
# Trabajo sobre la implementación de modelos de aprendizaje automático supervisado.

Realizado por el grupo 19

**Integrantes:**
Juan del Junco Ollero
Emilio José Nicasio Díaz

# Indice

# Resumen del problema y consideraciones previas

*Realizado por Emilio José Nicasio Díaz.*

Todas las gráficas mostradas en el documento están sacadas y pueden encontrarse en el archivo .ipynb correspondiente. En este documento se plasman para que se tomen en cuenta a la hora de la lectura, pero para observarlas con todo detalle y/o se quisieran modificar (tamaño, limites x o y, temas de visibilidad) aconsejamos encarecidamente que se visualicen desde la implementación.

A la hora de realizar la implementación, se tendrá en cuenta los apartados mencionados en el enunciado, los cuales son:

**1. Análisis inicial de los datos.**

- o Identificar tipo de variables (numéricas, categóricas).
- o Estudio de la existencia de valores perdidos y sustitución de los mismos.
- o Considerar si es necesario aplicar preprocesamiento: normalización, estandarización.
- o Justificar y explicar las decisiones tomadas.
- o Incluir visualizaciones que ayuden a entender los datos.

**2. Implementación desde cero de los tres modelos, sin utilizar librerías de aprendizaje automático.**

- o Se pueden emplear las librerías usadas en clase para operaciones matemáticas y visualización de resultados.
- o Solo se permite el uso de la función de optimización fmin_cg para regresión logística y red neuronal.
- o Para cada modelo se debe incluir:
  - ▪ Inicialización de los parámetros.
  - ▪ Fase de entrenamiento, incluyendo ajuste de parámetros.
  - ▪ Fase de predicción.
  - ▪ Evolución de la función de coste a lo largo del entrenamiento (gráfico de convergencia).

▪ Métrica de evaluación apropiada.

**3. Entrenamiento y evaluación.**

o División del conjunto de datos en datos de entrenamiento y datos de test. Elegir según convenga validación cruzada o

holdout, justificando las decisiones tomadas.

o Cálculo de las métricas adecuadas, por ejemplo:

▪ Para regresión lineal: error absoluto medio (MAE), error cuadrático medio (MSE), error relativo medio en

porcentaje (MAPE).

▪ Para regresión logística y red neuronal: tasa de acierto y matriz de confusión.

**4. Comparación y análisis de resultados.**

o Comparación del rendimiento de los modelos. Se valora positivamente introducir los resultados en tablas y de forma gráfica. Tenga en cuenta que el análisis no se debe resumir a frases como “se ha obtenido un 90% de precisión”, sino que debe tratarse de un análisis en profundidad para llegar a interpretar por qué el modelo obtiene un 90% de precisión o por qué se obtiene un 10% de error.

o Comentar ventajas e inconvenientes observados en cada caso.

Para dicha implementación, se tendrán en cuenta las restricciones impuestas en el enunciado, tales como:

- El lenguaje en el cual se realiza la implementación es en **Python**.
- **No** se permite el uso de librerías que incluyen modelos de forma directa. **Si** se permite el uso de: **Numpy**, **Pandas** y **Matplotlib** o **Seaborn**.
- La entrega debe estar compuesta de **tres notebooks**, uno para cada modelo. El notebook que requiera de modificaciones (en nuestro caso, **Regresión lineal**) debe tener incluida dicha modificación. Además del código, hay que aportar este documento, donde debe recogerse toda la información sobre los datos a tratar, las implementaciones realizadas y las decisiones tomadas.

# Introducción y objetivos

*Realizado por Juan del Junco Ollero y Emilio José Nicasio Díaz.*

Nuestro trabajo se basa en un fichero csv que trata sobre los datos de 8000 personas en la plataforma Spotify. Podemos ver que el archivo a partir del cual elaboraremos nuestros modelos muestra sus datos en doce columnas diferenciadas:

- user_id: El identificador único de cada usuario. Dado que no tenemos constancia de que su valor pueda tener relevancia para el estudio (por ejemplo, si un número menor denotara mayor antigüedad del usuario) no será considerado como variable.
- gender: El género del usuario. Variable categórica.

```
Female: 2659    |    Male: 2691    |    Other: 2650
```

- age: La edad del usuario. Variable numérica.

```
16: 172    |    17: 177    |    18: 185    |    19: 181    |    20: 202
21: 199    |    22: 200    |    23: 175    |    24: 176    |    25: 168
26: 173    |    27: 146    |    28: 173    |    29: 151    |    30: 192
31: 173    |    32: 184    |    33: 186    |    34: 182    |    35: 179
36: 177    |    37: 202    |    38: 170    |    39: 173    |    40: 170
41: 196    |    42: 199    |    43: 173    |    44: 176    |    45: 168
46: 192    |    47: 190    |    48: 188    |    49: 200    |    50: 184
51: 207    |    52: 190    |    53: 185    |    54: 173    |    55: 174
56: 170    |    57: 202    |    58: 183    |    59: 184
```

- country: Localización del usuario. Variable categórica.

```
AU: 1034    |    CA: 954    |    DE: 1015    |    FR: 989
IN: 1011    |    PK: 999    |    UK: 966    |    US: 1032
```

- subscription_type: Tipo de suscripción del usuario. Variable categórica.

```
Family: 1908    |    Free: 2018    |    Premium: 2115    |    Student: 1959
```

- listening_time: Minutos que el usuario dedica al día a escuchar música. Variable numérica.
- songs_played_per_day: Número de canciones que el usuario escucha al día. Variable numérica.
- skip_rate: Porcentaje de canciones que el usuario elige saltar. Variable numérica.
- device_type: Dispositivo empleado por el usuario. Variable categórica.

```
Desktop: 2778    |    Mobile: 2599    |    Web: 2623
```

- ads_listened_per_week: Número de anuncios que escucha el usuario cada semana. Variable numérica.
- offline_listening: Si el usuario emplea el modo offline de la aplicación. Variable categórica. 0 negativo, 1 afirmativo.

```
0: 2018     |     1: 5982
```

- is_churned: Si el usuario está activo en la aplicación o se ha dado de baja. Será el objetivo de estudio de estos modelos. 0 activo, 1 dado de baja.

```
0: 5929     |     1: 2071
```

El objetivo del trabajo es:

- Entender los datos del dataset: que datos tienen relación con cuales, cuales son las mejores columnas para predecir qué atributo, etc
- A partir de un estudio preliminar o a través de pruebas, determinar, para una clase, los atributos que la van a predecir.

Se deben elegir una clase para cada modelo. Nos hemos decantado por:

En el caso de la regresión lineal, se predecirá la columna “ads_listened_per_week” con las columnas “subscription_type”, “listening_time”, “songs_played_per_day”, “skip_rate”, “device_type”. Se utilizarán estas para comprobar si el tipo de la suscripción tiene relacion con los anuncios (tiene una baja relación que se explica en la descripcion del conjunto de datos), si el uso de la aplicacion (tiempo de escucha, canciones y el numero de saltos) influye e incluso si el tipo de dispositivo usado puede ser una variable (ya que se ha comprobado que en Spotify existen dispositivos en los que se reproducen más anuncios que en otros, con el mismo tiempo de uso).

En el caso de la regresión logística, se predecirá la columna “is_churned”, para comprobar de qué depende el abandono del servicio.

En el caso de las redes neuronales, se predecirá la columna “subscription_tipe”, para comprobar si el tiempo de escucha, el número de canciones que oye y la edad están relacionados con el tipo de suscripción.

# Descripción del conjunto de datos

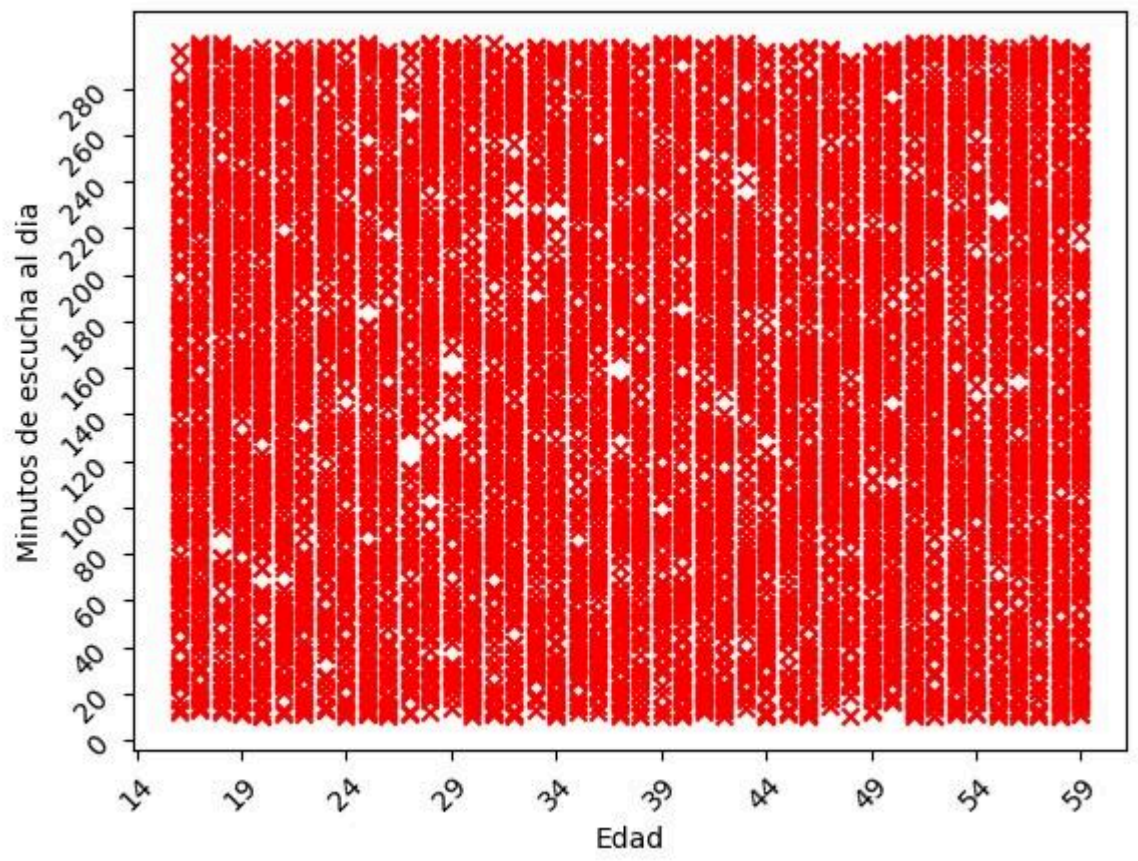
*Realizado por Emilio José Nicasio Díaz.*

(Toda la implementación descriptiva del conjunto de datos, la búsqueda de relaciones y posibles columnas relacionadas se encuentra en la implementación de la **regresión lineal**.)

Antes de empezar a implementar modelos, es muy importante que se entienda como son los datos, pues no podemos predecir cualquier clase con cualquier característica.

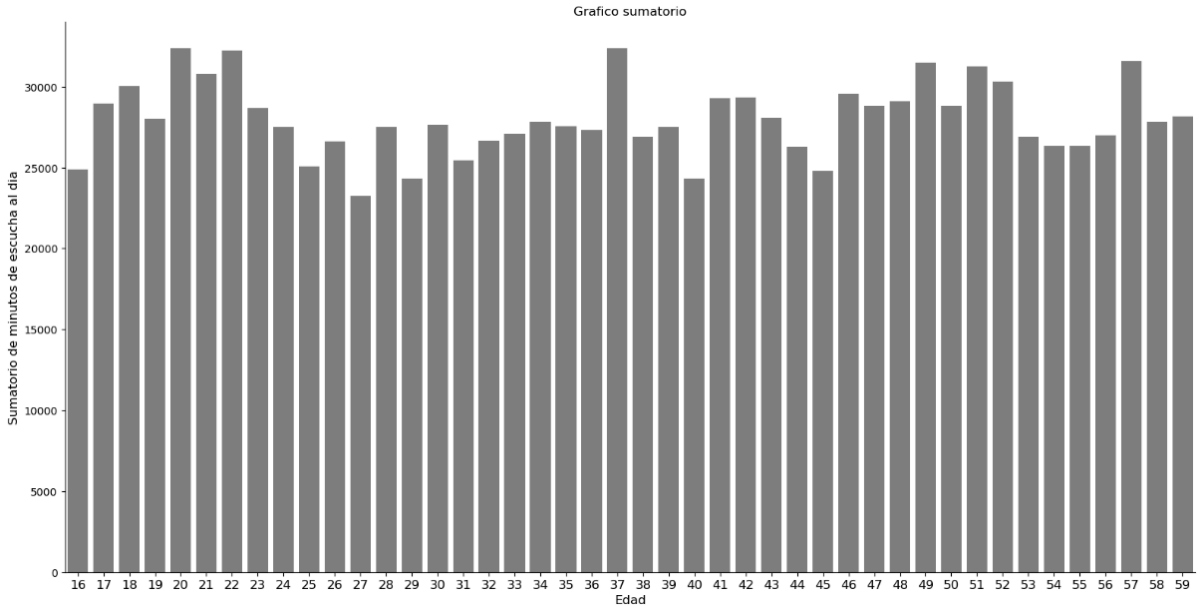
Para entender los datos, creemos que la mejor forma es de manera visual, a si que en primera instancia se han comparado las diferentes columnas del dataset contra otras columnas, para comprobar si existía o no una relación.

Las primeras columnas que se han comprobado han sido: “age” y “listening_time”, puesto que podría existir una clara relación, los estudiantes suelen escuchar música con asiduidad, mientras que las personas mas adultas podrían no estar tan acostumbradas al uso de este tipo de aplicaciones, por este motivo se ha decidido investigar la relación entre estas 2 columnas. La gráfica resultante es la siguiente:

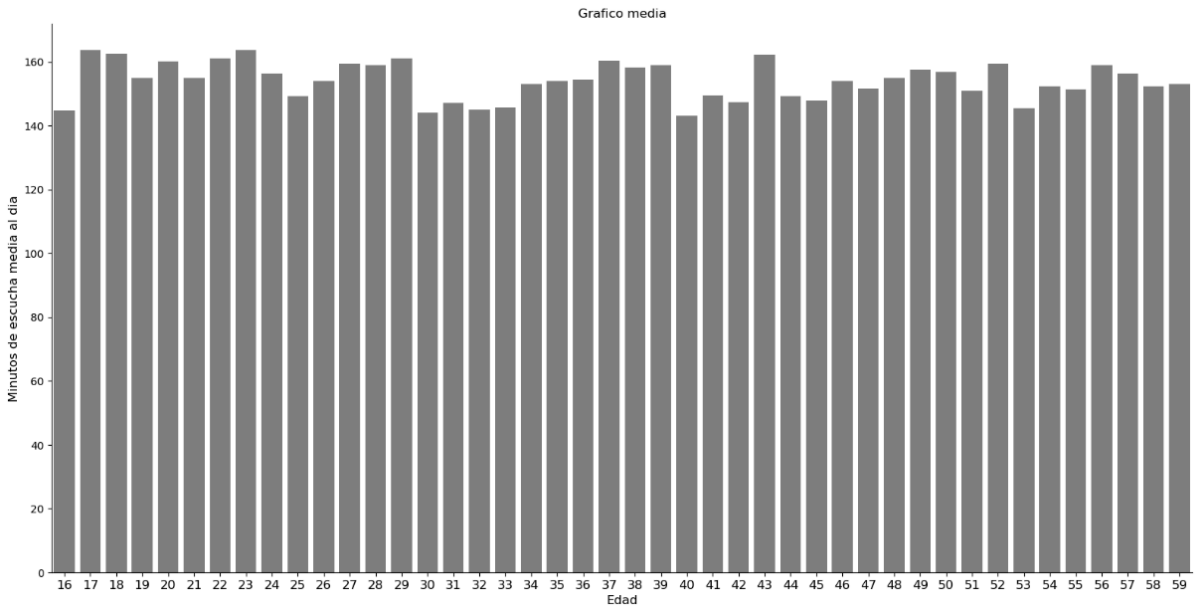


Lo cierto es, que basándonos en la información de la gráfica, no hay diferencia visible entre una persona joven y una adulta en términos de minutos de escucha.

Para afianzar este resultado y evitar posibles problemas por puntos superpuestos (ya que podría darse que la zona de edad menor tenga mas puntos, pero al estar superpuestos no se distingan), se ha realizado otra gráfica con el sumatorio de los minutos por edades:



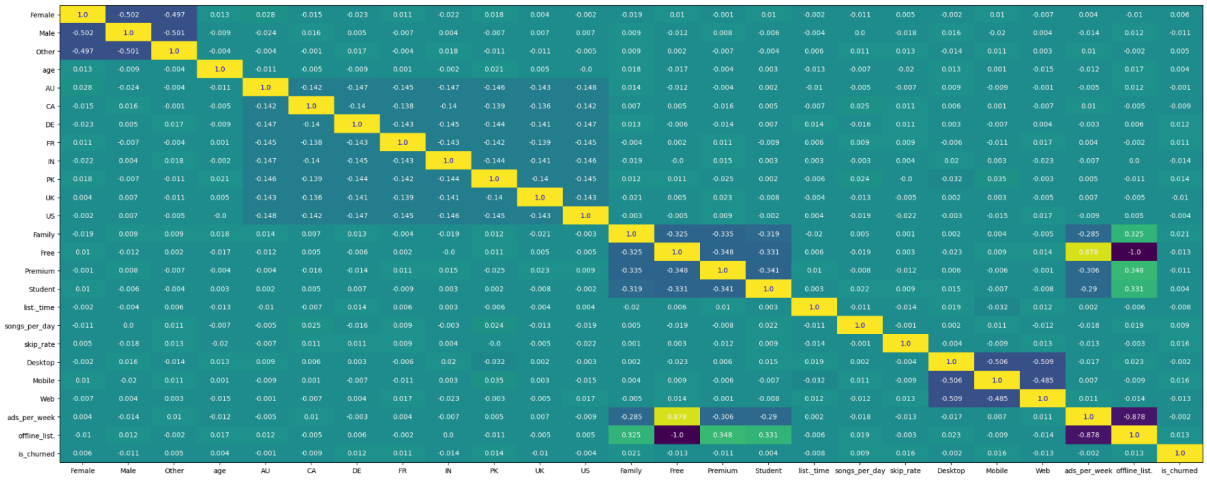

Esta grafica no es decisiva, ya que pueden existir edades con un numero de instancias dispares, sin embargo si todas estan en un rango similar (la cantidad de instancias por edad está entre 150 y 200 en el dataset), una media si es descriptiva del problema:



En esta gráfica se puede observar que no hay diferencia plausible entre una persona de mayor edad y una de menor edad.

Para estar seguros de que no existe ninguna relación entre columnas, se han comprobado el resto de columnas contra todas las demás.

Mapa de calor de las correlaciones de Pearson:




Se ha comparado con el coeficiente de correlación de Pearson (se ha hecho también con gráficas, pero está comentado en la implementación puesto que la información es la misma y los resultados son más claros con la correlación) y el resultado ha sido similar a las gráficas presentadas en el documento, lo que significa que ninguna columna tiene una relación fuerte con ninguna otra, con la excepción de la columna anuncios por dia, que tiene buena relación con el tipo de subscripción y con si tienen el modo offline activado.

Esta relación es lógica, ya que normalmente las plataformas del estilo de Spotify ofrecen una subscripción de pago para evitar los anuncios, y si se tiene el modo sin conexión activado sucede lo mismo. Por lo tanto estas columnas podrían servirnos para saber cuando un usuario no va a tener anuncios con total seguridad, pero para una persona con una subscripción gratuita y que esté con conexión no podemos predecir cuántos anuncios le saldrá con estás columnas. Por lo tanto, aunque tengan una buena relación, la información que otorga no es suficiente para hacer un buen modelo.

Con esto finalizamos diciendo que no existe una buena relación directa entre una columna y otra, por lo tanto carece de sentido implementar modelos univariables, por lo que haremos modelos multivariables.

# Preprocesamiento aplicado

*Realizado por Juan del Junco Ollero y Emilio José Nicasio Díaz.*

El primer paso en el preprocesado de los datos será la búsqueda de valores perdidos, esto es, aquellos datos que se muestren vacíos (valor nulo) en la vista del conjunto. La misma librería Panda empleada nos permite analizar rápidamente nuestro Archivo Separado por Comas (.csv):

```
spotify_file_path = 'spotify_churn_dataset.csv'
spotify_data = pd.read_csv(spotify_file_path)

df = pd.DataFrame(spotify_data)

totalPerdidos = df.isnull().sum()

print(totalPerdidos)
```

✓ 0.1s

```
user_id                 0
gender                  0
age                     0
country                 0
subscription_type       0
listening_time          0
songs_played_per_day    0
skip_rate               0
device_type             0
ads_listened_per_week   0
offline_listening       0
is_churned              0
dtype: int64
```

Como podemos comprobar, el archivo no presenta datos nulos, por lo que es innecesaria realizar una sustitución de los mismos.

Posteriormente, tenemos que decidir si hay que normalizar o estandarizar los datos. En la mayoría de casos va a ser necesario, ya que (por ejemplo) la edad y el índice de salto de canciones están en valores muy distintos (edad entre 10 y 70, skip_rate entre 0 y 1).

En regresión lineal: Ha sido necesario normalizar los valores, que (en la implementación final) se quiere predecir el número de anuncios a la semana, que tiene valores en el rango

de 0 a 60, mientras que se utiliza para predecir columnas como: el tiempo de escucha diario (de 20 a 300), el número de canciones diario (de 5 a 100), el ratio de salto de canciones (de 0 a 1), y más. Por esto se ha decidido normalizar los datos.

En regresión logística:

En redes neuronales: Ha sido necesario normalizar los valores, que se quiere predecir la columna “is_churned”, que tiene valores que pueden ser 0 o 1, mientras que se utiliza para predecir columnas como: el tiempo de escucha diario (de 20 a 300), el número de canciones diario (de 5 a 100) y el número de anuncios por semana (de 0 a 49). Por esto se ha decidido normalizar los datos.

# Resultados y comparativa

## Resultados de la implementación de la Regresión Lineal

*Realizado por Emilio José Nicasio Díaz.*

Como ya se expuso en el apartado: “Descripción del conjunto de datos”, el dataset no tiene columnas con buena relación entre ellas. Tampoco nos da pistas sobre qué conjunto de columnas podrían tener una relación oculta que nos permita predecir correctamente otra columna. Por lo tanto se ha realizado una implementación inicial, la cual se ha realizado sobre todo para realizar el estudio del dataset, una implementación con un conjunto de datos distinto al de Spotify, para validar la veracidad de la implementación, y una implementación final que contiene todos los aspectos que se piden en el enunciado (división train-test, validación cruzada, cálculo del error, …).

### Implementacion 1

Para la primera implementación se han tomado las siguientes consideraciones:

Dataset: “spotify_churn_dataset.csv”.
Clase: “age”.
Columnas para predecir: “listening_time”, “songs_played_per_day” y “skip_rate”.
¿Por qué estas columnas?: Para comprobar si dado su tiempo medio diario, sus canciones escuchadas por días y su índice de cambio de canciones se puede predecir.

En esta implementación se estudió sobre todo el conjunto en inicio, como resultaba coger columnas aun sabiendo que en principio no guardarían relación, ver como se adecuaría el conjunto y que resultados mostraría.

La gráfica que se adjunta es una comparación de cada usuario, con su valor real (azul) y predicho (rojo):

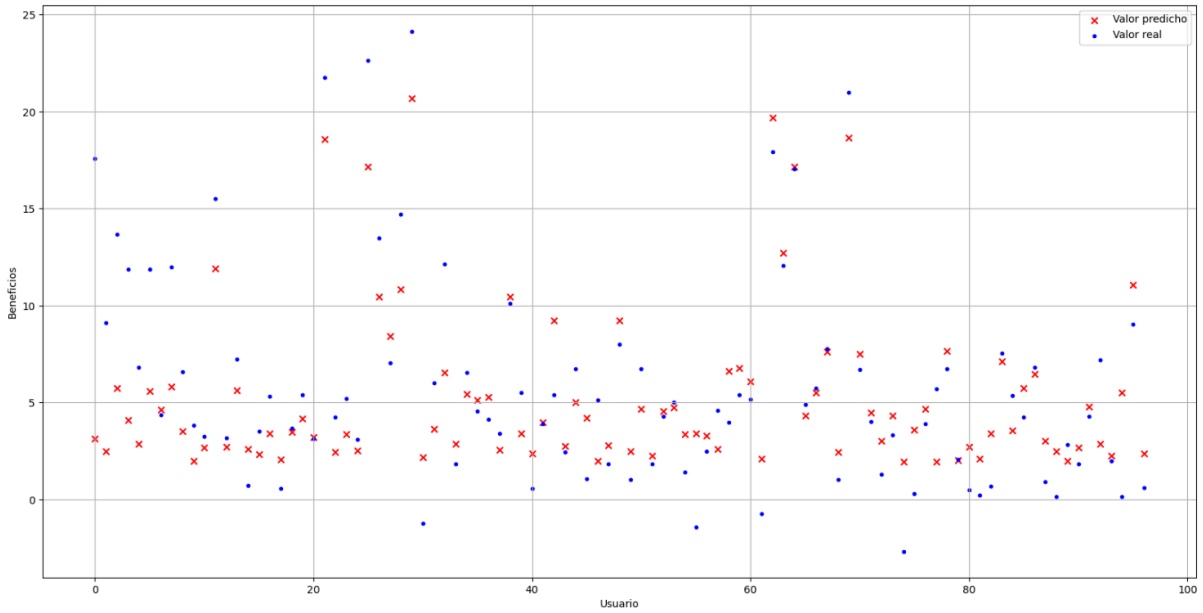

Como se puede apreciar, el modelo predice a la media independientemente de los valores de sus columnas predictoras. Para confirmar si el problema era de implementación o del conjunto de datos, se ha realizado una segunda implementación con otro conjunto de datos, el cual si tiene columnas relacionadas.

## Implementacion 2

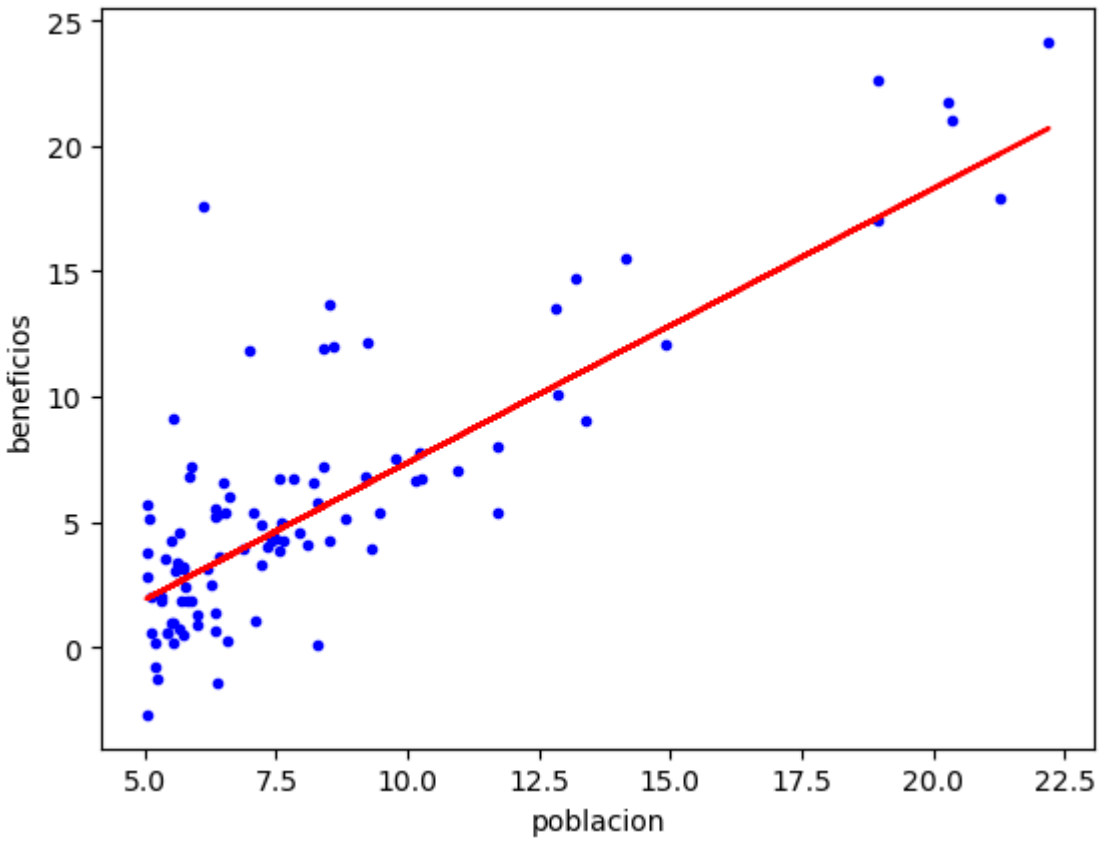
Para la segunda implementación se han tomado las siguientes consideraciones:

Dataset: “ex1data1.txt”.
Clase: “beneficio”.
Columnas para predecir: “poblacion”.
¿Por qué estas columnas?: Realmente tomar una columna para clase u la otra resulta en el mismo resultado, puesto que la correlación es la misma.

Como se comentó, se va a probar con otro conjunto de datos, el cual tiene dos columnas y el modelo será univariable. Como se ha mostrado en la implementación, la correlación de estas dos columnas es muy buena (0.80), por lo tanto si esta vez el modelo no es correcto el problema sería de la implementación. Se realiza una grafica similar a la  de la anterior implementación, comparando para cada instancia su valor real con el predicho:




También, como el modelo es univariable, puede pintarse su línea de regresión:



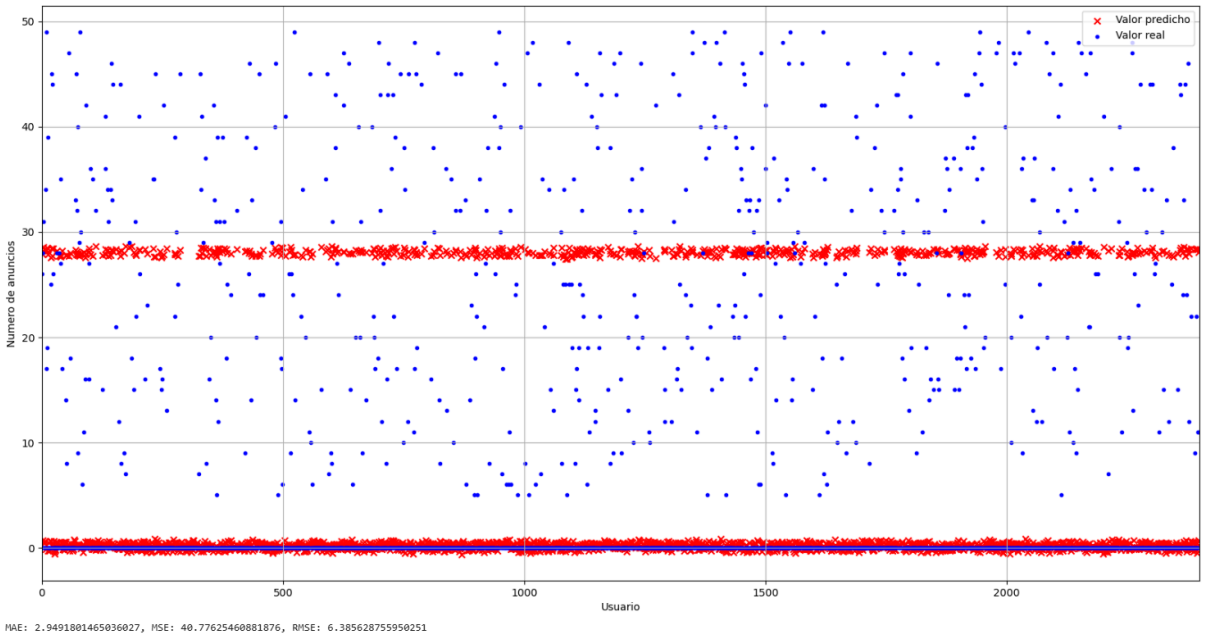

Se observa que la linea se adecua de buena forma y que las predicciones no son perfectas pero son buenas, por lo tanto se deduce que el problema de la primera implementación es el dataset dado, que no cuenta con suficientes relaciones como para formar un modelo robusto. Con todo, se realizará una tercera implementación que aplique todo lo aprendido sobre el dataset de Spotify y que en términos de implementación contará como la implementación definitiva del modelo de regresión lineal.

## Implementacion 3

Para la tercera implementación se han tomado las siguientes consideraciones:

Dataset: “spotify_churn_dataset.csv”.
Clase: “ads_listened_per_week”.
Columnas para predecir: “subscription_type”, “listening_time”, “songs_played_per_day” y “skip_rate”, “device_type”.
¿Por qué estas columnas?: Se utilizarán estas columnas para comprobar si el tipo de la suscripción tiene relación con los anuncios (tiene una baja relación que se explicó en la descripción del conjunto de datos), si el uso de la aplicación (tiempo de escucha, canciones y el número de saltos) influye e incluso si el tipo de dispositivo usado puede ser una variable (ya que se ha comprobado que en Spotify existen dispositivos en los que se reproducen más anuncios que en otros, con el mismo tiempo de uso).

Imagen de la gráfica final de la implementación:




Sobre las consideraciones del enunciado:

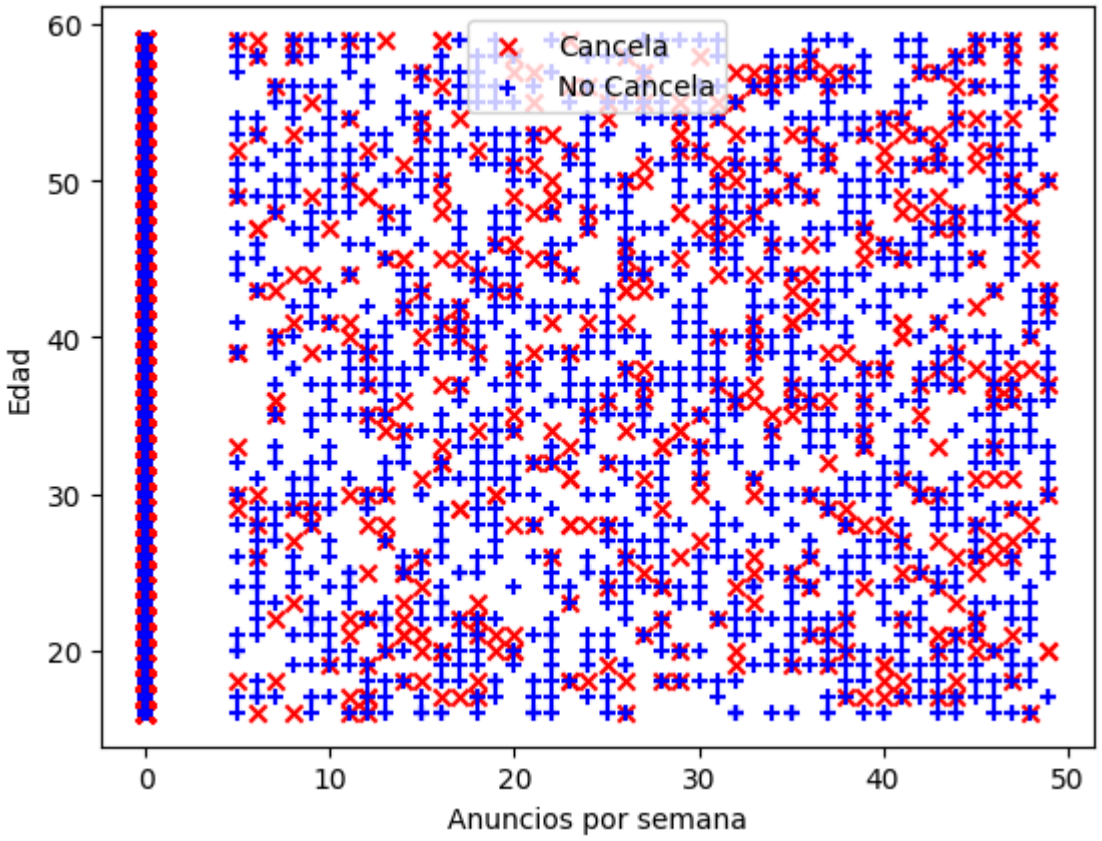
- Se ha dividido el conjunto de datos en entrenamiento y test antes de proceder a la creación del modelo. Esto permite probar el modelo, una vez ha sido terminado su entrenamiento, con un conjunto que no ha tocado nunca, pudiendo asi medir valores de errores más reales que si se hiciera el test con el conjunto con el que se entrenó el modelo.
- Se ha utilizado para entrenar el modelo validación cruzada, puesto que permite entrenar el modelo con mayor cantidad de datos de validación. Además, los resultados en validación cruzada son más homogéneos, para distintas iteraciones, que para hold-out (en hold-out pueden darte 2 iteraciones unos resultados parecidos y en una tercera un resultado muy dispar, mientras que en cross validation los resultados suelen ser más similares).
- Se han contemplado también los errores MAE, MSE y RMSE. Antes de poder hablar de ellos hay que entender sus diferencias.
  MAE es el error absoluto medio, es decir, se calcula el error para una instancia (y - h), se dispone en valor absoluto (para evitar cancelaciones. Por ejemplo, sin valor absoluto, un error de 100 y otro de -100 se cancelarían. Con el valor absoluto esto se evita siendo un error total de 200), se suma al resto de errores y se calcula la media.
  MSE es muy similar, con la diferencia de que al error no se le aplica el valor absoluto, si no el cuadrado. Esto tiene una doble función: elimina cancelaciones (como el valor absoluto) y dispara los errores altos. Un error muy alto podría verse

mermado en el MAE si existían otros errores menores que lo redujeran en la media, en el caso del MSE, un error alto será aún más alto, por lo que tendrá mucha repercusión incluso haciendo la media. Por esto se considera que MSE es mas sensible a errores que el MAE.
RMSE es muy parecido a MSE, con la única diferencia que a la media se le aplica una raiz cuadrada. Esto sirve para reducir un tanto la sensibilidad extrema del MSE.
Por sensibilidad a los errores se podrían ordenar (de menor a mayor):
MAE -> RMSE -> MSE.
Explicados los tipos de errores, tras observar los valores que tienen estos en la implementación nos podemos dar cuenta de que, naturalmente MSE tiene que ser más grande que MAE, pero es mucho más grande, lo que significa que han existido muchos fallos de gran tamaño. Además como el MAE es muy pequeño podemos deducir que se trata por la tendencia del modelo a predecir los datos en la media, queriendo reducir el error general, pero teniendo fallos extremos (por ejemplo, que un valor real esté en el máximo de los valores de la clase), lo que incrementa en gran medida el MSE. El RMSE no nos aporta mucha más información solo que, como es más grande que el MAE, nos recalca que se ve influenciado por valores de error muy altos. Si no existieran tantos errores altos (por ejemplo, en vez de existir picos de errores altos y pequeños, todos los errores fueran similares, siendo el total de errores igual), MSE sería más pequeño y a su vez RMSE también aunque MAE no cambie.

## Resultados de la implementación de la Regresión Logistica

*Realizado por Juan del Junco Ollero.*

Para este ejercicio intentaremos predecir si el usuario es más dado a cancelar (is_churned) en función de su edad (age) y el número de anuncios que escucha por semana (ads_listened_per_week), habiendo considerado que son las variables para las que más fácilmente puede la empresa corregir su modelo con el objetivo de aumentar la permanencia de los usuarios.




Aunque en una primera vista del conjunto no se lograban apreciar patrones claros, al finalizar se puede comprobar que a mayor edad y anuncios por semana, mayor es la probabilidad de cancelación del servicio, siendo los anuncios por semana el parámetro más influyente.

La interpretación de interés para la compañía sería que, en vistas a aumentar la permanencia de los usuarios, deberían, si no reducir los anuncios que les hacen escuchar, tomar la medida de redirigirlos al público de edad más avanzada, que es menos dado a cancelar por ello probablemente por estar los jóvenes más

acostumbrados al cambio de plataforma y ser más conscientes de las opciones y alternativas disponibles.

# Resultados de la implementación de la Red Neuronal

*Realizado por Emilio José Nicasio Díaz.*

Para la implementación de la red neuronal se ha tenido en cuenta lo aprendido en la exploración de los datos realizada en la regresión lineal. Con esto en mente, se ha tomado la columna “is_churned” como clase de estudio y las columnas “listening_time”, “songs_played_per_day” y “ads_listened_per_week” como clases predictoras. Se han tomado estas clases porque podría ser muy útil para la empresa a cargo de Spotify saber si los clientes que se van tienen relación con el uso que le dan a la app (tiempo de escucha y canciones escuchadas por día) y el número de anuncios que oyen.

Como aclaración, en la implementación, concretamente en la función training, existe una variable llamada maxiter, que delimita el número máximo de iteraciones que pueden haber en dicho entrenamiento. Se ha probado con diferentes valores de maxiter y se ha llegado a la conclusión de que, para este problema concreto, con una única iteración funciona igual de bien que para más iteraciones. Se intenta reducir el número de iteraciones porque cada una de ellas eleva el tiempo de cómputo bastante, y se debe aplicar el entrenamiento 11 veces (10 para comprobar cuál es el número óptimo de neuronas en la capa oculta y 1 para el entrenamiento final, con todos los datos de entrenamiento). Se muestran los datos resultantes para los valores probados de maxiter:

Para maxiter = 10

```
**** El número de neuronas de la capa oculta óptimo es:  1
**** Con esas neuronas en la capa oculta, el accuracy del conjunto de validación es:  73.21428571428571
         Current function value: 0.574469
         Iterations: 10
         Function evaluations: 27
         Gradient evaluations: 27
Accuracy del conjunto de test:  74.70833333333333
```

Para maxiter = 1

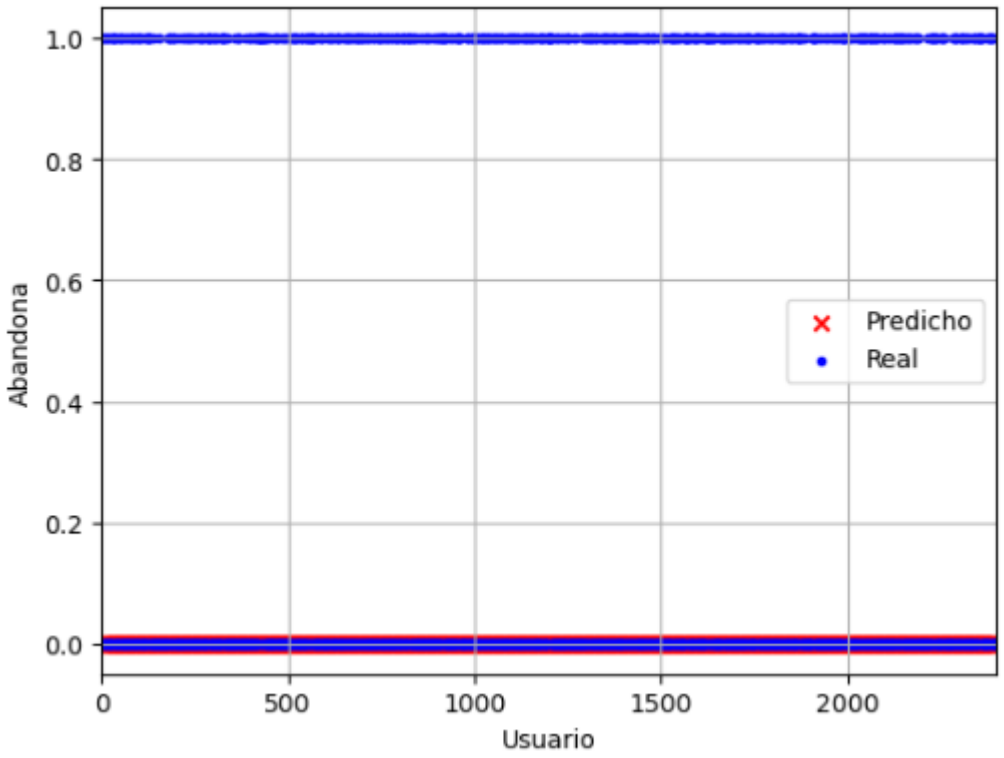
```
**** El número de neuronas de la capa oculta óptimo es:  1
**** Con esas neuronas en la capa oculta, el accuracy del conjunto de validación es:  74.52380952380952

         Current function value: 0.575606
         Iterations: 1
         Function evaluations: 3
         Gradient evaluations: 3

Accuracy del conjunto de test:  74.875
```

Como los resultados son prácticamente 74% de accuracy siempre, se deja por defecto maxiter a 1 para reducir el tiempo de cómputo.

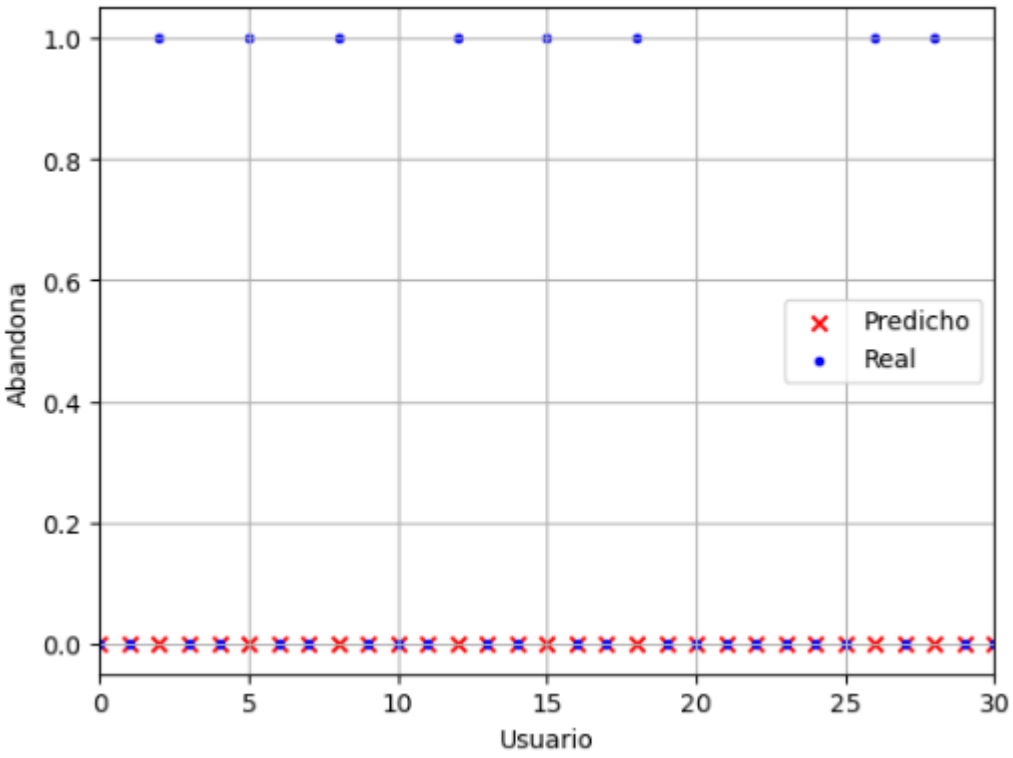
Tal como se ve en las dos imágenes anteriores, para el conjunto de test se obtiene un 74% de aciertos. Esto es una buena noticia, ya que el modelo es capaz de predecir

aproximadamente 3 de cada 4 veces cuando un usuario va darse de baja o cuando va a quedarse, pero como conocemos el conjunto de datos y sabemos por experiencia previa que no tienen buenas relaciones sus columnas, vamos a plasmar gráficamente los resultados predichos contra los resultados reales a ver si existe algo anómalo:

Gráfica obtenida al imprimir para X entre 0 y el tamaño del conjunto de test:




Gráfica obtenida al imprimir para X entre 0 y 30 (para observar mejor qué ocurre):




¿Qué es lo que está ocurriendo? El modelo, como comenté antes, es complicado que encuentre unas relaciones, ya que el dataset parece no tener ninguna relación entre sus columnas (como ya se mostró con el mapa de calor de los coeficientes de correlación de Pearson), por tanto, se ha adecuado a la media, como sucedió en la regresión lineal. Como se podrá observar, hay mayor cantidad de valores que son 0 en los valores reales, si se recuerda, en la introducción se comenta el número de instancias con valores distintos que existen para cada variable, en el caso de is_churned era:

```
0: 5929     |     1: 2071
```

Esto quiere decir que, en el conjunto total, un 75% (aproximadamente) de los usuarios tenían como valor de esta columna 0. Si el modelo se ajustara a la media de los valores de todos los usuarios, probablemente le saldría un valor cercano a 0.25, lo cual al redondearse para dar una solución binaria, saldría que la predicción es siempre 0, tal y como sucede en el modelo implementado.

Por tanto, aunque el accuracy del modelo sea bastante alto (75%) y realmente se adecue a los datos (porque representa esta alta tendencia de tener en esa columna dicho valor), no se puede decir tampoco que sea un buen modelo, porque en verdad no está teniendo en cuenta los valores de entrada, independientemente de ellos el modelo predecirá 0. De hecho, como última prueba, se predijo el valor de la clase para todo el conjunto (8000 valores) y el número de ceros en dicha predicción fue igual al tamaño del conjunto, 8000.

## Tabla comparativa de los errores

*Realizado por Juan del Junco Ollero y Emilio José Nicasio Díaz.*

Siendo obtenidos estos errores/accuracy al predecir con el conjunto de test

| | Regresión Lineal | Regresión Logística | Redes Neuronales |
|---|---|---|---|
| MAE | 3.15460 | - | - |
| MSE | 46.12161 | - | - |
| RMSE | 6.79128 | - | - |
| Accuracy | - | 2% (*) | 74.875% |

(*) Realmente, el accuracy debe de ser de 74.875, pero dada la implementación de la regresión logística nos da ese valor extraño de 2%. Decimos que debería dar 74.875 porque se evalúa la misma clase que en la red neuronal, y por tanto debería darse el mismo caso (prediciendose la media y acabando por predecir siempre en 0). Con esto aclaramos que conocemos el error, aunque en la implementación no esté reflejado.

# Modificaciones

*Realizado por Emilio José Nicasio Díaz.*

## Apartado 1:

En la implementación, se ha tomado como clase "ads_listened_per_week", y como columnas predictoras "listening_time", "age", "offline_listening" y "songs_played_per_day".

**i.** ¿Qué variable se comporta como mejor predictora?

La mejor columna predictora ha sido offline_listening. Esto es debido a que el error que comete es menor. No es perfecta, tal como se ve en la gráfica y por lo explicado a lo largo de la investigación del dataset.

```
Error implementación 1: 196.61586604188022
Error implementación 2: 187.0839178740751
Error implementación 3: 43.818079019715164
Error implementación 4: 189.47560288856724
El menor error es 43.818079019715164, de la implementacion 3.
```

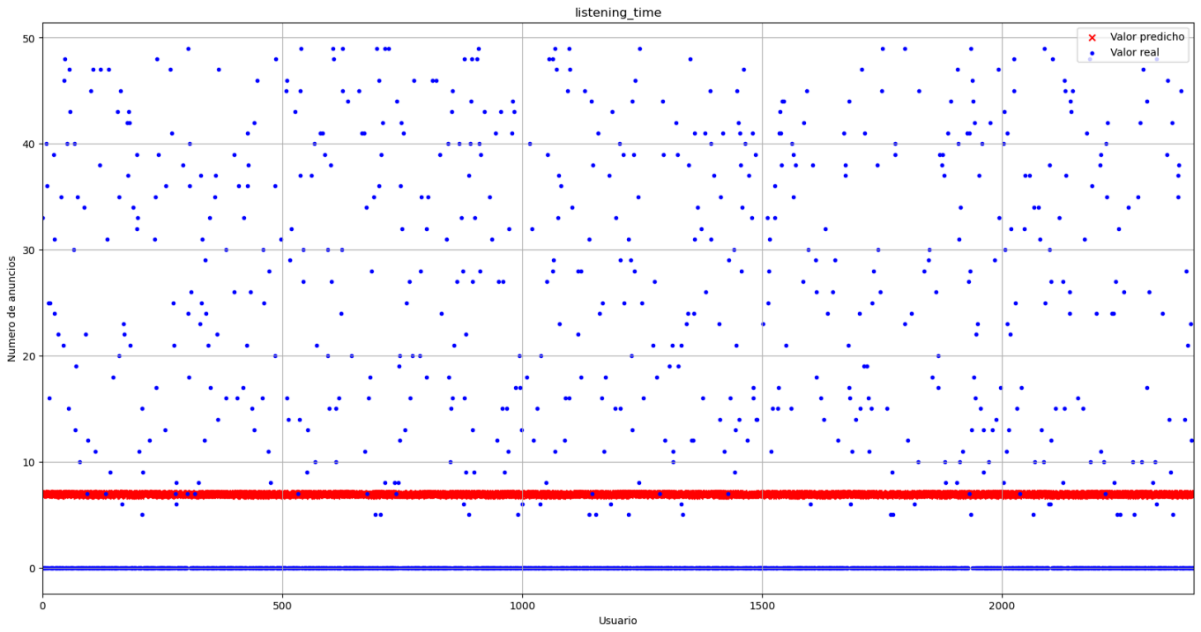
**ii.** ¿Por qué puede ocurrir esto?

Ocurre porque offline_listening es la única columna entre las 4 que tiene una buena relación con la columna ads_listened_per_week, esto se puede observar en la descripción del conjunto, en el mapa de calor que muestra los coeficientes de correlación de Pearson, donde estas dos columnas tienen un coeficiente de -0.878 (relacion excelente) y las otras columnas contra la clase tienen coeficientes menores a 0.1.

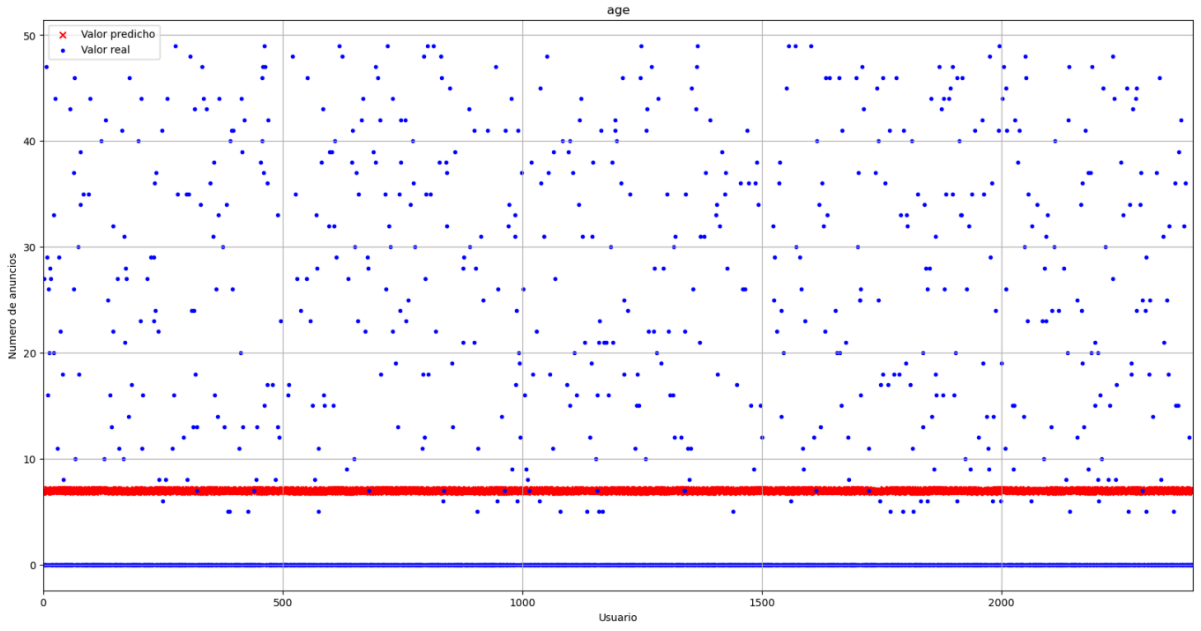
**iii.** ¿Se observan relaciones lineales claras entre las variables?

No, ya que aunque tengan una relación evidente, la unica información que da esa relación es si hay anuncios o no (si offline_listening es 1, entonces los anuncios son 0) pero no indica cuantos anuncios hay en el caso de que los haya (si offline_listening es 0 sabemos que hay anuncios, pero no sabemos cuantos), por lo tanto no es una relación lineal.

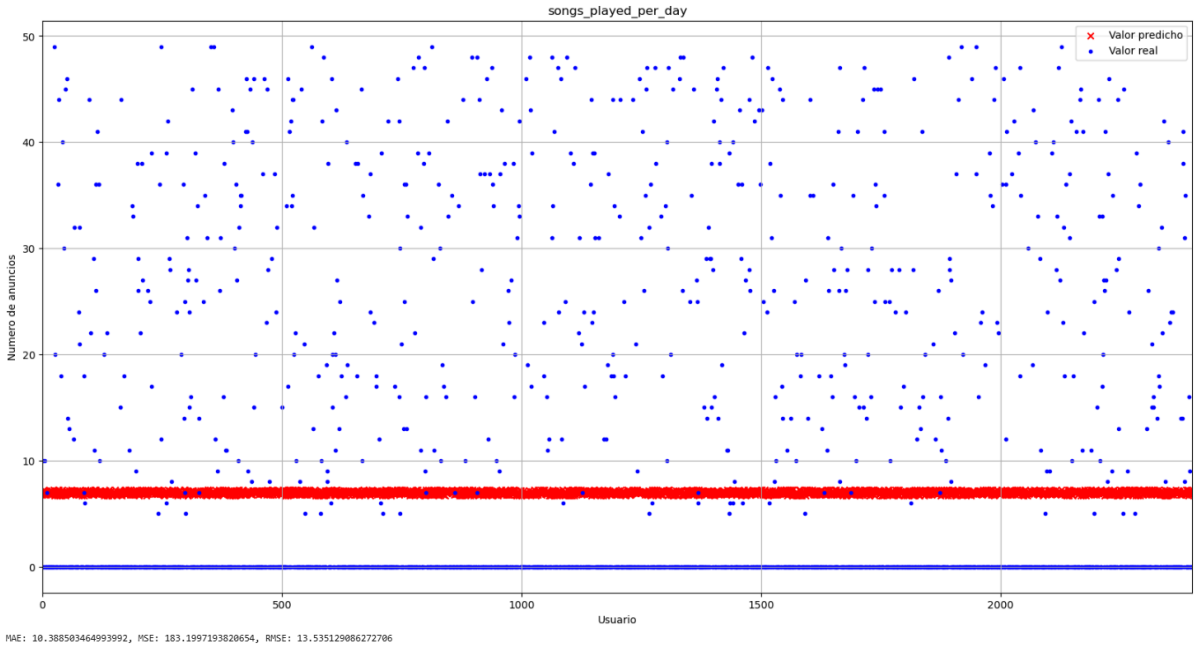
Adicionalmente a las preguntas, cabe recalcar que las gráficas de las columnas listening_time, age y songs_played_per_day son muy parecidas:



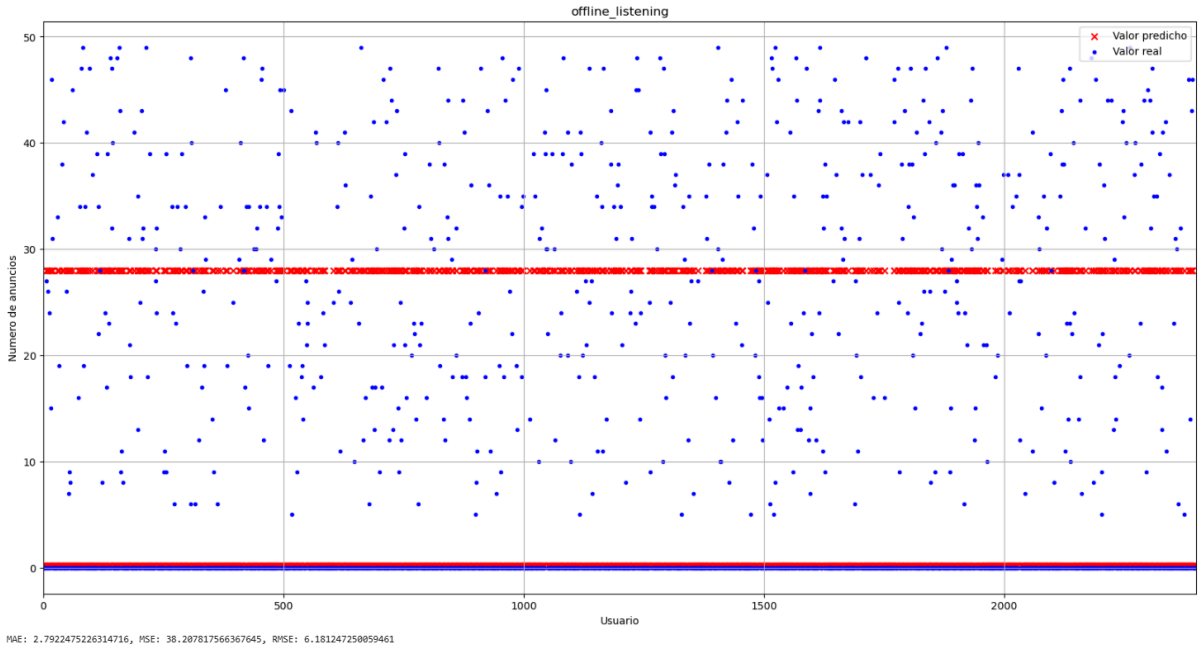

MAE: 10.494024884915175, MSE: 190.45738844754203, RMSE: 13.800630001834772




MAE: 10.423490723743893, MSE: 184.17917211867498, RMSE: 13.571262731178518




Mientras que la de offline_listening es muy distinta por lo anteriormente comentado:




Es capaz de predecir exitosamente los valores que son 0, pero con los demas valores le ocurre como a las demás gráficas y predice a la media

## Apartado 2:

En la implementación se ha tomado la columna “ads_listened_per_week” como la clase y los siguientes subconjuntos como predictores: “listening_time” y “age” para el primer subconjunto, para el segundo “offline_listening” y “songs_played_per_day” y para el tercero todas las columnas (salvo la clase). La elección de estas columnas se explica en la implementación.

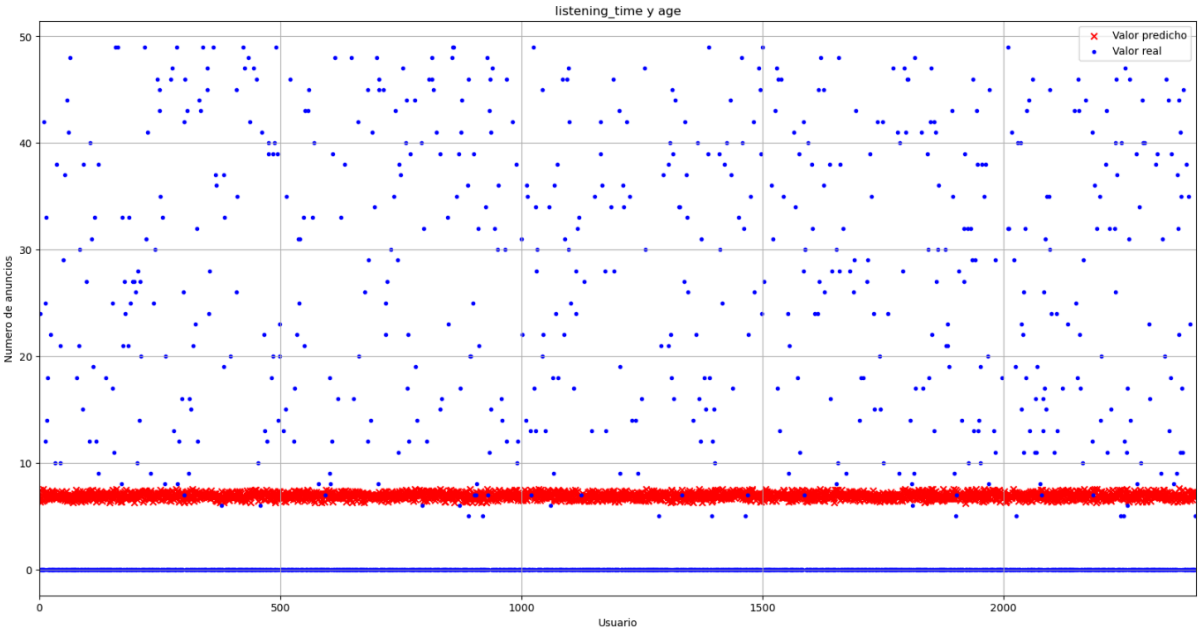
**i.** ¿Con qué subconjunto de variables has obtenido el menor error?

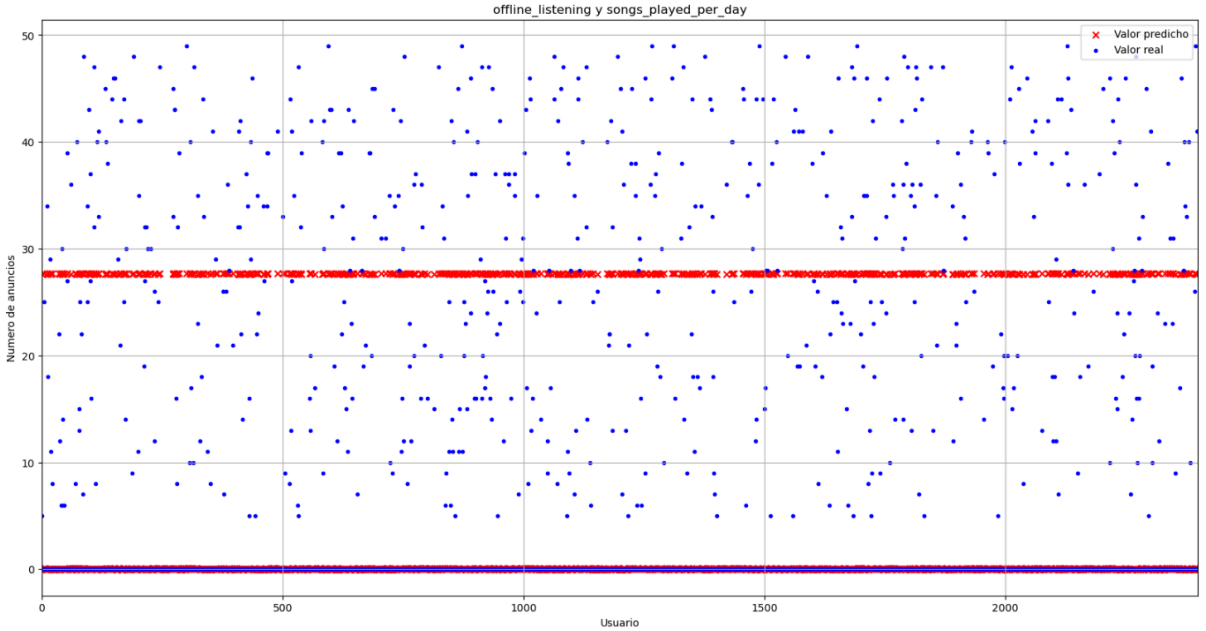
Se obtiene menor MSE con la segunda implementación. Se está teniendo en cuenta el MSE puesto que es más sensible a valores extremos y según lo visto en las gráficas los errores se dan bastante lejos (por el hecho de que predice a la media).

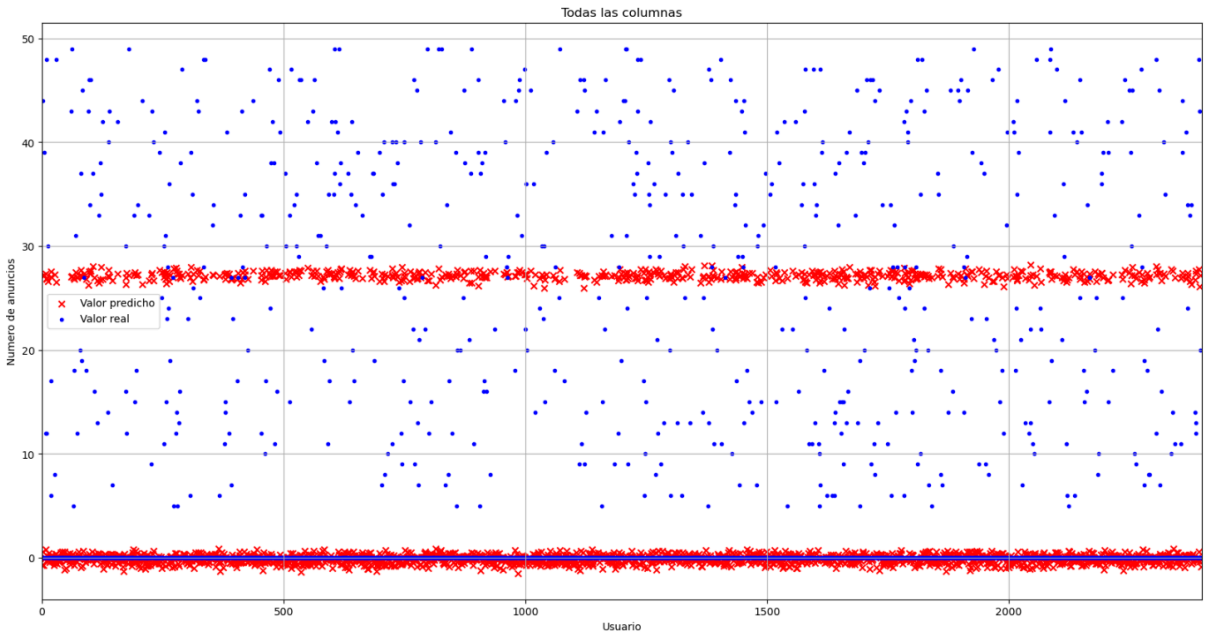
```
Error implementación 1: 193.26439543370702
Error implementación 2: 42.405711651521315
Error implementación 3: 45.52234746106082
El menor error es 42.405711651521315, de la implementacion 2.
```

La implementación con menor error es la segunda, osea, la del subconjunto de las columnas “offline_listening” y “songs_played_per_day”. Esto es debido a:
El primer subconjunto no es capaz de diferenciar las personas que tienen anuncios de las que no las tienen, por eso tiene el error más alto (ademas de que como también es la implementación con errores a mayor distancia de su valor real, su error es mucho mas grande que el del resto de implementaciones, recordamos que el MSE penaliza gravemente los valores altos de error).
El segundo subconjunto si es capaz de diferenciarlos gracias a su columna “offline_listening”, entonces aunque no sea bueno, es mejor y es capaz de ofrecer los datos de las personas cuyo numero de anuncios sea 0. Su otra columna no aporta valor a modelo.
Para el tercer subconjunto, se es capaz de diferenciar las personas con anuncios o sin anuncios, pero la unica columna con valor real es “offline_listening” o incluso los tipos de suscripción, pero el resto de columnas solo aporta ruido al modelo, por lo tanto esto produce que los valores en el 0 sean mas dispersos que en el segundo subconjunto y por tanto con más error, pero no por mucho.

Gráficas en las cuales se pueden observar lo que se acaba de explicar, de el primer subconjunto al tercero en orden:


listening_time y age
Valor predicho
Valor real
Numero de anuncios
Usuario
0
10
20
30
40
50
0
500
1000
1500
2000


offline_listening y songs_played_per_day
Valor predicho
Valor real
Numero de anuncios
Usuario
0
10
20
30
40
50
0
500
1000
1500
2000




El ruido que se observa en el punto Y = 0 es al error al que nos referimos. Comparándolo con la segunda grafica, se puede observar como aumenta la dispersión de los puntos.

**ii.** ¿Qué porcentaje de mejora se obtiene en el error con respecto al error cometido cuando se usan todas las variables?

El error obtenido con el segundo subconjunto (el que menor error tiene) es un 7% menor que el error cometido al usar todas las variables:

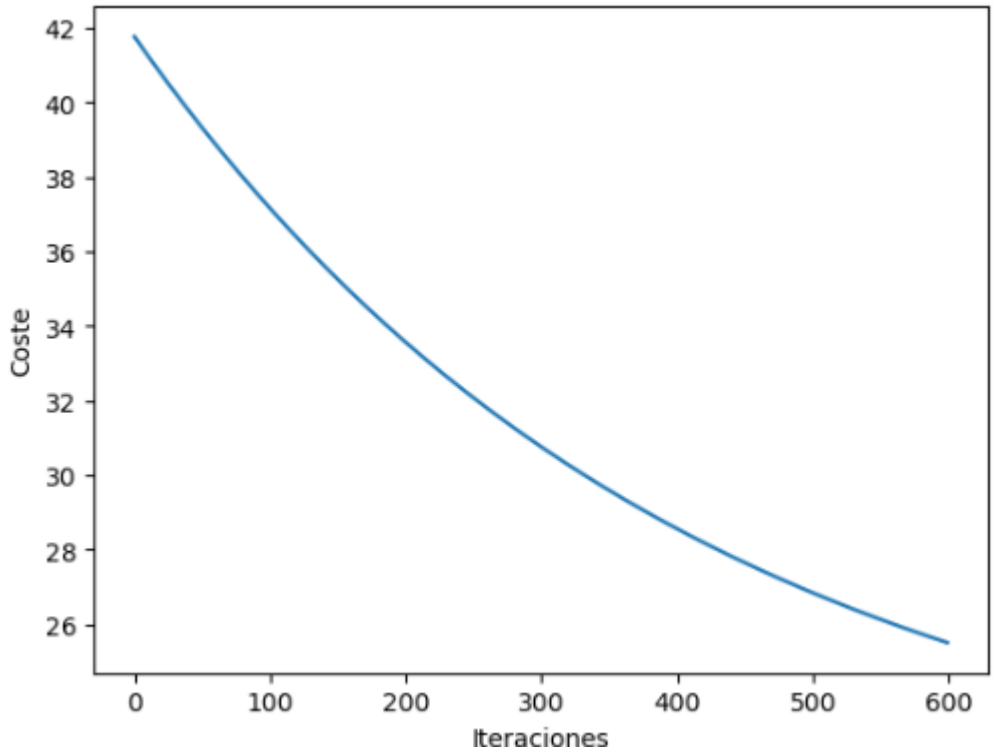
```
print(f"Porcentaje de mejora de la segunda implementación con respecto a la tercera: { ((MSE3 - MSE2)/MSE2)*100 }%")
```

```
Porcentaje de mejora de la segunda implementación con respecto a la tercera: 7.349566103621073%
```

## Apartado 3:

**i.** ¿Qué ocurre cuando la tasa de aprendizaje es demasiado baja?

Cuando la tasa de aprendizaje es demasiado baja se da una gráfica similar a esta (alpha = 0.001):

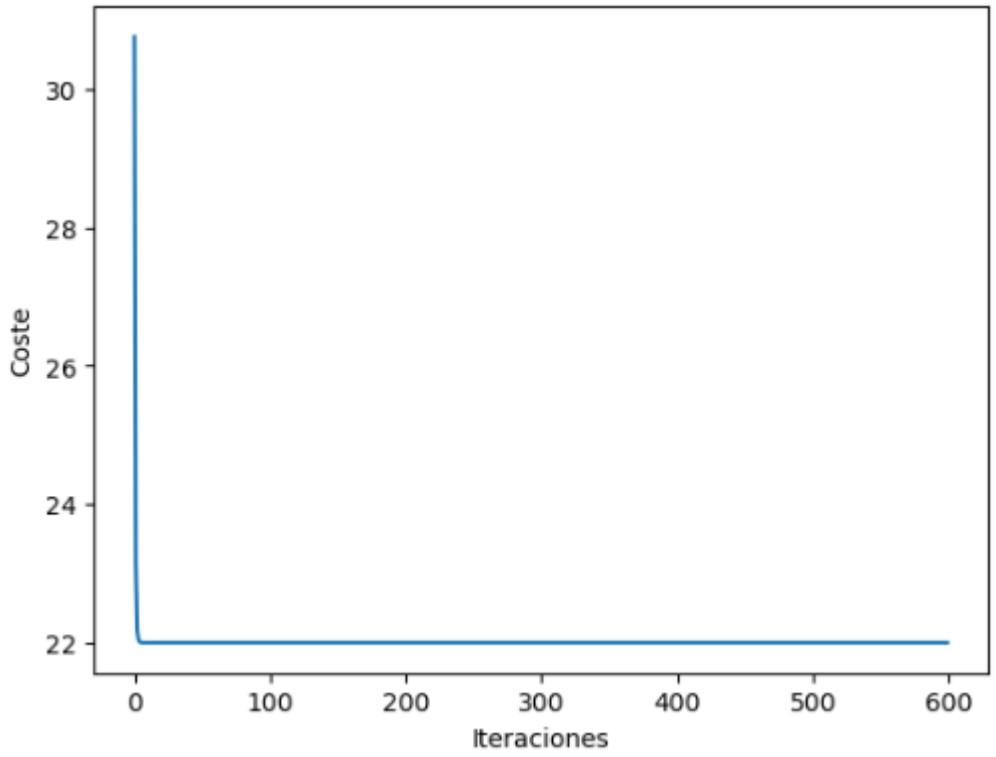



En esta gráfica se observan 2 puntos importantes:

- El coste no converge. Esto se podría solucionar o incrementando el alpha o incrementando el número de iteraciones.
- Al no converger el coste, no se llega al menor coste posible.

**ii.** ¿Y cuándo es demasiado alta?

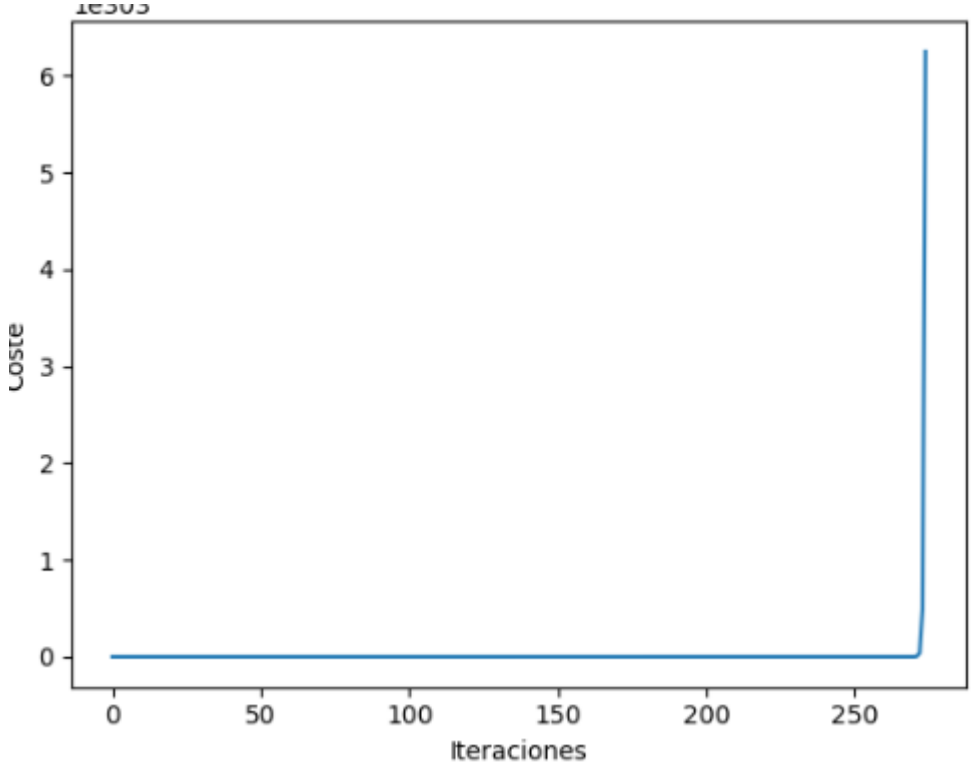
Cuando la tasa de aprendizaje es demasiado alta se da una gráfica similar a esta (alpha = 1):




En esta gráfica se observa un punto interesante:

- El coste converge, pero muy pronto, por lo que el resto de iteraciones son innecesarias.

En el caso de que alpha hubiera sido aun mas grande, el resultado sería el siguiente:



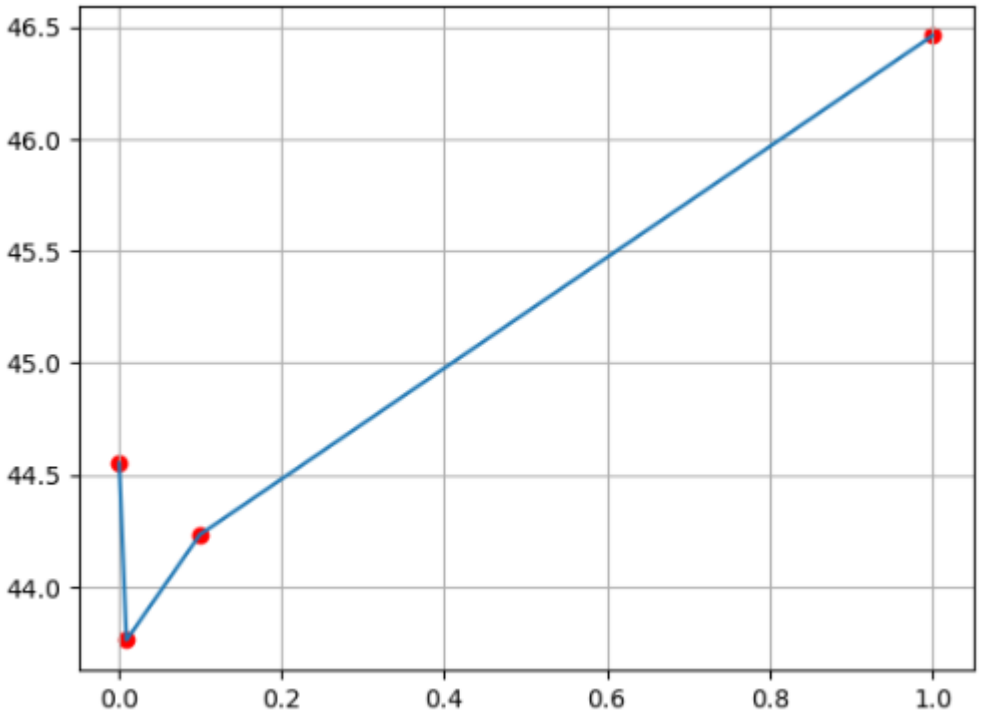

Esto se debe a que, al ser alpha tan grande, pasa del punto del que debería converger.

Esta gráfica se ha dado con alpha = 3.

**iii.** Represente gráficamente la curva de error para cada valor y discuta la gráfica.

En la gráfica se va a comparar el MSE, ya que es más sensible a valores extremos y son esos los que más denotan en nuestro modelo.

La gráfica es la siguiente:




Dado que los primeros puntos estan muy cercanos, recordamos que los puntos son, de izquierda a derecha en el eje X = 0.001, 0.01, 0.1, 1. En el eje Y están marcados sus errores.

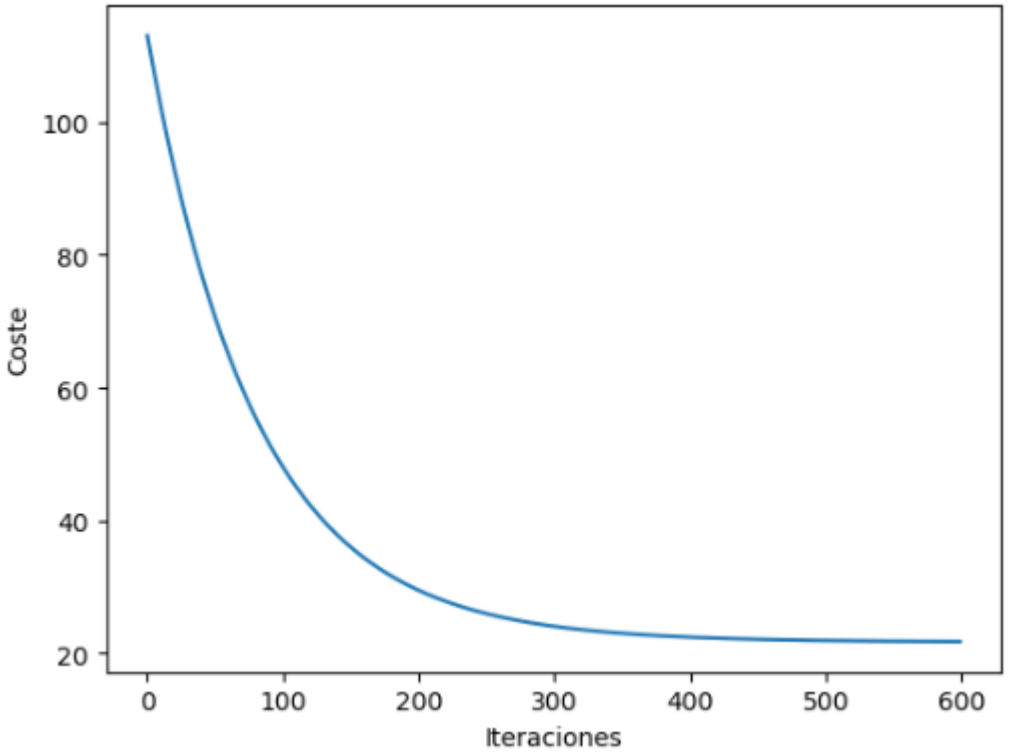
Claramente se puede ver cómo, de entre los 4 valores, el alpha que ofrece menor error es con el valor de 0.01, seguido por 0.1. Esto se debe a que son los dos valores que convergen (para 0.001 no converge) y que menos iteraciones usan (para 1 si converge, pero utiliza muchas iteraciones innecesarias).

## Apartado 4:

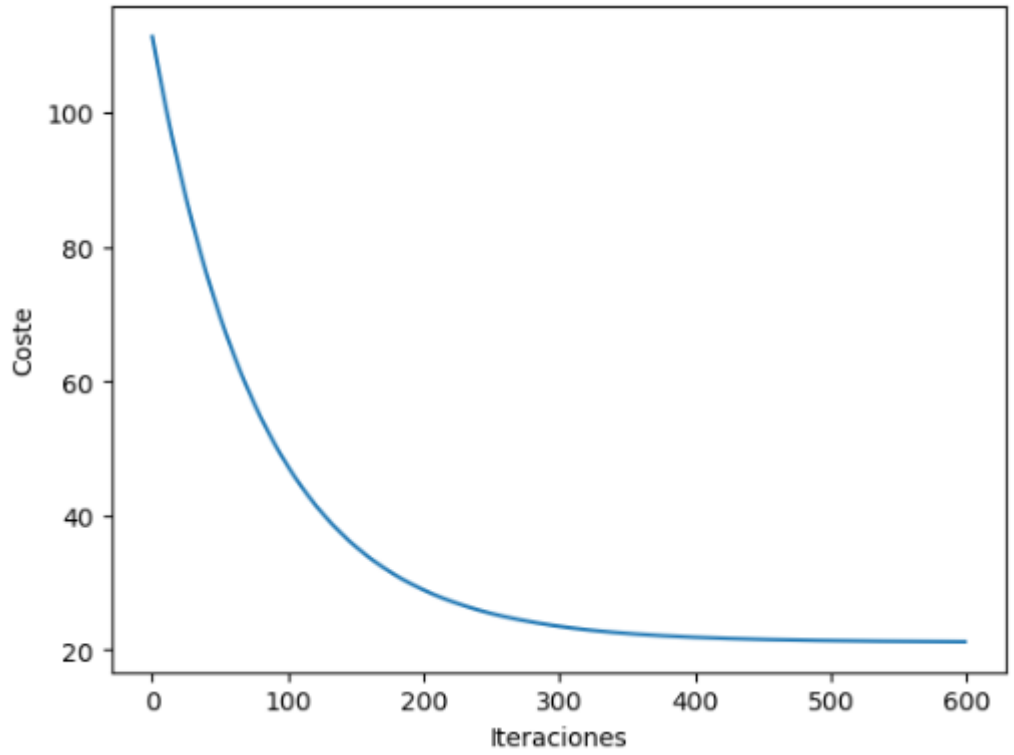
**i.** ¿Influye la inicialización en la convergencia del modelo?

Se comparará la gráfica del coste de la primera iteración de k-folds entre todos los modelos:

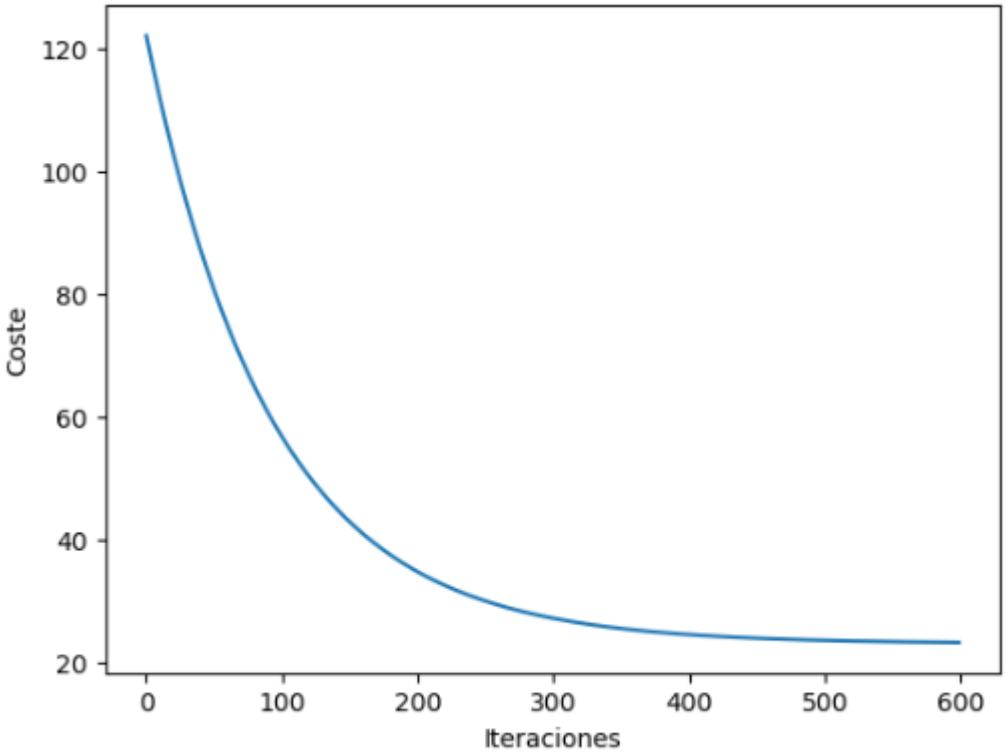
Para el modelo con theta inicializado a 0.



Para el modelo con theta inicializado a valores pequeños.




Para el modelo con theta inicializado a valores grandes.

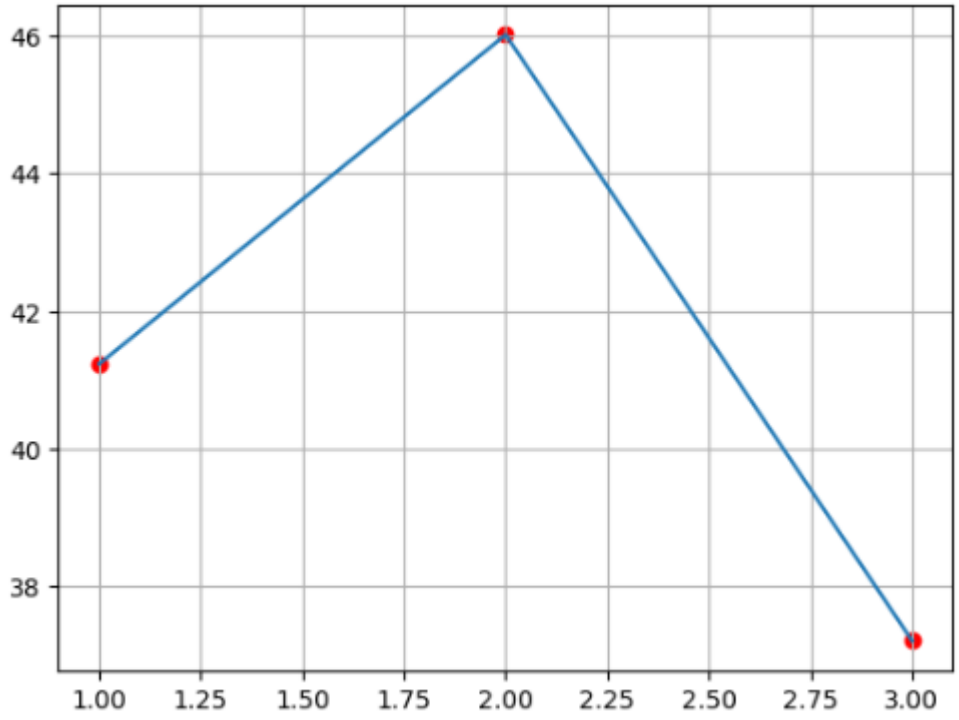


La forma de la gráfica es igual, sin embargo, cambiar los theta inicial cambia sobretodo el punto de partida, por eso eso en el último modelo, como los theta son más grandes y distintos al valor que debieran de tener, empiezan con un coste elevado, mientras que el primer y el segundo modelos, como empiezan relativamente cerca, no se ve un cambio significativo en la primera gráfica.

Por lo tanto si, influye en cuánto de lejos esté del theta óptimo, pues si el theta inicial está más cercano le costará menos iteraciones llegar hasta él, mientras que si es uno lejano necesitará más.

**ii.** ¿Afecta al valor final del error o solo a la velocidad de aprendizaje?

Se han comparado los 3 errores en una gráfica, en la cual el eje X solo simboliza si es de la prueba 1, la 2 o la 3, y el eje Y indica el MSE:




Los errores parecen extraños despues de lo comentado en el apartado **i.**, ya que si alguno debiera de ser mejor, deberían de ser los thetas inicializados a 0 y con valores menores simplemente por estar más cerca del óptimo.

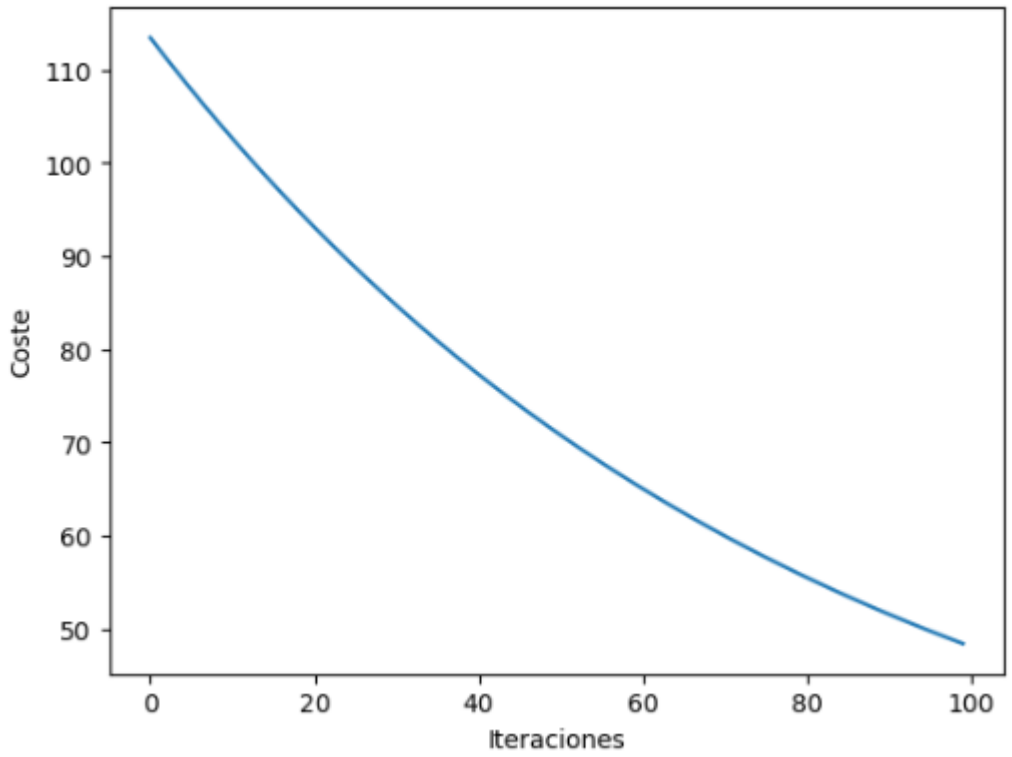
¿Qué es lo que ocurre? Lo que se demuestra en la gráfica es una coincidencia, pues si se observan más valores de MSE para la tercera implementación se pueden ver como sus valores oscilan entre 43 y 48 (incluso llegando a valores inusuales como 50 o 30). Que existan tantos cambios es por lo comentado a lo largo de todo el trabajo, el modelo no intenta predecir, se ajusta a la media con un tanto de dispersión. En algunos casos ajustarse a la media con esa dispersión ofrece buenos resultados (30) y otras veces no tantos (50). Por lo tanto, que en el conjunto de test el tercer modelo de mejores resultados

es porque los puntos reales están más cercanos a su media dispersa. En este caso, comparar estos valores no es representativo.
Verdaderamente, si el modelo predijera realmente, los valores tendrían que haber sido o iguales o ligeramente con menor error para los modelos de thetas inicializados a 0 y con valores menores, porque al estar inicializados con unos parámetros con menor coste que el tercer modelo, es seguro que encontrarán el theta óptimo antes (en esta implementación, al no haber relaciones entre clases no existe ese theta óptimo, porque no se pueden predecir las columnas).

## Apartado 5:

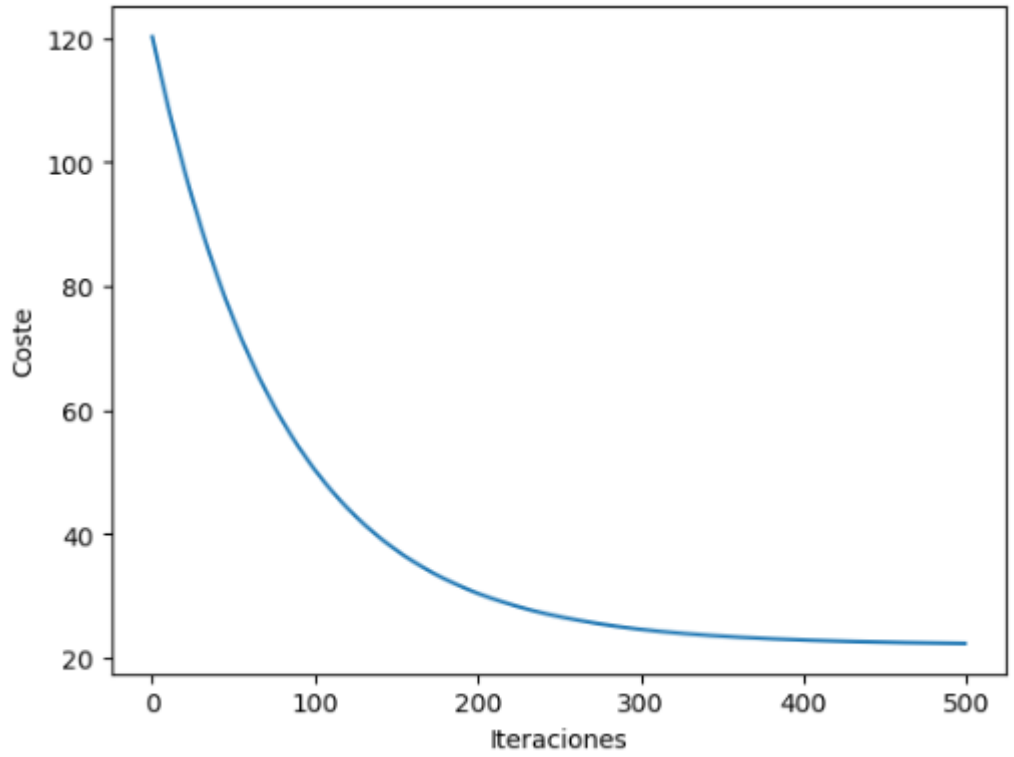
**i.** ¿Cuándo se estabiliza la función de coste?

Para el modelo con 100 iteraciones el coste nunca llega a estabilizarse, teniendo en todas las iteraciones del k-fold esta forma:

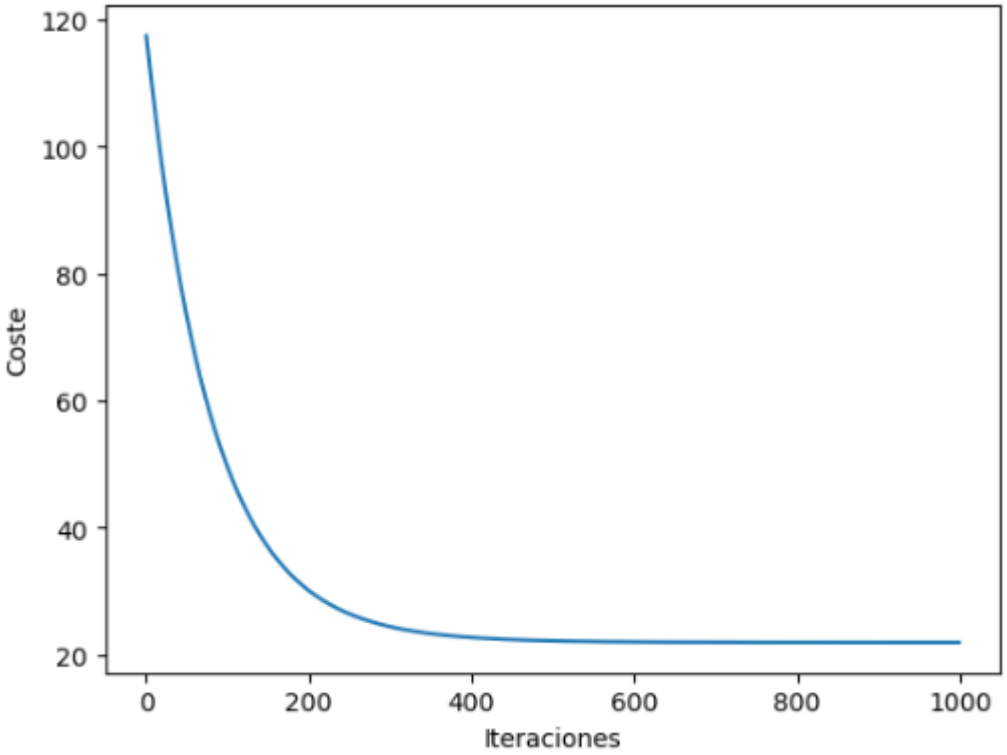



Para los demás modelos si logra converger desde la primera iteración:

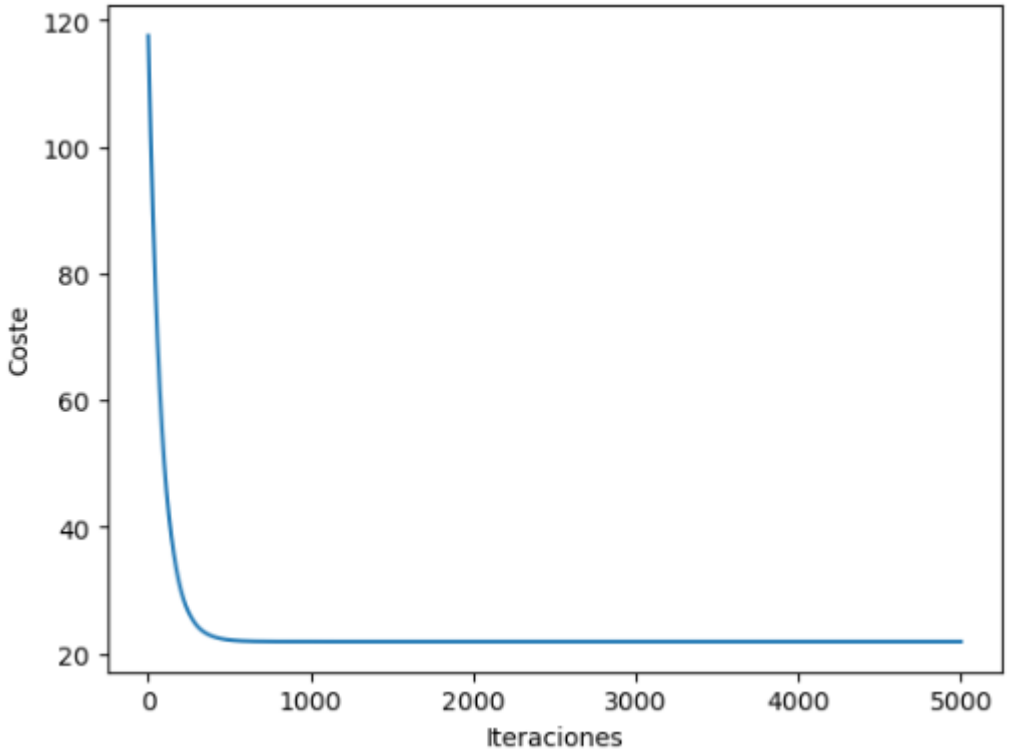
Iteraciones = 500.




Iteraciones = 1000.



Iteraciones = 5000.




**ii.** ¿Qué ocurre si se entrena demasiado poco o demasiado tiempo?

Si se utilizan pocas iteraciones se corre el riesgo de que no logre obtener un buen resultado, ya que puede que no logre converger. Sin embargo, será más rápido.

Si se utilizan muchas iteraciones se estará seguro de que convergerá, pero el tiempo que conlleva aumenta con cada iteración añadida.

**iii.** ¿Se observan síntomas de sobreajuste?

En este caso concreto no, ya que aunque el modelo se sobreentrene en el caso de 5000 iteraciones, el modelo se entrena con k-folds, lo que evita en cierta medida el sobreajuste por estar entrenandose y validandose con conjuntos distintos cada vez. Ademas que el modelo no tiene datos suficientes para adaptarse a situaciones especificas, en vez predice a la media. Por estos dos motivos, el modelo no sufre sobreajuste.