Predicción de fluctuaciones y pánico en criptodivisas

TFM - VI MASTER DATA SCIENCE KSCHOOL

Juan Roncero Philipp y Gonzalo Aguilar Messa

Introducción

El Trabajo Fin de Máster (TFM) supone, como cabría esperar, una gran oportunidad para demostrar lo aprendido durante el máster y nuestra capacidad para ponerlo en práctica. Además de ello, ofrece la posibilidad de aplicarlo a un tema que nos parezca digno de estudio y que nos aporte la motivación requerida para afrontarlo como un hobby y no como una carga. Ambos venimos del mundo técnico, de roles que pasaron por el desarrollo software y han evolucionado hacia la arquitectura de software, y compartimos la pasión por dar sentido a los datos y encontrar explicación o justificación analítica a nuestras teorías a partir de ellos.

Uno de esos fenómenos que nos intrigaban cuando comenzamos el máster era el de las criptomonedas y la fluctuación salvaje que tienen. Si hay un mercado volátil actualmente, es el de las criptomonedas, y dicha volatilidad creemos que no puede ser meramente aleatoria, sino que de algún modo ha de poder predecirse en base a tendencias y comportamientos cíclicos, al menos lo suficiente para poder anticiparnos a ciertos movimientos bruscos de subida o bajada.

Adquisición de datos

Para estudiar la fluctuación de las criptomonedas, hemos buscado los datos en los portales de trading y en los históricos de cotizaciones de los mismos.

Algunas páginas ofrecen la descarga de históricos de cotización de las criptomonedas en CSV, donde facilitan el valor de la cotización agrupado de forma diaria. Sin embargo, dada la alta fluctuación de una criptodivisa a lo largo de un día, entendíamos que esa granularidad no era lo suficientemente fina y que tampoco obteníamos información sobre el número de operaciones por intervalo y la cantidad total de capital que se movía con las transacciones, es decir, lo “caliente” que está el mercado.

Es por ello que decidimos “fabricarnos” nuestros propios datos, recurriendo para ello al uso de la API de uno de estos portales de trading y montar un demonio que fuese consultando las transacciones y guardándolas en una base de datos. El demonio se hizo con C# .NET y se guardó la información en un SQL Server.

El portal escogido fue BITSTAMP, que es un exchange para criptodivisas asentado en la Unión Europea desde 2011. A diferencia de los otros exchanges, Bitstamp se centra exclusivamente en Bitcoin, Ethereum, Litecoin y Ripple. Tiene una API bien documentada y nos permitía obtener los datos que necesitábamos.

El proceso se ejecuta cada minuto y obtiene todas las ultimas 2000 transacciones de las 4 diferentes criptomonedas, lo que supone una cantidad superior al número de transacciones que se han podido realizar en ese intervalo. Para evitar almacenar en nuestra base de datos transacciones repetidas, se comprueba el campo identificador de transacción TID para diferenciar las nuevas transacciones que son las que se van guardando en la base de datos.

Cada registro de la base de datos que se va almacenando contiene los siguientes campos de interés:

- AMOUNT: Cantidad comprada o vendida de la criptomoneda (float de unidades de la criptomoneda)
- FECHA: Fecha y hora de la operación (en formato YYYY-MM-DD HH:mm:ss.mmm)
- TIMESPAN: La marca de tiempo a partir de la cuál se obtiene el campo FECHA
- PRICE: Precio de la moneda al que se realiza la transacción (en dólares)
- TID: El identificador único de la transacción que provee el portal
- TIPOTRANSACCION: Valor booleano que indica que es una compra (1) o una venta (0)
- ID_MONEDA: Valor numérico que indica de qué moneda se trata (1-Bitcoin, 2-Ethereum, 5-Ripple, 6-Litecoin). Este valor se lo asignamos nosotros en base a las peticiones en la API.

AMOUNT	FECHA	TIMESPAN	PRICE	TID	TIPOTRANSACCION	ID_MONEDA
0,015416	2018-05-29 22:25:02.000	1527625502	7495.270000	67268677	0	1
0,871195	2018-05-29 22:25:00.000	1527625500	7487.000000	67268676	1	1
0,06	2018-05-29 22:24:58.000	1527625498	7487.000000	67268674	1	1
0,09261048	2018-05-29 22:24:57.000	1527625497	7495.120000	67268672	0	1
0,25489124	2018-05-29 22:24:57.000	1527625497	7495.120000	67268673	1	1
0,10336052	2018-05-29 22:24:54.000	1527625494	7495.120000	67268670	0	1
0,00189251	2018-05-29 22:24:53.000	1527625493	7491.950000	67268669	0	1
1	2018-05-29 22:24:52.000	1527625492	568.240000	67268668	1	2
2	2018-05-29 22:24:48.000	1527625488	568.230000	67268667	1	2
0,04	2018-05-29 22:24:47.000	1527625487	7487.220000	67268666	1	1
0,03785028	2018-05-29 22:24:46.000	1527625486	7492.930000	67268665	0	1
0,13397213	2018-05-29 22:24:34.000	1527625474	7487.200000	67268663	1	1
0,00585447	2018-05-29 22:24:34.000	1527625474	7491.880000	67268664	0	1
0,07755524	2018-05-29 22:24:25.000	1527625465	7492.930000	67268661	0	1

El demonio de recogida de datos exhaustivos (en su versión definitiva) entró en funcionamiento el 18 de febrero de 2018, y en 4 meses había guardado casi 6 millones de transacciones desglosándose por moneda en:

- 3.236.152 Bitcoin
- 1.067.454 Etherum
- 588.811 Litcoin
- 1.073.592 Ripple

Por otro lado, también queríamos estudiar la relación de las noticias relativas a las criptomonedas sobre la fluctuación. Para ello, se hizo otro demonio que utilizase la API de Twitter para descargar cada 15 minutos los tweets escritos en una serie de cuentas escogidas previamente, entre las cuales están las principales cuentas oficiales de las criptomonedas. Otro posible enfoque habría sido hacer búsquedas a través de Tags, pero se afrontará en una próxima iteración del proyecto, ya que supone en sí un proyecto poder determinar cuales de esos comentarios son reales y cuáles pertenecientes a bots, algo muy común en este mundo de criptos.

Los campos almacenados son el identificador de la cuenta, el texto del tweet y la fecha.

ID_CUENTA	TWEET	FECHA
1	More on our pilot with @SAMA_GOV and its potential imp...	2018-02-15 20:01:25.137
1	RT @OANDAbusiness: 2nd article in our miniseries with ...	2018-02-15 20:01:25.243
1	We´re excited about our work with @WesternUnion towa...	2018-02-15 20:01:25.243
1	RT @diliprao: A watershed event: @Ripple and @SAMA...	2018-02-15 20:01:25.243
1	.@bgarlinghouse spoke with @emilychangtv today about...	2018-02-15 20:01:25.243
1	#Blockchain is the game-changer every payments comp...	2018-02-15 20:01:25.243

Tendremos a su vez otra tabla para poder identificar las cuentas, ya que se trata de una base de datos relacional.

Para dar sentido a la información almacenada, se realizó un script en R que actualiza cada tweet con una variable adicional con el sentimiento obtenido a través de la API de Google Cloud que nos devuelve un valor entre -10 y 10, teniendo como negativo absoluto -10 y positivo absoluto 10 y así poder valorar el sentimiento de cada tweet guardado.

También consideramos relevante estudiar la influencia de lo que definimos como "ballenas", es decir, aquellas carteras de gran volumen de movimiento. Esta idea surgió a través de la lectura de una noticia que indicaba que allá por el 15 de diciembre se realizó una compra de una entidad privada de la criptomoneda Ripple por valor de 612 millones de USD, lo que constituía una cantidad de 900 millones de XRP a la cotización de aquel momento.

Averiguamos la existencia de una plataforma oficial de Ripple (https://xrpcharts.ripple.com) donde se podían verificar las transacciones de una cuenta. Tras una intensa búsqueda en internet, llegamos a la cartera en cuestión: *rGcyM8aFJwLEFajEeCJwtQJsCTYt9qXKsu* ya que al ser una moneda basada en tecnología blockchain, todas las transacciones son públicas, pues así es cómo puede funcionar este entorno distribuido.

Nos dimos cuenta de que llamando a la web antes indicada, concatenando una cartera, podríamos acceder a las operaciones de dicha cartera en su historia. Es anónima, pero sus transacciones son públicas. A Modo de ejemplo aquí se puede las transacciones comentadas:

https://xrpcharts.ripple.com/#/graph/rGcyM8aFJwLEFajEeCJwtQJsCTYt9qXKsu

La compra de 900 mill. XRP se realizó el 2017-12-15 a las 18:36 y la venta de 890 mill. de ripples el 2018-01-15 a las 20:20 hora peninsular. Compró a 0.50 USD y vendió a 1.80 logrando como se ve una altísima rentabilidad.

Nuestra conclusión fue que, bien fuera porque usaban software, conocimiento o información privilegiada, este tipo de operaciones se hacían con mucha seguridad y sería óptimo incluir esta información como variable de entrada, pues además estas operaciones de tanto volumen entendemos que tienen que provocar sacudidas en el mercado.

Se decidió tejer una tela de araña para ir relacionando unas ballenas con otras. Haciendo uso de la web anteriormente indicada, se puede realizar un seguimiento de las operaciones de la cartera que metamos como input. Con cada cartera obtenemos todas sus operaciones, y las carteras destino. Tal como se ha descrito, se utilizó la cartera inicial que llamó nuestra atención sobre este tema, y a partir de ella se sacaron las transacciones y carteras destino. Todas aquellas transacciones de más de 1 millón de XRP son guardadas, además de sus carteras destino. En cada iteración del script tiene nuevas carteras a estudiar, y así se va ampliando la colección de ballenas y, por ende, de transacciones.

El script realiza los siguientes pasos:

- Se conecta a cada cartera, abriendo un navegador con la cartera en la querystring.
- Se hace scrapping con expresiones regulares sobre el HTML y se obtienen la información descrita.
- Se guardan las transacciones y las carteras que no existieran ya en la base de datos.

Todos los procesos indicados almacenan sus datos en un servidor SQL Server dedicado y montado ad-hoc para el proyecto, instalado en una máquina virtual Windows Server 2012 donde también están corriendo los procesos de descarga descritos. En el repo se adjuntan, a modo ilustrativo, 2 de los ficheros C# más significativos para el proyecto, el de recogida de ballenas y de transacciones de criptodivisas (CargaDatos.cs) y el de recogida de tweets (cargaTweets.cs), así como el script en R para atacar la API de Google y evaluar el sentimiento de los tweets (ProcesamientoSentimientoTweets.R). El objetivo de compartir ese código es meramente ilustrativo y no con la intención de replicar todo el proceso, pues sería algo probablemente de poco interés en el ámbito puro del data science, y los datos están actualmente expuestos en un servidor nuestro.

Metodología de estudio de los datos y composición del dataset

A lo largo de los diferentes notebooks de análisis del proyecto, será necesario el uso de algunas librerías que no vienen inicialmente en Anaconda (que es el entorno recomendado para ejecutar todos estos notebooks implementados para Python 3.6), y que requerirán que abramos la consola de Anaconda y ejecutemos: *pip install nombrepaquete*

Hay paquetes como pandas_datareader que se instalan fácilmente, pero por ejemplo la instalación de XGBoost es un quebradero de cabeza en entornos Windows. Si fuese el caso, recomendamos consultar la siguiente referencia: https://medium.com/@rakshithvasudev/how-i-installed-xgboost-after-a-lot-of-hassels-on-my-windows-machine-c53e972e801e pues a nosotros nos sacó del contratiempo.

Una vez puestos manos a la obra, el primer notebook que tendremos que considerar es el *ANALISIS_V1.ipynb* que será el que nos introducirá en el problema objeto de estudio y nos dará unas primeras nociones del tratamiento y procesado de los datos.

Aunque los datos se han ido almacenando en una base de datos relacional de tipo SQL Server, para trabajar con ellos se han extraído a diversos ficharos CSV para su procesamiento. Y los notebooks acceden en línea a ellos y los cargan en dataframes sin que sea necesario descargar los ficheros previa.

La manipulación de los dataframes es muy parecida tanto para transacciones como ballenas y tweets. Inicialmente se convierten las fechas a tipo datetime, se establecen como índice y se ordenan. Acto seguido, se realizan diferentes agrupamientos estableciendo para ello en minutos la granularidad del agrupamiento.

Es precisamente esa granularidad de agrupamiento la que nos permite jugar con nuestros datos y así escoger si agruparlos en intervalos de 5 minutos para ver la rápida volatilidad del mercado o agruparlos cada hora o incluso más. A nosotros nos ha parecido que el estudio ha de ser lo más fino posible, pero se ha intentado que todo el código esté estructurado en diferentes funciones completamente parametrizadas para que cualquiera pueda hacer el estudio como mejor considere y sacar sus propias conclusiones. Una posible idea sería embuclar todas las llamadas iterando con diferentes rangos de tiempo para así poder montar un nuevo dataframe con esta nueva variable, pero por tiempo no hemos podido abordarlo.

Al hacer el agrupado, se está generando información que nos parece muy significativa, como es la cotización de apertura y cierre en el intervalo, así como la máxima y mínima también en el intervalo. A partir de esos valores, podemos obtener nuevas features con los porcentajes de variación en el intervalo (que podrán ser positivos o negativos) y unos porcentajes de variación máxima (que siempre serán positivos), y que nos sirven para ver lo bruscos que han sido los movimientos en cada uno de los intervalos de estudio.

Además de ellos, tendremos la posibilidad de generar nuevas features sobre las tendencias del mercado, que nosotros hemos querido entender como “tensión” acumulada que tiene el mercado, y que habíamos interpretado como posibles preludios de los cambios de tendencia. No hay que perder de vista que, a pesar de ser volátil, el mercado de las criptodivisas tiende a buscar un equilibrio como todos los demás.

Recomendamos echar un vistazo al mencionado *ANALISIS_V1.ipynb* para hacerse una idea del punto de partida del problema.

Una vez familiarizados con los datos, es cuando ya puede uno zambullirse en el notebook que trata de desarrollar toda nuestra idea con el proyecto, y que es el que intenta estudiar como un problema de regresión el fenómeno de la volatilidad en las criptomonedas. Ese notebook será el *REGRESION.ipynb*

Decidimos atacar el problema como una regresión porque queríamos ver si existía manera alguna de predecir en mayor o menor medida la fluctuación del mercado y el porcentaje de subida o bajada de la cotización. Apoyándonos en la librería sklearn, hemos implementado regresores lineales, ridge, svm (estos

no pudimos terminar de ejecutarlos debido al alto coste que tienen) y otros basados en algoritmos típicamente empleados para clasificación pero que tienen su versión para problemas de regresión.

Se realizó una agrupación de datos cada 5 minutos y se incluyeron 5 intervalos anteriores para darle al algoritmo suficiente información sobre la tendencia como para poder hacer una regresión fiable. Del mismo modo, para el horizonte a predecir, se decidió que intentaríamos predecir el futuro a 3 intervalos vista. Esto es, si agrupamos cada 5 minutos, pues intentar predecir lo que sucedería a los 15 minutos. La idea de hacerlo así es precisamente con la intención de poder hacer de este estudio un proyecto que entre en producción y no se quedase en un análisis teórico sin posibilidad de implementación. Ejecutando este algoritmo en vivo, gozaríamos de margen de tiempo suficiente como para poder operar vía API contra el portal de trading del que obtenemos los datos, y por tanto poder establecer umbrales de operación para planificar operaciones automáticas.

Se probó a hacer una selección de variables con las variables principales del dataset (previo a la inclusión de los 5 intervalos anteriores) mediante diversos métodos, para comprobar si podíamos eliminar alguna de nuestras features, pero no todos los métodos coincidieron en las mismas features a eliminar. Además, querían indicarnos que precisamente la "tensión" que habíamos observado como importante no lo era en realidad. Finalmente, no se desestimó ninguna variable porque las primeras pruebas eliminando features no daban mejores resultados.

Como se puede apreciar en el notebook, fueron precisamente el regresor lineal y el ridge los que nos ofrecieron unos mejores resultados y unas mejores métricas en comparación con los demás.

Como no tuvimos bastante hasta aquí, y nos había picado la curiosidad con hacer más y más análisis, pues nos lanzamos a atacar el problema como si fuese un clasificador, intentando por tanto predecir si la cotización iba a subir o bajar. Dicho análisis se encuentra en el notebook *CLASIFICACION.ipynb*

Tras enfocar todo el problema como una regresión, nos resultó difícil replantearlo como una clasificación. De hecho, a pesar del tiempo invertido, los resultados no han sido tan satisfactorios como con la regresión. Nos hemos encontrado con unos resultados que dejan entrever que hemos tenido overfitting, y aunque hemos intentado solucionarlo mediante la reducción de features, nos hemos encontrado con que entonces la tasa de acierto se desplomaba. Es posible que al intentar meter un backlog de resultados anteriores estemos empujando al problema a memorizar en lugar de aprender, o que no hayamos dado con la tecla a la hora de parametrizar correctamente los algoritmos y que por ello no hayamos obtenido los resultados deseados por falta de tiempo. Una idea implementada en algunos algoritmos y que provoca ese overfitting es la de probar con las variantes de sus principales parámetros en bucle, quedándonos siempre con el mejor resultado. Somos conscientes de este erro, pero nuestra idea era plasmar como se podría automatizar la búsqueda de los mejores coeficientes para los parámetros posibles de cada algoritmo. Sin duda, este estudio formará parte de esa segunda iteración del problema que comenzaremos ya fuera del alcance del TFM.

Por último, se intentó validar el algoritmo LSTM y así probar las redes neuronales. Se puede consultar en el notebook *LSTM - MODELO 4 (TRANSACCIONES).ipynb* que está basado en *LSTM – MODELO_LSTM.ipynb* que fue un notebook encontrado en internet, el cual copiamos en uno nuestro.

Los ejemplos encontrados sobre el mismo producen claramente memorización de los datos, ya que se han aplicado con los datos públicos de internet, es decir, como input sólo tienen el precio y poco más. Nosotros al tener la posibilidad de agrupar los datos en periodos más pequeños que un día (los datos públicos siempre son diarios) además de nuevas features, vimos una posibilidad de poder afrontarlo. No hemos conseguido aplicarlo a nuestros datos, pero sí nos ha servido para conocer un poco más en profundidad las redes neuronales. Sin duda somos conscientes del camino que necesitamos recorrer para poder afrontar Deep Learning, y atacaremos el problema en esa siguiente iteración que tenemos en mente.

Resumen del resultado

El proyecto ha sido un desafío desde el primer momento y sin duda es un proyecto vivo, es decir, que no se termina con la entrega del TFM. Nos planteamos un problema que entendíamos que era complejo y ambicioso, más si cabe dada nuestra inexperiencia, y que además tenía el componente añadido de las series temporales, que nos han supuesto un quebradero de cabeza desde el primer momento.

El proceso de obtención de los datos, así como el estudio de los mismos previo a pasarle cualquier método de predicción, ha sido de lo más entretenido. Hemos podido constatar como el trabajo de un data scientist consiste en una gran parte del tiempo en obtener y limpiar datos, así como estudiarlos para determinar las features y configurar el dataset que luego se utilizará con los algoritmos. Los datos de transacciones recolectados nos han permitido estudiar el problema como quisimos, aunque no pudimos obtener suficiente histórico de ballenas y tweets para que fuesen relevantes en los resultados del estudio.

Por otro lado, hemos obtenido interesantísimos resultados con regresores, que entendemos que es el ataque más idóneo del proyecto, y nos hemos quedado con la miel en los labios con los clasificadores, ya que no hemos conseguido los resultados esperados. Además, no logramos emplear satisfactoriamente una red neuronal en nuestro problema.

Tras estos primeros resultados, comenzamos ya una nueva iteración intentando afinar nuestros regresores y ampliando nuestros conocimientos sobre clasificadores y redes neuronales. Es por ello que el proyecto seguirá vivo, ya que seguimos recolectando transacciones, ballenas y tweets. ¿Hasta cuándo? Pues es difícil de determinar, pues queremos que esto no sea más que el principio del periplo de continuo aprendizaje y reciclaje que queremos tener en el mundo del data science.

Gracias por dedicar tiempo a revisar nuestro proyecto y agradecemos cualquier comentario o sugerencia que se nos pueda hacer llegar.