UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Facultad de Estudios Superiores
Acatlán

Practica 5:

Máquinas de Soporte Vectorial para clasificación.

Aprendizaje de maquina

Integrantes:

Gustavo Adolfo Alvarez Hernández

Profesor:

Eduardo Eloy Loza Pacheco

FES Acatlán, 17 de octubre 2022

Objetivo

Que el alumno aprenda los fundamentos de las maquinas de soporte vectorial para en consecuente, poner a prueba sus conocimientos al implementar el modelo para clasificar de manera adecuada a un conjunto de datos con variable objetivo-binaria. Además de analizar su desempeño con los algoritmos antes vistos.

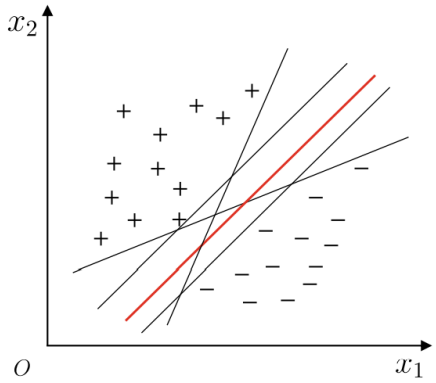
Resumen

En la presente práctica se abarca la problemática de clasificar a usuarios de una presente campaña de publicidad electrónica, con el fin de saber si el usuario compra o no el producto. Esta tarea se realizó por medio de una maquina de soporte vectorial, que hará una separación lineal tomando como características del usuario su salario y edad. Su correspondiente implementación se realiza en Python auxiliándonos de la biblioteca Sklearn para el modelo y Matplotlib para la visualización de los resultados.

El modelo tiene un performance bastante bueno al clasificar en su mayoría de manera adecuada, y permite observar una tendencia a la compra dada una edad y salario grande; y se mantiene este buen comportamiento aún en datos no observados por lo cual generaliza de una manera adecuada el modelo.

Antecedentes

Una maquina de soporte vectorial tiene como tarea principal que, dado un conjunto de datos ya etiquetados, encontrar un hiperplano que logré separar a las clases o etiquetas en el espacio vectorial; pero puede existir varios hiperplanos que logren hacer esta tarea, por lo cual tomaremos el hiperplano que mantenga mayor distancia entre las clases, como se muestra:



Nuestra elección en este particular caso sería la línea roja porque divide a ambas clases y la distancia del plano a cada clase, parece ser considerable y equitativa, este margen se define como la tolerancia. Su construcción depende de encontrar un hiperplano de la forma:

$$w^T x + b = 0$$

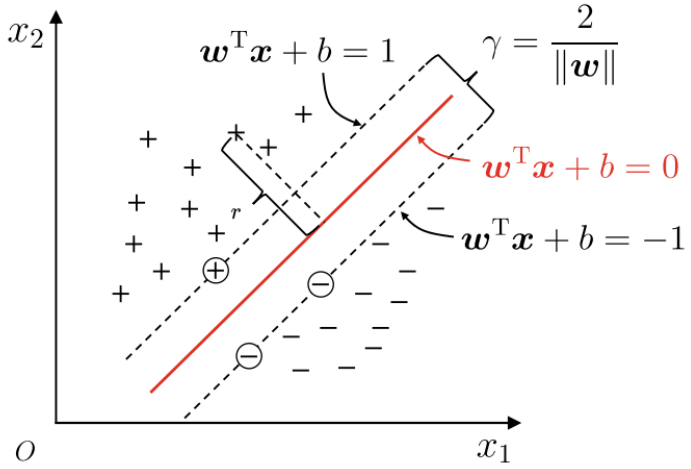
Donde w es el vector normal que controla dirección del hiperplano y b s es el sesgo que determina la distancia entre el mismo y el origen, por lo cual estamos interesados en la distancia de los puntos al hiperplano, la cual se obtiene de la siguiente manera:

$$r = \frac{|w^T x + b|}{||w||}$$

Y por medio de este hiperplano buscamos que arroje respuestas entre [-1,1] para clasificar correctamente a las clases -1 y 1, siendo esto de la forma.

$$\begin{cases} \boldsymbol{w}^\top \boldsymbol{x}_i + b \geqslant +1, & y_i = +1, \\ \boldsymbol{w}^\top \boldsymbol{x}_i + b \leqslant -1, & y_i = -1. \end{cases}$$

Y partiendo de los puntos que se encuentran mas cercanos al hiperplano que llamaremos como vectores soporte, entre ambos habrá una distancia $\gamma = \frac{2}{||w||}$ como se muestra intuitivamente en la imagen:



Y por tanto se trata de un problema de optimización donde queremos maximizar la distancia γ, lo cual es equivalente a nuestra siguiente función de perdida:

$$\min_{\boldsymbol{w},b} \ \frac{1}{2}\|\boldsymbol{w}\|^2$$
$$\text{s.t.} \ \ y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geqslant 1, \quad i = 1, 2, \ldots, m.$$

Desarrollo

1. Importar los módulos

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import plot_confusion_matrix
from matplotlib.colors import ListedColormap
```

2. Cargamos los datos de las campañas de marketing con pandas

```
[ ] data= pd.read_csv("AnunciosRedesSociales.csv")
```

```
data.head()
```

	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19.0	19000.0	0
1	15810944	Male	35.0	20000.0	0
2	15668575	Female	26.0	43000.0	0
3	15603246	Female	27.0	57000.0	0
4	15804002	Male	19.0	76000.0	0

3. Dividimos nuestros datos en entrenamiento y test para tener con que evaluar a nuestro modelo.

```
[ ] x_train,x_test,y_train,y_test= train_test_split(X,y,test_size= .25, random_state=0)
```

4. Al ser un modelo que utiliza distancias, es importante escalar los datos para reducir la dispersión de los datos

```
scx= StandardScaler()
x_train=scx.fit_transform(x_train)
x_test= scx.fit_transform(x_test)
```

5. Instanciamos el modelo de SVM para clasificación y le pedimos que sea una división lineal

```
[ ] from sklearn.svm import SVC
    clasificador = SVC(kernel= 'linear', random_state=0)
    clasificador.fit(x_train, y_train)
```

```
SVC(kernel='linear', random_state=0)
```

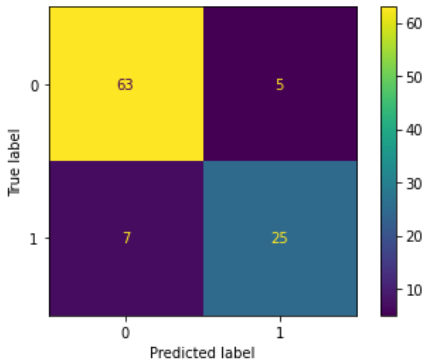
6. Predecimos con los datos test y contrastamos con los valores reales

```
[ ] y_pred = clasificador.predict(x_test)
```

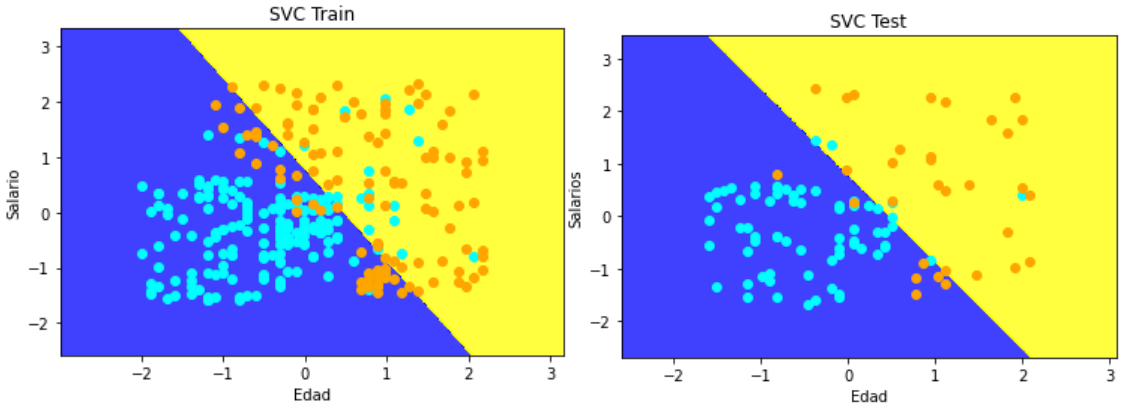
```
y_pred-y_test
```

```
array([ 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  1,  0,
        0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0, -1,  0,  0,
        0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  0,  1,  0, -1,  0,  0, -1,  0,  0,  0,  0,  0,  0,  0,  0,  0,
        0,  1,  0,  0,  0, -1,  0,  0,  1,  0,  0,  0,  0,  1,  0,  0,  0,
        0,  0,  0, -1,  0,  0,  0,  0,  0,  0, -1,  0, -1,  0,  0])
```

7. A manera de facilitar la lectura generamos una matriz de confusión



8. Por último, graficamos los datos de train y test para ver como se realizo la separación y en cuales valores acertó o falló.



Conclusiones

Como podemos observar la cantidad de errores al clasificar están presentes, debido a que los datos no son del todo linealmente separables, añadiendo que hay datos que no se llegan a comportar como esperaría, hay varios puntos azules que tienen una edad y salario alto y aun así no lo compra el producto.

No obstante, el modelo hace un buen desempeño a la hora del test, además de que permite identificar patrones claros a la hora de comprar o no el producto, por lo cual parece ser que el modelo generaliza y que tiene pocos fallos, por lo cual es buen modelo y se tiene la mejor división lineal posible.

Referencias

[1] Z.-H. Zhou, Machine Learning, Singapore: Springer Nature Singapore, 2016.