Aplicación de predicción en bolsa basada en indicadores técnicos y computación evolutiva

Carlos Beceiro Castillo
Cristina Valentina Espinosa Victoria
José Javier Martínez Pagés

GRADO EN INGENIERÍA INFORMÁTICA

FACULTAD DE INFORMÁTICA

DE DEPARTAMENTO DE SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

UNIVERSIDAD COMPLUTENSE DE MADRID

TRABAJO FIN DE GRADO EN INGENIERÍA INFORMÁTICA

Madrid, 19 de Junio de 2015

Director: Ismael Rodríguez Laguna

Autorización de difusión y utilización

Nosotros, Carlos Beceiro Castillo, Cristina Valentina Espinosa Victoria y José Javier Martínez Pagés, autores de este documento, autorizamos a la Universidad Complutense de Madrid a la difusión y utilización del mismo.

Carlos Beceiro Castillo
DNI: 47293637-W

Cristina Valentina Espinosa Victoria
DNI: 01672111-B

José Javier Martínez Pagés
DNI: 72350825-R

Madrid, 19 de Junio de 2015

Índice

Contenido

Índice de abreviaturas

ADX: Average Directional Index
ATR: Average True Range
DI: Directional Indicator
DM: Directional Movement
EMA: Exponential Moving Average
HCH: Hombro Cabeza Hombro
IRPF: impuesto sobre la renta de las personas físicas
IVA: impuesto sobre el valor añadido
MACD: Moving Average Convergence Divergence
máx.: máximo
mín.: mínimo
nº : número
pp: Pivot Point
RS: Relative Strength
RSI: Relative Strength Index
TSI: True Strength Index
PSO: Particle Swarm Optimization
DE: Differential Evolution

Resumen

Los inversores de bolsa utilizan dos factores principales para decidir cuándo invertir y cuándo no:

- Análisis fundamental: análisis en base a las noticias.
- Análisis técnico: análisis en base a la evolución de los precios y volúmenes.

El objetivo de este proyecto es crear una aplicación que, utilizando algoritmos genéticos, y otros tipos de algoritmos evolutivos y de aprendizaje automático, sea capaz de utilizar exclusivamente el análisis técnico para tratar de maximizar las ganancias obtenidas en un período de tiempo.

Se realizará un análisis de los resultados para comprobar si la utilización del análisis técnico es suficiente para la inversión en bolsa y si existe una estrategia general que permita ganar más dinero para cualquier período de tiempo dado que la estrategia básica de comprar el primer día del período y vender el último.

El programa deberá también poder representar de manera gráfica la evolución de los precios y los métodos analíticos e indicadores usados para el análisis técnico.

También se controlarán los impuestos y los cánones para hacer que los resultados sean lo más realistas posibles.

Palabras clave

Bolsa, análisis técnico, computación evolutiva, algoritmo genético, PSO, evolución diferencia, Java, Octave.

Abstract

Stock market investors use two main tools to decide when to invest and when not to:

- Fundamental analysis: analysis based on the news.
- Technical analysis: analysis based on the evolution of prices and volumes.

The objective of this project is to create an application that using genetic algorithms, other evolutionary algorithms and machine learning, and applying technical analysis, tries to maximize profits over a period of time.

An analysis of the results will be performed to check whether the use of technical analysis is sufficient for investing in stock markets and whether there is a general strategy to earn, on any period of time, more money than the basic strategy of buying the first day and selling the last.

The program should also be able to graphically represent the evolution of the stock and of the analytical methods and indicators used for technical analysis.

Taxes and fees shall also be monitored in order to make the results as realistic as possible.

Keywords

Stock market, technical analysis, evolutionary computation, genetic algorithm, PSO, differential evolution, Java, Octave.

Capítulo 1 - Introducción

1. Conceptos

Los principales conceptos que es necesario manejar para el entendimiento de la memoria están relacionados con el análisis técnico de la bolsa, algoritmos de optimización heurísticos, algoritmos evolutivos y algoritmos de aprendizaje automático.

Los métodos utilizados para el análisis técnico de la bolsa utilizados en el proyecto consisten en la observación de ciertos patrones tales como el Hombro-cabeza-hombro y la Curva de Elliot, el precio máximo alcanzado, el precio mínimo y el volumen de un período concreto de días para generar un sistema de reglas que permita lanzar señales de compra y venta. Cada uno de los indicadores técnicos utilizados se explicará en el capítulo 2.

Los algoritmos de programación evolutiva son algoritmos de optimización y de búsqueda de soluciones basados en las teorías de evolución biológicas. En concreto el primer algoritmo evolutivo que utilizamos para el proyecto fue el algoritmo genético.

Los algoritmos genéticos tratan de optimizar la solución de un problema tratando de emular la evolución biológica, centrándose en la variación genética mediante el intercambio de genes, mutaciones aleatorias y seleccionando al mejor individuo mediante una función de *fitness* que indique lo apto que es. Hace falta representar la solución del problema como una cadena de datos y fijar los puntos por la que se puede cortar para poder cruzar esa solución concreta con otras soluciones.

Utilizamos también el algoritmo de optimización por enjambre de partículas, este algoritmo evoca el comportamiento de los enjambres de abejas, es una versión simplificada y adaptada de modelos de conducta basados en enjambre. Se realiza una analogía entre cada solución concreta encontrada hasta el momento del problema y una abeja; inicialmente cada solución avanza de manera errática e individual, pero cuando se encuentra una nueva mejor solución al problema, el resto de las soluciones desvían en cierta medida su movimiento hacia la nueva solución.

Los algoritmos de aprendizaje automático pretenden conseguir inducir comportamientos y patrones dados unos datos de entrada no necesariamente estructurados con atributos que los definen, tratan de simular el aprendizaje. Para tratar de reconocer el patrón Hombro-cabeza-hombro utilizamos redes neuronales, las redes neuronales son un tipo de algoritmo de aprendizaje automático que simula el funcionamiento de las neuronas cerebrales, esencialmente combina polinomialmente los atributos de los ejemplos asignándoles un cierto peso a cada uno.

2. Antecedentes

La idea del desarrollo del proyecto surgió porque el Director del proyecto realizó una implementación del algoritmo genético y de tres indicadores técnicos, entrenó el algoritmo con distintos índices y lo probó con otros, obteniendo predicciones que parecían tener ganancias pero no pudiendo determinar que los resultados fueran concluyentes.

En [1] se aplicó un algoritmo genético a los datos del Dow Jones para tratar de obtener ganancias, sin embargo, a diferencia de nuestro algoritmo genético, los parámetros que maneja el algoritmo no son los indicadores técnicos, si no el cambio de precio, volumen y los días restantes para que se produzca el pago de dividendo. Entrenando el algoritmo con los datos de una semana y probándolo en la siguiente obtuvieron buenos resultados.

En [2] se aplican algoritmos genéticos utilizando la varianza, covarianza y otras técnicas estadísticas a los precios pasados para obtener reglas de compra y venta. Los resultados fueron buenos, pero se avisa de que es imperativo ampliar la investigación para poder determinar si los resultados son concluyentes o no.

En [3] se aplicaron algoritmos genéticos utilizando los indicadores RSI y MACD y se evaluó si se habían obtenido ganancias en un determinado período de tiempo, los resultados fueron positivos y se concluye que es posible encontrar las zonas que proporcionan ganancias, aunque se considera necesario una futura investigación que busque encontrar un patrón que se repita.

En [4] se utilizaron algoritmos genéticos usando como parámetros osciladores estocásticos, bandas de Bollinger, el ratio de cambio de los precios y los indicadores MACD y RSI; se logró generar una gran cantidad de individuos que obtienen ganancias con el conjunto de entrenamiento pero no en el conjunto de test, y unos pocos que obtienen ganancias en ambos, por lo que se puede decir que el resultado fue positivo pero necesita de futuras investigaciones.

En [5] se utilizaron los indicadores MACD, RSI, ADX, EMA y osciladores estocástico como parámetros para un algoritmo genético aplicado a la bolsa egipcia y obtuvieron buenos resultados.

En [6] se cuenta cómo utilizando los retornos del capital empleado y los ratios de precio/ganancia como parte de los individuos de un algoritmo genético se obtienen ganancias.

En [7] se utilizan las subidas del valor de las cotizaciones como entradas para ambos algoritmos y obtienen ganancias utilizando ambos, sin embargo siendo el algoritmo genético mucho más rápido y obteniendo ganancias parecidas.

En [8] se utilizaron medias móviles de los precios para entrenar un algoritmo genético y obtuvieron ganancias.

Exceptuando el último artículo citado, no se tienen en cuenta impuestos ni cánones en las simulaciones realizadas; además en todos se realiza la simulación bajo la hipótesis de un mercado líquido, es decir, se considera despreciable el peso que tienen las operaciones de un individuo sobre el mercado.

En [9] se utiliza el algoritmo PSO con los indicadores Directional Movement Index (DMI), Linear Regression (LIN), Moving Average Convergence-Divergence (MAC), Moving Average (MAV) y Parabolic Stop and Reverse (PSR) eligiendo el peso de cada uno de ellos, sobre el índice Dow Jones. El estudio refleja buenos resultados, mejorando los obtenidos por cada uno de los inidcadores.

Se puede concluir que en todos los artículos mencionados anteriormente los autores encuentran resultados positivos, aunque suelen avisar de la necesidad de futuras investigaciones que reafirmen los resultados. Para contribuir al avance del estado del arte, nosotros consideramos necesario una simulación más realista que las ya realizadas, y por eso se tendrá en cuenta las pérdidas que ocasionan los impuestos y los cánones.

3. Plan de trabajo

El desarrollo del proyecto se realizó mediante iteraciones de dos semanas de duración. Cada dos semanas se mantenía una reunión con el profesor, en dicha reunión se acordaban los siguientes avances a realizar y al salir de ésta se dividía el trabajo.

A continuación se mostrará el trabajo realizado por cada persona:

3.1 Carlos Beceiro Castillo

Durante el verano del 2014 realizó una lectura de la bibliografía propuesta por el profesor y un repaso del funcionamiento del algoritmo genético. Al comienzo del curso realizó un estudio del trabajo inicial realizado por el profesor para poder incorporarlo al proyecto. También realizó una investigación inicial del estado del arte en aquel momento.

Realizó un estudio acerca de los diferentes métodos analíticos que podrían implementarse, aprendiendo de qué manera eran aplicados a la inversión en bolsa.

Comenzó la implementación de las clases Individuo y Población del paquete Algoritmo_genético basándose en el estudio previo del código existente en C++ aportado por el profesor.

Realizó junto con los otros dos miembros del grupo distintas pruebas para ver los resultados iniciales de la implementación del algoritmo genético, primero sin los impuestos y después con ellos.

Llevó a cabo varias pruebas para comprobar los resultados iniciales de la implementación del algoritmo genético junto con los otros dos miembros del grupo. Estas pruebas se realizaron aplicando impuestos y sin aplicarlos.

Descubrió varios errores que tenían que ver con una mala copia de los parámetros del individuo, y junto con José Javier vió un mal funcionamiento de la inicialización aleatoria de los parámetros del algoritmo genético, arreglando la parte correspondiente a utilizar una sola semilla aleatoria estática para todas las inicializaciones de los parámetros.

Participó en la detección del overfitting y sus efectos a la hora de probar los resultados obtenidos en distintos entornos (índices o distintos momentos temporales de las cotizaciones) mientras se trabajaba sin impuestos.

Investigó otros algoritmos meta-heurísticos que se pudieran aplicar a la solución del problema tratado. De la investigación, y en común con los demás integrantes, decidió implementar el algoritmo de enjambre de partículas o PSO (Particle Swarm Optimization).

Investigó posibles implementaciones para dicho algoritmo y concluyó implementarlo utilizando la librería SwarmOpsJava que facilitó dicha implementación

y aportó la posibilidad de extender el número de algoritmos de optimización que se pudieran aplicar al problema en un futuro.

Implementó la clase PsoIndicesBursatiles que representa el problema de manera adecuada para poder utilizar la librería.

Implementó la clase Algoritmo_Pso que crea el entorno del algoritmo estableciendo sus parámetros y lo hace disponible desde la GUI y desde las pruebas.

Desarrolló la conversión de un individuo representado como una lista de métodos analíticos a una representación como un array de números reales. Esto permite, además del uso de la librería, la posibilidad de simplificar futuras implementaciones de algoritmos basados en múltiples agentes que traten de resolver el problema.

Añadió a las clases ya existentes creadas por sus compañeros José Javier y Cristina Valentina los métodos y las funciones necesarias para el tratamiento de los individuos por parte de la librería.

Modificó la función de fitness para adaptarla a la premisa de la librería SwarmOps, que busca minimizar la solución.

Desarrolló la adaptación del código a la implementación de la librería SwarmOps Java del algoritmo de evolución diferencial tras un estudio de su viabilidad para acometer la posible solución al problema.

Realizó las pruebas acerca del funcionamiento del algoritmo PSO comparando su optimización con la del algoritmo genético.

Realizó las modificaciones necesarias para incluir el menú Algoritmo PSO en la GUI y la adaptación del código del algoritmo.

Realizó la adaptación del código de las pruebas simples (trabajando sobre índices individualmente) para poder realizarlas sobre el algoritmo PSO y ejecutó la pruebas.

Diseñó pruebas para comprobar la aleatoriedad de los resultados obtenidos de los algoritmos y la necesidad de realizar varias repeticiones sobre las distintas partes de las pruebas para mejorar el comportamiento de éstas.

Analizó los resultados obtenidos de las pruebas y redactó las secciones de la memoria correspondientes.

Redactó las secciones de la memoria que tratan sobre la librería SwarmOps Java y los dos algoritmos implementados con ella, el PSO y el DE (Differential Evolution).

3.2 Cristina Valentina Espinosa Victoria

Dedicó el verano de 2014 a leer la bibliografía propuesta por el profesor, la cual incluye un trabajo de fin de máster que estudia el MACD, un libro sobre indicadores y un artículo en el que intentaron aplicar algoritmos genéticos.

Al comienzo del curso investigó e implementó los indicadores técnicos EMA, MACD, RSI, TSI, ADX y los Soportes y Resistencias junto con José Javier Martínez, buscando en diferentes páginas webs sus definiciones y las fórmulas matemáticas necesarias para aplicarlos.

También realizó el diseño seguido por los indicadores técnicos a través de la interfaz Método_analítico y la clase abstracta Método_parametrizable con José Javier Martínez.

Implementó la carga de datos desde archivos excel presente en la clase leeExcel del paquete lógica.cargaDatos para poder obtener los valores de los diferentes índices haciendo uso de la librería JExcelAPI, además de incluir alguna pequeña adaptación del formato de los datos para que fuera compatible con Java.

Más adelante añadió una clase exportaExcel en el paquete lógica.cargaDatos con la implementación para poder almacenar en un archivo excel los valores resultantes de aplicar cada indicador implementado a un índice desde el programa.

La implementación de la clase Estadísticas del paquete Utilidades.matemáticas también se realizó en conjunto con José Javier Martínez. Los métodos de esta clase se fueron implementando a medida que surgían nuevas necesidades en el resto del proyecto.

Llevó a cabo una pequeña búsqueda sobre los índices más importantes del Ibex35, comprobando la disponibilidad de los datos y buscando webs que proporcionaran dichos datos en archivos excel con un formato adecuado para poder probar el algoritmo avanzado, el cual actúa sobre varios índices a la vez.

Hizo la interfaz gráfica de la ventana principal del programa en el paquete vistas, desde la que podemos visualizar los indicadores, carga/exportar datos y ejecutar los algoritmos con ayuda de José Javier Martínez. Esto incluye la ventana de preferencias a la que se accede desde el menú y desde la que determinamos las preferencias sobre los algoritmos, y las gráficas de los diferentes indicadores disponibles, las cuales se implementaron en un paquete aparte vistas.indicadores creando una clase por indicador.

Cuando fue necesario cargar más de un índice, buscó la forma de mostrar un explorador de archivos con el que poder seleccionar el archivo deseado para que cargar los archivos excel fuera más sencillo y lo implementó en la clase SelectorArchivos del paquete Controlador.

Más adelante añadió ella sola una segunda ventana encargada de mostrar los resultados de ejecutar los algoritmos y realizar las simulaciones.

Realizó un estudio sobre los cánones, comisiones e impuestos presentes en la bolsa, así como otros posibles agentes relacionados con la bolsa que supongan un gasto de dinero, buscando información en diferentes webs de internet y preguntando en algún banco y a personas con alguna experiencia en bolsa para posteriormente

comparar datos y poder realizar la implementación de las diferentes clases presentes en el paquete lógica.impuestos de la forma más genérica posible.

Llevó a cabo varias pruebas para comprobar los resultados iniciales de la implementación del algoritmo genético junto con los otros dos miembros del grupo. Estas pruebas se realizaron aplicando impuestos y sin aplicarlos.

Investigó con José Javier Martínez el funcionamiento de dos indicadores técnicos, cuyo concepto es algo más abstracto: el Hombro-Cabeza-Hombro y la Curva de Elliot. Los dos realizaron en conjunto una implementación inicial, la cual está basada en un método determinista que idearon ellos mismos. También realizaron los dos juntos el etiquetado manual de los ejemplos de entrenamiento, necesarios para la Red Neuronal utilizada en la segunda implementación, y llevaron a cabo una investigación sobre la posibilidad de aplicar técnicas de Percepción Computacional para la resolución del problema.

Realizó la fase de prueba del nuevo algoritmo genético junto con José Javier Martínez.

Se encontraron algunos problemas debido a que en días festivos no hay bolsa y estos son diferentes en cada país, además de tener diferente número de días en cada índice a invertir. Estos problemas se solucionaron en conjunto con José Javier Martínez.

Para poder realizar las pruebas con los algoritmos genéticos, realizó la implementación de un código que automatizara dichas pruebas junto con José Javier Martínez para agilizar el proceso. La interpretación de los resultados obtenidos también se hizo en conjunto con José Javier Martínez.

Redactó los resultados obtenidos en las pruebas de las ejecuciones de los algoritmos genéticos junto con José Javier Martínez.

Redactó en la memoria la explicación sobre las partes del código que se encargó de implementar.

También se encargó de redactar el índice de abreviaturas y el índice de figuras, encargándose de poner título, enumerar y establecer un formato para cada una de las figuras presentes en la memoria.

Se encargó de guardar las diferentes versiones por las que iba pasando el código del proyecto para tener un seguro en caso de que fuera necesario volver atrás en algún momento.

3.3 José Javier Martínez Pagés

Durante el verano del 2014 realizó una lectura de la bibliografía propuesta por el profesor y un repaso del funcionamiento del algoritmo genético. Al comienzo del curso realizó un estudio del trabajo inicial realizado por el profesor para poder incorporarlo al proyecto. También realizó una investigación inicial del estado del arte en aquel momento.

Al comienzo del curso realizó la investigación y la implementación de los indicadores técnicos EMA, MACD, RSI, TSI, ADX y los Soportes y Resistencias de manera conjunta con Cristina Valentina Espinosa.

Asimismo, realizó el diseño que siguen todos los indicadores técnicos mediante la interfaz Método_analítico y la clase abstracta Método_parametrizable también de manera conjunta con Cristina Valentina Espinosa.

También realizó junto con Cristina Valentina Espinosa la implementación de las clase Estadísticas del paquete Utilidades.matemáticas según se fué viendo necesario el uso de distintas técnicas estadísticas para el proyecto.

Él solo realizó la clase Vectores del paquete Utilidades.matemáticas y la clase UtilidadesFechas del paquete Utilidades.fechas.

Realizó la adaptación de las clases Cotizaciones, Indicadores y Signal respecto al código del profesor, en el cual eran estructuras de C++; amplió la funcionalidad de las Cotizaciones para poder manejar fechas (usando la clase Date de Java), máximos, mínimos y volúmenes diarios.

Especificó e implementó la clase Controlador que del patrón utilizado Modelo-Vista-Controlador del proyecto.

Ayudó ligeramente a Cristina Valentina Espinosa a realizar la interfaz gráfica de la ventana principal.

Acabó la implementación comenzada por Carlos Beceiro de las clases Individuo y Población del paquete Algoritmo_genético que estaban basa. Realizó todos los cambios necesarios en Individuo para que en vez de ser una array de números reales fuera una lista de Metodo_parametrizable seguida de los Parametros_control.

Realizó la implementación de la clase Parametro del paquete Metodos_analiticos y la implementación de la clase Parametros_control del paquete Algoritmo_genetico.

Implementó la clase AlgoritmoGenetico y adaptó la simulación de la bolsa realizada por el profesor a nuestro código, con la excepción de todo el código relativo a los impuestos, cánones y comisiones.

Realizó junto con los otros dos miembros del grupo distintas pruebas para ver los resultados iniciales de la implementación del algoritmo genético, primero sin los impuestos y después con ellos.

Descubrió varios errores que tenían que ver con una mala copia de los parámetros del individuo, y junto con Carlos Beceiro vió un mal funcionamiento de la inicialización aleatoria de los parámetros del algoritmo genético, arreglando la parte correspondiente a utilizar una sola semilla aleatoria estática para todas las inicializaciones de los parámetros.

Investigó junto con Cristina Valentina Espinosa el funcionamiento de los dos indicadores técnicos conocidos como Hombro-Cabeza-Hombro y Curva de Elliot; y junto con ella realizó las primera implementación basada en un método determinista

ideado por nosotros, y la segunda implementación que usa una Red Neuronal. También realizó con ella la etiquetación manual de los ejemplos de entrenamiento y la investigación sobre si era posible utilizar técnicas propias de Percepción Computacional para resolver el problema.

El sólo investigó la manera de realizar un suavizado de la gráfica de las cotizaciones de la bolsa y diseñó e implementó la conexión a Octave desde Java para realizar esto, también implementó en Octave el suavizado gaussiano.

Tras esto inició la fase de diseño, modelado e implementación de un algoritmo genético que pudiera invertir en más de un índice de manera simultánea; realizó todas las adaptaciones de las clases Parámetros_control, Individuo y Población, también implementó la clase AlgoritmoGenéticoAvanzado.

Hizo la nueva simulación de la bolsa (clase Simulación_compleja) menos las partes correspondientes a los impuestos.

Adaptó la clase Controlador para que permitiera conectarse a los nuevos componentes de la interfaz gráfica.

La fase de prueba del nuevo algoritmo genético la realizó junto con Cristina Valentina Espinosa y juntos también solventaron los errores encontrados relativos a los días festivos que no hay bolsa y a la diferencia de número de días que hay en cada índice a invertir.

También junto con Cristina Valentina Espinosa realizaron la implementación de código que permitiera la automatización de los experimentos con ambos algoritmos genéticos. Con ella también realizó la interpretación de los resultados.

Redactó la partes de la memoria de “Conceptos” y “Antecedentes” correspondientes a la introducción y también su participación en el proyecto.

En el capítulo “Propuesta Software” de la memoria redactó todas las partes que trataban sobre aquello en lo que había trabajado, de tal manera que la redacción de las partes en las que trabajó con otro miembro del grupo también fueron conjuntas.

Escribió todos los resultados obtenidos de las ejecuciones de los algoritmos genéticos de manera conjunta con Cristina Valentina Espinosa Victoria.

Capítulo 2 - Propuesta Software

Este capítulo tiene como objetivo explicar la arquitectura de la aplicación, las decisiones de diseño tomadas, explorar los distintos métodos considerados y utilizados para modelar el problema, proponer un pequeño manual de usuario, y finalmente realizar un estudio sobre la licencia que debería tener el software propuesto.

Lo primero que se tuvo que decidir al comenzar el desarrollo del proyecto fue el lenguaje que se utilizaría. Se decidió utilizar Java por ser un lenguaje de propósito general que todos los miembros del grupo conocíamos y que dispone de una alta cantidad de librerías que podríamos utilizar en caso de que fuera necesario, incluidas las librerías que permiten conectar Java a otro tipo de lenguajes que pueden llegar a ser más útiles que Java para conseguir determinados objetivos, como más tarde en la memoria se puede llegar a comprobar.

1. Bolsa: Cotizaciones, Indicadores y Señales de compra/venta

Se comenzará detallando la información más básica que se necesita para representar la Bolsa:

- Cómo cotiza su precio diariamente,
- El volumen de acciones en un día determinado,
- Señales de compra y de venta,
- Indicadores que indiquen cuándo se han de lanzar las señales de compra y venta.

1.1 Cotizaciones

La representación del precio y el volumen diaria de las acciones se realizará mediante la clase Cotizaciones (Figura 2.1):

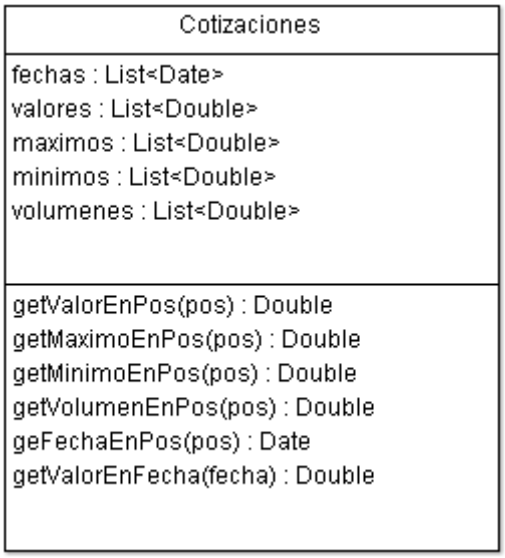


Figura 2.1. Clase Cotizaciones

Además del precio de cierre y del volumen también se guarda el valor máximo y el valor mínimo alcanzado cada día. Se almacenan en listas ordenadas según la fecha de manera creciente.

La clase dispone de métodos para obtener alguno de los valores según la posición que ocupen en las listas, o también se permite su obtención según la fecha, aunque no se hayan incluido todas las posibilidades en la figura por ahorrar espacio.

1.2 Señales de compra y venta

Se utiliza un tipo enumerado llamado *Signal* con tres posibles valores para representar lo que se puede hacer con las acciones en un determinado día:

- Compra,
- Venta,
- Nada.

1.3 Indicadores

Se utilizará la clase Indicadores (Figura 2.2) para almacenar las posibles señales de compra/venta lanzadas y una lista con un valor numérico para cada día que permite decidir si se compra, se vende o no se hace nada.

Figura 2.2. Clase Indicadores

La clase dispone de métodos para obtener la señal lanzada en un día determinado, el valor numérico que hay en el día; y métodos para cambiar estos valores.

Los métodos utilizados que se encargan de indicar cuándo se lanzan señales de compra/venta y de asignar un valor a cada día se detallan en el siguiente apartado.

2. Métodos análiticos para invertir en bolsa

Existen diversos indicadores o métodos, que a través de técnicas de análisis matemático y conocimiento empírico se pueden utilizar para generar señales de compra y venta y alcanzar ganancias. Su objetivo es estimar dónde es probable que el precio de las acciones esté cercano a su mínimo local (para dar una señal de compra) o a su máximo local (para dar una señal de venta), usando para hacer tal estimación únicamente los precios y volúmenes de la acción en el pasado. Generalmente estos indicadores dependen de una serie de parámetros que hay que ajustar en base a pasadas experiencias. Se utilizarán las ventajas ofrecidas por las interfaces y las clases abstractas para modelar estos métodos de tal manera que se mantenga una estructura lógica similar a la estructura real (Figura 2.3):

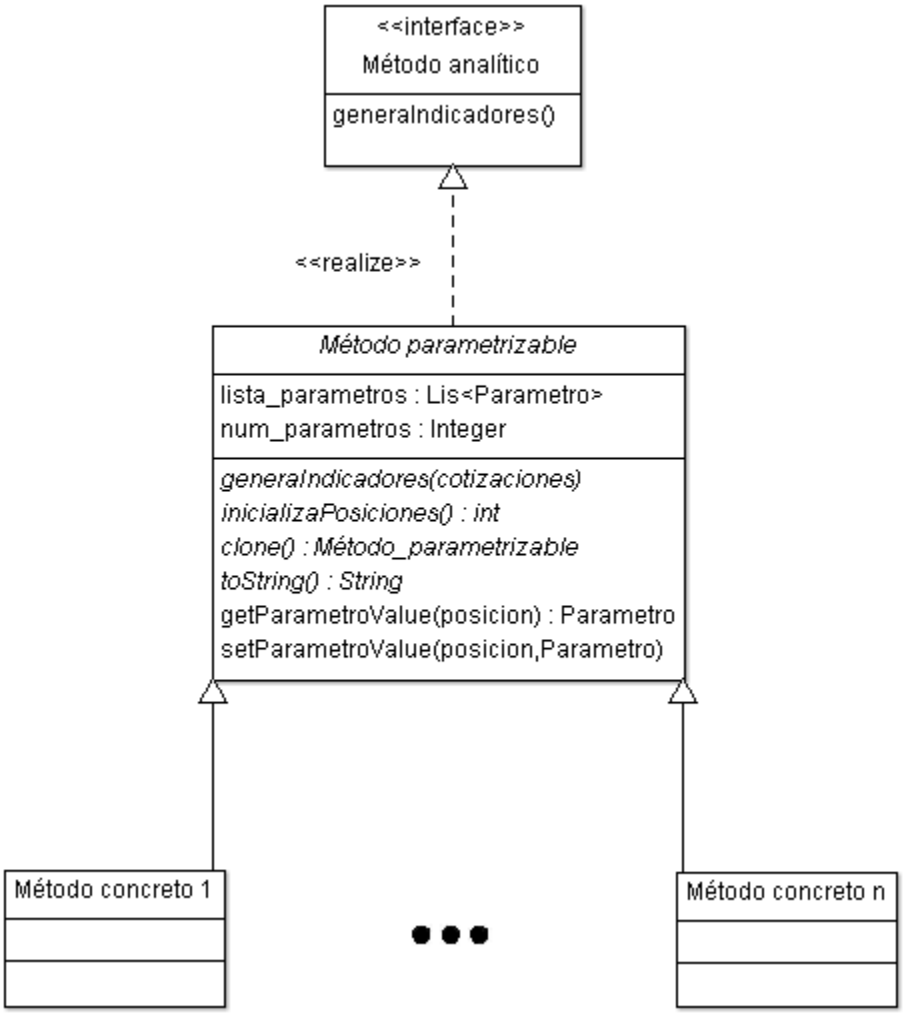


Figura 2.3. Estructura métodos analíticos

Método analítico es una interfaz que representa todos los métodos que generan señales de compra y venta. La clase abstracta Método parametrizable agrupa a todos los métodos analíticos que contienen algún parámetro y ofrece métodos que simplifican el tratamiento de estos parámetros. Los métodos concretos heredan de Método parametrizable y deberán implementar los métodos abstractos.

Para representar los parámetros se utiliza la siguiente clase (Figura 2.4):

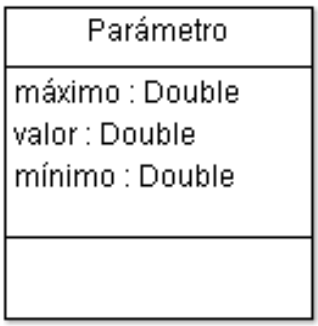


Figura 2.4. Clase Parámetro

Los métodos *getters* y *setters* no se han incluido en la figura. El atributo valor representa el valor que tiene actualmente el parámetro, y máximo y mínimo son respectivamente el valor máximo y el valor mínimo que es recomendable que alcance el parámetro. Por tanto: $valor\ \varepsilon\ [máximo, mínimo]$

Ahora se procederá a explicar uno a uno los métodos utilizados en el proyecto, la leyenda para las señales de compra y venta será la siguiente:

Comprar	
Vender	

2.1 *Exponential Moving Average*

El Exponential Moving Average es uno de los indicadores más básicos, que además de ser bastante utilizado para calcular otros indicadores, se puede utilizar por sí mismo de distintas maneras. Se puede definir con la siguiente fórmula recursiva en función de un cierto intervalo de tiempo:

$$EMA(0, intervalo) = precio_0$$
$$EMA(t, intervalo) = \alpha \times precio_t + EMA(t-1) \times (1-\alpha)$$
Donde $\alpha = 2/(intervalo + 1)$

Para generar las señales de compra o venta en un determinado día decidimos utilizar la estrategia más sencilla posible:

$$Comprar(t) \rightarrow EMA(t) = precio_t \wedge creciente(EMA(t))$$
$$Vender(t) \rightarrow EMA(t) = precio_t \wedge decreciente(EMA(t))$$

Es decir, si la función EMA corta la gráfica de la bolsa y la función EMA está creciendo se compra, si está decreciendo se vende (Véase la Figura 2.5).

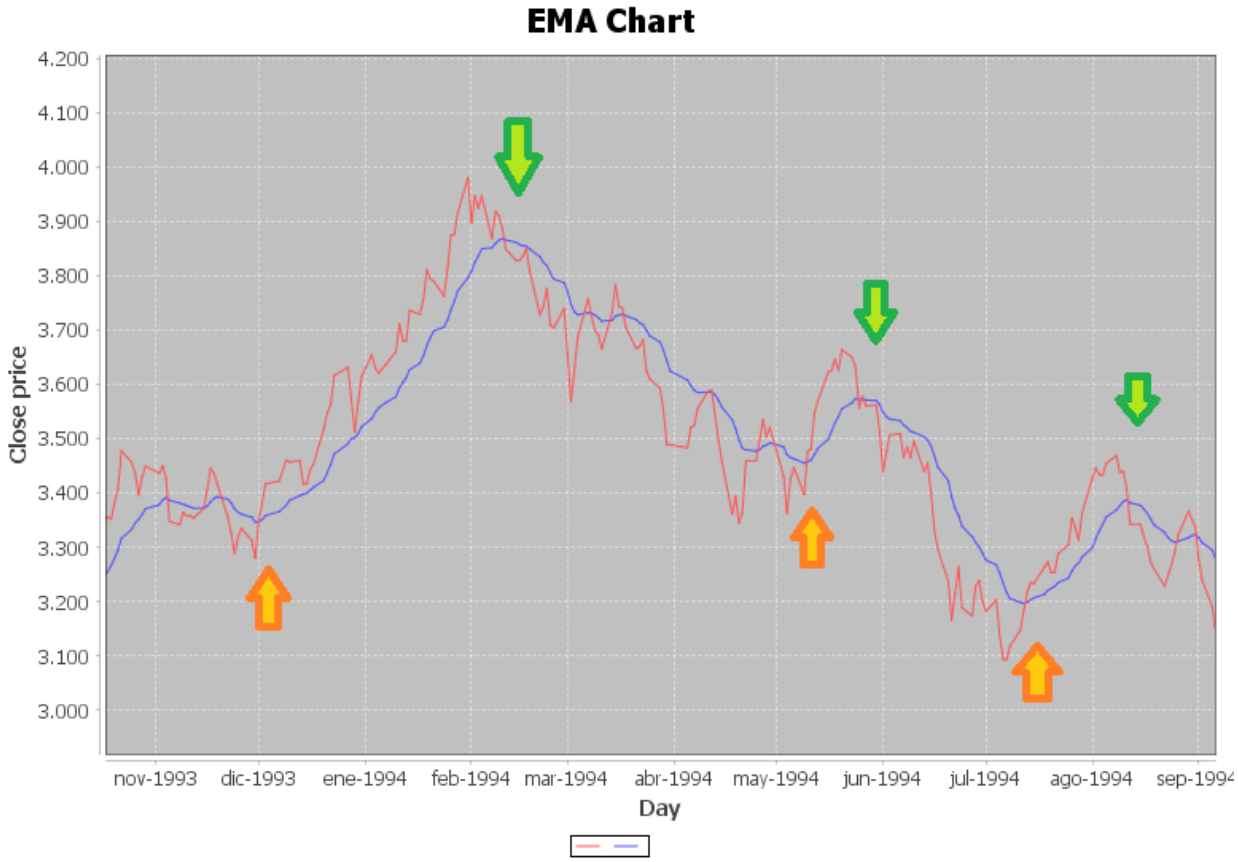


Figura 2.5. Gráfica EMA

Los parámetros variables en este caso son:

- El intervalo de tiempo,
- peso de la señal de compra,
- peso de la señal de venta.

2.2 *Moving Average Convergence Divergence*

El MACD es otro de los indicadores más usados, como tal es una diferencia de dos EMA, pero para generar señales de compra y venta se acostumbra a calcular también su histograma:

$$MACD(t) = EMA(t, intervalo) - EMA(t, intervalo')$$

$$Histograma(t) = EMA(\ MACD(t), intervalo'')$$

Donde generalmente, aunque no necesariamente $intervalo \leq intervalo'$

Se generan dos señales de compra y venta distintas. Cada una de estas señales tiene un cierto peso:

$$Compra_1(t) \rightarrow Histograma(t) > Histograma(t-1)$$
$$Vender_1(t) \rightarrow Histograma(t) < Histograma(t-1)$$
$$Comprar_2(t) \rightarrow Histograma(t) > 0 \wedge Histograma(t-1) \leq 0$$
$$Vender_2(t) \rightarrow Histograma(t) < 0 \wedge Histograma(t-1) \geq 0$$

Es decir, se compra cuando el histograma ha crecido o ha salido de negativo a positivo, y se vende cuando ha decrecido o cuando cae a negativo.

Ejemplo (Figura 2.6):

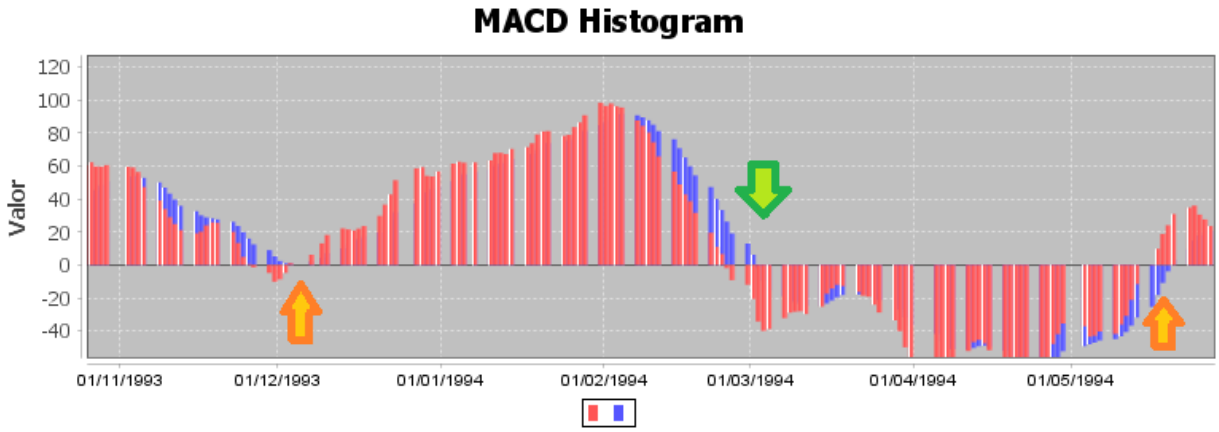


Figura 2.6. Histograma del MACD

En este caso los parámetros son:

- Los tres intervalos de los EMA para calcular la diferencia y el histograma;
- los pesos asignados a las señales de compra;
- los pesos asignados a las señales de venta.

2.3 *Relative Strength Index*

Método para calcular la fuerza relativa de la bolsa. Para calcular el RSI hace falta realizar algunos cálculos intermedios. Se comienza calculando dos vectores: U y D, que indican si ha habido crecimiento o pérdida de valor, después se utilizan los vectores para calcular la función RS, y finalmente con esta se calcula el RSI:

Si $precio_t > precio_{t-1}$ entonces:

$U = precio_t - precio_{t-1}$

$D = 0$

Si no:

$U = 0$

$D = precio_{t-1} - precio_t$

$RS = EMA(U, intervalo) \,/\, EMA(D, intervalo)$

$RSI = 100 - (100 \,/\, (1 + RS))$

Las señales se generan distinguiendo si entramos o salimos de zonas de sobrecompra y sobreventa:

$$Comprar_1(t) \rightarrow RSI(t) \geq V_{sobrecompra}$$
$$Vender_1(t) \rightarrow RSI(t) \leq V_{sobreventa}$$
$$Comprar_2(t) \rightarrow RSI(t) < V_{sobrecompra} \wedge RSI(t-1) \geq V_{sobrecompra}$$
$$Vender_2(t) \rightarrow RSI(t) > V_{sobreventa} \wedge RSI(t-1) \leq V_{sobreventa}$$

En este método los parámetros son:

- El intervalo para el cálculo del EMA;
- el valor a partir del cual se considera sobrecompra ($V_{sobrecompra}$);
- el valor a partir del cual se considera sobreventa ($V_{sobrecompra}$);
- el peso de las señales de compra y venta.

2.4 *True Strength Index*

En contraposición al RSI, el TSI no calcula la fuerza relativa, sino la verdadera. Su cálculo requiere de varios suavizados sobre el precio utilizando el EMA:

$$Suavizado_1(t) = EMA(\, EMA(precio_t - precio_{t-1}, intervalo), intervalo' \,)$$
$$Suavizado_2(t) = EMA(\, EMA(\, |\, precio_t - precio_{t-1}\, |, intervalo), intervalo' \,)$$
$$TSI(t) = 100 \times (Suavizado_1(t) \,/\, Suavizado_2(t))$$

La generación de señales de compra y venta es similar al RSI pero utilizando el indicador TSI en vez del RSI.

Los parámetros también son los mismos que en el indicador RSI.

2.5 *Average True Range*

Calcula la media de los rangos verdaderos que han alcanzado las acciones. Este indicador no genera señales de compra y venta, sino que se utiliza para calcular el indicador ADX, el siguiente que se explicará. El cálculo del ATR es:

$$TR(t) = \max\{\ (max_t - min_t), |\ max_t - precio_{t-1}\ |\ , |\ min_t - precio_{t-1}\ |\ \}$$
$$ATR(t) = (\ ATR(t-1) \times (intervalo - 1) + TR(t)\ ) \,/\, intervalo$$

Se podrá parametrizar:

- El intervalo para calcular el ATR.

2.6 *Average Directional Index*

El indicador ADX indica la fuerza que tiene la tendencia actual, y es uno de los indicadores más utilizados. Su cálculo requiere de varios cálculos intermedios:

$UpMove(t) = max_{t-1} - max_t$
$DownMove(t) = min_{t-1} - min_t$
Si $UpMove(t) > DownMove(t) \wedge UpMove(t) > 0$ entonces
    $+DM(t) = UpMove(t)$
Si no
    $+DM(t) = 0$
Si $DownMove(t) > UpMove(t) \wedge DownMove(t) > 0$ entonces
    $-DM(t) = DownMove(t)$
Si no
    $-DM(t) = 0$

$$+DI(t) = 100 \times EMA(\ +DM(t), intervalo)\ /\ ATR(t)$$
$$-DI(t) = 100 \times EMA(\ -DM(t), intervalo)\ /\ ATR(t)$$

$$ADX(t) = 100 \times EMA(\ |+DI(t) - -DI(t)|\ , intervalo'\ )\ /\ (+DI + -DI)$$

Las señales de compra y venta se deciden de la siguiente manera:

$$Vender(t) \rightarrow -DI(t-1) < +DI(t-1) \wedge -DI(t) \geq +DI(t) \wedge ADX(t) > S$$
$$Comprar(t) \rightarrow +DI(t-1) < -DI(t-1) \wedge +DI(t) \geq -DI(t) \wedge ADX(t) > S$$

Donde S es un valor real que indica que hay una fuerte tendencia, normalmente $S = 20$. Es decir, si las funciones -DI y +Di se cortan, se vende o se compra, dependiendo de cuál crezca más rápido.

Los parámetros son:

- Los intervalos de tiempo para calcular los EMA;
- el valor de fuerte tendencia;
- el peso de la señal de compra;
- el peso de la señal de venta.

2.7 Soportes y Resistencias

Las zonas de soporte y resistencia se utilizan para tratar de fijar zonas a partir de las cuales se cree que la bolsa no cruzará al alza ni a la baja, respectivamente. Se calculan partiendo de la suposición de que los máximos locales y los mínimos locales tienden a ubicarse sobre dos líneas rectas, la de los máximos y la de los mínimos respectivamente. A continuación se muestra cómo calcular las zonas, para los soportes se usará la letra *s* y para las resistencias la letra *r*:

$$pp(t) = (max_{t-1} + min_{t-1} + precio_{t-1}) \,/\, 3$$
$$s_1(t) = 2 \times pp(t) - max_{t-1}$$
$$r_1(t) = 2 \times pp(t) - min_{t-1}$$
$$s_2(t) = pp(t) + (r_1(t) - s_1(t))$$
$$r_2(t) = pp(t) - (r_1(t) - s_1(t))$$
$$s_3(t) = max_{t-1} + 2 \times (pp(t) - min_{t-1})$$
$$r_3(t) = min_{t-1} - 2 \times (max_{t-1} - pp(t))$$

Las zonas resultantes se pueden observar en la siguiente imagen (Figura 2.7), a diferente color cada zona:

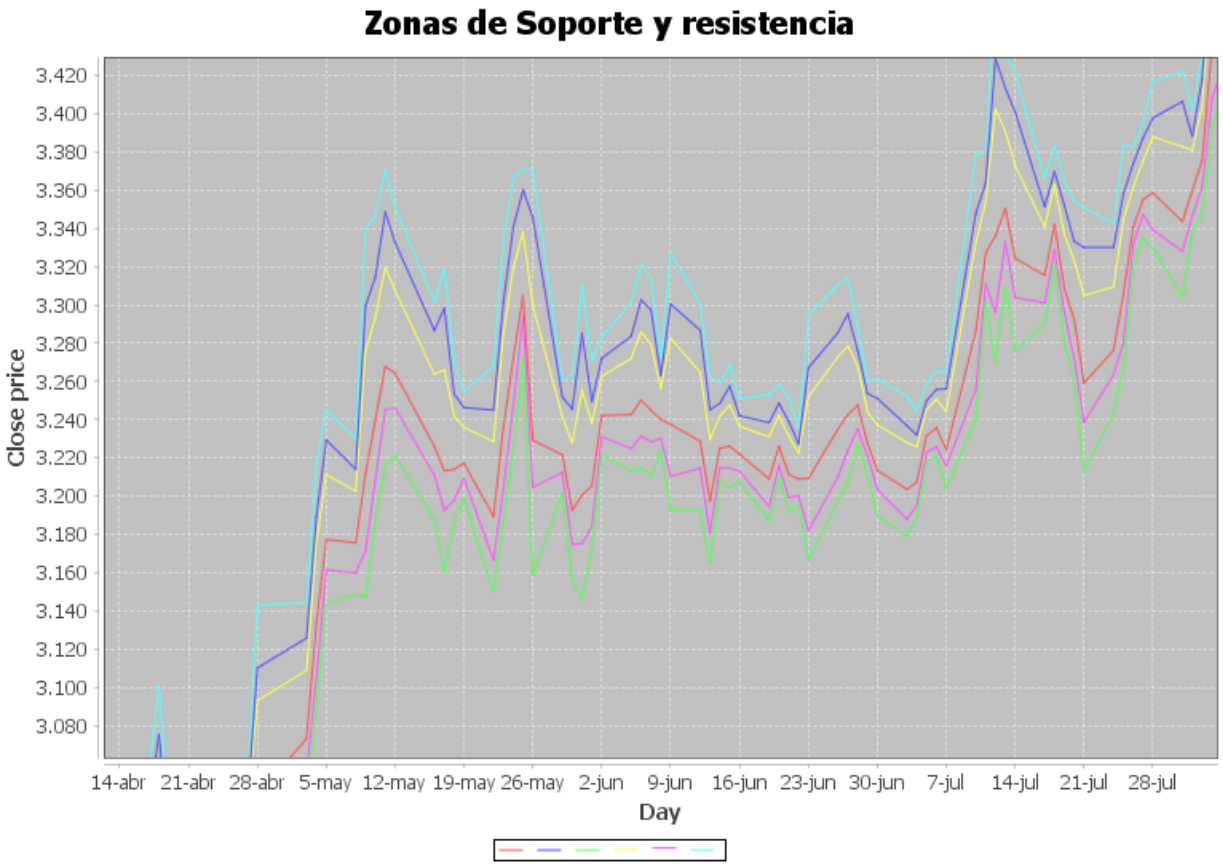


Figura 2.7. Gráfica soportes y resistencias

Se genera una señal de compra por nivel de resistencia y una señal de venta por nivel de soporte:

$$Compra_i(t) = s_i(t) \leq S_i$$
$$Vender_i(t) = r_i(t) \geq R_i$$

Los parámetros utilizados en este caso son:

- Los valores de $S_i \, \forall i \, \varepsilon \, [0,3]$;
- los valores de $R_i \, \forall i \, \varepsilon \, [0,3]$;

2.8 Hombro Cabeza Hombro

La figura Hombro Cabeza Hombro no es un indicador analítico en sí mismo, pero el satisfactorio reconocimiento del inicio de ésta figura y la predicción del intervalo de tiempo en el que ocurrirá abre puertas a nuevas estrategias de compra y venta.

La figura ideal consiste en tres parábolas consecutivas de vértices positivos, con las parábolas izquierda y derecha simétricas y la parábola de en medio debiendo tener un vértice mayor que las otras dos. Cómo conseguir que se formen tres

parábolas es difícil en la realidad, lo que se acostumbra a hacer es suavizar la gráfica y aceptar figuras que se parezcan a parábolas.

En este proyecto se utilizó la estrategia de comprar en el máximo valor de la cabeza (Figura 2.8):

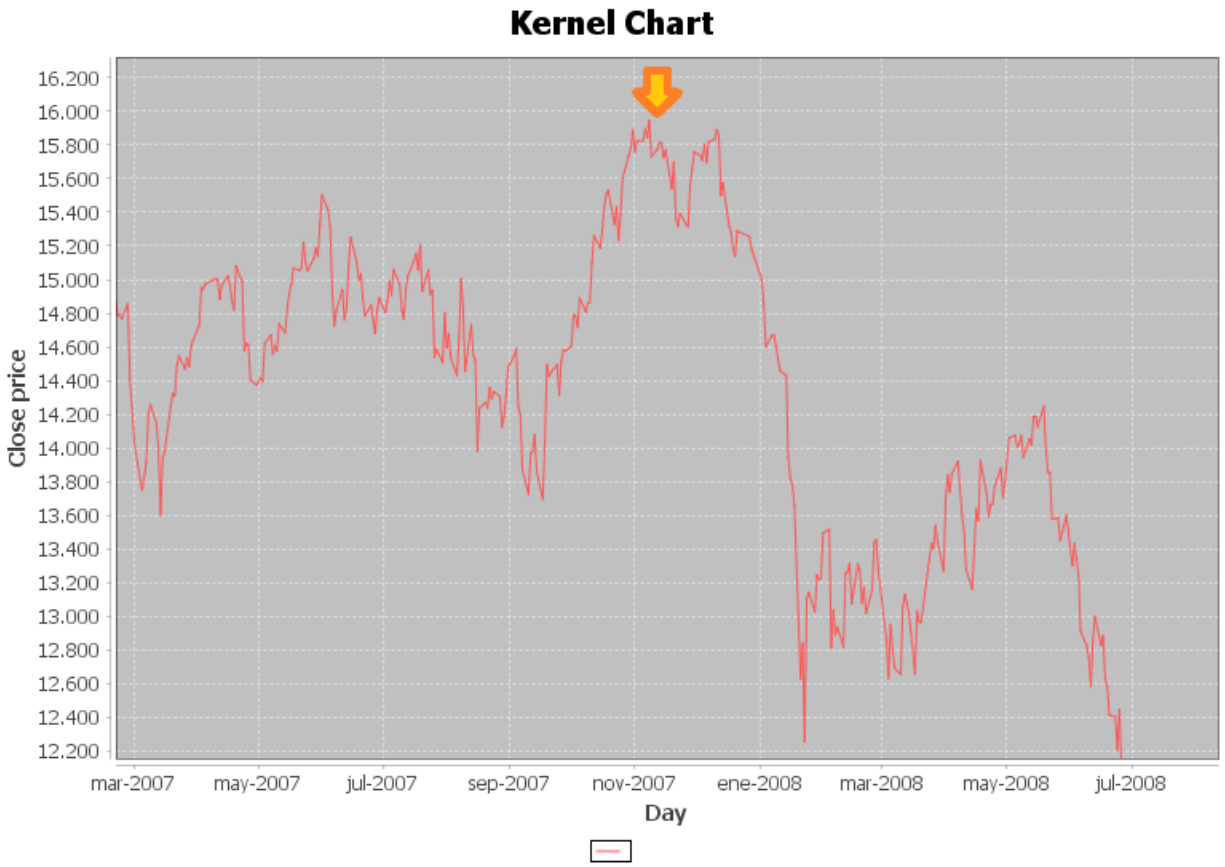


Figura 2.8. Figura Hombro Cabeza Hombro

$$Comprar(t) \rightarrow \exists t : (t \in HCH \ \wedge \ max\{ valor_t \})$$

Para tratar de reconocer esta figura se utilizaron diversas técnicas:

2.8.1 Punto de vista de inversor. Predictivo

Un inversor que sigue los movimientos de la bolsa día a día no puede conocer de manera totalmente certera si un determinado día forma parte de un Hombro Cabeza Hombro o no, como tal, debe intentar predecirlo.

Una de las maneras que teóricamente más acierta y que más beneficios consigue es esperar a que se haya producido un movimiento similar al de un hombro y se haya registrado una bajada del volumen del 15%, tras esto se debe esperar que el

valor suba y que supere al máximo alcanzado por el hombro, cuando el precio y el volumen comiencen a bajar es momento de vender, puesto que es probable que nos encontremos en una cabeza.

En la implementación se suavizó la gráfica con una implementación del kernel gaussiano realizada en Octave, se usó la estrategia anterior y no se obtuvieron buenos resultados, así que se decidió rebajar las condiciones y se desechó el peso que tiene el volumen, fijándonos así sólo en el valor de las acciones en cada día.

Con esta técnica fueron utilizados los siguientes parámetros:

- Longitud de la ventana en la que buscar el primer hombro y la cabeza;
- parámetro σdel suavizado con kernel gaussiano;
- el peso de la señal de venta.

2.8.2 Reconocimiento a posteriori

Aquí se probó a reconocer la figura entera una vez hubiera ocurrido, de tal manera que se pudiera ver de manera segura si la utilización de esta figura para generar señales de compra y venta es útil.

Se consideró el reconocimiento de la figura utilizando métodos de Percepción Computacional, formando una gráfica con los valores de la bolsa y tratando estos datos como una imagen; pero al realizar un estudio previo sobre la viabilidad de estos métodos, se determinó que la falta de una definición formal de lo que se considera un Hombro Cabeza Hombro y la cantidad de información que se pierde al suavizar la gráfica para poder reconocer algo útil hacían extremadamente complejo el reconocimiento de la figura con los métodos de reconocimiento de imagen.

Tras rechazar la opción anterior se decidió utilizar una red neuronal multicapa para predecir la figura. Se etiquetaron de manera manual todos los días entre 2009 y 2015 para utilizarlos como conjunto de entrenamiento para alimentar a la red neuronal de la siguiente manera:

$$Etiqueta_{_t} = 1 \; si \; t \; \varepsilon \; HCH$$
$$Etiqueta_t = 0 \; en \; caso \; contrario$$

Inicialmente se probó utilizando una red neuronal con un número de capas equivalente a $(n\acute{u}meroAtributos + n\acute{u}meroClases)/2$ y con ejemplos de entrenamiento con cuatro atributos correspondientes a un determinado día: el precio de cierre, los valores máximo y mínimo alcanzados en el susodicho día y el volumen alcanzado. Utilizando un 66% de los datos del conjunto de entrenamiento para entrenar y el resto para validar, se alcanzó un 60% de ejemplos bien predichos, pero

esto fue gracias a la gran cantidad de días que no formaban parte de un HCH, la cantidad de falsos negativos rozaba el 90%.

Debido a los malos resultados obtenidos se decidió que cada ejemplo de entrenamiento tuviera como atributos los valores de cotización y los volúmenes de la bolsa en el siguiente intervalo: $[t-30,\ t+30]$, lo que hacía que cada ejemplo de entrenamiento dispusiera de los datos de 60 días como atributos, es decir, los datos de los 30 días anteriores a él y los 30 días posteriores a él. Utilizando el 66% de los datos del conjunto de entrenamiento para entrenar la red neuronal y el resto para validar se consiguió un 82% de ejemplos bien predichos, en este caso tanto el número de falsos positivos como el de falsos negativos era bajo.

Como los resultados parecían buenos se decidió a probar esta red neuronal entrenada con las cotizaciones del Ibex35 entre 1993 y 2015, consiguiendo los siguientes resultados (Figura 2.9):

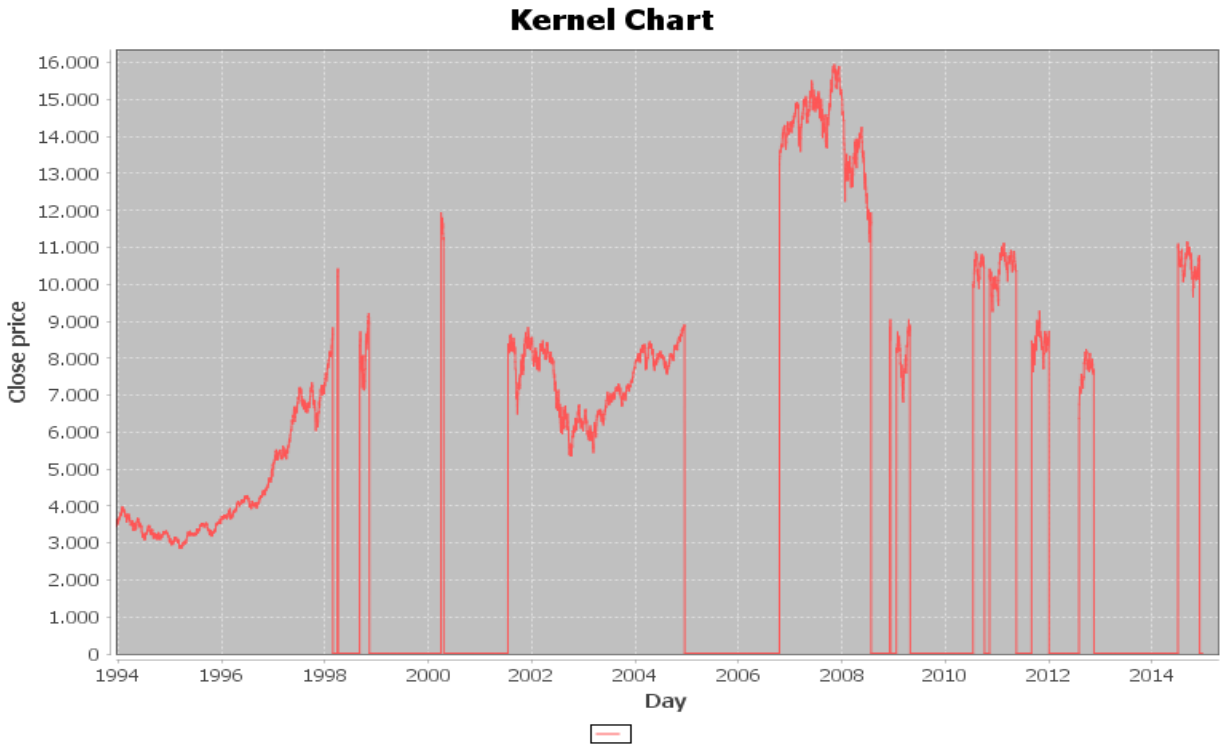


Figura 2.9. Reconocimiento del Hombro Cabeza Hombro (Ibex35)

Se puede observar que los HCH que la red neuronal encuentra solo son aceptables pasado el año 2006. Los conjuntos de días clasificados como

pertenecientes a un HCH anteriores a 2006 no son correctos, lo cual no es un buen resultado.

Se tomó la decisión de validar la red neuronal con las cotizaciones del Nikkei entre 1984 y finales de 2014 esta vez, y se obtuvieron los siguientes resultados (Figura 2.10):

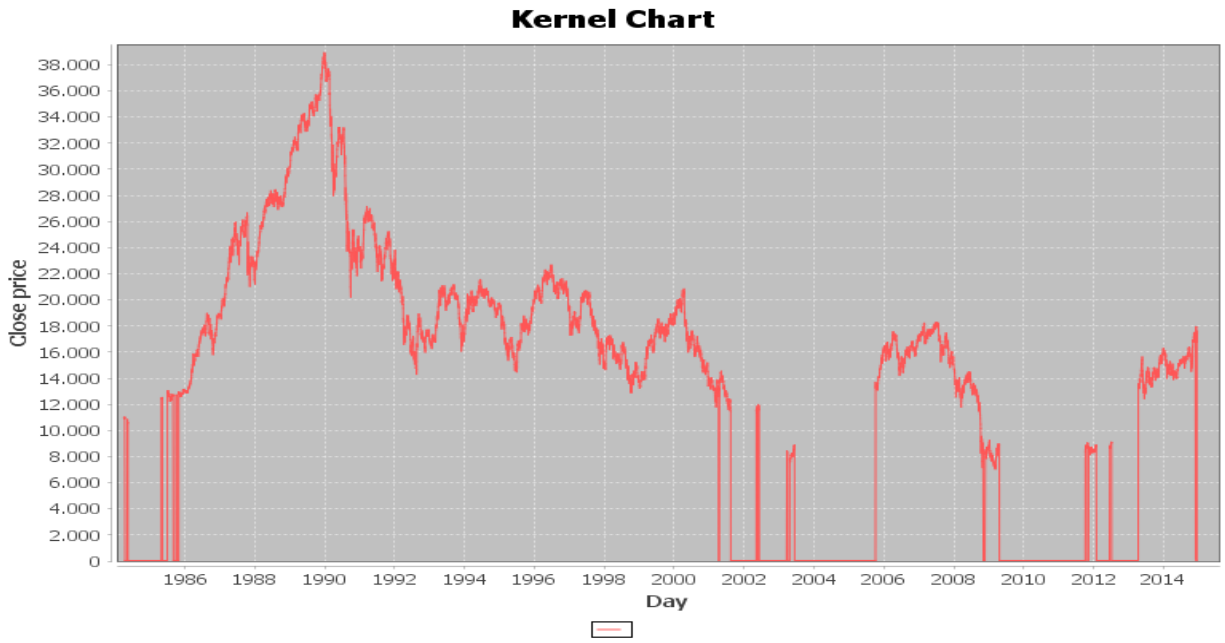


Figura 2.10. Reconocimiento del Hombro Cabeza Hombro (Nikkei)

Con excepción de la figura encontrada entre los años 2006 y 2008 no se encuentra ningún otro HCH, aunque la figura presente en el 2014 se le parezca.

Debido a los malos resultados se decidió abandonar esta línea de trabajo y utilizar la versión predictiva ya contada en el apartado anterior.

2.9 Curva de Elliot

La Curva de Elliot tampoco es un indicador analítico en sí mismo, sino una figura que se puede utilizar para generar señales de compra y venta.

Como el reconocimiento de esta figura presenta problemas similares a los que presenta el Hombro Cabeza Hombro, se decidió tratar de implementar un reconocedor sencillo para evitar así perder tiempo en solucionar un problema que puede ser demasiado complejo de resolver.

La implementación busca cuatro parábolas con el vértice positivo de tal manera que el vértice de la parábola n sea mayor que el de la parábola n-1; y una quinta parábola final con su vértice menor que la cuarta.

A pesar de tratar de suavizar la gráfica para que el reconocimiento fuera satisfactorio, el programa no encontró ninguna Curva de Elliot ni el Ibex35 ni en el Nikkei.

3. Conexión a Octave

Octave es un lenguaje de alto nivel orientado a los cálculos numéricos. Por este motivo decidimos utilizarlo para realizar el suavizado de las gráficas, para esta conexión se utilizó JavaOctave, por ser sencillo de usar y además gratuito.

Octave presenta algunas diferencias respecto a Java, como por ejemplo las posiciones en las que empiezan los arrays. Por este motivo decidimos utilizar dos clases para conectarnos a Octave: una que se encargue de adaptar los datos que hay que enviar de Java a Octave y viceversa, y otra que realice los cálculos en Octave.

Como ejemplo se incluye el diagrama de secuencia del suavizado Gaussiano (Figura 2.11):

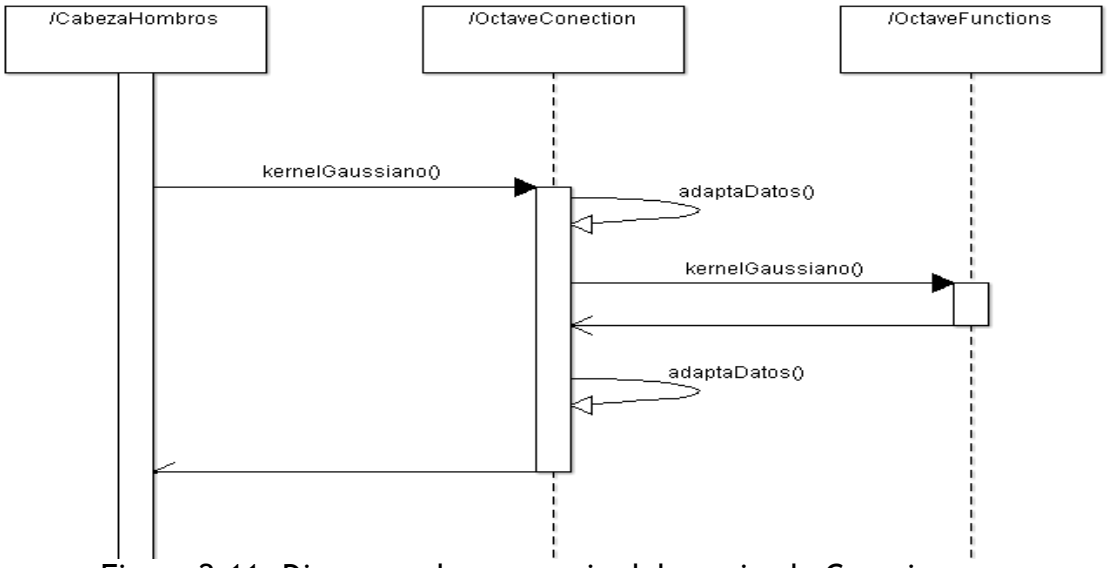


Figura 2.11. Diagrama de secuencia del suavizado Gaussiano

Los resultados de suavizar la gráfica son correctos:

- Original (Figura 2.12):

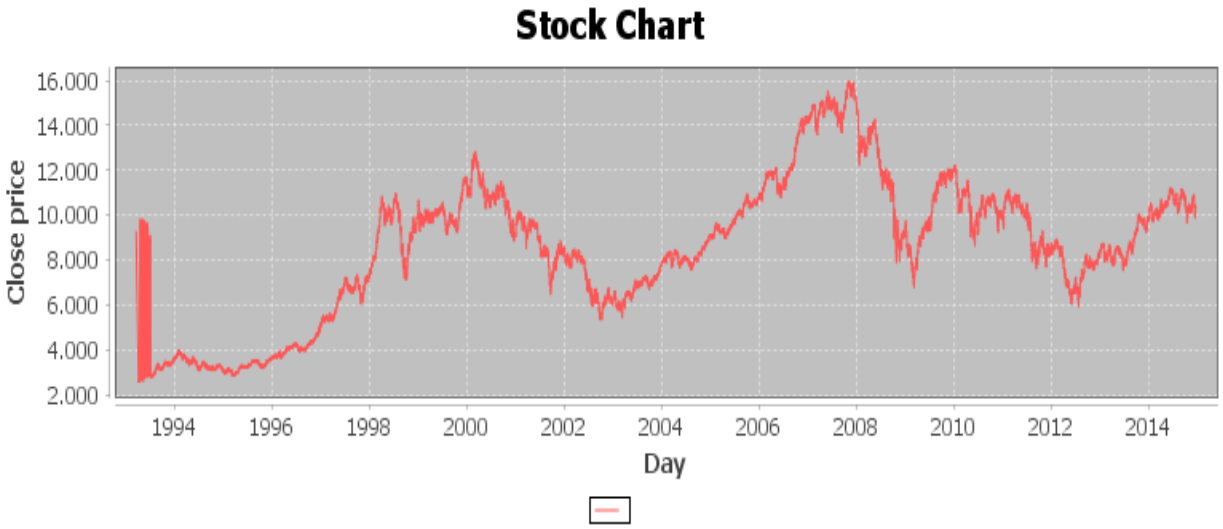


Figura 2.12. Suavizado Gaussiano. Gráfica original.

- Suavizado (Figura 2.13):

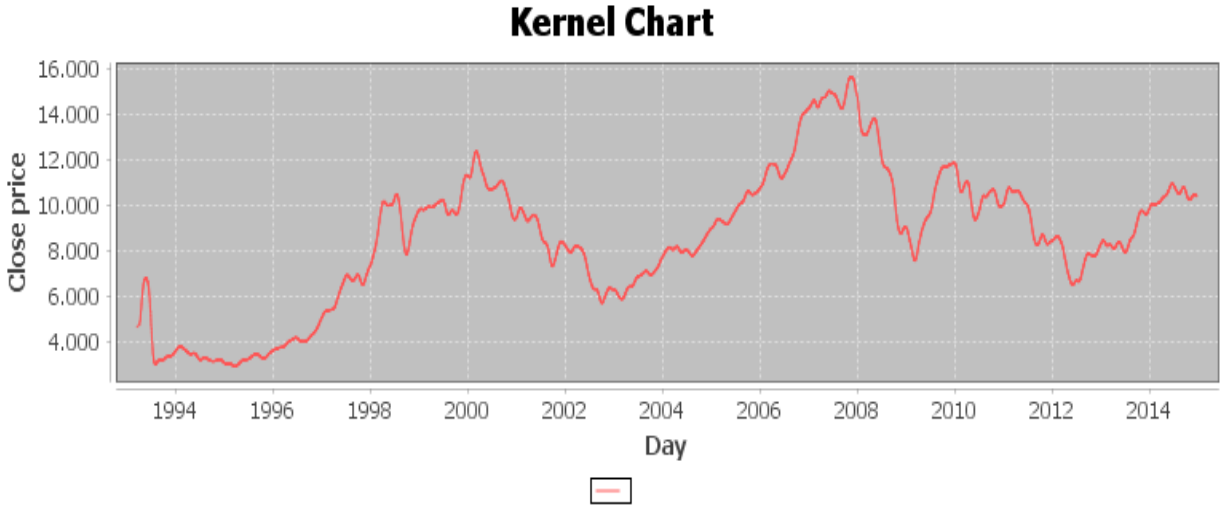


Figura 2.13. Suavizado Gaussiano. Gráfica suavizada.

4. Algoritmo genético

Esta primera versión del algoritmo genético busca encontrar el individuo que, utilizando los métodos analíticos mencionados anteriormente y utilizándolos conjuntamente de manera determinada, gane la mayor cantidad de dinero al invertir sobre una única lista de cotizaciones, es decir, sobre el Ibex35, o el Nikkei, o el Nasdaq, pero únicamente sobre uno, de tal manera que sólo ha de decidir cuándo comprar y cuándo vender, sin tener la necesidad de elegir sobre qué índice invertir. La versión que gestiona varios valores se verá en la siguiente versión.

El modo de reproducir los individuos entre sí y la simulación de la Bolsa está basada en una implementación preliminar que realizó el profesor Ismael Rodríguez Laguna antes de iniciar el presente proyecto.

4.1 Individuo

El objetivo del algoritmo genético es buscar la combinación de parámetros de los métodos analíticos que consiga ganar la mayor cantidad de dinero, pero sólo con estos parámetros no sirve; también son necesarios varios parámetros que decidan a partir de qué umbral los pesos de las señales de compra y venta son efectivos y se tienen que tomar en cuenta y parámetros que representen cuántos días es conveniente estar sin lanzar una señal de compra y venta, estos parámetros se agrupan en una clase llamada Parámetros_control.

Con esto, el Individuo tendrá el siguiente cromosoma (Figura 2.14):

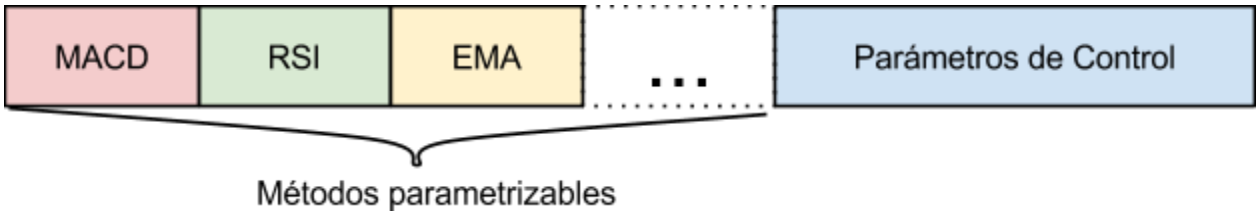


Figura 2.14. Cromosoma del Individuo.

Como se explicó anteriormente, los métodos parametrizables están constituidos por parámetros y cálculos, así que también se puede ver al Individuo como una sucesión de parámetros.

En cuanto a los métodos, la clase Individuo dispone de los setters y los getters usuales, métodos para calcular la función de fitness, y getters y setters especiales que ayudan a generalizar la gestión de los distintos parámetros, ya sean éstos de Control o pertenecientes a Métodos.

4.2 Población

La clase Población representa el conjunto de individuos que formará parte del algoritmo genético y contiene métodos para cruzar los individuos entre sí, ordenar los individuos según su fitness y descartar los peores individuos.

Contiene atributos que permiten modificar el número máximo de individuos de la población, y el número inicial de individuos de la población.

El cruce entre los individuos utilizado es el típico de los algoritmos genéticos: se genera una decisión probabilista para decidir cuáles individuos se cruzan, luego se genera otra para decidir el punto de corte, y luego se genera otra decisión probabilista para decidir si hay mutaciones.

4.3 Algoritmo genético

La clase Algoritmo_genetico se encarga de iterar un número determinado de veces y llamar a Población para que cruce los individuos, también se encarga de añadir nuevos individuos a la población tras cada iteración, y tras finalizar todas las iteraciones devuelve el mejor individuo.

4.4 Simulación

La función de fitness de los individuos consiste en calcular cuánto dinero hubiera conseguido el inversor en un determinado conjunto de días invirtiendo en un índice en concreto tal y como indica el cromosoma del individuo. Como tal, es necesario elaborar una simulación de la Bolsa.

La simulación elaborada no tiene en cuenta el peso que pueden tener en el mercado las compras y las ventas de acciones del individuo que esté generando la simulación, ya que, teniendo en cuenta el gran volumen de compra y venta de cada día particular se considera despreciable el peso que tienen las operaciones del individuo.

La estrategia que se sigue para decidir si se invierte o no en un día determinado es la siguiente:

$$Si\ (\ \Sigma\ pesos\ \leq\ -UMBRAL\ DISPARAR\ SEÑAL\ )\ entonces$$
$$Lanzar\ señal\ de\ Venta$$
$$Sino\ Si\ (\ \Sigma\ pesos\ \geq\ UMBRAL\ DISPARAR\ SEÑAL\ )\ entonces$$
$$anzar\ L\ señal\ de\ Compra$$
$$Si\ (\ Lanzada\ señal\ de\ V\ enta\ )\ entonces$$
$$DiasSinSeñalDeCompra\ ++$$
$$Si\ (\ Lanzada\ señal\ de\ Compra\ )\ entonces$$
$$DiasSinSeñalDeV\ enta\ ++$$

Si ($SeñalDeCompra \wedge$ ($DiasSinSeñalDeCompra \geq UMBRAL\ NO\ ESPERA\ EN\ COMPRA$))
$Comprar()$
$Sino$
$ComprarLuego = true$
Si ($SeñalDeVenta \wedge$ ($DiasSinSeñalDeVenta \geq UMBRAL\ NO\ ESPERA\ EN\ VENTA$))
$Vender()$
$Sino$
$VenderLuego = true$
Si ($ComprarLuego \wedge$ ($DiasSinSeñalDeCompra \geq UMBRAL\ ESPERA\ EN\ COMPRA$))
$Comprar()$
$Sino\ Si$ ($VenderLuego \wedge$ ($DiasSinSeñalDeVenta \geq UMBRAL\ ESPERA\ EN\ VENTA$))
$Vender()$

Al comprar, el individuo utiliza todo el dinero que posee, y al vender, vende todas las acciones que había comprado. A la cantidad de dinero gastada se le aplican los cánones e impuestos correspondientes explicados más tarde en la memoria.

Por supuesto, todos los umbrales que se utilizan forman parte de los parámetros de control que debe ajustar el algoritmo genético.

5. Algoritmo genético avanzado

Esta versión del algoritmo genético busca encontrar al individuo que gane más dinero al tratar de invertir en varios índices a la vez, teniendo que tomar la decisión de si debería comprar o vender acciones de cada uno de los índices que maneje y cuánto dinero debería gastar en cada uno.

Dado que los únicos cambios respecto al algoritmo genético anterior están en la clase Individuo y en Simulación, serán los únicos apartados que se explicarán en este apartado.

5.1 Individuo

Dado que ahora se manejan N índices, el individuo se adaptará a estos N índices, de tal manera que mantiene N listas de métodos parametrizables y N parámetros de control (Figura 2.15).

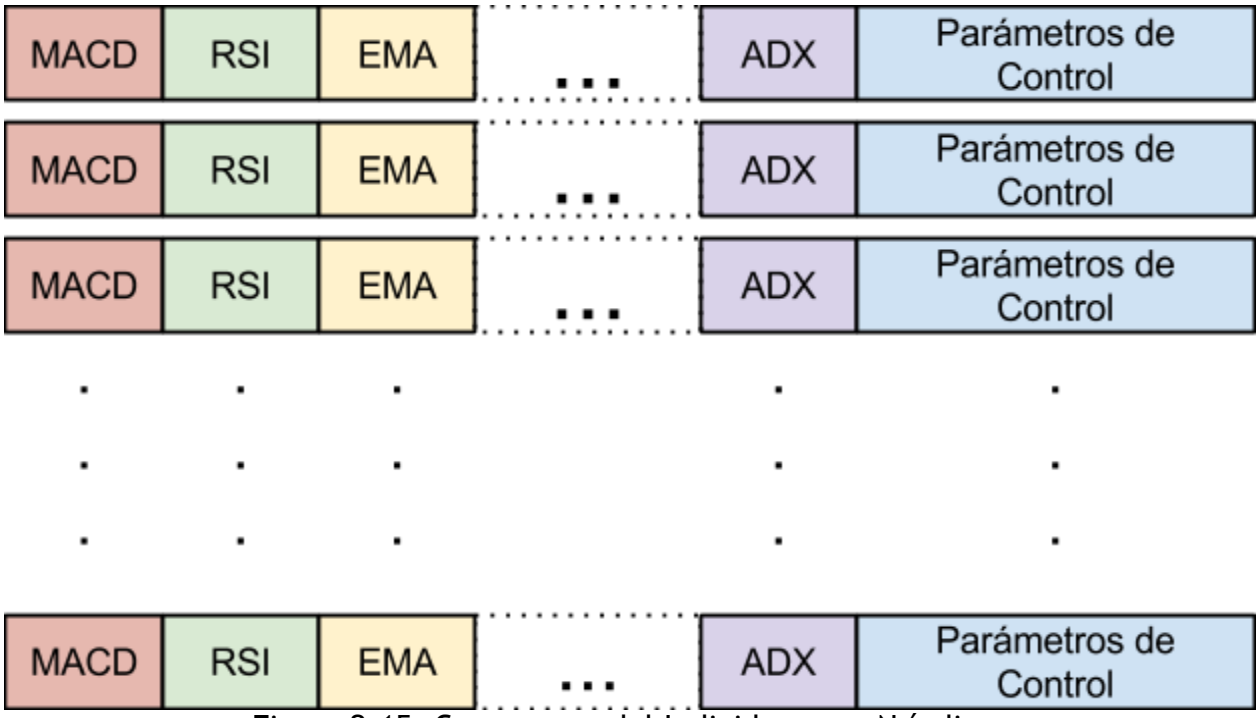


Figura 2.15. Cromosoma del Individuo para N índices

La relación entre cada lista de métodos parametrizables y parámetros de control es 1 a 1 con respecto a los índices. Cada método parametrizable de esa lista se utiliza para calcular los pesos de un índice en concreto, mientras que los parámetros de control fijan los umbrales para lanzar las señales de compra y venta y para indicar cuánto dinero invertir en cada uno de los índices.

Para facilitar el manejo del cromosoma del individuo, en la implementación se tomó la decisión de concatenar todas las listas y manejar una sola lista.

5.2 Simulación

Debido a que el manejo de varios indicadores es más complejo que el de uno solo, se decidió simplificar el lanzamiento de señales de compra y venta; ahora en vez de utilizar períodos de tiempo en los que no se han lanzado señales para decidir cuándo comprar y vender, se compara la suma del peso de los indicadores directamente con los umbrales del individuo, y se compara si los otros índices dan señales de compra o venta para poder comprar y vender de un índice que no da señales muy fuertes.

De esta forma la estrategia de compra y venta para cada índice queda:

$\forall$ *índice hacer*

Si (($\Sigma\, pesos[índice] \leq -UMBRAL\ V\ ENTA\ FUERTE$) $\vee$ $ventaDébil(\,)$) $entonces$

$Vender(\,)$

$Sino\ Si$ (($\Sigma\, pesos[índice] \geq UMBRAL\ COMPRA\ FUERTE$) $\vee$ $compraDébil(\,)$) $entonces$

$Comprar()$

Donde:

$ventaDebil(\,) \rightarrow (\Sigma\, pesos[índice] \leq -UMBRAL\ VENTA\ DÉBIL) \vee$
$(\Sigma\, pesos(\, índices - \{índice\}\,) \geq UMBRAL\ COMPRA\ CONJUNTO)$

$compraDebil(\,) \rightarrow (\Sigma\, pesos[índice] \geq UMBRAL\ COMPRA\ DÉBIL) \vee$
$(\Sigma\, pesos(\, índices - \{índice\}\,) \leq -UMBRAL\ COMPRA\ CONJUNTO)$

En esta simulación ya no se puede gastar todo el dinero en comprar acciones, hay que decidir la cantidad a invertir en cada índice:

$CantidadAComprar$
$= dinero * CANTIDAD[índice] - aplicaImpuestos(dinero * CANTIDAD[índice])$

$CANTIDAD\ \varepsilon\ [0, 1]$

Basta repetir esto para cada índice y aplicar los impuestos anuales.

6. Otros algoritmos de optimización

6.1 Algoritmo de enjambre de partículas (PSO).

El algoritmo de enjambre de partículas, o PSO por sus siglas en inglés (Particle Swarm Optimization), es un algoritmo meta-heurístico de optimización basado en el comportamiento de los enjambres de insectos. Es descrito por primera vez por Kennedy and Eberhart [10] . Se establecen una cantidad de "partículas" que se sitúan dentro del espacio de búsqueda en función de su fitness y que se van moviendo a lo largo del espacio de soluciones siguiendo los movimientos que describen las partículas que los rodean tratando de encontrar una solución óptima para el problema.

El objetivo es el mismo que con el algoritmo genético antes citado: encontrar el individuo (en éste caso partícula) que utilizando los métodos analíticos de manera conjunta y determinada obtenga la mayor cantidad de beneficios sobre una lista de cotizaciones.

6.1.1. Librería SwarmOps Java

Para la implementación del algoritmo de enjambre de partículas se ha utilizado la librería SwarmOps, una librería escrita en lenguaje Java que implementa varios algoritmos de optimización.

Para ser utilizada es necesario implementar la clase que representa el problema, que extiende a la clase Problem. En ésta se establecen los parámetros del problema, la cantidad de parámetros (o dimensión) de las partículas, la función que calcula el fitness de las partículas, los rangos entre los que se encuentran los diferentes parámetros y los métodos auxiliares necesarios para la adaptación y reutilización del código existente.

La clase Optimizer describe el optimizador que se utilizará, y es donde se relaciona el problema con el optimizador. En concreto, para el PSO es necesario definir unos parámetros que condicionan el comportamiento de las partículas. La propia librería incluye una variedad de combinaciones de estos parámetros previamente definidos e indicados para según qué optimizaciones, en función de la dimensión del problema y del número de iteraciones que se realizan.

SwarmOpsJava busca minimizar la función de fitness, así que ha sido necesaria una adaptación del comportamiento de algunas funcionalidades con respecto a la implementación del algoritmo genético.

6.1.2. Implementación

Para la implementación de éste se han reutilizado partes de código del desarrollo del algoritmo genético tales como la clase Individuo (que en este caso representa las diferentes partículas), la implementación de los Métodos Analíticos y la parte de Simulación de inversión (que representa la función de fitness) con la adaptación necesaria para poder ser utilizada por la librería. Se han utilizado también las clases correspondientes a las implementaciones de los impuestos. Ha sido necesario añadir métodos y funciones a las clases anteriormente mencionadas para adaptarlas a las necesidades de la librería utilizada.

6.2. Algoritmo de evolución diferencial.

Este algoritmo está basado al igual que el algoritmo genético en computación evolutiva, pero muestra diferencias significativas con él a la hora de realizar la búsqueda en el espacio de soluciones que se pensó que podrían mejorar los resultados obtenidos.

Se ha implementado también mediante la librería SwarmOpsJava, que ha permitido reutilizar el trabajo de adaptación realizado para el desarrollo del PSO.

Sobre éste algoritmo se han realizado las pruebas más básicas, obteniendo resultados muy parecidos a los otros dos y considerándose por tanto poco relevante a la hora del estudio del problema.

7. Impuestos

En la Bolsa hay varios tipos de impuestos que han sido distribuidos en diferentes clases.

7.1 Canon

Son los impuestos que se aplican en cada operación de compra y venta. Se ha implementado una clase abstracta Canon de la que heredan los tres tipos de cánones y una clase Cánones que los almacena y se encarga de aplicarlos (Figura 2.16).

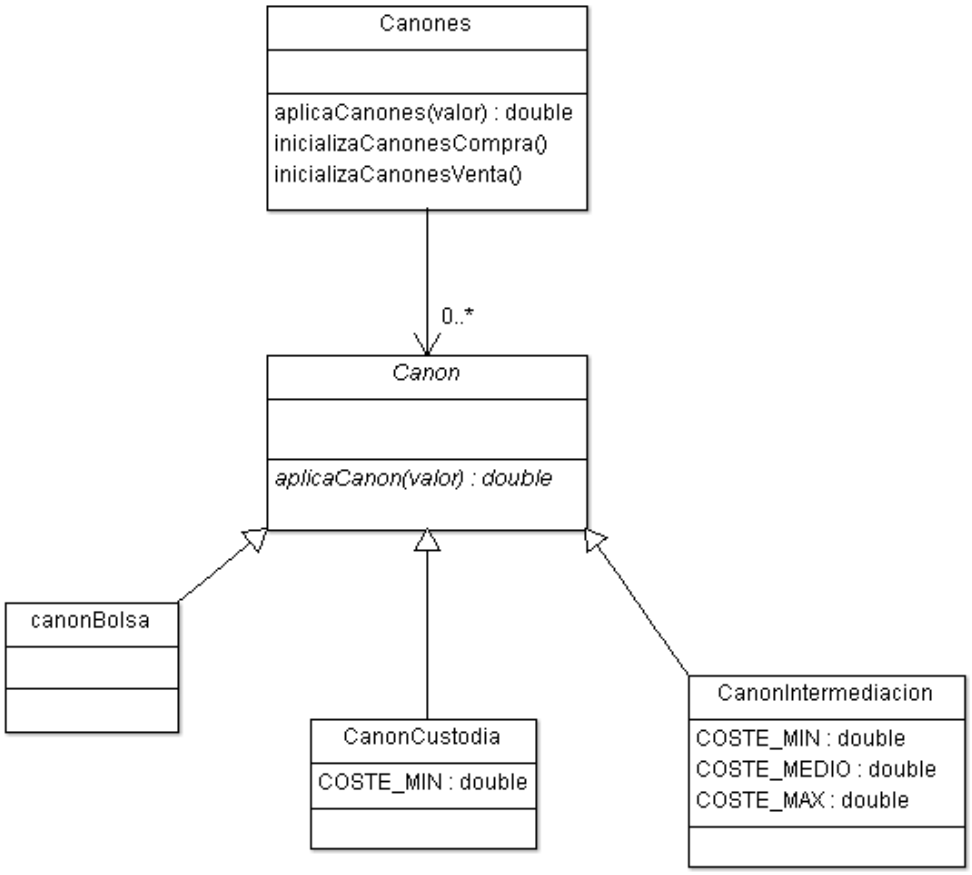


Figura 2.16. Estructura cánones

7.1.1 Canon Bolsa

Es un impuesto que depende del valor efectivo de la operación, como puede verse en la siguiente tabla. Se aplica en cada compra y venta de acciones.

Valor	≤ 300	$> 300\ y \leq 3000$	$> 3000\ y \leq 35000$	$> 35000\ y \leq 70000$	$> 70000\ y \leq 140000$	> 140000
Impuesto	1.10	2.45 + 0.024%	4.65 + 0.012%	6.40 + 0.007%	9.20 + 0.003%	13.40

7.1.2 Canon Custodia

Impuesto a pagar por la venta de acciones. Su valor depende del intermediario, por lo que se utiliza el coste mínimo más el IVA.

7.1.3 Canon Intermediación

Porcentaje sobre el valor de la operación que cobra el intermediario en cada compra y venta de acciones. Para aplicarlo se decidió utilizar la media entre el porcentaje máximo y el porcentaje mínimo que los intermediarios suelen aplicar en Madrid.

7.2 Impuesto Anual:

Son los impuestos que hay que pagar una vez al año. Se ha implementado una clase abstracta ImpuestoAnual de la que hereda el único impuesto anual aplicado y una clase ImpuestosAnuales que lo almacena y se encarga de aplicarlo (Figura 2.17).

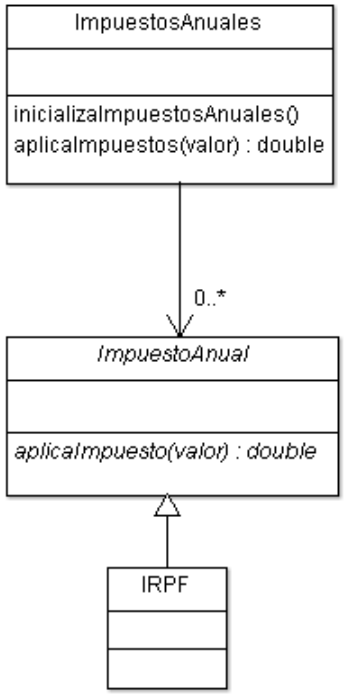


Figura 2.17. Estructura impuestos anuales

7.2.1 IRPF

Se aplica cada 365 iteraciones (imitando 365 días) y depende de las ganancias obtenidas en esas 365 iteraciones, como puede verse en la siguiente tabla:

Valor	≤ 600	$> 600\ y \leq 50000$	> 50000
Impuesto	20%	22%	24%

7.3 Mantenimiento Acciones

Impuesto a pagar por la posesión de acciones.Se trata de un porcentaje aplicado sobre el valor de las acciones y forma parte del canon de custodia, pero al aplicarse de forma diferente se optó por ponerlo en una clase aparte.

Depende del banco, por lo que para aplicarlo se calculó la media de los porcentajes anuales de diferentes bancos. Generalmente, este impuesto se suele cobrar cada tres o cada seis meses, pero en nuestra simulación no tiene sentido aplicarlo de dicha forma, por lo que para que siga dependiendo del tiempo durante el que se han tenido las acciones, se calcula el porcentaje diario y se multiplica por el

número de días de posesión de las acciones antes de aplicarlo, cada 30 iteraciones (imitando 30 días).

7.4 IVA

Se utiliza para calcular el canon de custodia en la clase CanonCustodia.

8. Carga y escritura de datos

Encontramos que los mejores datos disponibles sobre los índices los ofrece Yahoo Finance, por lo que todos los datos utilizados provienen de esa página web. Ya que los datos están en formato .xls para Excel de Microsoft se buscó una librería que permitiera leer los ficheros .xls y cargarlos en estructuras de Java. La librería escogida fue JExcelApi.

Tanto la carga como la escritura de datos se realiza desde el paquete cargaDatos. La carga en la clase leeExcel y la escritura en la clase exportaExcel. La estructura generada por la carga de datos sigue la misma estructura que la clase Cotizaciones ya explicada anteriormente. Los datos escritos por la aplicación corresponden al cálculo de los métodos analíticos utilizando parámetros por defecto.

9. Vistas

El objetivo principal del proyecto era experimentar si los indicadores y los métodos analíticos son útiles para la inversión en bolsa, por lo tanto, la interfaz gráfica desarrollada tiene como objetivo facilitarnos esta tarea.

9.1 Gráficas

La funcionalidad principal que se decidió que ofreciera la interfaz es la representación gráfica de las cotizaciones de la bolsa y de los distintos indicadores.

Para la representación de las gráficas se utiliza la librería jFreeChart, que se encarga de formar una gráfica dados una serie de puntos.

Como cada indicador genera una gráfica distinta, se diseñó una jerarquía de herencia para facilitar la implementación (Figura 2.18).

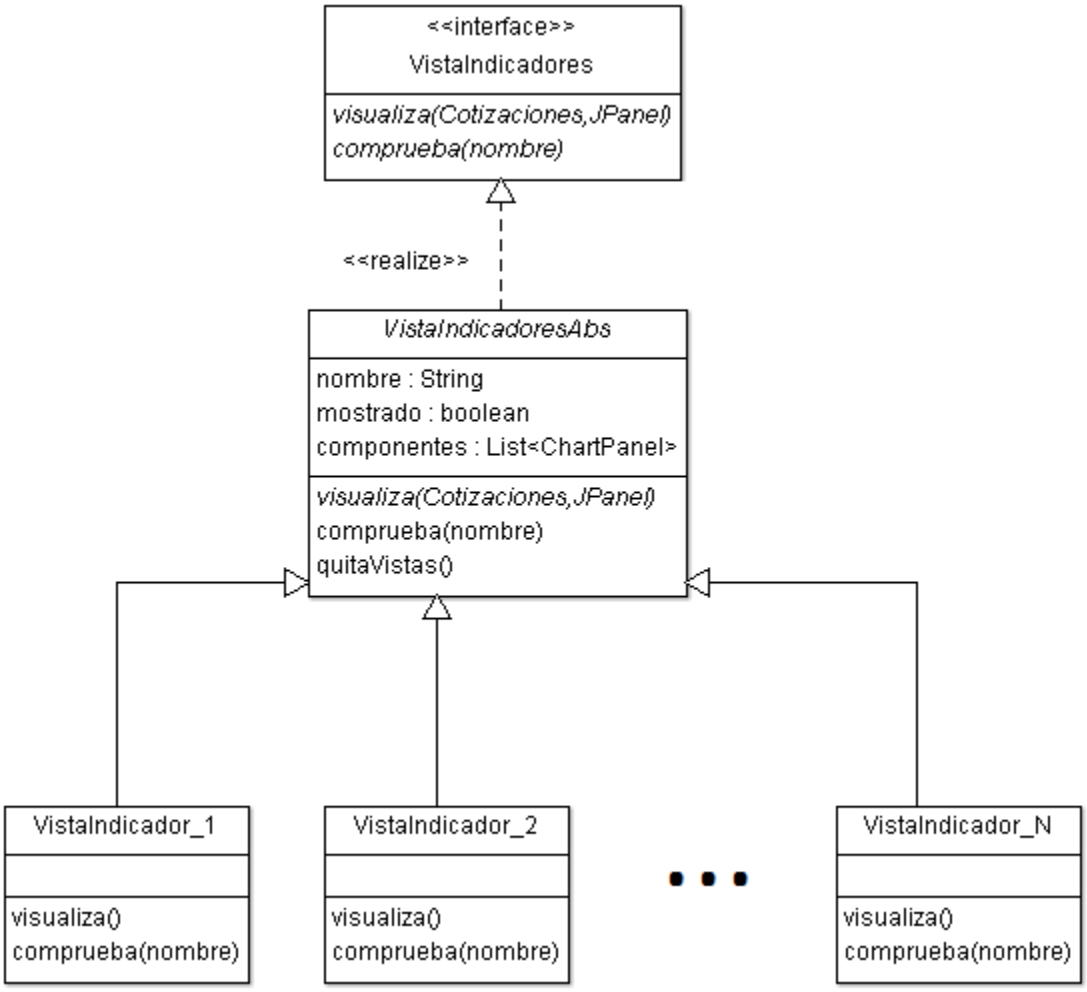


Figura 2.18. Estructura vistas

La interfaz VistaIndicadores contiene dos métodos abstractos: visualiza, que muestra las cotizaciones en forma de gráfica en un JPanel; y comprueba, que recibe un nombre y si el nombre del indicador coincide con el parámetro del método devuelve true.

La clase abstracta VistaIndicadoresAbs implementa comprueba y quitaVistas y deja el método *visualiza* como abstracto para que los implementen las vistas de cada indicador concreto.

9.2 Menús

Toda la funcionalidad ofrecida por el programa que no implique la interacción con gráficas se realiza a través de los menús desplegables superiores de la ventana principal.

Para facilitar el control de los eventos generados al pulsar sobre las opciones del menú, se decidió aplicar el patrón Modelo-Vista-Controlador. Existe una interfaz Vista con un método fijarControlador(EventListener controlador) que implementa la clase Menu y que asigna el manejador de eventos sobre cada botón del menú.

Cada opción seleccionable de los menús dispone de un nombre característico que se utiliza como identificador para saber qué evento generar.

9.3 Controlador

La clase Controlador captura los eventos generados por la interfaz gráfica y actúa según el componente que generó el evento. Hace de puente entre la interfaz gráfica y el modelo lógico (Figura 2.19).

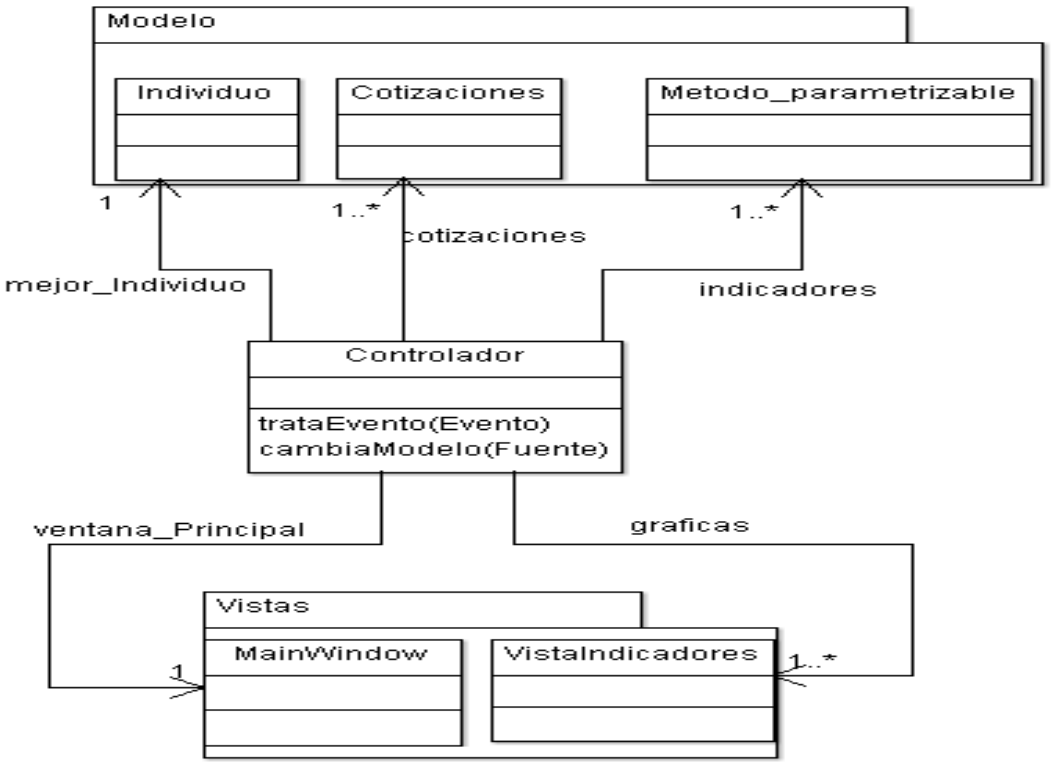


Figura 2.19. Estructura controlador

10. Estudio de licencias

En este apartado realizaremos un estudio de las diferentes licencias que poseen las distintas librerías utilizadas y de cómo pueden afectar estas licencias a la licencia que pudiera tener nuestro proyecto.

Las distintas librerías utilizadas son:

- JCommon v1.0.23 : librería utilizada por JFreeChart, con licencia GNU Lesser General Public License v2.1;
- JFreeChart v1.0.19 : librería utilizada para la representación de gráficas, con licencia GNU Lesser General Public License v2.1;
- JExcelAPI v2.6.12 : librería utilizada para leer y exportar datos en formato compatible con Microsoft Excel, con licencia GNU Lesser General Public License v2.0;
- JavaOctave v0.6.1 : librería utilizada para la conexión a Octave desde Java, con licencia Apache 2.0;
- Commons-logging v1.1.1 : librería utilizada por JavaOctave, con licencia Apache 2.0.
- SwarmOps v1.0 :librería utilizada para los algoritmos PSO y . Tiene una licencia propia libre.

La licencia GNU Lesser General Public License no afecta para nada a la licencia que se podría poner al software mientras no se hicieran modificaciones del código de las librerías con esta licencia. Si así se hiciera, este software debería tener una licencia *copyleft* de GNU que decidiéramos nosotros.

La licencia Apache 2.0 no es una licencia *copyleft* y no afecta a la licencia del software, sólo requiere que se incluya un archivo de texto que indique qué partes del código tienen licencia Apache e información sobre su origen y autor.

11. Manual de usuario

Como ya se ha mencionado antes, el objetivo de la interfaz gráfica era ayudarnos a realizar el trabajo de investigación, por lo que la funcionalidad de la aplicación es limitada y el prototipo de usuario capaz de utilizar la aplicación necesita conocimiento sobre la bolsa y algoritmos genéticos.

El diseño de la ventana principal (Figura 2.20) está basado en el diseño usualmente utilizado en aplicaciones de escritorio que requieren un nivel alto de interacción por parte del usuario. En la parte superior de la ventana principal hay varios menús desplegables con los que se puede interactuar, y debajo de ellos hay un hueco utilizado para la representación de las gráficas, que al ser el componente de la aplicación con el mayor nivel de interactividad, se sitúan en el centro de la ventana para conseguir captar el centro de atención del usuario.

Figura 2.20. Vista gráfica de la ventana principal

Ahora se procederá a explicar una a una todas las opciones de los menús:

11.1 Menú File

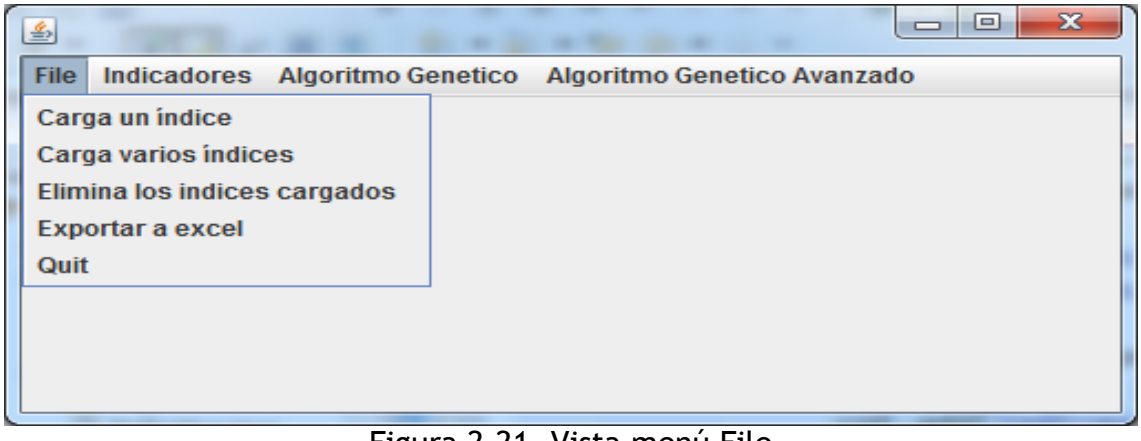


Figura 2.21. Vista menú File

El menú File (Figura 2.21) ofrece tres opciones:

- Carga un índice: al pulsar la opción se abrirá el Explorador de archivos para que el usuario pueda seleccionar un archivo .xls con las cotizaciones que interesa cargar. Estas cotizaciones se utilizarán para el algoritmo genético que funciona con **un solo índice**. El archivo .xls debe tener las filas ordenadas de menor fecha a mayor fecha, y las columnas deben guardar los datos del siguiente modo: la primera debe contener las fechas, la segunda el precio de apertura, en la tercera el precio mínimo alcanzado, en la cuarta el máximo, en la quinta el precio de cierre y en la sexta el volumen. Al cargar los datos correctamente se mostrará un mensaje de confirmación (Figura 2.22):

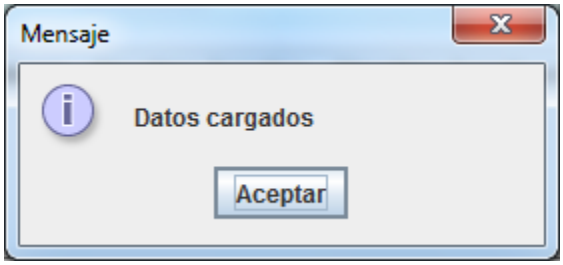


Figura 2.22. Vista mensaje "datos cargados"

- Carga varios índices: al pulsar la opción se abre el Explorador de archivos para seleccionar un archivo .xls con el mismo formato ya explicado antes. Cada índice cargado con esta opción se almacenará, y serán utilizados en conjunto para el algoritmo genético que funciona con **varios índices**. Véase la siguiente opción para eliminar los índices cargados.
- Elimina los índices cargados: al pulsar la opción se eliminan todos los índices que han sido cargados utilizando la opción "Cargar varios índices".
- Exportar a excel: esta opción calcula los indicadores asociados a las cotizaciones cargadas, y los guarda en un archivo "indicadores.xls". Si el archivo ya existe mostrará un mensaje de error (Figura 2.23):

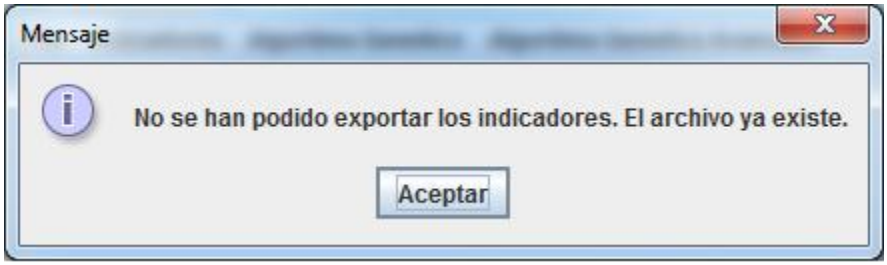


Figura 2.23. Vista mensaje de error al exportar los indicadores

- Quit: cierra la aplicación.

11.2 Menú Indicadores

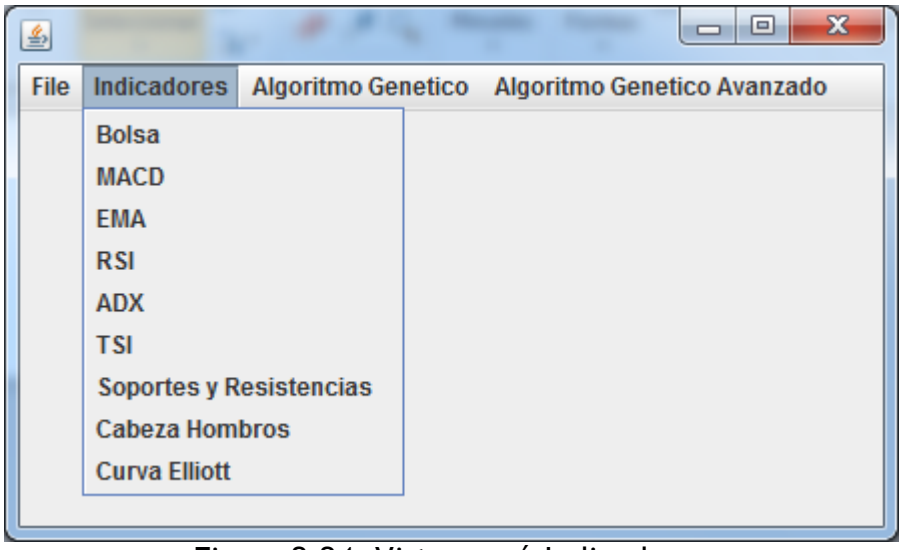


Figura 2.24. Vista menú Indicadores

El menú Indicadores (Figura 2.24) permite que aparezca en pantalla la representación gráfica de los indicadores asociados a las cotizaciones cargadas utilizando la opción del menú File **"Carga un índice"**, que se calculan utilizando parámetros por defecto. Si se quiere que la gráfica desaparezca de la pantalla se deberá pulsar otra vez el indicador que se pulsó para que apareciera inicialmente.

Si no hay datos cargados se mostrará un mensaje de error (Figura 2.25).

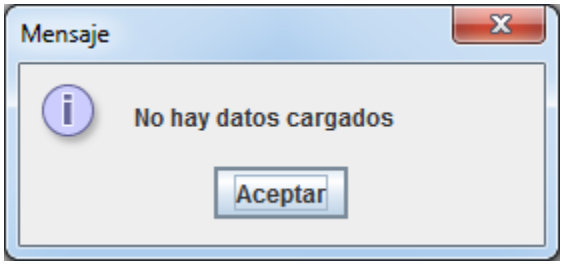


Figura 2.25. Vista error "no hay datos cargados"

Se puede interactuar con las gráficas, de tal forma que se puede hacer zoom en un rectángulo determinado de la gráfica manteniendo pulsado el botón izquierdo del ratón y desplazando el ratón sobre la zona sobre la que se quiere hacer zoom (Figura 2.26).

Figura 2.26. Seleccionar para ampliar la gráfica

El resultado obtenido es el siguiente (Figura 2.27):

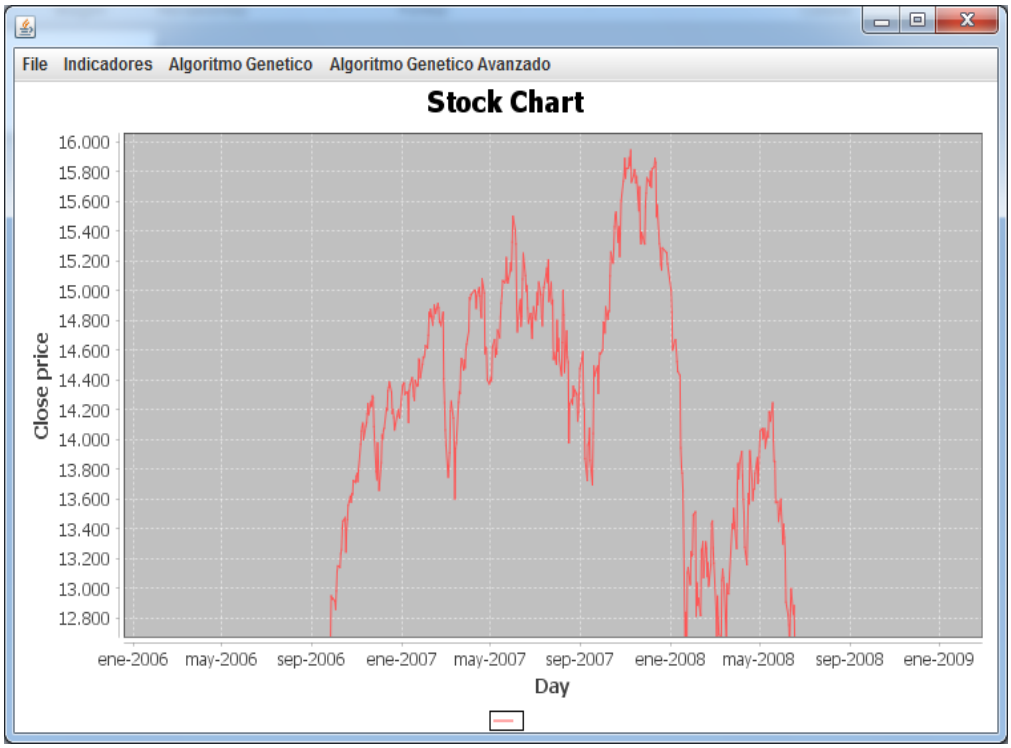


Figura 2.27. Sección de la gráfica ampliada

Al pulsar el botón izquierdo sobre la gráfica aparecerá un menú contextual que permite realizar varias operaciones sobre la gráfica (Figura 2.28).

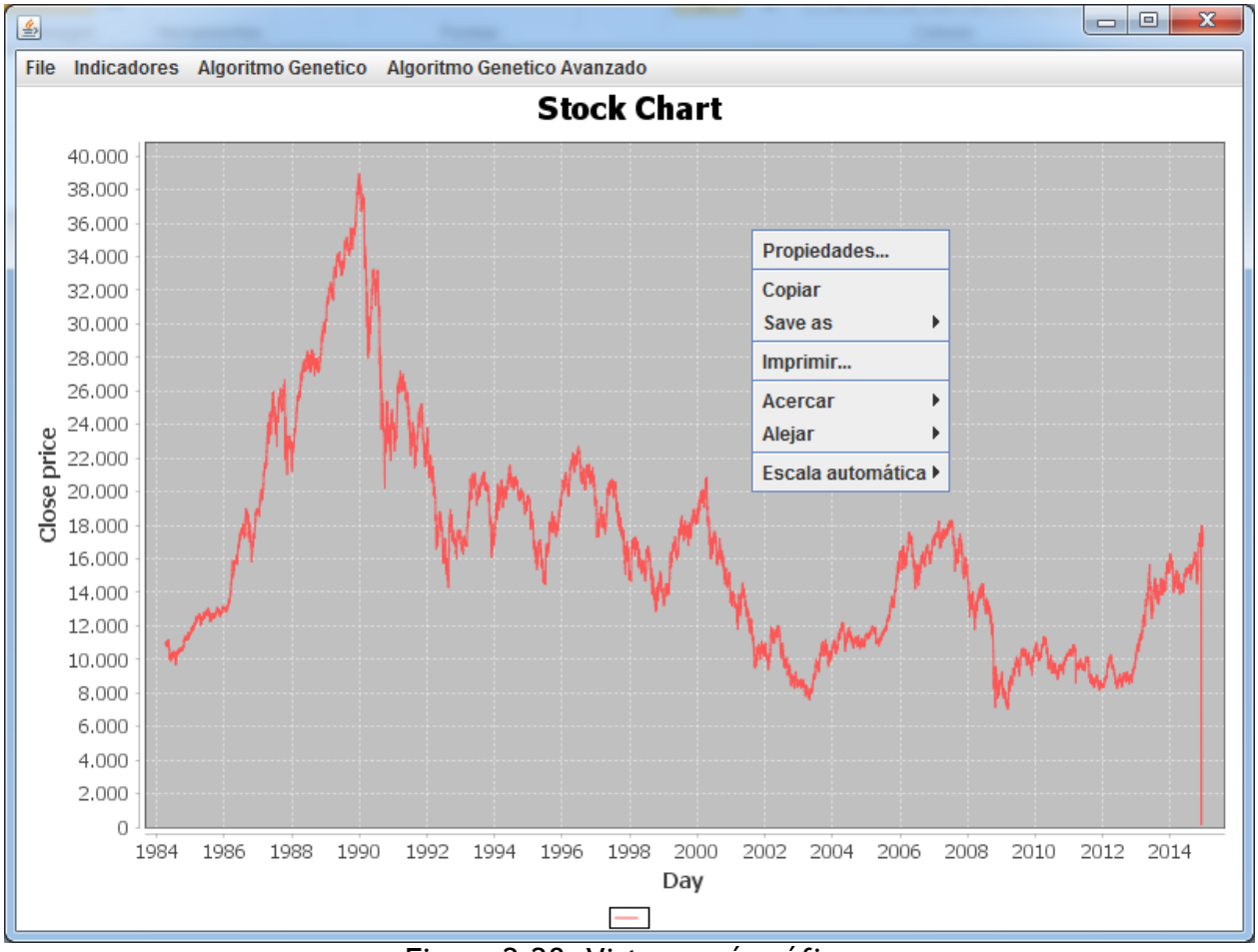


Figura 2.28. Vista menú gráfica

11.3 Menú Algoritmo Genético

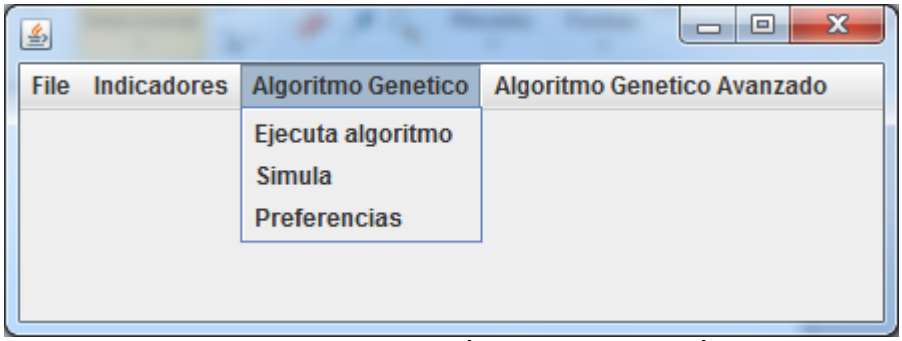


Figura 2.29. Vista menú Algoritmo Genético

El menú Algoritmo Genético (Figura 2.29) tiene las siguientes opciones:

- Ejecuta algoritmo: ejecuta el algoritmo genético con el nº de individuos y el nº de iteraciones especificados en el código en un rango de fechas especificado en Preferencias, sobre las cotizaciones cargadas, es decir, sobre un solo índice. El resultado se muestra en una ventana emergente.
- Simula: calcula las ganancias obtenidas por el mejor individuo encontrado por el algoritmo genético en las cotizaciones cargadas. Nótese que se puede entrenar al algoritmo genético con unas cotizaciones, cargar otras y hacer la simulación con estas últimas. El resultado se muestra en una ventana emergente.
- Preferencias (Figura 2.30): al pulsar sobre preferencias se abre un menú flotante que permite especificar el rango de días con los que se ejecutará el algoritmo genético:

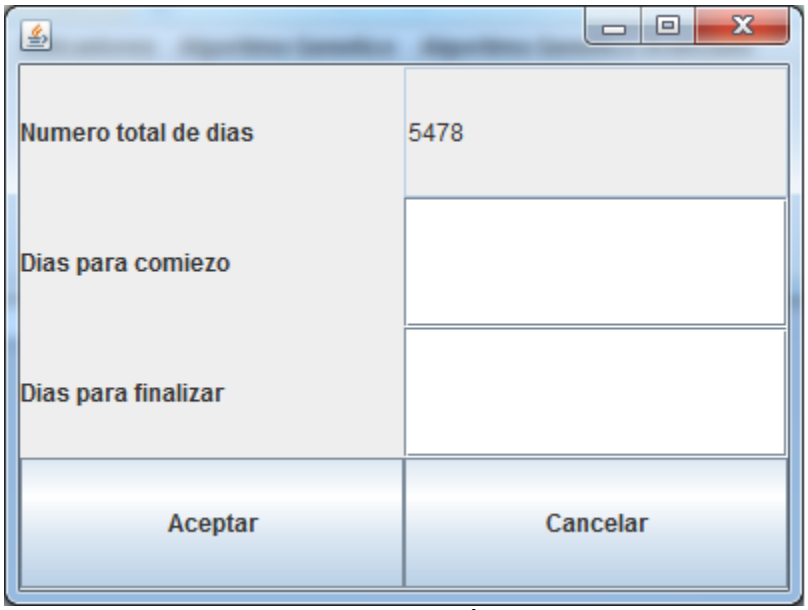


Figura 2.30. Vista menú Preferencias

11.4 Menú Algoritmo PSO

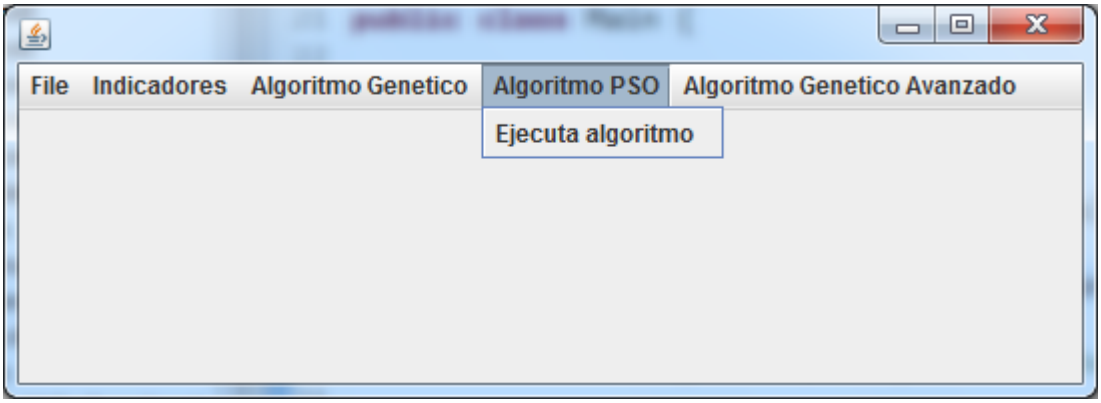


Figura 2.31. Vista menú Algoritmo PSO

- Ejecuta algoritmo: ejecuta el algoritmo PSO con el nº de individuos y el nº de iteraciones especificados en el código en un rango de fechas especificado también en el código, sobre las cotizaciones cargadas, es decir, sobre un solo índice. El resultado se muestra por consola.

11.5 Menú Algoritmo Genético Avanzado

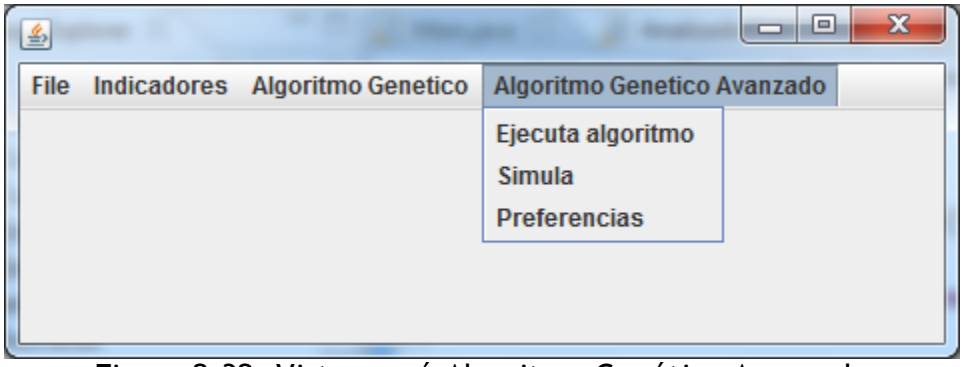


Figura 2.32. Vista menú Algoritmo Genético Avanzado

- Ejecuta algoritmo: ejecuta el algoritmo genético sobre los índices cargados utilizando la opción "Cargar varios índices" con el n.º de individuos y el n.º de iteraciones especificados en el código. El período de tiempo sobre el que se ejecuta depende del rango de los índices cargados. Si por ejemplo hemos cargado tres índices y el primer dato que tenemos de uno de ellos es de una fecha más tardía que el de los demás, el algoritmo comenzará en esa fecha. Ocurre lo mismo con el último dato que tenemos, pero utilizando la fecha más temprana.
- Simula: calcula las ganancias obtenidas por el mejor individuo encontrado por el algoritmo genético avanzado en los índices cargados. Para poder simular es necesario que el individuo se haya entrenado con el mismo número de índices que con los que se quiere realizar la simulación. Si eso no ocurre, se mostrará un mensaje de error (Figura 2.32).

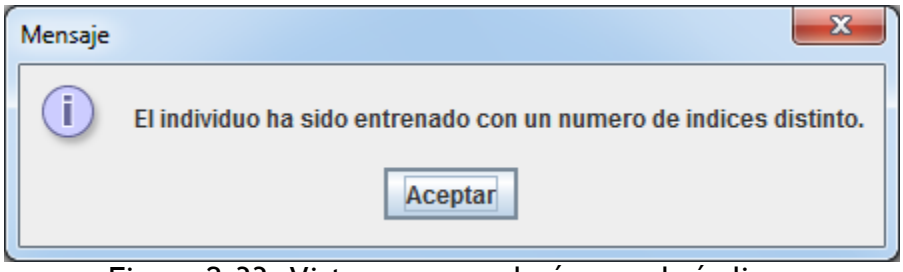


Figura 2.33. Vista error en el número de índices

- Preferencias: similar a las preferencias del algoritmo genético anteriormente mencionado. Sin embargo, hay que tener en cuenta que el número de días a partir del cual se quiere comenzar se contará a partir de la fecha inicial más tardía.

Capítulo 3 - Resultados

En este capítulo se expondrán los resultados obtenidos tras ejecutar los diversos algoritmos y se expondrán los planes de pruebas utilizados.

1. Algoritmo genético simple

El algoritmo genético simple trata de buscar la estrategia que consiga más dinero invirtiendo sólo en un índice dado. Para nuestras pruebas dispusimos de los índices IBEX-35, Nasdaq, Nikkei y London Stock Exchange.

Aunque ya se ha comentado antes, conviene recordar que las ganancias representadas es el cociente $dinero\ inicial\ /\ dinero final$.

El plan de pruebas consiste en entrenar el algoritmo genético un número *n* de veces utilizando un conjunto de distintos números de iteraciones y de tamaños máximos de la población del algoritmo genético, y después probar la estrategia encontrada con los demás índices. Es decir, si tenemos un conjunto de iteraciones y de tamaños: $I = \{i_0,\ i_1,\ \dots,\ i_{m-1}\}$ y $T = \{t_0,\ t_1,\ \dots, t_{m-1}\}$, el algoritmo genético se ejecutará $m \times k \times n$ veces; cada combinación de iteración y tamaño de población se ejecutará *n* veces y se escogerá la mejor solución de estas *n*. Los conjuntos utilizados son: $I = \{50, 100, 500\}$ y $T = \{50, 100, 500\}$, y *n* = 5.

Primero se explicarán los resultados obtenidos utilizando sólo los indicadores básicos RSI, EMA y MACD; después los resultados al utilizar los indicadores anteriores y además los indicadores TSI, ADX y Soportes y Resistencias en un período comprendido entre 1994 y 2014 inclusives ambos para el IBEX, entre 2001 y 2014 para el London Stock Exchange y entre 1984 y 2014 para los demás; al tratarse de intervalos de tiempo que comienzan en períodos de tiempos tan distintos, es más fácil comprobar si la estrategia encontrada es general o no. Así también se puede ver si, a mayor número de días para entrenar al algoritmo, se obtiene un mejor resultado.

Asimismo se realizará otra prueba solo con el RSI, EMA y MACD sin tener en cuenta impuestos y cánones, para ver las diferencias, y después se añadirán los demás indicadores.

Finalmente se realizarán pruebas más pequeñas utilizando conjuntos de entrenamiento y validación totalmente disjuntos.

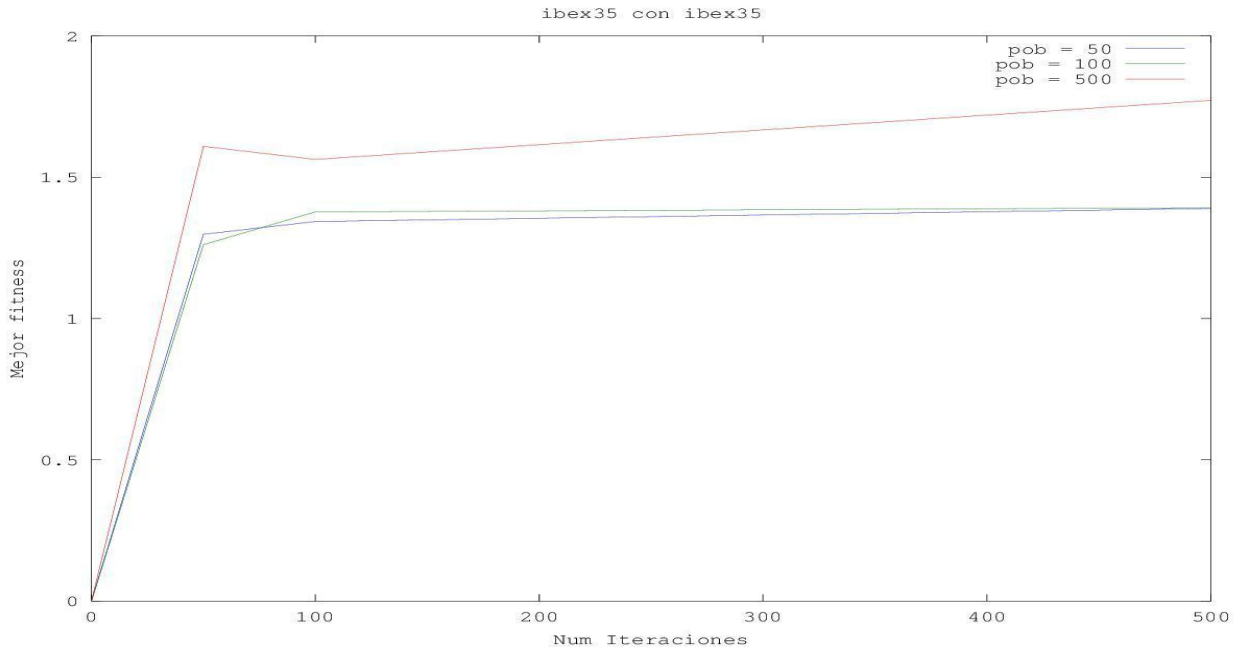
Los resultados se expondrán de manera gráfica para que sea más fácil observar cómo evolucionan las ganancias encontradas en función del número de iteraciones y del tamaño de la población. La cabecera del gráfico tiene el formato:
“ ‘indicador de entrenamiento’ con *‘indicador de prueba’ ”*.

1.1 Usando indicadores básicos durante 8 años

Dada la gran cantidad de gráficas generadas, se utilizarán cuatro subapartados para facilitar la lectura de los resultados. Cada subapartado tendrá como nombre el nombre del índice que se ha utilizado para entrenar al algoritmo genético.

1.1.1 IBEX-35

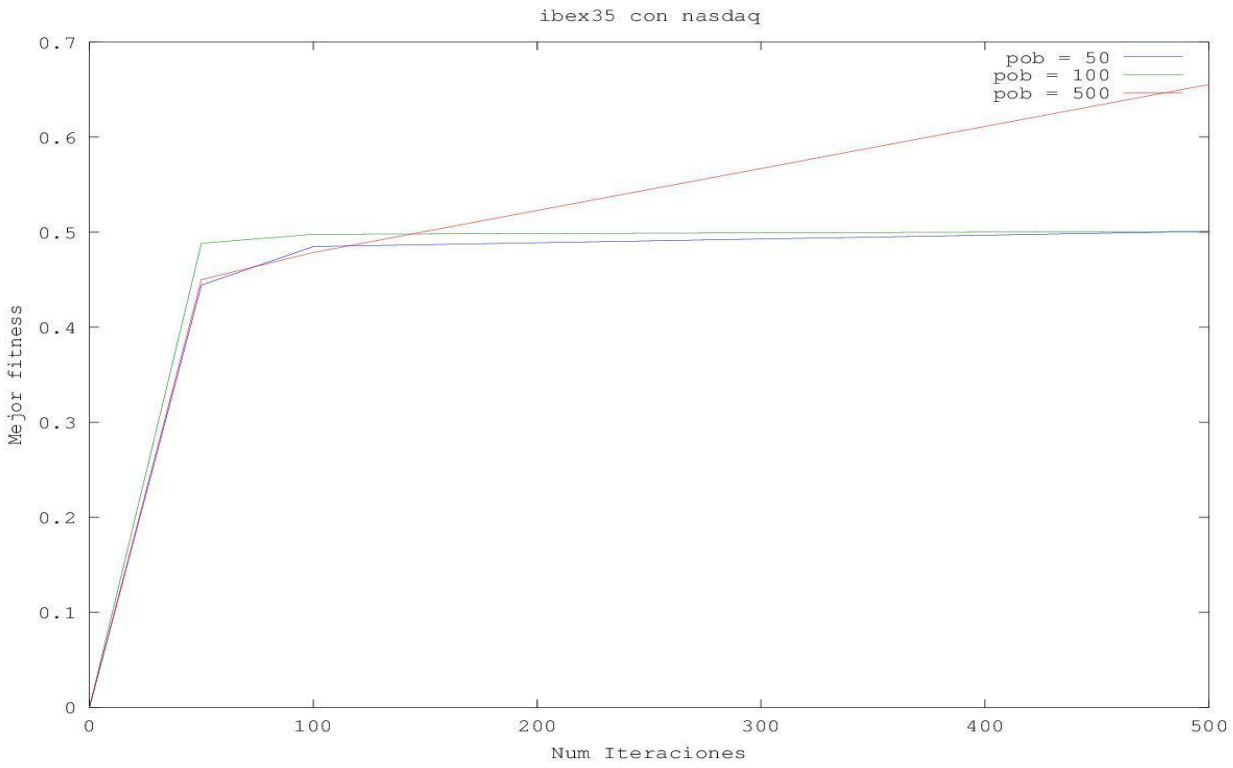
Las siguientes gráficas muestran cómo evolucionan las ganancias encontradas en función del número de iteraciones y de la población[1]:

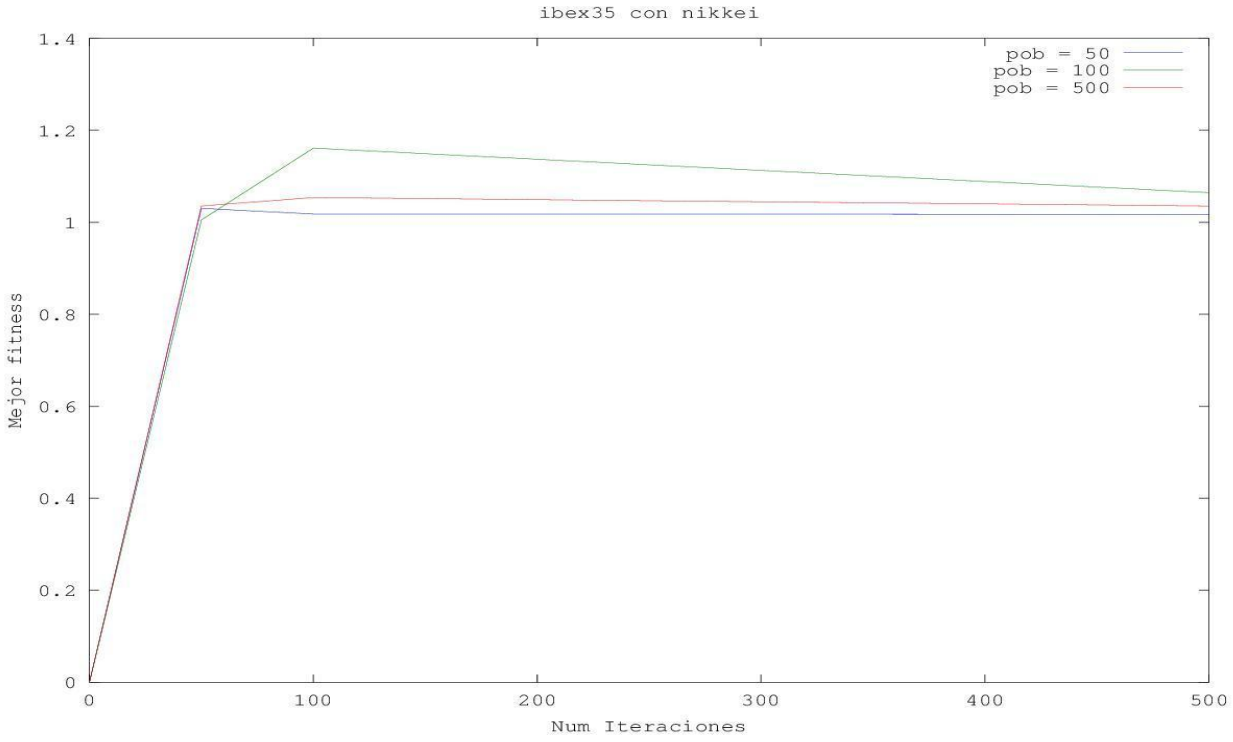


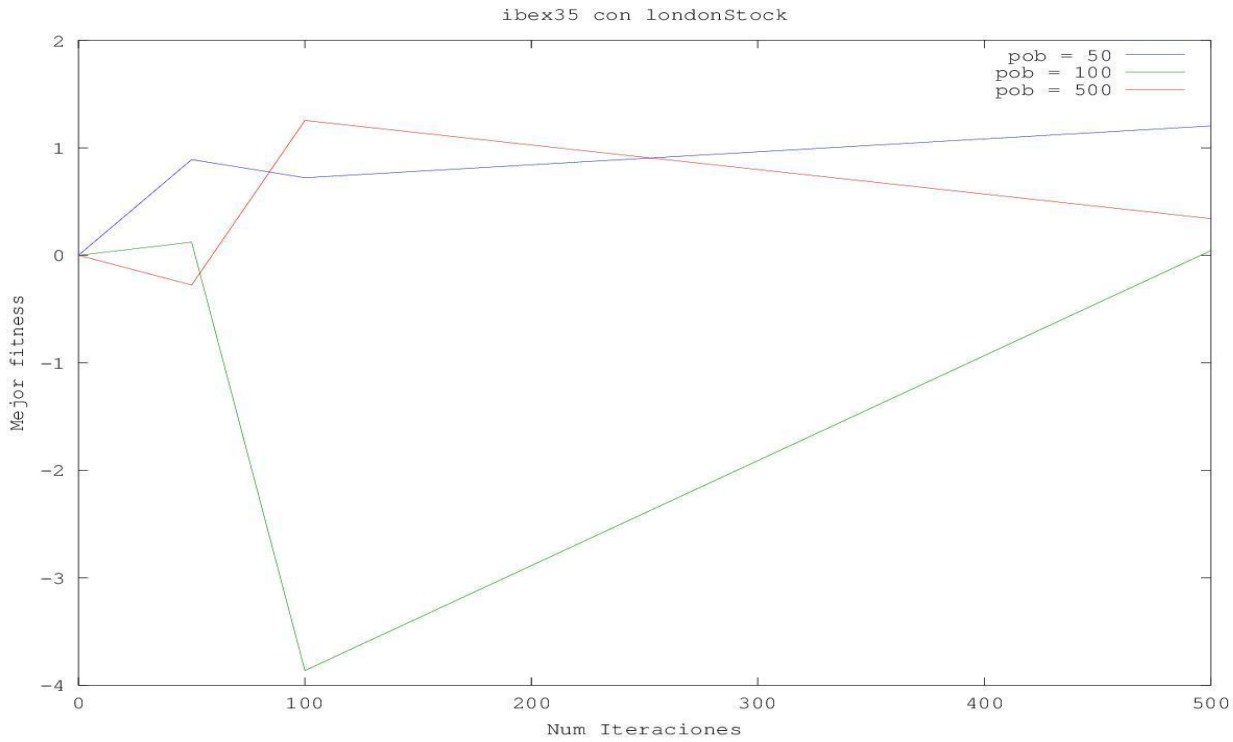
Como se puede observar, a mayor número de iteraciones y mayor tamaño de la población, más aumentan las ganancias, lo cual es un buen indicador temprano de que el algoritmo genético puede ser capaz de resolver el problema de invertir en bolsa utilizando análisis técnico. Se puede ver que a partir de 100 iteraciones o de una población de 100, las ganancias aumentan de manera más lenta.

Ahora se incluirán las gráficas que indican cómo evolucionan las ganancias en función del número de iteraciones al utilizar la estrategia encontrada con el IBEX-35 en el resto de índices.

[1] Figuras 3.1.1-3.1.4. Gráficas del IBEX-35


ibex35 con nasdaq
pob = 50
pob = 100
pob = 500
Mejor fitness
Num Iteraciones
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0
100
200
300
400
500


ibex35 con nikkei
pob = 50
pob = 100
pob = 500
Mejor fitness
Num Iteraciones
0
0.2
0.4
0.6
0.8
1
1.2
1.4
0
100
200
300
400
500

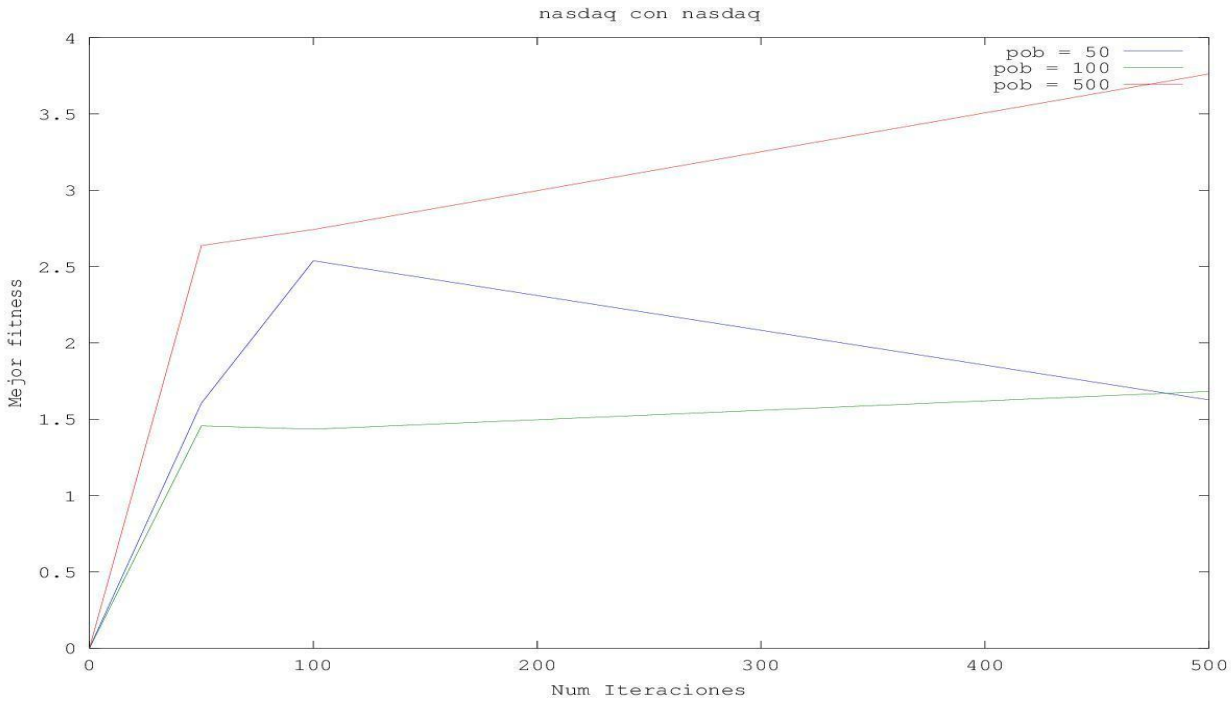


Al probar la estrategia encontrada al usar el IBEX-35 con los otros índices se puede observar como en general se alcanzan las ganancias máximas utilizando entre 50 y 100 iteraciones y 50 y 100 individuos en la población, salvo el caso de la bolsa de Londres que parece mejorar con 500 individuos sin que parezca que se ha producido un sobreajuste de la solución. Se denomina sobreajuste al fenómeno que se produce cuando la solución encontrada por el algoritmo se ha ajustado tanto a los datos que se han usado para entrenar al algoritmo que no funciona para otros datos. Además, paradójicamente, es con este índice con el que se consiguen más ganancias. Las ganancias obtenidas no son muy altas, aunque resultan mejores que utilizar la estrategia de comprar el primer día y vender el último, la cual no resulta en absoluto útil al tener en cuenta impuestos.

1.1.2 Nasdaq

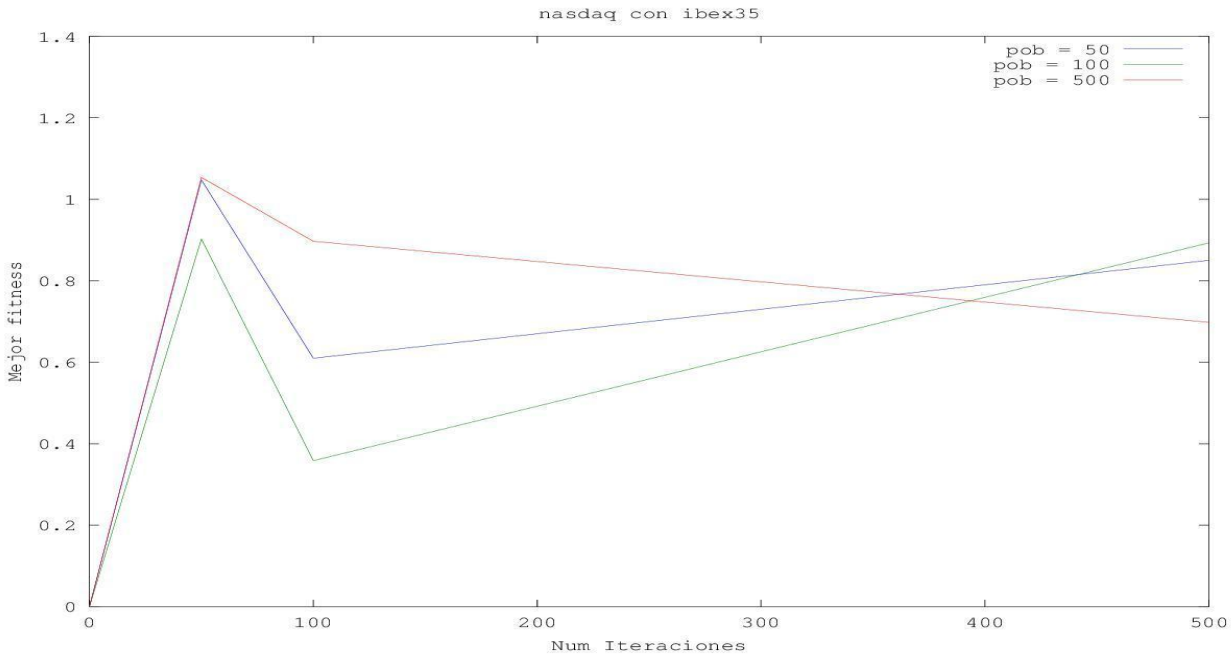
Los resultados se presentarán del mismo modo que se hizo en el apartado anterior, pero volviendo a obviar las gráficas que muestran la evolución del fitness en función del número de individuos por resultar redundantes[2].

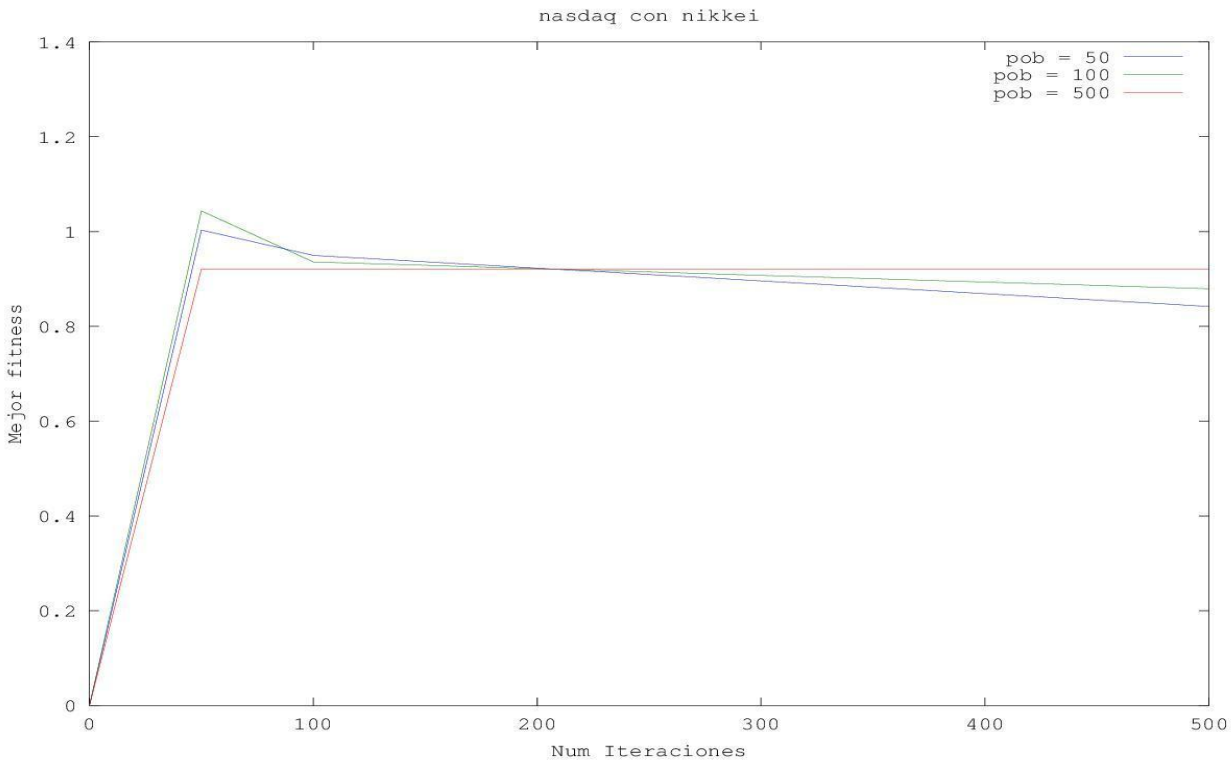
[2] Figuras 3.1.5-3.1.8. Gráficas del Nasdaq

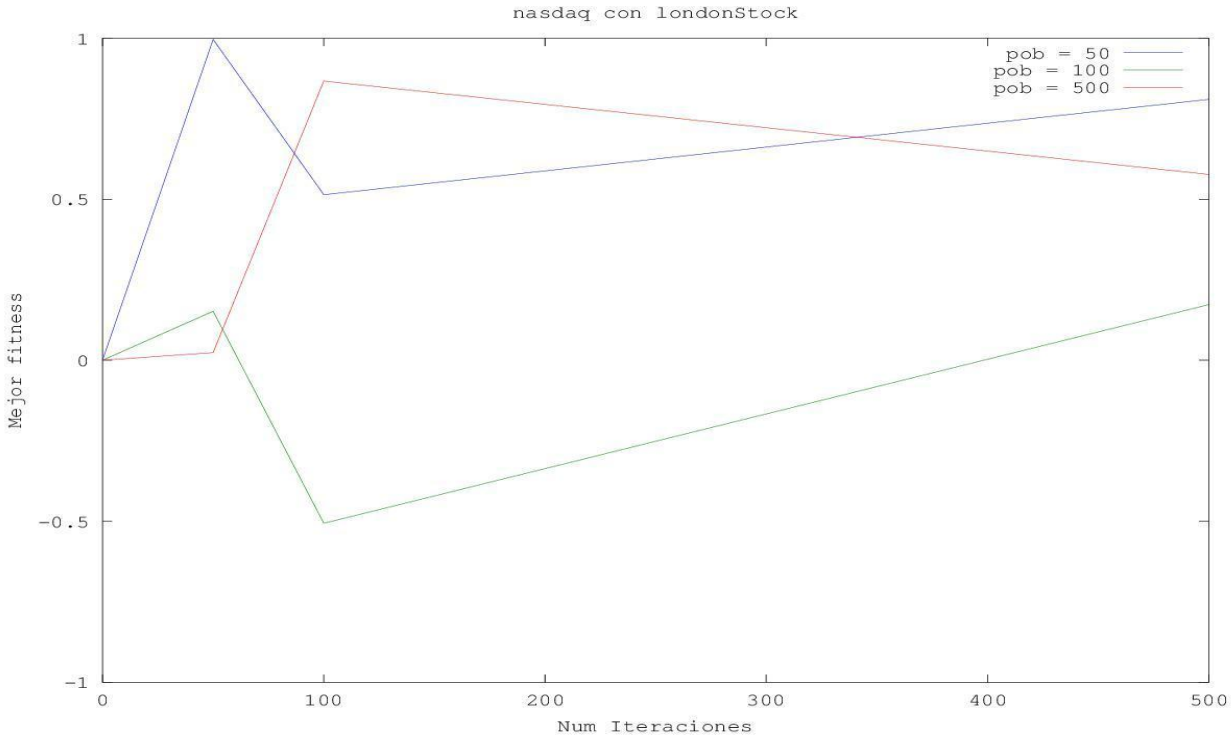


Se puede observar en las gráficas anteriores la misma evolución que en el apartado anterior, aunque en este caso cuando hay 50 individuos en la población se producen unos resultados buenos en cuanto a ganancias pero un tanto extraños en cuanto a la evolución de éstas, pues debería aumentar al aumentar el número de iteraciones pero en el caso de 50 individuos decrecen, quizá por el bajo número de iteraciones y el tan alto número de individuos.

Las siguientes gráficas muestran los resultados de probar las estrategias en otros índices:







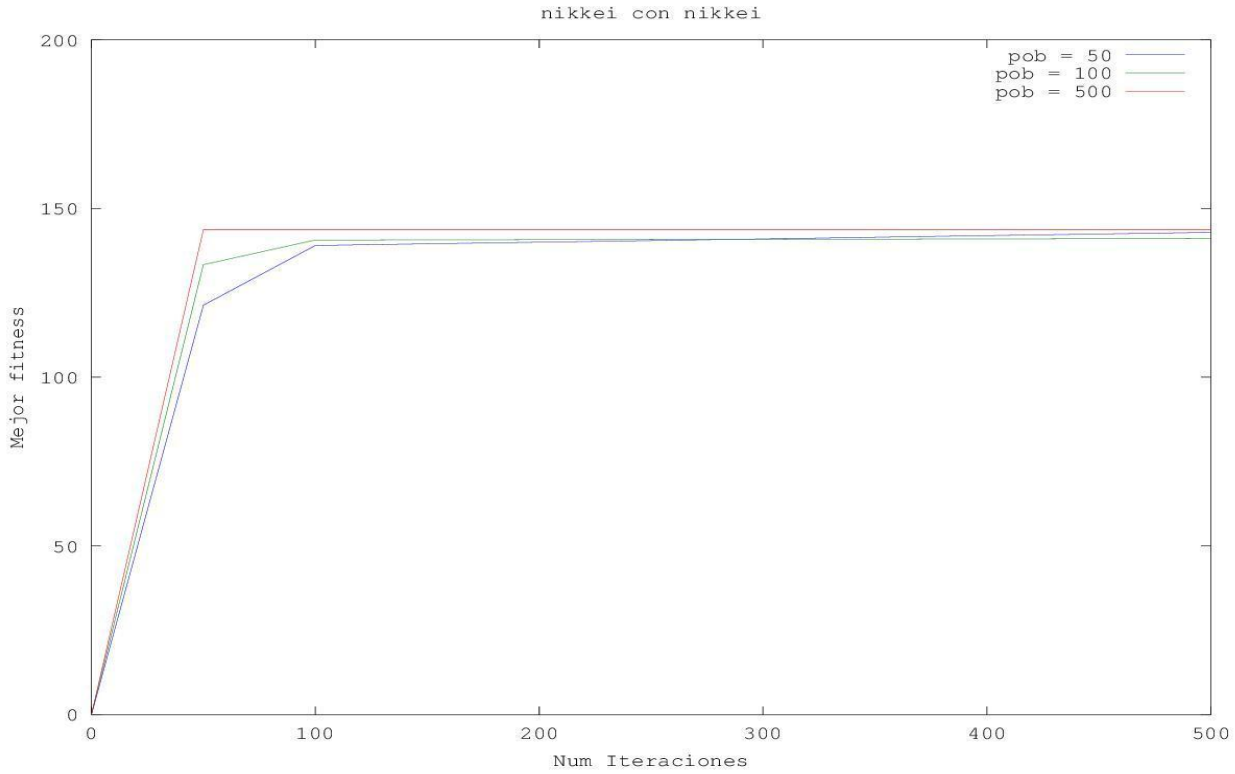
En general se puede ver que se alcanzan los máximos al utilizar entre 50 y 100 iteraciones y 50 y 100 individuos, al igual que pasaba al entrenar al algoritmo con el IBXEX-35. Sin embargo, al probar la estrategia que utiliza 50 individuos con el IBEX-35

y con el London Stock, parece que la solución mejora con 500 iteraciones y no sigue el esquema usual; posiblemente esté también relacionado con la anomalía que se ha observado antes, en la cual al utilizar 50 individuos para entrenar el algoritmo no se mejoraba la solución al aumentar el número de iteraciones. Es extraño que esto solo ocurra con la bolsa española y la londinense, pero seguramente solo pase con estos dos índices porque sus cotizaciones fueron relativamente similares durante esos años.

Las ganancias que encuentra el algoritmo son buenas, pero al aplicar estas estrategias a los otros índices se puede ver que no son muy útiles; sólo consigue alcanzar ganancias significativas con el IBEX-35.

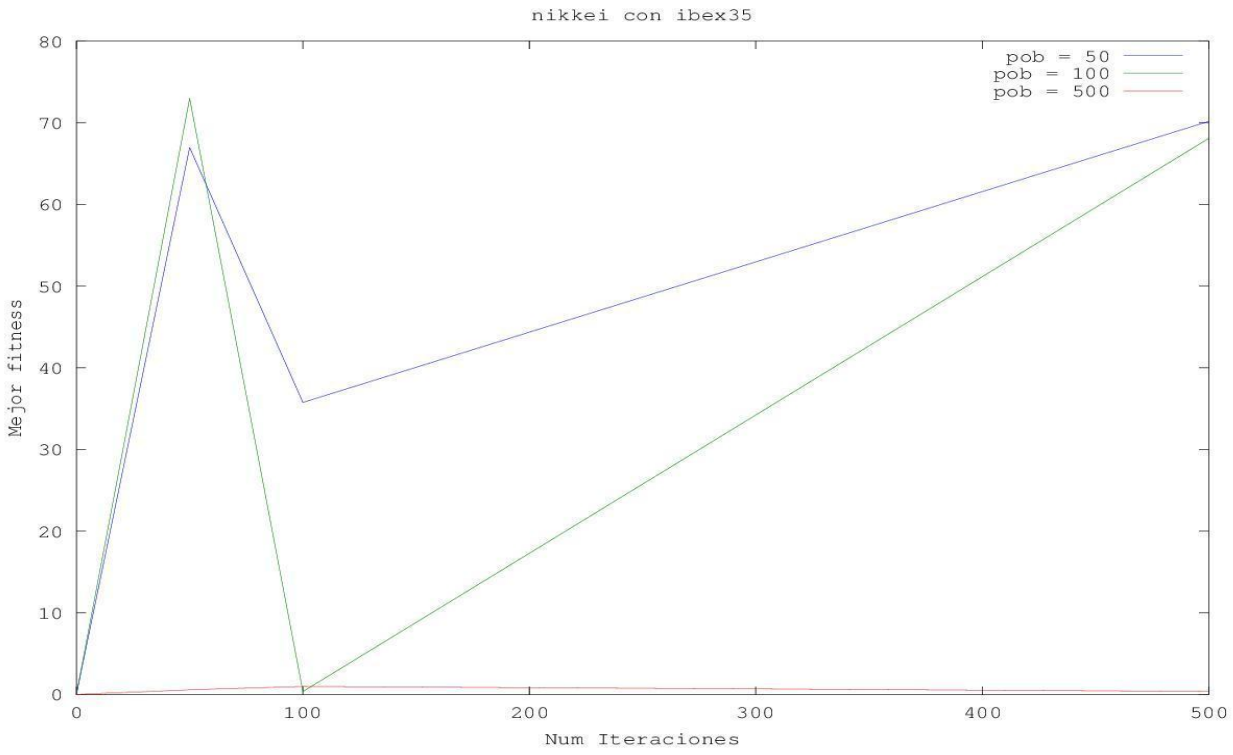
1.1.3 Nikkei

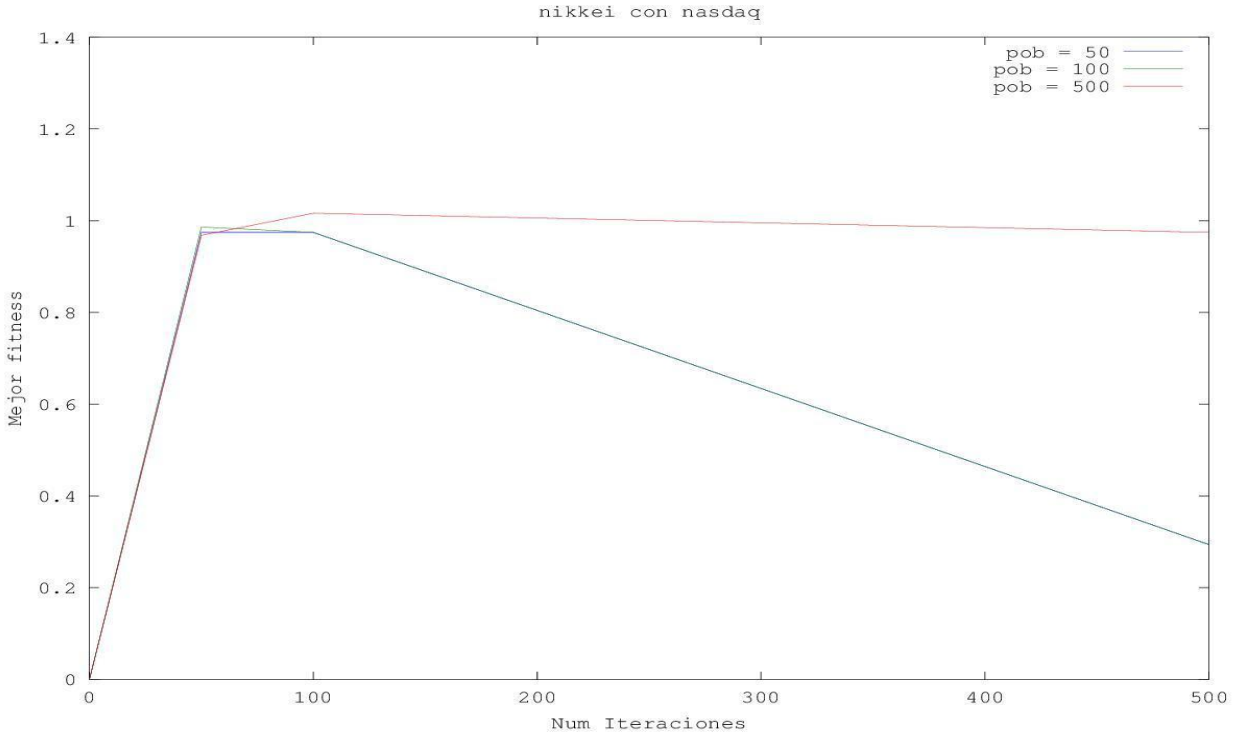
Se seguirá el mismo formato que antes[3].

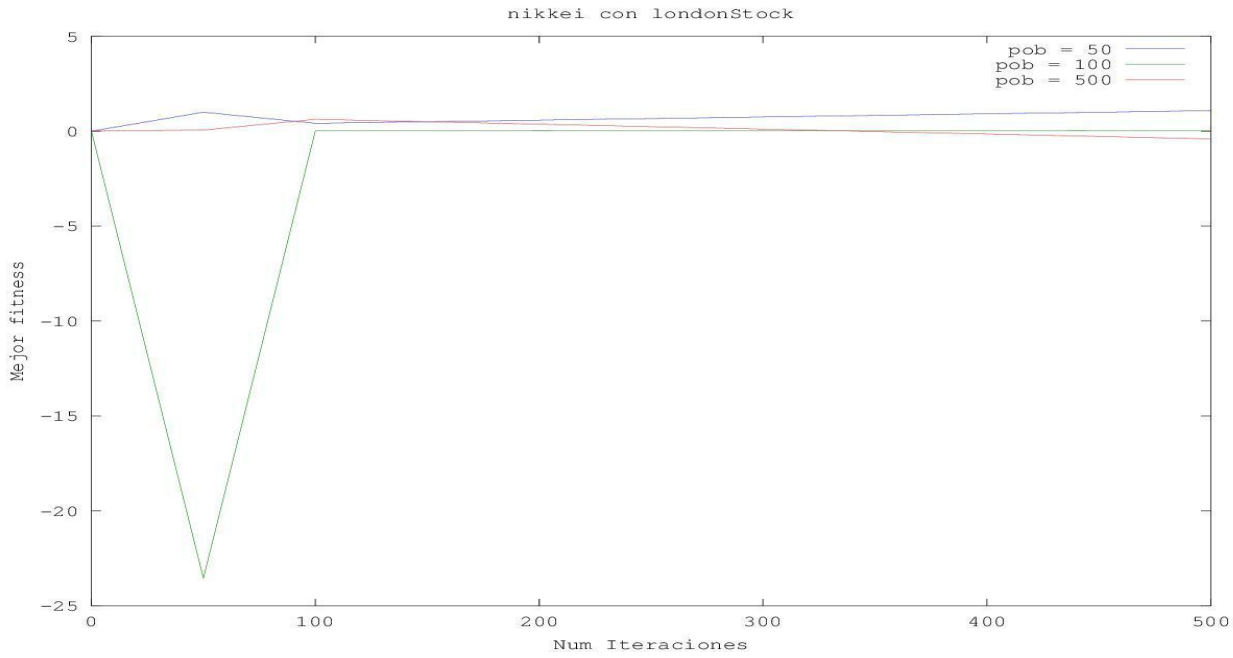


El comportamiento es similar al ya comentado anteriormente, sin embargo en este caso se consiguen ganancias altísimas, alcanzando a multiplicar por 150 la inversión.

[3] Figuras 3.1.9-3.1.12. Gráficas del Nikkei

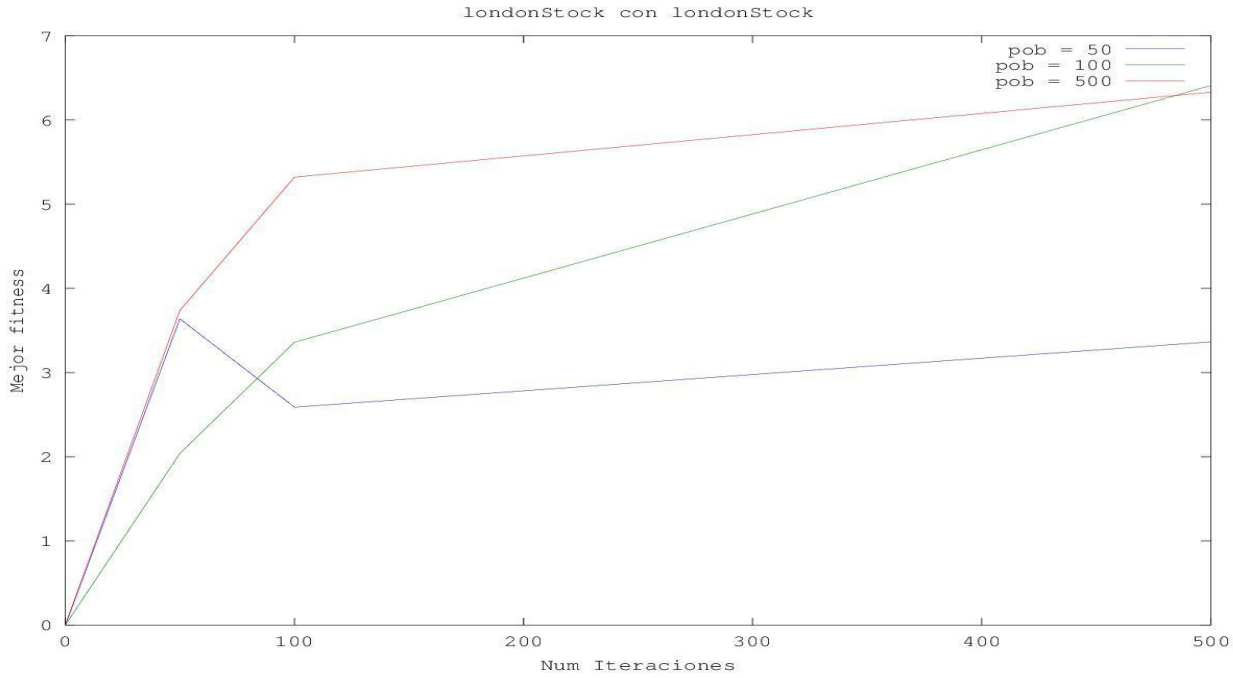

nikkei con ibex35
pob = 50
pob = 100
pob = 500
Mejor fitness
Num Iteraciones
0
10
20
30
40
50
60
70
80
0
100
200
300
400
500


nikkei con nasdaq
pob = 50
pob = 100
pob = 500
Mejor fitness
Num Iteraciones
0
0.2
0.4
0.6
0.8
1
1.2
1.4
0
100
200
300
400
500

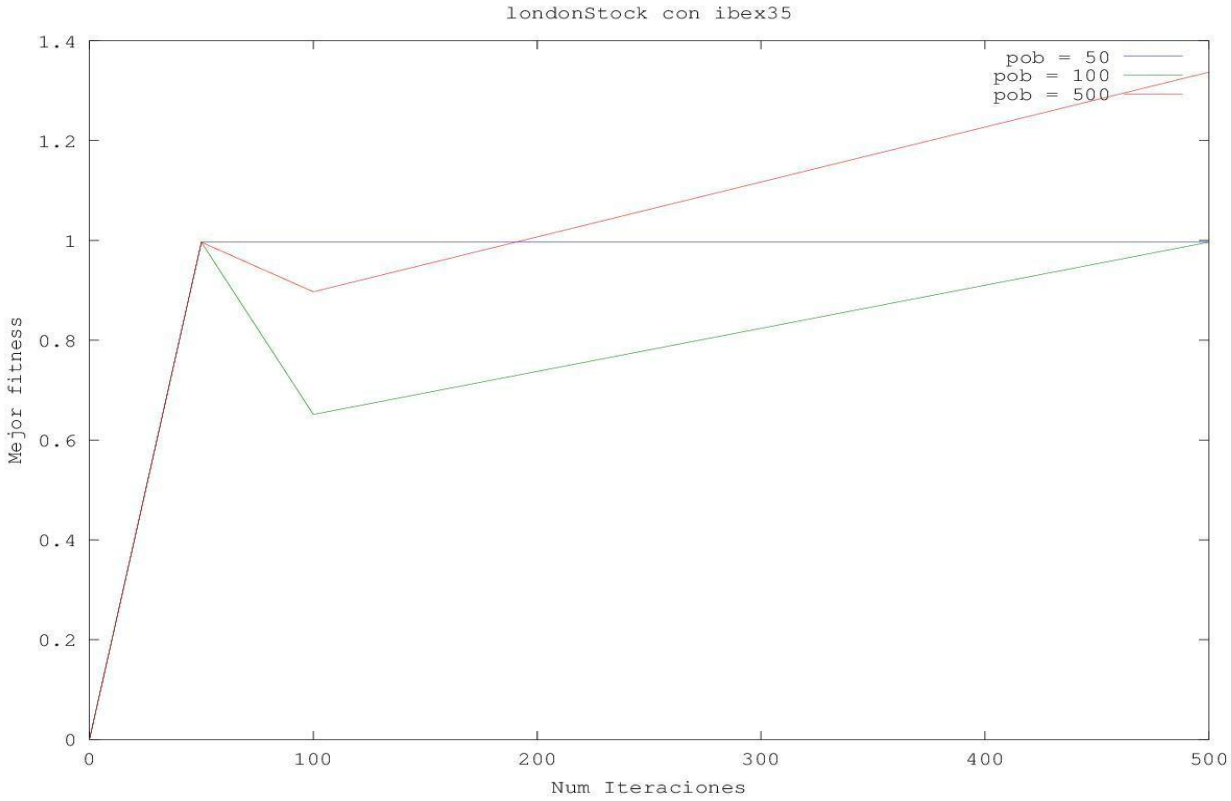


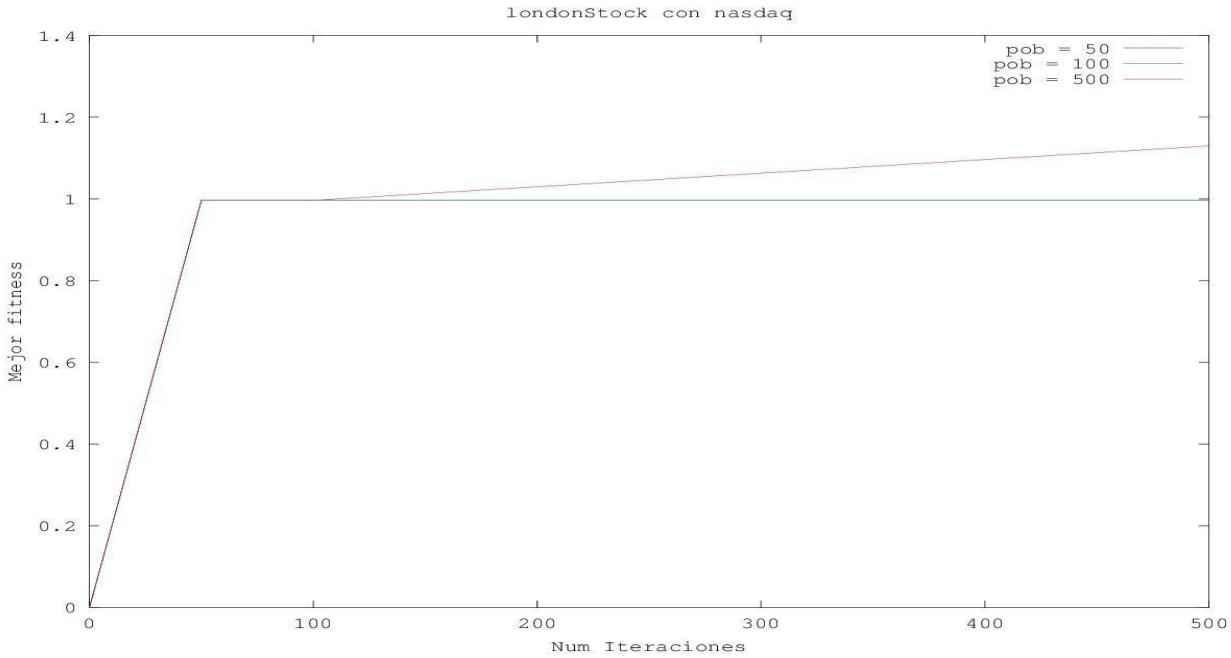
Al igual que antes, parece que los mejores resultados se alcanzan cuando el número de iteraciones y de individuos se sitúa entre 50 y 100. Obtiene ganancias muy buenas con el IBEX-35, pero con los demás índices no sólo pierde dinero, sino que con la bolsa de Londres se endeuda debido a los impuestos y las comisiones por comprar acciones y venderlas cuando están muy bajas.

1.1.4 London Stock Exchange

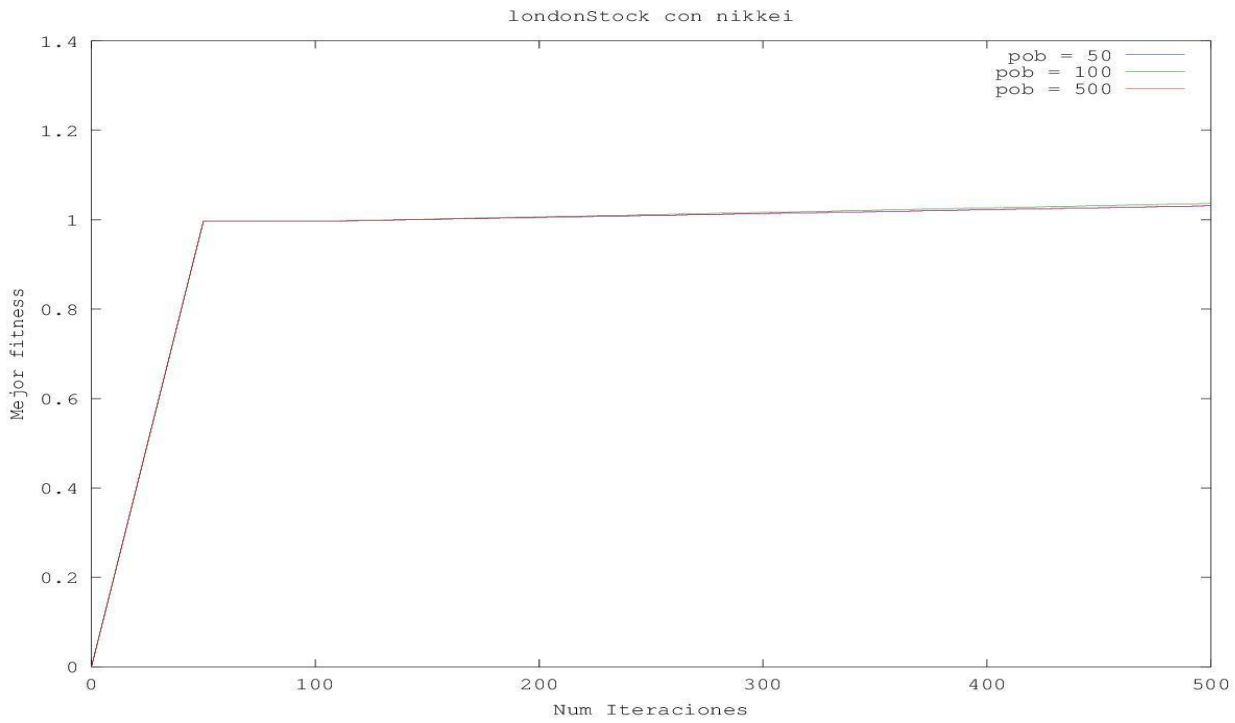


Obtiene ganancias cercanas a 7 al entrenar, por lo que los resultados son buenos[4]. También evoluciona del mismo modo que los anteriores índices.





[4] Figuras 3.1.13-3.1.16. Gráficas del London Stock Exchange



Los resultados son decepcionantes, ya que no se alcanzan ganancias muy altas; además, en general se comporta de manera extraña respecto al comportamiento usual de los algoritmos evolutivos y de aprendizaje automático, ya que a mayor cantidad de individuos e iteraciones se debería producir un sobreajuste de la solución, sin embargo con este índice mejora.

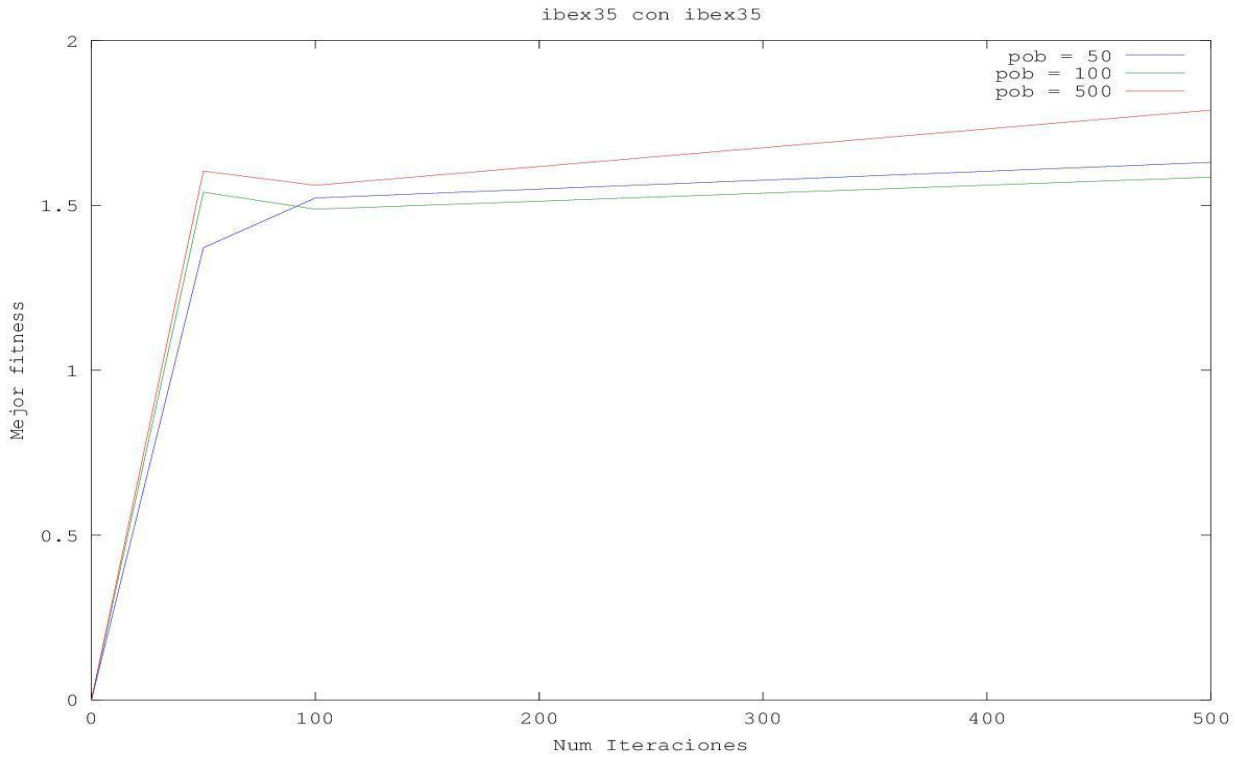
1.1.5 Conclusiones

Los resultados parecen ser un tanto aleatorios y no son buenos; hay demasiados ejemplos de comportamientos anómalos en cuanto a la evolución del fitness se refiere que deberían ser casos aislados, pero que en realidad no lo son. La principal hipótesis es que esto se debe a los impuestos, cánones y comisiones. Se tratará de comprobar esta hipótesis en las subsiguientes pruebas.

1.2 Usando seis indicadores

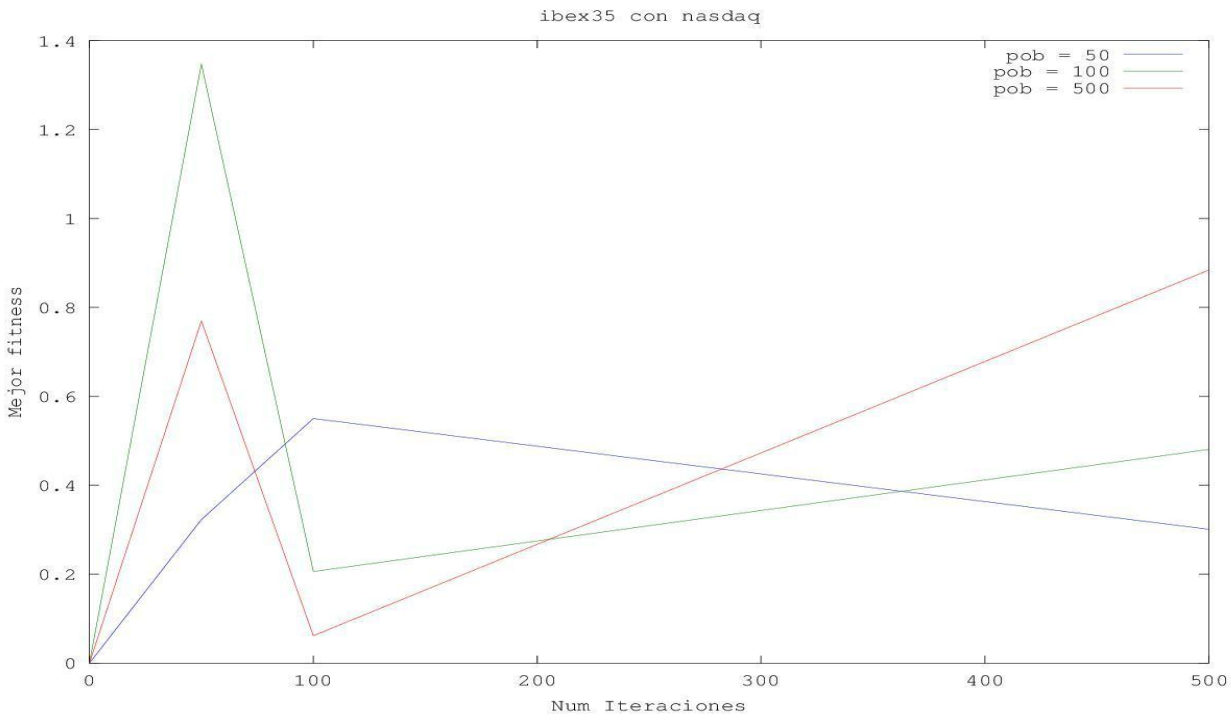
En este apartado se usarán los seis indicadores mencionados anteriormente, durante el mismo período de tiempo que la prueba anterior. Se volverán a omitir las gráficas que muestren la evolución del fitness en función del número de individuos por los mismos motivos comentados antes. El formato a seguir también será el mismo.

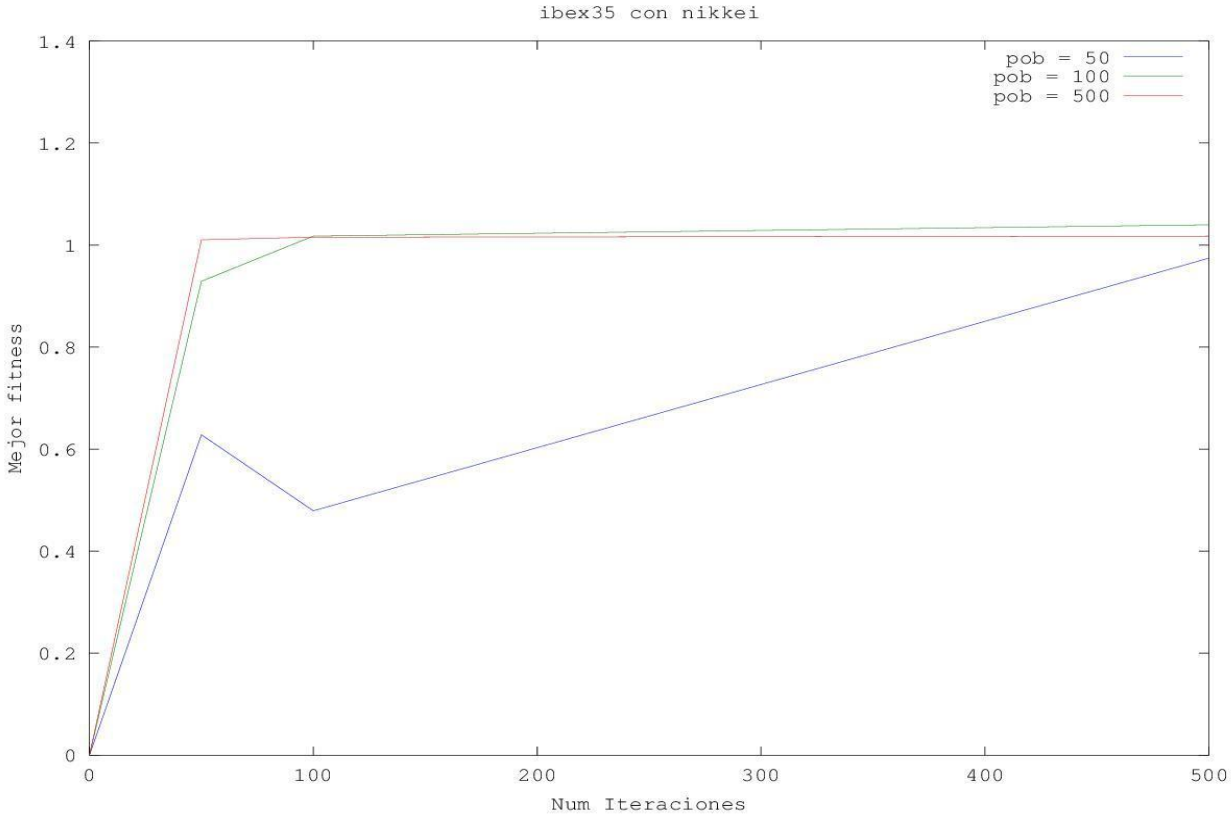
1.2.1 IBEX-35

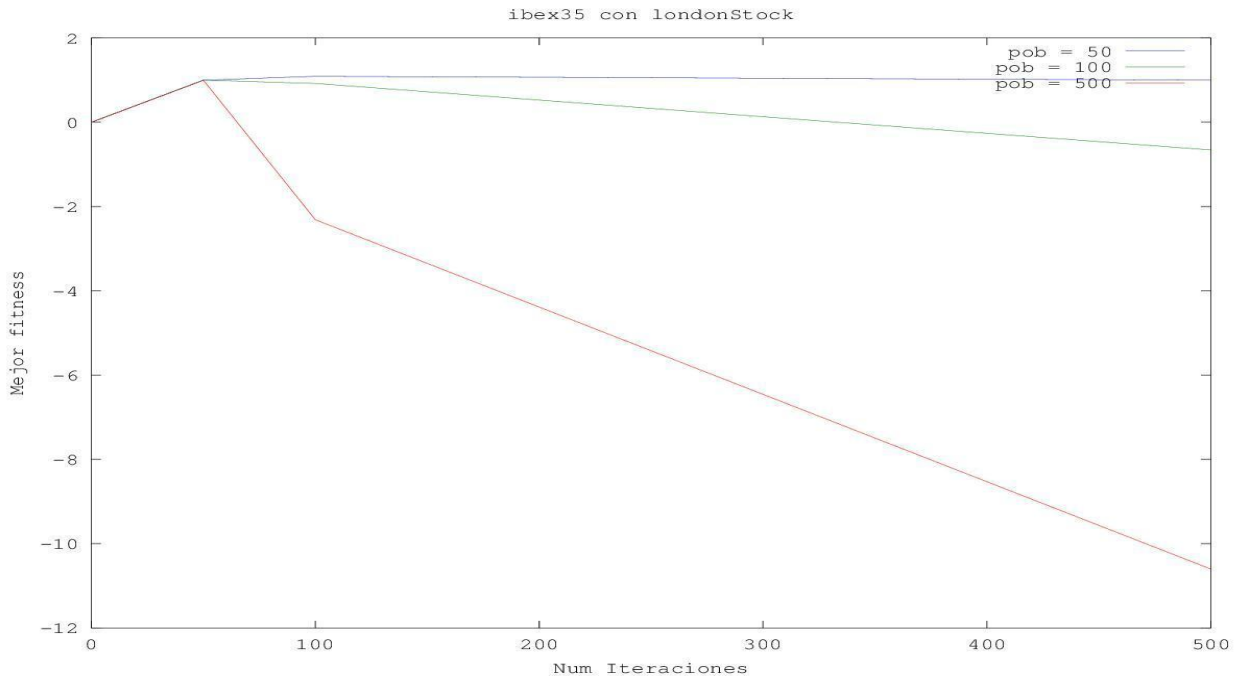


Las gráficas resultantes son similares a las obtenidas al usar solo tres indicadores, aunque al usar seis indicadores se alcanzan las ganancias del experimento anterior con un número menor de iteraciones y de individuos[5].

[5] Figuras 3.1.17-3.1.20. Gráficas del IBEX-35 usando 6 indicadores

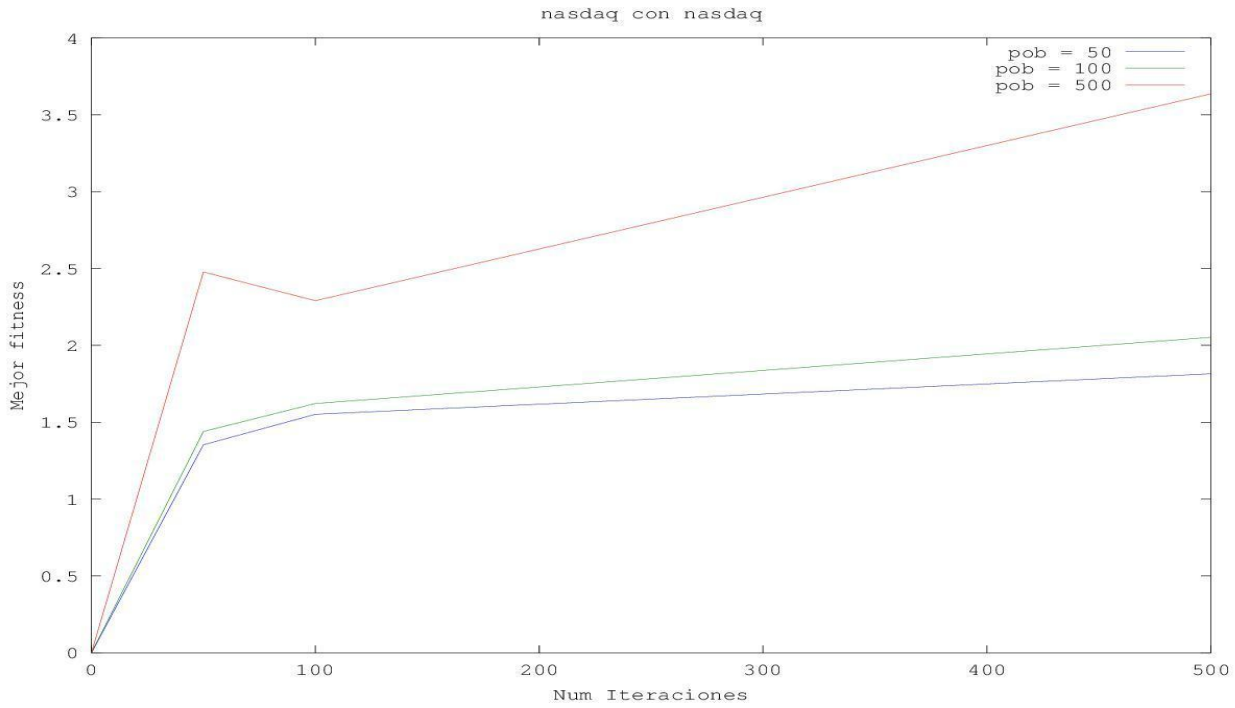

ibex35 con nasdaq
pob = 50
pob = 100
pob = 500
Mejor fitness
Num Iteraciones
0
0.2
0.4
0.6
0.8
1
1.2
1.4
0
100
200
300
400
500


ibex35 con nikkei
pob = 50
pob = 100
pob = 500
Mejor fitness
Num Iteraciones
0
0.2
0.4
0.6
0.8
1
1.2
1.4
0
100
200
300
400
500

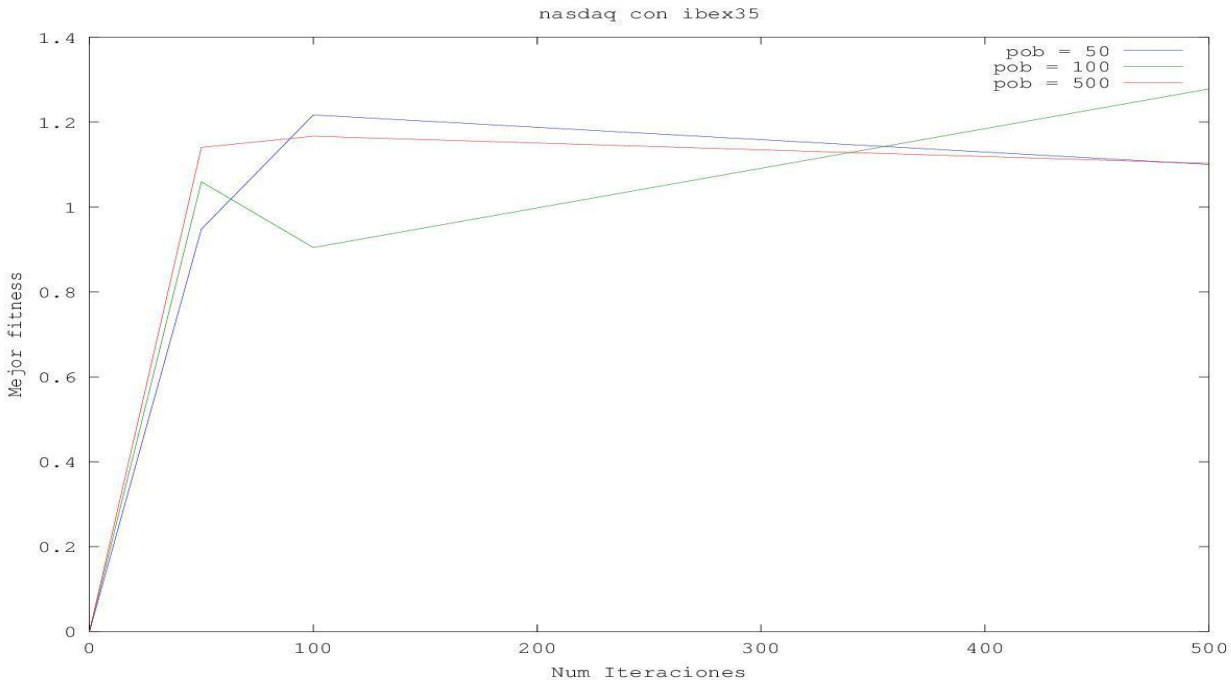


Se observa que los resultados obtenidos no son buenos, hasta el punto de endeudarse por los impuestos y los cánones; posiblemente se deba a que al añadir más indicadores se produzca un sobreajuste, esto se tratará de comprobar con los siguientes índices.

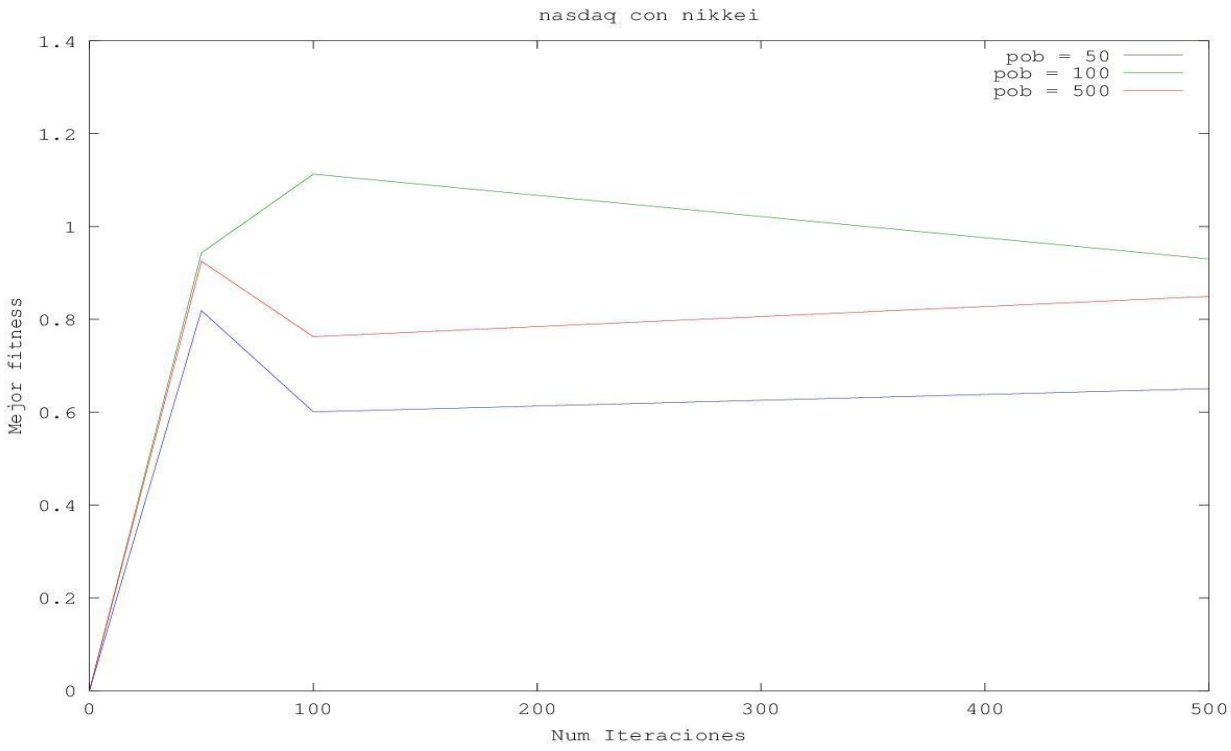
1.2.2 Nasdaq

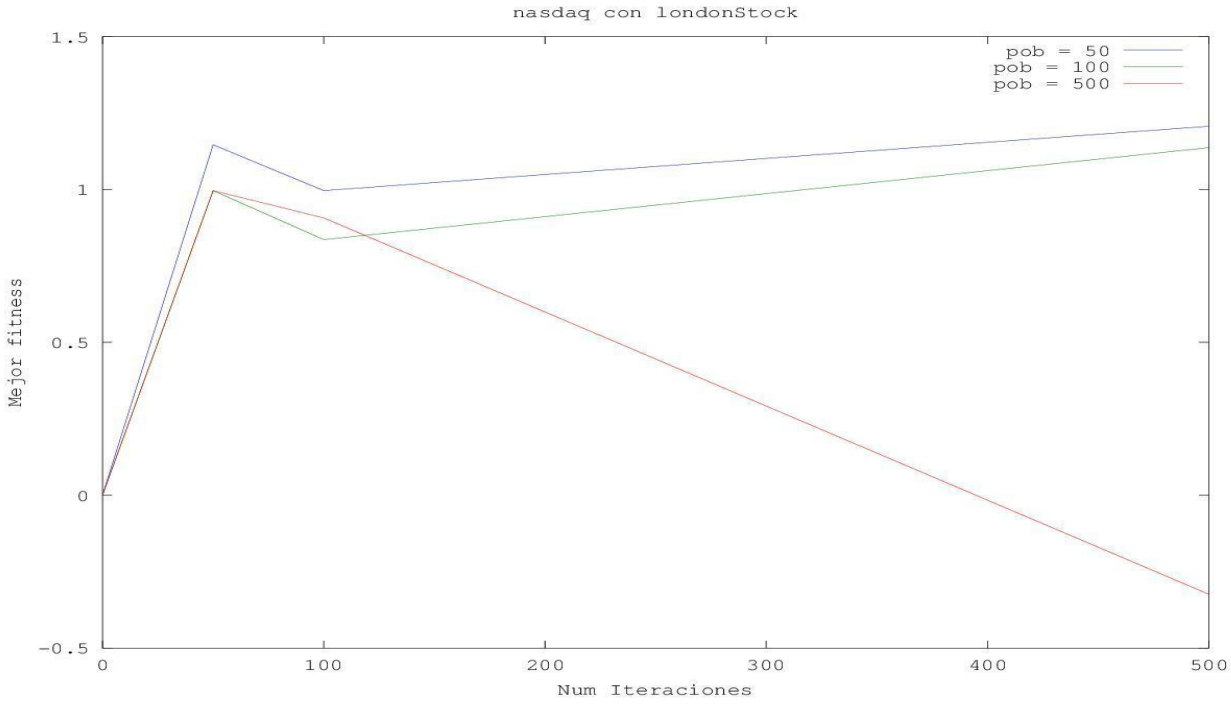


La evolución de las ganancias es similar a la resultante de usar solo tres indicadores; sin embargo, sorprendentemente las ganancias obtenidas son menores, lo que indica que en este caso los indicadores añadidos no sólo no ayudan sino que entorpecen[6].



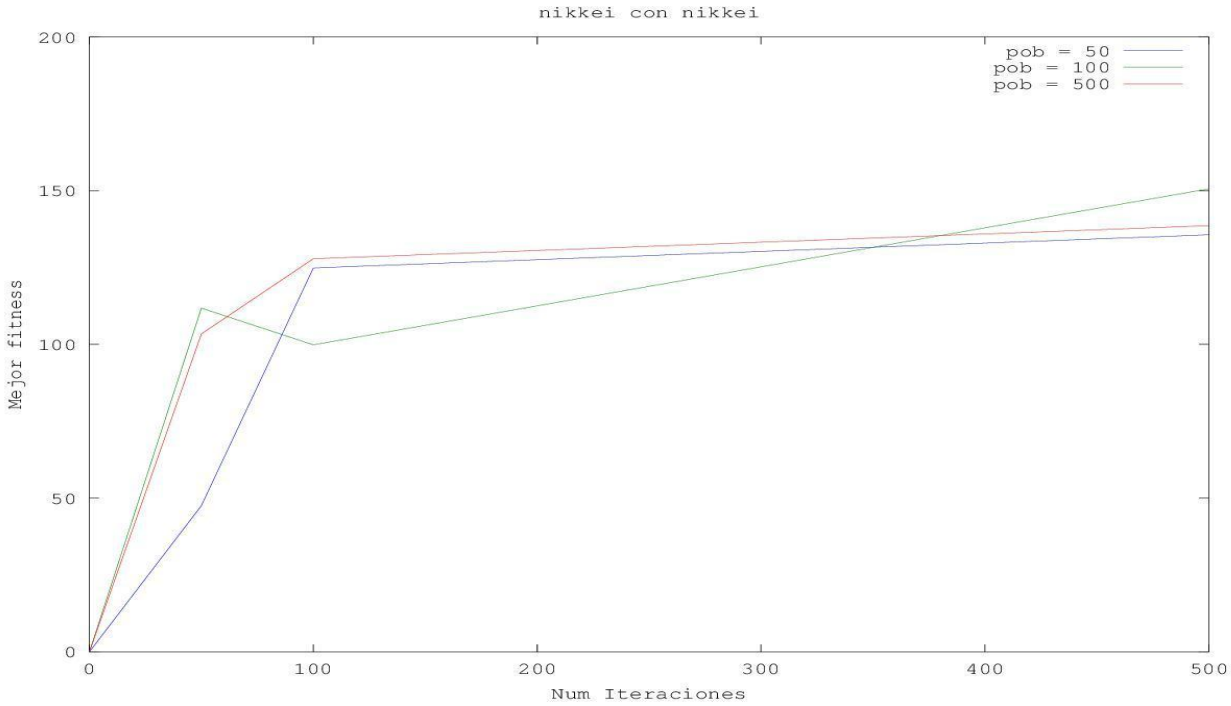
[6] Figuras 3.1.21-3.1.24. Gráficas del Nasdaq usando 6 indicadores





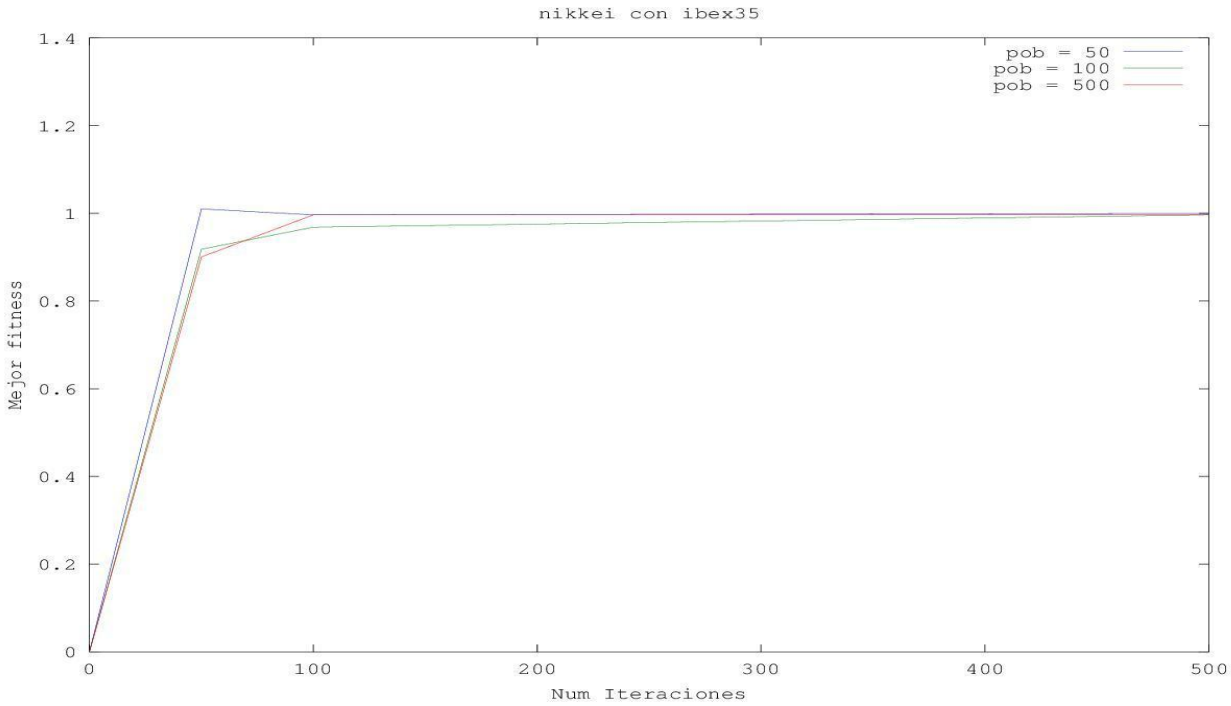
No se obtienen apenas ganancias; las mejores ganancias se suelen situar entre 50 y 100 iteraciones e individuos. Parece que, efectivamente, al añadir indicadores se produce un problema claro de sobreajuste.

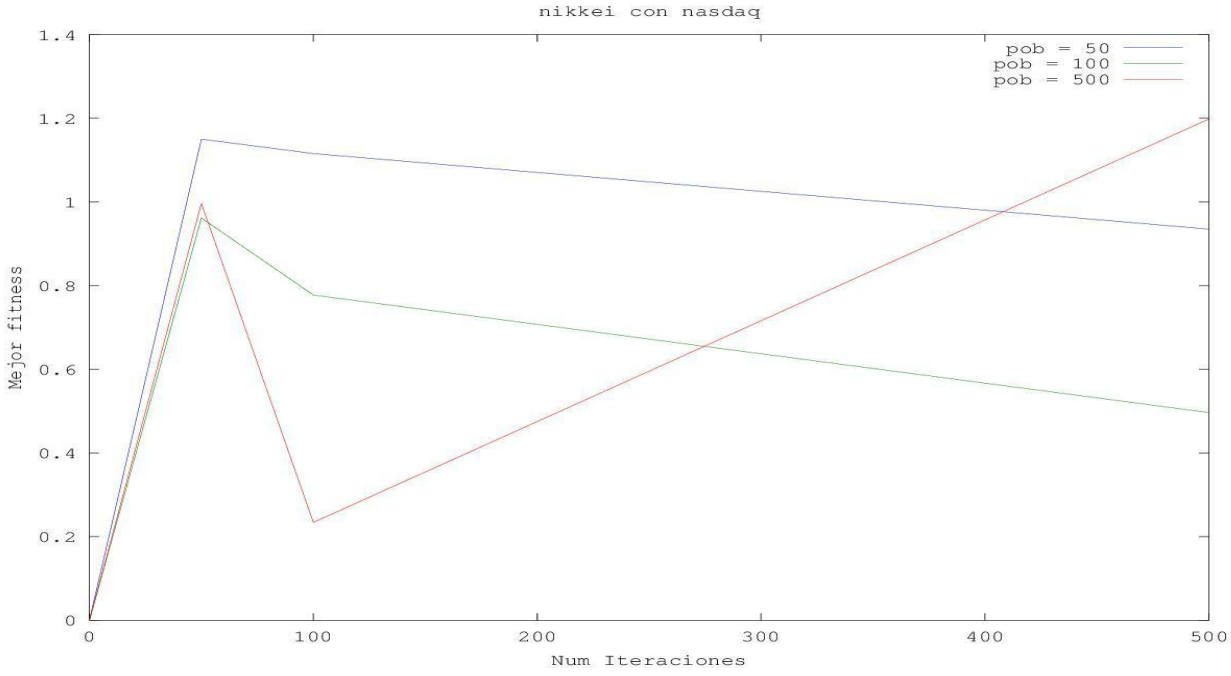
1.2.3 Nikkei

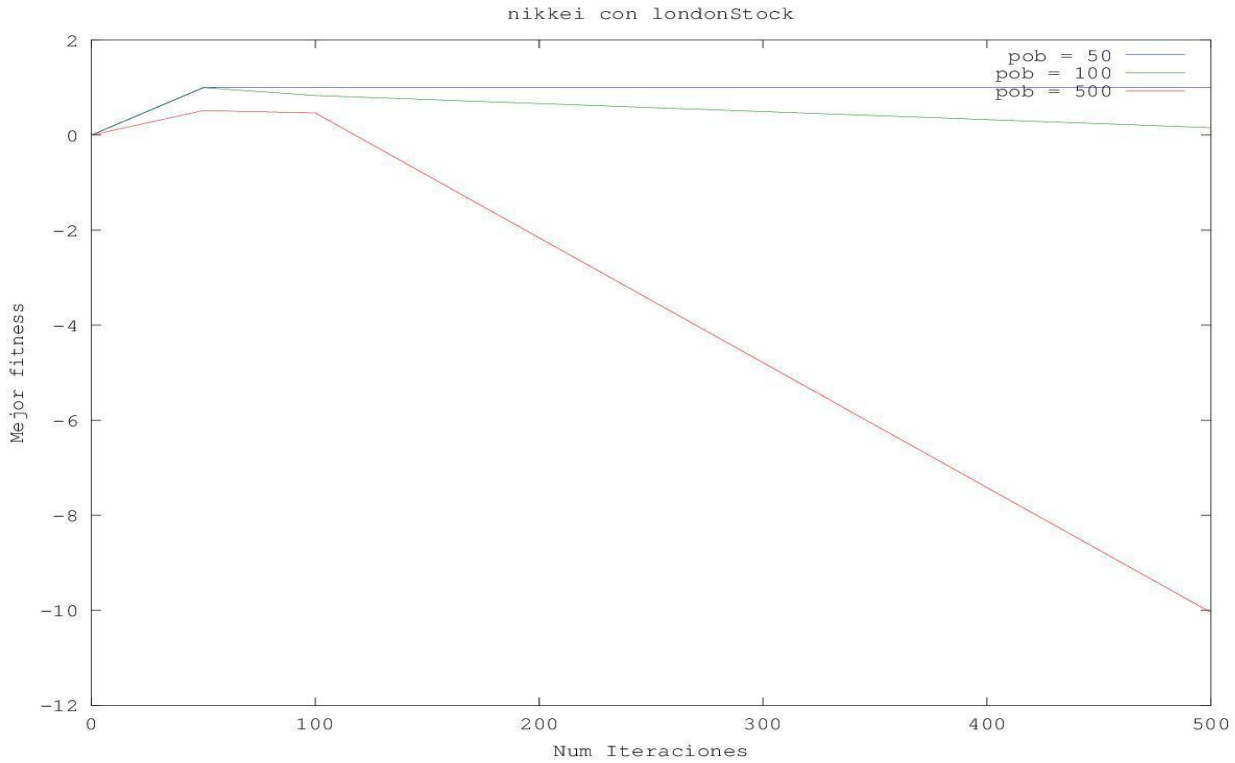


Los resultados son parecidos a los obtenidos utilizando sólo tres indicadores, aunque parece que el crecimiento de las ganancias respecto al número de iteraciones no se frena de manera tan rápida como hacía antes; sin embargo no parece que el añadir indicadores mejore significativamente las ganancias obtenidas antes[7].

[7] Figuras 3.1.25-3.1.28. Gráficas del Nikkei usando 6 indicadores

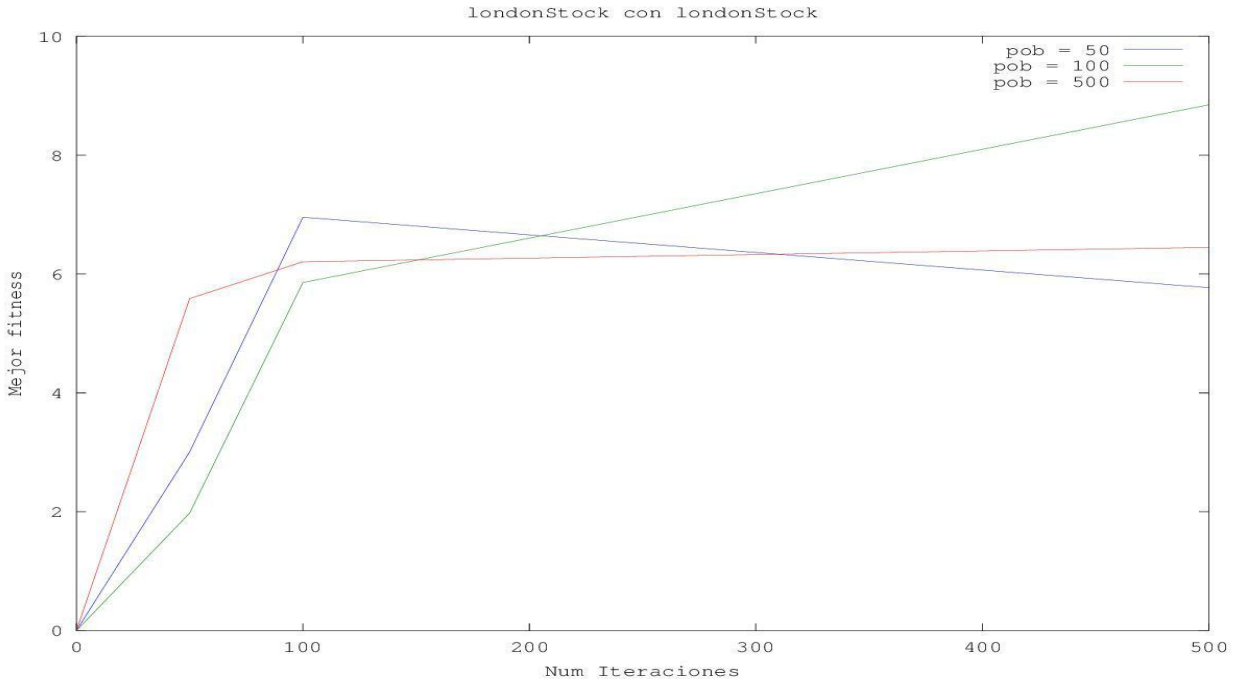

nikkei con ibex35
pob = 50
pob = 100
pob = 500
Mejor fitness
Num Iteraciones
0
0.2
0.4
0.6
0.8
1
1.2
1.4
0
100
200
300
400
500


nikkei con nasdaq
pob = 50
pob = 100
pob = 500
Mejor fitness
Num Iteraciones
0
0.2
0.4
0.6
0.8
1
1.2
1.4
0
100
200
300
400
500

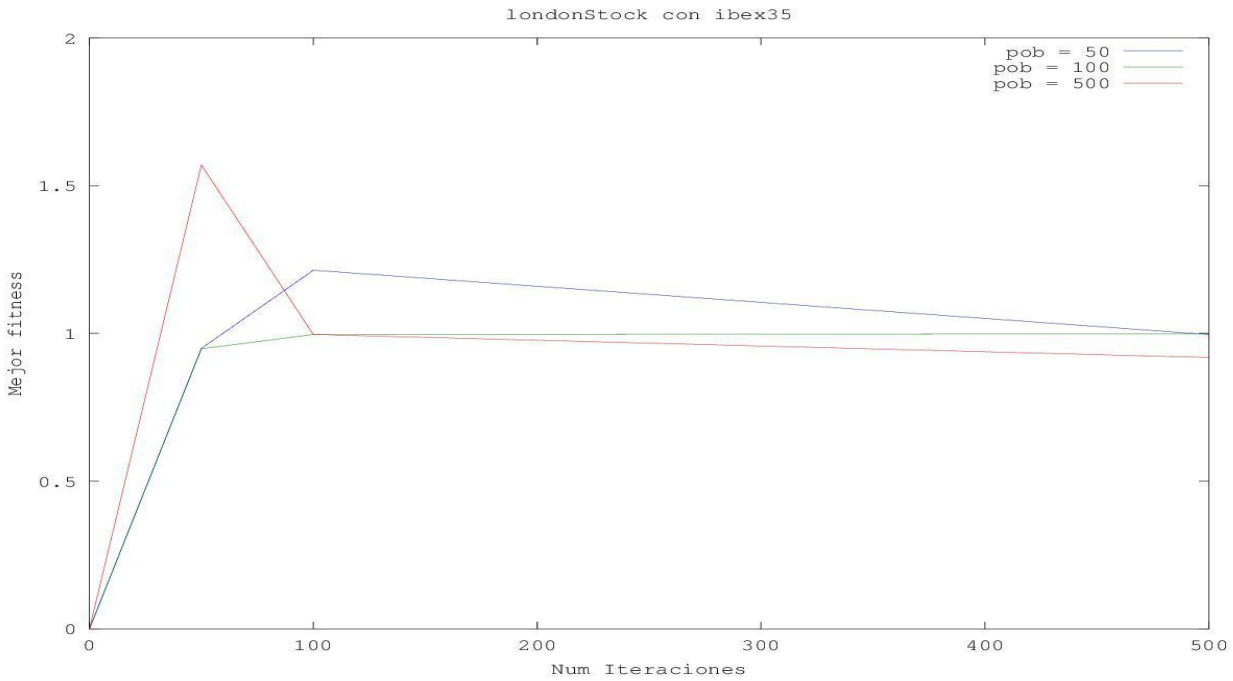


Estos resultados vuelven a ser malos, de hecho en comparación con utilizar solo tres indicadores son mucho peores. En general tampoco se conseguían muchas ganancias al usar tres indicadores, pero con el IBEX-35 conseguía multiplicar la inversión por setenta, y en este caso no. Seguramente se deba a un sobreajuste de la solución.

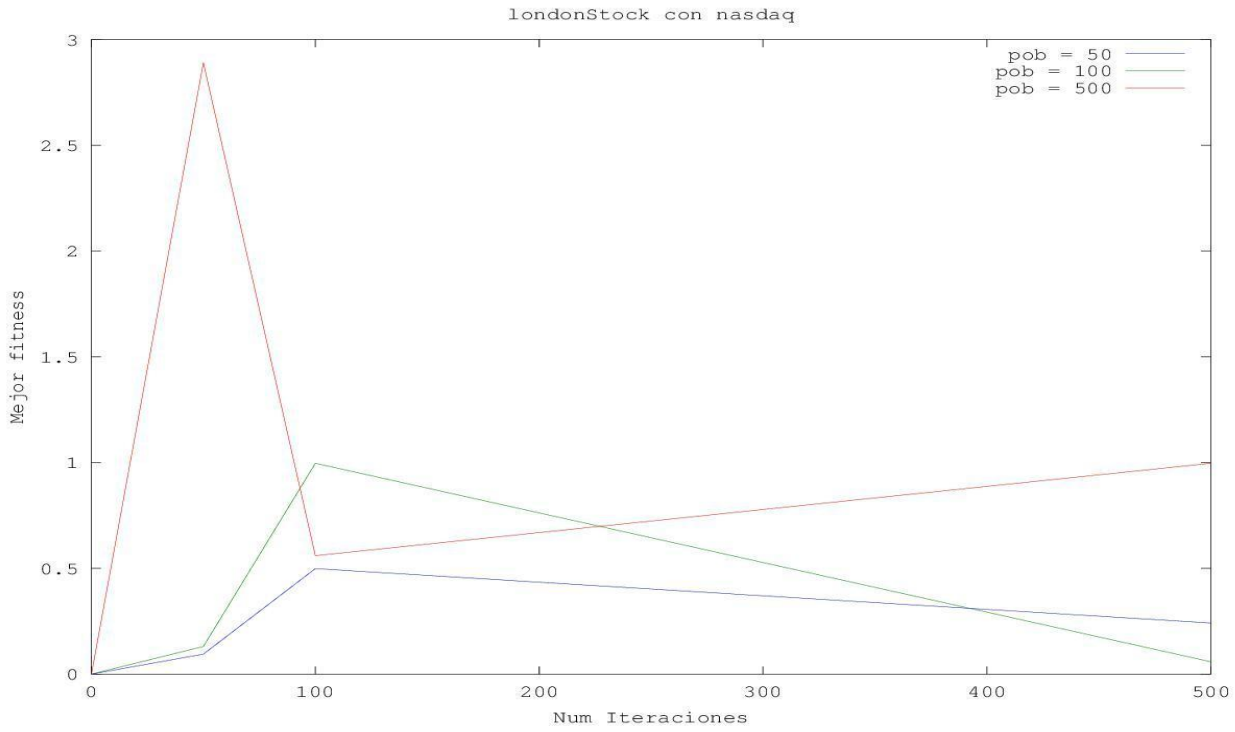
1.2.4 London Stock Exchange

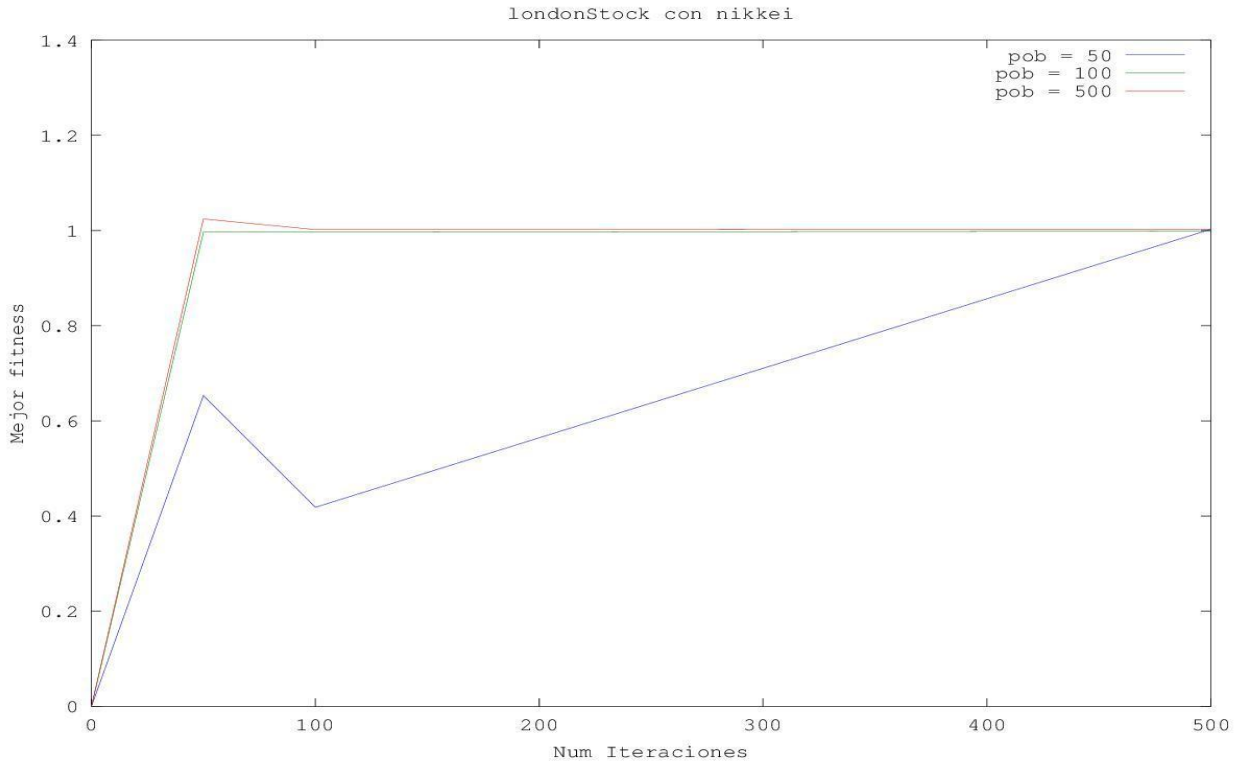


En este caso sí parece que se sigue el comportamiento esperado y las ganancias aumentan al haber más indicadores; por lo demás el comportamiento es similar al de utilizar tres indicadores[8].



[8] Figuras 3.1.29-3.1.32. Gráficas del London Stock Exchange usando 6 indicadores





Al entrenar con la bolsa de Londres obtenemos resultados menos anómalos que los que se han ido obteniendo antes; las mejores ganancias se suelen encontrar al utilizar entre 50 y 100 iteraciones e individuos, y se obtienen ganancias con todos los índices, que aunque bajas, resultan más altas que al utilizar sólo 3 índices.

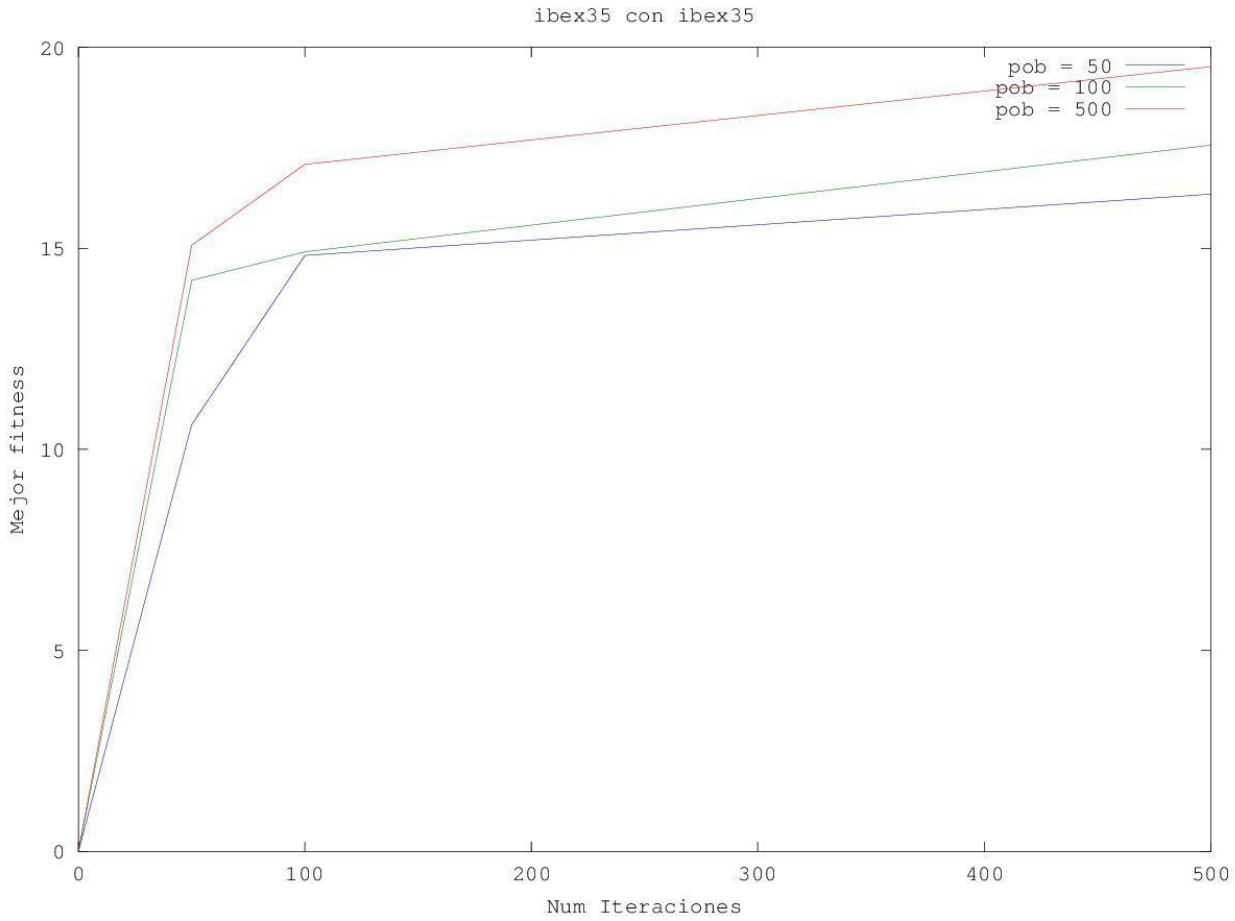
1.2.5 Conclusiones

Parece que, al aumentar el número de indicadores a usar, se alcanzan soluciones sobreajustadas al índice que se ha utilizado para entrenar al algoritmo. No merece la pena el tiempo extra de computación que se requiere al añadir indicadores con las mejoras obtenidas. También pasa como al utilizar sólo tres índices, y hay varios casos anómalos que deberían ser pocos en cantidad y sin embargo no lo son. Quizás los impuestos y los cánones tengan demasiado peso como para que se pueda encontrar una estrategia general que funcione medianamente bien en todos los índices. En los siguientes apartados se desarrolla mejor esta idea. Tampoco parece que exista una relación entre el número de días utilizado a entrenar o el período de tiempo concreto, ya que el Nikkei y el Nasdaq comparten número de días y período de tiempo, y uno da resultados bastante mejores que el otro.

1.3 Sin impuestos ni comisiones con indicadores básicos

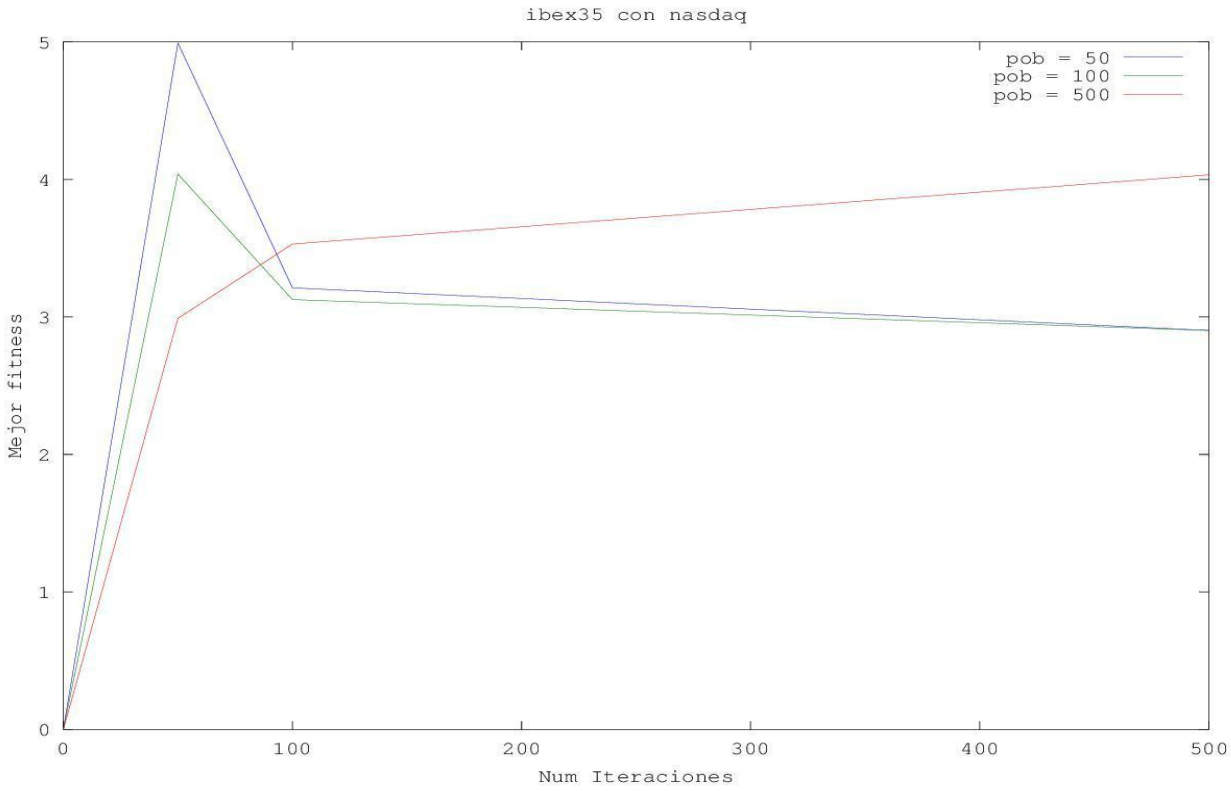
Los siguientes resultados se obtuvieron al quitar todos los costes causados por impuestos, cánones y comisiones, tomando los datos de la bolsa entre los mismos años que los anteriores experimentos.

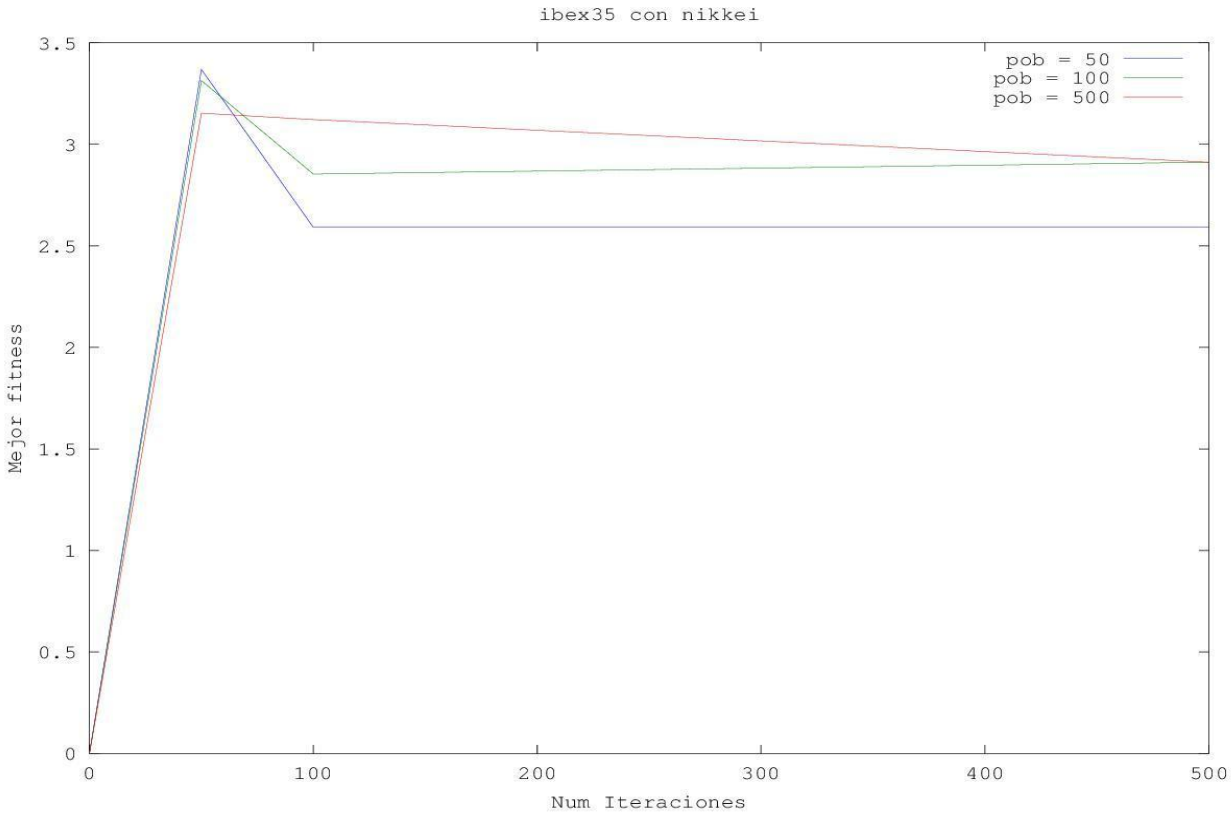
1.3.1 IBEX-35

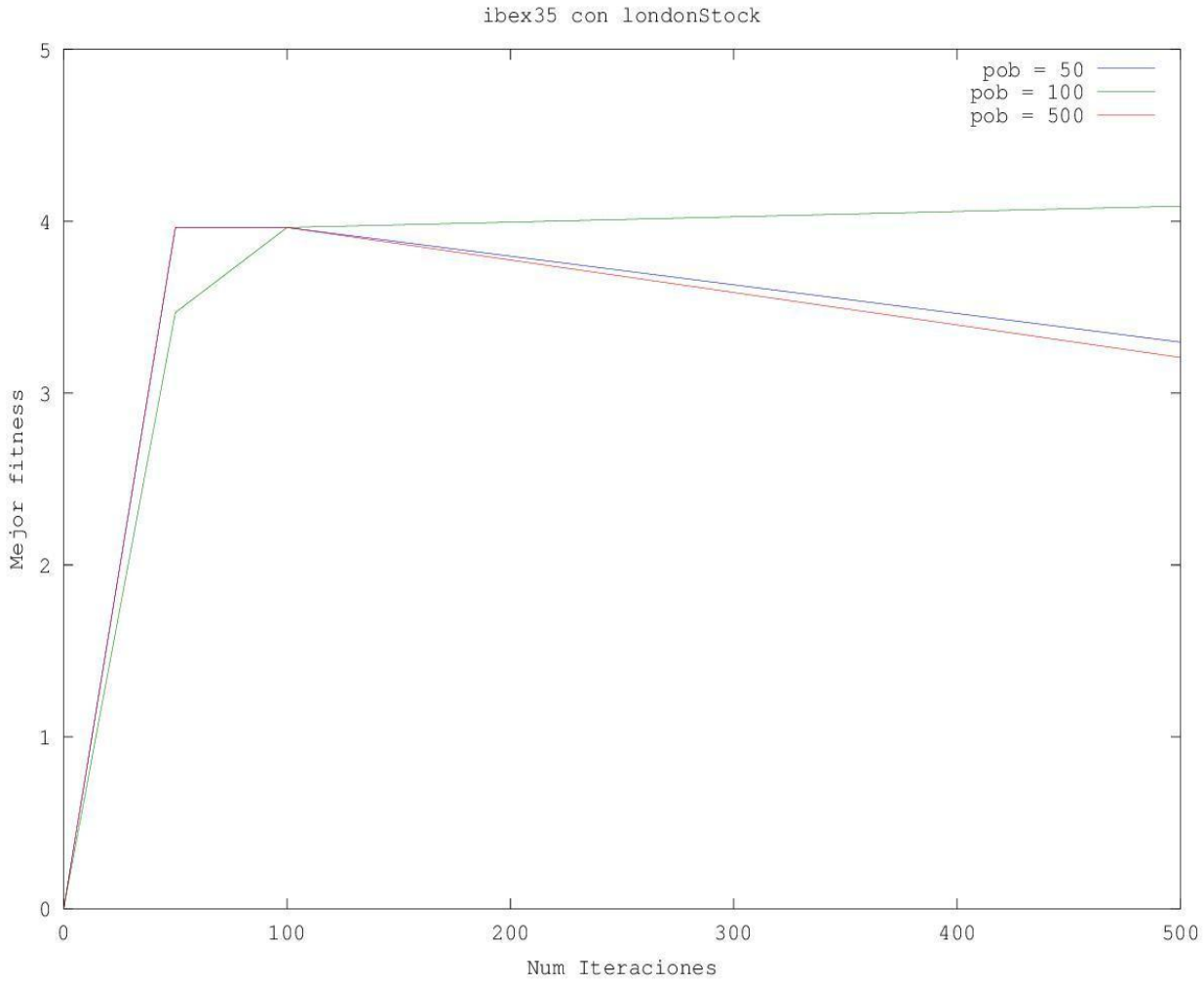


Estos resultados muestran claramente que, cuando no se tienen en cuenta los impuestos, el algoritmo genético adquiere lo que debería ser su comportamiento usual: al aumentar el número de individuos y de iteraciones se mejora bastante el fitness encontrado, aunque a partir de 100 iteraciones la función de fitness crece más lentamente. Se obtienen ganancias mejores que las que da la estrategia de comprar el primer día y vender el último[9].

[9] Figuras 3.1.33-3.1.36. Gráficas del IBEX-35 sin impuestos

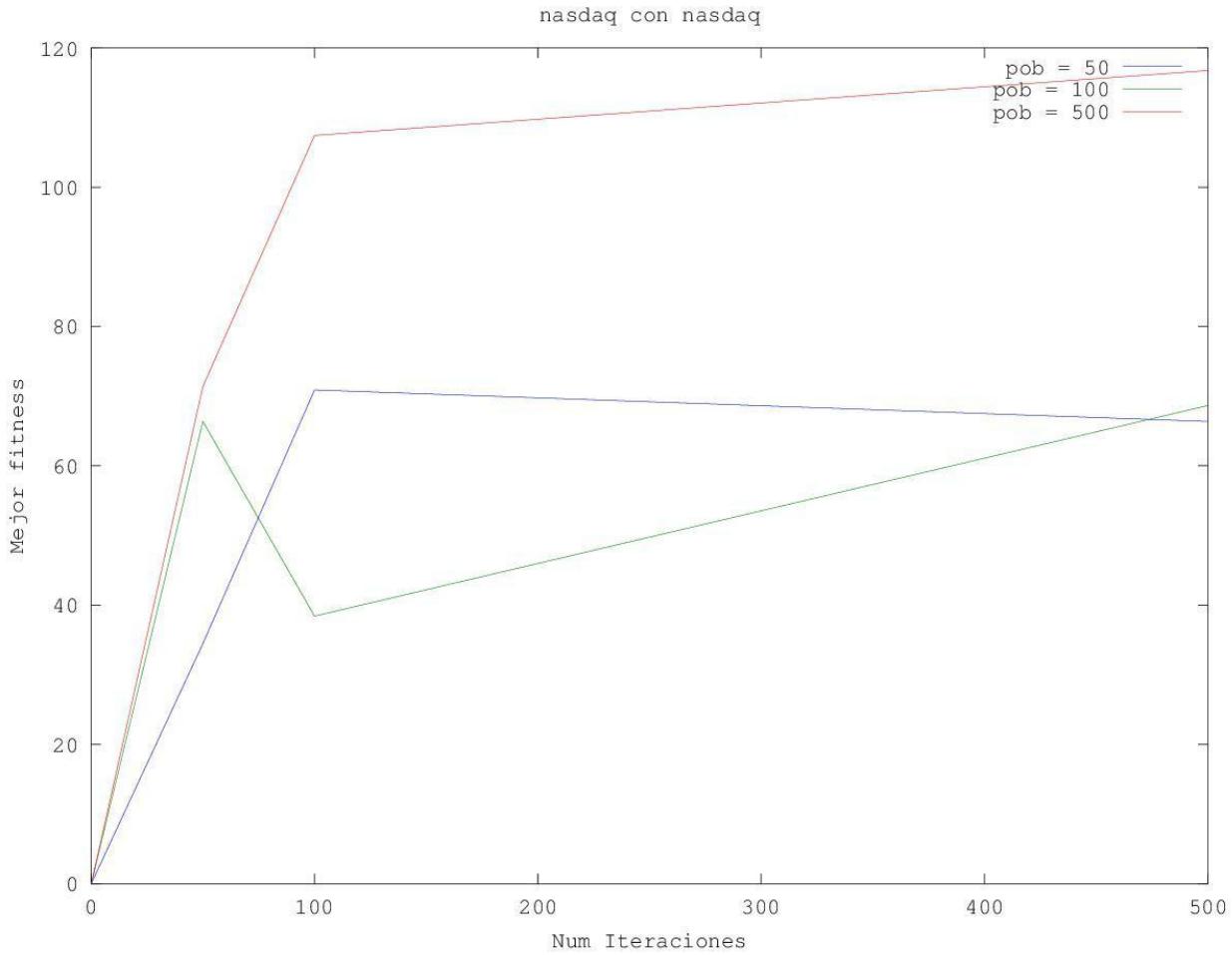

ibex35 con nasdaq
pob = 50
pob = 100
pob = 500
Mejor fitness
Num Iteraciones
0
1
2
3
4
5
0
100
200
300
400
500


ibex35 con nikkei
pob = 50
pob = 100
pob = 500
Mejor fitness
Num Iteraciones
0
0.5
1
1.5
2
2.5
3
3.5
0
100
200
300
400
500



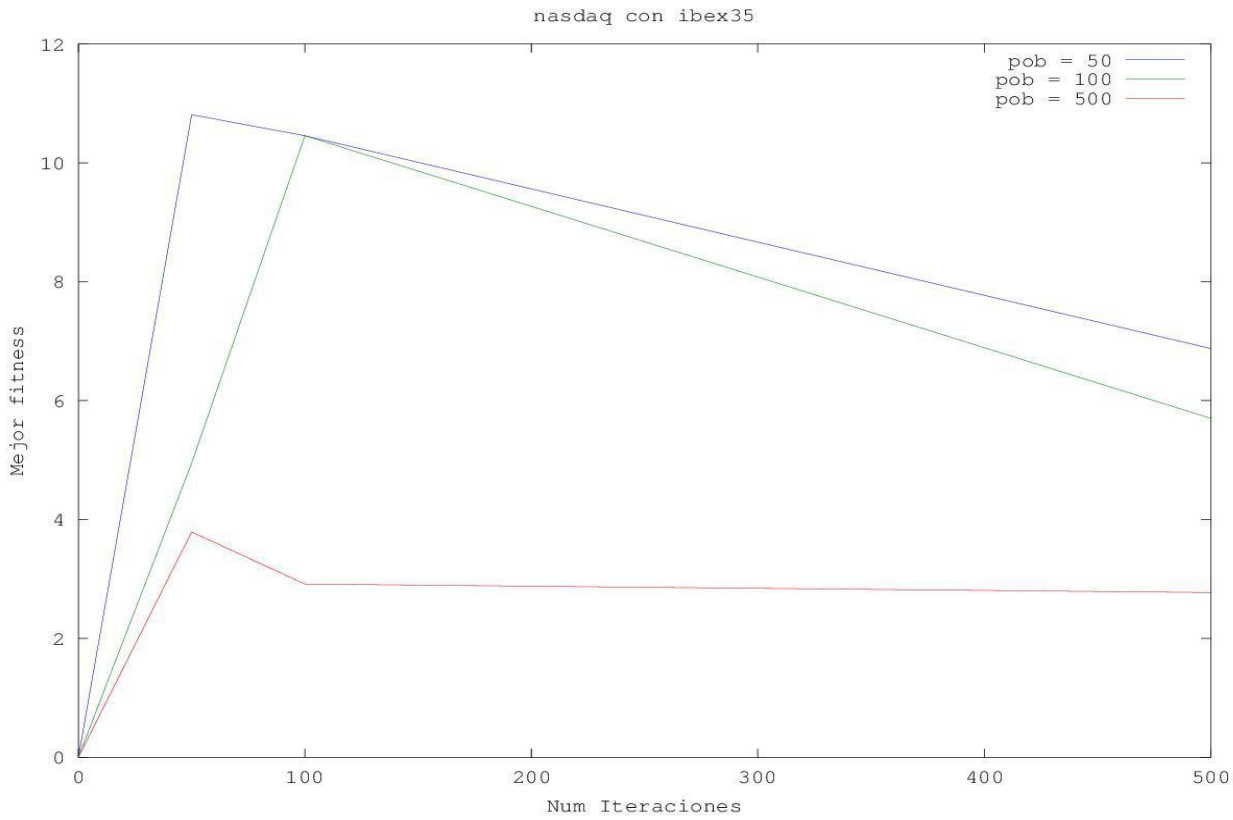
Al probar la estrategia con otros índices se obtienen mejores resultados que cuando hay impuestos, aunque no son resultados excesivamente buenos. Se puede observar que se alcanza el máximo local en torno a 50 y 100 iteraciones e individuos, aunque con el London Stock Exchange hay una excepción. Merece la pena comentar que la estrategia encontrada gana más de dinero invirtiendo en el London Stock Exchange que en el Nikkei y el Nasdaq, a pesar de que en estos últimos hay casi 20 años más de datos y por lo tanto bastantes más posibilidades de ganar dinero, por lo que se infiere que la estrategia encontrada no es muy general y parece sobreajustada al índice sobre el que se ha entrenado el algoritmo.

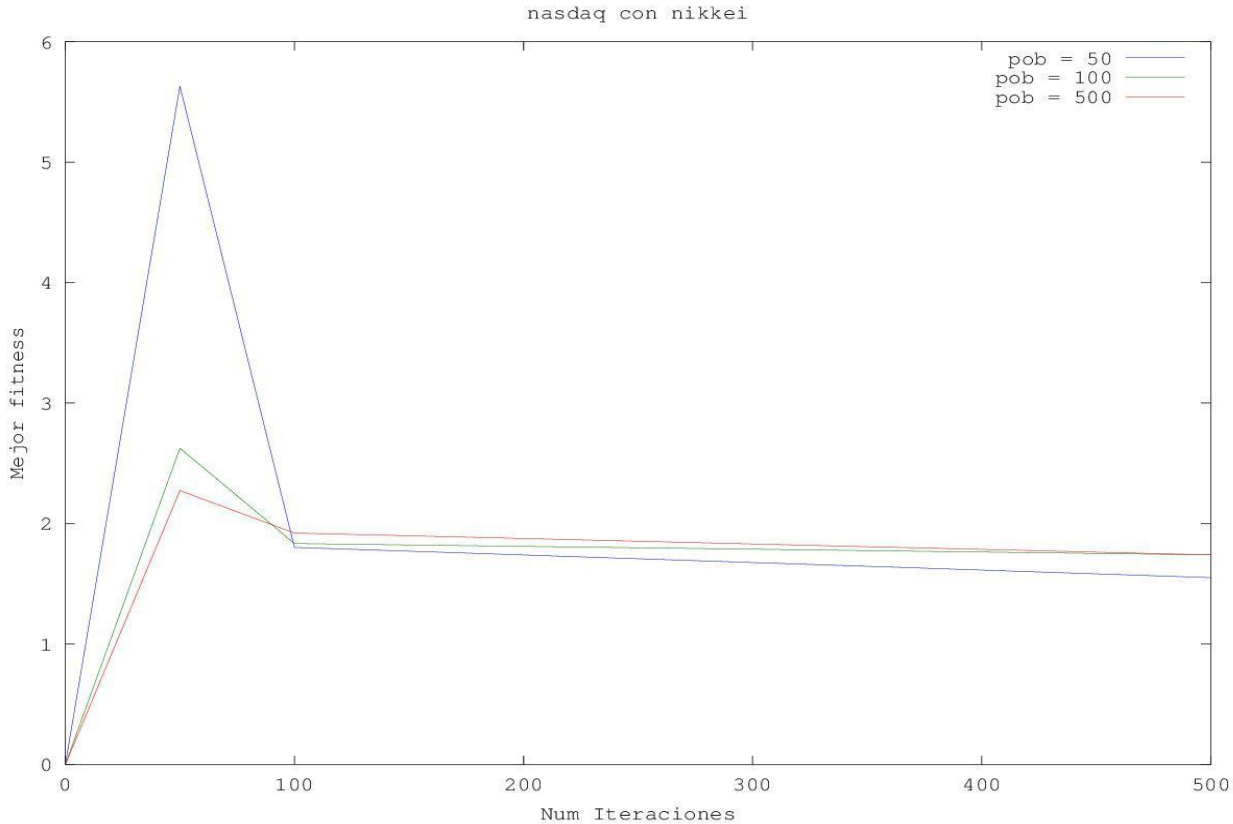
1.3.2 Nasdaq

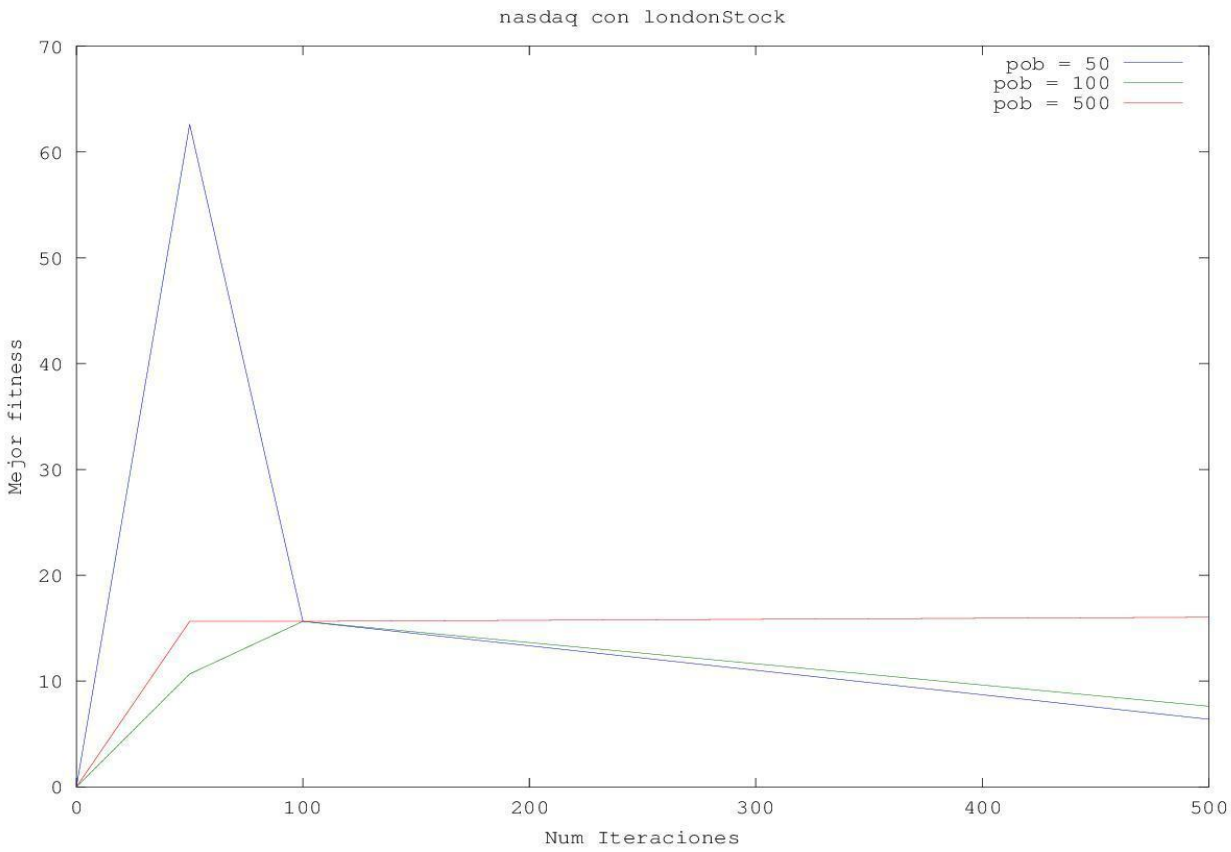


Igual que en el caso del IBEX se puede observar cómo mejoran los resultados al quitar los impuestos y cómo ahora sí sigue un esquema normal, parece que cuando hay impuestos y cánones de por medio el algoritmo genético se comporta de manera extraña[10].

[10] Figuras 3.1.37-3.1.40. Gráficas del Nasdaq sin impuestos

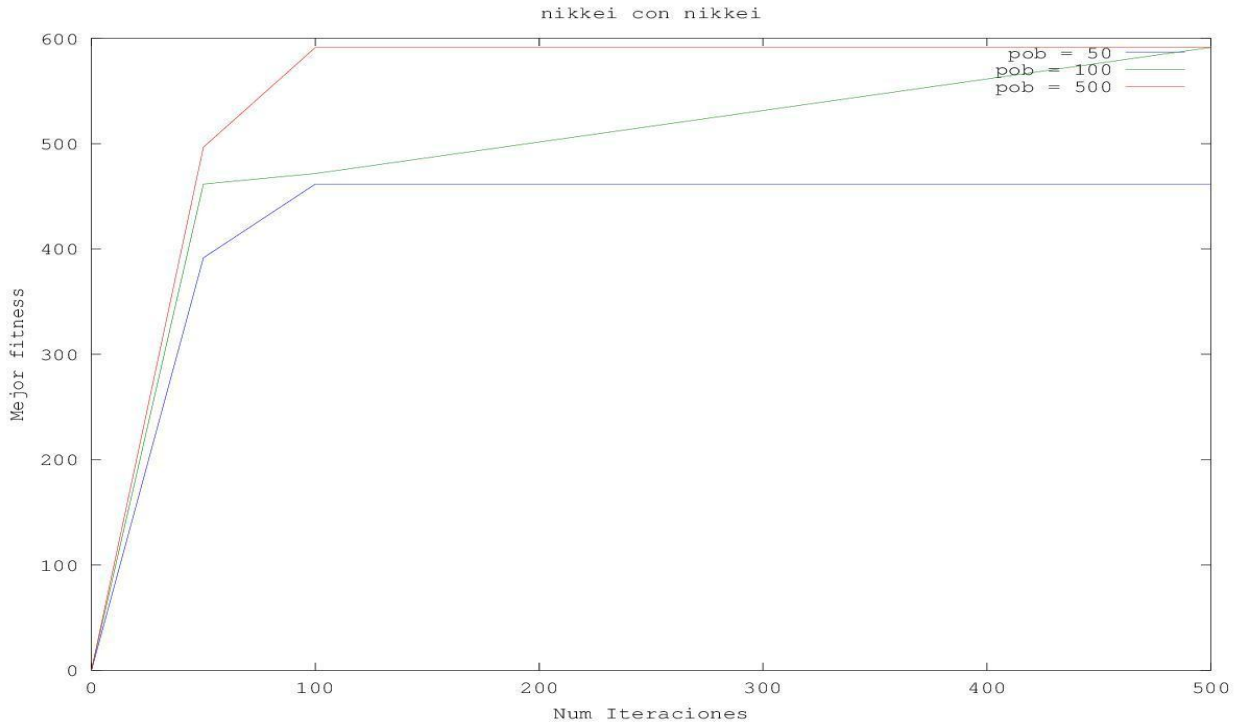

nasdaq con ibex35
pob = 50
pob = 100
pob = 500
Mejor fitness
Num Iteraciones
0
2
4
6
8
10
12
0
100
200
300
400
500


nasdaq con nikkei
pob = 50
pob = 100
pob = 500
Mejor fitness
Num Iteraciones
0
1
2
3
4
5
6
0
100
200
300
400
500

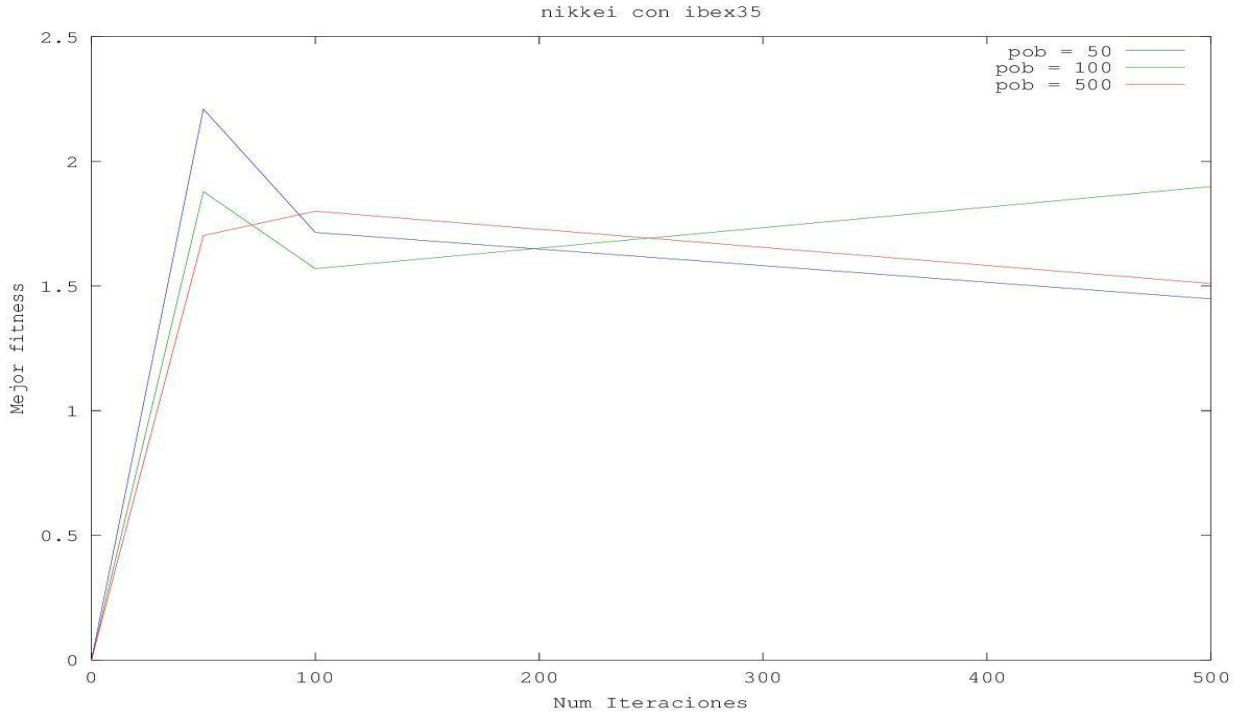


En estas gráficas se puede observar cómo se obtienen los mejores resultados al escoger una población de 50 y un número de iteraciones entre 50 y 100 en la mayoría de los casos; es extraño que la estrategia que encuentra sea la que menos gane en el Nikkei, ya que utilizan ambos el mismo conjunto de años e intuitivamente uno pensaría que se adaptaría mejor al Nikkei que a los demás, pero en este caso no parece que el uso del mismo período de tiempo tenga mucho peso.

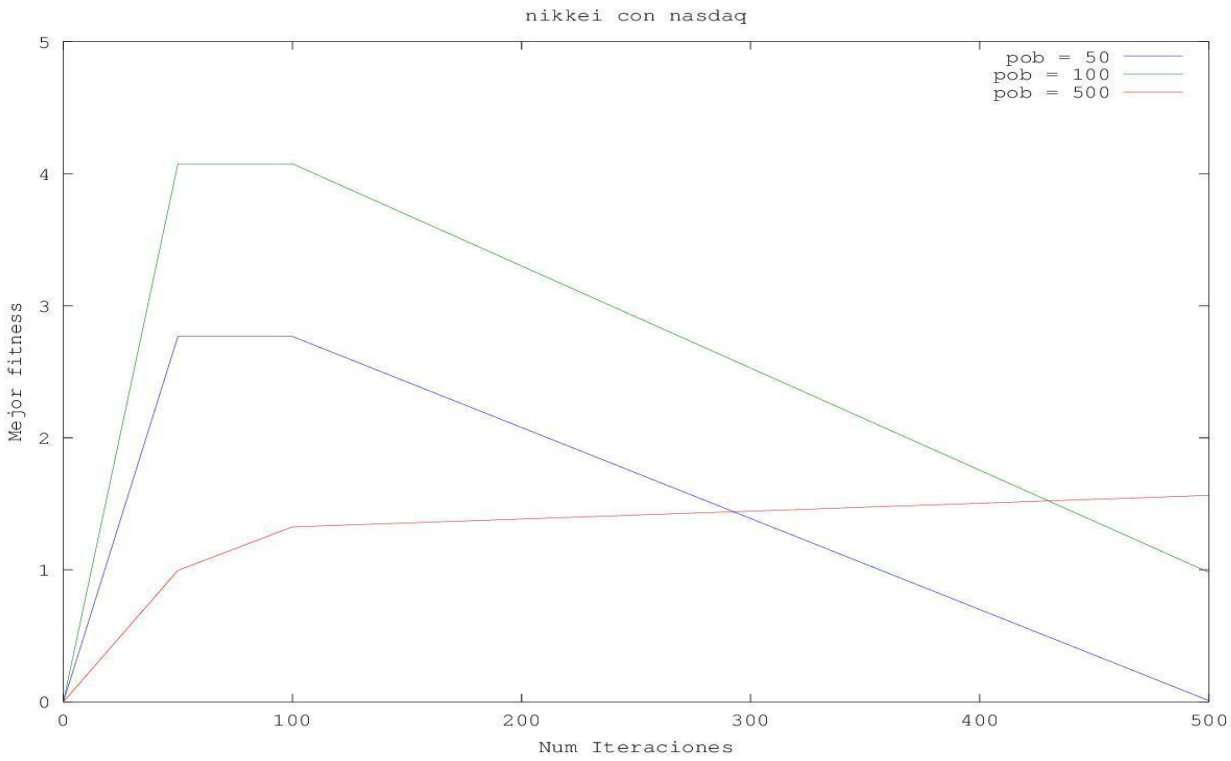
1.3.3 Nikkei

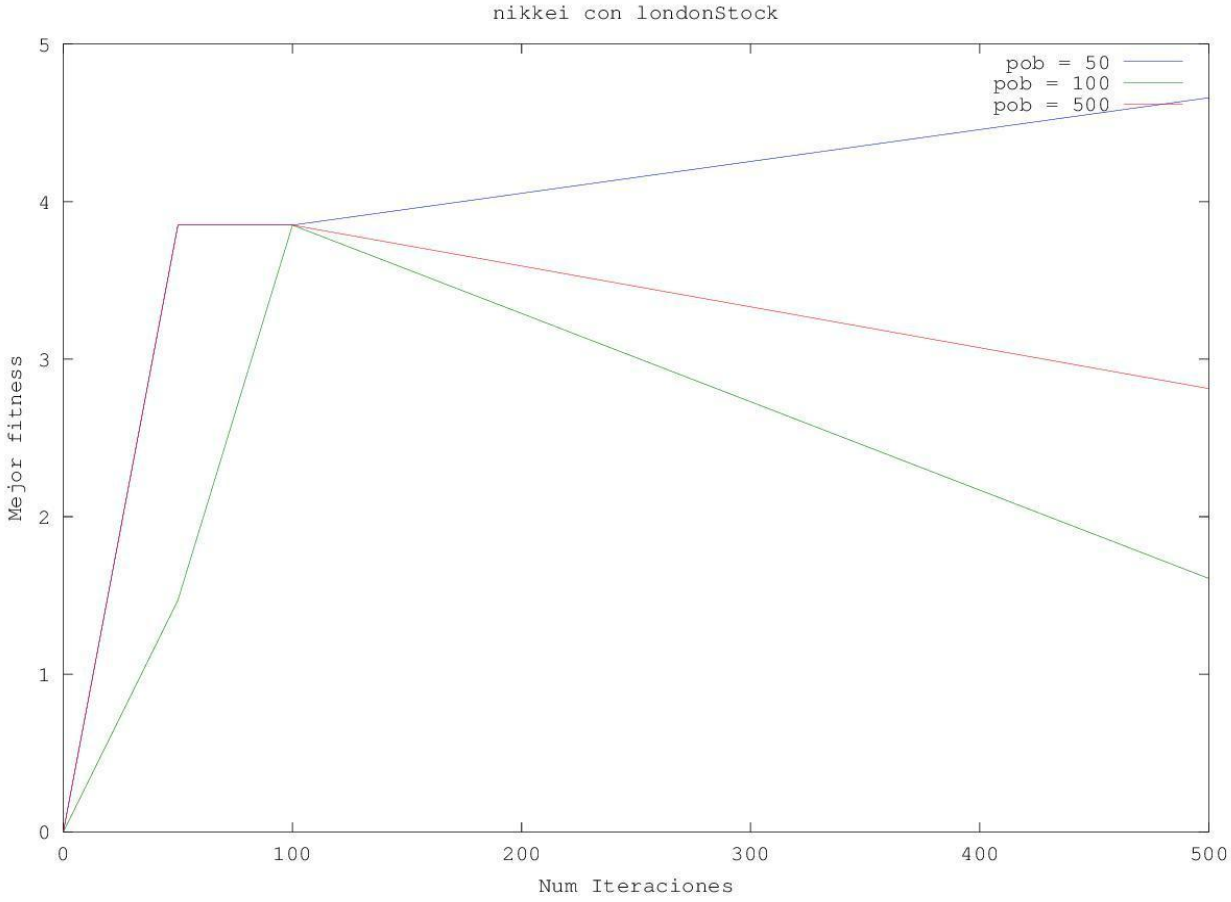


Al igual que en el anterior experimento, donde sí se aplicaban impuestos, se encuentran ganancias exageradamente grandes en comparación con los demás índices. Por lo demás la forma de las gráficas es similar a las ya vistas hasta ahora[11].



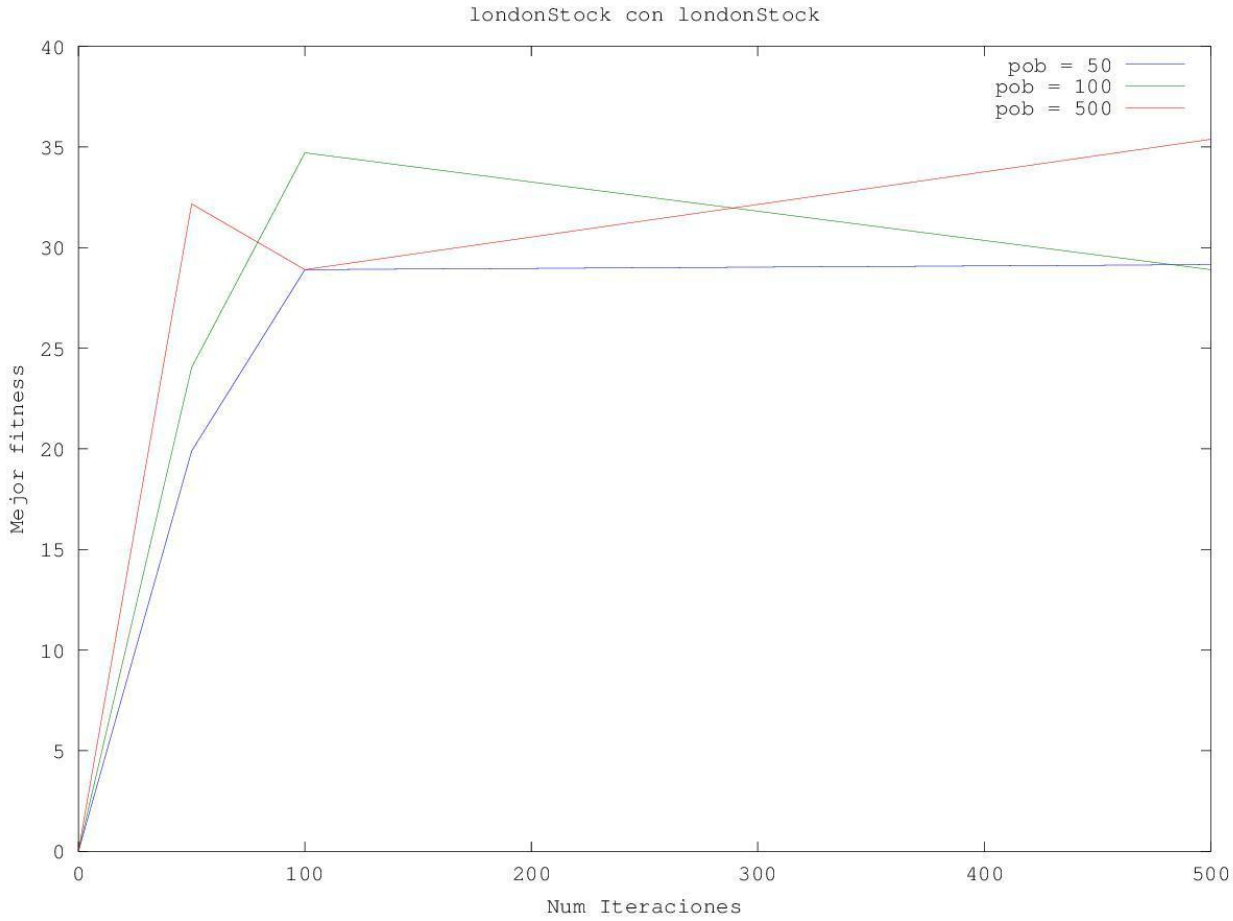
[11] Figuras 3.1.41-3.1.44. Gráficas del Nikkei sin impuestos


nikkei con nasdaq
pob = 50
pob = 100
pob = 500
Mejor fitness
Num Iteraciones
0
1
2
3
4
5
0
100
200
300
400
500


nikkei con londonStock
pob = 50
pob = 100
pob = 500
Mejor fitness
Num Iteraciones
0
1
2
3
4
5
0
100
200
300
400
500

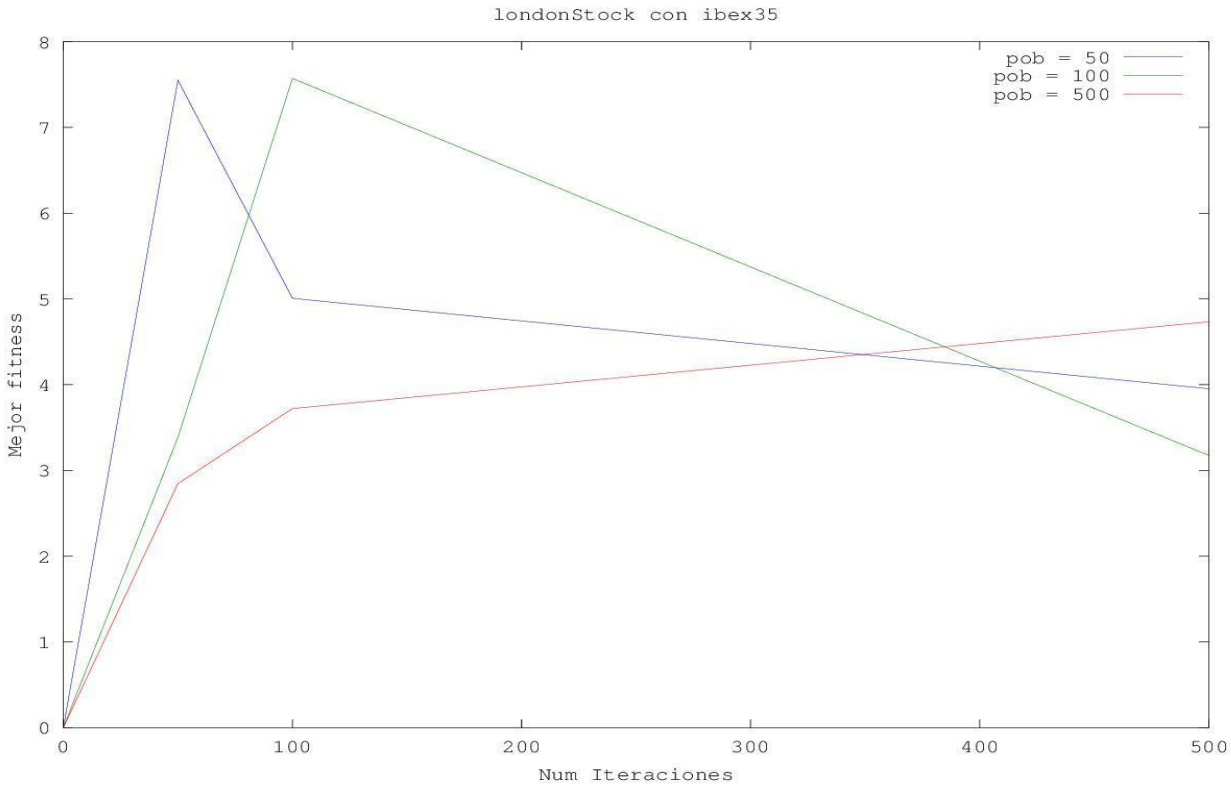
En general, las ganancias encontradas al probar la estrategia en otros índices son peores que al entrenar el algoritmo con el IBEX o el Nasdaq. Posiblemente el Nikkei sea muy propenso a producir un sobreajuste en la solución.

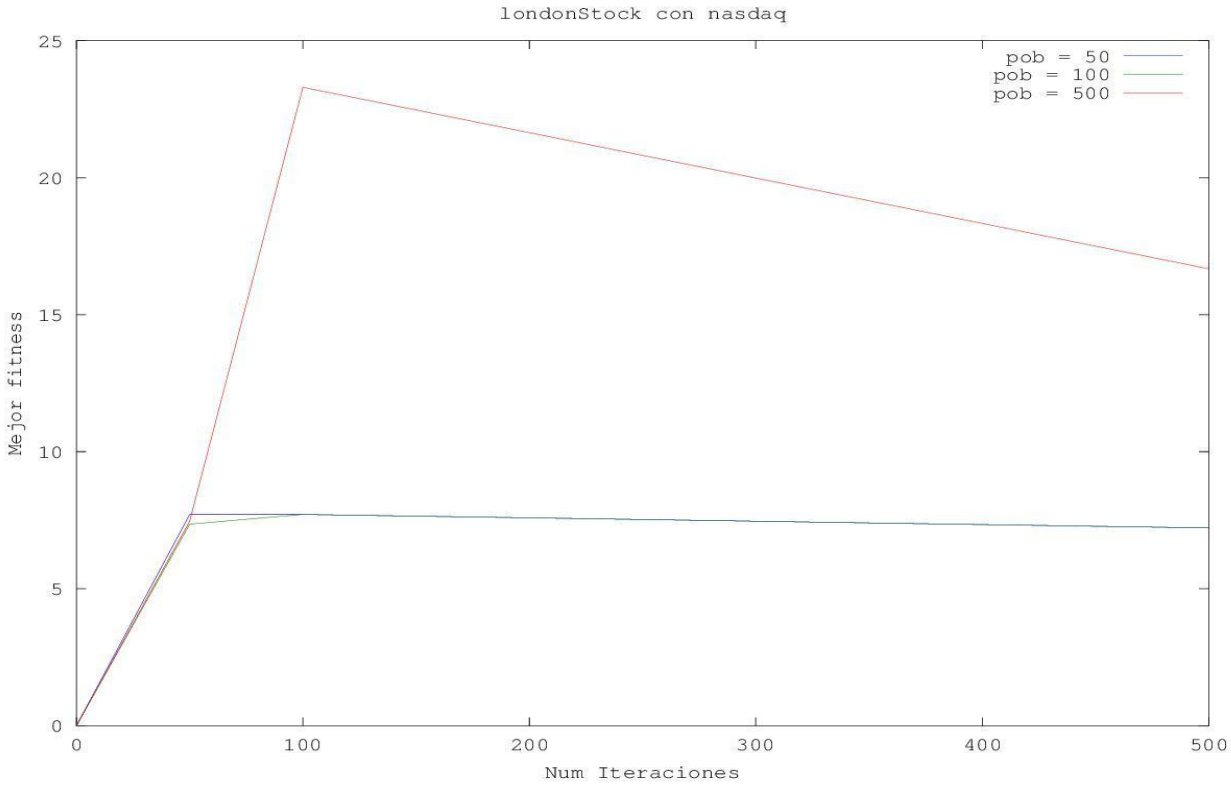
1.3.4 London Stock Exchange

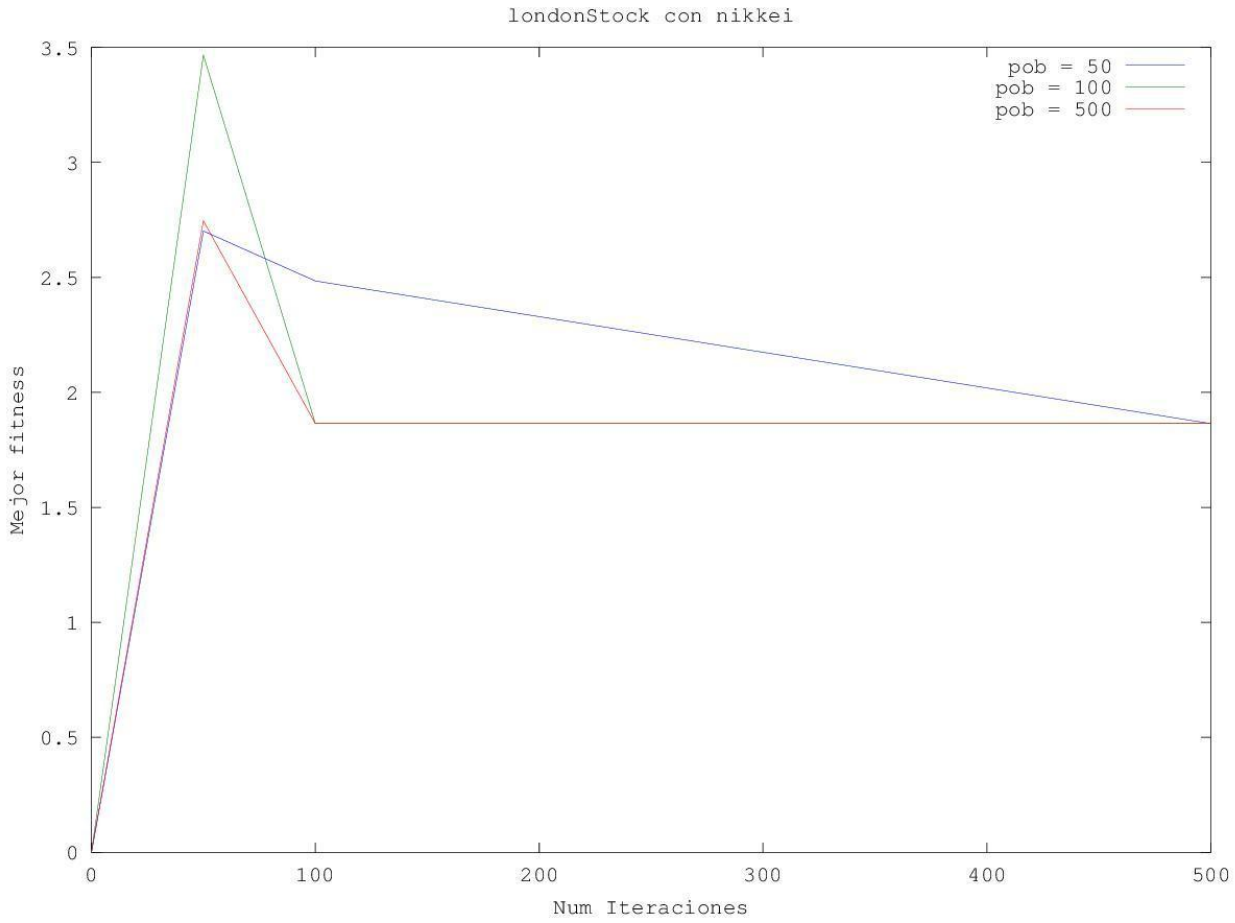


Las ganancias encontradas son bastante altas para los pocos años con los que se realiza la simulación de la bolsa londinense. De hecho, teniendo un período ocho años menor que el IBEX, encuentra casi el doble de ganancias[12].

[12] Figuras 3.1.45-3.1.48. Gráficas del London Stock Exchange sin impuestos


londonStock con ibex35
pob = 50
pob = 100
pob = 500
Mejor fitness
Num Iteraciones
0
1
2
3
4
5
6
7
8
0
100
200
300
400
500


londonStock con nasdaq
pob = 50
pob = 100
pob = 500
Mejor fitness
Num Iteraciones
0
5
10
15
20
25
0
100
200
300
400
500



Uno pensaría que, al encontrar tan buenos resultados en tan poco tiempo, se hubiera producido un sobreajuste, pero no parece ser el caso; se encuentran ganancias buenas al utilizar entre 50 y 100 individuos e iteraciones, aunque también se pueden observar situaciones extrañas como la del Nasdaq, en la que la estrategia parece funcionar muy bien utilizando 500 individuos y 100 iteraciones.

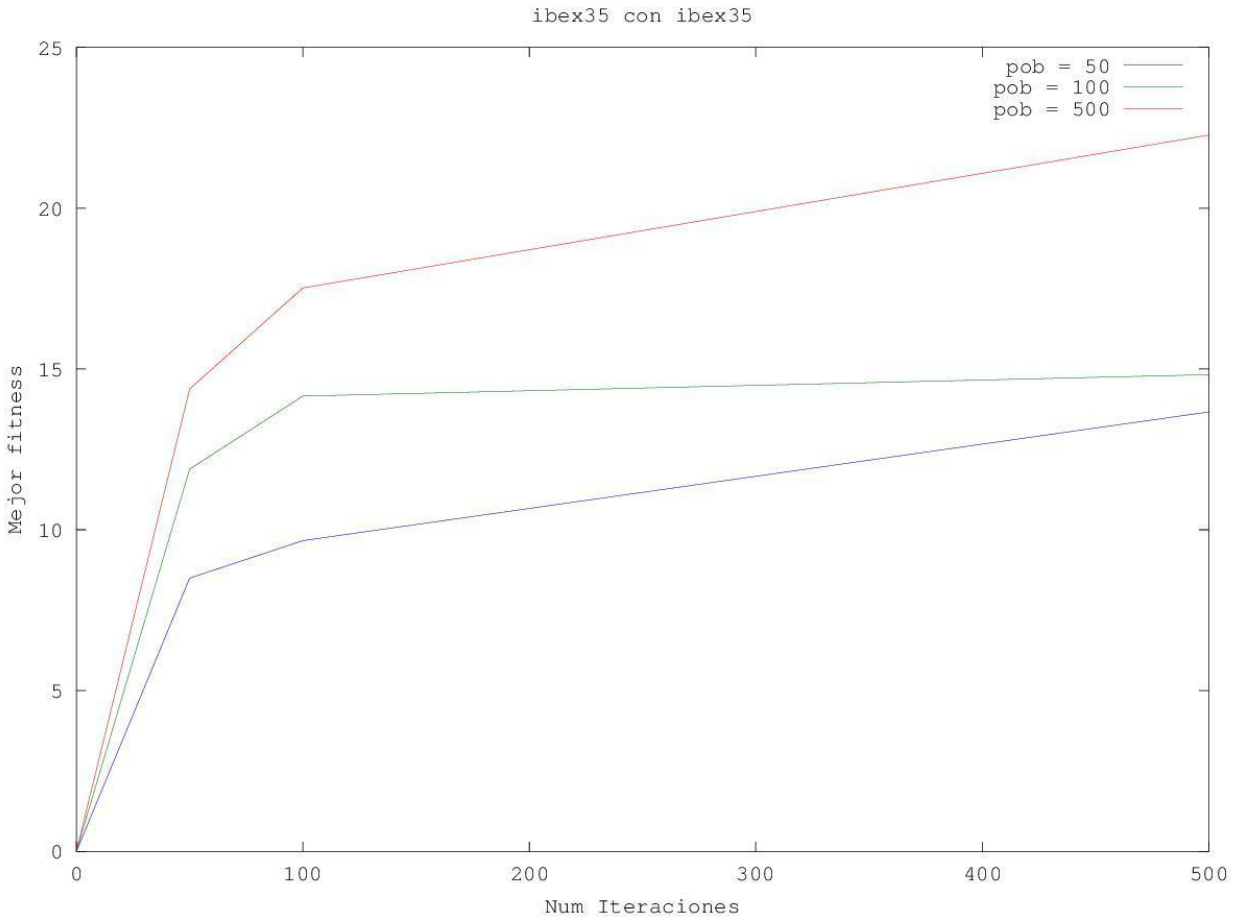
1.3.5 Conclusiones

Al quitar los impuestos se puede observar cómo el número de situaciones extrañas se reduce (aunque todavía están presentes) y cómo la evolución de la función de fitness es más parecida a la esquemática. Se puede ver que el peso que tienen los impuestos en el algoritmo genético es mayor que el peso que teóricamente deberían tener, y no parece que el algoritmo genético se adapte a este problema cuando hay que tener en cuenta impuestos, cánones y comisiones tal y como se haría en el mundo real.

1.4 Sin impuestos ni comisiones con seis indicadores

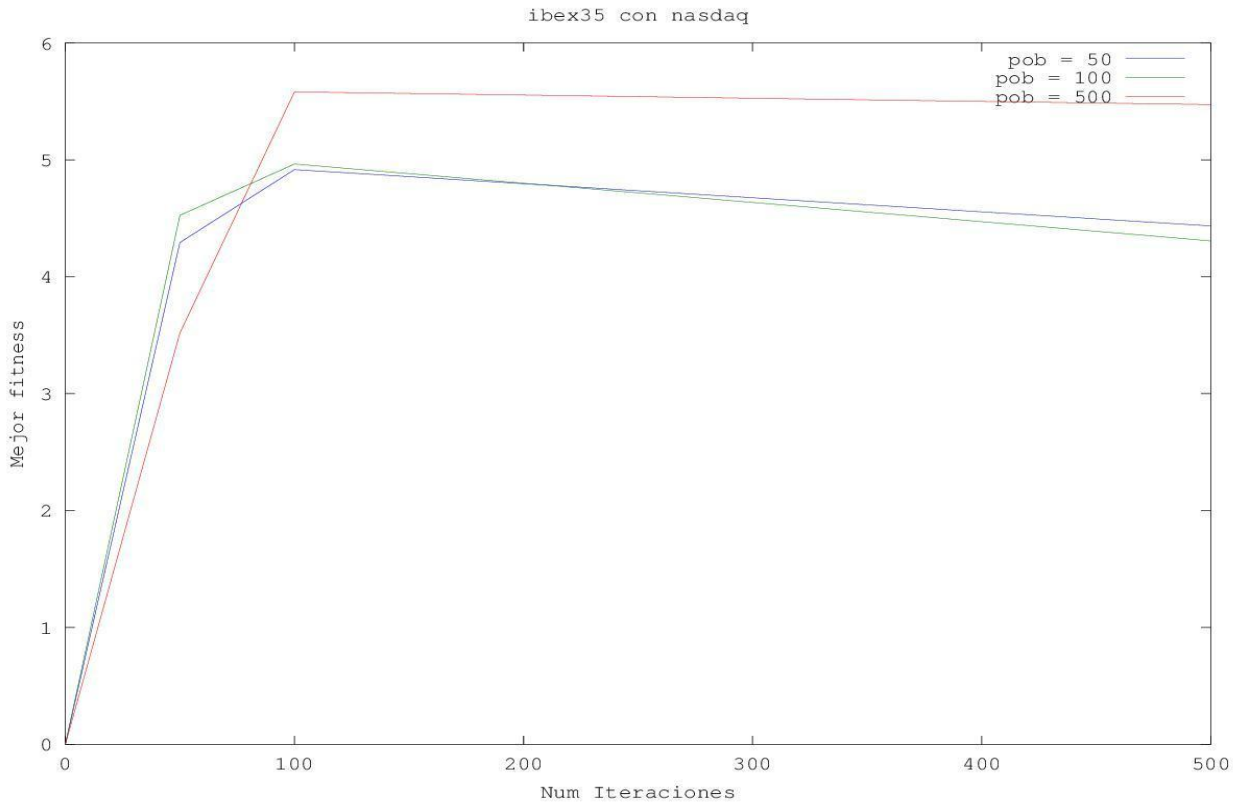
Ahora se realizarán las mismas pruebas que antes pero utilizando los seis indicadores mencionados al principio.

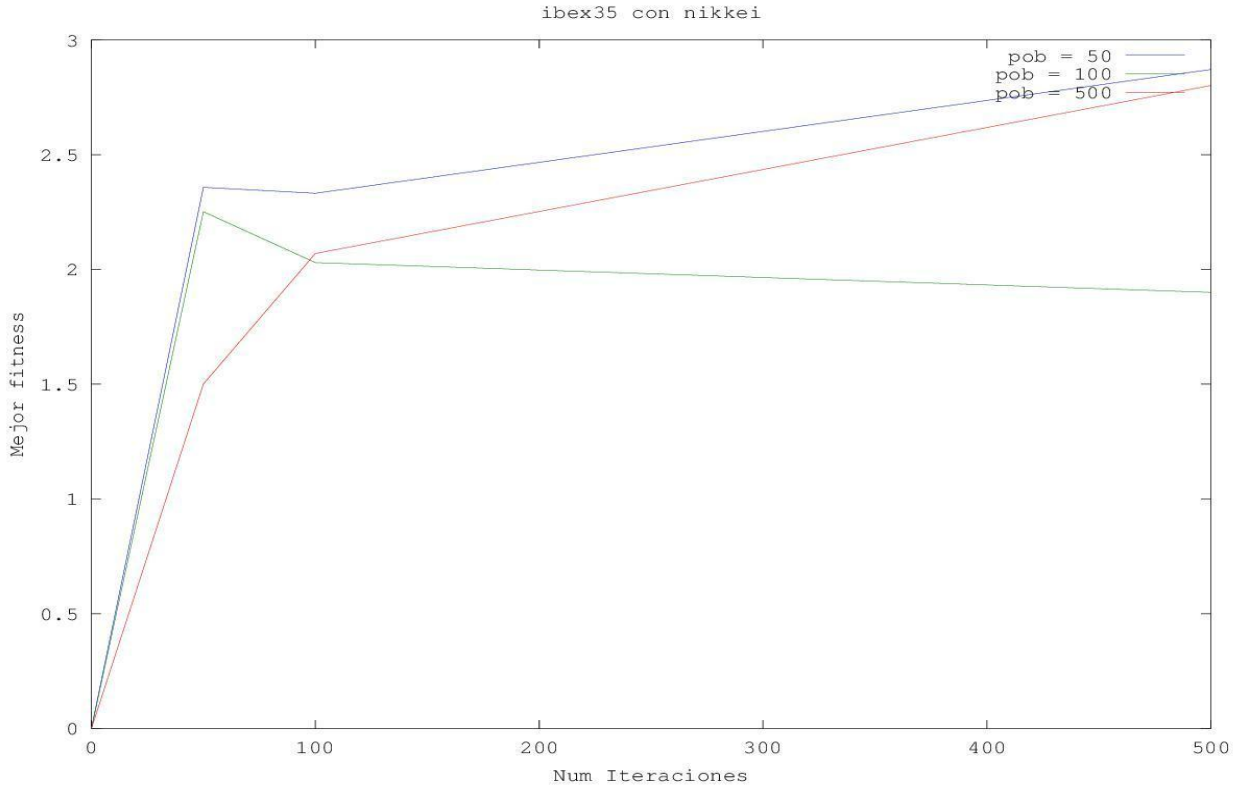
1.4.1 IBEX-35

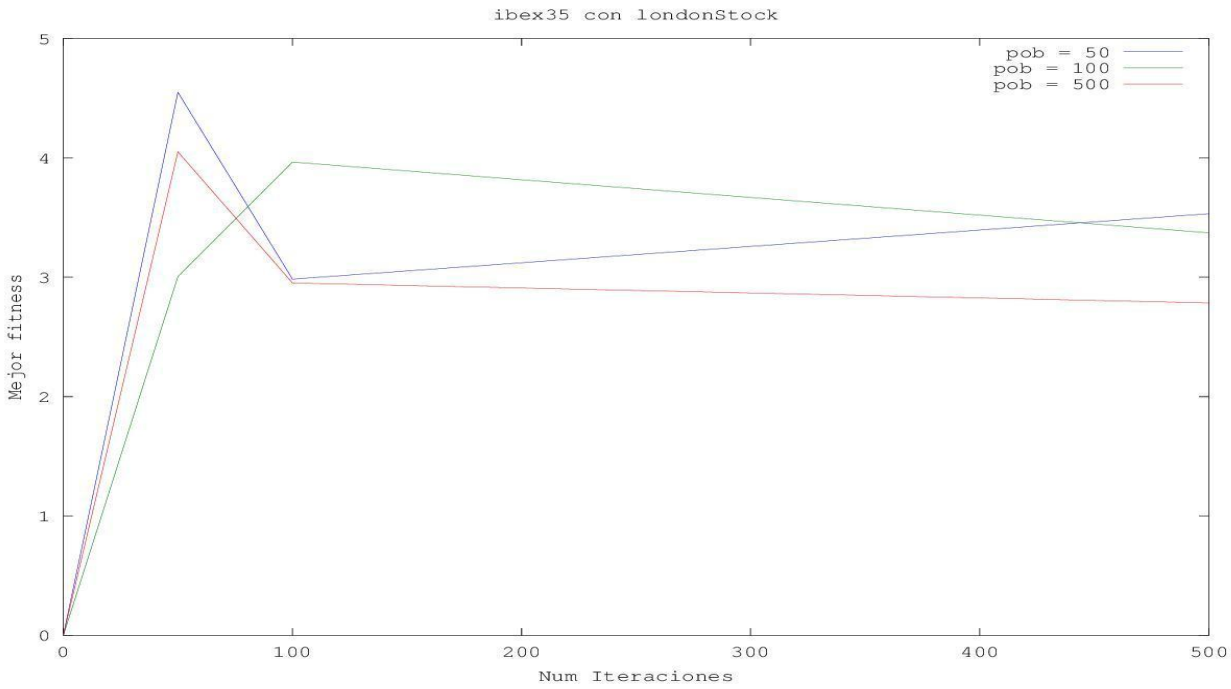


El uso de indicadores extra no parece ayudar a que se consigan mejores ganancias. De hecho, se consiguen resultados peores; también se observa que la función de fitness no crece tan lentamente como al usar tres indicadores. Posiblemente al añadir más indicadores el algoritmo necesite más tiempo para ajustar los parámetros entre todos sin que se produzcan contradicciones. Aun así, no parece que el mayor gasto de tiempo y espacio que supone el uso de estos indicadores extra merezca la pena vistas las ganancias[13].

[13] Figuras 3.1.49-3.1.52. Gráficas del IBEX-35 sin impuestos usando 6 indicadores

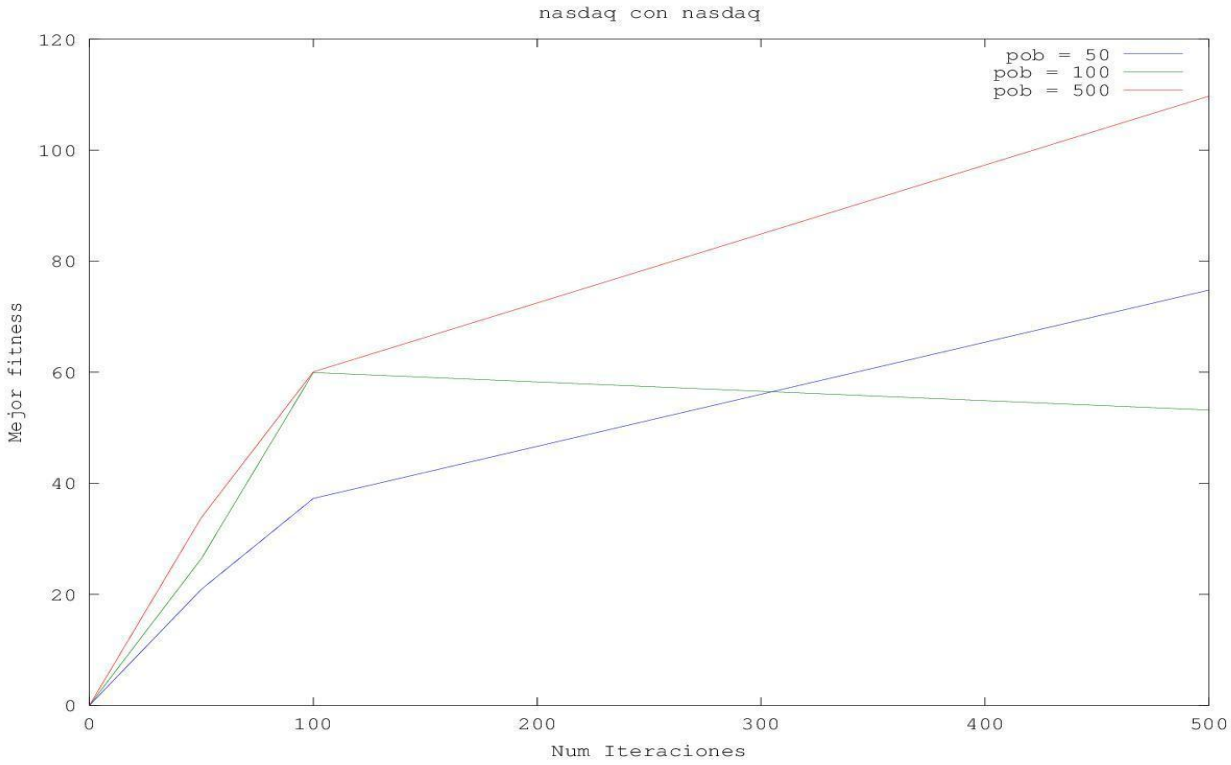

ibex35 con nasdaq
pob = 50
pob = 100
pob = 500
Mejor fitness
Num Iteraciones
0
1
2
3
4
5
6
0
100
200
300
400
500


ibex35 con nikkei
pob = 50
pob = 100
pob = 500
Mejor fitness
Num Iteraciones
0
0.5
1
1.5
2
2.5
3
0
100
200
300
400
500

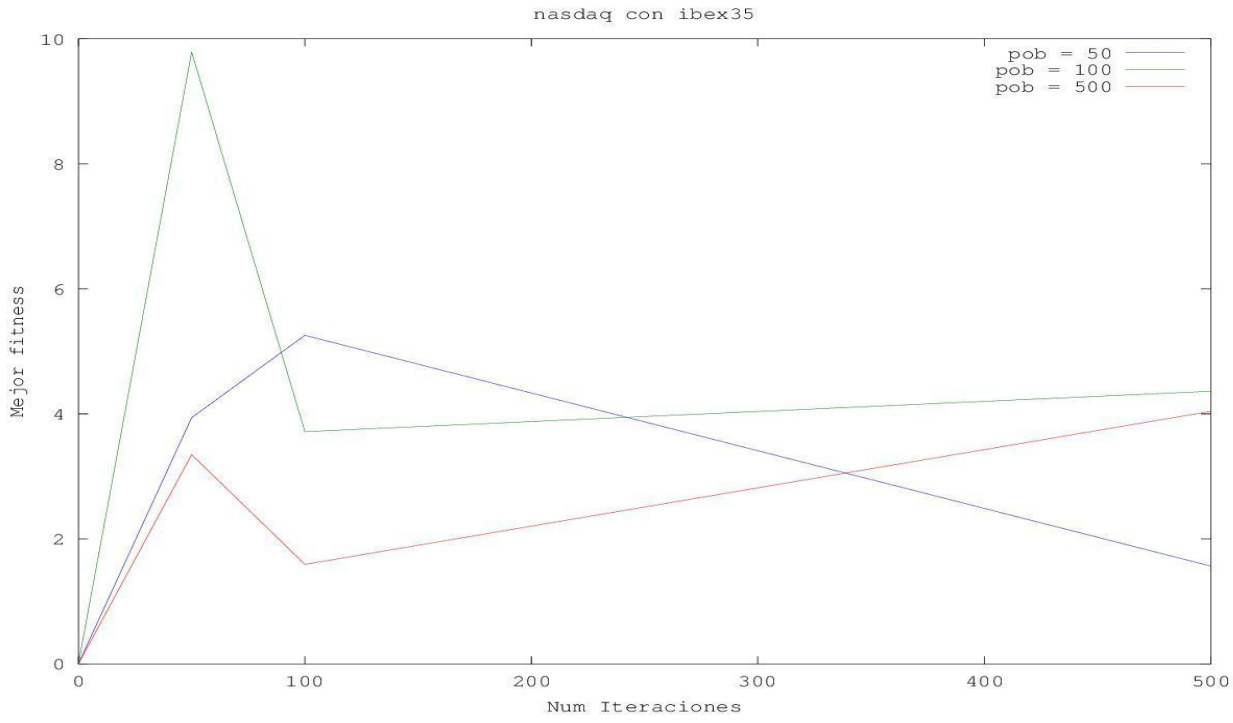


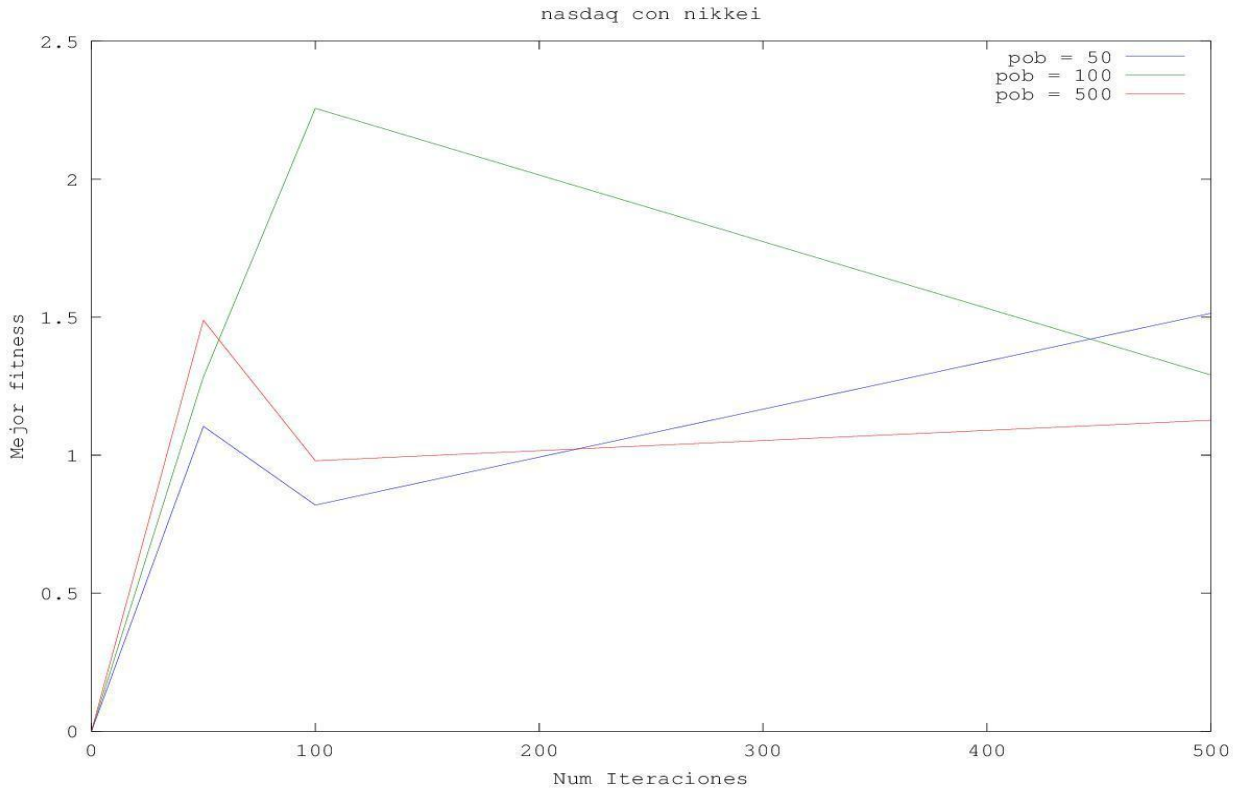
Se consiguen resultados parecidos a usar solo tres indicadores aunque en casos como el Nasdaq son ligeramente superiores; en general parece que no se llega al punto de sobreajuste tan rápidamente como antes, por lo menos en este caso.

1.4.2 Nasdaq

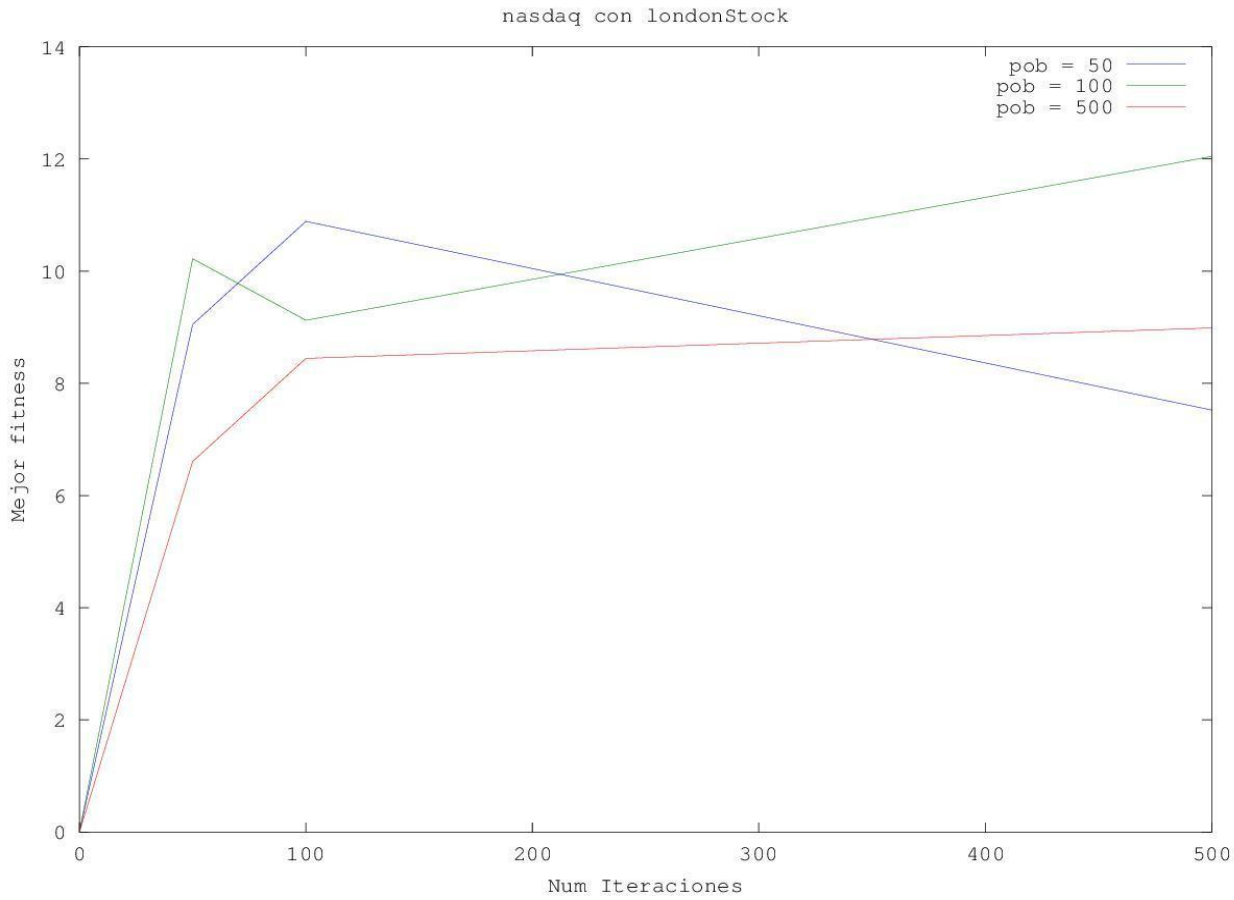


En este caso, se vuelve a observar que no aumentan mucho las ganancias respecto al uso de solo tres indicadores[14].



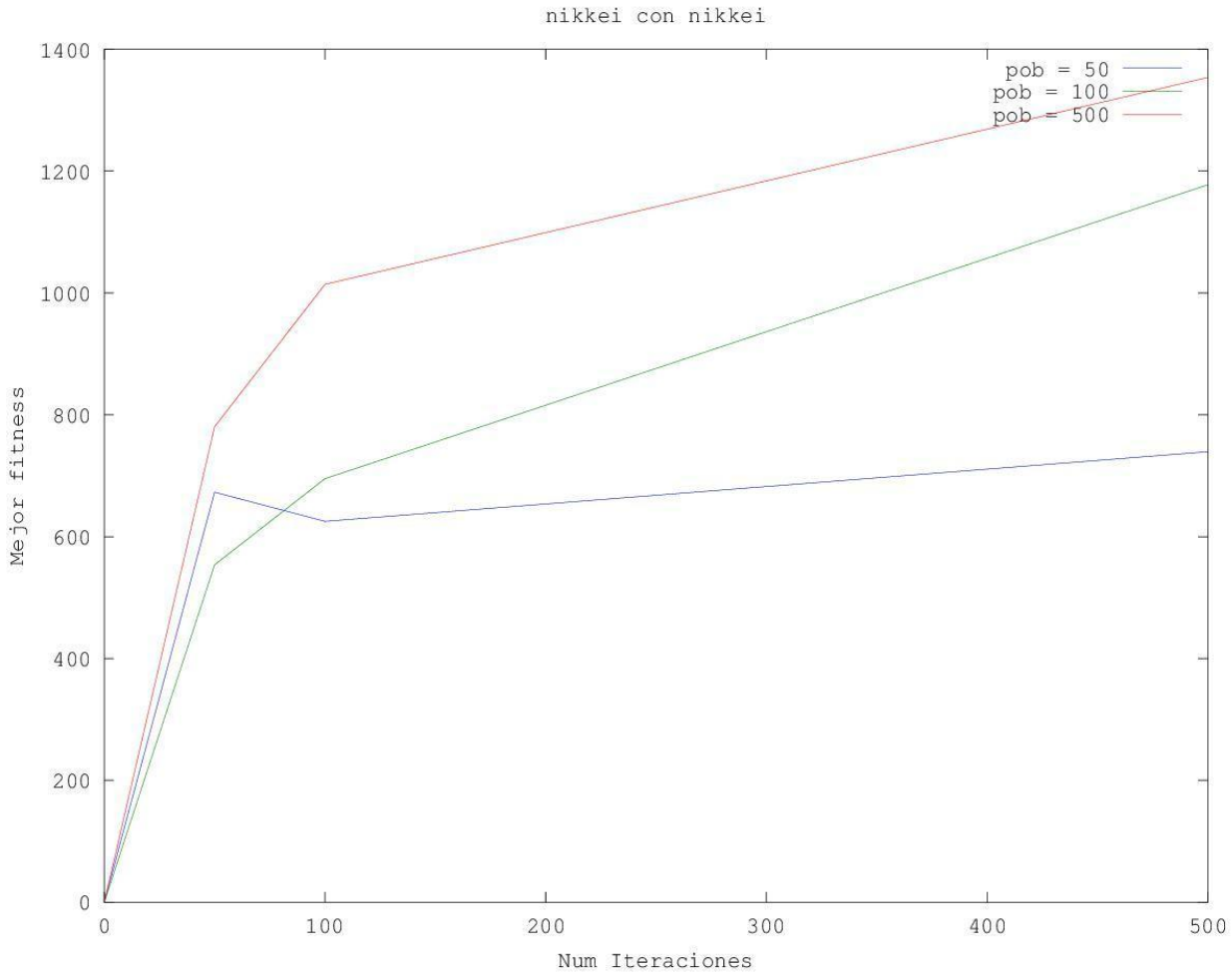


[14] Figuras 3.1.53-3.1.56. Gráficas del Nasdaq sin impuestos usando 6 indicadores



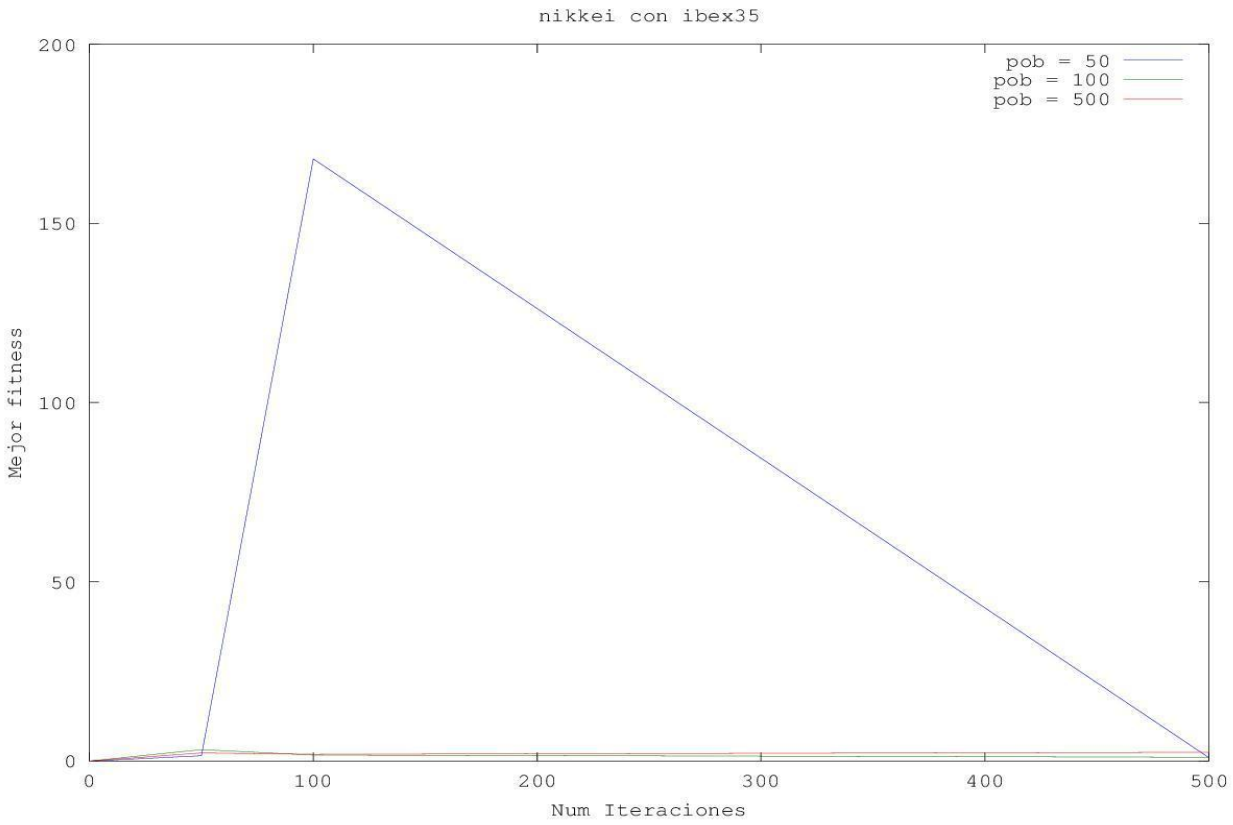
Se observa que en este caso se obtienen peores resultados que al utilizar solo tres indicadores. Esto se debe a un sobreajuste y es un fenómeno que suele ocurrir con los algoritmos evolutivos. Al añadir más atributos, se suele alcanzar más rápidamente el punto de sobreajuste.

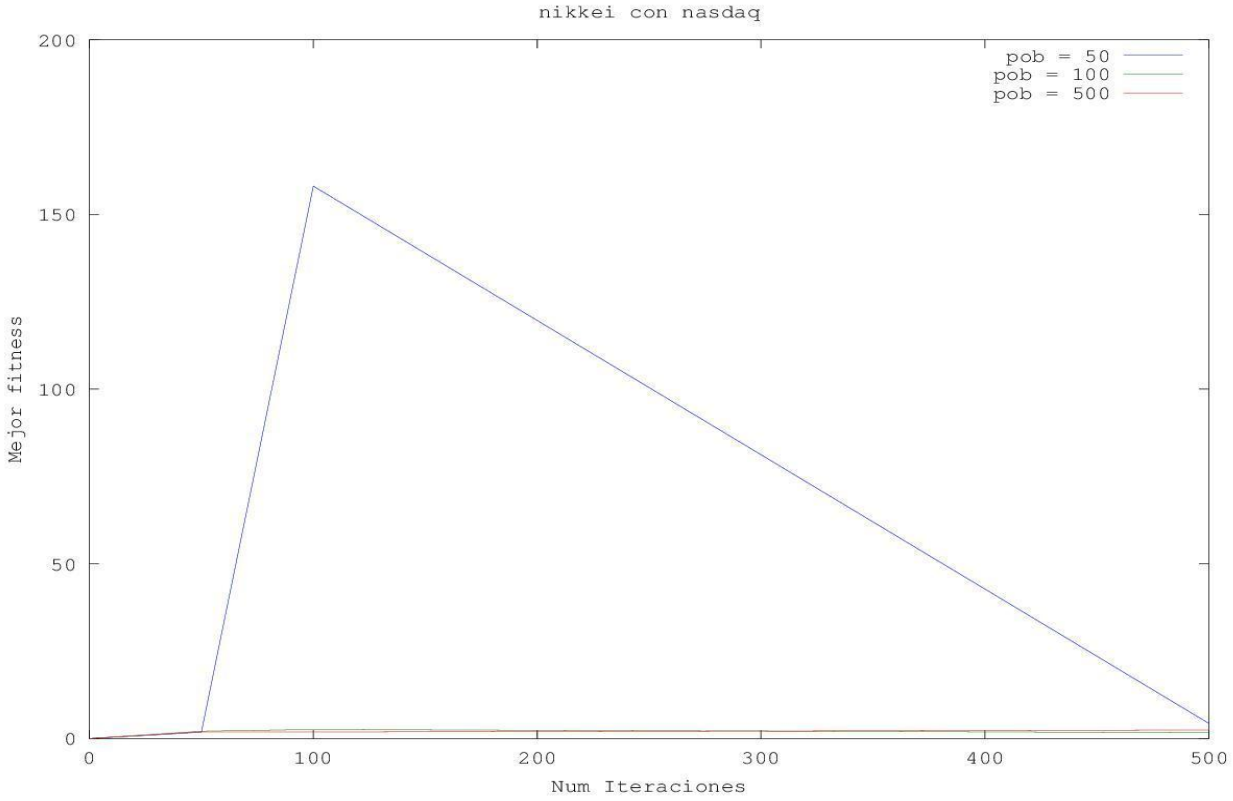
1.4.3 Nikkei

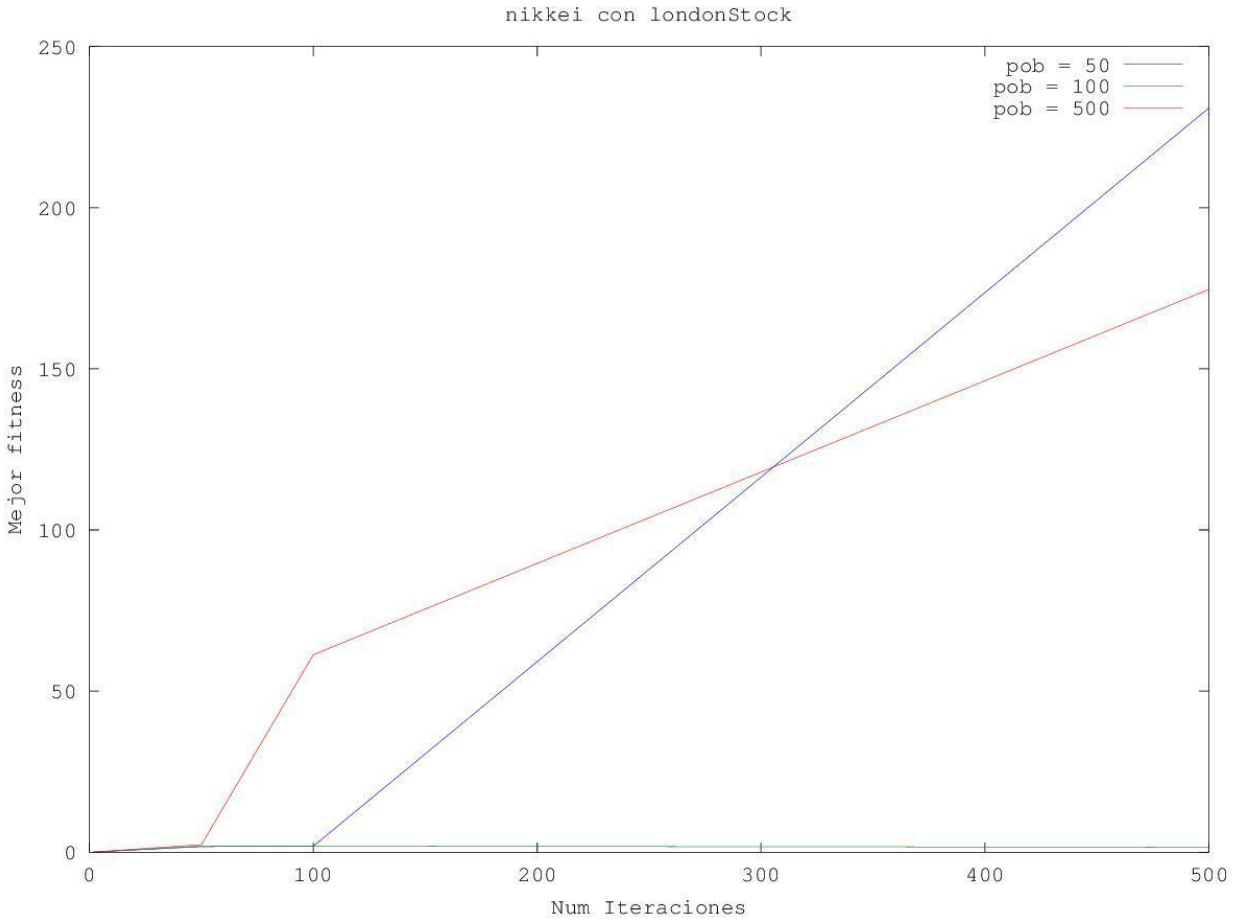


El Nikkei vuelve a obtener resultados exageradamente buenos. Añadir más índices ha mejorado casi en el doble las ganancias obtenidas antes, aunque posiblemente esto haya causado un sobreajuste gigantesco[15].

[15] Figuras 3.1.57-3.1.60. Gráficas del Nikkei sin impuestos usando 6 indicadores

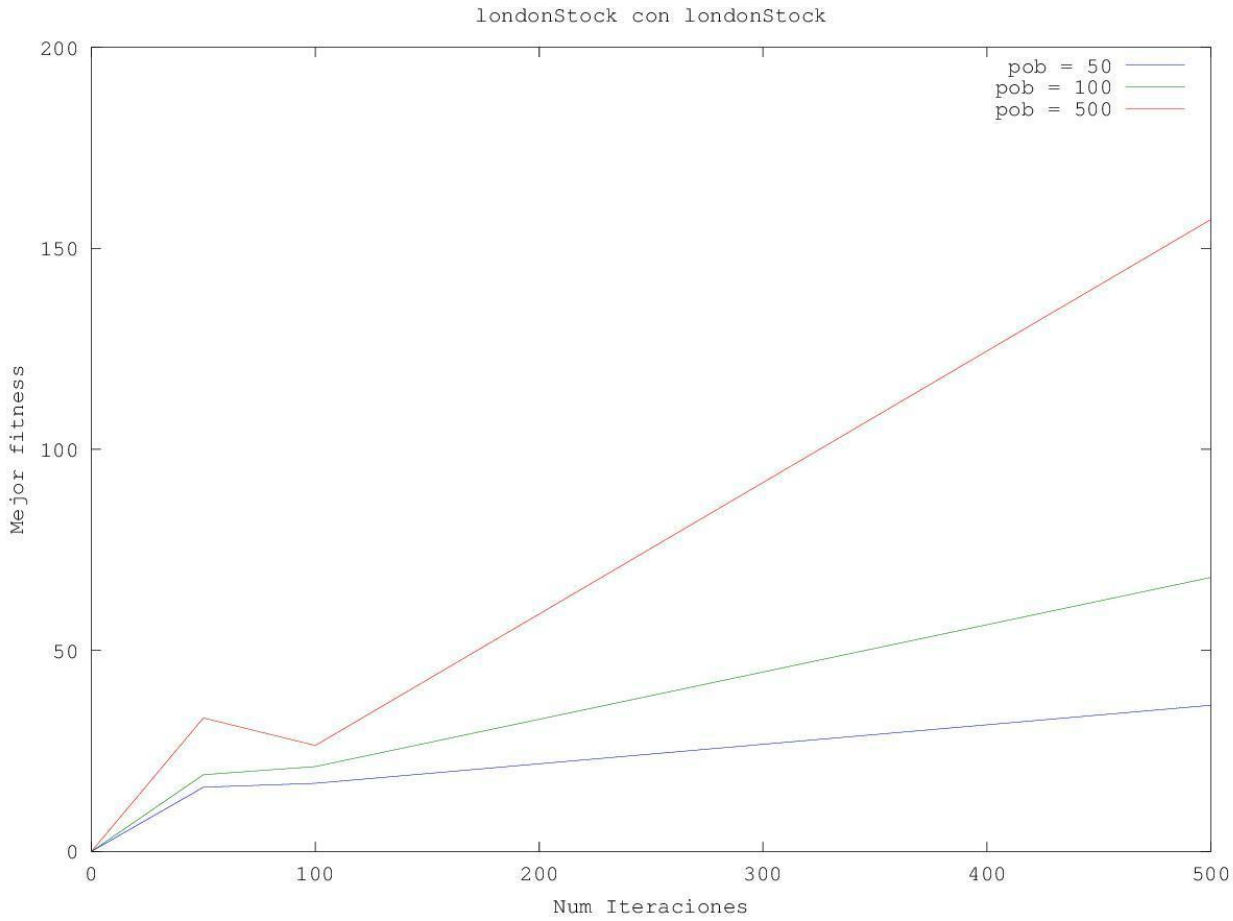

nikkei con ibex35
pob = 50
pob = 100
pob = 500
Mejor fitness
Num Iteraciones
0
50
100
150
200
0
100
200
300
400
500


nikkei con nasdaq
pob = 50
pob = 100
pob = 500
Mejor fitness
Num Iteraciones
0
50
100
150
200
0
100
200
300
400
500



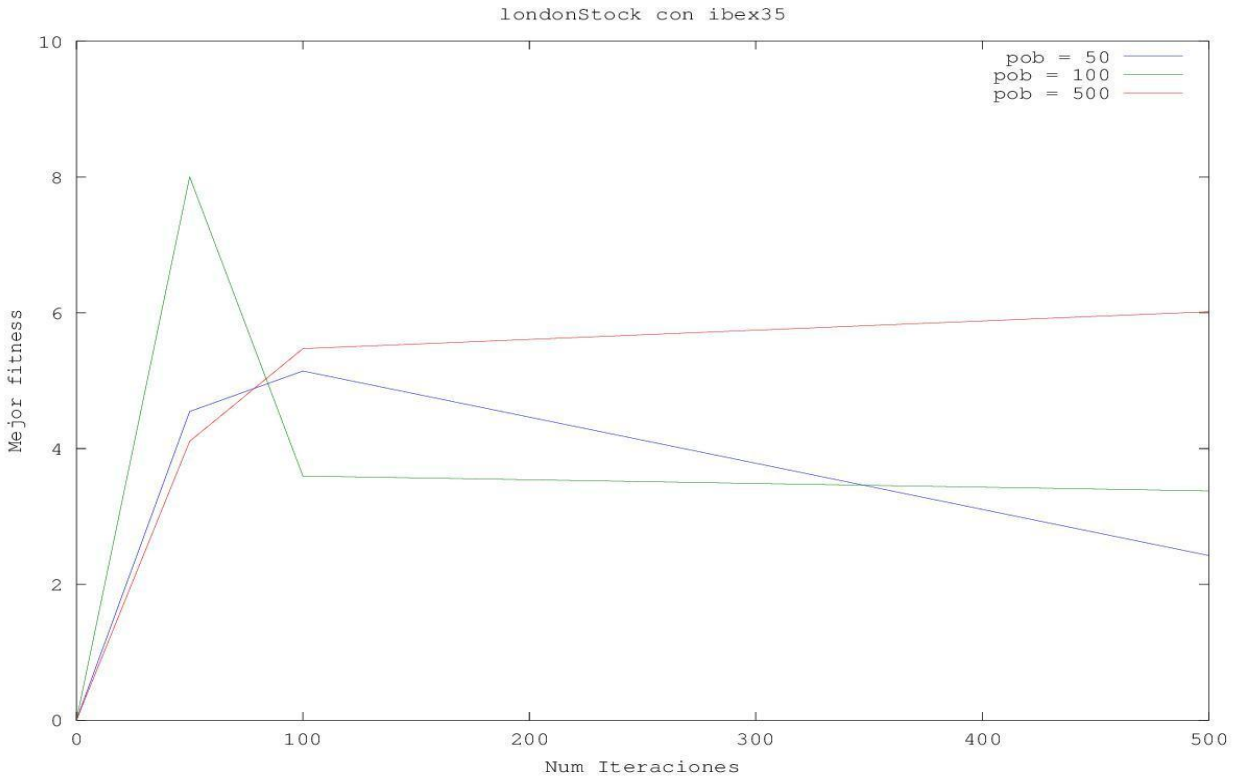
Al contrario de lo pensado inicialmente y de lo ocurrido con el Nasdaq, se obtienen resultados mucho mejores que al usar solo tres indicadores. De hecho, parece que al aumentar los indicadores se ha conseguido que el Nikkei se convierta en un índice idóneo para entrenar al algoritmo genético utilizando un número bajo de individuos e iteraciones. Es una pena que al incluir impuestos este comportamiento ya no sea así.

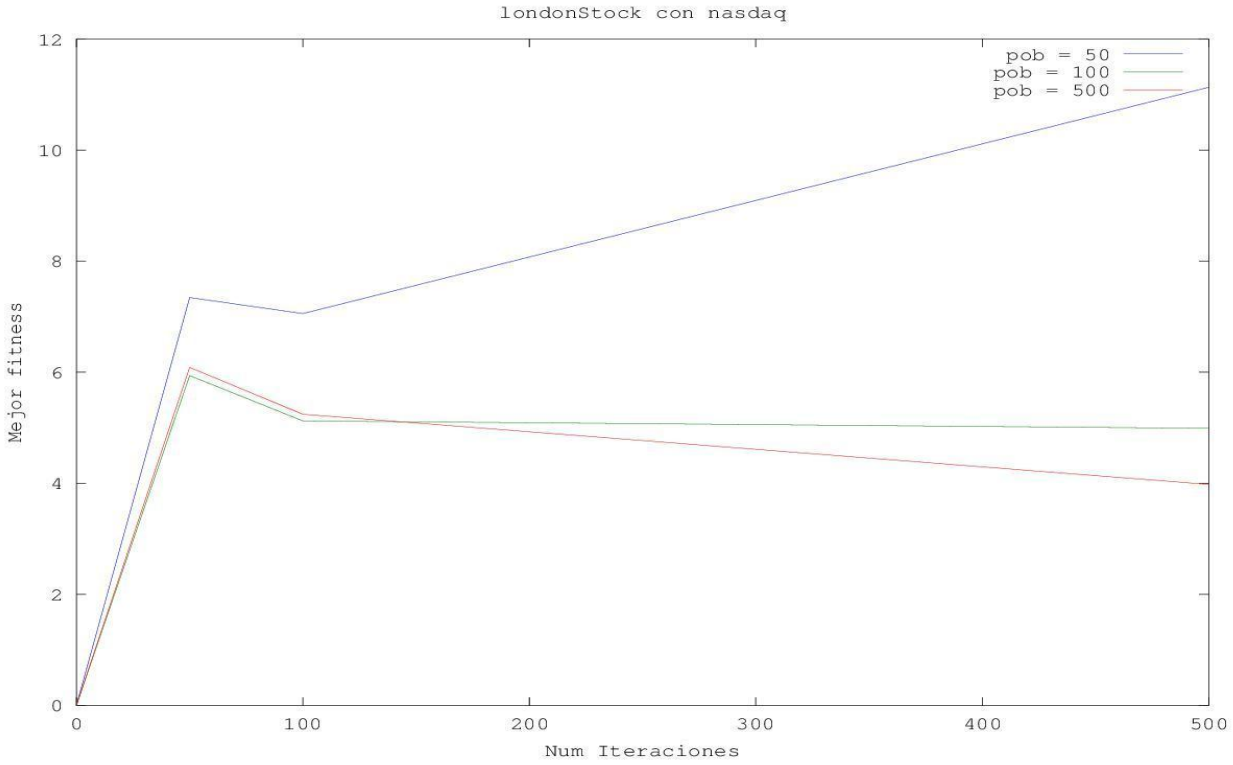
1.4.4 London Stock Exchange

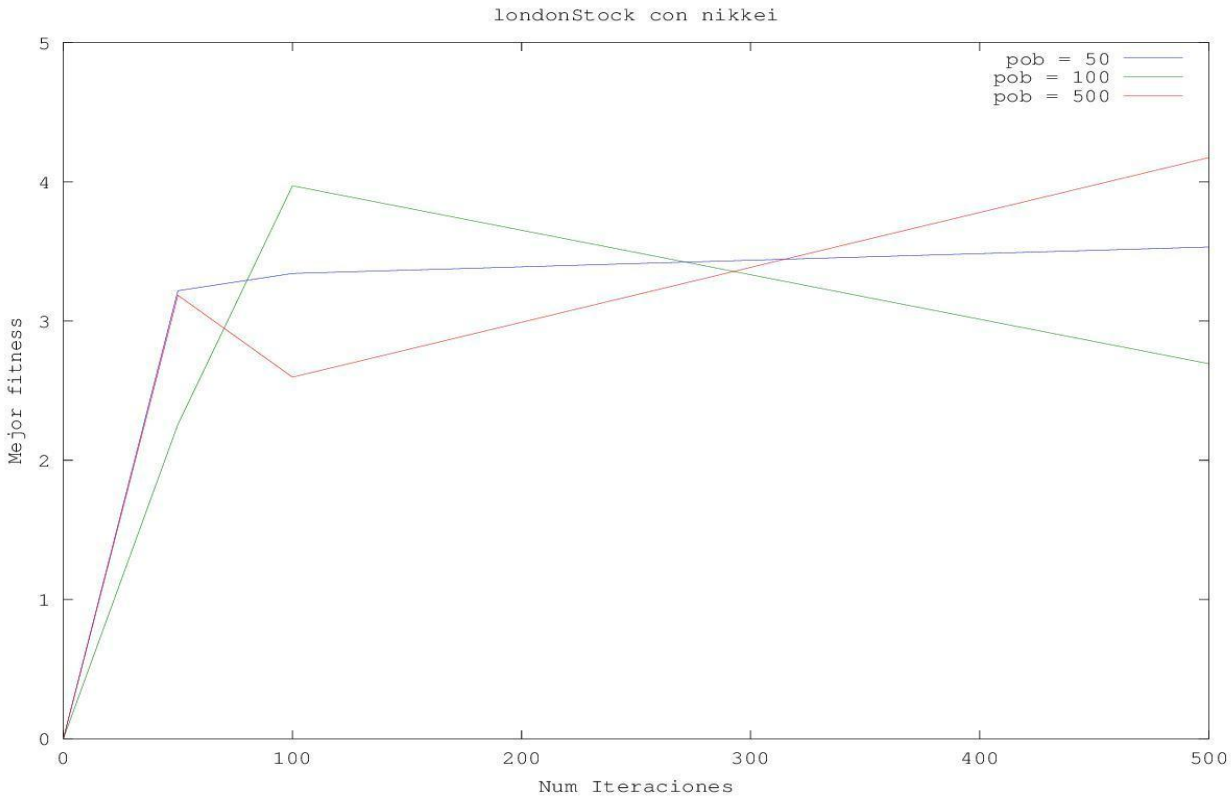


Los resultados se mejoran bastante al añadir indicadores, y además se puede volver a observar cómo la función de fitness crece bastante más deprisa en función al número de individuos y de iteraciones que al utilizar sólo tres indicadores[16].

[16] Figuras 3.1.61-3.1.64. Gráficas del London Stock Exchange sin impuestos usando 6 indicadores


londonStock con ibex35
pob = 50
pob = 100
pob = 500
Mejor fitness
Num Iteraciones
0
2
4
6
8
10
0
100
200
300
400
500


londonStock con nasdaq
pob = 50
pob = 100
pob = 500
Mejor fitness
Num Iteraciones
0
2
4
6
8
10
12
0
100
200
300
400
500



En este caso se obtienen ganancias parecidas a las de usar solo tres indicadores, pero en conjunto ligeramente peores, seguramente se haya producido un sobreajuste.

1.4.5 Conclusiones

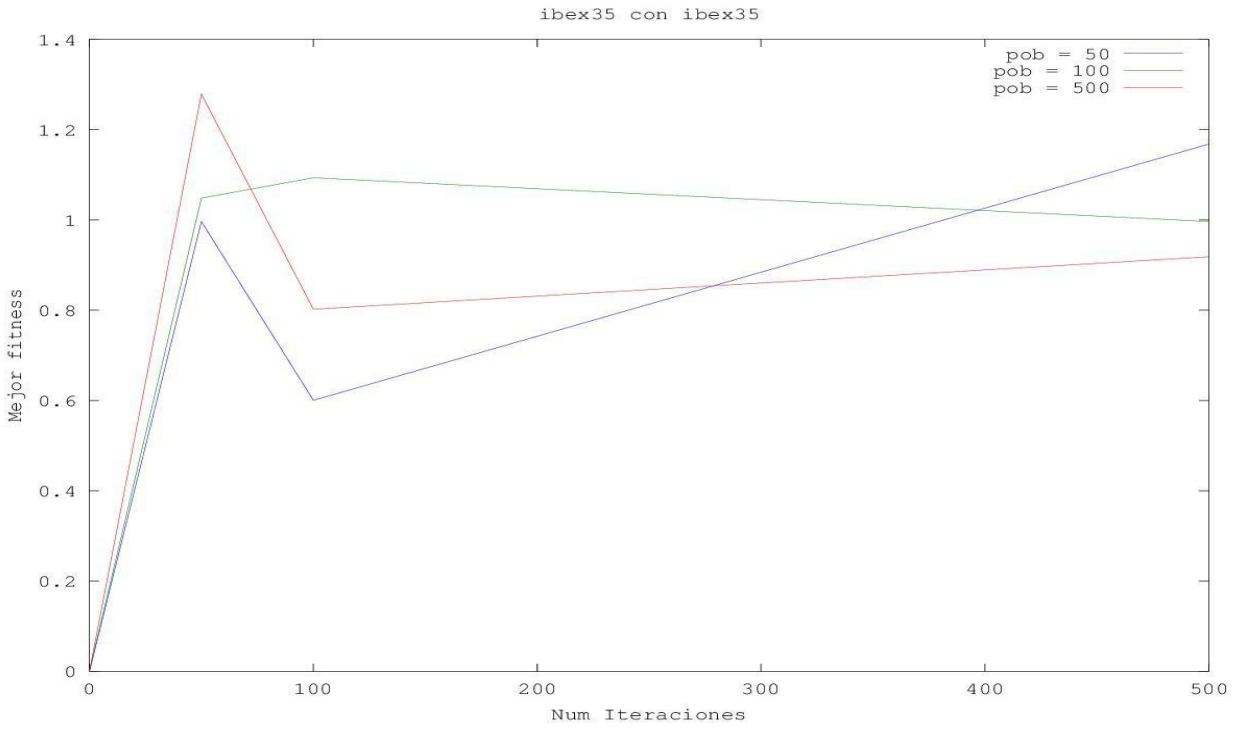
Los resultados son muy dispares en función de qué indicador haya sido usado para entrenar el algoritmo y es difícil sacar una conclusión. En el IBEX no se mejoran los resultados, y en el Nasdaq y en la bolsa de Londres se mejoran solo al entrenar y se produce un sobreajuste en los casos de validación. Sin embargo, con el Nikkei se mejoran mucho los resultados. En general parece que no merece la pena usar más indicadores debido al tiempo de computación y al espacio que añaden, ya que tarda el doble y ocupa dos veces más; además hay que tener en cuenta que esta es una versión sin impuestos no realista, y ya se han visto los resultados obtenidos al aplicar impuestos.

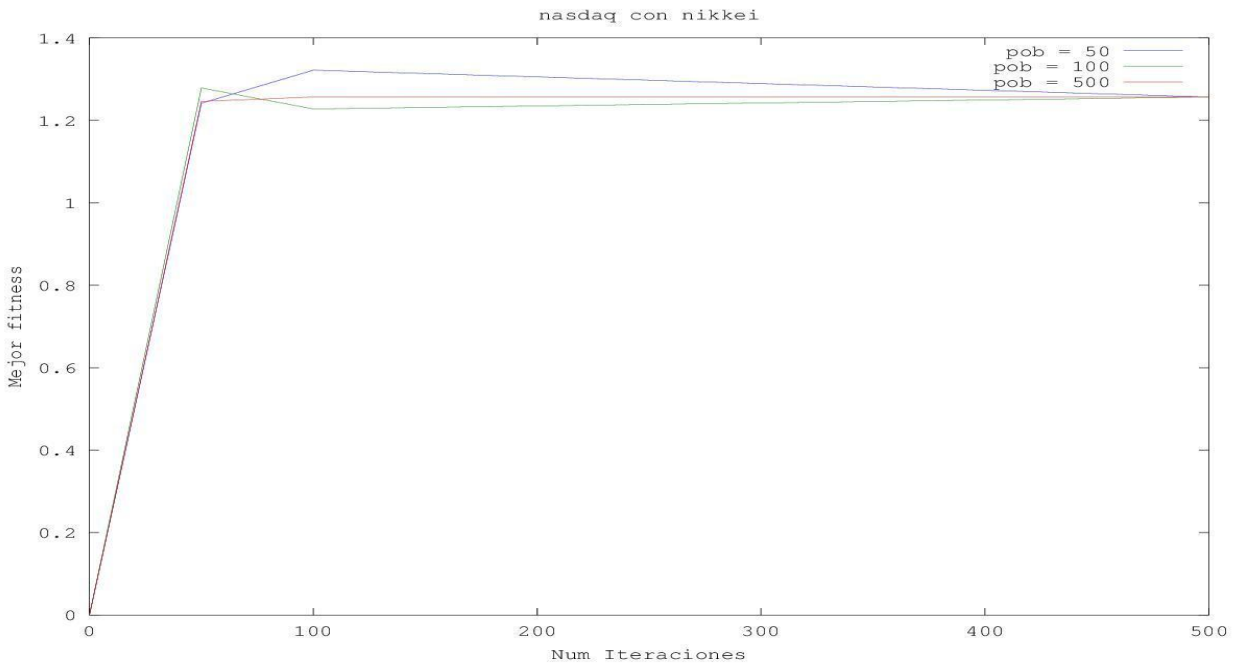
1.5 Tiempos disjuntos

Para estas pruebas se buscará una estrategia de inversión utilizando la primera mitad de los conjuntos de tiempo mencionados al principio, y se aplicará la estrategia a la segunda mitad de los conjuntos; primero con impuestos y después sin ellos. Como los resultados son muy parecidos a los ya presentados antes, no se tantas gráficas, solo las más significativas.

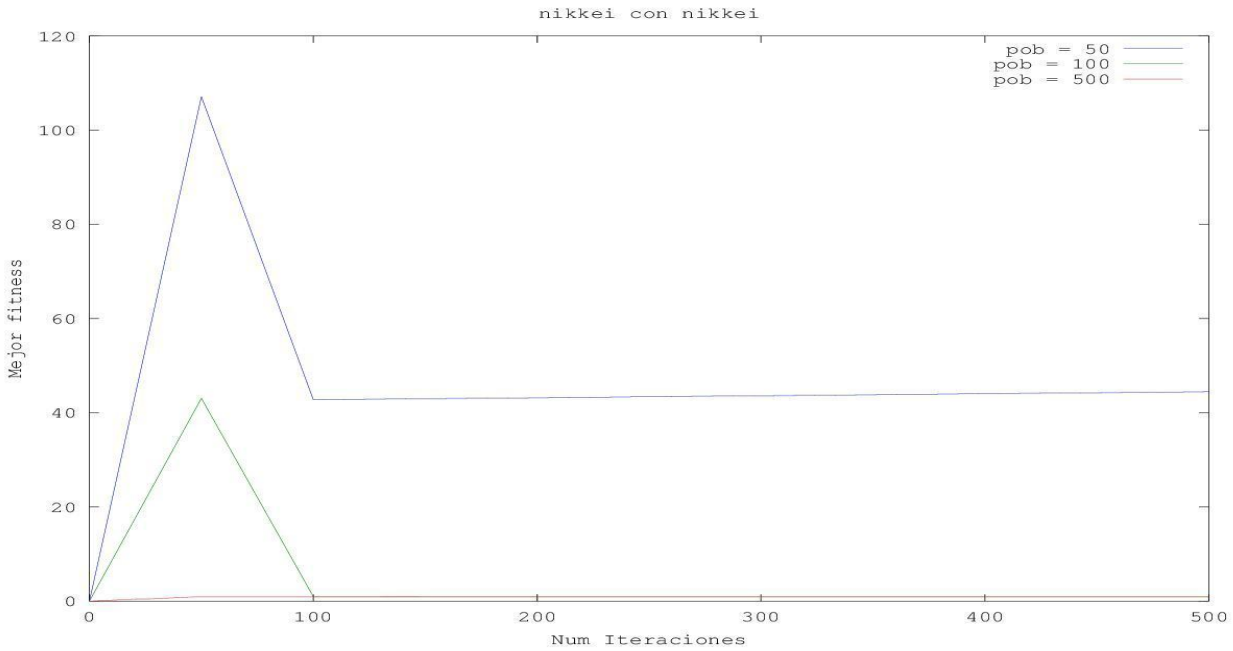
1.5.1 Con impuestos

Los resultados son semejantes a los anteriores, aunque en general se obtienen menos ganancias, es normal, teniendo en cuenta que ahora se dispone de la mitad de tiempo para invertir. Se incluyen algunas gráficas como ejemplo[17]:





[17] Figuras 3.1.65-3.1.67. Gráficas con tiempos disjuntos y con impuestos



Se puede observar que las estrategias encontradas cuando hay impuestos en la primera mitad del conjunto de tiempo funciona en la segunda mitad, incluso el Nikkei sigue obteniendo beneficios altos.

1.5.2 Sin impuestos

En este caso, no se obtienen tantas ganancias como antes, y no exclusivamente debido a que se utilice un conjunto de tiempo menor, si no también a que la primera mitad del conjunto fue anterior a la crisis, y la segunda mitad pertenece a la crisis, y las cotizaciones de la bolsa cambian drásticamente. Como ejemplo, se incluyen las cotizaciones del Nikkei durante el período de tiempo especificado y las ganancias obtenidas.

Figura 3.1.68. Cotizaciones del Nikkei

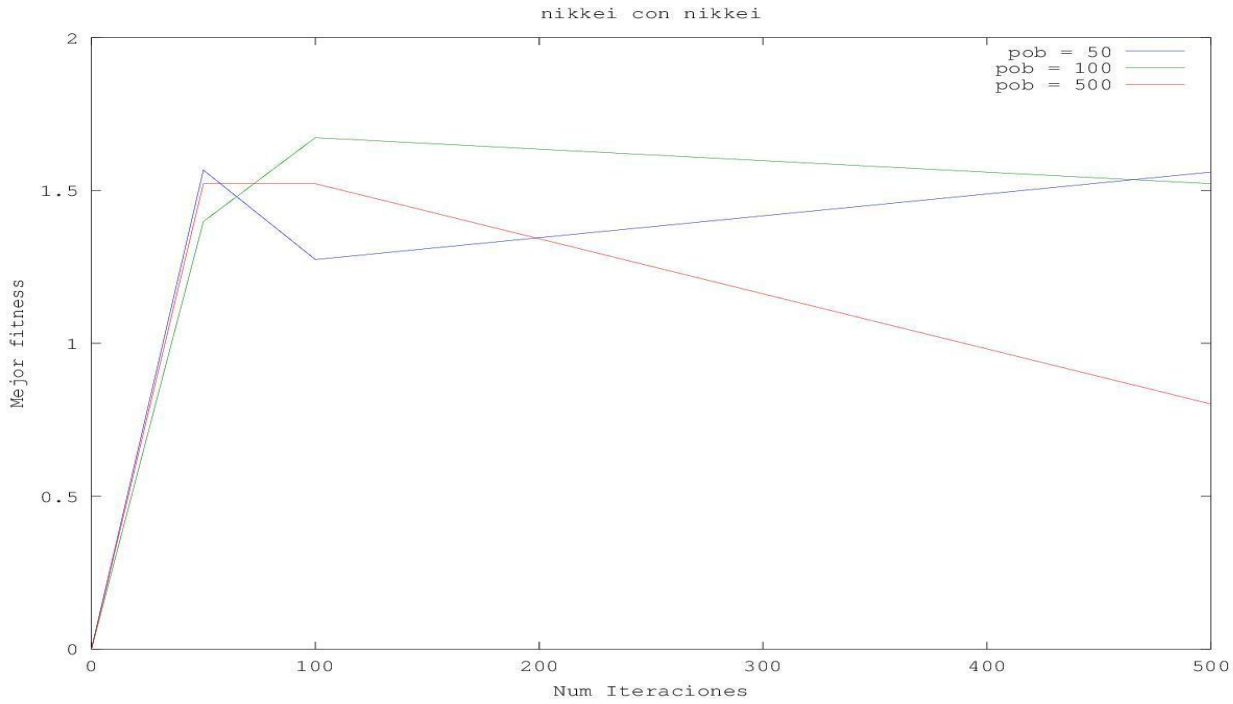


Figura 3.1.69. Gráfica Nikkei con tiempos disjuntos y sin impuestos

Como se puede ver en las cotizaciones del Nikkei, en la primera mitad hay una gran subida de la bolsa que se prolonga durante cuatro años. Al entrenar al algoritmo sólo con la primera mitad del tiempo, ajusta los parámetros de tal manera que compra antes de que comience la subida y vende al al alcanzar el máximo; esta

estrategia produce muy pocas operaciones de compra y venta, por eso, al tratar de aplicarla a la segunda mitad, produce tan pocas ganancias.

Este fenómeno resulta extraño, ya que al tener en cuenta impuestos se encuentran mejores ganancias. ¿Por qué ocurre esto? Cuando se tienen en cuenta impuestos y demás gastos, el peso que tienen los costes de custodia es mayor al peso de las comisiones; en consecuencia, cuando hay impuestos, decide realizar más operaciones de compra y de venta, lo que hace que la estrategia sea más general, en vez de comprar las acciones cuando están bajas, mantenerlas durante cuatro años, y venderlas cuando están altas.

Este fenómeno, en menor medida, se produce también con el resto de índices, y donde antes se tenían altas ganancias, ahora son menores, y en general resultan similares a la estrategia de comprar el primer día y vender el último.

1.5.3 Conclusiones

En conclusión, si se quiere entrenar al algoritmo es recomendable utilizar un período de tiempo con un comportamiento regular para que sea más general. Como es inevitable que aún así se puedan presentar largas subidas o bajadas en la bolsa en el futuro, se podría considerar modificar el algoritmo de tal manera que actualizara sus datos de manera automática, y hacer que la función de fitness diera más peso a los datos de los últimos días que a los demás, para así tratar de adaptarse más rápidamente a las nuevas tendencias.

1.6 Conclusiones generales

El algoritmo genético no funciona correctamente cuando se tienen en cuenta impuestos, cánones y comisiones. Sin embargo, al quitarlos tiene un funcionamiento bastante bueno incluso al probarlo con distintos índices en conjuntos de tiempos no iguales; no parece que merezca la pena en general añadir más indicadores además del EMA, MACD y RSI salvo en el caso excepcional del Nikkei, que parece un índice idóneo para entrenar al algoritmo siempre y cuando podamos evadir impuestos y demás gastos, y se utilicen todos los datos conocidos para entrenar al algoritmo.

2. Algoritmo genético avanzado

El algoritmo genético avanzado trata de maximizar la cantidad de dinero ganando invirtiendo a la vez en varios índices. Para nuestras pruebas dispusimos de las cotizaciones de las empresas del IBEX-35 Santander, Telefónica, Inditex, Iberdrola y Amadeus, organizadas en dos conjuntos: $C_1 = \{ Santander,\ Telefónica,\ Inditex \}$ y $C_2 = \{ Inditex,\ Iberdrola,\ Amadeus \}$.

Se entrena el algoritmo en uno de los conjuntos y se prueba en el otro, de tal manera que aplica la estrategia encontrada para la empresa *i* del primer conjunto en la empresa *i* del segundo conjunto.

Igual que antes, recordemos que las ganancias representadas es el cociente $dinero\ inicial\ /\ dinero final$.

El plan de pruebas es similar al anterior aunque, debido a la mayor complejidad de este algoritmo genético, se reducirá el número máximo de iteraciones y de individuos. Los conjuntos utilizados son: $I = \{50, 100, 200\}$ y $T = \{50, 100, 200\}$, y *n* = 3.

Se mostrarán primero los resultados de entrenar el algoritmo con un conjunto y probarlo en el otro con impuestos. Después se probará a quitar los costes de custodia, repetir el proceso y comparar resultados. Ambas pruebas se realizarán utilizando solo los indicadores MACD, EMA y RSI, y utilizando las cotizaciones entre los años 2001 y 2013. Se realizan menos pruebas que antes por la mayor complejidad de este algoritmo.

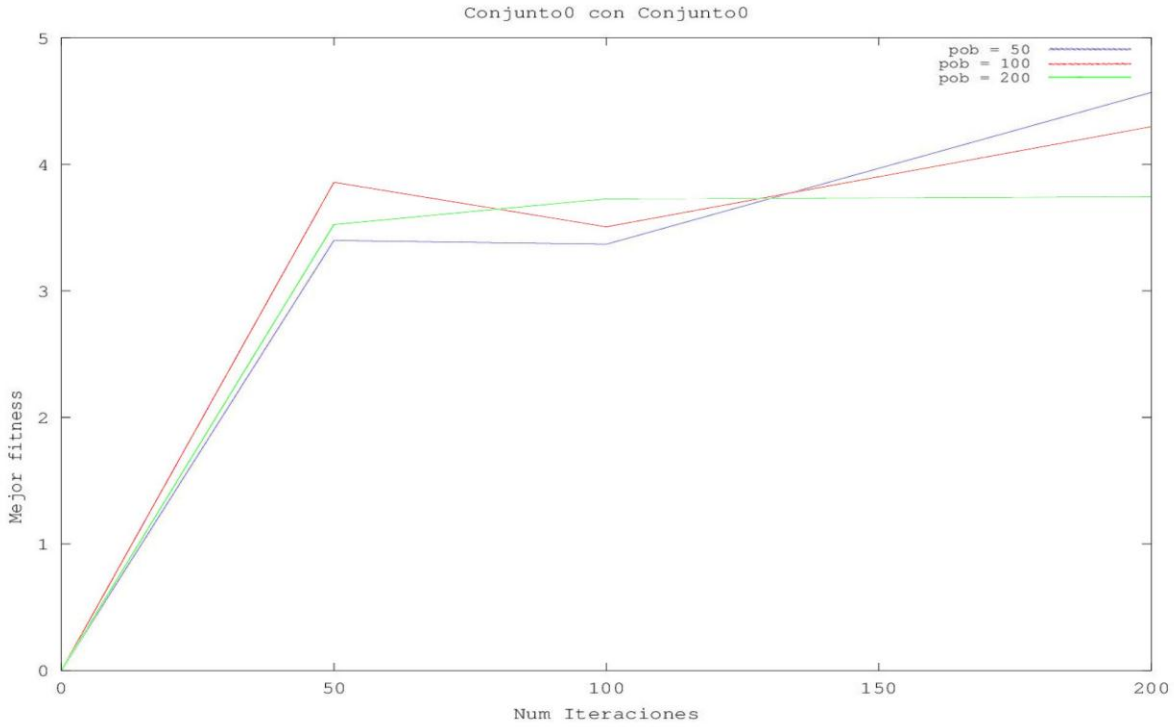
De manera gráfica, se expondrán las ganancias encontradas en función del número de iteraciones y del número de días en el que se decide a invertir todo en una sola empresa (en la gráfica viene representado como número de unos, ya que la cantidad de dinero a invertir cada día viene dada por la expresión $Inversión_i = dinero * cantidad_i$, y si invierte todo $cantidad = 1$). La cabecera del gráfico indica tiene el formato:

" '*conjunto de entrenamiento*' con '*conjunto de prueba*' ".

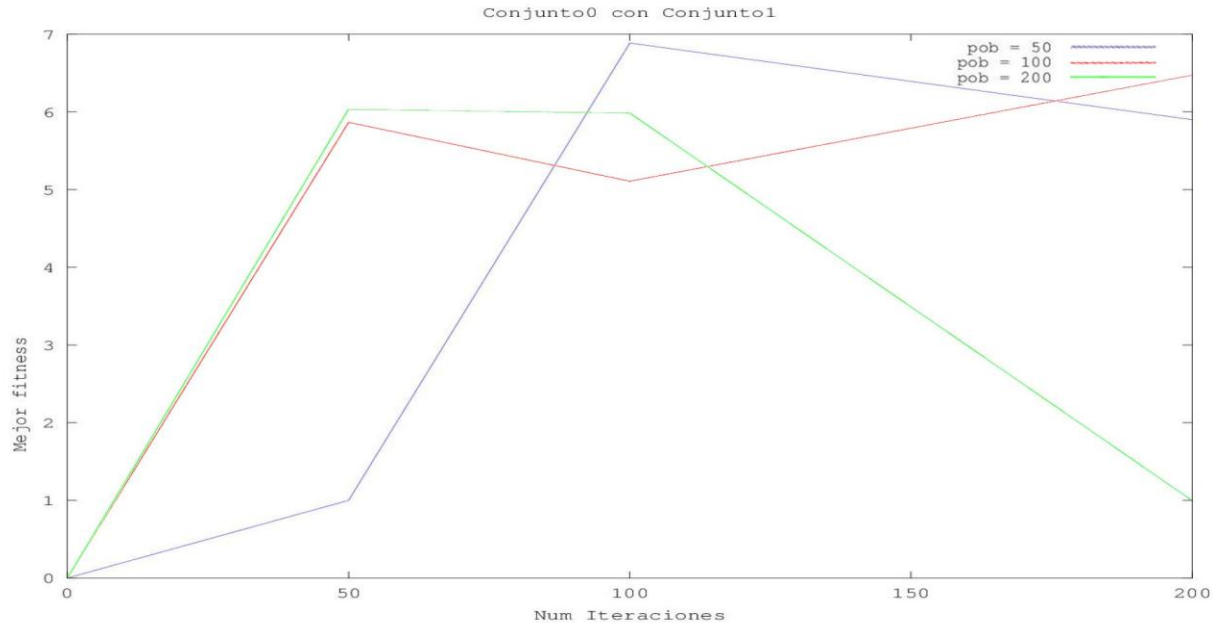
2.1 Con costes de custodia

Veremos qué ganancias se obtienen y cuántos días se decide invertir todo el dinero en una sola empresa.

2.1.1 Conjunto 0

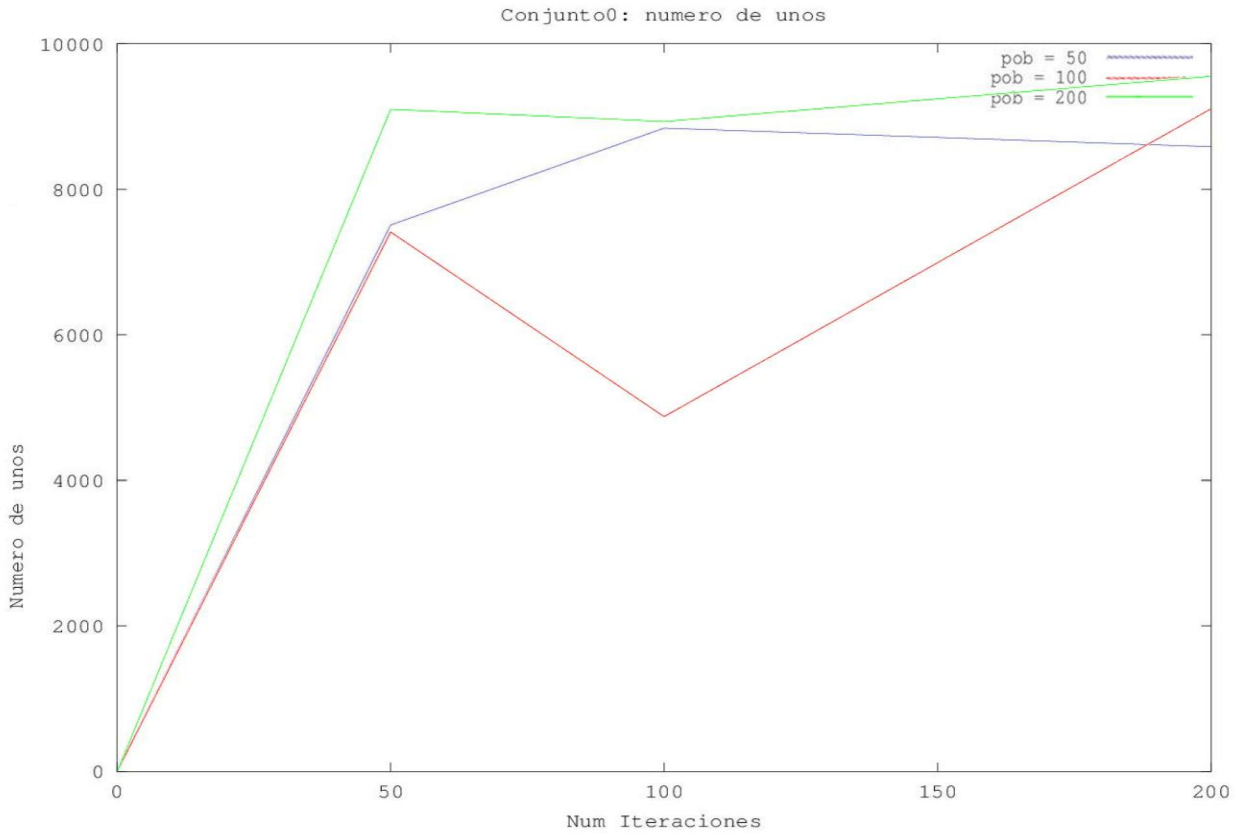


Al entrenar el algoritmo se obtienen buenas ganancias, de hecho mejores que las encontradas al utilizar las cotizaciones del IBEX-35. Al igual que con el algoritmo que maneja un solo índice, también se puede observar cómo a partir de 50 y 100 iteraciones las ganancias crecen más despacio[18].



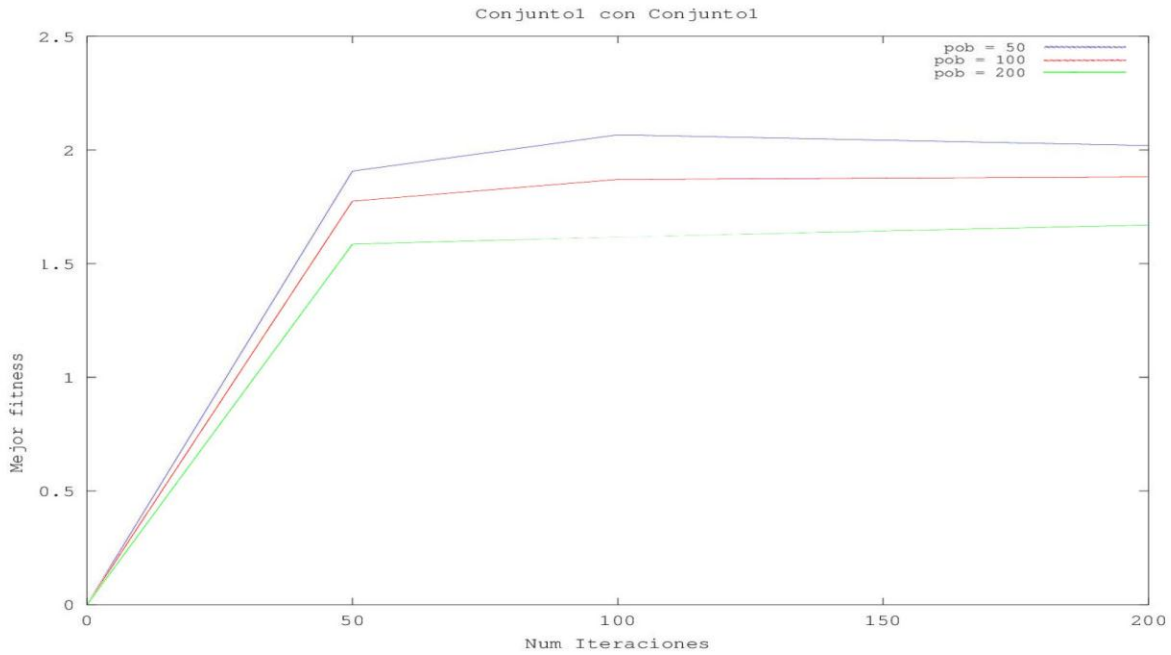
[18] Figuras 3.2.1-3.2.3. Gráficas del conjunto 0 con costes de custodia

Al probar la estrategia encontrada con el segundo conjunto, se observan también buenas ganancias, de hecho incluso mejores que las que se obtienen con el conjunto de entrenamiento. La mejor estrategia encontrada se obtiene al utilizar 50 individuos y 100 iteraciones. Parece que, a mayor número, hay un sobreajuste.

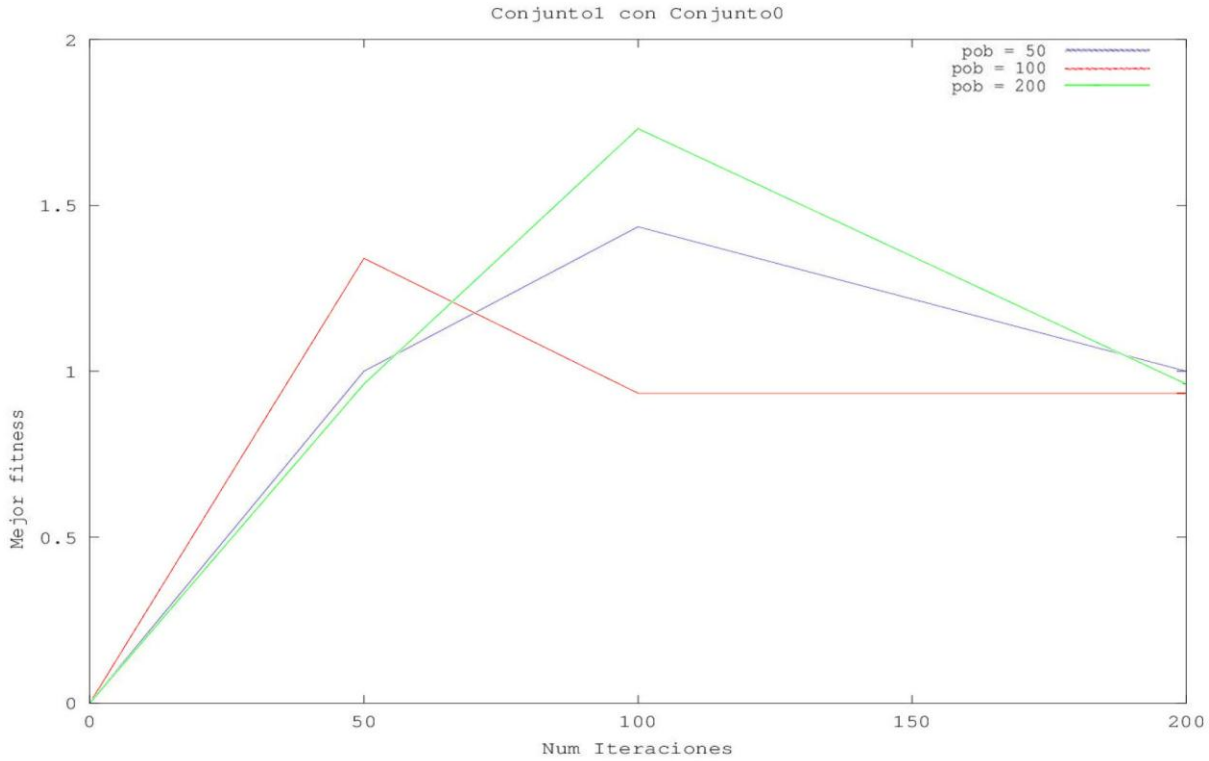


Se puede observar cómo el algoritmo tiende a no dividir el dinero entre varias empresas y, si un determinado día tiene que invertir, prefiere hacerlo en una sola. También se ve cómo el mejor resultado obtenido al probar la estrategia con el segundo conjunto se obtiene al utilizar una estrategia que decide gastar todo el dinero en invertir en una sola empresa durante casi todos los días que decide invertir. Y con 100 iteraciones y 100 individuos, que es la combinación que resulta en una mayor división del dinero, obtiene una ganancia menor.

2.1.2 Conjunto 1

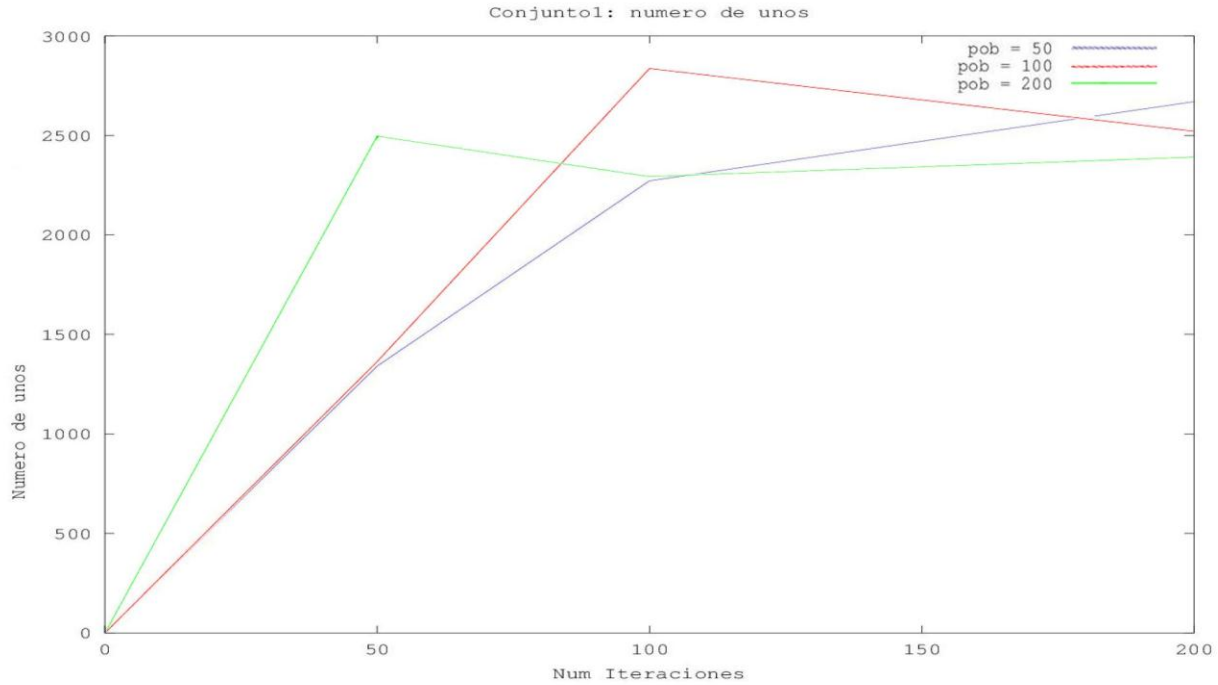


Al entrenar el algoritmo con el segundo conjunto, obtiene varias estrategias que resultan peores que la estrategia que encontró para el primer conjunto, lo cual no es muy buena señal[19].



[19] Figuras 3.2.4-3.2.6. Gráficas del conjunto 1 con costes de custodia

Al usar la estrategia con el primer conjunto resulta en unas ganancias bastantes pobres. Además, la mejor ganancia se obtiene al usar 100 iteraciones y 200 individuos. Mirando cómo evolucionan las ganancias al entrenar, será esperable que se produzca un sobreajuste en ese punto, y que el resultado fuera al revés.



Vemos cómo con esta estrategia decide dividir mucho más la inversión y obtiene peores resultados.

2.1.3 Conclusiones

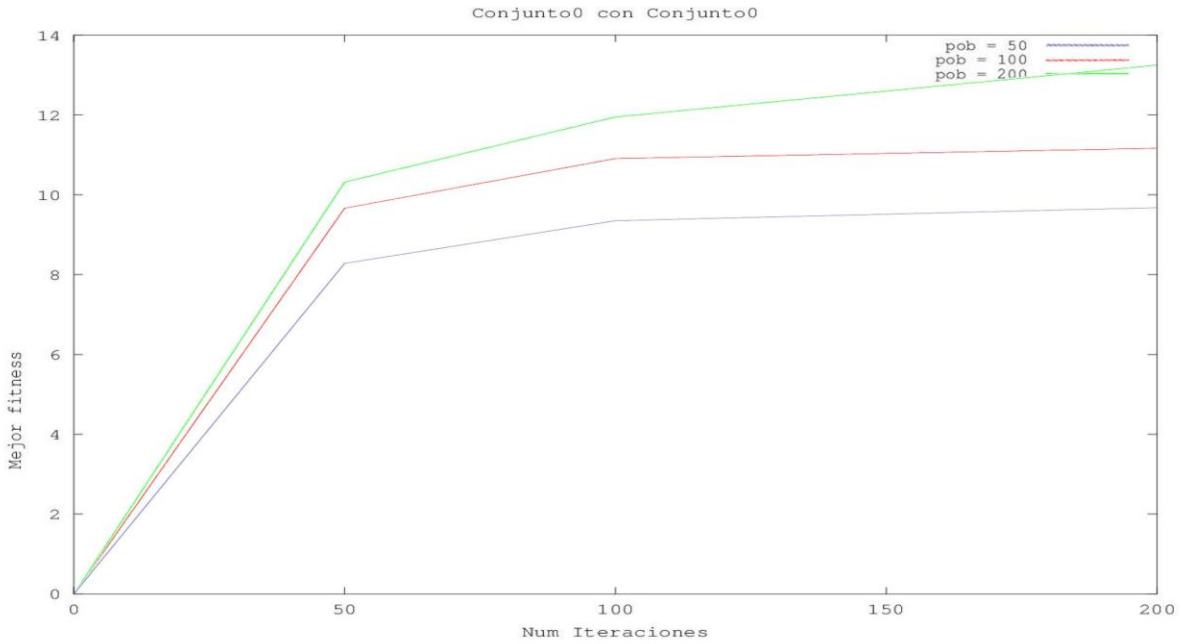
Se obtienen mejores ganancias que al utilizar el algoritmo que solo puede invertir en el índice IBEX-35, pero el comportamiento es similar. Lo que parece ser un punto de sobreajuste resulta no serlo, y se obtienen mejores resultados al usar el conjunto de validación que el de entrenamiento. También se observa cómo al no dividir la inversión se obtienen mejores resultados que al dividirla. No parece que el algoritmo en general pueda resultar muy útil debido a este comportamiento errático.

2.2 Sin costes de custodia

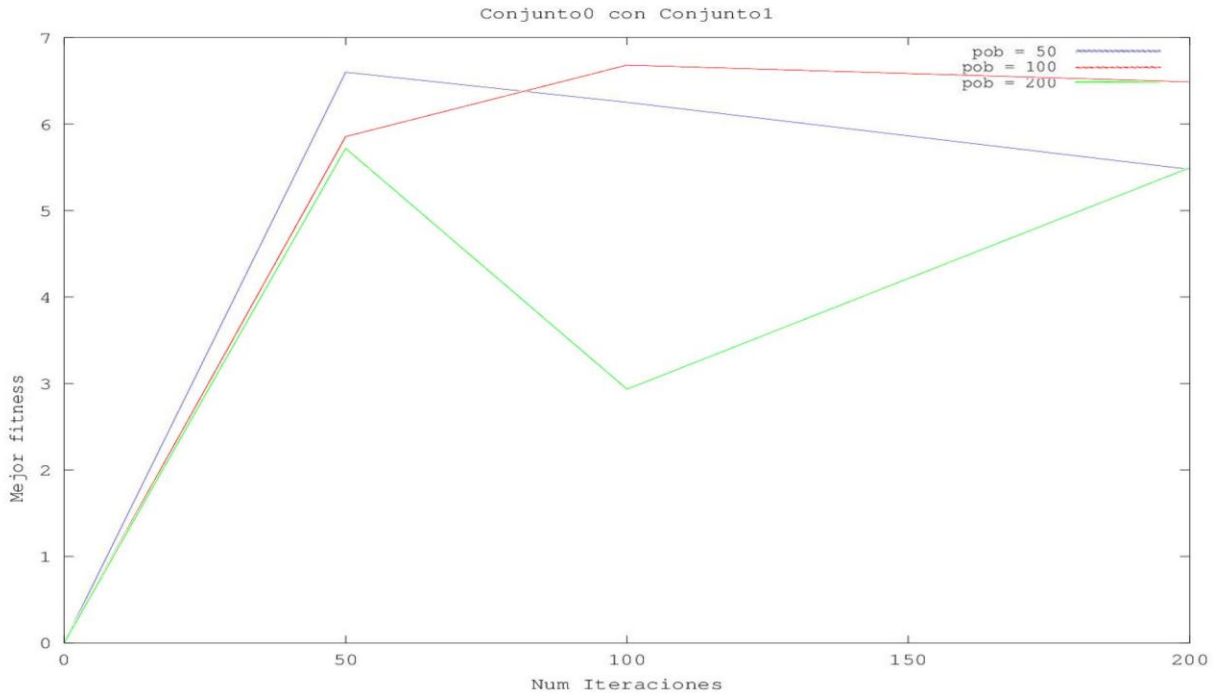
El objetivo de estas pruebas es comprobar si los costes de custodia influyen en cómo divide la inversión en las distintas empresas[20].

[20] Figuras 3.2.7-3.2.9. Gráficas del conjunto 0 sin costes de custodia

2.2.1 Conjunto 0

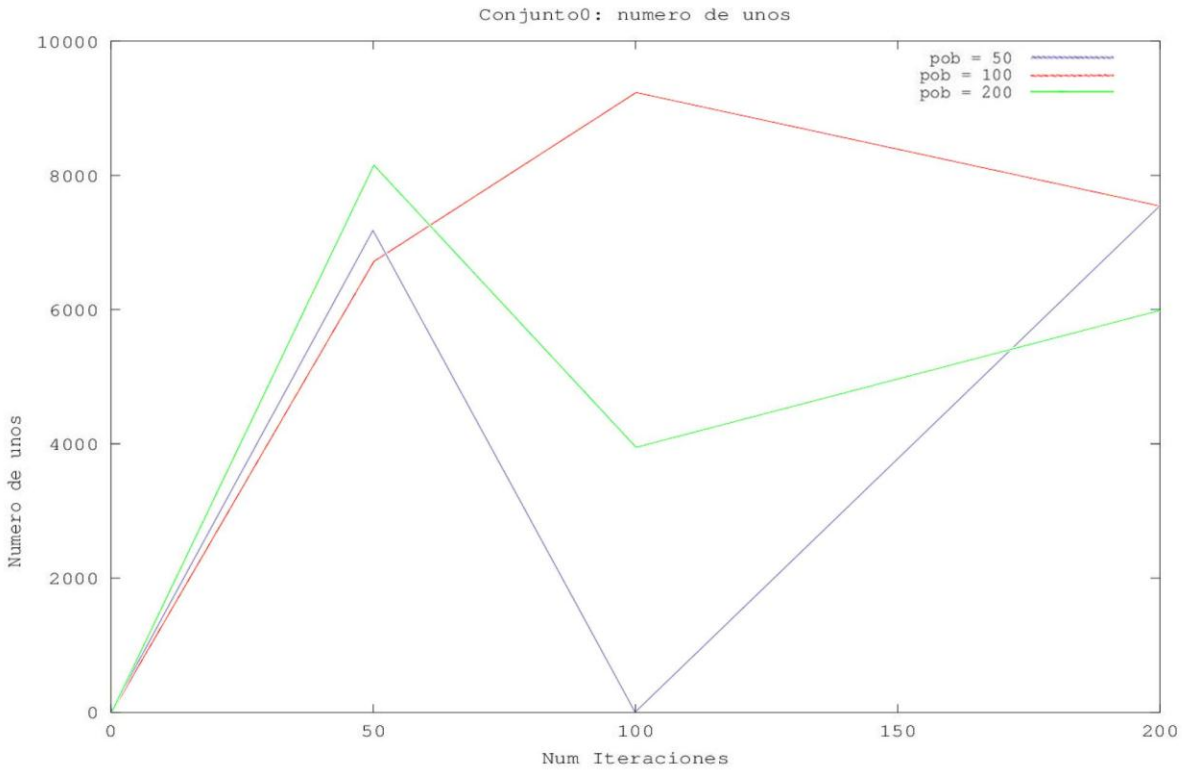


Al quitar los costes de custodia se observa una mejora significativa de las ganancias, y hasta el momento también un comportamiento más parecido al que suelen tener los algoritmos genéticos, obteniendo mejores ganancias cuanto más se aumenten las iteraciones y el número de individuos.



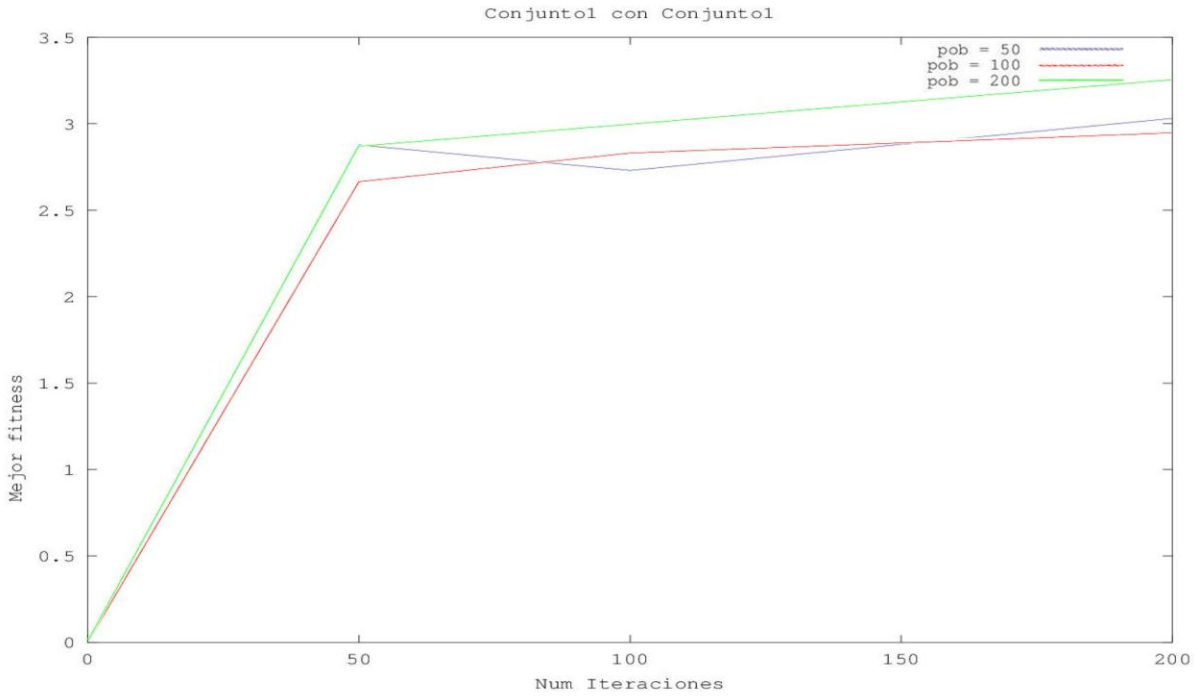
También se encuentran buenas ganancias con el conjunto de validación. A diferencia de cuando se tenía en cuenta el coste de custodia, en este caso son peores

que las encontradas usando el conjunto de entrenamiento (como es lógico), y la combinación que da mejores resultados es usar 100 individuos y 100 iteraciones.

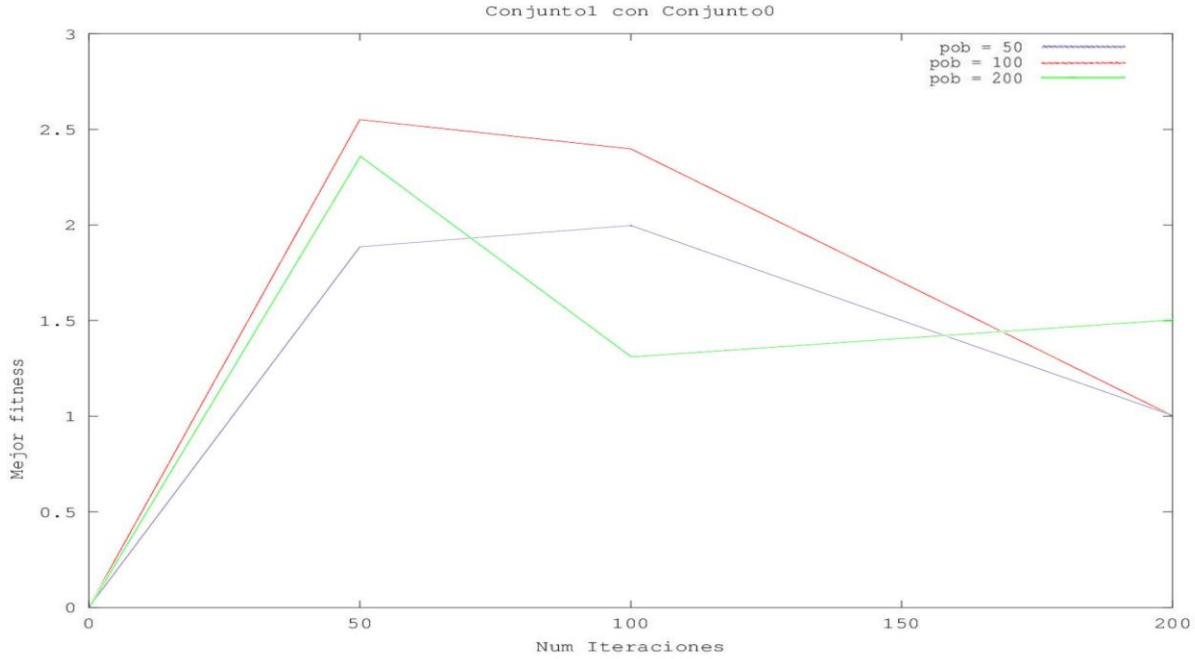


La mejor solución también se obtiene casi no dividiendo la inversión en ningún día. Sin embargo, con una población de 50 individuos y 100 iteraciones nunca decide invertir todo en una sola empresa y también obtiene buenas ganancias. Posiblemente se deba a que, cuantos más tipos de acciones distintas se tienen a la vez, mayor es el coste de custodia.

2.2.2 Conjunto 1

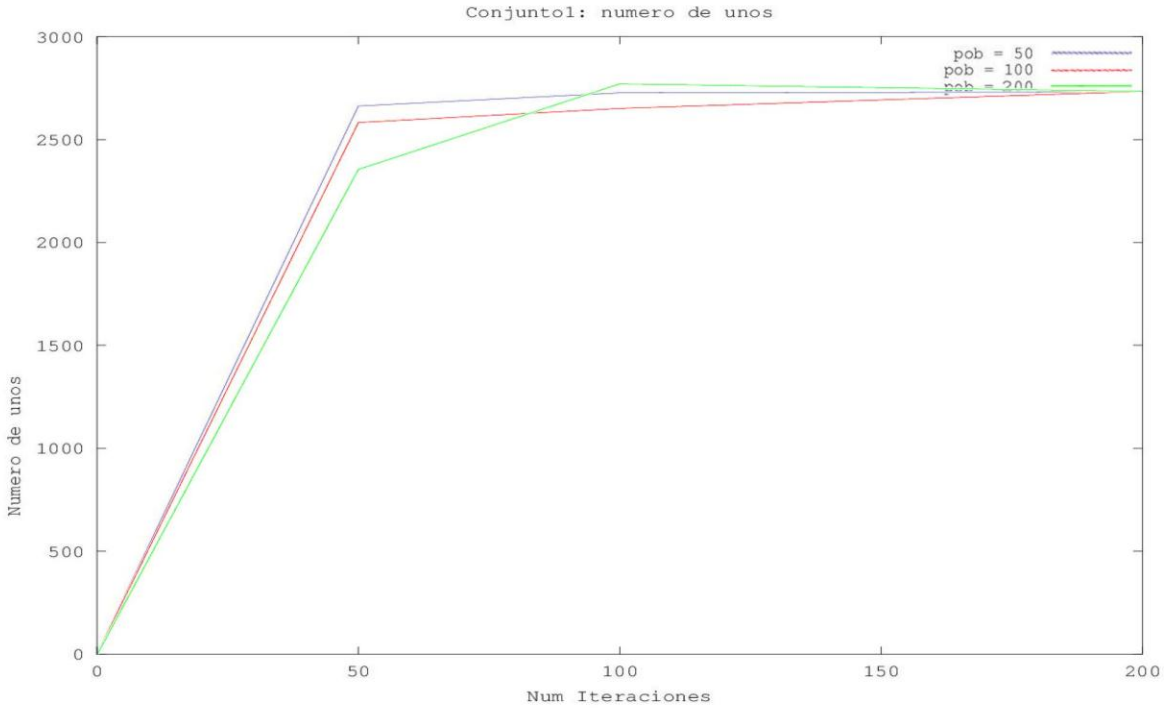


Vuelve a conseguir menos ganancias que con el primer conjunto, lo cual no es buena señal[21].



Las ganancias con el conjunto de validación vuelven a ser bajas.

[21] Figuras 3.2.10-3.2.12. Gráficas del conjunto 1 sin costes de custodia



Al igual que antes decide invertir pocos días en un solo índice, pero esta vez no le da buenos resultados.

2.2.3 Conclusiones

El método de inversión vuelve a tener un comportamiento un tanto errático. No es un comportamiento normal de los algoritmos genéticos que se encuentren mejores resultados para un determinado conjunto al usarlo como conjunto de validación que al usarlo como conjunto de entrenamiento. Parece que, cuando no se tienen en cuenta los costes de custodia, se obtienen ganancias parecidas al invertir todo el dinero en una sola empresa que al dividirlo en varias.

2.3 Conclusiones generales

El algoritmo no se comporta de manera correcta y parece que las ganancias que se obtienen son un tanto aleatorias como ya se ha comentado antes. Tampoco parece que el número de días en los que se decide dividir la inversión en varias empresas dependa del número de individuos o del número de iteraciones, y en caso de que no haya costes de custodia, tampoco parece que intervenga en las ganancias.

3. Algoritmo PSO

Sobre el PSO se han realizado pruebas basadas en las diseñadas para el algoritmo genético de tal manera que los resultados pudieran ser comparables. No se han realizado pruebas tan extensas debido a que los resultados obtenidos reflejan que el comportamiento del algoritmo es, al igual que el del algoritmo genético, errático.

De nuevo, el sobreajuste al contexto de entrenamiento, el comportamiento extraño del algoritmo trabajando con impuestos y la aleatoriedad de los beneficios obtenidos por los distintos individuos juegan un papel fundamental.

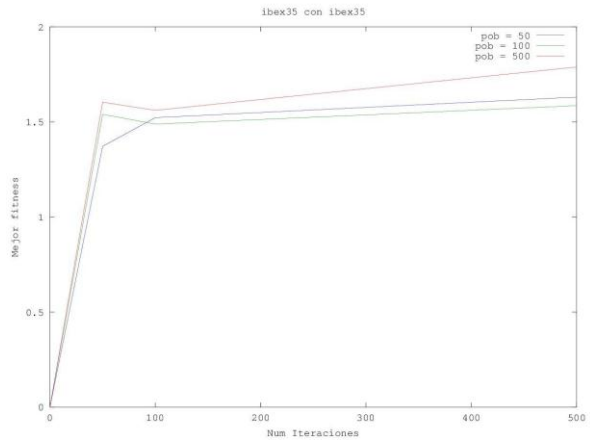
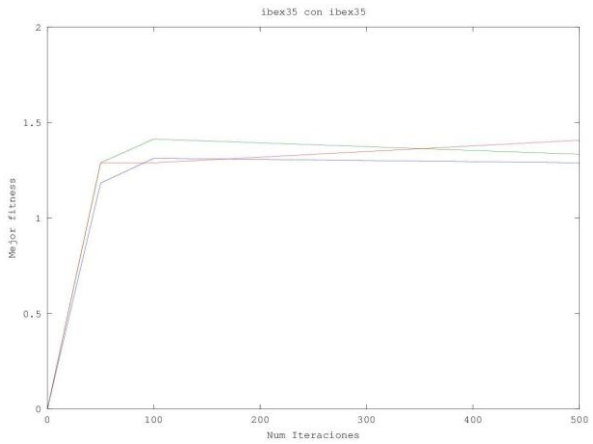
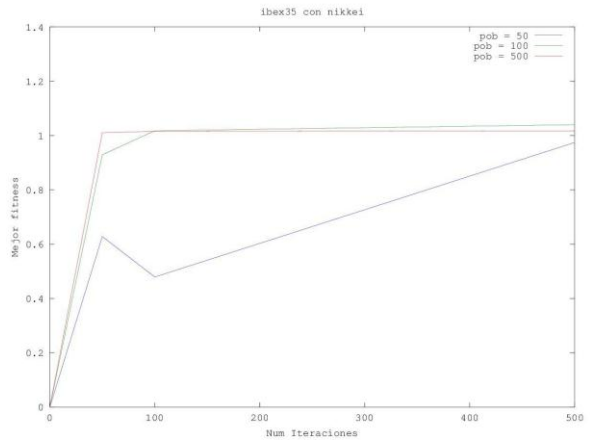
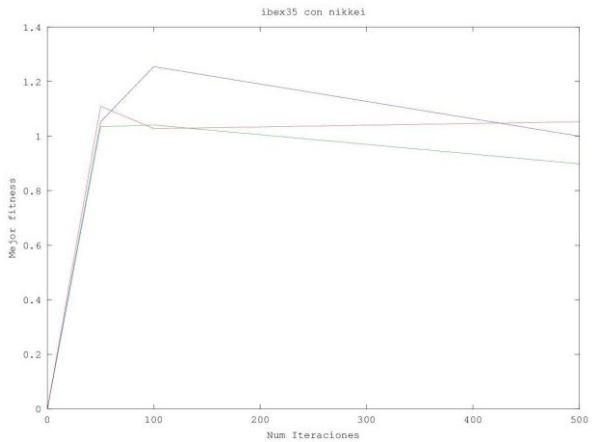
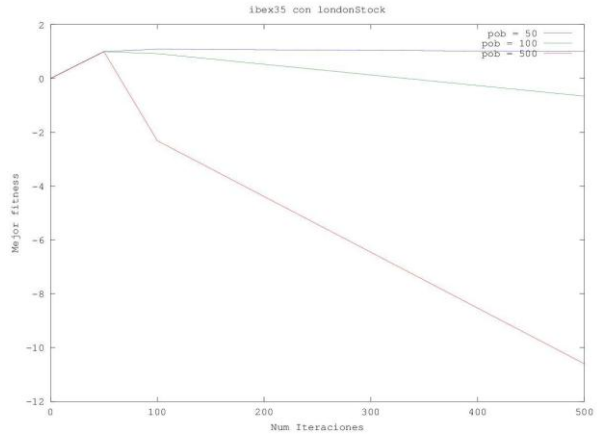
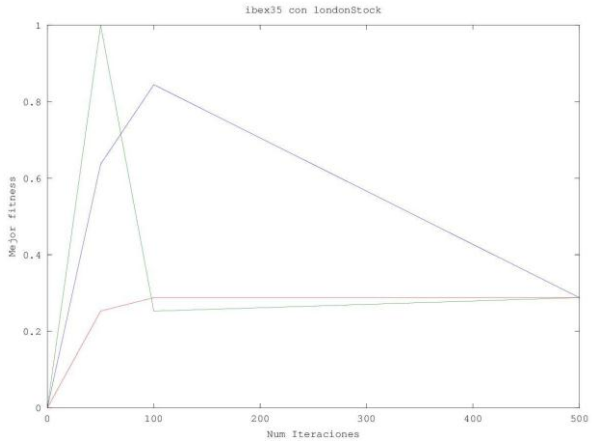
A continuación se muestran algunos de los resultados obtenidos en las pruebas realizada con 5 métodos analíticos (MACD,RSI,TSI,EMA,ADX) y combinando poblaciones de 50, 100 y 500 partículas con ejecuciones durante 50, 100 y 500 iteraciones. Cada combinación se repite 5 veces y se selecciona el mejor individuo encontrado, buscando así reducir la variabilidad de una ejecución a otra de las pruebas. Se presentan sólo casos curiosos y que ayudan a explicar algunas de las conclusiones explicadas en los análisis anteriores realizados sobre el algoritmo genético.

3.1 Con Impuestos

3.1.1 IBEX

En las siguientes gráficas se puede observar que el PSO (derecha) encuentra menores ganancias al ser entrenado sobre el ibex35, sin embargo, al utilizar otros índices como validación las ganancias son mejores. Ésto se explica por el efecto del sobreajuste antes mencionado[22].

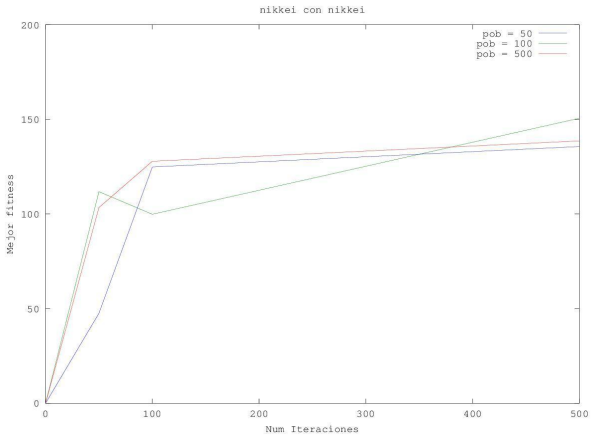
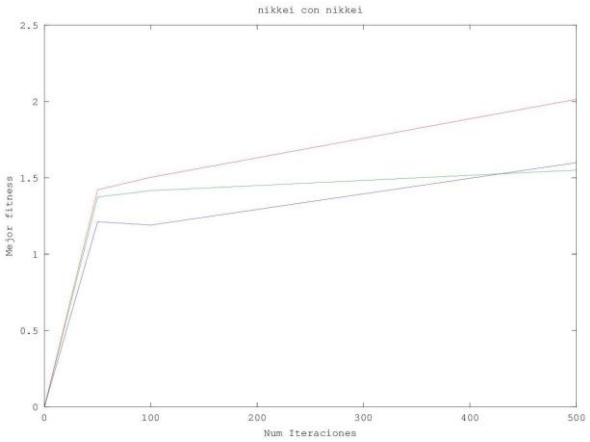
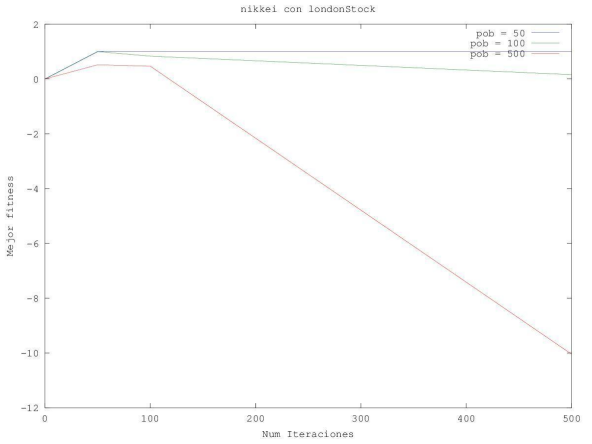
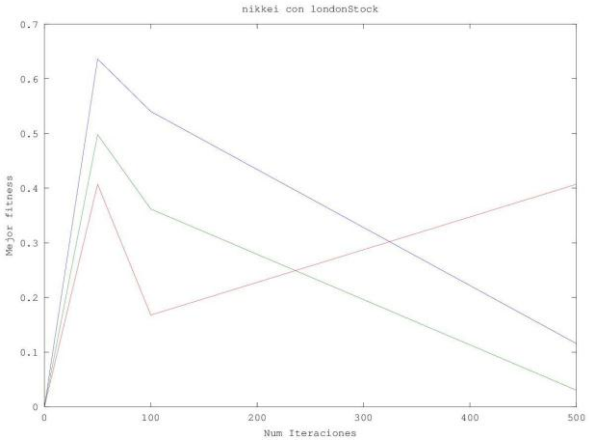
[22] Figura 3.3.1. Comparativa PSO y Algoritmo Genético sobre Ibex35

Algoritmo Genético	PSO
	
	
	

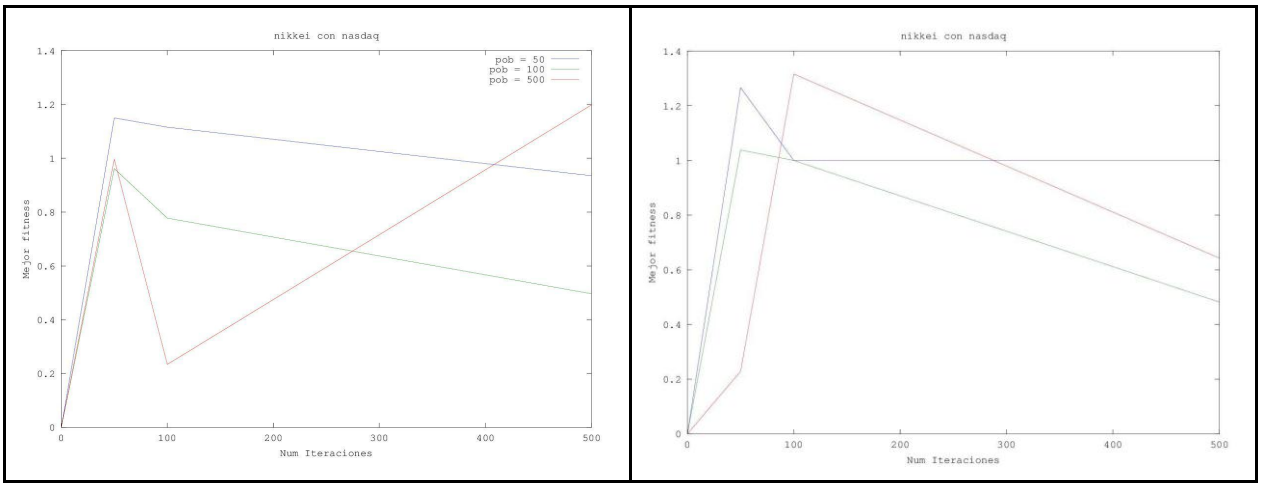
3.1.2 Nikkei

Sobre el Nikkei, el PSO encuentra unas ganancias más parecidas a las obtenidas sobre el entrenamiento en otros índices en comparación con el algoritmo

genéticos. Sin embargo, las simulaciones de validación sobre otros índices con el individuo obtenido son parecidas e incluso evitan el endeudamiento producido por el excesivo sobreajuste que se da en el entrenamiento mediante el algoritmo genético[23].

Algoritmo Genético	PSO
	
	

[23] Figura 3.3.2. Comparativa PSO y Algoritmo Genético sobre Nikkei



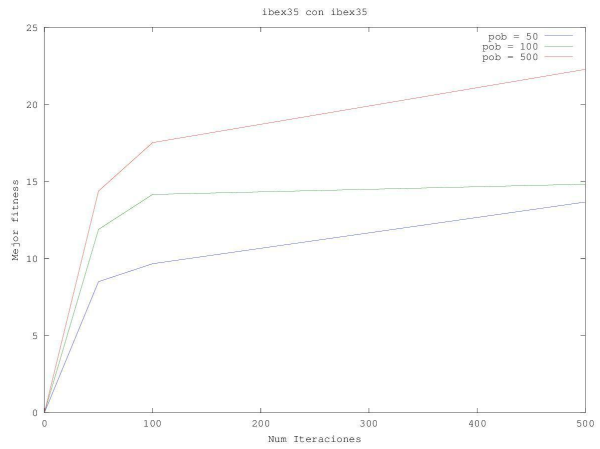
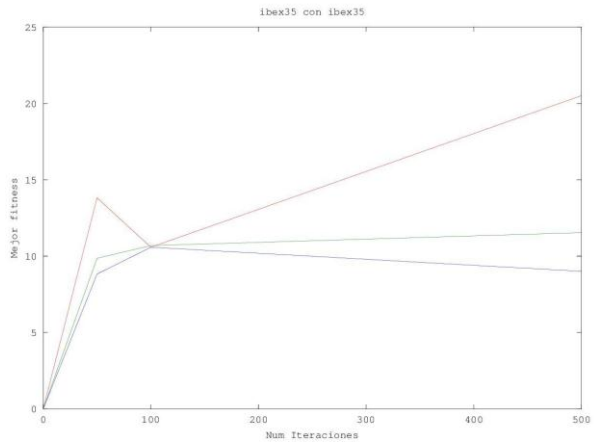
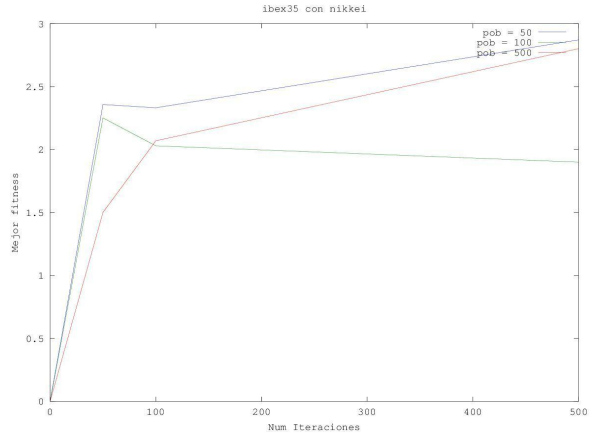
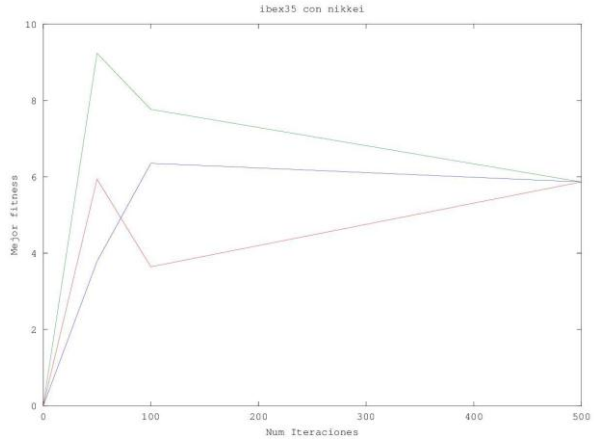
3.2 Sin Impuestos ni comisiones

En las pruebas sin impuestos se observan otros fenómenos que confirman conclusiones anteriormente expuestas.

3.2.1 IBEX

En el entrenamiento sobre el Ibex, se pueden observar que se obtienen ganancias muy similares sin embargo, los individuos que provocan estas ganancias son diferentes, ya que al ser validados sobre el nikkei o el nasdaq, obtienen mejores resultados. Ésto confirma que la diversidad de individuos es muy grande, y que individuos que se desenvuelven igual en un índice, pueden desenvolverse de maneras muy dispares en otros[24].

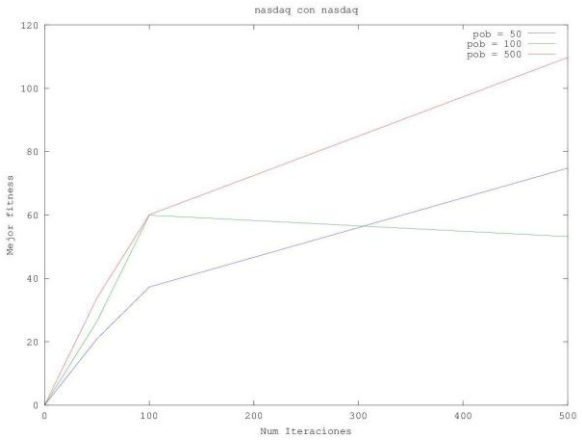
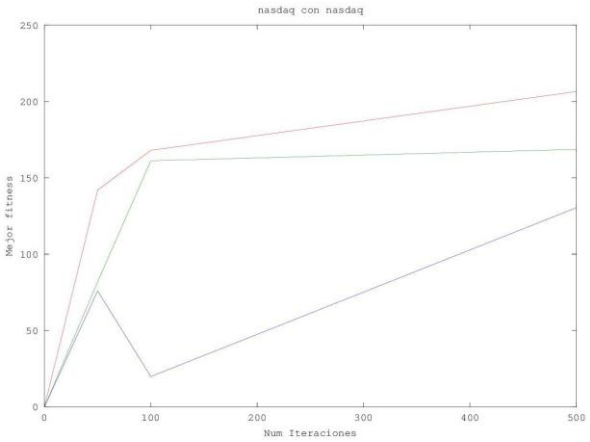
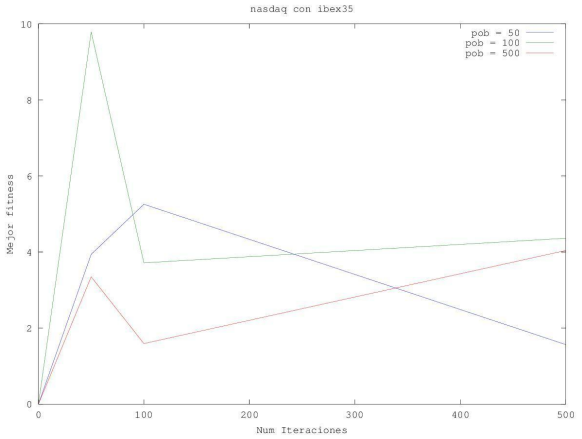
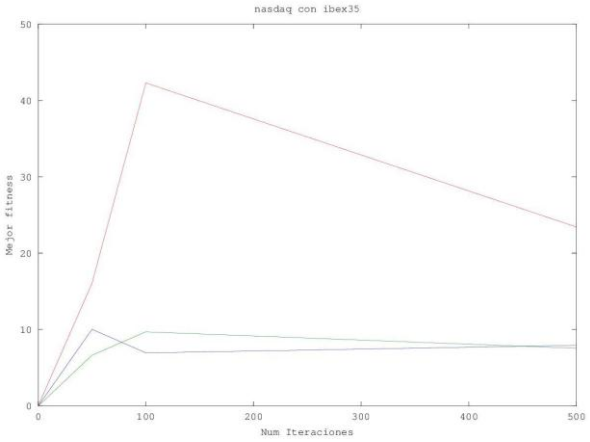
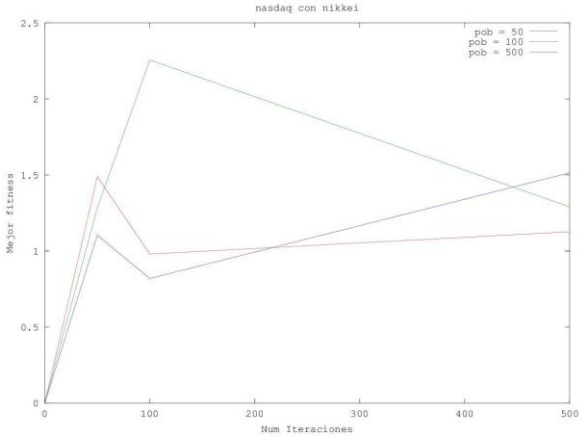
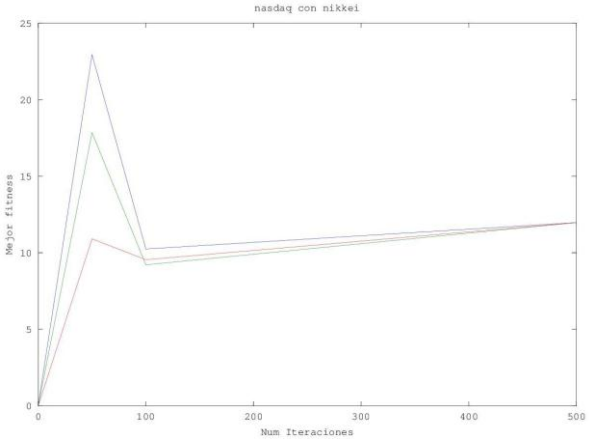
[24] Figura 3.3.3. Comparativa PSo y Algoritmo Genético sobre Ibex sin impuestos.

Algoritmo Genético	PSO
	
	

3.2.2 Nasdaq

Entrenando sobre el índice Nasdaq el PSO encuentra resultados bastante mejores que el algoritmo genético, y en sus correspondientes validaciones, se observa que ésta mejora no parece producir sobreajuste alguno[25].

[25] Figura 3.3.4. Comparativa PSO y Algoritmo Genético sobre Nasdaq sin impuestos.

Algoritmo Genético	PSO
	
	
	

3.3 Análisis de aleatoriedad de resultados en el PSO

A la vista de las pruebas realizadas con el PSO anteriormente presentadas se decidió realizar un estudio sobre la necesidad de realizar varias iteraciones sobre un mismo par de prueba {iteraciones,población} para así disminuir la probabilidad de que un conjunto que supuestamente debiera obtener mejores resultados no lo hiciera.

A continuación se presenta una ejecución del PSO sobre el índice Nikkei, repitiendo tan solo una vez cada par de valores {iteraciones,población}.

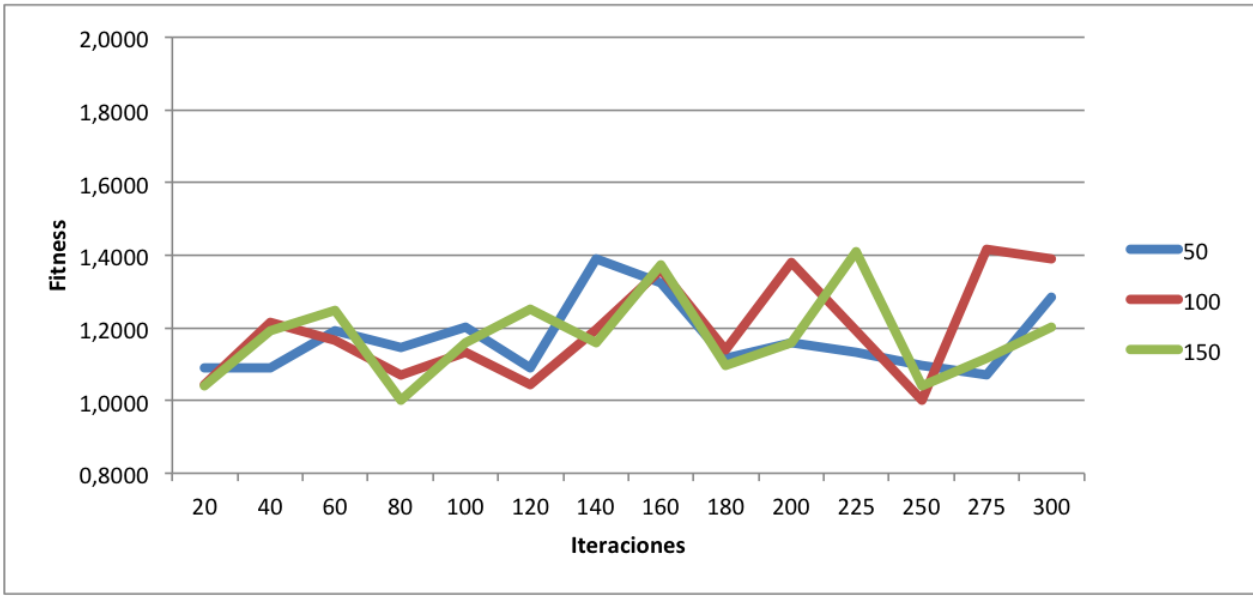


Figura 3.3.5. Gráfico de resultados entrenamiento PSO sobre Nikkei.

Se puede observar que los resultados obtenidos no siguen la lógica de lo que debería pasar, a mayor número de iteraciones y población, mejores resultados.

En la siguiente gráfica se muestran los resultados tras ejecutar la misma prueba, pero realizando 5 veces el entrenamiento para cada par de valores.

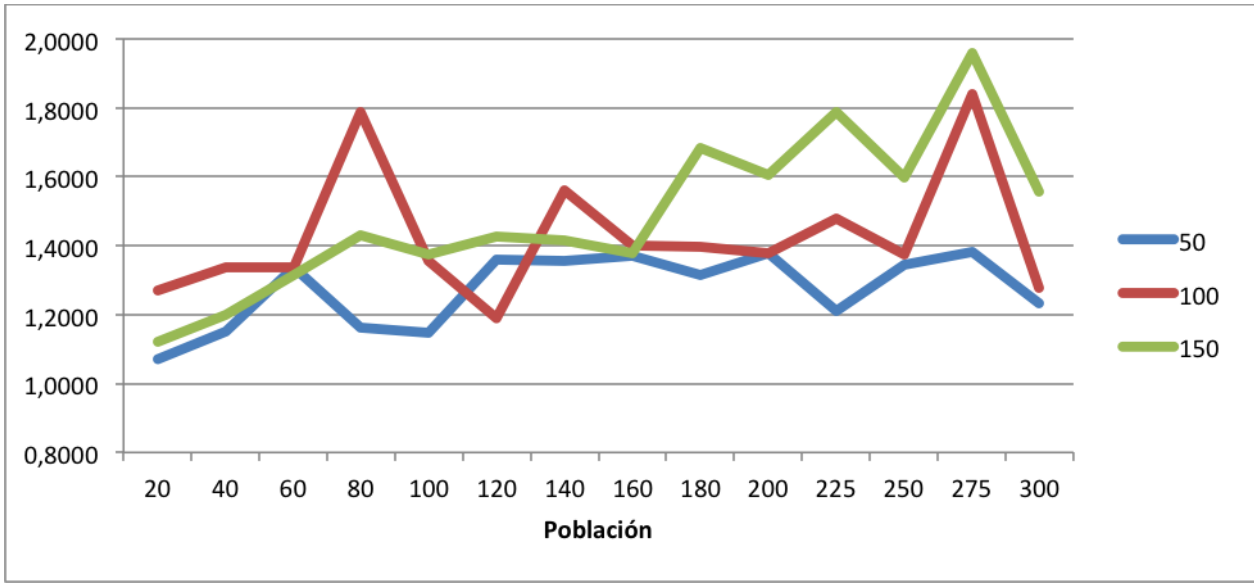


Figura 3.3.6. Gráfica de entrenamiento PSO sobre Nikkei con 5 repeticiones para cada par {iteraciones,población}

Se observa que esto mejora los resultados globales obtenidos y hace que la gráfica sea más aproximada al comportamiento esperado de éste tipo de algoritmos, con obteniendo una función con menor ruido.

Éste fenómeno es mencionado en un estudio realizado por Tarek Aboueldahab and Mahumod Fakhreldin [11]. El estudio referencia casos en los que al aplicar PSO sobre métodos de predicción el bolsa, el algoritmo converge de manera prematura a un máximo local, imitando la búsqueda a una parte del espacio de soluciones. Como solución proponen un algoritmo híbrido GA/PSO (Algoritmo Genético y Particle Swarm Optimization) que evita en parte el problema aumentando la ventana de búsqueda en el espacio de soluciones.

3.4 Conclusiones

El análisis comparativo de los resultados obtenidos no permite discernir si uno de los dos algoritmos funciona mejor que el otro. Se detectan casos en los que su funcionamiento es diferente, sin embargo, la mayoría de las veces es similar y es probable que estos casos sean fruto de la aleatoriedad propia de la búsqueda de soluciones óptimas mediante algoritmos metaheurísticos.

Los resultados sí ayudan a comprender que individuos o partículas que obtienen un beneficio similar sobre un índice pueden no ser igual de útiles cuando son aplicados sobre otros contextos.

Capítulo 4 - Trabajo futuro

En esta sección se detalla el trabajo que se podría realizar en el futuro.

Desde el punto de vista de la propuesta software, se puede mejorar la arquitectura del modelo-vista-controlador para controlar mejor los eventos que le llegan al controlador, teniendo varios controladores específicos, uno para cada menú, y un controlador principal que delegue el evento al controlador específico que necesite.

Desde el punto de vista del estudio de los resultados se puede tratar de ampliar los conjuntos de iteraciones e individuos usados para realizar las pruebas, aumentar el número de datos de las cotizaciones de los índices y de las empresas y, en general, tratar de hacer pruebas más extensas con los algoritmos de los que disponemos. Se puede también probar con más algoritmos evolutivos, y si se consiguiera encontrar un número suficiente de ejemplos de días etiquetados se podría probar con algoritmos de aprendizaje automático como redes neuronales o *support vector machines*.

Se podría también tratar de modificar los algoritmos de tal manera que la función de fitness diera diera más peso al dinero en los últimos días que a los primeros. Hipotéticamente así se conseguiría que el algoritmo se adaptara más rápidamente a las nuevas tendencias surgidas. Sería necesario la realización de nuevos experimentos y un estudio de éstos. Ayudaría que este programa fuera capaz de mantener los datos de la bolsa actualizados para que fuera lo más realista posible.

Capítulo 5 - Conclusiones

Los experimentos realizados muestran que los algoritmos presentan un comportamiento anómalo cuando se tienen en cuenta impuestos, cánones y comisiones, alcanzando pocas ganancias menos con el índice Nikkei. Al despreciar el peso de los gastos extra, los algoritmos encuentran estrategias válidas que consiguen ganancias significativas, pero aun así las soluciones se encuentran generalmente sobreajustadas y funcionan por el relativo parecido de las cotizaciones. Si se utiliza para entrenar a los algoritmos un período de tiempo anterior a la crisis y se prueba la estrategia en uno posterior, las ganancias se reducen significativamente, demostrando que existe un claro problema de sobreajuste.

No parece que los algoritmos de computación evolutiva sirvan para poder encontrar una estrategia genérica que sirva para ganar dinero con la bolsa en cualquier período de tiempo; aunque, tal y como se ha propuesto en el capítulo 4, es posible que se puedan modificar los algoritmos para que se vayan adaptando a las nuevas tendencias de la bolsa de manera incremental.

No es posible determinar si el uso exclusivo del análisis técnico resulta suficiente para construir una estrategia genérica que permita ganar dinero en cualquier período de tiempo, ya que el hecho de que los algoritmos no la encuentren no significa que no exista. Aunque definitivamente se puede concluir que el uso de indicadores técnicos es útil para ganar dinero en un período de tiempo específico siempre y cuando los impuestos, las comisiones y los cánones no tengan peso.

Chapter 5 - Conclusions

The experiments show that the algorithms have abnormal behavior when taxes, fees and commissions are taken into account, obtaining little profit except when using the Nikkei index. When neglecting the weight of the extra costs, the algorithms find valid strategies that obtain significant gains, yet the solutions are usually overfitted and their success depend on the similarities between the prices of the different stock markets. If a pre-crisis time period is used to train the algorithms, and a post-crisis time period is used to test the said strategy, profits are significantly reduced, showing that there is a clear problem of overfitting.

It does not seem that evolutionary computation algorithms can be used to find a generic strategy that manages to make money by investing in the stock market at any time; although, as proposed in Chapter 4, it seems possible to modify the algorithms so that they are able to adapt incrementally to the new stock market trends.

It is not possible to determine whether the exclusive use of technical analysis is sufficient to build a generic strategy to make money at any time, since the fact that the algorithms do not find it does not mean said strategy does not exist. It can be definitely concluded that the use of technical indicators is useful for making money on a specified period of time as long as taxes and fees are not considered.

BIBLIOGRAFÍA

1. M. Scott Brown, M.J. Pelosi, H. Dirska. *Dynamic-Radius Species-Conserving Genetic Algorithm for the Financial Forecasting of Dow Jones Index Stocks.*Machine Learning and Data Mining in Pattern Recognition, Lecture Notes in Computer Science, Springer (2013)
2. R.P. Kanungo. *Genetic Algorithms: Genesis of Stock Evaluation*. Economics WPA Working Paper (2004)
3. D. de la Fuente, A. Garrido, J. Laviada, A. Gómez. *Genetic Algorithms to Optimize the Time to Make Stock Market Investmen*. In Proc. GECCO 2006, USA (2006)
4. C. Schoreels, B. Logan, J.M. Garibaldi. *Agent based genetic algorithm employing financial technical analysis for making trading decisions using historical equity market data*. IAT '04: Proceedings of the Intelligent Agent Technology, IEEE/WIC/ACM International Conference on (IAT '04), IEEE Computer Society, Washington, DC, USA (2004)
5. F.A. Badawy, H.Y. Abdelazim, M.G. Darwish. *Genetic Algorithms for Predicting the Egyptian Stock Marke*t, Information and Communications Technology, 2005. Enabling Technologies for the New Knowledge Society: ITI 3rd International Conference (2005)
6. C. Zhou, L. Yu, T. Huang, S. Wang, K. K. Lai. *Selecting valuable stock using genetic algorithm*. In Proc. SEAL'06. Springer (2006).
7. L. Lin, L. Cao, J. Wang, C. Zhang. *The applications of genetic algorithms in stock market data mining optimisation*. A. Zanasi, N. Ebecken, C. Brebbia (Eds.), Data Mining V: Data Mining, Text Mining and their Business Applications (DATA MINING 2004), WIT Press, Southampton, Boston (2004)
8. F. Fernández-Rodríguez, C. González-Martel, S. Sosvilla-Rivero. *Optimisation of Technical Rules by Genetic Algorithms: Evidence from the Madrid Stock Market*. Fundación de Estudios de Economía Aplicada (2001)
9. Antonio C. Briza, Prospero C. Naval Jr. *Stock trading system based on the multi-objective particle swarm optimization of technical indicators on end-of-day market data*. Applied Soft Computing 11 (2011).
10. J. Kennedy, R. Eberhart. *Particle swarm optimization, in: Proceedings of the 1995 IEEE International Conference on Neural Networks, Piscataway, New Jersey*. IEEE Service Center (1995).
11. T. Aboueldahab, M. Fakhreldin. *Prediction of Stock Market Indices using Hybrid Genetic Algorithm/ Particle Swarm Optimization with Perturbation Term*. International Conference on swarm intelligence (ICSI) (2011).