Series de Tiempo

Series de Tiempo en la APN

Una API REST es un servicio web que permite hacer consultas a una base de datos o aplicación en línea. Muchas APIs pueden usarse simplemente como una URL configurable / parametrizable en el navegador.

La Administración Pública Nacional dispone de APIs de datos en las que todos los organismos pueden publicar: https://apis.datos.gob.ar

Una de ellas permite consultar indicadores con evolución temporal de distintos ministerios (actualmente +20 mil series):

- **Documentación API**: https://apis.datos.gob.ar/series
- **Buscador web de series**: http://datos.gob.ar/series (permite buscar los ids de las series deseadas)

El buscador permite llevarse una URL a la API que descarga un CSV actualizado de los indicadores elegidos:

- **Tipo de cambio vendedor BNA**: http://apis.datos.gob.ar/series/api/series/?limit=1000&ids=168.1_T_CAMBIOR_D_0_0_26&format=csv
- **IPC Nacional. Nivel General**: http://apis.datos.gob.ar/series/api/series/?limit=1000&ids=148.3_INIVELNAL_DICI_M_26&format=csv
- **EMAE**: http://apis.datos.gob.ar/series/api/series/?limit=1000&ids=143.3_NO_PR_2004_A_21&format=csv

Como te podrás imaginar, `limit`, `ids` y `format` son algunos de esos parámetros que te permiten personalizar la consulta:

- `ids`: el parámetro más importante! Permite pedir una lista de series por id, separados por comas.
- `format`: puede ser "csv" o "json".
- `limit`: por default la API devuelve 100 filas, pero se puede extender hasta 1000.

En la mayoría de las APIs REST, los parámetros empiezan después del `?` y se separan por `&`.

Time series dataframes

Descargar CSVs de la API de series de tiempo

a. Descargar el IPC en un dataframe de R. Hint: `df = read.csv(*)`

```
R df = read.csv("http://apis.datos.gob.ar/series/api/series/?
limit=1000&ids=148.3_INIVELNAL_DICI_M_26&format=csv") tail(df)
```

	indice_tiempo	ipc_nivel_general_nacional
17	2018-04-01	136.7512
18	2018-05-01	139.5893
19	2018-06-01	144.8053
20	2018-07-01	149.2966
21	2018-08-01	155.1034
22	2018-09-01	165.2383

b. Descargar el Estimador Mensual de la Actividad Económica (EMAE) general, de la Construcción y de la Industria Manufacturera en el mismo dataframe. Hint: `ids=*,*,*.`

```
R df = read.csv("http://apis.datos.gob.ar/series/api/series/?
limit=1000&ids=143.3_NO_PR_2004_A_21,11.3_VMASD_2004_M_23,11.3_VMATC_2004_M_12&
```

```
tail(df)
```

	indice_tiempo	indice_serie_original	industria_manufacturera	construccion
171	2018-03-01	155.6816	134.2285	160.7265
172	2018-04-01	152.7067	130.3412	150.3414
173	2018-05-01	161.5906	140.0652	155.4314
174	2018-06-01	151.2358	128.5660	149.6231
175	2018-07-01	147.2753	130.8963	151.7200
176	2018-08-01	146.4046	132.6932	161.1219

c. Descargar el tipo de cambio mínimo, promedio y máximo mensual, usando la API (sin programar para eso en R). Hint: `&collapse=month` transforma la llamada en "mensual" y `&ids=*:min,*:avg` indica cómo agregar los valores de esas series.

```
R df = read.csv("http://apis.datos.gob.ar/series/api/series/?
limit=1000&ids=168.1_T_CAMBIOR_D_0_0_26:min,168.1_T_CAMBIOR_D_0_0_26:avg,168.1_
tail(df)
```

	indice_tiempo	tipo_cambio_bna_vendedor	tipo_cambio_bna_vendedor.1	tipo_cambio_bna_vendedor.2
44	2018-06-01	24.90	26.67433	28.85
45	2018-07-01	27.21	27.60765	28.85
46	2018-08-01	27.29	29.93758	37.60
47	2018-09-01	36.85	38.49607	41.25
48	2018-10-01	35.95	37.03032	39.60
49	2018-11-01	35.67	NA	35.67

d. Descargar el IPC, la inflación mensual y la inflación inter-anual en un mismo dataframe, usando la API (sin programar para eso en R) sólo desde 2017 en adelante. Hint: `&ids=*:percent_change,*:percent_change_a_year_ago` / `&start_date=*`.

```
R df = read.csv("http://apis.datos.gob.ar/series/api/series/?
limit=1000&ids=148.3_INIVELNAL_DICI_M_26,148.3_INIVELNAL_DICI_M_26:percent_char
tail(df)
```

	indice_tiempo	ipc_nivel_general_nacional	ipc_nivel_general_nacional.1	ipc_nivel_general_nacional.2
16	2018-04-01	136.7512	0.02739032	0.2549816
17	2018-05-01	139.5893	0.02075375	0.2629076
18	2018-06-01	144.8053	0.03736676	0.2946650
19	2018-07-01	149.2966	0.03101613	0.3120916
20	2018-08-01	155.1034	0.03889439	0.3442611
21	2018-09-01	165.2383	0.06534286	0.4054234

Convertir el dataframe en uno de series de tiempo

a. Plotear la relación entre la inflación mensual y las expectativas de inflación futura. Hint: `plot(df$*, df$*)`

```
R df = read.csv("http://apis.datos.gob.ar/series/api/series/?
limit=1000&ids=148.3_INIVELNAL_DICI_M_26:percent_change,430.1_MEDIANA_IP_12_M_(

R df
```

indice_tiempo	ipc_nivel_general_nacional	rem_ipc_nac_var_ia_t_12
2017-01-01	0.01585900	0.1970
2017-02-01	0.02067216	0.1950
2017-03-01	0.02374190	0.1840
2017-04-01	0.02655830	0.1751
2017-05-01	0.01434750	0.1700
2017-06-01	0.01192073	0.1700
2017-07-01	0.01732266	0.1710
2017-08-01	0.01403258	0.1720
2017-09-01	0.01898045	0.1690
2017-10-01	0.01514733	0.1730
2017-11-01	0.01375083	0.1750
2017-12-01	0.03141974	0.1740
2018-01-01	0.01757354	0.1860
2018-02-01	0.02419034	0.1764
2018-03-01	0.02341063	0.1780
2018-04-01	0.02739032	0.1820
2018-05-01	0.02075375	0.2224
2018-06-01	0.03736676	0.2415
2018-07-01	0.03101613	0.2370
2018-08-01	0.03889439	0.3150
2018-09-01	0.06534286	0.3190
2018-10-01	NA	0.3060

```
R plot(df$ipc_nivel_general_nacional, lag(df$rem_ipc_nac_var_ia_t_12,
0))
```

b. ¿Tal vez las expectativas de inflación futura inciden con un mes de retraso en la inflación mensual real? Para esto hace falta comparar una regresión entre las dos variables, contra una en la que las expectativas están 1 mes adelantadas. Esto requiere tratar las variables como series de tiempo.

```
b1. install.packages("dynlm"); library("dynlm") (Instala y carga una librería c
b2. library(zoo); df_ts = read.zoo(df, index = 1, tz = "", format = "%Y-%m-%d")
b3. Regresar la inflación mensual contra las expectativas de inflación futura.
b3. Regresar la inflación mensual contra las expectativas de inflación futura c
b4. Comparar los R2 de cada regresión. ¿Mejoró el % de variabilidad de la infla

R install.packages("dynlm"); library("dynlm")

Updating HTML index of packages in '.Library'
Making 'packages.html' ... done
Loading required package: zoo

Attaching package: ‘zoo’

The following objects are masked from ‘package:base’:

    as.Date, as.Date.numeric

R library(zoo); df_ts = read.zoo(df, index = 1, tz = "", format = "%Y-
%m-%d")

R summary(dynlm( df_ts$ipc_nivel_general_nacional ~
df_ts$rem_ipc_nac_var_ia_t_12 ))

Time series regression with "zoo" data:
Start = 2017-01-01, End = 2018-09-01

Call:
dynlm(formula = df_ts$ipc_nivel_general_nacional ~ df_ts$rem_ipc_nac_var_ia_t_1

Residuals:
      Min        1Q    Median        3Q       Max
-0.011481 -0.004117 -0.001782  0.003908  0.014066

Coefficients:
                             Estimate Std. Error t value Pr(>|t|)
(Intercept)                 -0.020611   0.007137  -2.888  0.00943 **
df_ts$rem_ipc_nac_var_ia_t_12  0.225352   0.035009   6.437 3.59e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.006989 on 19 degrees of freedom
  (0 observations deleted due to missingness)
Multiple R-squared:  0.6856,    Adjusted R-squared:  0.6691
F-statistic: 41.44 on 1 and 19 DF,  p-value: 3.594e-06

R summary(dynlm( df_ts$ipc_nivel_general_nacional ~
lag(df_ts$rem_ipc_nac_var_ia_t_12, -1) ))

Time series regression with "zoo" data:
Start = 2017-02-01, End = 2018-09-01

Call:
dynlm(formula = df_ts$ipc_nivel_general_nacional ~ lag(df_ts$rem_ipc_nac_var_ia
    -1))

Residuals:
       Min         1Q     Median         3Q        Max
-0.0087399 -0.0039750 -0.0003016  0.0028987  0.0123917

Coefficients:
```

```
                                    Estimate Std. Error t value Pr(>|t|)
(Intercept)                         -0.03552    0.00638  -5.568 2.76e-05 ***
lag(df_ts$rem_ipc_nac_var_ia_t_12, -1)  0.31170    0.03249   9.593 1.69e-08 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.005114 on 18 degrees of freedom
  (0 observations deleted due to missingness)
Multiple R-squared:  0.8364,    Adjusted R-squared:  0.8273
F-statistic: 92.03 on 1 and 18 DF,  p-value: 1.685e-08
```

LASSO

Sin embargo, la inflación es un fenómeno multicausal. Probablemente el mejor modelo explicativo sea uno que incluya muchas variables... La base de series de tiempo ofrece 20 mil! Si bien no tiene sentido probar con todas, habría que buscar un método que elija el mejor modelo al enfrentarse a muchas variables. (Ver tutorial de LASSO en R para más detalles: https://web.stanford.edu/~hastie/glmnet/glmnet_alpha.html)

a. Cargar en un dataframe de series de tiempo la inflación mensual, la tasa de interés, la variación mensual del tipo de cambio nominal, las expectativas de inflación futura y el tipo de cambio real multilateral, desde 2017. Hint: `df2 = read.csv("https://apis.datos.gob.ar/series/api/series/?limit=1000&ids=*:percent_change,*,*:percent_change,*,*&format=csv&start_date=20`

```
R df2 = read.csv("https://apis.datos.gob.ar/series/api/series/?
limit=1000&ids=148.3_INIVELNAL_DICI_M_26:percent_change,89.2_TS_INTE_PM_0_D_16,

R colnames(df2)
```

1. 'indice_tiempo'
2. 'ipc_nivel_general_nacional'
3. 'tasas_interes_pm'
4. 'tipo_cambio_bna_vendedor'
5. 'rem_ipc_nac_var_ia_t_12'
6. 'tipo_cambio_real_multilateral_actual'

```
R df2
```

indice_tiempo	ipc_nivel_general_nacional	tasas_interes_pm	tipo_cambio_bna_vendedor	rem_ipc_nac_var_ia_t_12	tipo_cambio_real_multilateral_actual
2017-01-01	0.01585900	24.75000	0.003517234	0.1970	90.65791
2017-02-01	0.02067216	24.75000	-0.020135332	0.1950	88.83677
2017-03-01	0.02374190	24.75000	-0.004182818	0.1840	86.54637
2017-04-01	0.02655830	25.70000	-0.010676656	0.1751	83.57614
2017-05-01	0.01434750	26.25000	0.023613498	0.1700	84.02443
2017-06-01	0.01192073	26.25000	0.023845717	0.1700	84.86319
2017-07-01	0.01732266	26.25000	0.066109731	0.1710	90.87417
2017-08-01	0.01403258	26.25000	0.014622869	0.1720	92.30178
2017-09-01	0.01898045	26.25000	-0.011230780	0.1690	90.44356
2017-10-01	0.01514733	26.58871	0.013274022	0.1730	88.88115
2017-11-01	0.01375083	28.51667	0.001905806	0.1750	87.22771
2017-12-01	0.03141974	28.75000	0.006644870	0.1740	86.64583
2018-01-01	0.01757354	28.02419	0.068057261	0.1860	93.11830
2018-02-01	0.02419034	27.25000	0.042691021	0.1764	95.78648
2018-03-01	0.02341063	27.25000	0.019872585	0.1780	95.08458
2018-04-01	0.02739032	27.65000	0.001074931	0.1820	91.52574
2018-05-01	0.02075375	39.15323	0.165391095	0.2224	100.62604
2018-06-01	0.03736676	40.00000	0.130246733	0.2415	108.41812
2018-07-01	0.03101613	40.00000	0.034989134	0.2370	108.56195
2018-08-01	0.03889439	44.03226	0.084394575	0.3150	112.35245
2018-09-01	0.06534286	60.50000	0.285877677	0.3190	135.58553
2018-10-01	NA	71.83500	-0.038075165	0.3060	126.68329

b. Usar el método lasso (librería `glmnet`) para encontrar el mejor modelo posible entre estas variables.

```
b1. install.packages("glmnet", repos = "http://cran.us.r-project.org"); library
b2. Remover las filas que tengan algún valor nulo. Hint: * = *[complete.cases(*
b3. Crear una matriz de predictores x (sin la variable a predecir). Hint: x = c
b4. Correr lasso usando cross validation. Hint: cvfit = cv.glmnet(x, y)
b5. Encontrar los coeficientes del modelo que minimiza el error de predicción.
```

```
R install.packages("glmnet", repos = "http://cran.us.r-project.org");
library(glmnet)

Updating HTML index of packages in '.Library'
Making 'packages.html' ... done
Loading required package: Matrix
Loading required package: foreach
Loaded glmnet 2.0-16

R df2 = df2[complete.cases(df2), ]

R df2
```

indice_tiempo	ipc_nivel_general_nacional	tasas_interes_pm	tipo_cambio_bna_vendedor	rem_ipc_nac_var_ia_t_12	tipo_cambio_real_multilateral_actual
2017-01-01	0.01585900	24.75000	0.003517234	0.1970	90.65791
2017-02-01	0.02067216	24.75000	-0.020135332	0.1950	88.83677
2017-03-01	0.02374190	24.75000	-0.004182818	0.1840	86.54637
2017-04-01	0.02655830	25.70000	-0.010676656	0.1751	83.57614
2017-05-01	0.01434750	26.25000	0.023613498	0.1700	84.02443
2017-06-01	0.01192073	26.25000	0.023845717	0.1700	84.86319
2017-07-01	0.01732266	26.25000	0.066109731	0.1710	90.87417
2017-08-01	0.01403258	26.25000	0.014622869	0.1720	92.30178
2017-09-01	0.01898045	26.25000	-0.011230780	0.1690	90.44356
2017-10-01	0.01514733	26.58871	0.013274022	0.1730	88.88115
2017-11-01	0.01375083	28.51667	0.001905806	0.1750	87.22771
2017-12-01	0.03141974	28.75000	0.006644870	0.1740	86.64583
2018-01-01	0.01757354	28.02419	0.068057261	0.1860	93.11830
2018-02-01	0.02419034	27.25000	0.042691021	0.1764	95.78648
2018-03-01	0.02341063	27.25000	0.019872585	0.1780	95.08458
2018-04-01	0.02739032	27.65000	0.001074931	0.1820	91.52574
2018-05-01	0.02075375	39.15323	0.165391095	0.2224	100.62604
2018-06-01	0.03736676	40.00000	0.130246733	0.2415	108.41812
2018-07-01	0.03101613	40.00000	0.034989134	0.2370	108.56195
2018-08-01	0.03889439	44.03226	0.084394575	0.3150	112.35245
2018-09-01	0.06534286	60.50000	0.285877677	0.3190	135.58553

```
R colnames(df2)
```

1. 'indice_tiempo'
2. 'ipc_nivel_general_nacional'
3. 'tasas_interes_pm'
4. 'tipo_cambio_bna_vendedor'
5. 'rem_ipc_nac_var_ia_t_12'
6. 'tipo_cambio_real_multilateral_actual'

```
R predictors = subset(df2, select=c("tasas_interes_pm",
"tipo_cambio_bna_vendedor", "rem_ipc_nac_var_ia_t_12")) x =
data.matrix(predictors) y = df2$ipc_nivel_general_nacional

R cvfit = cv.glmnet(x, y)

Warning message:
"Option grouped=FALSE enforced in cv.glmnet, since < 3 observations per fold"

R plot(cvfit)
```

```
R cvfit$lambda.min
```

0.000836805298067895

```
R cvfit$lambda.1se
```

0.00337819742599459

```
R coef(cvfit, s = "lambda.min")

4 x 1 sparse Matrix of class "dgCMatrix"
                                    1
(Intercept)              -0.0111626705
tasas_interes_pm          0.0009288312
tipo_cambio_bna_vendedor  .
rem_ipc_nac_var_ia_t_12   0.0338024639
```

```
R coef(cvfit, s = "lambda.1se")

4 x 1 sparse Matrix of class "dgCMatrix"
                                      1
(Intercept)               -0.0004582777
tasas_interes_pm           0.0007777370
tipo_cambio_bna_vendedor   .
rem_ipc_nac_var_ia_t_12    0.0034981108
```

c. Comparar la predicción del modelo con la realidad en un plot. Hint: plot(y, predict(cvfit, newx = *, s = "lambda.min"))

```
R y_predict = predict(cvfit, newx = x, s = "lambda.min")

R plot(y, y_predict)
```


d. Calcular el R2 de un ajuste lineal tradicional sobre este modelo y comparar los coeficientes.

Hint: `summary(lm(* ~ * + *))`

```
R summary(lm( df2$ipc_nivel_general_nacional ~ df2$tasas_interes_pm +
df2$rem_ipc_nac_var_ia_t_12 ))

Call:
lm(formula = df2$ipc_nivel_general_nacional ~ df2$tasas_interes_pm +
    df2$rem_ipc_nac_var_ia_t_12)

Residuals:
      Min        1Q    Median        3Q       Max
-0.012602 -0.003812 -0.001163  0.004489  0.010360

Coefficients:
                             Estimate Std. Error t value Pr(>|t|)
(Intercept)                -0.0146997  0.0068576  -2.144   0.0460 *
df2$tasas_interes_pm        0.0009765  0.0004098   2.383   0.0284 *
df2$rem_ipc_nac_var_ia_t_12 0.0441606  0.0822485   0.537   0.5979
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.00626 on 18 degrees of freedom
Multiple R-squared:  0.761, Adjusted R-squared:  0.7345
F-statistic: 28.66 on 2 and 18 DF,  p-value: 2.543e-06
```