SERIES TEMPORALES UNIVARIANTES.

Las series temporales univariantes son observaciones que se registran en sucesos temporales equidistantes en la cual representan un fenómeno a través del tiempo. Estas series son de gran interés por su capacidad para representar eventos repetitivos o definir tendencias en observaciones futuras, que pueden ser proyectadas a través de algoritmos de aprendizaje automático como lo son *Random Forest*, *Support Vector Machine, ARIMA y LSTM.* Para utilizar estos algoritmos de forma idónea es necesario realizar ciertas transformaciones al conjunto de datos de entrada.

PROBLEMA SUPERVISADO

La serie temporal univariante es de la forma:

Tiempo	Observación
t_1	O_1
t_2	O_2
.	.
.	.
.	.
t_n	O_n

UN RETRASO DE TIEMPO:

Para esta forma de datos es necesario proceder con una transformación para obtener un problema supervisado, es decir, para un registro etiqueta y sea éste representado a través de la variable o variables característica $x_1, x_2, \dots, x_l$. En estos términos se construye un conjunto de datos donde las características x sean las observaciones y en un retraso de tiempo.

$$x_t = y_{t-1}$$

El conjunto de datos para un problema supervisado con un retraso temporal es de la forma:

$\boldsymbol{x}$	$\boldsymbol{y}$
0	O_1
O_1	O_2
O_2	O_3
.	.
.	.
.	.
O_{n-1}	O_n

MULTIPLES RETRASOS DE TIEMPO:

Es posible representar un registro y con más de un retraso temporal. Esta transformación es ejecutada para incluir más información temporal, en lugar de

usar una sola observación, se usan m observaciones retrasadas en el tiempo para describir la variable etiqueta y en un instante de tiempo t.

$$y_t = \{x_1, x_2, \dots, x_m\}$$

$\boldsymbol{x_1}$	$\boldsymbol{x_2}$	$\cdots$	$\boldsymbol{x_m}$	$\boldsymbol{y}$
O_1	O_2	$\cdots$	O_m	O_{m+1}
O_2	O_3	...	O_{m+1}	O_{m+2}
$\vdots$	$\ddots$	...	$\vdots$	$\vdots$
O_p	O_{p+1}	...	O_{m+n-1}	O_{m+n}

Cabe señalar que estas transformaciones solamente son aplicables para las estructuras de datos de Random Forest, Support Vector Regressor y Long Short Therm Memory (una sola proyección). Para el modelo de Long Short Therm Memory para más de una proyección se utilizan m observaciones retrasadas en el tiempo y m observaciones adelantadas.

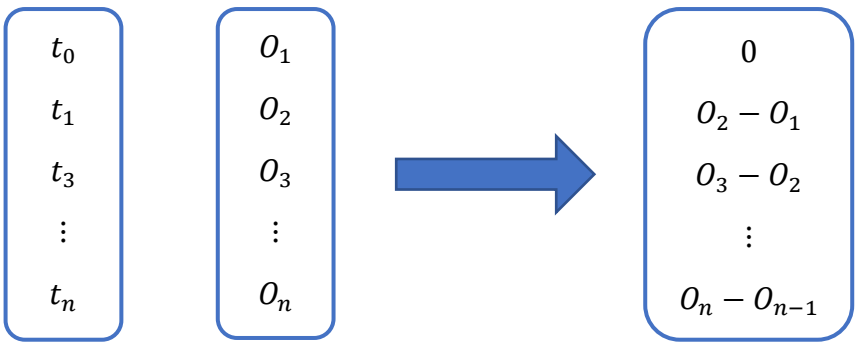
$$\{y_1, y_2, \dots, y_m\} = \{x_1, x_2, \dots, x_m\}$$

$\boldsymbol{x_1}$	$\boldsymbol{x_2}$	$\cdots$	$\boldsymbol{x_m}$	$\boldsymbol{y_{m+1}}$	$\boldsymbol{y_{m+2}}$	$\cdots$	$\boldsymbol{y_{m+m}}$
O_1	O_2	$\cdots$	O_m	O_{m+1}	O_{m+2}	$\cdots$	O_{m+m}
O_2	O_3	...	O_{m+1}	O_{m+2}	O_{m+3}	$\cdots$	O_{m+m+1}
$\vdots$	...	$\ddots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
O_p	O_{p+1}	...	O_{m+n-1}	O_{m+n}	...	...	O_{m+m+p}

SERIE DE TIEMPO DE DIFERENCIAS

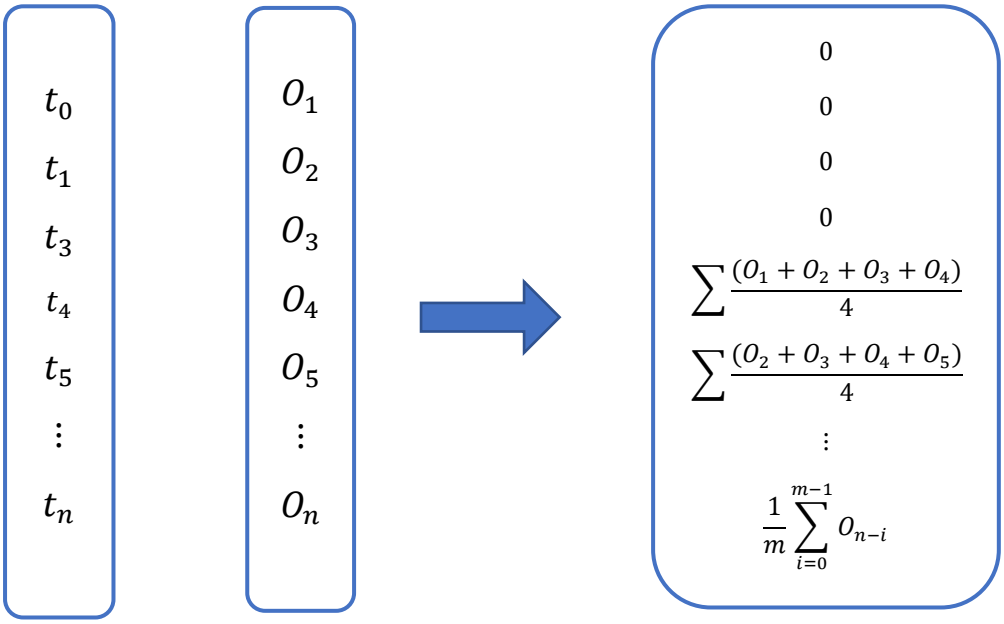
Las series temporales no estacionarias suponen un problema a la hora de intentar hacer proyecciones con los algoritmos de ML puesto que las observaciones en el tiempo no poseen un carácter repetitivo o predecible. Las series temporales de diferencias corresponde a la diferencia entre un valor y_t y y_{t-1} en el mismo instante de tiempo t. Al proceder con esta transformación se disminuye o se anula la tendencia en la serie temporal.

$$y_t = y_t - y_{t-1}$$



SERIE DE TIEMPO DE PROMEDIOS MÓVILES

La serie de tiempo de promedios móviles consiste en realizar promedios en las observaciones con respecto a una ventana móvil que recorre la totalidad de la serie temporal. La ventana móvil es un actor estabilizador de la serie temporal logrando suavizar la media y la varianza a lo largo de la serie



Registro de tiempo

Observaciones

Serie de tiempo de promedios

RANDOM FOREST

El modelo Random Forest es un algoritmo *bagging* (potenciado) que consiste en la generación de múltiples árboles de decisión basado en reglas y en el cual se promedia las proyecciones resultantes de los árboles generados. Este algoritmo permite realizar "regresiones" o "proyecciones" con respecto a un conjunto de datos de entrenamiento seleccionados aleatoriamente.

DEFINICIÓN:

"*Un bosque aleatorio es un clasificador que consiste en una colección de clasificadores estructurados por árboles* $\{h(x, \Theta_k),\ k = 1, \dots\}$, *donde* Θ_k *son vectores aleatorios independientes idénticamente distribuidos y cada árbol emite un voto unitario para la clase más popular en el vector de entrada* x"[1]

Convergencia de Random Forest:

De un conjunto de clasificadores $h_1(x), h_2(x), \dots, h_k(x)$ y con un conjunto de entrenamiento con la distribución del vector aleatorio Y, X, se define como la función margen como.

$$mg(X, Y) = av_k I(h_k(X) = Y) - \max_{j \neq Y} av_k I(h_k(X) = j\ )$$

Donde $I(.)$ es la función indicadora. El margen mide la medida en que el promedio de votos X, Y para la clase derecha supera el promedio de votos para cualquier otra clase.

$$PE^* = P_{X,Y}(mg(X,Y) < 0)$$

Donde los subíndices X, Yindican que la probabilidad está sobre el espacio X, Y. En un bosque aleatorio $h_k(X, \Theta_k) = h(X, \Theta_k)$ para un gran número de árboles se cumple la ley de los grandes números como:

A medida que el número de árboles aumenta, seguramente todas las secuencias $\Theta_1, \dots, PE^*$*converge a:*

$$P_{X,Y}(P_\Theta(h(X,\Theta) = Y) - \max_{j \neq Y} P_\Theta(h(X,\Theta) = j) < 0)$$

La ecuación anterior muestra el por qué random forest no presenta sobreajuste en más árboles agregados, pero produce una limitación en la generalización del error.

RANDOM FOREST PARA REGRESIÓN:

Random Forest para la regresión son conformados por árboles que crecen dependiendo de un vector aleatorio Θ de tal forma que el predictor de $h(x, \Theta)$ toma valores numéricos y las salidas son numéricas donde se asume que el conjunto de entrenamiento es independiente de la distribución del vector aleatorio Y, X. El error cuadrático medio de generalización para cualquier predictor numérico $h(x)$ es de la forma:

$$E_{X,Y}\big(Y - h(X)\big)^2$$

Las predicciones se realizan tomando el promedio de los k árboles $\{h(x, \Theta_k)\}$. Se define como error medio generalizado de un árbol como.

$$PE^*(tree) = E_\Theta E_{X,Y}\big(Y - h(X,\Theta)\big)^2$$

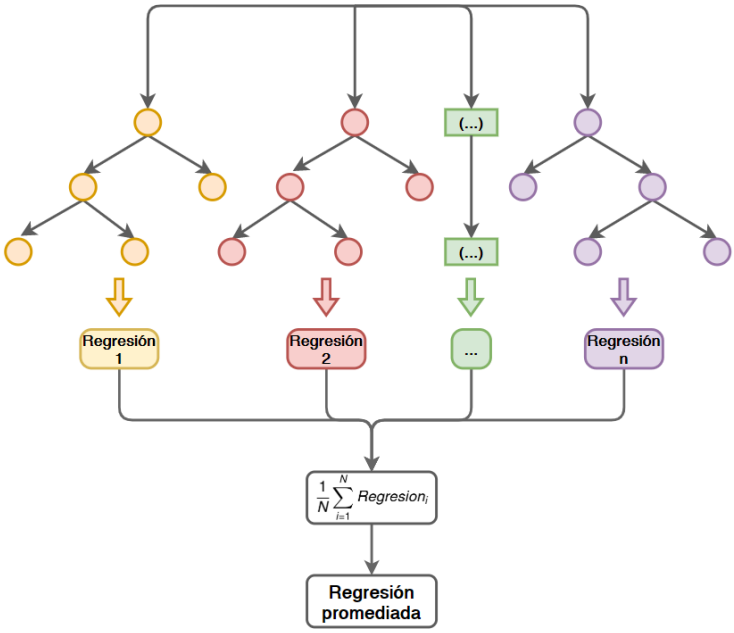


Ilustración 1. Regresor Random Forest.

CONDICIONES Y RESTRICCIONES DE USO:

- Se debe controlar la profundidad de los árboles ya que podría consumir espacio en memoria de forma considerable además de la complejidad al momento de entrenar el modelo.
- Los árboles usan las n características de forma automática. Se establece el parámetro *max_features = "auto".*
- Para construir un bosque de árboles para el conjunto de entrenamiento (X, Y) .
    - X: Es requisito ingresar los datos en forma de vector unidimensional o como matriz dispersa de la forma $(\#\ ejemplos, \#\ características)$.
    - Y : Es requisito ingresar los datos como un vector unidimensional o de la forma $(\#\ ejemplos, \#\ salidas)$.
- No controlar los parámetros para la creación de los árboles provoca poca generalización en los datos de entrenamiento.
- La complejidad en la construcción de un árbol binario es $O(n_{\,ejemplos} n_{características} \log(n_{ejemplos}))$ y el tiempo de consulta $O(\log(n_{ejemplos}))$.
-

REQUISITOS SOFTWARE (ACTUALMENTE EN GOOGLE COLABORATORY)

- Python 3.5 – 3.8
- Numpy 1.15.5
- Sklearn 0.22
- Arquitectura 64 bits
- Matplotlib 3.2.2

SUPPORT VECTOR REGRESSION:

El modelo Support Vector Machine (SVM) es un método ampliamente usado para problemas de clasificación y de regresión. Este modelo tiene la capacidad de realizar regresiones sobre un conjunto de datos de prueba de la forma $\{(x_1, z_1), \dots, (x_l,\ z_l)\}$ donde $x_i \in \mathbb{R}^n$es un vector característica y $z_i \epsilon \mathbb{R}^1$es la etiqueta de salida. Bajo los parámetros $C > 0$ y $\varepsilon > 0$, la forma estándar de un vector de regresión de soporte (SVR) es.[2]

$$\min_{w,b,\xi,\xi^*} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}\xi_i + C\sum_{i=1}^{l}\xi_i^*$$

El primer término de la ecuación anterior indica el problema de maximización al que se somete el vector o hiperplano w para obtener el máximo margen que abarque la mayoría de los datos. Por otro lado, los términos restantes se someten a un problema de minimización. Allí la constante C mantiene un umbral de equilibrio que describe la tolerancia a desviaciones mayores a los márgenes de soporte, así como ξ y ξ^* que indican los errores de los datos que están por fuera de los márgenes de soporte.[2]

$$w^T\phi(x_i) + b - z_i \leq \epsilon + \xi_i$$

$$z_i - w^T\phi(x_i) - b \leq \epsilon + \xi_i^*$$

$$\xi_i, \xi_i^* \geq 0, i = 1, \ldots, l$$

El problema doble es:

$$min_{\alpha,\alpha^*} \frac{1}{2}(\alpha - \alpha^*)^T Q(\alpha - \alpha^*) + \varepsilon e^T(\alpha + \alpha^*) - y^T(\alpha - \alpha^*)$$

Sujeto a:

$$e^T(\alpha - \alpha^*) = 0$$

$$0 \leq \alpha_i, \alpha^* \leq C, 1 = 1, \ldots, n$$

Donde e es un vector de unos, Q es una matriz semi - definida positiva de n por n. $Q_{ij} \equiv K(x_i, x_j) = \phi(x_i)^T\phi(x_j)$ es un kernel.

La predicción es de la forma:

$$\sum_{i=1}^{l} (\alpha_i - \alpha_i^*)K(x_i, x) + b$$

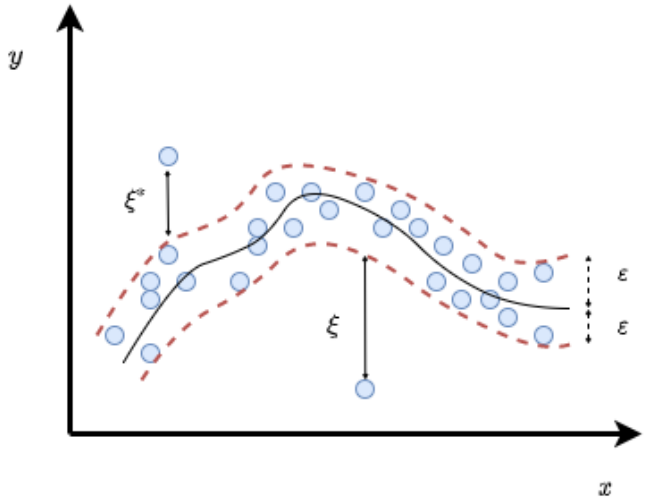


Ilustración 2. Support Vector Regressor

CODICIONES Y RESTRICCIONES DE USO:

- La complejidad en el tiempo de entrenamiento es mayor que $O(n^2)$. Poco escalable si se usan más de 10.000 ejemplos de entrenamiento. Los autores recomiendan usar otros modelos de regresión como *LinearSVR* o *SGDRegressor* dicho escenario.
- Altamente recomendable escalar los datos.
- Ajustar los hiperparámetros según el problema. Uso de kernels *Linea, Poly, rbf, sigmoid, precomputed* para diferentes distribuciones de datos.

- Hiperparámetro ϵ indica el margen de no penalización asociado a los márgenes de soporte.
- Para construir el modelo SVR para un conjunto de datos representados por (X, Y) .
    - X: Es requisito ingresar los datos en forma de vector unidimensional o como matriz dispersa de la forma $(\#\ ejemplos, \#\ características)$ o $(\#\ ejemplos, \#\ ejemplos)$.
- Y : Es requisito ingresar los datos como un vector unidimensional de la forma $(\#\ ejemplos)$

REQUISITOS SOFTWARE (ACTUALMENTE EN GOOGLE COLABORATORY)

- Python 3.5 – 3.8
- Numpy 1.15.5
- Sklearn 0.22
- Arquitectura 64 bits
- Matplotlib 3.2.2

ARIMA

El modelo integrado autorregresivo de medias móviles (ARIMA) es un modelo ampliamente usado para la predicción de observaciones x_t por medio de observaciones históricas $\{x_{t-1}, x_{t-2}, \dots, x_{t-p}\}$, donde p es el indicativo de la cantidad de pasos históricos tenidos en cuenta.

El operador autorregresivo de orden p, abreviado de la forma $AR(p)$ tiene la forma:

$$x_t = \phi_1 x_{t-1} + \phi_2 x_{t-2} + \cdots + \phi_p x_{t-p} + w_t$$

Donde x_t es estacionario, $w_t \sim wn(0, \sigma_w^2)$ y $\phi_1, \phi_2, \dots, \phi_p$ son constantes diferentes de cero. La ecuación anterior representa una observación x_t de la serie temporal univariante a través de las observaciones pasadas junto con las constantes $\phi_1, \phi_2, \dots, \phi_p$ y w_t. Cabe resaltar que si la media de x_t no es cero se reemplaza x_t por $x_t - \mu$, quedando de la siguiente forma.

$$x_t - \mu = \phi_1(x_{t-1} - \mu) + \phi_2(x_{t-2} - \mu) + \cdots + \phi_p(x_{t-p} - \mu) + w_t$$

Reduciendo como:

$$x_t = \alpha + \phi_1 x_{t-1} + \phi_2 x_{t-2} + \cdots + \phi_p x_{t-p} + w_t$$

$$\alpha = \mu(1 - \phi_1 - \cdots - \phi_p)$$

En adición el modelo ARIMA contiene el modelo de promedios móviles de orden q, abreviado como $MA(q)$.

$$x_t = w_t + \theta_1 w_{t-1} + \theta_2 w_{t-2} + \cdots + \theta_q w_{t-q}$$

Donde $w_t \sim wn\,(0, \sigma_w^2\,)$ y $\theta_1, \theta_2, \ldots, \theta_q$ son parámetros diferentes de cero. Este modelo es similar al modelo autorregresivo. A partir de las anteriores ecuaciones se construye el modelo autorregresivo de medias móviles (ARMA).

Una serie temporal de la forma $\{x_t; t = 0, \pm 1, \pm 2, \ldots\}$ es $ARMA(p, q)$ si es estacionaria y sí.

$$x_t = \phi_1 x_{t-1} + \phi_2 x_{t-2} + \cdots + \phi_p x_{t-p} + w_t + \theta_1 w_{t-1} + \theta_2 w_{t-2} + \cdots + \theta_q w_{t-q}$$

Donde $\phi_p \neq 0, \phi_q \neq 0$ y $\sigma_w^2 > 0$. Los parámetros p, q son llamados como órdenes autorregresivos y de medias móviles respectivamente. Si x_t no tiene media cero, se reescribe como:[3]

$$\alpha = \mu(\,1 - \phi_1 - \cdots - \phi_p)$$

$$x_t = \alpha + \phi_1 x_{t-1} + \phi_2 x_{t-2} + \cdots + \phi_p x_{t-p} + w_t + \theta_1 w_{t-1} + \theta_2 w_{t-2} + \cdots + \theta_q w_{t-q}$$

CONDICIONES Y RESTRICCIONES DE USO:

- Se debe especificar el orden (p, d, q) definido en la teoría de ARIMA para la generación del modelo. El parámetro autorregresivo $AR(p)$, el de diferencias (d) y el de promedios móviles $MA(q)$
- Los datos de entrenamiento y prueba corresponden a una estructura de arreglo unidimensional.
- Importante especificar la tendencia que tiene la serie temporal (constante o no constante).
- Los valores predichos son generados a través de un arreglo *numpy*.
- Los métodos para el entrenamiento del modelo son:
    - *Css-mle*
    - *Mle*
    - *Css*

  Si no se especifica el método automáticamente se asigna *css-mle* como método por defecto.

- La implementación de ARIMA de *statsmodel* indica que el parámetro $q < 5$, debido a que si p es mayor su tiempo de entrenamiento se verá afectado considerablemente.

REQUISITOS SOFTWARE

- Python >= 3.5
- Numpy >= 1.14
- Scipy >= 1.0
- Pandas>=0.21
- Arquitectura 64 bits
- Matplotlib >=2.2
- Entorno de ejecución (recomiendan los autores Jupyter notebook).

Long Short Therm Memory.

Es un tipo de red neuronal recurrente la cual posee celdas de memoria y compuertas para la adición o sustracción de elementos dependiendo de las activaciones en las compuertas de memoria. Dentro de las celdas de memoria se agrega una compuerta multiplicativa de entrada, esto para proteger el estado de memoria almacenado en la celda j de perturbaciones o entradas irrelevantes. A su vez también se agrega una compuerta multiplicativa de salida, esto para proteger las otras unidades de perturbaciones o irrelevancias en el estado de memoria en la celda j.

$$y^{in_j}(t) = f_{in_j}\left(net_{in_j}(t)\right)$$

$$y^{out_j}(t) = f_{out_j}\left(net_{out_j}(t)\right)$$

La entrada $y^{in_j}(t)$ se transforma por la unidad multiplicativa en la celda j por $f_{in_j}\left(net_{in_j}(t)\right)$ y la salida $y^{out_j}(t)$ por $f_{out_j}\left(net_{out_j}(t)\right)$. Donde.

$$net_{in_j}(t) = \sum_u w_{in_j u} y^u(t-1)$$

$$net_{out_j}(t) = \sum_u w_{out_j u} y^u(t-1)$$

Se generaliza como.

$$net_{c_j}(t) = \sum_u w_{c_j u} y^u(t-1)$$

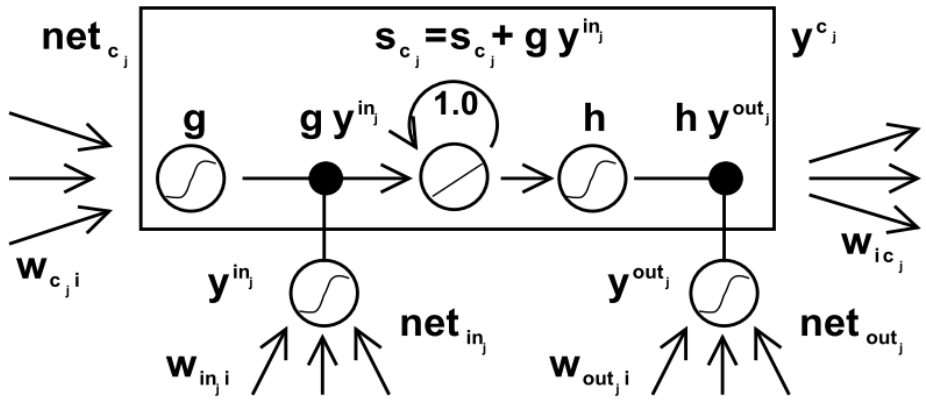
La metodología basada en “cintas transportadoras” y la compuertas multiplicativas de entrada y salida permiten que a lo largo de más de una celda de memoria se añada o se remueva información según su relevancia.[4]

$$y^{c_j}(t) = y^{out_j}(t) h(s_{c_j}, (t))$$

El estado interno de la celda j

$$s_{c_j}(0) = 0$$

$$s_{c_j}(t) = s_{c_j}(t-1) + y^{in_j}(t) g\left(net_{c_j}(t)\right); \;\; t > 0$$



CONDICIONES Y RESTRICCIONES DE USO:

- Se maximiza el rendimiento si se posee una GPU dedicada para el entrenamiento del modelo LSTM.
- Funciones de activación en las compuertas del bloque de memoria pueden ser especificados. Se usa $tanh$ como función de activación por defecto.
- La entrada de datos al modelo debe ser un tensor 3D de la forma $[batch, timesteps, feature]$.
- Siempre escalar los datos a una dimensión menor.

ERRORES:

Los modelos de proyecciones RF, SVR, ARIMA y LSTM deben ser evaluados según su capacidad de predicción de observaciones de una serie temporal, por esto se acude a métricas que permitan diferenciar cuantitativamente el rendimiento de los modelos.

En la literatura se encuentran muchas métricas las cuales representan, en diferentes cantidades, el error en las regresiones. Durante la ejecución del proyecto se implementaron seis diferentes métricas de error.

- **Error cuadrático medio (MSE):** Este error mide el promedio de los errores al cuadrado de un conjunto de observaciones predichas $\hat{y}$ y observaciones reales y.
- **Error absoluto medio (MAE):** Mide el error absoluto medio que existe entre las observaciones reales y y las observaciones predichas $\hat{y}$.
- **Error de porcentaje absoluto medio (MAPE):** Mide en forma de porcentual el error absoluto de y y de $\hat{y}$, en donde un error cercano a 1.0 se considera un modelo pobre en predicciones.
- **Error absoluto mediano :** A diferencia del MAE, el error absoluto medio se basa tomando las medianas de todas las diferencias absolutas entre y y $\hat{y}$.
- **Error máximo:** Genera el mayor error presente en el conjunto de predicciones $\hat{y}$ con respecto a y. Un error máximo cercano a 0.0 implica que el modelo predictor se ajusta muy bien a los datos de prueba.

- **R Cuadrado (r^2):** Mide el coeficiente de determinación de una regresión. Un coeficiente de determinación cercano a 1.0 implica una buena aproximación de las observaciones predichas a las observaciones reales.

METODOLOGÍA

Esta etapa es fundamental para entender los diferentes métodos y transformaciones que se le aplican a la serie temporal de entrada. Los algoritmos difieren en algunos procesos en consecuencia a los requisitos de entrada que así lo especifican sus documentaciones.

DIAGRAMA OPERACIONAL RANDOM FOREST:

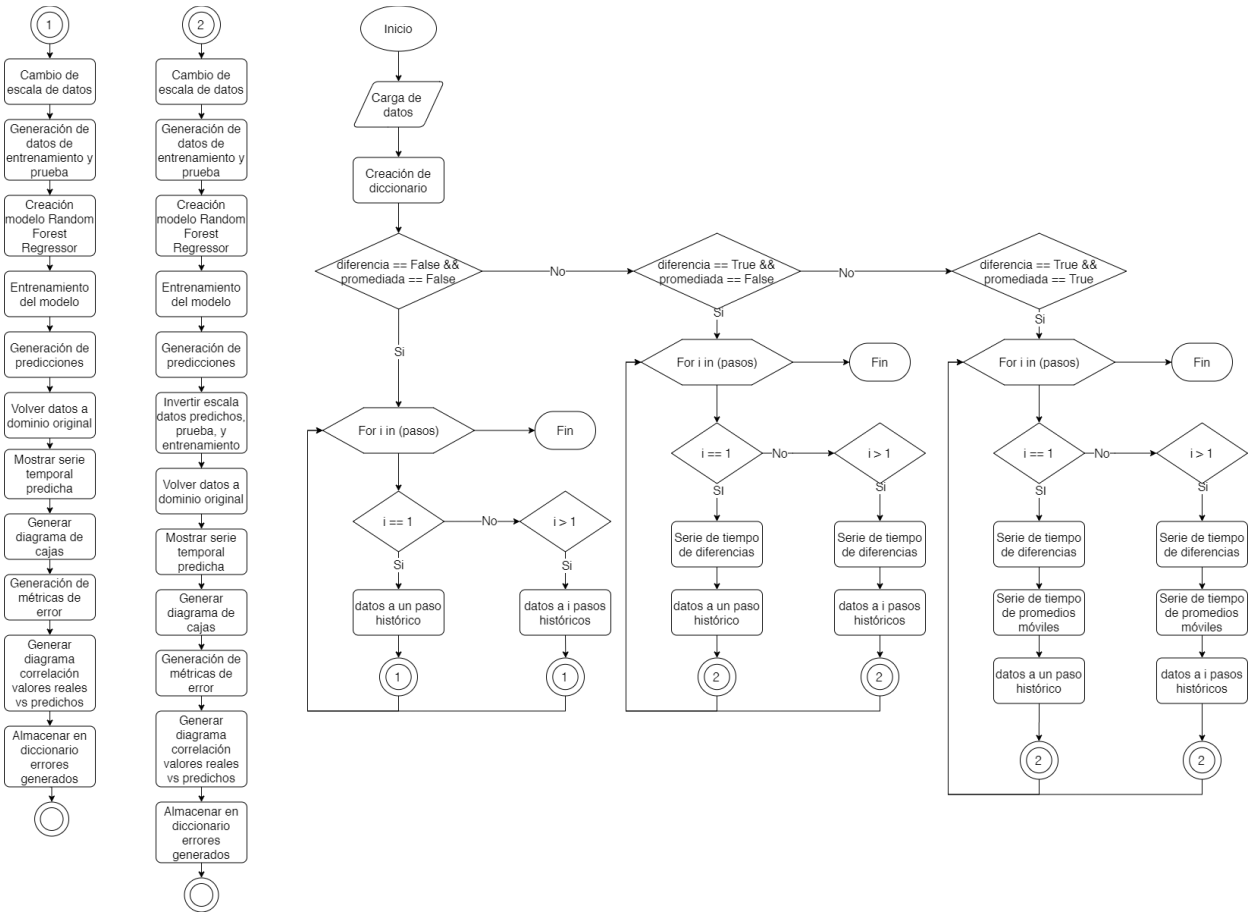


DIAGRAMA OPERACIONAL SUPPORT VECTOR REGRESSOR:

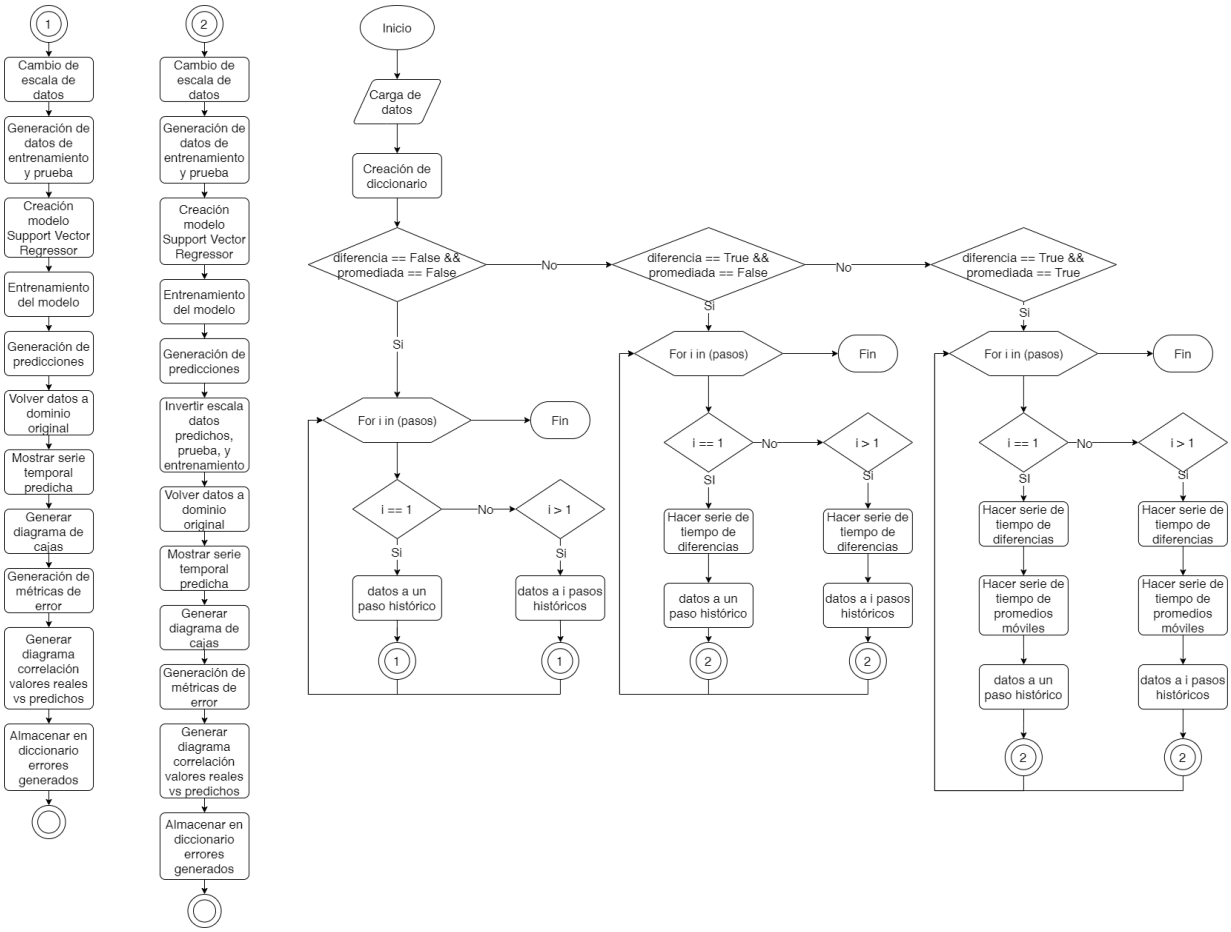


DIAGRAMA OPERACIONAL ARIMA:

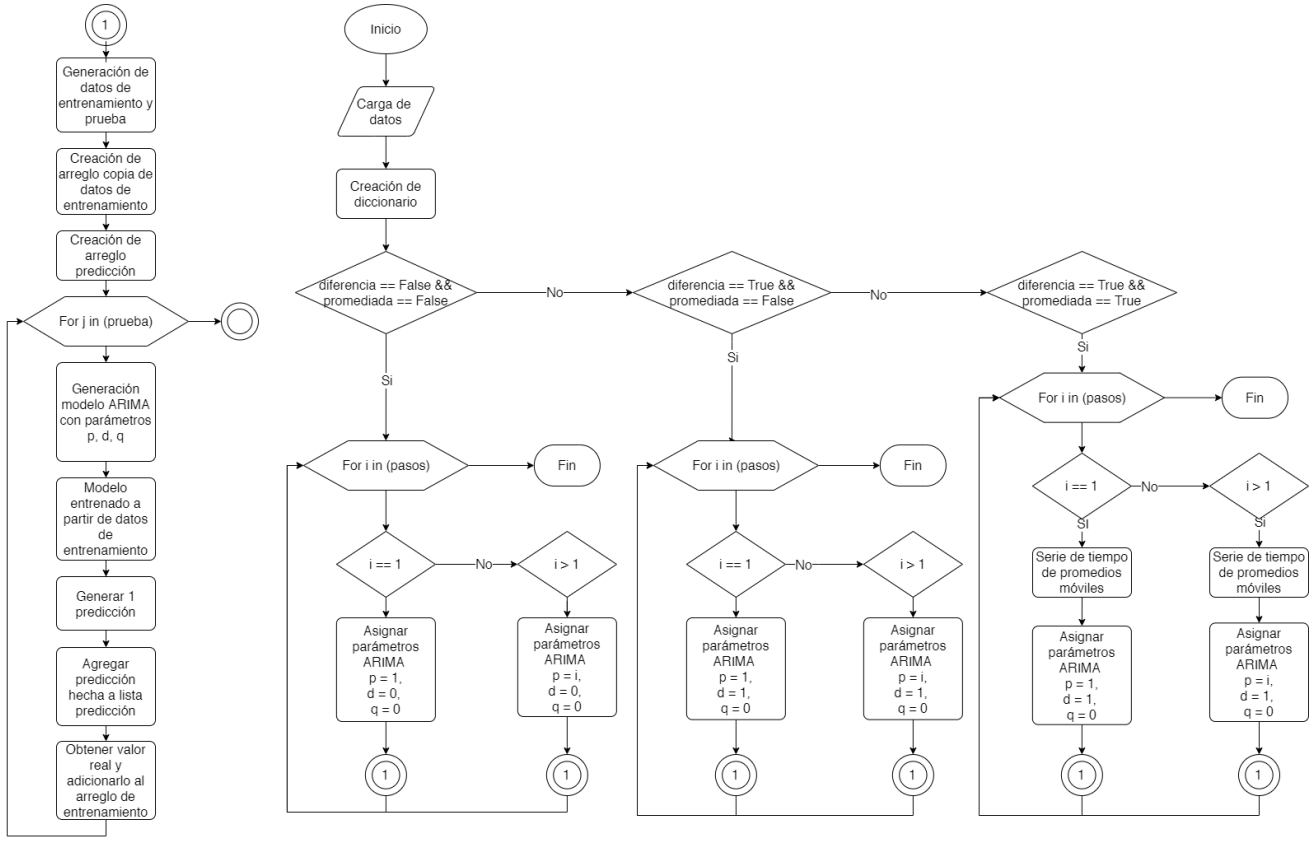
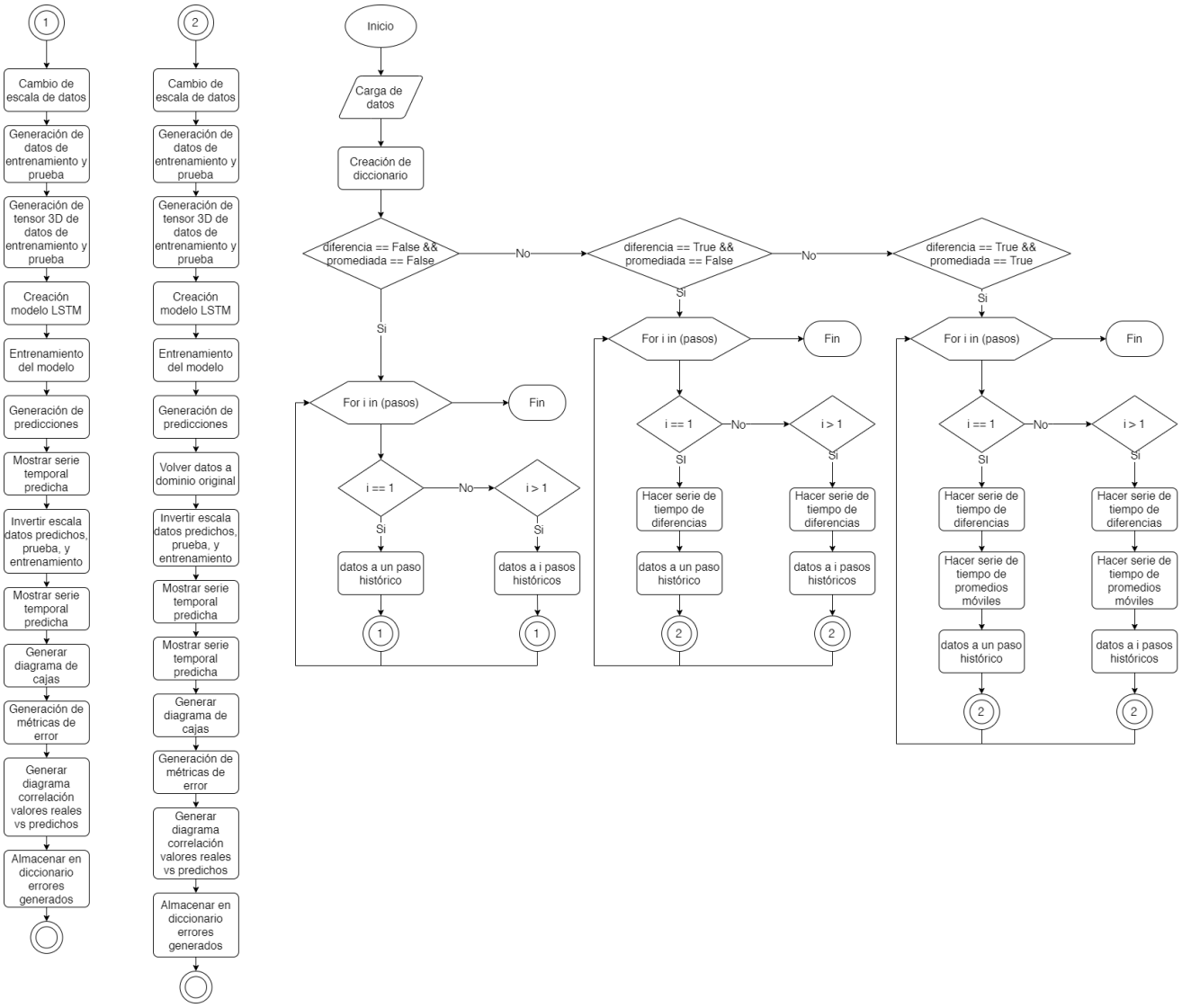


DIAGRAMA OPERACIONAL LSTM:



EXPERIMENTOS:

En esta etapa se ejecutarán los modelos de RF, SVR, ARIMA y LSTM para la misma cantidad de pasos históricos, esto quiere decir que para una cantidad de pasos n los modelos deberán generar una proyección $\hat{y}$. Como no se recomienda un parámetro $p > 5$ en el modelo ARIMA de *statsmodel*, entonces se ejecutarán los cuatro modelos con hasta tres pasos históricos, luego se aumentará los pasos históricos sólo para los modelos de RF, SVR y LSTM.

UNO A CUATRO PASOS HISTÓRICOS:

Se ejecutan los modelos junto con las transformaciones en los datos necesarias para evaluar su desempeño en diferentes tipos de datos, esto implica que la serie temporal univariante será ingresada a los modelos de la siguiente forma:

- Serie de tiempo sin diferencias: Aquí la serie temporal no sufre ningún tipo de transformación para la supresión de la no estacionalidad, sin embargo si sufre un cambio de escala para los algoritmos de RF, SVR y LSTM.
- Serie de tiempo de diferencias: Como indica la teoría de series de tiempo, para la supresión de la tendencia de una serie temporal se recurre a la diferencia entre observaciones. Esta suposición indica que la tendencia en el tiempo t es próxima a la existente en el tiempo $t-1$. Por esta razón la serie temporal resultante es libre de tendencia y es estacionaria.
- Serie de tiempo de diferencias y promedios móviles: Se aplican los mismos conceptos y transformaciones que el anterior ítem, con la diferencia que esta serie temporal es el resultado de la aplicación de promedios móviles a la serie temporal sin transformaciones. La aplicabilidad de promedios móviles permite el amortiguamiento de picos o valores extremos que pudiesen existir en los datos originales.

MÁS DE CUATRO PASOS HISTÓRICOS:

Para más de cuatro observaciones históricas no se utilizará ARIMA por su elevado costo computacional y elevado tiempo de entrenamiento, en lugar se utilizarán los algoritmos de RF, SVR y LSTM porque estos tienen un tiempo de entrenamiento mucho menor. Los experimentos se ejecutan con la siguiente cantidad de pasos históricos.

- Diez – once pasos históricos.
- Veinte - veintiún pasos históricos.
- Cincuenta – cincuentaiún pasos históricos.

Cabe resaltar que se aplican los mismos tres ítems anteriores en el pretratamiento de la serie temporal univariante.

BIBLIOGRAFÍA

[1] L. Breiman, “Random Forests - Random Features, Tecnical Report 567, Statistic Department, University of California, Berkeley, (https://www.stat.berkeley.edu/~breiman/random-forests.pdf, 08.10.2018’de erişildi),” pp. 1–29, 1999.

[2] C. C. Chang and C. J. Lin, “LIBSVM: A Library for support vector machines,” *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 1–39, 2011.

[3] Z. Zhang, *Time series: a data analysis approach using R*. 2020.

[4] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.