Entendimiento y Preparación de Datos

EP7144 - Técnicas de Minería de Datos

Mg. Enver Gerald Tarazona Vargas

enver.tarazona@pucp.edu.pe

Escuela de Post-Grado

Universidad Nacional Agraria La Molina (UNALM)

Resumen I

Resumen II

- Análisis de Componentes Principales (PCA)
- Definiciones

¿Por qué preparar los datos? I

- Algún tipo de preparación de datos siempre es necesario para la mayoría de herramientas de minería de datos.
- El propósito de la preparación es transformar los conjuntos de datos de tal forma que la información que contienen esté mejor expuesta para la herramienta de minería de datos que se utilizará.
- Los errores de predicción deberían ser menores (o en el peor caso similares) luego de la preparación de datos, en comparación con la data inicial.
- La preparación de datos también prepara al analista para producir mejores modelos y de manera más rápida.
- Tener buenos datos es un prerrequisito para producir modelos efectivos de cualquier tipo.
- Los datos necesitan ser formateados para cada software en particular.

¿Por qué preparar los datos? II

- Los datos necesitan ser adecuados para un método en particular
- Los datos en la vida real están "sucios":
  - **incompletos**: Falta de valores en los atributos, carecen de algunos atributos de interés, sólo contienen datos agregados:
    ej., ocupación = " "
  - **anómalos**: errores y outliers
    ej., Salario = "-10"
  - **inconsistentes**: contienen discrepancias en códigos y nombres
    ej., Edad = "42" , Cumpleaños = "03/07/1997"
    ej., Rating previo "1,2,3," Rating actual "A, B, C"
    ej., Discrepancia con registros duplicados

¿Por qué los datos están sucios?

- Los datos incompletos pueden venir de
  - Datos "No aplicables.ªl momento de ser colectados.
  - Diferentes consideraciones de tiempo cuando fueron recolectados y cuando son analizados
  - Problemas Humanos/hardware/software
- Datos anómalos (valores incorrectos) pueden venir de
  - Instrumentos de recolección de datos defectuoso
  - Errores humanos o de computadora en la entrada de los datos
  - Errores en la transmisión de datos
- Datos inconsistentes pueden venir de
  - Diferentes fuentes de datos
  - Violación de dependencias funcionales (ej., modificación en algunos datos relacionados)
- Registros duplicados también necesitan ser limpiados

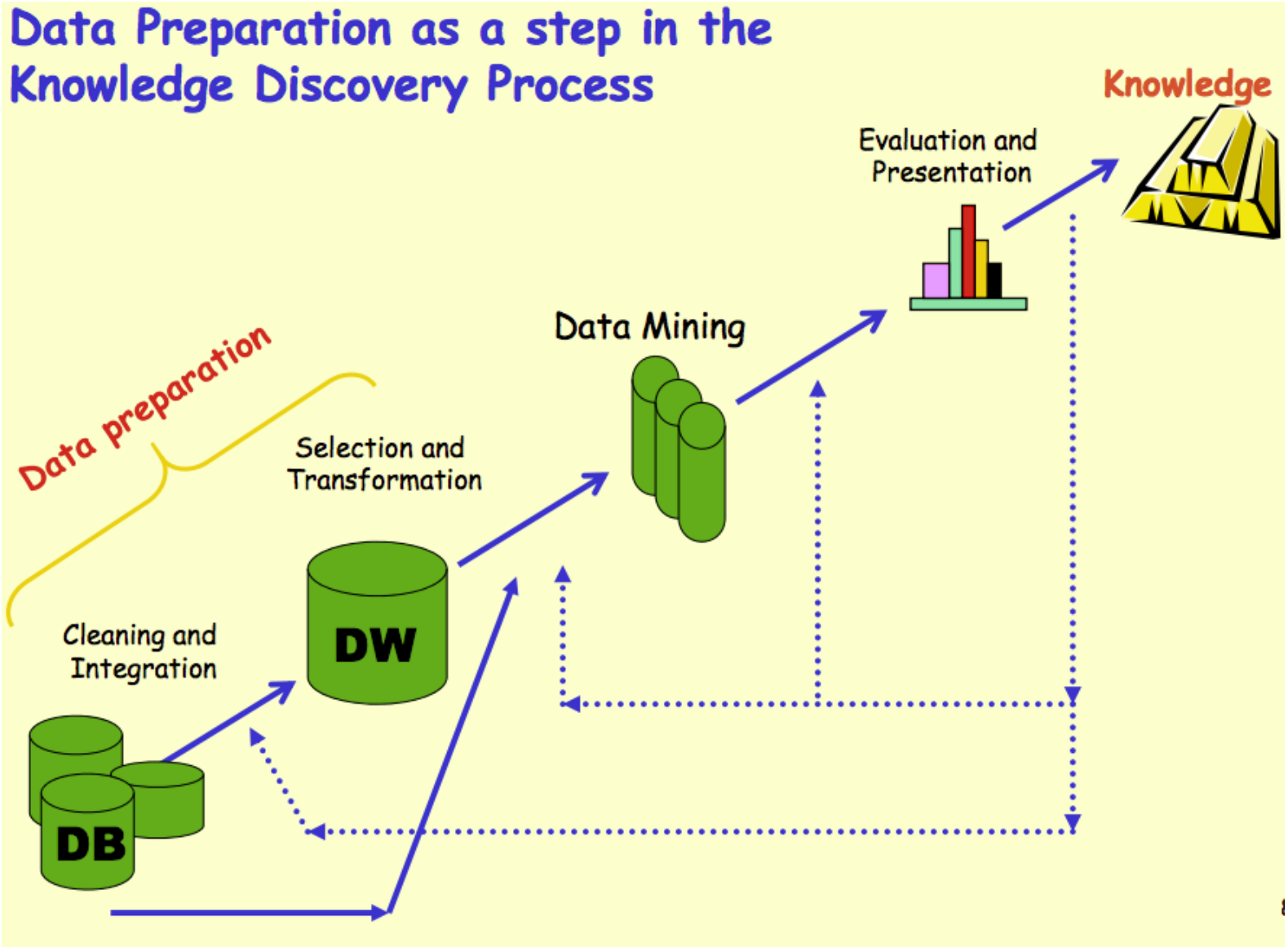
¿Por qué los datos están sucios?

- ¡No hay calidad en los datos, no hay calidad en los resultados!
  - Decisiones de calidad deben de basarse en datos de calidad
  - ej., datos duplicados o perdidos pueden producir estadísticas engañosas o incorrectas.
  - Data warehouse necesita una integración consistente de datos de calidad
  - La selección de datos, la limpieza y la transformación comprende la mayor parte del trabajo de construir una data warehouse

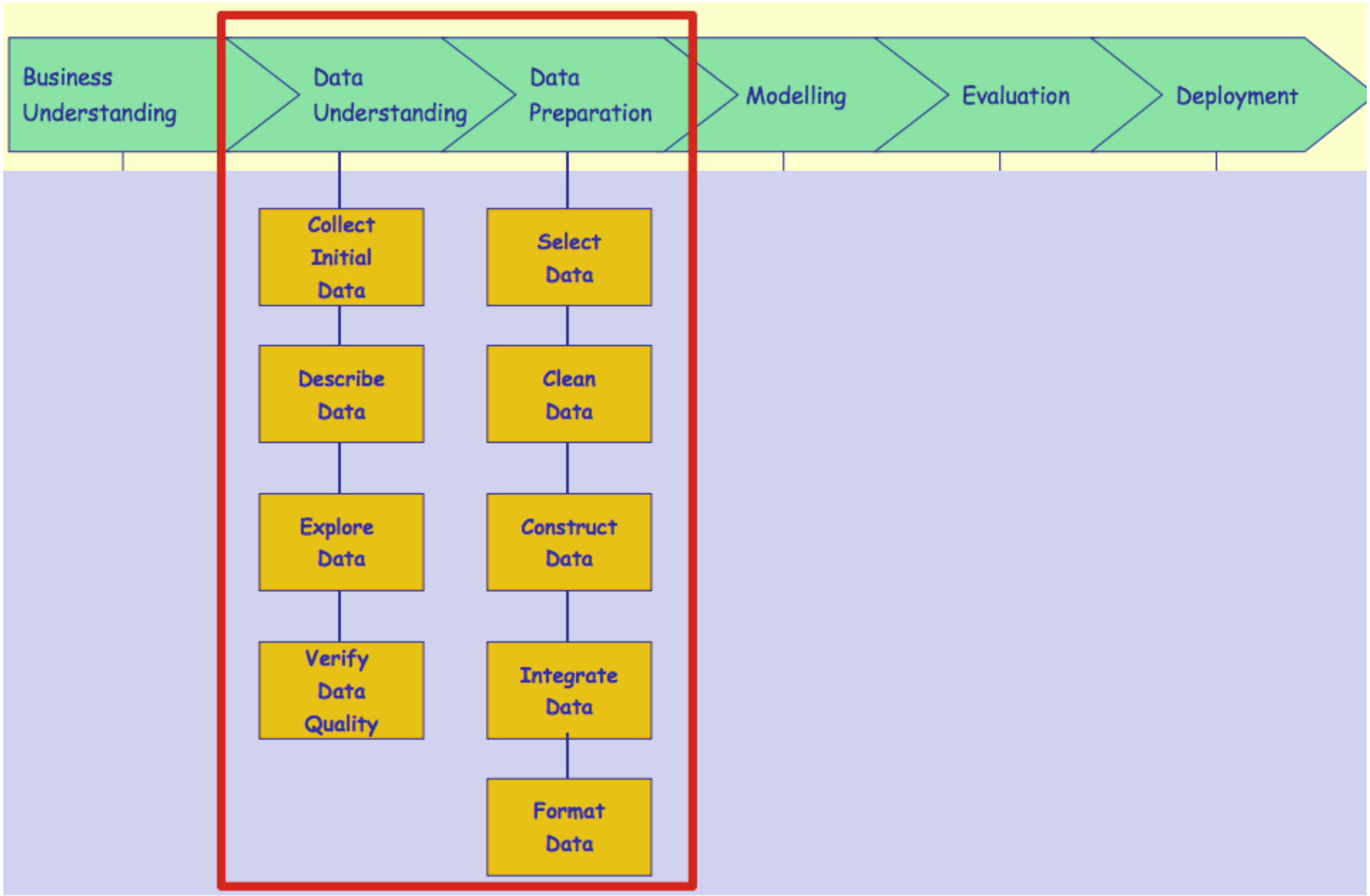
Principales Tareas en la Preparación de Datos I

- Limpieza de datos
  - Completa valores faltantes, suavizar datos ruidosos, identificar o remover outliers y resolver inconsistencias.
- Integración de datos
  - Integración de múltiples bases de datos, cubos de datos, archivos.
- Transformación de datos
  - Normalización y agregación (totalización)
- Reducción de datos
  - Se obtiene una representación más reducida en volumen pero que produce los mismos o similares resultados analíticos.
- Discretización de datos
  - Parte de la reducción de datos pero con particular importancia, especialmente para datos numéricos

La preparación de datos como una fase del KDD



CRISP-DM: Fases y Tareas



CRISP-DM: Preparación de Datos I

- **Selección de datos**
  - Reconsiderar el criterio de selección de los datos.
  - Decidir el conjunto de datos que será usado.
  - Recolectar data adicional que sea apropiada (interna o externa).
  - Considerar el uso de técnicas de muestreo.
  - Explicar por qué ciertos datos son incluidos o excluidos.
- **Limpieza de datos**
  - Corregir, remover o ignorar ruido.
  - Decidir como proceder con valores especiales y su significado (99 para estado civil)
  - Niveles de totalización, valores perdidos, etc.
  - outliers?
- **Construccion de Datos**
  - Derivación de atributos.
  - Conocimiento previo.

CRISP-DM: Preparación de Datos II

- ¿Los datos perdidos pueden imputarse o reconstruirse?

- **Formato de Datos**
  - Reordenamiento de los atributos (Algunas herramientas tienen requerimientos en relación al orden de los atributos, ej. el primer campo debe ser un identificar único para cada registro o el último campo debe ser la variable respuesta a ser predicha).
  - Reordenamiento de registros (Puede ser que la herramienta de modelamiento requiera que los registros estén ordenados de acuerdo al valor de la variable respuesta)
  - Reformateo de valores (Cambios puramente sintácticos para satisfacer los requerimeintos de una herramienta específica de modeloamiento, ej. NA para datos perdidos en vez de 99, remover caracteres ilegales, letras mayúsculas o minúsculas, etc.)

Datos Perdidos I

- Los datos no siempre están disponibles.
- La falta de valores se puede deber a:
  - Mal funcionamiento de equipos.
  - Inconsistencia con otros datos registrados y por lo tanto eliminados.
  - Datos no ingresados debido a equivocaciones.
  - Algunos datos pudieron no considerarse importantes al momento de ingresar datos.
  - No se registró historial o cambios en los datos.
- Puede ser necesario estimar estos valores faltantes.
- Los valores faltantes son un problema común en análisis estadístico.
- Se ha propuesto muchos métodos para el tratamiento de valores faltantes. Muchos de estos métodos fueron desarrollados para el tratamiento de valores faltantes en encuestas por muestreo.
- Bello (1995), tratamiento de valores faltantes in regression

Datos Perdidos II

- Troyanskaya et al (2001), tratamiento de datos faltantes en clasificacion no supervisada.
- Estudios relacionados con clasificación supervisada:
  - Chan and Dunn (1972) - Imputation en LDA para problemas con dos clases.
  - Dixon (1975) - Imputacion k-nn para lidiar con valores faltantes en clasificacion supervisada.
  - Tresp (1995)- el problema de valores faltantes en aprendizaje supervisado usando redes neurales.

Datos Perdidos: Impacto

Impacto de los valores faltantes:

- 1 % datos faltantes – trivial.
- 1-5 % – manejable
- 5-15 % – requiere métodos sofisticados
- Más del 15 % – interpretación perjudicial

Mecanismos de datos perdidos I

- **Valores faltantes completamente al azar (MCAR)**:La probabilidad que una instancia tenga un valor faltante para un atributo es la misma para todas las instancias. Es decir,esta probabilidad no depende ni de los valores observados ni de los valores faltantes. La mayoria de los valores faltantes no son MCAR.
  - Por ejemplo en el caso de tener en un estudio las variables ingreso y edad. Estaremos bajo un modelo MCAR cuando al analizar conjuntamente edad e ingresos, suponemos que la falta de respuesta en el campo ingresos es independiente del verdadero valor de los ingresos y la edad.

  Este mecanismo es mas adecuado para datos a ser usados en clasificacion no supervisada.
- **Valores faltantes al azar (MAR)**: La probabilidad que una instancia tenga un valor faltante en un atributo depende de los valores observados, como por ejemplo la clase a la cual pertenece la instancia, pero no depende de los valores faltantes. Este mecanismo es mas adecuado para datos usados en clasificacion supervisada.

Mecanismos de datos perdidos II

- En el ejemplo anterior si suponemos que los ingresos son independientes de los ingresos del miembro del hogar pero puede depender de la edad estaremos bajo un modelo MAR.

Este mecanismo es mas adecuado para datos usados en clasificacion supervisada.

- **Valores faltantes no al azar o no ignorables (NMAR)**: La probabilidad de que una instancia tenga un valor faltante en un atributo depende de los valores faltantes en el conjunto de datos. Ocurre cuando las personas entrevistadas no quieren revelar algo muy personal acerca de ellas. El patron de valores faltantes no es aleatorio. Este tipo de valores faltantes es el mas dificil de tratar y es el que ocurre más frecuentemente.
  - En el ejemplo anterior, se obtiene que la función respuesta de la variable ingresos depende del propio valor de la variable ingresos, además de poder depender de otros factores.

Consideraciones prácticas I

- Para conjuntos de datos con un bajo porcentaje de valores faltantes el mecanismo se puede considerar MCAR.
- Para conjuntos de datos con un alto porcentaje de valores faltantes el mecanismo se puede considerar NMAR.
- En muchas aplicaciones lo prudente será considerar distintos modelos plausibles para el mecanismo de no respuesta y realizar un análisis de sensibilidad de las estimaciones.

Datos faltantes: Ejemplo

- Ejemplo: conjunto de datos `census`.
- Este conjunto de datos proviene de la librería `dprep` que contiene funciones para el pre-procesamiento de datos. Esta librería fue desarrollada por el Prof. Edgar Acuña de la Universidad de Puerto Rico-Mayaguez.
- Disponible en:
  `http://ftp.ics.uci.edu/pub/machine-learning-databases/`
- Donantes: Ronny Kohavi y Barry Becker (1996).

Datos faltantes: Ejemplo

- A data frame with 32561 observations on the following 14 variables.
  - V1 age: continuous
  - V2 workclass
  - V3 fnlwgt: continuous
  - V4 education
  - V5 marital-status
  - V6 occupation
  - V7 relationship
  - V8 race
  - V9 sex
  - V10 capital-gain: continuous
  - V11 capital-loss: continuous
  - V12 hours-per-week: continuous
  - V13 native-country
  - V14 class: $> 50K$, $<= 50K$

Datos faltantes: Ejemplo

- Ejemplo:

```
#Para ver que columnas tienen valores perdidos
which(colSums(is.na(censusn))!=0)

#Para ver que filas tienen valores perdidos
rmiss=which(rowSums(is.na(censusn))!=0,arr.ind=T)

#Para ver el porcentaje de filas con valores perdidos
length(rmiss)*100/dim(censusn)[1]

#Para ver el porcentaje de valores perdidos en las columnas
colmiss=c(2,6,13)
per.miss.col=100*colSums(is.na(censusn[,colmiss]))/dim(censusn)[1]
per.miss.col
```

Datos faltantes: Ejemplo

- Podemos utilizar la librería VIM

```
library(VIM)
a=aggr(censusn,numbers=T)
a
summary(a)
```

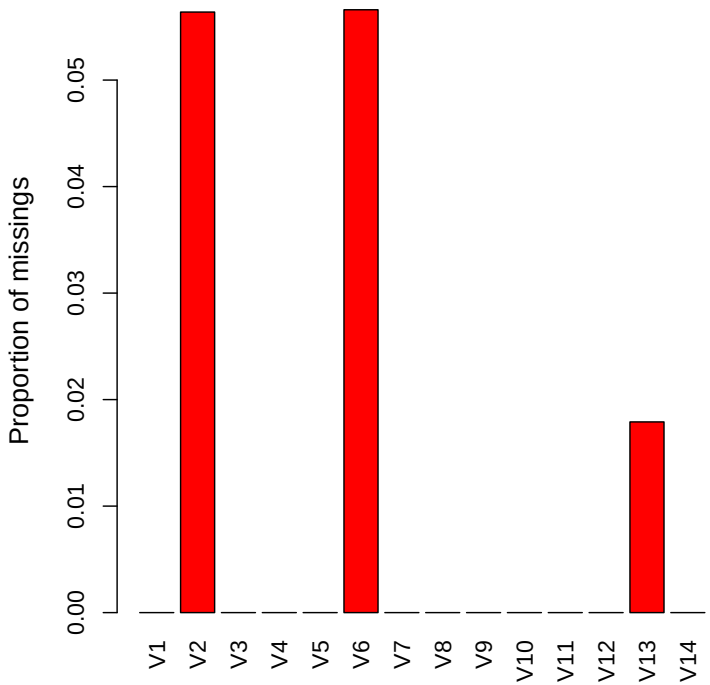
Datos faltantes: Ejemplo

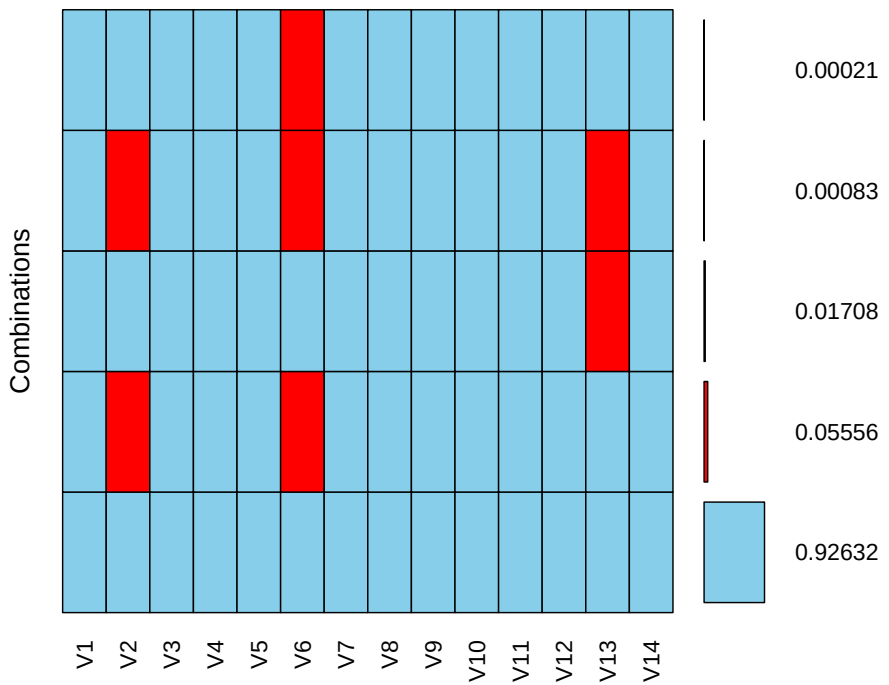
```
> summary(a)
 Missings per variable:
 Variable Count
       V1     0
       V2  1836
       V3     0
       V4     0
       V5     0
       V6  1843
       V7     0
       V8     0
       V9     0
      V10     0
      V11     0
      V12     0
      V13   583
      V14     0
```

Datos faltantes: Ejemplo

```
Missings in combinations of variables:
                                  Combinations Count     Percent
 0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0 30162 92.63229016
 0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:1:0   556  1.70756426
 0:0:0:0:0:1:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0     7  0.02149811
 0:1:0:0:0:1:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0  1809  5.55572618
 0:1:0:0:0:1:0:0:0:0:0:0:0:0:0:0:0:0:0:0:1:0    27  0.08292129
```

Datos faltantes: Ejemplo

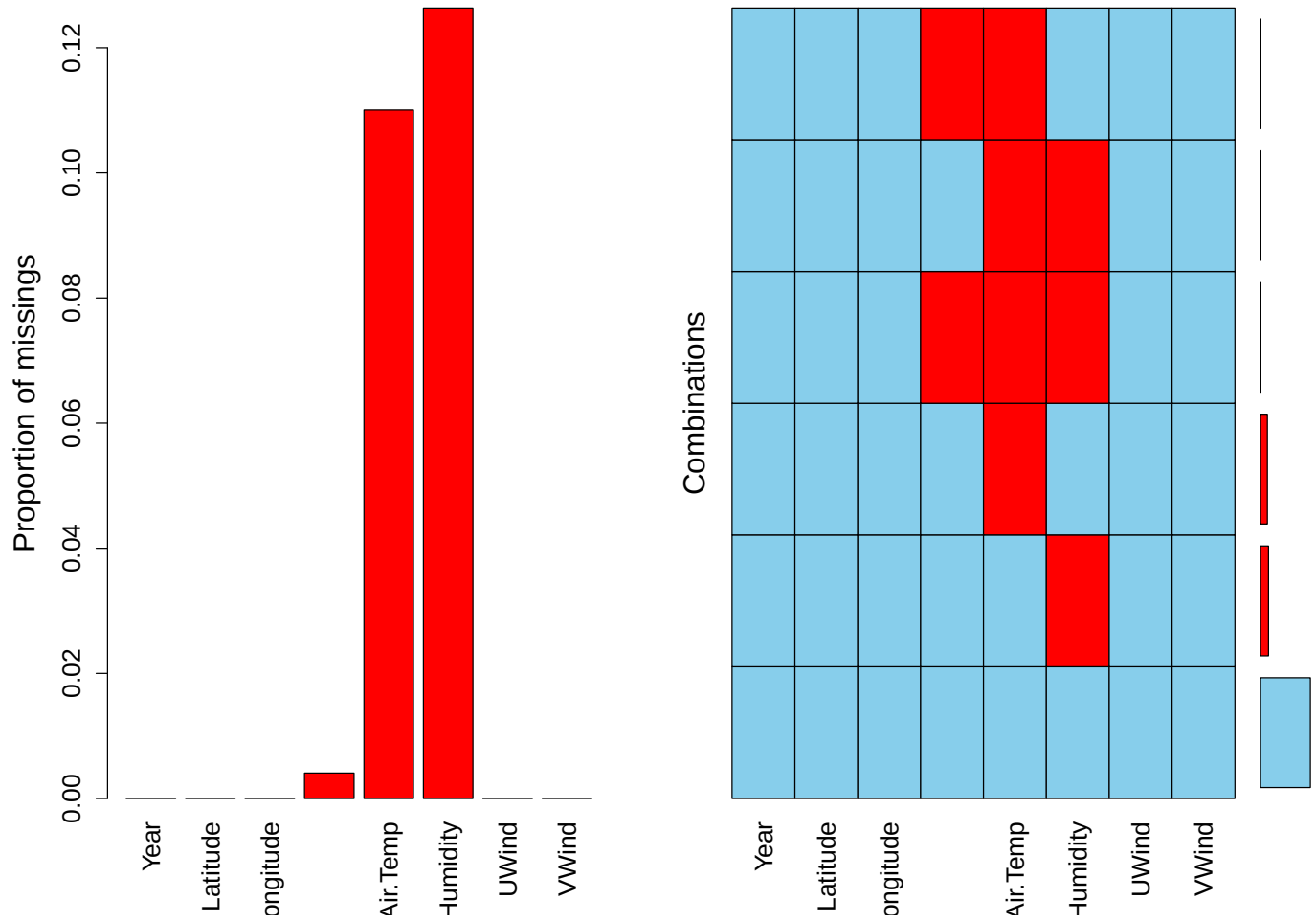




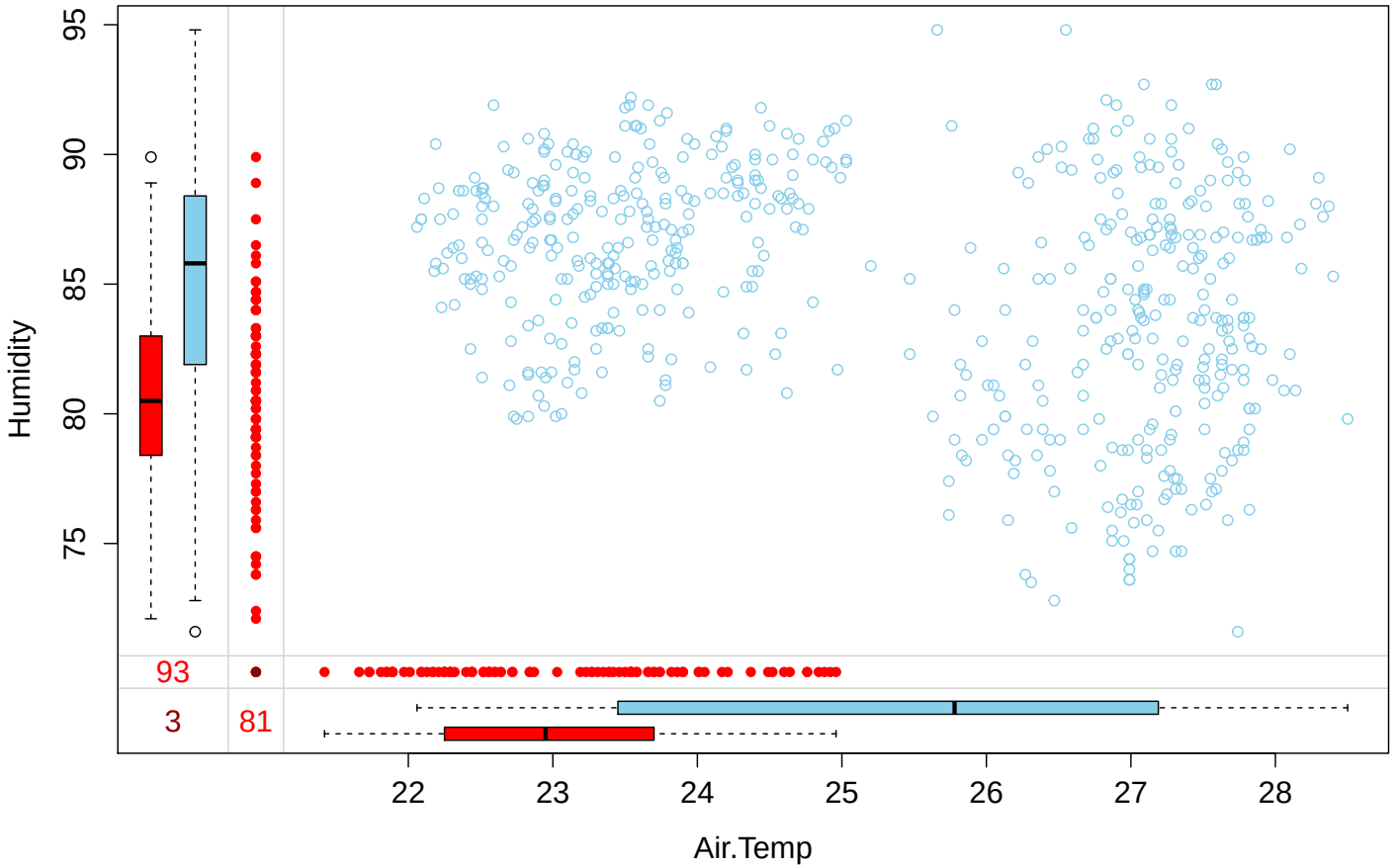
Datos faltantes: Ejemplo

```
#Ejemplo 2
data(tao)
b<-aggr(tao)
b
marginplot(tao[,c("Air.Temp", "Humidity")])
```

Datos faltantes: Ejemplo



Datos faltantes: Ejemplo

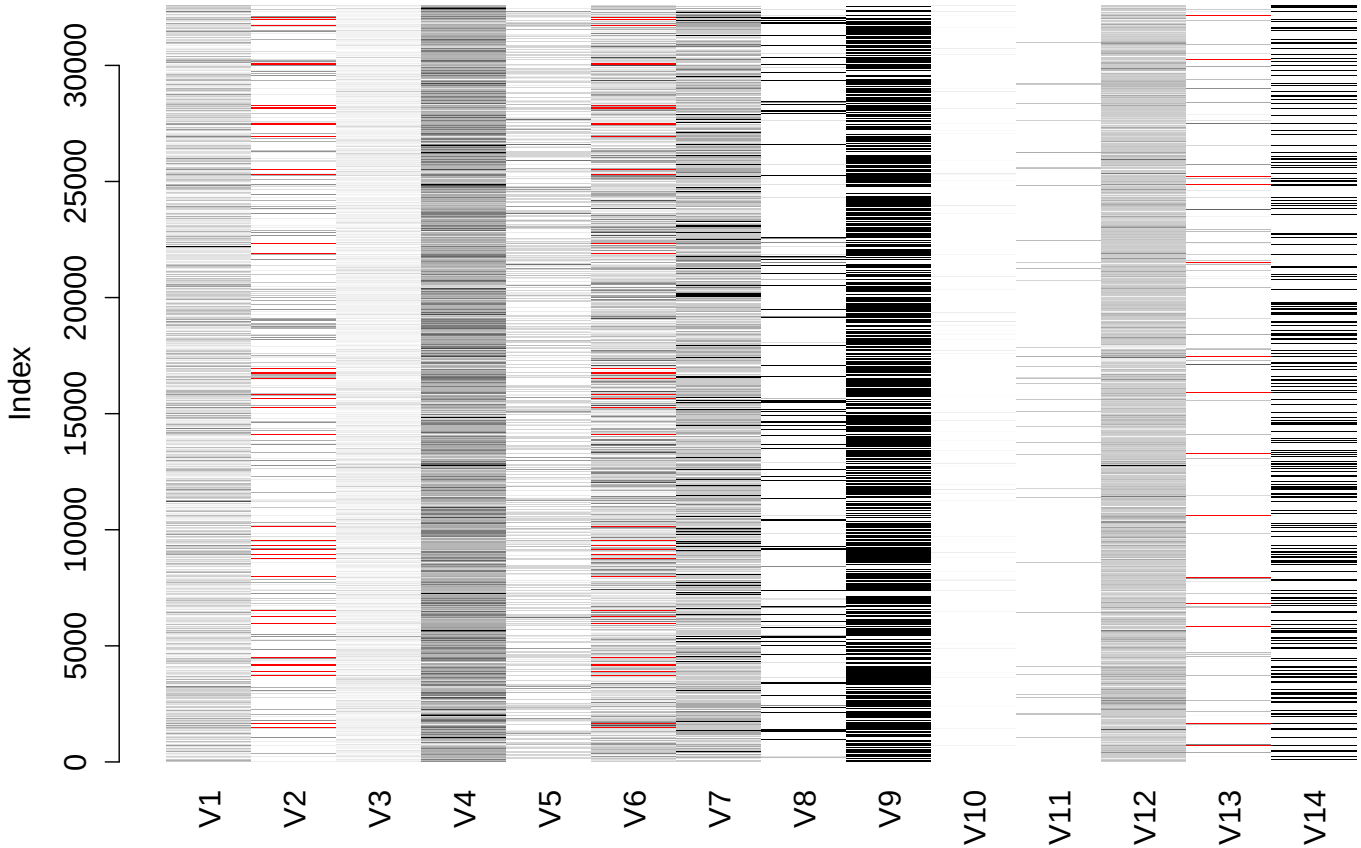


Datos faltantes

- Otra forma de visualizar valores faltantes es mediante un gráfico de matriz, en el cual la celda de cada matriz es visualizada como un rectángulo. Los datos continuos serán presentados por una escala de grises y los valores missing por el color rojo.

```
matrixplot(censusn)
```

Datos faltantes: Ejemplo



Tratamiento de la no respuesta I

- **Eliminar**: Es la opción mas sencilla y consiste en eliminar las observaciones o variables que tengan los datos perdidos. Solamente debe realizarse si es poco el porcentaje de observaciones a eliminar y si es posible asumir que los valores faltantes provienen de un proceso MCAR.
- **Reemplazar (imputar)**: Reemplazar el valor perdido con un valor conocidos. Variedad de métodos, desde opciones sencillas (reemplazar por la media o mediana) hastas otras más complejas (modelos de regresión).
- **Mantener**: No realizar imputación. A veces es factible analizar la información por separado. Por ejemplo, en algunas situaciones los procedimientos de Máxima Verosimilitud que usan variantes del algoritmo EM (Expectation-Maximization) pueden manejar la estimación de parámetros en presencia de valores faltantes.

Datos faltantes

- Eliminación de casos.
- Utilizaremos la función `na.omit`.

```
census.cl=na.omit(censusn)
```

Datos faltantes

- Imputación: Los valores faltantes son reemplazados con valores estimados basados en la información disponible.
  - Imputación por la media
  - Imputación por la mediana
  - Imputación por la moda

Datos faltantes

- La librería `DMwR` tiene la función `centralImputation` que reemplaza los valores faltantes de la siguiente manera:
  - Si la variable es numérica (`numeric` o `integer` en R) reemplaza los valores faltantes con la mediana.
  - Si la variable es categórica (`factor` en R) reemplaza los valores faltantes con la moda.

Datos faltantes

- La librería `VIM` tiene la función `initialise` que reemplaza los valores faltantes de la siguiente manera:
  - Si la variable es numérica continua (`numeric` en R) reemplaza los valores faltantes con la media.
  - Si la variable es numérica discreta (`integer` en R) reemplaza los valores faltantes con la mediana.
  - Si la variable es categórica (`factor` en R) reemplaza los valores faltantes con la moda.

Datos faltantes

```
census<-censusn

for(h in c(2,4,5,6,7,8,9,13,14)){
census[,h]<-as.factor(census[,h])
}

library(DMwR)
census.c<-centralImputation(census)
census.d<-initialise(census,method="median")
```

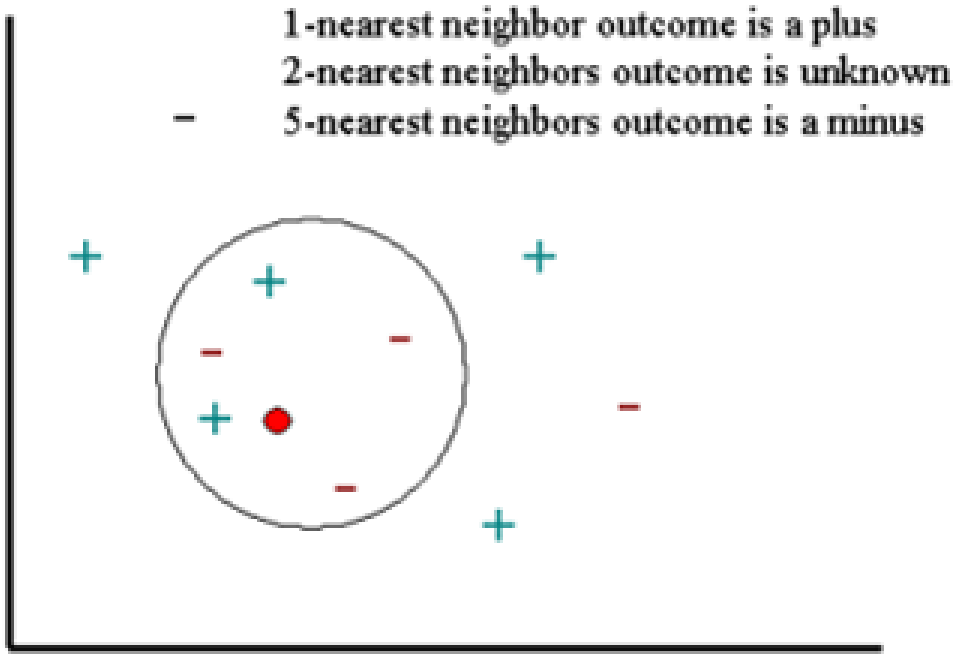
Datos faltantes

K-vecinos más cercanos

- El método consiste en que para cada valor faltante se encuentran las k-observaciones o instancias que están más cercanas considerando las otras variables.
- Luego se reemplaza el valor faltante de la siguiente manera:
  - Si la variable es categórica se reemplaza por la moda de las k-observaciones más cercanas.
  - Si la variable es numérica se reemplaza por la media de las k-observaciones más cercanas.

Datos faltantes

- K-vecinos mas cercanos.



Ejemplo de Imputacion K-nn

X1	X2	X3
3	6	7
4	5	?
6	?	4

La imputacion usando k=1
da la siguiente matriz completa

Datos faltantes

- La librería `DMwR` tiene la función `knnImputation` que reemplaza los valores faltantes mediante el método de k-valores más cercanos:

```
knnImputation(data, k = 10, scale = T,
meth = "weighAvg", distData = NULL)
```

  - data: conjunto de datos
  - k: número de vecinos más cercanos.
  - scale: indica si para calcular las distancias primero se estandarizan las variables.
  - meth: método para reeemplazar el valor faltante para variables numéricas. Opciones: 'median' (mediana) or 'weighAvg' (media ponderada por la distancia). En variables categóricas se usa la moda

Datos faltantes

- K-nn

```
census.k<-knnImputation(census)
```

Datos faltantes

Utilizando Modelos de Regresión

- El método consiste en estimar un modelo de regresión en función a las otras variables.
- Luego se reemplaza el valor faltante utilizando el modelo de regresión.

Datos faltantes

- La librería `VIM` tiene la función `irmi` que reemplaza los valores faltantes mediante modelos de regresión, el método es denominado de Iterative robust model-based imputation:

  ```
  irmi(data)
  ```

- `data`: conjunto de datos

  ```
  imputed.tao <- irmi(tao)
  summary(imputed.tao)
  ```

Efecto del tratamiento

- Para conjuntos de datos con una pequeña cantidad de valores faltantes se observa poca diferencia entre la eliminación de casos y otros metodos de imputación.
- Cuando se usa eliminación de casos la variabilidad del estimado del error de clasificación aumenta.
- Casi no hay diferencia entre usar imputacion por la media e imputacion por la mediana.
- El efecto de los valores faltantes depende de la forma que se distribuyen en la matriz de datos y en su localización con respecto a las variables mas importantes.
- El porcentaje de instancias con valores faltantes tiene mayor efecto en el proceso de clasificación que el porcentaje total de valores faltantes en la matriz de datos
- El tratamiento de los valores falantes en el procesos de clasificación depende del clasificador que esta siendo usado.

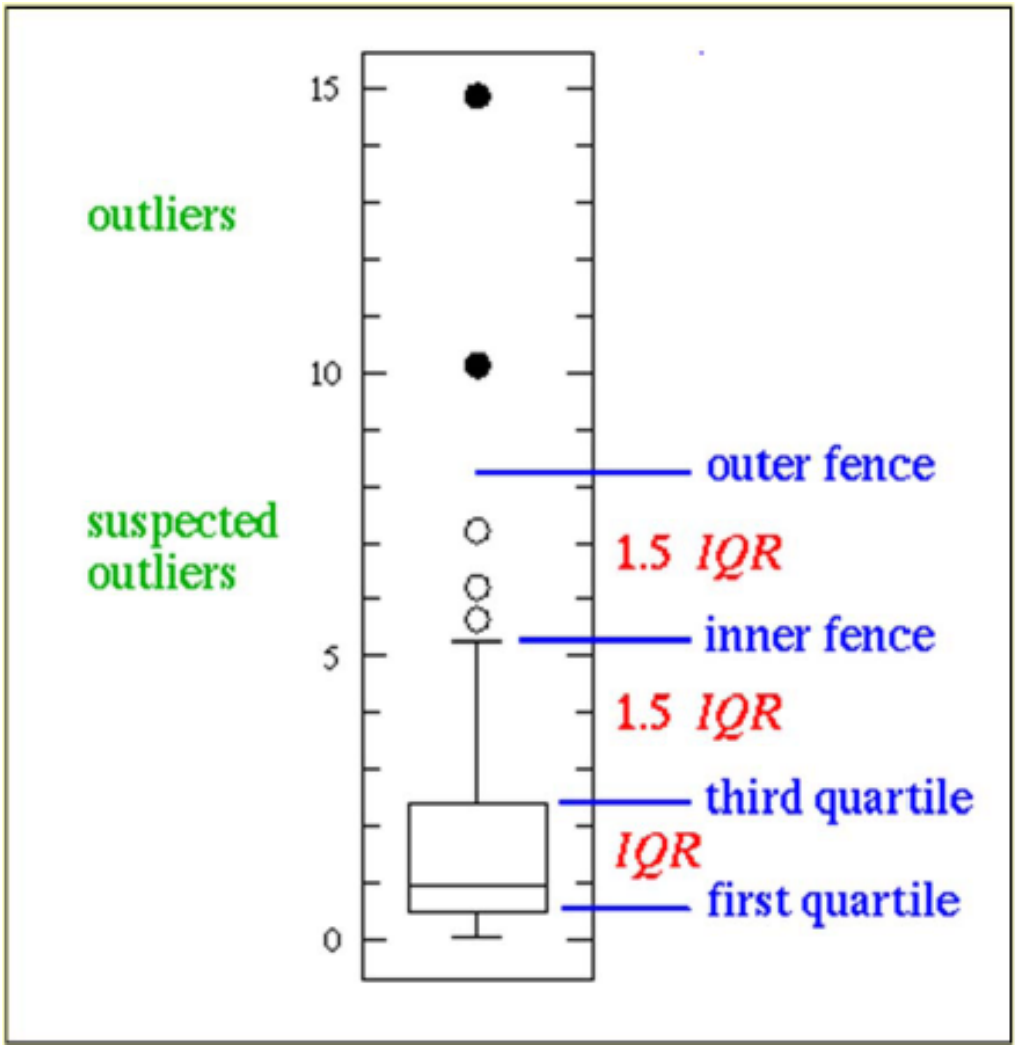
Valores Outlier

- Un "outlier" es una observación que se desvía tanto de las otras observaciones como para crear la sospecha de que fue generado por un mecanismo diferente.

Outliers Univariados

- Considerar outliers valores que $\frac{|x-\overline{x}|}{s} > k$
- donde k es 2 ó 3 si consideramos normalidad.
- Considerando el Boxplot (Tukey, 1977), se considera outlier a los valores que caen fuera de este intervalo. $(Q_1 - 3 \times IQR,\ Q_3 + 3 \times IQR)$

Outliers univariados



http://www.physics.csbsju.edu/stats/box2.html

Dataset: Bupa liver disease

1. Creador: BUPA Medical Research Ltd.
2. Información Relevante: Las primeras 5 variables son resultados de pruebas sanguíneas que se piensan pueden ser sensitivas (y posibles predictores) ante transtornos hepáticos producidos por un consumo excesivo de alcohol. Cada linea de la base de datos bupa.txt constituye un registro de un individuo de sexo masculino.
3. Número de instancias: 345
4. Información de atributos:
   - V1 volumen corpuscular
   - V2 fosfatosa alcalina
   - V3 alamine aminotransferase
   - V4 aspartate aminotransferase
   - V5 gamma-glutamyl transpeptidase
   - V6 número de bebidas alcohólicas
   - V7 1(hígado enfermo) 2(hígado sano)

Ejemplo

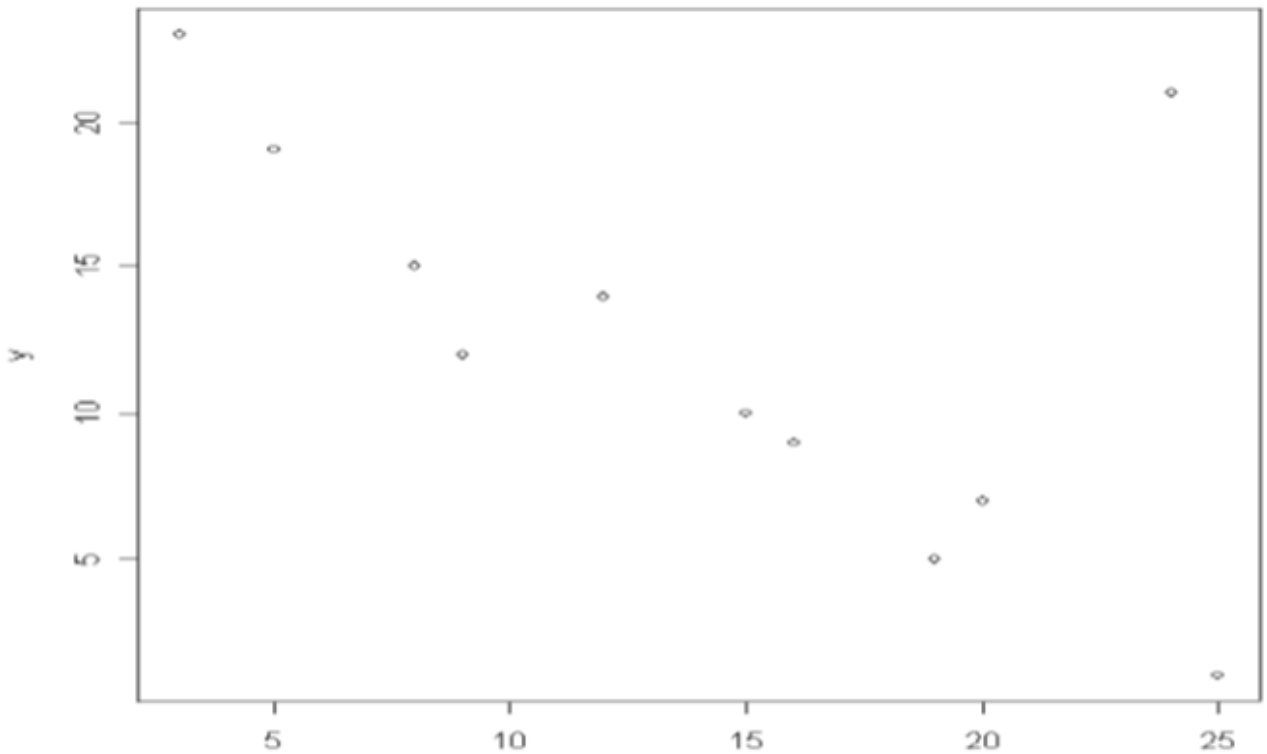
```
zbupa=cbind(scale(bupa[,-7]),bupa[,7])
zbupa1=zbupa[,1]
rownames(bupa[abs(zbupa1)>2,])

outliers=boxplot(bupa$V1,plot=F)$out
nout=as.character(outliers)
boxplot(bupa$V1,col="blue")
for(i in 1:length(outliers))
{
text(outliers[i],as.character(which(bupa$V1==outliers[i])),
cex=.8,pos=4)
}
```

Outliers Multivariados

- Consideremos un conjunto de datos D con p variables y n instancias. Supongamos que también conocemos las clases a las cuales pertenecen cada una de las instancias.
- El objetivo es detectar todas las instancias que parecen ser no usuales, estas serán los outliers multivariados.
- Uno podría pensar que los outliers multivariados pueden ser detectados basados en los outliers univariados en cada una de las variables, pero no es cierto. Una instancia puede tener valores que son outliers en varias variables, pero la instancia como todo podría no ser un outlier multivariado.

Outliers Multivariados



Un outlier bi-dimensional que no es outlier en cualquiera de sus projecciones.

Métodos para detectar Outliers Multivariados

- Métodos basados en estadística robusta
- Métodos basados en clustering,
- Métodos basados en distancia, y
- Métodos basados en densidad local.

Outliers Multivariados: Distancia de Mahalanobis I

- Sea x una observación de un conjunto de datos multivariado consistente de n observaciones y p variables.
- Sea $\bar{x}$ el centroide del conjunto de datos, el cual es un vector p-dimensional que tiene como componentes la media de cada variable.
- Sea $\tilde{x}$ la matriz del conjunto de datos original con columnas centradas por sus medias.
- Luego, la matriz $S = \frac{1}{n-1}\tilde{x}'\tilde{x}$ de orden $p \times p$ representa la matriz de covarianza de p variables.
- La versión multivariada de la ecuación anterior es

$$D^2(x, \bar{x}) = (x - \bar{x})' S^{-1} (x - \bar{x}) > k$$

donde D^2 es llamada la distancia de Mahalanobis cuadrada estimada desde x al centroide del conjunto de datos.

Outliers Multivariados: Distancia de Mahalanobis II

- Una observación con una distancia de Mahalanobis grande puede ser considerada como un outlier.
- Si se asume que los datos vienen de una distribución normal multivariada (p dimensiones):
  - Entonces la distancia de Mahalanobis cuadrada de las observaciones siguen una distribución Chi-cuadrado con p grados de libertad.
  - Es posible realizar una gráfica QQ de la distribución Chi-cuadrado para detectar a los outliers.
  - En R: `qqplot()`
- Consideraciones prácticas:
  - La distribución de chi-cuadrado sigue siendo razonablemente buena para la distancia de Mahalanobis estimada.
  - Si los datos no siguen una distribución normal multivariada, los puntos con una distancia de Mahalanobis grande son todavía potenciales outliers.

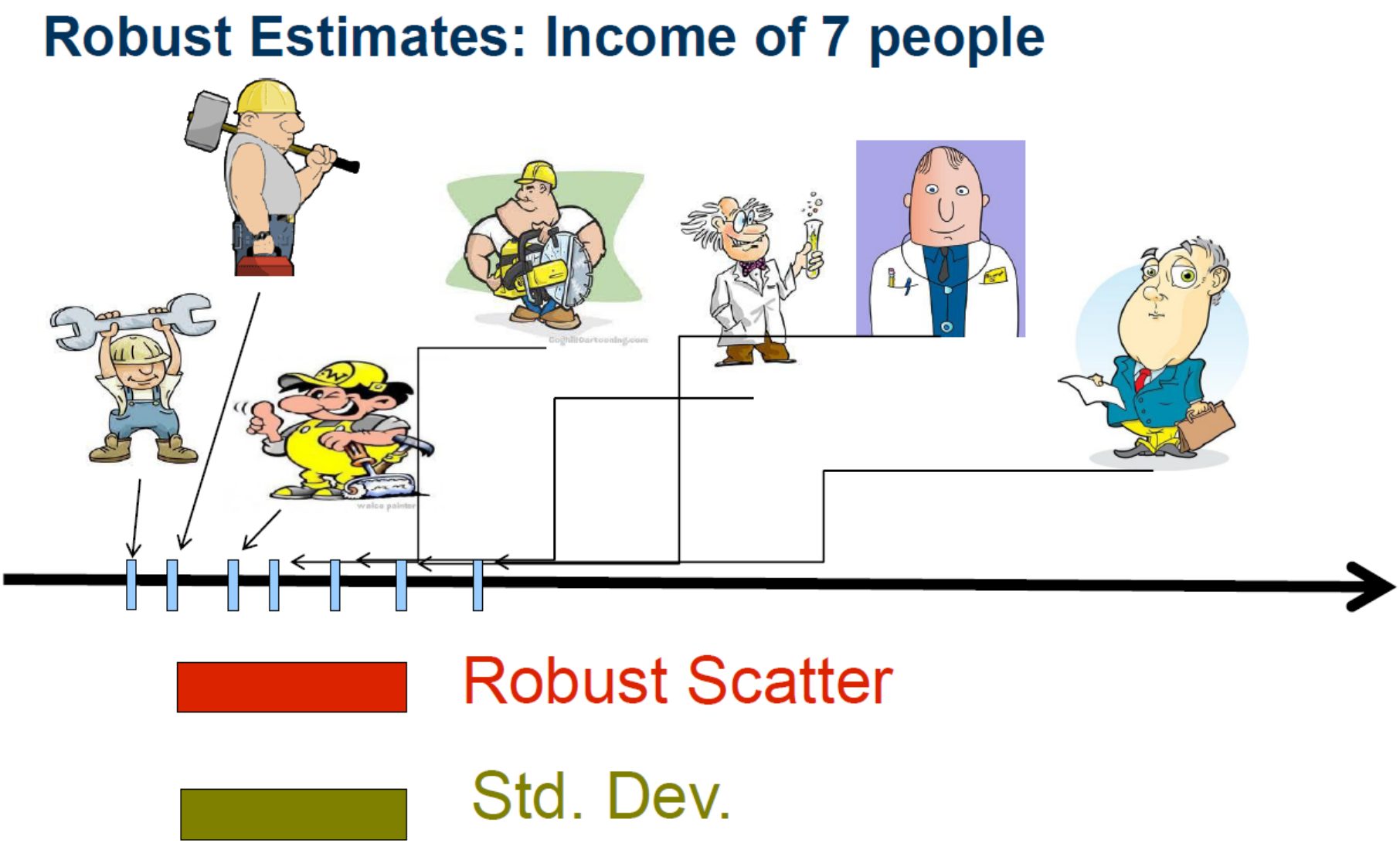
Outliers Multivariados: Estimadores robustos

- Efecto de enmascaramiento: Ocurre cuando después de eliminar un outlier, otra instancia se puede volver outlier.
- Efecto de cubrimiento: Ocurre cuando después de eliminar un outlier, otro outlier se vuelve una buena observación.

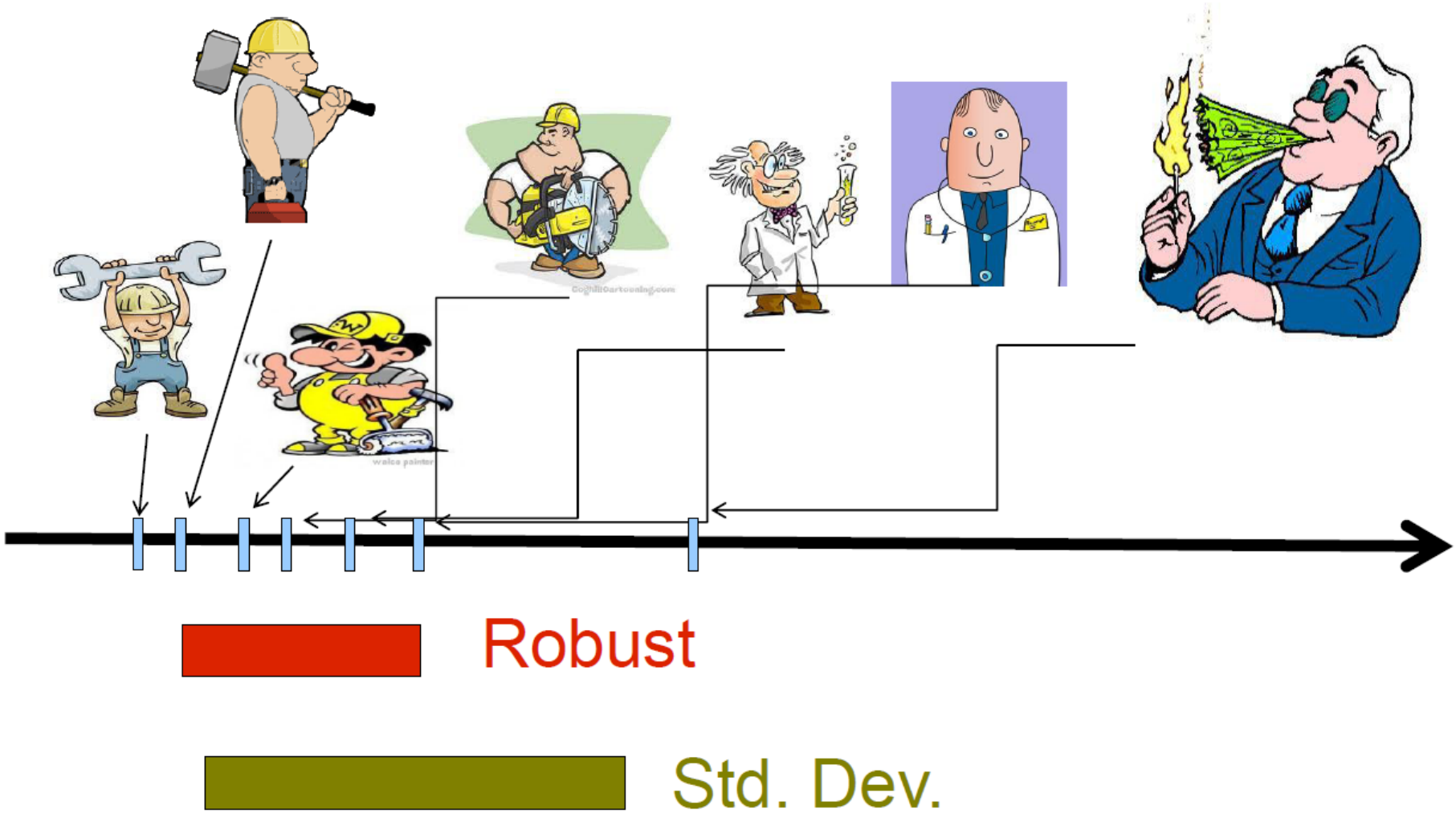
Outliers Multivariados: Estimadores robustos

- Para lidiar con estos efectos se recomienda usar un estimador robusto de la distancia de Mahalanobis. Hay dos propuestas: El estimador de elipsoide de volúmen mínimo (MVE) y el estimador de determinante de covarianza mínima (MCD).

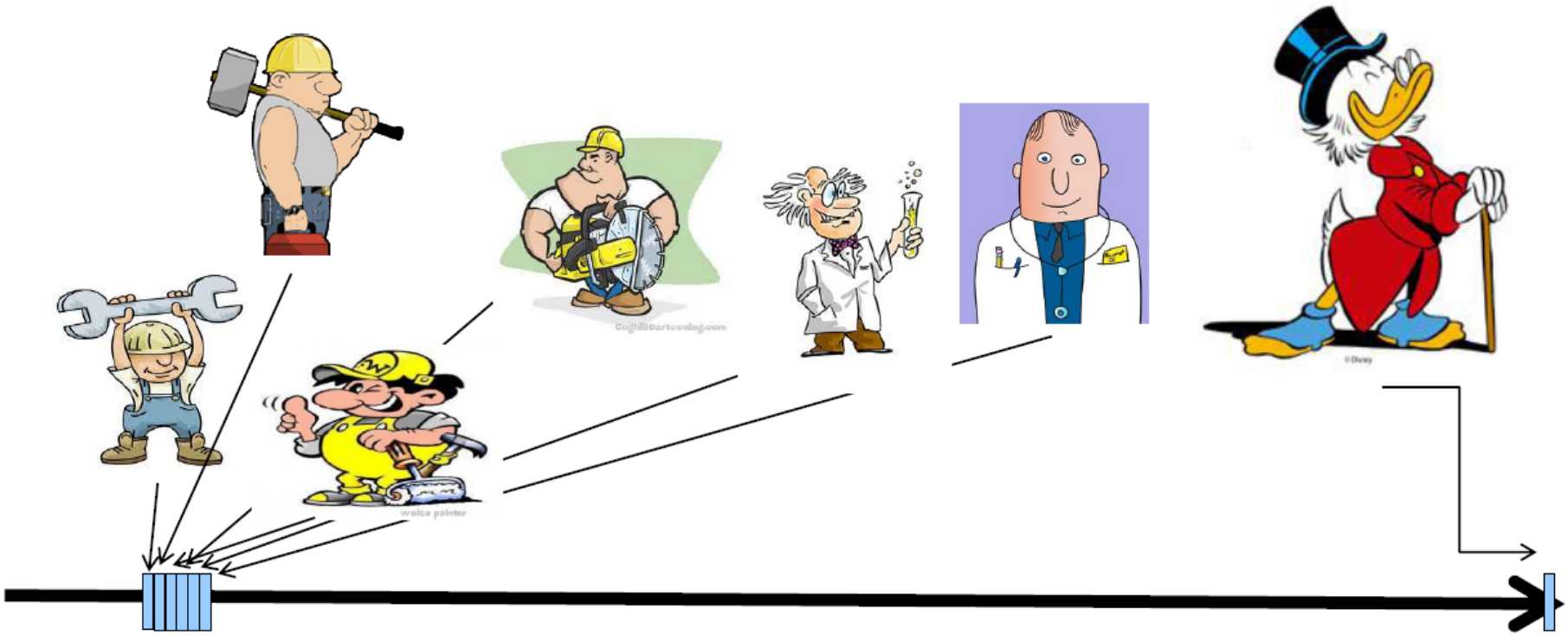
Outliers Multivariados: Estimadores robustos



Outliers Multivariados: Estimadores robustos



Outliers Multivariados: Estimadores robustos



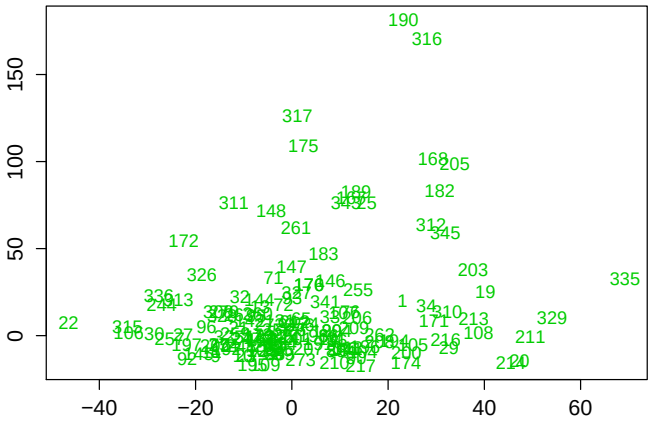
Robust

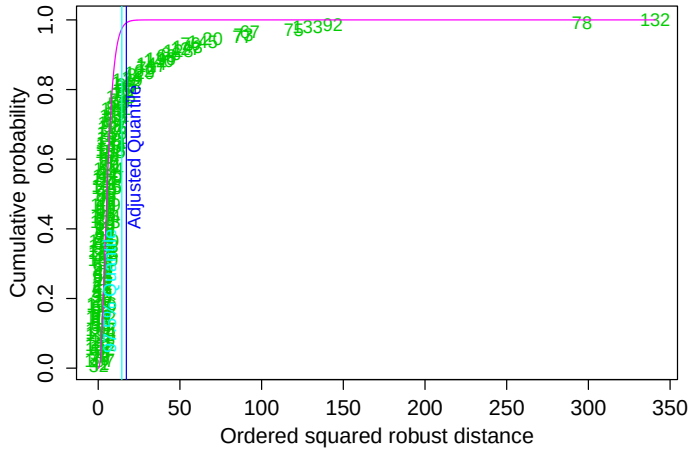
Std. Dev.

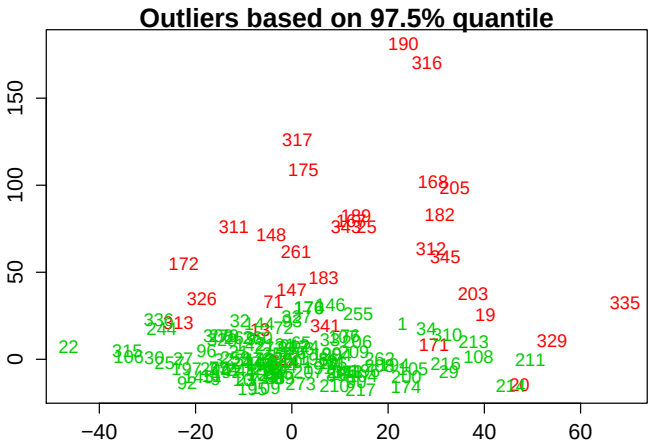
Ejemplo

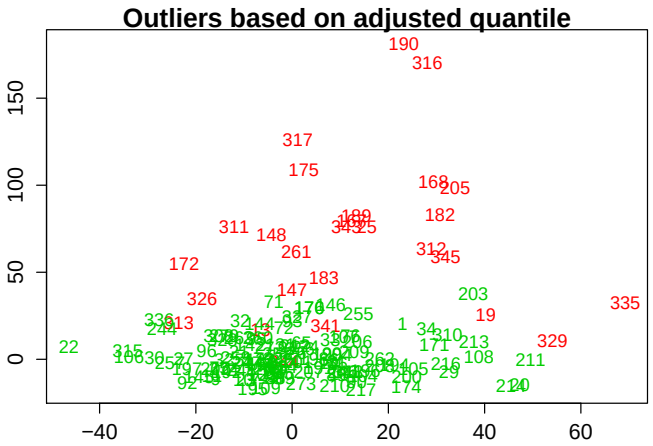
```
library(mvoutlier)
aq.plot(bupa[bupa$V7==1,1:6],alpha=0.01)
```

Ejemplo




Cumulative probability
Adjusted Quantile
Ordered squared robust distance


Outliers based on 97.5% quantile


Outliers based on adjusted quantile

Outlier basado en densidad local

- En este tipo de outliers la densidad de los vecinos de una distancia juega un crucial rol Además, una instancia no es explícitamente clasificada como outlier ó no-outlier; lo que se hace es calcular para cada instancia un factor de outlier local (LOF) y esta medida da una idea de que tan fuerte una instancia puede ser un outlier.

Ejemplo

- Se puede usar la función `lofactor` de la librería `DMwR`.

```
lofactor(data, k)
```

- Argumentos:
  - data: conjunto de datos
  - k: número de vecinos más cercanos a ser utilizados para el cálculo del factor de oulier local.

```
bupa1=bupa[bupa$V7==1,1:6]
indice=as.numeric(rownames(bupa1))
lof=lofactor(bupa1,10)
lof
indice [order(lof,decreasing=T)][1:10]
```

Otros Métodos

```
bupa1=bupa[bupa$V7==1,1:6]
indice=as.numeric(rownames(bupa1))
outlier=pcout(bupa1)
outlier
indice [order(outlier$wfinal,decreasing=F)][1:10]

outlier1=sign2(bupa1)
outlier1
indice [order(outlier1$x.dist,decreasing=T)][1:10]
```

Detección outlier - Cluster

```
library(cluster)
bupa1=bupa[bupa[,7]==1,1:6]
pambupa1=pam(bupa1,20,stand=T)
pambupa1$clusinfo
bupa1[pambupa1$clustering==19,]
```

Transformación de Datos

- Suavizamiento: Remover datos ruidosos
- Agregación: resumen, construcción de cubos de datos
- Normalización
  - Normalización min-max
  - Normalización z-score
  - Normalización por escalamiento decimal
- Construcción de Atributos
  - Nuevos atributos construidos basados en los anteriormente especificados.

Normalización

- Consiste en reescalar los valores de los datos a un rango pre-especificado.
- Normalizar los datos de entrada ayudará a acelerar la fase de aprendizaje.
- Los atributos con rangos grandes de valores tendrán más peso que los atributos con rangos de valores más pequenos, y entonces dominarán la medida de distancia.
- Por ejemplo, el clasificador K-nearest usando la medida de distancia euclideana depende de que todas las dimensiones de los valores de entrada estén en la misma escala.
- También puede ser necesario aplicar algún tipo de normalización de datos para evitar problemas numéricos tales como pérdida de precisión y desbordamientos aritméticos (overflows).

Normalización Z-score

Los valores son normalizados de la siguiente manera:

$$V' = \frac{V - \overline{x}}{S}$$

Este tipo de normalización funciona adecuadamente cuando:

- No se conoce el mínimo ni el máximo de los datos originales.
- Valores outlier pueden afectar el rango de los datos (pero no los elimina).

La función rescaler

- En R para realizar la normalización Softmax se puede usar la función `rescaler` de la librería `reshape`.

Aplicación de la normalización z-score

```
> library(reshape)
> zbupa<-rescaler(x=bupa[,-7],type="sd")
> summary(zbupa)
       V1                  V2                  V3
 Min.   :-5.65622   Min.   :-2.5545   Min.   :-1.3533
 1st Qu.:-0.71029   1st Qu.:-0.7014   1st Qu.:-0.5845
 Median :-0.03584   Median :-0.1564   Median :-0.2258
 Mean   : 0.00000   Mean   : 0.0000   Mean   : 0.0000
 3rd Qu.: 0.63861   3rd Qu.: 0.5521   3rd Qu.: 0.1842
 Max.   : 2.88676   Max.   : 3.7133   Max.   : 6.3854
       V4                  V5                  V6
 Min.   :-1.9518   Min.   :-0.8479   Min.   :-1.0351
 1st Qu.:-0.5607   1st Qu.:-0.5932   1st Qu.:-0.8853
 Median :-0.1633   Median :-0.3384   Median :-0.1363
 Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
 3rd Qu.: 0.2341   3rd Qu.: 0.1966   3rd Qu.: 0.7624
 Max.   : 5.6989   Max.   : 6.5907   Max.   : 4.9568
```

Normalización Min-Max

- Los valores son transformado en forma lineal a un rango pre-especificado [a,b]

$$V^{'} = \frac{(V - X_{min})\,(b - a)}{X_{max} - X_{min}} + a$$

donde X_{min} es el valor mínimo en los datos originales y X_{max} el valor máximo.

- Este tipo de normalización preserva las relaciones entre los datos.
- Como desventaja se puede mencionar que si un nuevo dato cae fuera del rango original ocasionará un error.

La función `ReScaling`

- En R para realizar la normalización Min-Max se puede usar la función `ReScaling` de la librería `DMwR`.

  ```
  ReScaling(x, t.mn, t.mx)
  ```

- Argumentos:
  - x: conjunto de datos a ser normalizado.
  - t.mn: el nuevo valor mínimo
  - t.mx: el nuevo valor máximo

La función ReScaling

Aplicación de la normalización Min-Max

```
> library(DMwR)
> mmbupa=bupa
> mmbupa[,-7]<-sapply(bupa[,-7],FUN=ReScaling, t.mn=0, t.mx=1)
> summary(mmbupa)
       V1                V2                V3                V4
 Min.   :0.0000   Min.   :0.0000   Min.   :0.00000   Min.   :0.0000
 1st Qu.:0.5789   1st Qu.:0.2957   1st Qu.:0.09934   1st Qu.:0.1818
 Median :0.6579   Median :0.3826   Median :0.14570   Median :0.2338
 Mean   :0.6621   Mean   :0.4076   Mean   :0.17487   Mean   :0.2551
 3rd Qu.:0.7368   3rd Qu.:0.4957   3rd Qu.:0.19868   3rd Qu.:0.2857
 Max.   :1.0000   Max.   :1.0000   Max.   :1.00000   Max.   :1.0000
       V5                V6                V7
 Min.   :0.00000   Min.   :0.0000   Min.   :1.00
 1st Qu.:0.03425   1st Qu.:0.0250   1st Qu.:1.00
 Median :0.06849   Median :0.1500   Median :2.00
 Mean   :0.11399   Mean   :0.1728   Mean   :1.58
 3rd Qu.:0.14041   3rd Qu.:0.3000   3rd Qu.:2.00
 Max.   :1.00000   Max.   :1.0000   Max.   :2.00
```

Normalización por Escalamiento Decimal

- La normalización se realiza moviendo el punto decimal de los valores. El número de puntos decimales depende del máximo valor absoluto.

$$V' = \frac{V}{10^j}$$

donde j es el entero mas pequeño tal que $max\left(|V'|\right) < 1$

- Sólo es útil cuando los valores de los atributos son mayores que 1 en valor absoluto.
- Esta normalización transforma los datos al rango [-1,1]
- Ejemplo: Si el valor de A varía entre -986 y 917, el valor máximo de A en val.abs. es 986. Para normalizar se divide entonces por 1000: -986 –normalizado–¿ -0.986.

La función decscale

- En R para realizar la normalización por escalamiento decimal se puede usar la función `decscale` de la librería `dprep`.

Aplicación de la normalización por escalamiento decimal

```
> dsbupa<-decscale(bupa)[,-7]
> dsbupa<-cbind(dsbupa,bupa[,7])
> summary(dsbupa)
       V1                V2                V3                V4
 Min.   :0.06500   Min.   :0.02300   Min.   :0.00400   Min.   :0.0500
 1st Qu.:0.08700   1st Qu.:0.05700   1st Qu.:0.01900   1st Qu.:0.1900
 Median :0.09000   Median :0.06700   Median :0.02600   Median :0.2300
 Mean   :0.09016   Mean   :0.06987   Mean   :0.03041   Mean   :0.2464
 3rd Qu.:0.09300   3rd Qu.:0.08000   3rd Qu.:0.03400   3rd Qu.:0.2700
 Max.   :0.10300   Max.   :0.13800   Max.   :0.15500   Max.   :0.8200
       V5                V6          bupa[, 7]
 Min.   :0.00500   Min.   :0.00000   Min.   :1.00
 1st Qu.:0.01500   1st Qu.:0.00500   1st Qu.:1.00
 Median :0.02500   Median :0.03000   Median :2.00
 Mean   :0.03828   Mean   :0.03455   Mean   :1.58
 3rd Qu.:0.04600   3rd Qu.:0.06000   3rd Qu.:2.00
 Max.   :0.29700   Max.   :0.20000   Max.   :2.00
```

Normalización Sigmoidal

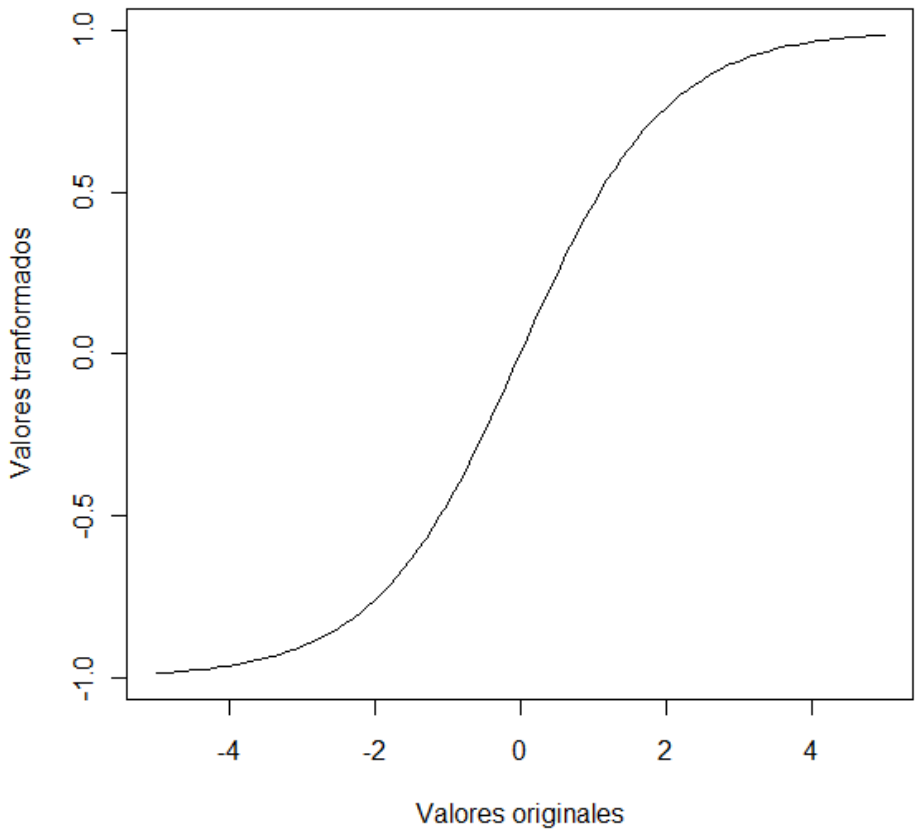
- Se realiza una transformación no lineal de los datos para llevarlos al rango [-1,1]

$$V' = \frac{1 - e^{-a}}{1 + e^{-1}}$$

donde $a = \frac{V - \overline{x}}{s}$

- Los valores dentro de una desviación estándar de la media son mapeados a la región casi linear del sigmoide. Los puntos anómalos son comprimidos a lo largo de las colas de la función sigmoidal.
- La normalización sigmoidal es especialmente apropiada cuando se tienen datos anómalos que se desean incluir en el conjunto de datos. Este previene que los valores que ocurren más comúnmente sean comprimidos en los mismos valores, sin perder la habilidad de representar grandes valores anómalos.

Normalización sigmoidal



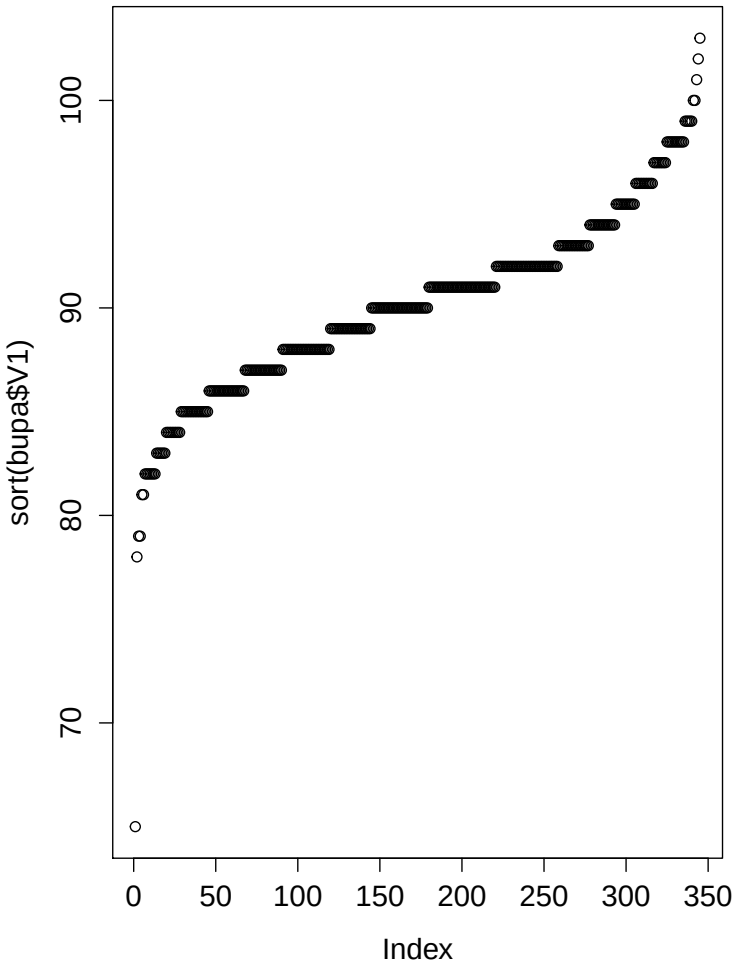
La función `signorm`

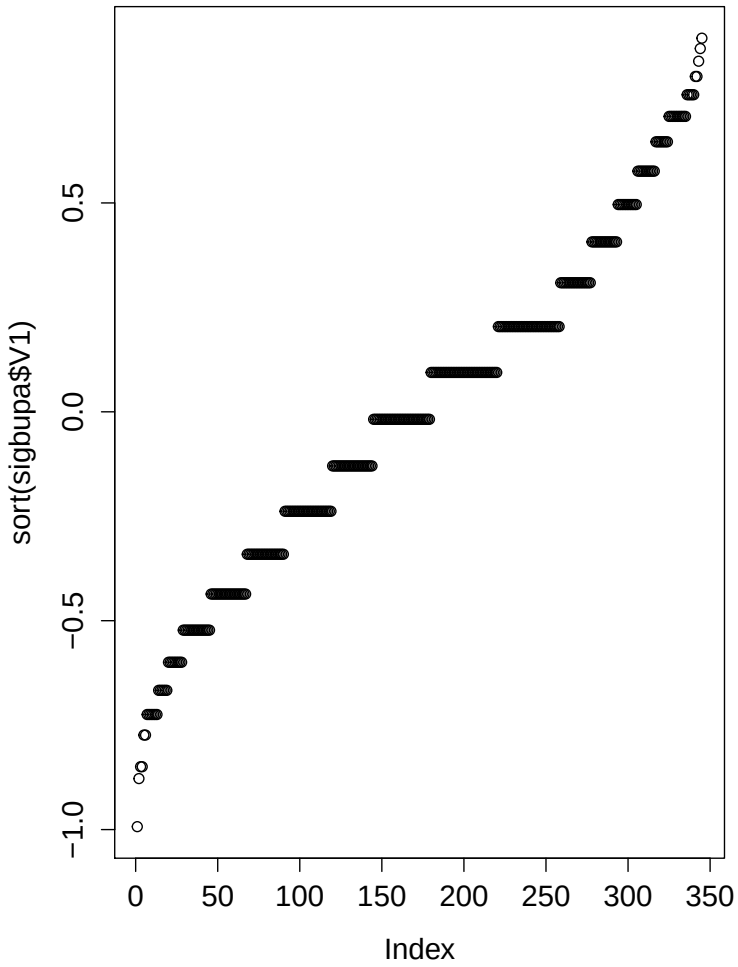
- En R para realizar la normalización sigmoidal se puede usar la función `signorm` de la librería `dprep`.

Aplicación de la normalización sigmoidal

```
> sigbbupa<-bupa
> sigbupa[,-7]<-signorm(bupa[,-7])
> summary(sigbupa)
       V1                  V2                  V3                  V4
 Min.   :-0.993033   Min.   :-0.85575   Min.   :-0.58933   Min.   :-0.75128
 1st Qu.:-0.340929   1st Qu.:-0.33701   1st Qu.:-0.28422   1st Qu.:-0.27324
 Median :-0.017918   Median :-0.07804   Median :-0.11242   Median :-0.08147
 Mean   : 0.001603   Mean   :-0.01624   Mean   :-0.03786   Mean   :-0.03316
 3rd Qu.: 0.308876   3rd Qu.: 0.26926   3rd Qu.: 0.09184   3rd Qu.: 0.11654
 Max.   : 0.894376   Max.   : 0.95237   Max.   : 0.99663   Max.   : 0.99332
       V5                  V6             bupa[, 7]
 Min.   :-0.40025   Min.   : 0.000   Min.   :1.00
 1st Qu.:-0.28818   1st Qu.: 0.500   1st Qu.:1.00
 Median :-0.16761   Median : 3.000   Median :2.00
 Mean   :-0.04280   Mean   : 3.455   Mean   :1.58
 3rd Qu.: 0.09797   3rd Qu.: 6.000   3rd Qu.:2.00
 Max.   : 0.99726   Max.   :20.000   Max.   :2.00
par(mfrow=c(1,2))
plot(sort(bupa$V1))
plot(sort(sigbupa$V1))
```

Efecto de la transformación sigmoidal





Normalización softmax

- Es denominada de esta forma porque tiende suavemente hacia su valor máximo o mínimo sin llegar absolutamente. La transformación es mas o menos lineal en el rango medio, y tiene una ligera no linealidad a ambos extremos.
- Esta transformación lleva los valores al rango [0,1]
- La transformación asegura que no ocurran valores futuros que caigan fuera del rango.

$$V^{'} = \frac{1}{1 + e^{-a}}$$

donde $a = \frac{V - \overline{x}}{s}$

La función SoftMax

- En R para realizar la normalización Softmax se puede usar la función SoftMax de la librería DMwR.
- En esta función debe de considerarse al parámetro $\lambda = 2\pi$

Aplicación de la normalización softmax

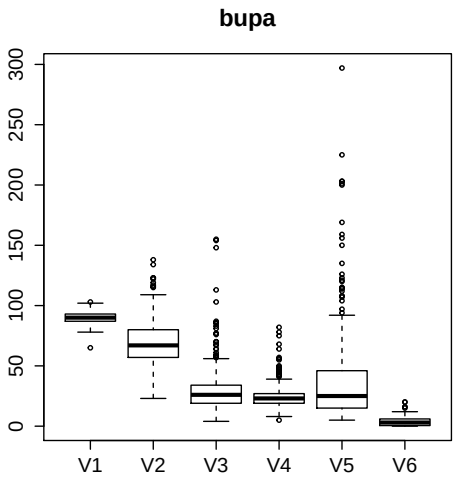
```
> library(DMwR)
> sigbbupa<-bupa
> sigbupa[,-7]<-signorm(bupa[,-7])
> summary(sigbupa)
       V1                  V2                  V3                  V4
 Min.   :-0.993033   Min.   :-0.85575   Min.   :-0.58933   Min.   :-0.75128
 1st Qu.:-0.340929   1st Qu.:-0.33701   1st Qu.:-0.28422   1st Qu.:-0.27324
 Median :-0.017918   Median :-0.07804   Median :-0.11242   Median :-0.08147
 Mean   : 0.001603   Mean   :-0.01624   Mean   :-0.03786   Mean   :-0.03316
 3rd Qu.: 0.308876   3rd Qu.: 0.26926   3rd Qu.: 0.09184   3rd Qu.: 0.11654
 Max.   : 0.894376   Max.   : 0.95237   Max.   : 0.99663   Max.   : 0.99332
       V5                  V6             bupa[, 7]
 Min.   :-0.40025   Min.   : 0.000   Min.   :1.00
 1st Qu.:-0.28818   1st Qu.: 0.500   1st Qu.:1.00
 Median :-0.16761   Median : 3.000   Median :2.00
 Mean   :-0.04280   Mean   : 3.455   Mean   :1.58
 3rd Qu.: 0.09797   3rd Qu.: 6.000   3rd Qu.:2.00
```

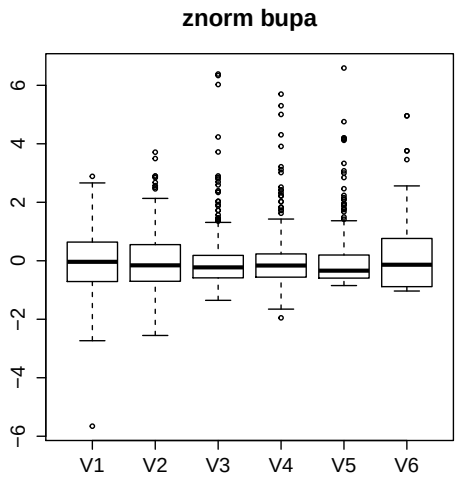
Efecto de normalizar los datos

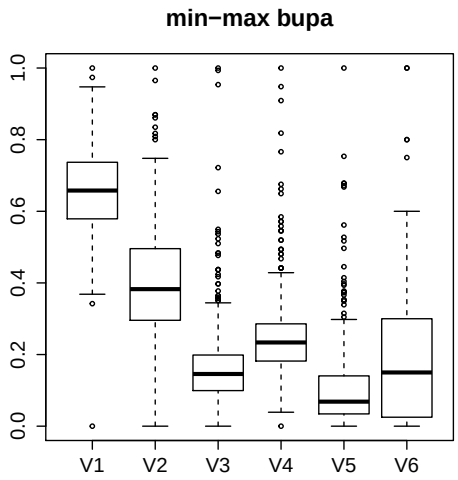
Boxplots de efectos de las transformaciones

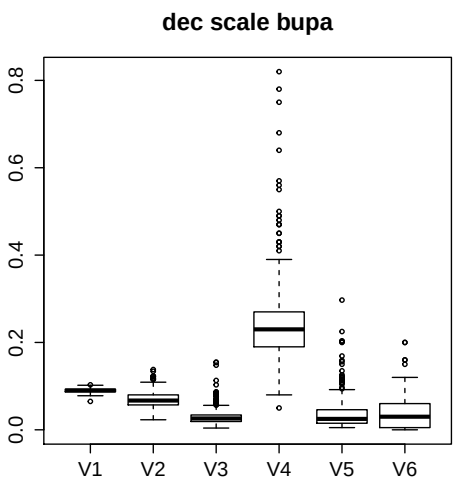
```
par(mfrow=c(2,3))
boxplot(bupa[,1:6],main="bupa")
boxplot(zbupa[,1:6],main="znorm bupa")
boxplot(mmbupa[,1:6],main="min-max bupa")
boxplot(dsbupa[,1:6],main="dec scale bupa")
boxplot(sigbupa[,1:6],main="signorm bupa")
boxplot(softbupa[,1:6],main="softmax bupa")
```

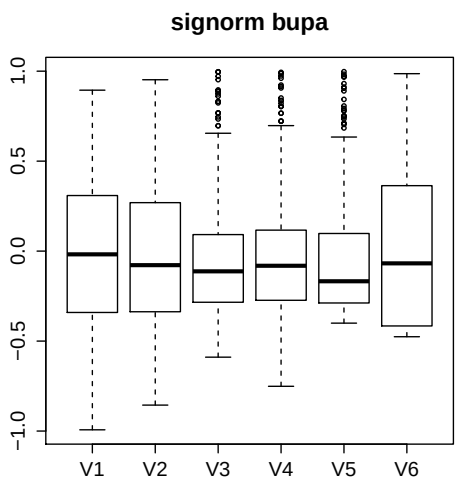
Efecto de normalizar los datos

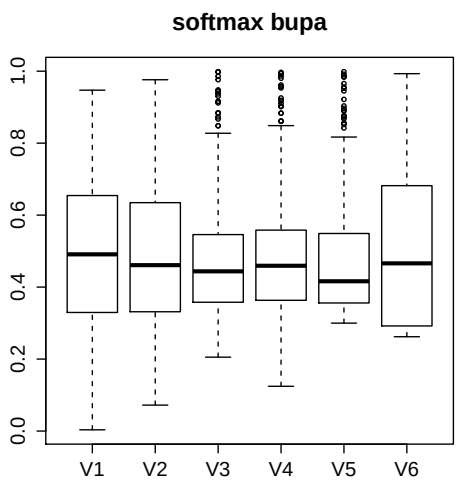












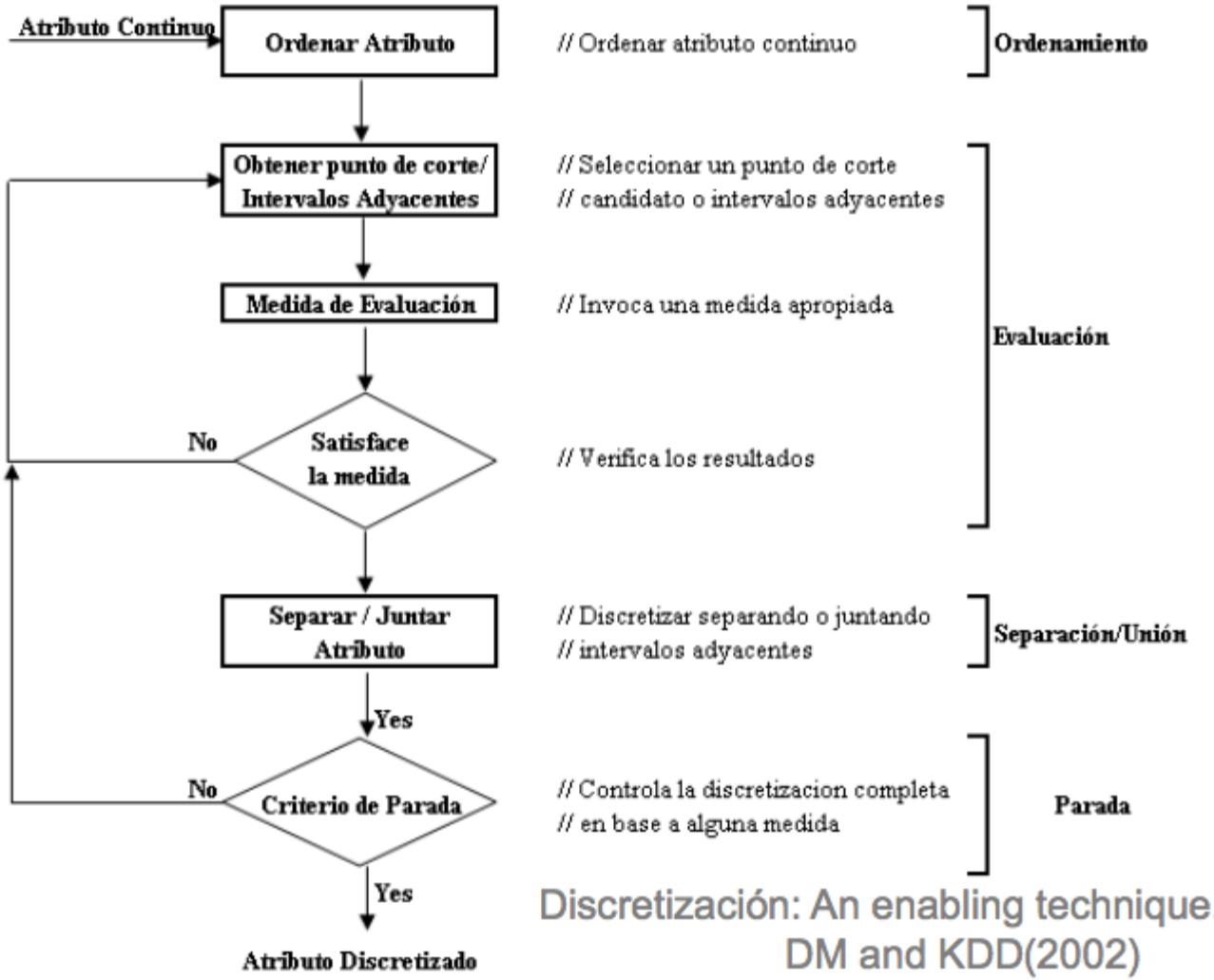
Reducción de Datos

- **Importancia**
  - Warehousing puede resultar en terabytes de datos: Tareas complejas de datamining pueden demorar mucho tiempo en ejecutarse sobre el conjunto completo de datos...
  - Busca obtener una representación reducida del conjunto. de datos que es mucho más pequeña en volumen pero produce los mismos (o casi iguales) resultados analíticos.
- **Estrategias en Reducción de Datos**
  - Agregación del cubo de datos.
  - Discretización.
  - Reducción de la dimensionalidad.

Discretización

- Es un método que transforma datos cuantitativos en cualitativos
- Algunas metodologías solo aceptan atributo categóricos. Ejemplo : Naive Bayes.
- El proceso de aprendizaje es frecuentemente menos eficiente cuando las datos son solo cuantitativos

Discretización



Discretización: An enabling technique. Liu et al DM and KDD(2002)

Discretización

- **Métodos Top-Down**: se inicia con una lista vacía de puntos de corte y se continúan agregando nuevos puntos a la lista "separando" los intervalos mientras la discretización progresa.
- **Métodos Bottom-Up**: se inicia con la lista completa de todos los valores continuos de la variable como puntos de corte y se eliminan algunos de ellos "juntando" los intervalos mientras la discretización progresa.

Discretización

- **Discretización Dinámi**ca: algunos algoritmos de clasificación tienen incorporados mecanismos para discretizar atributos continuos (por ejemplo, árboles de decisión). Los atributos continuos son discretizados durante el proceso de clasificación.
- **Discretización Estáti**ca: Es un paso más en el preprocesamiento de datos. Los atributos continuos son previamente discretizados antes de la tarea de clasificación.
- No existe una ventaja clara de algunos de los métodos (Dougherty, Kohavi, and Sahami, 1995)
- También es conocido como "*Binning*".

Discretización

- **Métodos Supervisados**: Utilizan la información de la clase para la discretización.
- **Métodos No Supervisados**: No utilizan la información de la clase para la discretización.

Discretización

- Intervalo de igual amplitud
- Intervalos de igual frecuencia
- Discretización 1R
- Discretización por Entropía
- Discretización por chiMerge

Intervalos de Igual Amplitud

- Es similar a la elaboración de tablas de frecuencia y de contingencia.
- Se divide el rango de la variable en k intervalos de igual tamaño.
- Sea X_{min} y X_{max} el valor mínimo y máximo de la variable, el ancho del intervalo es:

$$C = \frac{X_{max} - X_{min}}{k}$$

- Luego los puntos de corte son dados por:

$$X_{min} + C,\ X_{min} + 2C, \ldots, X_{min} + (k-1)\mathrm{C}$$

- Desventajas: No supervisado. Sensible a outliers.
- Ventajas: Fácil de implementar. Produce una abstracción de los datos razonable.

Intervalos de Igual Amplitud

Como determinar el número de intervalos

- Formula de Sturges:$k = 1 + \log_2(n) = 1 + 3.322 \log_{10}(n)$
- Formula de Friedman-Diaconis: $C = 2\frac{RIC}{n^{\frac{1}{3}}}$
- Formula de Scott: $C = 3.5\frac{s}{n^{\frac{1}{3}}}$

Para Friedman-Diaconis y Scott luego se calcula

$$k = \frac{X_{max} - X_{min}}{C}$$

Intervalos de Igual Frecuencia

- Se debe dividir el rango en k intervalos.
- Cada intervalo de contener aproximadamente el mismo número de instancias.
- No se utiliza la información de la clase.

Ejemplo

- En R para realizar la discretización en intervalos de igual frecuencia se puede utilizar la función `discretize` de la librería `arules`.

```
discretize(x, method="frequency", categories )
```

- Argumentos:
  - x: vector de datos
  - method : Usar frequency
  - categories : número de categorías

```
#Discretizacion con intervalos de igual frecuencia
dbupa=bupa
for(h in 1:6){
dbupa[,h]<-discretize(bupa[,h],method="frequency",categories=10)
}
table(dbupa[,1])
table(dbupa[,2])
table(dbupa[,3])
table(dbupa[,4])
table(dbupa[,5])
table(dbupa[,6])
```

Ejemplo

- En R para realizar la discretización en intervalos de igual frecuencia se puede utilizar la función `discretize` de la librería `arules`.

```
discretize(x, method=?frequency", categories )
```

- Argumentos:
  - x: vector de datos
  - method : Usar frequency
  - categories : número de categorías

```
#Discretizacion con intervalos de igual frecuencia
dbupa=bupa
for(h in 1:6){
dbupa[,h]<-discretize(bupa[,h],method="frequency",categories=10)
}
table(dbupa[,1])
table(dbupa[,2])
table(dbupa[,3])
table(dbupa[,4])
table(dbupa[,5])
table(dbupa[,6])
```

Discretización por entropía

- Se calcula la entropía o contenido de información en base a la clase.
- Luego se encuentra la mejor partición posible de modo que las divisiones sean las más puras posibles (la mayoria de los valores en una división correspondan a una misma clase) Entropía $= -\sum_{i=1}^{n} p_i log_2(p_i)$
- Si la entropía es pequeña el conjunto es relativamente puro, si la entropía es grande el conjunto está muy mezclado. La Entropía varía entre 0 y 1.

Discretización por entropía

- Se tiene el siguiente conjunto de datos: S = (0,Y), (4,Y),(12,Y), (16,N), (16,N), (18,Y), (24,N), (26,N), (28,N).
- Sea p1 =4/9 la fracción de pares con clase=Y, y p2 = 5/9 la fracción de pares con clase=N.
- Entropía(S)=0.991076
- Dado un conjunto de muestras S, si S es particionado en dos intervalos S_1 y S_2 usando el punto de corte T, la entropía después de particionar es: $E(S,T) = \frac{|S_1|}{|S|}Ent(S_1) + \frac{|S_2|}{|S|}Ent(S_2)$

Discretización por entropía

- Por ejemplo si T=14
- S_1= (0,Y), (4,Y), (12,Y) y S_2= (16,N), (16,N), (18,Y), (24,N), (26,N), (28,N)
- E(S,T)=(3/9)*E(S_1)+(6/9)*E(S_2)=3/9*0+(6/9)* 0.6500224
- E(S,T)=.4333
- Ganancia de información de la partición, Gain(S,T) = Entropía(S) - E(S,T).
- Ganacia de Información=.9910-.4333=.5577

Discretización por entropía

- Para $T = 21$, Ganancia de Información=.9910-.6121=.2789. Por lo que v=14 es una mejor partición.
- El objetivo de este algoritmo es encontrar la partición con la máxima ganancia de información. La ganancia máxima se obtiene cuando E(S,T) es mínima.
- La mejor partición(es) se encuentran examinando todas las posibles particiones y seleccionando la óptima. El punto de corte que minimiza la función de entropía sobre todos los posibles puntos de corte se selecciona como una discretización binaria.
- El proceso es aplicado recursivamente a particiones obtenidas hasta que se cumpla algún criterio de parada, $Ent(S) - E(T, S) > \partial$

Discretización por entropía

- Donde $\partial = \frac{log(N-1)}{N} + \frac{\triangle(T,S)}{N}$
- y $\triangle(S,T) = log_2(3^c - 2) - [cEnt(S) - c_1 Ent(S_1) - c_2 Ent(S_2)]$
- Aquí c es el número de clases en S, c_1 es el número de clases en S_1 y c_2 es el número de clases en S_2. Esto es llamado el Principio de Longitud de Descripción Mínima (MDLP).

Ejemplo

- En R para realizar la discretización en intervalos por entropía se puede utilizar la función `mdlp` de la librería `discretization`.

```
mdlp(data)
```

- Argumentos:
  - data : conjunto de datos, se asume que la última columna contiene a las clases.

```
#Discretizacion por entropía
dbupa=mdlp(bupa)$Disc.data
table(dbupa[,1])
table(dbupa[,2])
table(dbupa[,3])
table(dbupa[,4])
table(dbupa[,5])
table(dbupa[,6])
```

ChiMerge

Características de ChiMerge:

- Las frecuencias relativas de clase deben ser bastante parecidas dentro de un intervalo (de lo contrario se debe dividir el intervalo).
- Dos intervalos adyacentes no deben tener similares frecuencias relativas de clase (de lo contrario se debe juntar).

ChiMerge

Tabla de contingencia

	Clase 1	Clase 2	Total
Intervalo I	A_{11}	A_{12}	R_1
Intervalo II	A_{21}	A_{22}	R_2
Total	C_1	C_2	N

ChiMerge

- Este valor puede ser obtenido así: $\chi^2 = \sum\sum \frac{(A_{ij}-E_{ij})^2}{E_{ij}}$
  - k: número de clases
  - A_{ij}: número de datos en el i-ésimo intervalo, j-ésima clase
  - E_{ij}: frecuencia esperada de $A_{ij} = \frac{(R_i \times C_j)}{N}$
  - R_i: número de datos en el i-ésimo intervalo
  - C_j: número de datos en la j-ésima clase
  - N: número total de datos en los dos intervalos
- Si $E_{ij} = 0$ entonces asignar a E_{ij} un valor pequeño, por ejemplo 0.1

ChiMerge

1. Obtener el valor de χ^2 para cada par de intervalos adyacentes.
2. Juntar el par de intervalos adyacentes que tengan el menor valor de χ^2.

- Repetir 1 y 2 hasta que los valores χ^2 de todos los pares adyacentes excedan un valor dado (threshold)
- Threshold: es determinado por el nivel de significancia y el grado de libertad = número de clases - 1.

ChiMerge

Muestra:	F	K
1	1	1
2	3	2
3	7	1
4	8	1
5	9	1
6	11	2
7	23	2
8	37	1
9	39	2
10	45	1
11	46	1
12	59	1

ChiMerge

- Considerando cada punto como una partición los limites de los intervalos serian: 0, 2, 5, 7.5, 8.5, 10, etc.
- En este ejemplo se obtiene el minimo χ^2 en los intervalos $[7.5, 8.5]$ y $[8.5, 10]$.

	$K=1$	$K=2$	$\sum$
Intervalo $[7.5, 8.5]$	$A_{11}=1$	$A_{12}=0$	$R_1=1$
Intervalo $[8.5, 10]$	$A_{21}=1$	$A_{22}=0$	$R_2=1$
$\sum$	$C_1=2$	$C_2=0$	$N=2$

- Los valores esperados serán:
  - $E_{11} = 2/2 = 1$
  - $E_{12} = 0/2 \approx 0.1$
  - $E_{21} = 2/2 = 1$
  - $E_{22} = 0/2 \approx 0.1$

ChiMerge

- χ^2=2
- Para grados de libertad d $= 1$, y $\chi^2 = 0.2 < 2.706$ (el valor de la tabla chi-cuadrado para un $\alpha = 0.1$),
- Luego se concluye que no hay diferencias significativas y se procede a juntar las particiones.
- Luego el proceso se repite juntando particiones

ChiMerge

	$K=1$	$K=2$	$\sum$
Intervalo $[0, 7.5]$	$A_{11}=2$	$A_{12}=1$	$R_1=3$
Intervalo $[7.5, 10]$	$A_{21}=2$	$A_{22}=0$	$R_2=2$
$\sum$	$C_1=4$	$C_2=1$	$N=5$

- Los valores esperados serán:
  - $E_{11}=12/5=2.4$
  - $E_{12}=3/5=0.6$
  - $E_{21}=8/5=1.6$
  - $E_{22}=2/5=0.4$
- $\chi^2=0.843$. Para grados de libertad $=1$, y $\chi^2=0.834<2.706$ (el valor de la tabla chi-cuadrado para un $\alpha=0.1$),
- Luego se concluye que no hay diferencias significativas y se procede a juntar las particiones.

ChiMerge

	$K=1$	$K=2$	$\sum$
Intervalo $[0, 10.0]$	$A_{11}=4$	$A_{12}=1$	$R_1=5$
Intervalo $[10.0, 42.0]$	$A_{21}=1$	$A_{22}=3$	$R_2=4$
$\sum$	$C_1=5$	$C_2=4$	$N=9$

- Los valores esperados son: $E_{11}=2.78$, $E_{12}=2.22$, $E_{21}=2.22$, $E_{22}=1.78$, y $\chi^2=2.72>2.706$
- Luego se concluye que hay diferencias significativas y ya no se puede juntar las particiones.

Ejemplo

- En R para realizar la discretización por el método de ChiMerge se utiliza la función `chiM` de la librería `discretization`.

```
chiM(data, alpha = 0.05)
```

- Argumentos:
  - data: conjunto de datos, se asume que la última columna contiene a las clases.
  - alpha: nivel de significancia

```
#Discretizacion con chiMerge
dbupa=chiM(bupa,0.05)$Disc.data
table(dbupa[,1])
table(dbupa[,2])
table(dbupa[,3])
table(dbupa[,4])
table(dbupa[,5])
table(dbupa[,6])
```

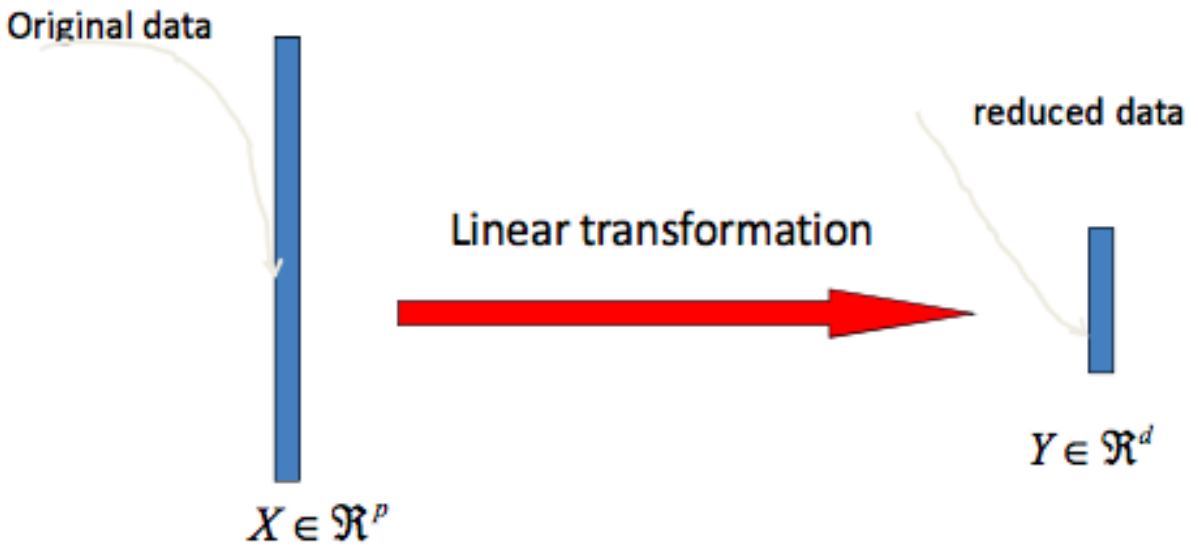
¿Qué es el Análisis de Componentes Principales (PCA)?

- Método Exploratorio: Ver como más de 3 variables están relacionadas
- Método de Reducción de Datos: Variación en menos variables. Nuevas variables independientes.
- Encuentra una representación de los datos en menos dimensiones.
- Busca la mejor combinación lineal de variables de tal forma que recoja la mayor parte de la variabilidad (información) original de los datos.
- Paso previo al uso de otras técnicas.
- Util para comprimir y clasificar datos.

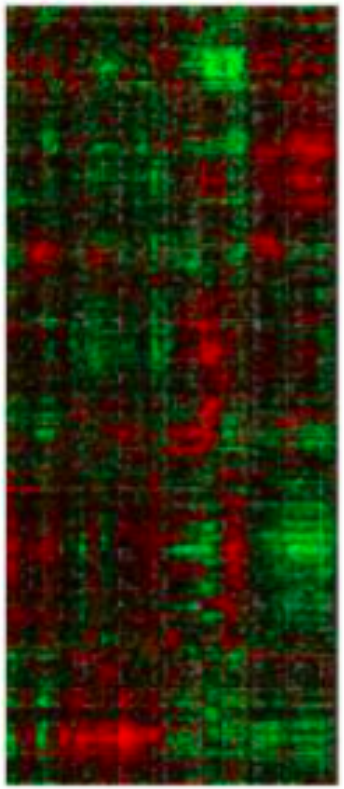
¿Por qué usar PCA?

- Dado un conjunto de p variables: $X_1, X_2, \ldots, X_p$
- Se desea calcular una transformación lineal (proyección) tal que:

$$X \in \Re^p \rightarrow Y \in \Re^m$$



Datos en Grandes Dimensiones



Gene expression

Face images

Handwritten digits

¿Por qué usar PCA?

- La mayoría de técnicas de Minería de Datos y Machine Learning no suelen ser efectivas para datos en grandes dimensiones.
  - Maldición de la Dimensionalidad.
  - La eficiencia y presición en la clasificación se deteriora rapidamente mientras la dimensión de los datos se incrementa.
- La dimensión intrínseca puede ser más pequeña
  - Por ejemplo, el número de genes responsables de cierta enfermedad puede ser menor.

¿Por qué usar PCA?

- Visualización: Proyeción de datos de grandes dimensiones en 2D o 3D.
- Compresión de Datos: Almacenamiento y recuperación eficiente.
- Extracción de Atributos: Extraer atributos útiles.

Algoritmos de Dimensionalidad

- Criterio
  - No Supervisados: Minimizar la pérdida de información.
  - Supervisados: Minimizar errores de clasificación.
- No Supervisados
  - Indexación Semántica Latente (LSI)
  - Análisis de Componentes Independientes (ICA)
  - Análisis de Componentes Principales (PCA)
  - Análisis de Correlación Canónica (CCA)
- Supervisados
  - Análisis Discriminante Lineal (LDA)

Componentes Principales I

- La primera componente principal para un conjunto de atributos $X_1, X_2, \ldots$ es la combinación lineal normalizada de los atributos tal que:

$$Y_1 = \phi_{11}X_1 + \phi_{21}X_2 + \ldots + \phi_{p1}X_p$$

  tenga la mayor varianza.
- Por normalizado entendemos que $\sum_{j=1}^{p} \phi_{j1}^2 = 1$
- Los coeficientes $\phi_{11}, \ldots, \phi_{p1}$ corresponden a las cargas del primer componente principal.
- El primer vector de cargas de componentes principales estará dado por $\phi_1 = (\phi_{11}, \phi_{21}, \ldots, \phi_{p1})^T$.
- Restricción: $\sum_{j=1}^{p} \phi_{j1}^2 = 1$

Componentes Principales II

- En otras palabras, la primera componente principal resuelve el problema de optimización:

$$\underset{\phi_{11},\ldots,\phi_{p1}}{maximizar} \left\{ \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{j=1}^{p} \phi_{j1} x_{ij} \right)^2 \right\} \text{ sujeto a } \sum_{j=1}^{p} \phi_{j1}^2 = 1$$

- El objetivo es maximizar la varianza muestral de los n valores de y_{i1}.
- Los valores $y_{11}, \ldots, y_{n1}$ son conocidos como las puntuaciones o scores.
- El problema anterior puede ser resuelto por descomposición de autovalores u otros métodos.
- El siguiente componente principal será la siguiente combinación lineal de los atributos que maximice la varianza y que no esté correlacionado con el primer componente principal.