

# AKADEMIA GÓRNICZO-HUTNICZA

Wydział Elektrotechniki, Automatyki,  
Informatyki i Elektroniki



KATEDRA INFORMATYKI

Praca magisterska

Zastosowanie metod eksploracji danych  
oraz sieci neuronowych w przewidywaniu  
i analizie danych giełdowych.

Autor: **Daniel Mroczka**

Kierunek: **Informatyka**

Promotor: **dr inż. Robert Marcjan**

Kraków 2006

---

## SPIS TREŚCI

<b>WSTĘP.....</b>	<b>5</b>
<b>CZĘŚĆ I SZTUCZNA INTELIGENCJA.....</b>	<b>7</b>
<b>1 EKSPŁORACJA DANYCH.....</b>	<b>9</b>
1.1 WPROWADZENIE DO EKSPŁORACJI DANYCH .....	10
1.2 ZDEFINIOWANIE ZADANIA .....	11
1.3 PROCES ODKRYWANIA WIEDZY .....	12
1.4 GROMADZENIE DANYCH.....	13
1.5 PRZYGOTOWANIE DANYCH.....	14
1.5.1 CZYSZCZENIE DANYCH ( <i>DATA CLEANING</i> ) .....	14
1.5.2 INTEGRACJA DANYCH ( <i>DATA INTEGRATION</i> ) .....	17
1.5.3 TRANSFORMACJA DANYCH ( <i>DATA TRANSFORMATION</i> ).....	18
1.5.4 REDUKCJA DANYCH ( <i>DATA REDUCTION</i> ) .....	20
1.6 METODOLOGIE EKSPŁORACJI DANYCH I ICH ZASTOSOWANIE.....	20
<b>1.6.1 ODKRYWANIE ASOCJACJI.....</b>	<b>21</b>
1.6.2 ODKRYWANIE KLASYFIKACJI .....	21
1.6.3 KLASTROWANIE .....	22
1.6.4 ODKRYWANIE WZORCÓW SEKWENCJI.....	23
1.6.5 WYKRYWANIE ZMIAN I ODCHYLEŃ .....	23
1.6.6 ODKRYWANIE PODOBIEŃSTW W PRZEBIEGACH CZASOWYCH .....	24
1.6.7 PRZEWIDYWANIE PRZEBIEGÓW CZASOWYCH .....	24
1.6.8 ZASTOSOWANIA EKSPŁORACJI DANYCH – PODSUMOWANIE. ....	25
1.7 PODSUMOWANIE .....	25
<b>2 SZTUCZNE SIECI NEURONOWE .....</b>	<b>26</b>
2.1 ROZWÓJ SIECI NEURONOWYCH.....	27
2.2 PODSTAWY BIOLOGICZNE.....	27
2.3 MODEL SZTUCZNEGO NEURONU .....	28
2.3.1 MODEL NEURONU MCCULLOCHA-PITTSA .....	29
2.3.2 NEURON SIGMOIDALNY .....	30
2.4 RODZAJE SIECI NEURONOWYCH.....	31
2.5 UCZENIE SIECI NEURONOWYCH .....	32
2.5.1 METODY UCZENIA .....	33

---

2.5.2	REGUŁY UCZENIA.....	34
2.5.3	ALGORYTMY UCZENIA.....	35
2.5.4	DOBÓR PARAMETRÓW UCZENIA .....	38
2.5.5	ZDOLNOŚĆ GENERALIZACJI SIECI .....	40
2.5.6	POMIAR JAKOŚCI DZIAŁANIA SIECI NEURONOWYCH .....	42
2.6	SIECI JEDNOKIERUNKOWE TYPU SIGMOIDALNEGO .....	43
2.7	ZADANIA SIECI NEURONOWYCH .....	44
2.8	ZASTOSOWANIE.....	45
<b>3</b>	<b>ALGORYTMY GENETYCZNE .....</b>	<b>46</b>
3.1	NAZEWNICTWO .....	46
3.2	TEORIA DOBORU NATURALNEGO .....	48
3.3	OPTIMALIZACJA, CZYLI POSZUKIWANIE NAJLEPSZEGO ROZWIĄZANIA .....	48
3.4	METODY KODOWANIA.....	50
3.5	PROSTY ALGORYTM GENETYCZNY .....	51
3.6	ZASTOSOWANIE.....	57
3.7	OPTIMALIZACJA SIECI NEURONOWEJ ZA POMOCĄ AG .....	58
3.8	TWIERDZENIE O SCHEMATACH .....	58
	<b>CZĘŚĆ II ANALIZA GIEŁDOWA .....</b>	<b>60</b>
<b>4</b>	<b>ANALIZA GIEŁDOWA.....</b>	<b>60</b>
<b>4.1</b>	<b>NOTOWANIA GIEŁDOWE .....</b>	<b>60</b>
4.1.1	CENY I WOLUMEN .....	61
4.1.2	INDEKSY GIEŁDOWE .....	62
4.2	ANALIZA TECHNICZNA .....	63
4.2.1	TRZY PODSTAWOWE PRZESŁANKI ANALIZY TECHNICZNEJ .....	63
4.2.2	WYBRANE WSKAŹNIKI ANALIZY TECHNICZNEJ .....	65
4.2.3	FORMACJE CENOWE.....	68
4.2.4	LINIE TRENDU .....	71
4.3	ANALIZA FUNDAMENTALNA.....	71
4.4	ANALIZA PORTFELOWA .....	72
4.5	PODSUMOWANIE .....	72

---

---

<b>CZĘŚĆ III REALIZACJA PRAKTYCZNA.....</b>	<b>73</b>
<b>5 ARCHITEKTURA SYSTEMU AMADEUS .....</b>	<b>74</b>
5.1 BUDOWA SYSTEMU.....	74
5.2 DEFINIOWANIE I NADZOROWANIE ORAZ WYKONYWANIE ZADAŃ .....	76
5.3 SIEĆ NEURONOWA .....	83
5.4 ALGORYTM GENETYCZNY .....	84
5.5 DOBÓR DANYCH WEJŚCIOWYCH .....	91
5.6 REALIZACJA PRAKTYCZNA .....	92
5.6.1 PAKIETY I KLASY ZWIĄZANE Z WARSTWĄ BIZNESOWĄ .....	92
5.6.2 BAZA DANYCH .....	107
5.6.3 ORGANIZACJA STRON JSP.....	110
<b>6 WYNIKI DOŚWIADCZALNE.....</b>	<b>111</b>
6.1 PARAMETRY MAJĄCE WPŁYW NA PRACĘ SIECI NEURONOWEJ .....	112
6.1.1 WPŁYW ILOŚCI NEURONÓW W WARSTWIE UKRYTEJ .....	112
6.1.2 POPRAWNOŚĆ WYBORU ZBIORÓW UCZĄCYCH.....	115
6.1.3 WPŁYW KSZTAŁTU PRZEBIEGU DANYCH NA JAKOŚĆ NAUCZANIA.....	119
6.1.4 ZALEŻNOŚĆ JAKOŚCI PREDYKCJI OD CZASU TRWANIA PREDYKCJI.....	120
6.1.5 AKTUALIZACJA ZBIORU UCZĄCEGO PODCZAS UCZENIA SIECI .....	122
6.2 OPTIMALIZACJA TOPOLOGII SIECI ALGORYTMEM GENETYCZNYM .....	125
6.3 PODSUMOWANIE .....	128
<b>7. WNIOSKI .....</b>	<b>131</b>
<b>BIBLIOGRAFIA .....</b>	<b>133</b>
<b>SPIS ILUSTRACJI .....</b>	<b>135</b>

---

# Wstęp

Przez prognozę danego zjawiska należy rozumieć wskazanie najbardziej prawdopodobnej drogi jego rozwoju w oparciu o posiadaną wiedzę, dotychczasowy przebieg tego zjawiska oraz wiedzę o aktualnym jego stanie. W starożytnej Grecji Hipokrates<sup>1</sup> oparł medycynę na zasadach racjonalnych poprzez wnikliwą obserwację lekarską i logiczne wyciąganie wniosków prognostycznych. Stąd też pochodzi etymologia tego słowa (z greckiego: gnoza – wiedza, prognoza – uprzednia wiedza).

Prognozować można zjawiska gospodarcze, społeczne, polityczne, rozwoju całego świata, lokalnej społeczności, rozwoju technologii etc. Przedmioty budowania prognoz można by wymieniać niemalże w nieskończoność.

Istota racjonalnego prognozowania polega na budowaniu modeli rzeczywistości, a następnie obserwowaniu, jak się one zachowują z upływem (modelowego) czasu<sup>2</sup>. Jakość prognozy zależy głównie od dwóch czynników – od tego na ile model jest ścisły, tzn. sformalizowany matematycznie i na ile wiernie odzwierciedla on realia.

Posiadanie wiedzy na temat kształtowania się w przyszłości pewnych zachowań jest bardzo cenne i pożądane. Głównym powodem dążenia do pozyskiwania takich informacji jest chęć większego zysku poprzez lepsze zarządzanie przedsiębiorstwem, czy też dokonanie trafniejszych inwestycji finansowych (np. giełdowych).

Wykorzystanie metody sztucznej inteligencji, jaką jest eksploracja danych, to w analizie giełdowej trudne i zarazem ryzykowne zadanie. Trudne, bo wymaga zgłębienia obszernej wiedzy dotyczącej zagadnień sztucznej inteligencji, a także rynku giełdowego. Ryzykowne, gdyż panuje przekonanie o niemożności dokonywania prognoz rynku giełdowego. Często za przyczynę podaje się tezę, że nie istnieje związek pomiędzy danymi historycznymi a przyszłymi cenami na giełdzie (według Hipotezy Rynku Efektywnego), a także brak matematycznego modelu rynku giełdowego. Z pewnością jednak zachowań giełdowych nie można określić jako zachowania w pełni losowego. Przyczyn zmian kursu akcji w czasie należy szukać w wydarzeniach gospodarczych, politycznych, klimatycznych. Oczywiście to nie wszystkie możliwe składowe mające wpływ na kształtowanie się zmian.

---

<sup>1</sup> (ok. 460–377 p.n.e.), lekarz grecki., zwany ojcem medycyny

<sup>2</sup> [Wyrozumski]

Celem niniejszej pracy jest przedstawienie podstawowych zagadnień i pojęć związanych ze sztuczną inteligencją oraz rynkiem giełdowym zwieńczone realizacją narzędzia umożliwiającego analizę i predykcję rynku giełdowego. W pracy starano się umieścić informacje niezbędne do ogólnego poznania i zrozumienia tej tematyki.

Dodatkowo poprzez realizację praktyczną systemu i poprzez wykonanie doświadczeń starano się wykazać, że wykorzystując własności rynku giełdowego, a w szczególności przesłanki analizy technicznej, istnieje możliwość (przy określonych warunkach) stworzenia predykcji odzwierciedlającej przyszłe trendy notowań giełdowych.

Praca została podzielona na trzy części: dwie pierwsze omawiają aspekty teoretyczne zagadnień, ostatnia natomiast opisuje realizację praktyczną pracy oraz uzyskane wyniki doświadczeń.

Część pierwsza pracy została poświęcona zagadnieniom sztucznej inteligencji.

W rozdziale pierwszym omówiono aspekty eksploracji danych. Został tu opisany cały proces postępowania od momentu sformułowania zadania, aż do etapu otrzymania odpowiedzi na pytanie, które postawiono w zadaniu.

Drugi rozdział poświęcono sieciom neuronowym, które są jedną z metod eksploracji danych, stanowiących temat niniejszej pracy.

Rozdział trzeci opisuje natomiast zagadnienia dotyczące algorytmów genetycznych użytych w pracy do optymalizacji sieci neuronowych, by w rezultacie osiągnąć wyniki o lepszej jakości.

Część druga przybliży zagadnienia analizy danych giełdowych. Sama znajomość rynku giełdowego nie jest konieczna do predykcji za pomocą sieci neuronowych. Jednak podstawowy zasób wiedzy z dziedziny analizy technicznej pozwoli wyjaśnić zaobserwowane w części praktycznej zachowania, a dodatkowo poprawnie wykorzystana wiedza pozwala zwiększyć jakość prognozy. Omówione zostały wskaźniki będące pomocne przy uczeniu sieci neuronowej oraz charakterystyczne formacje i trendy.

Część trzecia - praktyczna - obejmuje implementację systemu, który realizuje proces predykcji notowań giełdowych wraz z zamieszczeniem wyników i wniosków z przeprowadzonych doświadczeń.

---

# CZEŚĆ I SZTUCZNA INTELIGENCJA

*„Połowa tego, co się mówi o sztucznej inteligencji nie jest prawdą,  
druga połowa nie jest możliwa”*

D. PATRIDGE

Sztuczna inteligencja (ang. *Artificial Intelligence* - AI) to dział informatyki, którego zadaniem jest konstruowanie maszyn i oprogramowania zdolnego rozwiązywać problemy nie poddające się algorytmizacji - wymagające inteligencji, gdy te są rozwiązywane przez człowieka.

Problemy takie określa się mianem AI-trudnymi, a zalicza się do nich np. analizę i syntezę języka naturalnego, rozumowanie logiczne, gry logiczne z szachami będącymi charakterystycznym przykładem intelektualnej rywalizacji człowieka i sztucznej maszyny.

Do grona przykładów wykorzystania sztucznej inteligencji należy także zaklasyfikować dział związany z manipulacją wiedzy, a mianowicie z systemami doradczymi i diagnostycznymi

Historia badań nad sztuczną inteligencją rozpoczęła się w latach pięćdziesiątych ubiegłego wieku wraz z rozwojem techniki mikroprocesorowej, jednak znacznie wcześniej w umysłach naukowców rodziły się marzenia o budowie inteligentnej maszyny.

Rozróżnia się dwa kierunki prac nad sztuczną inteligencją. Pierwszy z nich zakłada tworzenie modeli matematycznych analizowanych problemów i implementowanie ich w postaci programu komputerowego. Drugi kierunek preferuje właściwość „samouczenia” się na podstawie danych uczących, a następnie wykorzystywaniu zdobytej wiedzy w praktycznych systemach (np. decyzyjnych). Zaletą tych drugich jest mniejsza ingerencja człowieka w proces podejmowania decyzji przez sztuczną maszynę. Człowiek co prawda dostarcza gotowych pytań i odpowiedzi, jednak w rzeczywistych warunkach pracy to maszyna podejmuje decyzję na podstawie zdobytych umiejętności w trakcie nauczania. Człowiek pełni jedynie rolę „nauczyciela” a maszyna rolę „ucznia”. Do takich struktur samouczących się zaliczyć można sztuczne sieci neuronowe, którym poświęcono w całości rozdział drugi niniejszej pracy.

Szybko zorientowano się, że umiejętnie zbudowany system może posłużyć do prognozowania sytuacji w środowisku opisanym liczbami. Rozdział pierwszy dotyczący eksploracji danych ukazuje korzyści wynikające z zastosowania metod sztucznej inteligencji w systemach doradczych i decyzyjnych. Prognoza oparta o wynik pracy procesu odkrywania wiedzy

dorównuje jakości prognozy stworzonej przez człowieka, a częstokroć jest jedynym narzędziem mogącym taką prognozę wygenerować (ze względu na bardzo dużą ilość danych branych pod uwagę w analizie).

Badania nad sztuczną inteligencją są ciągle prowadzone, gdyż do tej pory nie udało się uzyskać satysfakcjonujących wyników w wielu dziedzinach. Do tej pory tworzone systemy naśladowujące konwersację ludzką w trakcie testów Turinga<sup>3</sup> wykazują przewagę człowieka nad maszyną.

Ciągle prowadzone są także spory o to, czy maszyna potrafi skutecznie wygenerować zysk poprzez grę na giełdzie akcji. Prowadzone dotychczas prace nie przyniosły zadawalającego rezultatu. Z drugiej jednak strony wynikać to może ze zbyt wygórowanych oczekiwań dotyczących sztucznej inteligencji. Trudno jest przewidzieć sytuacje losowe wywołane np. kataklizmami lub wojną, jednak w pozostałych przypadkach można z dużą dozą prawdopodobieństwa przewidywać trend korzystając z wiedzy zdobytej poprzez analizę rynków giełdowych.

---

<sup>3</sup> Sposób określania zdolności maszyny zaproponowany przez Alana Turinga w 1950 roku, polega na konwersacji człowieka w języku naturalnym z pozostałymi stronami. Jeżeli człowiek pełniący rolę sędziego nie jest w stanie rozróżnić rozmówcy czy jest maszyną czy człowiekiem wówczas stwierdzamy że maszyna przeszła test Turinga.



---

# 1 Eksploracja danych

„Sekretem biznesu jest wiedzieć to,  
czego nie wiedzą inni.”

ARISTOTELES ONASSIS

Rozwój systemów komputerowych spowodował, że z roku na rok rośnie w coraz większym stopniu ilość pozyskiwanych i gromadzonych danych. Dane pobierane są z wielu źródeł, począwszy od tych, z którymi można mieć do czynienia na co dzień (dane transakcji w supermarketach, użytkowanie kart kredytowych w płatnościach, billingi telefoniczne czy też rządowe dane statystyczne), po te związane z badaniami naukowymi (np. astronomia, fizyka i medycyna).

W miarę wzrostu ilości danych zauważono, że zgromadzone przez firmy dane nie muszą służyć wyłącznie do prowadzenia księgowości przedsiębiorstw. Zauważono, że w danych tych zawarta jest dodatkowa (ukryta) wiedza, którą można wykorzystać do powiększania zysków firmy poprzez polepszenie oferty handlowej lub lepsze rozmieszczenie towarów na półkach czy też oszczędności w optymalnej organizacji działania przedsiębiorstwa. Można dowiedzieć się np. co jest przyczyną spadku sprzedaży produktów lub rosnących kosztów produkcji.

Zagadnienia związane z wydobywaniem zawartej wiedzy w danych (*knowledge discovery*)<sup>4</sup> przyjęto nazywać eksploracją danych (*data mining*), w polskojęzycznej literaturze naukowej istnieje także pod pojęciem zgłębiania lub drążenia danych.

Eksploracja danych<sup>5</sup> to analityczny proces odkrywania istotnych korelacji, wzorców i trendów poprzez dogłębną analizę dużych zbiorów danych przechowywanych w bazach danych, przy użyciu technologii automatycznego rozpoznawania wzorców (uczenia maszynowego), oraz technik statystycznych i matematycznych. Rezultatem odkrywania zależności niewidocznych gołym okiem są dane łatwo przyswajalne dla posiadacza bazy danych z informacjami<sup>6</sup> pomagając w podejmowaniu (lepszyc) decyzji. Termin eksploracja danych jest często używany jako synonim procesu odkrywania wiedzy w bazach danych. W literaturze<sup>7</sup> czasami jednak rozróżnia się te dwa pojęcia, według której termin odkrywania wiedzy odnosi się do

---

<sup>4</sup> [Bigus]

<sup>5</sup> [www.businessintelligence.pl/slowniczek.htm](http://www.businessintelligence.pl/slowniczek.htm)

<sup>6</sup> [Hand]

<sup>7</sup> [Fayyad]

---

całego procesu, natomiast eksploracja danych stanowi tylko jeden z etapów tego procesu odnoszący się do generowania reguł.

Pozostałe etapy procesu odnoszą się do przygotowania danych, wyboru danych do eksploracji, czyszczenia danych, definiowania dodatkowej wiedzy przedmiotowej, interpretacji wyników eksploracji i ich wizualizacji.

## 1.1 Wprowadzenie do eksploracji danych

Zainteresowanie eksploracją danych wynika z faktu, że wiele przedsiębiorstw, instytucji administracji publicznej, czy wreszcie ośrodków naukowych zgromadziło w ostatnim czasie bardzo wiele danych przechowywanych w bazach danych i stanęło przed problemem, w jaki sposób efektywnie i racjonalnie wykorzystać zgromadzoną w tych bazach wiedzę dla celów wspomagania działalności biznesowej.

Sposób, w jaki użytkownik korzysta z bazy danych (w jaki realizuje do niej dostęp) nazywa się *modelem przetwarzania*. Na przestrzeni ostatnich trzydziestu lat oprócz eksploracji danych można wyróżnić następujące modele przetwarzania:

- Tradycyjny sposób korzystania z baz danych ogranicza się zwykle, do realizacji systemów pytających – raportujących (*Q&R* - ang. *Queries and Report*). Narzędzia tego typu „odpowiadają” na pytanie typu „co się stało?”.
- Model przetwarzania danych w trybie on-line (ang. *On Line Analytical Processing OLAP*) to model umożliwiający użytkownikom wielowymiarowych baz danych tworzenie interakcyjnych zapytań analitycznych pokazujących przekroje (np. statystyczne) danych, np. szeregi czasowe lub trendy. OLAP jest w pełni satysfakcjonujący w przypadku bieżącej obsługi działalności danej firmy, dla dobrze zdefiniowanych procesów (obsługa bankowa klienta, rejestracja zamówień, obsługa sprzedaży, itp.). OLAP pozwala nie tylko uzyskać odpowiedź na pytanie „kto?”, „co?” i „kiedy?” ale również „dlaczego?”.

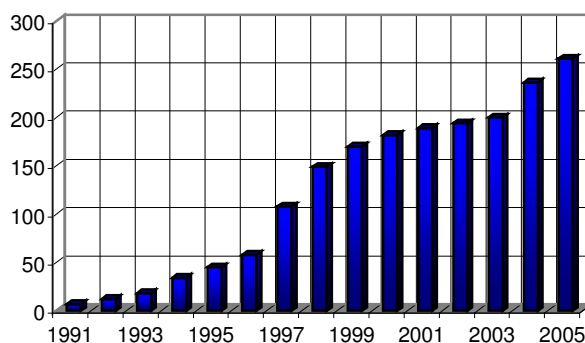
Zapytania i raporty	OLAP	Data mining
Wybór pojedynczych i zsumowanych danych	Podsumowania, trendy i prognozy	Odkrywanie niewidocznych związków
Informacja	Analiza	Wgląd i przewidywanie
Kto kupił produkt w ostatnim roku?	Jaki jest średni przychód od klienta, który kupił produkt w ciągu ostatnich dwóch lat z podziałem na regiony?	Kto i dlaczego kupi produkt w najbliższym półroczu?

Rys. 1.1 Porównanie zadań modeli przetwarzania: Q&R, OLAP i Data Mining<sup>8</sup>

<sup>8</sup> Gazeta IT nr 2(21) luty 2004

Na rys. 1.1 zostały zaprezentowane zadania wyżej wymienionych modeli przetwarzania. Tradycyjny model przetwarzania danych nie wspomaga procesów analizy danych oraz aplikacji wspomagających podejmowanie decyzji. OLAP z kolei umożliwia obrazowanie trendów i tworzenie prognoz, natomiast użycie Data Miningu umożliwi odkrycie niewidocznych związków w zbiorze danych.

Posiadanie wiedzy dotyczącej przyszłości jest bardzo cenne, pozwala korzystnie zainwestować pieniądze unikając ryzyka. W przypadku rynku giełdowego, posiadając niezbędną wiedzę oraz korzystając z wielu technik [Rozdział 4] można z mniejszym bądź większym prawdopodobieństwem oszacować kierunek i wartość zmian. W tym celu z przebiegów czasowych reprezentujących kursy akcji, należy wyodrębnić: formacje, trendy i inne punkty charakterystyczne. Wykonanie tego bez użycia komputera w dzisiejszych czasach jest bezspornie nieefektywne, dodatkowo ilość spółek giełdowych z każdym rokiem wzrasta (rys. 1.2)



Rys. 1.2 Ilość spółek giełdowych notowanych na Warszawskiej Giełdzie Papierów Wartościowych

Wykorzystując Data Mining, a dokładnie jedną z jego metodologii, jaką są sieci neuronowe można stworzyć swego rodzaju „silnik” (engine), który łatwo będzie można dostosować do innych zadań, oczywiście po drobnych modyfikacjach na łączy baza danych – interpretacja danych.

Niestety do zadania nie należy podchodzić z wielkim entuzjazmem. Grono specjalistów zajmujących się dziedziną analizy giełdowej poprzez sieci neuronowe, dzieli się na dwa fronty: tych którzy twierdzą, że rynek giełdowy da się przewidzieć za pomocą sieci neuronowej oraz tych którzy twierdzą że takiej możliwości nie ma.

## 1.2 Zdefiniowanie zadania

Przygotowanie modelu do predykcji rynku giełdowego, należy poprzedzić precyzyjnym zdefiniowaniem oczekiwań, z których pierwsze dotyczy ilości dni prognozy, czyli informacji o czasie trwania inwestycji. Należy odpowiedzieć także na pytanie, jakie dane są potrzebne

inwestorowi w dokonywaniu inwestycji giełdowych. Wydaje się być logiczne, że najważniejszą informacją jest cena zamknięcia za  $x$  sesji. Jednak z psychologicznego punktu widzenia, inwestor który chce na własne oczy zobaczyć przebieg notowań w okresie przyszłej inwestycji, na podstawie wykresu może ocenić czy osiągnięty wynik nie odbiega od przebiegu wykresu.

Kolejne pytanie, czy bardziej interesująca z punktu widzenia inwestora jest wartość akcji za  $x$  sesji, czy też *stopa zwrotu*<sup>9</sup> z inwestycji? Podczas inwestycji łatwiej sobie wyobrazić zysk procentowy.

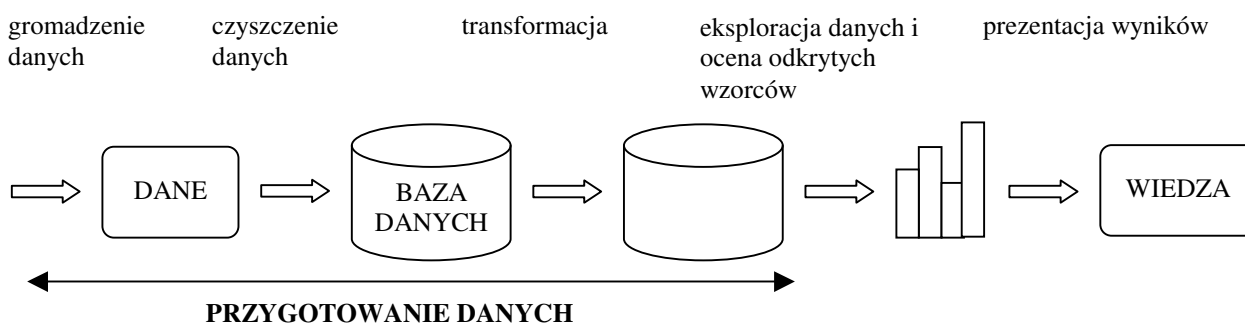
Bardzo istotne, aby już na poziomie definiowania zadania określić wskaźniki osiągnięcia sukcesu lub porażki

### 1.3 Proces odkrywania wiedzy

Algorytmy eksploracji danych wymagają przygotowanych danych spełniających odpowiednie kryteria, które należy uprzednio zgromadzić w bazie danych (ang. *Database*) lub w hurtowniach danych (ang. *Data Warehouse*). Dane mogą być jednak “zanieczyszczone” i dlatego wskazane może być ich czyszczenie [1.4.1]. W zależności od przyjętego algorytmu dane mogą wymagać transformacji do żądanej postaci tak, aby algorytm mógł funkcjonować lub aby działał efektywniej.

Po wykonaniu wyżej opisanych czynności następuje moment, kiedy dysponując gotowymi danymi, można zasilić mechanizmy eksploracji danych. Czynności wykonane do tej pory można zaklasyfikować jako *przygotowanie danych* (ang. *data preprocessing*).

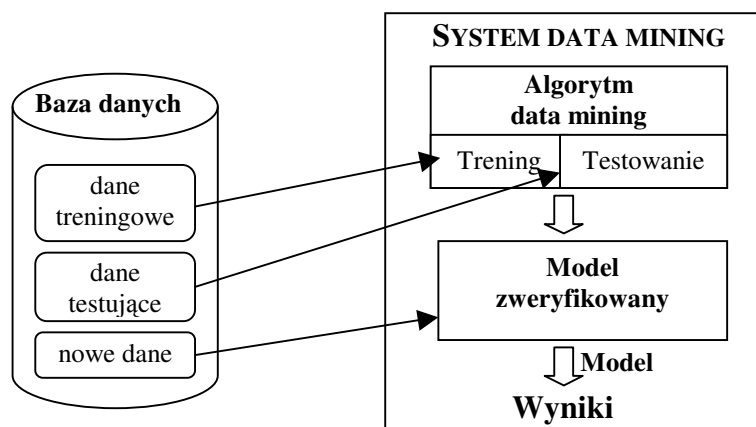
W zależności od wybranej metodologii [1.6] wykonywane są czynności opisane przez reguły postępowania poszczególnych algorytmów. Po zakończeniu zadania należy sprawdzić, jakiej „jakości” są uzyskane wyniki, a następnie przedstawić w wizualnej formie (ang. *postprocessing*), gdyż taka forma jest bardziej zrozumiała dla człowieka.



Rys. 1.3 Przebieg procesu odkrywania wiedzy

<sup>9</sup> Stopa zysku to stosunek wysokości zysku do wysokości zaangażowanego kapitału; jest względną miarą zyskowności inwestycji, umożliwia porównywanie zyskowności różnych inwestycji.

Dane historyczne gromadzone w bazie dzieli się na trzy rodzaje w zależności od wykorzystania ich w trakcie eksploracji danych. Są to mianowicie: dane treningowe (na których system uczy się zależności), dane testowe (często używa się podziału tego zbioru na dane sprawdzające i testujące), nowe dane (jako wynik działania algorytmu).

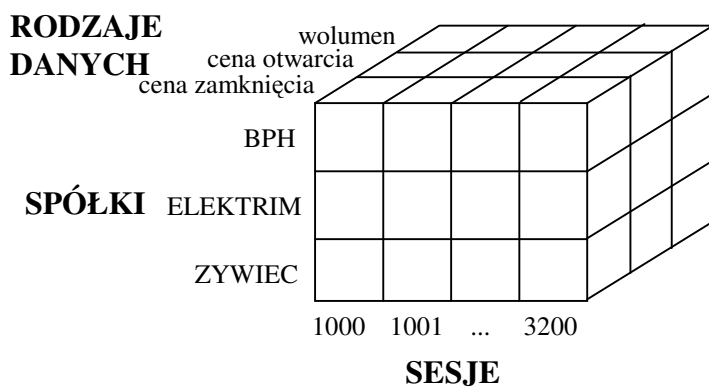


Rys. 1.4 Rodzaje i udział danych w procesie eksploracji danych

Rezultatem algorytmu data miningu jest wypracowanie odpowiedniego modelu (np. w przypadku sztucznych sieci neuronowych będzie nim zbudowanie optymalnej topologii sieci). Dopiero po podaniu na wejście takiego modelu danych można otrzymać odpowiedź na postawione wcześniej pytanie.

## 1.4 Gromadzenie danych

Każda metoda analityczna wymaga do działania danych, gromadzonych w specjalnie dostosowanych strukturach.. W niniejszej pracy zbudowano bazę danych zaprezentowaną w rozdziale [5], przystosowaną do przechowywania danych następującej postaci:



Rys. 1.5 Kostka danych przedstawia trójwymiarowy charakter danych z notowań giełdowych <sup>10</sup>

<sup>10</sup> [Han]

Przedstawiona powyżej struktura zwana kostką danych (ang. *data cube*) posiada trzy wymiary określone przez takie wielkości jak: numer sesji, nazwa spółki giełdowej i rodzaj danych.

Posiadając wszystkie trzy parametry można odczytać elementarną informację z tejże kostki będącą np. ceną zamknięcia spółki *ŻYWIEC* dla sesji numer *1000*.

## 1.5 Przygotowanie danych

Posiadanie surowych danych wymaga odpowiedniego przygotowania do analiz. Proces ten zabiera ok. 60-90%<sup>11</sup> czasu poświęconego na eksplorację danych, a od jakości jego wykonania zależy 75-90% powodzenia całego przedsięwzięcia.

Należy pamiętać, że niezależnie od przyjętej metody analizy odpowiednia jakość danych jest kluczowym czynnikiem wpływającym na wyniki analizy. Właściwe przeprowadzenie wstępnej analizy danych jest niezbędnym warunkiem uzyskania pożądanego efektu końcowego, którym jest skonstruowanie modelu opisującego w poprawny sposób badany fragment rzeczywistości. Znane jest powiedzenie: śmieci na wejściu – śmieci na wyjściu (ang. *garbage in – garbage out*), oddające wiernie tę regułę.

### 1.5.1 Czyszczenie danych (*Data cleaning*)

Zgromadzone dane mogą być niekompletne, obciążone błędami (zazumione) i niespójne. Do zadań mechanizmu czyszczenia danych należą odpowiednio: wypełnianie brakujących danych, wygładzanie danych (neutralizowanie wpływu szumów), korygowanie niespójnych danych.

#### Brakujące dane

Dane z notowaniami giełdowymi mogą zawierać brakujące informacje w ciągłym notowaniu dla poszczególnych spółek i sesji. Może brakować całego rekordu lub tylko poszczególnych danych.

Sesja	Cena otwarcia	Cena minimalna	Cena maksymalna	Cena zamknięcia	Wolumen
2000	99,00	97,00	101,00	100,00	1000
2001	99,50	<b><u>NULL</u></b>	<b><u>NULL</u></b>	101,00	1020
2003	99,75	98,00	101,25	101,25	<b><u>NULL</u></b>
2004	<b><u>NULL</u></b>	97,75	101,50	101,25	1030

Rys. 1.6 Przykład brakujących danych

<sup>11</sup> [Iebeling] strona 366

Powody braku danych mogą być różne: spółka mogła nie być w ogóle notowana, nie było zainteresowania ze strony inwestorów. Z brakującymi danymi można postąpić następująco:

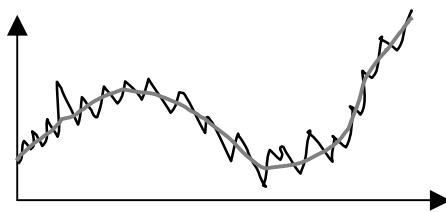
- Zignorować brakujące dane. Metoda ta jest tym bardziej nieefektywna im stopień brakujących danych jest większy.
- Ręcznie uzupełnić dane. Sposób ten z kolei jest nieefektywny w przypadku dużej ilości danych.
- Wstawić rekordy z wartościami typu *NULL* w przypadku braku całego rekordu (np. gdy brakuje danych z poszczególnych sesji). Korzyścią z zastosowania tej metody jest ciągłość danych względem kolejnych sesji oraz informacja o tym, że dane pierwotne były niekompletne.
- Uzupełnić brakujące dane wartością, będącą średnią z sąsiadujących danych. Metoda ta jest prosta i skuteczna w użyciu.
- Użyć najbardziej prawdopodobnej wartości. Brakująca dana może być wygenerowana z wykorzystaniem *drzew decyzyjnych*, jej wartość nie musi zależeć wyłącznie od wartości sąsiadujących danych tej samej kategorii, ale także od innych parametrów. Metoda bardzo skuteczna wymaga jednak już zastosowania zaawansowanych algorytmów do interpretacji danych.

### Zaszumione dane

Szum jest czynnikiem sprawiającym, że dodany do danych powoduje pogorszenie ich jakości. W analizie giełdowej może być także uważany za chwilową fluktuację nie mającą swoich konsekwencji w kolejnych notowaniach. Jednak przeprowadzanie analiz takich danych jest bardzo trudne, a w przypadku użycia algorytmów *uczenia maszynowego* czasami bywa wręcz niemożliwe.

Sesja	Cena otwarcia	Cena minimalna	Cena maksymalna	Cena zamknięcia	Wolumen
2000	99,00	97,00	101,00	100,00	1000
2001	99,50	<b><u>82</u></b>	<b><u>119</u></b>	101,00	1020
2003	99,75	98,00	101,25	101,25	<b><u>1095</u></b>
2004	<b><u>80</u></b>	97,75	101,50	101,25	1030

Rys. 1.7 Przykład zaszumionych danych



Rys. 1.8 Zaszumione dane przed (kolor czarny) i po wygładzeniu (kolor niebieski)

Zaszumione dane koryguje się poprzez następujące techniki wygładzania (ang. *smoothing techniques*).

- Dyskretyzacja (ang. *Binning*), w metodzie tej dokonuje się sortowania wartości a następnie grupuje się w mniejsze części (ang. *buckets*). Wygładzanie może odbywać się przez uśrednianie (ang. *smoothing by bin medians*) lub rozgraniczanie (ang. *smoothing by bin boundaries*) w danej lokalnej grupie. Metoda pierwsza przyjmuje dla wszystkich elementów grupy wartość średnią, metoda druga nie modyfikuje wartości brzegowych, a jedynie dostosowuje wartości w środku grupy do wartości minimum lub maksimum w zależności od tego, które jest bliższe wartości średniej grupy.

Dane posortowane: 4, 8, 15, 21, 21, 24, 25, 28, 34

Podzielenie w grupy: {(4, 8, 15), (21, 21, 24), (25, 28, 34)}

Wynik uśredniania: {(9, 9, 9), (22, 22, 22), (29, 29, 29)}

Wynik rozgraniczania: {(4, 4, 15), (21, 21, 24), (25, 25, 34)}

- W metodzie klastrowania (ang. *Clustering*) lub inaczej klasteryzacji odchylenia wartości rozpoznaje się poprzez grupowanie obiektów, w tzw. klastry. Wartości będące poza klastrami uznaje się, za dane zaszumione. Pojęcie klastrowania zostało szerzej omówione w [1.6.3].
- Metoda kombinowana - „Badanie” danych odbywa się początkowo przez komputer badający odchylenia od normalnego przebiegu. W sytuacji wykrycia anomalii decyzję o sposobie „naprawy” pozostawia człowiekowi.
- Metoda regresji - Dane próbuje się dostosować do przebiegu funkcji odwzorowanej wzorem matematycznym. W miejscach uznanych za zaszumione przyjmuje się wartości uzyskane dzięki regresji liniowej (najczęściej wielopunktowej). Należy pamiętać, że nie każdy przebieg czasowy można zapisać funkcją matematyczną, stąd ograniczenia w zastosowaniu tej metody.



**Niezgodne dane:**

Dane mogą być obarczone błędami wynikłymi z konwersji, mogą zawierać dane nieprawidłowego typu, bądź o niespodziewanych wartościach. Przykładowo, jeśli wartość wolumenu przyjmie postać liczby zmiennoprzecinkowej bądź cena przyjmie wartość ujemną można mieć pewność, że wartość pola jest nieprawidłowa i należy je traktować jako nieważne.

Sesja	Cena otwarcia	Cena minimalna	Cena maksymalna	Cena zamknięcia	Wolumen
2000	99,00	97,00	101,00	100,00	1000
2001	99,50	<u>ABC</u>	<u>-101,00</u>	101,00	1020
2003	99,75	98,00	101,25	101,25	<u>0,01</u>
2004	<u>0</u>	97,75	101,50	101,25	1030

Rys. 1.9 Przykład danych uznanych za niezgodne

**1.5.2 Integracja danych (*Data integration*)**

Dane do analizy są zbierane z wielu źródeł (tabel, baz danych, plików). Dobrze zaprojektowana baza danych eliminuje przedstawione poniżej zagrożenia wynikłe w trakcie integracji danych.

**Redundancja**

Redundancją nazywa się ilość informacji przekraczającą wymagane do rozwiązania problemu minimum. Analizowanie dwóch redundantnych atrybutów jest nie wskazane, nie prowadzi do lepszego rozwiązania a jedynie obciąża zasoby. Wykrywaniu nadmiarowych atrybutów pomaga analiza korelacji. W przypadku analizy danych giełdowych korelację mierzy się względem stóp zwrotu.

Korelacja (powiązanie) stóp zwrotu akcji określa jak duży wpływ mają zmiany cen jednej spółki na ceny innej spółki, bądź czy ulegają takim samym zmianom, co ceny innej spółki. Powiązania można badać zarówno między danymi tej samej spółki giełdowej, jak i pomiędzy danymi różnych spółek. Wzór na wyliczenie korelacji jest przedstawiony w następującej postaci:

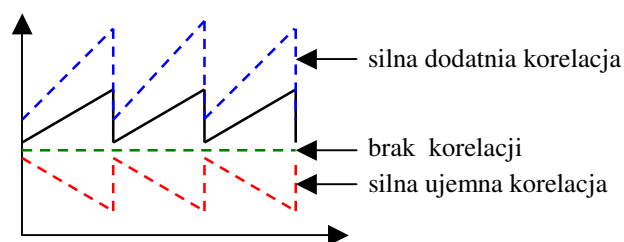
$$\rho_{ij} = \frac{\sum_{i=1}^n (R_{1i} - R_1) * (R_{2i} - R_2)}{(n-1) * (s_1 * s_2)} \quad (1.1)$$

Gdzie:

$\rho_{ij}$  - współczynnik korelacji dla stopy zwrotu akcji pierwszej i drugiej,

$s_1$  – odchylenie standardowe stopy zwrotu akcji pierwszej,

$s_2$  – odchylenie standardowe stopy zwrotu akcji drugiej.



Rys.1.10 Przykład korelacji danych

Jeżeli rezultat równania (1.1) zmierza ku wartości 1 oznacza to, że badane dane są ze sobą silnie skorelowane. Gdy wynik jest dodatni mówi się o dodatniej korelacji, gdy wynik jest ujemny analogicznie mówi się o ujemnej korelacji (dane są odbiciem lustrzanym względem osi  $ox$ ), lub o jej braku, gdy wynik jest zerowy.

### Konflikt typów danych

Występuje w przypadku użycia różnych jednostek w poszczególnych tabelach.

Spółka	Wolumen [szt.]	Spółka	Wolumen [tysiące szt.]
BPH	5000	BPH	5
ELEKTRIM	53000	ELEKTRIM	53
ZYWIEC	17000	ZYWIEC	17

Rys. 1.11 Przykład konfliktu danych związanych z przyjęciem różnych jednostek

### 1.5.3 Transformacja danych (*Data transformation*)

Transformacja danych polega na przekształcaniu lub łączeniu danych w bardziej użyteczną postać. Realizacja procesu bywa konieczna, ponieważ niektóre algorytmy wymagają danych spełniających warunki ze względu na zakres wartości (np. sieci neuronowe przyjmują wartości z przedziału:  $[-1, 1]$  lub  $[0, 1]$ ). Niektóre rodzaje normalizacji mogą okazać się potrzebne w celu uniknięcia problemów numerycznych, takich jak np. utrata precyzji przy działaniach na liczbach zmiennoprzecinkowych.

W celu przeprowadzenia transformacji danych wyróżnia się następujące metody:

#### Normalizacja

Jednym ze sposobów przekształcania danych jest *normalizacja* polegająca na takim przekształceniu zmiennej, aby była porównywalna z ustalonym punktem odniesienia, co jest przydatne, gdy niekompatybilność pomiarów pomiędzy zmiennymi może mieć wpływ na

wyniki analizy. Wynikiem normalizacji może być na przykład takie przekształcenie zmiennej, by jej wartości zawierały się w przedziale  $[0,1]$ .

Istnieje kilka sposobów normalizacji danych. W pierwszym nazywanym *normalizacją min-max* znając minimalną i maksymalną wartość w zbiorze danych, nową znormalizowaną wartość wyznacza się według wzoru

$$v' = \frac{v - \min}{\max - \min} \quad (1.2)$$

Zaletą tej metody jest to, że zachowuje ona prawidłowe relacje pomiędzy wartościami, nie wprowadza żadnych odchyłeń od wartości. Jej wadą z kolei jest to, że można napotkać błędy, jeśli dane wejściowe przekroczą zadany pierwotnie przedział  $[\min, \max]$  (np. dane wyjściowe sieci neuronowej mogą nie spełniać tego warunku).

Wady takiej nie posiada druga metoda *z-score*, która jest przydatna zwłaszcza wtedy, kiedy trudno jest określić wartości minimalne i maksymalne. Do obliczeń potrzebne jest odchylenie standardowe (ang. *standard deviation*) liczone według wzoru (1.4) gdzie  $v_i$  to kolejne wartości ciągu danych przed normalizacją.

$$v' = \frac{v - \bar{v}}{std} \quad (1.3)$$

$$std = \sqrt{\frac{1}{N-1} \sum_{i=1}^N (v_i - \bar{v})^2} \quad (1.4)$$

Normalizacja przez skalowanie dziesiętne (ang. *normalization by decimal scaling*) polega na przesuwaniu przecinka w liczbach dziesiętnych. Ten typ skalowania przekształca dane do przedziału  $[-1, 1]$  według wzoru:

$$v' = \frac{v}{10^j} \quad (1.5)$$

$$\max(|v'|) < 1 \quad (1.6)$$

Przecinek jest przesuwany w prawo (wartość wykładnika  $j$  jest inkrementowana) do momentu, aż wartość bezwzględna po normalizacji będzie mniejsza od liczby 1.

### **Agregacja**

Polega na wyliczeniu jednej z wielu statystyk takich jak suma lub średnia dla grup obserwacji wyznaczonych przez kategorie zmiennych grupujących. I tak do analizy miesięcznej sprzedaży używa się danych agregowanych względem miesiąca, analiza danych za dzień jest tu zbędna.

## Wyglądanie

Ma na celu usunięcie z danych szumów. Techniki takie jak binaryzacja, klasteryzacja i regresja zostały przybliżone w podrozdziale [1.5.1]

### 1.5.4 Redukcja danych (*Data reduction*)

Nie zawsze pożądana jest analiza wszystkich dostępnych danych. Przyjmuje się, że nie powinno się analizować pierwszych 1000 sesji WGPW, kiedy to giełda dopiero się rozwijała, ilość spółek była jeszcze bardzo niewielka, metody analizy giełdowej nie zawsze się sprawdzały etc. Strategie przyjęte przy redukcji danych to m.in.: agregacja danych, kompresja danych, redukcja rozmiarów.

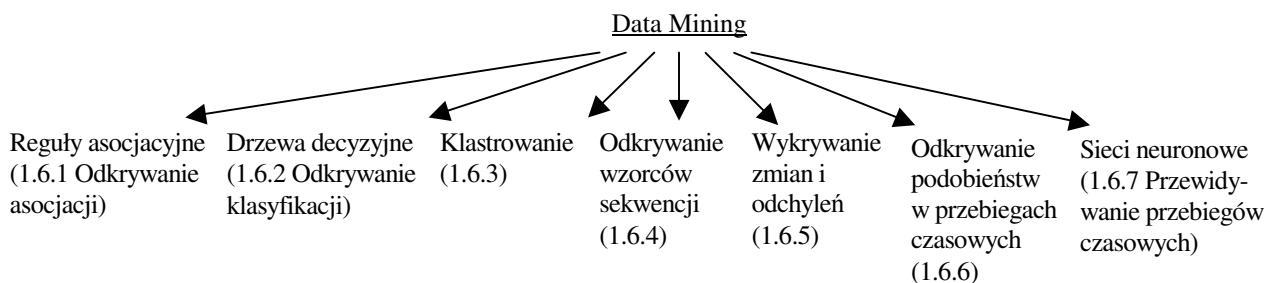
Redukcja rozmiarów ma na celu zminimalizowanie czynników biorących udział w analizie, które wydają się być zbędne lub nie wnoszą większych korzyści przy analizie (mogą być nadmiarowe).

W literaturze<sup>12</sup> wyróżnia się trzy podstawowe metody redukcji danych:

- Selekcja typu *step-wise forward* rozpoczyna się z pustym zbiorem czynników, czynnik znaczący jest dodawany do zbioru.
- Selekcja typu *step-wise backward* w przeciwieństwie do poprzedniej rozpoczyna się z pełnym zbiorem czynników, które są sukcesywnie usuwane, jeśli nie wnoszą niczego znaczącego.
- Kombinacja powyższych metod, podczas której na każdym kroku dodawany jest najlepszy czynnik a usuwany najgorszy.

## 1.6 Metodologie eksploracji danych i ich zastosowanie

Główne techniki eksploracji danych reprezentuje poniższy diagram na rys 1.12 i zostały one szerzej omówione w kolejnych podpunktach.



Rys. 1.12 Najważniejsze techniki eksploracji danych

<sup>12</sup> [Han]



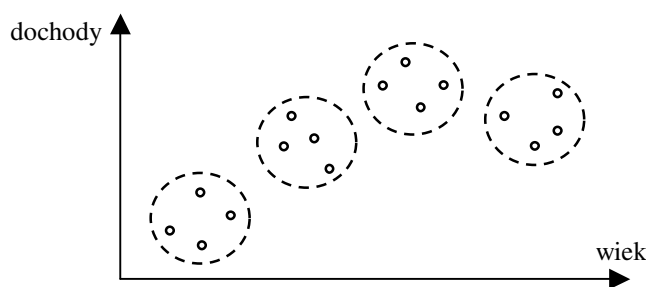
Klasyfikacja to problem znany i analizowany od dawna, zwłaszcza, w dziedzinie sztucznej inteligencji i uczenia maszynowego. Na fundamencie tych dziedzin opracowano wiele metod odkrywania klasyfikacji takich jak: drzewa decyzyjne, sieci neuronowe, czy algorytmy genetyczne

Zastosowanie: charakterystyka pacjentów, klientów kart kredytowych, pożyczkobiorców etc.

### 1.6.3 Klastrowanie

Klastrowanie (ang. *clustering*) lub inaczej grupowanie to poszukiwanie skończonego zbioru kategorii, opisujących dane. Klasyfikacja polega na przypisywaniu obiektów do klas, natomiast klastrowanie poszukuje skończonego zbioru klas obiektów (klastrow) w zbiorze danych odznaczających się podobnymi cechami. Początkowo liczba klastrow nie jest znana i dlatego proces grupowania przebiega dwuetapowo. Pierwszy wyznacza możliwe klastry by podczas drugiego cyklu znaleźć optymalny podział obiektów pomiędzy klastry.

Problem klastrowania danych, nazywany także taksonomią danych, jest analizowany w ramach sztucznej inteligencji i uczenia maszynowego od kilku lat. Wyróżnia się wiele algorytmów klastrowania danych. Różnice pomiędzy algorytmami wynikają z rodzaju danych: dane mogą być ciągłe, numeryczne, a także symboliczne. Występują dwa podstawowe typy algorytmów klastrowania danych: algorytmy podziału i algorytmy hierarchiczne. Podobnie jak w przypadku algorytmów odkrywania klasyfikacji, podstawowym problemem w przypadku korzystania z algorytmów opracowanych na bazie uczenia maszynowego jest ich niska skalowalność w zakresie rozmiaru przestrzeni.



Rys. 1.15 Grupowanie dwunastu obiektów w cztery dwuwymiarowe klastry w zależności od wieku i osiąganego dochodu

Zastosowanie: segmentacja rynku klientów (telekomunikacja, ubezpieczenia), segmentacja obrazów, biologia, medycyna etc.

### 1.6.4 Odkrywanie wzorców sekwencji

Zagadnienie odkrywania wzorców sekwencji (ang. *sequential patterns*) dotyczy często powtarzających się wzorców w zachowaniach np. zakupów w sklepach przez klientów. Na poniższym rysunku (rys. 1.16) zaprezentowano listę zakupów dokonywanych przez pięciu klientów w czasie. Można zauważyć następującą prawidłowość - zakup produktu A jest poprzedzony zakupem tego samego produktu w przypadku dwóch (1 i 4) z pięciu istniejących klientów. W takim przypadku o podsekwencji (lub inaczej wzorcu sekwencji) „A poprzedza A”, można powiedzieć, iż posiada 40% wsparcie. Takie samo wsparcie posiada podsekwencja „A poprzedza C i E” (odpowiednio dla klientów: 2 i 4).

ID klienta	Data	Produkt
1	1 luty	A
1	15 luty	A
2	4 luty	A
2	16 luty	B
2	2 czerwca	C, E
3	4 stycznia	A, E
4	3 luty	A
4	3 marca	C, D, E
4	2 czerwca	A
5	5 czerwca	A

Rys. 1.16 Lista zakupów klientów w czasie

Zastosowania: analiza dostępu do stron internetowych, analiza koszyka zakupów, bezpośredni marketing, medycyna, ubezpieczenia, telekomunikacja etc.

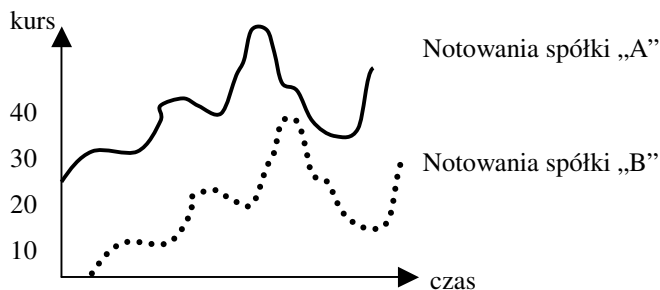
### 1.6.5 Wykrywanie zmian i odchyleń

Wykrywanie zmian i odchyleń to poszukiwanie różnic pomiędzy bieżącymi a oczekiwanymi wartościami danych (ang. *changes and deviations detection*) np. znajdowanie anomalnych zachowań klientów ubezpieczalni, posiadaczy kart kredytowych, abonentów firm telekomunikacyjnych.

W takim samym stopniu ważne jest odnajdywanie zależności i prawidłowości w bazach danych, jak i odnajdywanie odchyleń, anomalii od normalnych zachowań a także wyjątków. Odchylenie to różnica pomiędzy bieżącą a żadaną wartością. Analiza danych ma na celu zrozumienie trendów i zmian zachodzących w procesach generujących te dane.

### 1.6.6 Odkrywanie podobieństw w przebiegach czasowych

W zbiorach danych opisujących określone procesy można odnaleźć podobieństwa polegające na podobnym kształtowaniu się wartości w czasie. W ten sposób można znaleźć te elementy, które wykazują się podobnymi cechami w celu ich pogrupowania.



Rys. 1.17 Notowania spółek A i B różnią się wartościami są także wzajemnie przesunięte w czasie, odznaczają się jednak podobną charakterystyką

Zastosowania: poszukiwanie klientów o podobnej konsumpcji energii elektrycznej, identyfikacja spółek giełdowych o podobnej dynamice ruchu cen akcji, identyfikacja surowców o podobnej charakterystyce sprzedaży etc.

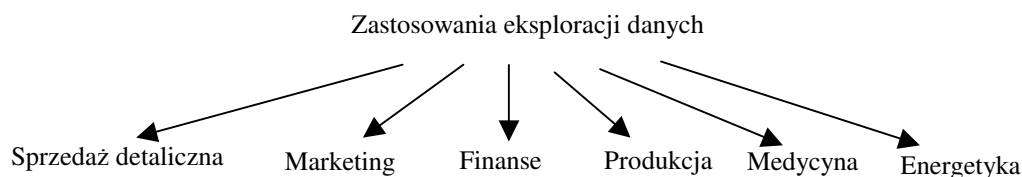
### 1.6.7 Przewidywanie przebiegów czasowych

Przewidywanie przebiegów czasowych (ang. *Time series forecasting*) odbywa się na podstawie dostępnych danych oraz modelu predykcji wartości specyficznych atrybutów w zbiorach danych. Dysponując danymi z przeszłości można odkrywać podobieństwa do bieżących sytuacji i wykorzystując np. sztuczne sieci neuronowe [Rozdział 2] przewidywać zachowania w przyszłości. Ta metodologia jest tematem niniejszej pracy i zostanie szerzej omówiona w kolejnych rozdziałach.

Zastosowanie: Predykcja rynku giełdowego (ang. *Stock market forecasting*), predykcja konsumpcji energii elektrycznej.



### 1.6.8 Zastosowania eksploracji danych – podsumowanie.



Rys. 1.18 Zastosowania eksploracji danych z podziałem na branże

Przedstawione przykłady zastosowań wykazują, że data mining nie ma na celu zastąpienia człowieka w procesach decyzyjnych. Jest narzędziem pomocnym w podejmowaniu decyzji w sytuacji, gdy posiada się zasób wiedzy, której człowiek nie jest w stanie ogarnąć.

## 1.7 Podsumowanie

Wykorzystanie narzędzi statystycznych do przygotowania danych, a następnie do prezentacji wyniku nie są trudnymi czynnościami w praktycznym wykonaniu.

Problem polega na różnorodności metod postępowania w zależności od rodzaju danych oraz specyfiki zadania. Dodatkowo wiele zadań może mieć wielorakie możliwości rozwiązania. Eksploracja danych jako algorytmy postępowania to często osobne działy wiedzy, takie jak np. sieci neuronowe, algorytmy genetyczne etc. Poznanie ich zasady działania a także zalet i wad, jakimi się charakteryzują, a co za tym idzie implementacja odpowiedniego algorytmu i technik postępowania w postawionym zadaniu gwarantują duże prawdopodobieństwo w uzyskaniu satysfakcjonującego wyniku pracy.

---

## 2 Sztuczne sieci neuronowe

„Uczenie maszynowe to urządzenie na działania którego mają wpływ doświadczeniami z przeszłości”

NILS NILSSON

W rozdziale tym zostanie omówione zagadnienie sztucznych sieci neuronowych - SSN (ang. *Artificial Neural Networks* - ANN) rozpoczynając od rysu historycznego, poprzez budowę neuronu, aż po wykorzystywaną w części praktycznej pracy wielowarstwową sieć perceptronową – MLP (ang. *Multi Layered Perceptron*), wraz z omówieniem algorytmu wstecznej propagacji błędów (ang. *Backpropagation*).

Zasada działania SSN opiera się na aproksymowaniu pewnej funkcji, uczenie odbywa się poprzez obserwowanie przykładów jej działania. Najprostszym przykładem może być SSN ucząca się funkcji XOR, jednak w ten sam sposób może ona uczyć się rozpoznawania języka na podstawie tekstu bądź sprawdzania, czy na zdjęciu rentgenowskim znajduje się obraz nowotworu. Jeżeli SSN ma być zdolna do rozwiązywania jakiegoś problemu, zagadnienie musi być zdefiniowane jako pewna funkcja ze zbiorem zmiennych wejściowych i wyjściowych, poparta przykładowymi zestawami danych wejściowych i wyjściowych (tzw. zbiorem uczącym lub treningowym). Problem taki może wystąpić, kiedy funkcja XOR jest już z założenia funkcją z dwoma zmiennymi na wejściu (o możliwych wartościach 0 i 1) i jedną na wyjściu, a przykłady zdefiniowane są jako cztery różne kombinacje wejściowe. Oczywiście bardziej skomplikowane problemy mogą być trudniejsze do zdefiniowania jako funkcje. Zmiennymi wejściowymi, w przypadku problemu wyszukiwania nowotworu na zdjęciu rentgenowskim, mogą być wartości pikseli na danym obrazie, ale mogą też to być inne informacje uzyskane ze zdjęcia. Na wyjściu może się wtedy pojawić wartość binarna lub zmiennoprzecinkowa, określająca prawdopodobieństwo wystąpienia nowotworu na zdjęciu. W przypadku znormalizowanego rozkładu prawdopodobieństwa wartość zmiennoprzecinkowa przyjmowałaby wartości od 0 do 1 włącznie. Sieci neuronowych nie trzeba programować, metody uczenia i samouczenia sieci pozwalają rozwiązać zadanie nawet wówczas, kiedy algorytm rozwiązania zadania nie jest znany. W ten sposób znajdują zastosowanie jako skuteczne narzędzia obliczeniowe tam, gdzie brak jest jednoznacznych, dokładnych rozwiązań określonych problemów. Sieci neuronowe mogą zdobywać całą swoją wiedzę wyłącznie na etapie nauki, muszą odznaczać się jednak wystarczającym stopniem złożoności.

## 2.1 Rozwój sieci neuronowych

Historia sztucznych sieci neuronowych rozpoczęła się wraz z opublikowaniem historycznej pracy autorstwa W.S.McCulloch i W.H.Pitts „A Logical Calculus of Ideas Immanent in Nervous Activity” (1943r.), w której po raz pierwszy przedstawiono matematyczny opis komórki neuronowej. Były to proste neurony, które mogły modelować funkcje logiczne takie jak OR lub AND. Dotychczasowe zainteresowanie badaczy skupiało się na opisanu mechanizmów działania mózgu lub pojedynczych komórek układu nerwowego.

Pierwszym szeroko znanym przykładem sztucznej sieci neuronowej był perceptron. Sieć tego typu jako układ częściowo elektromechaniczny, a częściowo elektroniczny została zbudowana w 1957 roku. Perceptron F.Rosenblatta używany był do rozpoznawania znaków alfanumerycznych.

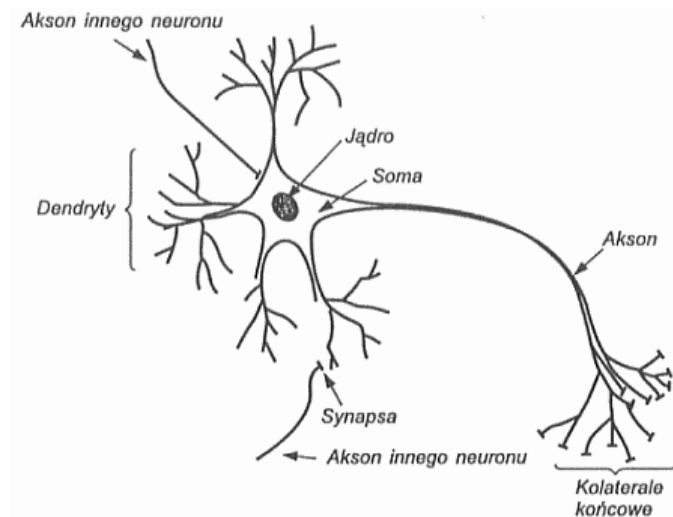
W 1960 roku B.Widrow tworzy analogowe urządzenie o nazwie ADALINE (skrót od angielskiej nazwy: *Adaptive linear element*). Po opublikowaniu pracy autorstwa M.Minsky i S.Papert w 1969 roku, która zawierała formalny dowód, że sieci jednowarstwowe (podobne do perceptronu) mają bardzo ograniczony zakres zastosowań tempo badań nad problematyką sztucznych sieci neuronowych gwałtownie osłabło. Stan taki utrzymywał się przez blisko piętnaście lat, aż do ukazania się serii publikacji, które w sposób bardzo sugestywny pokazywały, że sieci nieliniowe wolne są od ograniczeń pokazanych w powyższej pracy.

W międzyczasie powstało wiele konstrukcji sieci neuronowych m.in. P.J.Werbosa (1974 r.) zaproponował metodę wstecznej propagacji błędów, J.A.Anderson i T.Kohonen opracowali koncepcje sieci asocjacyjnych, a S.Grossberger i G.Carpenter zbudowali w oparciu o analogie biologiczne sieci ART (Adaptive Resonance Theory). Od końca lat osiemdziesiątych, na skutek nagromadzenia pozytywnych przykładów i rozwoju komputerów pozwalających na symulacje sieci neuronowych, nastąpił gwałtowny rozwój badań nad sieciami neuropodobnymi.

## 2.2 Podstawy biologiczne

Pierwowzorem sztucznej sieci neuronowej jest biologiczny układ nerwowy. Jest to złożona struktura, składająca się z komórek nerwowych (neurony) i z połączeń pomiędzy nimi (synapsy).

Liczba komórek nerwowych w mózgu człowieka wynosi około  $10^{18}$ , natomiast liczba połączeń pomiędzy nimi sięga  $10^{15}$ . Szybkość przetwarzania mózgu oceniana jest na  $10^{18}$  operacji na sekundę.

Rys. 2.1 Budowa komórki nerwowej<sup>13</sup>

Pojedyncza komórka nerwowa (rys. 2.1) składa się z jądra, wielu dendrytów (stanowiących wejścia) i *aksonu* (stanowiącego jedyne wyjście komórki nerwowej). Dendryty i aksony zakończone są *synapsami*, które służą do przekazywania informacji pomiędzy neuronami. Akson neuronu połączony jest poprzez synapsy z dendrytami innych neuronów, (ale tylko z jednym dendrytem każdego neuronu). Sygnały wewnątrz komórki nerwowej przewodzone są na drodze elektrycznej, natomiast pomiędzy komórkami na drodze chemicznej. Biologiczna sieć neuronowa działa według następującego mechanizmu. Każdy z dendrytów dostarcza do komórki sygnał o określonym poziomie; w komórce następuje kumulowanie tych sygnałów i jeśli sygnał wyjściowy osiągnie wartość progową, jest przewodzony przez synapsę do wejścia kolejnej komórki nerwowej. Występują zatem dwa stany komórki nerwowej - pobudzenie i brak aktywności.

## 2.3 Model sztucznego neuronu

Podstawowym składnikiem użytym do budowy sieci neuronowych są sztuczne neurony.

Z technicznego punktu widzenia sztuczny neuron jest elementem, którego właściwości odpowiadają wybranym właściwościom biologicznego neuronu. Sztuczny neuron nie jest więc wierną kopią neuronu biologicznego, a jedynie elementem, który stara się spełniać określone funkcje w sztucznej sieci neuronowej.

Modele sztucznych neuronów charakteryzują się występowaniem wielu wejść  $x_i$  ( $i = 1, 2, \dots, n$ ) i jednego wyjścia  $y_i$  przyjmujących wartości, odpowiadające pewnym informacjom.

<sup>13</sup> [Osowski]

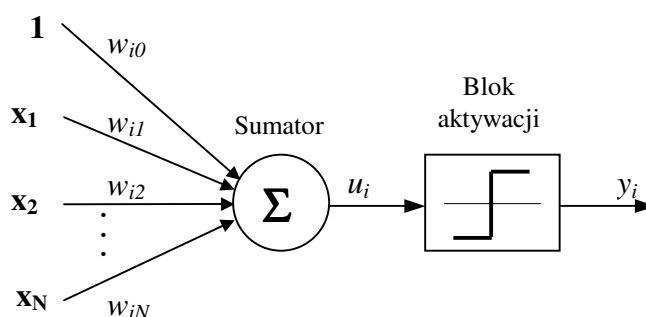
Zadaniem neuronu jest przetworzenie informacji wejściowych  $x_i$  na pewien wynik  $y$ . Z każdym wejściem neuronu związany jest parametr nazywany wagą. Sygnał podany na wejście neuronu jest przemnażany przez wagę tego wejścia, w ten sposób sygnał może zostać wzmocniony (gdy waga jest większa od 1) lub stłumiony (gdy waga przyjmuje wartości mniejsze od 1) lub może otrzymać wartość przeciwną (gdy waga posiada ujemną wartość). Sygnały wejściowe po przemnożeniu przez odpowiednie wagi są sumowane, do tej sumy w niektórych typach sieci dodaje się składnik niezależny od sygnałów wejściowych (ang. *bias*). Sygnał wyjściowy w zależności od rodzaju neuronu może być obliczany bezpośrednio z sumy przemnożonych sygnałów wejściowych oraz wag lub pośrednio z tejże sumy za pomocą pewnej nieliniowej zależności zachodzącej pomiędzy pobudzeniem a sygnałem wyjściowym (np. neuron sigmoidalny).

Każdy neuron dysponuje wewnętrzną pamięcią (reprezentowaną przez aktualne wartości wag i progów) oraz możliwościami przetwarzania sygnałów w sygnał wyjściowy.

Ze względu na fakt, że możliwości obliczeniowe pojedynczego neuronu są bardzo małe (a nieraz niewystarczające do rozwiązania problemu) neurony łączy się w większą ilość.

### 2.3.1 Model neuronu McCullocha-Pittsa

W jednym z pierwszych modeli neuronu, modelu McCullocha-Pittsa, przyjęto neuron jako jednostkę binarną. Schemat elektryczny takiego modelu przedstawiono na rys. 2.2.



Rys. 2.2 Model komórki nerwowej według McCullocha-Pittsa

Sygnały wejściowe  $x_j$  ( $j=1, 2, \dots, N$ ) sumowane są z odpowiednimi wagami  $w_{ij}$  w sumatorze, a otrzymany wynik porównuje się z progiem  $w_{i0}$ . Sygnał wyjściowy neuronu  $y_i$  wyraża się wówczas zależnością:

$$y_i = f\left(\sum_{j=1}^N w_{ij} x_j(t) + w_{i0}\right) \quad (2.1)$$

Argumentem funkcji jest sygnał sumacyjny  $u_i = \sum_{j=1}^N w_{ij}x_j(t) + w_{i0}$ . Funkcja  $f(u_i)$  jest zwana funkcją aktywacji. W modelu McCullocha jest to funkcja skokowa:

$$f(u) = \begin{cases} 1 & \text{dla } u > 0 \\ 0 & \text{dla } u \leq 0 \end{cases} \quad (2.2)$$

Współczynniki  $w_{ij}$  występujące we wzorze (2.1) reprezentują wagę połączeń synaptycznych. Dodatnia wartość współczynnika oznacza synapsę pobudzającą, ujemna – hamującą, natomiast wartość zerowa świadczy o braku połączenia pomiędzy  $i$ -tym a  $j$ -tym neuronem. W modelu McCullocha-Pittsa stan neuronu w chwili  $(t+1)$  określany jest na podstawie stanu sygnałów wejściowych neuronów w chwili poprzedniej  $t$ .

Sieci zbudowane z wykorzystaniem neuronów opisanych zależnością (2.1) nazywa się sieciami liniowymi. W literaturze sieci takie nazywa się również sieciami *Madaline*, natomiast tworzące je elementy (neurony) nazywa się *Adaline*.

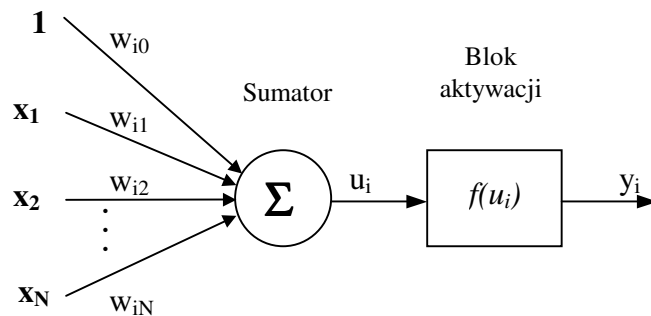
### 2.3.2 Neuron sigmoidalny

Neuron typu sigmoidalnego wykazuje budowę podobną do modelu McCullocha-Pittsa, z tą jednak różnicą, że funkcja aktywacji jest ciągła i przyjmuje postać funkcji sigmoidalnej unipolarnej<sup>14</sup>:

$$f(x) = \frac{1}{1 + e^{-\beta x}} \quad (2.3)$$

lub bipolarnej:

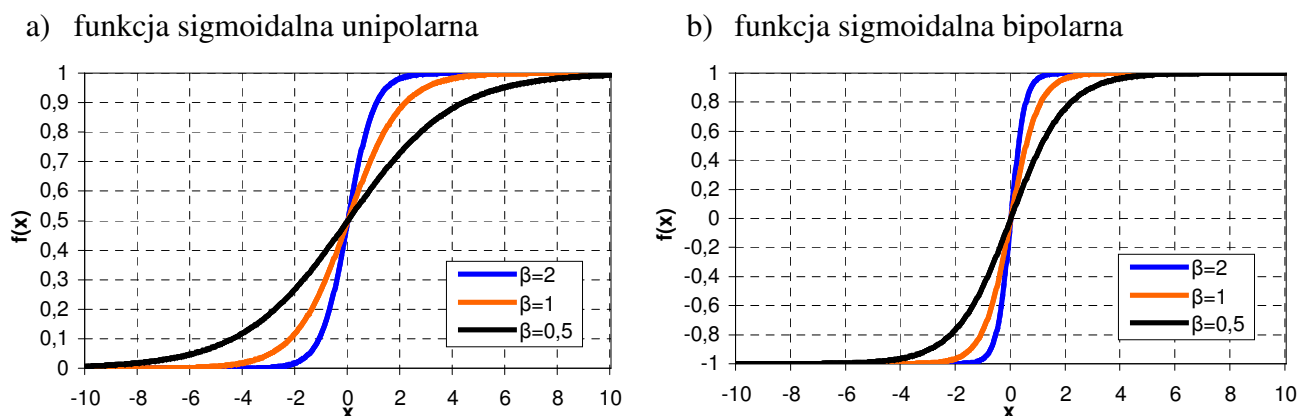
$$f(x) = \tanh(\beta x) \quad (2.4)$$



Rys. 2.3 Model neuronu sigmoidalnego

W powyższych wzorach parametr  $\beta$  jest dobierany przez użytkownika. Wpływ tego parametru na kształt funkcji aktywacji przedstawiono na rys. 2.4

<sup>14</sup> [Osowski]



Rys 2.4 Przebieg funkcji sigmoidalnej.

Istotną cechą funkcji sigmoidalnej jest jej różniczkowalność, dla funkcji unipolarnej ma ona postać:

$$\frac{df(x)}{dx} = \beta f(x)(1 - f(x)) \quad (2.5)$$

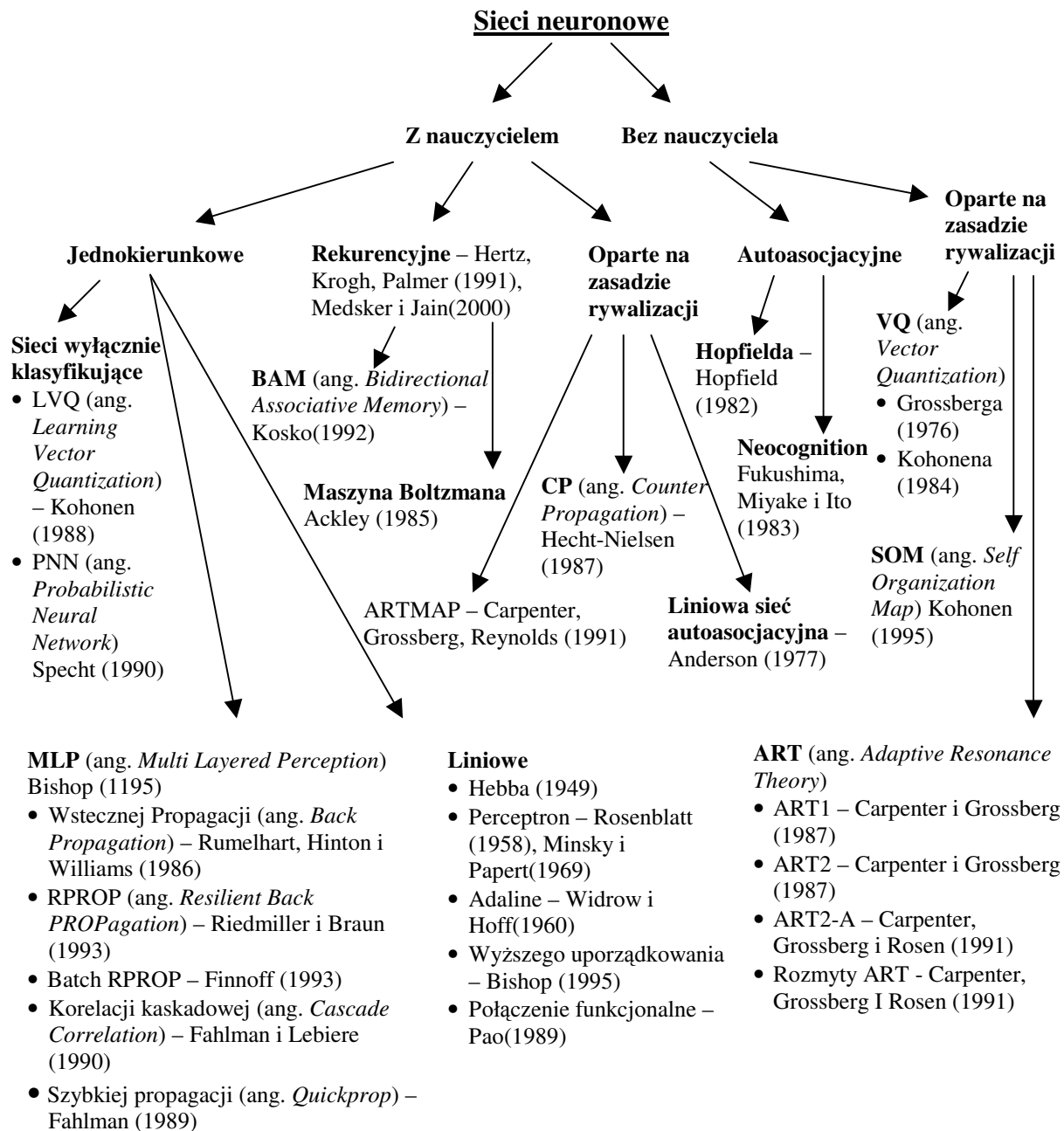
a dla funkcji bipolarnej:

$$\frac{df(x)}{dx} = \beta(1 - f^2(x)) \quad (2.6)$$

Dla obu funkcji wykres pochodnej względem zmiennej  $x$  ma kształt dzwonu, z maksimum w punkcie  $x=0$ .

## 2.4 Rodzaje sieci neuronowych

Podziału sieci neuronowych można dokonać ze względu na sposób połączenia neuronów w tejże sieci, kierunku przepływu sygnałów w sieci, a także od metody doboru wag, czyli uczenia. Istnieje wiele rodzajów sieci neuronowych, najbardziej podstawowe, obrazujące budowę i sposób działania zostały zaprezentowane na rys. 2.5



Rys 2.5 Rodzaje sieci neuronowych wraz z nazwiskami autorów i datą powstania.  
Opracowano na podstawie: <http://pawelrosczak.republika.pl/mlp/mlp.html>

## 2.5 Uczenie sieci neuronowych

Jedną z najważniejszych cech sieci neuronowych jest ich zdolność uczenia się, czyli zdolność do samodzielnego dostosowywania współczynników wagowych. Dzięki temu charakteryzują się one właściwościami sztucznej inteligencji, potrafią, bowiem samodzielnie przystosowywać się do zmieniających się warunków. Uczenie ma na celu taki dobór wag w poszczególnych neuronach, aby sieć mogła rozwiązywać stawiane przed nią problemy.



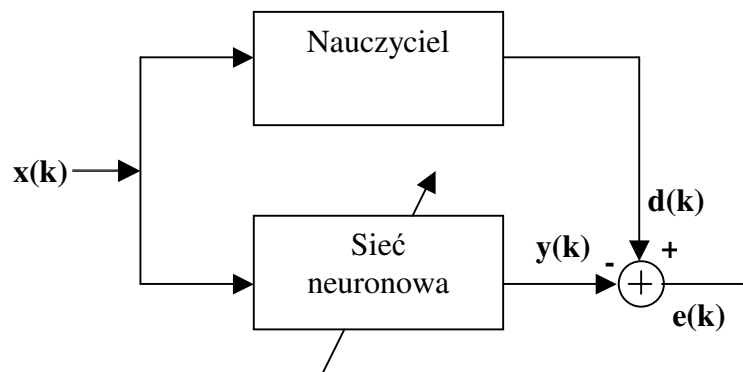
### 2.5.1 Metody uczenia

Z punktu widzenia systemu uczącego wyróżnia się dwa warianty uczenia: z nauczycielem (ang. *supervised learning*) i bez nauczyciela (ang. *unsupervised learning*).

- **Uczenie z nauczycielem**

Dysponując informacjami o oczekiwanych wartościach wyjściowych można skorzystać z uczenia pod nadzorem zewnętrznego nauczyciela.

Ogólny schemat procesu uczenia pod nadzorem może być przedstawiony w postaci systemu adaptacyjnego (rys. 2.6), w którym adaptacja wag sieci odbywa się tak, aby minimalizować błąd sieci  $e(k)$ , czyli różnicę sygnału wyjściowego  $y(k)$  i oczekiwanego sygnału wyjściowego  $d(k)$ . W kolejnych cyklach uczących sieć dobiera wagi tak, aby jej wyjścia były jak najbardziej zgodne z wzorcami uczącymi. Charakterystyczną cechą takiego sposobu uczenia jest istnienie sprzężenia zwrotnego, umożliwiającego korelację wag sieci prowadzącą do minimalizacji różnic między aktualną odpowiedzią układu wyrażoną przez wektor  $y(k)$ , a wartościami żądanymi reprezentowanymi przez wektor  $d(k)$ .



Rys. 2.6 Sieć neuronowa jako układ adaptacyjny

- **Uczenie bez nauczyciela**

Metoda różni się od metody uczenia pod nadzorem tym, że nie występuje w niej informacja o pożądanach wartościach na wyjściu, podawany na wejście jest jedynie szereg przykładowych danych. Odpowiednio skonstruowana sieć potrafi jedynie w oparciu o obserwację sygnałów wejściowych nauczyć się odwzorowań.

Występuje tu informacja czy akcja podjęta przez system odnosi pozytywne czy negatywne wyniki. Jeżeli działanie odnosi pozytywny wynik wówczas następuje wzmocnienie tendencji do właściwego zachowania się systemu w przyszłości i odwrotnie, jeżeli wynik odniesie negatywny rezultat tendencje zostaną osłabione.

Samouczenie lub też samoorganizacja bo tak również nazywane jest uczenie bez nauczyciela, jest bardzo interesujące z punktu widzenia zastosowań, ponieważ nie wymaga zewnętrznej wiedzy dostarczonej do sieci, a sieć i tak potrafi zmodyfikować swoje parametry.

### 2.5.2 Reguły uczenia

Wyróżnia się dwie podstawowe reguły uczenia: regułę perceptronu leżącą u podstaw większości algorytmów uczenia z nauczycielem oraz regułę Hebba będącą przykładem uczenia bez nauczyciela.

#### **Reguła Perceptronu**

Najpopularniejszą metodą uczenia perceptronu jest tzw. reguła perceptronu.

Początkowo wartości wag  $w_{ij}$  są inicjalizowane w sposób losowy. Na wejście podaje się wektor uczący  $x$  a następnie oblicza wartość sygnału wyjściowego  $y_i$ . W wyniku porównania aktualnej wartości  $y_i$  oraz wartości zadanej  $d_i$  dokonuje się aktualizacji wag przy wykorzystaniu korekcji wag zgodnie z równaniem:

$$\Delta w_{ij} = x(d_i - y_i) \quad (2.7)$$

Wyróżnia się trzy przypadki postępowania w zależności od wartości  $y_i$ .

- Jeżeli  $y_i = d_i$ , wówczas wagi  $w_{ij}$  pozostają bez zmian,
- Jeżeli  $y_i = 0$ , a  $d_i = 1$ , wówczas  $w_{ij}(t+1) = w_{ij}(t) + x_j$ ,  
gdzie  $t$  jest oznaczeniem cyklu poprzedniego, a  $t+1$  cyklu bieżącego,
- Jeżeli  $y_i = 1$ , a  $d_i = 0$ , wówczas  $w_{ij}(t+1) = w_{ij}(t) - x_j$ .

Po aktualizacji wag, podawany jest na wejście kolejny wektor uczący  $x$  i następuje kolejna aktualizacja wag względem kolejnej wartości oczekiwanej  $d$ . Proces powtarza się na wszystkich próbkach uczących wielokrotnie aż uzyska się minimalizację różnic między wszystkimi wartościami  $y_i$  a odpowiadającymi jej wartościami  $d_i$ .

Cechą charakterystyczną reguły perceptronowej jest wykorzystywanie w uczeniu jedynie informacji o aktualnej wartości sygnału wyjściowego neuronu i wartości żądanej.

### Reguła Hebba

To jedna z najbardziej popularnych metod samouczenia sieci neuronowych. W modelu Hebba wykorzystuje się obserwacje neurobiologiczne, polegające na tym, że waga powiązań między dwoma neuronami wzrasta przy jednoczesnym stanie pobudzenia obu neuronów, w przeciwnym wypadku maleje. Zgodnie z regułą Hebba, zmiana wagi  $w_{ij}$  połączenia pomiędzy  $i$ -tą i  $j$ -tą komórką o sygnałach wyjściowych odpowiednio:  $y_i$ ,  $y_j$  odbywa się proporcjonalnie do iloczynu sygnałów wyjściowych:

$$\Delta w_{ij}(n) = \eta y_i(n) x_j(n) \quad (2.8)$$

gdzie  $\eta$  jest stałą uczenia z przedziału (0, 1).

Reguła Hebba może być stosowana do różnego typu struktur sieci neuronowych i różnych funkcji aktywacji neuronu. Przy wielokrotnej prezentacji takiego samego wymuszenia  $x_j$  obserwuje się wykładniczy wzrost wag, czego efektem jest nasycenie neuronu. Aby uniknąć takiej niepożądanej sytuacji modyfikuje się tą regułę przez wprowadzenie współczynnika zapomnienia.

### 2.5.3 Algorytmy uczenia

Wyróżnia się wiele algorytmów uczenia sieci neuronowych. Jednym z najważniejszych jest klasyczny algorytm wstecznej propagacji błędów, który zostanie omówiony poniżej. Dodatkowo omówione zostaną dwa algorytmy będące jego modyfikacjami: *Batch Back Prop* oraz *RPROP*.

#### Klasyczny algorytm wstecznej propagacji błędów

Algorytm wstecznej propagacji (ang. *Back Propagation*) definiuje metodę doboru wag w sieci MLP przy wykorzystaniu gradientowych metod optymalizacji. Podstawę algorytmu stanowi funkcja celu, zdefiniowana jako suma kwadratów różnic między aktualnymi wartościami sygnałów wyjściowych sieci a wartościami oczekiwanymi (zadanymi) – uczenie zatem odbywa się w trybie z nauczycielem. Analizując jedynie pojedynczą próbkę uczącą ( $x$ ,  $d$ ) funkcja celu ma postać:

$$E(w) = \frac{1}{2} \sum_{k=1}^M (y_k - d_k)^2 \quad (2.9)$$

gdzie wartość  $y_k$  uzyskuje się z równania (2.1). W przypadku, gdy analizuje się więcej próbek uczących  $j$  ( $j = 1, 2, \dots, p$ ) funkcja celu stanowi sumę  $E$  po wszystkich próbkach:

$$E(w) = \frac{1}{2} \sum_{j=1}^p \sum_{k=1}^M (y_k^{(j)} - d_k^{(j)})^2 \quad (2.10)$$

Uaktualnienie wag może odbywać się po każdorazowej prezentacji próbki lub jednorazowo po prezentacji wszystkich próbek (np. algorytm *Batch BackProp*). Celem uczenia sieci jest określenie wartości wag neuronów wszystkich warstw sieci w taki sposób, aby przy zadanym wektorze wejściowym  $x$  uzyskać na wyjściu sieci wartości sygnałów wyjściowych  $y_i$  odpowiadające z określoną dokładnością wartościom zadanym  $d_i$ .

Uczenie składa się z kilku etapów. W pierwszym etapie następuje prezentacja próbki uczącej  $x$  i obliczenie wartości sygnałów poszczególnych neuronów w sieci. Obliczane są wartości sygnałów  $v_i$  warstwy ukrytej (rys 2.10) a następnie wartości  $y_i$  neuronów warstwy wyjściowej.

Do obliczenia tych wartości wykorzystuje się zależności 2.11 i 2.12.

$$v_i = f\left(\sum_{j=0}^N w_{ij}^{(1)} x_j\right) \quad (2.11)$$

w której wskaźnik  $0$  odpowiada sygnałowi i wagom polaryzacji, zakładając, że  $v_0 \equiv 1$ ,  $x_0 \equiv 1$ .

W przypadku warstwy wyjściowej  $k$  – ty neuron wytwarza sygnał wyjściowy opisany następująco:

$$y_k = f\left(\sum w_{ki}^{(2)} v_i\right) = f\left(\sum w_{ki}^{(2)} f\left(\sum w_{ij}^{(1)} x_j\right)\right) \quad (2.12)$$

Znając wartość sygnału wyjściowego  $y_i$  możliwe staje się określenie aktualnej wartości funkcji celu  $E(w)$  zgodnie z wzorem (2.9).

Drugi etap algorytmu polega na minimalizacji wartości tej funkcji. Adaptacja wektora wag dokonuje się według wzoru:

$$w(k+1) = w(k) + \Delta w \quad (2.13)$$

gdzie:

$$\Delta w = \eta p(w),$$

$\eta$  - współczynnik uczenia,

$p(w)$  – kierunek w przestrzeni wielowymiarowej.

Uczenie sieci wielowarstwowej z wykorzystaniem metod gradientowych wymaga dla wyznaczenia kierunku  $p(w)$  określenia wektora gradientu względem wszystkich warstw sieci.

W przypadku wag warstwy wyjściowej zadanie to jest wykonywane w sposób natychmiastowy, natomiast pozostałe warstwy wymagają zastosowania specjalnej strategii postępowania, którą nazywa się algorytmem propagacji wstecznej (ang. *backpropagation*), będącej fundamentalnym algorytmem procedury uczenia sieci neuronowej.

Zgodnie z nim, w każdym cyklu uczącym wyróżnia się następujące etapy uczenia<sup>15</sup>:

1. Analiza sieci neuronowej o zwykłym kierunku przepływu sygnałów przy założeniu sygnałów wejściowych sieci równych elementom aktualnego wektora  $x$ . W wyniku analizy otrzymuje się wartości sygnałów wyjściowych neuronów warstw ukrytych oraz warstwy wyjściowe a także odpowiednie pochodne  $\frac{df(u^{(1)})}{du_i^{(1)}}, \frac{df(u^{(2)})}{du_i^{(2)}}, \dots, \frac{df(u^{(m)})}{du_i^{(m)}}$  funkcji aktywacji w poszczególnych warstwach;
2. Utworzenie sieci propagacji wstecznej przez odwrócenie kierunków przepływu sygnałów, zastąpienie funkcji aktywacji przez ich pochodne, a także podanie do byłego wyjścia (obecnie wejścia) sieci wymuszenia w postaci odpowiedniej różnicy między wartością aktualną i żadaną. Dla tak utworzonej sieci należy obliczyć wartości odpowiednich różnic wstecznych;
3. Adaptacja wag (uczenie sieci) odbywa się na podstawie wyników uzyskanych w punkcie 1 i 2 dla sieci zwykłej i sieci o propagacji wstecznej według odpowiednich wzorów;
4. Omówiony proces opisany w punktach 1, 2, 3 należy powtórzyć dla wszystkich wzorców uczących, kontynuując go do chwili spełnienia warunku zatrzymania algorytmu. Działanie algorytmu kończy się w momencie, gdy norma gradientu spadnie poniżej pewnej wartości  $\varepsilon$  określającej dokładność procesu uczenia.

### Algorytm Batch BackProp

Zakłada minimalizację funkcji błędu średniokwadratowego wyznaczanego dla wszystkich obserwacji ze zbioru uczącego. Modyfikacja wag sieci neuronowej następuje w tym przypadku dopiero po zaprezentowaniu całego zbioru uczącego par wektorów wartości wejść oraz wzorców. Jest obliczana jako średnia z wartości gradientów dla poszczególnych obserwacji. Odpowiednia dla tej odmiany algorytmu funkcja celu, bazująca na metryce błędu średniokwadratowego, ma formę:

$$E(w) = \frac{1}{2} \frac{1}{T} \sum_{i=1}^T \sum_{j=1}^N (y_k^{(j)} - d_k^{(j)})^2 \quad (2.14)$$

Algorytm powinien zatem pamiętać zmiany konieczne do zastosowania po zakończeniu epoki (dla dużej liczby wzorców może to spowodować duże zapotrzebowanie pamięci). Jednak w szczególnych warunkach osiąga się zbieżność wcześniej niż w przypadku klasycznego algorytmu.

<sup>15</sup> [Osowski]

**Algorytm RPROP (ang. *Resilient BackProp*)**

W algorytmie do adaptacji wag uwzględnia się jedynie znak składowej gradientu, ignorując jej wartość.

$$\Delta w_{ij}(k) = -\eta_{ij}(k) \operatorname{sgn}\left(\frac{\partial E(w(k))}{\partial w_{ij}}\right) \quad (2.15)$$

Współczynnik uczenia jest dobierany w każdym cyklu dla każdej wagi  $w_{ij}$  indywidualnie na podstawie zmian wartości gradientu.

$$\eta_{ij}(k) = \begin{cases} \min(a\eta_{ij}(k-1), \eta_{\max}) & \text{dla } S_{ij}(k)S_{ij}(k-1) > 0 \\ \max(b\eta_{ij}(k-1), \eta_{\min}) & \text{dla } S_{ij}(k)S_{ij}(k-1) < 0 \\ \eta_{ij}(k-1) & \text{w pozostałych przypadkach} \end{cases} \quad (2.16)$$

przy czym  $S_{ij}(k) = \frac{\partial E(w(k))}{\partial w_{ij}}$ ,  $a$  i  $b$  są stałymi:  $a = 1.2$ ,  $b = 0.5$ , natomiast  $\eta_{\min}$  i  $\eta_{\max}$  oznaczają

odpowiednio minimalną i maksymalną wartość współczynnika uczenia. Współczynniki te w algorytmie RPROP wynoszą odpowiednio  $10^{-6}$  oraz 50, a funkcja  $\operatorname{sgn}$  oznacza znak argumentu. Algorytm RPROP, umożliwia znaczne przyspieszenie procesu uczenia w tych obszarach, gdzie nachylenie funkcji celu jest niewielkie. Strategia doboru wag zakłada ciągły wzrost współczynnika uczenia, jeśli w dwóch kolejnych krokach znak gradientu jest taki sam, natomiast jego redukcję, gdy ten znak jest różny.

**2.5.4 Dobór parametrów uczenia**

Metoda gradientowa zastosowana w uczeniu neuronu zapewnia osiągnięcie minimum lokalnego, przy czym wyjście ze strefy przyciągania określonego minimum lokalnego nie jest możliwe przy zastosowaniu prostego algorytmu największego spadku. Rozwiązaniem jest stosowanie uczenia z tzw. *momentem*, w którym proces aktualizacji wag uwzględnia nie tylko informację o gradiencie funkcji, ale również aktualny trend zmian wag. Sposób tego uczenia można zapisać jako:

$$\Delta w_{ij}(t+1) = -\eta \delta_i x_j + \alpha \Delta w_{ij}(t) \quad (2.17)$$

Gdzie składnik pierwszy odpowiada metodzie największego spadku, natomiast składnik drugi, zwany składnikiem momentu, uwzględnia ostatnią zmianę wag i jest niezależny od aktualnej wartości gradientu. Wartość współczynnika uczenia  $\eta$  może być stała lub w sposób adaptacyjny dobierana w trakcie uczenia z zakresu  $[0,1]$ . Współczynnik momentu  $\alpha$  przybiera również wartość z tego samego zakresu co współczynnik uczenia  $\eta$ . Znaczenie momentu

wzrasta w pobliżu minimum lokalnego, w tym przypadku możliwe są zmiany wag prowadzące do pokonania bariery ograniczającej minimum lokalne.

Dobór współczynnika uczenia  $\eta$  ma bardzo duży wpływ na zbieżność algorytmu optymalizacyjnego do minimum funkcji celu<sup>16</sup>. Im bardziej współczynnik uczenia odbiega od wartości, przy której  $E(w)$  osiąga minimum na danym kierunku, tym większa ilość iteracji potrzebna do wyznaczenia optymalnego rozwiązania.

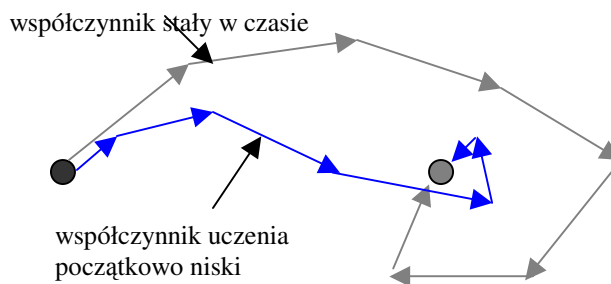
Zbyt mała wartość  $\eta$  wydłuża czas potrzebny na wyznaczenie minimum funkcji, ponieważ potrzeba większej ilości iteracji podczas uczenia. Zbyt duża wartość  $\eta_k$  powoduje ominięcie punktu optymalnego, co wiąże się z koniecznością dodatkowych iteracji w celu powrotu, co wydłuża czas uczenia.

Najprostsza metoda doboru współczynnika polega na dobieraniu stałej wartości podczas uczenia dla każdej z warstw. Jest to sposób mało efektywny, ponieważ nie uzależnia wartości współczynnika od aktualnego wektora gradientu, a co za tym idzie również kierunku w danej iteracji. Może zaistnieć sytuacja w której błędy będą wzrastać a współczynnik pozostanie niezmienny. Podejście to jest jednak często stosowane ze względu na prostotę implementacji.

Optymalne wartości współczynników uczenia i momentu mogą być różne dla kolejnych iteracji, a także różne dla każdej z wag danej iteracji. Dlatego kolejna ze strategii zakłada stosowanie większych wartości współczynnika uczenia  $\eta$  i współczynnika momentu  $\alpha$  na początku uczenia i zmniejszanie ich w miarę zbliżania się funkcji błędu do minimum. Inna, bardziej złożona strategia, przyjmuje na początku procesu uczenia małe wartości współczynnika  $\eta$  (ewentualnie również współczynnika  $\alpha$ ). Jest to uzasadnione tym, że w pierwszych krokach procesu uczenia następuje wstępna preorientacja neuronów. Wagi, początkowo ustawione losowo i z reguły bliskie zera, zostają spolaryzowane w kierunku hamujących lub pobudzających połączeń. Mniejsze tempo uczenia na tym etapie procesu może ułatwić korektę ewentualnych błędów, umożliwiając ustalenie właściwego kierunku zmian w przestrzeni wag. Gdy zostanie osiągnięta pożądana tendencja zmiany wag zwiększa się współczynnik uczenia do pewnej ustalonej wartości zwiększając tym samym tempo procesu. Końcowa faza nauczania wymaga ponownego ograniczenia wartości  $\eta$ , umożliwiającego precyzyjne osiągnięcie minimum błędu.

---

<sup>16</sup> [Tadeusiewicz]



Rys. 2.8 Różnice w uczeniu dla współczynnika stałego w czasie oraz dla dynamicznie modyfikowanego

Innym rozwiązaniem jest uzależnienie współczynnika uczenia od wartości funkcji błędu średniokwadratowego [2.5.6]. Jeśli w kolejnych iteracjach błąd średniokwadratowy maleje, można zdecydować się na zwiększenie współczynnika uczenia. Wzrost funkcji miary błędu oznacza, że proces zmierza w niewłaściwym kierunku i należy zmniejszyć  $\eta$ . Badania dowodzą, że dobrze jest zwiększać współczynnik uczenia o wartość stałą, a zmniejszać go geometrycznie:

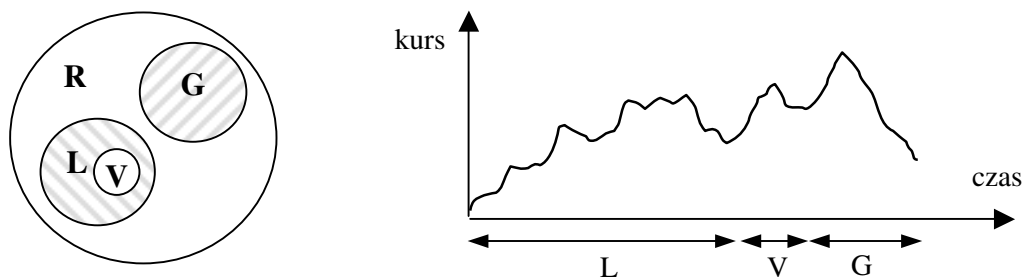
$$\Delta\mu(n) = \begin{cases} +a & \text{gdy } \Delta Q < 0 \\ -b \cdot \mu(n-1) & \text{gdy } \Delta Q > 0 \\ 0 & \text{gdy } \Delta Q = 0 \end{cases} \quad (2.18)$$

gdzie  $\Delta Q$  jest przyrostem funkcji miary błędu, natomiast  $a$  i  $b$  to odpowiednie stałe dodatnie.

W przypadku dokonania błędnego kroku można również nie przeprowadzać modyfikacji wag, a ponadto aż do osiągnięcia właściwego kierunku przebiegu procesu uczenia przyjąć  $\alpha = 0$ .

### 2.5.5 Zdolność generalizacji sieci

Sieć neuronowa posiada zdolność do generalizacji, jeśli jest w stanie prawidłowo zaklasyfikować dane należące do zbioru  $R$  po nauczaniu jej za pomocą danych ze zbioru  $L$  (dane uczące) i zweryfikowaniu jej działania za pomocą danych ze zbioru  $V$  (dane sprawdzające). Ocenę jakości wytrenowania sieci wyznacza się za pomocą zbioru  $G$  (dane testujące).



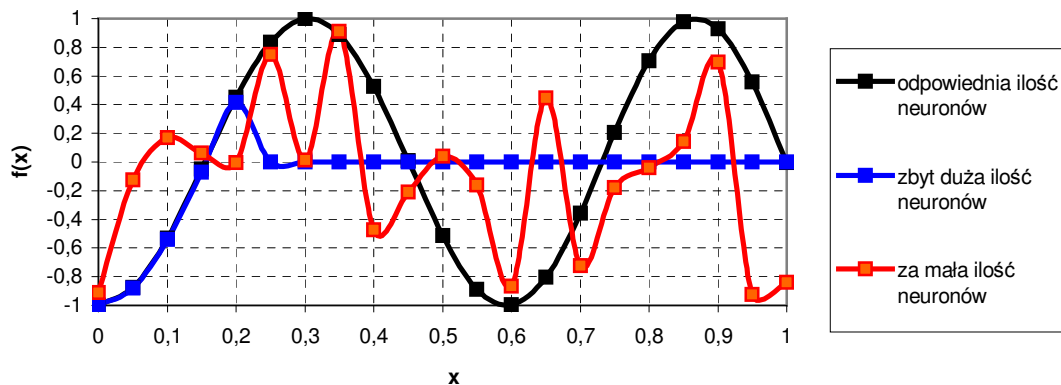
Rys. 2.9 Reprezentacja zbioru uczącego  $L$ , zbioru testującego  $G$  i zbioru sprawdzającego  $V$  spełniających pewną regułę  $R$ .



Odpowiedni dobór wag w procesie uczenia powoduje takie ukształtowanie jej własności, aby najlepiej odtwarzała ciąg zadanych par uczących  $(x_i, d_i)$ . Występuje tu ścisły związek liczby wag sieci z liczbą wzorców uczących. Żeby sieć miała dobre właściwości generalizujące, musi być uczona na zbiorze nadmiarowym gdyż wtedy wagi mogą dostosować się do statystycznej reprezentacji, a nie tylko do pojedynczych danych.

Z uwagi na błąd generalizacji ważny jest stosunek liczby próbek uczących do liczby wag sieci. Mała ilość próbek przy ustalonej liczbie wag będzie wykazywać dobre dopasowanie sieci do próbek uczących, ale złą generalizację. W zależności od ilości neuronów w warstwie ukrytej spotyka się następujące przypadki zaprezentowane kolejno na rys. 2.10:

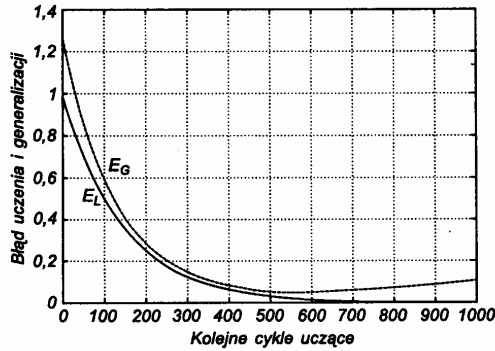
- przeuczenie (zbyt duża ilość neuronów) :
- odpowiednia ilość neuronów:
- za mała ilość neuronów:



Rys. 2.10 Ilustracja zdolności uogólniania sieci w zależności od ilości neuronów

Pierwszy przykład przedstawia sieć, która pomimo małego błędu uczenia dokonała niepoprawnej aproksymacji, czego powodem była zbyt duża ilość wag względem podawanych na wejście wzorców. Zmniejszenie ilości neuronów w warstwie ukrytej spowodowało, że sieć już poprawnie dokonywała aproksymacji, natomiast dalsze próby zmniejszenia ilości neuronów spowodowały niezdolność sieci do odwzorowania danych uczących. Jak widać, z punktu widzenia budowy sieci neuronowych prawidłowa ilość neuronów jest bardzo ważnym parametrem, wbrew pozorom jakość sieci nie zależy od czasu treningu, źle zaprojektowana sieć nigdy nie nauczy się odwzorowania wzorców. W pracy do optymalizacji topologii sieci użyto algorytmów genetycznych [Rozdział 3]. Przebieg błędu uczenia maleje monotonicznie z liczbą epok, z kolei błąd generalizacji charakteryzuje się tym, że maleje razem z błędem uczenia, ale tylko do pewnego momentu. Zbyt długie uczenie może doprowadzić do przeuczenia sieci, czyli do zbyt drobiazgowego dopasowania wag do nieistotnych szczegółów danych uczących. Taka

sytuacja ma miejsce w przypadku nadmiarowości neuronów w warstwie ukrytej względem potrzebnej ilości.



Rys. 2.11 Wpływ długości uczenia na błąd uczenia  $E_L$  oraz błąd generalizacji  $E_G$ <sup>17</sup>

W celu uniknięcia przeuczenia ze zbioru uczącego wydziela się część danych sprawdzających (zbiór  $V$ ), które służą do okresowego sprawdzania zdolności generalizacyjnych w trakcie uczenia. Decyzje o zaprzestaniu uczenia podejmuje się np. w przypadku gdy błąd generalizacji zacznie się powiększać.

### 2.5.6 Pomiar jakości działania sieci neuronowych

Pomiar jakości działania sieci neuronowej jest konieczny zarówno w procesie uczenia, jak i w ocenie pracy już nauczonej sieci. Większość miar polega na obliczaniu błędu pomiędzy wartościami wzorcowymi podanymi na wejście sieci, a wygenerowanymi wartościami na jej wyjściu. Najczęściej wykorzystuje się klasyczne i powszechnie używane statystyczne miary błędów, do których należą:

- RMSE (ang. Root Mean Square Error) – pierwiastek błędu średniokwadratowego:

Niech:  $y$  oznacza wartość wzorcową,  $p$  wartość otrzymaną na wyjściu sieci wówczas:

$$RMSE_j = \sqrt{\frac{1}{T} \sum_{i=1}^T (y_{i,j} - p_{i,j})^2}, \quad RMSE_j > 0 \quad (2.19)$$

- RMSPE (ang. Root Mean Square Percentage Error) – pierwiastek procentowego błędu średniokwadratowego:

$$RMSPE_j = \sqrt{\frac{1}{T} \sum_{i=1}^T \left( \frac{y_{i,j} - p_{i,j}}{y_{i,j}} \right)^2} * 100[\%], \quad RMSPE_j > 0 \quad (2.20)$$

Miary związane z błędem średniokwadratowym są najbardziej użyteczne w procesie uczenia sieci, ze względu na to, że właśnie funkcja oparta na błędzie średniokwadratowym, będącym

<sup>17</sup> [Osowski]

podstawą zaprezentowanych miar, jest najczęściej minimalizowana w algorytmie wstecznej propagacji błędu.

- ME (ang. *Mean Error*) – błąd średni:

$$ME_j = \frac{1}{T} \sum_{i=1}^T (y_{i,j} - p_{i,j}), ME_j \in R \quad (2.21)$$

- MPE (ang. *Mean Percentage Error*) – średni błąd procentowy:

$$MPE_j = \frac{1}{T} \sum_{i=1}^T \frac{y_{i,j} - p_{i,j}}{y_{i,j}} * 100[\%], MPE_j \in R \quad (2.22)$$

- MAE (ang. *Mean Absolute Error*) – średni błąd bezwzględny

$$MAE_j = \frac{1}{T} \sum_{i=1}^T |y_{i,j} - p_{i,j}|, MAE_j > 0 \quad (2.23)$$

- MAPE (ang. *Mean Absolute Percentage Error*) – średni absolutny błąd procentowy

$$MAPE_j = \frac{1}{T} \sum_{i=1}^T \left| \frac{y_{i,j} - p_{i,j}}{y_{i,j}} \right| * 100[\%], MAPE_j > 0 \quad (2.24)$$

Porównanie błędu średniego (ME, MPE) oraz średniego absolutnego (MAE, MAPE) pozwala określić charakter rozbieżności pomiędzy zależnościami zawartymi w danych treningowych, a wartością na wyjściu sieci, uwarunkowaną funkcją opisaną przez współczynniki wagowe sieci. Jeżeli wartości średnie oraz absolutne błędu są znaczne i bardzo do siebie zbliżone pod względem wartości absolutnej, wskazuje to, że wartości generowane przez sieć są systematycznie większe lub mniejsze od wzorców. Powodem jest zbyt słabe dopasowanie sieci do przybliżanych zależności. W przeciwnym przypadku, gdy wartości błędu średniego są bliskie zeru, a błędu absolutnego wyraźnie wyższe, obserwuje się zjawisko różnokierunkowego odchyłania się wartości wyjść sieci w stosunku do wartości wzorców. Opisana właściwość jest następstwem tendencji do znoszenia się błędów o znakach przeciwnych w miarach średnich i może np. wskazywać na zbyt uproszczoną strukturę sieci, która nie jest w stanie odwzorować bardzo złożonych zależności zawartych w danych ze zbioru treningowego.

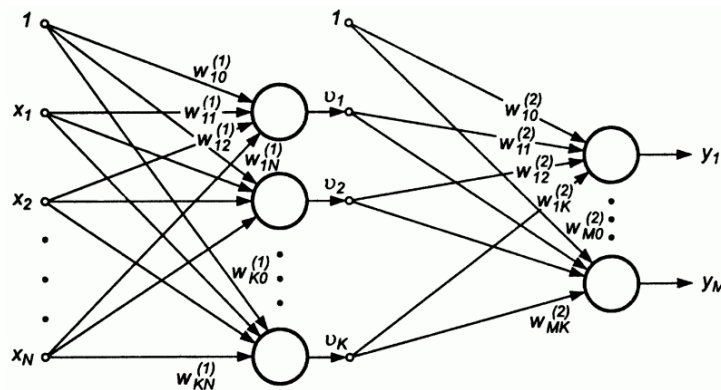
## 2.6 Sieci jednokierunkowe typu sigmoidalnego

Obok sposobu uczenia (z nauczycielem lub bez) sieci neuronowych, kolejną klasyfikacją jest podział ze względu na sposób połączenia neuronów. Mogą to być np. sieci jednokierunkowe, rekurencyjne czy też autoasocjacyjne etc. Najczęściej stosowaną siecią jest sieć jednokierunkowa, wielowarstwowa o neuronach typu sigmoidalnego, zwana także perceptronem wielowarstwowym. W sieciach tych przepływ sygnałów odbywa się tylko w

jednym kierunku od wejścia do wyjścia. Uczenie perceptronu odbywa się zazwyczaj z nauczycielem, na podstawie podawanych par uczących ( $x$  – wektor wejściowy,  $d$  – oczekiwany wektor na wyjściu). W sytuacji gdy  $x=d$  sieć nosi nazwę autoasocjacyjnej [2.7], w przeciwnym przypadku sieć nazywana jest heteroasocjacyjną.

Przykładem sieci jednokierunkowej wielowarstwowej jest *perceptron wielowarstwowy*.

Sieć wielowarstwową tworzą neurony ułożone w wielu warstwach. Oprócz warstwy wejściowej i wyjściowej istnieje, co najmniej jedna, warstwa ukryta.



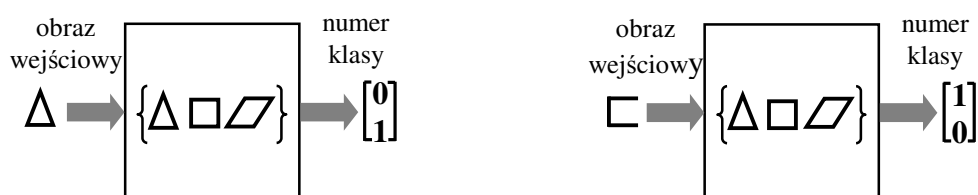
Rys. 2.12 Ogólny schemat sieci neuronowej sigmoidalnej dwuwarstwowej.<sup>18</sup>

Sygnały wejściowe podawane są na pierwszą warstwę ukrytą neuronów, a te z kolei stanowią sygnały źródłowe dla kolejnej warstwy. W sieci tej występują połączenia pełne między warstwami. Neurony warstw ukrytych stanowią bardzo istotny element sieci, umożliwiając uwzględnienie związków między sygnałami, wynikającymi z zależności statystycznych wyższego rzędu. Uczenie perceptronu wielowarstwowego odbywa się zwykle z nauczycielem według algorytmów przedstawionych w [2.5.3]. Sieci jednokierunkowe wielowarstwowe często wykorzystują nieliniową funkcję aktywacji typu sigmoidalnego i stanowią naturalne uogólnienie perceptronu.

## 2.7 Zadania sieci neuronowych

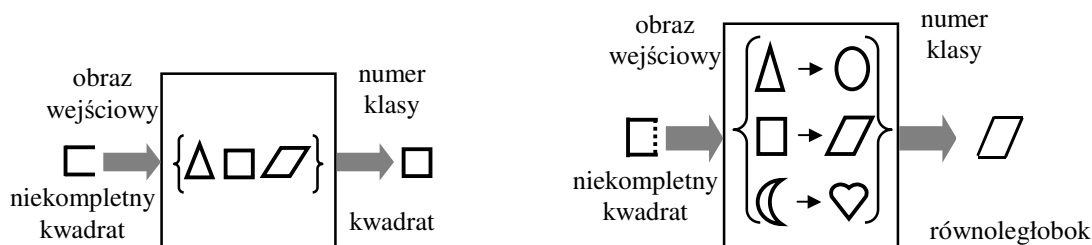
Dobrze nauczona sieć wykazuje się możliwościami klasyfikacji (zaszeregowania danych wejściowych do danych, na których opierało się uczenie) i rozpoznawania wzorców (podobnie jak w klasyfikacji z tą jednak różnicą, że obraz wejściowy wykazuje wyłącznie cechy podobieństwa a nie identyczności).

<sup>18</sup> [Osowski]



Rys. 2.13 Klasyfikacja i rozpoznawanie.

Sieć charakteryzuje się również możliwościami autoasocjacji (po otrzymaniu nowego wzorca reakcją sieci jest wytworzenie zapamiętanego wzorca najbardziej podobnego do wzorca na wejściu) oraz heteroasocjacji (sieć zapamiętuje i kojarzy pary obrazów tak, że po otrzymaniu na wejście nawet zniekształconego obrazu może on wywołać właściwą heteroasocjację na wyjściu, czyli drugi element pary dla rozpoznanego wzorca).



Rys. 2.14 Autoasocjacja i heteroasocjacja

## 2.8 Zastosowanie

Sieci neuronowe znajdują zastosowanie w bardzo wielu dziedzinach życia i techniki. Klasycznym wykorzystaniem są szeroko pojęte układy rozpoznające. Sieci neuronowe są wykorzystywane do rozpoznawania i identyfikacji obrazów radarowych, rozpoznawania (w sensie klasyfikacji) nawet tych o niepełnej bądź zafałszowanej informacji, do zadań optymalizacyjno - decyzyjnych, do szybkiego przeszukiwania dużych baz danych. Inne możliwości wykorzystania to systemy eksperckie, tworzenie analiz finansowych, a także zadania sterowania. Oprócz tego można znaleźć wiele zastosowań sieci m.in. w: diagnostyce układów elektronicznych, prognozie giełdowej, prognozowaniu sprzedaży, analizie badań medycznych, interpretacji wyników badań biologicznych, analizie problemów związanych z produkcją, sterowaniu procesami produkcyjnymi, optymalizacji różnego rodzaju działalności, tworzeniu prognozy pogody, systemach rezerwacji biletów lotniczych etc..

---

## 3 Algorytmy genetyczne

*„Zasadę, na mocy której każda zmiana,  
jeśli jest korzystna, zostaje zachowana,  
nazwałem «doborem naturalnym»,  
by wskazać na jej stosunek do  
doboru przeprowadzonego przez człowieka.”*

CHARLES DARWIN

Teoria ewolucji według Charlesa Darwina ogłoszona w 1859 roku stwierdza, że złożoność form życia na Ziemi jest skutkiem ewolucji i doboru naturalnego. Algorytmy genetyczne powstały w efekcie obserwacji naturalnych procesów zachodzących w przyrodzie. Do algorytmów genetycznych przeniesiono zjawiska ewolucji i naturalnej selekcji, które zachodzą wśród populacji żywych osobników. W 1975 roku, John Holland z Uniwersytetu Michigan, wzorując się na mechanizmach ewolucji, a także biorąc pod uwagę fakt, że ewolucja zachodzi na chromosomach, a nie na żywych istotach, opracował pierwszy algorytm genetyczny.

Algorytm genetyczny jest algorytmem poszukiwania, opartym na mechanizmach doboru naturalnego oraz dziedziczności. Łącząc w sobie ewolucyjną zasadę przeżycia najlepiej przystosowanych osobników z systematyczną, aczkolwiek losową wymianą informacji, tworzą metodę poszukiwania charakteryzującą się dążeniem do osiągnięcia najlepszego rozwiązania.

W każdym pokoleniu powstaje nowy zespół sztucznych organizmów (ciągów bitowych), utworzonych z połączenia fragmentów najlepiej przystosowanych przedstawicieli poprzedniego pokolenia. Sporadycznie dochodzić może do dołączania nowych osobników.

Pomimo wykorzystywania wyników losowania, algorytmy genetyczne nie przypominają błędzenia przypadkowego. Wykorzystują one efektywnie dotychczasowe doświadczenia do określenia nowego obszaru poszukiwań o spodziewanej podwyższonej wydajności. W ten sposób, poprzez odnalezienie rozwiązania o najwyższej jakości, algorytmy genetyczne znajdują zastosowanie w optymalizacji.

### 3.1 Nazewnictwo

W algorytmach genetycznych wiele pojęć zostało przejętych z genetyki, dlatego niezbędne jest krótkie omówienie podstawowych pojęć używanych w dalszej części rozdziału.

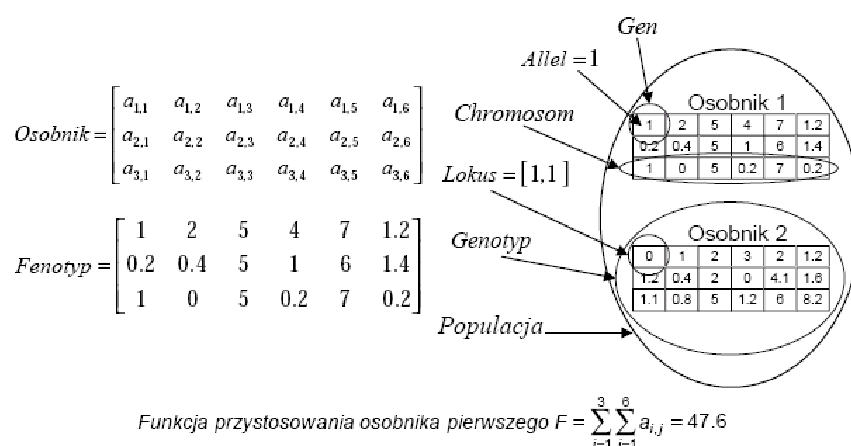
*Populacja* to zbiór osobników (organizmów) o określonej *liczebności*<sup>19</sup>. Odpowiada ona zbiorowi elementów poszukiwań określonych przez chromosomy osobników tworzących populację.

Ze względu na to, że algorytm genetyczny ma charakter iteracyjny, w każdym kolejnym kroku zwanym *generacją* tworzone jest nowe *pokolenie* osobników.

Algorytm genetyczny przetwarza populację osobników, z których każdy jest propozycją rozwiązania postawionego problemu. Działa on w środowisku, które można zdefiniować na podstawie problemu rozwiązywanego przez algorytm. W środowisku każdemu osobnikowi jest przyporządkowana wartość określająca jakość reprezentowanego przez niego rozwiązania. Wartość ta jest nazwana *przystosowaniem osobnika*. Każdy osobnik jest wyposażony w informację stanowiącą jego *genotyp*, na podstawie którego tworzony jest *fenotyp*, czyli zestaw cech, który podlega ocenie środowiska. Właśnie ta ocena jest przystosowaniem osobnika opisywanym wcześniej. Proces tworzenia fenotypu z genotypu nazywa się *dekodowaniem*. Można powiedzieć, że fenotyp należy do przestrzeni rozwiązań problemu, natomiast genotyp do przestrzeni kodów. Sposób oceniania fenotypu opisuje się za pomocą funkcji przystosowania, która opisuje środowisko. Funkcja ta może być stacjonarna, zmienna w czasie bądź podlegać randomizacji.

Genotyp to zespół chromosomów danego osobnika. Natomiast każdy chromosom składa się z *genów*. Genem nazywa się cechę, znak, detektor, który może występować w pewnej liczbie odmian zwanych *allelami*.

*Locus* - inaczej pozycja, określa miejsce położenia danego genu w łańcuchu, czyli chromosomie.



Rys. 3.1 Graficzna reprezentacja podstawowych pojęć genetyki komputerowej<sup>20</sup>

<sup>19</sup> [Goldberg]

<sup>20</sup> [Murawski]

## 3.2 Teoria doboru naturalnego

Algorytm genetyczny opiera się na teorii doboru naturalnego Darwina, który swoją teorię oparł na następujących prawach:<sup>21</sup>

- zmienność i dziedziczenie - istoty żywe rozmnażają się; potomstwo jest podobne do rodziców, choć nie identyczne. Potomstwo charakteryzuje duża zmienność dziedziczna i nie dziedziczna, jedynie zmiany dziedziczne mają znaczenie w ewolucji. Osobniki, które przeżyją przekazują korzystne cechy swemu potomstwu;
- nadmiar reprodukcyjny - organizmy wydają zwykle znacznie więcej potomstwa niż może się wyżywić, utrzymać przy życiu i wydać następne pokolenie potomstwa;
- walka o byt - ponieważ rodzi się więcej osobników niż może przeżyć, istnieje współzawodnictwo o pożywienie, przestrzeń życiową itp. (jawne lub niejawne). Walka o byt może się odbywać bezpośrednio między dwoma różnymi gatunkami w układzie ofiara - drapieżnik lub pośrednio, w obrębie jednego gatunku w wyniku konkurencji o tę samą niszę ekologiczną;
- konsekwencja: prawo doboru naturalnego - w walce o byt przeżywają osobniki najlepiej przystosowane, formy pośrednie wymierają, co prowadzi do coraz większej rozbieżności cech w następnych pokoleniach i powstania z czasem form bardzo różniących się od form wyjściowych i powstawania nowych gatunków;
- konsekwencja: zróżnicowana propagacja cech = ewolucja.

## 3.3 Optymalizacja, czyli poszukiwanie najlepszego rozwiązania

Optymalizacja jest jednym z podstawowych zastosowań algorytmów genetycznych. Polega ona na znalezieniu potencjalnego ekstremum w ściśle określonym zbiorze. Może nim być przykładowo wierzchołek szczytu górskiego. Stopień trudności polega na tym, czy ów szczyt jest jedynym wzniesieniem w swoim otoczeniu (ekstremum globalne), czy też takich wzniesień może być więcej (ekstremum lokalne). Jeżeli wierzchołek nie jest jedyny w swoim otoczeniu wówczas dla algorytmu poszukiwań zadanie komplikuje się.

Wyróżnia się następujące metody poszukiwań:<sup>22</sup>

- metody analityczne,
- metody enumeracyjne,
- metody losowego poszukiwania rozwiązań.

---

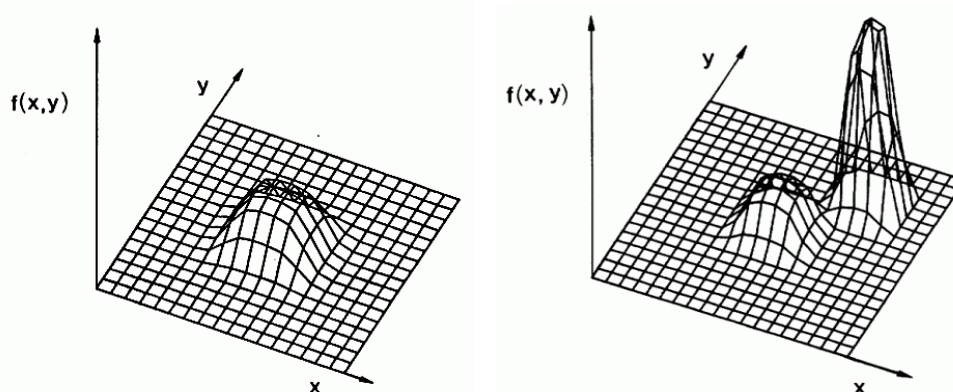
<sup>21</sup> [Grygiel]

<sup>22</sup> [Goldberg]



Metody analityczne i enumeracyjne należą do konwencjonalnych metod optymalizacyjnych. Charakteryzują się małą odpornością na znalezienie globalnego ekstremum, nie znaczy to jednak, że są nieużyteczne. Obie metody a także ich warianty i kombinacje były i są używane z powodzeniem w wielu zastosowaniach. Jednak wobec coraz bardziej złożonych problemów, stają się potrzebne nowe metody. Techniki analityczne dają dobre wyniki w wąskiej klasie problemów, ale szybko przestają być efektywne poza nią. Można je uznać za techniki wyspecjalizowane. Technika enumeracyjna (przeglądowa) charakteryzuje się nieefektywnością w całym spektrum problemowym (konieczność analizy całego zbioru argumentów).

W metodach analitycznych szukanie optimum ma zasięg lokalny, odbywa się na zasadzie szukania optimum w sąsiedztwie danego punktu. Przy funkcji jednomodalnej (rys 3.2) metoda analityczna zwróci poprawny rezultat, lecz przy funkcji bardziej skomplikowanej jak np. funkcja wielomodalna (rys 3.2) metoda analityczna ma niewielkie szanse znaleźć optimum globalne.



Rys. 3.2 Wykres funkcji jednomodalnej i wielomodalnej<sup>23</sup>

W metodach analitycznych wyróżnia się metodę pośrednią i bezpośrednią. Z metodą pośrednią wiąże się rozwiązywanie układu równań, co przy małym obszarze poszukiwań i płaskiej funkcji może dawać dobre efekty. Wynikiem korzystania z tej metody jest znalezienie przede wszystkim ekstremum lokalnego. W metodzie analitycznej bezpośredniej niejako „skacze się” po wykresie funkcji celu w kierunku wyznaczonym przez lokalny gradient. Realizuje się w ten sposób „wspinaczkę górską” wspinając się po najbardziej stromej z możliwych dróg. Ta metoda doskonale pozwala znajdować jedynie ekstrema lokalne. Obie metody mają lokalny charakter poszukiwań, szukają optymalnego rozwiązania w sąsiedztwie konkretnego punktu. Metody enumeratywne są to metody oparte na przeszukiwaniu całej przestrzeni rozwiązań i obliczanie dla każdego punktu wartości funkcji celu. Niestety jest to metoda wysoce nieefektywna.

<sup>23</sup> [Goldberg]

Zdając sobie sprawę z ograniczeń metod analitycznych i enumeracyjnych zaczęto interesować się algorytmami poszukiwania losowego. Algorytm genetyczny jest przykładem procedury używającej wyboru losowego jako przewodnika w prowadzeniu wysoce ukierunkowanego poszukiwania w zakodowanej przestrzeni rozwiązań. Wydaje się być niecelowym stosowanie go tam gdzie funkcja posiada algorytmiczny obraz. We wszystkich pozostałych przypadkach gdy nie zna się wzoru funkcji (np. wynik jest wyjściem z tzw. „czarnej skrzynki”) użycie AG może przynieść wymierne korzyści.

### Właściwości Algorytmów genetycznych

Zasadnicze cechy algorytmów genetycznych odróżniające je od tradycyjnych metod poszukiwawczych to:

- Przetwarzanie zakodowanej postaci parametrów zadania zamiast przetwarzania bezpośredniego. Informacja może być zapisana m.in. w kodzie binarnym;
- Poszukiwania rozpoczynają się od populacji, której elementy zostały dołączone w sposób losowy, metody tradycyjne poszukiwania rozpoczynają od najczęściej jednego punktu początkowego. Oznacza to, że AG pracują w tym czasie na szerszej bazie punktów populacji ciągów kodowych tym samym zbieżność do lokalnego ekstremum jest eliminowana (odporność na lokalne ekstremum);
- AG korzysta z funkcji celu a nie z pomocniczych informacji. Algorytm wymaga postawienia jasno sprecyzowanego zadania, którego oceną się zajmuje;
- Losowe reguły wyboru zamiast deterministycznych. Losowanie jednak nie powoduje poszukiwań na „oślep”, wyznacza natomiast kierunek poszukiwań ku obszarom, w których należy się spodziewać polepszenia wyników.

## 3.4 Metody kodowania

Oprócz prostego kodowania binarnego istnieje kodowanie oparte na *kodzie Graya*. Ma ono tę zaletę, że chromosomy odpowiadające dwóm kolejnym liczbom całkowitym różnią się tylko jednym bitem. Zatem niewielkiej zmianie w dziedzinie liczb całkowitych towarzyszy zmiana tylko na jednym bicie (np.  $011_{(2)} \rightarrow 100_{(2)}$  czyli zmiana z  $3_{(10)}$  na  $4_{(10)}$  odpowiada zmiana w kodzie Graya z  $010 \rightarrow 110$ ). W ten sposób reprezentacja informacji zapisanych w chromosomie jest bardziej naturalna, a przez to wpływa na poprawę pracy algorytmów.

Kolejnym przykładem modyfikacji kodowania jest kodowanie logarytmiczne. Ten sposób zapisu chromosomów stosowany jest w celu zmniejszenia długości chromosomów w

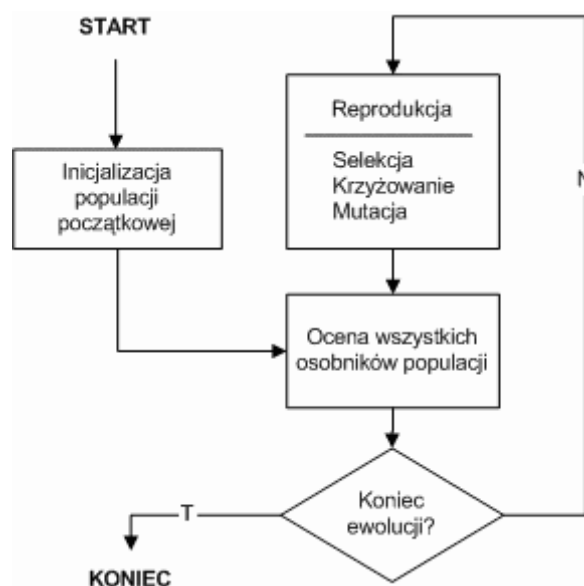
algorytmie genetycznym. W kodowaniu logarytmicznym pierwszy bit ( $\alpha$ ) ciągu kodowego oznacza znak funkcji wykładniczej, drugi bit ( $\beta$ ) oznacza znak wykładnika funkcji, a pozostałe bity są reprezentacją wykładnika funkcji.

Inną modyfikacją metod kodowania jest wykorzystanie zapisu zmiennopozycyjnego. Każdy chromosom jest zapisany jako wektor liczb zmiennopozycyjnych. Dokładność takiego podejścia jest z reguły znacznie wyższa niż przy tradycyjnym kodowaniu binarnym, co wynika z faktu, że reprezentacja zmiennopozycyjna może objąć szerszy obszar dziedziny. W przypadku kodowania zmiennopozycyjnego znacznie łatwiej jest zaprojektować specjalistyczne narzędzia ułatwiające postępowanie w przypadku nietrywialnych ograniczeń.

Jedną z podstawowych zasad przy wyborze kodowania jest *zasada minimalnego alfabetu*, która mówi, że: należy wybrać najmniejszy alfabet, w którym dane zadanie wyraża się w sposób naturalny.

### 3.5 Prosty algorytm genetyczny

Prosty algorytm genetyczny SGA (ang. *Simple Genetic Algorithm*) został zaproponowany przez Johna Hollanda (1975 r.)



Rys. 3.3 Ogólny schemat działania algorytmu genetycznego

W prostym algorytmie genetycznym wyróżnia się charakterystyczne etapy, zostały one wymienione w kolejnych podpunktach.

#### Inicjalizacja

Czyli utworzenie populacji początkowej, polega na wyborze żądanej liczby chromosomów reprezentowanych przez ciągi binarne określonej długości.

Algorytm rozpoczyna działanie od określenia rozmiaru populacji, który powinien być liczbą parzystą większą od zera. W zależności od parametrów zdefiniowanego zadania, należy zdefiniować długość chromosomu.

### Ocena przystosowania chromosomów

Polega na obliczeniu wartości funkcji przystosowania (funkcji celu) dla każdego chromosomu z tej populacji. Im większa jest wartość tej funkcji, tym lepsza „jakość” chromosomu.

### Sprawdzenie warunku zatrzymania

Określenie warunku zatrzymania algorytmu genetycznego zależy od konkretnego zastosowania i może nastąpić w przypadkach:

- Kiedy uzyskano pewną wartość maksymalną którą uznano za wystarczającą;
- Podczas, gdy kolejne generacje populacji nie wnoszą poprawy osiągniętych wyników;
- Po określonej liczbie generowanych populacji.

Możliwe też są kombinacje poszczególnych przypadków.

### Reprodukcja (selekcja)

Jest to proces, w którym indywidualne ciągi kodowe zostają powielone proporcjonalnie do wartości funkcji celu (przystosowania)  $f_i$  danego ciągu kodowego. Funkcja celu stanowi pewien miernik jakości danego osobnika, którą chce się maksymalizować. Reprodukcja zależna od przystosowania polega na tym, że ciągi kodowe o wyższym przystosowaniu mają większe prawdopodobieństwo wprowadzenia jednego lub więcej potomków do następnego pokolenia.

Reprodukcja jest namiastką darwinowskiej zasady doboru naturalnego, zgodnie z którą przetrwa najlepiej przystosowany osobnik.

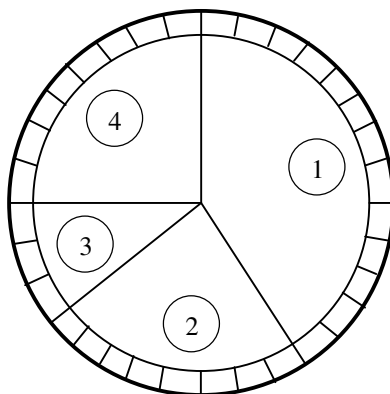
Wyróżnia się trzy podstawowe metody selekcji:

- **Selekcja proporcjonalna** (ang. *fitness proportionate selection*)

Polega na  $n$  krotnym losowaniu ( $n$ - liczebność populacji) ze starej populacji osobników, które zostaną przepisane do nowej populacji. Każdy osobnik może mieć różne prawdopodobieństwo wylosowania, wyrażone przez iloraz przystosowania tego osobnika i sumy przystosowań wszystkich osobników (3.1).

$$p_{select} = \frac{f_i}{\sum f} \quad (3.1)$$

Metoda inaczej nazywana *metodą ruletki* zawdzięcza swoją nazwę analogii do losowania poprzez ruletkę. Tarczę ruletki dzieli się pomiędzy osobników w taki sposób, że pole powierzchni zajęte przez osobnika jest proporcjonalne do jego ilorazu przystosowania i sumy przystosowań wszystkich osobników.



Rys. 3.4 Operacja reprodukcji za pomocą ruletki

Można spodziewać się, że w wyniku losowania oczekiwana liczba kopii osobników zasilających tzw. *pulę rodzicielską*, czyli nowe pokolenie wynosić będzie:

$$\frac{f_i}{\bar{f}} = \frac{f_i}{\frac{\sum f}{N}} \quad (3.2)$$

W przypadku naszkicowanym na rys. 3.4 można założyć, że osobnik o numerze (1) zostanie wylosowany, co najmniej raz ze względu na dużą powierzchnię zajmowaną na planszy, natomiast osobnik o numerze (3) prawdopodobnie nie zostanie wcale wylosowany;

- **Ranking liniowy** (ang. *rank selection*)

Polega na umieszczeniu osobników na liście posortowanej względem wartości funkcji oceny. Każdy osobnik otrzymuje pewną wartość (przystosowanie) zależną od pozycji, jaką zajmuje na liście.

I tak dla populacji z trzema osobnikami, prawdopodobieństwo wybrania najsilniejszego osobnika wynosi  $\frac{3}{6}$ , a najsłabszego osobnika wynosi  $\frac{1}{6}$  (przykład na rys.3.5).

Ranking liniowy zmniejsza przewagę najlepszych rozwiązań, gdy ich wartość funkcji jest bardzo duża i zwiększa przewagę, gdy wartość ta jest bardzo mała. W ten sposób ranking liniowy eliminuje przedwczesną zbieżność oraz przeciwdziała stagnacji.

Wartości funkcji przystosowania	Wartość prawdopodobieństwa		
	Ruletka	Lista rankingowa	
3	3/25 12%	1 (najgorszy na liście)	1/6 16,6%
15	15/25 60%	3 (najlepszy na liście)	3/6 50%
6	6/25 24%	2	2/6 33,3 %

Rys. 3.5 Przykład porównujący metodę ruletki i listy rankingowej

Do wad selekcji rankingowej należą: potrzeba porządkowania populacji, trudności w analizie teoretycznej oraz brak uzasadnienia biologicznego.

- **Turniej** (ang. *tournament selection*)

Z całej populacji losowo wybiera się kilku osobników (tzw. grupę turniejową). Z grupy turniejowej wybiera się najlepiej przystosowanego osobnika i umieszcza w nowo tworzonej populacji. Operacja powtarzana jest do czasu utworzenia całej nowej populacji. Istotne jest optymalne zdefiniowanie liczebności grupy turniejowej (zbyt mała wprowadza ryzyko znalezienia się w puli rodzicielskiej osobnika niemal najslabszego, zbyt duża może spowodować, że praktycznie zawsze do puli przechodzić będzie osobnik najsilniejszy).

Ważna jest jedynie informacja o „wyższości” jednego rozwiązania nad innymi (nie jest wymagana maksymalizacja funkcji oceny, jak w powyższych metodach). Do zalet selekcji turniejowej należy zaliczyć:

- brak przedwczesnej zbieżności,
- brak stagnacji,
- brak potrzeby porządkowania całej populacji,
- niepotrzebna jawna funkcja przystosowania,
- odwzorowanie selekcji naturalnej.

### Podsumowanie

Nie zawsze korzystna jest selekcja ilości osobników proporcjonalna do wartości funkcji przystosowania danego osobnika na tle całej populacji. W ściśle określonych sytuacjach może to spowodować zagrożenie zdominowania puli rodzicielskiej tylko jednym rodzajem osobników. Oczywiście jest, że najsilniejszy osobnik powinien być wyróżniony i taką cechą właśnie charakteryzuje się ranking liniowy. Zmniejsza on niebezpieczeństwo dominacji. Turniej natomiast wprowadza dodatkowy element losowości, ponieważ najsilniejszy element

nie zawsze musi się znaleźć w grupie turniejowej. Tym samym turniej zwiększa szanse innych osobników niż najsilniejszy, odznaczających się jednak dużą wartością funkcji przystosowania. Reasumując metoda ruletki promuje zawsze najsilniejszego osobnika proporcjonalnie do jego przystosowania, metoda rankingu liniowego dąży do niwelowania różnicy pomiędzy najsłabszymi a najsilniejszymi osobnikami. Turniej promuje silnych osobników niekoniecznie będących najsilniejszymi.

W metodzie ruletki z kolei osobniki podobne, dostają równy wycinek pola ruletki i w ten sposób presja selekcyjna maleje. Algorytm słabiej rozróżnia osobniki dobre od słabszych.

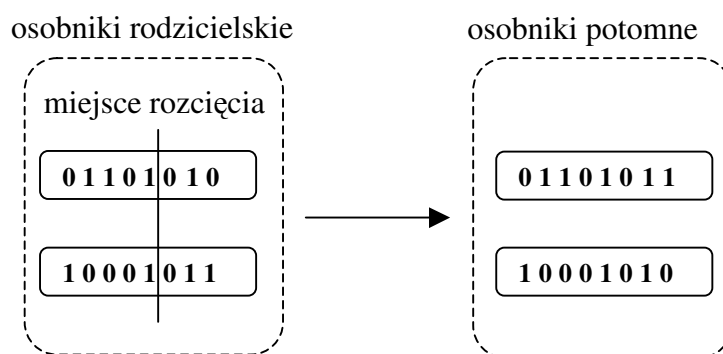
W procesie reprodukcji należy wyróżnić dwie szczególne strategie:

- Strategia elitarna – Najlepsze chromosomy są chronione w kolejnych iteracjach (w algorytmie klasycznym najlepiej przystosowane osobniki niekoniecznie muszą wchodzić w skład następnej generacji). Podstawowa zaleta to gwarancja zaistnienia zbieżności, a podstawowa wada to ryzyko wpadnięcia w ekstremum lokalne
- Algorytm genetyczny z częściową wymianą populacji – Część populacji przechodzi do następnej generacji bez modyfikacji (nie podlega omówionym w kolejnych podrozdziałach operacjom krzyżowania i mutacji)

### Krzyżowanie

Proces krzyżowania nazywany też krosowaniem (ang. *crossover*) polega na przypadkowym kojarzeniu osobników z puli rodzicielskiej. W tym celu wybiera się punkt krzyżowania (ang. *crosspoint*) dzielący chromosom na części (w przypadku kojarzenia pary osobników rodzicielskich). Miejsce to podobnie jak osobników wybiera się w sposób losowy, a pozycja może przybierać wartość z zakresu liczb od 0 do  $l-1$  gdzie  $l$  jest długością chromosomu.

Krzyżowanie polega na zamianie miejscami wszystkich znaków od pozycji  $k+1$  do  $l$  włącznie w obu elementach pary, tworząc w ten sposób dwa nowe ciągi.



Rys. 3.6 Schemat krzyżowania prostego dwóch osobników rodzicielskich

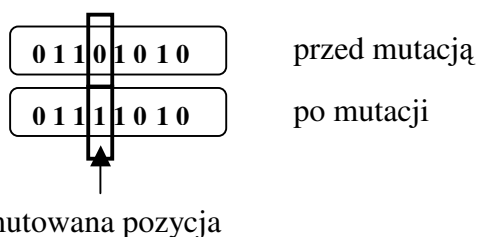
W zależności od ilości punktów krzyżowania wyróżnia się:

- krzyżowanie jednopunktowe,
- krzyżowanie wielopunktowe (chromosomy rodziców podzielone są na kilka części, a osobniki potomne tworzone są na podstawie przeplatanych wycinków rodziców),
- krzyżowanie równomierne (odbywa się zgodnie z wylosowanym wzorcem wskazującym, które geny są dziedziczone od pierwszego z rodziców, pozostałe natomiast dziedziczone są od drugiego z rodziców).

Prosty algorytm genetyczny zakłada, że prawdopodobieństwo krzyżowania  $p_c = 1$ . Jednak wiele modyfikacji tego algorytmu (w zależności od przyjętej strategii reprodukcji) zapewnia możliwość sytuacji, aby do puli rodzicielskiej przechodziły osobniki nie poddane operacji krzyżowania.

### Mutacja

W sztucznych systemach genetycznych mutacja polega na sporadycznej (zachodzącej z niewielkim prawdopodobieństwem), przypadkowej zmianie wartości elementu ciągu kodowego (genu). W przypadku kodu binarnego oznacza to negację bitu (*allele*), czyli zamianę jedynki na zero i na odwrót. Mutacja jest przypadkowym błędzeniem w przestrzeni ciągów kodowych. Wykonywana jest po procesie krzyżowania. Dla każdego genu w chromosomach potomków sprawdzane jest prawdopodobieństwo zajścia mutacji, które zwykle przybiera wielkość 0.001 bądź jeszcze rzadziej. Głównym jej zadaniem jest umożliwienie odtworzenia zniszczonego podczas krzyżowania chromosomu (wartość funkcji przystosowania potomka jest mniejsza od wartości funkcji rodzica).



Rys. 3.7 Schemat procesu mutacji



## 3.6 Zastosowanie

Wyróżnia się następujące zastosowania algorytmów genetycznych:

### **Rozwiązywanie problemów NP**

Algorytmy genetyczne znajdują zastosowanie tam, gdzie nie jest dobrze określony lub znany sposób rozwiązywania problemu, ale znany jest sposób, w jaki należy ocenić jakość rozwiązania. Przykładem jest np. problem komiwojażera, w którym należy znaleźć najkrótszą drogę łączącą wszystkie miasta, tak by przez każde miasto przejść tylko raz. Ocena jakości proponowanej trasy jest natychmiastowa, natomiast znalezienie optymalnej trasy kwalifikuje się do klasy problemów NP zupełnych. Przy zastosowaniu podejścia ewolucyjnego dobre rozwiązanie można znaleźć bardzo szybko, ale oczywiście można być jedynie pewnym uzyskania rozwiązań sub-optymalnych, co wynika z formalnie opisanej trudności problemów klasy NP.

### **Projektowanie genetyczne**

Algorytmy genetyczne wykorzystywane mogą być do zarządzania populacją sieci neuronowych, projektowania maszyn lub obwodów elektrycznych. Inżynierowi podczas tworzenia nowych pomysłów nie chodzi o znalezienie najlepszego możliwego rozwiązania. Wystarczy tylko przybliżone spełnienie granicznych warunków oraz optymalizacja projektu. Algorytmy genetyczne w odróżnieniu od człowieka nie działają schematycznie. Program nie zna wcześniejszych projektów i dlatego czasami wykazuje się pewną inwencją. Co więcej, człowiek często opiera się na bardzo przybliżonych modelach, które dają fałszywy obraz problemu. Algorytm genetyczny może przeanalizować złożony model zagadnienia i znaleźć rozwiązanie, które dla człowieka stanowiłoby dużą trudność.

### **Przeszukiwanie**

Algorytmy genetyczne umożliwiają tworzenie skutecznych mechanizmów przeszukiwania dużych przestrzeni rozwiązań. Ponieważ grupowanie należy do tej kategorii zadań to oczywiste jest, że algorytmy genetyczne stosowane są w grupowaniu. Algorytmy genetyczne są bardziej niezależne od wstępnej inicjalizacji oraz mniej skłonne do znajdowania lokalnych rozwiązań w miejsce optymalnych. Przykładem może być zagadnienie grupowania, w którym w miejsce klasycznych algorytmów z powodzeniem stosuje się algorytmy genetyczne.

### 3.7 Optymalizacja sieci neuronowej za pomocą AG

Wybór odpowiedniej topologii sieci neuronowych jest szczególnie ważny, ponieważ nieodpowiednia topologia<sup>24</sup> nie pozwoli sieci na rozwiązanie problemu. Sieć albo nie będzie w stanie nauczyć się danych wejściowych, albo nie będzie w stanie uogólniać wprowadzanych danych. Rozwiązaniem optymalnym jest dobrze uogólniająca sieć o najmniejszej topologii, potrafiąca się szybko uczyć. Jednak trudno znaleźć ręcznie odpowiednią dla danego rozwiązania topologię. Złożoność problemu polega na odpowiednim przedstawieniu topologii sieci neuronowej (wszystkich albo tylko wybranych elementów sieci neuronowej) za pomocą chromosomu który jako osobnik będzie uczestniczył w pętli algorytmu genetycznego.

### 3.8 Twierdzenie o schematach

W przypadku prostego algorytmu genetycznego mają zastosowanie niektóre rozważania analityczne, tłumaczące zasadę działania oraz dowodzące zbieżności. Pomocne mogą być przedstawione poniżej: twierdzenie o schematach i hipoteza cegiełek. Należy jednak stwierdzić, że na dzień dzisiejszy nie istnieją metody dowodzące zbieżności w stricte deterministycznym sensie<sup>25</sup>.

Schematem nazywa się wzorzec opisujący podzbiór ciągów podobnych ze względu na ustalone pozycje<sup>26</sup>. Schemat buduje się wprowadzając symbole *nieistotne* (\*) do struktury DNA. Schemat reprezentuje wszystkie łańcuchy, które zgadzają się z nim na wszystkich pozycjach innych niż (\*). I tak dla przykładu do schematu (\*0101) pasują dwa łańcuchy

$$\{(10101), (00101)\}$$

a do schematu (\*10\*1) cztery łańcuchy

$$\{(01001), (01011), (11001), (11011)\}$$

Rzędem schematu  $S$  (oznaczony jako  $o(S)$ ) nazywa się liczbę ustalonych (nieistotnych) pozycji w schemacie, czyli wartość wynoszącą długość szablonu pomniejszoną o liczbę symboli nieistotnych<sup>27</sup>. O poprzednich schematach (\*0101 i \*10\*1) można zatem powiedzieć, że są czwartego i trzeciego rzędu.

<sup>24</sup> Przez topologię sieci neuronowej rozumie się liczbę neuronów w sieci oraz sposób ich połączenia

<sup>25</sup> [PHD]

<sup>26</sup> [Goldberg]

<sup>27</sup> [Goldberg]

*Rozpiętością schematu S (oznaczoną jako  $\delta(S)$ ) nazywa się odległość między dwiema skrajnymi pozycjami ustalonymi<sup>28</sup>. Określa ona zwartość informacji zawartej w schemacie, jest też wykorzystywane przy obliczaniu prawdopodobieństwa przeżycia po krzyżowaniu.*

Dla tych samych schematów (\*0101 i \*10\*1) długość schematu wynosi 3.

Im rząd schematu oraz długość definiująca schemat są mniejsze, tym większe prawdopodobieństwo przetrwania schematu. Bardziej zwarty schemat trudniej przeciąć w trakcie operacji krzyżowania, a krótszy trudniej zmienić przez operator mutacji. Tezę tą potwierdza twierdzenie o schematach:

*Krótkie niskiego rzędu i oceniane powyżej średniej schematy uzyskują wykładniczo rosnącą liczbę łańcuchów w kolejnych pokoleniach<sup>29</sup>.*

W literaturze jest też definiowana hipoteza o blokach budujących:

*Algorytm genetyczny poszukuje działania zbliżonego do optymalnego przez zestawienie krótkich, niskiego rzędu schematów o dużej wydajności działania, zwanych blokami budującymi (cegietkami).<sup>30</sup>*

Oceniając poszczególne ciągi kodowe algorytm genetyczny pośrednio ocenia jakość reprezentowanych przez nie schematów. Krzyżowanie pozwala na zestawianie ze sobą wysokiej jakości schematów niskiego rzędu, budując tak samo dobre lub lepsze schematy wyższego rzędu.

---

<sup>28</sup> [Goldberg]

<sup>29</sup> [Goldberg]

<sup>30</sup> [Goldberg]

---

# CZĘŚĆ II ANALIZA GIEŁDOWA

*„Nigdy naprawdę nie straciłeś,  
dopóki nie przestałeś próbować”*

MIKE DITKA

## 4 Analiza giełdowa

Giełda Papierów Wartościowych na której dokonuje się obrotu akcjami może przysporzyć wymiernych zysków w czasie hossy ale wraz z zyskiem istnieje niebezpieczeństwo poniesienia strat w chwili gdy kurs akcji będzie malał. Nie istnieje model matematyczny potrafiący opisać zachowania giełdowe. Istnieją jednak instrumenty takie jak Analiza Techniczna które z większym bądź mniejszym prawdopodobieństwem potrafią przewidzieć zmianę trendu lub jego podtrzymanie na podstawie danych historycznych.

W poniższym rozdziale zostanie omówiona analiza giełdowa, czyli ogół czynności zmierzających do rozpoznania obecnego trendu i próbie odpowiedzi na pytanie jak potoczy się dalszy kurs badanych walorów giełdowych. Ponadto wyjaśnione zostaną podstawowe pojęcia związane z Giełdą Papierów Wartościowych..

Pomimo tego, że sztuczne sieci neuronowe mogą nadawać się zarówno do analizy technicznej jak i fundamentalnej w pracy zdecydowano, że do dokonywania prognoz wykorzystana zostanie wiedza z zakresu analizy technicznej. Dlatego też tejże analizie poświęcona zostanie większość uwagi.

### 4.1 Notowania giełdowe

Giełda papierów wartościowych jest miejscem, gdzie kupuje się i sprzedaje papiery wartościowe. Papier wartościowy to dokument stwierdzający istnienie określonego prawa majątkowego, zbywalnego na rzecz innego właściciela. Transakcje są zawierane pomiędzy pośrednikami, reprezentującymi właścicieli akcji czy innych instrumentów, a kursy są określane na bieżąco, na podstawie popytu i podaży.

Na giełdzie dochodzi do konfrontacji ofert sprzedających papiery wartościowe z zapotrzebowaniem, zgłaszanym przez kupujących.

Na podstawie złożonych zleceń określa się kursy akcji, co umożliwia przeprowadzenie transakcji. W początkowej fazie istnienia Warszawskiej Giełdy kursy każdej akcji były określane tylko raz dziennie, obecnie w ten sposób handluje się wyłącznie akcjami w systemie

pojedynczego fixingu. Zupełnie inny jest mechanizm notowań ciągłych; tu kurs jest ustalany na bieżąco, może się więc zmieniać wielokrotnie w czasie jednej sesji.

Przedmiotem obrotu na giełdzie są przede wszystkim akcje spółek, akcje NFI, ponadto prawa poboru nowych akcji, obligacje oraz instrumenty pochodne.

Niniejsza praca opiera się na analizie Warszawskiej Giełdy Papierów Wartościowych, której pierwsza sesja odbyła się 16 kwietnia 1991 roku, a od 1994 roku notowania odbywają się 5 razy w tygodniu. W obecnej chwili (grudzień 2005) na WGPW notowanych jest 265 walorów giełdowych, nie wszystkie z nich są spółkami akcyjnymi, oprócz nich notowane są Narodowe Fundusze Inwestycyjne oraz indeksy giełdowe [4.1.2] (notowane są także inne walory takie jak profile walut, futures oraz towary, ale ze względu na niewykorzystywanie ich w pracy obecność takowych nie będzie dalej omawiana).

#### 4.1.1 Ceny i wolumen

W pracy do analizy giełdowej posłużono się historycznymi informacjami z ubiegłych sesji. Przykładowa sesja numer 3200 z dnia 18 listopada 2005 r. zawiera m.in. notowania poniższych walorów:

Spółka	Kurs otwarcia	Kurs najniższy	Kurs najwyższy	Kurs zamknięcia	Wolumen	Zmiana
01N	10.50	10.60	10.55	10.60	18918	-0.47%
INT	15.21	15.80	15.75	16.00	7973	0%
KGH	51.60	51.60	53.10	53.30	488132	1.36%
PKN	59.20	59.50	61.00	61.30	691833	1.68%
VST	34.40	34.50	34.40	34.90	609	-1.45%
WIG	33564.92	33667.04	33854.83	33909.63	20721681	-0.23%

Rys. 4.1 Przykład notowań wybranych walorów na sesji 3200

Każde notowanie charakteryzuje się m.in. następującymi informacjami<sup>31</sup>:

**Cena otwarcia** (ang. *open*) jest to cena, po której zostaje zawarta pierwsza transakcja w danym okresie (np. danego dnia).

**Cena zamknięcia** (ang. *close*) jest to cena, po której została zawarta ostatnia transakcja w danym okresie. Dzięki swej dostępności cena zamknięcia to najczęściej wykorzystywana cena w analizie technicznej.

<sup>31</sup> [Tarczyński]

Relacje pomiędzy ceną otwarcia i zamknięcia w danym okresie uznawane są przez większość analityków za bardzo istotne.

**Cena najniższa (minimum)** jest to najniższa cena osiągnięta przez papier w danym okresie. Wielkość ta reprezentuje minimum, jakie gotowi są zaakceptować potencjalni sprzedający pozbywając się papieru.

**Cena najwyższa (maksimum)** jest to najwyższa cena osiągnięta przez papier w ciągu analizowanego okresu. Wyznacza ona maksimum, jakie kupujący są skłonni zaakceptować nabywając w danym momencie walor.

**Wolumen** – informuje o zaangażowaniu inwestorów, pozwala na prostą ocenę łatwości i siły ruchu cenowego, a dzięki temu pomaga określić, czy większe prawdopodobieństwo ma kontynuacja trendu, czy jego zmiana. Podstawowa zasada mówi, że wolumen jest wskaźnikiem wyprzedzającym cenę (wcześniej niż ceny sygnalizuje kontynuację lub zmianę trendu). Kontynuacja trendu wzrostowego jest bardziej prawdopodobna, jeżeli towarzyszy mu rosnący wolumen, analogicznie w trakcie ruchów korekcyjnych należy się spodziewać niższego wolumenu. Jeżeli dzieje się odwrotnie, to istnieje spore prawdopodobieństwo zmiany trendu.

### 4.1.2 Indeksy giełdowe

Indeksy giełdowe to wskaźniki, obrazujące zmiany cen na giełdzie - odzwierciedlają one tym samym stan rynku lub wybrane jego segmenty. Giełda warszawska publikuje następujące indeksy:

- **WIG (Warszawski Indeks Giełdowy)** - Uwzględnia zmiany cen akcji największych oraz średnich spółek (w tym zagranicznych) notowanych na giełdzie, których łączna wartość rynkowa stanowi 99% kapitalizacji giełdy;
- **WIG20** - Odzwierciedla zmiany cen akcji dwudziestu spółek o największej wartości rynkowej i największych obrotach na giełdzie. W jego skład mogą wchodzić zarówno spółki polskie jak i zagraniczne;
- **WIG-PL** - Uwzględnia zmiany cen akcji największych i średnich spółek polskich, notowanych na rynku podstawowym i równoległym, których łączna wartość rynkowa stanowi 99% kapitalizacji krajowych spółek giełdowych;
- **MIDWIG** - Odzwierciedla zmiany cen grupy spółek giełdowych tzw. średniej wielkości;
- **WIRR** - Odzwierciedla zmiany cen najmniejszych spółek, nie zaliczonych do indeksów WIG20 i WIG;

- TechWIG - Informuje, co się dzieje z akcjami należącymi do sektora SiTech, czyli spółkami zaliczanymi do tzw. spółek nowych technologii;
- NFI - Uwzględnia zmiany cen akcji czternastu Narodowych Funduszy Inwestycyjnych;
- Subindeksy sektorowe: WIG Banki, WIG Budownictwo, WIG Informatyka, WIG Spożywczy, WIG Telekomunikacja. Są to indeksy dla sektorów gospodarki, które mają największe znaczenie dla obrotu giełdowego. Opierają się na metodologii indeksu WIG, ale grupują jedynie spółki z danej branży.

## 4.2 Analiza techniczna

Analiza techniczna wymaga jedynie posiadania informacji o aktualnym oraz historycznym kursie akcji. Efektem analizy jest przede wszystkim wyznaczenie punktów kupna i sprzedaży, które można wytyczyć znając przyszły trend kursów akcji. W dalszej części opisane zostaną najważniejsze właściwości analizy technicznej mające wpływ na analizę i predykcje rynku giełdowego.

### 4.2.1 Trzy podstawowe przesłanki analizy technicznej

Literatura<sup>32</sup> wymienia podstawowe przesłanki, na których opiera się teoria Analizy Technicznej.

#### **Rynek dyskontuje wszystko**

Podstawowe założenie analizy technicznej polega na tym, że wszelkie czynniki, które mają wpływ na cenę są już w niej uwzględnione. Uważa się, że zachowania cen odzwierciedlają zmiany w relacjach popytu i podaży. Analiza techniczna wychodzi z założenia, że nie trzeba badać czynników wpływających na cenę waloru lub doszukiwać się przyczyn spadków bądź wzrostów. Rynek posiada wiedzę i to on kształtuje cenę, dlatego też na podstawie wykresów i wskaźników analizy technicznej, można skutecznie prognozować, w którą stronę podąża rynek. Oczywiście nie odrzuca się stwierdzenia, że przyczyną trendów na giełdzie są uwarunkowania gospodarcze. Analiza techniczna uważa jedynie, że rynek łatwiej można zrozumieć i przewidzieć, dokonując analizy jego zachowań, czyli wykresów.

---

<sup>32</sup> [Murphy]

### **Ceny podlegają trendom**

W wykresie cen analiza techniczna stara się odnaleźć trend, czyli kierunek, w którym podążają ceny. Wczesne rozpoznanie kształtującego się trendu pozwala dokonać transakcji, która powinna przynieść zyski (kupić, gdy tworzy się trend rosnący, sprzedać, gdy malejący). Analitycy zakładają mianowicie, że istnieje większe prawdopodobieństwo kontynuacji trendu niż jego odwrócenie.

### **Historia się powtarza**

Analiza wykresów pozwala znaleźć powtarzające się wzory (formacje), według których zmieniają się ceny. Związane jest to z powtarzalnością zachowań ludzkich w określonych sytuacjach. Analitycy, znając najczęściej występujące wzory (formacje) starają się odnaleźć je w aktualnych notowaniach i na tej podstawie prognozować przyszłość.

Podsumowując: Pierwsza przesłanka mówi o tym, że rynek w pełni dyskontuje wszelkie czynniki mogące mieć wpływ na kurs akcji, z czego wynika, że badanie jedynie samych cen jest podejściem wystarczającym. Druga przesłanka związana jest z trendem, jednym z podstawowych pojęć analizy technicznej. Stwierdza ona, że kursy poruszają się w trendach, a zadaniem analizy technicznej jest odnajdywanie ich zmian. Trzecia przesłanka dotyczy psychologii rynków i zakłada, że w warunkach podobnych do przeszłych, sytuacja rynkowa może kształtować się w sposób podobny.

Analiza techniczna zajmuje się zatem badaniem skutków zachowań rynku (ceny) i poszukiwaniem na wykresach cen historycznych, trendów i formacji, które pomogą przewidzieć zachowanie rynku. W swojej pracy analitycy wspomagają się m.in. wskaźnikami, które mają pomóc w lepszym zrozumieniu ruchów cen.

Dzięki temu można znaleźć odpowiedź na trzy główne pytania, które zadaje sobie inwestor:

- Co kupić lub sprzedać?
- Kiedy kupić?
- Kiedy sprzedać?

O ile w analizie fundamentalnej poszukuje się odpowiedzi na powyższe pytania, starając się interpretować dane o sytuacji spółki, jej pozycji finansowej, ocenie nowo wprowadzanych produktów itd., analiza techniczna koncentruje się na wpływie tych informacji na zachowanie inwestorów. Analitycy techniczni nie interesują się samą informacją, lecz jej skutkiem, którego wyrazem są zmiany w wycenie papieru i wahania obrotów.



### 4.2.2 Wybrane wskaźniki analizy technicznej

Wyróżnia się następujące rodzaje wskaźników analizy technicznej:

- Wskaźniki śledzące trend, działają dobrze, gdy kurs znajduje się w trendzie wzrostowym lub spadkowym, dla trendu horyzontalnego sygnały wskaźnika będą nieprawidłowe;
- Oscylatory potrafią znajdować punkty zwrotne dla trendów horyzontalnych. Dla trendów wzrostowych lub spadkowych są źródłem informacji o wykupieniu lub wyprzedaniu rynku, dostarczając w ten sposób informacji wyprzedzających trend;
- Wskaźniki nastroju, pozwalają badać „psychologię mas”. Wskaźniki te mogą być prowadzącymi lub równoczesnymi w stosunku do trendu.

**Prosta średnia krocząca** – (ang. *Simple Moving Average - SMA*) jest jednym z najbardziej uniwersalnych i najpowszechniej stosowanych wskaźników technicznych<sup>33</sup>. Wartość obliczana jest jako średnia arytmetyczna z  $k$  ostatnich dni.

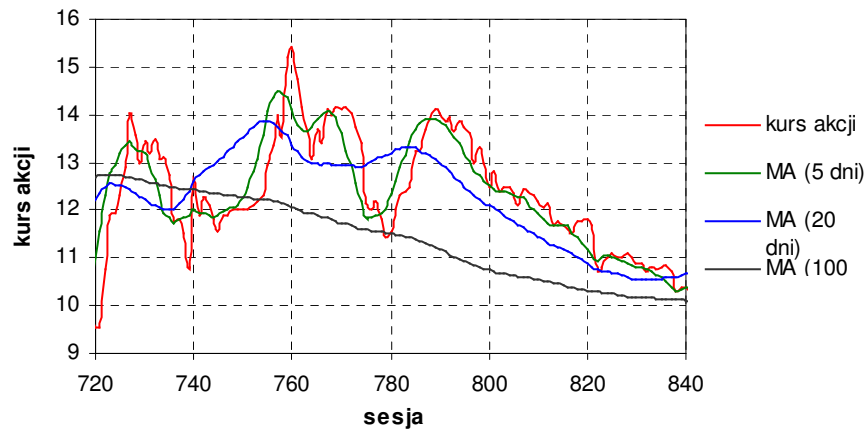
$$SMA(t, k) = \frac{Kurs(t) + Kurs(t-1) + \dots + Kurs(t-(k-1))}{k} \quad (4.1)$$

Zadaniem tego wskaźnika jest badanie istniejącego lub też nowo kształtującego się trendu. Nie prognozuje on zachowań rynku, a jedynie podąża za trendem. Średnia krocząca jest też narzędziem służącym do niwelacji odchyłeń głównego ruchu cen. Uśrednienie cen pozwala na uzyskanie mniej pofalowanej linii, bez przypadkowych fluktuacji (szumów), ułatwiając w ten sposób obserwację zasadniczego trendu.

Sygnał kupna otrzymuje się, gdy kurs przebija średnią od dołu, sygnał sprzedaży, gdy przebija ją od góry (według teorii tendencja, w której kurs jest większy od średniej z  $k$ -sesji jest podstawą do optymistycznej prognozy, co do zachowania się kursu w przyszłości). W zależności od długości średnich są to sygnały (5-25 dni) krótko, (25-75 dni) średnio lub (powyżej 75 dni) długoterminowe. Inna metoda analizy średnich polega na naniesieniu na wykres dwóch średnich (np. 5 i 25 dniową). Wtedy przejście szybszej średniej (5 sesyjnej) ponad wolniejszą daje sygnał kupna, a spadek poniżej - sprzedaży.

---

<sup>33</sup> [Plummer]

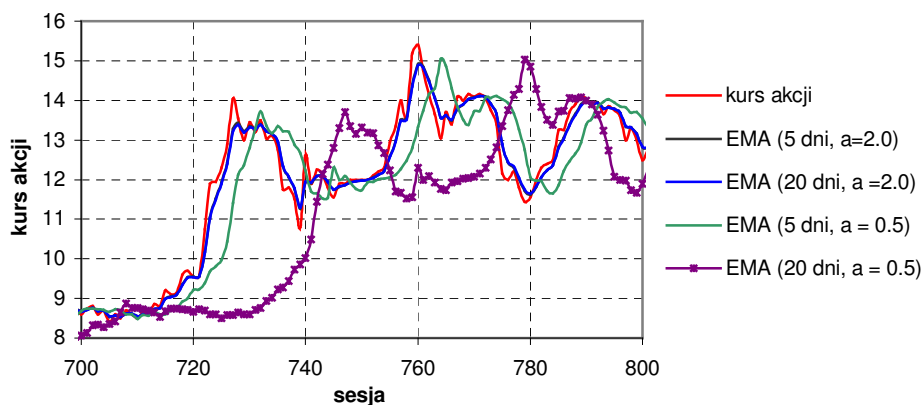


Rys. 4.3 Prosta średnia krocząca - SMA

**Wykładnicza średnia krocząca** (ang. *Exponential Moving Average - EMA*) – rozwiązuje główną wadę prostej średniej kroczącej, a mianowicie fakt, że wszystkie przeszłe ceny mają takie samo znaczenie<sup>34</sup>. W zależności od parametru  $a$ , wykładnicza średnia krocząca przykłada większą wagę bardziej aktualnych cen ( $a > 1$ ), dla najstarszych cen ( $a < 1$ ), a może być także szczególnym przypadkiem prostej średniej kroczącej ( $a = 1$ ).

$$EMA = \frac{Kurs(t) + Kurs(t-1) \cdot a + Kurs(t-(k-1))}{1 + a + a^2 + a(n-1)} \quad (4.2)$$

$$a = 1 - \frac{2}{(k+1)a(i)} \quad (4.3)$$



Rys. 4.4. Wykładnicza średnia krocząca

**Wskaźnik zbieżności/rozbieżności średnich kroczących** – (ang. *Moving Average Convergence/Divergence - MACD*) to oscylator średnich wykładniczych ze ściśle ustalonymi parametrami. MACD jest różnicą wartości długo i krótkoterminowej średniej wykładniczej.

<sup>34</sup> [Plummer]

Najczęściej stosuje się średnie o opóźnieniu 8 i 17 dni (dla inwestycji długoterminowych przyjmuje się średnie 12 i 26 dniową). Analiza MACD związana jest z linią *SIGNAL*, która jest średnią wykładniczą o okresie 9 dni wyliczoną ze wskaźnika:

$$MACD(x, y) = EMA(x) - EMA(y) \quad \text{dla: } x < y \quad (4.4)$$

W trakcie tendencji wzrostowej średnia wykładnicza z  $x$  sesji będzie rosłać szybciej, niż średnia z  $y$  sesji, co powoduje zwiększenie wartości oscylatora. Podczas tendencji spadkowej sytuacja jest odwrotna. Przecięcie przez wykres MACD swojej linii *SIGNAL* od góry interpretuje się jako sygnał *sprzedaży*, a od dołu jako sygnał *kupna*. Potwierdzeniem tych sygnałów jest przecięcie przez wykres MACD linii równowagi znajdującej się na poziomie 0.

**Wskaźnik zmiany** – (ang. *Rate Of Change - ROC*) jest to procentowa zmiana kursu z obecnej sesji w stosunku do kursu z przed  $k$  sesji. Do obliczania wskaźnika ROC służy poniższy wzór:<sup>35</sup>

$$ROC(t, k) = \frac{Kurs(t) - Kurs(t - k)}{Kurs(t - k)} * 100\% \quad (4.5)$$

ROC podaje o ile procent aktualna cena jest wyższa lub niższa od ceny sprzedaży sprzed  $k$  sesji. Zalecane wartości parametru  $k$  to 5 dla inwestycji krótkoterminowych i 10 dla średnioterminowych. Wartość wskaźnika ROC oscyluje wokół zera przyjmując dodatnie bądź ujemne wartości.

Z reguły wyznacza się linie wysprzedań i wykupienia akcji (rynku), która łączy punkty maksymalne (minimalne) na wykresie. Linie te są poziome i wyznaczają możliwe zahamowanie wzrostu lub spadku wskaźnika, a więc i ceny, a nawet jego spadek (wzrost). Jeśli wzrostowi ceny towarzyszy wzrost wskaźnika, to tą zmianę ceny można uznać za trwałą. Adekwatnie ma to zastosowanie w sytuacji, gdy spadkowi ceny towarzyszy spadek wskaźnika.

Przy interpretacji tego wskaźnika analitycy kierują się kilkoma zasadami:

- jednoczesne osiągnięcie przez kurs i ROC maksimum wzmacnia trend rosnący,
- jednoczesne osiągnięcie przez kurs i ROC minimum wzmacnia trend malejący,
- osiągnięcie przez kurs maksimum (minimum) przy jednoczesnym spadku (wzroście) wskaźnika oznacza osłabienie trendu i możliwość jego zmiany na odwrotny,
- długotrwała konsolidacja (stabilizacja) wskaźnika zmian na jednym poziomie zapowiada zmianę trendu.

<sup>35</sup> [Tarczyński]

**Wskaźnik relatywnej siły** – (ang. *Relative Strenght Index - RSI*) określa wewnętrzną siłę akcji. Wskaźnik RSI przyjmuje wartości w skali od 0 do 100 - przy czym osiągnięcie wartości krańcowej (100) jest w praktyce niemożliwe. Natomiast wskaźnik przyjmuje wartość (0) tylko wtedy, gdy w ciągu ostatnich  $k$  sesji nie odnotowano zysków.

Sposób obliczania wskaźnika względnej siły:<sup>36</sup>

$$RSI(n, k) = 100 - \frac{100}{1 + RS} \quad (4.6)$$

$$RS = \frac{\text{\textit{Średni wzrost ceny w ostatnich k notowaniach}}}{\text{\textit{Średni spadek ceny w ostatnich k notowaniach}}} \quad (4.7)$$

$RSI$  – wartość wskaźnika względnej siły,

$RS$  – średnia wartość cen zamknięcia (netto) dla wybranej liczby dni.

Działanie wskaźnika RSI opiera się na kilku podstawowych zasadach:

- gdy wskaźnik osiągnie wartość 100 odwrócenie trendu w zniżkowy jest bardzo możliwe,
- gdy wskaźnik osiągnie wartość 0 odwrócenie trendu w zwyżkowy jest bardzo możliwe,
- gdy RSI osiągnie poziom 70% (linia wykupienia) lub większy oznacza to sygnał sprzedaży,
- gdy RSI osiągnie poziom 30% (linia wyprzedania) lub mniejszy oznacza to sygnał kupna.

Wartości 30% i 70% nie zawsze okazują się trafne, w okresie hossy oscylator przyjmuje często wartości powyżej 70%, w okresie bessy często są one niższe niż 30%. Wówczas należy dobrać poziomy wykupienia i wyprzedania na podstawie zachowania wskaźnika w przeszłości.

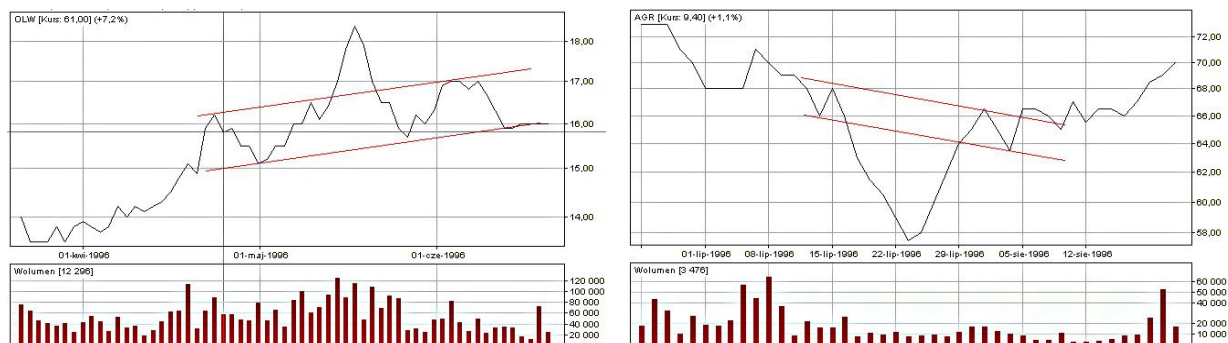
### 4.2.3 Formacje cenowe

Na podstawie obserwacji historycznych notowań, można wyróżnić pewne formacje cenowe, dla których z dużym prawdopodobieństwem można przewidzieć charakter trendu w przyszłości. Formacje cenowe w graficzny sposób obrazują powtarzalny efekt ścierania się rządzących rynkiem sił popytu i podaży, zapowiadający często krótkoterminowe zwycięstwo jednej ze stron. Należy wyróżnić trzy zasadnicze rodzaje formacji cenowych:

- formacje zapowiadające odwrócenie się trendu,
- formacje sygnalizujące kontynuację trendu,
- formacje uniwersalne, czyli takie, które spełniają oba warunki.

<sup>36</sup> [Tarczyński]

**Głowa i ramiona** – formacja ta najczęściej występuje na szczytach trendów wzrostowych informując, że rynek jest coraz słabszy i należy spodziewać się odwrócenia trendu. Charakterystycznym czynnikiem towarzyszącym tej formacji jest spadkowy charakter wolumenu dla kolejnych punktów: lewego ramienia, głowy oraz prawego ramienia. Oznacza to, że kapitał inwestycyjny wyczerpuje się.



Rys. 4.8 Formacja głowy i ramion oraz formacja odwróconej głowy i ramion

**Odwrócona głowa i ramiona** – sygnalizuje zakończenie spadków i możliwość początku trendu wzrostowego. Formacja ta jest lustrzanym odbiciem formacji *głowa i ramiona*.

**Formacja czworokąta** – składa się z dwóch równoległych linii wyznaczonych podobnie jak dla trendu horyzontalnego. Górna linia reprezentuje *opór*, czyli miejsce gdzie strona popytowa traci środki oraz przekonanie do dalszego utrzymywania wzrostów. Natomiast dolna linia *wsparcia* jest sygnałem, że rynek jest wysprzedany i prawdopodobne jest wybiecie w górę.



Rys. 4.9 Formacja czworokąta

**Formacja trójkątów** – zalicza się do formacji uniwersalnych, może ona sygnalizować odwrócenie trendu lub jego kontynuację. Trójkąty można podzielić na trzy zasadnicze grupy, w zależności od kąta tworzonego przez ich ramiona:

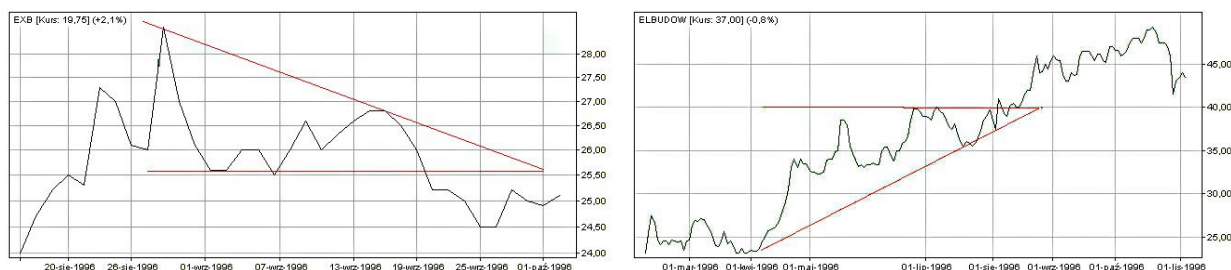
- trójkąty symetryczne,
- trójkąty zwyżkujące,
- trójkąty zniżkujące.

Trójkąty symetryczne charakteryzuje to, że górna i dolna linia są nachylone do siebie pod tym samym kątem. Formacja ta odzwierciedla równowagę, jaka istnieje na rynku i z dużym prawdopodobieństwem sygnalizuje kontynuację trendu.



Rys. 4.10 Formacje cenowe – trójkąt symetryczny.

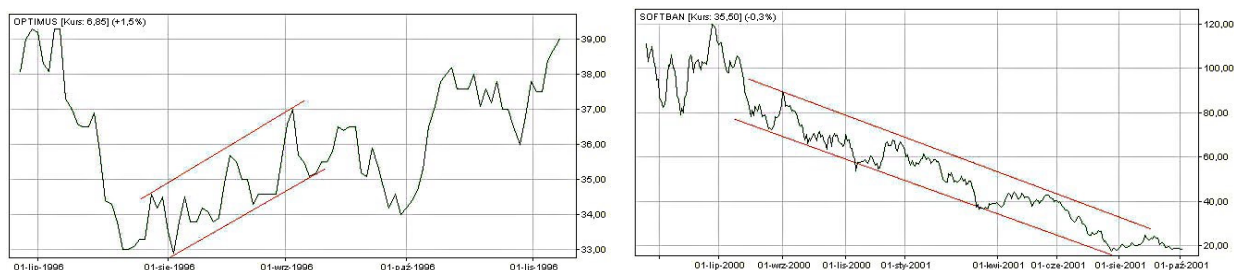
Trójkąty zniżkujące zbudowane są w oparciu o dwie linie: dolną poziomą oraz malejącą skierowaną w dół. Kąt nachylenia linii informuje o wyczerpującym się kapitale byków, a rosnącej przewadze niedźwiedzi. Trójkąty zniżkujące zapowiadają zwykle wybiecie cen akcji w dół.



Rys. 4.11 Formacje cenowe – trójkąt zniżkujący oraz zwyżkujący

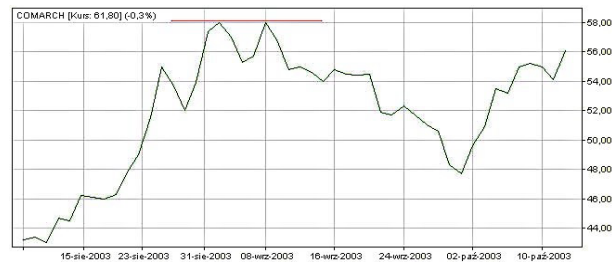
Trójkąty zwyżkujące zbudowane są z poziomej linii górnej, oraz dolnej skierowanej w górę. Zwyżkująca dolna linia mówi o zmniejszającym zaangażowaniu niedźwiedzi i rosnącej przewadze byków. Należy więc spodziewać się wzrostów i wybicia cen w górę.

**Formacje flag** – są specyficznym rodzajem czworokątów. Składają się z dwóch linii równoległe skierowanych w dół albo w górę. Istnieje zależność, która mówi, że wybicia z tej formacji następują zwykle w kierunku przeciwnym do jej kierunku nachylenia.



Rys. 4.12 Formacje cenowe – flaga wznosząca i opadająca

**Podwójne szczyty i dna** – to specyficzny rodzaj formacji cenowej. Formacja podwójne szczyty ma miejsce wtedy, gdy ceny zwyżkują do poziomu na którym ukształtował się poprzedni wierzchołek. Analogicznie formacja podwójnego dna występuje, gdy ceny spadają po raz drugi do poziomu poprzedniego dołka.



Rys. 4.13 Formacje cenowe – podwójny szczyt.

#### 4.2.4 Linie trendu

Trend jest to kierunek zmian w długim okresie, dzięki czemu eliminuje zjawiska nieregularne. Kursy walorów są podporządkowane trendom i związanymi z nimi zasadami. Wyszukując trend można znaleźć sygnały do kupna bądź sprzedaży. Na rynku giełdowym wyróżnia się trzy rodzaje trendów:

- wzrostowy, w którym ceny akcji kształtują się na coraz wyższym poziomie,
- spadkowy, w którym ceny akcji kształtują się na coraz niższym poziomie,
- horyzontalny (często nazywany brakiem trendu), w którym większa część zwyżek i zniżek kształtuje się na podobnym pułapie cenowym. Trend horyzontalny jest oznaką stabilizacji rynku, ale może też oznaczać fazę przejściową pomiędzy trendem wzrostowym a spadkowym lub odwrotnie.



Rys. 4.14 Linie trendu spółki INTERIA PL. Kolejno: horyzontalny, spadkowy, wzrostowy

### 4.3 Analiza fundamentalna

Szczególną rolę w analizie fundamentalnej, przy wyborze spółek, odgrywa analiza finansowa, która dzieli się na cztery podstawowe grupy technik inwestycyjnych.

Pierwszą z nich jest badanie bilansu spółki. Analiza ta zawiera się w określeniu prawidłowości finansowania działalności inwestycyjnej oraz sposobu finansowania środków trwałych, metody i prawidłowości finansowania zapasów bieżących, zdolności płatniczej spółki oraz stosunku należności do zobowiązań. Analiza bilansu określa dynamikę i możliwości rozwojowe spółki, a także jej wartość rynkową.

Drugim sposobem jest analiza zmian w kapitale obcym spółki. Wysokość kapitału obcego obrazuje stopień bezpieczeństwa finansowego, jakie posiada jednostka, informuje również o wielkości kapitału, który pozostanie wewnątrz jednostki po spłaceniu przez nią bieżących zobowiązań. Gdy kapitał obcy kształtuje się na wysokim poziomie świadczy to o wysokim ryzyku inwestycyjnym i niesie za sobą ewentualne niebezpieczeństwo dla spółki w razie niespłacenia zaciągniętych zobowiązań. Optymalną sytuacją jest stan, gdy kapitał obcy kształtuje się na stosunkowo niskim poziomie, co pozwala na stwierdzenie, że dana jednostka jest w dobrej kondycji finansowej a naruszenie jej pozycji wśród konkurencji jest mało prawdopodobne.

## 4.4 Analiza portfelowa

Analiza portfelowa dokonuje wyboru i zestawiania ze sobą odpowiednich akcji w celu obniżenia ryzyka inwestycyjnego (tzw. dywersyfikacja portfela). Istnieje wiele metod doboru spółek i ich wkładu do portfela. Dywersyfikacji można dokonać pośrednio na podstawie analizy fundamentalnej inwestując w spółki o różnym charakterze działalności. Do portfela można włączyć spółki należące do przeciwstawnych branż, bądź posiadających odmienne struktury kapitału. Istnieje też metoda polegająca na wykorzystaniu analizy technicznej, poprzez wybór kilku spółek, dla których istnieją duże szanse na wzrost notowań, jednakże należy pamiętać, że nigdy nie ma pewności poprawnej prognozy. Aby zapobiec ewentualnej stracie inwestorzy bazujący na analizie technicznej kupują, więc kilka akcji, o największym potencjale wzrostowym. W takiej sytuacji często zdarza się, że każda z prognoz została opracowana na podstawie różnych sygnałów technicznych.

## 4.5 Podsumowanie

W rozdziale zostały przedstawione trzy najważniejsze sposoby analizy rynku giełdowego.

Wybierając Analizę Techniczną do badania rynku giełdowego przez sieć neuronową należy zapewnić niezbędne rodzaje danych, ponieważ tylko wówczas sieć będzie w stanie nauczyć się charakterystycznych formacji i trendów. Dodatkowo uczenie sieci wskaźnikami wyprzedzającymi (np. oscylatory) zwiększy zdolności przewidywania zmian trendów.

Uczenie sieci neuronowych danymi spółki, która ma niewielką ilość notowań powoduje, że w zbiorze uczącym wystąpi niedobór większości charakterystycznych formacji, trendów. W rezultacie sieć nie będzie w stanie prawidłowo rozpoznawać podstawowych zależności.



---

# CZEŚĆ III REALIZACJA PRAKTYCZNA

„Jedynym źródłem wiedzy jest doświadczenie”

ALBERT EINSTEIN

Celem pracy było stworzenie narzędzia realizującego analizę i predykcję rynku giełdowego różnymi metodami w oparciu o wiedzę opisaną w poprzednich rozdziałach części teoretycznej. W ten sposób wykazano przydatność elementów sztucznej inteligencji w przewidywaniu rynku giełdowego. Podczas wykonywania doświadczeń za pomocą systemu skupiono się jednak w głównej mierze na wykorzystaniu sztucznych sieci neuronowych w połączeniu z algorytmami genetycznymi. Tym nie mniej narzędzie zaprojektowano tak aby było ono rozwojowe oraz aby jego architektura w łatwy sposób umożliwiała wykorzystanie innych metod. Umożliwia ono szeroką analizę i przeprowadzenie doświadczeń dla wszystkich walorów giełdowych w pełnym spektrum sesji giełdowych. Użytkownik ma możliwość sprecyzowania zadania postawionego systemowi (parametry związane z inwestycją giełdową ale także parametry związane z uczeniem sieci neuronowej, a także parametry dotyczące ewolucji genetycznej). Całą resztę parametrów (dobór optymalnej dającej najlepsze wyniki topologii sieci neuronowej) ma za zadanie dobrać opracowany algorytm genetyczny.

W rezultacie stworzony został system o nazwie AMADEUS (skrót w pośredniej formie pochodzi od *Another Data Mining System*) zawierający strukturę bazodanową i implementujący metody oparte na sztucznych sieciach neuronowych. Jego realizacja praktyczna została opisana w pierwszym rozdziale niniejszej części. Zawarty został w nim także opis problemów na jakie napotkano, oraz opis stworzonych i zastosowanych algorytmów w systemie.

W kolejnym rozdziale na podstawie przeprowadzenia szeregu doświadczeń w pracy ze zbudowanym systemem zostały wyciągnięte wnioski i spróbowano odpowiedzieć na pytanie kiedy sieci neuronowe dobrze spełniają swoje zadanie a kiedy stosowanie sieci zakończy się niepowodzeniem oraz jakie parametry wpływają na skuteczność predykcji.

---

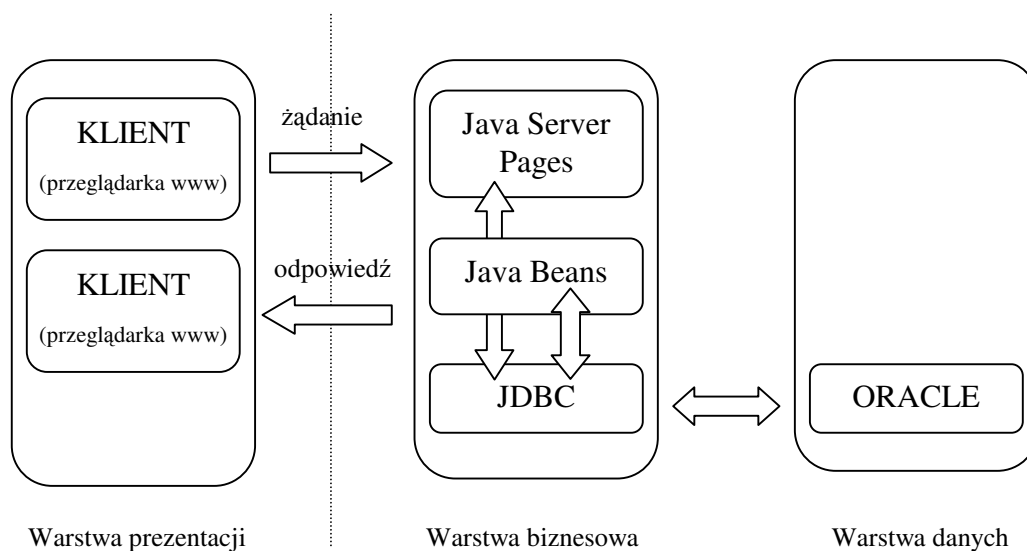
## 5 Architektura systemu Amadeus

System został wykonany w technologii J2EE (ang. *Java 2 Enterprise Edition*) przy wykorzystaniu serwera bazy danych Oracle 10g. Wykorzystanie technologii *webowej* i wykonanie systemu w technologii trójwarstwowej zapewnia możliwość szerokiego wykorzystania dowolnej przeglądarki internetowej i dowolnego komputera połączonych w sieć z serwerem aplikacji, na którym uruchomiono system. W ten sposób z wyników pracy systemu może korzystać większa liczba użytkowników niż gdyby system został zrealizowany jako jednostanowiskowa aplikacja typu „standalone”.

Wykorzystanie bazy danych Oracle zapewnia wysoką wydajność i bezpieczeństwo dla zgromadzonych danych historycznych, danych związanych z topologią konstruowanych sieci a także dla gromadzonych wyników prognoz.

### 5.1 Budowa systemu

Jak już wspomniano system jest wykonany w technologii trójwarstwowej. Schemat blokowy trójwarstwowej aplikacji J2EE został zaprezentowany poniżej.

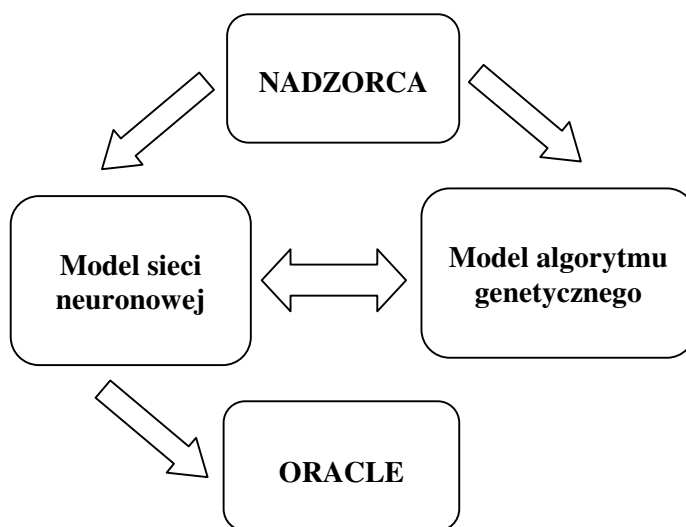


Rys. 5.1 Schemat blokowy trójwarstwowej aplikacji J2EE

System otrzymuje odpowiednie żądania wygenerowane przez klienta. Na ich podstawie w warstwie biznesowej dokonywane są stosowne działania (obliczenia, prezentacja danych), których bezpośrednim rezultatem jest odesłanie odpowiedzi. Do realizacji żądań konieczne mogą być dane zgromadzone w bazie danych.

Klientem może być dowolna przeglądarka internetowa. Rolę warstwy biznesowej pełni serwer aplikacji lub web kontener (ang. *web container*), w prezentowanym systemie rolę tą pełni Jakarta Apache Tomcat<sup>37</sup>, rolę warstwy danych realizuje serwer bazy danych Oracle 10g Express Edition<sup>38</sup>. W ten sposób wykorzystane w projekcie narzędzia są darmowe i ogólnodostępne.

W warstwie biznesowej można wyróżnić trzy główne moduły funkcjonalne będące zbiorem klas implementujących zadania modelu sieci neuronowej, algorytmu genetycznego a także zarządzania tymi procesami oraz dostarczania danych z warstwy danych.



Rys. 5.2 Podstawowe moduły systemu Amadeus

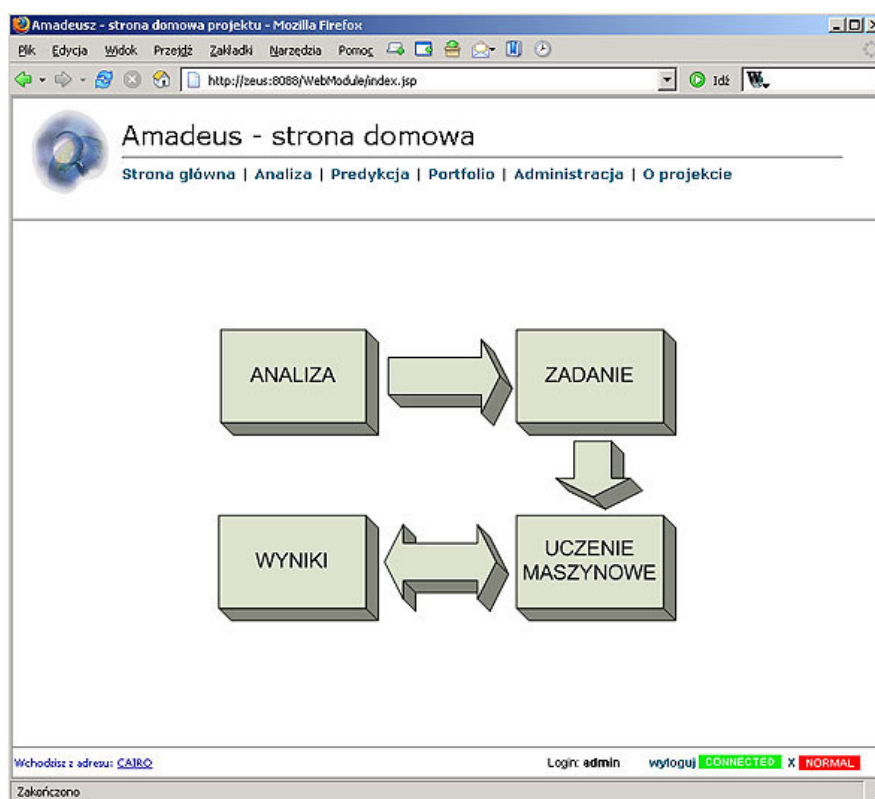
System realizuje cztery podstawowe funkcjonalności:

- analiza danych (wykresy historycznych notowań, prezentacja informacji o spółkach, prezentacja danych dowolnej sesji),
- zadania (umożliwia zdefiniowanie zadania jakie system będzie realizował),
- uczenie sieci neuronowej optymalizowanej algorytmem genetycznym,
- wyniki (moduł realizujący możliwość predykcji notowań na podstawie przygotowanych modeli sieci neuronowych uzyskanych podczas uczenia).

Powyższe funkcjonalności zostały przedstawione na głównej stronie systemu w postaci diagramu przepływu.

<sup>37</sup> <http://tomcat.apache.org/index.html>

<sup>38</sup> <http://www.oracle.com/technology/products/database/xe/index.html>



Rys. 5.3 Strona główna systemu Amadeus

## 5.2 Definiowanie i nadzorowanie oraz wykonywanie zadań

W poniższym podrozdziale zostanie omówiony proces interakcji użytkownika z systemem od momentu zdefiniowania zadania aż po uzyskanie żądanych wyników działania. W ten sposób zostanie zaprezentowany krótki przewodnik obsługi programu.

Pierwszym etapem pracy z systemem jest zdefiniowanie zadania, jakie zamierza się dzięki niemu rozwiązać. System wyróżnia cztery rodzaje parametrów zadania: ogólne, związane z siecią neuronową, związane z algorytmem ewolucyjnym (rys 5.4) oraz określające spółki giełdowe wybrane do analizy w tymże zadaniu (rys. 5.5).

Pierwszy parametr ogólny definiuje unikalną nazwę zadania, priorytet jego wykonania (pilne, normalne, wykonywane w ostatniej kolejności) oraz okres prognozy. W ten sposób można stworzyć kilka zadań o różnym okresie prognozy nadając im różne priorytety wykonania (ponieważ zasoby obliczeniowe są ograniczone nie zawsze istnieje gwarancja wykonania wszystkich zadań w określonym terminie) identyfikując je w prosty sposób po nadanej nazwie.

Pozostałe dwa rodzaje parametrów wymagają szerszego omówienia. Definiują one bowiem parametry pracy sieci neuronowej i algorytmu ewolucyjnego. Podzielono je na parametry niezbędne do ustawienia (są one podpowiadane wartościami domyślnymi, poprawnymi

aczkolwiek nie zawsze najbardziej efektywnymi) oraz opcjonalne dzięki którym można poprawić jakość wykonywanych działań (zostały one zaznaczone na rys. 5.4 odpowiednio jako czerwone i zielone gwiazdki umieszczone przy nazwie parametru)

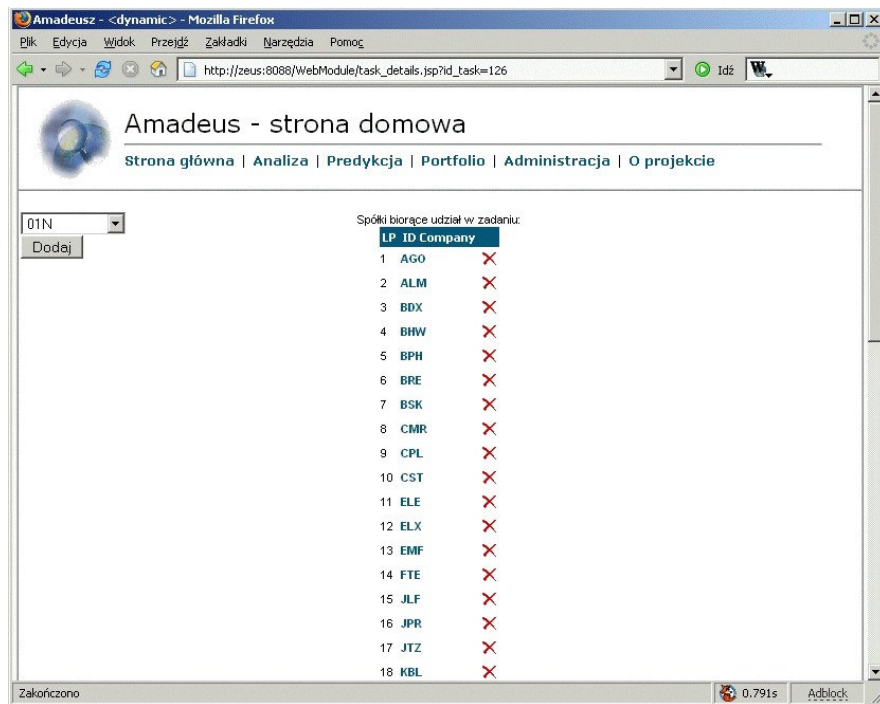
W ten sposób do niezbędnych parametrów zalicza się m.in.: okres początkowy treningu, ilość wzorców, współczynnik uczenia, wartość współczynnika momentum a także te związane z parametrami algorytmu genetycznego a mianowicie: ilość osobników w pokoleniu, ilość pokoleń oraz ilość punktów krzyżowania.

Parametrami opcjonalnymi są natomiast: parametry związane z testowaniem, inne algorytmy uczenia, początkowa wartość współczynnika uczenia a także jego zmiana w czasie.

Parametry zadania	Parametry sieci neuronowej	Parametry algorytmu ewolucyjnego
Okres predykcji* 60	Ilość epok* 50	Ilość osobników w pokoleniu* 6
Priorytet 0	Ilość wzorców* 1000	Ilość pokoleń* 6
Opis PRACA dług	Przesunięcie sesji początkowej 100	Ilość krzyżowań* 1
	Ilość epok douczania* 0	
	Start DA* 0	
	DA step* 0	
	DA rate* 0	
	LR* 0.2	
	LR start* 0.0	
	LR rate* 0.0	
	Momentum* 0.8	
	Początek uczenia* 1200	
	Początek testowania* 200	
	Ilość wzorców testowania* 200	
	Algorytm uczenia* 0	

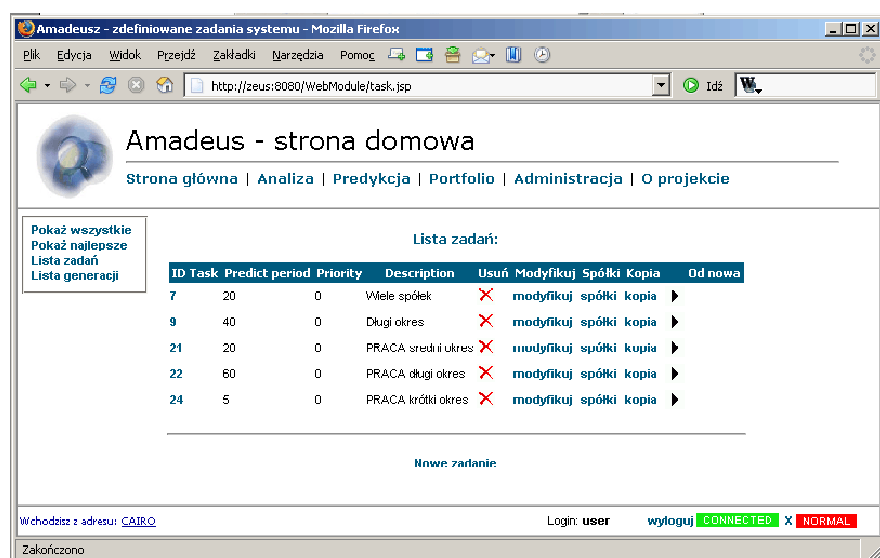
Rys 5.4 Menu systemu Amadeusz – definiowanie zadań.

Gdy już te parametry zadania zostaną zdefiniowane, należy jeszcze określić, dla których spółek giełdowych będzie wykonywane nauczanie sieci neuronowej. Ze względu na kryterium kosztowe, jakim jest czas obliczeń, nie warto poddawać uczeniu sieci danych związanych ze spółkami, które podczas wstępnej analizy wydają się być nieatrakcyjne (np. w wyniku analizy fundamentalnej) lub wykazują spore ryzyko inwestycji (analiza portfelowa), czy też działają w branży chwilowego ryzyka inwestycyjnego.



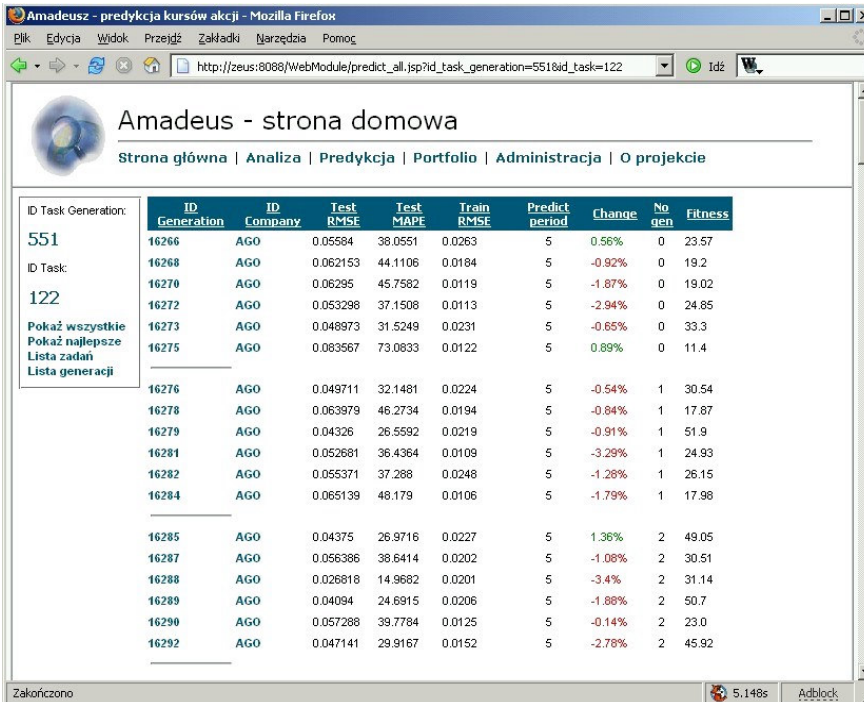
Rys 5.5 Menu systemu Amadeusz – definiowanie spółek

Poprawnie zdefiniowane zadanie zostanie umieszczone na liście zadań (rys. 5.6), na której można je modyfikować, usunąć oraz uruchomić jego wykonanie. W danej chwili czasowej może być wykonywane tylko jedno zadanie, należy jednak zauważyć, że rozdzielając zadania na osobne komputery zwiększa się wydajność systemu. W ten sposób można zbudować bardzo prosto rozproszony system w którym każdy z komputerów zajmowałby się jedynie powierzonym mu zadaniem.



Rys. 5.6 Lista zadań systemu Amadeusz

Cały czas w trakcie uczenia można na bieżąco kontrolować wyniki, obserwując po każdym treningu wyniki uczenia (rys. 5.7) kolejnego osobnika. Wśród wyników można zaobserwować błędy testowania oraz uczenia a także procentową zmianę wartości notowań dla danej spółki dla zbioru testowego. Dodatkowo można zaobserwować pozycję: *fitness* określającą przystosowanie osobnika (szerzej omówione w [5.4]).



Amadeusz - predykcja kursów akcji - Mozilla Firefox

Strona główna | Analiza | Predykcja | Portfolio | Administracja | O projekcie

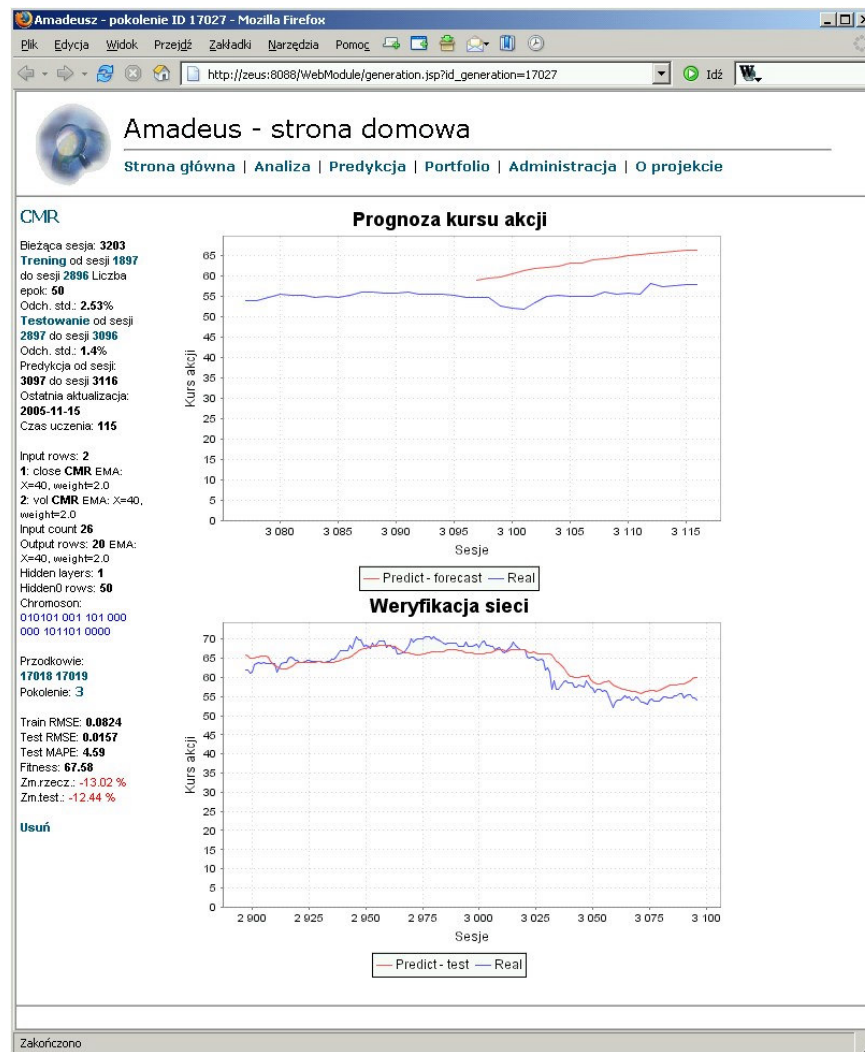
ID Task Generation: 551  
ID Task: 122  
Pokaż wszystkie  
Pokaż najlepsze  
Lista zadań  
Lista generacji

ID Generation	ID Company	Test RMSE	Test MAPE	Train RMSE	Predict period	Change	No gen	Fitness
16266	AGO	0.05584	38.0551	0.0263	5	0.56%	0	23.57
16268	AGO	0.062153	44.1106	0.0184	5	-0.92%	0	19.2
16270	AGO	0.06295	45.7582	0.0119	5	-1.87%	0	19.02
16272	AGO	0.053298	37.1508	0.0113	5	-2.94%	0	24.85
16273	AGO	0.048973	31.5249	0.0231	5	-0.65%	0	33.3
16275	AGO	0.083567	73.0833	0.0122	5	0.89%	0	11.4
16276	AGO	0.049711	32.1481	0.0224	5	-0.54%	1	30.54
16278	AGO	0.063979	46.2734	0.0194	5	-0.84%	1	17.87
16279	AGO	0.04326	26.5592	0.0219	5	-0.91%	1	51.9
16281	AGO	0.052681	36.4364	0.0109	5	-3.29%	1	24.93
16282	AGO	0.055371	37.288	0.0248	5	-1.26%	1	26.15
16284	AGO	0.065139	48.179	0.0106	5	-1.79%	1	17.98
16285	AGO	0.04375	26.9716	0.0227	5	1.36%	2	49.05
16287	AGO	0.056386	38.6414	0.0202	5	-1.08%	2	30.51
16288	AGO	0.026818	14.9682	0.0201	5	-3.4%	2	31.14
16289	AGO	0.04094	24.6915	0.0206	5	-1.88%	2	50.7
16290	AGO	0.057288	39.7784	0.0125	5	-0.14%	2	23.0
16292	AGO	0.047141	29.9167	0.0152	5	-2.78%	2	45.92

Zakończono 5.148s AdBlock

Rys. 5.7 Lista parametrów osobników dla kolejnych iteracji

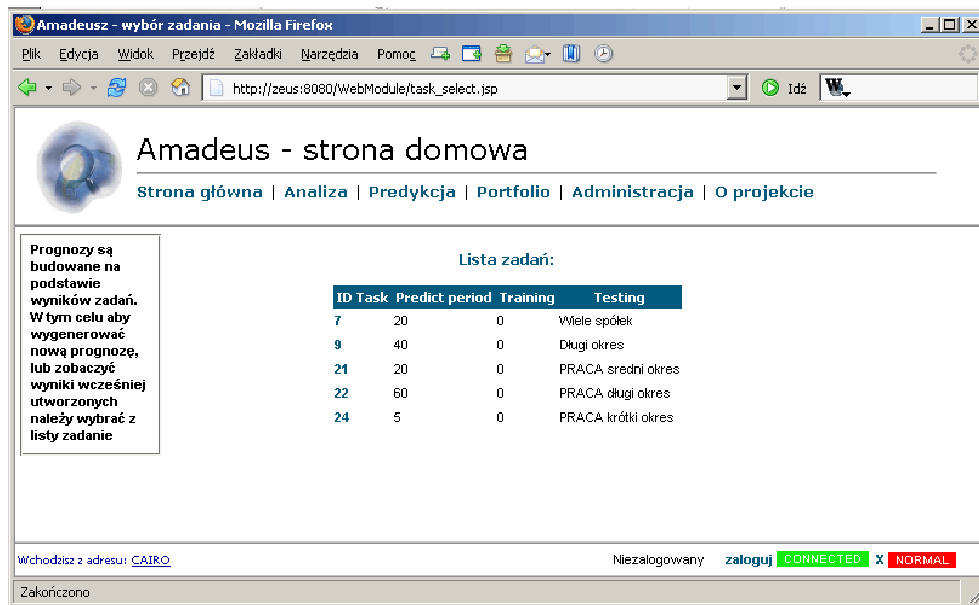
Dla każdego z osobników istnieje możliwość analizowania wyniku testowania sieci (rys. 5.8). Dodatkowo można odczytać wartości ważniejszych parametrów definiujących proces uczenia (takie m.in. jak topologia sieci, zapis binarny chromosomu, informacja o przodkach etc.) oraz zobaczyć na wykresie przebieg predykowanych wartości dla zbioru walidacyjnego (górny wykres) a także testowego (wykres dolny).



Rys.5.8 Szczegółowe parametry osobnika

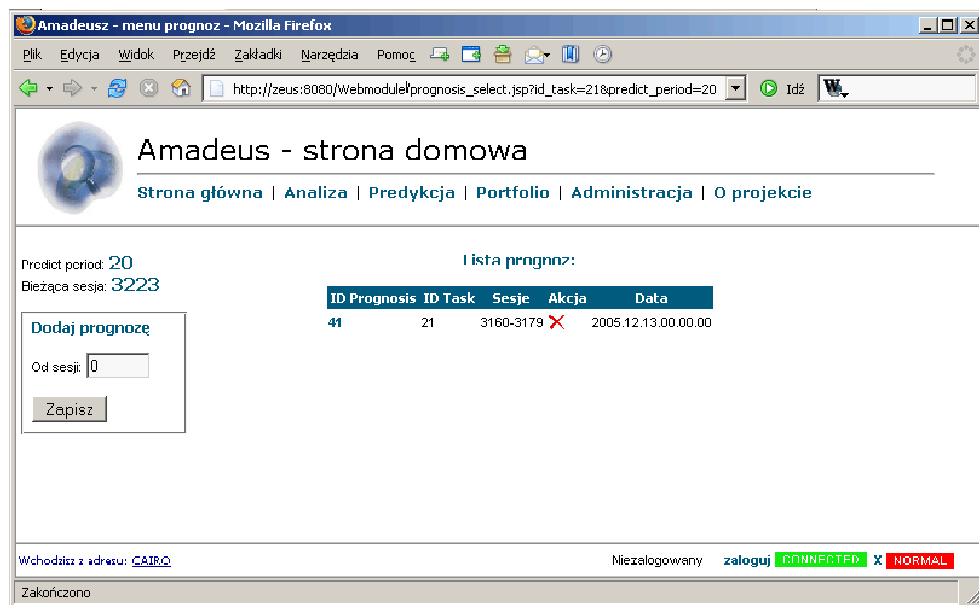
Po zakończeniu uczenia wszystkich sieci związanych z zadaniem jest ono widoczne na liście prognoz (rys. 5.9) do wykonania. Zakłada się, że raz nauczona sieć zachowuje swoje zdolności predykcyjne dla danej spółki przez pewien okres zależny m.in. od okresu i przebiegu wartości zbioru uczącego. W sytuacji gdy błędy predykcji będą wzrastać konieczne jest przeprowadzenie uczenia również na aktualniejszych danych jeżeli np. nowe zachowania giełdowe nie mają swojego odzwierciedlenia w historii notowań.





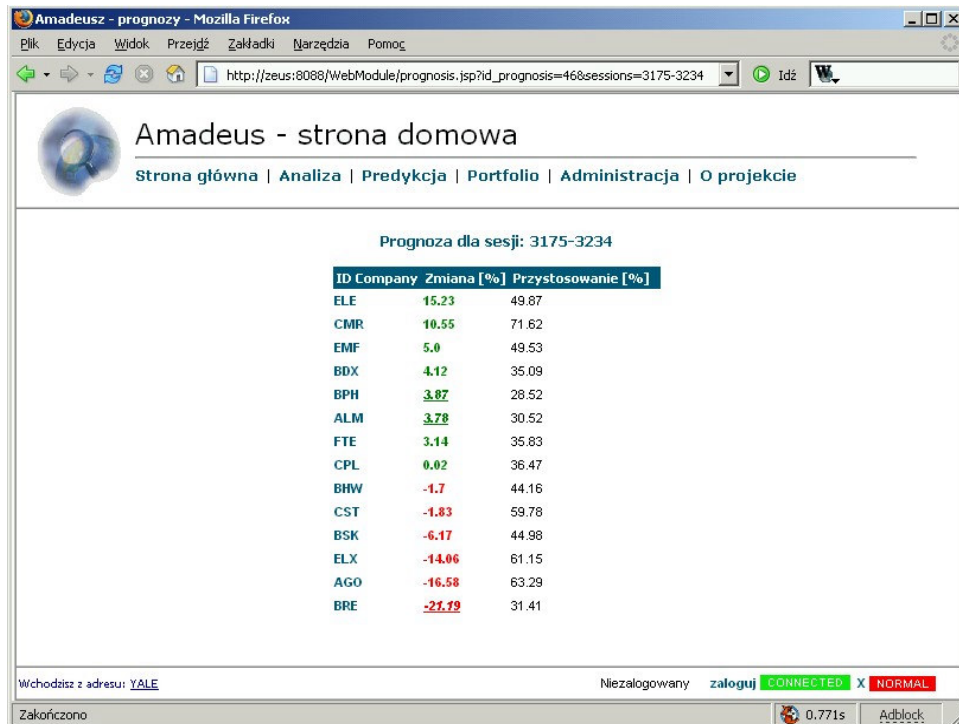
Rys. 5.9 Wybór zadania w procesie tworzenia prognoz

Można teraz wygenerować prognozy dla tych spółek, dla których przeprowadzono uczenie sieci. Przy stałym okresie predykcji modyfikowalnym parametrem jest początkowa sesja predykcji (rys 5.10). Predykcje nie mogą nakładać się czasowo.



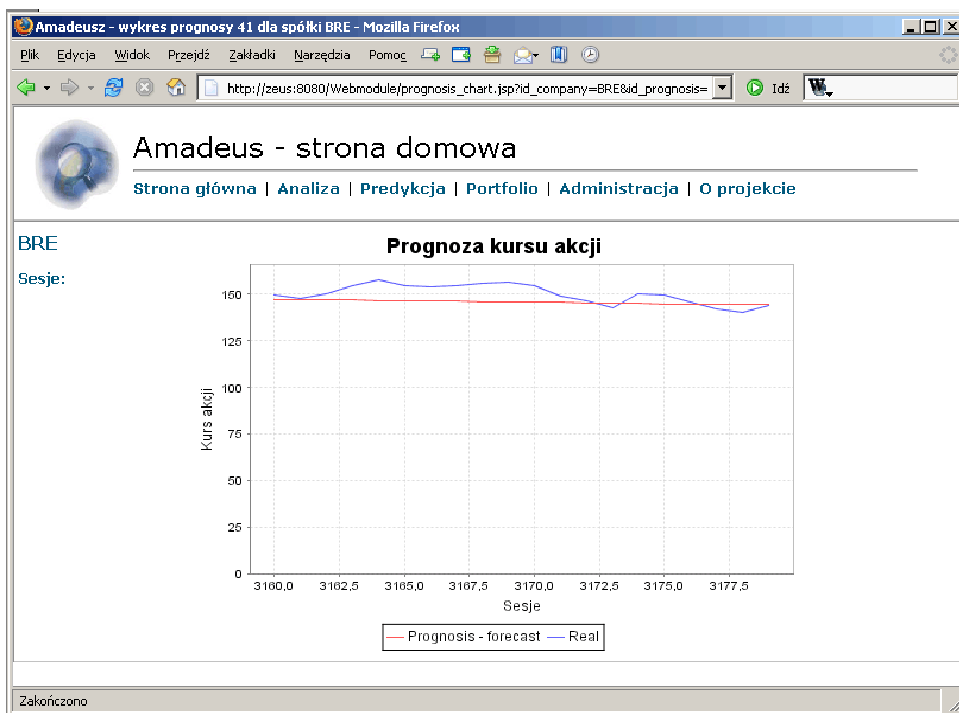
Rys. 5.10 Tworzenie prognozy

Podkreślona zmiana procentowa na liście prognoz oznacza niskie przystosowanie osobnika a co za tym idzie wyższe prawdopodobieństwo zaistnienia niepoprawnej prognozy.



Rys. 5.11 Wygenerowana prognoza dla spółek biorących udział w zadaniu

Wybierając za początkowy okres predykcji sesję notowaną  $k$  sesji temu (gdzie  $k$  to ilość dni predykcji) można porównać wygenerowaną prognozę z wartością rzeczywistą kursu akcji.



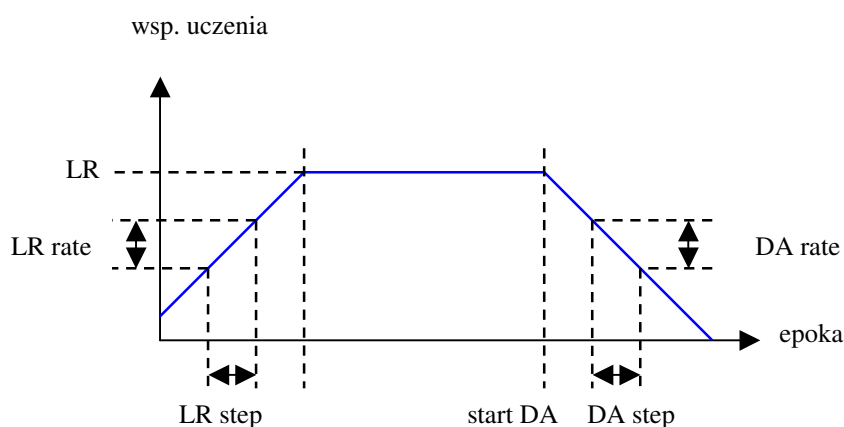
Rys. 5.12 Graficzna interpretacja prognozy (kolor czerwony)

### 5.3 Sieć neuronowa

W projekcie zaimplementowano sieć typu MLP [2.6]. Większość parametrów sieci jest modyfikowalna na etapie definiowania zadania (rys 5.4) pozostałe wartości są modyfikowane w trakcie ewolucji genetycznej. Dla warstwy wyjściowej do parametrów modyfikowalnych należą typy pól oraz ilość neuronów. Natomiast dla warstwy ukrytej algorytm genetyczny będzie starał się optymalizować ilość warstw a także ilość neuronów dla każdej z warstw ukrytych. Innymi słowy modyfikacji ulega topologia sieci. Doświadczalnie przyjęto ograniczenie, że warstw ukrytych może być jedna lub dwie.

Ze względu na indywidualny charakter każdej ze spółek, każdy walor będzie posiadał osobną architekturę sieci neuronowej. Spółki przyjmują różne wartości notowań, dodatkowo przynależność do różnych branż powoduje różne zachowania kursów w tym samym czasie, czego dowodem jest to, że spółki posiadają bardzo szerokie spektrum wartości korelacji w stosunku do notowań WIG. Istnieje wówczas trudność realizacyjna takiego modelu sieci, który miałby przynieść zadowalające wyniki dla wszystkich spółek.

Wartość współczynnika uczenia może być stała lub w sposób adaptacyjny dobierana w trakcie uczenia. Na samym początku, gdy ustalają się wektory wag neuronów korzystnie jest przeprowadzać uczenie z małym współczynnikiem, stopniowo go powiększać i po kilkunastu-kilkudziesięciu epokach uczyć z docelowym współczynnikiem.



Rys. 5.13 Parametry związane z procesem uczenia

W końcowej fazie uczenia, gdy błąd zaczyna rosnąć, współczynnik uczenia powinien dynamicznie maleć a tym samym odległość wektora wag nie będzie powiększała odległości tak znacznie jak dla pełnej wartości współczynnika.

W pierwszej fazie algorytmu ewolucyjnego zostanie stworzona populacja zawierająca większą ilość sieci neuronowych, które będą ze sobą rywalizowały w procesie ewolucji. Sieci będą

uczone określoną ilość epok (tak, aby nie nastąpiło zjawisko przetrenowania) a następnie po zakończeniu rywalizacji (ewolucji) i wyłonieniu sieci o najwyższej jakości będzie ona kontynuować uczenie do chwili, w której nie zacznie zwiększać się jej błąd testowania.

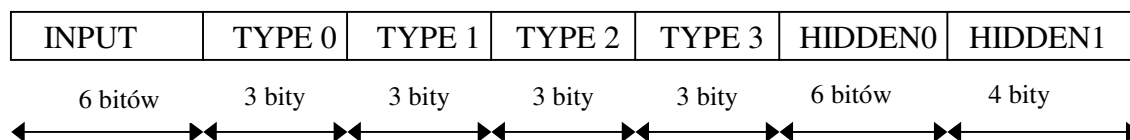
Dodatkowo do zbioru treningowego dołączone będą szumy o określonej amplitudzie dobrane doświadczalnie zwiększając w ten sposób właściwości generalizacji przy bardzo długich zbiorach treningowych przy których może zachodzić bardzo duża różnica pomiędzy wartościami na początku zbioru i na jego końcu.

## 5.4 Algorytm genetyczny

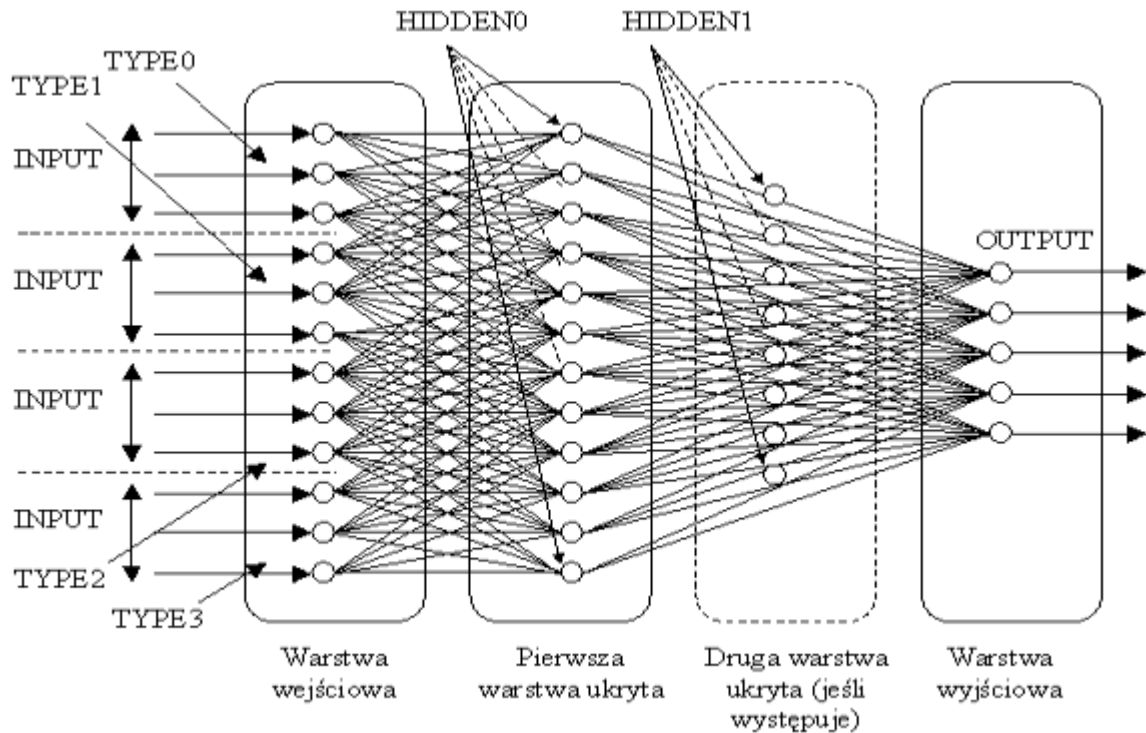
Moduł ten ma za zadanie optymalizację topologii zastosowanej sieci neuronowej. Posiada on jedynie trzy parametry modyfikowalne na poziomie zadania. Są nimi: liczebność populacji, ilość pokoleń, ilość punktów krzyżowania.

Zgodnie z teorią algorytmów genetycznych, praca tego modułu rozpoczyna się od wygenerowania populacji początkowej. Ponieważ celem jest optymalizacja sieci ze względu na parametry warstwy wejściowej (typ i ilość) oraz warstwy ukrytej (liczebność warstw i neuronów dla każdej z nich) parametry te zostaną zainicjalizowane w sposób losowy.

W trakcie ewolucji powyższe parametry muszą mieć charakter binarny tak, aby móc stworzyć chromosom o poniższej strukturze:



Rys. 5.14 Budowa chromosomu



Rys 5.15 Struktura sieci na podstawie parametrów pobranych z chromosomu.

Pole *INPUT* określa ilość neuronów dla jednego wejścia (*TYPE<sub>x</sub>*), ponieważ jest ono 6 bitowe może zawierać wartości od 0 do 63. Założono wstępnie minimalną ilość równą 5 stąd zakres docelowy wynosi 5-68. Podobnie rozwiązano zakresy dla warstw ukrytych, gdzie na podstawie wzoru:

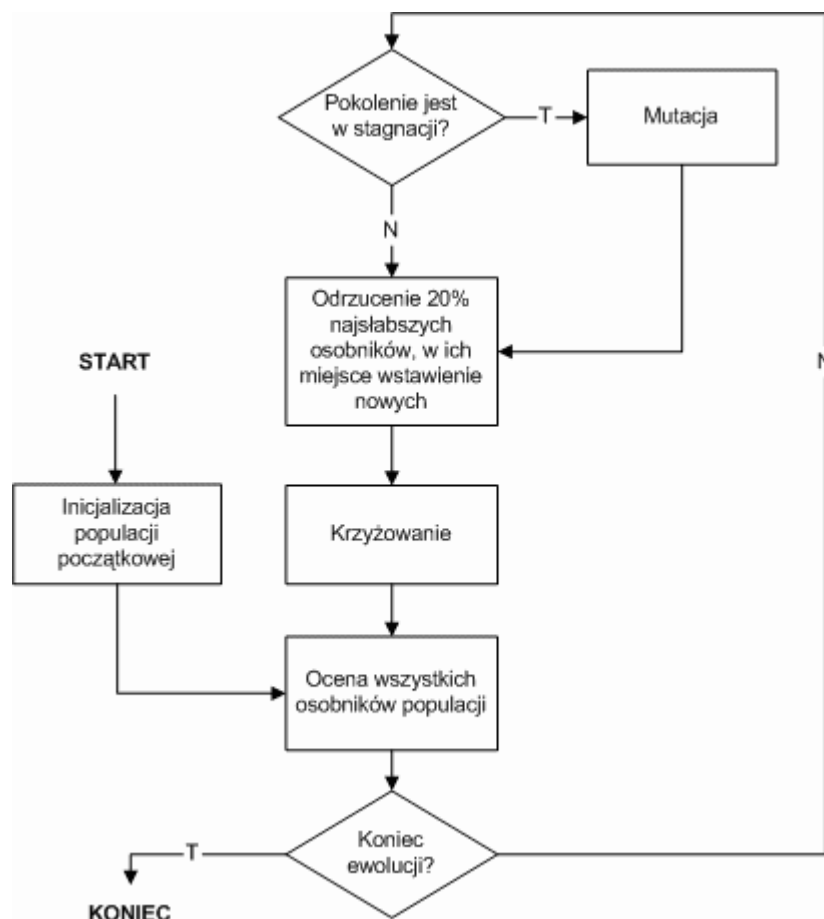
$$N_h = \sqrt{N_i N_o} \quad (5.1)$$

określającego optymalną wartość neuronów w warstwie ukrytej dla  $N_i = 5$  oraz  $N_o = 5$  (minimalnych wartości, dla jakich skonstruowano sieć) wartość  $N_h = 5$ . Jeżeli wartość genów odpowiedzialnych za drugą warstwę ukrytą wynosi zero oznacza to, że warstwy tej nie uwzględnia się w implementacji i sieć MLP posiada tylko trzy warstwy.

Pozostała jeszcze sprawa typów danych podawanych na wejście sieci. Na pierwsze wejście nie biorące udziału w optymalizacji będzie podawana zawsze wartość, którą będzie się predykować (najczęściej będzie to cena zamknięcia *close*). Kolejnym etapem jest losowanie ilości wejść z zakresu 1-5. Wylosowanie wartości 1 oznacza, że na wejście sieci podawane będzie jedynie uprzednio zdefiniowane pole. W przypadku wylosowania wartości większej od 1, konieczne jest wylosowanie, jakiego typu to będą dane. Wcześniej należy zdefiniować „słownik” dostępnych kombinacji pól zamieszczony w poniższej tabeli:

Wartość pola	Typ danych
000	puste
001	close
010	close – open
011	max – min
100	close - high
101	close - low
110	vol
111	(high-low)*vol

W tym momencie system może przejść do realizacji turnieju, postępowanie dla każdego pokolenia wyjaśnia poniższy diagram.



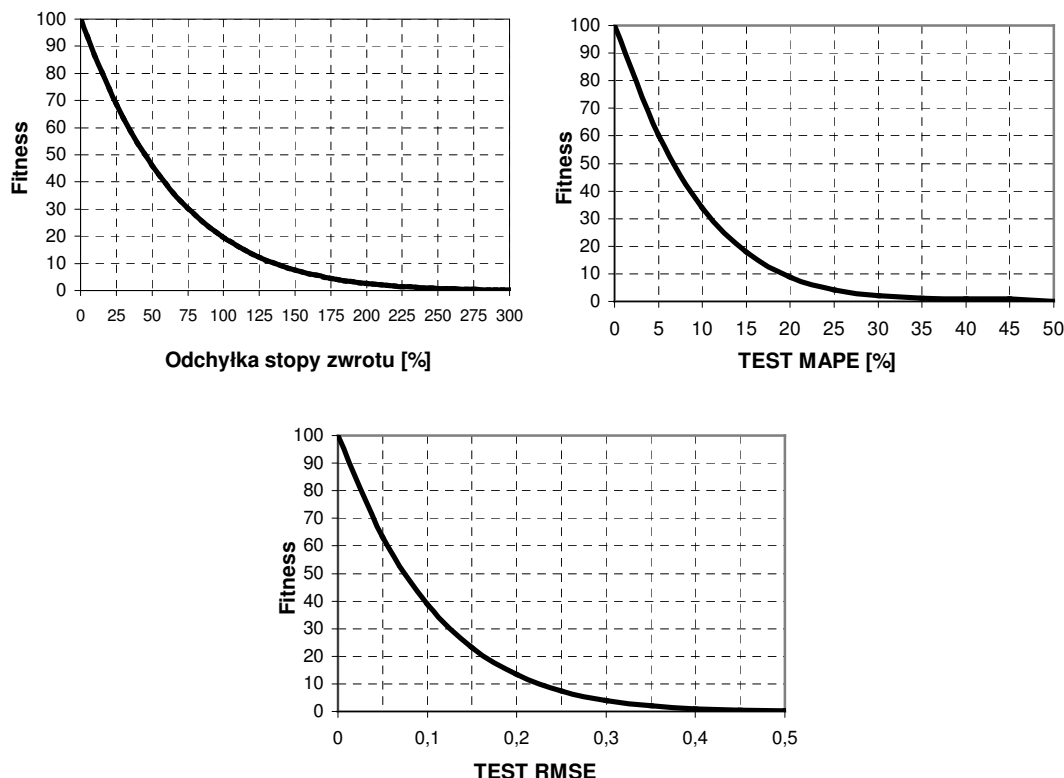
Rys. 5.16 Diagram realizacji turnieju w procesie ewolucyjnym

Tyle razy ile osobników w pokoleniu, dokonaj treningu sieci (zbudowanej na podstawie danych zakodowanych w osobniku). Po zakończeniu uczenia oblicz funkcje przystosowania *fitness* dla pojedynczego osobnika a także sumę dla wszystkich osobników.

Wartość funkcji przystosowania składa się z trzech składowych:

- składowej mówiącej czy stopa zysku w trakcie testowania sieci jest zbliżona do wzorca,
- wartości błędu MAPE w okresie testowania,
- wartości błędu RMSE w okresie testowania.

$$fitness = 0.5(fitness_{turnrate}) + 0.25(fitness_{test\_rmse}) + 0.25(fitness_{test\_mape}) \quad (5.2)$$

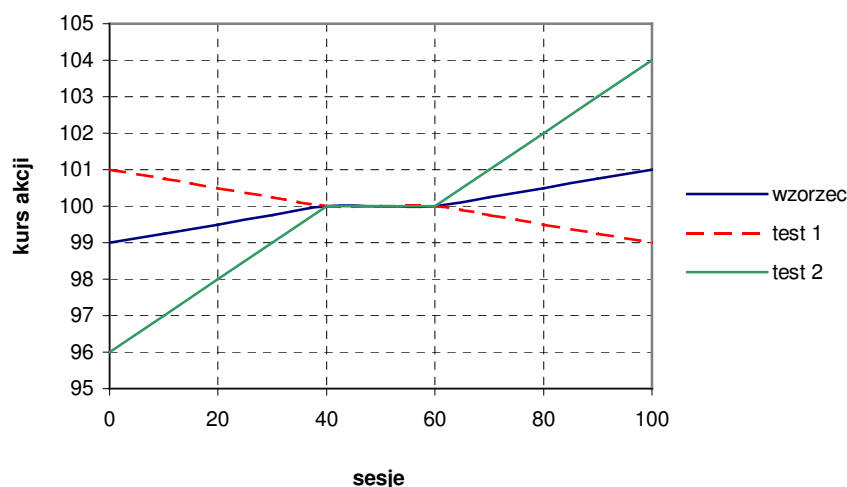


Rys. 5.17 Krzywa funkcji dopasowania względem odchyłki stopy zwrotu i błędów testowania

Każda z tych częściowych funkcji przystosowania przyjmuje wartości z zakresu [0, 100]. Należy zwrócić uwagę, że charakterystyka funkcji przystosowania jest poprowadzona tak, aby silniej niż liniowo promować osobników najlepszych, a jeżeli wyniki osiągną niezadowalające rezultaty wówczas, aby zminimalizować możliwość przejścia do puli rodzicielskiej.

Rozpatrując np. częściową funkcję przystosowania na podstawie odchyłki stopy zwrotu (rys 5.10). Wyniki, które można zaakceptować (do 20%) są silnie promowane, jednak aż do odchyłki 300% funkcja przyjmuje wartości powyżej zera. Dzieje się tak, dlatego, że podczas prognoz interesuje nas co prawda bardzo dokładna ocena stopy zwrotu za okres, ale równie ważny jest kierunek postępujących zmian. Miarą rzetelności oceny sieci będzie wartość przystosowania, jednakże analizując rys 5.17 odchyłki od wzorca przebiegów *test1* i *test2* są takie same, jednak ich kierunek przeciwny. W praktyce oznaczałoby to, że prognoza oparta na

*test1* byłaby błędna. Algorytm powinien silniej promować spółki o zachowaniu podobnym do *test2*.



Rys 5.18 Przypadek niewielkiej rozbieżności wartości o przeciwnym znaku

Do tego celu stosuje się funkcję kary, wartość funkcji dostosowania powinna być tak dobrana, aby dany osobnik nie zdominował pokolenia lub też, aby w ogóle nie był reprodukowany.

W projekcie wykorzystano selekcję proporcjonalną z wprowadzeniem nowych osobników co każdą iterację. Na podstawie wyniku *fitness* wyznaczana jest ilość kopii danego osobnika przechodzącego do dalszego turnieju. Jeżeli na skutek zaokrągleń suma osobników przechodzących do następnego pokolenia różni się od zdefiniowanej liczebności populacji, wówczas w zależności od wyniku do populacji dołącza się nowego osobnika lub zmniejsza o jeden liczbę kopii dla osobnika najslabszego.

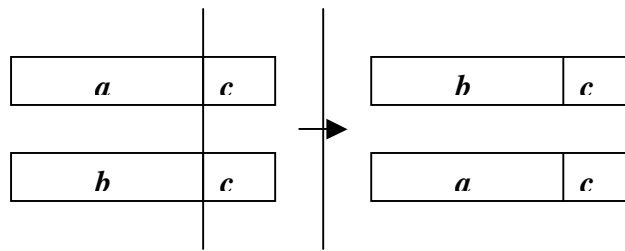
Należy zauważyć, że funkcja przystosowania może składać się z innych składników. Obok błędów testowania, odchyłki w prognozie, może to być np. czas, wartość zyskanych/straconych pieniędzy w wyniku inwestycji na podstawie prognozy etc. W zadaniu założono predykcję stóp zwrotu i dlatego też odchyłka od prognoz stanowi ważny czynnik. Błędy testowania są dodatkowym czynnikiem zwiększającym jakość poprawnej prognozy.

Kolejną czynnością jest operacja krzyżowania osobników z puli rodzicielskiej.

Operacja krzyżowania w zastosowanym algorytmie wymaga dokładniejszego opisu. W zależności od zdefiniowanej uprzednio ilości punktów krzyżowania dokonuje się krzyżowania jedno lub wielopunktowego. Specjalny algorytm zapobiega krzyżowaniu elementów takich samych (np. w mocno zdominowanej populacji osobnikami odznaczającymi

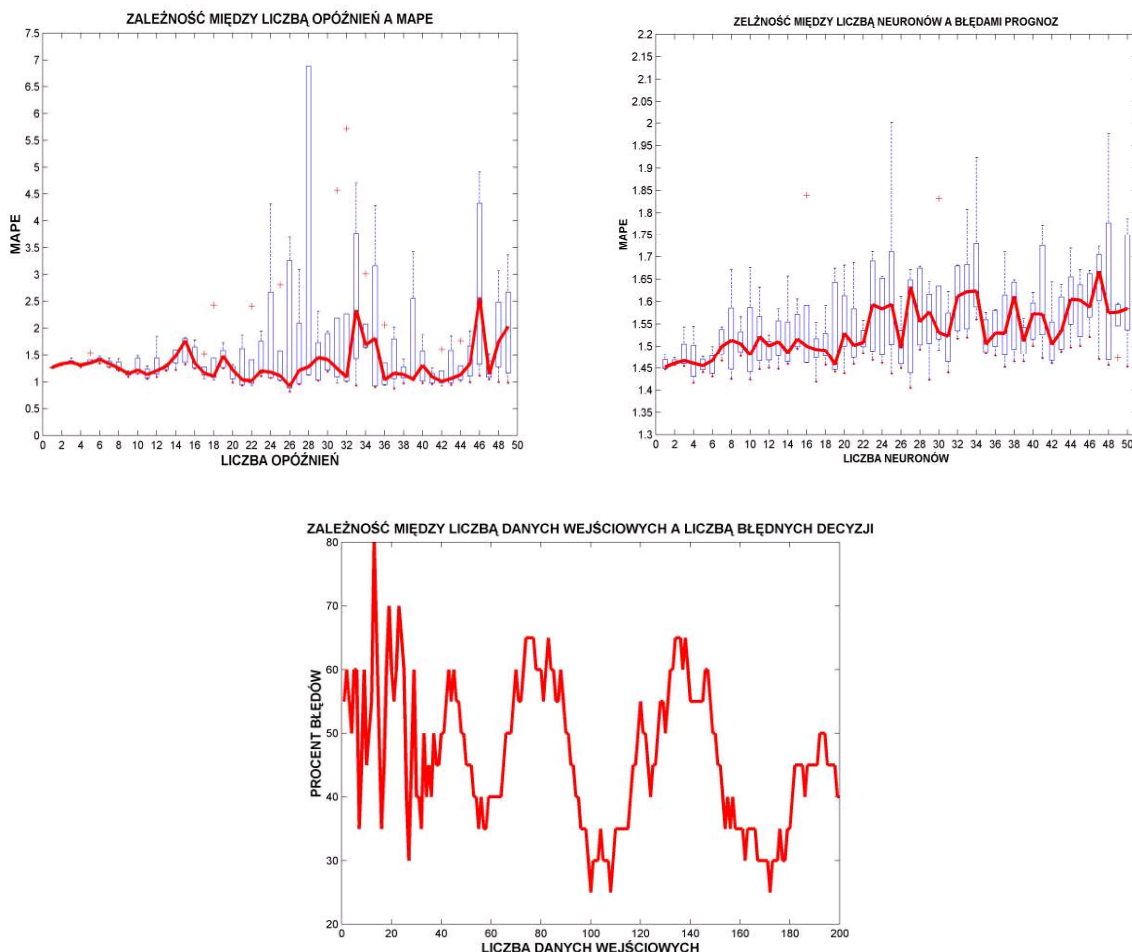


się takim samym schematem). Mechanizm dba także o to, aby wynik krzyżowania doprowadzał do nowych rozwiązań uniemożliwiając zachodzenie sytuacji przedstawionej na rysunku 5.19.



Rys. 5.19 Niekorzystna postać krzyżowania

Taka sytuacja byłaby nieefektywna ze względu na czas poświęcony ponownie na trening sieci o identycznych parametrach. Dodatkowo po każdym krzyżowaniu dokonuje się sprawdzenia czy wartości *fitness* osobników są zbliżone czy też mocno odchylone. Idealnie byłoby, gdyby z kroku na krok maksymalna średnia odchylenia rosła. Nie zawsze jest to możliwe.

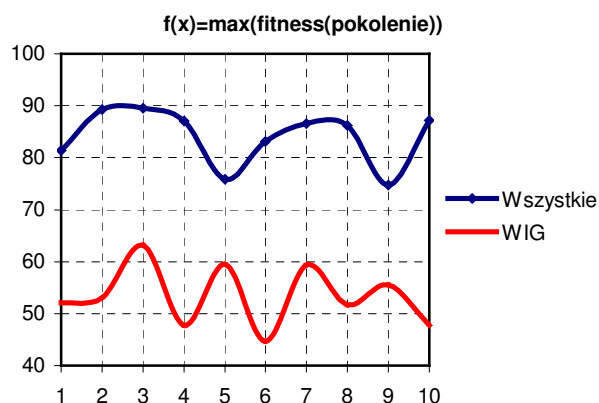


Rys. 5.20 Wpływ parametrów sieci neuronowej na wynik prognozy<sup>39</sup>

<sup>39</sup> Rysunek zaczerpnięty z prac badawczych dr K. Najman z Uniwersytetu Gdańskiego  
<http://panda.bg.univ.gda.pl/~Najman/sn/sn.htm>

Wykorzystując wyniki pracy naukowej dr Krzysztofa Najmana [Najman] dotyczącej wpływu parametrów sieci neuronowej (liczba danych wejściowych, opóźnienia, ilości neuronów) na wynik prognozy (rys 5.20) można zauważyć, że na wykresach dla każdego z trzech analizowanych przypadków istnieją ekstrema lokalne.

Zatem w trakcie turnieju realizowane są dwa zadania: poszukiwanie ekstremum globalnego i polepszenie uzyskanych wyników. Zastosowany algorytm nie zakłada, że kolejne pokolenie będzie reprezentować lepsze wyniki, może się tak zdarzyć, że będzie ono miało dość zmienny charakter (rys 5.14).



Rys. 5.21 Zależność wartości najlepszego przystosowania w pokoleniu względem kolejnych iteracji algorytmu genetycznego

Należy także przeciwdziałać zjawisku stagnacji w sytuacji, gdy wyniki *fitness* są dalekie od oczekiwanego. Dlatego po każdym turnieju, dokonywany jest pomiar odchylenia standardowego

$$s = \sqrt{\frac{1}{N-1} \sum_{t=1}^N (R_t - R)^2} \quad (5.3)$$

gdzie:

$R$  – średnia wartość *fitness* w pokoleniu,

$R_t$  – wartość *fitness*  $t$ -osobnika w pokoleniu,

$N$  – liczebność populacji.

Od wyniku  $s$  uzależnia się współczynnik prawdopodobieństwa mutacji, im niższe  $s$  tym konieczność mutowania chromosomu jest wyższa, ponieważ populacja osiąga bardzo podobne wyniki i warto sprawdzać czy wyprowadzenie ich z takiego stanu „równowagi” nie przyniesie wymiernych korzyści. Jeżeli odchylenie jest duże, wówczas na etapie ewolucji słabsze osobniki będą usuwane a wzmacniane te o najsilniejszym przystosowaniu.

W takim przypadku mutowanie genów wydaje się być bezcelowe a nawet szkodliwe.

Mutowanie chromosomów należy zatem przeprowadzać w końcowym etapie ewolucji, kiedy zachodzi prawdopodobieństwo zajścia stagnacji, czyli sytuacji kiedy kolejne rozwiązania przyjmują podobną postać.

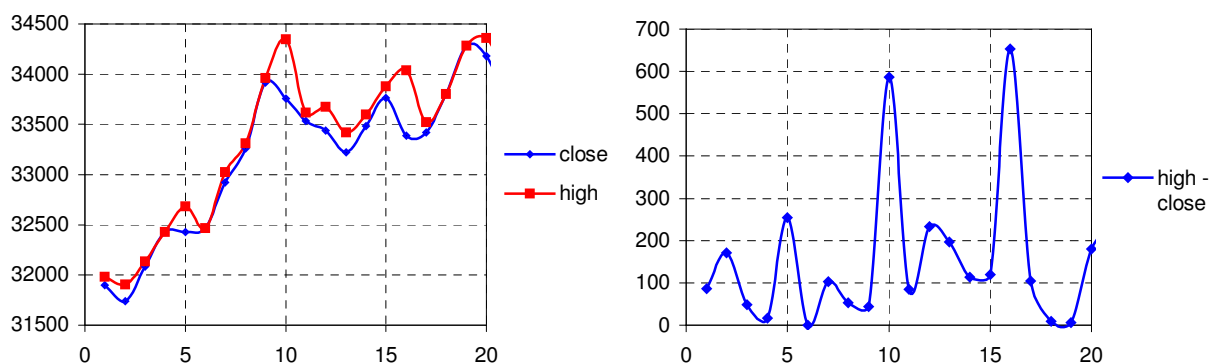
## 5.5 Dobór danych wejściowych

Bardzo duża część sukcesu zależy od „jakości” danych. Przeprowadzając *preprocessing* danych zgodnie z wytycznymi rozdziału [1] otrzymuje się w rezultacie „czyste” dane. Kolejnym zadaniem jest podjęcie decyzji o tym, jakie dane będą podawane na wejście sieci neuronowej.

Logicznym jest, iż jeżeli zamierza się predykować kurs zamknięcia takową wartość należy dostarczać w trakcie uczenia sieci. Z analizy technicznej wiadomo natomiast, że nie jest to wystarczająca ilość danych, aby sieć była w stanie poprawnie odwzorować zachowania nauczone z przeszłości.

Jak zostało to już wcześniej zasygnalizowane algorytm genetyczny ma na celu wytypowanie najbardziej optymalnej topologii sieci neuronowej. Jako topologię należy rozumieć nie tylko strukturę wewnętrzną sieci ale także jej wejścia.

W danych historycznych zawarta jest informacja o cenach: otwarcia, zamknięcia, najniższej i najwyższej w trakcie sesji a także o wolumenie (obrocie) w transakcjach kupna – sprzedaży danej akcji. Powyższe informacje można dostarczać bezpośrednio na wejścia sieci, albo odpowiednio je przygotowując. Sieć otrzymując informacje na wejście o wszystkich czterech cenach, które z reguły posiadają bardzo podobne wartości, może mieć problem z selekcją istotnych zależności pomiędzy cenami.



Rys. 5.22 Prezentacja danych jako dwa sygnały oraz różnica pomiędzy nimi.

Lepiej jest dostarczyć sieci informacji o bezwzględnej zmianie w stosunku do ceny zamknięcia w postaci różnic ( patrz rys 5.22 high – close). Sieć wówczas łatwiej nauczy się zmian trendów, gdyż już na wejściu będzie miała wyraźne znaki zmian. Różnice przyjmują wartości zarówno

dodatnie jak i ujemne a co za tym idzie bardzo łatwo odróżnić spadki i wzrosty. Dane są normalizowane w znacznie węższym zakresie przez co zwiększa się także w istotny sposób dokładność danych. Tak przedstawione dane są podobne do prezentowania cen poprzez świece japońskie, będące jedną z najlepszych metod prezentacji cen, przez co zyskuje się potwierdzenie słuszności wyboru metody.

Szczegóły zależności zmian cen na przyszłe ceny zamknięcia wyjaśnia diagram drzewa decyzyjnego (rys. 1.15).

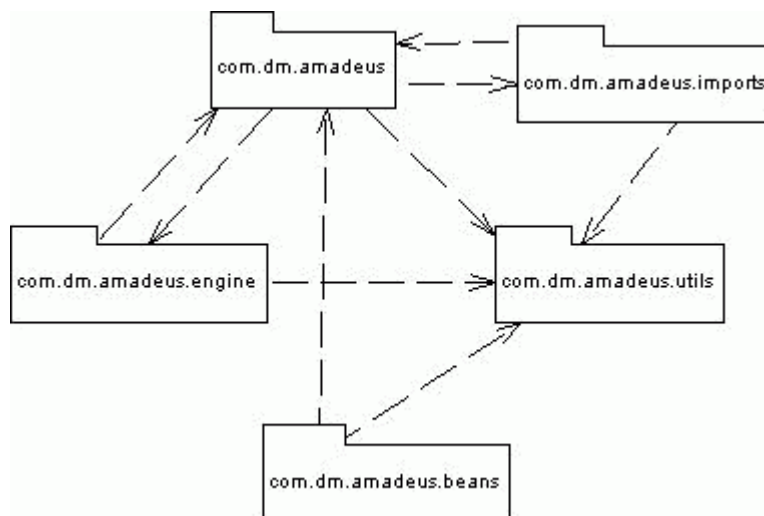
Dane wejściowe zawierają dużo danych będących przypadkowymi fluktuacjami, gwałtownymi skokami nie posiadającymi swoich sygnałów w poprzednich sesjach. Mogą być wywołane czynnikami politycznymi, ekonomicznymi bądź też „psychologią rynku” czyli reakcją na niepotwierdzone informacje tak, aby zaraz wrócić do poprzedniego trendu. Sieć nie jest w stanie poprawnie odwzorować sytuacji losowych, niestety sieć nie potrafi odróżnić czy dana sekwencja cen jest sytuacją, której powinna się nauczyć czy też nie. Dane powinny być tak przygotowane, aby uchronić sieć przed uczeniem się przypadkowych sytuacji. W projekcie zadanie to zrealizowano poprzez niwelowanie ostrych przebiegów stosując średnią wykładniczą (EMA). Problemem jest odpowiedni dobór parametrów (opóźnienie i waga). Opóźnienie powinno być indywidualnie dobrane do spółki a także być związane z długością predykcji. Jeżeli prognoza ma być długotrwała, opóźnienie powinno być znacznie większe. Nie interesują nas fluktuacje na przestrzeni dni a jedynie trend długoterminowy tym bardziej wyraźny im bardziej wygładzony zostanie przebieg. Dla prognozy krótkotrwałej uśrednianie nie powinno eliminować ważnych sygnałów.

## 5.6 Realizacja praktyczna

Architektura programowa systemu Amadeus składa się ze stron JSP widocznych podczas pracy z systemem, bazy danych, a także pakietów i klas związanych z warstwą biznesową realizujących zadania związane z zachowaniem aktualnych informacji w bazie danych (import), uczeniem sieci neuronowych oraz generowaniem wyników.

### 5.6.1 Pakiety i klasy związane z warstwą biznesową

Klasy realizujące funkcje warstwy biznesowej zostały pogrupowane tematycznie w zależności od zastosowania na pięć pakietów. Diagram pakietów, który został umieszczony na rys. 5.23 obrazuje zależności pomiędzy poszczególnymi pakietami systemu Amadeus.



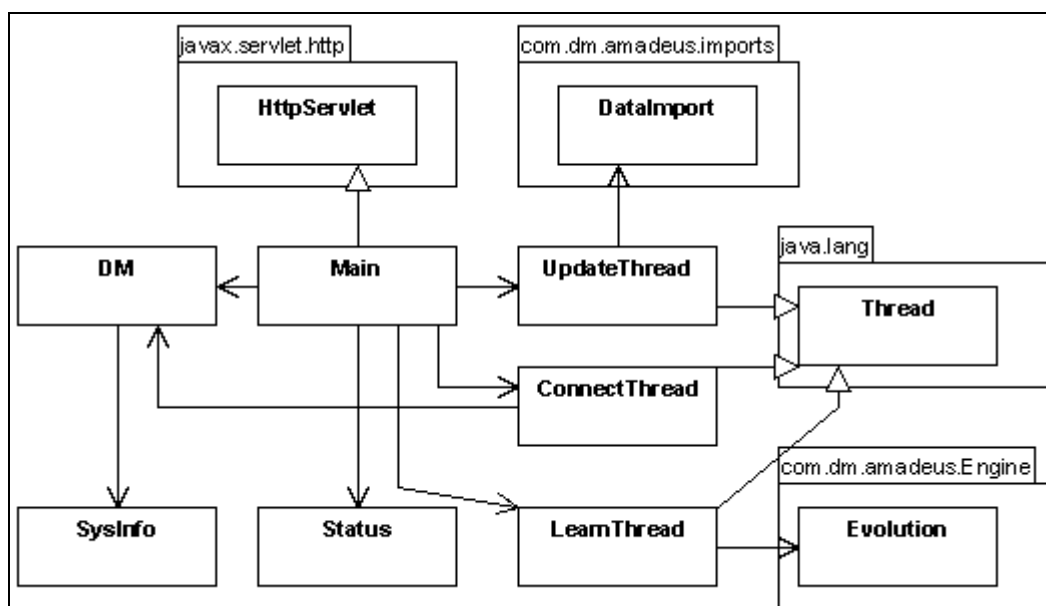
Rys. 5.23 Diagram pakietów systemu Amadeus

**Pakiet com.dm.amadeus**

Podczas rozpoczęcia pracy systemu, serwer aplikacji tworzy egzemplarz klasy *Main*. Klasa ta dziedziczy po *HttpServlet* a zatem jest serwletem. Do jej zadań należy utworzenie następujących wątków:

- *ConnectThread* – odpowiedzialnego za zestawienie połączenia z bazą danych.
- *UpdateThread* – zajmującego się uaktualnianiem bazy danych nowymi notowaniami tak szybko jak pojawią się na stronie internetowej skąd zostaną pobrane i zaimportowane do bazy danych.
- *LearnThread* – wykorzystywanego w procesie uczenia sieci neuronowej dopóki nie nastąpi żądanie wykonywania obliczeń wątek ten pozostanie w stanie nieaktywnym.

W ten sposób zagwarantowano „aplikacji webowej” wykonywanie sprecyzowanych uprzednio zadań „w tle”. Rozpoczęcie lub zakończenie uczenia to wprowadzenie wątku *LearnThread* w stan aktywny lub jego zakończenie.



Rys. 5.24 Diagram klas pakietu com.dm.amadeus

Klasa *DM* (skrót od *Data Module*) to zbiór funkcji realizujących dostęp do baz danych, posiada statyczny atrybut *Connection* będący fizycznym połączeniem z serwerem bazy danych gwarantującym wyłącznie jedną instancję połączenia. W ciele tej klasy zaimplementowano wątek *ConnectThread*.

Klasy *Status* i *SysInfo* realizują funkcje informujące o aktualnym stanie działania systemu.

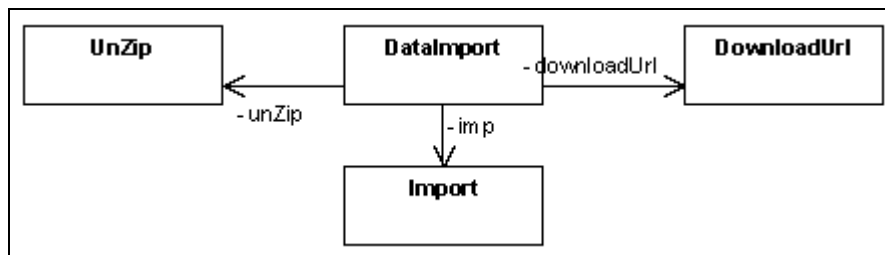
Wyróżnia się następujące tryby pracy systemu:

- TRAINING – uczenie sieci neuronowej.
- TESTING – testowanie sieci neuronowej.
- FORECASTING – tworzenie predykcji.
- INITIALIZING – inicjowanie algorytmu genetycznego początkową populacją, bądź też proces przygotowania do kontynuacji wcześniej przerwanej ewolucji.
- DOWNLOADING – pobieranie plików z serwera www.
- UNZIPPING – rozpakowywanie pobranych plików.
- IMPORTING – wstawianie pobranych notowań do bazy danych.
- COMPUTE\_CORELATIONS – wykonywanie obliczeń korelacji notowań pomiędzy dwiema spółkami giełdowymi
- IDLE – stan bezczynności, pozostawanie w gotowości na przyjęcie zadań do wykonania.

### Pakiet com.dm.amadeus.imports

System do poprawnej pracy wymaga aktualnych danych, uaktualnianych każdego dnia po zakończeniu sesji giełdowej. Dlatego wątek *UpdateThread* na bieżąco co określony interwał

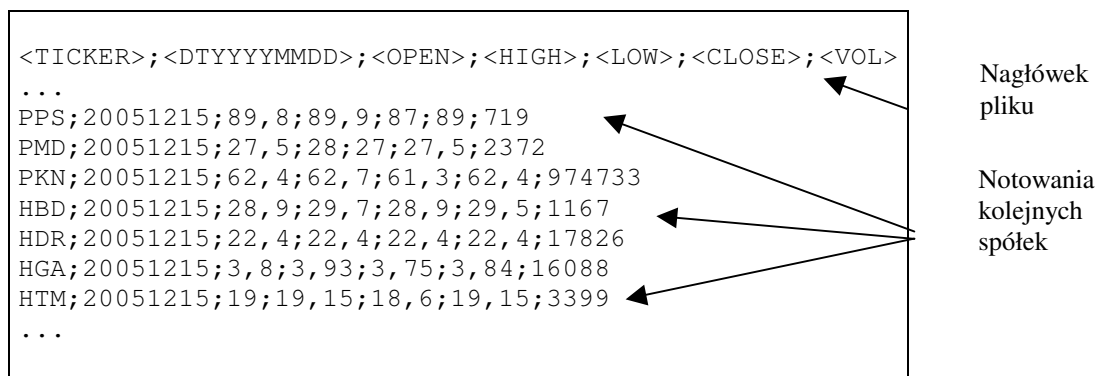
czasu (zwykle wystarcza 15 minut) wywołuje metodę *update()* klasy *com.dm.amadeus.imports.DataImport*, która porównuje ostatnią datę notowań w bazie danych z aktualnymi danymi na serwerze www i w przypadku różnicy wykonuje operacje importu.



Rys. 5.25 Diagram klas pakietu *com.dm.amadeus.imports*

W projekcie dane pobierane są w postaci plików z portalu [www.parkiet.com](http://www.parkiet.com) (adres: [www.parkiet.com/dane/danesesji/bazytxt/akcje/!akcjetxt.zip](http://www.parkiet.com/dane/danesesji/bazytxt/akcje/!akcjetxt.zip) dla archiwum wszystkich notowań oraz [www.parkiet.com/dane/danesesji/archiwum/dzis/d<ddmmyy>.zip](http://www.parkiet.com/dane/danesesji/archiwum/dzis/d<ddmmyy>.zip) dla wszystkich spółek dla danego dnia). Pierwszy wymieniony adres potrzebny jest w przypadku gdy system nie posiada żadnych danych w bazie (pierwsze uruchomienie) a drugi w sytuacji gdy istnieje potrzeba aktualizacji bazy od pewnej daty do chwili obecnej.

Ściągnięty (za pomocą klasy *DownloadUrl*) i zdekompresowany (przy użyciu klasy *UnZip*) plik posiada następującą przykładową zawartość:



Rys. 5.26 Budowa pliku tekstowego z notowaniami giełdowymi

Klasa *Import* ma za zadanie rozpoznać na podstawie odczytanego nagłówka, rodzaj informacji następujący w kolejnych liniach oddzielonych znakiem średnika - „;”.

Ściągnięte dane poddaje się analizie *preprocessingu* mającego na celu wykrycie brakujących i uzupełnienie danych (średnią arytmetyczną z sąsiednich pól co oznacza, że: w przypadku braku np. ceny zamknięcia z sesji *x* uzupełnia się ją średnią arytmetyczną z sesji wcześniejszej i

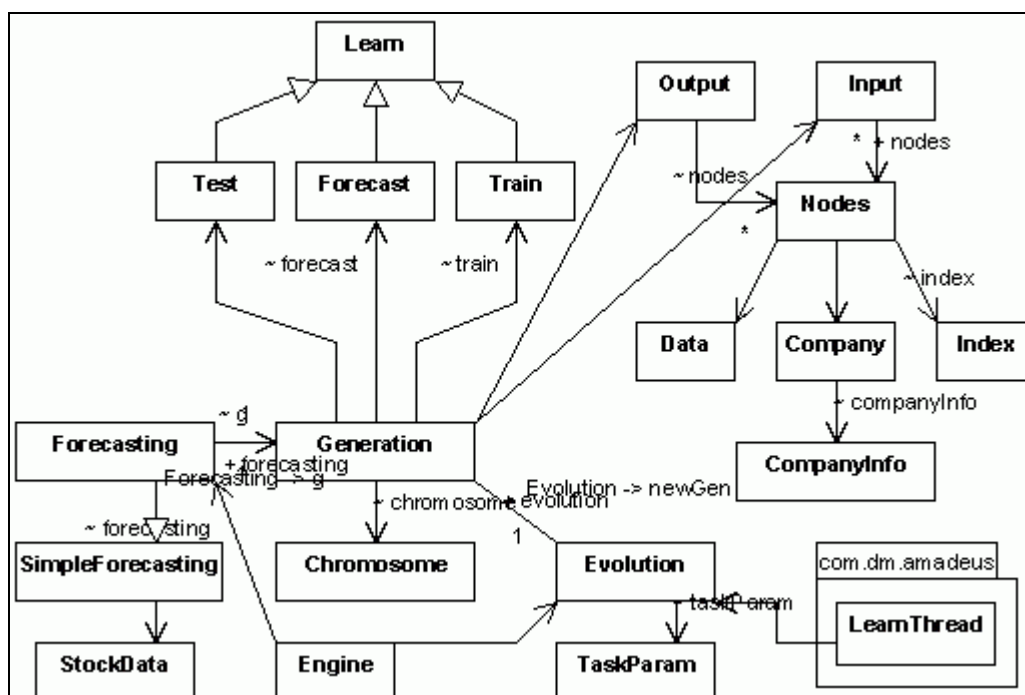
późniejszej dbając aby wartość spełniała warunki ceny maksymalnej i minimalnej o ile informacja o takowych będzie dostępna).

### Pakiet `dm.com.amadeus.beans`

Pakiet zawiera klasy wraz z funkcjami (zbudowanymi jako metody pobierające - *getters* i ustawiające – *setters* a zatem spełniających specyfikację *JavaBeans*<sup>40</sup>) koniecznymi do wymiany informacji pomiędzy warstwą prezentacyjną a biznesową. Zapobiega umieszczaniu zaawansowanych fragmentów kodu na stronach JSP a także tworzeniu identycznego kodu w wielu miejscach programu.

### Pakiet `dm.com.amadeus.engine`

Pakiet ten stanowi najważniejszą część aplikacji, zaimplementowana jest tu bowiem zarówno sieć neuronowa jak i algorytm ewolucyjny.



Rys. 5.27 Diagram klas pakietu `com.dm.amadeus.engine`

Algorytm ewolucyjny można rozpocząć na dwa sposoby. Można go inicjować od losowania parametrów podlegających randomizacji tworząc populację początkową, można także kontynuować rozpoczęte uprzednio a nie zakończone zadanie. W drugim przypadku

<sup>40</sup> <http://java.sun.com/products/javabeans>



przywracane są z bazy danych informacje pozwalające utworzyć ponownie osobniki o takich samych właściwościach co w przedostatnim pokoleniu (o ile w ostatnim pokoleniu liczba osobników jest większa od zera ale mniejsza od pełnej populacji) lub w ostatnim (jeżeli ostatnie pokolenie zawiera liczbę osobników dokładnie taką samą co populacja). Dalszy proces ewolucji w jednym i drugim przypadku wygląda tak samo.

Wywołanie rozpoczęcia algorytmu ewolucyjnego może zostać zainicjowane przez użytkownika (gdy system pozostaje w stanie bezczynności *IDLE*) lub na podstawie priorytetów zadania w chwili gdy istnieje większa ilość zadań do wykonania w kolejce zadań.

Rozpoczęcie algorytmu ewolucyjnego rozpoczyna się od wywołania metody *execute()* w klasie *Evolution*, która zapewnia:

- Walidację poprawności parametrów zadania (parzysta i większa od zera liczba osobników populacji; większa od zera liczba epok, wzorców, okresu predykcji, ilości pokoleń czy też spółek giełdowych).
- Przywrócenie danych z ostatniego pokolenia lub stworzenie nowych osobników populacji początkowej.
- Wykonanie „turnieju” którego algorytm omówiono w [5.4] a uproszczony przykład realizacji programowej zamieszczono na rys 5.16 aż do osiągnięcia warunku końcowego. Jeżeli zadanie obejmuje większą ilość spółek niż jedna, czynności są powtarzane od początku dla kolejnej spółki giełdowej.

Po pozytywnie wykonanej walidacji parametrów zadania następuje utworzenie nowych lub przywróconych z bazy danych osobników. Pojedynczy osobnik populacji jest reprezentowany przez obiekt klasy *GENERATION*. Informacje zawarte w klasie (rys. 5.28) są związane z jego topologią sieci neuronowej którą reprezentuje (klasy *INPUT* i *OUTPUT* – rys. 5.30) oraz ze sposobem przeprowadzenia treningu, testowania i predykcji (*TRAIN*, *TEST*, *FORECAST* – rys 5.31).

Generation			
* isOk	: boolean	= false	
* input	: Input		
* output	: Output		
* train	: Train		
* test	: Test		
* forecast	: Forecast		
* type	: int		
* hidden	: int[]		
* chromosome	: Chromosome		
* noGen	: int	= 0	
* fitness	: double	= 0	
* noCopy	: int	= 0	
* idTaskGeneration	: int	= 0	
* id_generation	: int		
* testRMSE	: double		
* testMAPE	: double		
* trainRMSE	: double		
* testRMSPE	: double		
* trainingTime	: long	= 0	
* changeStocksReal	: double		
* changeStocksPredict	: double		
* parents	: int[]	= new int[2]	

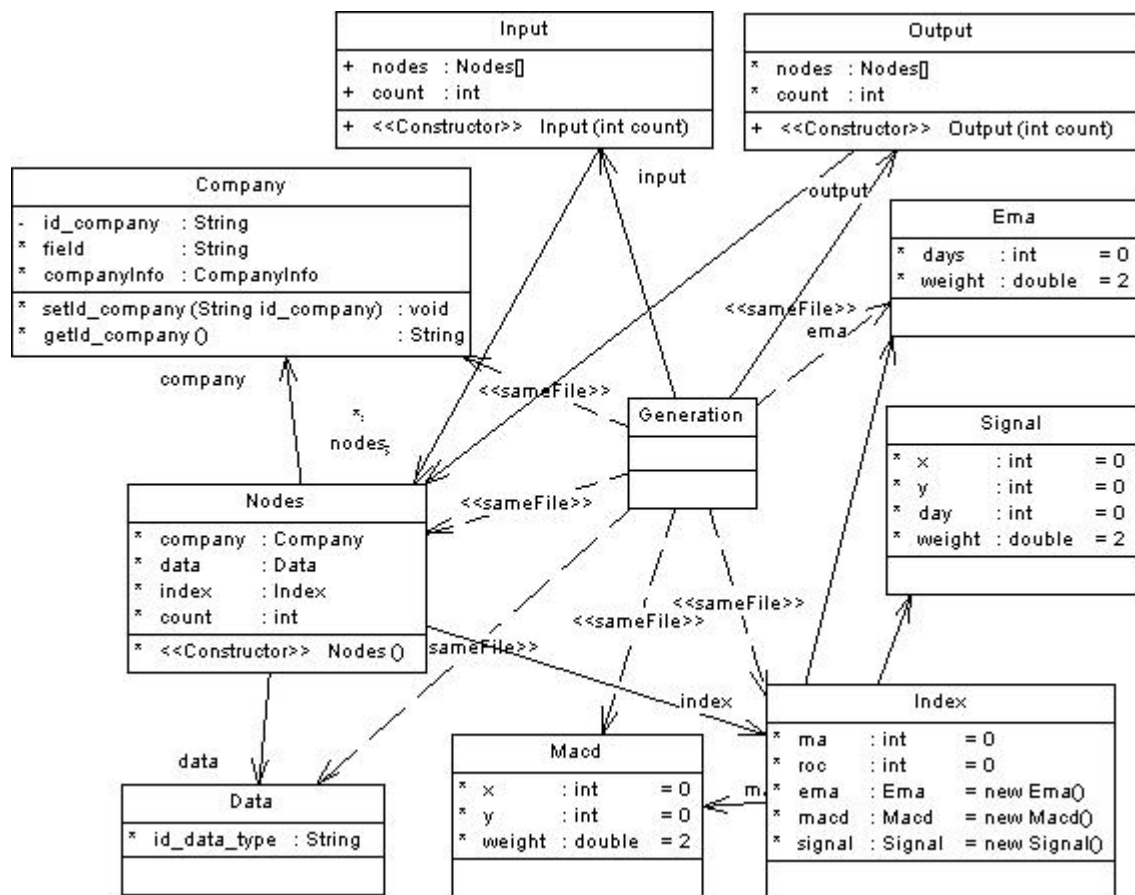
Rys.5.28 Diagram klasy *GENERATION*

Dodatkowo w klasie znajdują się informacje o numerze pokolenia, wskaźniku do rodziców osobnika, czasie nauczania, mierze błędów nauczania i testowania oraz predykcji. Wykorzystanie obiektu klasy *CHROMOSOME* będącej odzwierciedleniem binarnego zapisu struktury sieci ([5.29]) umożliwia szybki proces translacji pomiędzy tradycyjnym opisem topologii sieci do zapisu binarnego (i odwrotnie) którym posłużono się w trakcie wykonywania operacji krzyżowania i mutacji, które to również zostały zaimplementowane w tej klasie.

Chromosome			
- valBin	: String		
- valDec	: long		
- random	: Random	= new Random()	
+ <<Constructor>>	Chromosome ()		
+ <<Constructor>>	Chromosome (String valBin)		
+ <<Constructor>>	Chromosome (long valDec)		
+	getValBin ()		: String
+	getValDec ()		: long
+	setValDec (long valDec)		: void
+	setValBin (String valBin)		: void
+	crossover (Chromosome ch2, int pointOfCrossover)		: boolean
+	crossover (Chromosome ch2, int points, int firstIndex, int lastIndex)		: boolean
+	crossover (Chromosome ch1, Chromosome ch2, int points, int firstIndex, int lastIndex)		: boolean
+	mutation ()		: void
+	mutation (Chromosome ch)		: void
+	mutation (double probability, int firstIndex, int lastIndex)		: void
+	mutation (Chromosome ch, double probability, int firstIndex, int lastIndex)		: void

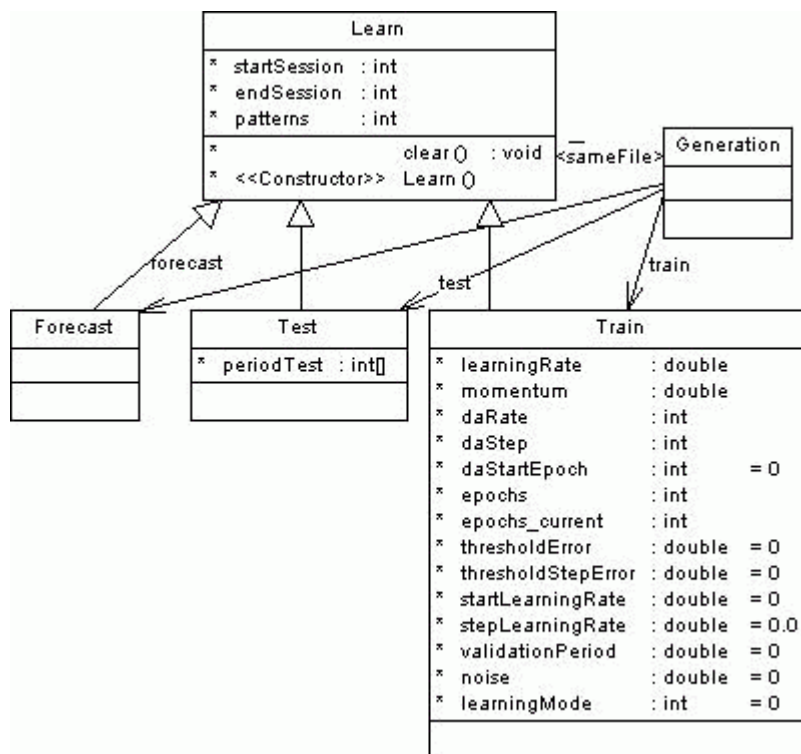
Rys. 5.29 Diagram klasy *CHROMOSOME*

Wspomniane klasy *INPUT* i *OUTPUT* mają za zadanie określenie jednoznacznej konfiguracji połączeń w warstwie wejściowej i wyjściowej. W klasach tych umieszczony jest z kolei obiekt klasy *NODES*, który definiuje jakiego typu informacja jest dołączona (jaka spółka giełdowa lub jakie dane innego typu np. kurs walut, złota etc.). W klasie *NODES* znajduje się natomiast wskazanie do obiektu klasy *INDEX*. Za pomocą tej klasy możliwe jest przetworzenie „surowych” danych na wyniki wybranych indeksów analizy technicznej (ROC, EMA, MACD). W tym celu wartość atrybutu *day* klasy *SIGNAL* musi posiadać wartość większą od zera.



Rys. 5.30 Klasy kooperujące z klasą *GENERATION* związane z topologią sieci.

Natomiast klasy *TRAIN*, *TEST*, *FORECAST* definiują początek i koniec zbiorów oraz dodatkowo w przypadku klasy *TRAIN* zdefiniowane są parametry uczenia sieci. Ten sposób (nieco nadmiarowy) zapewnia wysoką niezawodność działania a także sporą elastyczność modyfikowania pewnych parametrów w trakcie turnieju.

Rys. 5.31 Klasy kooperujące z klasą *GENERATION* związane z uczeniem sieci

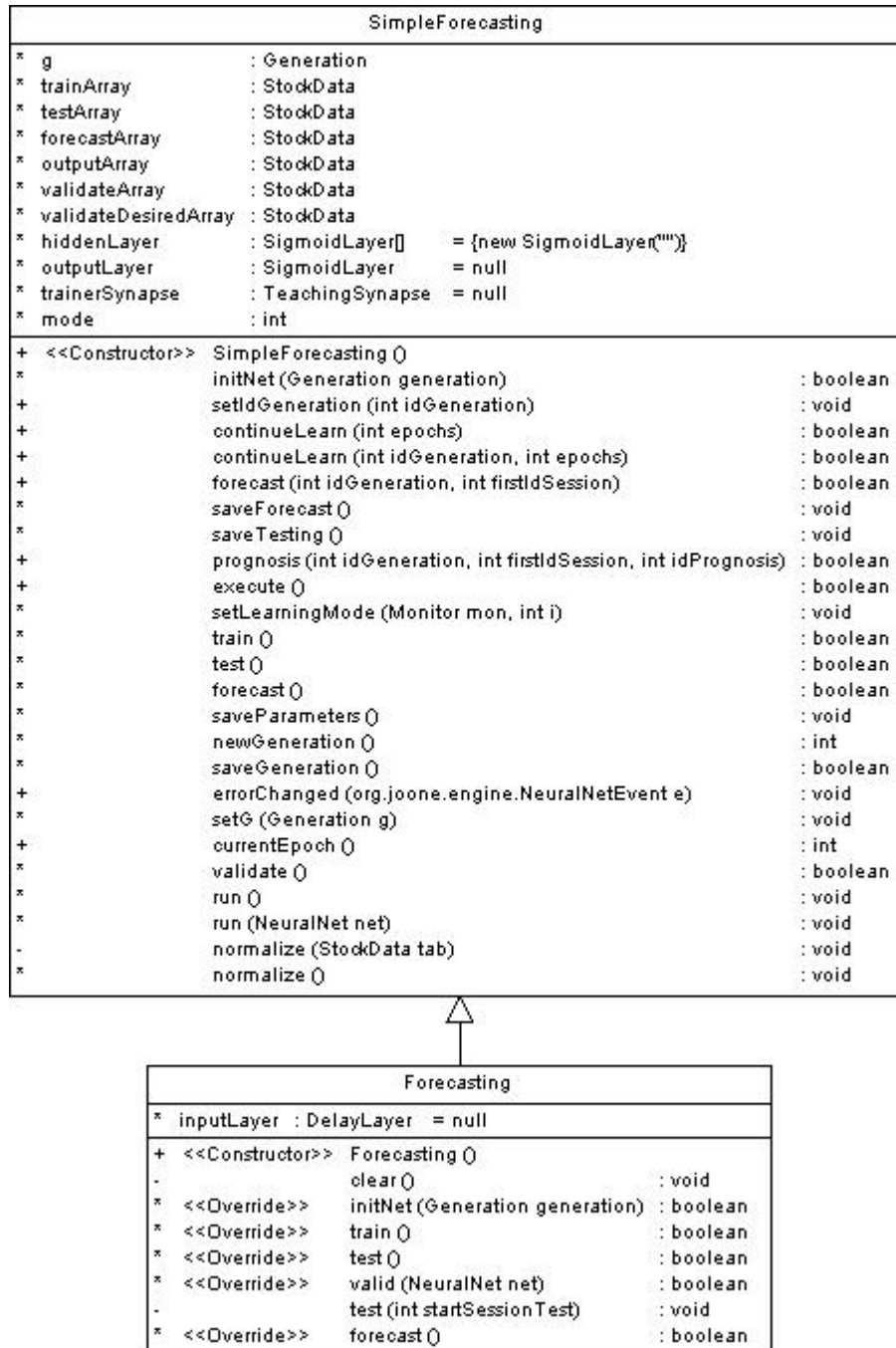
Gdy zostaną już zdefiniowane obiekty klasy *GENERATION* można przystąpić do realizacji „turnieju”, poniższy uproszczony fragment kodu obrazuje jego implementację.

```

while (noGen < condMaxGen) {
    //trening
    train();
    //suma przystosowań
    sum = sumFitness();
    //oblicz chromosom dla wszystkich osobników
    computeChromosome();
    //określ ilość egzemplarzy do reprodukcji
    computeNoCopy(sum, loop);
    //krzyżowanie
    crossover();
    //zapisanie pokolenia potomków jako przodków
    prepareNextGen(++noGen);
}
  
```

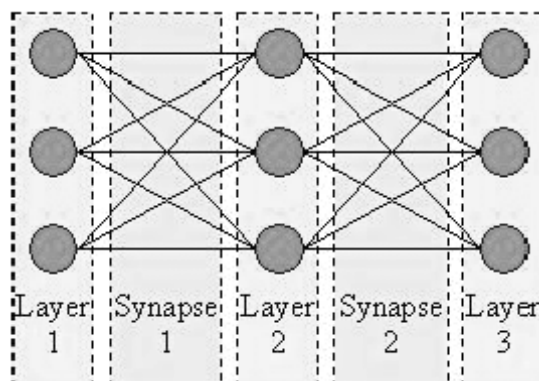
Rys.5.32 Implementacja programowa turnieju

Pierwsza wymieniona metoda – *train()* w algorytmie „turnieju” tworzy nowy obiekt klasy *Forecasting* dziedziczącej po klasie *SimpleForecasting*. Do zadań tej klasy należy przeprowadzenie uczenia sieci neuronowej łącznie z przygotowaniem danych.



Rys. 5.33 Diagram klasy Forecasting

W klasie *Forecasting* do implementacji sieci neuronowej użyto gotowego frameworka JOONE<sup>41</sup> (ang. *Java Object Oriented Neural Engine*) dostępnego na licencji GNU GPL. W ten sposób uniknięto mozolnej pracy implementacyjnej nie będącej tematem niniejszej pracy.



Rys.5.34 Ilustracja klas implementujących warstwy i połączenia frameworku JOONE<sup>42</sup>

Implementacja sieci neuronowej rozpoczyna się od zadeklarowania warstw sieci neuronowej. Konieczne jest zatem zbudowanie warstwy wejściowej i wyjściowej i co najmniej jednej warstwy ukrytej (w projekcie zakłada się, że ukrytych warstw sieci może być jedna lub dwie; dla dalszego uproszczenia schematu postępowania przyjęto, że występuje tylko jedna warstwa ukryta). Stworzenie trzech warstw następuje zatem po poniższym zapisie:

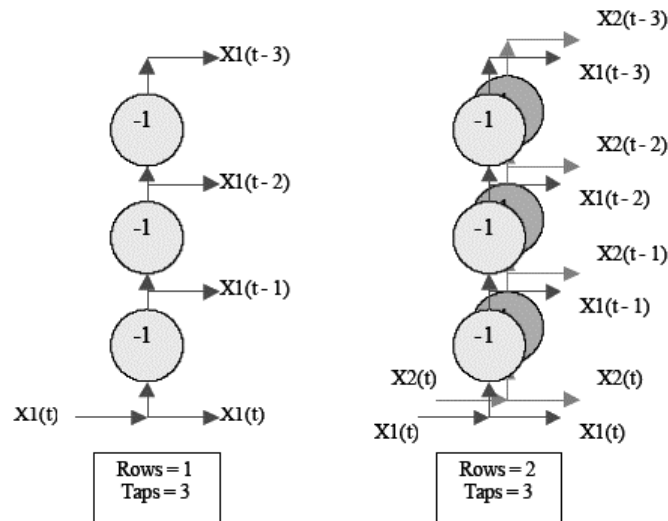
```
DelayLayer inputLayer = new DelayLayer("INPUT_LAYER");
SigmoidLayer hiddenLayer[] = {new SigmoidLayer("HIDDEN_LAYER")};
SigmoidLayer outputLayer = new SigmoidLayer("INPUT_LAYER");
```

Klasa *SigmoidLayer* to implementacja warstwy składającej się z neuronów pobudzanych funkcją sigmoidalną. Z uwagi na fakt występowania jednej lub dwóch warstw ukrytych zostały one zapisane w kodzie jako tablica obiektów.

Użycie klasy *DelayLayer* ułatwia organizację podawania danych wejściowych w przypadku uczenia ciągów czasowych. Klasa sama zapewnia podawanie odpowiednich danych w kolejnych cyklach uczenia (rys. 5.35)

<sup>41</sup> <http://sourceforge.net/projects/joone>

<sup>42</sup> Joone Complete Guide

Rys. 5.35 Schemat działania klasy DelayLayer<sup>43</sup>

Stworzone warstwy należy następnie odpowiednio ze sobą połączyć. Połączenia realizowane są w sposób „każdy z każdym”. Odpowiednikiem w kodzie tego połączenia jest zapis (dla jednej warstwy ukrytej).

```
FullSynapse INPUT_HIDDEN = new FullSynapse();
FullSynapse HIDDEN_OUTPUT = new FullSynapse();

inputLayer.addOutputSynapse(INPUT_HIDDEN);
hiddenLayer[0].addInputSynapse(INPUT_HIDDEN);
hiddenLayer[0].addOutputSynapse(HIDDEN_OUTPUT);
outputLayer.addInputSynapse(HIDDEN_OUTPUT);
```

W ten sposób powstały dwa “pełne” połączenia, które zostały dołączone do odpowiednich warstw sieci neuronowych.

Każda z tych warstw wymaga sparametryzowania pod względem ilości neuronów w każdej z nich. Warstwy ukryte i wyjściowe posiadają tylko jeden parametr dotyczący liczby neuronów:

```
hiddenLayer.setRows(INPUT_LAYER_DIMENSION);
outputLayer.setTaps(OUTPUT_LAYER_DIMENSION);
```

<sup>43</sup> Joone Complete Guide

Natomiast dla warstwy wejściowej parametryzacja ta wygląda nieco inaczej. Dla tej warstwy należy sparametryzować zarówno ilość rodzajów informacji (np. cena zamknięcia i wolumen) jak również ilość dni opóźnień dla każdej z nich. Przyjmuje się uproszczenie, że dla każdego rodzaju informacji opóźnienie będzie posiadać tą samą długość. Poniższy zapis oznacza zatem, że warstwa wejściowa posiada 14 neuronów pogrupowanych w dwa rodzaje wejść o szerokości 7. W ten sposób możliwe jest np. wczytanie wartości kursu akcji i wolumenu z 7 dniowym opóźnieniem.

```
inputLayer.setRows(2);  
inputLayer.setTabs(6);
```

Kolejnym krokiem jest załadowanie zbiorów: treningowego, testującego, predykcji i wyjściowego. O ile przeznaczenie dwóch pierwszych zbiorów powinno być zrozumiałe, zawierają one dane odczytane z bazy danych dla konkretnych sesji i spółek giełdowych, o tyle zastosowanie dwóch ostatnich wymaga wyjaśnienia. Zbiór predykcji zawiera dane z bazy danych potrzebne do zasilania sieci, natomiast zbiór wyjściowy zawiera wyłącznie pozyskane informacje w etapie uczenia sieci. W projekcie wszystkie te zbiory zostały zrealizowane jako dwuwymiarowe tablice. W rzeczywistości dane te są przechowywane w obiektach klasy *StockData* która oprócz atrybutu zawierającego dwuwymiarową tablicę, zawiera jeszcze informacje potrzebne w procesie normalizacji i denormalizacji (zrealizowane metodą z-score [1.5.3]).

```
double[][] data;
```

Rozmiar tych tablic jest zależny od rodzaju zbioru oraz parametrów predykcji. Zakładając że:

- $n$  – ilość dni opóźnienia na wejściu,
- $m$  – długość zbioru treningowego,
- $t$  – ilość dni zbioru testującego,
- $k$  – ilość wyjść warstwy wyjściowej.

poniższe wyrażenia będą definiować rozmiary poszczególnych zbiorów:

```
double[][] trainArray = new double[n+k][m];  
double[][] testArray = new double[t+n][m];  
double[][] forecastArray = new double[n][m];  
double[][] outputArray = new double[1][m];
```



Z wyjątkiem zbioru treningowego wszystkie zbiory posiadają podobny format (specjalnym przypadkiem jest zbiór wyjściowy będący podzbiorem tego formatu).

Wartość wejściowa				Wartość wyjściowa
$X_1$	$X_2$	...	$X_n$	$X_1$
$X_2$	$X_3$	...	$X_{n+1}$	$X_2$
...	...	...	...	...
$X_{m-1}$	$X_m$	...	$X_{m+n-1}$	$X_{m-1}$
$X_m$	$X_{m+1}$	...	$X_{m+n}$	$X_m$

Rys. 5.36 Format zbiorów: testującego i predykującego (z lewej) oraz wynikowego (z prawej)

Zbiór treningowy wymaga nieco innego zapisu, w kolejnych wierszach po  $n$ -tym indeksie następują wartości oczekiwane na wyjściu. Taki sposób zapisu danych został wprowadzony ze względu na wymagania użytego frameworka.

Wartość wejściowa				Wartość oczekiwana
$X_1$	$X_2$	...	$X_n$	$X_{n+1}$
$X_2$	$X_2$	...	$X_{n+1}$	$X_{n+2}$
...	...	...	...	...
$X_{m-1}$	$X_m$	...	$X_{m+n-1}$	$X_{m+n}$
$X_m$	$X_{m+1}$	...	$X_{m+n}$	$X_{m+n+1}$

Rys. 5.37 Format zbioru treningowego

Po stworzeniu obiektów (*MemoryInputSynapse*) odpowiedzialnych za połączenie pomiędzy warstwą ukrytą a danymi wejściowymi (osobnych dla danych treningowych - *input* i oczekiwanych – *desired*) i załadowaniu tablicami należy jeszcze zadeklarować które kolumny są danymi służącymi do treningu a które są wartościami oczekiwanymi.

```
MemoryInputSynapse memInput = new MemoryInputSynapse();
MemoryInputSynapse memDesired = new MemoryInputSynapse();
memInput.setInputArray(trainArray);
memDesired.setInputArray(trainArray);
memInput.setAdvancedColumnSelector("1-7");
memDesired.setAdvancedColumnSelector("8");
```

Wszystkie dane wymagają uprzednio znormalizowania do zakresu z przedziału liczb [0..1]. Otrzymane wyniki na wyjściu sieci są ponownie przywracane do pierwotnego zakresu (denormalizacja). Zapewnia to zapamiętywanie w strukturach podzielnika dla każdego ze zbiorów z osobna.

Aby ustawić parametry sieci (m.in.: współczynnik uczenia – *LearningRate*, współczynnik momentum, ilość wzorców treningowych – *TrainingPatterns*, epok – *TotCicles*, algorytmu uczenia *LeraningMode* oraz ustawienie w tryb uczenia – *Learning*) a także zarządzać rozpoczynaniem i kończeniem jej pracy tworzony jest obiekt nadzorcy:

```
Monitor monitor = new Monitor();
monitor.setLearningRate(0.8);
monitor.setMomentum(0.3);
monitor.setTrainingPatterns(200);
monitor.setTotCicles(500);
setLearningMode(mon, "org.joone.engine.BasicLearner");
monitor.setLearning(true);
```

który potem jest przypisywany każdej z warstw z osobna.

```
inputLayer.setMonitor(monitor);
hiddenLayer[0].setMonitor(monitor);
outputLayer.setMonitor(monitor);
```

Uruchomienie uczenia odbywa się poprzez:

```
mon.Go();
```

Po zakończeniu uczenia odbywa się testowanie sieci a następnie obliczenie błędów uczenia (RMSE) i testowania.(RMSE i MAPE).

W tym momencie zakończony jest proces dotyczący sieci neuronowych. Otrzymane wyniki związane z błędami oraz różnica w wyniku predykowanym a rzeczywistym na podstawie wzoru (5.1) są przekształcane w wartość funkcji przystosowania *fitness*. Dodatkowo dokonuje się obliczenia sumarycznego przystosowania całego pokolenia. Na tej podstawie wyznacza się ilość osobników przechodzących do następnego pokolenia.

**Pakiet dm.com.amadeus.utils**

W pakiecie zawarto klasy z użytecznymi funkcjami bibliotecznymi wykorzystywanymi w całym projekcie (m.in. obsługa dat, plików, katalogów oraz generowaniem liczb losowych a także konwersja liczb).

### 5.6.2 Baza danych

Dane historyczne oraz wyniki obliczeń a także parametry systemowe przechowywane są w strukturach bazy danych. W ten sposób zachowana jest trwałość danych oraz szybki i sprawny do nich dostęp.

Najważniejszą rolę w systemie pełni tabela *STOCKS* zawierająca informacje (ujęte w [4.1.1]) o wszystkich notowaniach spółek giełdowych zawartych w tabeli *COMPANY*. Obie tabele realizują trójwymiarową kostkę danych [1.4] o trzech następujących wymiarach: rodzaj danych oraz sesja i spółka giełdowa. W ten sposób jednoznacznie określając dostęp do poszukiwanej danej.

Z kolei z tabelą *COMPANY* połączone są dwie inne tabele: *BRANCH* zawierające informacje o branży spółki giełdowej oraz *CORELATIONS* zawierające informacje o korelacjach pomiędzy wartościami notowań w wspólnym okresie czasu.

Tabela *SESSIONS* zawiera wpisy typu „klucz-wartość”, kluczem jest tu numer kolejnej sesji notowań giełdowych a wartością jest jej data. W ten sposób można w sposób dualny identyfikować sesję w bazie: po jej numerze oraz po jej dacie. Relacja *jeden do wielu* oznacza, że nie może mieć miejsca sytuacja w której nie istnieje żadna sesja w tabeli *STOCKS* pomimo obecności w tabeli *SESSIONS*. Dla łatwiejszego posługiwania się podczas programowania systemu obie tabele zostały połączone w wspólny widok.

W tabeli *IMPORTS* umieszczone są zapisy o dokonanych poprawnie importach do bazy (uaktualnieniach). W ten sposób system może kontrolować czy dysponuje aktualnymi informacjami w bazie.

W tabeli *GENERATIONS* w trakcie uczenia sieci zapisywane są wszystkie dane związane z poszczególnymi osobnikami populacji. W danych tych zawarta jest pełna informacja o topologii sieci neuronowej, uzyskane rezultaty na etapie uczenia i testowania. Wszystkie te dane są potrzebne do reprodukcji jeżeli dany osobnik spełni warunki do jej przystąpienia.

Struktura bazy zapewnia kontynuowanie uczenia od dowolnego momentu, aż do decyzji o usunięciu danych w bazie trzymane są wszystkie szczegółowe informacje.

Z tabelą *GENERATIONS* pośrednio poprzez *TASK\_GENERATIONS* połączona jest tabela *TASKS*, w której zdefiniowane są zadania. Każde zadanie może mieć kilka „wykonań” (*TASK\_GENERATIONS*) zadań np. w skutek kontynuacji przerwanej zadania lub decyzji o uczeniu od nowa.

Po zakończonym uczeniu sieci, wyniki predykcji zapisywane są do tabeli *FORECAST* natomiast do tabeli *VERIFY* zapisywane są wyniki w trakcie testowania. Tabela *PROGNOSIS* zawiera listę predykcji generowanych przez użytkownika, pole *id\_session* definiuje początkową sesję od której predykcja następuje a długość prognozy jest zapisana w polu *predist\_period* tabeli *TASKS* która jest połączona z tabelą *PROGNOSIS* w relacji *jeden do wielu*.

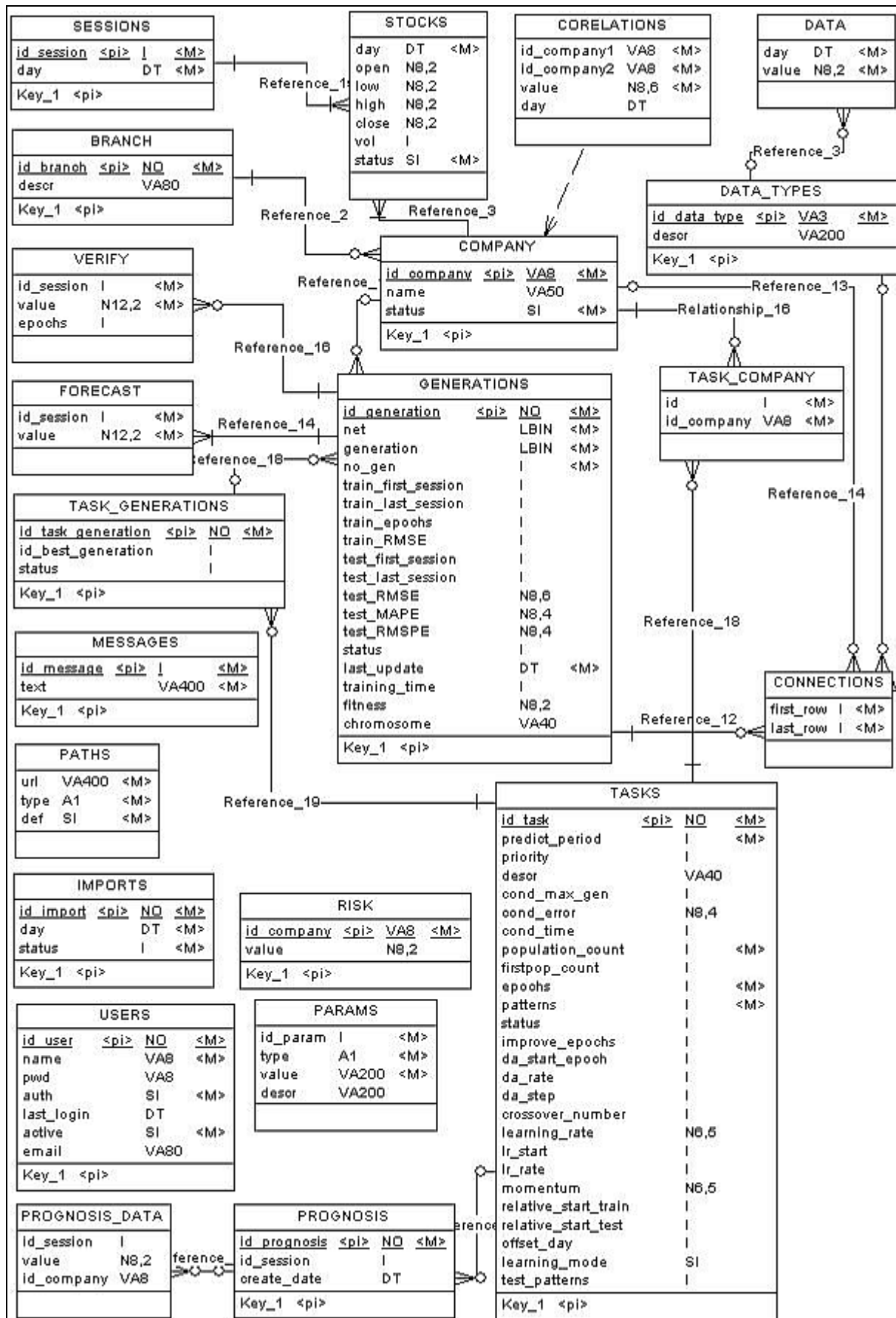
Zbiór danych testujących jest wydzielonym zbiorem w tabeli *STOCKS* na podstawie zawartości pól *relative\_start\_train* i *test\_pattern* tabeli *TASKS*. Pola te wyznaczają początkową sesję i ilość sesji odczytywanych z tabeli *STOCKS*. Pole *offset\_day* jeżeli jest różne od zera informuje, że punktem odniesienia jest numer sesji wyznaczony jako różnica numeru ostatniej sesji giełdowej i wartości pola *offset\_day*. Użycie tego parametru jest bardzo pomocne w fazie testowania systemu, ustawienie go na wartość dodatnią umożliwia na bieżąco walidacji rzeczywistych wartości predykcji. Analogicznie zbiór uczący wyznaczają wartości *relative\_start\_train* i *patterns*.

Pozostałe pola tabeli *TASKS* są odzwierciedleniem definiowanego zadania omówionego w podrozdziale [5.2]. Tabela *CONNECTIONS* ma zadanie konfiguracji danych umieszczanych na wejściu sieci neuronowej. Wraz z tabelami *DATA* i *DATA\_TYPES* umożliwiają dowolne zdefiniowanie kombinacji spółek i indeksów analizy technicznej.

Klika słów omówienia należy się grupie tabel definiujących parametry pracy systemu. Tabela *PARAMS* zawiera parametry systemowe, pole *type* oznacza typ danych zawartych w tym polu (liczba całkowita, data, ciąg tekstowy) a pole *value* zawiera wartość tego parametru.

Dane użytkowników systemu są przechowywane w tabeli *USERS*. Treści komunikatów, błędów i ostrzeżeń są zapisane w tabeli *MESSAGES*.

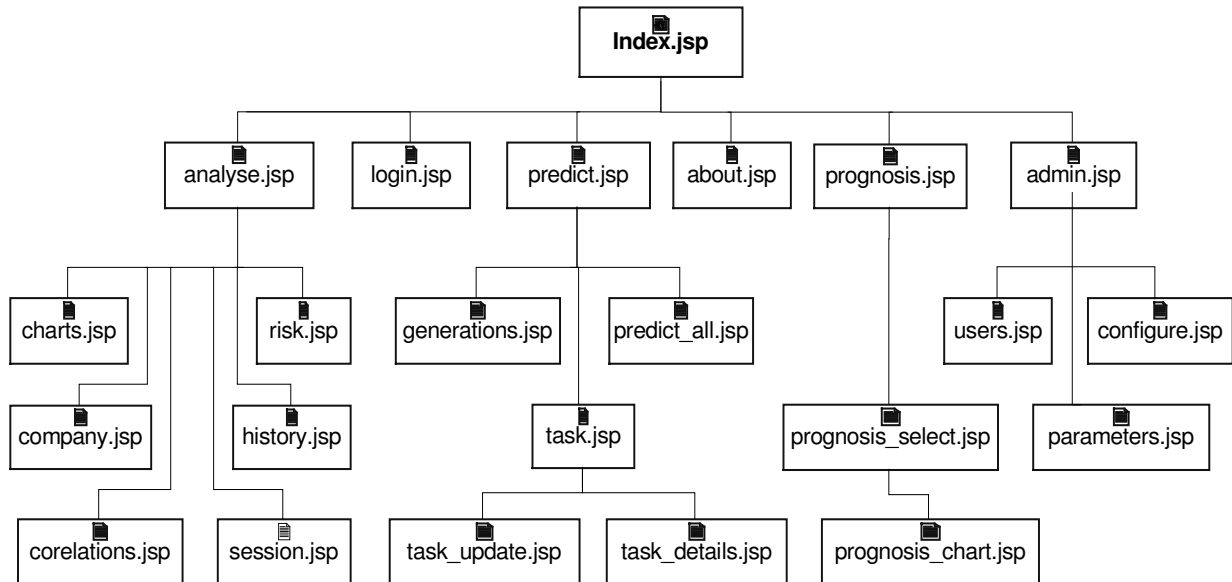
Złożone zadania systemu operujące na danych zostały zaimplementowane poprzez funkcje (ang. *Function*) lub poprzez procedury składowane (ang. *Stored Procedure*). W ten sposób wyeliminowano obecność w kodzie programu skomplikowanych procedur operujących na danych z bazy w wyniku czego kod zyskał na czytelności a efektywność pracy systemu wzrosła.



Rys. 5.38 Diagram encji bazy danych zastosowanej w projekcie AMADEUS.

### 5.6.3 Organizacja stron JSP

Kliku słów omówienia wymaga sposób organizacji interfejsu klienta. Został on zrealizowany poprzez strony JSP. Tradycyjnie strona o nazwie *index.jsp* stanowi główną stronę z odnośnikami do tematycznych działów zawartych w systemie. Diagram stron JSP został umieszczony na poniższym rysunku.



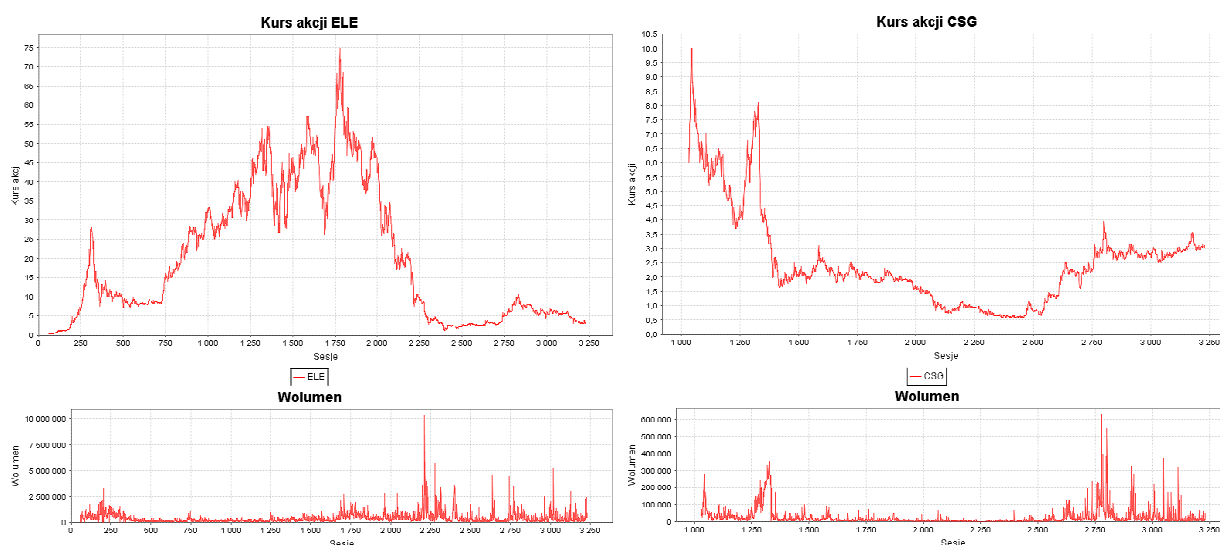
Rys. 5.39 Diagram stron JSP systemu Amadeus

## 6 Wyniki doświadczenia

Sieć neuronowa nie zawsze potrafi nauczyć się notowań spółki giełdowej. Istnieje wiele uwarunkowań, które uzależniają powodzenie użycia danej sieci w danych warunkach, jednak niepowodzenia, jakie towarzyszą zadaniu predykcji rynku giełdowego powodują, że powszechny jest pogląd o niemożności przewidzenia przyszłego ruchu cen na giełdzie.

Wykorzystując wiedzę z dziedziny analizy technicznej rynku giełdowego, w poniższym rozdziale została dokonana próba wykazania przydatności sieci neuronowej. Oczywistym jest to, że nie można od sieci oczekiwać gotowych wyników prezentujących zmiany cen dzień po dniu z minimalnym błędem. Przyjęto założenie, iż docelowym wynikiem jest zmiana stopy zysku w okresie średnioterminowym (20 sesji). Od sieci oczekuje się zatem sygnalizowania zmiany trendu lub potwierdzenia jego kontynuacji. Jednocześnie należy zrezygnować z konieczności uzyskania informacji o zmianach krótkoterminowych aczkolwiek dobrze nauczona sieć powinna odwzorowywać również część z takowych sygnałów.

W poniższych podpunktach, zamieszczone zostały przypadki mogące wpływać na wynik predykcji. Do analizy wybrano dwie spółki giełdowe Elektrim S.A. (ELE) oraz Centrostal S.A. (CSG). Pierwsza z nich działa w obszarze telekomunikacji, energetyki i kabli, druga zaś pochodzi z branży przemysłu hutniczego. Zatem obie spółki działają w przemyśle, odznaczając się wyższymi od przeciętnych obrotami. W danych historycznych obu spółek odnaleźć można notowania wzrostowe, spadkowe a także pewną część notowań będących w trendzie horyzontalnym.



Rys. 6.1 Przebieg cen i wolumenu spółek użytych w badaniach  
[Wykres pochodzi z systemu Amadeus]

W badaniach dokonano wstępnych założeń związanych z topologią sieci neuronowych i, o ile nie zasygnalizowano zmian, zakłada się, że poniższe parametry obowiązują w każdym przypadku.

Parametr	Wartość
Liczba neuronów w warstwie wejściowej	20
Liczba neuronów w warstwie wyjściowej	20
Liczba epok	50
Nominalny współczynnik uczenia	0,05
Początkowa wartość wsp. uczenia / skok	0,005/0,005
Współczynnik momentum	0,8
Algorytmy uczenia	BackPropagation, Batch BackPropagation, RPROP
Amplituda szumów dodanych do zbioru treningowego	0,4
Liczba punktów krzyżowania	2

## 6.1 Parametry mające wpływ na pracę sieci neuronowej

W pracy wykonano doświadczenia mające na celu wykazanie, jaki wpływ na pracę sieci neuronowej mają poszczególne parametry: ilość neuronów w warstwie ukrytej, wybór odpowiednich zbiorów uczących i stopień ich zaszumienia.

### 6.1.1 Wpływ ilości neuronów w warstwie ukrytej

Istnieje wiele zaleceń dotyczących optymalnej ilości neuronów w warstwie ukrytej. O ile ilość neuronów w warstwie wejściowej i wyjściowej bezpośrednio wynikają z założeń zadania, tak ten parametr uzależnia w bardzo istotny sposób jakość (błąd testowania) uczenia sieci.

Inkrementując ilość neuronów w warstwie ukrytej w przedziale od 1 do 60 zbadano obie sieci, w tym jedną (CSG) dla różnych algorytmów uczenia.

Wybrano odpowiednie przedziały danych dla zbiorów uczących i testujących, starając się, aby zbiory danych dla spółki ELE odznaczały się większym odchyleniem standardowym od populacji niż w przypadku spółki CSG - w ten sposób, przy okazji porównania, dokonano analizy jak zachowa się zależność błędu testowania od liczby neuronów w warstwie ukrytej dla różnych zbiorów danych.

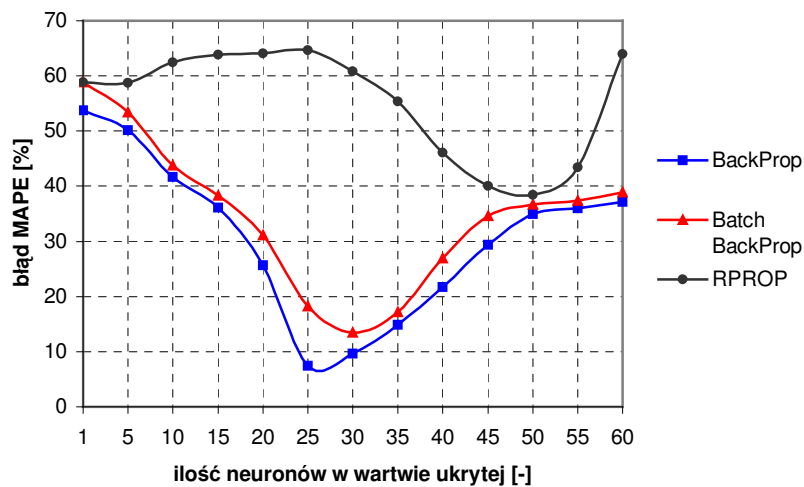
Spółka	Uczenie sieci	Odchylenie standardowe kursu akcji	Testowanie	Odchylenie standardowe kursu akcji
ELE	Sesje: 1110 – 1463 (16.07.1997 – 14.12.1998)	2,76	Sesje: 1464 – 1688 (15.12.1998 – 5.11.1999)	2,63
CSG	Sesje: 1400 – 1999 (15.09.1998 – 06.02.2001)	2,26	Sesje: 2850 – 3099 (02.07.2004 – 27.06.2005)	1,50

Rys. 6.2 Zbiory uczące i testujące dla spółek ELE i CSG

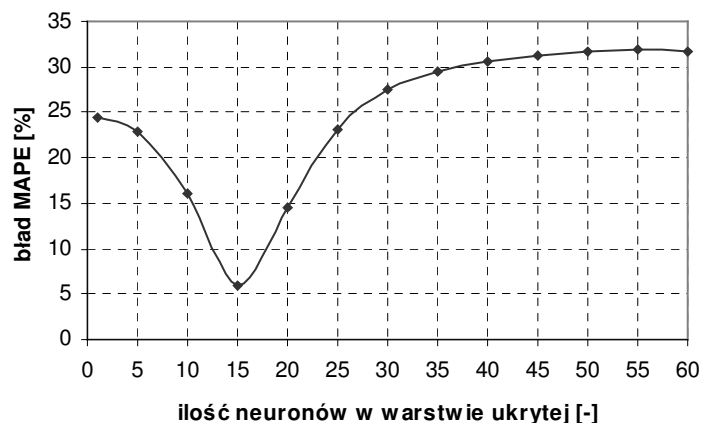




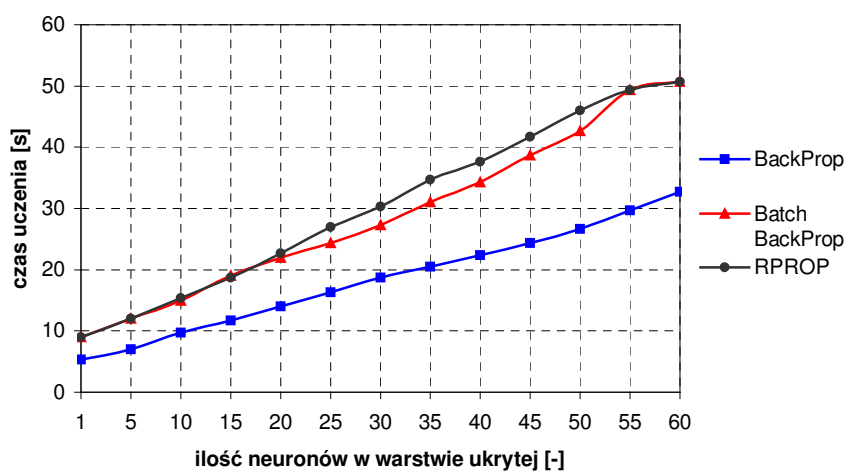
Rys. 6.3 Wykres zbiorów uczących i testujących dla spółek ELE i CSG  
[Wykres pochodzi z systemu Amadeus]



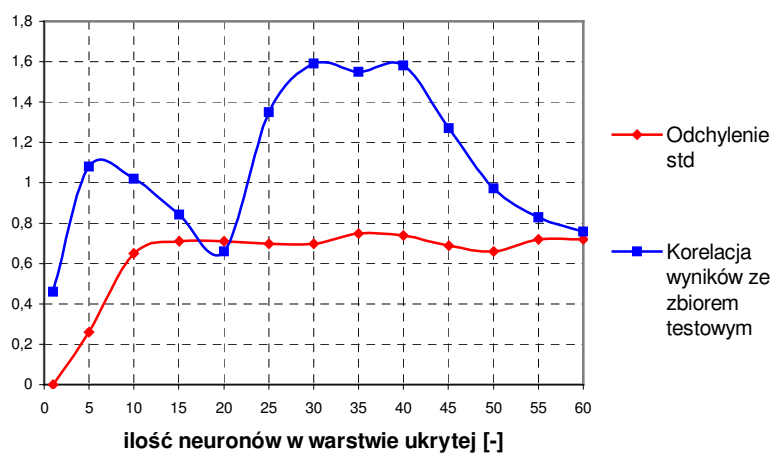
Rys. 6.4 Wykres zależności błędu MAPE od ilości neuronów w warstwie ukrytej (ELE)



Rys. 6.5 Wykres zależności błędu MAPE od ilości neuronów w warstwie ukrytej (CSG)



Rys. 6.6 Zależność czasu uczenia od ilości neuronów w warstwie ukrytej dla trzech wybranych algorytmów nauczania (ELE)



Rys. 6.7 Zależność odchylenia standardowego oraz korelacji wyników ze zbiorem testowym od ilości neuronów w warstwie ukrytej (ELE)

Analizując otrzymane wyniki zaprezentowane na rys 6.4 i 6.5 można stwierdzić, że sieć dla spółki o łagodniejszym przebiegu notowań (CSG) potrafi nauczyć się lepiej, wykorzystując przy tym mniejszą ilość niezbędnych neuronów w warstwie ukrytej. Czas uczenia jest, jak widać, proporcjonalny do ilości synaps. Zwiększając ilość neuronów w warstwie ukrytej, proporcjonalnie zwiększa się ilości połączeń.

To dowodzi raz jeszcze jak ważny jest etap preprocessingu danych wejściowych w całym procesie uczenia.

Badania wykazały nieprzydatność algorytmu RPROP, a także wyższy koszt użycia algorytmu Batch BackProp względem klasycznego algorytmu wstecznej propagacji błędów.

Dokonując z kolei analizy korelacji wyników ze zbiorem testowym oraz odchylenia standardowego, a także z obserwacji przebiegów wyjściowych dochodzi się do wniosków, że dla bardzo małych ilości neuronów (1-5) w warstwie ukrytej sieć jest niezdolna do generalizacji. Przebieg wyjściowy jest linią prostą, której amplituda jest bliska średniej wartości w okresie testującym. Innymi słowy, sieć nie ma wystarczająco dużo neuronów do odwzorowania przebiegu zbioru testującego. Dla kolejnych wartości, wynik staje się coraz bardziej skorelowany z oczekiwanym. Jednak dla punktu, w którym błąd testowania jest najniższy, korelacja maleje. Dzieje się tak dlatego - gdyż odwzorowany wynik jest bardzo gładki. W miarę kolejnego wzrostu neuronów w warstwie ukrytej, odchylenie standardowe oraz korelacja nie przyjmują już niższych wartości, sygnał staje się coraz bardziej „poszarpany” oscylując wokół prawidłowego rozwiązania.

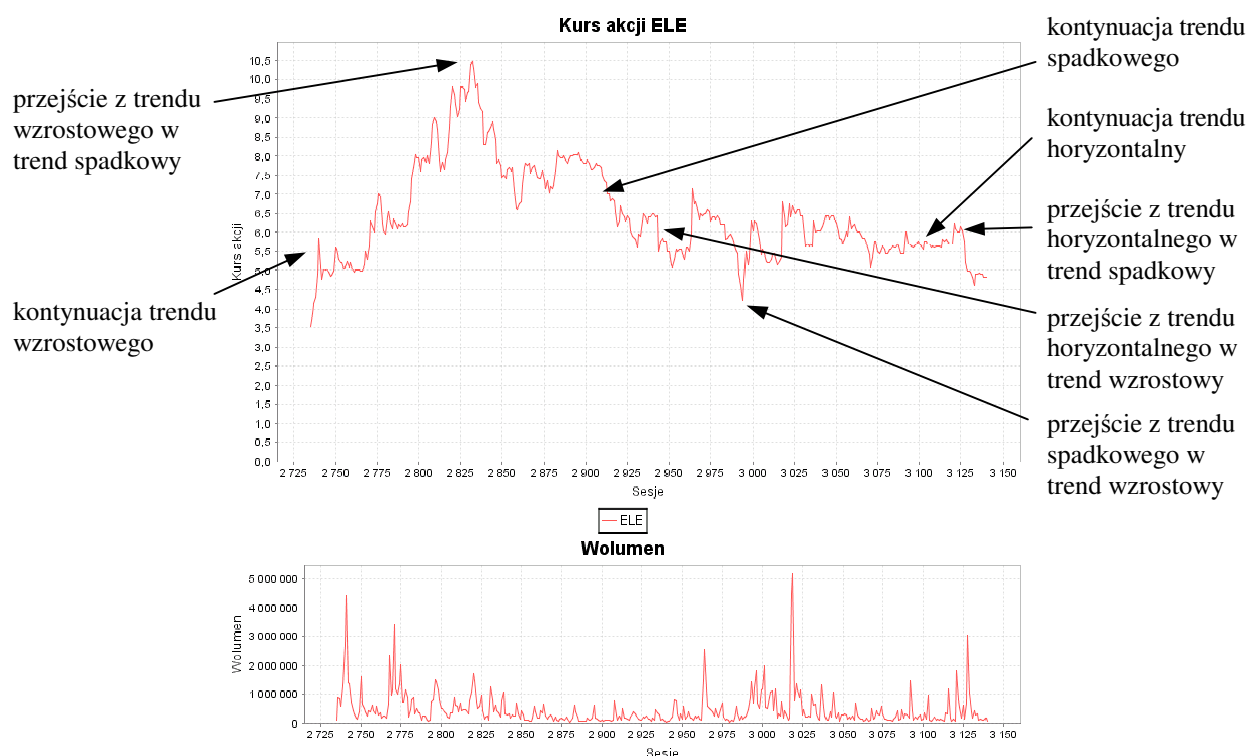
### 6.1.2 Poprawność wyboru zbiorów uczących

Kolejnym analizowanym tematem związanym z poprawą wydajności był dobór danych w procesie uczenia. Dane w procesie uczenia powinny być tak dobrane, aby odzwierciedlały jak największą kombinację formacji, trendów i innych charakterystycznych punktów w przebiegach notowań giełdowych.

W badaniu wykorzystano dane spółki Elektrim S.A., wydzielając z niej cztery zbiory:

- zbiór z przeważającym trendem wzrostowym,
- zbiór z przeważającym trendem spadkowym,
- zbiór z przeważającym trendem horyzontalnym,
- zbiór łączny, zawierający wszystkie powyższe przypadki.

Następnie wyróżniono siedem charakterystycznych punktów zmian, lub potwierdzenia kontynuacji bieżącego trendu.



Rys. 6.8 Charakterystyczne punkty zmian lub potwierdzenia kontynuacji  
[Wykres pochodzi z systemu Amadeus]




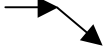


Ponieważ dostarczanie jedynie cen zamknięcia w procesie uczenia z punktu widzenia analizy technicznej nie jest wystarczające spróbowano uczyć sieć następującymi kombinacjami danych:

1	2	3	4	5	6	7	8	9
close	close vol	close vol high-low	close vol high-low close-open	close close-open	close high-low	close high-close	Close close-low	close vol close-open

Rys. 6.9 Zaproponowana konfiguracja danych wejściowych

Konfiguracje 1-4 zostały użyte dla wszystkich zbiorów, dodatkowo dla zbioru łącznego zastosowano dodatkowo kombinacje 5-9. Wyniki z przeprowadzonych badań zostały umieszczone w tabeli na rys. 6.10. Wyniki które odzwierciedlały poprawną prognozę zostały pogrubione tak aby łatwiej zobrazować poprawność zbiorów uczących wraz z odpowiednimi kombinacjami danych.

## ROZDZIAŁ 6 – WYNIKI DOŚWIADCZALNE

Sesja początkowa i zmiana na końcu okresu	Zbiór uczący z sesji 730-1349 ↗	Zbiór uczący z sesji 1778-2378 ↘	Zbiór uczący z sesji 2310-2710 —	Zbiór uczący z sesji 730-2620 ↗↘
2908 (-30,97%) 	1. 58,25% 2. -5,22% 3. 30,41% 4. -27,53%	1. -16,02 2. -22,15 3. -20,14 4. -21,97	1. -35,14% 2. -20,31% 3. -40,37% 4. -40%	1. -26,46% 2. -10,2% 3. -28,54% 4. -37,85% 5. -3,97% 6. -44,38% 7. -6% 8. -28,77% 9. -21,88%
2962 (+12,73%) 	1. 23,28% 2. 5,13% 3. 14,73% 4. 42,58%	1. -13,87 2. -18,27 3. -19,11 4. -21,92	1. -18,35% 2. -5,52% 3. -28,98% 4. -27,75%	1. -9,86% 2. 1,63% 3. -21,47% 4. -12,38% 5. -17,23% 6. -14,50% 7. 16,55% 8. 16,45% 9. -7,45%
2994 (+29,45%) 	1. -2,97% 2. 73,18% 3. 21,56% 4. 37,04%	1. -12,58 2. -18,99 3. -13,79 4. -19,19	1. -30,59% 2. -12,20% 3. -10,59% 4. -40,8%	1. -4,7% 2. 7,82% 3. -4,93% 4. -3,34% 5. 1,44% 6. -20,6% 7. 21,75% 8. 22,59% 9. 3,93%
3125 (-28,45%) 	1. 7,73% 2. 53,25% 3. -1,37% 4. 49,79%	1. -13,89 2. -19,51 3. -17,86 4. -16,22	1. 33,61% 2. -24,79% 3. -10,75% 4. -30,88%	1. -21,78% 2. -5,81% 3. -15,96% 4. -19,17% 5. -18,73% 6. -11,72% 7. -49% 8. 13,09% 9. -5,16%
2830 (-20,10%) 	1. 37,41% 2. 27,9% 3. 9,14% 4. 6,68%	1. -20,10 2. -21,92 3. -27,21 4. -23,49	1. -45,31% 2. -34,98% 3. -47,57% 4. -47,02%	1. -39,03% 2. -32,84% 3. -44,9% 4. -58,82% 5. -54,18 6. -47,45% 7. -41,81% 8. 1,6% 9. -46,50%
2735 (+42,8%) 	1. 1,45% 2. -18,31% 3. -1,43% 4. -24,71%	1. -16,7 2. -21,69 3. -13,67 4. -20,54	1. -1,51% 2. -2,99% 3. -3,72% 4. -3,33%	1. -8,5% 2. 13,81% 3. 4,42% 4. 30,58% 5. -4,24% 6. -3,6% 7. 1,89% 8. 7,36% 9. 13,16%

3100 (0%) →→	1. 2,15%	1. -7,23%	1. -4,56%	1. -18,56%
	2. 1,87%	2. -2,54%	2. -5,31%	2. -10,6%
	3. 6,54%	3. -1,42%	3. -8,38%	3. -6,68%
	4. 5,36%	4. -1,43%	4. -8,33%	4. -5,14%
				5. -7,11%
				6. -13,37%
				7. -41,87%
				8. -20,67%
				9. 0,3%

Rys. 6.10 Wyniki prognoz w zależności od zbioru uczącego

Wyniki nie powinny budzić zaskoczenia, zbiory w których przeważał tylko jeden trend nie dały pozytywnych rezultatów. Sieć uczona jedynie na trendzie wzrostowym, nie potrafiła rozpoznać i zasygnalizować zmiany trendu na spadkowy. Analogicznie sieć uczona na trendzie spadkowym nie rozróżniała sygnałów zmiany trendu na wzrostowy. Sieć nie mając wystarczających danych w procesie uczenia nie potrafiła dokonać prawidłowej predykcji. Dane wyjściowe należy uznać za przypadkowe. W przypadku predykcji zmian podobnych do takich na jakich sieć się uczyła, należy zauważyć dość dobrze dobraną wartość zmiany predykcji. Można dostrzec nawet, że sieci uczone jedynie na danym trendzie wykazują lepszą jakość predykcji tego trendu niż te uczone na zbiorze zawierającym wszystkie przypadki trendów.

Trzeci zbiór, uczący się jedynie wartościami w trakcie trwania trendu horyzontalnego wykazywał jeszcze gorszą jakość predykcji, zawsze predykował spadek trendu. Wynikać z tego może wniosek, że trudno dobrać długi okres uczący horyzontalny nie mający drobnych odchyłeń i widocznie takowy miał tu miejsce.

Niestety brak wykrywania innych trendów dyskwalifikował ogólne zastosowanie sieci uczonych jedynie na zdominowanym trendzie.

Pozostaje analiza najbardziej ciekawego zbioru pod względem zachowania. Analizując wyniki można dojść do wniosku że sieci neuronowe o konfiguracji wejść według kombinacji numer 2 oraz 7 jako jedyne poprawnie sygnalizowały zmianę bądź kontynuację trendu. Można wysnuć wniosek, że dla spółki Elektrim S.A. uczonej na zbiorze uczącym zawierającym wszystkie cztery charakterystyczne trendy, sieć neuronowa potrafiła dobrze przewidzieć przyszłe notowania mając do dyspozycji informację o przeszłych cenach zamknięcia i informację o wolumenie lub różnicy cen zamknięcia i ceny najwyższej.

Sieć na podstawie korelacji pomiędzy ceną zamknięcia i wolumenu lub ceny maksymalnej potrafiła dobrze rozpoznać podobny przypadek znaleziony gdzieś w danych historycznych.

Oczywiście każda spółka posiada inną specyfikę przebiegu danych historycznych, stąd niekoniecznie powyższe kombinacje muszą być słuszne.

Sieć neuronowa zbudowana w oparciu o konfigurację numer 9 nie potrafiła przewidzieć zmiany trendu horyzontalnego na wzrostowy, natomiast pozostałe kombinacje miały 2 lub 3 błędne decyzje na 6 możliwych.

Pozostał ostatni przypadek, przy którym sieć neuronowa w każdej sytuacji nie poradziła sobie wystarczająco dobrze. Jak widać kontynuacja trendu horyzontalnego była bardzo trudna do predykcji. Sieć usilnie doszukuje się w dostarczanych sygnałach, pomimo niskiej ich wartości zmian mogących zajść w przyszłości. Dlatego konieczne było zastosowanie dodatkowych mechanizmów mających na celu wyeliminowanie błędnych decyzji.

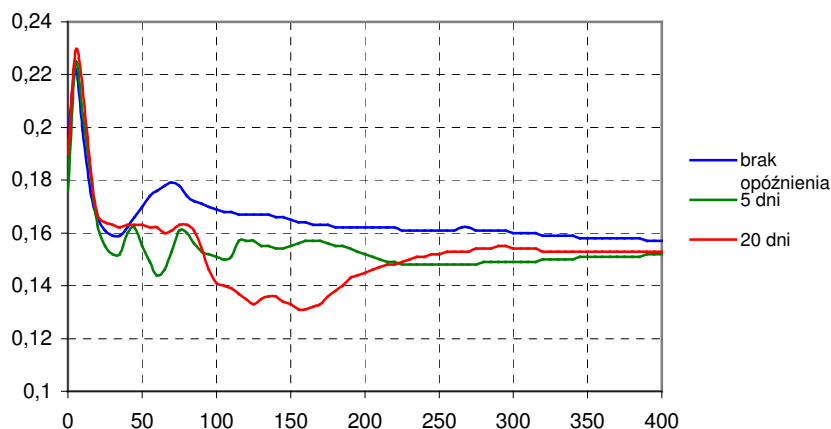
Problemem było jednak to, że jak wcześniej wspomniano, trudno jest znaleźć wystarczająco długi trend będący cały czas w stanie równowagi. Należy też obrać odpowiednią tolerancję, przekroczenie której dopiero sygnalizuje zmianę trendu.

### 6.1.3 Wpływ kształtu przebiegu danych na jakość nauczania

Podawanie na wejście surowych danych bez odpowiedniego przetworzenia, powodowało że sieć nie była zdolna nauczyć się zbioru treningowego. Powodem było zbyt wiele bardzo różniących się przebiegów, które nigdy nie mogły się powtórzyć. Zastosowanie średniej kroczącej do wygładzenia przebiegu umożliwiło sieci poprawne nauczanie się przebiegu. Zmniejszenie standardowego odchylenia wartości poszczególnych punktów przyczyniło się do zneutralizowania przez sieć w trakcie uczenia chwilowych zawahań kursów akcji nie mających większego wpływu na długoterminowy kierunek podążania trendu.



Rys. 6.11 Wycinek badanego przebiegu bez opóźnień, z 5 dniowym opóźnieniem oraz z opóźnieniem 20 dniowym



Rys. 6.12 Wartość błędu treningu w zależności od wartości opóźnienia przebiegu uczącego

Na rys. 6.12 można zauważyć, iż przebieg bez opóźnień nie potrafił nauczyć się odwzorowania, stopniowo pomniejszał błąd uczenia ale mógł on być okupiony przetrenowaniem sieci. Większe opóźnienia charakteryzowały się coraz to niższymi wartościami błędu. Decyzja o zaprzestaniu uczenia w punktach osiągnięcia minimum wydaje się być wystarczająca.

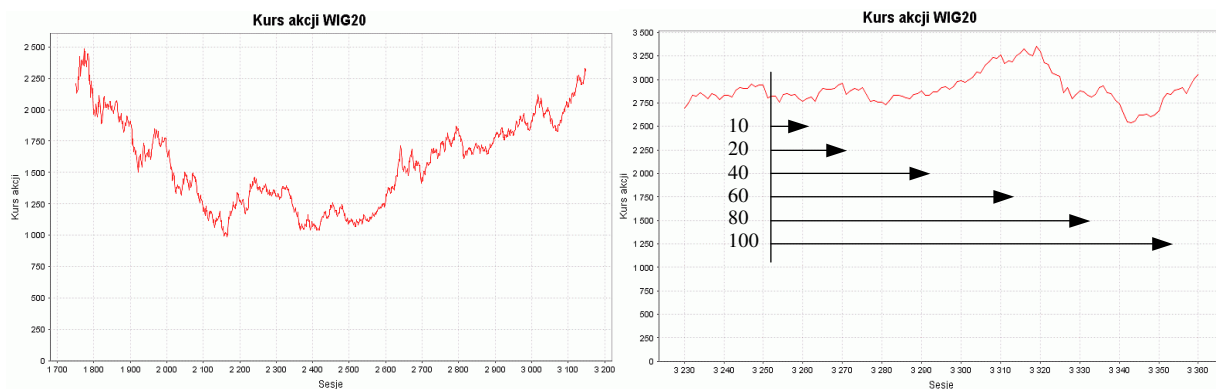
#### 6.1.4 Zależność jakości predykcji od czasu trwania predykcji

Sama konstrukcja systemu uniemożliwia dokonywanie prognoz intraday<sup>44</sup>. Trudne też będzie dokonywanie prognozy na następny dzień ze względu na to, że pojedyncza wartość może być obciążona błędem i na jej podstawie niemożliwe będzie dokonanie analizy kierunku zmian. Rozwiązaniem tu mogło by być zastosowanie następującego kryterium: notowania wzrosną, zmaleją lub nie zmieniają swojej wartości ale sam system był z założenia budowany dla badania co najmniej 5 sesji naprzód. Informacja o 20 sesjach naprzód (mniej więcej kalendarzowy miesiąc) pozwala na inwestycje dłuższe niż krótkoterminowe.

W analizie wykorzystano notowania indeksu WIG20, sieć uczono danymi z dość rozległego zbioru (od sesji 1750 do sesji 3150) dobranego tak, aby zawierał w miarę równomierną ilość zarówno trendu rosnącego jak i malejącego. Na rys. 6.13 zaprezentowano przebieg danych uczących oraz przebieg, który predykowano (sesja początkowa 3250).

<sup>44</sup> Umożliwia to śledzenie przebiegu notowań w trakcie trwania sesji



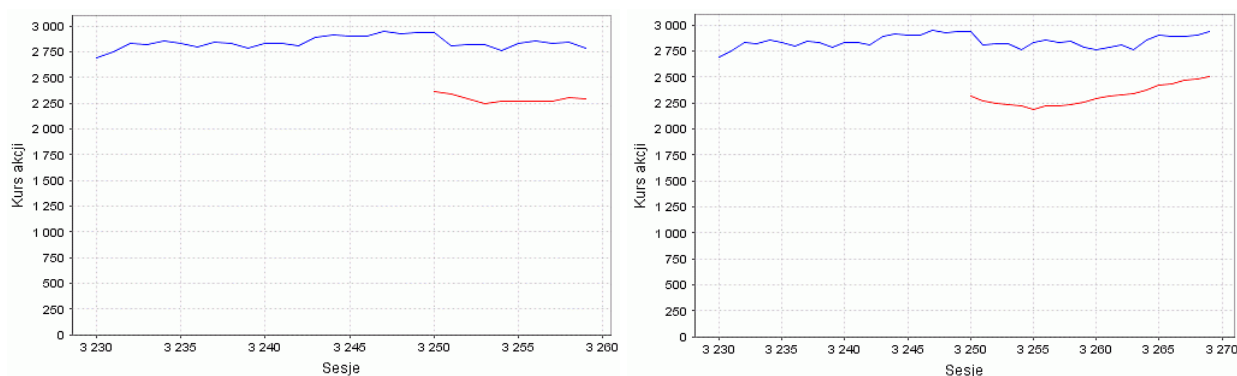


Rys. 6.13 Zbiór uczący oraz wykres kursu akcji WIG20 który jest celem tworzonej predykcji.

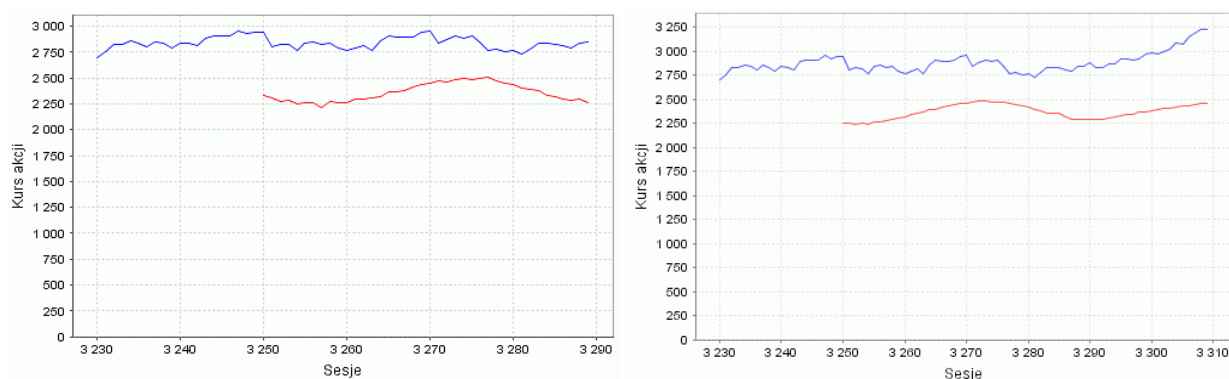
W doświadczeniu wykonano 6 predykcji dla okresu 10, 20, 40, 60, 80 i 100 dni, za każdym razem budując nową sieć neuronową zmieniając jedynie potrzebną ilość neuronów w warstwie ukrytej (od 20 – 360) w zależności od ilości wyjść determinowanej długością predykcji.

Wykonano 6 modeli sieci tak aby każdy jak najlepiej dostosował się do długości prognozy.

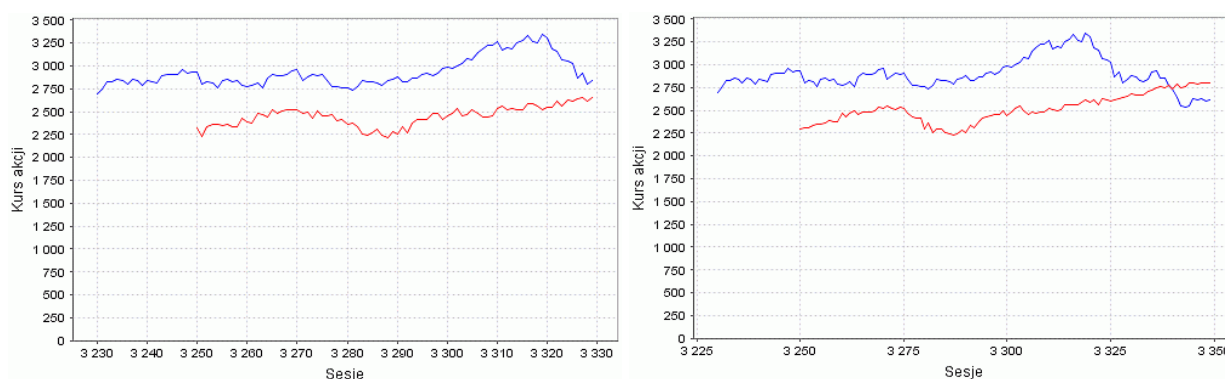
Otrzymane wyniki zamieszczone są na poniższych rysunkach:



Rys. 6.14 Wyniki predykcji: 10 i 20 dniowej dla WIG20



Rys. 6.15 Wyniki predykcji: 40 i 60 dniowej dla WIG20



Rys. 6.16 Wyniki predykcji: 80 i 100 dniowej dla WIG20

Analizując wykresy można zauważyć, że o ile w zakresie pierwszych 60 notowań (jeżeli prognoza obejmowała aż taki okres) wyniki sieci są poprawne, o tyle powyżej tego okresu widać, że sieci miały problem z wykryciem dość wyrazistego szczytu. Próbowaliśmy w zamian za to sygnalizować spokojny trend wzrostowy przez całą pozostałą długość. Wyniki i tak można uznać za dobre, ponieważ 60 notowań giełdowych to okres trzech miesięcy kalendarzowych wystarczający na średnioterminowe inwestycje. Należy jednak podkreślić fakt, że w/w prognoza miała miejsce w trakcie trwania trendu wzrostowego (styczeń – czerwiec 2006) z korektą w międzyczasie (sesja 3320 – 12 maj 2006) do wykrycia której potrzebne były aktualne dane (np. wolumen) lub informacja o nastrojach panujących na giełdzie.

Należy zauważyć zatem jak wielki wpływ na prognozę ma posiadanie aktualnych danych i jak bardzo będzie zawsze ograniczona możliwość dokonywania prawidłowej prognozy. Sieć prawidłowo odtworzyła wzrostowy trend, który dalej był kontynuowany, jednak prognoza nie oddawała szczegółów przebiegu kursu akcji.

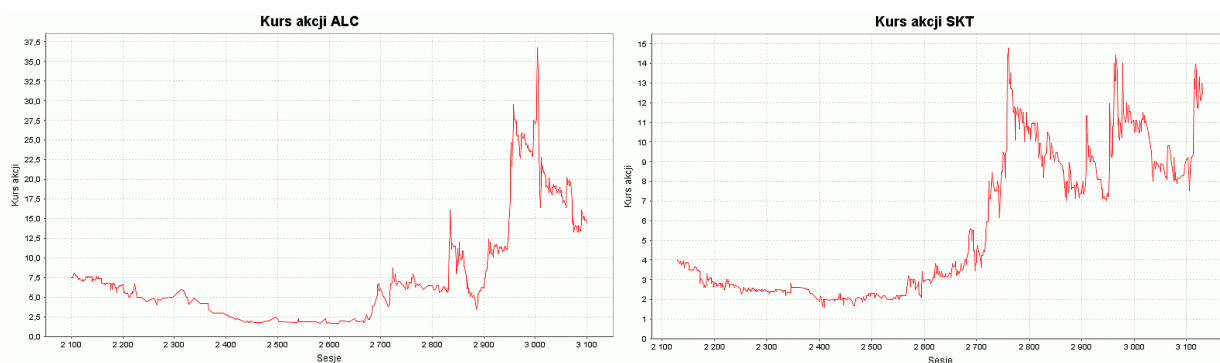
### 6.1.5 Aktualizacja zbioru uczącego podczas uczenia sieci

W podrozdziale [6.1.2] został opisany wpływ poprawności danych zbioru uczącego na jakość predykcji, wymagający obecności reprezentantów charakterystycznych trendów w notowaniach akcji. Założono tam jednak niezmiennosć zbioru uczącego w trakcie uczenia sieci. W poniższym podrozdziale spróbowano na przykładzie spółek osiągających bardzo znaczne wzrosty (a co za tym idzie nie mających w żaden sposób odzwierciedlenia w notowaniach historycznych) aktualizować cyklicznie tenże zbiór włączając do niego nowe nieobecne dotąd przypadki. W ten sposób, co 20 sesji dokonywano nowej prognozy na kolejne 20 sesji.

W drugiej połowie 2005 roku a także w pierwszej połowie roku 2006 na WGPW miało miejsce wiele spektakularnych wzrostów spółek giełdowych. Uzyskane stopy zwrotu z inwestycji mogły wynosić 1000% (spółka Alchemia) lub nawet blisko 2000% (spółka Skotan) w przeciągu

około 150 sesji giełdowych (około 8 miesięcy kalendarzowych). Powodów tak dużego przyrostu wartości należy szukać m.in. w uprzednio zaniżonych notowaniach w/w spółek, w opublikowaniu korzystnych raportów finansowych oraz prognoz dotyczących inwestycji i zysków w przyszłości. Nie da się ukryć, że końcowe notowania spółek były nadmiernie zawyżone z powodu bardzo dużego popytu na ich akcje głównie ze strony drobnych inwestorów zachęconych dotychczasowymi wręcz niebotycznymi zyskami.

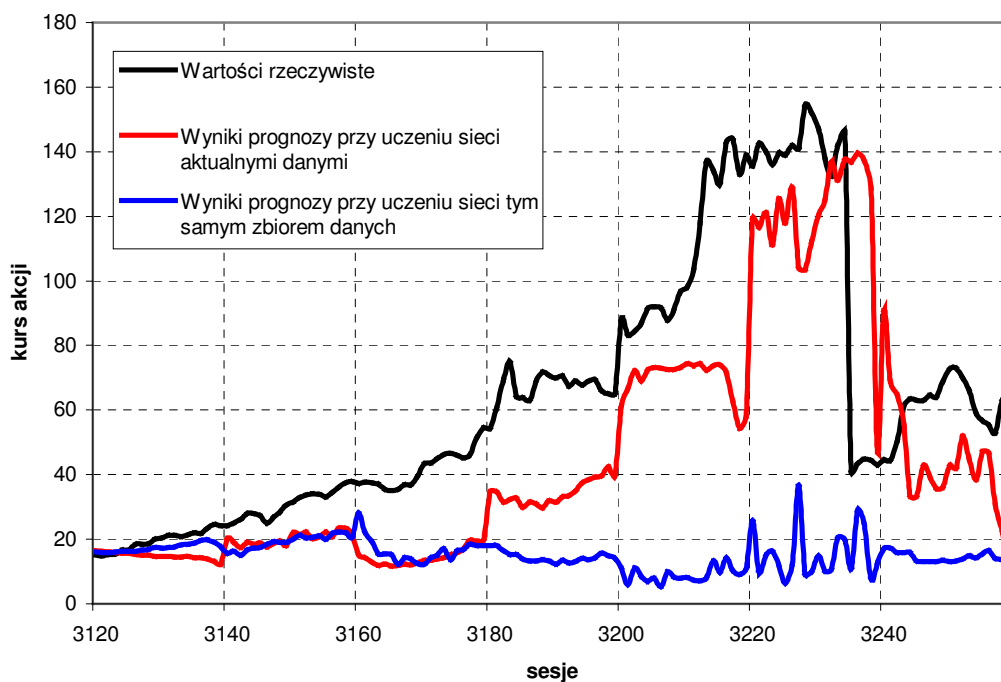
Spróbowano zatem sprawdzić jak zbudowany system poradziłby sobie w predykcji kolejnych notowań w/w spółek. Dokonano kilkukrotnych prognoz na kolejne 20 sesji począwszy od sesji 3120 w przypadku Alchemii oraz od sesji 3170 w przypadku Skotana. Jako zbiór uczący przyjęto ostatnie 1020 notowań. Przebieg tych notowań można zaobserwować na rys. 6.17



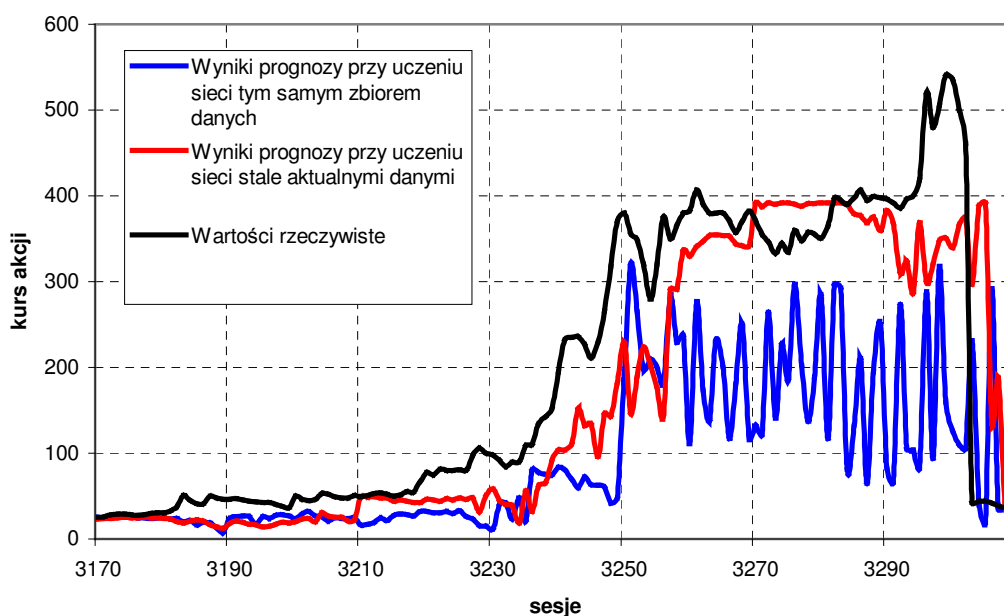
Rys. 6.17 Zbiór danych uczących dla spółki Alchemia (sesje 2100-3100) oraz dla spółki Skotan (sesje 2130-3130)

W przebiegach można odnaleźć fragmenty trendu horyzontalnego, wzrostowego jak i spadkowego. W ten sposób zapewniono nauczanie sieci dla szerokie spektrum możliwych przypadków.

Osiągnięte wyniki predykcji (rys. 6.18 i 6.19) przy stałym zbiorze danych uczących (kolor niebieski) znacząco różnią się jednak od wartości oczekiwanych (Alchemia) lub osiągają wartości zbyt mocno oscylujące, aby na ich podstawie można było dokonać prawidłowej decyzji inwestycyjnej. Drobne usprawnienie w postaci przesuwania zbioru uczącego o kolejne 20 sesji (kolor czerwony) spowodowało, że sieć potrafiła już dużo lepiej predykować.



Rys. 6.18 Prognoza notowań spółki ALCHEMIA



Rys. 6.19 Prognoza notowań spółki SKOTAN

Charakterystyczny w przebiegach notowań obu spółek jest gwałtowny spadek w końcowej fazie notowań. Wyniki sieci uczonej stałym zbiorem danych nie wykryły zmiany trendu, jej prognozy są jak najbardziej niepoprawne. Nie powinno to jednak dziwić, ponieważ sieć nie

uksztaltowała wyraźnego szczytu obecnego w rzeczywistych notowaniach, a tym samym trudno, aby zasygnalizowała gwałtowny z niego spadek.

Sieć uczona według drugiej metody, co prawda w sposób opóźniony zasygnalizowała gwałtowny spadek, ale to opóźnienie (około 5 sesji) jest dużo mniejsze od okresu wykonywanej prognozy. Przy tak długoterminowej inwestycji powyższy błąd jest nieznaczący.

Nie należy jednak w tych konkretnych przypadkach przypisywać sieciom neuronowym nadzwyczajnych właściwości. Spadki w obu przypadkach od dłuższego czasu były sygnalizowane m.in. przecięciem się średnich kroczących [4.2.2], przebicciem od góry wartości 70% wskaźnika *RSI* [4.2.2] a także przecięciem średniej *SIGNAL* i wskaźnika *MACD* [4.2.2].

Reasumując: Zachodzi zatem konieczność uczenia od nowa lub tzw. douczania sieci w pewnych określonych sytuacjach. Nauczenie jednokrotne sieci i następnie korzystanie z jej wyników w dłuższym okresie czasu nie zawsze będzie dawało pozytywne wyniki.

## 6.2 Optymalizacja topologii sieci algorytmem genetycznym

Poszukiwanie optymalnej kombinacji wejść jakiego dokonywano w punkcie 6.1 w przypadku jednej spółki giełdowej doprowadziło do osiągnięcia wymiernych sukcesów, jednak obarczone było ono czasem koniecznym do przeanalizowania bardzo wielu kombinacji topologii sieci neuronowej. W przypadku większej ilości spółek giełdowych powyższa metoda przypomina błądzenie przypadkowe. W powyższym rozdziale poszukiwaniem optymalnego rozwiązania zajmie się algorytm genetyczny.

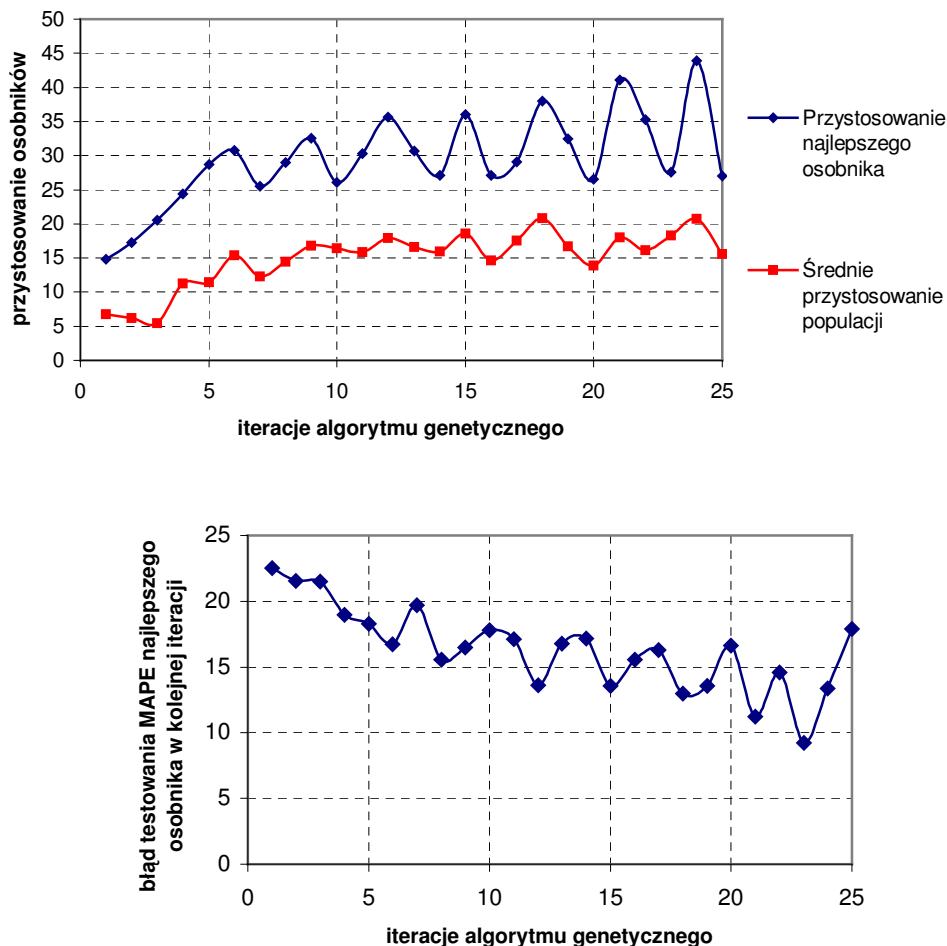
Systemowi Amadeus powierzono zadania takie same jak w punkcie 6.1, poszukiwanie ograniczono wyłącznie do zbioru uczącego zawierającego wszystkie charakterystyczne trendy.

Korzystając z wcześniejszych wyników, dokonano wymienionych poniżej wstępnych założeń. Minimalna ilość typów danych podawanych na wejście jest liczbą z zakresu [2..5].

Populację początkową tworzy 20 osobników, przyjęto krzyżowanie dwupunktowe oraz możliwość mutowania. Na zamieszczonych poniżej wykresach widać, że wraz ze wzrostem iteracji, przystosowanie najlepszego osobnika staje się coraz wyższe. A zatem korzystając z algorytmu genetycznego otrzymuje się coraz to lepszego „jakościowo” osobnika.

Przebieg posiada charakterystyczne pulsacje, w których po każdym szczycie osiąga niższą wartość niż poprzednio ale i tak w kolejnych iteracjach osiągnie wynik lepszy niż dotychczas. Jeżeli w trzech kolejnych iteracjach wynik się nie poprawiał następował koniec tworzenia nowych pokoleń. Wraz ze wzrostem przystosowania najlepszego osobnika również wzrastało średnie przystosowanie a to oznaczało, że kolejni osobnicy byli tworzeni z lepszych jakościowo

rodziców. Oczywiście mogło zdarzyć się wystąpienie jednorodności rozwiązań, temu zjawisku przeciwdziała wprowadzenie nowych osobników i mutacja, której prawdopodobieństwo zajścia zostało ściśle powiązane z tym czy reprezentacja pokolenia przedstawiała podobne czy też różne rozwiązania.



Rys. 6.20 Wykres przyrostu przystosowania osobników w kolejnych iteracjach oraz malenia minimalnego błędu testowania MAPE.

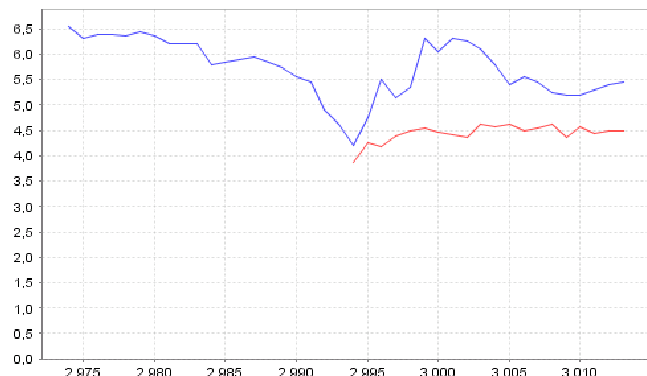
W efekcie końcowym za najlepszego reprezentanta w trakcie całej ewolucji uznano osobnika z 24 pokolenia, który osiągnął najwyższe przystosowanie.

Z informacji zapisanych w chromosomie tego osobnika można odczytać, że najlepsze wyniki osiągnęła sieć posiadająca 26 dniowe opóźnienie, dwa wejścia: cena zamknięcia i wolumen, posiadająca jedną warstwę ukrytą, w której znalazło się 18 neuronów.

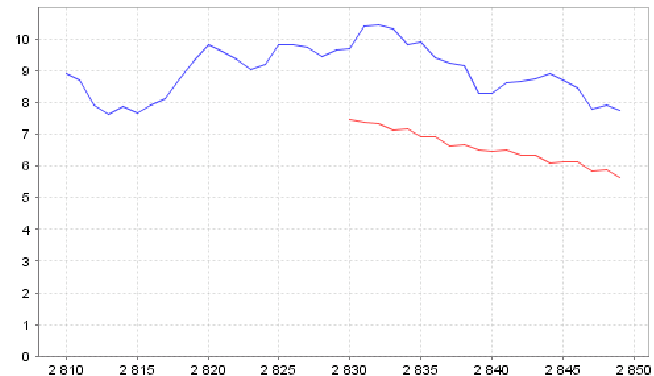
Dla tej sieci dokonano prognozy zachowania dla charakterystycznych punktów wykresu (rys 6.8), wyniki zostały zaprezentowane na rys 6.21. Na poniższych wykresach wartości predykcji

ulegają odchyleniom od wartości rzeczywistych, jednak należy brać pod uwagę przede wszystkim sam kształt oraz kierunek i zmianę procentową przebiegu.

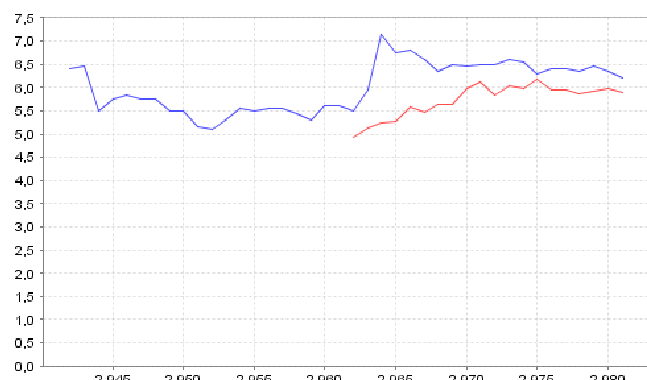
Przejście trendu spadkowego w trend wzrostowy



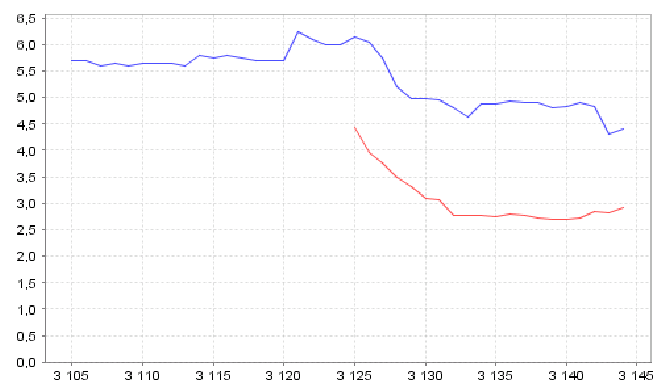
Przejście trendu wzrostowego w trend spadkowy



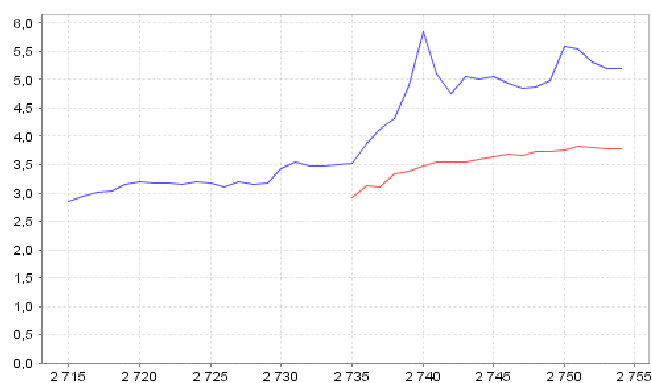
Przejście trendu horyzontalnego w trend wzrostowy



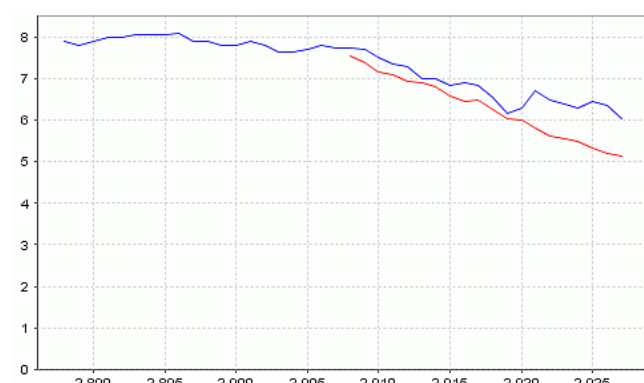
Przejście trendu horyzontalnego w trend spadkowy



Kontynuacja trendu wzrostowego



Kontynuacja trendu spadkowego



— Rzeczywiste wartości notowań  
— Wynik prognozy

Rys. 6.21 Reprezentacja wyników prognozy oraz rzeczywistych wartości notowań spółki Elektrim S.A. [Wykres pochodzi z systemu Amadeus]

Jak widać sieć poradziła sobie bardzo dobrze z rozpoznaniem zmian notowań. Jakościowo wartości prognozowane różniły się od przebiegów rzeczywistych, jednak tak przedstawione wyniki pozwoliłyby potencjalnemu inwestorowi dokonać trafnej decyzji inwestycyjnej. Skonstruowanie funkcji przystosowania zakładające przede wszystkim predykcje stopy zwrotu powodowało, że przebiegi nie zawsze odzwierciedlały zmiany krótkoterminowe, które nie były bardzo istotne przy analizie.

## 6.3 Podsumowanie

Przeprowadzone doświadczenia wykazały, że istnieją przesłanki ku temu, że pomimo braku jednoznacznego opisu rynku giełdowego przy spełnieniu pewnych wymogów sieć neuronowa potrafiła dokonać skutecznej prognozy w większości zaprezentowanych przypadków[6.2].

Równocześnie zauważono, że jakość predykcji była niska kiedy zbiór treningowy był zdeterminowany jedynie notowaniami jednego trendu, a także podawany zbiór danych (zakres, rodzaj) był niewystarczający z punktu widzenia analizy technicznej.

Wybranie zbioru treningowego, w którym zawarta była możliwie największa ilość charakterystycznych zachowań rynku giełdowego, a także dodanie oprócz ceny zamknięcia informacji o wartości wolumenu lub zależności ceny najwyższej i zamknięcia sesji, spowodowały wyraźną poprawę jakości prognoz. Dodatkowo odpowiednio dodany szum (o ograniczonej stałej amplitudzie) do zbioru treningowego zwiększył możliwości generalizacji, gdyż już nie tylko podobieństwa o tym samym rzędzie wielkości mogły być wykorzystywane w procesie decyzyjnym.

Działo się tak dlatego, że sieć potrafiła odwzorować przypadki z przeszłości i na zasadzie podobieństwa przełożyć na zachowania w przyszłości, które były celem dokonywania prognoz.

Należy jednak pamiętać o tym, że sieć, która otrzyma na wejście zbyt dużą liczbę nieistotnych szczegółów nie usuniętych lub nie przetworzonych (np. uśrednionych za pomocą średniej kroczącej) w trakcie *preprocessingu* może w rezultacie końcowym dokonywać gorszych uogólnień. Lepiej jest [6.1.3] uśrednić przebiegi wejściowe, tracąc tym samym chwilowe fluktuacje notowań ale wyraźnie wzmacniając obraz kształtowanego trendu.

Sieć dobrze aproksymowała rzeczy powtarzalne, niekoniecznie o tych samych wartościach, wystarczało jej jedynie ogólne podobieństwo pomiędzy wielkościami. Zaletą sieci neuronowej jest to, że do budowy takiego systemu nie jest konieczna wiedza na temat działania samego rynku giełdowego. Zwiększanie zdolności predykcyjnych sieci odbywa się wyłącznie poprzez



uczenie się sieci na danych historycznych, wykorzystując podstawową przesłankę analizy technicznej mówiącą, że historia lubi się powtarzać.

Nie było zatem potrzeby stosować skomplikowanych algorytmów zachowań giełdowych, które same w sobie stanowiłyby trudność implementacyjną, gdyż każda ze spółek posiada specyficzną charakterystykę notowań.

Sieć, podobnie jak człowiek, nie jest w stanie przewidzieć sytuacji losowych (kataklizmów, wojen). Trudno jej jednak nauczyć się tzw. *psychologii rynku giełdowego*, którą posługuje się w swoich decyzjach inwestycyjnych makler giełdowy. Na korzyść sztucznych sieci neuronowych przemawia większa moc przetwarzania

Sieć, pomimo wielu modyfikacji, nie potrafiła jednoznacznie rozpoznawać trendu horyzontalnego. Spowodowane to być mogło brakiem odpowiednio dużego zbioru treningowego, gdyż trend horyzontalny nie jest zbyt częstym przypadkiem zachowania giełdowego. Niemożność wykorzystania sieci dla tego przypadku objawiała się tym, że sieć odczytywała już drobne zmiany wartości cen jako sygnał zmiany trendu.

Przeprowadzone doświadczenia związane z różnymi okresami predykcji wykazały istnienie granicy możliwości przeprowadzania dłuższych prognoz [6.1.4]. Wartość okresu predykcji dla której podejmowane decyzje są ważne zależy od wielu czynników (branża, nastroje na giełdzie etc.) ale bez dostarczania aktualnych danych sieć nie jest w stanie długoterminowo dokonać predykcji.

Modyfikacje polegające na optymalizacji topologii poprawiały wynik jakościowy [6.1.1] ale w głównej mierze kierunek predykcji był prawidłowy. Wykorzystanie algorytmu genetycznego [6.2] do zoptymalizowania topologii sieci neuronowej umożliwiło odnalezienie takiej kombinacji parametrów sieci przy której odznaczała się ona najwyższym wskazaniem funkcji przystosowania. W zależności od tego, z jakich czynników składała się funkcja przystosowania można było wybrać priorytety, którymi optymalizacja się kierowała. Mogły nimi być wartość błędu testowania, zgodność kierunku zmian trendów, a także np. czas uczenia sieci, w zależności od tego który z czynników wydaje się być bardziej cenny.

W trakcie ewolucji genetycznej obserwowano stałe poprawianie się jakości osobników w kolejnych iteracjach, co udowadniało poprawność działania algorytmu genetycznego.

Badania przeprowadzone w pracy wykazały, że najlepsze właściwości charakteryzowały algorytm genetyczny składający się z kilkunastu do kilkudziesięciu osobników w populacji. Ich mniejsza ilość szybko doprowadzała do zbieżności i w efekcie sieć szybko nie mogła osiągać lepszych wyników. W wyniku doświadczeń zauważono, że najkorzystniejsza sytuacja

występowała, kiedy w populacji można było wyróżnić kilku „liderów”, których potomstwo pozyskiwało coraz to lepsze cechy. Zbyt mała ilość osobników w populacji powodowała, że w populacji przewodził tylko jeden taki osobnik, a poprawianie się wyników mogło odbywać się jedynie poprzez mutowanie, bądź krzyżowanie z nowym osobnikiem. Obserwując jednak wykres średniego przystosowania osobników widać, że średnia ta rośnie, a zatem im dalsza iteracja tym krzyżowanie z osobnikiem z populacji może owocować lepszym potomstwem niż krzyżowanie z nowym osobnikiem. Wprowadzenie nowego osobnika do populacji eliminuje przedwczesną zbieżność ale nie ma na celu poprawy samego wyniku najlepszego przystosowania.

Bardzo ważne jest zapewnienie odpowiednich metod zapewniających opóźnienie zbieżności tak, aby algorytm ciągle był zdolny odkrywać nowe rozwiązania. W tym celu wprowadza się nowe osobniki co iterację, krzyżując dwupunktowo osobniki, wykorzystując mutowanie osobników.

Dodatkowo w sytuacji, gdy w notowaniach historycznych nie istnieją podobne zachowania notowań giełdowych, aktualizacja zbioru uczącego z każdym krokiem kolejnej predykcji umożliwia dołączanie do zbioru uczącego nowych – bieżących sytuacji [6.1.5] pozwalając dokonywać sieci poprawnych decyzji w oparciu o szerszy zbiór możliwych rozwiązań.

Reasumując: Wykazano przydatność sieci neuronowych do dokonywania prognozy stopy zysku. W tym celu muszą być spełnione pewne warunki początkowe. Wartości predykcji mogą odbiegać od rzeczywistych wartości ale będą dążyć w prawidłowym kierunku.

---

## 7. WNIOSKI

Liczne publikacje (m.in. [Gately]) wykazywały już wcześniej przydatność użycia sieci neuronowych do dokonywania prognoz giełdowych, dlatego spostrzeżenia z przeprowadzonych doświadczeń zamieszczone w podsumowaniu rozdziału [6] nie powinny być zaskakujące. Wnioski po przestudiowaniu przesłanek analizy technicznej, a także ich wykorzystanie w praktyce dowodzą, że w pewnym ograniczonym stopniu zachowanie rynku giełdowego jest jak najbardziej możliwe do przewidzenia.

Zaimplementowanie powyższej wiedzy z dziedziny analizy giełdowej umożliwia zbudowanie systemu eksperckiego, którego zadaniem byłoby ogólnie rzecz mówiąc dostarczenie informacji pomagającej inwestorowi. Wykorzystanie wskaźników takich jak np. *RSI*, *MACD*, *ROC* a także szeregu innych specjalistycznych wskaźników nie omówionych w tejże pracy pozwalałoby z dużym prawdopodobieństwem generować sygnały kupna lub sprzedaży. Do budowy takiego systemu niezbędna jednak byłaby specjalistyczna wiedza z zakresu rynku giełdowego.

Sama analiza wykresu kursu akcji bez odpowiedniego przetworzenia nie zawsze dostarcza jasnych wskazówek co do kształtowania się trendu, dlatego często próba dokonania odpowiedzi na pytanie: „jak będzie się kształtował trend w najbliższych  $x$  dniach?” może przypominać próbę odgadnięcia wyniku w trakcie rzutu monetą.

Dane historyczne którymi posłużono się w pracy powinny zostać odpowiednio przetworzone (poprzez czyszczenie, integracje, transformacje oraz redukcje danych) zgodnie z wyznaczonym zadaniem. Pominięcie któregoś etapu przygotowania danych skutkuje pogorszeniem jakości generowanych prognoz bądź wręcz uniemożliwia otrzymanie żadanego rezultatu. Dlatego też tak istotne jest zbudowanie prawidłowo działających mechanizmów przechowywujących i przygotowujących dane historyczne oraz wyniki prognoz.

Dopiero dla tak przygotowanych danych wejściowych sieć neuronowa w trakcie kolejnych iteracji algorytmu genetycznego, zajmującego się doбором optymalnej topologii potrafiła zwiększać jakość swojej predykcji. Kolejne pokolenia charakteryzowały się coraz to lepszymi cechami zapewniając coraz lepszą jakość predykcji sieci zbudowanej w oparciu o przekazywane przez osobników parametry.

Sieć neuronowa sama znajdowała powiązania pomiędzy pewnymi wielkościami dostarczonymi jej w zbiorze danych historycznych. Dane historyczne były jedyną „wiedzą” dostarczoną sieci, która ucząc się z osobna dla każdego waloru giełdowego potrafiła rozpoznać jego charakterystyczne zachowania inne niż dla innych walorów.

Nie oznacza to jednak, że sieć, niezależnie od tego jak zostanie zbudowana i jakie dane jej się dostarczy, prędzej czy później odnajdzie prawidłowe rozwiązanie. Sieć taka będzie charakteryzować się znacznie większą ilością błędnych decyzji dyskryminując w ten sposób jej użycie.

Na podstawie uzyskanych doświadczeń w trakcie pracy można wysnuć wniosek, że dobrze opracowany model systemu potrafi tworzyć decyzje inwestycyjne równie wartościowe co inwestycje przeprowadzone przez analityka giełdowego, którego decyzje inwestycyjne oparte są wyłącznie na podstawie analizy technicznej bez uwzględnienia innych zewnętrznych czynników. Sieć w trakcie uczenia (doboru wag połączeń między neuronami) zdobywała wiedzę na temat rynku giełdowego poprzez dosłowną „naukę na błędach”. Była niejako karana za błędne decyzje (brak dalszej ewolucji takiego osobnika) oraz równocześnie nagradzana (ilość osobników potomnych większa od zera) w przeciwnych przypadkach.

Niestety brak informacji o danych fundamentalnych oraz obecnej „psychologii rynku giełdowego” powoduje, że nie zawsze decyzje podjęte przez system będą poprawne. Byłyby one poprawne w sytuacji, gdyby aktualne sytuacje gospodarczo-ekonomiczne oraz polityczne w obu okresach czasu były podobne.

Oczywiście możliwe jest rozwinięcie systemu o analizę fundamentalną spółek giełdowych ale ze względu na trudność realizacji porzeczono jedynie na analizie technicznej.

---

# BIBLIOGRAFIA

- [Berry] Michael J.A.Berry, Gordon S.Linoff - *Data Mining Techniques For Marketing, Sales, and Customer Relationship Management Second Edition*, WILEY 2004
- [Bigus] Bigus Joseph P. - *Data Mining with Neural Networks (Solving Buisness Problem from Application Development to Decision Support)*, MCGRAW-HILL 1996
- [Brause] R. Brause, T. Langsdorf, M. Hepp - *Neural Data Mining for Credit Card Fraud Detection*, UNIWERYSTET GOETHEGO FRANKFURT NAD MENEM 1999
- [Dasu] Dasu T., Johnson T. – *Exploratory Data Mining and Data Cleaning* WILEY 2003
- [Elder] A. Elder – *Zawód inwestor giełdowy* OFICyna EKONOMICZNA ODDZIAŁ POLSKICH WYDAWNICTW PROFESJONALNYCH 2001
- [Fausett] Laurene Fausett – *Fundamentals of Neural Networks: Architectures, algorithm and applications* PRENTICE HALL 1994
- [Fayyad] U. Fayyad, G. Piatetsky-Shapiro, *Advances in Knowledge Discovery and Data Mining*, MIT PRESS, 1996
- [Gately] Ed Gately – *Sieci neuronowe, prognozowanie finansowe i projektowanie systemów transakcyjnych* WIG-PRESS 1999
- [Goldberg] Goldberg David E. – *Algorytmy genetyczne i ich zastosowania* WNT 2003
- [Grygiel] Grygiel K. – *Wstęp do obliczeń ewolucyjnych i neuronowych* INSTYTUT INFORMATYKI UNIWERSYTETU WARSZAWSKIEGO 2005
- [Han] Han J., Kamber M. – *Data Mining: Concepts and Techniques*. MORGAN KAUFMANN 2000
- [Hand] Hand David J., Heikki Mannila and Padhraic Smyth – *Principles of Data Mining*
- [Iebling] Kaastra Iebling, Boyd Milton – *Designing a neural network for forecasting financial and economic time series* NEUROCOMPUTING 10/1996 str 215-236
- [Kantardzic] Mehmed Kantardzic - *Data Mining: Concepts, Models, Methods, and Algorithms* WILEY 2003
- [Larose] Larose Daniel T.– *Discovering Knowledge in Data (An Introduction to Data Mining)* WILEY 2005

- [Michalewicz] Michalewicz Z. – *Genethic Algorithms + Data Structures = Evolution Programs* SPRINGER 1992
- [Murawski] Krzysztof Murawski - *Obliczenia ewolucyjne geneza i zastosowanie*  
BIULETYN INSTYTUTU AUTOMATYKI I ROBOTYKI WAT NR 15/2001
- [Murphy] Murphy J.J – *Analiza techniczna rynków finansowych* WIG-PRESS 1999
- [Najman] Kamila Migdał Najman, Krzysztof Najman, *Sieci neuronowe – wykorzystanie do prognozowania WIG*, Profesjonalny Inwestor, nr 8/2000, str. 10-18
- [Nong] Ye Nong - *Handbook of Data Mining* LAWRENCE ERLBAUM ASSOCIATES PUBLISHERS 2003
- [Osowski] Osowski Stanisław – *Sieci neuronowe do przetwarzania informacji* Oficyna Wydawnicza Politechniki Warszawskiej 2000
- [Plummer] Plummer T. - *Psychologia rynków finansowych* WIG-PRESS 1995
- [Tadeusiewicz] Tadeusiewicz Ryszard – *Sieci neuronowe* Akademicka Oficyna Wydawnicza RM 1993.
- [Tarczyński] Tarczyński W. – *Rynki kapitałowe* Agencja Wydawnicza PLACET 1997
- [Wyrozumski] Wyrozumski Tomasz - *Sieci neuronowe a energetyka prawdy i mity o prognozowaniu* X Konferencja PLOUG 2004

---

## SPIS ILUSTRACJI

RYS. 1.1 PORÓWNANIE ZADAŃ MODELI PRZETWARZANIA: Q&R, OLAP I DATA MINING .....	10
RYS. 1.2 ILOŚĆ SPÓŁEK GIEŁDOWYCH NOTOWANYCH NA WGPW .....	11
RYS. 1.3 PRZEBIEG PROCESU ODKRYWANIA WIEDZY .....	12
RYS. 1.4 RODZAJE I UDZIAŁ DANYCH W PROCESIE EKSPLORACJI DANYCH.....	13
RYS. 1.5 KOSTKA DANYCH PRZEDSTAWIA TRÓJWYMIAROWY CHARAKTER DANYCH.....	13
RYS. 1.6 PRZYKŁAD BRAKUJĄCYCH DANYCH .....	14
RYS. 1.7 PRZYKŁAD ZASZUMIONYCH DANYCH.....	15
RYS. 1.8 ZASZUMIONE DANE PRZED (KOLOR CZARNY) I PO WYGŁADZENIU (KOLOR NIEBIESKI) ..	16
RYS. 1.9 PRZYKŁAD DANYCH UZNANYCH ZA NIEZGODNE .....	17
RYS.1.10 PRZYKŁAD KORELACJI DANYCH .....	18
RYS. 1.11 PRZYKŁAD KONFLIKTU DANYCH ZWIĄZANYCH Z PRZYJĘCIEM RÓŻNYCH JEDNOSTEK	18
RYS. 1.12 NAJWAŻNIEJSZE TECHNIKI EKSPLORACJI DANYCH.....	20
RYS. 1.13 PRZYKŁAD TZW. KOSZYKA ZAKUPÓW .....	21
RYS. 1.14 PRZYKŁAD DRZEWA DECYZYJNEGO PRZEDSTAWIAJĄCEGO ZMIANĘ TRENDU W ZALEŻNOŚCI OD CEN ZAMKNIĘCIA, OTWARCIA I WOLUMENU .....	21
RYS. 1.15 GRUPOWANIE DWUNASTU OBIEKTÓW W CZTERY DWUWYMIAROWE KLASTRY W ZALEŻNOŚCI OD WIEKU I OSIĄGANEGO DOCHODU .....	22
RYS. 1.16 LISTA ZAKUPÓW KLIENTÓW W CZASIE .....	23
RYS. 1.17 NOTOWANIA SPÓŁEK A I B RÓŻNIĄ SIĘ WARTOŚCIAMI SĄ TAKŻE WZAJEMNIE PRZESUNIĘTE W CZASIE, ODZNACZAJĄ SIĘ JEDNAK PODOBNĄ CHARAKTERYSTYKĄ ....	24
RYS. 1.18 ZASTOSOWANIA EKSPLORACJI DANYCH Z PODZIAŁEM NA BRANŻE .....	25
RYS. 2.1 BUDOWA KOMÓRKI NERWOWEJ.....	28
RYS. 2.2 MODEL KOMÓRKI NERWOWEJ WEDŁUG MCCULLOCHA-PITTSA.....	29
RYS. 2.3 MODEL NEURONU SIGMOIDALNEGO .....	30
RYS 2.4 PRZEBIEG FUNKCJI SIGMOIDALNEJ. ....	31
RYS 2.5 RODZAJE SIECI NEURONOWYCH WRAZ Z NAZWISKAMI AUTORÓW I DATĄ POWSTANIA..	32
RYS. 2.6 SIEC NEURONOWA JAKO UKŁAD ADAPTACYJNY .....	33
RYS. 2.8 RÓŻNICE W UCZENIU DLA WSPÓŁCZYNNIKA STAŁEGO W CZASIE ORAZ DLA DYNAMICZNIE MODYFIKOWANEGO .....	40
RYS. 2.9 REPREZENTACJA ZBIORU UCZĄCEGO L, ZBIORU TESTUJĄCEGO G I ZBIORU SPRAWDZAJĄCEGO V SPEŁNIAJĄCYCH PEWNĄ REGUŁĘ R .....	40
RYS. 2.10 ILUSTRACJA ZDOLNOŚCI UOGÓLNIANIA SIECI W ZALEŻNOŚCI OD ILOŚCI NEURONÓW.	41
RYS. 2.11 WPŁYW DŁUGOŚCI UCZENIA NA BŁĄD UCZENIA $E_L$ ORAZ BŁĄD GENERALIZACJI $E_G$ .....	42
RYS. 2.12 OGÓLNY SCHEMAT SIECI NEURONOWEJ SIGMOIDALNEJ DWUWARSTWOWEJ. ....	44
RYS. 2.13 KLASYFIKACJA I ROZPOZNAWANIE.....	45
RYS. 2.14 AUTOASOCJACJA I HETEROASOCJACJA.....	45
RYS. 3.1 GRAFICZNA REPREZENTACJA PODSTAWOWYCH POJĘĆ GENETYKI KOMPUTEROWEJ .....	47
RYS. 3.2 WYKRES FUNKCJI JEDNOMODALNEJ I WIELOMODALNEJ .....	49
RYS. 3.3 OGÓLNY SCHEMAT DZIAŁANIA ALGORYTMU GENETYCZNEGO .....	51
RYS. 3.4 OPERACJA REPRODUKCJI ZA POMOCĄ RULETKI.....	53

RYS. 3.5 PRZYKŁAD PORÓWNUJĄCY METODĘ RULETKI I LISTY RANKINGOWEJ.....	54
RYS. 3.6 SCHEMAT KRZYŻOWANIA PROSTEGO DWÓCH OSOBNIKÓW RODZICIELSKICH .....	55
RYS. 3.7 SCHEMAT PROCESU MUTACJI.....	56
RYS. 4.1 PRZYKŁAD NOTOWAŃ WYBRANYCH WALORÓW NA SESJI 3200.....	61
RYS. 4.3 PROSTA ŚREDNIA KROCZĄCA - SMA.....	66
RYS. 4.4. WYKŁADNICZA ŚREDNIA KROCZĄCA .....	66
RYS. 4.8 FORMACJA GŁOWY I RAMION ORAZ FORMACJA ODWRÓCONEJ GŁOWY I RAMION .....	69
RYS. 4.9 FORMACJA CZWOROKĄTA .....	69
RYS. 4.10 FORMACJE CENOWE – TRÓJKĄT SYMETRYCZNY.....	70
RYS. 4.11 FORMACJE CENOWE – TRÓJKĄT ZNIŻKUJĄCY ORAZ ZWYŻKUJĄCY .....	70
RYS. 4.12 FORMACJE CENOWE – FLAGA WZNOSZĄCA I OPADAJĄCA .....	70
RYS. 4.13 FORMACJE CENOWE – PODWÓJNY SZCZYT. ....	71
RYS. 4.14 LINIE TRENDU SPÓŁKI INTERIA PL. KOLEJNO: HORYZONTALNY, SPADKOWY, WZROSTOWY	71
RYS. 5.1 SCHEMAT BLOKOWY TRÓJWARSTWOWEJ APLIKACJI J2EE .....	74
RYS. 5.2 PODSTAWOWE MODUŁY SYSTEMU AMADEUS .....	75
RYS. 5.3 STRONA GŁÓWNA SYSTEMU AMADEUS .....	76
RYS. 5.4 MENU SYSTEMU AMADEUS – DEFINIOWANIE ZADAŃ. ....	77
RYS. 5.5 MENU SYSTEMU AMADEUS – DEFINIOWANIE SPÓŁEK.....	78
RYS. 5.6 LISTA ZADAŃ SYSTEMU AMADEUS.....	78
RYS. 5.7 LISTA PARAMETRÓW OSOBNIKÓW DLA KOLEJNYCH ITERACJI .....	79
RYS. 5.8 SZCZEGÓŁOWE PARAMETRY OSOBNIKA .....	80
RYS. 5.9 WYBÓR ZADANIA W PROCESIE TWORZENIA PROGNOZ .....	81
RYS. 5.10 TWORZENIE PROGNOZY .....	81
RYS. 5.11 WYGENEROWANA PROGNOZA DLA SPÓŁEK BIORĄCYCH UDZIAŁ W ZADANIU .....	82
RYS. 5.12 GRAFICZNA INTERPRETACJA PROGNOZY .....	82
RYS. 5.13 PARAMETRY ZWIĄZANE Z PROCESEM UCZENIA.....	83
RYS. 5.14 BUDOWA CHROMOSOMU .....	84
RYS. 5.15 STRUKTURA SIECI NA PODSTAWIE PARAMETRÓW POBRANYCH Z CHROMOSOMU. ....	85
RYS. 5.16 DIAGRAM REALIZACJI TURNIEJU W PROCESIE EWOLUCYJNYM .....	86
RYS. 5.17 KRZYWA FUNKCJI DOPASOWANIA WZGLĘDEM ODCHYLEŃ STOPY ZWROTU I BŁĘDÓW TESTOWANIA .....	87
RYS. 5.18 PRZYPADEK NIEWIELKIEJ ROZBIEŻNOŚCI WARTOŚCI O PRZECIWNYM ZNAKU .....	88
RYS. 5.19 NIEKORZYSTNA POSTAĆ KRZYŻOWANIA .....	89
RYS. 5.20 WPŁYW PARAMETRÓW SIECI NEURONOWEJ NA WYNIK PROGNOZY.....	89
RYS. 5.21 ZALEŻNOŚĆ WARTOŚCI NAJLEPSZEGO PRZYSTOSOWANIA W POKOLENIU WZGLĘDEM KOLEJNYCH ITERACJI ALGORYTMU GENETYCZNEGO.....	90
RYS. 5.22 PREZENTACJA DANYCH JAKO DWA SYGNAŁY ORAZ RÓŻNICA POMIĘDZY NIMI. ....	91
RYS. 5.23 DIAGRAM PAKIETÓW SYSTEMU AMADEUS .....	93
RYS. 5.24 DIAGRAM KLAS PAKIETU COM.DM.AMADEUS .....	94
RYS. 5.25 DIAGRAM KLAS PAKIETU COM.DM.AMADEUS.IMPORTS .....	95
RYS. 5.26 BUDOWA PLIKU TEKSTOWEGO Z NOTOWANIAMI GIEŁDOWYMI .....	95



RYS. 5.27 DIAGRAM KLAS PAKIETU COM.DM.AMADEUS.ENGINE.....	96
RYS. 5.28 DIAGRAM KLASY <i>GENERATION</i> .....	98
RYS. 5.29 DIAGRAM KLASY <i>CHROMOSOME</i> .....	98
RYS. 5.30 KLASY KOOPERUJĄCE Z KLASĄ <i>GENERATION</i> ZWIĄZANE Z TOPOLOGIĄ SIECI.....	99
RYS. 5.31 KLASY KOOPERUJĄCE Z KLASĄ <i>GENERATION</i> ZWIĄZANE Z UCZENIEM SIECI.....	100
RYS.5.32 IMPLEMENTACJA PROGRAMOWA TURNIEJU.....	100
RYS. 5.33 DIAGRAM KLASY FORECASTING .....	101
RYS.5.34 ILUSTRACJA KLAS IMPLEMENTUJĄCYCH WARSTWY I POŁĄCZENIA FRAMEWORKU JOONE	102
RYS. 5.35 SCHEMAT DZIAŁANIA KLASY DELAYLAYER .....	103
RYS. 5.36 FORMAT ZBIORÓW: TESTUJĄCEGO I PREDYKUJĄCEGO ORAZ WYNIKOWEGO.....	105
RYS. 5.37 FORMAT ZBIORU TRENINGOWEGO .....	105
RYS. 5.38 DIAGRAM ENCJI BAZY DANYCH ZASTOSOWANEJ W PROJEKCIE AMADEUS.....	109
RYS. 5.39 DIAGRAM STRON JSP SYSTEMU AMADEUS.....	110
RYS. 6.1 PRZEBIEG CEN I WOLUMENU SPÓŁEK UŻYTYCH W BADANIACH.....	111
RYS. 6.2 ZBIORY UCZĄCE I TESTUJĄCE DLA SPÓŁEK ELE I CSG.....	112
RYS. 6.3 WYKRES ZBIORÓW UCZĄCYCH I TESTUJĄCYCH DLA SPÓŁEK ELE I CSG .....	113
RYS. 6.4 WYKRES ZALEŻNOŚCI BŁĘDU MAPE OD ILOŚCI NEURONÓW W WARSTWIE UKRYTEJ (ELE)113	
RYS. 6.5 WYKRES ZALEŻNOŚCI BŁĘDU MAPE OD ILOŚCI NEURONÓW W WARSTWIE UKRYTEJ (CSG)114	
RYS. 6.6 ZALEŻNOŚĆ CZASU UCZENIA OD ILOŚCI NEURONÓW W WARSTWIE UKRYTEJ .....	114
DLA TRZECH WYBRANYCH ALGORYTMÓW NAUCZANIA (ELE) .....	114
RYS. 6.7 ZALEŻNOŚĆ ODCHYLENIA STANDARDOWEGO ORAZ KORELACJI WYNIKÓW ZE.....	114
RYS. 6.8 CHARAKTERYSTYCZNE PUNKTY ZMIAN LUB POTWIERDZENIA KONTYNUACJI .....	116
RYS. 6.9 ZAPROPONOWANA KONFIGURACJA DANYCH WEJŚCIOWYCH .....	116
RYS. 6.10 WYNIKI PROGNOZ W ZALEŻNOŚCI OD ZBIORU UCZĄCEGO.....	118
RYS. 6.11 WYCINEK BADANEGO PRZEBIEGU BEZ OPÓŹNIEŃ, Z 5 DNIOWYM OPÓŹNIENIEM ORAZ Z OPÓŹNIENIEM 20 DNIOWYM .....	119
RYS. 6.12 WARTOŚĆ BŁĘDU TRENINGU W ZALEŻNOŚCI OD WARTOŚCI OPÓŹNIENIA PRZEBIEGU UCZĄCEGO .....	120
RYS. 6.13 ZBIÓR UCZĄCY ORAZ WYKRES KURSU AKCJI WIG20 KTÓRY JEST CELEM TWORZONEJ PREDYKCJI. ....	121
RYS. 6.14 WYNIKI PREDYKCJI: 10 I 20 DNIOWEJ DLA WIG20 .....	121
RYS. 6.15 WYNIKI PREDYKCJI: 40 I 60 DNIOWEJ DLA WIG20 .....	121
RYS. 6.16 WYNIKI PREDYKCJI: 80 I 100 DNIOWEJ DLA WIG20 .....	122
RYS. 6.17 ZBIÓR DANYCH UCZĄCYCH DLA SPÓŁKI ALCHEMIA (SESJE 2100-3100) ORAZ DLA SPÓŁKI SKOTAN (SESJE 2130-3130).....	123
RYS. 6.18 PROGNOZA NOTOWAŃ SPÓŁKI ALCHEMIA .....	124
RYS. 6.19 PROGNOZA NOTOWAŃ SPÓŁKI SKOTAN.....	124
RYS. 6.20 WYKRES PRZYROSTU PRZYSTOSOWANIA OSOBNIKÓW W KOLEJNYCH ITERACJACH ORAZ MALENIA MINIMALNEGO BŁĘDU TESTOWANIA MAPE .....	126
RYS. 6.21 REPREZENTACJA WYNIKÓW PROGNOZY ORAZ RZECZYWISTYCH WARTOŚCI NOTOWAŃ SPÓŁKI ELEKTRIM S.A .....	127