Predicción COVID-19

Omar Emmanuel Villaseñor Murillo

Universidad de Guadalajara

Centro Universitario de Ciencias Exactas e Ingenierías

Seminario de Solución de Problemas de Inteligencia Artificial II

COVID-19: ¿Nuevo problema?

El nuevo tema del que es imposible no hablar es el *Coronavirus,* que nos tiene a todos en cuarentena con tal de evitar una catástrofe aún mayor que la que actualmente se vive a nivel global. Como paso inicial, en Jalisco se tomaron las medidas de suspender clases y eventos grandes.

Para determinar los días más peligrosos, la UDG creó un algoritmo con el cuál determinaron la fórmula para predecir el aumento de contagios con un flujo normal de personas, este arrojó que el periodo comprendido del día 21 al 25 de marzo serían los días más decisivos, por ende, había que resguardarse.

Como actividad, utilizando un dataset de contagios por país (y utilizando sólo la provincia de Hubei en China) se creó un modelo autoregresivo que ayuda con la predicción de datos para el día siguiente.

Código:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler

data = pd.read_csv("time_series_covid_19_confirmed.csv")

days = data[data["Province/State"] == "Hubei"]

days = days.drop(columns=[
   "Province/State",
   "Country/Region",
   "Lat",
   "Long"
])
x = np.asanyarray(days)
x = np.squeeze(x)
```

Inicialmente, importamos nuestras librerías y cargamos nuestros datos, convirtiéndolos a un arreglo de una dimensión.

```
plt.plot(x)
plt.xlabel('Tiempo')
plt.ylabel('Contagios')
plt.title('Razón de contagios')
plt.show()
p = 3
plt.scatter(x[p:],x[:-p])
plt.title('autocorrelacion')
plt.show()
```

Acto seguido, mostramos las gráficas de los valores y después la matriz de correlación.

```
from pandas.plotting import autocorrelation_plot

autocorrelation_plot(x)
plt.show()
```

Después, importamos la función de pandas para la matriz de autocorrelación y la mostramos.

```
dat = pd.DataFrame({"Contagios": x})
for i in range(0, 3):
   dat = pd.concat([dat, dat.Contagios.shift(-i)], axis=1)
dat = dat[:-p]
```

Para dar inicio a nuestros cálculos, tomamos valores de contagios en un nuevo Dataframe y le concatenamos los 7 días anteriores, para usar esa información en el entrenamiento.

```
x = np.asanyarray(dat.iloc[:, 1:])
y = np.asanyarray(dat.iloc[:, 0])
```

Le asignamos a X el valor de todas las columnas del Dataframe menos la original (salidas) la cuál será asignada a Y.

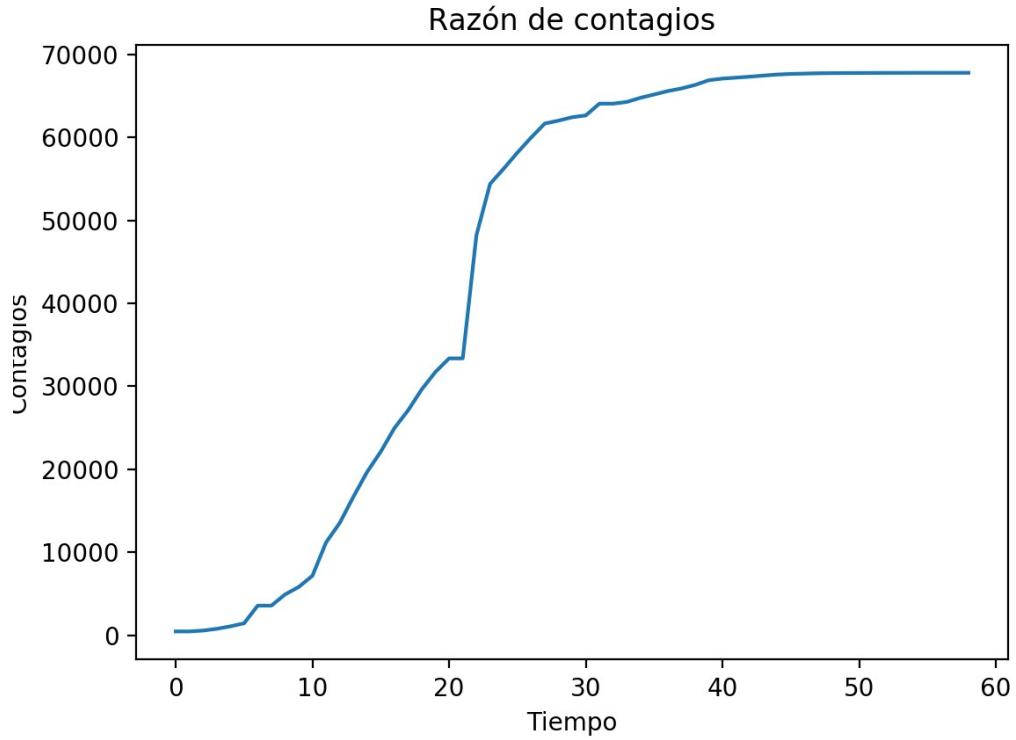
```
from sklearn.model_selection import train_test_split

xtrain, xtest, ytrain, ytest = train_test_split(x,y)

from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
from sklearn.tree import DecisionTreeRegressor

model = Pipeline([
   ('scaler', StandardScaler()),
   ('lin_reg', DecisionTreeRegressor(min_samples_leaf=0.0005))
])

model.fit(xtrain,ytrain)

print('Train: ',model.score(xtrain,ytrain))
print('Test: ',model.score(xtest,ytest))

plt.close('all')
plt.title("Función encontrada")
plt.plot(y, 'bo')
plt.plot(model.predict(x), 'r-')
plt.show()

seven_days_before  = np.linspace(x.min(), x.max(), 7)

print(seven_days_before)

print(model.predict( [seven_days_before] ))
```

En el código anterior, realizamos lo siguiente:

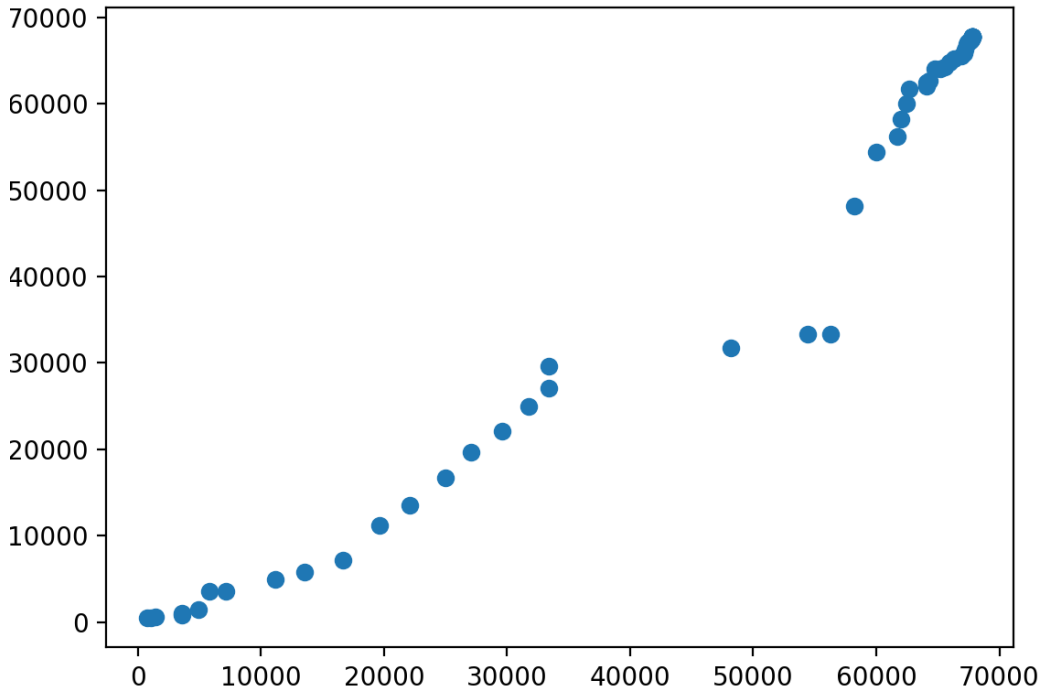
- Separamos los datos para entrenar y probar.
- Importamos nuestras librerías.
- Creamos el modelo con un Pipeline, con el escalador estándar y el modelo de árbol de decisión regresivo.
- Entrenamos.
- Mostramos los scores.
- Cerramos las gráficas anteriores para evitar que se encimen.
- Mostramos la función encontrada según el modelo para los datos que ya teníamos.
- Creamos un nuevo conjunto de contagios que esté acotado entre el máximo y el mínimo de X y tenga 7 valores (para los 7 días).
- Lo mostramos y después mostramos su valor de predicción.

Al ejecutar el código, obtenemos:

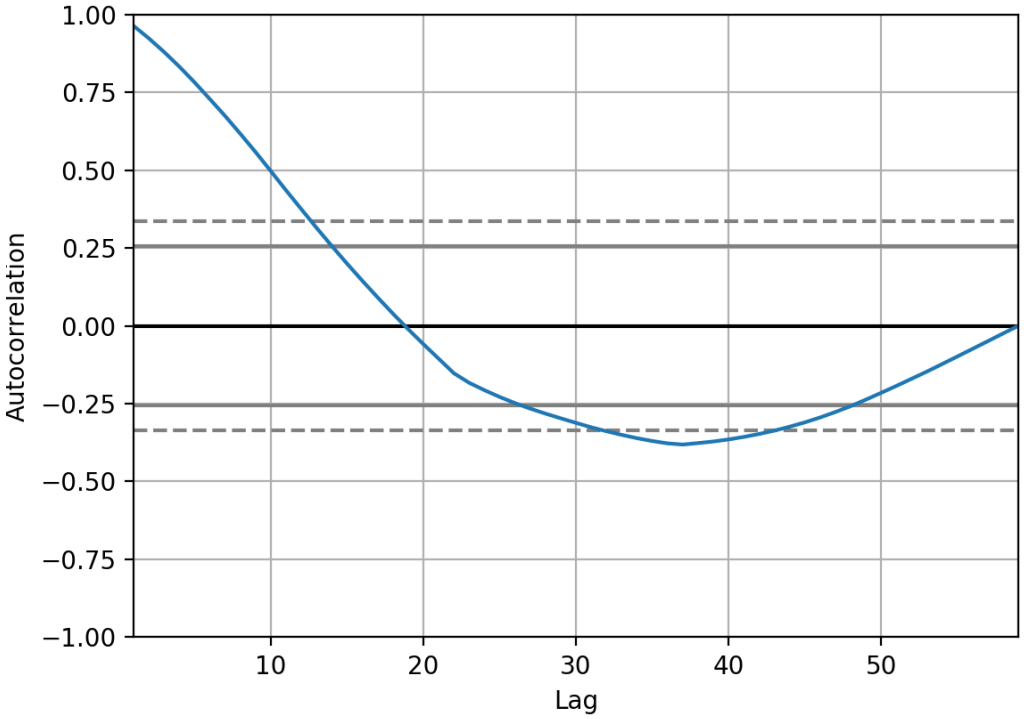
Valores originales:



Correlación:



Autocorrelación:

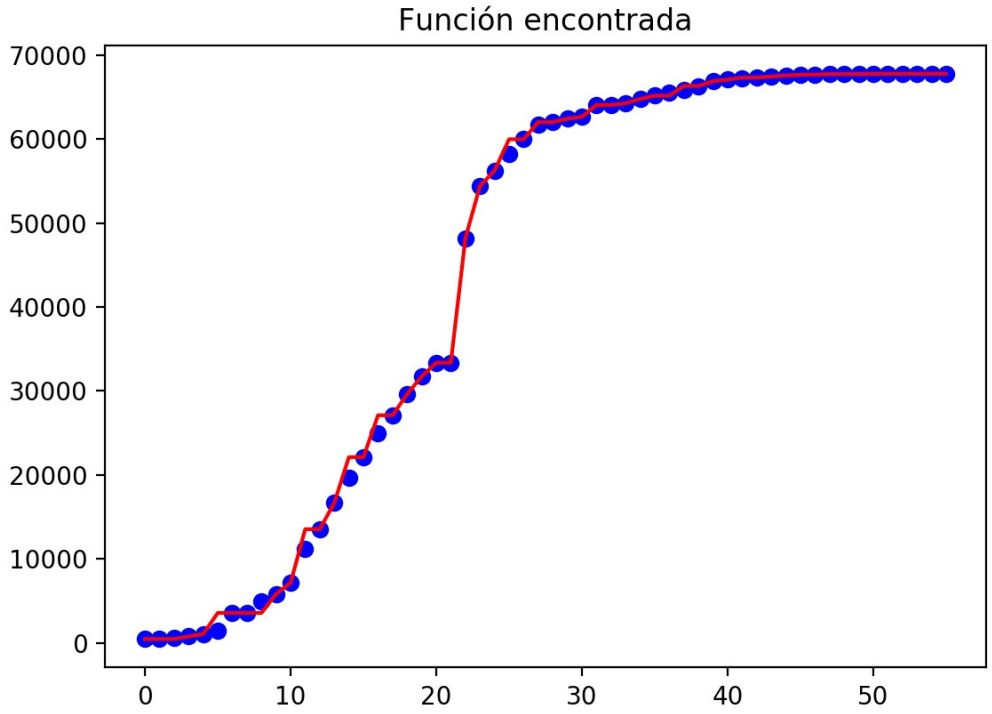


Scores:

Train: 1.0

Test: 0.9975513198554598

Función encontrada:



Valores encontrados y predicción:

[444. 11670. 22896. 34122. 45348. 56574. 67800.]

[22112.]