



**UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO**

---

**Facultad de Estudios Superiores  
Acatlán**

## **Practica 3:**

*SVM y Arboles de decisión para  
regresiones*

**Aprendizaje de maquina**

**Integrantes:**

Gustavo Adolfo Alvarez Hernández

**Profesor:**

Eduardo Eloy Loza Pacheco



**FES Acatlán, 26 de septiembre 2022**

## Objetivo

Que el alumno aprenda a utilizar otros algoritmos de machine learning para problemas de predicción numéricas, además de mostrar que hay varios algoritmos que son normalmente utilizados en clasificación se pueden implementar para valores numéricos.

## Materiales

- Google colab: Utilizando Sklearn, pandas y matplotlib.

## Resumen

En la presente práctica se utilizarán la máquina de soporte vectorial y arboles de decisión sobre el conjunto de datos “Salarios”, el cual cuenta con salario y años de experiencia, para crear predictores que nos servirá para estimar cual debería de ser el salario de un nuevo empleado en relación de cuantos años de experiencia tiene. Por último, se contrasta el desempeño de los algoritmos y que tan bien se ajustan a los datos.

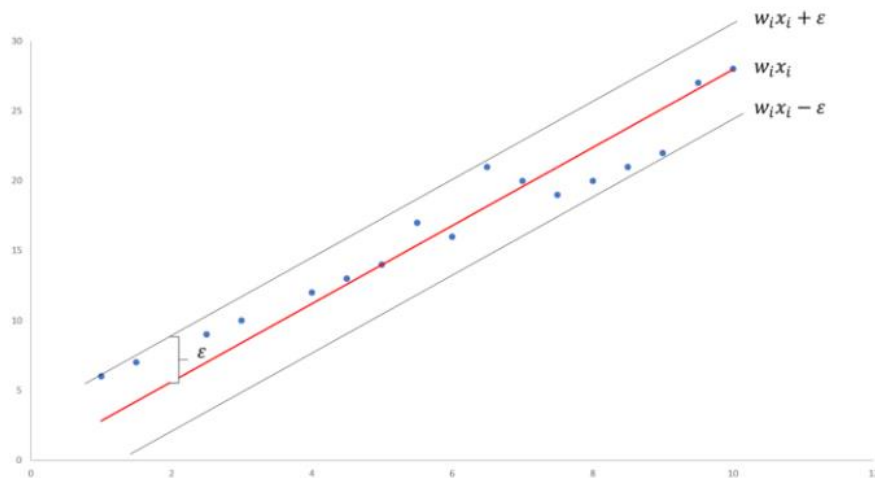
## Antecedentes

El uso de máquinas de soporte vectorial suele ser utilizados para problemas de clasificación, no obstante, tiene su utilidad en los problemas de regresión. Partimos de una regresión lineal simple, la cual obtiene sus parámetros al minimizar los errores al cuadrado o en su defecto OLS.

Ahora hay métodos de regulación que buscan simplificar el modelo a través de penalizar los pesos de nuestros coeficientes y así anulando o reduciendo parámetros. En el caso de SVMR nuestra función a minimizar está dada por los pesos y teniendo como restricción no superar cierto umbral.

$$\text{Min} \frac{1}{2} \|w\|^2, \quad |y_i - w_i x_i| \leq \epsilon$$

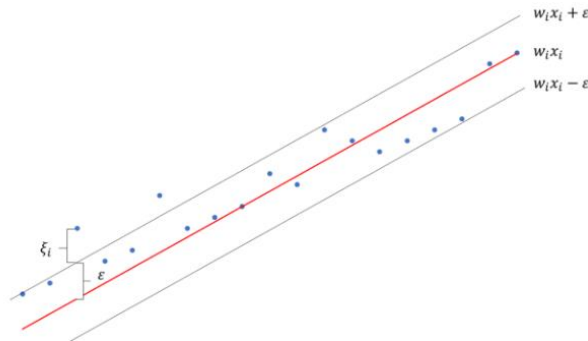
Dando así un hiperplano en donde tenemos la recta o hiperparámetro con sus dos respectivos bordes que representan nuestros errores.



Cuando no se tiene una buena recta, también podemos añadir métodos de penalización, en el cual se añade la distancia de los puntos que no entraron en nuestro hiperplano.

$$\text{Min} \frac{1}{2} \|w\|^2 + C \sum_i |\xi_i|, \quad |y_i - w_i x_i| \leq \epsilon + |\xi_i|$$

Gráficamente se ve así:



En cuanto un árbol de regresión, se busca encontrar los puntos de entrenamiento en donde al dividir nuestros datos, la estimación de la media nos da el menor error cuadrado. Esto es se hace particiones en el dataset y se calcula el MSE, se toma el nodo con menor error y este será el nodo raíz y así se van seleccionando los mejores nodos hasta estimar todos los valores de  $y$ .

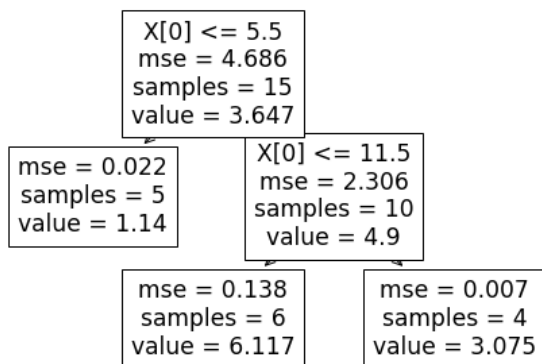


fig 3.1: The resultant Decision Tree

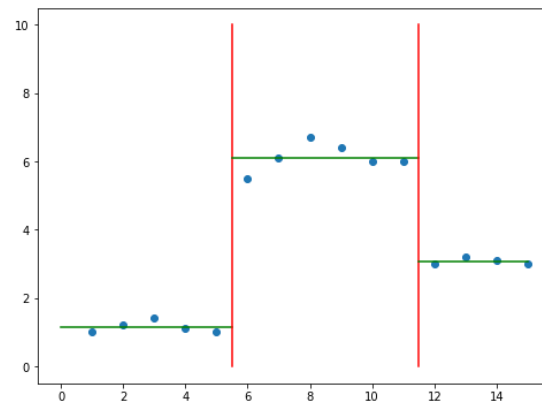


fig 3.2: The Decision Boundary

## Desarrollo

1. Cargamos los módulos necesarios

```
[ ] import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

from sklearn.svm import SVR
```

2. Y utilizamos los datos de salarios respecto al puesto

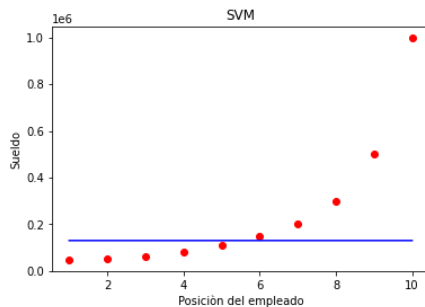
```
[ ] datos= pd.read_csv("/content/Salarios.csv")
X= datos.iloc[:,1:2].values
y= datos.iloc[:, -1].values
```

3. Y sin realizar alguna transformación generamos un SVM

### ▼ Sin escalar

```
[ ] regresion= SVR(kernel= "rbf")
regresion.fit(X,y)
```

4. Procedemos a graficar nuestra estimación para ver su desempeño.

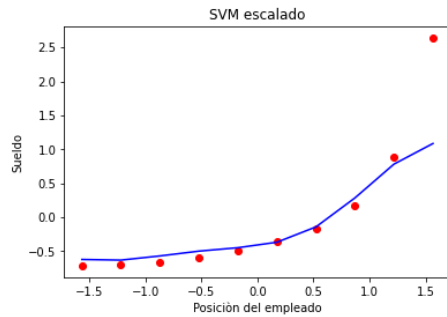


5. Ahora procedemos a escalar los datos para que estén en el mismo intervalo

```
▶ from sklearn.preprocessing import StandardScaler
ssx=StandardScaler()
ssy=StandardScaler()
```

```
[ ] X= ssx.fit_transform(X)
y= ssy.fit_transform(y.reshape(-1,1))
```

6. Generamos el mismo modelo, pero con los datos escalados y dan como resultado



7. Ahora abordamos el mismo problema con los mismos datos, pero con árboles de decisión, entonces importamos los módulos necesarios.

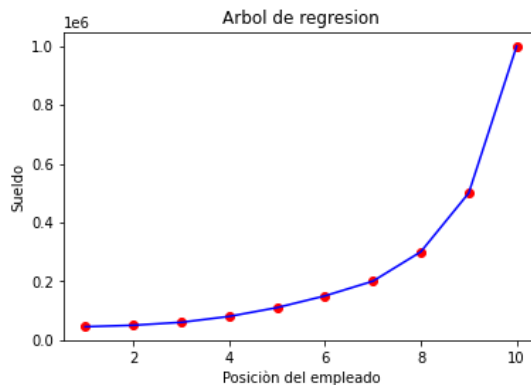
```
[2] import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.tree import DecisionTreeRegressor
```

8. Los datos son los mismos, por lo que procedemos a instanciar el árbol

```
[5] regresion= DecisionTreeRegressor(random_state=0)
regresion.fit(X,y)

DecisionTreeRegressor(random_state=0)
```

9. Y graficamos para ver que tal se ajusta la regresión



## Conclusión

La maquina de soporte vectorial en primera instancia lo hace bastante mal debido que, al no escalar los datos, nos quedan muy distanciados los datos por lo cual establecer límites donde contenga la mayor parte de los datos se vuelve inaccesible. Por tanto, al escalar se compacta el espacio vectorial y la generación del hiper plano y sus límites son mucho más sencillos.

En el caso del árbol de regresión se ajusta bastante bien pues aglomera los datos y como están cercanos entre sí, la media es una buena estimación del valor. Por lo cual se ve un comportamiento que se ajusta a los datos, tanto que es importante preguntarse si es un buen modelo a la hora de generalizar.

## Bibliografía

- [1] T. Sharp, «Towards Data Science,» 3 Marzo 2020. [En línea]. Available:  
<https://towardsdatascience.com/an-introduction-to-support-vector-regression-svr-a3ebc1672c2>. [Último acceso: 20 Septiembre 2022].
  
- [2] A. Prasad, «Analytics Vidhya,» 8 Agosto 2021. [En línea]. Available:  
<https://medium.com/analytics-vidhya/regression-trees-decision-tree-for-regression-machine-learning-e4d7525d8047>. [Último acceso: 20 Septiembre 2022].