Instrukcje do laboratorium

Marcin Korzeń

Zachodniopomorski Uniwersytet Technologiczny, Wydział Informatyki,
Katedra Metod Sztucznej Inteligencji i Matematyki Stosowanej
mkorzen@wi.ps.pl

Spis treści

Rozdział 1

Metody Sztucznej Inteligencji

Rozliczenie zajęć

Rozliczenie zajęć odbywa się na podstawie:

1. Obecności i czynnej pracy na zajęciach
2. Ocen z wejściówek
3. Ocen z zadań praktycznych i sprawozdań

Część materiału i zadań ma charakter dodatkowy co na leży rozumieć: trzeba czytać i robić więcej jeżeli chce się mieć ocenę powyżej 4.0.

1.1 Wprowadzenie do programu Matlab i pakietów: Netlab i Neural Network Toolbox

Wymagania Podstawowa znajomość programowania w Matlabie oraz znajomość podstawowych funkcji do obliczeń numerycznych.

Zadania do wykonania

1. Nauczyć się wczytywać dane różnych formatów do Matlaba, dane można znaleźć tu:
   - `http://mlr.cs.umass.edu/ml/datasets.html`
   - `http://weka.sourceforge.net/wiki/index.php/Datasets`
   - `http://lib.stat.cmu.edu/`
   - `http://www.inf.ed.ac.uk/teaching/courses/dme/html/datasets0405.html`
2. Zapoznać się z pakietem `Neural Network Toolbox` (zaczynając od: `» help nnet`
3. Zapoznać się z pakietem `Netlab` (autorzy: Ian Nabney, Christopher Bishop) dostępny pod adresem: `http://www.ncrg.aston.ac.uk/netlab/book.php`
4. Nauczyć się wykonywać w Matlab'ie znane ze statystyki i metod numerycznych metody jak: wyznaczanie podstawowych statystyk, obliczanie macierzy korelacji, PCA, regresja liniowa, rozwiązywanie zadania optymalizacji, itp.

5. Nauczyc się rysowania wykresów funkcji jednej i dwóch zmiennych, rysowania histogramów, itp.

literatura i uwagi Wszystkie potrzebne informacje można znaleźć na stronie `http://www.mathworks.com/access/helpdesk/help/techdoc/`.

1.2 Perceptron

Wymagania Zbiór liniowo separowalny, margines separacji. Perceptron, struktura sieci, model neuronu, reguła perceptronu. Zbieżność perceptronu, twierdzenie Novikoff'a. Interpretacja geometryczna wag.

Zadania do wykonania

1. Napisać algorytm uczenia perceptronu, funkcja powinna brać na wejście dane uczące, oraz zwracać wagi neuronu/ów i warość/ści prrogowe. Prototyp funkcji może wyglądać następująco: `[w,b]=regulaPerceptronu(X,d)`. Uwaga funkcja powinna w jakiś sposób zabezpieczać się przed danymi nieseparowalnymi np. ustawiając maksymalną liczbę kroków algorytmu na 1e6 lub uzależniając stop od czasu działania.

2. Wygenerować dowolne dane liniowo seprowalne[1] np:
```
X=rand(100,2);
d=2*(X(:,1)-X(:,2)>0)-1;
```

3. Uruchomić algorytm
`[w,b]=regulaPerceptronu(X,d)`

4. Zwizualizować wynik działania algorytmu.

5. Poeksperymentować z różnymi danymi wejściowymi np. zwiększając rozmiar zbioru uczącego zachowując margines separacji, zmieniając rozkład danych.

6. Wyznaczyć maksymalny margines w danych oraz minimalny promień danych, a następnie sprawdzić, że zachodzi teza twierdzenia Novikoff'a.

* Zmodyfikować algorytm tak, aby w sensowny sposób zachowywał się dla danych nieseparowalnych. Poeksperymentować z różnymi zbiorami danych.

literatura i uwagi Większość informacji można znaleźć w pracach: [4, rozdział 1.], [2, Rozdzaił 5], przeczytać ze zrozumieniem [2, §5.3]. Książkę [4] można traktować jako podstawowy podręcznik do sieci neuronowych na naszych zajęciach.

1.3 Sieci neuronowe jednokierunkowe warstwowe

Wymagania Struktura sieci, model neuronu, uczenie metodą wstecznej propagacji błędu w postaci prostej (on-line) i wsadowej (off-line), przegląd zastosowań.

[1] Uwaga: Kolejne dane mogą być zapisane wierszami jak w przykładnie lub kolumnami – ten sposób preferowany jest przez pakiet Neural Network Toolbox. Oba sposoby zapisu są równie często spotykane w literaturze. Należy mieć to na uwadze transponując odpowiednio wzory w sytuacjach, gdy jest to potrzebne.

Zadania do wykonania

1. Napisać algorytm[2] uczenia sieci neuronowej metodą wstecznej propagacji błędu w postaci on-line, dla pojedynczej warstwy neuronów. Prototyp funkcji może mieć postać: `[w,b]=backprop(X,d)`, gdzie (X, d) to pary uczące.

2. Zmodyfikować algorytm do tak, aby pracował w trybie wsadowym.Powinna wystarczyć tu zmiana pojedynczej linii, podmienioną linię wziąść w komentarz.

* Zmodyfikować algorytm do tak, aby pracował dla sieci dwuwarstwowej, proponuje tryb wsadowy oraz w miejsce zwykłej wstecznej propagacji błędu wykorzystać jakiś rodzaj gradientów sprzężonych. Wywołanie może mięć postać: `[w1,b1,w2,b2]=backprop(X, d, K)`, gdzie K określa liczbę neuronów warstwy ukrytej, do obsługi funkcji ze zmienną liczbą argumentów można użyć stałej `nargin`.

3. Napisać funkcję, która dla zadanych wag, oraz próbek wejściowych wyznaczy wartościowe sieci neuronowej. Funkcja może obsługiwać zmienną liczbę argumentów:
`y=mlp(X,w,b)`
`y=mlp(X,w1,b1,w2,b2)`

* Napisać funkcje do wizualizacji, wywołania: `sepline(X,d,w,b)` lub `sepline(X,d,w1,b1,w2,b2)`. Funkcja powinna ona rysować prostą separacji na tle próbek uczących w przypadku sieci jednowarstwowej, oraz linię separacji w przypadku sieci dwuwarstwowej. Zastanowić się nad przypadkiem gdy danych wejściowe są więcej niż dwuwymiarowe, wtedy można posłużyć się metodą PCA. (wskazówka: przypadek danych dwuwymiarowych jest prosty należy posłużyć się funkcjami: `meschgrid` i `contour`.)

4. Poeksperymentować z różnymi zbiorami danych, porównać działanie z algorytmem reguła perceptronu.

literatura i uwagi Większość informacji można znaleźć w pracach: [4, rozdział 2.], parz również [2].

1.4 Sieci neuronowe jednokierunkowe (ang. *feed-forward, MLP*)

Wymagania MLP, struktura sieci model neuronu. Uczenie sieci metodą wstecznej propagacji błędu i jej uogólnienia: dodanie czynnika momentu, minimalizacja kierunkowa, gradienty sprzężone, metody drugiego rzędu. Optymalna prosta separująca dla danych linowo separowalnych i nieseparowalnych, metoda SVM.

Zadania do wykonania

1. Pobrać dane dowolne dane z repozytoriom UCI:
`http://archive.ics.uci.edu/ml/datasets.html`.
Powinny dotyczyć one interesującego zagadnienia o którym mamy pewną wiedzę (np. `wine`).

2. Wczytać dane do Matlab'a

[2]Proszę wziąść pod uwagę, że podstawowym typem Matlab'a są macierze, dla których przeciążono odpowiednie działania i funkcje, czyli należy korzystać z wygodnych notacji wektorowych i nie powinno się nadużywać pętli `for`.

3. Podzielić dane na dwie częći: część uczącą (P, T) i część testującą (PT, TT)

4. Podzielić zbiór uczący (P, T) na część uczącą (PU, TU) i walidującą (PV, TV)

5. Nauczyć sieć neuronową na całym zbiorze uczącym (bez podziału na części uczącą i walidującą) i dokonać oceny błędu na zbiorze testowym

6. Nauczyć sieć neuronową na całym zbiorze uczącym z wykorzystaniem zbioru walidującego i dokonać oceny błędu na zbiorze testowym

* Dokonać wizualizacji danych (w rzutach na wybrane podprzestrzenie lub lepiej używając PCA) oraz wizualizacji modelu.

* Zastosować uczenie sieci metodą SVM

7. Dokonać oceny modelu: oszacowanie błędu klasyfikacji (odsetek błędnych klasyfikacji na zbiorze testowym), krosswalidacja, AUC.

8. Powtórzyć uczenie dla sieci jednowarstwowych i dwuwarstwowych (z przyjętą arbitralnie liczbą neuronów ukrytych), porównać różne rodzaje sieci neuronowych oraz porównać różne algorytmy uczenia, do porównania można wykorzystać również program WEKA, wyniki zebrać w tabeli.

* Dobrać liczbę neuronów warstwy ukrytej z wykorzystaniem zbioru walidującego.

* Spróbować dokonać oceny istotności i współzależności zmiennych. (do oceny współzależności można użyć współczynnika korelacji liniowej miary przyrostu informacji lub innych, istotność można ocenić biorąc pod uwagę jaki wpływ na jakość modelu ma usuniecie jednej lub kilku zmiennych,)

9. Analiza i wnioski dotyczące eksperymentu (należy brać pod uwagę wielkość błędu, złożoność modelu, praktyczną przydatność takiego klasyfikatora w konkretnym przypadku, dokona jakościowej analizy).

1.5 Klasyfikacja i regresja

Wymagania Ocena dokładności maszyny uczącej, testowanie, kroswalidacja, MAE, MSE, dokładność klasyfikacji, błąd klasyfikacji, czułość, specyficzność, krzywa ROC, AUC.

Zadania do wykonania

1. Wczytać do Matlab'a dowolne dane dotyczące problemu klasyfikacji dla dwóch klas.

2. Podzielić dane na trzy części ucząca L, walidującą V i testującą T

3. Nauczyć na danych uczących następujące liniowe maszyny uczące: perceptron, MLP (sieć jednowarstwowa różne algorytmy uczenia dostępne w matlabie), (ewentualnie SVM, regresja logistyczna)), powinniśmy dostać różne zestawy wag prostej separującej w, b, oraz różne błędy klasyfikacji[3].

[3] Należy rozróżniać błąd sieci (w przypadku MLP jest to zwykle MSE lub SSE, w przypadku regresji logistycznej to DEV lub log-likelihood) oraz błąd klasyfikacji

4. Wybrać najlepszy z modeli używając próby walidującej, a następnie ocenić jego dokładność używając próby testowej

* Napisać funkcję która dla zadanych wag i danego zbioru testowego narysuje krzywą ROC.

* Obliczyć pole pod ta krzywą.

literatura i uwagi Większość informacji na ten temat można znaleźć w książce: [3, §2.3.2], zachęcam jednak do szerszego zapoznania się z ta pozycją..

literatura i uwagi O metodzie maszyn wektorowych (SVM) można przeczytać w [1] lub [3, §6.2].

1.6 Sieci neuronowe typu RBF

Wymagania Sieci RBF, struktura sieci, model neuronu, uczenie sieci (samoorganizacja + regresja), algorytm K-środków, algorytm EM, mieszanka modeli gaussowskich (GMM), sieci probabilistyczne, optymalny klasyfikator Bayesowski

Zadania do wykonania

1. Pobrać dane `autoPrices.txt` ze strony `ksir.wi.ps.pl` lub inne dowolne dane z repozytorioum UCI:
   `http://archive.ics.uci.edu/ml/datasets.html` dotyczące zadania regresji.

2. Wczytać dane do Matlab'a

3. Dokonać normalizacji zmiennych: zmienne przyjmujące wartości różnych znaków normalizujemy do przedziału $[-1, 1]$ z zachowaniem zera, zmienne przyjmujące wartości dodatnie normalizujemy do zakresu $[0, 1]$.

4. Podzielić dane na dwie częći: część uczącą (`PU`, `TU`) i część testującą (`PT`, `TT`)

5. Nauczyć sieć neuronową typu RBF na całym zbiorze uczącym (bez podziału zbioru uczącego na części uczącą i walidującą) i dokonać oceny błędu na zbiorze testowym.

6. Dokonać oszacowania błędu sieci RBF (`mse` lub `mae`)

7. Powtórzyć uczenie dla sieci z różną liczbą liczbą neuronów RBF.

8. Analiza istotności poszczególnych atrybutów (zmiennych) warunkowych.

9. Stworzyć dla porównania standardowy model regresji liniowej, porównać błędy (`mse` lub `mae`), spojrzeć na współczynniki i stad wnioskować nt. istotności zmiennych.

10. Analiza i wnioski dotyczące eksperymentów (należy brać pod uwagę wielkość błędu, praktyczną przydatność prognozy (np. szacowanie ceny samochodu dla potrzeb komisu samochodowego), dyskusję czym spowodowany jest błąd, dyskusję nt. brakujących czynników mających wpływ na błąd prognozy, itp.).

Zadania dodatkowe:

1. Dokonać wizualizacji danych (w rzutach na wybrane podprzestrzenie lub uzywając PCA)
2. Spróbować dokonać oceny istotności i współzależności zmiennych. (do oceny współzależności można użyć współczynnika korelacji liniowej, istotność można ocenić biorąc pod uwagę jaki wpływ na jakość modelu ma usuniecie jednej lub kilku zmiennych)
3. Dobrać liczbę neuronów warstwy ukrytej lub szerokość neuronów RBF z wykorzystaniem zbioru walidującego.

literatura i uwagi Większość informacji można znaleźć w pracach: [4, rozdział 2.], o metodzie maszynach wektorowych (SVM) można przeczytać w [1] lub [3, §6.2].

1.7 Sieci rekurencyjne

Wymagania Sieci Hopfielda, struktura sieci model neuronu, tryb pracy, tryb odtwarzania, reguła Hebba, pamięci skojarzeniowa, funkcja Lapunowa

Rozdział 2

Aplikacje Sztucznej Inteligencji

2.1 Klasyfikacja i regresja

Zadania do wykonania

1. Pobrać dwa dowolne, nietrywialne zbiory danych z repozytorium UCI: `http://archive.ics.uci.edu/ml/datasets.html`, lub innego. (przykładowo może to być zbiór laeukemia, zbiory z KDD CUP 2004, zbiór streszczeń artykułów z bazy Reuters-21578, lub inny)
2. Wczytać dane do Matlab'a, oraz/lub do Weka (oraz/lub innego programu)
3. Podzielić dane na dwie części: część uczącą (PL, TL) i część testującą (PT, TT)
4. Przedstawić wizualnie dane wielowymiarowe z zaznaczonymi klasami decyzyjnymi, najlepiej posłużyć się metodą PCA, gdy nie jest to możliwe dokonać redukcji atrybutów mniej istotnych
5. Na części uczącej dokonać wybory najlepszego klasyfikatora (spośród następujących metod: sieci neuronowe o różnej strukturze i różnych alg. uczenia, metody SVM, regresja logistyczna, naiwny klasyfikator Bayes'a, inne), klasyfikatory należy oceniać stosując metodę kroswalidacji na zbiorze uczącym
6. Dokonać selekcji atrybutów wybraną metodą (np. spróbować dla zbioru leukaemia wybrać jedynie kilka atrybutów które wystarczają do dość dokładnej klasyfikacji obiektów)
7. Porównać wyniki kroswalidacji z wynikami na zbiorze testowym.
8. Oceny jakości klasyfikacji dokonywać na podstawie:
   - błędów/dokładności klasyfikacji
   - krzywej ROC (krzywe te przedstawić graficznie dla różnych klasyfikatorów)
   - AUC (pole pod krzywą ROC)
9. Podsumowanie i wnioski dotyczące między innymi:

- krytycznej oceny jakości klasyfikacji w danym konkretnym problemie
- stosowalności wybranych metod uczenia maszynowego w danym konkretnym problemie pod względem możliwości numerycznych oraz jakości działania
- oceny wybranych metod (które lepsze, które gorsze)
- do prezentacji wyników używać tabel i wykresów, zachęcam do używania generatora raportów Matlab'a

2.2 Optymalizacja dyskretna

Zadania do wykonania Do wyboru mamy dwa tematy:

- Wyszukiwanie optymalnej ścieżki komiwojażera
- Problem kolorowania grafów (kolorowanie wierzchołków)

Zadania należy rozwiązać stosując algorytm ewolucyjny, jednak proponuję zacząć od napisania, któregoś ze znanych algorytmów heurystycznych

Rozdział 3

Podstawy Sztucznej Inteligencji - ZIP

3.1 Porównani algorytmów uczących sieci neuronowych z pakietu Neural Network Toolbox

Zadania do wykonania

1. Wczytać do Matlab'a trzy różne zbiory danych dotyczące problemu klasyfikacji dla dwóch klas (np. wine, breast-cancer, waveform-5000).
2. Podzielić dane na dwie równe części ucząca L, testującą T
3. Porównać czasy działania i dokładność klasyfikacji algorytmów uczących przy ustalonej strukturze sieci. Należy wziąść pod uwagę algorytmy: `traingd, traingdm, trainrp, trainscg, traincgf, trainlm`, oraz struktury: 1- warstwową, 2-warstwową (5:5:100 neuronów w warstwie ukrytej), kilka przykładów sieci 3-warstwowych. Wyniki przedstawić w tabeli (przykładowo tab 3.1) oraz na wykresach. Przedstawić wyniki pogrupowane względem algorytmów uczenia, należy patrzeć na czas działania i liczbę parametrów oraz błąd na zbiorze testowym (zdolność do uogólniania)
4. Podsumowanie i wnioski
5. Jako załącznik wkleić wykorzystywane skrypty Matlab'a

3.2 Sieci neuronowe typu RBF

Wymagania Sieci RBF, struktura sieci, model neuronu, uczenie sieci (samoorganizacja + regresja), algorytm K-środków

Zadania do wykonania

1. Pobrać dane `autoPrices.txt` ze strony `ksir.wi.ps.pl` lub inne dowolne dane z repozytorioum UCI:
   `http://archive.ics.uci.edu/ml/datasets.html` dotyczące zadania regresji.
2. Wczytać dane do Matlab'a

3. Dokonać normalizacji zmiennych: zmienne przyjmujące wartości różnych znaków normalizujemy do przedziału $[-1, 1]$ z zachowaniem zera, zmienne przyjmujące wartości dodatnie normalizujemy do zakresu $[0, 1]$.

4. Podzielić dane na dwie częći: część uczącą (`PU`, `TU`) i część testującą (`PT`, `TT`)

5. Nauczyć sieć neuronową typu RBF na całym zbiorze uczącym (bez podziału zbioru uczącego na części uczącą i walidującą) i dokonać oceny błędu na zbiorze testowym. (przydatne polecenia `newrb`, `newrbe`, `newgrnn`)

6. Wyznaczyć błąd sieci RBF na zbiorze uczącym i testowym (`mse` lub `mae`)

7. Powtórzyć uczenie dla sieci z różną liczbą neuronów RBF.

8. Stworzyć dla porównania standardowy model regresji liniowej, porównać błędy (`mse` lub `mae`), spojrzeć na współczynniki i stad wnioskować nt. istotności zmiennych.

9. Spróbować dokonać oceny istotności i współzależności zmiennych. (do oceny współzależności można użyć współczynnika korelacji liniowej, istotność można ocenić biorąc pod uwagę jaki wpływ na jakość modelu ma usuniecie jednej lub kilku zmiennych)

10. Analiza i wnioski dotyczące eksperymentów (należy brać pod uwagę wielkość błędu, praktyczną przydatność prognozy (np. szacowanie ceny samochodu dla potrzeb komisu samochodowego), dyskusję czym spowodowany jest błąd, dyskusję nt. brakujących czynników mających wpływ na błąd prognozy, itp.). Wyniki przedstawić w tabeli i na wykresach.

numer eksperymentu	...
rozmiar zbioru uczącego	...
struktura sieci	...
rodzaj algorytmu uczenia	...
czas działania	...
błąd na zb. uczącym	...
błąd na zb. testowym	...

Tablica 3.1: Wyniki należy prezentować w tabeli zbliżonej do tej oraz na wykresach.

Rozdział 4

Sztuczne Sieci Neuronowe - INF

4.1 Porównani algorytmów uczących sieci neuronowych z pakietu Neural Network Toolbox

Zadania do wykonania Celem ćwiczenia jest przebadanie algorytmów uczenia sieci neuronowych. Przy ocenie algorytmów należy wziąć pod uwagę dwie rzeczy: a) czas działania algorytmu, b) zdolność do uogólniania mierzona jako błąd klasyfikacji na zbiorze testowym. Szczegółowy plan zadań:

1. Wczytać do Matlab'a trzy różne zbiory danych dotyczące problemu klasyfikacji dla dwóch klas.
2. Podzielić dane na dwie równe części ucząca L, testującą T
3. Porównać czasy działania i dokładność klasyfikacji algorytmów uczących przy ustalonej strukturze sieci. Należy wziąść pod uwagę algorytmy: `traingd, traingdm, trainrp, trainscg, traincgf, trainlm`, oraz struktury: 1- warstwową, 2-warstwową (5:5:100 neuronów w warstwie ukrytej), kilka przykładów sieci 3-warstwowych. Wyniki przedstawić w tabeli (przykładowo tab 3.1) oraz na wykresach. Przedstawić wyniki pogrupowane względem algorytmów uczenia, należy patrzeć na: czas działania i liczbę parametrów sieci (stukturę) oraz błąd na zbiorze testowym (zdolność do uogólniania)
4. Sprawdzić jak skalują się powyższe algorytmy gdy zmienia się rozmiar zbioru uczącego
5. Podsumowanie i wnioski
6. Zachęcam do publikowania wyników (do HTML'a lub LaTeX'a) bezpośrednio z poziomu Matlaba, w przeciwnym razie jako załącznik wkleić wykorzystywane skrypty Matlab'a. Sprawozdanie nie powinno przekraczać 4 stron, Zasadniczą częścią są: tabela, wykresy, oraz podsumowanie.

4.2 Kompresja obrazów z wykorzystaniem sieci neuronowych

Zadania do wykonania

1. Przygotować obraz testowy.
2. Ustalić strukturę sieci neuronowej liczbę we/wy oraz liczbę parametrów warstwy ukrytej, rodzaje funkcji aktywacji zaczynając od `purelin`.
3. Przygotować wzorce uczące dzieląc obraz na wzorce zadanej wymiarowości.
4. Nauczyć sieć neuronową odtwarzać wzorce uczące, wyznaczyć błąd sieci oraz współczynnik kompresji.
5. Poeksperymentować z różną liczbą wejść/wyjść oraz z różną liczbą neutronów z warstwy ukrytej, parząc za każdym razem na błąd sieci oraz współczynnik kompresji.
6. Prezentacja wyników (rysunki oryginał i po kompresji obok siebie oraz tabela z wynikami), podsumowanie i wnioski
7. Zachęcam do publikowania wyników (do HTML'a lub LaTeX'a) bezpośrednio z poziomu Matlaba, w przeciwnym razie jako załącznik wkleić wykorzystywane skrypty Matlab'a. Sprawozdanie nie powinno przekraczać 4 stron.

4.3 Sieci neuronowe typu RBF

Wymagania Sieci RBF, struktura sieci, model neuronu, uczenie sieci (samoorganizacja + regresja), algorytm K-środków

Zadania do wykonania

1. Pobrać dane `autoPrices.txt` ze strony `wikizmsi.zut.edu.pl` lub inne dowolne dane z repozytorioum UCI: `http://archive.ics.uci.edu/ml/datasets.html` dotyczące zadania regresji.
2. Wczytać dane do Matlab'a
3. Dokonać normalizacji zmiennych: zmienne przyjmujące wartości różnych znaków normalizujemy do przedziału $[-1, 1]$ z zachowaniem zera, zmienne przyjmujące wartości dodatnie normalizujemy do zakresu $[0, 1]$.
4. Podzielić dane na dwie częći: część uczącą (`PU`, `TU`) i część testującą (`PT`, `TT`)
5. Nauczyć sieć neuronową typu RBF na całym zbiorze uczącym (bez podziału zbioru uczącego na części uczącą i walidującą) i dokonać oceny błędu na zbiorze testowym. (przydatne polecenia `newrb`, `newrbe`, `newgrnn`)
6. Wyznaczyć błąd sieci RBF na zbiorze uczącym i testowym (`mse` lub `mae`)
7. Powtórzyć uczenie dla sieci z różną liczbą neuronów RBF. Wybrać najlepszą liczbę neuronów posługując się zbiorem uczącym (można wydzielić część walidującą)

8. Stworzyć dla porównania standardowy model regresji liniowej, porównać błędy obu modeli (`mse` oraz `mae`). Spojrzeć na współczynniki i stad wnioskować nt. istotności zmiennych.

9. Spróbować dokonać oceny istotności i współzależności zmiennych. (do oceny współzależności można użyć współczynnika korelacji liniowej, istotność można ocenić biorąc pod uwagę jaki wpływ na jakość modelu ma usuniecie jednej lub kilku zmiennych)

10. Analiza i wnioski dotyczące eksperymentów (należy brać pod uwagę wielkość błędu, praktyczną przydatność prognozy (np. szacowanie ceny samochodu dla potrzeb komisu samochodowego), dyskusję czym spowodowany jest błąd, dyskusję nt. brakujących czynników mających wpływ na błąd prognozy, itp.). Wyniki przedstawić w tabeli i na wykresach.

11. Zachęcam do publikowania wyników (do HTML'a lub LaTeX'a) bezpośrednio z poziomu Matlaba, w przeciwnym razie jako załącznik wkleić wykorzystywane skrypty Matlab'a. Sprawozdanie nie powinno przekraczać 4 stron.

Bibliografia

[1] Ch J. C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition, in *Data Mining and Knowledge Discovery*, **2**, 121–167 (1998), Kluwer Academic Publishers, Boston.

[2] J. Hertz, A. Krogh, R. G. Palmer *Wstęp do teorii obliczeń neuronowych*. WNT, Warszawa, 1993.

[3] J. Koronacki, J. Ćwik, *Statystyczne systemy uczące*. WNT, Warszawa 2005.

[4] S. Osowski. *Sieci neuronowe w ujeciu algorytmicznym*. WNT, Warszawa, 1998.

[5] A. Piegat, *Modelowanie i sterowanie rozmyte*. Akademicka oficyna wydawnicza EXIT, Warszawa 2000.

[6] I. H. Witten, E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2005