Instituto Politécnico Nacional

Escuela Superior de Cómputo

Práctica 7

Aplicación de tareas de aprendizaje supervisado.

Grupo: 3CV18

Integrantes:

Cazares Martínez Maximiliano

Ramos Nieves Adrián

Profesor:

Roberto Eswart Zagal Flores

Introducción

A lo largo del semestre se ha trabajado un dataset de ayuda acerca de la precipitación pluvial en la CDMX, el cual podremos analizar para tener conocimiento acerca de los lugares en los que existe mayor número de registro de PP, que puede llegar a ser un causante de algunos acontecimientos importantes, desde la falta de agua en algunas colonias hasta lo que podría llegar a ser inundaciones en las diferentes alcaldías de la ciudad de México.

Los datos recabados en la práctica fueron llenados con los informes y base de datos extraída de la página de la CDMX para darle más formalidad a este tipo de análisis. Se obtuvieron datos desde antes del 2000 hasta 2019, pero al realizar nuestras diferentes tablas insertadas con un programa en python solo se tomaron los datos a partir del 2010, ya que fueron años en los que los datos tenían más datos consistentes. Nuestro dataset final incluye todo acerca de los diferentes lugares donde se tienen registros, el número, alcaldía, fecha y Id de cada una de las tablas.

Teniendo así como tabla principal donde insertamos los datos más importantes la Tabla de hechos, que gracias a la ayuda de los conocimientos aprendidos acerca del machine learning vistos en el curso, se desarrolló una propuesta de solución a este tema con la aplicación de tareas de aprendizaje supervisado, y en el siguiente apartado mostraremos los resultado encontrados a lo largo de la realización de esta práctica, además algunas gráficas extras realizadas en tableau para tener un poco más de entendimiento acerca del tema. Al final se aplica un modelo de aprendizaje de máquina, el cual se usa SVM donde se le dan 4 dimensiones de entrenamiento con el 80 % de los datos, para llevar el proceso de aprendizaje con el 19 % restante y 1 % con datos diferentes para el testing, que sería un prueba en la vida real.

Desarrollo

Capturas de Pantalla del código presentado

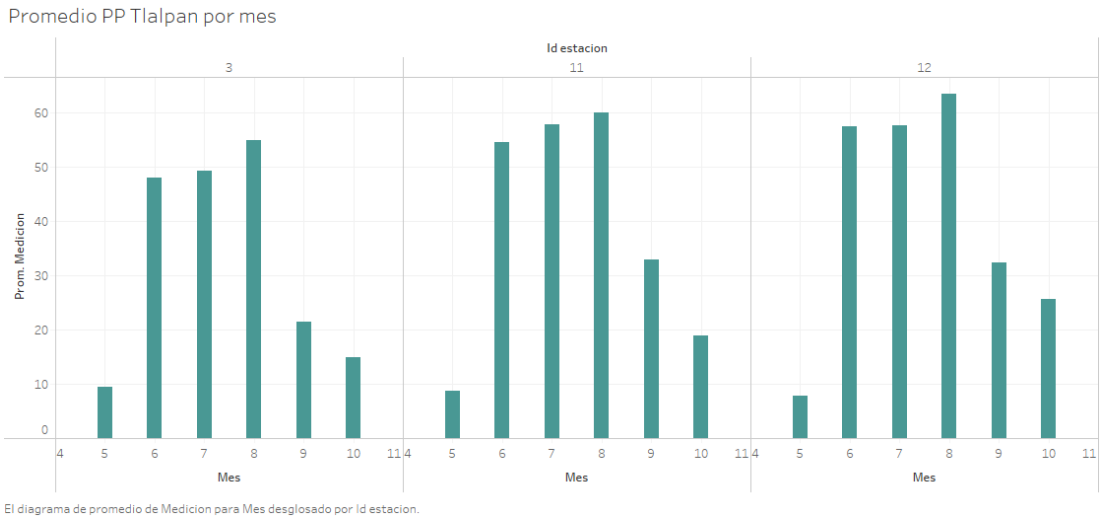
Se presenta el código utilizado en el lenguaje python, se guardaron los datos importantes en el archivo de set Training y el set Testing para mayor facilidad, y se utilizaron las variables de día, mes y año para poder realizar los cálculos precisos , donde se expone cual es el valor predecido, el valor real y el porcentaje de error en la última columna.

```python
import os
import numpy as np
import pandas as pd
from sklearn.svm import SVR
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import cross_val_score
from sklearn.metrics import precision_score,accuracy_score

script_directory = os.path.dirname(__file__)
dataSetTraining = f'{script_directory}/{"SetTraining.csv"}'
dataSetTesting = f'{script_directory}/{"SetTesting.csv"}'

dfTraining = pd.read_csv(dataSetTraining)
dfTesting = pd.read_csv(dataSetTesting)

X_train = dfTraining[["Anio","Mes","Dia"]]
y_train=dfTraining.Medicion

X_testing = dfTesting[["Anio","Mes","Dia"]]
y_testing = dfTesting.Medicion

print("-------------------- Normal SVM -------------------------")

clf = SVR(C=1.0, epsilon=0.2)
clf.fit(X_train, y_train)
SVR(C=1.0, cache_size=200, coef0=0.0,
    degree=3, gamma='auto', kernel='rbf',
    max_iter=-1,  shrinking=True,
    tol=0.001, verbose=False)
scores = cross_val_score(clf, X_train, y_train, cv = 10)
res1=clf.predict(X_testing)

print("\n\n")
index=0
for element in res1:
    error=(abs((element-y_testing[index]))/y_testing[index])*100
    print(f'Valor predecido: {element} -- Valor real: {y_testing[index]} -- % de Error: {error}')
    index=index+1

import csv
pathCsvfile=f'{script_directory}/{"outRealTesting1.csv"}'
with open(pathCsvfile, 'w', encoding='utf-8', newline='') as csvFile:
    fieldnames = ['Valor predecido', 'Valor real','% de Error']
    writer = csv.DictWriter(csvFile, fieldnames=fieldnames)
    writer.writeheader()
    index=0
    for element in res1:
        error=(abs((element-y_testing[index]))/y_testing[index])*100
        print(f'Valor predecido: {element} -- Valor real: {y_testing[index]} -- % de Error: {error}')
        writer.writerow({'Valor predecido': "'"+format(element)+"'",
        'Valor real': "'"+format(y_testing[index])+"'", '% de Error': "'"+format(error)+"'"})
        index=index+1
```

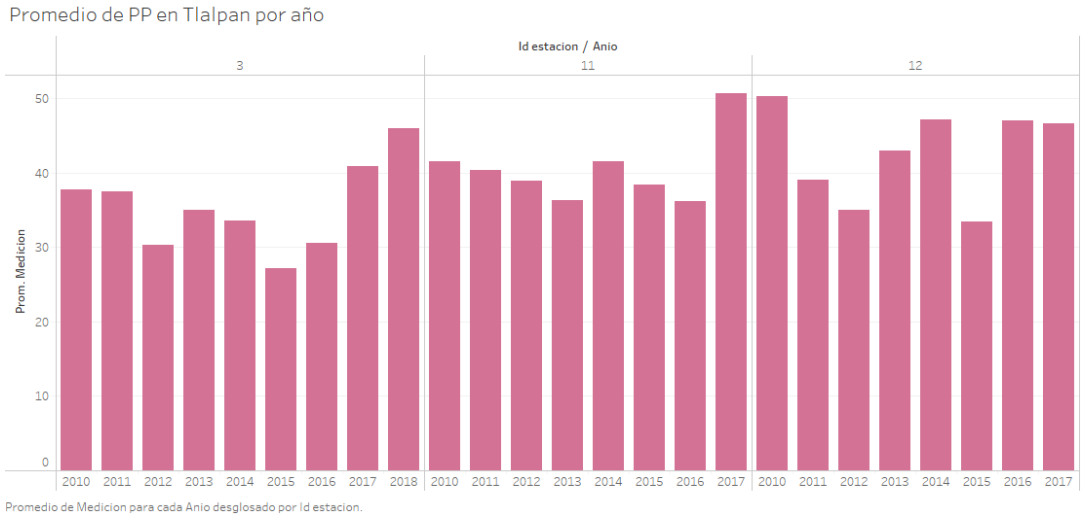
Los datos arrojados al correr el programa son los siguientes, donde marcaremos con un cuadro rojo las partes más importantes, y los porcentajes con menos errores encontrados, que serían la aproximación más real al valor esperado de nuestras variables calculadas.

```
Maxo@Maxo MINGW64 ~/OneDrive/Documentos/ESCOM/Data Mining/Codigos
$ C:/Users/MaxoC/AppData/Local/Programs/Python/Python39/python.exe "c:/Users/MaxoC/OneDrive/D
tor.py"
-------------------- Normal SVM -------------------------



Valor predecido: 33.11037054749498 -- Valor real: 1.75 -- % de Error: 1792.0211741425703
Valor predecido: 33.1112974740511 -- Valor real: 25.24 -- % de Error: 31.18580615709628
Valor predecido: 33.11222413308496 -- Valor real: 3.99 -- % de Error: 729.8803040873422
Valor predecido: 33.10904386107683 -- Valor real: 33.92 -- % de Error: 2.3907905039008677
Valor predecido: 33.10997099382108 -- Valor real: 28.94 -- % de Error: 14.409022093369309
Valor predecido: 33.11089800476896 -- Valor real: 41.91 -- % de Error: 20.99523262999531
Valor predecido: 33.111824791961844 -- Valor real: 79.72 -- % de Error: 58.464845970946
Valor predecido: 33.10877671369613 -- Valor real: 58.37 -- % de Error: 43.27775104729119
Valor predecido: 33.10970386265368 -- Valor real: 46.6 -- % de Error: 28.949133341944894
Valor predecido: 33.110630919033085 -- Valor real: 101.87 -- % de Error: 67.49717196521735
Valor predecido: 33.11155778086736 -- Valor real: 36.12 -- % de Error: 8.329020540234328
Valor predecido: 33.10837711681449 -- Valor real: 53.68 -- % de Error: 38.32269538596407
Valor predecido: 33.109304273645236 -- Valor real: 50.89 -- % de Error: 34.939468906179535
Valor predecido: 33.110231381686404 -- Valor real: 37.92 -- % de Error: 12.683988972345986
Valor predecido: 33.111158338961744 -- Valor real: 44.1 -- % de Error: 24.918008301674053
Valor predecido: 33.112085043522235 -- Valor real: 40.11 -- % de Error: 17.446808667359175
Valor predecido: 33.1089046757365 -- Valor real: 7.98 -- % de Error: 314.8985548337907
Valor predecido: 33.109831815944595 -- Valor real: 46.5 -- % de Error: 28.796060610871837
Valor predecido: 33.110758849171155 -- Valor real: 48.89 -- % de Error: 32.27498701335415
Valor predecido: 33.10863752831294 -- Valor real: 3.59 -- % de Error: 822.2461707050959
Valor predecido: 33.11141864334111 -- Valor real: 0.31 -- % de Error: 10581.102788174552
Valor predecido: 33.10851629961975 -- Valor real: 9.98 -- % de Error: 231.74866031683115
Valor predecido: 33.109443455360044 -- Valor real: 17.46 -- % de Error: 89.63026033997733
Valor predecido: 33.11037054749498 -- Valor real: 2.24 -- % de Error: 1378.141542298883
Valor predecido: 33.1112974740511 -- Valor real: 25.17 -- % de Error: 31.55064550675843
Valor predecido: 33.11222413308496 -- Valor real: 2.69 -- % de Error: 1130.9377001146825
Valor predecido: 33.10904386107683 -- Valor real: 45.1 -- % de Error: 26.58748589561679
Valor predecido: 33.10997099382108 -- Valor real: 44.9 -- % de Error: 26.258416494830556
Valor predecido: 33.11089800476896 -- Valor real: 19.86 -- % de Error: 66.7215408095114
Valor predecido: 33.111824791961844 -- Valor real: 109.56 -- % de Error: 69.77745090182378
Valor predecido: 33.10877671369613 -- Valor real: 69.85 -- % de Error: 52.600176501508756
Valor predecido: 33.10970386265368 -- Valor real: 48.09 -- % de Error: 31.15054301797946
Valor predecido: 33.110630919033085 -- Valor real: 84.61 -- % de Error: 60.86676407158363
Valor predecido: 33.11155778086736 -- Valor real: 55.68 -- % de Error: 40.532403410798565
Valor predecido: 33.10837711681449 -- Valor real: 23.55 -- % de Error: 40.58758860643094
```

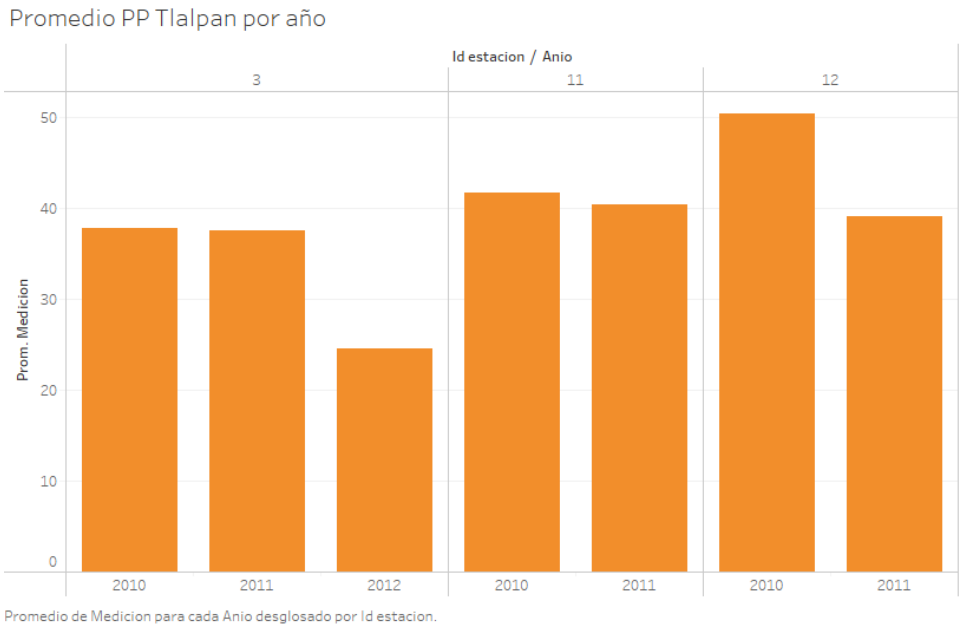
Se anexan algunas tablas realizadas en tableau de los resultados obtenidos de la tabla creada de los data set.

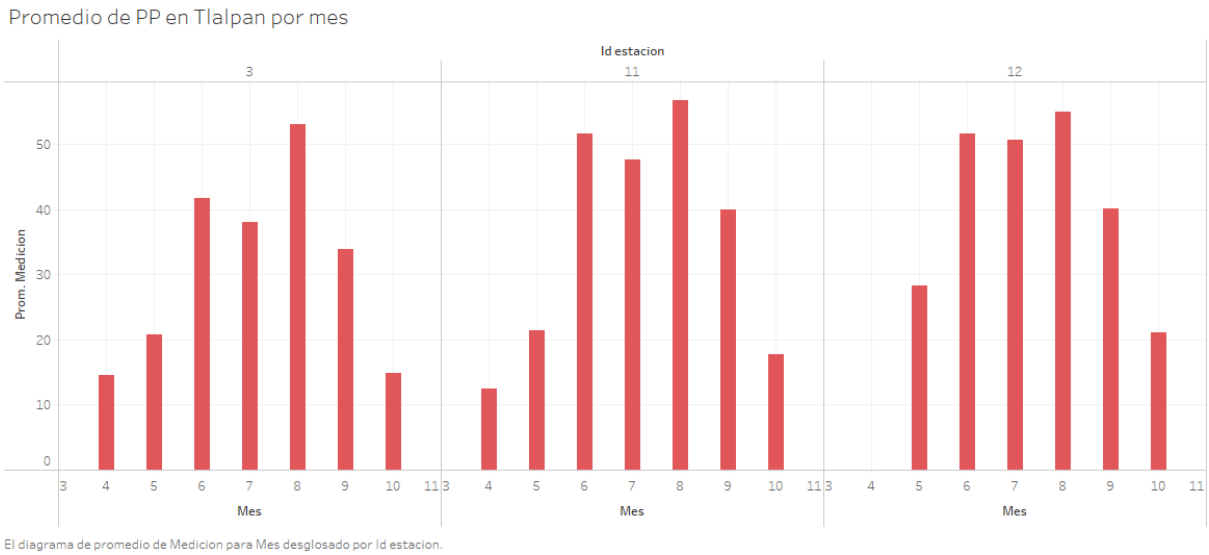


En la primer tabla obtenemos la medición de la PP por mes en la delegación de Tlalpan, donde en las tres estaciones que se tienen registradas de este lugar, encontramos como Agosto el mes con mayores registros obtenidos en el transcurso del año, justamente en las fechas donde las lluvias empiezan a tener un mayor impacto gracias al cambio de estación.



En la siguiente tabla, se hizo uso del tiempo como en la anterior gráfica, para obtener los datos por estaciones de monitoreo, mostrando el año con más registros dentro de nuestro dataset, es impresionante ver como los años 2017 y 2018 tenemos más registros presentes, algo que podemos destacar tal vez es el crecimiento de la contaminación , que es un factor clave para que las lluvias sean más concurrentes, ya que el mismo aire trata de limpiarse con las lluvias, pero esto no siempre tiene que ser bueno porque llegan a haber inundaciones.





En la presente últimas dos gráficas, se efectuó el mismo procedimiento que en la primer parte que vimos, pero ahora hicimos uso solo el 20 % de nuestro dataset, ayudando así a que los datos se analizen de manera más precisa, arrojando diferentes datos en comparación con la primer columna pero obteniendo que las conclusiones sean las mismas, pueden llegar a influenciar las inundaciones en esto.

Conclusiones

Finalmente creo que en esta práctica aunque fue un poco más cortas que las demás, fue muy interesante ver cómo se realiza un análisis más crudo de los datos, un poco más precisos y el funcionamiento al codificar el aprendizaje o Machine Learning, que aunque fue algo simple de realizar a diferencia de los costos computacionales que tiene un programa más avanzado de machine learning, creo que el poder adentrarnos a estos temas fue muy interesante de analizar, así como los procesos para las predicciones y los diferentes modelos que se pueden representar para este tipo de trabajos, ya que no todos son posibles de aplicar con los datos que se tienen, que también es un muy buen analisis.