Regresión lineal y Machine Learning*

David Aguirre** Carlos Mundo*** José Rivera****

Resumen

Every day we ask to the computer to learn for themselves. It's a real necessity that the computer could self-program, where the Machine Learning has become in the solution. The importance of this text is given for the explication that we will give about the fundamental that will be machine learning and we are gonna explain the theory that give the bases to apply the machine learning.

Resumen

Todos los días pedimos a la computadora que aprenda por sí misma. Es una necesidad real que la computadora pueda auto-programarse, donde Machine Learning se ha convertido en la solución. La importancia de este texto se da para la explicación que daremos sobre lo fundamental que será Machine Learning y explicaremos la teoría que proporciona las bases para aplicarlo.

1. Introducción

En la actualidad el aprendizaje automático, machine learning, big data, data science... son temas muy comentados por todos. De hecho, la profesión de data scientist ha sido calificada como la de más interes del siglo XXI. Muchos hablan de la revolución de los datos y la inteligencia artificial, pero, ¿De que se trata realmente el machine learning y por qué se habla tanto de ello? Una respuesta a esta pregunta es dada por Drew Conway*, que con el siguiente diagrama de Venn definió machine learning como la unión de habilidades con los ordenadores y los conocimientos de las matemáticas y estadística.

El aprendizaje automático o machine learning consiste en un conjunto de algoritmos que aprenden y resuelven problemas gracias a la experiencia. Existen diversos tipos de problemas que se abordan con multiples técnicas de machine learning, entre ellos se encuentran los problemas de clasificación (donde queremos predecir una clase), los de regresión, las series temporales, etc. Los algoritmos de aprendizaje automático pueden aprender de 4 formas distintas: mediante un aprendizaje supervisado, con aprendizaje no supervisado, con aprendizaje semi-supervisado o con aprendizaje por refuerzo.

El aprendizaje supervisado es el método más sencillo de realizar. En ellos se parte de un conocimiento con

*Profesor reponsable, Clifford Torres, docente de la especialidad de Matemática, ctorresp@uni.edu.pe

**Alumno del 3er ciclo, especialidad de Matemática, david.aguirre.h@uni.pe.

***Alumno del 4to ciclo, especialidad de Matemática, crmundol@uni.pe

****Alumno del 3to ciclo, especialidad de Matemática, jose.rivera.h@uni.pe

*http://drewconway.com/zia/2013/3/26/the-data-science-venn-diagram

antelacion. El objetivo es, mediante unos datos de entrenamiento, obtener una función que haga lo mejor posible el mapeo entre unas entradas y una salida. Así, la regresión lineal, seguramente es el algoritmo más sencillo de aprendizaje supervisado dentro del paradigma del machine learning.

En consecuencia, el análisis de Regresión es un subcampo de "machine learning supervisado". Su propósito es establecer un modelo para la relación entre un cierto número de características y una variable objetivo continua. En los problemas de regresión perseguimos obtener una respuesta cuantitativa, como por ejemplo, predicciones sobre precios de inmuebles o el número de segundos que alguien dedicará a visualizar un vídeo, entre otros.

El modelo clásico de regresión lineal nos permite establecer un modelo para ajustar la relación de dependencia entre otras características específicas independientes (valor de las variables independientes X) y el valor resultado correspondiente (un valor de la variable dependiente y). Esta relación se realiza estableciendo una línea arbitraria y calculando la distancia de la recta a los puntos de datos correspondientes a los valores y y x. Esta distancia, las líneas verticales, son los "residuos.º los "errores de predicción". El algoritmo de regresión recalculará (y moverá) la recta con cada interacción, buscando aquella que mejor se ajuste a los puntos de datos,o sea, la línea con el menor error (la más cercana al máximo número de puntos).

Usando el paradigma de machine learning supervisado y manipulando uno de sus métodos básico, regresión lineal, se buscará que al resolver un problema; el algoritmo a plantear pueda aprenden por sí mismos y en este caso, pueda obtener automáticamente esa «recta» que buscamos con la tendencia de predicción, es decir, nuestro interés es predecir soluciones a nuestro problema.

El siguiente proyecto plantea resolver un problema en el cual se desea saber cuantas veces sera compartido un artículo que se refiere a machine learning en la web, para esto nosotros planteamos definir que es una regresión lineal, desde el punto de vista multivariado y definir las cosas necesarias de machine learnig para hacer posible tal objetivo, para luego usando un conjunto de datos podamos implementar el algoritmo de machine learning, en python, y poder realizar la predicción que deseamos saber. Ante esto haremos uso de nuestras herramientas conocidas hasta el momento, en la universidad, Álgebra lineal, Cálculo de probabilidades, Estadística inferencial, Cálculo diferencial e integral avanzado, Análisis real, programación en python y el editor de textos Latex para poder presentar los avances y el informe final del proyecto.

2. Regresión lineal

2.1. El Modelo Lineal

La forma genérica del modelo de regresión lineal es:

$$\begin{aligned} y_i &= f(x_{i1}, x_{i2}, \cdots, x_{iK}) + \epsilon_i \\ &= \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots \\ &+ \beta_K x_{iK} + \epsilon_i,\ i = 1, \cdots, n. \end{aligned} \tag{1}$$

donde y es la variable **dependiente** o explicada, $x_{i1}, x_{i2}, \cdots, x_{iK}$ son las variables **independientes** o explicativas y el subíndice i indica las n observaciones muéstrales. Una teoría especificará la función $f(x_{i1}, x_{i2}, \cdots, x_{iK})$. esto es lo que comúnmente se conoce como **Ecuación de regresión poblacional** de y sobre $x_{i1}, x_{i2}, \cdots, x_{ik}$. En este contexto, y es el **regresando** y x_k, $k = 1, \cdots, K$, son los **regresores**. Al término ϵ se le llama perturbación aleatoria,

por que "perturba"la que de otra manera, sería una relación estable determinada.

2.2. Supuestos del modelo clásico de regresión lineal

Los supuestos del modelo hacen referencia a las siguientes cuestiones:

1. **Linealidad del modelo de regresión**: Sea el vector columna $\mathbf{x}_k$ que contiene las n observaciones de la variable x_k, $k = 1, \cdots, K$ añadamos este vector columna a la matriz $\mathbf{X}$ de tamaño $n \times K$. la primera columna de $\mathbf{X}$ será una columna de unos, por lo que β_1 será el término constante del modelo. Llamaremos y a las n observaciones, $y_1, y_2, \cdots, y_n$ y ϵ al vector columna que contiene las n perturbaciones. Ahora el modelo (1) puede escribirse como

$$\mathbf{y} = \mathbf{x}_1\beta_1 + \cdots + \mathbf{x}_K\beta_K + \epsilon \tag{2}$$

o, en forma matricial:

$$\begin{aligned}\mathbf{y}_{n\times 1} &= \begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_K\end{bmatrix}\begin{bmatrix}\beta_1 \\ \vdots \\ \beta_K\end{bmatrix} + \begin{bmatrix}\epsilon_1 \\ \vdots \\ \epsilon_K\end{bmatrix} \\ &= \mathbf{X}_{n\times K}\beta_{K\times 1} + \epsilon_{n\times 1}. \quad (\textbf{Supuesto 1})\end{aligned} \tag{3}$$

Así cada observación se puede expresar como $y_i = \mathbf{x}_{i1}\beta_1 + \cdots + \mathbf{x}_{iK}\beta_K + \epsilon_i$ donde $\begin{bmatrix}\mathbf{x}_{i1} & \cdots & \mathbf{x}_{iK}\end{bmatrix}$ es la i-ésima fila de la matriz $\mathbf{X}$, por lo tanto

$$y_i = \begin{bmatrix}\mathbf{x}_{i1} & \cdots & \mathbf{x}_{iK}\end{bmatrix}^t \beta \tag{4}$$

La parte de nuestro interés principal es la estimación e inferencia de los parámetros del vector β.

El supuesto de linealidad del modelo de regresión incluye una perturbación en forma aditiva. Para que la regresión sea estrictamente lineal, debe ser de la forma (1) ya sea en sus variables originales o después de alguna transformación adecuada.

2. **Rango completo**: Un supuesto muy importante debe hacerse sobre los regresores. Consideramos que no existe una relación lineal exacta entre las variables.

$$\mathbf{X} \text{ es una matriz } n\times K \text{ con rango } K. \quad (\textbf{Supuesto 2}) \tag{5}$$

Significa que se tiene el rango de columnas completo, las columnas de $\mathbf{X}$ son linealmente independientes, y hay al menos K observaciones. Este supuesto se conoce como **condición de identificación**.

3. **Regresión**: Consideremos que el valor esperado de la perturbación aleatoria debe ser cero para cualquier observación, lo cual podemos escribir como

$$E[\epsilon_i|\mathbf{X}] = 0. \tag{6}$$

Para todo el conjunto de observaciones, podemos escribir los siguiente:

$$E[\epsilon|\mathbf{X}] = \begin{bmatrix}E[\epsilon_1|\mathbf{X}] \\ E[\epsilon_2|\mathbf{X}] \\ \vdots \\ E[\epsilon_n|\mathbf{X}]\end{bmatrix} = 0_{n\times 1}. \quad (\textbf{Supuesto 3}) \tag{7}$$

La parte izquierda nos dice, que la medida de cada ϵ_i condicionada con todas las observaciones $\mathbf{x}_i$ es cero. Entonces las observaciones en $\mathbf{x}$ no conllevan información sobre el valor esperado de la perturbación. El supuesto en este punto es que no hay información sobre $E[\epsilon_1|\cdot]$

contenida en cualquier observación de $\mathbf{x}_j$. También supondremos que las perturbaciones no contienen información sobre las otras , es decir, $E[\epsilon_i|\epsilon_1, \cdots, \epsilon_{i-1}, \epsilon_{i+1}, \cdots, \epsilon_n] = 0$. Es decir, hemos considerado que las perturbaciones siguen un camino aleatorio.

Observación 1. *a) Que la media condicionada sea cero implica que la media no condicionada también sea cero y así*

$$E[\epsilon_i] = E[E[\epsilon_i|\mathbf{x}_i]] = E[0] = 0.$$

ya que, $Cov[\mathbf{x}, \epsilon] = Cov[\mathbf{x}, E[\epsilon|\mathbf{X}]$*, el Supuesto 3 implica que* $Cov[\mathbf{X}, \epsilon] = 0$*.*

b) En muchos casos el supuesto de media cero, $E[\epsilon_i] = 0$*, no es restrictivo, es decir se puede dar el caso de que* $E[\epsilon_i] = \mu_i \neq 0$

c) La idea básica de una caminata aleatoria es que el valor de mañana de un conjunto de datos es el valor de hoy más un cambio impredecible, en nuestro caso

$$\epsilon_i = \epsilon_{i-1} + \gamma_i, \text{ donde } \gamma_i \text{ es i.i.d. con } N(0,1),$$

es decir, todos los ϵ_i *tienen la misma distribución y a su vez son independientes entre si.*

El Supuesto 3 también implica que

$$E[\mathbf{y}|\mathbf{X}] = \mathbf{X}\beta \tag{8}$$

Es así que los Supuestos 1 y 3 abarcan el *modelo de regresión lineal*. La **regresión** de $\mathbf{y}$ sobre $\mathbf{X}$ es la esperanza condicionada, $E[\mathbf{y}|\mathbf{X}]$, por lo que sin el Supuesto 3, $\mathbf{X}\beta$ no es la función media condicionada.

Observación 2. *a) El resto de supuestos especificarán las características de las perturbaiones en el modelo las condiciones con las que se obtienen las observaciones muestrales* $\mathbf{X}$*.*

4. **Perturbaciones esferícas**: El cuarto supuesto será sobre las varianzas y covarianzas de las perturbaciones

$$Var[\epsilon_j|\mathbf{X}] = \sigma^2, \text{ para } i = 1, \cdots, n \quad (*)$$

y

$$Cov[\epsilon_i, \epsilon_j|\mathbf{X}] = 0, \text{ para } i \neq j. \quad (**)$$

La varianza constante es conocida como **homocedasticidad** (comportamiento homogéneo), en el caso que ésta no sea constante, lo cual también puede manifestarse será conocida como **heterocedasticidad** (comportamiento heterogéneo). Vease Figura (1) La incorrelación entre observaciones es conocida como **no autocorrelación**. Es así que este Supuesto consiste en que las desviaciones de las observaciones de su valor esperado están incorrelacionados. Entonces $(*)$ y $(**)$ implican que

$$E[\epsilon\epsilon^t|\mathbf{X}] = \begin{bmatrix} E[\epsilon_1\epsilon_1|\mathbf{X}] & E[\epsilon_1\epsilon_2|\mathbf{X}] & \cdots & E[\epsilon_1\epsilon_n|\mathbf{X}] \\ E[\epsilon_2\epsilon_1|\mathbf{X}] & E[\epsilon_2\epsilon_2|\mathbf{X}] & \cdots & E[\epsilon_2\epsilon_n|\mathbf{X}] \\ \vdots & \vdots & \ddots & \vdots \\ E[\epsilon_n\epsilon_1|\mathbf{X}] & E[\epsilon_n\epsilon_2|\mathbf{X}] & \cdots & E[\epsilon_n\epsilon_n|\mathbf{X}] \end{bmatrix}$$

$$= \begin{bmatrix} \sigma^2 & 0 & \cdots & 0 \\ 0 & \sigma^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma^2 \end{bmatrix}$$

que podemos resumir en el siguiente supuesto

$$E[\epsilon\epsilon^t|\mathbf{X}] = \sigma^2 I(n). \quad (\textbf{Supuesto 4}) \tag{9}$$

Observación 3. *a) Una vez más, hay que resaltar que este supuesto describe la información sobre las varianzas y covarianzas entre perturbaciones que es proporcionada por las variables independientes.*

b) Las perturbacines que satisfacen los supuestos de homocedasticidad y de no autocorrelación son conocidas como perturbaciones **esféricas**.

5. **Regresores no estocásticos**: Es usual suponer que $\mathbf{x}_i$ es no **estocástico**[**], como ocurriría en una situación experimental. En este caso, el supuesto de regresores no estocásticos es ampliamente conveniente. con él podremos utilizar los resultados estadísticos para obtener nuestros resultados ; el vector $\mathbf{x}_i$ es simplemente una constante conocida en la función de probabilidad de y_i. Con esta simplificación, los Supuestos 3 y 4 pueden convertirse en incondicionales.

   Un punto de vista alternativo, el cual seguiremos, consiste en que las observaciones de $\mathbf{x}_i$ son "fijas en muestras repetidas", lo que equivale a realizar el análisis estadístico condicional en la muestra que hemos observado. De este modo, sólo supondríamos que el modelo de regresión y sus supuestos se aplican al conjunto particular de las $\mathbf{X}$ que hemos observado. Así, cualquier tratamiento nos permite hacer caso omiso de el origen fundamental de la variación de $\mathbf{X}$ y concentrarnos en la relación entre $\mathbf{y}$ y $\mathbf{X}$

$$\begin{gathered}\mathbf{X}\text{es una matriz conocida} \\ n \times K \text{ de constantes.} \qquad (10) \\ \textbf{(Supuesto 5)}\end{gathered}$$

   Observación 4. *a) Los científicos sociales raramente son capaces de analizar datos experimentales y relativamente pocos de sus modelos están construidos de regresores no estocásticos.*

   b) Dada la observación previa, se puede relajar el supuesto de regresores fijos no estocásticos sin ningún costo alguno. Por lo tanto, el supuesto importante es el Supuesto 3, la no correlación de $\mathbf{X}$ *y* ϵ.

6. **Normalidad**: Nos conviene suponer que las perturbaciones están normalmente distribuidas , con media cero y varianza constante. Es decir, añadimos la normalidad de la distribción a los Supuestos 3 y 4.

$$\epsilon|\mathbf{X} \sim N(0, \sigma^2 I(n)). \quad (\textbf{Supuesto 6}) \qquad (11)$$

   En vista de nuestra descripción de las fuentes ϵ, las condiciones del teorema del límite central pueden generalmente aplicarse y el supuesto de normalidad puede considerarse. Una aplicación útil del Supuesto 6 es que implica que las observaciones de ϵ_i son estadísticamente independientes así como no correlacionadas.

   Observación 5. *a) La normalidad no es necesaria para obtener muchos de los resultados que utilizamos en el aálisis de la regresión múltiple, aunque nos permite obtener algunos resultados estadísticos exactos, es decir, permite las construcción de contrastes estadísticos.*

   b) También es posible relajar este Supuesto y mantener la mayoría de lo resultados estadísticos que normalmente se obtienen.

[**] Una cantidad cuyo valor se determina como resultado de un experimento se denomina variable estocástica. En un determinado experimento, una variable estocástica $\mathbf{x}_i$ puede tomar diferentes valores x_{ji}; debe tenerse cuidado de distinguir la variable $\mathbf{x}_i$ de los distintos resultados $\{x_{ji}\}$ posibles.

3. Regresión por mínimos cuadrados

Los parámetros desconocidos de la relación estocástica $y_i = x_i^{'}\beta + \epsilon_i$ son el objetivo de la estimación. La regresión poblacional es $E[y_i|x_i] = x_i\beta$, mientras que la estimación de $E[y_i|x_i]$ la llamaremos:

$$\widehat{y_i} = x_i^{'}b.$$

La perturbación aleatoria asociada al punto i-ésimo es:

$$\epsilon_i = y_i - x_i^{'}\beta.$$

Para cada valor de **b**, estimamos ϵ_i a partir de los residuos

$$e_i = y_i - x_i^{'}b.$$

A partir de las definiciones,

$$y_i = x_i^{'}\beta + \epsilon_i = x_i^{'}b + e_i$$

El valor poblacional β es un vector de parámetros desconocidos de la distribución de probabilidad y_i cuyos valores esperamos estimar a partir de los datos de nuestra muestra. Consideramos el problema algebraico puro de elección del vector **b** de modo que la linea ajustada x_ib se acerque a los puntos de los datos. La medida de este acercamiento constituye el **criterio de ajuste**. Aunque se han sugerido numerosas opciones. la utilizada más frecuentemente es la de los **mínimos cuadrados**[***].

3.1. El vector de coeficientes de mínimos cuadrados

El vector de coeficientes de mínimos cuadrados minimiza la suma de los cuadrados de los residuos:

$$\sum_{i=1}^{n} \epsilon_{i_0}^2 = \sum_{i=1}^{n} (y_i - \beta_0^{'} x_i) \tag{12}$$

Donde β_0 es la elección arbitraria del vector de coeficientes. El problema de la minimización consiste en elegir un β_0 tal que:

$$minimizar_{\beta_0}\ S(\beta_0) = \epsilon_0^{'}\epsilon_0 = (y - X\beta_0)^{'}(y - X\beta_0) \tag{13}$$

Operando obtenemos:

$$\epsilon_0^{'}\epsilon_0 = y^{'}y - 2y^{'}X\beta_0 + \beta_0^{'}X^{'}X\beta_0 \tag{14}$$

La condición necesaria de mínimo es:

$$\frac{\partial S(\beta_0)}{\partial \beta_0} = -2X^{'}y + 2X^{'}X\beta_0 = 0 \tag{15}$$

Supongamos que **b** sea la solución. Entonces **b** satisface las **ecuaciones normales de mínimos cuadrados,**

$$X^{'}Xb = X^{'}y$$

Si la inversa de $X^{'}X$ existe, que se sigue del supuesto de rango completo, la solución es:

$$b = (X^{'}X)^{-1}X^{'}y \tag{16}$$

debe ser una matriz definida positiva. Supongamos que $q = e^{'}X^{'}Xe$ para un vector arbitrario e distinto de cero. Entonces,

$$q = v^{'}v = \textstyle\sum_{i=1}^{n} v_i^2, \text{ donde } v = Xe$$

A menos que cada elemento de v sea cero, q es positivo. Pero si v pudiera ser cero, v seria una combinación lineal de columnas X iguales a 0. Esto contradice el supuesto de que X tiene rango completo. Por lo tanto, si X tiene rango completo, la solución b de mínimos cuadrados minimiza la suma de los cuadrados de los residuos.

[***]Tendremos que demostrar que el enfoque de ajustar la línea tanto como sea posible por mínimos cuadrados, conduce a estimadores con buenas propiedades estadísticas. Esto tiene sentido intuitivamente y es, de hecho, el caso.

4. Bondad del ajuste y análisis de la varianza

La variación de la variable dependiente se define en términos de desviaciones respecto de su media $(y_i - \bar{y})$. La variación total de y es la suma de las desviaciones al cuadrado:

$$\text{SST} = \sum_{i=1}^{n} (y_i - \bar{y})^2$$

Se utilizan los cuadrados ya que la suma de las desviaciones es siempre cero. En términos de la ecuación de regresión, podríamos escribir todo el conjunto de observaciones como

$$y = Xb + e = \hat{y} + e$$

Para una observación individual, tenemos:

$$y_i = \hat{y}_i + e_i$$
$$a_i = x_i' b + e_i$$

Si la regresión tiene un termino constante, los residuos sumaran cero y la media de los valores predichos de y_i se igualara a la media de los valores actuales. Restando $\bar{y}$ a ambos lados de la expresión y utilizando este resultado $\bar{y} = xb$ se obtiene

$$y_i - \bar{y} = \hat{y}_i - \bar{y} + e_i = (x_i - \bar{x})'b + e_i$$

Puesto que ambos términos en esta descomposición suman cero, para cuantificar este ajuste, utilizamos la suma de los cuadrados de los residuos. Para todo el conjunto de observaciones, tenemos

$$M^o y = M^o X b + M^o e$$

donde M^o es una matriz idempotente $n \times n$ que transforma las observaciones en desviaciones de las medias muéstrales. La columna $M^o X$ correspondiente al término constante es cero y, ya que los residuos también tienen media cero, $M^o e = e$. Entonces, ya que $e' M^o X = e' X = 0$, la suma total de los cuadrados es

$$y' M^o y = b' X' M^o X b + e' e$$

Esto se puede escribir como la suma total de cuadrados que seria igual a la suma de cuadrados de la regresión mas la suma del cuadrado de los errores, es decir,

$$\text{SST} = \text{SSR} + \text{SSE} \tag{17}$$

Ahora podemos obtener una media de la bondad del ajuste mediante la utilización del coeficiente de determinación:

$$\frac{SSR}{SST} = \frac{b' X' M^o X b}{y' M^o y} = 1 - \frac{e' e}{y' M^o y} \tag{18}$$

El coeficiente de determinación se representa por R^2. Debe estar comprendido entre cero y uno, y mide la proporción del total de la variación de y que es representada por la variación de los regresores. Toma el valor cero si la regresión es una linea horizontal, es decir, todos los elementos de b excepto el término constante son cero. En este caso, el valor predicho de y es siempre $\bar{y}$, ya que desviaciones de x respecto de su media no se traducen en diferentes predicciones para y. Como tal, x no tiene poder explicativo. El otro extremo, $R^2 = 1$, que ocurre si los valores de x y de y están en el mismo hiperplano (en una línea recta para una regresión de dos variables) por lo que los residuos son cero. Si todos los valores de y_i están en una línea vertical, R^2 no tiene sentido y no puede calcularse.

Una manera equivalente de calcular R^2 es:

$$b' X' M^o X b = \hat{y}' M^o \hat{y}$$

pero $\hat{y} = Xb, y = \hat{y} + e, M^o e = e$ y $X' e = 0$, ya que $\hat{y}' M^o \hat{y} = \hat{y}' M^o y$. Multiplicamos $R^2 = \hat{y}' M^o \hat{y} / y' M^o y$

$= \hat{y}'M^o y/y'M^o y$ por $1 = \hat{y}'M^o y/\hat{y}'M^o\hat{y}$, se obtiene

$$R^2 = \frac{[\sum_i (y_i - \bar{y})(\hat{y}_i - \hat{\bar{y}})]^2}{[\sum_i (y_i - \bar{y})^2][\sum_i (\hat{y}_i - \hat{\bar{y}})^2]} \qquad (19)$$

que es la correlación al cuadrado entre los valores observados de y y las predicciones calculadas por la ecuación de regresión estimada.

5. Machine Learning

El Machine Learning -traducido al Español como «Aprendizaje Automático»- es un subcampo de la Inteligencia Artificial que busca resolver el «cómo construir programas de computadora que mejoran automáticamente adquiriendo experiencia».

Esta definición indica que el programa que se crea con ML no necesita que el programador indique explícitamente las reglas que debe seguir para lograr su tarea si no que este mejora automáticamente.

Grandes volúmenes de datos están surgiendo de diversas fuentes en los últimos años y el Aprendizaje Automático relacionado al campo estadístico consiste en extraer y reconocer patrones y tendencias para comprender qué nos dicen los datos. Para ello, se vale de algoritmos que pueden procesar Gygas y/o Terabytes y obtener información útil.

5.1. Como aproximarse a las soluciones de estos problemas

Para poder resolver este tipo de problemas surgen soluciones de tipo heurísticas que intentan dar «intuición» al camino correcto a tomar para resolver un problema. Estos pueden obtener buenos resultados en tiempos menores de procesamiento, pero muchas veces su intuición es arbitraria y pueden llegar a fallar. Los algoritmos de ML intentan utilizar menos recursos para «entrenar» grandes volúmenes de datos e ir aprendiendo por sí mismos. Podemos subdividir el ML en 2 grandes categorías: Aprendizaje Supervisado o Aprendizaje No Supervisado.

5.2. Aprendizaje Supervisado

En el Aprendizaje Supervisado los datos para el entrenamiento incluyen la solución deseada, llamada «etiquetas» (labels). Un claro ejemplo es al clasificar correo entrante entre Spam o no. Entre las diversas características que queremos entrenar deberemos incluir si es correo basura o no con un 1 o un 0. Otro ejemplo son al predecir valores numéricos por ejemplo precio de vivienda a partir de sus características (metros cuadrados, nº de habitaciones, incluye calefacción, distancia del centro, etc.) y deberemos incluir el precio que averiguamos en nuestro set de datos.

Los algoritmos más utilizados en Aprendizaje Supervisado son:

k-Nearest Neighbors
Linear Regression
Logistic Regression
Support Vector Machines
Bayesian Classifiers
Decision Tress and Random Forest
Neural Networks
Deep Learning

5.3. Predicción

Similar a la clasificación pero para valores continuos, nos permite intentar predecir qué valor obtendremos dado un conjunto de datos de entrada con resultado desconocido. Como comentábamos antes, se puede utilizar para predecir precios de inmuebles, alquileres, coches o la probabilidad de que ocurra algún evento utilizado con frecuencia en estadística

con Regresión Lineal. Pero si lo que queremos hacer es Pronóstico de Series temporales, será mejor utilizar redes neuronales ú otros modelos estadísticos como ARIMA.

5.4. ¿Cómo funciona el algoritmo de regresión lineal en Machine Learning?

Recordemos que los algoritmos de Machine Learning Supervisados, aprenden por sí mismos y -en este caso- a obtener automáticamente esa «recta» que buscamos con la tendencia de predicción. Para hacerlo se mide el error con respecto a los puntos de entrada y el valor «Y» de salida real. El algoritmo deberá minimizar el coste de una función de error cuadrático y esos coeficientes corresponderán con la recta óptima. Hay diversos métodos para conseguir minimizar el coste. Lo más común es utilizar una versión vectorial y la llamada Ecuación Normal que nos dará un resultado directo.

6. Problema de aplicación

6.1. Regresión lineal simple

En este notebook intentaremos predecir cuántas veces será compartido en Redes Sociales un artículo de Machine Learning según algunas de sus características. Comencemos por importar las librerías que utilizaremos

```
# Imports necesarios
import numpy as np
import pandas as pd
import seaborn as sb
import matplotlib.pyplot as plt
%matplotlib inline
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
plt.rcParams['figure.figsize'] = (16, 9)
plt.style.use('ggplot')
from sklearn import linear_model
from sklearn.metrics import mean_squared_error,
r2_score
```

Leemos el archivo csv y lo cargamos como un dataset de Pandas. Y vemos su tamaño.

```
#cargamos los datos de entrada
data = pd.read_csv("./articulos_ml.csv")
data
#veamos cuantas dimensiones y registros
contiene
data.shape
```

Nos devuelve (161,8). Veamos esas primeras filas:

```
#son 161 registros con 8 columnas. Veamos
los primeros registros
data.head()
```

Vease la figura (2) Se ven algunos campos con valores NaN (nulos) por ejemplo algunas urls o en comentarios.
En nuestro caso la columna Shares será nuestra salida, es decir nuestro valor "Y", el valor que queremos predecir.

```
# Ahora veamos algunas estadísticas de nuestros datos
data.describe()
```

Vease la figura (3) Aqui vemos que la media de palabras en los artículos es de 1808. El artículo más corto tiene 250 palabras y el más extenso 8401.
Y en cuanto a las saldas, vemos mínimo de 0 veces compartido y máximo de 350000 (eso es mucho!)
Intentaremos ver con nuestra relación lineal, si hay

una correlación entre la cantidad de palabras del texto y la cantidad de Shares obtenidos.
Hacemos una visualización en general de los datos de entrada:

6.1.1. Visualización General

```
# Visualizamos rápidamente las características
de entrada
data.drop(['Title','url',
'Elapsed days'],1).hist()
plt.show()
```

Vease la figura (4) En estas gráficas vemos entre qué valores se concentran la mayoría de registros.

6.1.2. Visualizamos Cantidad de palabras vs Compartidos

```
#vamos a Visualizar los datos de entrada
colores=['orange','blue']
tamanios=[30,60]
f1 = data['Word count'].values
f2 = data['# Shares'].values
# Vamos a pintar en 2 colores los puntos
por debajo de la media de Cantidad
de Palabras
asignar=[]
for index, row in data.iterrows():
if(row['Word count']>1808):
asignar.append(colores[0])
else:asignar.append(colores[1])
plt.scatter(f1, f2, c=asignar, s=tamanios[0])
plt.show()
```

Vease la figura (5) Vamos a filtrar los datos de cantidad de palabras para quedarnos con los registros con menos de 3500 palabras y también con los que tengan Cantidad de compartidos menos a 80.000.

```
#Vamos a RECORTAR los datos en la zona donde
se concentran más los puntos
# esto es en el eje X: entre 0 y 3.500
# y en el eje Y: entre 0 y 80.000
filtered_data =
data[(data['Word count'] <= 3500)
& (data['# Shares'] <= 80000)]
f1 = filtered_data['Word count'].values
f2 = filtered_data['# Shares'].values
# Vamos a pintar en colores los puntos por
debajo y por encima de la media de Cantidad
de Palabras
asignar=[]
for index, row in filtered_data.iterrows():
if(row['Word count']>1808):
asignar.append(colores[0])
else: asignar.append(colores[1])
plt.scatter(f1, f2, c=asignar, s=tamanios[0])
plt.show()
```

Vease la figura (6)

```
# Veamos como cambian los valores una vez
filtrados filtered_data.describe()
```

Vease la figura (7)

6.1.3. Regresión Lineal con Python y SKLearn

Vamos a crear nuestros datos de entrada por el momento sólo Word Count y como etiquetas los Shares. Creamos el objeto LinearRegression y lo hacemos «encajar» (entrenar) con el método fit(). Finalmente imprimimos los coeficientes y puntajes obtenidos.

```
#Asignamos nuestra variable de entrada X
para entrenamiento y las etiquetas Y.
dataX =filtered_data[["Word count"]]
X_train = np.array(dataX)
```

```
y_train = filtered_data['# Shares'].values


#Creamos el objeto de Regresión Linear
regr = linear_model.LinearRegression()
#Entrenamos nuestro modelo
regr.fit(X_train, y_train)
#Hacemos las predicciones que en definitiva
una línea (en este caso, al ser 2D)
y_pred = regr.predict(X_train)
#Veamos los coeficienetes obtenidos,
En nuestro caso, serán la Tangente
print('Coefficients: \n', regr.coef_)
#Este es el valor donde corta el eje Y
(en X=0)
print('Independent term: \n', regr.intercept_)
#Error Cuadrado Medio
print("Mean squared error: %.2f" %
mean_squared_error(y_train, y_pred))
#Puntaje de Varianza.
% El mejor puntaje es un 1.0
print('Variance score:
%.2f' % r2_score(y_train, y_pred))


    Coefficients:
 [5.69765366]
Independent term:
 11200.303223074163
Mean squared error: 372888728.34
Variance score: 0.06
```

De la ecuación de la recta $y = mX + b$ nuestra pendiente «m» es el coeficiente 5,69 y el término independiente «b» es 11200. Tenemos un Error Cuadrático medio enorme por lo que en realidad este modelo no será muy bueno. Esto también se ve reflejado en el puntaje de Varianza que debería ser cercano a 1.0.

6.1.4. Visualicemos la recta

Vease la Figura (8)

```
plt.scatter(X_train[:,0], y_train,  c=asignar, s=tama
plt.plot(X_train[:,0], y_pred, color='red', linewidth
plt.xlabel('Cantidad de Palabras')
plt.ylabel('Compartido en Redes')
plt.title('Regresión Lineal')
plt.show()
```

6.1.5. Predicción 1:

Vamos a intentar probar nuestro algoritmo, suponiendo que quisiéramos predecir cuántos «compartir» obtendrá un articulo sobre ML de 2000 palabras.

```
#Vamos a comprobar:
#Quiero predecir cuántos "Shares" voy a
obtener por un artículo con 2.000 palabras,
#según nuestro modelo, hacemos:
y_Dosmil = regr.predict([[2000]])
print(int(y_Dosmil))
22595
```

Nos devuelve una predicción de 22595 «Shares» para un artículo de 2000 palabras. Vamos a mejorar un poco el modelo.

6.2. Regresión Lineal Múltiple

Vamos a extender el ejercicio utilizando más de una variable de entrada para el modelo. Esto le da mayor poder al algoritmo de Machine Learning, pues de esta manera podremos obtener predicciones más complejas. Nuestra «ecuación de la Recta», ahora pasa a ser:

$$Y = b + m_1X_1 + m_2X_2 + \cdots + m_nX_n \qquad (20)$$

En nuestro caso, utilizaremos 2 «variables predictivas» para poder graficar en 3D, pero recordar que

para mejores predicciones podemos utilizar más de 2 entradas y prescindir del gráfico.

Nuestra primer variable seguirá siendo la cantidad de palabras y la segunda variable será la suma de 3 columnas de entrada: la cantidad de enlaces, comentarios y cantidad de imágenes

```
#Vamos a intentar mejorar el Modelo,
con una dimensión más:
# Para poder graficar en 3D,
haremos una variable nueva que será la suma
de los enlaces, comentarios e imágenes
suma = (filtered_data["# of Links"]
+ filtered_data['# of comments'].fillna(0)
+ filtered_data['# Images vídeo'])
dataX2 =  pd.DataFrame()
dataX2["Word count"] =
filtered_data["Word count"]
dataX2["suma"] = suma
XY_train = np.array(dataX2)
z_train = filtered_data['# Shares'].values
```

Ya tenemos nuestras 2 variables de entrada en $XYtrain$ y nuestra variable de salida pasa de ser «Y» a ser el eje «Z».

Creamos un nuevo objeto de Regresión lineal con SKLearn pero esta vez tendrá las dos dimensiones que entrenar: las que contiene $XYtrain$. Al igual que antes, imprimimos los coeficientes y puntajes obtenidos:

```
# Creamos un nuevo objeto de Regresión Lineal
regr2 = linear_model.LinearRegression()
# Entrenamos el modelo, esta vez,
con 2 dimensiones
# obtendremos 2 coeficientes,
para graficar un plano
regr2.fit(XY_train, z_train)
# Hacemos la predicción con la que tendremos
puntos sobre el plano hallado
z_pred = regr2.predict(XY_train)
# Los coeficientes
print('Coefficients: \n', regr2.coef_)
# Error cuadrático medio
print("Mean squared error:
%.2f" % mean_squared_error(z_train, z_pred))
# Evaluamos el puntaje de varianza
(siendo 1.0 el mejor posible)
print('Variance score:
%.2f' % r2_score(z_train, z_pred))
Coefficients:
[6.63216324 -483.40753769]
Mean squared error: 352122816.48
Variance score: 0.11
```

Como vemos, obtenemos 2 coeficientes (cada uno correspondiente a nuestras 2 variables predictivas), pues ahora lo que graficamos no será una linea si no, un plano en 3 Dimensiones.

El error obtenido sigue siendo grande, aunque algo mejor que el anterior y el puntaje de Varianza mejora casi el doble del anterior (aunque sigue siendo muy malo, muy lejos del 1).

6.2.1. Esta vez visualizamos un plano:

Graficaremos nuestros puntos de las características de entrada en color azul y los puntos proyectados en el plano en rojo. Recordemos que en esta gráfica, el eje Z corresponde a la «altura» y representa la cantidad de Shares que obtendremos.

```
fig = plt.figure()
ax = Axes3D(fig)
# Creamos una malla, sobre la cual
graficaremos el plano
xx, yy = np.meshgrid(np.linspace(0, 3500, num=10),
```

```
np.linspace(0, 60, num=10))
# calculamos los valores del plano
para los puntos x e y
nuevoX = (regr2.coef_[0] * xx)
nuevoY = (regr2.coef_[1] * yy)
# calculamos los correspondientes valores
para z. Debemos sumar el punto de
intercepción
z = (nuevoX + nuevoY + regr2.intercept_)
# Graficamos el plano
ax.plot_surface(xx,yy,z,alpha=0.2,cmap='hot')
# Graficamos en azul los puntos en 3D
ax.scatter(XY_train[:, 0], XY_train[:, 1],
z_train, c='blue',s=30)
# Graficamos en rojo, los puntos que
ax.scatter(XY_train[:, 0], XY_train[:, 1],
z_pred, c='red',s=40)
# con esto situamos la "cámara"
con la que visualizamos
ax.view_init(elev=30., azim=65)
ax.set_xlabel('Cantidad de Palabras')
ax.set_ylabel
('Cantidad de Enlaces,Comentarios e Imágenes')
ax.set_zlabel('Compartido en Redes')
ax.set_title
('Regresión Lineal con Múltiples Variables')
```

Véase Figura (9)

6.2.2. Predicción 2:

Veamos ahora, que predicción tendremos para un artículo de 2000 palabras, con 10 enlaces, 4 comentarios y 6 imágenes.

```
# Si quiero predecir cuántos "Shares"
voy a obtener por un artículo con:
# 2000 palabras y con enlaces:
10, comentarios: 4, imágenes: 6
# según nuestro modelo, hacemos:
z_Dosmil = regr2.predict([[2000, 10+4+6]])
print(int(z_Dosmil))
```

Esta predicción nos da 20518 y probablemente sea un poco mejor que nuestra predicción anterior con 1 variables.

6.2.3. Comparemos las predicciones obtenidas en los modelos:

En este caso, Obtuvimos mejora en el modelo de 2 dimensiones:

```
#Restamos los errores calculados antes:
#Obviamente, "menos error" es mejor
mejoraEnError =
mean_squared_error(y_train, y_pred) -
mean_squared_error(z_train, z_pred)
print(mejoraEnError)
```

20765911.860715985

```
#También calculamos la mejora en la varianza:
mejoraEnVarianza = r2_score(z_train, z_pred) -
r2_score(y_train, y_pred)
print(mejoraEnVarianza)
# Aunque no parezca mucho, recordemos que
el valor más alto que se puede obtener es 1.0
```

0.052615337462582956

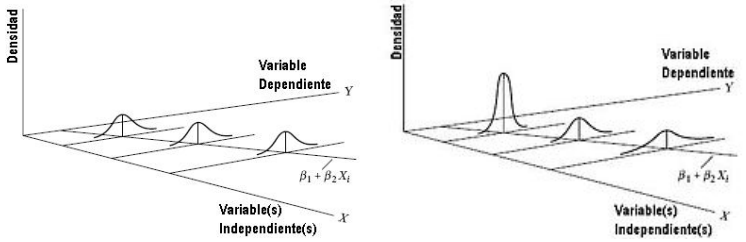
```
#Finalmente, mejoramos en nuestra predicción
de un artículo de 2.000 palabras,
#pues aunque disminuyen los "Shares"
que obtendremos en el 2do modelo,
#seguramente será un valor más cercano
a la realidad
diferenciaComparir = z_Dosmil - y_Dosmil
print(int(diferenciaComparir))
```

-2077

7. Conclusiones

Hemos visto cómo utilizar SKLearn en Python para crear modelos de Regresión Lineal con 1 o múltiples variables. En nuestro ejercicio no tuvimos una gran confianza en las predicciónes. Por ejemplo en nuestro primer modelo, con 2000 palabras nos predice que podemos tener 22595 pero el margen de error haciendo raíz del error cuadrático medio es mas menos 19310. Para mejorar nuestro modelo, deberíamos utilizar más dimensiones y encontrar datos de entrada mejores. También es posible, que no exista ninguna relación nunca entre nuestras variables de entrada y el éxito en "Shares"del artículo... con lo cual... nunca podremos predecir con certeza esta salida. En este ejemplo utilizamos información de artículos sobre Machine Learning con algunos datos ficticios en nuestro .csv pero si quisiéramos tratar de mejorar las predicciones, deberíamos utilizar más de 2 variables (recordemos que aquí lo hicimos para poder graficar en 3D). Sería conveniente además conseguir mejores características de entrada, pues no es lo mismo un enlace de un portal con multitud de visitas al mes, que otros Blogs más modestos.

8. Anexo:



(a) Homocedasticidad (b) Heterocedasticidad

Figura 1: Perturbaciones

	Title	url	Word count	# of Links	# of comments	# Images video	Elapsed days	# Shares
0	What is Machine Learning and how do we use it ...	https://blog.signals.network/what-is-machine-l...	1888	1	2.0	2	34	200000
1	10 Companies Using Machine Learning in Cool Ways	NaN	1742	9	NaN	9	5	25000
2	How Artificial Intelligence Is Revolutionizing...	NaN	962	6	0.0	1	10	42000
3	Dbrain and the Blockchain of Artificial Intell...	NaN	1221	3	NaN	2	68	200000
4	Nasa finds entire solar system filled with eig...	NaN	2039	1	104.0	4	131	200000

Figura 2: Tabla 1

	Word count	# of Links	# of comments	# Images video	Elapsed days	# Shares
count	161.000000	161.000000	129.000000	161.000000	161.000000	161.000000
mean	1808.260870	9.739130	8.782946	3.670807	98.124224	27948.347826
std	1141.919385	47.271625	13.142822	3.418290	114.337535	43408.006839
min	250.000000	0.000000	0.000000	1.000000	1.000000	0.000000
25%	990.000000	3.000000	2.000000	1.000000	31.000000	2800.000000
50%	1674.000000	5.000000	6.000000	3.000000	62.000000	16458.000000
75%	2369.000000	7.000000	12.000000	5.000000	124.000000	35691.000000
max	8401.000000	600.000000	104.000000	22.000000	1002.000000	350000.000000

Figura 3: Tabla 2

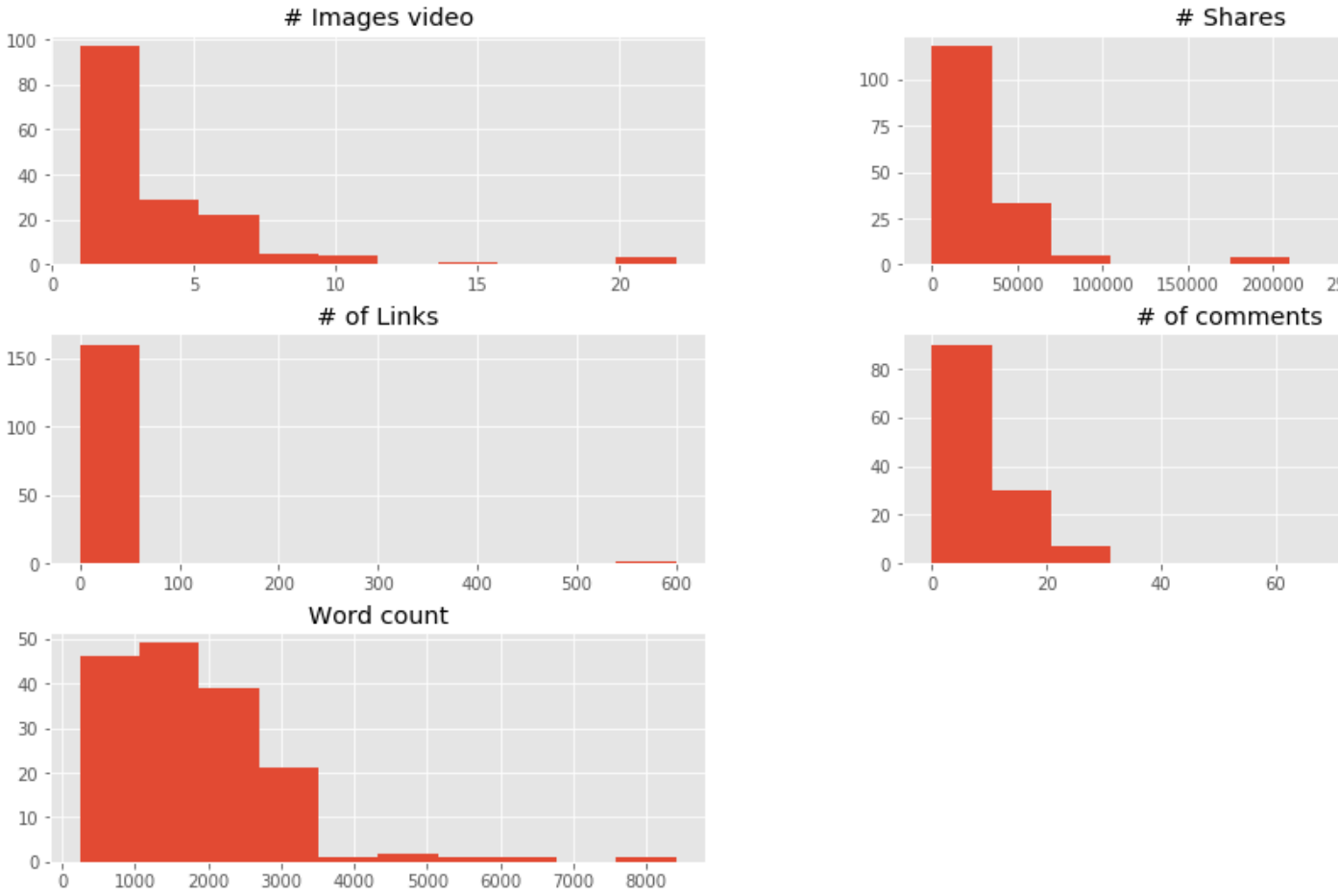


Figura 4: Grafica 1

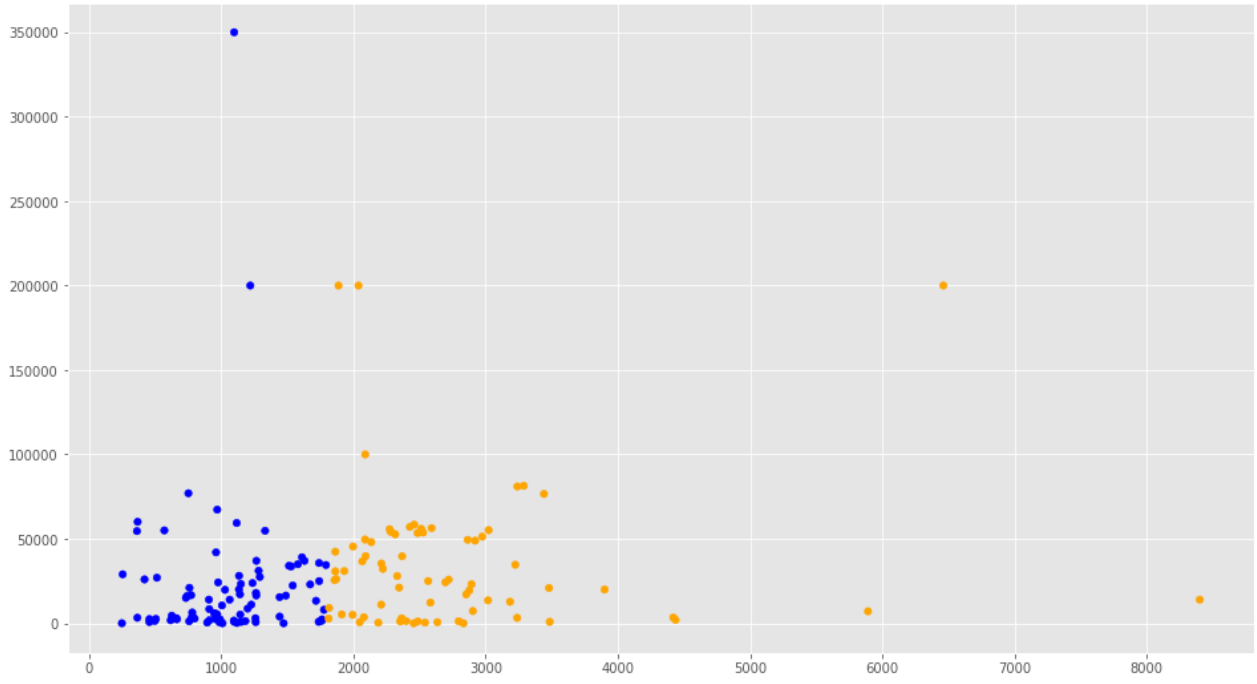


Figura 5: Grafica 2

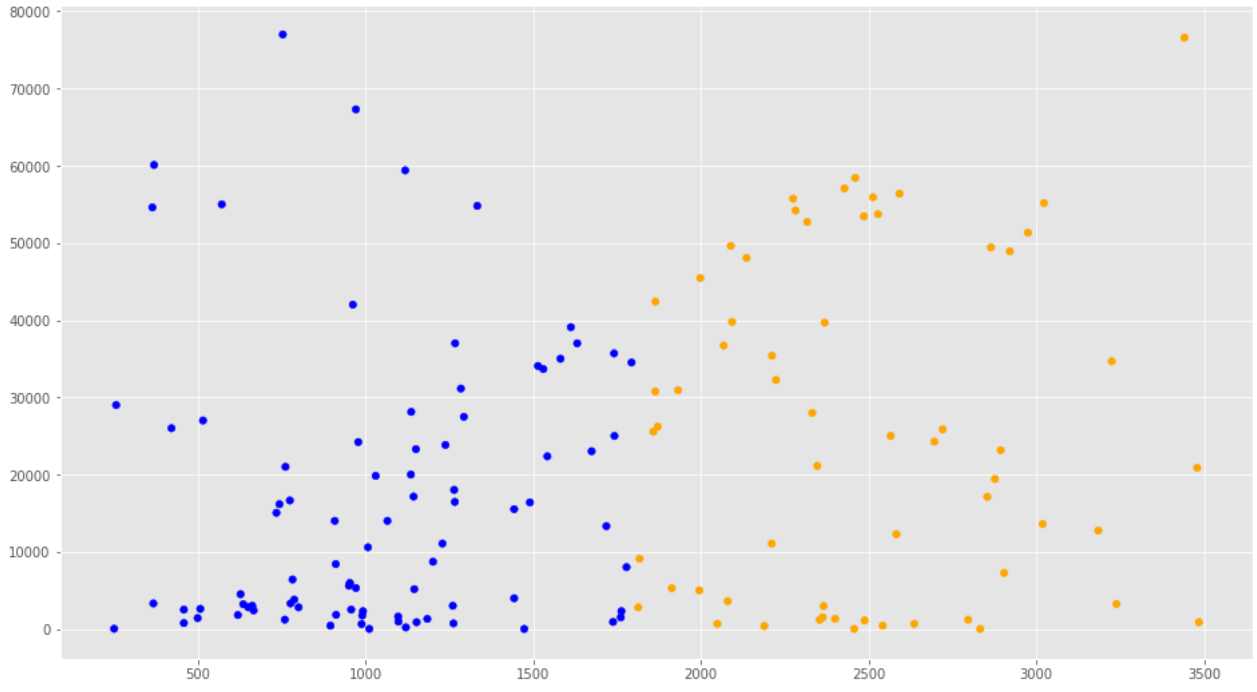


Figura 6: Grafica 3

	Word count	# of Links	# of comments	# Images video	Elapsed days	# Shares
count	148.000000	148.000000	121.000000	148.000000	148.000000	148.000000
mean	1640.209459	5.743243	7.256198	3.331081	91.554054	20545.648649
std	821.975365	6.064418	6.346297	2.706476	91.143923	19933.865031
min	250.000000	0.000000	0.000000	1.000000	1.000000	0.000000
25%	971.000000	3.000000	2.000000	1.000000	28.750000	2750.000000
50%	1536.000000	5.000000	6.000000	3.000000	60.000000	15836.000000
75%	2335.750000	7.000000	11.000000	4.000000	110.500000	34177.500000
max	3485.000000	49.000000	30.000000	22.000000	349.000000	77000.000000

Figura 7: Tabla3

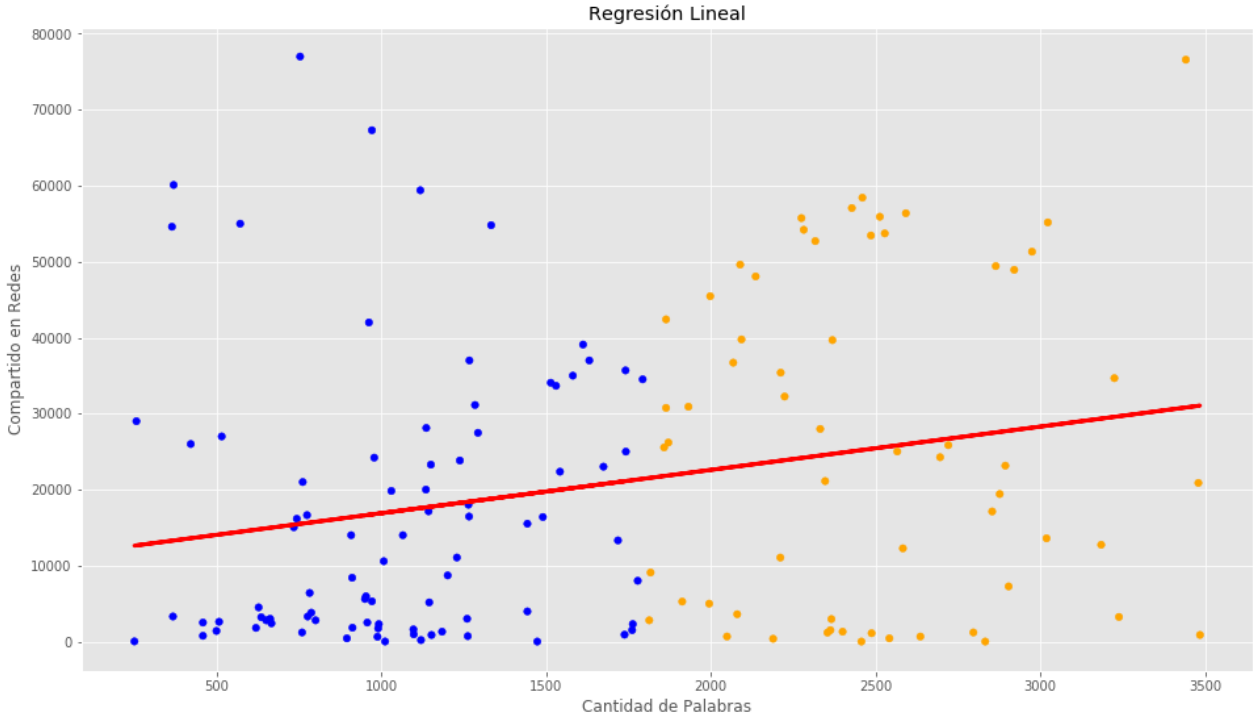


Figura 8: Grafica 4

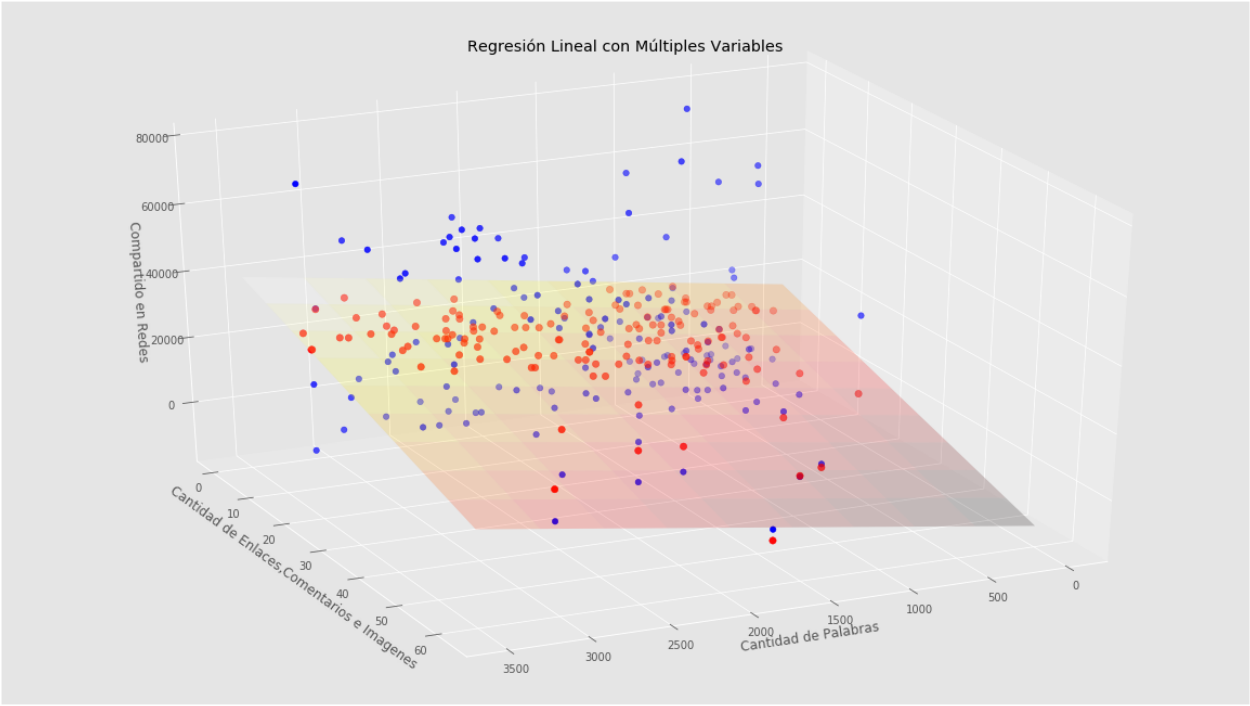


Figura 9: Grafica 5

REFERENCIAS

[1.] Tverberg, H., A genereralization of Randon's theorem, ***J. London Math. Soc.* 37** (1966), 123–128.

[2.] Eckhoff, Jürgen; Randon's theorem revisited. ***Contributions to geometry(Proc. Geom. Sympos., Siegen, 1978)***, pp. 164-185, ***Birkhäuser, Basel-Boston, Mass.***, 1979.

[3.] Eckhoff, Jürgen; Helly Radon, and Carathéodory type theorems. ***Handbook of convex geometry, Vol. A, B***, 389–448, ***North-Holland, Amsterdam***, 1993.

[4.] Matoušek, Jiří; Lectures on discretre geometry. Graduate Texts in Mathematics, 212. ***Springer-Verlag, New York***, 2002. xvi+481 pp. ISBN: 0-387-9537-6

[5.] Bárány, Imre, Blagojević, Pavle V. M.; Ziegler, Günter M.; Tverberg's theorem at 50: extensions and counterexamples. ***Notices Amer. Math. Soc.*** 63(2016), no. 7, 732-739

[6.] De Loera, Jesus A. and Goaoc, Xavier and Meunier, Frédéric and Mustafa, Nabil; The discrete yet ubiquitous theorems of Carathéodory, Helly, Sperner, Tucker, and Tverberg. ***arXiv preprint arXiv:1706.05975***, 2017.

[7.] Blagojević, Pavle V. M. and Ziegler, Günter M.; Beyond the Borsuk-Ulam theorem: the topological Tverberg story. ***A journey through discrete mathematics***, 273–341, ***springer, cham***, 2017.

[8.] Radon, Johann; Mengen konvexer Körper, die einen gemeinsamen Punkt Enthalten. (German) ***Math. Ann.*** 8381921), no. 1-2, 113–115.

[9.] Birch, B. J., On $3N$ points in a plane. ***Proc. Cambridge Philos. Soc.*** 5581959), 289–293.

[10.] Tverberg, Helge; A combinatorial mathematician in Norway: some personal reflections. Selected papers in honor of Helge Tverberg. ***Discrete Math.*** 241(2001), no. 1-3, 11–22.