
***INSTITUTO POLITÉCNICO NACIONAL***

***ESCUELA SUPERIOR DE CÓMPUTO***

# *Data Mining*

***Grupo: 3CV14***

**Guzmán Hernández Luis Daniel**
**López Sánchez Kevin Ian**

## Práctica 8. Aplicación de tareas de aprendizaje supervisado

10 de junio de 2022

# INTRODUCCIÓN

En la presente práctica, se aplicarán conocimientos previos vistos en el curso, temas referentes con la introducción con el aprendizaje de máquina, por medio de técnicas de clasificación y predicción referentes al análisis de datos. Para esto se probará un modelo de predicción de datos obtenido del dataset de denuncias ante la PAOT que utilizamos en el proyecto semestral, ligeramente seccionado y modificado, para poder entrenar con el 80% de los datos el modelo SVR (Support Vector Regression) y posteriormente, usar el 20% para realizar las pruebas.

Todo esto se podrá realizar gracias al software libre de Anaconda, que nos permite implementar estos modelos sin un análisis tan profundo mediante librerías de Python.

Finalmente, realizaremos una breve exploración de los datos de entrenamiento y los de prueba para poder obtener conclusiones más precisas relevantes al desempeño del modelo implementado.

# DESARROLLO

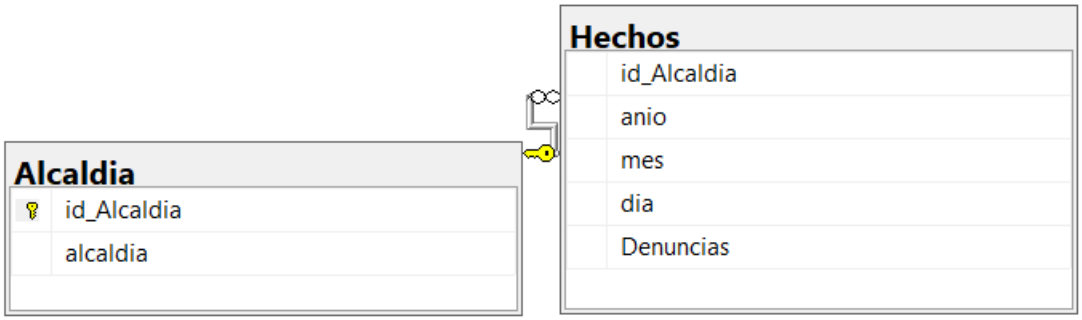
1. **Elija un problema de clasificación o predicción aplicado a alguna situación útil de su proyecto, indique en un párrafo de texto el problema elegido y que desea obtener.**

**Problema de Predicción:**

Se intenta pronosticar el número de denuncias realizadas en cada alcaldía por la dimensión del tiempo, desde la granularidad de año, mes y día, considerando 16 años de datos, desde 2002 hasta el 2018.

2. **Diseñe dos tablas de hechos, o datasets, para la fase de entrenamiento y de pruebas. Documente y explique porque eligió las dimensiones o columnas y el nivel de granularidad de datos.**

   Para el dataset de entrenamiento y pruebas usamos la misma estructura mostrada a continuación.




   Utilizamos la herramienta de Pentaho para poder seleccionar y acomodar los datos en dos archivos diferentes, uno que utilizaremos para el entrenamiento del SVR con el 80% de los datos, y otro para el testeo con el 20% de los datos. Estos datasets los generamos a partir del dataset de “denuncias_PAOT” el cual previamente limpiamos y utilizamos en el proyecto.

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | id_Alcaldia | anio | mes | dia | Denuncias |
| 2 | 9 | 2010 | 11 | 4 | 10 |
| 3 | 9 | 2018 | 11 | 5 | 1 |
| 4 | 9 | 2016 | 2 | 10 | 1 |
| 5 | 9 | 2014 | 1 | 22 | 2 |
| 6 | 9 | 2016 | 1 | 26 | 1 |
| 7 | 9 | 2018 | 7 | 17 | 2 |
| 8 | 9 | 2013 | 5 | 31 | 1 |
| 9 | 9 | 2015 | 10 | 7 | 3 |
| 10 | 9 | 2015 | 2 | 23 | 1 |
| 11 | 9 | 2011 | 5 | 20 | 1 |
| 12 | 9 | 2010 | 4 | 12 | 2 |
| 13 | 9 | 2009 | 11 | 17 | 1 |
| 14 | 9 | 2003 | 6 | 11 | 1 |
| 15 | 9 | 2018 | 9 | 14 | 1 |
| 16 | 9 | 2013 | 10 | 18 | 1 |
| 17 | 9 | 2010 | 6 | 10 | 1 |
| 18 | 9 | 2014 | 5 | 12 | 2 |
| 19 | 9 | 2006 | 4 | 3 | 2 |
| 20 | 9 | 2004 | 12 | 6 | 1 |
| 21 | 9 | 2015 | 11 | 13 | 1 |
| 22 | 9 | 2018 | 1 | 16 | 3 |
| 23 | 9 | 2015 | 5 | 20 | 1 |
| 24 | 9 | 2010 | 10 | 26 | 1 |
| 25 | 9 | 2003 | 4 | 21 | 1 |
| 26 | 9 | 2016 | 5 | 23 | 1 |
| 27 | 9 | 2006 | 5 | 26 | 1 |
| 28 | 9 | 2016 | 9 | 23 | 3 |
| 29 | 9 | 2018 | 8 | 22 | 4 |
| 30 | 9 | 2014 | 11 | 11 | 1 |

De esta forma, tenemos dimensiones de tiempo, que cambian con mes y día, y una geográfica (alcaldía) pues los valores si cambian dependiendo la alcaldía. A continuación mostramos la tabla con los ID's correspondientes a cada alcaldía.

| | id_Alcaldia | alcaldia |
|---|---|---|
| 1 | 1 | Benito Juárez |
| 2 | 2 | Tlalpan |
| 3 | 3 | Venustiano Carranza |
| 4 | 4 | Cuauhtémoc |
| 5 | 5 | Álvaro Obregón |
| 6 | 6 | Gustavo A. Madero |
| 7 | 7 | Iztapalapa |
| 8 | 8 | Azcapotzalco |
| 9 | 9 | Xochimilco |
| 10 | 10 | Miguel Hidalgo |
| 11 | 11 | Coyoacán |
| 12 | 12 | Tláhuac |
| 13 | 13 | Cuajimalpa de Morelos |
| 14 | 14 | Iztacalco |
| 15 | 15 | La Magdalena Contreras |
| 16 | 16 | Milpa Alta |

3. **Utilice el código adjunto con nombre "svmPredictorBetaParaModificarParaClasificador.zip" el cual consiste en un ejemplo en python para predecir valores usando SVM**.

Entre los principales cambios realizados al archivo desarrollado en python, fueron los nombres de los archivos para testeo y entrenamiento y también el nombre de las columnas trabajadas. Se realizó el proceso de la siguiente manera:

```
dfTraining = pd.read_excel("80.xls")
dfTesting = pd.read_excel("20.xls")


X_train = dfTraining[["anio","mes","dia"]]
y_train=dfTraining.Denuncias

X_testing = dfTesting[["anio","mes","dia"]]
y_testing = dfTesting.Denuncias
```

Al ejecutar el programa, a la salida nos entregó valores variables, teniendo como porcentaje mínimo de error un 0.4%, pero en promedio cerca del 30%-50%

```
Predicted Value:  2.2784892573393654  Real value:  1  % Error:  127.84892573393654
Predicted Value:  1.1344882170346973  Real value:  2  % Error:  43.275589148265134
Predicted Value:  1.1450934520246074  Real value:  3  % Error:  61.83021826584641
Predicted Value:  1.8999745281838107  Real value:  2  % Error:  5.001273590809463
Predicted Value:  1.200057345837173  Real value:  1  % Error:  20.005734583717306
Predicted Value:  1.2001402898882234  Real value:  1  % Error:  20.01402898882234
Predicted Value:  1.2002627988598094  Real value:  1  % Error:  20.026279885980934
Predicted Value:  1.1098216490123374  Real value:  1  % Error:  10.982164901233737
Predicted Value:  1.4923085232648758  Real value:  1  % Error:  49.230852326487586
Predicted Value:  1.2001786919065784  Real value:  2  % Error:  39.991065404671076
Predicted Value:  1.1324147134861329  Real value:  1  % Error:  13.241471348613288
Predicted Value:  1.1996282354584267  Real value:  1  % Error:  19.962823545842667
Predicted Value:  1.3407499959857212  Real value:  3  % Error:  55.308333467142624
Predicted Value:  1.225431825590419  Real value:  5  % Error:  75.49136348819162
Predicted Value:  1.200133959253173  Real value:  4  % Error:  69.99665101867068
Predicted Value:  1.8057611071787893  Real value:  1  % Error:  80.57611071787893
Predicted Value:  1.2000131015302486  Real value:  1  % Error:  20.001310153024864
Predicted Value:  1.200345140824196  Real value:  3  % Error:  59.988495305860134
Predicted Value:  1.199832813725595  Real value:  2  % Error:  40.008359313720256
Predicted Value:  1.5396777470736758  Real value:  1  % Error:  53.967774707367575
Predicted Value:  1.1997902959217812  Real value:  2  % Error:  40.01048520391094
Predicted Value:  1.2209956999878484  Real value:  3  % Error:  59.30014333373839
```

Por lo que procedimos a cambiar los parámetros, ahora tomaremos en cuenta la alcaldía y eliminaremos el año en que se realizaron las denuncias

```
X_train = dfTraining[["id_Alcaldia","mes","dia"]]
y_train=dfTraining.Denuncias

X_testing = dfTesting[["id_Alcaldia","mes","dia"]]
y_testing = dfTesting.Denuncias
```

También modificamos el valor de épsilon, que identifica que tan permisible es el margen de error dentro del vector.

```
clf = SVR(C=1.0, epsilon=0.01)
```

Finalmente ejecutamos de nuevo el programa y obtenemos los siguientes valores:

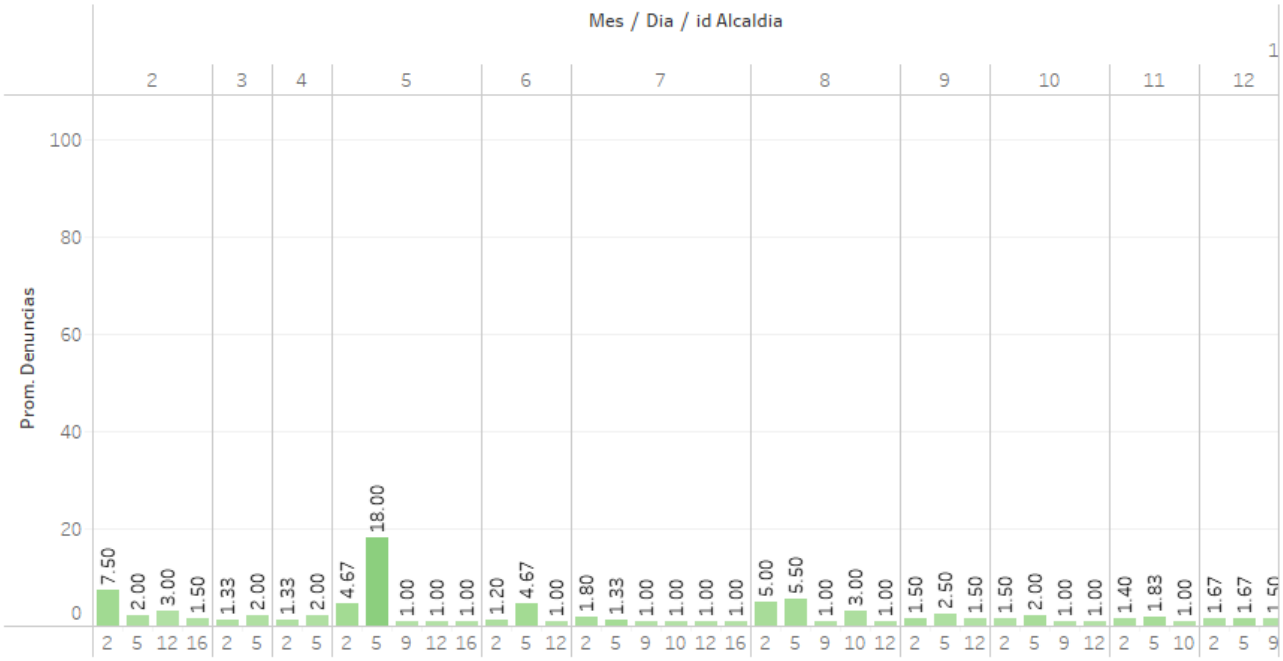
```
Predicted Value:  1.009807061691871  Real value:  1  % Error:  0.9807061691871066
Predicted Value:  1.3169882653915523  Real value:  2  % Error:  34.15058673042238
Predicted Value:  1.001161085059728  Real value:  2  % Error:  49.941945747013605
Predicted Value:  1.0099395101057853  Real value:  4  % Error:  74.75151224735536
Predicted Value:  1.0096846765132652  Real value:  1  % Error:  0.9684676513265211
Predicted Value:  0.9903131965779896  Real value:  1  % Error:  0.9686803422010448
Predicted Value:  0.9980004165139451  Real value:  1  % Error:  0.199958348605489
Predicted Value:  1.0103962883449014  Real value:  1  % Error:  1.039628834490136
Predicted Value:  1.167712928071439  Real value:  1  % Error:  16.77129280714389
Predicted Value:  1.005763665651905  Real value:  1  % Error:  0.5763665651904892
Predicted Value:  1.0472392074405439  Real value:  1  % Error:  4.723920744054388
Predicted Value:  0.860532737700738  Real value:  1  % Error:  13.946726229926199
Predicted Value:  0.9898082807932228  Real value:  1  % Error:  1.0191719206777194
Predicted Value:  1.0754697556883828  Real value:  1  % Error:  7.546975568838277
Predicted Value:  0.8952371177849895  Real value:  1  % Error:  10.47628822150105
Predicted Value:  0.9945142231466297  Real value:  1  % Error:  0.5485776853370261
Predicted Value:  1.3517336443522487  Real value:  3  % Error:  54.94221185492504
Predicted Value:  1.0199013985175993  Real value:  1  % Error:  1.9901398517599311
Predicted Value:  1.0103770680337985  Real value:  1  % Error:  1.0377068033798453
Predicted Value:  0.9897740454915727  Real value:  1  % Error:  1.0225954508427337
Predicted Value:  0.9898316471162556  Real value:  1  % Error:  1.0168352883744403
Predicted Value:  1.1365393663735603  Real value:  1  % Error:  13.653936637356034
Predicted Value:  0.9897350922623103  Real value:  1  % Error:  1.0264907737689732
```

Si bien seguimos obteniendo algunos valores erróneos, la mayoría de estos no pasan del 50% de error.
Ahora, procedemos a realizar 15 pruebas individuales para revisar el algoritmo implementado:

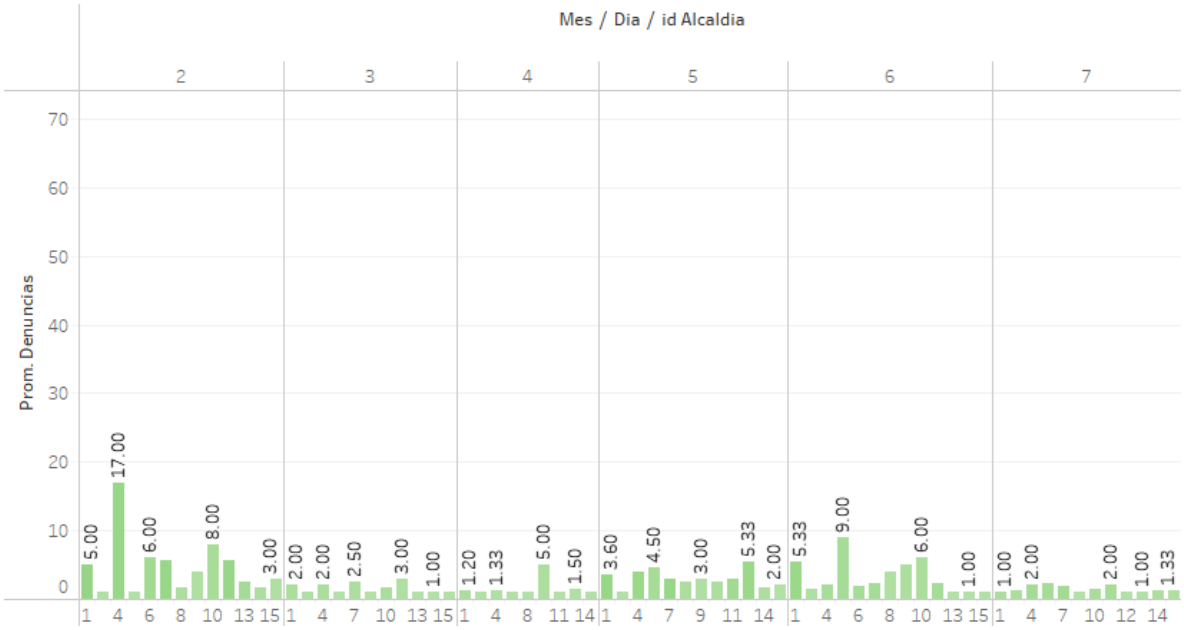
1.
```
Predicted Value:  1.5101058060246393  Real value:  4  % Error:  62.24735484938402
```
2.
```
Predicted Value:  1.331900329331082  Real value:  2  % Error:  33.4049835334459
```
3.
```
Predicted Value:  1.8952254422245336  Real value:  3  % Error:  36.825818592515546
```
4.
```
Predicted Value:  0.9897068774529647  Real value:  1  % Error:  1.0293122547035294
```
5.
```
Predicted Value:  1.0100936502909375  Real value:  1  % Error:  1.009365029093745
```
6.
```
Predicted Value:  1.0097971510092525  Real value:  1  % Error:  0.9797151009252492
```
7.
```
Predicted Value:  1.2552154833257148  Real value:  1  % Error:  25.521548332571477
```
8.
```
Predicted Value:  1.1231584801547094  Real value:  1  % Error:  12.315848015470943
```
9.
```
Predicted Value:  1.0153268298700575  Real value:  2  % Error:  49.23365850649712
```
10.
```
Predicted Value:  1.0927523496305966  Real value:  2  % Error:  45.36238251847017
```

11.
Predicted Value: 0.9902442552362016 Real value: 1 % Error: 0.9755744763798369
12.
Predicted Value: 1.0924324024774157 Real value: 1 % Error: 9.243240247741568
13.
Predicted Value: 0.9977492846640983 Real value: 1 % Error: 0.22507153359017096
14.
Predicted Value: 0.9907098225748087 Real value: 1 % Error: 0.9290177425191337
15.
Predicted Value: 1.3425257138509756 Real value: 3 % Error: 55.24914287163415

**Análisis exploratorio del dataset de prueba:**



**Análisis exploratorio del dataset de prueba:**




Dentro del análisis exploratorio, podemos ver un detalle que nos pareció importante dentro del algoritmo, y es que, no parece haber una relación clara entre los datos, aunque si se puede localizar un cierto patrón en algunas partes, sin embargo, en los valores no son del todo constantes, esto es debido a naturaleza de los fenómenos que se denuncian que en su mayoría no dependen de los demás, tanto es posible que en un día específico existan muchas denuncias a como no exista ninguna.

# CONCLUSIONES

En esta práctica, pudimos poner en práctica algo de lo visto en la unidad 4, que fue lo relacionado con aprendizaje de máquina y breves conceptos para la predicción y categorización de datos. Aplicamos un algoritmo de SVR (Support Vector Regression), que principalmente divide de un lado de un vector, los valores que acepta el algoritmo, y del otro aquellos que no coinciden con el entrenamiento.

Al momento de trabajar con esto, teníamos bastante incertidumbre de los resultados, ya que no sabíamos la forma en que trabajaba el algoritmo, sin embargo, después de hacer varias pruebas logramos identificar los parámetros correctos para que maximizara su funcionamiento. Aunque sobra decir que este tipo de datos no son los adecuados para experimentar con el aprendizaje máquina logramos llevar a la práctica los conceptos vistos en clase, lo que nos ayudó a un mejor entendimiento de estos.