Computación Blanda

# Lesson 6 Covid19 - Pereira

Autor: Maria Camila Muñoz Mejía

Genny Paola Rivera Becerra

*IS&C, Universidad Tecnológica de Pereira, Pereira, Colombia*

Correo-e: [m.munoz1@utp.edu.co](mailto:m.munoz1@utp.edu.co)

Genny.rivera@utp.edu.co

***Resumen*—** En este documento se realiza una breve explicación del código, el cual muestra de manera gráfica cómo aumentan los casos de covid-19 en la ciudad de Pereira, desde el día que se inicia con el primer caso que es el 9 de marzo de 2020, hasta el 15 de septiembre 2020.

***Palabras clave—*** casos, aumento, contagio, gráficos

Compilar, ejecutar, estructura, datos

***Abstract*—** In this document a brief explanation of the code is made, which graphically shows how the cases of covid-19 increase in the city of Pereira, from the day that the first case begins, which is March 9, 2020, until September 15, 2020.

***KeyWord*—** cases, increase, contagion, graphics, compile, run

Structure, data

**INTRODUCCIÓN**

El covid-19 es un virus el cual nos ha afectado de distintas maneras a toda la humanidad, este se transmite principalmente a partir de otra persona infectada. El aumento de estos casos se ha producido por la falta de cuidado y por no acatar las recomendaciones dadas. Esto ha generado un gran aumento de casos de contagio en el mundo y muertes por este mismo.

I. COVID-19 - PEREIRA

**1.** Se importa la librería del sistema operativo y la librería de Numpy, Scipy y Matplotlib.

import os

Import Numpy as np

import scipy as sp

import matplotlib.pyplot as plt

Se importan directorios chart y data en el directorio de trabajo.

* DATA\_DIR: Directorio de los datos
* CHART\_DIR: Directorio gráficos generados.

From utils import DATA\_DIR, CHART\_DIR

Como función principal se eliminan las advertencias por el uso de funciones que pueden cambiar en el futuro.

np.seterr(all='ignore')

Se establecen los datos de trabajo de la siguiente manera

data = np.genfromtxt

(os.path.join(DATA\_DIR, "tgg.tsv"),

delimiter="\t")

Se establece el tipo de dato

data = np.array(data, dtype=np.float64)

print(data[:10])

print(data.shape)

* Se crea el vector x, correspondiente a la primera columna de data
* Se crea el vector y, correspondiente a la segunda columna de data

x = data[:, 0]

y = data[:, 1]

La función isnan (vector) devuelve un vector en el cual los TRUE son valores de tipo nan, y los valores FALSE son valores diferentes a nan.

print("Número de entradas incorrectas:", np.sum(np.isnan(y)))

Se aplica la eliminación de datos incorrectos sobre el vector y, lo cual hace que tanto x como y queden completamente sincronizados: Sin valores nan.

x = x[~np.isnan(y)]

y = y[~np.isnan(y)]

2. Definimos los modelos con la siguiente función, la cual contiene el comportamiento de un ajuste con base a un grado polinomial

def **plot\_models**(x, y, models, fname, mx=None, ymax=None, xmin=None):

''' dibujar datos de entrada '''

* Crea una nueva figura
* Num = identificador
* Fingsize = Anchura y altura.

plt.figure(num=None, figsize=(25, 8))

Los primeros corchetes ([]) se refieren a las marcas en x

Los siguientes corchetes ([]) se refieren a las etiquetas

En el primer corchete se tiene: 1\*7+ 2\*7+..., hasta completar el total de puntos en el eje horizontal, según el tamaño del vector x.

Además, se aprovecha para calcular los valores de w, los cuales se agrupan en paquetes de w\*7.

Estos valores se utilizan en el segundo corchete para escribir las etiquetas basadas en estos valores de w.

Por tanto, se escriben etiquetas para w desde 1 hasta 45, lo cual constituye las semanas analizadas

S = Semana, la cual se muestra en la parte inferior de la gráfica.

plt.xticks(

[w \* 7 for w in range(45)],

['S %i' % w for w in range(45)])

3. linspace devuelve números espaciados uniformemente durante un intervalo especificado. En este caso, sobre el conjunto de valores x establecido.

Si no se define ningún valor para mx (revisar el código más adelante), el valor de mx será calculado con la función linspace

if mx is None:

mx = np.linspace(0, x[-1], 1000)

* La función Zip () toma elementos iterables (puede ser cero o más), los agrega en una tupla y los devuelve,
* Se realiza un ciclo, la función Zip nos permite unir diferentes listas en una sola lista así Zip (models, linestyles, color).

for model, style, color in zip(models, linestyles, colors):

Se utiliza plt.plot para la graficación de cada modelo que haya dentro de la lista según su color y estilo.

plt.plot(mx, model(mx), linestyle=style, linewidth=2, c=color)

Crea una leyenda en la parte más superior izquierda mostrando de forma visual la dimensión de cada modelo según su estilo de línea.

plt.legend(["d=%i" % m.order for m in models], loc="upper left")

Se definen los límites que tendrán las funciones en la gráfica, esto con el fin de poder extrapolar los resultados o el comportamiento de las funciones a futuro por defecto en caso de no ser definidas, se tomara como limites nulos

plt.ylim(ymin=0)

if ymax:

plt.ylim(ymax=ymax)

if xmin:

plt.xlim(xmin=xmin)

plt.grid(True,linestyle='-',color='0.75')

4. En la siguiente función grafica únicamente los datos dados sin modelo, sin límites y asigna la ubicación y el nombre para el archivo de la gráfica.

plot\_models(x,y,None,os.path.join(CHART\_DIR,"1400\_01\_01.png"))

Creación y dibujo del modelo con las siguientes funciones:

Todos estos valores son devueltos por la función polyfit quien hace dichos calculos a partir de los datos y la dimensión del modelo

fp1, res1, rank1, sv1, rcond1

= np.polyfit(x, y, 1, full=True)

print("Parámetros del modelo fp1:%s"% fp1)

print("Error del modelo fp1:", res1)

f1 = sp.poly1d(fp1)

**fp1** = Coeficientes polinomiales

**res1**= La suma de los residuos al cuadrado del ajuste de mínimos cuadrados

**rank1**= Rango de la matriz

**sv1**= Valores singulares

fp2, res2, rank2, sv2, rcond2

= np.polyfit(x, y, 2, full=True)

print("Parámetros del modelo fp2:%s"% fp2)

print("Error del modelo fp2:", res2)

f2 = sp.poly1d(fp2)

**5.** Se gráfica el modelo de grado 1 sin límites (proyección) en la ruta y con el nombre indicado

plot\_models(x, y, [f1], os.path.join

(CHART\_DIR,"1400\_01\_02.png"))

Se grafican simultáneamente los modelos de grado 1 y 2 sin límites (proyección) en la ruta y con el nombre indicado

plot\_models(x, y, [f1, f2], os.path.join

(CHART\_DIR, “1400\_01\_03.png"))

Se grafican simultáneamente los modelos de grado 1, 2, 3, 10, 53 sin límites (proyección), en la ruta y con el nombre indicado

plot\_models(x,y,[f1,f2,f3,f10,f100],os.path.join (CHART\_DIR, "1400\_01\_04.png"))

Calcula el punto de inflexión en la semana 15, ya que es donde se ve el incremento de casos contagiados

Inflexión = 15 \* 7

Toma solo los datos que hay hasta el punto de inflexión

xa = x[:int(inflexión)]

ya = y[:int(inflexión)]

Se grafican dos líneas rectas y se construye el modelo de una dimensión para los datos que hay hasta el punto de inflexión

fa = sp.poly1d(np.polyfit(xa, ya, 1))

fb = sp.poly1d(np.polyfit(xb, ybx, 1))

Se grafican los modelos de inflexión construidos anteriormente sin límites (sin proyección), en la ruta y con el nombre indicado

plot\_models(x,y,[fa,fb],os.path.join

(CHART\_DIR, "1400\_01\_05.png"))

**6.** Se extrapola de modo que se proyecten respuestas en el futuro y se grafican todos los modelos simultáneamente con limites (con proyección a futuro) específicamente hasta la semana 40 en la ruta y con el nombre indicado para su almacenamiento.

plot\_models(

x, y, [f1,f2,f3,f10,f100],

os.path.join(CHART\_DIR,"1400\_01\_06.png"),

mx=np.linspace(0 \* 7, 40 \* 7, 100),

ymax= 250, xmin= 0 \* 7)

Se construyen nuevamente los modelos de 1,2,3,10,100 dimensiones pero a partir de únicamente los datos que hay desde el punto de inflexión en adelante.

fb1 = fb

fb2 = sp.poly1d(np.polyfit(xb, yb, 2))

fb3 = sp.poly1d(np.polyfit(xb, yb, 3))

fb10 = sp.poly1d(np.polyfit(xb, yb, 10))

fb100 = sp.poly1d(np.polyfit(xb, yb, 100))

Se grafican los modelos construidos solo con los datos que se encuentran desde el punto de inflexión en adelante, con limites (proyección a futuro) específicamente hasta la semana 40, en la ruta y con el nombre de archivo indicado.

plot\_models

(x, y, [fb1, fb2, fb3, fb10, fb100],

os.path.join(CHART\_DIR,"1400\_01\_07.png"),

mx = np.linspace (0 \* 7, 40 \* 7, 100),

ymax = 250, xmin = 0 \* 7)

Se crea un rango de posiciones desde 0 hasta la longitud de xb (datos después del punto de inflexión) random.permutation permite mezclar todos los elementos de una lista al azar.

shuffled= sp.random.permutation

(list(range(len(xb))))

Se toma el primer 30% de las posiciones y se ordenan

test = sorted(shuffled[:split\_idx])

Se toma el % restante y de igual forma se ordenan dichas posiciones

train = sorted(shuffled[split\_idx:])

Se crean los nuevos modelos a partir de los datos que hay después del punto de inflexión solo tomando las posiciones calculadas en la lista train.

fbt1 =

sp.poly1d (np.polyfit(xb[train],yb[train], 1))

fbt2 =

sp.poly1d(np.polyfit(xb[train],b[train], 2))

En esta función se muestra el polinomio generado a través de poly1d

print("fbt2(x)= \n%s" % fbt2)

print("fbt2(x)-100,000= \n%s" % (fbt2-100000))

fbt3=sp.poly1d

(np.polyfit(xb[train], yb[train], 3))

fbt10=sp.poly1d

(np.polyfit(xb[train],yb[train],10)

fbt100=sp.poly1d

(np.polyfit(xb[train],yb[train],100))

Se grafican los nuevos modelos a partir del punto de inflexión y las posiciones de entrenamiento, con limites (proyección a futuro), precisamente hasta la semana 40, con la ruta y el nombre de archivo indicado

plot\_models(

x, y, [fbt1, fbt2, fbt3, fbt10, fbt100],

os.path.join(CHART\_DIR, "1400\_01\_08.png"),

mx=np.linspace (0 \* 7, 40 \* 7, 100),

ymax=250, xmin=0 \* 7)

Se hace uso de la función fsolve la cual a partir del modelo dado(al cual se le resta el número del cual deseamos pronosticar su tiempo de llegada) nota=x0 estimación inicial, o punto de partida de la posible respuesta nota2= se divide el resultado obtenido en los días de la semana (7) para obtener el resultado dado en días

alcanzado\_max = fsolve(fbt2 - 0, x0=240) / (7)

print("\n0 casos por día de COVID-19 esperados en la semana %f" %

alcanzado\_max[0])

**7.** Luego de terminar el código, lo ejecutamos y se muestra lo siguiente:

* Nos muestra el archivo tgg, el cual es donde nos quedó guardado los 183 días de pandemia con sus respectivos contagios diarios. (Los del lado izquierdo es el numero consecutivo hasta 183 y los del lado derecho son los contagios diarios)

1 1

2 2

3 0

4 1

5 1

6 2

7 0

8 3

9 2

10 0

11 0

12 0

13 8

14 0

15 5

16 0

17 0

18 2

19 0

20 0

21 0

22 0

23 5

24 1

25 2

26 3

27 2

28 1

29 3

30 2

31 0

32 29

33 8

34 4

35 0

36 2

37 5

38 3

39 1

40 25

41 3

42 0

43 6

44 0

45 6

46 1

47 4

48 0

49 0

50 0

51 0

52 0

53 1

54 2

55 0

56 0

57 0

58 18

59 2

60 0

61 0

62 1

63 0

64 0

65 0

66 1

67 0

68 1

69 4

70 1

71 0

72 2

73 0

74 0

75 2

76 0

77 0

78 0

79 1

80 1

81 2

82 3

83 1

84 0

85 2

86 4

87 3

88 2

89 4

90 1

91 0

92 1

93 3

94 23

95 3

96 6

97 3

98 2

99 5

100 11

101 4

102 8

103 5

104 2

105 3

106 0

107 11

108 9

109 10

110 16

111 2

112 4

113 18

114 10

115 40

116 8

117 14

118 6

119 18

120 49

121 23

122 26

123 27

124 16

125 0

126 3

127 47

128 46

129 50

130 44

131 29

132 2

133 47

134 71

135 65

136 57

137 112

138 124

139 8

140 90

141 49

142 89

143 98

144 68

145 23

146 80

147 123

148 119

149 87

150 104

151 134

152 150

153 15

154 85

155 98

156 42

157 129

158 172

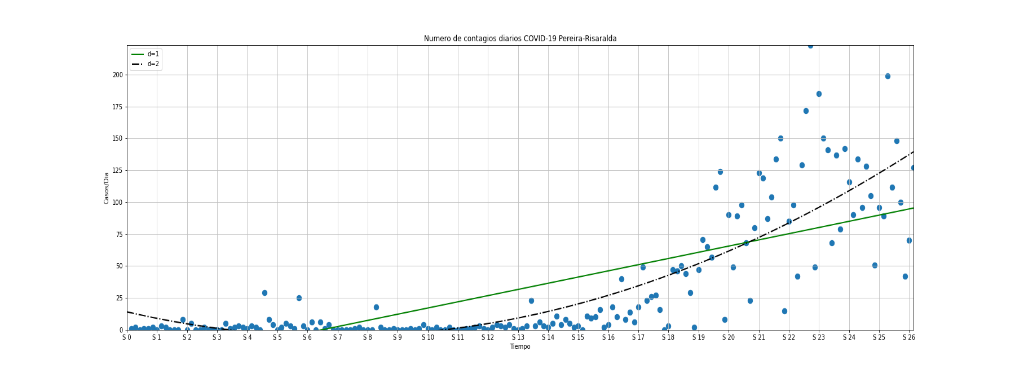
159 223

160 49

161 185

162 150

163 141

****164 68

165 137

166 79

167 142

168 116

169 90

170 134

171 96

172 128

173 105

174 51

175 96

176 89

177 199

178 112

179 148

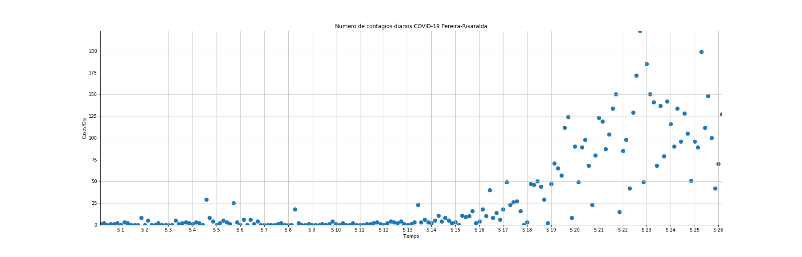
180 100

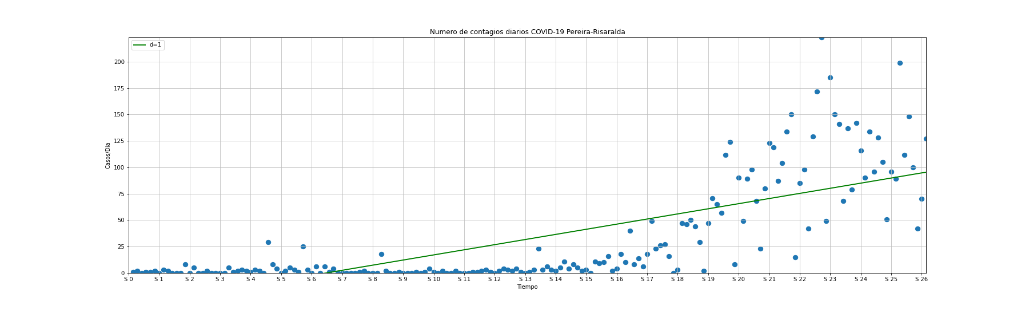
181 42

182 70

183 127

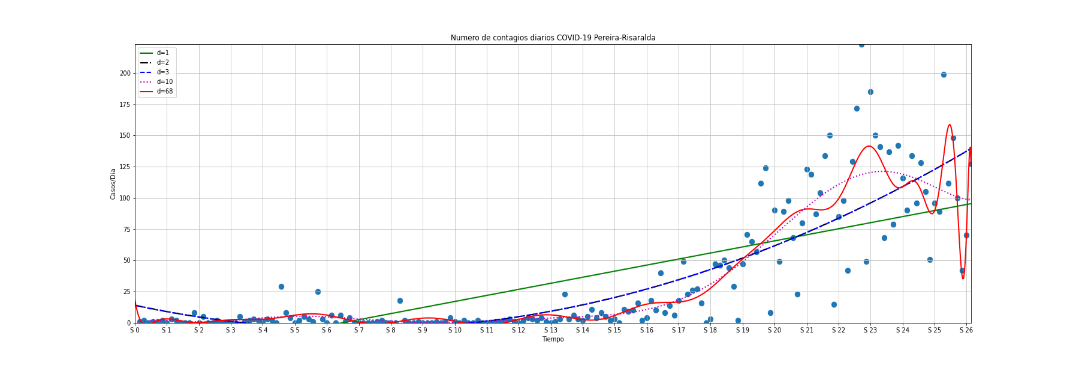
**Primera gráfica:** La cual nos muestra el aumento de contagios diarios, se puede observar que en las primeras semanas se lleva un límite de contagios sin aumento pero en la semana 15 es donde se incrementan.

****

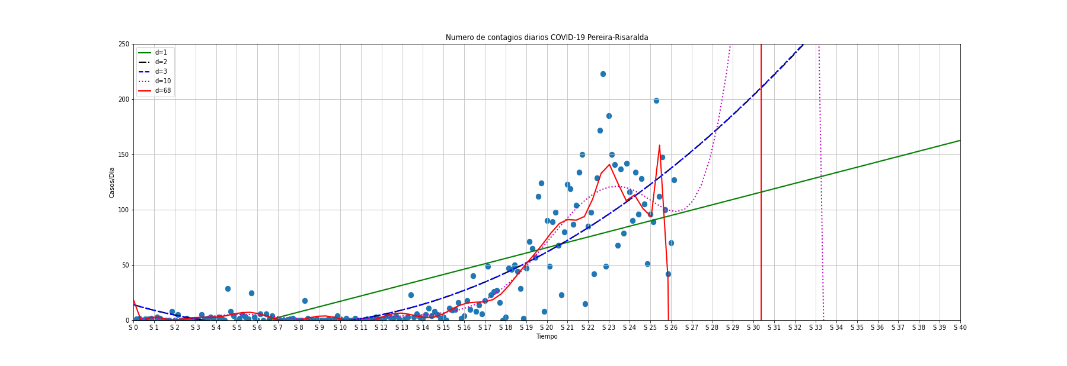
**Segunda gráfica:** Nos muestra el primer modelo del polinomio de una dimensión, el cual agrega una recta para poder obtener el punto de inflexión que es la semana 15.

**Tercera gráfica:** Se muestra el modelo de una dimensión que es la recta y el modelo de segunda dimensión que es una parábola, los cuales nos dan el punto de inflexión y se muestra que en la semana 15 es donde se disparan los contagios.

**Cuarta gráfica:** Es el polinomio de tercera dimensión, el cual muestra de manera más clara el incremento de los contagiosdesde la semana 15 desde el punto de inflexión.

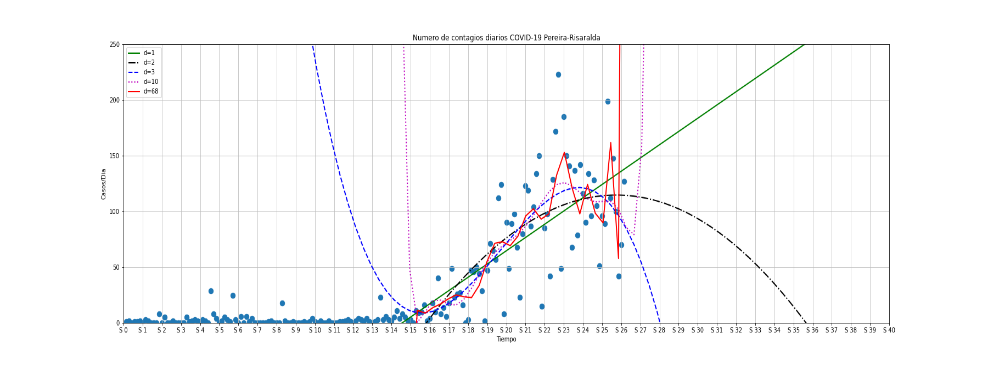
****

**Quinta gráfica:** Se grafican otras dimensiones para ver si se obtienen otros cambios diferentes.



**Sexta gráfica**: Como en las gráficas anteriores habíamos tomado los datos desde la semana 1 hasta la 26 que fueron la toma de los datos, aquí en esta grafica se puede visualizar que ya sol se trabara con los datos a partir del punto de inflexión que sería desde la semana 15 en adelante. Esta nos muestra de cierta manera que llega hasta un punto de inflexión alto y luego empieza a decaer en la semana 26 entonces esta se extrapola hasta la semana 40.

Esto nos quiere decir que los cero contagiados que habrá serian en la semana 36, esto quiere decir esta base de datos que esa sería la semana en donde no abrían contagiados.



II. CONCLUSIONES COVID-19 - PEREIRA

* A partir de la predicción realizada con el modelo de grado 2, se llegó a la conclusión que los 0 contagios llegaran en la semana 36.
* Gracias al punto de inflexión que se tomó en la semana 15, con el criterio que es el punto en la gráfica en el cual se aumentan los contagios, los datos pueden ser obtenidos con mayor precisión.
* Con las gráficas se puede obtener con más claridad los aumentos de contagios en cada semana.

III. BIBLIOGRAFIA

* <https://www.datos.gov.co/Salud-y-Protecci-n-Social/Vista-casos-positivos-COVID-19-Pereira/mehb-4wt8/data#Filter>