*Anuario Jurídico y Económico Escurialense,* **LIII (2020) 245-258 / ISSN: 1133-3677**

# Un modelo de red neuronal para el Índice de Producción de la Construcción Total en España

*A neural network model for the Quarterly Index of Total Construction in Spain*

**Agustín ALONSO RODRIGUEZ**
Real Centro Universitario
"Escorial-María Cristina"
San Lorenzo del Escorial

**Resumen:** Con los datos del Índice de la Construcción Trimestral Total en España, se establece un modelo de red neuronal para visionar el futuro de la construcción desde el tercer trimestre de 2019 hasta el segundo cuatrimestre de 2022.

**Abstract**: Using the quarterly Index of Total Construction in Spain, a neural network model is presented to envision the future of construction in Spain, from the third quarter of 2019 to the second quarter of 2022.

**Palabras clave**: Índice Trimestral de la Construcción Total en España, modelos de redes neuronales, OECD, previsión, paquete estadístico R, paquete estadístico *forecast*.

**Keywords**: Total Quarterly Construction Index of Spain, neural netwoks models, OECD, forecasts, statistical package R, statistical package *forecast*.

**Sumario:**

**Recibido: septiembre de 2019.**
**Aceptado: noviembre de 2019.**

## I. LA INDUSTRIA DE LA CONSTRUCCIÓN EN ESPAÑA

La construcción en sentido amplio puede ser descrita como "la industria que engloba el conjunto de actividades que tiene como fin último la provisión de toda la gama de edificaciones e infraestructuras dentro de un territorio, generando espacio especializado para su utilización por las actividades productivas y la cobertura de las necesidades sociales" (Paloma Taltavull de la Paz, en *Lecciones de Economía Española*, 2017, p. 199). Las pinceladas que siguen están tomadas de esta obra.

El carácter motor de la actividad constructora en la economía española queda reflejado en sus aportaciones al valor añadido final, a la formación bruta de capital y al empleo. En relación al valor añadido, en promedio, su aportación se sitúa en torno al 8%, si bien entre 2005 y 2007 alcanzó el 11%. En lo que respecta a la formación bruta de capital, su aportación supera el 60% del total, convirtiéndose en el primer sector inversor de la economía. En cuanto a su capacidad generadora de empleo, se sitúa, en promedio, en torno al 8%, habiendo superado el 10% en los años de mayor crecimiento.

Como ilustración gráfica de la evolución de la actividad constructora, en la figura 1, se recogen los datos trimestrales desestacionalizados del *Índice de la Producción de la Construcción Total*, desde 1988, primer cuatrimestre, hasta 2019, segundo cuatrimestre (OECD: Production of Total Construction in Spain,[ESPROCONQISMEI], datos tomados del Federal Reserve Bank of St. Louis, septiembre 2019).

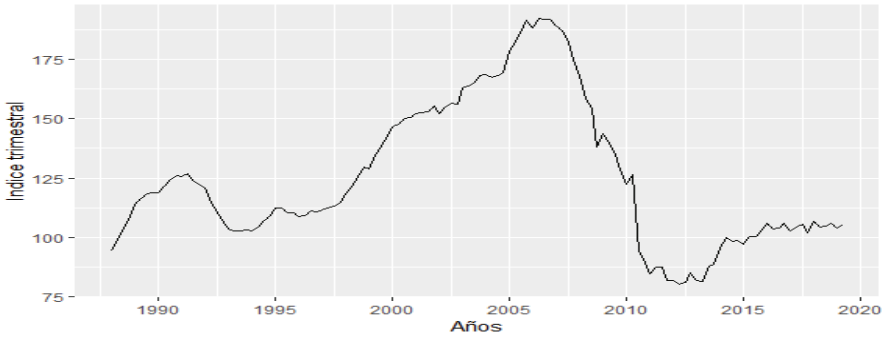


*Figura 1. Índice de la Producción de la Construcción en España: 1988 a 2019, 2015=100.*

La figura muestra claramente diferenciadas las fases de la actividad constructora en estos años. A mediado de los 90, y coincidiendo con la entrada de España en la Unión Europea, aparece una fuerte fase expansiva, que se mantiene hasta 2007, resultante de un ciclo a largo plazo que superó máximos históricos, coincidiendo con una fuerte inversión en infraestructuras. A partir de 2007, y a lo largo de siete años, comienza una fuerte recesión como resultado de la crisis financiera internacional, alcanzando mínimos históricos en actividad y producción. Desde 2014 se paraliza la caída, con muestras de estabilización, si bien con mayor debilidad que en otros sectores productivos. (Cf. Taltavull, P., Op. cit., pp. 201-202).

Este es el marco en el que se quiere establecer un *modelo de red neuronal* para la previsión de los valores futuros de este Índice trimestral.

## II. ANÁLISIS DE SERIES TEMPORALES MEDIANTE REDES NEURONALES

Bradley Efron y Trevor Hastie (2016) realizan el intento de explorar el desarrollo de la estadística en la época actual, en la que la potencia de cálculo de los ordenadores está abriendo horizontes imposibles de vislumbrar pocos años atrás.

Sea como sea, la forma en que nuestro cerebro procesa la información, se ha incorporado al ámbito de la estadística, y se ha ido plasmando en los denominados *modelos de redes neuronales*.

En su forma más simple, cabe describir un modelo de *red neuronal* como un conjunto, o red, de capas, denominadas técnicamente, *layers*. El layer inicial, o *input layer*, la capa intermedia, o *hidden layer* y el *output layer* o capa final. En cada *layer* existen los nudos o *neuronas*, encargadas del procesamiento de los datos. En teoría es posible la existencia de redes con múltiples layers, técnicamente designadas como *redes MLP: multilayer perceptron*. Los coeficientes vinculados a los nudos se denominan técnicamente ponderaciones, o *weights*.

En la figura 2 tenemos un ejemplo de red neuronal.

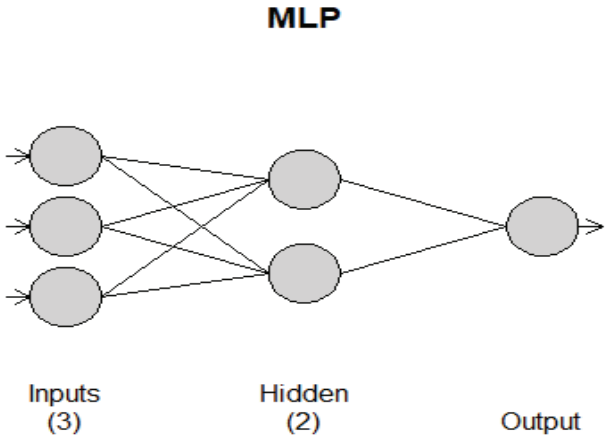


*Figura 2. Ejemplo de red neuronal: MLP*

En esta figura 2 tenemos una red o *network* con tres inputs, llamémoslos: X1, X2, X3, en el *layer input*; dos nudos o neuronas en el *hidden layer*, sean N1 y N2, y un resultado, en el *layer output*. En los nudos tienen lugar las combinaciones lineales entre los inputs así como su transformación mediante funciones no lineales, siendo estas transformaciones no lineales el output de los nudos que se trasmiten al *layer* siguiente.

Con el paso del tiempo han ido apareciendo nuevos tipos de redes. Desde el punto de vista de la predicción con series temporales, se ha ido imponiendo la red *feed-forward network*, en la que la información se procesa en una única dirección, de forma parecida a lo que se realiza en el modelo de regresión lineal.

Si desentrañamos un poco lo expuesto, se puede decir que al llegar los tres inputs al *hidden layer*, de nuestra figura 2, son combinados, para formar los *nudos* (neuronas) de acuerdo con

$$N_j = b_j + \sum_{i=1}^{3} w_{ij} X_i$$

siendo $b_j$ una constante, u ordenada en el origen, llamado el sesgo, *bias,* en esta literatura, y los $w_{ij}$ las ponderaciones de la red.

Es decir, para el primer nudo

$$N_1 = b_1 + w_{11}X_1 + w_{21}X_2 + w_{31}X_3$$

y para el segundo

$$N_2 = b_2 + w_{12}X_1 + w_{22}X_2 + w_{32}X_3$$

Estas transformaciones son procesadas luego por las funciones *no lineales* denominadas *funciones de activación*, entre las que figura la función *sigmoide*

$$f(n) = \frac{1}{1 + e^{-n}}$$

Esta función reduce el resultado de la transformación lineal anterior a un número real en el rango $[0,1]$, indicando el *cero* el bloqueo de la transfomación, y el *uno* el paso de la misma.

La última etapa del proceso consiste en promediar los outputs del *hidden layer* para generar el *output.*

Los parámetros $b_1, b_2,\ w_{11}, w_{21}, w_{31}, w_{12}, w_{22}, w_{32}$ son *deducidos*, son *aprendidos*, de los datos. Los parámetros son, además, controlados restringiendo sus valores mediante un parámetro de decrecimiento, cuyo valor más común es 0.1.

El proceso comienza dando a las ponderaciones valores aleatorios, que se van ajustando partir de los datos. En consecuencia, hay un elemento de aleatoriedad en las predicciones, por lo que la red es *entrenada* muchas veces utilizando diferentes puntos de arranque, y promediando los resultados.

Conviene poner de relieve la gran flexibilidad de los modelos de redes neuronales. De hecho, se puede demostrar, gracias al *Universal Convergence Theorem* que una red neuronal es capaz de aproximar cualquier relación entre el *input* y el *output* con una determinada precisión, dependiendo del número de neuronas. (Cf., Ramsundar, B. y Zadeh, R. B., *TensorFlow for Deep Learning*, 2018, p. 87) Además, trabajos teóricos han demostrado que un único *hidden layer* puede aproximar cualquier relación entre input y output dependiendo del número de nudos.

Recientemente existe gran interés en redes con múltiples *layers* intermedios, dando lugar al enfoque denominado *deep learning*. *Deep learning* parece un enfoque útil en temas de reconociemnto de voz, reconocimiento de imágenes, procesamiento del lenguage, pero no existe clara evidencia de que sea útil en temas de predicción.(Cf., Kourentzes, N. en *Principles of Business Forecasting*, 2017, p. 346)

## III. DATOS *TRAINING* Y DATOS *TEST*

A la hora de establecer la precisión de las predicciones es necesario utilizar nuevos datos, es decir, datos no utilizados para la estimación del modelo. Esto nos lleva a particionar la muestra en estudio.

La partición establece dos submuestras: la submuestra a utilizar para la estimación del modelo, técnicamente, los datos *training* , y la submuestra de validación del modelo, los datos *test*. Con los datos *training* se estiman los coeficientes del modelo, y con los datos *test* se evalúa la precisión de las predicciones.

Aunque no existe una norma establecida para la partición de la muestra, se suele dedicar a la submuestra *test* el 20% del total de la muestra. Ciertamente los datos *test* deben ser, al menos, tantos como el horizonte de la predicción a realizar.

Las ponderaciones $w_i$ tiene dos objetivos básicos: transformar y captar la relación entre las variables del modelo. Por ello, es aconsejable la normalización de los inputs. De esta manera, las ponderaciones captarán mejor las relaciones. (Cf., Kourentzes, N. en Op.cit., p. 345)

De ordinario, las predicciones un *paso adelante*, se realizan con los datos del bloque *training*, son los valores ajustados, *fitted values*, y las predicciones multiples *hacia adelante*, con los datos *test*.

## IV. REDES AUTOREGRESIVAS

Tratándose de series de tiempo se pueden utilizar como inputs para la red los últimos valores de la serie temporal en estudio. Tenemos así los modelos autoregresivos para la red: *NNAR(Neural Networks AR)*. Son modelos unidireccionales con un único *hidden layer*. (Cf. Hyndman, R. y Athanasopoulos, G., *Forecasting, Principles and practice*, 2018, p. 337.)

Siguiendo la terminología del paquete *forecast* del profesor Robert Hyndeman, y colaboradores, tenemos que un *NNAR(p,k)* hace referencia a un modelo de red, que toma las *p* últimas observaciones de la serie, para el *hidden layer*, en el que hay *k* nudos o neuronas. Así, un *NNAR(5,3)* describe una red con las 5 últimas observaciones de la serie como inputs para el *hidden layer* con 3 nudos.

Se puede demostrar que un *NNAR(p,0)* es equivalente a un *ARIMA(p,0,0)*, pero sin restricciones en los parámetros para asegurar la estacionariedad.

Con series estacionales es util incorporar como input la última observación de la serie temporal. Así, $NNAR(3,1,2)_{12}$ hace referencia a un modelo con inputs $y_{t-1}$, $y_{t-2}$, $y_{t-3}$ e $y_{t-12}$, con 2 nudos en el *hidden layer*. En general, un $NNAR(p,P,k)_{12}$, describe un modelo de red con *p* últimas observaciones de la serie temporal mensual, y *P* observaciones de la serie estacional, con *k* nudos en el *hidden layer*. Es decir, los inputs del hidden layer son: $y_{t-1}, y_{t-2} \cdots, y_{t-p}, y_{t-m}, y_{t-2m}, y_{t-m}, y_{t-2m}, \cdots, y_{t-Pm}$. Se puede demostrar que un $NNAR(p,P,0)_m$ es equivalente a un $ARIMA(p,0,0) \times (P,0,0)_m$ pero sin restricciones en los parametros para asegurar la estacionariedad.

## V. LOS MODELOS NNAR

Con ayuda del ya aludido paquete *forecast,* tenemos la función *NNAR* que estima un modelo de red neuronal, permitiendo su validación y su utilización para predicción de valores futuros.

Antes de nada, particionemos nuestra serie *Índice de la Construcción Total Trimestral española* en dos submuestras. La submuestra *training* desde el comienzo hasta el 2009, cuarto trimestre, y la submuestra *test*, a partir del 2010, ambas representadas en la figura 3.

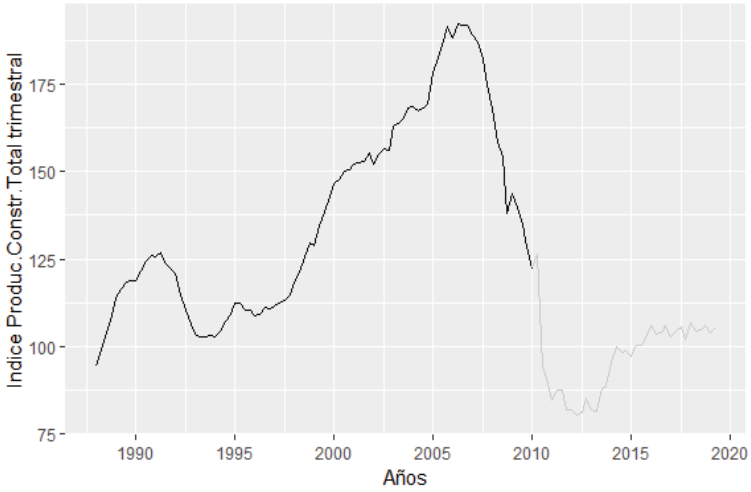


*Figura 3. Gráfico de las dos submuestas de la Serie Índice de la Construcción Total.*

Utilizamos la serie *train* para la estimación del modelo. La función *NNAR* permite estimar el modelo de forma automática o manual. Tratándose de series no estacionales, en el modo automático, el número de datos de la serie a incluir como inputs, se obtiene mediante el criterio *AIC* aplicado al correspondiente modelo lineal *AR(p)*. Con series estacionales, por defecto, $P = 1$, y $p$ se toma del modelo lineal óptimo ajustado a los datos desestacionalizados. En cuanto al número de nudos, si no se especifica, $k = (p + P + 1)/2$, redondeando al entero inmediato.

En nuestro caso:

```
modelo.train = nnetar(train, scale.inputs=TRUE)
modelo.train
## Series: train
## Model:  NNAR(1,1,2)[4]
## Call:   nnetar(y = train, scale.inputs = TRUE)
##
## Average of 20 networks, each of which is
## a 2-2-1 network with 9 weights
## options were - linear output units
##
## sigma^2 estimated as 7.356
```

Es decir, podemos escribir el modelo como:

$$y_t = f(y_{t-1}) + \epsilon_t$$

siendo $y_{t_1} = (y_{t-1}, y_{t-4})'$ el vector de *inputs*, con dos nudos en el *hidden layer*, y el término de error del modelo, la serie $\in_t$, que se supone *homoscedástica* y posiblemente con distribución de probabilidad *normal*.

El comportamiento de los residuos de este modelo, aparece en la figura 4.

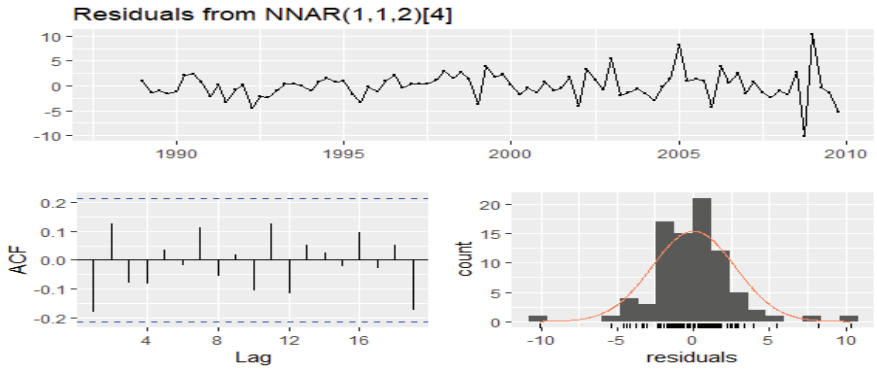


*Figura 4. Residuos del modelo estimado: modelo.train.*

Este comportamiento de los residuos es aceptable para validar el modelo estimado.

Pasemos a utilizar los datos *test* para la predicción. Estos datos utilizarán el modelo ya estimado.

```
modelo.test = nnetar(test,scale.inputs=T, model = modelo.train)
modelo.test
          ## Series: test
          ## Model:  NNAR(1,1,2)[4]
          ## Call:   nnetar(y = test, model = modelo.train,
          scale.inputs = T)
          ##
          ## Average of 20 networks, each of which is
          ## a 2-2-1 network with 9 weights
          ## options were - linear output units
          ##
          ## sigma^2 estimated as 20.31
```

siendo la predicción de 12 periodos, es decir, tres años, Cf. figura 5.

```
          fut.bis = forecast(modelo.test, h=12)
          fut.bis
          ##          Qtr1     Qtr2     Qtr3     Qtr4
          ## 2019                      106.2580 106.7674
          ## 2020 107.8117 108.6382 109.3944 110.1438
          ## 2021 110.7518 111.2548 111.6553 111.9382
          ## 2022 112.1194 112.2073
```

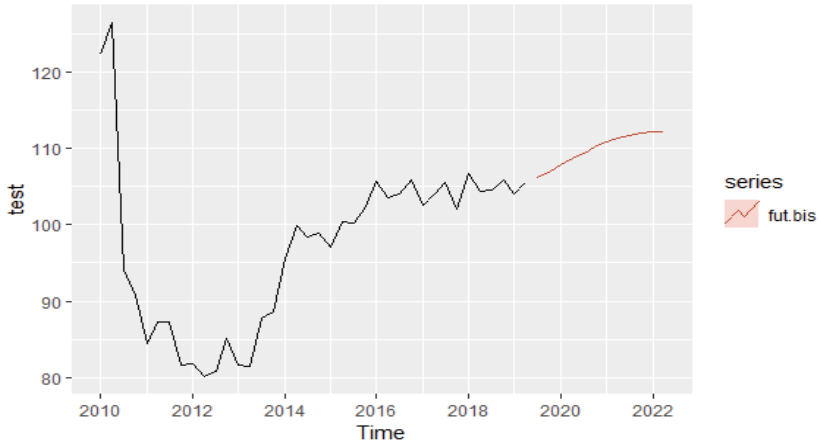


*Figura 5. Submuestra de datos Test y predicciones: fut.bis.*

Para mejor valorar estas cifras, recordemos los datos finales del *Índice*

```
tail(trim)
##          Qtr1     Qtr2     Qtr3     Qtr4
## 2018 106.6844 104.3232 104.4895 105.9195
## 2019 104.0572 105.4872
```

## VI. INTERVALO DE CONFIANZA PARA LAS PREDICCIONES NNAR

Los modelos de redes neuronales no tienen modelos estadísticos bien definidos, por lo que es necesario acudir a la simulación, para generar posibles sendas de valores futuros. Para ello, se pueden extraer valores futuros para la serie $e_t$, los residuos del modelo estimado, bien a partir de una distribución de probabilidad dada, o bien por muestrear repetidamente la serie de residuos muestrales ya obtenidos.

En la instrucción *forecast.nnetar* el argumento *npaths* controla el número de simulaciones, *1000* por defecto. También, por defecto, los errores son obtenidos de la distribución *normal*. El argumento *bootstrap* permite obtener errores *bootstrapped*, es decir aleatoriamente sacados de errores muestrales pasados.

La multitud de sendas generadas permiten alcanzar una buena visión de la distribución del término de error permitiendo establecer los intervalos para las predicciones con los modelos *nnetar*.

El paquete *forecast* permite realizar lo expuesto mediante la instrucción:

*forecast(modelo, PI=TRUE, h= ).*

Siendo *h* el número de periodos a predecir, y *PI: Prediction Intervals*.

Por defecto, como se ha indicado, la función *forecast* realiza *1000* simulaciones, con el término de error extraído de la distribución de probabilidad *normal*.

Tras lo expuesto, la predicción y sus bandas de confianza para las doce predicciones, así como su representación gráfica, viene dada en la figura 6

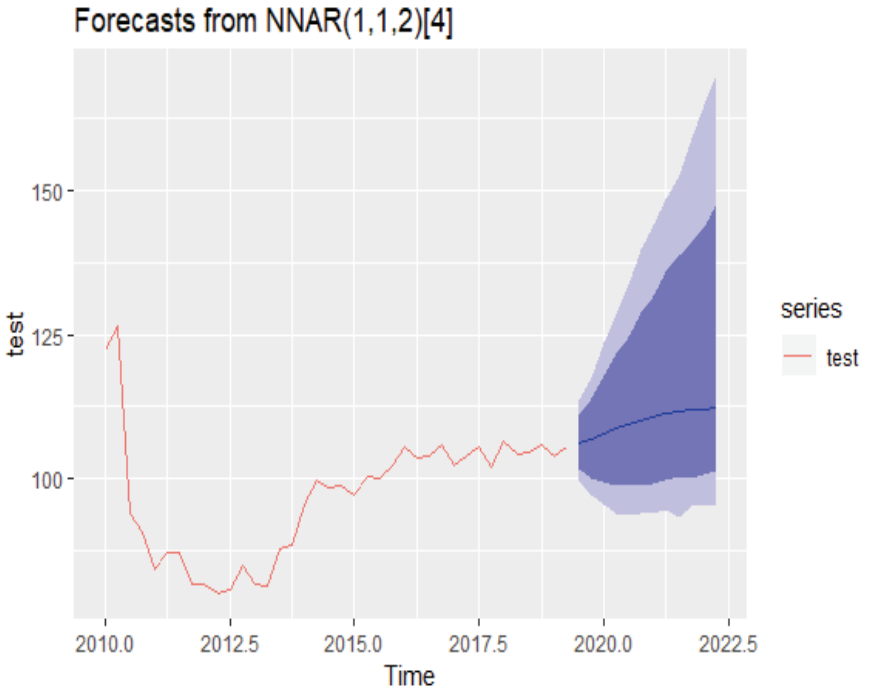


*Figura 6. Submuestra de datos Test y predicciones: fut.bis, con bandas de confianza.*

Siendo los valores concretos de la predicción y sus bandas,

```
fc = forecast(modelo.test,h=12,PI=T)
fc

##          Point Forecast     Lo 80    Hi 80    Lo 95    Hi 95
## 2019 Q3       106.2580 101.75901 110.8269 99.33694 113.3681
## 2019 Q4       106.7674 100.34782 114.0139 96.48542 117.4483
## 2020 Q1       107.8117 100.04840 117.5225 95.96090 122.9317
## 2020 Q2       108.6382  99.30653 121.1284 94.36400 128.3642
## 2020 Q3       109.3944  99.39986 124.8338 94.37757 133.0833
## 2020 Q4       110.1438 100.13541 128.4270 95.00729 136.9164
## 2021 Q1       110.7518 100.08811 131.6583 95.78464 141.5130
## 2021 Q2       111.2548 100.40555 135.4853 94.73458 146.8538
## 2021 Q3       111.6553 100.83729 138.8531 95.44096 151.1883
## 2021 Q4       111.9382 101.35893 142.4031 94.58236 156.4373
## 2022 Q1       112.1194 101.25449 145.1893 95.30050 160.6492
## 2022 Q2       112.2073 100.81090 149.2184 95.13605 167.5912
```

Para mejor interpretar estos resultados, los últimos valores del *Índice* son

```
tail(trim)
##          Qtr1     Qtr2     Qtr3     Qtr4
## 2018 106.6844 104.3232 104.4895 105.9195
## 2019 104.0572 105.4872
```

## VII. CONCLUSIONES

La previsión de los 12 valores del Índice, muestra un lento crecimiento del Índice de Construcción Total Trimestral en España. Las amplias bandas de confianza son muestra de la incertidumbre presente en la predicción.

Si pasamos a estimar la tasa porcentual de crecimiento de estos valores, el resultado muestra un comportamiento decreciente:

```
tasa = diff(log(fc))
tasa*100
0.4782301 0.9733742 0.7637066 0.6936120 0.6827405
0.5504414
0.4531615
0.3593117 0.2530701 0.1617272 0.0784386
```

Es decir, desde un 0.478% hasta un 0.078**%.**

## VIII. BIBLIOGRAFÍA


- CRAN, The *Comprehensive R Archive Network*. https://cloud.r-project.org/ Efron, B. y Hastie, T., *Computer Age Statistical Inference, Algorithms, Evidence, and Data Science*, Cambridge University Press, New York, 2016.

- García Delgado, J. L. y Myro, R., directores, *Lecciones de Economía Española*, 13ª edición, Civitas (Thomson Reuters), Editorial Aranzadi, Cizur Menor (Navarra), 2017.

- Hyndman, R. J. y Athanasopoulus, G., *Forecasting, Principles and Practice*, segunda edición, OTexts.org, 2018.

- Hyndman, R. J. y Athanosopoulos, G., *paquete fpp2*, en CRAN. Este paquete estadístico acompaña a la obra *Forecasting, Principles and Practice*, segunda edición.

- James, G., Witten, D,, Hastie, T. y Tibshirani, R., *An Introduction to Statistical Learning with Applications in R*, Springer, New York, 2013.

- Kourentzes, N., *package nnfor*, en CRAN. Este paquete estadístico complementa: Ord, K., Fildes, R. y Kourentzes, N., *Principles of Business Forecasting*, segunda edición, Wesssex Press Inc., New York, 2017.

- Matloff, N., *Statistical Regression and Classification, From Linear Models to Machine Learning*, CRC Press, Boca Raton, 2017.

- Matloff, N., *Probability and Statistics for Data Science, Math + R + Data*, CRC Press, Boca Raton, 2020.

- OECD, Organization for Economic Cooperation and Development, *Production of Total Construction in Spain*,[ESPROCONQISMEI], datos tomados del Federal Reserve Bank of St. Louis, https://fred.stlouisfed.org/series/ESPROCONQISMEI, septiembre 18, 2019.

- Ord, K., Fildes, R. y Kourentzes, N., *Principles of Business Forecasting*, segunda edición, Wessex Press Inc., New York, 2017.

- R Core Team (2019). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

- Ramsundar, B. y Zadeh, R. B., *TensorFlow for Deep Learning*, O'Reilly, Sebastopol, CA, 2018.

- Tsay, R. S., *Analysis of Financial Time Se*ries, J. Wiley and Sons, Inc., Hoboken, 2010.