Probabilidad-Parte II

Probabilidad Aplicada 3-602

Conceptos Importantes

Registro

Un **registro** es cada una de las observaciones realizadas. En R es cada fila de registros.

```
head(insurance, 1)
```

```
## # A tibble: 1 x 7
##     age sex      bmi children smoker region    charges
##   <dbl> <chr>  <dbl>    <dbl> <chr>  <chr>       <dbl>
## 1    19 female  27.9        0 yes    southwest  16884.
```

En este caso, la primera observación se refiere a una mujer de 19 años de edad, un un IMC de 27.9, fumadora, sin hijos y que vive en la región suroeste de los Estados Unidados, a la que su aseguradora le cobra 16884.924 por la póliza.

Variable

Una **variable** es una propiedad que define cada uno de los registros.

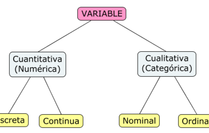
```
glimpse(insurance)
```

```
## Rows: 1,338
## Columns: 7
## $ age      <dbl> 19, 18, 28, 33, 32, 31, 46, 37, 37, 60, 25, 62, 23, 56, 27, 1~
## $ sex      <chr> "female", "male", "male", "male", "male", "female", "female",~
## $ bmi      <dbl> 27.900, 33.770, 33.000, 22.705, 28.880, 25.740, 33.440, 27.74~
## $ children <dbl> 0, 1, 3, 0, 0, 0, 1, 3, 2, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0~
## $ smoker   <chr> "yes", "no", "no", "no", "no", "no", "no", "no", "no", "no", ~
## $ region   <chr> "southwest", "southeast", "southeast", "northwest", "northwes~
## $ charges  <dbl> 16884.924, 1725.552, 4449.462, 21984.471, 3866.855, 3756.622,~
```

En este ejemplo las variables son: género, edad, IMC, número de hijos, fumador, región y costo de la póliza.

Cada una de estas variables, puede ser definida de diferentes maneras como el IMC que se calcula en función del peso y la altura.

El **tipo de variable** es una característica fundamental que determina cómo se manipula y analiza una variable.



En el ejemplo, tenemos:

Género: Variable cualitativa (categórica) y nominal.

Edad: Variable cuantitativa (numérica) y discreta.

En otros casos,

Peso: Variable cuantitativa (numérica) y continua.

Nivel de estudio alcanzado: Variable cualitativa (categórica) y ordinal.

Más variables y tipos de datos en R

Todas las variables se pueden clasificar en Cualitativas o Cuantitativas.

Pero en R, existen algunos tipos de datos especiales. Como por ejemplo:

Variables de tipo fecha

Ejemplo, supongamos que tenemos una columna más donde se registra la fecha de nacimiento del asegurado. Entonces, esta variable es cualitativa ordinal.

Repaso de R: Funciones de exploración inicial de datos

```
str(insurance)  #estructura del marco de datos
head(insurance, 10)  #Visualizamos los primeros 10 registros
tail(insurance)  #Visualizamos los últimos 6 registros
colnames(insurance)  #Muestramos los nombres de las variables del marco de datos
summary(insurance)  #Muestra un resumen estadístico de las variables del marco de datos
dim(insurance)  #dimensión del marco de datos
nrow(insurance)  #cantidad de filas
ncol(insurance)  #cantidad de columnas
```

Más de R: El paquete **dyplr**

Una importante contribución del paquete **dplyr** es que proporciona una "gramática" (particularmente verbos) para la manipulación y operaciones con data frames. Con esta gramática podemos comunicar mediante nuestro código que es lo que estamos haciendo en los data frames a otras personas (asumiendo que conozcan la gramática). Esto es muy útil, ya que proporciona una abstracción que anteriormente no existía. Por último, cabe destacar que las funciones del paquete dplyr son muy rápidas, puesto que están implementadas con el lenguaje C++.

Fuente: https://rsanchezs.gitbooks.io/rprogramming/content/chapter9/dplyr.html

Algunos ejemplo de verbos y funciones:

Algunos ejemplos de verbos y funciones dplyr

Verbo	Función
Seleccionar	select()
Filtrar	filter()
Ordenar/reordenar	arrange()
Agrupar por	group_by()
Resumir por estadisticos	summarise()/summarize()
Mutar/transformar	mutate()
Renombrar	rename()

Mas de R: El operador PIPE %>%

El operador pipeline **%>%** es útil para concatenar múltiples dplyr operaciones.

El operador **%>%** nos permite escribir una secuencia de operaciones de izquierda a derecha. El operador "pipe" es usado para conectar múltiples acciones en una única "pipeline" (tubería).

Fuente: https://rsanchezs.gitbooks.io/rprogramming/content/chapter9/dplyr.html

ATAJO DE TECLADO: CTRL+SHIFT+M = %>%

Más de R. Resúmenes estadísticos y subconjunto de datos

Resúmenes estadísticos: Pipe + summarise()

```
ejemplo_1 <- insurance %>%
    summarise(min(bmi) , mean(bmi), max(charges))
```

En este caso, definimos el objeto **ejemplo_1**. Para ello vamos a nuestro marco de datos/data frame **insurance** utilizando el pipe **%>%**. Una vez en el dataframe, con la función **summarise()** construimos nuestro resumen. ¿Qué queremos incluir en nuestro resumen? La respuesta a esta pregunta la incluimos en el argumento de la función **summarise**. Supongamos que deseamos conocer cuál es valor mínimo de bmi, el bmi promedio y el máximo valor de póliza asegurada. Entonces utilizamos las funciones **min()**, **mean()** y **max()** respectivamente, aplicadas a las variables de interés. Obtenemos...

min(bmi)	mean(bmi)	max(charges)
15.96	30.6634	63770.43

Vista del data frame Ejemplo_1

Resúmenes estadísticos: Pipe + group_by() + summarise()

La función **group_by()** tiene por objetivo agrupar los registros según los valores posibles de una variable del marco de datos/data frame. Pero esto agrupamiento no cambia la apariencia del marco de datos. Veamos el siguiente ejemplo:

```
ejemplo_2 <- insurance %>%
    group_by(sex)
```

En este caso, definimos el objeto **ejemplo_2**. Para ello vamos a nuestro marco de datos/data frame **insurance** utilizando el pipe **%>%**. Una vez allí, decidimos agrupar los datos por los valores posibles de la variable sex (famale, male). Pero si observamos el objeto **ejemplo_2** no notamos ningún cambio. Esto es porque la función **group_by()** realiza el agrupamiento pero, por sí sola no "muestra" ningún resultado.

Si deseamos ver los datos reordenados en función del agrupamiento previo, utilizamos la función **arrange()**:

```
ejemplo_3 <- insurance %>%
    group_by(sex) %>%
    arrange(sex)
```

La función **group_by()** es útil cuando la acompañamos con la función **summarise()**. Esta combinación provee información estadística (resumen) de los datos agrupados según los valores posibles de una o más variables. Veamos el siguiente ejemplo:

```
ejemplo_4 <- insurance %>%
    group_by(sex) %>%
    summarise(n())
```

En este caso, definimos el objeto **ejemplo_4**. Para ello vamos a nuestro marco de datos/data frame **insurance** utilizando el pipe **%>%**. Una vez allí, decidimos agrupar los datos por los valores posibles de la variable sex (famale, male). Una vez agrupados los datos, decidimos obtener algún estadístico. Para ello, invocamos con pipe a la función **summarise()** y dentro de su argumento, explicitamos el dato que queremos conocer. En este caso, queremos contar cuántos registros de asegurados hay según el sexo. Entonces, incorporamos en el argumento la función contar: **n()**

Si queremos, podemos asignarle el nombre "cantidad" a la columna que registra cuántos asegurados hay según el sexo.

```
ejemplo_4 <- insurance %>%
    group_by(sex) %>%
    summarise(cantidad = n())
```

Otros ejemplos

```
# ¿Cuántas personas hay por sexo y región?
ejemplo_7 <- insurance %>%
    group_by(sex, region) %>%
    summarise(cantidad = n())
ejemplo_7
```

```
## # A tibble: 8 x 3
## # Groups:   sex [2]
##   sex    region    cantidad
##   <chr>  <chr>        <int>
## 1 female northeast      161
## 2 female northwest      164
## 3 female southeast      175
## 4 female southwest      162
## 5 male   northeast      163
## 6 male   northwest      161
## 7 male   southeast      189
## 8 male   southwest      163
```

```
# ¿Cuántas personas hay por sexo y región y cuál es el promedio de bmi por
# región?
ejemplo_8 <- insurance %>%
    group_by(sex, region) %>%
    summarise(promedio_bmi = mean(bmi))
ejemplo_8
```

```
## # A tibble: 8 x 3
## # Groups:   sex [2]
##   sex    region    promedio_bmi
##   <chr>  <chr>            <dbl>
## 1 female northeast         29.3
## 2 female northwest         29.3
## 3 female southeast         32.7
## 4 female southwest         30.1
## 5 male   northeast         29.0
## 6 male   northwest         29.1
## 7 male   southeast         34.0
## 8 male   southwest         31.1
```

Subconjunto de datos: Pipe + select()

Si necesitamos quedarnos con algunas variables de nuestro data frame, como por ejemplo con las variables sex, age y smoker, utilizamos la función **select()**

```
subconjunto_5 <- insurance %>%
    dplyr::select(sex, age, smoker)  #Agrego dplyr::select para no entrar en conflicto con otra función
```

Subconjunto de datos: Pipe + filter()

Con la función **filter()** podemos crear subconjunto de datos, haciendo "consultas" por condiciones sobre las variables.

```
## queremos todos los registros correspondientes a varones
subconjunto_2 <- insurance %>%
    filter(sex == "male")
```

```
## queremos todos los registros de las personas asegurados que no viven en el
## suroeste
subconjunto_3 <- insurance %>%
    filter(region != "southwest")
```

```
## queremos ver cuáles son los registros donde region==southwest en el
## subconjunto_3 es decir, queremos chequear que no exista ningun registro con
## el valor region=southwest
ejemplo_10 <- subconjunto_3 %>%
    filter(region == "southwest")
```

```
## queremos todos los asegurados entre 30 y 60 años
ejemplo_11 <- insurance %>%
    filter(age >= 30 & age <= 60)
```

Tablas de frecuencias

Frecuencia absoluta

La **frecuencia absoluta** es el número de veces que se repite una observación.

Frecuencia relativa

La **frecuencia relativa** es el número total de veces que se repite una observación dividido por la cantidad total de observaciones.

Frecuencia porcentual

La **frecuencia porcentual** es la frecuencia relativa expresada en porcentaje. Para ello, se multiplica la frecuencia relativa por 100.

Frecuencia acumulada

La **frecuencia acumulada** se calcula como el sumatorio de la frecuencia de dicha observación y las anteriores. Puede ser tanto absoluta como relativa.

IMPORTANTE: En el caso de variables cuantitativas, será necesario definir intervalos de valores para generar una tabla de frecuencias. Esto es así porque:

- En el caso de variables discretas, el número de observaciones puede ser muy alto y la información quedaría poco agregada.
- En el caso de las variables continuas, dos observaciones nunca pueden ser iguales si consideramos un número infinito de decimales.