original variable *Ozone*.

Solution:

```
> imputed2 = impute(airquality, cols = list(Ozone = imputeMedian()))
> imp_ozone2=imputed2$data[,1]
```

(f) Replace the missing values in the variable *Ozone* in the original data with the constant 0.

Solution:

```
> imputed3 = impute(airquality, cols = list(Ozone = 0))
> imp_ozone3=imputed3$data[,1]
```

(g) Replace the missing values in the variable *Ozone* in the original data with random numbers drawn from an appropriate normal distribution.

Solution:

```
> imputed4 = impute(airquality, cols = list(Ozone = imputeNormal()))
> imp_ozone4=imputed4$data[,1]
```

(h) Replace the missing values in the variable *Ozone* in the original data with random numbers drawn from the actual distribution of the data.

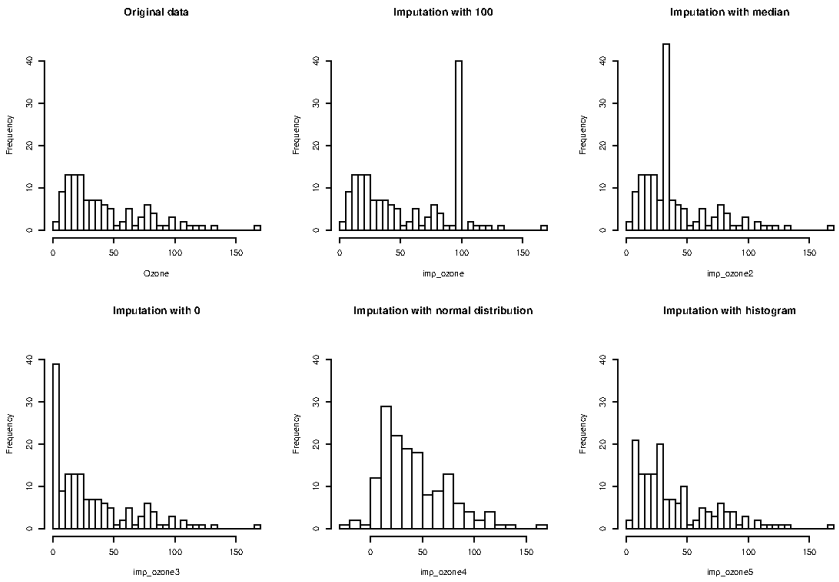
```
> imputed5 = impute(airquality, cols = list(Ozone = imputeHist()))
> imp_ozone5=imputed5$data[,1]
```

(i) Use one graphics window to plot six histograms for the original data and the resulting data from the five imputation methods used above (Function *par(mfrow=c(yourvalue, yourvalue))* to plot more than one histogram in the same window).

Each histogram should consist of 25 groups (bins). All histograms shall have the same y-axis scaling.

Solution:

```
> par(mfrow=c(2,3))
> hist(Ozone,breaks=25,ylim=(c(0,45)),main="Original data")
> hist(imp_ozone,breaks=25,ylim=(c(0,45)),main="Imputation with 100")
> hist(imp_ozone2,breaks=25,ylim=(c(0,45)),main="Imputation with median")
> hist(imp_ozone3,breaks=25,ylim=(c(0,45)),main="Imputation with 0")
> hist(imp_ozone4,breaks=25,ylim=(c(0,45)),main="Imputation with normal distr.")
> hist(imp_ozone5,breaks=25,ylim=(c(0,45)),main="Imputation with histogram")
```



(j) Calculate the mean value for the original data as well as the five means for the resulting data from the five imputation methods used above (one mean for each imputation method).

Solution:

```
> mean(imp_ozone5)
[1] 40.89542
> mean(imp_ozone4)
[1] 42.55786
> mean(imp_ozone3)
[1] 31.94118
> mean(imp_ozone2)
[1] 39.55882
> mean(imp_ozone)
[1] 56.12418
> mean(Ozone,na.rm=TRUE)
[1] 42.12931
```

(k) Compare the five distributions in e) and also compare the four mean values in f).
Which imputation method would you prefer? Why?
...

3. **Case Study:**
Direct mail marketing campaign for clothing

The data in this real-world case study are based on a direct mail marketing campaign conducted last year. Our final goal will be to develop classification models for this year's marketing campaign (so that we can predict if a customer is going to respond to the markeing campaign or not).

The dataset contains many different variables with customer information, useful or not, dirty or not ...

The way to reach our final goal is long and we've got to struggle a bit. In this assignment, we take a first glance at the data and do some initial data preparation. The case study will be continued in other assigments (also using the results of this assignment! Don't forget to save your results!).

(a) Load the dataset *Clothing_Store* into R. This dataset is described below.

Solution:

```
clothing = read.csv(file=''your_path/clothing_store.txt'')
```

(b) What are the variable names in this dataset?

Solution:

```
> names(clothing)
 [1] "HHKEY"        "ZIP_CODE"     "REC"          "FRE"          "MON"
 [6] "CC_CARD"      "AVRG"         "PC_CALC20"    "PSWEATERS"    "PKNIT_TOPS"
[11] "PKNIT_DRES"   "PBLOUSES"     "PJACKETS"     "PCAR_PNTS"    "PCAS_PNTS"
```

```
[16] "PSHIRTS"      "PDRESSES"     "PSUITS"       "POUTERWEAR"   "PJEWELRY"
[21] "PFASHION"     "PLEGWEAR"     "PCOLLSPND"    "AMSPEND"      "PSSPEND"
[26] "CCSPEND"      "AXSPEND"      "TMONSPEND"    "OMONSPEND"    "SMONSPEND"
[31] "PREVPD"       "GMP"          "PROMOS"       "DAYS"         "FREDAYS"
[36] "MARKDOWN"     "CLASSES"      "COUPONS"      "STYLES"       "STORES"
[41] "STORELOY"     "VALPHON"      "WEB"          "MAILED"       "RESPONDED"
[46] "RESPONSERATE" "HI"           "LTFREDAY"     "CLUSTYPE"     "PERCRET"
[51] "RESP"
```

(c) How many observations and variables are there in the dataset?

Solution:

```
> dim(clothing)
[1] 21740    51
```

(d) Use the function *sample* (in the base package) to randomly select 70% of the dataset to be in the training dataset and the remaining 30% of the dataset to be in the validation dataset. Setr the seed of the random number generator to 42, to make your sample replicable.
Hint:
Actually, you are first just going to randomly select 70% of the total **number** of observations in the dataset *Clothing_Store* and the corresponding 30% as you choose the validation data. As a second step, you then select the training data as the data rows in the dataset *clothing* having the row numbers for training data . The other rows in *clothing* are the validation data.
Another hint:
The functions *set_len* and *setdiff* can be useful.

Solution:

```
> set.seed(42)
> clothing_nobs = nrow(clothing)
```

To get the row numbers of the sampled training data:

```
> clothing_train_no = sample(nrow(clothing), 0.7*clothing_nobs)
```

To get the row numbers of the sampled validation data:

```
> clothing_validate_no = sample(setdiff(seq_len(nrow(clothing)),
clothing_train_no), 0.30*clothing_nobs)
```

To get the length of the future training and validation data:

```
> length(clothing_train_no)
[1] 15217
> length(clothing_validate_no)
[1] 6522
```

Now, create the training and validation data sets from the total data set, using the randomly selected row numbers above:

```
training_data=clothing[clothing_train_no,]
validation_data=clothing[clothing_validate_no,]
```

Why is the partitioning into training and validation data important for the modeling process?
Should sampling be with replacement or not?
How many observations are there in the training data set? How many observations are there in the validation data set?

Solution:

```
> dim(training_data)
[1] 15217    51
> dim(validation_data)
[1] 6522   51
```

(e) Find a variable with two unique levels!
Find a variable with 15217 unique levels!
Determine the number of unique levels for some of the other variables!

Hint:
You can use the function *unique* to determine the different levels of a variable.

Solution:

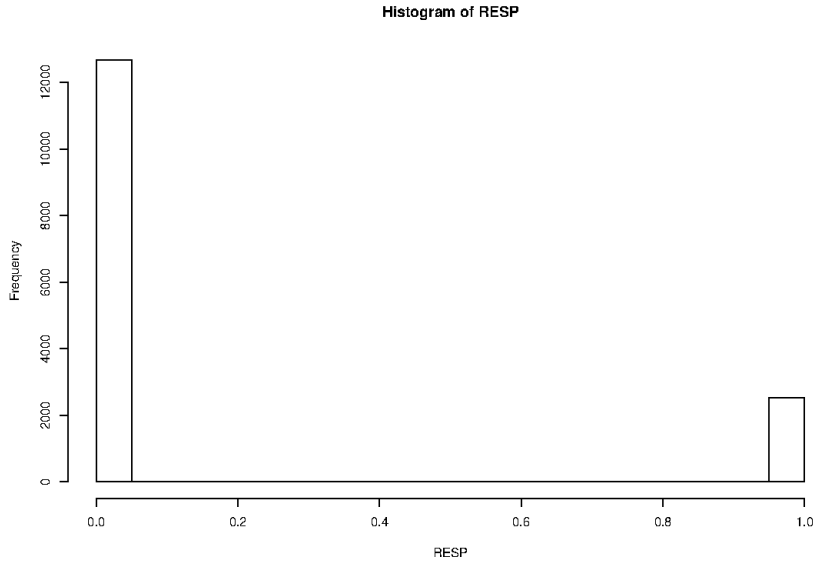
```
> attach(training_data)
> length(unique(HHKEY))
[1] 15217
> length(unique(RESP))
[1] 2
> length(unique(PKNIT_TOPS))
[1] 378
> length(unique(CLUSTYPE))
[1] 51
> length(unique(CC_CARD))
[1] 2
> length(unique(WEB))
[1] 2
```

(f) Distribution of the target variable
The target variable is the variable *RESP*. Construct a histogram to see the distribution of this variable. What are the different levels? What is the proportion of responders? What about symmetry/skewness?

Solution:

```
> hist(RESP)
```

```
> unique(RESP)
[1] 0 1
> data_one=training_data[RESP==1,]
> data_zero=training_data[RESP==0,]
> dim(data_zero)
[1] 12708    51
> dim(data_one)
[1] 2509   51
```



(g) Variables to reject

Let us first discuss if there are any variables that we can reject in the analysis. The first would be the variable containing the customer id *HHKEY*. Why shouldn't we use this variable as an input variable in the analysis?

Another variable that we are going to reject here is the variable *ZIP_CODE*. Could you think of any potential information that this variable could contain?

Create a new training dataset and a new validation dataset (to be used in the following tasks) without the variables *HHKEY* and *ZIP_CODE*.

Solution:

```
> tr=training_data[,-1]
> tr=tr[,-1]
> names(tr)
 [1] "REC"          "FRE"          "MON"          "CC_CARD"      "AVRG"
 [6] "PC_CALC20"    "PSWEATERS"    "PKNIT_TOPS"   "PKNIT_DRES"   "PBLOUSES"
[11] "PJACKETS"     "PCAR_PNTS"    "PCAS_PNTS"    "PSHIRTS"      "PDRESSES"
[16] "PSUITS"       "POUTERWEAR"   "PJEWELRY"     "PFASHION"     "PLEGWEAR"
```

```
[21] "PCOLLSPND"    "AMSPEND"      "PSSPEND"      "CCSPEND"      "AXSPEND"
[26] "TMONSPEND"    "OMONSPEND"    "SMONSPEND"    "PREVPD"       "GMP"
[31] "PROMOS"       "DAYS"         "FREDAYS"      "MARKDOWN"     "CLASSES"
[36] "COUPONS"      "STYLES"       "STORES"       "STORELOY"     "VALPHON"
[41] "WEB"          "MAILED"       "RESPONDED"    "RESPONSERATE" "HI"
[46] "LTFREDAY"     "CLUSTYPE"     "PERCRET"      "RESP"
```

```
> val=validation_data[,-1]
> val=val[,-1]
> names(val)
 [1] "REC"          "FRE"          "MON"          "CC_CARD"      "AVRG"
 [6] "PC_CALC20"    "PSWEATERS"    "PKNIT_TOPS"   "PKNIT_DRES"   "PBLOUSES"
[11] "PJACKETS"     "PCAR_PNTS"    "PCAS_PNTS"    "PSHIRTS"      "PDRESSES"
[16] "PSUITS"       "POUTERWEAR"   "PJEWELRY"     "PFASHION"     "PLEGWEAR"
[21] "PCOLLSPND"    "AMSPEND"      "PSSPEND"      "CCSPEND"      "AXSPEND"
[26] "TMONSPEND"    "OMONSPEND"    "SMONSPEND"    "PREVPD"       "GMP"
[31] "PROMOS"       "DAYS"         "FREDAYS"      "MARKDOWN"     "CLASSES"
[36] "COUPONS"      "STYLES"       "STORES"       "STORELOY"     "VALPHON"
[41] "WEB"          "MAILED"       "RESPONDED"    "RESPONSERATE" "HI"
[46] "LTFREDAY"     "CLUSTYPE"     "PERCRET"      "RESP"
```

(h) Skewness

Many of the numeric variables are right skewed. Unfortunately many standard data mining methods require that the variables are symmetric. But often is it possible to transform the original data to obtain symmetry.

Construct a histogram for the variable describing product uniformity, called *HI*. This variable takes large values for customers purchasing only from a few different classes of clothes (e.g. blouses, pants, ...) and small values for customers purchasing from many different cloth classes. This variable is clearly right skewed.

Try some different transformations (e.g. $t^{-2}, t^{-1}, t^{-1/2}, \ln(t), \sqrt{t}, t^1, t^2$) and use histograms to see if you get symmetry. Choose one of the transformations, apply this to *HI* in both the training and validation data and use these datasets in the following.

Solution:

```
> attach(tr)
> hi_trans=log(HI)
> tr=tr[,-45]
> tr=data.frame(tr,hi_trans)
```

```
> names(tr)
 [1] "REC"          "FRE"          "MON"          "CC_CARD"      "AVRG"
 [6] "PC_CALC20"    "PSWEATERS"    "PKNIT_TOPS"   "PKNIT_DRES"   "PBLOUSES"
[11] "PJACKETS"     "PCAR_PNTS"    "PCAS_PNTS"    "PSHIRTS"      "PDRESSES"
[16] "PSUITS"       "POUTERWEAR"   "PJEWELRY"     "PFASHION"     "PLEGWEAR"
[21] "PCOLLSPND"    "AMSPEND"      "PSSPEND"      "CCSPEND"      "AXSPEND"
[26] "TMONSPEND"    "OMONSPEND"    "SMONSPEND"    "PREVPD"       "GMP"
[31] "PROMOS"       "DAYS"         "FREDAYS"      "MARKDOWN"     "CLASSES"
[36] "COUPONS"      "STYLES"       "STORES"       "STORELOY"     "VALPHON"
[41] "WEB"          "MAILED"       "RESPONDED"    "RESPONSERATE" "LTFREDAY"
[46] "CLUSTYPE"     "PERCRET"      "RESP"         "hi_trans"
```

```
> attach(val)
> hi_trans=log(HI)
> val=val[,-45]
> val=data.frame(val,hi_trans)

> names(val)
 [1] "REC"          "FRE"          "MON"          "CC_CARD"      "AVRG"
 [6] "PC_CALC20"    "PSWEATERS"    "PKNIT_TOPS"   "PKNIT_DRES"   "PBLOUSES"
[11] "PJACKETS"     "PCAR_PNTS"    "PCAS_PNTS"    "PSHIRTS"      "PDRESSES"
[16] "PSUITS"       "POUTERWEAR"   "PJEWELRY"     "PFASHION"     "PLEGWEAR"
[21] "PCOLLSPND"    "AMSPEND"      "PSSPEND"      "CCSPEND"      "AXSPEND"
[26] "TMONSPEND"    "OMONSPEND"    "SMONSPEND"    "PREVPD"       "GMP"
[31] "PROMOS"       "DAYS"         "FREDAYS"      "MARKDOWN"     "CLASSES"
[36] "COUPONS"      "STYLES"       "STORES"       "STORELOY"     "VALPHON"
[41] "WEB"          "MAILED"       "RESPONDED"    "RESPONSERATE" "LTFREDAY"
[46] "CLUSTYPE"     "PERCRET"      "RESP"         "hi_trans"
```

(i) Derivation of new variables

Usually the variables "delivered" to us in the database are not the most appropriate variables for the analysis. So, often we have to derive new variables, based on some of the original ones. In our case study, let's look at the following three variables: *Amount spent by customer in the last month0* (*OMONSPEND*), *Amount spent by customer in the last three months* (*TMONSPEND*), *Amount spent by customer in the six last months* (*SMONSPEND*).

Would it be good to use all these three variables as input variables at the same time? Why/why not? Which variables would you derive? Do this! (Both for the training data and for the validation data.)

Solution:

```
> attach(tr)
> amount23=TMONSPEND-OMONSPEND
> amount456=SMONSPEND-TMONSPEND
> names(tr)
 [1] "REC"          "FRE"          "MON"          "CC_CARD"      "AVRG"
 [6] "PC_CALC20"    "PSWEATERS"    "PKNIT_TOPS"   "PKNIT_DRES"   "PBLOUSES"
[11] "PJACKETS"     "PCAR_PNTS"    "PCAS_PNTS"    "PSHIRTS"      "PDRESSES"
[16] "PSUITS"       "POUTERWEAR"   "PJEWELRY"     "PFASHION"     "PLEGWEAR"
[21] "PCOLLSPND"    "AMSPEND"      "PSSPEND"      "CCSPEND"      "AXSPEND"
[26] "TMONSPEND"    "OMONSPEND"    "SMONSPEND"    "PREVPD"       "GMP"
[31] "PROMOS"       "DAYS"         "FREDAYS"      "MARKDOWN"     "CLASSES"
[36] "COUPONS"      "STYLES"       "STORES"       "STORELOY"     "VALPHON"
[41] "WEB"          "MAILED"       "RESPONDED"    "RESPONSERATE" "LTFREDAY"
[46] "CLUSTYPE"     "PERCRET"      "RESP"         "hi_trans"


> tr=tr[,-26]
> names(tr)
 [1] "REC"          "FRE"          "MON"          "CC_CARD"      "AVRG"
 [6] "PC_CALC20"    "PSWEATERS"    "PKNIT_TOPS"   "PKNIT_DRES"   "PBLOUSES"
[11] "PJACKETS"     "PCAR_PNTS"    "PCAS_PNTS"    "PSHIRTS"      "PDRESSES"
[16] "PSUITS"       "POUTERWEAR"   "PJEWELRY"     "PFASHION"     "PLEGWEAR"
```

```
[21] "PCOLLSPND"    "AMSPEND"      "PSSPEND"      "CCSPEND"      "AXSPEND"
[26] "OMONSPEND"    "SMONSPEND"    "PREVPD"       "GMP"          "PROMOS"
[31] "DAYS"         "FREDAYS"      "MARKDOWN"     "CLASSES"      "COUPONS"
[36] "STYLES"       "STORES"       "STORELOY"     "VALPHON"      "WEB"
[41] "MAILED"       "RESPONDED"    "RESPONSERATE" "LTFREDAY"     "CLUSTYPE"
[46] "PERCRET"      "RESP"         "hi_trans"
> tr=tr[,-27]
> names(tr)
 [1] "REC"          "FRE"          "MON"          "CC_CARD"      "AVRG"
 [6] "PC_CALC20"    "PSWEATERS"    "PKNIT_TOPS"   "PKNIT_DRES"   "PBLOUSES"
[11] "PJACKETS"     "PCAR_PNTS"    "PCAS_PNTS"    "PSHIRTS"      "PDRESSES"
[16] "PSUITS"       "POUTERWEAR"   "PJEWELRY"     "PFASHION"     "PLEGWEAR"
[21] "PCOLLSPND"    "AMSPEND"      "PSSPEND"      "CCSPEND"      "AXSPEND"
[26] "OMONSPEND"    "PREVPD"       "GMP"          "PROMOS"       "DAYS"
[31] "FREDAYS"      "MARKDOWN"     "CLASSES"      "COUPONS"      "STYLES"
[36] "STORES"       "STORELOY"     "VALPHON"      "WEB"          "MAILED"
[41] "RESPONDED"    "RESPONSERATE" "LTFREDAY"     "CLUSTYPE"     "PERCRET"
[46] "RESP"         "hi_trans"

> tr=data.frame(tr,amount23,amount456)
> names(tr)
 [1] "REC"          "FRE"          "MON"          "CC_CARD"      "AVRG"
 [6] "PC_CALC20"    "PSWEATERS"    "PKNIT_TOPS"   "PKNIT_DRES"   "PBLOUSES"
[11] "PJACKETS"     "PCAR_PNTS"    "PCAS_PNTS"    "PSHIRTS"      "PDRESSES"
[16] "PSUITS"       "POUTERWEAR"   "PJEWELRY"     "PFASHION"     "PLEGWEAR"
[21] "PCOLLSPND"    "AMSPEND"      "PSSPEND"      "CCSPEND"      "AXSPEND"
[26] "OMONSPEND"    "PREVPD"       "GMP"          "PROMOS"       "DAYS"
[31] "FREDAYS"      "MARKDOWN"     "CLASSES"      "COUPONS"      "STYLES"
[36] "STORES"       "STORELOY"     "VALPHON"      "WEB"          "MAILED"
[41] "RESPONDED"    "RESPONSERATE" "LTFREDAY"     "CLUSTYPE"     "PERCRET"
[46] "RESP"         "hi_trans"     "amount23"     "amount456"

> attach(val)
> amount23=TMONSPEND-OMONSPEND
> amount456=SMONSPEND-TMONSPEND
> val=val[,-26]
> val=val[,-27]
> val=data.frame(val,amount23,amount456)
> names(val)
 [1] "REC"          "FRE"          "MON"          "CC_CARD"      "AVRG"
 [6] "PC_CALC20"    "PSWEATERS"    "PKNIT_TOPS"   "PKNIT_DRES"   "PBLOUSES"
[11] "PJACKETS"     "PCAR_PNTS"    "PCAS_PNTS"    "PSHIRTS"      "PDRESSES"
[16] "PSUITS"       "POUTERWEAR"   "PJEWELRY"     "PFASHION"     "PLEGWEAR"
[21] "PCOLLSPND"    "AMSPEND"      "PSSPEND"      "CCSPEND"      "AXSPEND"
[26] "OMONSPEND"    "PREVPD"       "GMP"          "PROMOS"       "DAYS"
[31] "FREDAYS"      "MARKDOWN"     "CLASSES"      "COUPONS"      "STYLES"
[36] "STORES"       "STORELOY"     "VALPHON"      "WEB"          "MAILED"
[41] "RESPONDED"    "RESPONSERATE" "LTFREDAY"     "CLUSTYPE"     "PERCRET"
[46] "RESP"         "hi_trans"     "amount23"     "amount456"
```

(j) So, now this exercise is finished! Note that all changes to variables (transformations, derivations of new variables, etc.) should be done for both the training data and the validation data. These datasets are going to be used in later assignments!!!

To save your data you can use the command *write.table* (if you have your data as a dataframe).

Solution:

```
> write.table(tr,"tr_clothing_prepared.txt",row.names=FALSE)
> write.table(val,"val_clothing_prepared.txt",row.names=FALSE)
```

Description of the variables in the dataset Clothing_Store:

- Customer ID (encrypted)
- Zip code
- Number of purchase visits
- Total net sales
- Average amount spent per visit
- Amount spent at each of four different franchises (four variables)
- Amount spent in the past month, the past three months, and the past six months
- Amount spent the same period last year
- Gross margin percentage
- Number of marketing promotions on file
- Number of days the customer has been on file
- Number of days between purchases
- Markdown percentage on customer purchases
- Number of different product classes purchased
- Number of coupons used by the customer
- Total number of individual items purchased by the customer
- Number of stores the customer shopped at
- Number of promotions mailed in the past year
- Number of promotions responded to in the last year
- Promotion response rate for the past year
- Product uniformity (low score = diverse spending patterns)
- Lifetime average time between visits
- Microvision lifestyle cluster type
- Percent of returns
- Credit card user (flag)
- Valid phone number on file (flag)
- Web shopper (flag)
- 15 variables providing the percentages spent by the customer on specific classes of clothing, including sweaters, knit tops, knit dresses, blouses, jackets, career pants, casual pants, shirts, dresses, suits, outerwear, jewelry, fashion, legwear, collectibles line and one variable showing the brand
- Response to promotion (target variable)