MÁSTER EN DESARROLLO Y AUTOMATIZACIÓN DE ESTRATEGIAS DE TRADING

Con la finalidad de facilitar la asimilación de los contenidos del máster consideramos de su interés facilitarle los iconos propuestos para este fin:

Este icono se utilizará para realizar preguntas que ayuden a reflexionar sobre determinadas situaciones o para realizar preguntas que ayuden a introducir conceptos.

Este icono se utilizará para resumir aquellos conceptos clave que los participantes deben recordar por su importancia.

Este icono pretende alertar de conceptos que frecuentemente no son tenidos en cuenta por la gran masa inversora.

Este icono servirá para poner ejemplos que, de un modo más amigable, permitan reflejar casos prácticos.

MÓDULO III: ESTRATEGIA DE TENDENCIA IMPULSADA POR EVENTOS.

1. Descripción detallada de la estrategia a implementar.
    1.1. Modelo.
    1.2. Filtro.
    1.3. Reglas de entradas.
    1.4. Reglas de salida.
2. Implementación de la estrategia mediante NinjaTrader.
3. Optimización de la estrategia.
4. Backtest de la estrategia.

1. Descripción detallada de la estrategia a implementar.

En esta segunda práctica se va a implementar una estrategia se pueda usar cuando se produzcan roturas de volatilidad.

1.5. Modelo.

El modelo pretende aprovechar las roturas de volatilidad que se producen en el mercado cuando los precios llevan unos periodos en tendencia, posteriormente entran en un rango y después lo rompen a favor de la tendencia previa.

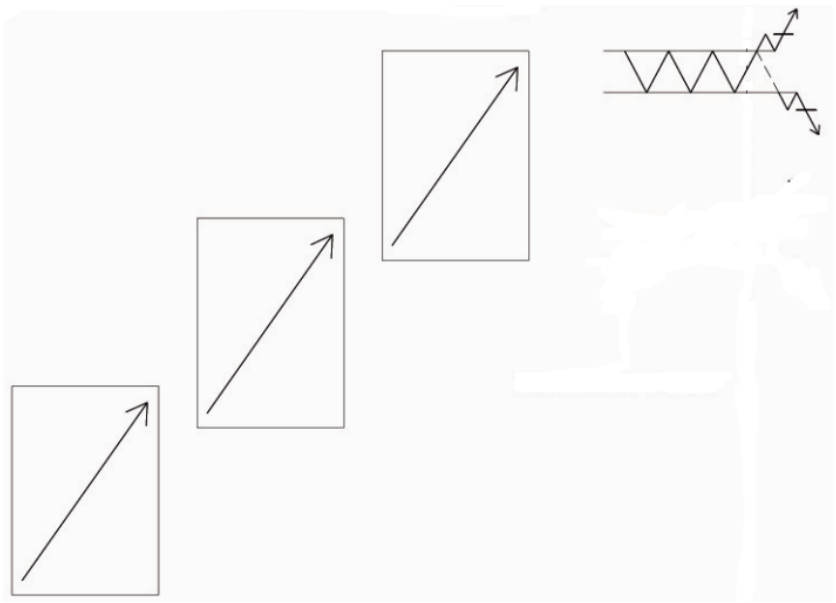


Figura 1: Sistema.

1.6. Filtro.

Puesto que se requiere que se produzcan aumentos de la volatilidad con cierta magnitud, se requieren valores que sean volátiles, con un gran ATR en relación con su precio y que posean gran liquidez. En este caso se han seleccionado un grupo de empresas que cumplen dicho requisitos lo que le otorga al sistema gran fiabilidad y profit factor. Como ocurre con todas las estrategias, si se eligen los valores equivocados la estrategia no funcionará correctamente.

Por su propia naturaleza, esta estrategia produce pocas entradas, puesto que los valores obtienen roturas de volatilidad cada cierto tiempo, por lo que ejecutándose de forma paralela a otras estrategias, este sistema aprovecharía esas condiciones del mercado que se producen en determinado días y que generan gran profit factor.

1.7. Reglas de entradas.

La entrada se producirá cuando el precio, tras llevar 3 o más periodos en tendencia, forme un rango en el siguiente periodo y en el posterior obtenga un movimiento mayor del 1,5*ATR de los 5 últimos periodos.

1.8. Reglas de salida.

La salida se producirá con un porcentaje del precio, en este caso se usara:

- Stop loss: 0,6% del precio.
- Target: 0,5% del precio.

Esta relación ha resultado ser la más óptima, pues en este caso se necesita que el ruido no haga salir las entradas por stop loss.

2. Implementación de la estrategia mediante NinjaTrader.

En este apartado se muestra el código final, tras la optimización, que ofrece los resultados que posteriormente se exponen.

En primer lugar, como en la práctica anterior, aparecen las sentencias internas que NinjaTrader genera de forma automática, en las cuales solo se puede modificar el nombre, que no puede coincidir con el de otra estrategia que esté programada previamente en el ordenador, y la descripción de la estrategia, que no afecta a la misma. Este código es siempre igual.

```
#region Using declarations
using System;
using System.ComponentModel;
using System.Diagnostics;
using System.Drawing;
using System.Drawing.Drawing2D;
using System.Xml.Serialization;
using NinjaTrader.Cbi;
using NinjaTrader.Data;
using NinjaTrader.Indicator;
using NinjaTrader.Gui.Chart;
using NinjaTrader.Strategy;
#endregion

// This namespace holds all strategies and is required. Do not
change it.
namespace NinjaTrader.Strategy
{
    /// <summary>
    /// Enter the description of your strategy here
    /// </summary>
    [Description("Enter the description of your strategy here")]
    public class TendenciaImpulsadaPorEventos : Strategy
```

Ejemplo 1: Declaraciones y sentencias que la platagoforma genera de manera automática.

En la segunda parte del código se encuentran las inicializaciones de las variables que se van a usar en la estrategia. Es importante distinguir entre

variables internas y externas. Las variables internas son las que se eliminan en cada vela, es decir, cada vez que se forma una vela se crean y se destruyen, por lo que no se pueden mantener de una vela a otra. Por su parte las externas son las que se pueden mantener, y modificar entre las distintas velas, a lo largo de todo el periodo en el que se ejecute. Las variables internas se pueden declarar e inicializar en la tercera parte del código, mientras que las variables externas hay que declararlas en la cuarta zona e inicializarlas en la segunda zona.

Las principal novedad que aparece con respecto a la práctica anterior es que se incluye Add(PeriodType.Minute, c); que sirve para, en la tercera parte del código, poder hacer referencia a un timeframe concreto. Si se añaden varios timeframes distintos, para elegirlos posteriormente en la tercera parte del código se hará referenciándolos en el orden que se ha escrito en esta segunda parte, es decir [1] hace referencia al primero, [2] hace referencia al segundo en escribirse, etc. Se puede cambiar Minute por Day o por otro intervalo. c hace referencia al número de minutos, días, etc.

Para establecer un target global para la estrategia se usa SetProfitTarget(CalculationMode.Percent, d); en el que Percent se puede cambiar por Price o por Tick. d Hace referencia a la cantidad, que si es en Percent se expresa en tanto por uno. Ejemplo: 10% -> 0.1.
Para establecer un StopLoss global para la estrategia se usa SetStopLoss(CalculationMode.Percent, 0.005); con el mismo formato que el anterior.

```
{
        #region Variables
        // Wizard generated variables
        // User defined variables (add any user defined variables
below)
            private int velas=0;
            private int c = 0;
            private int v = 0;
        #endregion

        /// <summary>
        /// This method is used to configure the strategy and is
called once before any strategy method is called.
        /// </summary>
        protected override void Initialize()
        {
                  SetProfitTarget(CalculationMode.Percent, 0.006);
                  SetStopLoss(CalculationMode.Percent, 0.005);
                  CalculateOnBarClose = true;
                  Add(PeriodType.Minute, 195);
                  Add(PeriodType.Day, 1);
        }

        /// <summary>
        /// Called on each bar update event (incoming tick)
        /// </summary>
```

Ejemplo 2: Inicialización de las variables y descripción de las mismas.

En la tercera parte se sitúan las órdenes propias de la estrategia, que se suele estructurar en condiciones, y acciones. En cada vela se evalúan una serie de condiciones y si alguna se cumple se ejecuta una acción.

Para establecer una variable en función de un timeframe distinto al global, lo primero es haber definido dicho timeframe en la segunda parte del código. Una vez definidos se hará referencia a cada timeframe en función del orden con el que se estableció en la segunda parte. Hay dos formas de hacer referencia a los timeframes dependiendo de la variable a la que se vaya a aplicar. Por un lado están las variables propias del precio (Close[b], Open[b],…), para estas variables se usará (Closes[a][b], Opens[a][b]), es decir, se añade una “s” al final del nombre, seguido de un número entre corchetes. El número entre corchetes hace referencia al timeframe elegido, y corresponde con el orden con el que se han declarado los timeframes en el segundo periodo. Es decir, [b] al primero, [b] al segundo, etc… El siguiente numero entre corchetes es igual que la variable original, es decir, hace referencia a la vela en la que se está comprobando la

información, donde [0] quiere decir que es sobre la última vela, [1] la penúltima, [2] la antepenúltima, etc…

```
protected override void OnBarUpdate()
        {
                Velas=0;
                if(c==0 && v==0 &&
            (Close[0]-Closes[1][0])>(1.5*ATR(BarsArray[1],5)[0])
            && Closes[1][3]>Opens[1][3] &&
            Closes[1][2]>Opens[1][2]&& Closes[1][1]>Opens[1][1])
                {
                      EnterLong(10, "Entrada Larga");
                      c=1;
                      Velas=0;
                }

                if(ToTime(Time[0]) < ToTime(15, 33, 0))
                      {c=0;}

            }
```

Ejemplo 3: Reglas del modelo.

En la cuarta parte se encuentran las declaraciones de las variables externas, que deben coincidir en nombre y tipo con las inicializaciones que se han hecho en la segunda zona.

```
#region Properties

            public int Velas
        {
            get { return velas; }
            set { velas = Math.Max(0, value); }
        }
            public int C
        {
            get { return c; }
            set { c = Math.Max(0, value); }
        }
            public int V
        {
            get { return v; }
            set { v = Math.Max(0, value); }
        }
        #endregion
    }
}
```

Ejemplo 4: Declaración de variables externas.

3. Optimización de la estrategia.

Para optimizar la estrategia se ha elegido en este caso un filtro de valores y una optimización en función de la duración del periodo usado.

Para la primera optimización se han elegido una serie de valores que tienen una gran liquidez y una gran volatilidad, y que además de poseer mucho volumen, tienen un alto ATR en relación con su precio. Para realizar este filtro, que es quizá la parte más importante y compleja a la hora de realizar de estrategias de trading, se pueden usar screnners, como el de Finviz: http://finviz.com/screener.ashx? que posee gran cantidad de indicadores y elementos técnicos.

La segunda optimización se ha basado en elegir el tamaño del intervalo de tiempo que han de durar los periodos. En primer lugar se eligieron periodos de un día, y posteriormente de medio día, que, como la sesión bursátil en el mercado americano dura 6:30 horas, cada periodo se ha establecido de 3:15 horas, 195 minutos.

4. Backtest de la estrategia.

Otra de las diferencias con respecto a la anterior práctica es que, dado que esta estrategia no requiere de un entorno de mercado concreto, se ha aplicado sobre varios entornos, los que se incluyen desde 2010 a la actualidad, obteniéndose unos resultados similares en todos ellos.

Se ha realizado un primer backtest usando un periodo de tiempo igual a un día y se han obtenido unos resultados de un 45% de fiabilidad y un profitfactor de 1,32.

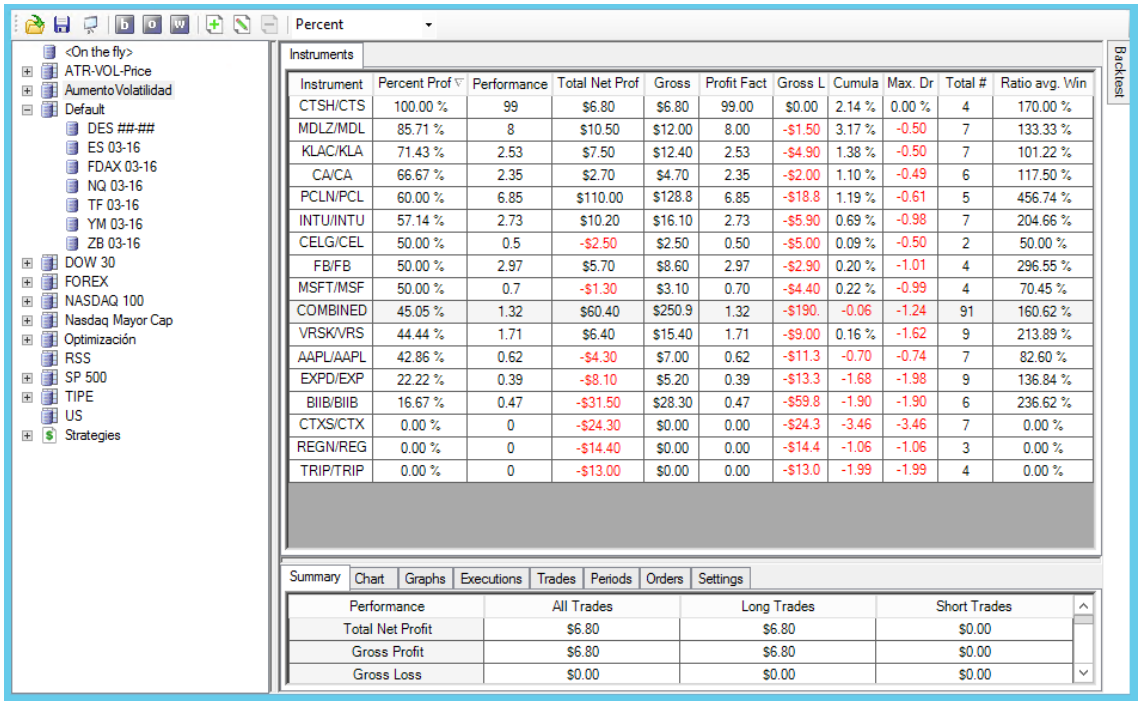
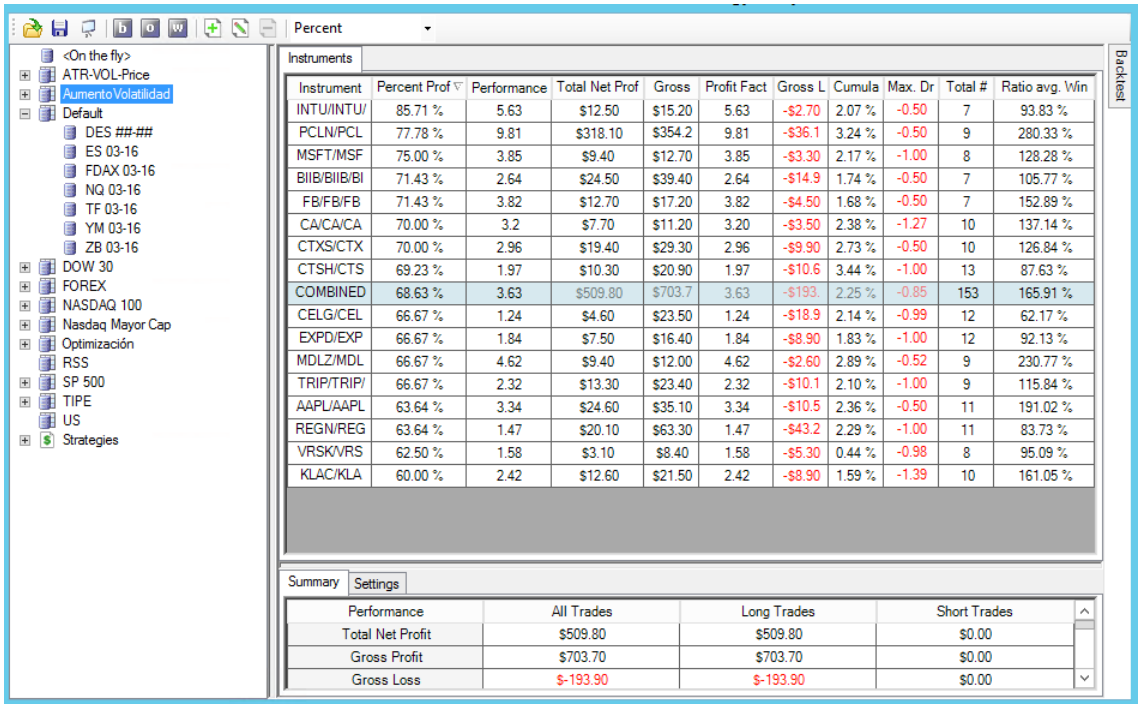


Figura 1: Backtest.

Posteriormente se ha realizado otro backtest tras la optimización comentada en el apartado anterior mejorando sustancialmente los resultados. En este segundo backtest se han obtenido unos resultados de un 68% de fiabilidad y un 3,63 de profit factor. Esto hace ver de nuevo la necesidad de los filtro y de la optimización de una estrategia para obtener los mejores resultados posibles.

Esta estrategia y la de la práctica anterior pueden funcionar, y de hecho es recomendable que lo hagan, en paralelo, así cada una puede aprovechar las distintas situaciones de mercado para las que han sido creadas.



Instrument	Percent Prof	Performance	Total Net Prof	Gross	Profit Fact	Gross L	Cumula	Max. Dr	Total #	Ratio avg. Win
INTU/INTU/	85.71 %	5.63	$12.50	$15.20	5.63	-$2.70	2.07 %	-0.50	7	93.83 %
PCLN/PCL	77.78 %	9.81	$318.10	$354.2	9.81	-$36.1	3.24 %	-0.50	9	280.33 %
MSFT/MSF	75.00 %	3.85	$9.40	$12.70	3.85	-$3.30	2.17 %	-1.00	8	128.28 %
BIIB/BIIB/BI	71.43 %	2.64	$24.50	$39.40	2.64	-$14.9	1.74 %	-0.50	7	105.77 %
FB/FB/FB	71.43 %	3.82	$12.70	$17.20	3.82	-$4.50	1.68 %	-0.50	7	152.89 %
CA/CA/CA	70.00 %	3.2	$7.70	$11.20	3.20	-$3.50	2.38 %	-1.27	10	137.14 %
CTXS/CTX	70.00 %	2.96	$19.40	$29.30	2.96	-$9.90	2.73 %	-0.50	10	126.84 %
CTSH/CTS	69.23 %	1.97	$10.30	$20.90	1.97	-$10.6	3.44 %	-1.00	13	87.63 %
COMBINED	68.63 %	3.63	$509.80	$703.7	3.63	-$193.	2.25 %	-0.85	153	165.91 %
CELG/CEL	66.67 %	1.24	$4.60	$23.50	1.24	-$18.9	2.14 %	-0.99	12	62.17 %
EXPD/EXP	66.67 %	1.84	$7.50	$16.40	1.84	-$8.90	1.83 %	-1.00	12	92.13 %
MDLZ/MDL	66.67 %	4.62	$9.40	$12.00	4.62	-$2.60	2.89 %	-0.52	9	230.77 %
TRIP/TRIP/	66.67 %	2.32	$13.30	$23.40	2.32	-$10.1	2.10 %	-1.00	9	115.84 %
AAPL/AAPL	63.64 %	3.34	$24.60	$35.10	3.34	-$10.5	2.36 %	-0.50	11	191.02 %
REGN/REG	63.64 %	1.47	$20.10	$63.30	1.47	-$43.2	2.29 %	-1.00	11	83.73 %
VRSK/VRS	62.50 %	1.58	$3.10	$8.40	1.58	-$5.30	0.44 %	-0.98	8	95.09 %
KLAC/KLA	60.00 %	2.42	$12.60	$21.50	2.42	-$8.90	1.59 %	-1.39	10	161.05 %

Performance	All Trades	Long Trades	Short Trades
Total Net Profit	$509.80	$509.80	$0.00
Gross Profit	$703.70	$703.70	$0.00
Gross Loss	$-193.90	$-193.90	$0.00

Figura 2: Backtest.