Fundamentos de Aprendizaje Automático

Práctica 3

Grupo1461
Mario Valdemaro García Roque
Manuel Reyes Sánchez

Apartado 1 – Detalles de la implementación

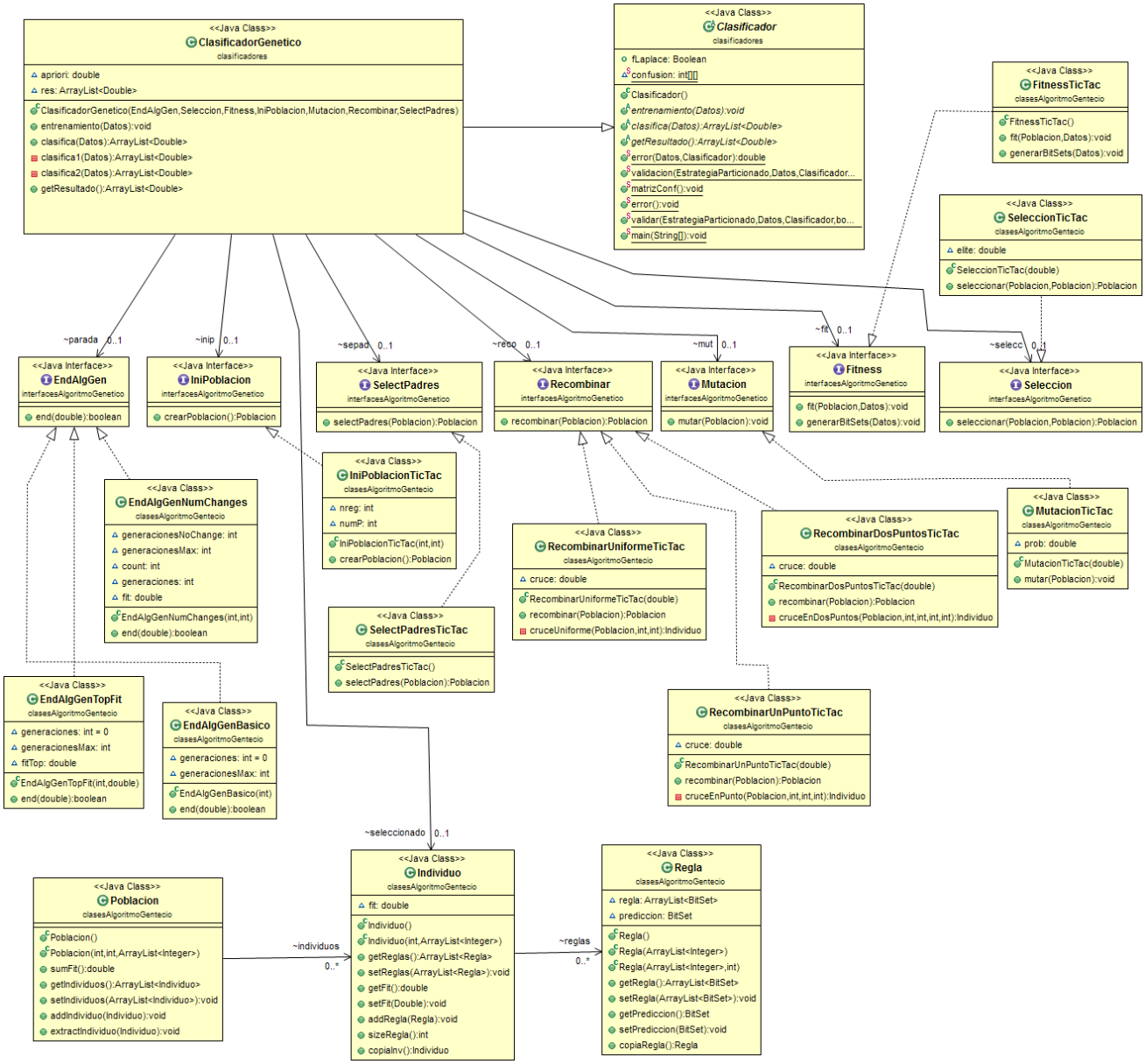
Hemos dotado a cada regla de nueve condiciones (una por cada atributo con los cuales cuenta un dato). Cada condición esta formada por tres bits que indican lo que se encuentra en el tablero (x, b, o), pudiendo una condición aceptar mas de una posibilidad (por ejemplo x, b). Para realizar esto cada condición es un BitSet de tamaño 3 y la regla es un ArrayList de estos. Elegimos esta configuración por aportar rapidez en la ejecución, así como evitar malgastar espacio.

A la hora de crear la población elegimos el numero de reglas y el numero de población. Ademas hay una clase IniPoblaciónTicTac que hereda de IniPoblacion (interfaz que usa el algoritmo genético) que se encarga de iniciar un ArrayList que se encarga de indicar a la clase población (que es genérica) el numero de condiciones así como el tamaño de cada una. Se consigue por lo tanto un diseño modular, pues para resolver un problema distinto deberíamos realizar simplemente su correspondiente IniPoblacion, heredando de la interfaz y indicando a Población como son las reglas, pero sin tener que preocuparnos de nuevo por la organización interna de la población y sus individuos.

Hemos realizado los métodos de cruce en un punto, cruce en dos punto y cruce uniforme. Todos implementan una interfaz común, por lo que la codificación del resto del algoritmo no cambia usemos uno u otro. Los mejores resultados los hemos obtenido con el cruce en un punto, aunque la diferencia no es muy acusada.

Para implementar la mutación realizamos un recorrido por todas las reglas de todos los individuos y para cada regla si se cumple la probabilidad de mutación, seleccionamos una de sus condiciones de la cual invertimos un bit aleatorio.

En la siguiente página puede verse el diagrama de clases correspondiente al algoritmo genético implementado.
Se incluye también en la entrega como fichero aparte.



Puede observarse por una lado que una población esta formada por individuos, que a su vez cuentan con reglas, o como el clasificador tiene un individuo, que es el mejor tras la fase de train, que se almacena para ser usado en test. Estas clases son genéricas del algoritmo y la diferencia entre datos vendrá marcada en como se llame al constructor de población.

Lo mas destacable a observar del diseño es ver como el algoritmo genéticos necesita unas interfaces que representan las diferentes partes que lo componen y como se han realizado clases especificas para el problema TicTac. La clases generadas para la interfaz que determina el fin del algoritmo son genéricas también (pues solo son diferentes maneras de poner fin al algoritmo), por lo cual no incluyen TicTac en su nombre.

Apartado 2 – Resultados

Población	Generaciones	Fit max inicial/final	Fit min inicial/final	Fit medio inicial/final	Test
10	100	0,367 / 0,41438	0,162 / 0,409	0,267 / 0,41437	39,58%
10	1000	0,376 / 0,676	0,201 / 0,646	0,277 / 0,671	38,22%
100	100	0,458 / 0,67	0,144 / 0,67	0,287 / 0,67	36,1%
100	1000	0,388 / 0,674	0,175 / 0,674	0,3 / 0,674	32,56%
500	100	0,482 / 0,799	0,14 / 0,783	0,291 / 0,797	31,42%
500	1000	0,509 / 0,868	0,172 / 0,864	0,291 / 0,867	29,82%

Número de reglas: 100; Parámetros usados: los indicados en el enunciado.

Apartado 3 – Análisis de resultados

Elegimos usar 100 reglas para que actuará el algoritmo genético en la mayor parte de las decisiones. Nuestra implementación cuando no se posee una regla que encaje con el dato a predecir utiliza un clasificador a priori. Con un conjunto de 30 reglas en el individuo se utilizaba a priori en un 25% o mas incluso (50%) de las ocasiones aproximadamente. Para paliar esto se debía aumentar el numero de reglas. Aumentamos este numero hasta obtener un cantidad de reglas que permita tener una predicción en la mayor parte de las ocasiones.

El tamaño de la población ha sido uno de los factores que mas creemos que ha influido en nuestros datos. Como podemos ver el numero inicial de individuos condiciona como nuestro algoritmo empieza. Si tenemos pocos individuos peor probabilidad tenemos de generar uno bueno con el que empezar. Podemos ver en la tabla que como norma general a mayor población el mejor individuo inicial es mejor. Así mismo, mas individuos implican un mayor número de posibilidades, cruces y en definitiva, combinaciones de genes en cada generación.

En cambio el numero de generaciones no ha sido tan determinante. De 100 a 1000 generaciones se obtienen mejoras, pero dado el incremento de tiempo puede no ser rentable. Nuestro programa imprime el mejor fit solo si es mejor que el anterior. Cuando ejecutamos el programa podemos ver como en las primeras generaciones se obtiene mejora prácticamente todo el tiempo. Según este numero aumenta esto empieza a no ocurrir. Por ejemplo según se llega a las 1000 generaciones se observan periodos de casi 20 generaciones sin variación en el mejor fit.

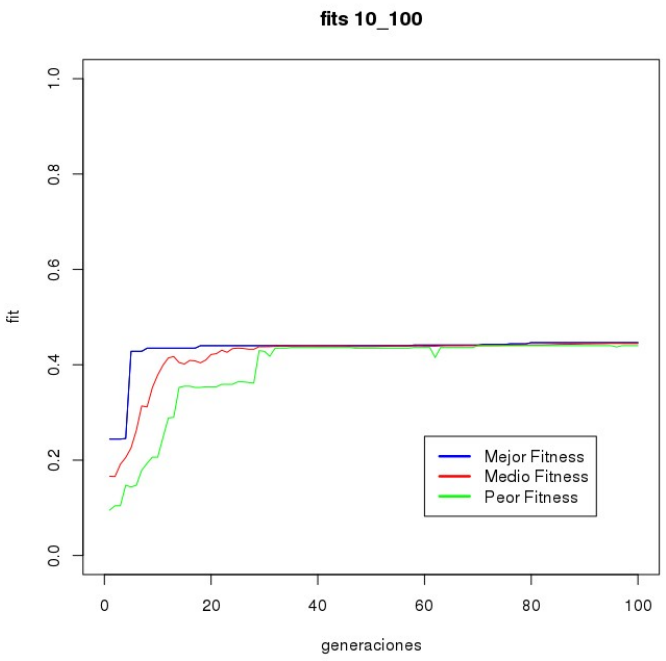
Partiendo de una tasa de cruce estándar de 60%. Aumentar el numero no produce diferencias significativas, ya que el elitismo ayuda a conservar los mejores. Estamos simplemente realizando mas variaciones, lo cual es positivo para encontrar un individuo mejor. Bajar el numero (a por ejemplo 20%) ocasiona que el fit del mejor individuo para de aumentar antes pues este tiene bajas probabilidades de cruzarse con otros individuos y por lo tanto mejorar.

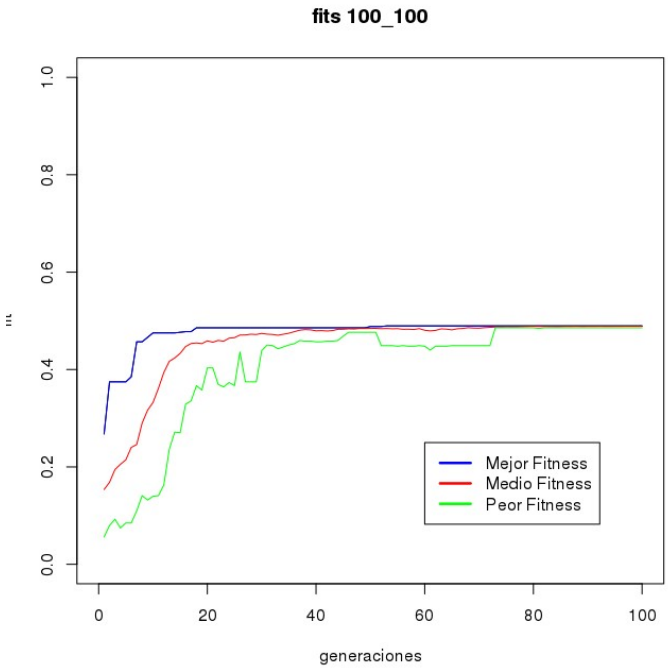
En el caso de la mutación hemos obtenido resultados positivos al aumentar su probabilidad al 1%. Quizá el numero dado en el enunciado era bajo y se producían muy pocas mutaciones, con este valor nos aseguramos mayor numero de mutaciones, incluyendo mas variación genética en los individuos, ayudando a que no se estanquen en un conjunto de reglas muy repetidas entre todos.

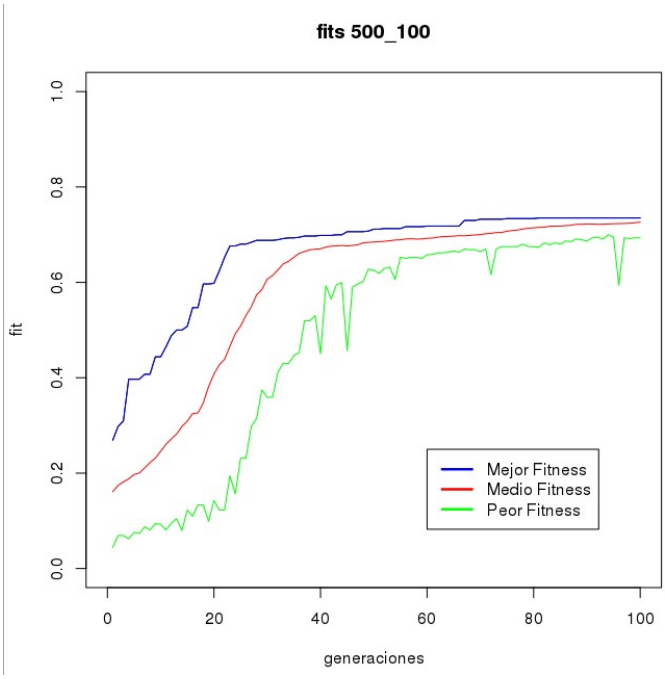
Apartado 4 – Evolución de la fase de entrenamiento de forma gráfica

Para mostrar este tipo de datos hemos considerado interesante mostrar tanto las gráficas de como va evolucionado un test en particular, es decir como mejora su finnes medio, mejor y peor. Y también ver en función de la población como va mejorando este fit.

A continuación mostramos todas las gráficas con **100 generaciones** y distintas cantidades de población.

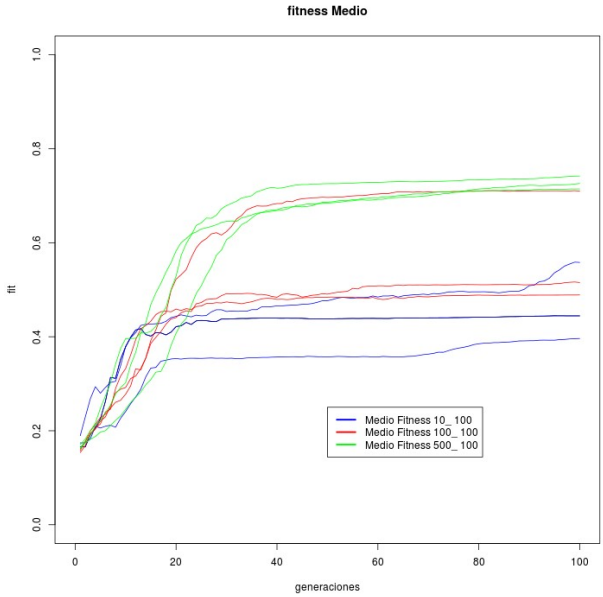


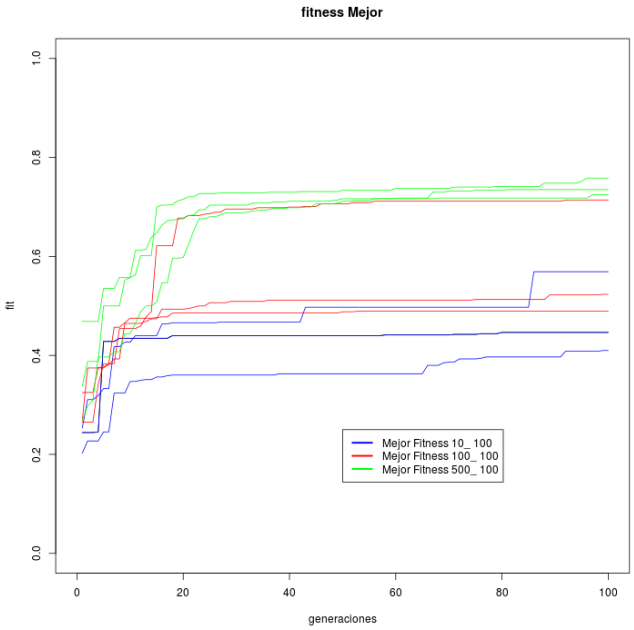




Como podemos observar el para esta cantidad de generaciones obtenemos que el mejor fitness se encuentra cuando usamos tamaño de población grandes, véase que la población de 500 individuos es la que obtiene el mejor fit al final, esto es lógico ya que cuanto mas individuos haya mas probable es obtener un muy bueno y mas mezclas y combinaciones genéticas se producen.

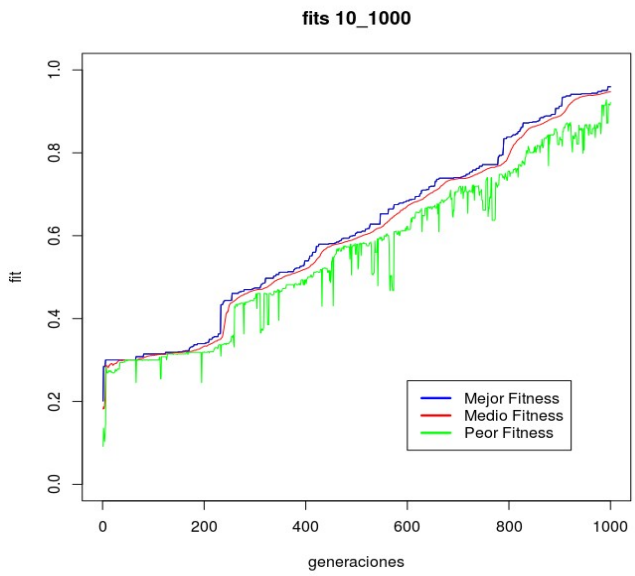
En las siguientes gráficas esto se pude observar mas claramente , estas consisten en repetir 3 veces una prueba, para obtener una media o tendencia con cada combinación, con la misma población y las mismas generaciones y comparar los fitnnes obtenido con otras pruebas con las mismas generación pero distintos individuos.

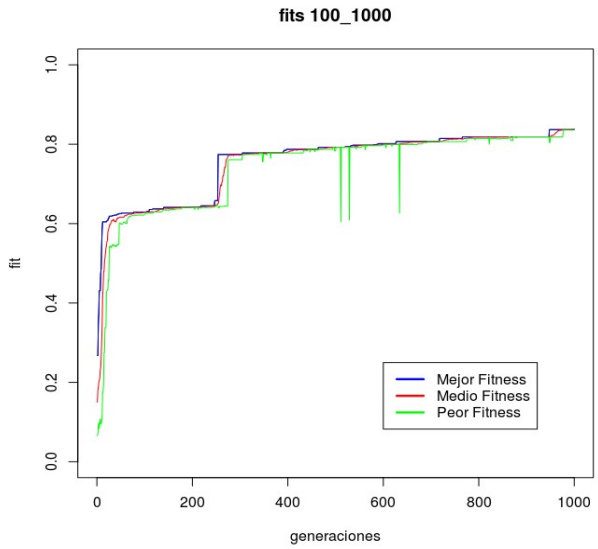


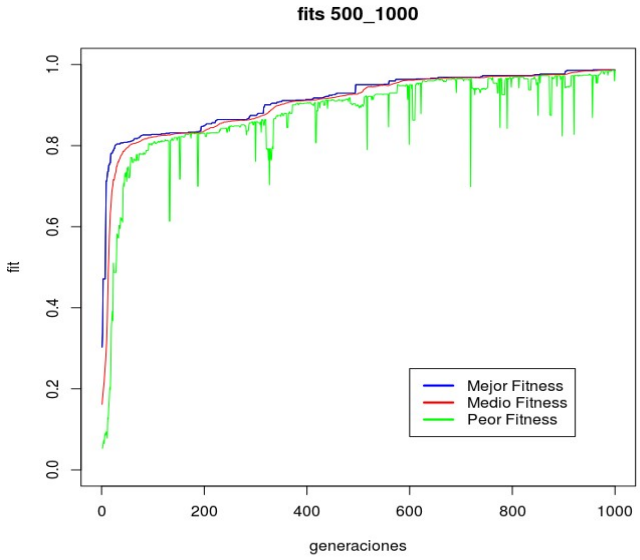


Como podemos observar y ya habíamos comentado, cuanta mas población haya mejores resultado obtendremos. También podemos observar otra cosa curiosas y es que cuantos mas individuos hay es menos probable obtener una prueba que sea significativamente mejor o peor que el resto como si ocurre con poblaciones bajas (100, 10 individuos). Esto se debe sin duda a que cuantos mas individuos hay es mas probable que aleatoriamente obtengamos individuos parecidos en una prueba y en otra mientras que cuanto mas pequeña es la población esto es mas difícil.

A continuación mostramos el mismo tipo de gráficas que hemos puesto antes solo que con la pruebas de **1000 generaciones**.



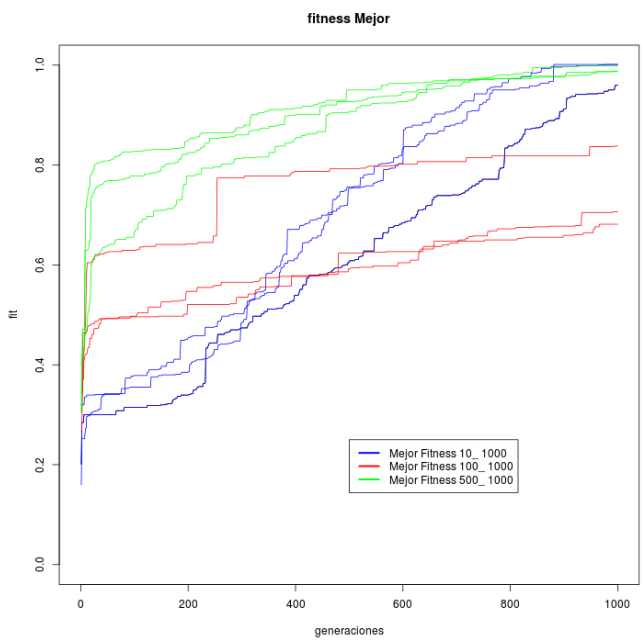


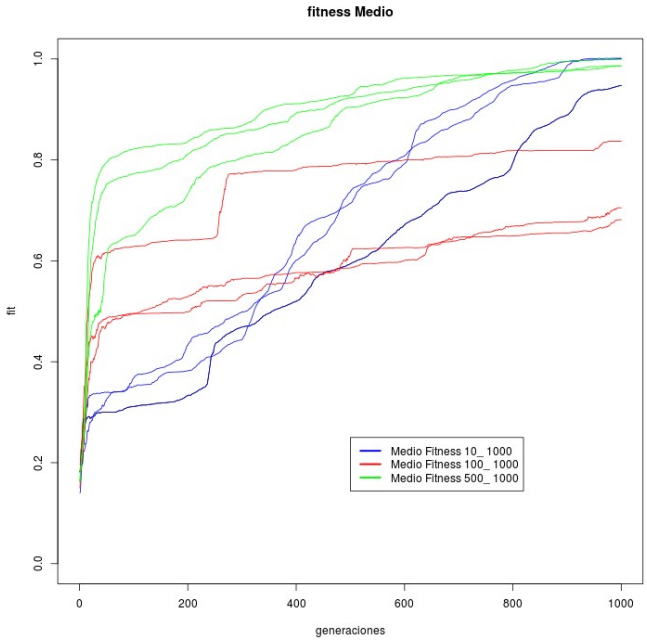


Como vemos obtenemos unas gráficas bastante interesantes, mientras que con generaciones de 100 individuos el fitness solía hacerse máximo muy rápidamente aquí observamos cosas distintas según la población. Lo primero destacable es el incremento casi constante del fitness para poblaciones de 10 individuos, entendemos que esto se debe a que al ser la población muy baja se producen pocos cruces por tanto es poco probable que al cruzar 2 individuos nos salga uno mucho mejor, por eso se ve como mas o menos el mejor de la población se mantiene constante y aumenta poco a poco.

Lo que observamos para una población de 100 es lo habitual salvo por el pequeño detalle de que en un determinado momento entre la generación 300 y 400 aparece un individuo muy bueno que dispara el fit de la población, esto es bastante curioso ya que los mas probable es que aleatoriamente se haya producido un cruce o una mutación que hayan generado un individuo muy bueno y se ve como en sucesivas generaciones la presencia de este individuo incrementa el fit de toda la población hasta que se vuelve a estabilizar.

En cuanto a lo que observamos en una población de 500 individuos no se puede destacar demasiado ya que es lo que esperamos obtener, una población que mejora muy rápidamente pero que llegada una cantidad de generaciones no se puede adaptar mas al medio ya que se esta convirtiendo en muy individuo para resolver el problema.





En cuanto a estas ultimas gráficas es muy llamativo observar como una población de 10 individuos se adapta mejor que una población de 100 individuos pasadas las 400 generaciones. Quizá esto se debe a que, aunque en la población de 10 haya menos individuos, estos son mas diferentes entre si, ya que es mas difícil que las primeras generaciones obtengamos un individuo muy bien adaptado, y sin embargo si obtenemos varios que se adaptan mas o menos al problema pero que al no ser tan buenos no se cruzaran con tanta población, manteniéndose así secuencias genéticas interesantes que con unos determinados cruces pueden obtener un buen resultado, mientras que en una población de mas individuos es mas probable que obtengamos un individuo que clasifique mejor al principio y de este modo se eliminen otras secuencias interesares del inicio al cruzarse mucho con la población.