

Análisis de información de documentos semiestructurados / Parsing information from semistructured documents

MANEJO DE DATOS - PROYECTO FINAL - RODRIGO LÓPEZ
HERNÁNDEZ

Objetivo

Mostrar cómo realizar la recopilación, procesamiento y limpieza de datos de una base de datos climáticos provenientes de archivos de texto descargados desde un servidor FTP, para obtener un archivo `.csv` que sea manejable para cualquier análisis estadístico.

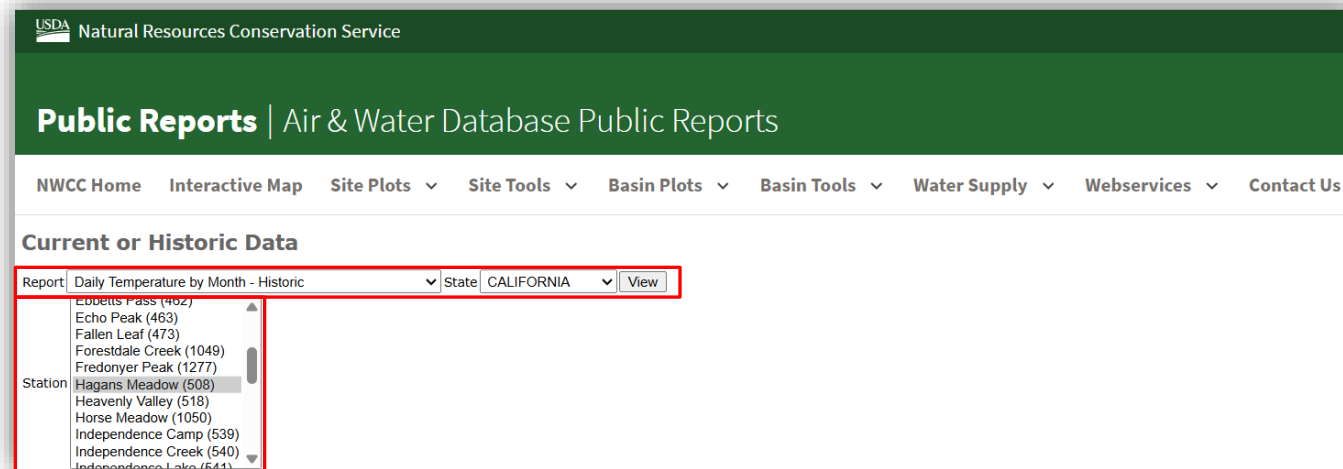
Línea de código para acceder al servidor FTP:

```
ftp <- ftp://ftp.wcc.nrcs.usda.gov/data/climate/table/temperature/history/california/
```

Se trabaja con los datos históricos diarios de la temperatura del estado de California.

Conflicto Principal

El servidor FTP, al cual el capítulo hace referencia, fue removido y no es posible consultarlo de la forma en la que el libro sugiere. Para resolver este problema, se consultó el sitio oficial de datos climáticos que ofrece el Servicio de Conservación de Recursos Naturales del Departamento de Agricultura de los Estados Unidos, particularmente los datos del estado de California.



The screenshot shows the USDA Natural Resources Conservation Service website. The header is green with the USDA logo and the text "Natural Resources Conservation Service". Below the header, there is a navigation bar with links: "NWCC Home", "Interactive Map", "Site Plots", "Site Tools", "Basin Plots", "Basin Tools", "Water Supply", "Webservices", and "Contact Us". The main content area is titled "Public Reports | Air & Water Database Public Reports". Underneath, there is a section "Current or Historic Data". This section contains a form with a "Report" dropdown menu set to "Daily Temperature by Month - Historic", a "State" dropdown menu set to "CALIFORNIA", and a "View" button. Below the form, there is a list of stations with a scrollbar. The stations listed are: Eubetts Pass (402), Echo Peak (463), Fallen Leaf (473), Forestdale Creek (1049), Fredonyer Peak (1277), Hagans Meadow (508), Heavenly Valley (518), Horse Meadow (1050), Independence Camp (539), Independence Creek (540), and Independence Lake (541).

El libro hace uso de la información de seis estaciones: ADIN MTN, INDEPENDENCE CAMP, SQUAW VALLEY, SPRATT CREEK, LEAVITT MEADOWS, POISON FLAT.

Conflicto Principal

Los datos disponibles en archivos de texto están semiestructurados, es decir, no se pueden incorporar directamente en R debido a su formato heterogéneo y poco fácil de manipular.

```
-----WARNING-----
#
# The data you have obtained from this automated Natural Resources Conservation Service
# database are subject to revision regardless of indicated Quality Assurance level.
# Data are released on condition that neither the NRCS nor the United States Government
# may be held liable for any damages resulting from its use.
#
# SNOTEL air temperature data contains a known bias. This bias is rooted in the sensor
# conversion equation and varies through the output range. Solutions are in development.
# For more information go to Air Temperature Bias Correction.
#
# Help and Tutorials: https://www.nrcs.usda.gov/sites/default/files/2023-03/Report%20Generator%20Help%20Guide.pdf
# Air Temperature Bias Correction: https://www.nrcs.usda.gov/wps/portal/wcc/home/snowClimateMonitoring/temperature/temperatureBiasCorrection/
#
# Support Contact: usdafpacbc@servicenowservices.com
#
-----
```

Encabezados sobre los datos

Sin filas o columnas delimitadas

[illegible]

Limpieza de Datos

```
# Creamos un vector de líneas.  
# Cada línea de texto del archivo "data1.txt" es un elemento del vector Vlíneas.
```

```
Vlineas <- readLines("data1.txt")  
cat("Los elementos de Vlineas es: ", length(Vlineas), "\n")  
  
print(head(Vlineas))
```

Los elementos de Vlineas es: 1272

```
[1] "#----- WARNING -----"  
[2] "# "  
[3] "# The data you have obtained from this automated Natural Resources Conservation Service "  
[4] "# database are subject to revision regardless of indicated Quality Assurance level. "  
[5] "# Data are released on condition that neither the NRCS nor the United States Government "  
[6] "# may be held liable for any damages resulting from its use. "
```

ature Minimum (degF),Air Temperature Maximum (degF),Air Temperature Average (degF),Air Temperature Minimum (degF),Air Temperature Maximum (degF),Air Temperature Average (degF),Air Temper
gF),Air Temperature Maximum (degF),Air Temperature Average (degF),Air Temperature Minimum (degF),Air Temperature Maximum (degF),Air Temperature Average (degF),Air Temperature Minimum (de
ire Maximum (degF),Air Temperature Average (degF),Air Temperature Minimum (degF),Air Temperature Maximum (degF),Air Temperature Average (degF),Air Temperature Minimum (degF),Air Temperatu
r,Air Temperature Average (degF),Air Temperature Minimum (degF),Air Temperature Maximum (degF),Air Temperature Average (degF),Air Temperature Minimum (degF),Air Temperature Maximum (deg
Average (degF),Air Temperature Minimum (degF),Air Temperature Maximum (degF),Air Temperature Average (degF),Air Temperature Minimum (degF),Air Temperature Maximum (degF),Air Temperatur
Air Temperature Minimum (degF),Air Temperature Maximum (degF),Air Temperature Average (degF),Air Temperature Minimum (degF),Air Temperature Maximum (degF),Air Temperatur

Limpieza de Datos

```
# Una vez que ya se limpiaron las líneas en blanco, vamos a preparar el archivo para ser manipulado. Vamos a extraer los años  
# del archivo de texto. Empleamos nuevamente la función "grepl( - , TRUE)" para extraer las líneas completas que coincidan con el patrón:  
# - \\d{4} (dígitos numéricos que tengan exactamente 4 posiciones, los años).
```

```
Danho <- grep("^\\d{4}", VlineasL, value = TRUE)
```

```
# Separamos los años por comas.
```

```
Dcomas <- strsplit(Danho, split = ",")  
print(head(Dcomas))
```

```
[[1]]  
[1] "1987" "01"    "32.0" "32.0" "32.0" "32.0" "32.0" "32.0" "32.0" "32.0"  
[11] "32.0" ""      ""      ""      "32.0" "32.0" "32.0" ""      ""      ""  
[21] ""      ""      ""      "32.0" "32.0" "32.0" ""      ""      ""      "33.6"  
[31] ""      "52.3" "37.6" "80.6" "56.8" "40.5" "75.4" "52.7"
```

```
[[2]]  
[1] "1987" "02"    "32.0" "32.0" "32.0" "32.0" "32.0" "32.0" "32.0" "32.0"  
[11] "32.0" ""      ""      ""      ""      ""      ""      ""      ""      ""  
[21] ""      ""      ""      ""      ""      ""      ""      ""      ""      ""  
[31] ""      ""      "39.6" "82.6" "59.4" "37.9" "71.8" "52.7"
```

Limpieza de Datos

```
# Con los datos separador por comas, ahora vamos a elaborar un dataframe (df) para una mejor manipulación de los datos.  
# Vamos a emplear la función "do.call( - )" para unir una lista de vectores en una sola matriz Así mismo, usaremos la  
# función rbind( - ) para unir por filas dichos vectores en una sola matriz  
# Después emplearemos as.data.frame( - ) para convertir la matriz en una tabla. "stringsAsFactors = FALSE" evita que las variables se vuelvan  
# categoricas.
```

```
df <- do.call(rbind, Dcomas)  
df <- as.data.frame(df, stringsAsFactors = FALSE)  
print(head(df))
```

```
      V1 V2  V3  V4  V5  V6  V7  V8  V9  V10 V11 V12 V13 V14  V15  V16  
1 1987 01 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0      32.0 32.0  
2 1987 02 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0      32.0 32.0  
3 1987 03 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0      32.0 32.0  
4 1987 04 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0      32.0 32.0  
5 1987 05 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0      32.0 32.0  
6 1987 06 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0      32.0 32.0  
      V17 V18 V19 V20  V21  V22  V23  V24  V25  V26 V27 V28 V29  V30 V31  V32  
1 32.0      32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0  
2 32.0      32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0  
3 32.0      32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0  
4 32.0      32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0  
5 32.0      32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0  
6 32.0      32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0 32.0  
      V33  V34  V35  V36  V37  V38  
1 37.6 80.6 56.8 40.5 75.4 52.7  
2 39.6 82.6 59.4 37.9 71.8 52.7  
3 40.1 81.1 59.5 37.0 70.2 51.8  
4 -36.4 84.7 59.9 33.8 66.6 49.3  
5 42.8 83.1 60.6 36.0 71.2 50.7  
6 42.3 82.8 59.9 33.6 71.6 49.5
```


Limpieza de Datos

```
# Para dar forma al data frame, damos nombre a las columnas.
```

```
colnames(df) <- c("Año", "Día", "Oct_Min", "Oct_Max", "Oct_Prom", "Nov_Min", "Nov_Max", "Nov_Prom",  
                 "Dic_Min", "Dic_Max", "Dic_Prom", "Ene_Min", "Ene_Max", "Ene_Prom", "Feb_Min",  
                 "Feb_Max", "Feb_Prom", "Mar_Min", "Mar_Max", "Mar_Prom", "Abr_Min", "Abr_Max",  
                 "Abr_Prom", "May_Min", "May_Max", "May_Prom", "Jun_Min", "Jun_Max", "Jun_Prom",  
                 "Jul_Min", "Jul_Max", "Jul_Prom", "Ago_Min", "Ago_Max", "Ago_Prom",  
                 "Sep_Min", "Sep_Max", "Sep_Prom")  
  
print(head(df))
```

	Año	Día	Oct_Min	Oct_Max	Oct_Prom	Nov_Min	Nov_Max	Nov_Prom	Dic_Min	Dic_Max
1	1987	01	32.0	32.0	32.0	32.0	32.0	32.0	32.0	32.0
2	1987	02	32.0	32.0	32.0	32.0	32.0	32.0	32.0	32.0
3	1987	03	32.0	32.0	32.0	32.0	32.0	32.0	32.0	32.0
4	1987	04	32.0	32.0	32.0	32.0	32.0	32.0	32.0	32.0
5	1987	05	32.0	32.0	32.0	32.0	32.0	32.0	32.0	32.0
6	1987	06	32.0	32.0	32.0					

	Dic_Prom	Ene_Min	Ene_Max	Ene_Prom	Feb_Min	Feb_Max	Feb_Prom	Mar_Min	Mar_Max
1	32.0				32.0	32.0	32.0		
2	32.0								
3	32.0								
4	32.0								
5	32.0								
6									

	Mar_Prom	Abr_Min	Abr_Max	Abr_Prom	May_Min	May_Max	May_Prom	Jun_Min	Jun_Max
1					32.0	32.0	32.0		
2									
3		32.0	32.0	32.0	32.0	32.0	32.0		
4					32.0	32.0	32.0		
5		32.0	32.0	32.0	32.0	32.0	32.0		
6		32.0	32.0	32.0					

	Jun_Prom	Jul_Min	Jul_Max	Jul_Prom	Ago_Min	Ago_Max	Ago_Prom	Sep_Min	Sep_Max
1		33.6		52.3	37.6	80.6	56.8	40.5	75.4
2					39.6	82.6	59.4	37.9	71.8
3					40.1	81.1	59.5	37.0	70.2
4					-36.4	84.7	59.9	33.8	66.6
5					42.8	83.1	60.6	36.0	71.2
6					42.3	82.8	59.9	33.6	71.6

	Sep_Prom
1	52.7
2	52.7
3	51.8
4	49.3
5	50.7
6	49.5

Gráfico de Datos

```
# Calcular promedio mensual para todos los años.
# Empleamos la función "grep( - )" para buscar las columnas que posean valores de promedios.
# Asignamos un valor de "na.rm = TRUE" para descartar los valores con "NA".

prom_mensual <- colMeans(df[, grep("_Prom", colnames(df))], na.rm = TRUE)
print(prom_mensual)

# Reorganizamos el vector de promedios dado que nuestros valores comienzan por el mes de octubre y es conveniente hacerlo con enero.
meses_ordenados <- c("Ene_Prom", "Feb_Prom", "Mar_Prom", "Abr_Prom", "May_Prom", "Jun_Prom",
                    "Jul_Prom", "Ago_Prom", "Sep_Prom", "Oct_Prom", "Nov_Prom", "Dic_Prom")
prom_mensual_ordenado <- prom_mensual[meses_ordenados]
print(prom_mensual_ordenado)

# Creamos un dataframe para graficar
meses <- 1:12
df_grafica <- data.frame(
  Mes = factor(month.abb, levels = month.abb),
  Temperatura_Promedio = prom_mensual_ordenado
)
```

Oct_Prom	Nov_Prom	Dic_Prom	Ene_Prom	Feb_Prom	Mar_Prom	Abr_Prom	May_Prom
39.98968	31.15052	25.31684	25.94289	26.53100	30.40391	34.73292	41.50418
Jun_Prom	Jul_Prom	Ago_Prom	Sep_Prom				
50.04781	56.18789	54.97421	171.12440				
Ene_Prom	Feb_Prom	Mar_Prom	Abr_Prom	May_Prom	Jun_Prom	Jul_Prom	Ago_Prom
25.94289	26.53100	30.40391	34.73292	41.50418	50.04781	56.18789	54.97421
Sep_Prom	Oct_Prom	Nov_Prom	Dic_Prom				
171.12440	39.98968	31.15052	25.31684				

Gráfico de Datos

```
# Graficamos los datos obtenidos
library(ggplot2)
ggplot(df_grafica, aes(x = Mes, y = Temperatura_Promedio)) +
  geom_line(group = 1, color = "steelblue", size = 1) +
  geom_point(color = "red", size = 2) +
  labs(title = "Temperatura Promedio por Mes",
       x = "Mes",
       y = "Temperatura Promedio (°F)")
```

Problema con Sep_Prom

Sep_Min	Sep_Max	Sep_Prom
37.4	73	51.8
NA	NA	2024
NA	NA	2025
NA	NA	2025

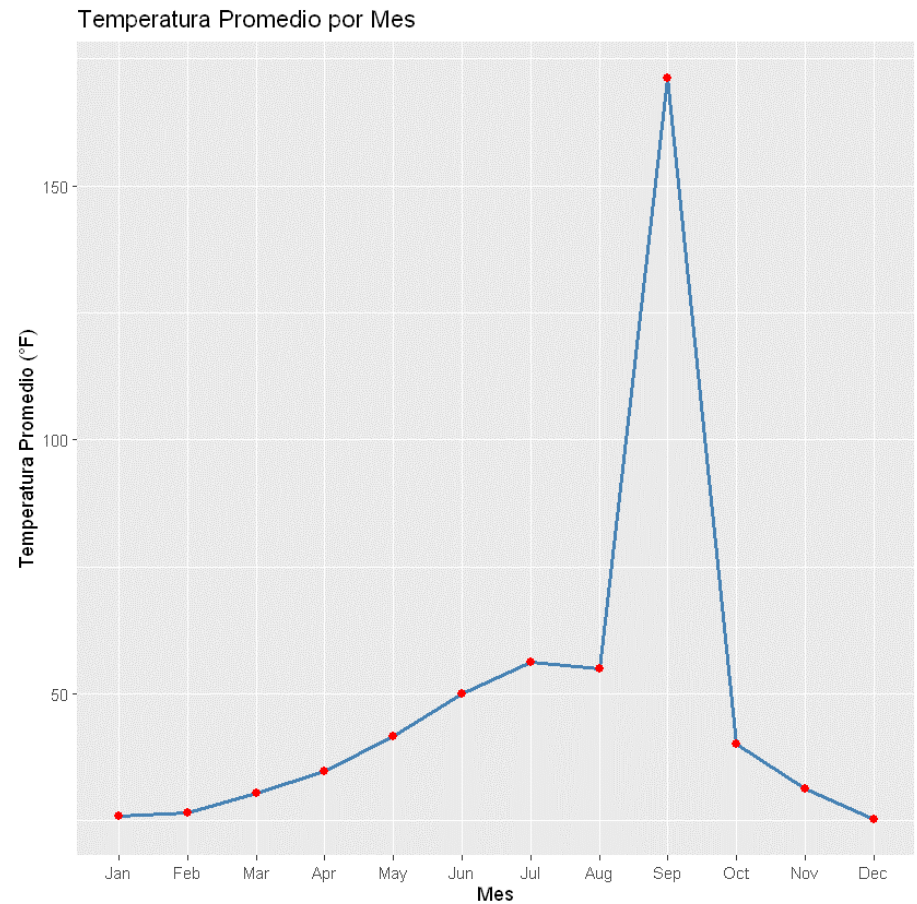


Gráfico de Datos

```
# Vemos ahora un punto atípico en la gráfica, esto se debe a un llenado erróneo de las celdas en el CSV generado.
# Al parecer, para el mes de septiembre, cuando las variables de temperatura mínima y máxima toman un valor de "NA",
# se autorellena con el valor de año correspondiente, es decir, 1990, 2020, 2025, entre otros. Por ello el valor del promedio
# es demasiado alto. Para esto vamos a descartar todos estos valores atípicos.
# Vamos entonces a detectar valores con exactamente 4 dígitos sin decimales en la columna Sep_Prom

filtro_4_digitos <- grepl("^\\d{4}$", df$Sep_Prom)
df$Sep_Prom[filtro_4_digitos] <- NA
print(df$Sep_Prom)
```

[illegible]

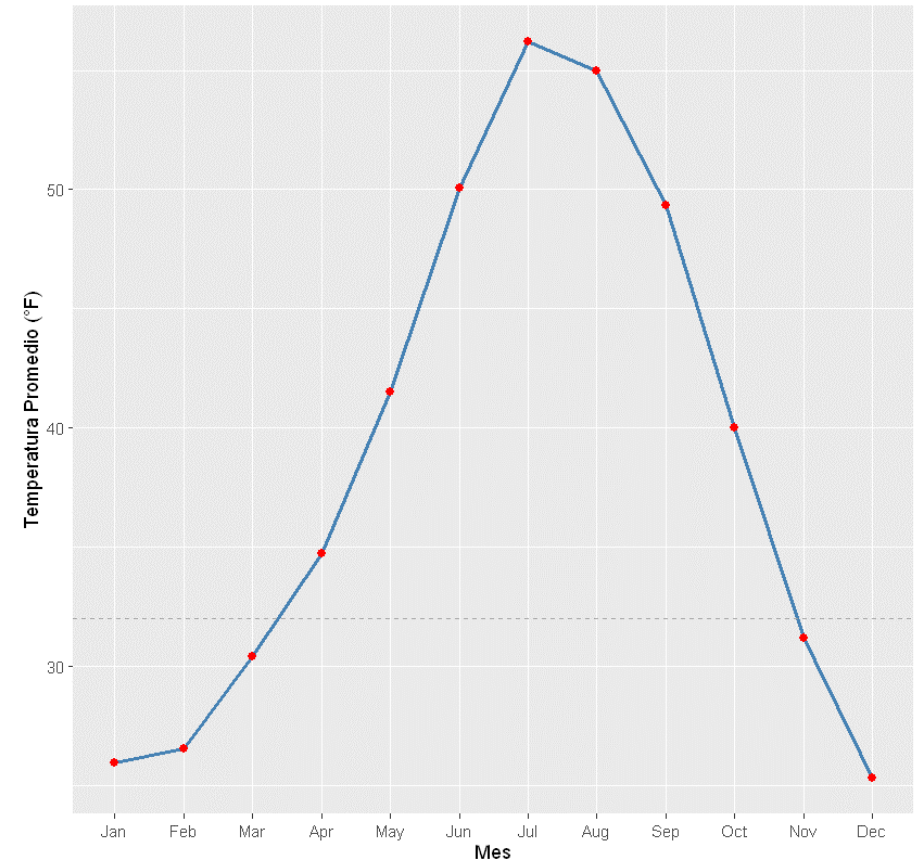
Gráfico de Datos

```
# Ejecutamos nuevamente las antepenultimas líneas de código para ver el gráfico final.
prom_mensual <- colMeans(df[, grep("_Prom", colnames(df))], na.rm = TRUE)
meses_ordenados <- c("Ene_Prom", "Feb_Prom", "Mar_Prom", "Abr_Prom", "May_Prom", "Jun_Prom",
                    "Jul_Prom", "Ago_Prom", "Sep_Prom", "Oct_Prom", "Nov_Prom", "Dic_Prom")
prom_mensual_ordenado <- prom_mensual[meses_ordenados]
meses <- 1:12
df_grafica <- data.frame(
  Mes = factor(month.abb, levels = month.abb),
  Temperatura_Promedio = prom_mensual_ordenado
)
library(ggplot2)
ggplot(df_grafica, aes(x = Mes, y = Temperatura_Promedio)) +
  geom_line(group = 1, color = "steelblue", size = 1) +
  geom_point(color = "red", size = 2) +
  geom_hline(yintercept = 32, linetype = "dashed", color = "darkgray") +
  labs(title = "Temperatura Promedio por Mes de la Estación: HAGAN'S MEADOW",
       x = "Mes",
       y = "Temperatura Promedio (°F)")
```

```
# Finalmente generamos el CSV final con todo el procesamiento de datos
```

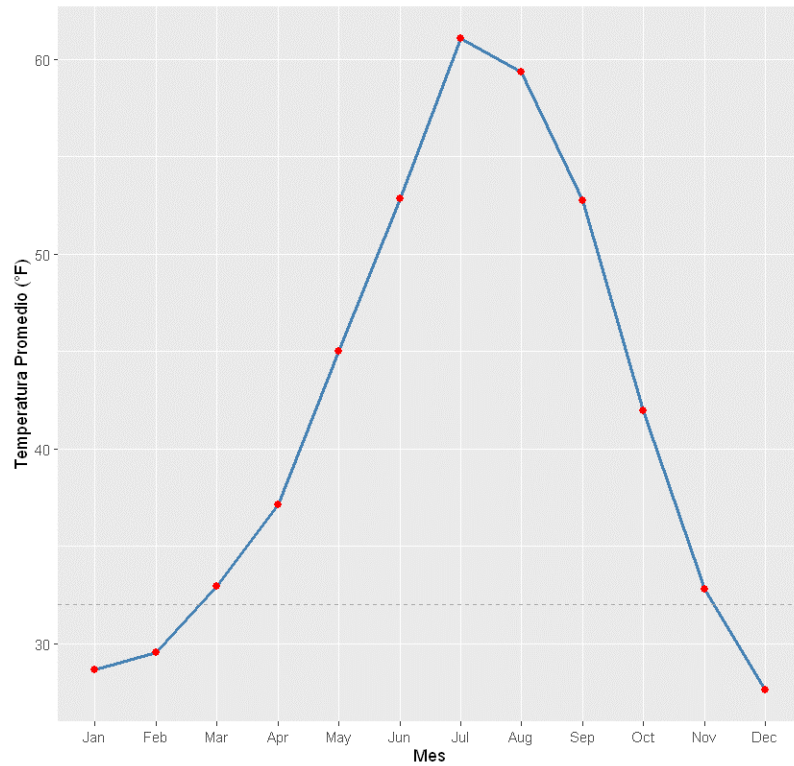
```
write.csv(df, "base_limpia_final1.csv", row.names = FALSE)
```

Temperatura Promedio por Mes de la Estación: HAGAN'S MEADOW

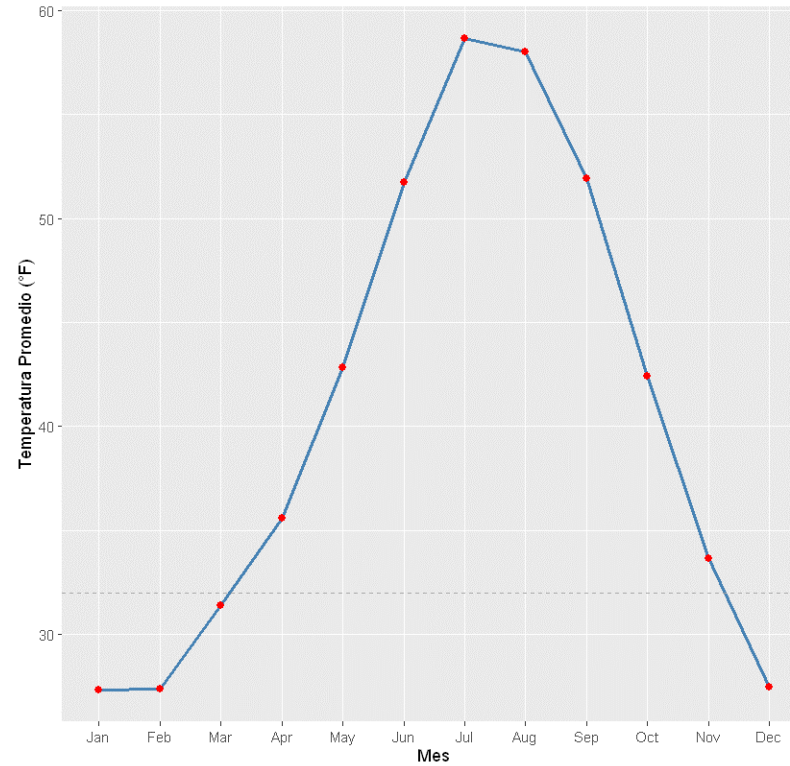


Resultados para distintas estaciones

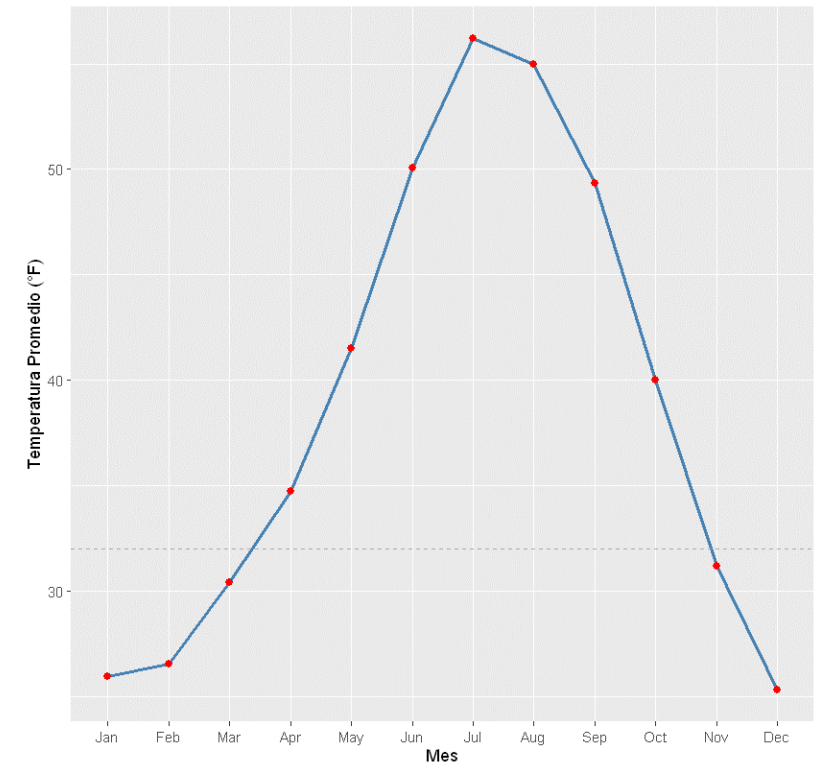
Temperatura Promedio por Mes de la Estación: ADIN MTN



Temperatura Promedio por Mes de la Estación: INDEPENDENCE CAMP

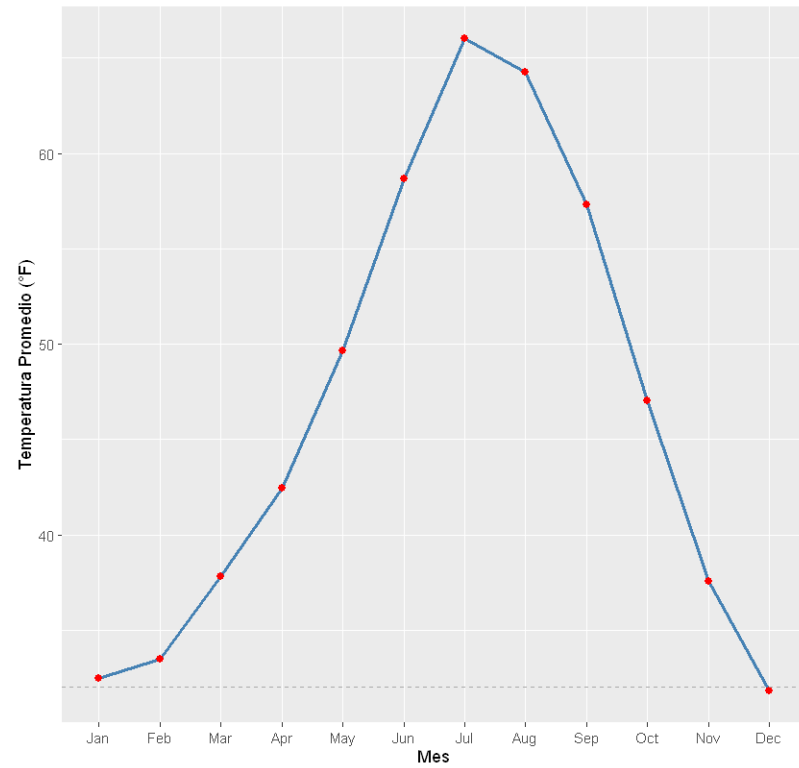


Temperatura Promedio por Mes de la Estación: HAGAN'S MEADOW

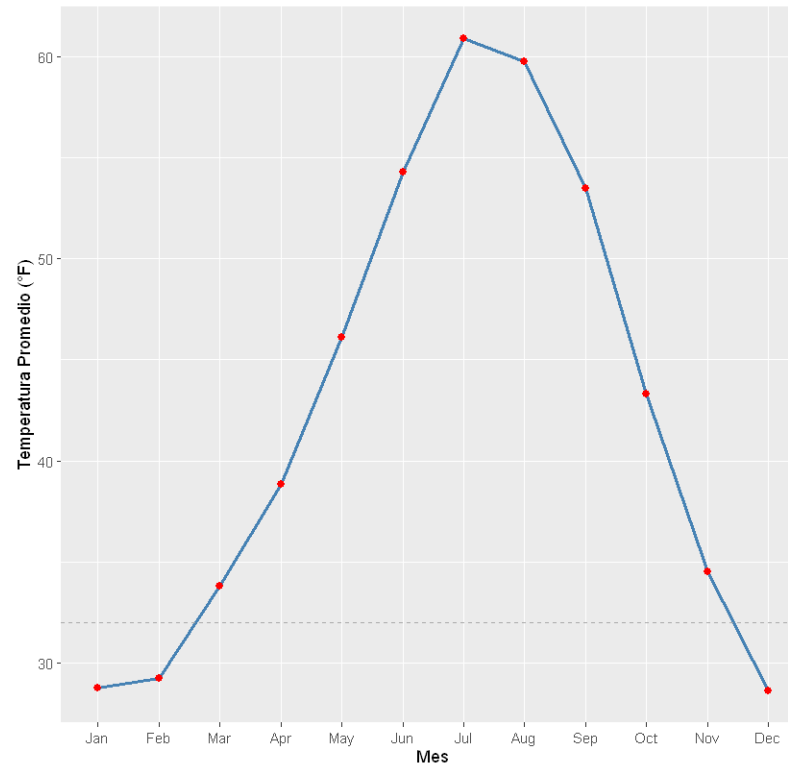


Resultados para distintas estaciones

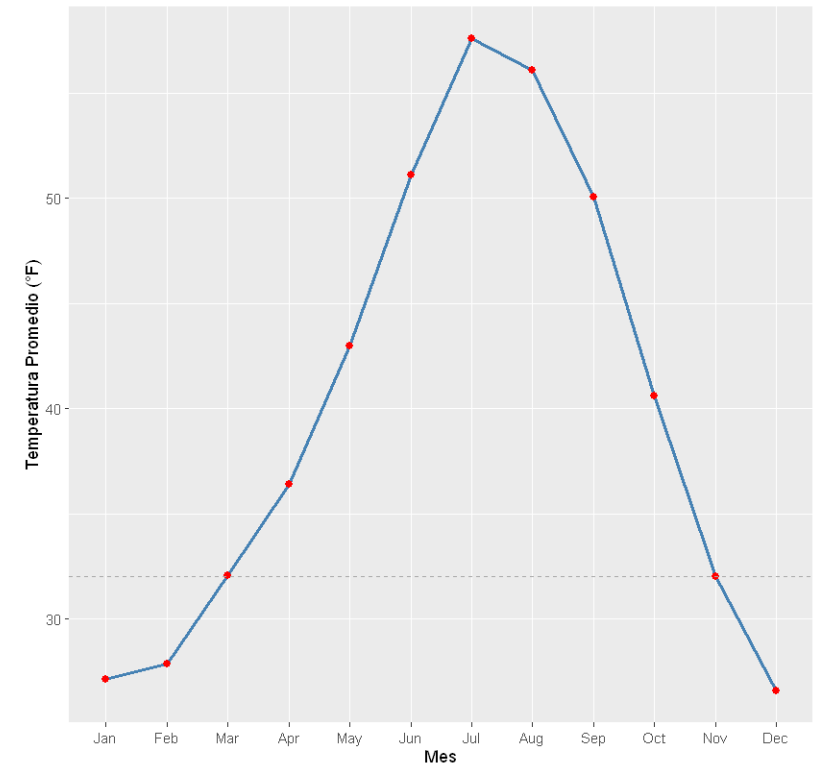
Temperatura Promedio por Mes de la Estación: SPRATT CREEK



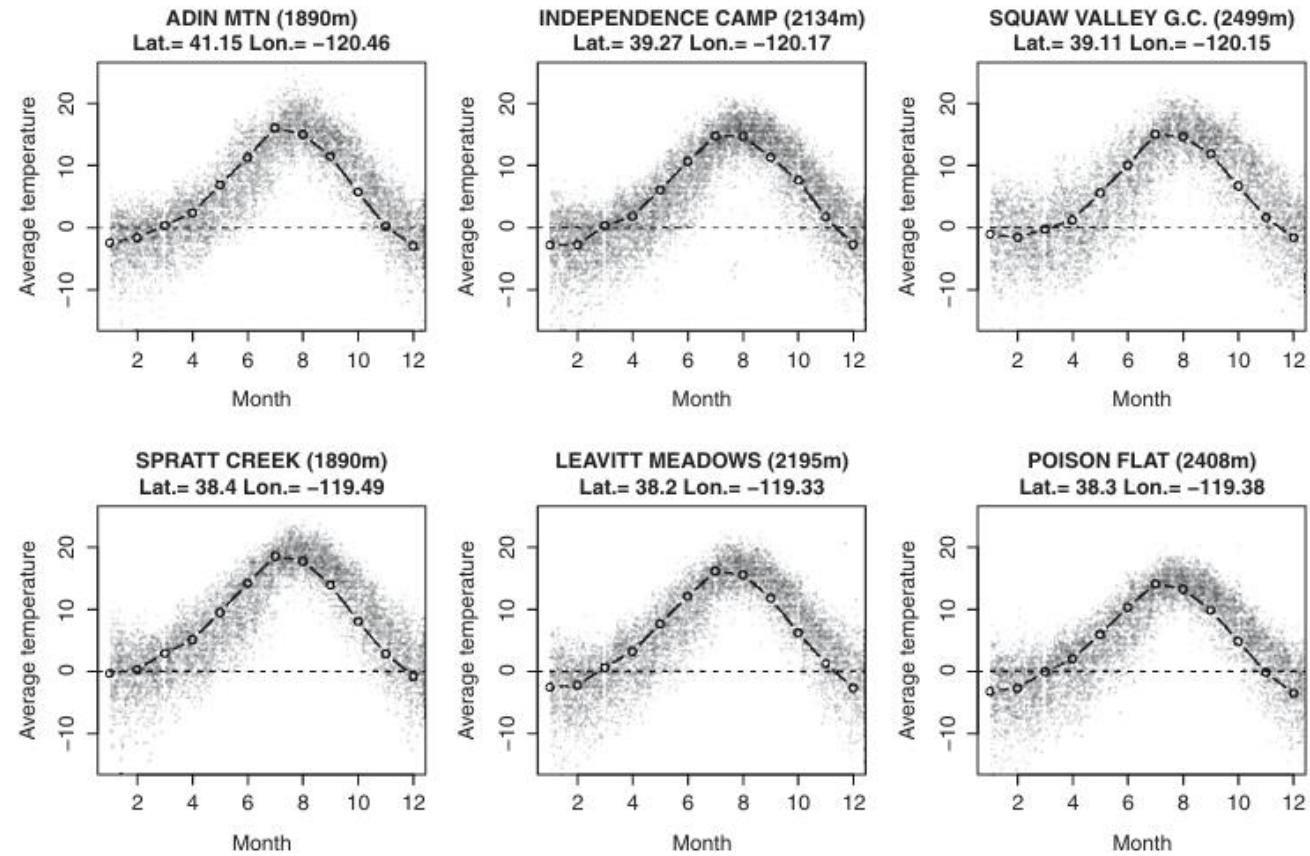
Temperatura Promedio por Mes de la Estación: LEAVITT MEADOWS



Temperatura Promedio por Mes de la Estación: POISON FLAT



Resultados del libro



Revisión de CSV's y Códigos

Resultados y Conclusiones

Podemos observar que derivado de la limpieza de datos se obtuvo un archivo .csv manejable para realizar cualquier análisis estadístico. Ahora los datos están ordenados de una forma en la que cualquier analista de datos puede emplearlos y realizar las estadísticas requeridas.

Se resalta la importancia de hacer manipulaciones de datos para visualizar su comportamiento y detectar posibles errores en la base.

En conclusión es importante tener buenos conocimientos sobre manipulaciones de datos ya que no siempre las bases de datos se encuentran en un estado óptimo para su uso. Así mismo, es importante saber localizar alternativas a las fuentes de datos, ya que como se muestra en este capítulo muchas veces los servidores dejan de estar disponibles.

¡Muchas Gracias por su Atención!