TRADING ALGORÍTMICO CON REDES NEURONALES

Mi robot tiene cerebro

Estamos en 2019, la tecnología nos acompaña desde que nos levantamos hasta que nos acostamos, incluso mientras dormimos. Es impensable para una gran mayoría levantarse de la cama cuando nuestro smartphone nos despierta sin echar un vistazo a los mensajes que han tenido lugar en nuestro descanso de redes sociales, correos electrónicos, etc. Mientras desayunamos, viajamos en transporte público a nuestros trabajos, vamos conectados a servidores de correos, al servidor de la empresa, rellenando nuestra agenda… Y así durante todo el día. Incluso cuando viajamos o estamos de vacaciones, nos sigue siendo necesario: una urgencia laboral, un cliente descontento, un proveedor que nos ha fallado, todo resolviéndolo desde cualquiera de nuestros dispositivos electrónicos con conectividad. ¿Impensable, indispensable o las dos cosas a la vez?

Como ejemplo o paradigma tomemos a un escritor. Antes de 1829, nacimiento de la máquina de escribir, los autores escribían a mano (dejemos a un lado debates de pergamino, papel, pluma, bolígrafo). A raíz de su patente en el citado año su expansión y dependencia de uso se hizo exponencial. Los escritores fueron haciendo uso de ella a pesar de la resistencia de algunos muchos que defendían que preferían a mano en vez de beneficiarse de una herramienta de trabajo que por el simple hecho de usarla no dejaba de ser escritura “manual”. Lo mismo ha pasado con el uso de editores de texto y ordenadores.

Encontraremos debates en foros y blogs sobre el trading discrecional frente al automático. Argumentan la preferencia de hacerlo manualmente en vez de beneficiarse de una herramienta de trabajo que podría mejorar la eficacia de su negocio. Desconocer cómo incorporar y usar esta tecnología no la invalida ni tampoco la hace inferior.

Al comenzar en este negocio lo primero que se hace es conocer utilidades, indicadores, plataformas que son comunes para todos. A medida que se va desarrollando

Esteban Pérez

Esteban Pérez, ingeniero en Telecomunicaciones e inversor en bolsa con una experiencia de más de 20 años. Actualmente es analista técnico independiente, trader intradiario a tiempo completo, programador de sus propios sistemas algorítmicos y formador en el trading especializado en Price Action y SoR calculados por lo que es muy conocido. Dirige el portal de formación www.forexdax.com y www.tradingalgoritmico.com.

El robot se debe conectar cada día para no influir en el sistema y por tanto, en la estadística.

la actividad, se van generando nuevas necesidades cada vez más profesionales.
Hoy, lo mismo que unos no conciben levantarse y no tener a mano su dispositivo electrónico para leer mensajes y/o correos electrónicos, otros no entenderían iniciar su jornada de trading sin poder conectarse a un servidor donde esté alojada la plataforma y las herramientas de trading incluido un sistema automático.

Trading algorítmico

El trading algorítmico, o trading basado en reglas y procesos, es una modalidad de operación en mercados financieros (trading) que se caracteriza por el uso de algoritmos, reglas y procedimientos automatizados en diferentes grados, para ejecutar operaciones de compra o venta de instrumentos financieros.
Dotarlo de una inteligencia artificial es una necesidad que nace en el momento que se observa un escenario desfavorable. Si seguimos una regla básica del trading algorítmico que dice: El sistema se debe conectar siempre, veamos lo que veamos, leamos lo que leamos, creamos lo que creamos. Si está programado siguiendo una estrategia ganadora y se ha comprobado que la relación pérdidas/ganancias es buena, el número de aciertos es mayor que cuando no acierta, en períodos medios obteniendo buena fiabilidad y rentabilidad, entonces hay que ponerlo a funcionar siempre. Es decir, el robot se debe conectar cada día para no influir en el sistema y por tanto, en la estadística.
Sin embargo todo sistema se enfrenta a escenarios o momentos de mercado que no le son favorables y si existiese una forma de detectarlos a tiempo se podría evitar pérdidas, pago de comisiones innecesarias y aumentar su fiabilidad. Pero es difícil hacerlo desde las propias reglas algorítmicas ya que al variar algunas para adaptarlo a todo tipo de momento de mercados, discriminando los peores dejaría de operar también en algunos buenos. Lo que se termina buscando en que haya un punto óptimo tolerando ciertas situaciones negativas.
Ahí es donde la necesidad de tomar una decisión de cuándo operar, sin que sea a través de reglas programadas, nos lleva a utilizar una red neuronal.

Keras.

Permite crear prototipos de forma rápida y fácil ya que esa es la filosofía que siguieron para su diseño.. Librería de alto nivel.

Tensorflow.

Desarrollada por Google y que ha terminado liberando y poniéndola a disposición del público mediante código abierto ya que es la misma que utiliza en sus productos.

Scikit-learn.

Escrita en Python, utilizada por Booking o Spotify entre otras compañías.

Pytorch.

Es un framework de Python. Su principal característica es que utiliza grafos computacionales dinámicos.

Para comenzar a utilizarlas Keras o Tensorflow serán una buena elección.

Red neuronal artificial- Definición y ejemplos

Podríamos definir el concepto de una red neuronal artificial como el intento de conseguir dotar de pensamiento a una máquina valiéndose por sí misma para el aprendizaje. Los humanos recibimos una serie de entradas desde el exterior (inputs) a través de los ojos, tacto, olfato, oídos y son transmitidos a nuestro cerebro donde procesamos con más de cien mil millones de neuronas que establecen conexiones entre unas y otras a través de impulsos eléctricos (proceso llamado sinapsis) y obtenemos unos datos de salida (pensamiento).
En una red de neuronas artificiales (RNA), recibiremos los datos de entrada a través de una capa (nodos de entrada), serán procesados en otra capa/s (nodos ocultos) y los datos finales del pensamiento será a través de la capa de salida (nodos de salida). Los nodos ocultos son fundamentales pues es donde se realiza el proceso de aprendizaje tomando la decisión más adecuada u óptima. Cada

neurona está conectada con otra a través de un enlace. El valor de salida de una neurona es multiplicada por un peso al llegar a la siguiente. El estado de ese peso pueda inhibir o activar a las siguientes neuronas (sopesa la decisión). Existe una función de activación que impone un límite que debe ser sobrepasado antes de llegar a las siguiente neuronas. Por eso estos sistemas se pueden entrenar, en esa fase aprenden por sí mismos en vez de ser algoritmos con reglas programadas que se ejecutan sin fase de pensamiento. El objetivo es que resuelvan problemas, tomen decisiones acertadas sin la intervención humana.

Dicho de otra manera, se consigue mediante un software dar entrada a unos datos, que serán procesados y el resultado será un dato de salida que permita tomar una decisión. Este software debe ser entrenado primeramente para que "aprenda" (machine learning).

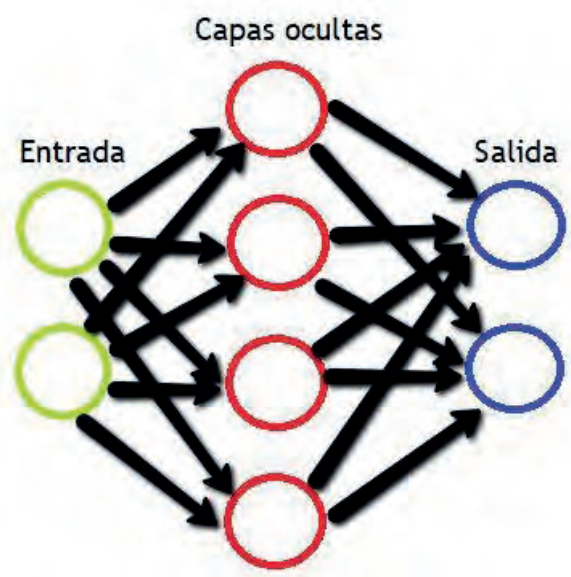
Por ejemplo, un coche tele-dirigido, eliminando el mando a distancia, instalando sensores, un pequeño ordenador con un sistema algorítmico y una red neuronal. En el algoritmo se establece que según reciba señal de un sensor que avisa de la presencia de un obstáculo tendrá que cambiar de dirección. A continuación se entrenará a la red enfrentándola de forma real en recorridos donde irá tomando decisiones hasta que el número de errores que comete sea igual a cero consiguiendo a partir de ese instante el 100% de efectividad. Ya no necesitará de más aprendizaje ni de optimizaciones futuras (en este ejemplo no valoraremos el desgaste de los mecanismos, neumáticos, etc. que volverá más imprecisa la conducción).

Otro ejemplo es el mítico juego de "Piedra, papel, tijeras" donde se entrena una red que aprende al cabo de un número de jugadas donde de forma aleatoria el jugador número 1 dice una cifra (o=piedra, 1=papel, 2=tijeras) y la red tiene que ir diciendo un número y al obtener el resultado sopesará si con la respuesta gana, pierde o empata. Aproximadamente, entre las jugadas 100 y 170 habrá aprendido a obtener el 100% de aciertos .

Aplicado al trading no es lo mimo

Su aplicación al trading no conseguirá un 100% de efectividad debido a diversas cuestiones como que el comportamiento de los mercados no obedece a unas leyes o reglas exactas como en los ejemplos anteriores. En el ejemplo del coche autónomo un sensor siempre que dé una señal de obstáculo será porque lo habrá, excepto en caso de avería o que alguien lo tape intencionadamente. Pero en los mercados, las noticias macro económicas o geopolíticas, los resultados empresariales, las correlaciones directas o inversas entre subyacentes entre otros

T1 Red neuronal artificial



Red neuronal artificial formada por un grupo interconectado de nodos de forma parecida a cómo lo están en el cerebro humano . Cada círculo representación una neurona artificial, color verde capa de entrada, color rojo capa/s intermedia/s, color azul capa de salida. En la zona de neuronas ocultas es donde se produce el aprendizaje.

Fuente: tradingalgoritmico.com

hechos hará que a pesar de un buen análisis técnico que ofrezca un escenario muy favorable de gran probabilidad, podrá ocurrir que el precio se gire y mueva de forma diferente a la esperada.

La buena noticia es que si el sistema es bueno, la estrategia es buena y el procedimiento por el que se dota de "inteligencia" a nuestro robot está bien hecho, la eficacia rozará cotas altas de eficacia.

Cerebro: Análisis y decisión sustituyendo al trader

El uso de una red neuronal artificial con machine learning se convierte en necesidad cuando se detectan y conocen aquellos momentos de mercados que podrían dar un escenario muy probable o probable que ocasionen pérdidas.

El concepto de un sistema infalible en la práctica no es posible. Porque aunque se den las condiciones matemáticas adecuadas si ignoramos el momento en que un activo se encuentra hará que tengamos pérdidas con mucha probabilidad.

Tomemos como ejemplo a un corredor. Se ha preparado para un determinado tipo de carreras de ciudad. Sabe dosificarse, cuándo debe acelerar atacando a los otros corredores y cuándo no. Se levanta un día y se prepara para ir a disputar una carrera . Llega a la pista y no encuentra sino a unos pocos atletas que están entrenado porque no

es día de carreras ya que es jueves y se disputan en fines de semana. No ganará porque no hay "volumen" de corredores y los pocos que hay al estar entrenando, su "volatilidad" será baja. Habrá perdido tiempo y esfuerzo y lo que es peor, desgaste de energía, acumulación de carga en sus músculos dos días antes de que produzca una carrera lo que podrá restarle posibilidades de competir bien. No miró nada más que lo que le interesaba en ese momento: tenía ganas de hacerlo, se sentía fuerte y necesitaba una victoria.

En el trading pasa algo similar. Sin volumen de negocio, volatilidad, momento tendencial fuerte es un escenario de baja probabilidad de éxito. El establecimiento de la regla de los tres escenarios: Muy probable, probable y poco probable.

Podremos encontrar navegando por Internet que la mayoría de los ejemplos buscan la predicción del futuro del precio. Es decir, saber o adivinar si mañana el precio va a estar alcista o bajista sabiendo de antemano dónde empezará y acabará.

Sin embargo podría ser un mal uso de la red neuronal. Predecir o adivinar no parece una buena estrategia de negocio. Y en realidad los resultados que se obtienen no suelen ser brillantes.

Discriminar escenarios según el momento de mercado

Un planteamiento diferente es tener bien identificado los escenarios favorables y desfavorables para nuestra estrategia. Encontrar la manera de discriminarlos puede ser una actividad de éxito.

Entrenar una red neuronal para que aprenda a decidir si hoy va a dejar al robot a actuar con un porcentaje de aciertos suficientemente alto para que no elimine momentos buenos y no exponga a la cuenta de trading en los malos es el objetivo que se propone.

Con estas premisas la red será entrenada del siguiente modo:

1. Período de 10 años, de los cuales 8 serán para entrenamiento y 2 para test de efectividad.
2. Decidir el escenario más probable: alcista, bajista, rango
3. Si es alcista permitirá abrir sólo largos mediante una variable de estado "true" en el algoritmo de compras. El algoritmo de ventas tendrá la variable en estado "false".
4. Si es bajista permitirá abrir sólo cortos mediante una variable de estado "true" en el algoritmo de ventas. El algoritmo de compras tendrá la variable en estado "false".
5. Si el escenario más probable es que se mueva en un lateral con un ancho menor al promedio diario mantener las dos variables en estado "false" lo que impedirá al robot abrir ningún tipo de posición.
6. Si el resultado final del día es que la operación acabó con éxito una variable recibirá el estado "1" de acierto. Si no acertó, recibirá "0" (por ejemplo calculó que el día sería alcista y la operación correcta era bajista o lateral, etc.)
7. El marco temporal elegido será el de 5 minutos debido a que la estrategia ha sido diseñada para huir del ruido que se genera en esta temporalidad ciertos momentos del día evitando falsas señales de entrada.

Se expondrá al 80% del período seleccionado y se dejará que vaya decidiendo qué escenario es el probable, a continuación los algoritmos de compras o de ventas se ejecutarán de forma independiente y la red neuronal recibirá el resultado del día a modo de 1=acierto, 0=error. A medida que vaya pasando el tiempo la estadística de aciertos empezará de forma exponencial a aumentar hasta alcanzar una cifra entre el 80-90%. Cuando finaliza, guardaremos el aprendizaje para su utilización diaria sin tener que repetir este proceso constantemente. Se expondrá al 20% restante para ver si mantiene la misma efectividad que en entrenamiento. Si lo hace habrá que efectuar procesos más complejos estadísticos como Montecarlo para finalmente enfrentarlo al mercado real en una cuenta papel trading que confirme su eficacia hasta ser validado para su uso comercial.

Sin volumen de negocio, volatilidad, momento tendencial fuerte es un escenario de baja probabilidad de éxito.

Usando Python y librerías Machine Learning

Hasta ahora no se ha mencionado el lenguaje de programación a utilizar, la plataforma. Es bien conocido el dato de que muchas plataformas traen incorporado un lenguaje de programación y permiten escribir estrategias automáticas, que es así como definen al trading algorítmico. Pero quizás debamos hacer una distinción importante:

Estrategias automáticas: Mediante un lenguaje de programación se podrá automatizar una estrategia para que se ejecute en el gráfico seleccionado. A medida que van transcurriendo barras o velas se irá reflejando en

el gráfico dónde toma una posición y dónde la cierra. La escritura de mensajes en una terminal donde se nos informe de eventos no suele estar contemplado.

Sistema automático independiente a la plataforma: La programación de estos sistemas se hacen en un lenguaje independiente a la plataforma del bróker. Mediante funciones API (Application Programming Interface) se conectará con ella y las órdenes serán colocadas o canceladas como si de forma manual se tratara. No sólo habrá programada una estrategia sino todo un sistema que mediante mensajes en una terminal informará de condiciones de mercado, escaneo de activos listos para ser operable, operaciones en curso, envío de emails con avisos, etc.

Dos potentes lenguajes están disponibles para ello: Python y R con librerías de trading, backtesting y el uso de Machine Learning.

Para el ejemplo elegiremos Python pero R es igual de bueno. En cuanto al uso de redes neuronales encontraremos cuatro librerías muy potentes: TensorFlow, Scikit-learn, Pytorch y Keras. Una breve descripción sobre ellas puede ver en el cuadro correspondiente.

Código de ejemplo con Keras

Se describirá a continuación el código de una estrategia que utiliza Keras para predicción del movimientos de precios de mini DAX utilizando para ello los datos de la serie diaria del futuro del mini DAX. El código se puede descargar desde aquí.

Primeramente se importan las librerías necesarias, Numpy es una poderosa librería científica, Pandas es usada para estructurar datos, en este caso, para coger los datos descargados desde Yahoo Finance del mini DAX desde el año 1996 hasta Abril del 2019 y poder tratarlos más tarde. Talib es una herramienta ampliamente utilizada por desarrolladores con más de 200 indicadores programados, muchos de ellos conocidos con MACD, RSI, media móviles, estocástico, Bollinger, ADX etc. Aquí será usada para coger alguno de ellos e implementar la estrategia a la hora de búsqueda de señales de compra o venta. A continuación otra función generará un número pseudo aleatorio para comenzar el entrenamiento.

El fichero "csv" generado en Yahoo Finance estará ubicado en c:\tmp y se prepara la data usando además una media móvil simple de 10 períodos y un RSI de 9 períodos.

Con la función dataset.dropna() se elimina de la data filas vacías si las hubiese.

Dos marcos de datos que almacenarán las entradas y las variables de salida (representado gráficamente en el esquema de una red neuronal de la figura 1). "X" almacenará las entidades de entrada. "Y" almacenará el valor a predecir: aumento del precio.

G1 Inicio de la fase de aprendizaje

```
Using TensorFlow backend.
WARNING:tensorflow:From C:\Users\Usuario\AppData\Local\Programs\Python\Python37\lib\site-packages\tensorflow\python\fra
ework\op_def_library.py:263: colocate_with (from tensorflow.python.framework.ops) is deprecated and will be removed in a
 future version.
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From C:\Users\Usuario\AppData\Local\Programs\Python\Python37\lib\site-packages\tensorflow\python\ops\
math_ops.py:3066: to_int32 (from tensorflow.python.ops.math_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
Epoch 1/100
2019-04-08 16:20:17.012725: I tensorflow/core/platform/cpu_feature_guard.cc:141] Your CPU supports instructions that thi
s TensorFlow binary was not compiled to use: AVX2
4707/4707 [==============================] - 2s 489us/step - loss: 0.2489 - acc: 0.5303
Epoch 2/100
4707/4707 [==============================] - 1s 292us/step - loss: 0.2481 - acc: 0.5356
Epoch 3/100
4707/4707 [==============================] - 2s 422us/step - loss: 0.2482 - acc: 0.5356
Epoch 4/100
4707/4707 [==============================] - 1s 179us/step - loss: 0.2482 - acc: 0.5356
Epoch 5/100
4707/4707 [==============================] - 1s 199us/step - loss: 0.2481 - acc: 0.5356
Epoch 6/100
4707/4707 [==============================] - 1s 178us/step - loss: 0.2481 - acc: 0.5356
Epoch 7/100
4707/4707 [==============================] - 1s 180us/step - loss: 0.2480 - acc: 0.5356
Epoch 8/100
4707/4707 [==============================] - 1s 169us/step - loss: 0.2479 - acc: 0.5356
Epoch 9/100
4707/4707 [==============================] - 1s 173us/step - loss: 0.2478 - acc: 0.5356
```

Observamos que cada vez que realiza una pasada (epoch o época) va ofreciendo un porcentaje de aciertos que comienza sobre el 53%

Fuente: Ejecución en Python con Keras por tradingalgoritmico.com

Dividiremos los datos de las variables de entrada y salida para crear la prueba y entrenar los datos. Para ello, se crea la variable "división" a la que se otorga el valor de "0.8" veces la longitud del conjunto de datos. Eso quiere decir que en el restante equivalente al 20% se comprobará la estrategia "pilotada" por la red neuronal.

Se divide la variable X en cuatro marcos separados: x_train, X_test, Y_train, Y,_test, parte esencial de cualquier algoritmo machine learning, los datos de entrenamientos serán utilizados por el modelo para llegar a otorgar los pesos.

Y luego importamos la librería Keras y definiremos su funcionamiento usando 128 neuronas en la capa central haciendo 100 pasadas por el conjunto total de datos.

Con esta última parte del código se realizará la predicción del precio de mañana, los valores se almacenarán en variables y serán mostrados en un gráfico a través de la función "plot".

El resultado que se obtiene tras las 100 pasadas es el de un aprendizaje que supera el 60% de las veces que acierta y la estrategia con red neuronal (color azul) gana al a la estrategia sin red neuronal (color verde):

Aunque la estrategia es muy simple y en realidad no es una buena como para ser usada en el mercado a tiempo real y con dinero, nos ayudará a comprender que si se dota de inteligencia artificial a un sistema automático de forma correcta y conveniente podría ayudarla a mejorar su fiabilidad y resultados en un plazo de tiempo sin tener que realizar continuos ajustes (parámetros optimizables).

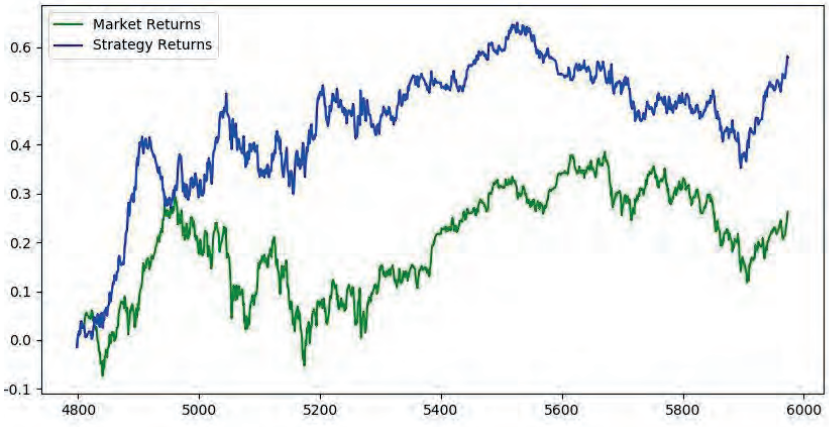
G2 Finalización de la fase de aprendizaje

```
Epoch 87/100
4690/4690 [==============================] - 1s 210us/step - loss: 0.2176 - acc: 0.6471
Epoch 88/100
4690/4690 [==============================] - 1s 272us/step - loss: 0.2170 - acc: 0.6531
Epoch 89/100
4690/4690 [==============================] - 1s 283us/step - loss: 0.2161 - acc: 0.6490
Epoch 90/100
4690/4690 [==============================] - 1s 180us/step - loss: 0.2161 - acc: 0.6544
Epoch 91/100
4690/4690 [==============================] - 1s 178us/step - loss: 0.2151 - acc: 0.6552
Epoch 92/100
4690/4690 [==============================] - 1s 175us/step - loss: 0.2146 - acc: 0.6614
Epoch 93/100
4690/4690 [==============================] - 1s 252us/step - loss: 0.2136 - acc: 0.6554
Epoch 94/100
4690/4690 [==============================] - 2s 401us/step - loss: 0.2129 - acc: 0.6588
Epoch 95/100
4690/4690 [==============================] - 2s 362us/step - loss: 0.2118 - acc: 0.6597
Epoch 96/100
4690/4690 [==============================] - 2s 331us/step - loss: 0.2117 - acc: 0.6618
Epoch 97/100
4690/4690 [==============================] - 1s 215us/step - loss: 0.2102 - acc: 0.6657
Epoch 98/100
4690/4690 [==============================] - 1s 172us/step - loss: 0.2094 - acc: 0.6699
Epoch 99/100
4690/4690 [==============================] - 1s 191us/step - loss: 0.2095 - acc: 0.6731
Epoch 100/100
4690/4690 [==============================] - 1s 171us/step - loss: 0.2089 - acc: 0.6672
```

Después observamos que al llegar a las 100 épocas termina con un nivel de aciertos del 67%

Fuente: Ejecución en Python con Keras por tradingalgoritmico.com

G3 Gráfico comparativo de retornos



La imagen muestra el resultado de la estrategia con machine learning frente a la misma antes de usarlo.

Fuente: Ejecución en Python con librería Keras y datos del futuro del mini DAX (1996/2019) por tradingalgoritmico.com

La dificultad estriba en saber hacer justo eso, en coger algo bueno que gana más que lo que pierde, que no necesita ayuda de optimizaciones constantes y que seleccione los momentos de mercado que le puedan ser favorables descartando los que son menos y que minimizando las situaciones negativas.

Aunque en el ejemplo anterior la estrategia intenta predecir el precio del día siguiente para tomar una posición no es lo más adecuado. Todo sistema tiene sus virtudes y su ineficiencia. Saber cuáles son sus puntos débiles ayudará a mejorarlos si hacemos uso de esta tecnología. No se podrá obtener un sistema que gane 100% de las veces al mercado operando con frecuencia ni ese debe ser el objetivo. Hay que aceptar las pérdidas como vienen pero siempre que estén perfectamente evaluadas y esperadas por la fiabilidad que arroja el conjunto. Conocer las herramientas de trading y hacer un uso correcto definen al trader.

Sistema Genius con red neuronal

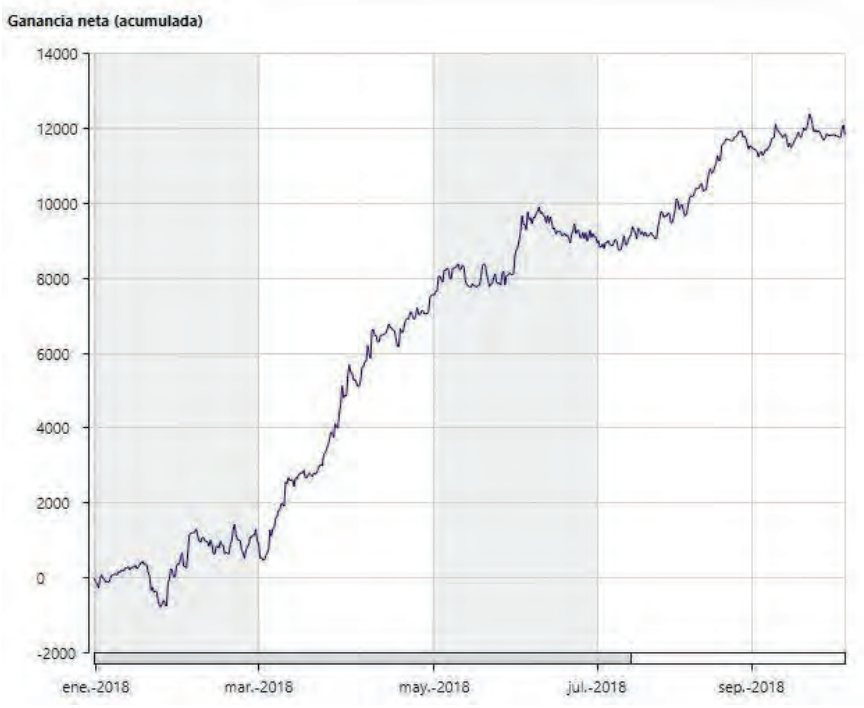
La implementación de un sistema de trading algorítmico que es consistente en el tiempo donde se muestra muy

fiable mediante una red neuronal que cumpla la premisa de distinguir, seleccionar y discriminar escenarios poco favorables puede generar dudas de si lo empeorará, mejorará o el resultado sea similar al que se está obteniendo como rendimiento del mercado.

Si en la fase inicial donde se realiza el organigrama de cómo se va a programar el sistema no se tiene en cuenta la posibilidad de incluir machine learning en un futuro puede ser una ardua tarea que o bien altere el funcionamiento o simplemente no se pueda hacer. Si se está pensando en crear un nuevo sistema sería el momento de evaluar la implementación con RNA o en el caso que se encuentre en la etapa de desarrollo primaria puede intentarse también hacer su uso.

El mes pasado fue publicado un artículo sobre el "curve fitting" y precisamente estudiábamos a la estrategia automática Genius viendo los resultados del año 2018 en un gráfico. Genius no fue concebido para ser implementado con posterioridad mediante esta técnica. Su filosofía es negociar cada día realizando "swing day trading" en el marco temporal de 5 minutos negociando el DAX (mini o DX) haciendo compras o ventas. La fiabilidad y comportamiento es excelente pero hay momentos de mercado de rango estrecho que generan señales de entrada al mercado tipo BUY o SELL que generarán pequeñas pérdidas. Sin embargo, esos escenarios desfavorables pueden ser evaluados antes de iniciar la jornada de negocio y con la decisión del usuario no conectar el sistema ese día. Pero eso quiere decir que va contra el principio de que sea lo más autónomo y automático posible con la mínima intervención humana. Incluirle un "cerebro artificial" que evalúe el mercado y

G4 Gráfico de ganancias netas acumuladas sin RNA



Podemos observar gráficamente la evolución del sistema Genius de t sin RNA (red neuronal artificial) a lo largo del año 2018 mediante la ejecución en Python y exportando los datos a una hoja de cálculo para su representación.

Fuente: tradingalgoritmico.com

G5 Gráfico de ganancias netas acumuladas con RNA

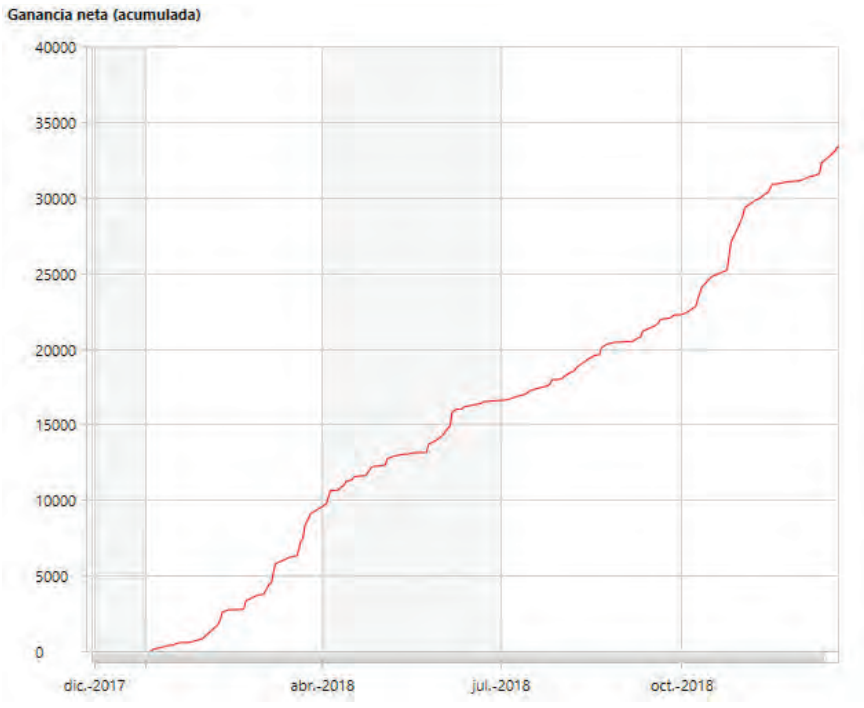


Gráfico que muestra el resultado del mismo sistema Genius con RNA mediante la ejecución en Python y exportando los datos a una hoja de cálculo para su representación.

Fuente: tradingalgoritmico.com

decida anular los algoritmos que provocan las entradas en un día determinado se traducirá en algunos cientos de horas para que el "alma del sistema" siga siendo la misma sin ser alterado en nada las reglas que componen la estrategia.
Una vez desarrollada esta fase y ser entrenada de forma conveniente, comprobado en el mercado en cuenta paper trading está disponible para la negociación real.
Debajo se podrá valorar dos gráficos, el primero muestra todas las operaciones que realiza en el año 2018 (roza las 800 en su totalidad) y el segundo con la red neurona incorporada que mejora los resultados.
Resulta obvio como desde el principio evita situaciones negativas. En la imagen 4 se observa una pérdida cercana a los 1.000 € por contrato del mini DAX en el mes de enero. Sin embargo, queda amortiguada tras ser "manipulado" su funcionamiento por la RNA no dejando que actúe en escenarios cuya probabilidad de ser negativos es suficientemente alta sabiendo reconocerlos.
Todo sistema por bueno que sea es susceptible de ser mejorado. La forma en que se decida y haga hará que el objetivo sea alcanzable o que el resultado sea contrario a la esperanza matemática que se espere.

Cómo comenzar

Publicaciones en formato libro las hay. En español Fernando Berzal ha publicado varios volúmenes con el título "Redes neuronales & Deep Learning". Victor Viera y Maritza Palacios poseen una publicación con ejemplos en Python muy prácticos: "Aprendizaje de máquinas. Redes neuronales". En inglés Ernert P. Chan nos ofrece uno específico para el trading: "Machine Trading". Estas son algunas referencias aunque la mayor parte del material estará en inglés. Lo mismo ocurre con el disponible en internet. Aunque hay webs como Quantopian.com que son un fondo de inversión pero que desde su plataforma web ofrece la posibilidad de programar en Python con material de aprendizaje en formato vídeo y texto incluyendo machine learning. Si el código pasa las pruebas podrá competir contra el de otros programadores y ser usado por Quantopian ganando dinero en premios y derechos de uso. El grado de dificultad es medio-alto y aunque anuncia el comenzar desde cero la realidad es que es mejor llegar ya con buenos conocimientos previos.

Conclusión

Hoy en día se sigue recurriendo al paradigma sobre la informática refiriéndose a la actividad del futuro cuando en realidad hace tiempo que no es así, lo ha sido en el pasado reciente y lo es en el presente. Puede que sea un error aprovechar una parte de los beneficios que aporta el software dedicado al mundo del trading y dejar de lado a otra parte por pereza al aprendizaje, temor a lo que no se conoce, amplificar errores por falta de formación, visión de mercado, estrategia ganadora o porque se pueda llegar a pensar que no está al alcance de uno mismo.
No se tiene que saber de programación, porque incluso teniendo esos conocimientos no hay una relación directa con tener los conocimientos y saber programar . Automatizar una estrategia puede llevar cientos o miles de horas dependiendo de su complejidad y eficiencia y eso lleva un coste económico cuya media puede alcanzar decenas de miles de euros o dólares. No se debe menospreciar la actividad ni embarcarse en un proyecto de este tipo sin haber valorado muy bien todos esos aspectos.
Sin embargo, el encontrar a disposición de cualquier particular o sociedad sistemas automáticos que incluyan machine learning producidos por un trader programador o empresas especialistas empieza a ser una tarea que poco a poco es menos ardua. No se puede generalizar con argumentos de que todos fallan a la larga o que son un engaño por el mero hecho de haber ocurrido en el pasado incluso a día de hoy. Igual que no todos los intermediarios a los que se compra y vende los contratos y/o acciones son unos estafadores y no se deja de elegir un bróker y negociar a diario por esa creencia tampoco se debería dejar de hacer uso del trading algorítmico.
Lo que hay que hacer es buscar en el mercado cosas que se adapten a nuestras circunstancias: poder de adquisición, tamaño de la cuenta de trading y de las posiciones, estrategias que sean bien explicadas y no se esté comprando una "caja negra" como decía Alexander Elder y que detrás haya alguien que respalde y de soporte. El "boca-a-boca" de usuarios reales aportan un plus a la hora de decidir. Como en todo compra sopesar los diferentes valores a la hora de decidirse.
Si se tiene buena experiencia en trading con éxito, conocimientos de programación e incluso ya se está utilizando el trading algorítmico hecho por uno mismo, dotarlo de una red neuronal podría ayudar a dar un salto cualitativo.
Se especula que más del 80% de los movimientos que vemos en cualquier producto financiero está actualmente generado por trading algorítmico y que más de las dos terceras partes llevan implícito redes neuronales (incluso redes neuronales colaborativas, una red entrena a la otra). No parece la mejor idea quedarse anclado al click del ratón para colocar la orden y quedarse mirando la pantalla.