MATEMATICA III

Trabajo Practico – Transacciones Fraudulentas

Resumen

El fraude en las transacciones personales es un problema cada vez más común, que afecta a usuarios de servicios financieros y genera pérdidas importantes a nivel individual. Con el avance del aprendizaje automático, se abren nuevas posibilidades para desarrollar sistemas que identifiquen patrones de fraude y ayuden a prevenirlo. Esta tesis aborda el análisis de transacciones personales mediante un conjunto de datos de operaciones realizadas por individuos.

Docentes a cargo: Bompensieri Josefina & Prudente Tomas
Integrantes: Nicolas Cernadas & Catalina Correa

Primeros pasos

En esta ocasión, el Data Set seleccionado fue “Credit Card Fraud”, tomado de la página “Kaggle”.

Para empezar, el Data Set ¡contaba con 1.000.000 de datos! Para poder trabajarlo con mas claridad y eficacia, redujimos esta cantidad. En el total, había 87.403 datos los cuales concluían en ser transacciones fraudulentas, lo cual nos deja con 912.597 transacciones que no lo son. Hicimos lo siguiente: Utilizando un script de funciones definido por nosotros, borramos las primeras 820.000 filas que terminaran por no ser fraudulentas, lo cual nos dejo con 251.648 datos, todavía bastante. Para no tener que repetir el proceso de borrado de muestras siempre que se quisiera probar algo nuevo, una vez borradas las 820.000 filas, se guardo como un nuevo csv, y se pasó a utilizar ese. Ambos csvs se encuentran en la carpeta de, valga la redundancia, csvs.

Variables continuas:

distance_from_home: Representa la distancia entre el lugar de la transacción y el hogar del cliente.

- Distancias largas pueden indicar actividad inusual.

distance_from_last_transaction: Mide la distancia entre la transacción actual y la última transacción realizada.

- Cambios repentinos en distancia pueden señalar actividad sospechosa.

ratio_to_median_purchase_price: Es la relación entre el precio de la compra y el precio de compra promedio.

- Transacciones con ratios inusualmente altos o bajos pueden ser anómalas.

Variables categóricas:

repeat_retailer: Indica si la transacción fue realizada en el mismo comercio que la anterior.

- Las transacciones en el mismo lugar pueden considerarse de menor riesgo.

used_chip: Indica si se utilizó chip en la tarjeta para la transacción.

- Las transacciones sin chip pueden ser más vulnerables a fraudes.

used_pin_number: Señala si se ingresó un PIN en la transacción.

- El uso de PIN añade seguridad y puede reducir el riesgo de fraude.

online_order: Identifica si la transacción fue una compra en línea.

- Las compras en línea suelen tener un riesgo de fraude mayor.

fraud: Variable objetivo que indica si la transacción fue clasificada como fraudulenta.

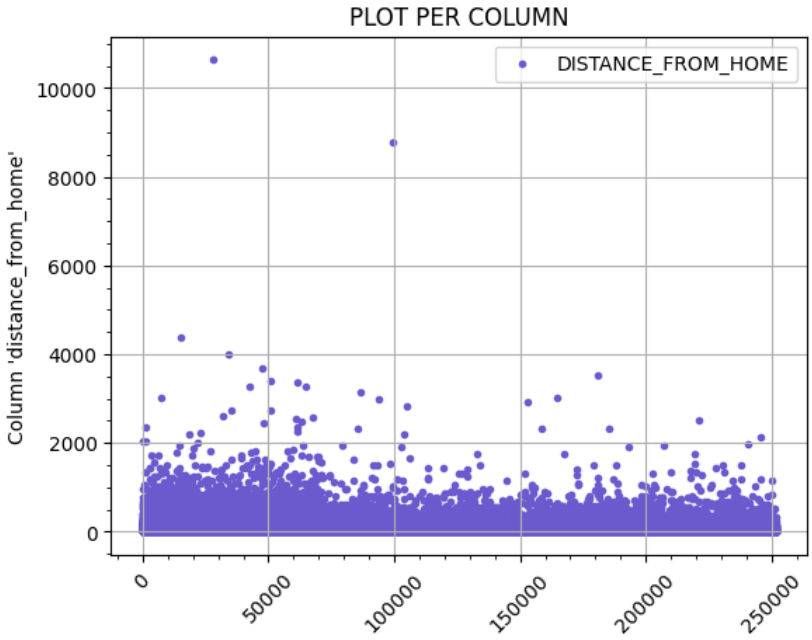
Primeras impresiones del Data Set

```
Drescripcion del DF sin procesar

       distance_from_home  distance_from_last_transaction  ratio_to_median_purchase_price  repeat_retailer      used_chip  used_pin_number   online_order          fraud
count       251648.000000                  251648.000000                   251648.000000    251648.000000  251648.000000    251648.000000  251648.000000  251648.000000
mean            37.774015                       7.267704                        3.016502         0.880762       0.323718         0.072160       0.734681       0.347322
std             93.023494                      34.096929                        4.234568         0.324069       0.467895         0.258754       0.441504       0.476120
min              0.025847                       0.000298                        0.007278         0.000000       0.000000         0.000000       0.000000       0.000000
25%              4.018835                       0.306884                        0.588911         1.000000       0.000000         0.000000       0.000000       0.000000
50%             10.941643                       1.044075                        1.424900         1.000000       0.000000         0.000000       1.000000       0.000000
75%             31.920588                       3.649177                        4.345763         1.000000       1.000000         0.000000       1.000000       1.000000
max          10632.723672                    4968.315477                      266.689692         1.000000       1.000000         1.000000       1.000000       1.000000

15 Primeras filas

    distance_from_home  distance_from_last_transaction  ratio_to_median_purchase_price  repeat_retailer  used_chip  used_pin_number  online_order  fraud
0            2.131956                       56.372401                        6.358667              1.0        0.0              0.0           1.0    1.0
1            3.803057                       67.241081                        1.872950              1.0        0.0              0.0           1.0    1.0
2           15.694986                      175.989182                        0.855623              1.0        0.0              0.0           1.0    1.0
3           26.711462                        1.552008                        4.603601              1.0        1.0              0.0           1.0    1.0
4           10.664474                        1.565769                        4.886521              1.0        0.0              0.0           1.0    1.0
5            2.530145                        3.689781                        8.297407              1.0        0.0              0.0           1.0    1.0
6           21.126116                        0.271987                        6.081771              1.0        0.0              0.0           1.0    1.0
7          151.370437                        5.340081                        1.171567              1.0        0.0              0.0           1.0    1.0
8            9.598401                        0.454556                        6.084829              1.0        0.0              0.0           1.0    1.0
9           22.545588                        0.353939                        4.095442              1.0        0.0              0.0           1.0    1.0
10         335.189320                        1.114168                        0.098243              1.0        0.0              0.0           1.0    1.0
11          43.326444                        0.205255                        4.250634              1.0        0.0              0.0           1.0    1.0
12           1.115881                        0.170237                        4.127313              0.0        0.0              0.0           0.0    1.0
13           5.793890                        2.758511                        5.085686              1.0        0.0              0.0           1.0    1.0
14          22.641962                        0.015625                        4.322207              1.0        0.0              0.0           1.0    1.0

NaNs por columna

Series([], dtype: float64)
```

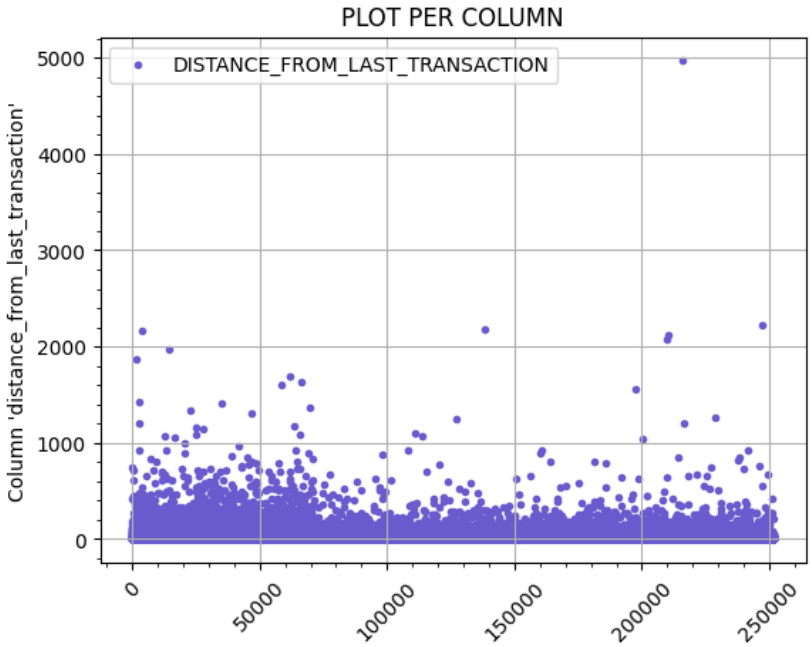
Bien, podemos empezar. ¡No tenemos valores faltantes! Eso es un notición, ya que no vamos a tener que alterar el valor de una fila completa por simplemente un valor, o eliminarla. A primera vista pareciera haber valores atípicos en las primeras 3 columnas, ya que el mínimo esta bastante alejado del máximo en los 3 casos, pero eso lo vamos a ver ahora.

Primero veamos algunos gráficos de las columnas que valen la pena ver gráficamente. (las 3 primeras, que son variables continuas, ya que las categóricas cuentan solamente con unos y ceros.)

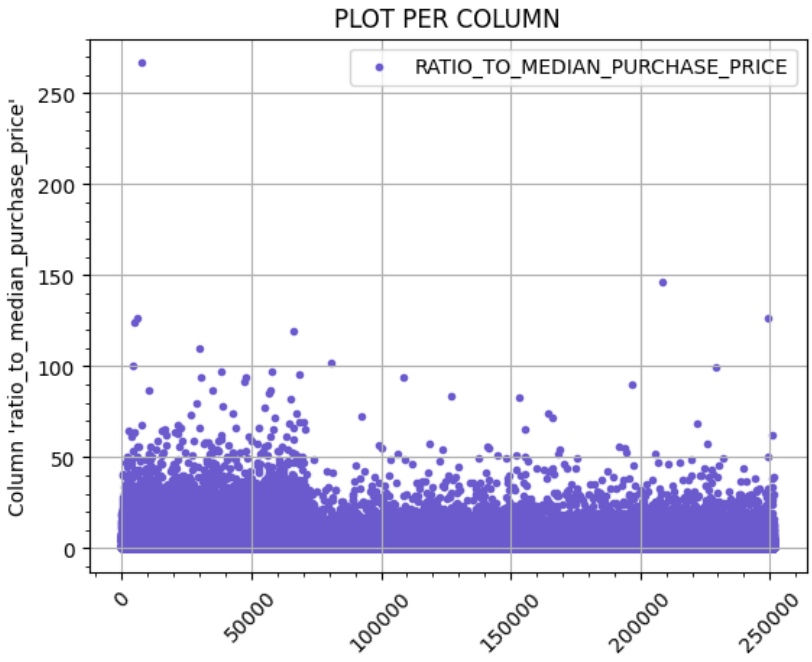
Tenemos primero la distancia desde el hogar hasta donde se realizo la transacción. Vemos que la mayoría se suelen hacer mas bien cerca del hogar. Esto parece ser mas seguro que las transacciones a largas distancias.



Distancia desde la última transacción

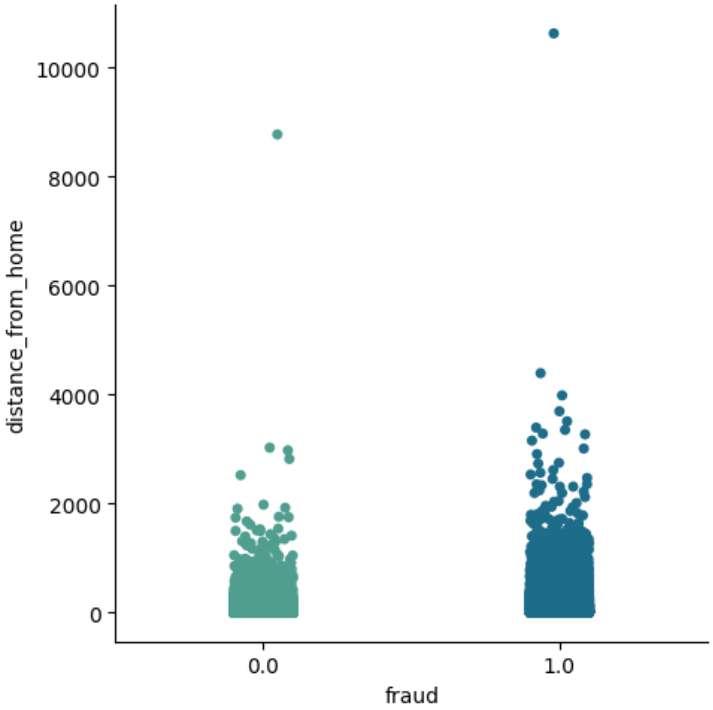


Y acá, podemos ver lo siguiente: cuando el ratio es cercano a 1, significa que el precio de la compra es cercano a la media del precio de las cosas que suele comprar el usuario. Osea que si el ratio es 0, compro algo mucho mas barato de las cosas que suele comprar. Un pequeño y tonto ejemplo sería, por ejemplo, alguien que tiene una tarjeta exclusiva para comprar ropa, se olvida de agarrar billetes y tiene que pagar unos chicles con la tarjeta. El ratio de esa compra, va a ser bajísimo, ya que está muy por debajo de una remera que hoy en día te termina saliendo cerca de los 20.000.

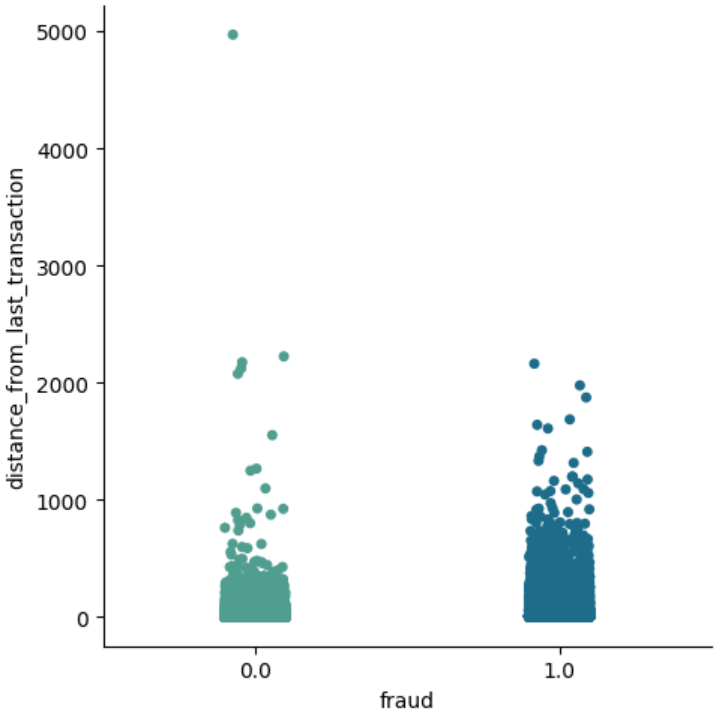


En los próximos 3 gráficos, los podemos ver representados en base a si terminan siendo o no, transacciones fraudulentas.

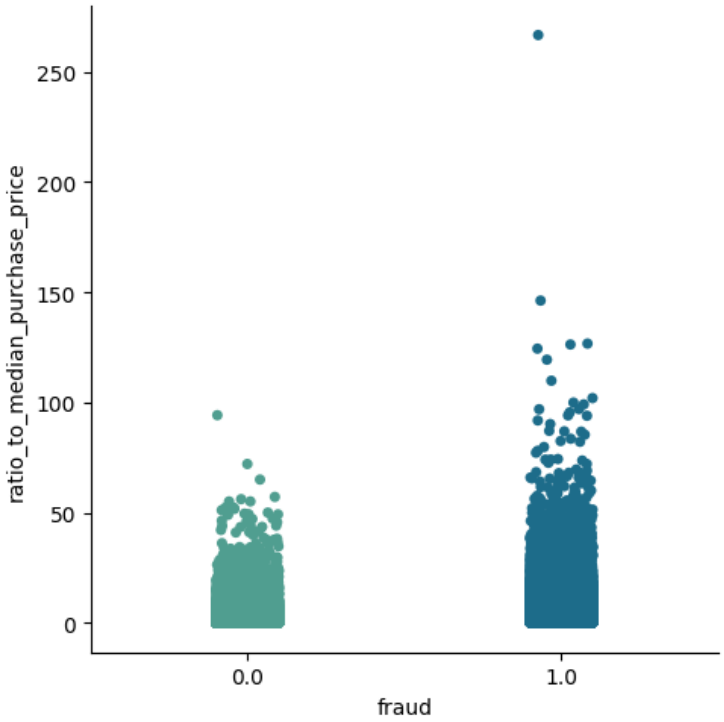
‘distance_from_home’



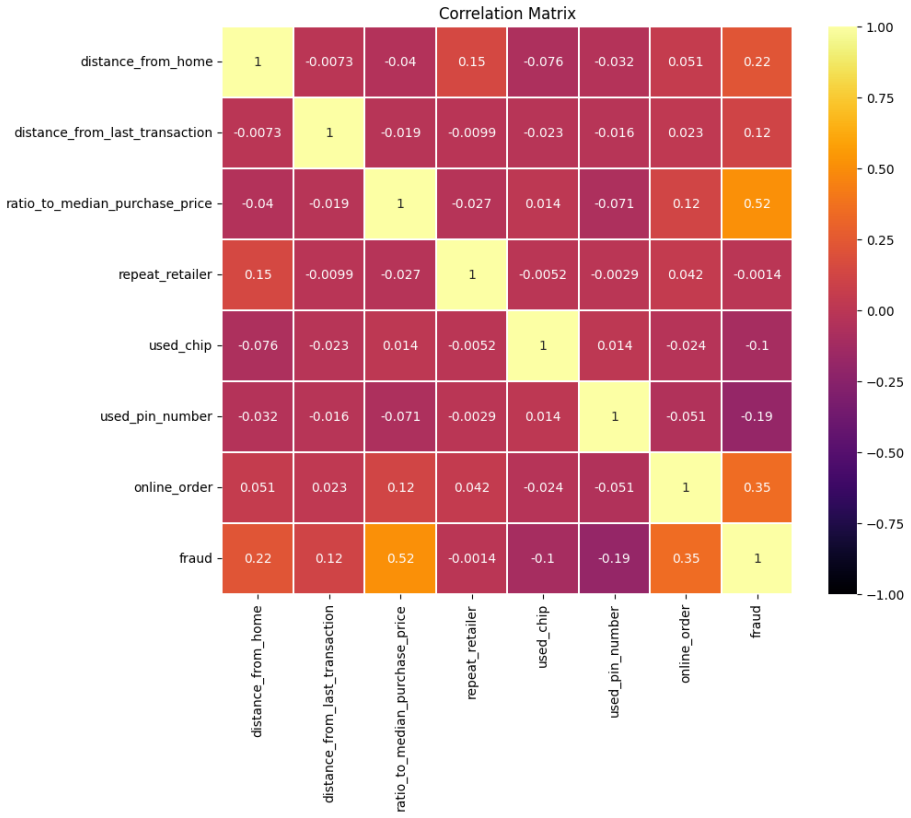
‘distance_from_last_transaction



‘ratio_to_median_purchase_price’



En los 3 gráficos, podemos ver como los valores mas altos (mas alejados del hogar o de la última transacción en los primeros 2 casos, y mas alejado del ratio en el tercero) parecen ser indicadores de fraude. Pareciera que, de alguna manera, estos se relacionan. ¿Cómo? ¿Osea que si compro algo lejos de mi casa va a ser un fraude? No creo que sea tan lineal, pero para verlo mejor, podemos usar una matriz de correlación:



En el grafico se puede ver la relación de una columna con otra. Por ejemplo, el segundo cuadradito de la primera fila de arriba de todo (léase fila 0, posición 1 si se quiere), nos muestra la relación de la columna ‘distance_from_home’ con la columna ‘distance_from_last_transaction’: -0,0073. ¿Qué quiere decir? La correlación mide la relación entre 2 valores. Va entre -1 y 1, indicando la fuerza y dirección de una relación lineal. Si este valor estuviese cerca de -1, significaría que están fuertemente ligados en una relación lineal que decrece, y viceversa si estuviese cerca de 1. Al estar cerca de 0, quiere decir que su relación lineal es casi nula, y son valores independientes. ¿A que columna deberíamos prestarle atención? Y…A la de fraude. Ahí nos damos cuenta de que valores influyen más en esto, como por ejemplo la relación de ‘fraud’ y ‘ratio_to_median_purchase’, la cual es de 0.52, bastante relacionados.

Tratamiento del Data Set

Una vez visualizados un poco los datos y explicados un par de conceptos, vamos a proceder a tratarlos.

Dado el hecho de que no tenemos celdas vacías, lo que resta por hacer, es deshacernos de los outliers, para después estandarizar los datos.

Una vez hecho esto, el Data Frame quedo de la siguiente manera:

Drescripcion del DF post procesamiento

	distance_from_home	distance_from_last_transaction	ratio_to_median_purchase_price	repeat_retailer	used_chip	used_pin_number	online_order	fraud
count	180304.000000	180304.000000	180304.000000	180304.000000	180304.000000	180304.000000	180304.000000	180304.000000
mean	14.573810	1.571295	2.434710	0.864706	0.347624	0.080459	0.710273	0.239179
std	15.659575	1.899988	2.417031	0.342038	0.476217	0.272003	0.453637	0.426584
min	0.025847	0.000298	0.007278	0.000000	0.000000	0.000000	0.000000	0.000000
25%	3.453440	0.252337	0.577178	1.000000	0.000000	0.000000	0.000000	0.000000
50%	8.533762	0.775117	1.361532	1.000000	0.000000	0.000000	1.000000	0.000000
75%	19.964630	2.147431	4.144441	1.000000	1.000000	0.000000	1.000000	0.000000
max	73.767535	8.721074	10.345535	1.000000	1.000000	1.000000	1.000000	1.000000

Como se puede apreciar, todas las columnas tienen la misma cantidad de datos, ya que estamos borrando completamente la fila donde aparezca un valor atípico, y no la celda, ya que esto, llevaría a que los datos se movieran de lugar, es decir, si yo borro la celda 0, la 1 pasa a su posición, y me cambia toda la fila 0, y no queremos que esto pase.

Ahora sí, vamos a hacer el 2do paso del que ya habíamos hablado, el cual es: estandarizar los datos. Con la media (promedio) de cada columna y la desviación estándar (que tanto se alejan los datos de la media), vamos a iterar, otra vez, sobre cada columna, realizando el siguiente calculo en cada uno de los datos: *(dato - media) / desviación estándar.* Esto, va a poner a absolutamente todos los valores de la columna en un mismo rango, ósea, ya no vamos a tener valores como 0.025 y 73.76, como en el caso de 'distance_from_home', sino que los valores se van a medir en cuanto a que tan alejados de la media se encuentren (a cuantas desviaciones estándar). Para entender mejor esto, veamos como queda el Data Frame después de este proceso:

Drescripcion del DF post procesamiento

	distance_from_home	distance_from_last_transaction	ratio_to_median_purchase_price	repeat_retailer	used_chip	used_pin_number	online_order	fraud
count	1.803040e+05	1.803040e+05	1.803040e+05	1.803040e+05	1.803040e+05	1.803040e+05	1.803040e+05	180304.000000
mean	-9.285520e-18	1.222437e-16	-3.052547e-16	1.613217e-16	4.224542e-17	-2.167442e-19	3.176288e-17	0.239179
std	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00	0.426584
min	-9.290139e-01	-8.268459e-01	-1.004304e+00	-2.528100e+00	-7.299695e-01	-2.958007e-01	-1.565729e+00	0.000000
25%	-7.101323e-01	-6.941931e-01	-7.685185e-01	3.955518e-01	-7.299695e-01	-2.958007e-01	-1.565729e+00	0.000000
50%	-3.857096e-01	-4.190436e-01	-4.440068e-01	3.955518e-01	-7.299695e-01	-2.958007e-01	6.386766e-01	0.000000
75%	3.442507e-01	3.032314e-01	7.073683e-01	3.955518e-01	1.369913e+00	-2.958007e-01	6.386766e-01	0.000000
max	3.780034e+00	3.763066e+00	3.272952e+00	3.955518e-01	1.369913e+00	3.380635e+00	6.386766e-01	1.000000

15 Primeras filas

	distance_from_home	distance_from_last_transaction	ratio_to_median_purchase_price	repeat_retailer	used_chip	used_pin_number	online_order	fraud
3	0.775095	-0.010151	0.897337	0.395552	1.369913	-0.295801	0.638677	1.0
4	-0.249645	-0.002908	1.014390	0.395552	-0.729969	-0.295801	0.638677	1.0
5	-0.769093	1.115000	2.425578	0.395552	-0.729969	-0.295801	0.638677	1.0
6	0.418422	-0.683851	1.508901	0.395552	-0.729969	-0.295801	0.638677	1.0
8	-0.317723	-0.587761	1.510166	0.395552	-0.729969	-0.295801	0.638677	1.0
9	0.509067	-0.640718	0.687096	0.395552	-0.729969	-0.295801	0.638677	1.0
11	1.836106	-0.718973	0.751304	0.395552	-0.729969	-0.295801	0.638677	1.0
13	-0.560674	0.624854	1.096790	0.395552	-0.729969	-0.295801	0.638677	1.0
14	0.515222	-0.818779	0.780916	0.395552	-0.729969	-0.295801	0.638677	1.0
16	-0.842568	-0.797273	1.008358	-2.528100	-0.729969	-0.295801	-1.565729	1.0
23	-0.426926	-0.556975	1.201026	0.395552	-0.729969	-0.295801	0.638677	1.0
26	-0.562557	-0.498294	2.536676	0.395552	1.369913	-0.295801	0.638677	1.0
27	-0.468962	1.293700	1.370029	0.395552	1.369913	-0.295801	0.638677	1.0
30	0.623496	-0.090883	1.570556	0.395552	1.369913	-0.295801	0.638677	1.0
32	-0.093247	-0.622312	0.816891	0.395552	-0.729969	-0.295801	0.638677	1.0

¿Qué paso? Ahora todos los datos están exactamente a 1 desviación estándar de la media, ósea: son cercanos entre sí. Si la media de la columna de 'distance_from_home' era 14.57, el valor mínimo anterior, nos va a quedar de la siguiente forma:

(0.025 – 14.57) / 15.69 = -0.92...

Y el valor máximo:

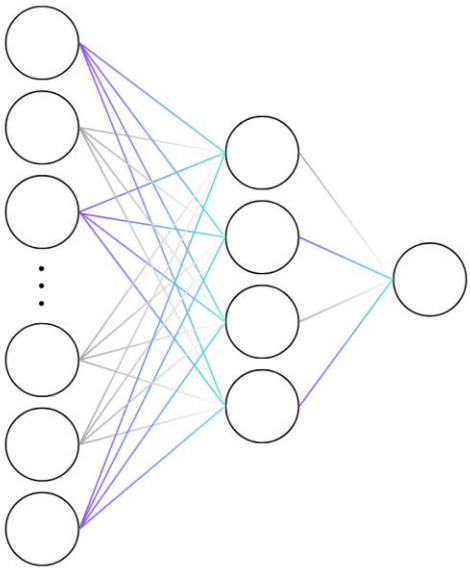
(73.76 – 14.57) / 15.69 = 3.77...

¡Epa! Ahora tenemos todos los valores “reducidos” a un mismo rango. Esto se hace para un optimo entrenamiento de la red, que viene a continuación.

Red Neuronal

Explicaciones y bocetos

Primero vamos a empezar armando un esquema de cómo va a ser nuestra red. De entrada, tenemos 7 datos (las 7 columnas), en la capa oculta, decidimos poner 4 neuronas, ya que fue lo que nos dio resultados mas consistentes. Y solamente 1 neurona de salida, que va a determinar si es, o no fraude. Algo de este estilo:



Pensamos también en las multiplicaciones matriciales. Vamos a pasar a explicar este concepto, pero después de un pantallazo visual:

$$Z1 = W_{hidden} \times X + B_{hidden}$$

$$Z1 = \begin{bmatrix} W1\ W2\ W3\ W4\ W5\ W6\ W7 \\ W8\ W9\ W10\ W11\ W12\ W13\ W14 \\ W15\ W16\ W17\ W18\ W19\ W20\ W21 \\ W22\ W23\ W24\ W25\ W26\ W27\ W28 \end{bmatrix} \begin{bmatrix} X1 \\ X2 \\ X3 \\ X4 \\ X5 \\ X6 \\ X7 \end{bmatrix} + \begin{bmatrix} B1 \\ B2 \\ B3 \\ B4 \end{bmatrix}$$

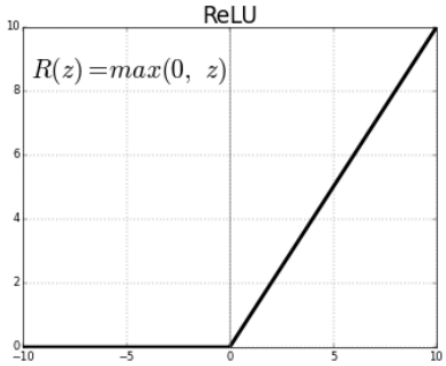
$$Z2 = W_{output} \times A1 + B_{output}$$

$$Z2 = \begin{bmatrix} W82\ W83\ W84\ W85 \end{bmatrix} \begin{bmatrix} A1 \\ A2 \\ A3 \\ A4 \end{bmatrix} + \begin{bmatrix} B10 \end{bmatrix}$$

Bien. Para empezar, decimos que *Z1 = Whidden x X + Bhidden*, ¿Y esto que es? Z1 es todo lo que se encuentra en las neuronas de la capa oculta, justo antes de que se aplique la función de activación (explicamos más adelante).

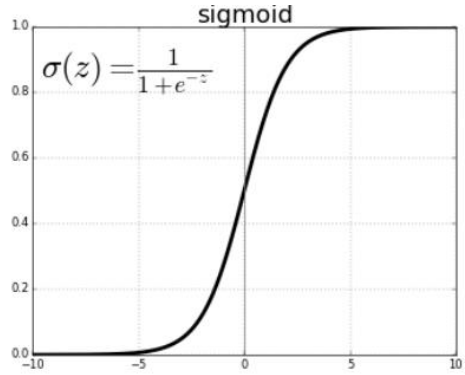
Pensemos cada neurona de entrada como una columna, la de arriba de todo sería la de 'distance_from_home', la de abajo la de 'distance_from_last_transaction'...y así. Osea, a cada neurona de la capa oculta le van a llegar 7 datos, 1 por columna. A esos datos, se les va a aplicar un peso random (0 < random < 1), al cual vamos a llamar W. Entonces no solo le llegan 7 datos, sino que le llegan 7 datos multiplicados cada uno por un valor random, y a eso, se le va a sumar un sesgo. ¿Qué es un sesgo? Es un valor de partida de las neuronas, es decir, si todo lo anterior da cero, todavía tenemos ese valor. Sirve para ajustar y flexibilizar la red para que pueda adaptarse a distintos tipos de patrones.

Ya se hizo toda la multiplicación, ¿Con que seguimos? Aplicamos la función de activación. Hay muchos tipos de función de activación para las capas ocultas, personalmente vamos a utilizar la ReLu. ¿Qué hace la función ReLu? Toma los valores negativos, y los convierte en 0. Los valores positivos, los deja como están.



Ahora que se aplicó la función de activación, la capa pasa a llamarse de otra manera, A1. Y así como dijimos que Z1 eran las multiplicaciones matriciales antes de la función de activación, Z2 son las multiplicaciones matriciales antes de la función sigmoide (¡pero posteriores a la función ReLu!), es decir, están justo antes de la salida de la última neurona.

Ahora, cuando se realizan las multiplicaciones matriciales entre A1, los nuevos pesos asignados a los datos y el bias (sesgo) de la última neurona, obtenemos Z2. Que como también paso anteriormente, es el dato previo a pasar por la función sigmoide, el cual lo convierte en A2, las predicciones de nuestra red.



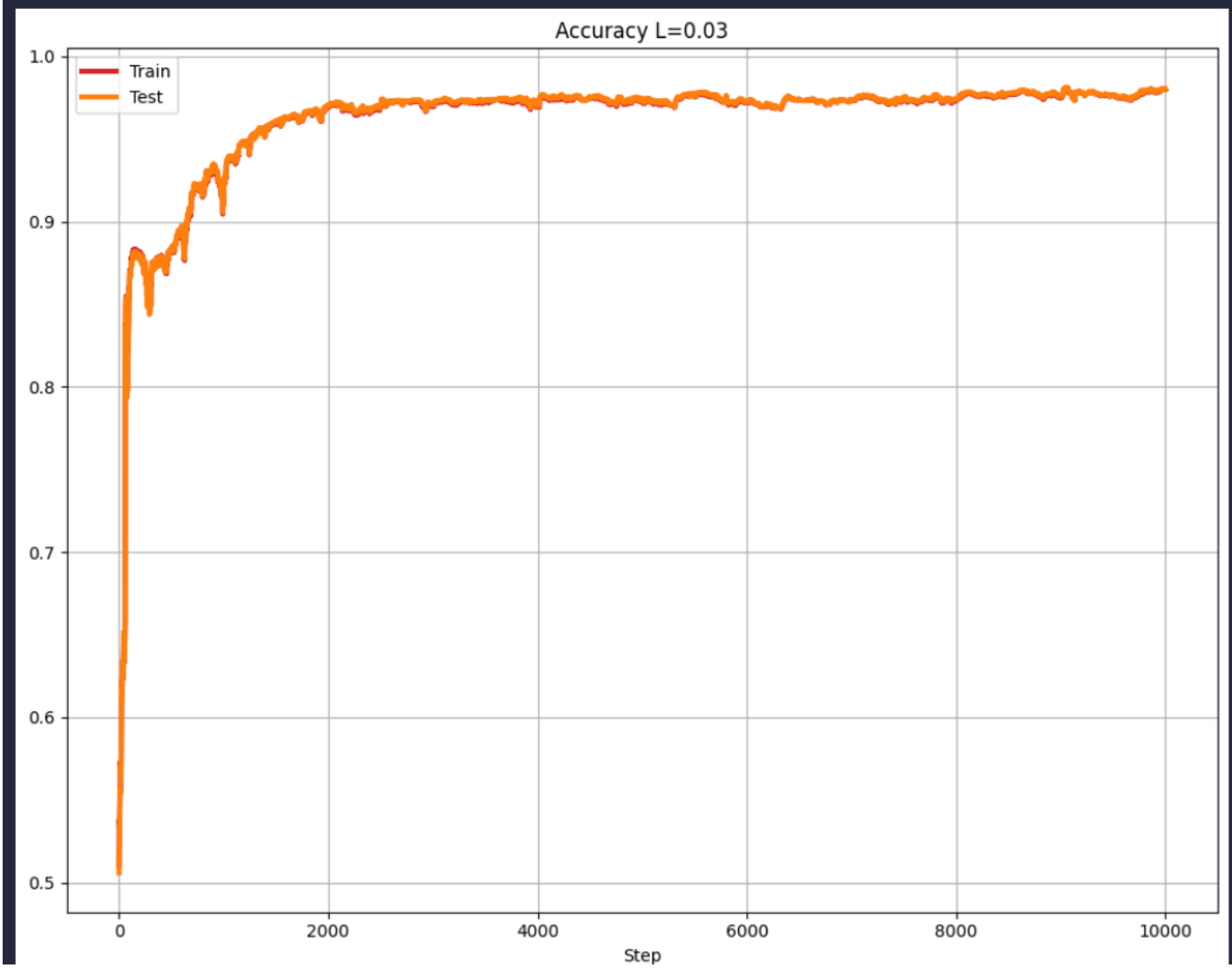
Hasta ahí, tenemos lo que se conoce como 'Forward Propagation', pero todavía no termina. Cuando llegamos al final, se llama a la función de 'Backward Propagation', que lo que hace es ajustar los pesos y sesgos previamente mencionados (recordemos que empezaron con valores aleatorios) para minimizar el error de la red. Empezando desde la última capa (capa de salida), la red distribuye el error hacia capas anteriores para identificar cómo cada peso contribuyó al error, y utilizando el descenso de gradiente, la retropropagación modifica los pesos y sesgos para reducir el error en cada paso. Esto se hace calculando gradientes (derivadas parciales) respecto al error y ajustando los parámetros en la dirección que lo minimiza.

Armado de la Red

Procedimos a armar nuestra Red Neuronal, separando por primera y única vez nuestros datos en entrenamiento y testeo. (Como los nombres bien lo indican, los datos de entrenamiento indican a la red como hacer las cosas, y los de testeo, valga la redundancia, testean el aprendizaje de la red)

Una vez que todo estaba listo, entrenamos y probamos la red. Se le da un determinado paso (step) y una cantidad de repeticiones (epochs). El paso indica que tan grande va a ser el avance por iteración, y los epochs las iteraciones.

```
TRAIN ACCURACY:  0.9799254588089091
TEST ACCURACY:  0.9803167947831957
Epochs: 10000
Paso: 0.03
```



Después de probar con distintos L y epochs, concluimos en que los mejores resultados venían con un paso de 0.03.

Se puede ver el proceso completo en el archivo '.ipynb' del repositorio, así como el Data Set completo y el script de funciones utilizado para el tratamiento de la red y gráficos. Además, al final de Jupyter hay un apartado de código para probar con nuevas muestras. Se recomienda copiar y pegar del CSV "card_transdata.csv" hasta la anteúltima columna (sin incluir la última, que sería la de fraude) y pegarla en la función "red_predict(acá iría lo copiado)".

¡Muchas gracias!