

Informatyka stosowana, studia dzienne, II st.

semestr II

Analiza danych złożonych

2020/2021

Prowadzący: dr hab. inż. Agnieszka Duraj

poniedziałek, 11:45

Etap 2

Autorzy:

Paweł Galewicz 234053

Karol Podlewski 234106

Spis treści

1. Cel	3
2. Implementacja	3
3. Opis klasyfikatorów	3
3.1. Naiwny klasyfikatora Bayesa	3
3.2. Maszyna wektorów nośnych	4
3.3. Klasyfikator k-najbliższych sąsiadów	4
3.4. Algorytm sztucznych sieci neuronowych	4
4. Badania	5
4.1. Naiwny klasyfikatora Bayesa	6
4.2. Maszyna wektorów nośnych	8
4.3. Klasyfikator k-najbliższych sąsiadów	16
4.4. Algorytm sztucznych sieci neuronowych	20
5. Wnioski	28

1. Cel

Zadanie polegało na analizie zachowania klasyfikatorów w przypadku wystąpienia dryftu w strumieniu danych dla różnych parametrów początkowych.

2. Implementacja

Program został stworzony w języku Python w wersji 3.8.6, przy wsparciu bibliotek scikit-multiflow oraz scikit w celu skorzystania z algorytmów przeznaczonych do detekcji dryftu oraz klasyfikacji.

Wybranym przez nas klasyfikatorem były algorytmy:

1. Naiwny klasyfikatora Bayesa
2. Maszyna wektorów nośnych
3. K-najbliższych sąsiadów
4. Sztuczne sieci neuronowe

Do detekcji dryftów wykorzystaliśmy algorytmy

1. DDM
2. EDDM
3. ADWIN
4. Page-Hinkley

Wykorzystano też bibliotekę argparse - w stworzonym rozwiązaniu w łatwy sposób można określić większość parametrów algorytmów, takich jak liczbę sąsiadów w KNN, rodzaj funkcji jądra w SVM czy maksymalną liczbę iteracji w sieciach neuronowych, a także podział zbioru treningowego oraz sam plik z danymi.

3. Opis klasyfikatorów

3.1. Naiwny klasyfikatora Bayesa

Naiwny klasyfikatora Bayesa dokonuje klasyfikacji na bazie twierdzenia Bayesa:

$$P(A | B) = \frac{P(B | A) P(A)}{P(B)}$$

gdzie:

- A, B – zdarzenia
- $P(A | B)$ – prawdopodobieństwo zdarzenia A , o ile zajdzie B
- $P(B | A)$ – prawdopodobieństwo zdarzenia B , o ile zajdzie A
- $P(A)$ – prawdopodobieństwo wystąpienia zdarzenia A

- $P(B)$ – suma prawdopodobieństw wszystkich potencjalnych skutków zdania: $P(B) = \sum P(B | A)P(A)$

Model naiwnego klasyfikatora Bayesa zakłada, że dana cecha klasy jest niepowiązana z pozostałymi cechami. Każda z cech indywidualnie wskazuje na prawdopodobieństwo przynależności do danej klasy. Sprawdza się najlepiej przy dużych zbiorach danych. Jest wykorzystywany m.in. przy filtrowaniu spamu, diagnozie medycznej, czy prognozowaniu pogody.

3.2. Maszyna wektorów nośnych

Maszyna wektorów nośnych jest klasyfikatorem liniowym. Algorytm polega na rozdzieleniu obiektów o różnej przynależności klasowej za pomocą hiperpłaszczyzn, które mają być od siebie możliwe jak najbardziej oddalone - taką odległość nazywa się marginesem klasyfikatora, a hiperpłaszczyzny z największym marginesem wektorami nośnymi.

Algorytm bardzo dobrze sobie radzi z danymi liniowo separowanymi, ale nie zawsze będzie istniała hiperpłaszczyzna rozdzielająca, która zapewni poprawną klasyfikację wszystkich elementów zbioru. W takich przypadkach maszyna wektorów nośnych za pomocą funkcji jądrowych transformuje przestrzeń do postaci liniowo separowanej.

3.3. Klasyfikator k-najbliższych sąsiadów

Algorytm k najbliższych sąsiadów jest klasyfikatorem (ściślej algorytmem regresji regresji nieparametrycznej). Algorytm ten zakłada dany zbiór uczący, w którym znajdują się już sklasyfikowane dane. Schemat składa się z szukania k obiektów najbliższych do obiektu klasyfikowanego. Następnie, przyporządkowuje się nowy obiekt do najczęściej występującej klasy w obrębie jego k-najbliższych sąsiadów.

3.4. Algorytm sztucznych sieci neuronowych

Sztuczna sieć neuronowa jest połączeniem wielu elementów nazywanych sztucznymi neuronami, które tworzą conajmniej trzy warstwy: wejściową, ukrytą oraz wyjściową. Neurony przetwarzają informacje dzięki nadaniu im parametrów które nazywane są wagami. Podstawą tworzenia sieci neuronowej jest modyfikowanie współczynnika wagowego połączeń w celu uzyskania poprawnych wyników.

4. Badania

Celem przeprowadzonych przez nas badań było sprawdzenie różnych klasyfikatorów pod względem odporności na zmiany dryftu na zadanym strumieniu danych. W tym celu stworzyliśmy modele klasyfikatorów z różnymi parametrami startowymi. Zbiór treningowy stanowił 20% wszystkich danych. Dla wyuczonych modeli generowaliśmy macierz pomyłek i na jej podstawie wyliczaliśmy 4 miary ewaluacyjne:

- Dokładność (ang. *Accuracy*) – określa jak duży odsetek obserwacji został prawidłowo zaklasyfikowany.
- Precyzja (ang. *Precision*) – oznacza jak dużo zaklasyfikowanych do danej klasy obserwacji rzeczywiście do niej należy.
- Czułość (ang. *Sensitivity*) – stosunek liczby obserwacji oznaczonych jako *true positive* do sumy *true positive* i *false negative*. Jej wartość interpretować można jako zdolność do prawidłowego zakwalifikowania obserwacji do odpowiedniej klasy.
- Specyficzność (ang. *Specificity*) – stosunek liczby obserwacji *true negative* do sumy *true negative* i *false positive*. Określa jak dobrze model rozpoznaje, że dana obserwacja nie należy do danej klasy.

Każdy klasyfikator badany był pod względem wpływu parametrów startowych na wystąpienia dryftu. Zjawisko to sprawdzaliśmy wykorzystując kolejne algorytmy detekcji. Wartości miar dla kolejnych próbek, wraz z momentami pojawienia się dryftu zostały zaprezentowane na wykresach.

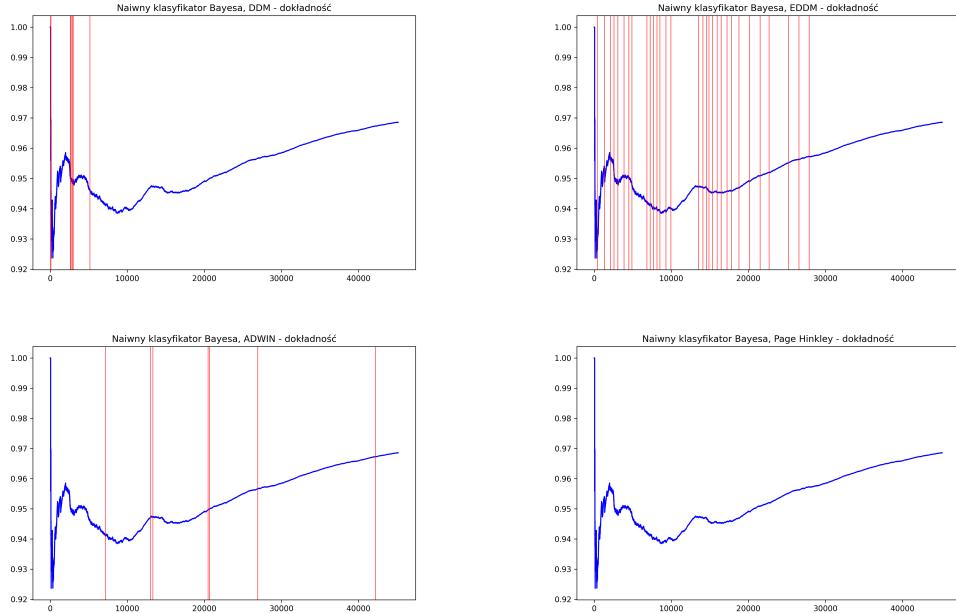
Wybrany przez nas zbiór danych to **Rain in Australia**, który zawiera historię danych pogodowych z 10 lat (data, lokalizacja, temperatury, opady, wiatr, ciśnienie, wilgotność, nasłonecznienie itp) wraz z informacją czy następnego dnia padało - jest to cel klasyfikacji dla tego zbioru danych.

W badaniach uwzględniliśmy następujące konfiguracje klasyfikatorów:

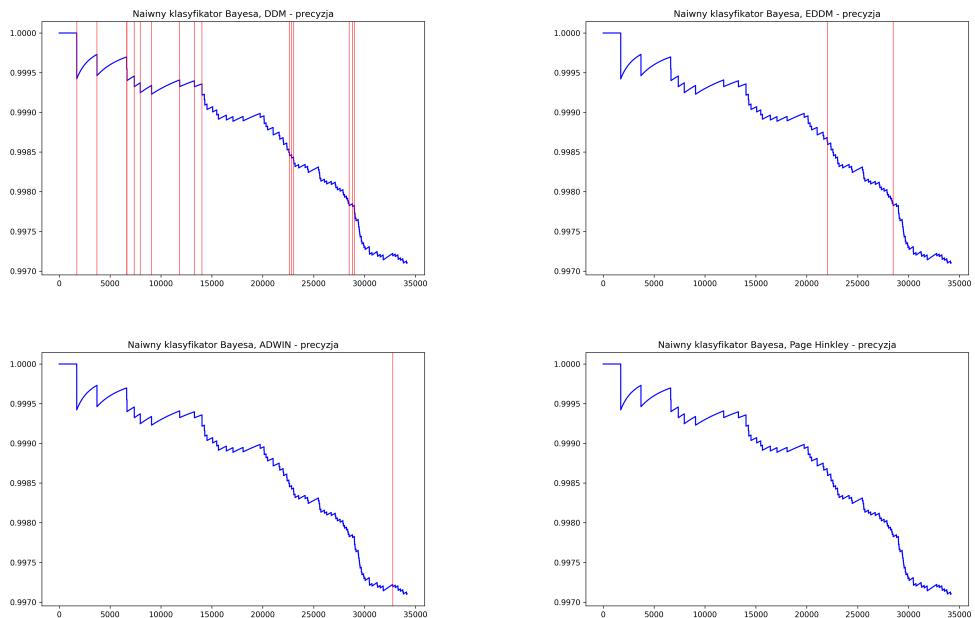
- Naiwny klasyfikator Bayesa – brak parametrów do zmiany.
- Maszyna wektorów nośnych – jądro wielomianowe lub radialne, regularizacja 0,1 lub 1.
- Klasyfikator k-najbliższych sąsiadów – 2 lub 5 sąsiadów.
- Algorytm sztucznych sieci neuronowych – 1 lub 10 ukrytych warstw, 500 lub 1000 iteracji.

Z badań otrzymaliśmy następujące wyniki:

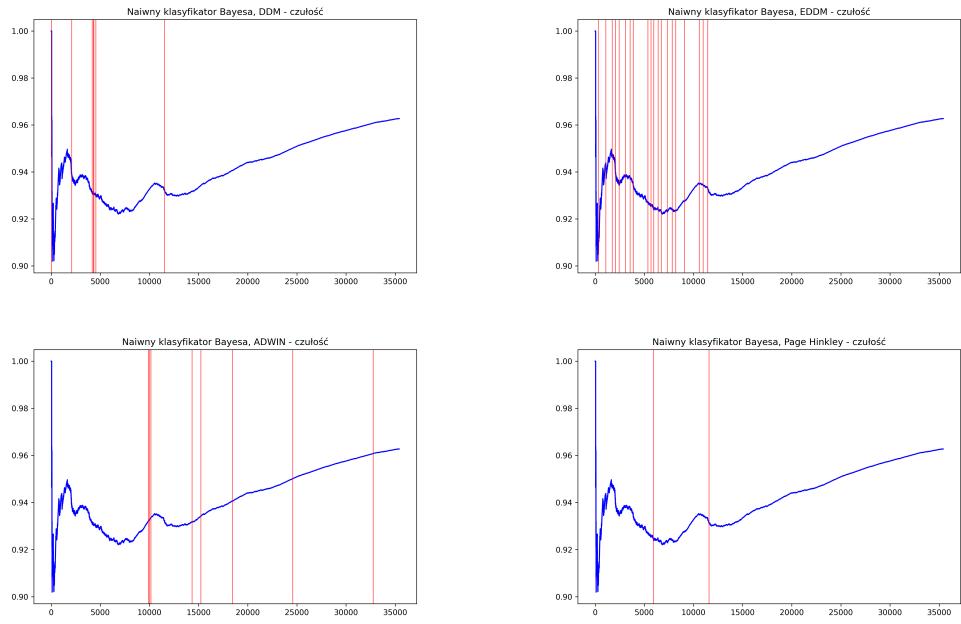
4.1. Naiwny klasyfikatora Bayesa



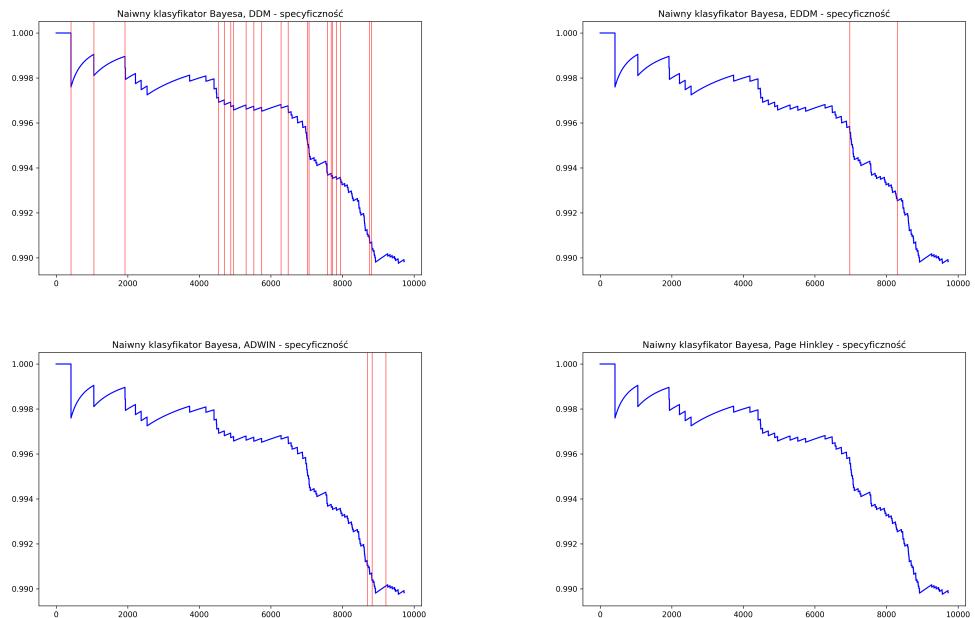
Rysunek 1: Dokładność dla naiwnego klasyfikatora Bayesa



Rysunek 2: Precyzja dla naiwnego klasyfikatora Bayesa

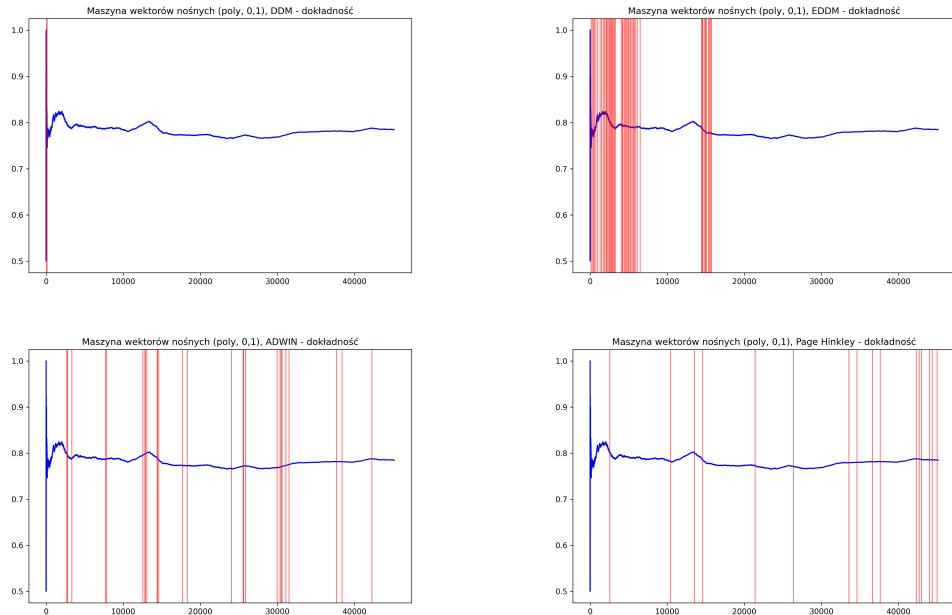


Rysunek 3: Czułość dla naiwnego klasyfikatora Bayesa

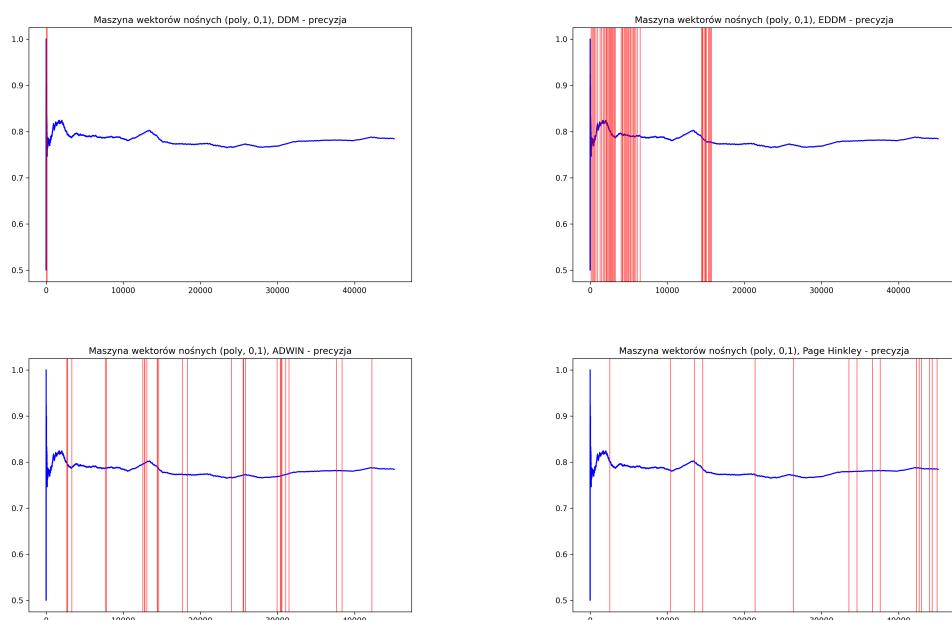


Rysunek 4: Specyficzność dla naiwnego klasyfikatora Bayesa

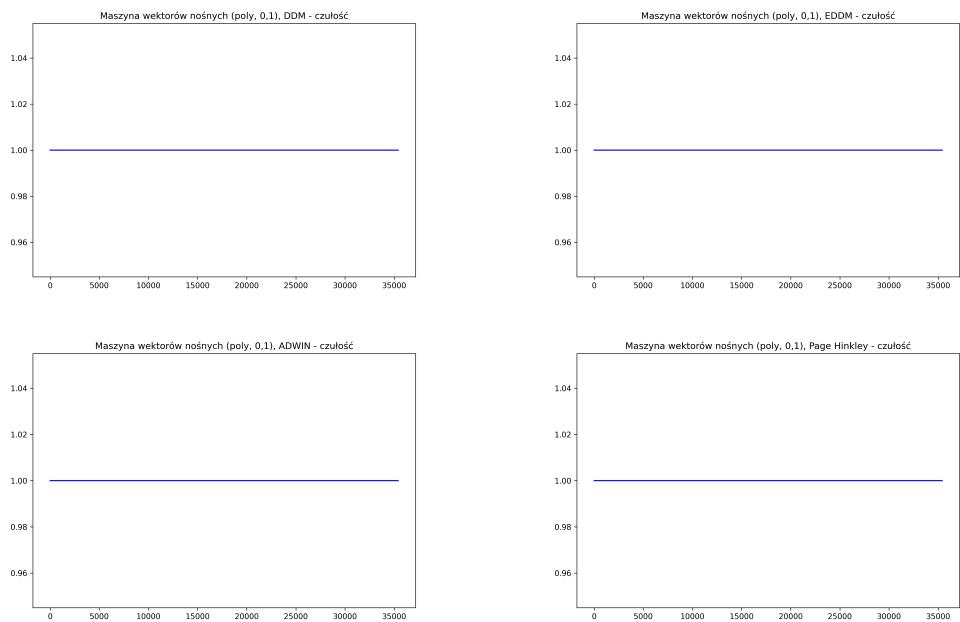
4.2. Maszyna wektorów nośnych



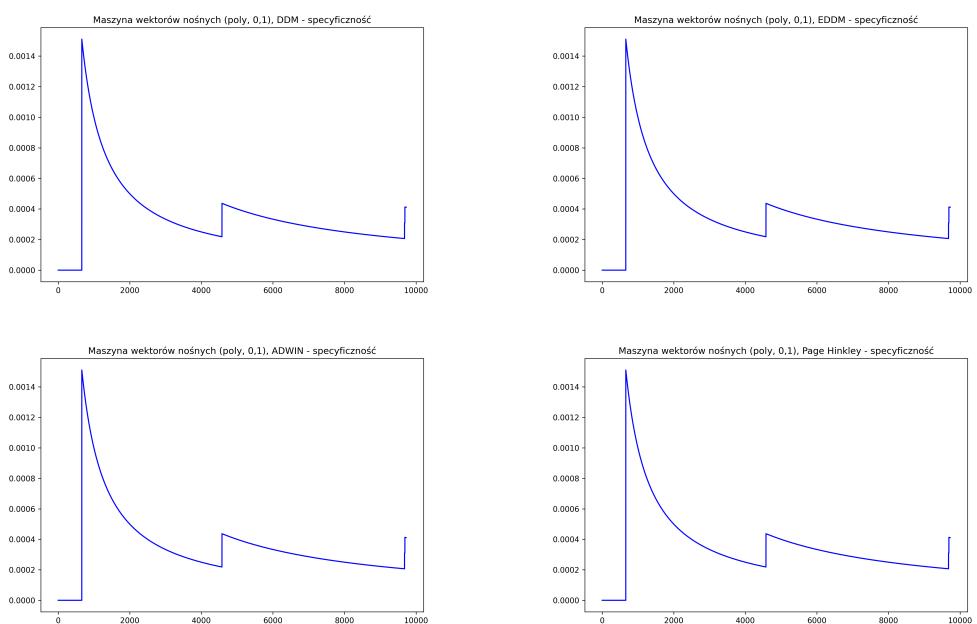
Rysunek 5: Dokładność dla maszyny wektorów nośnych - metoda wielomianowa, regularyzacja 0.1



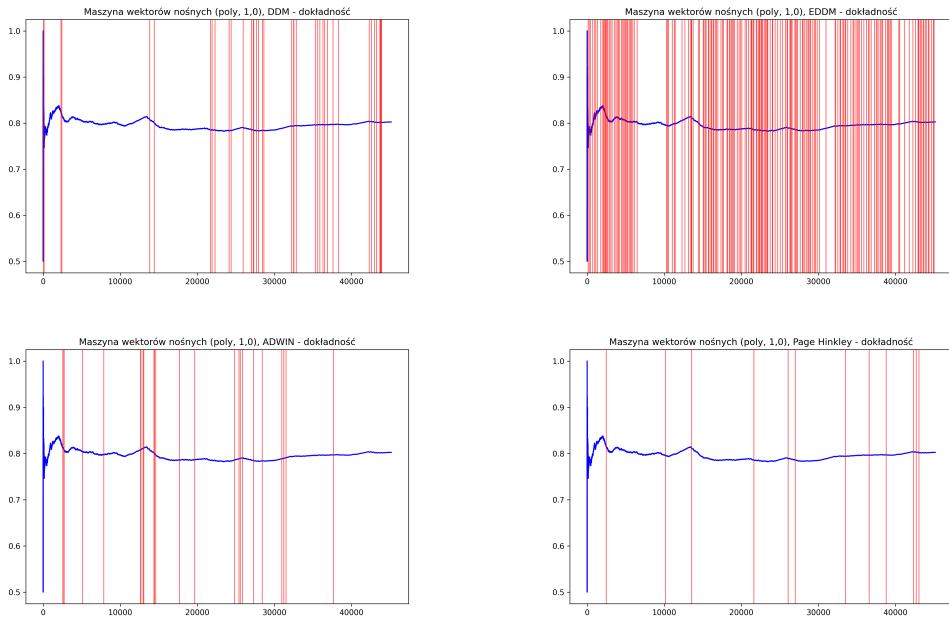
Rysunek 6: Precyza dla maszyny wektorów nośnych - metoda wielomianowa, regularyzacja 0.1



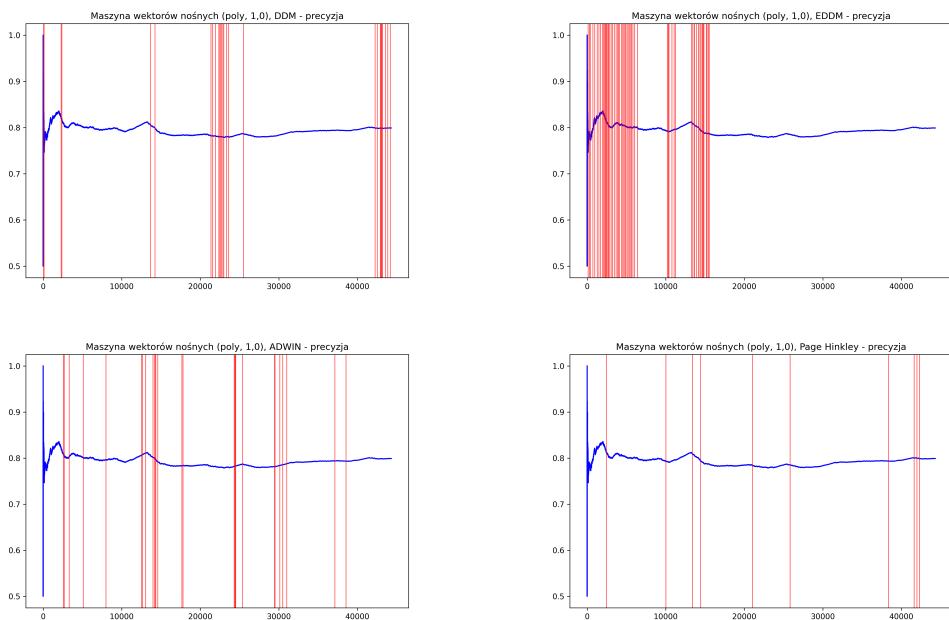
Rysunek 7: Czułość dla maszyny wektorów nośnych - metoda wielomianowa, regularyzacja 0.1



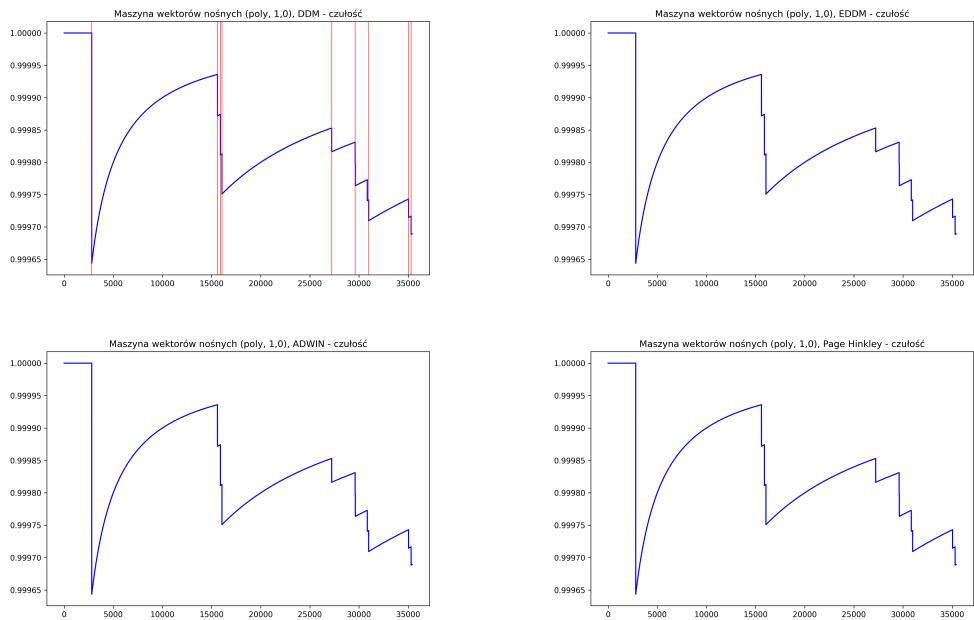
Rysunek 8: Specyficzność dla maszyny wektorów nośnych - metoda wielomianowa, regularyzacja 0.1



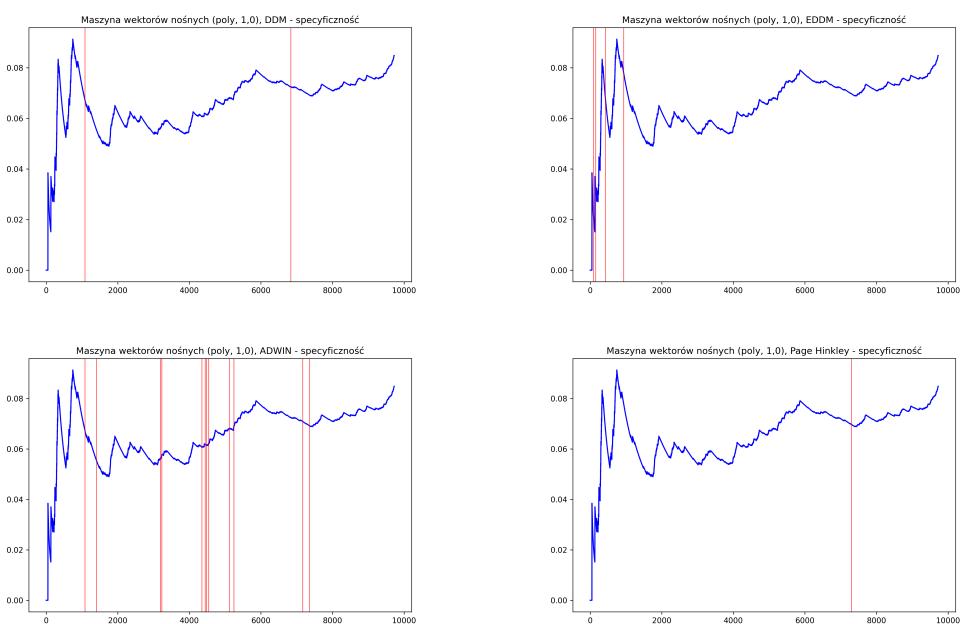
Rysunek 9: Dokładność dla maszyny wektorów nośnych - metoda wielomianowa, regularyzacja 1



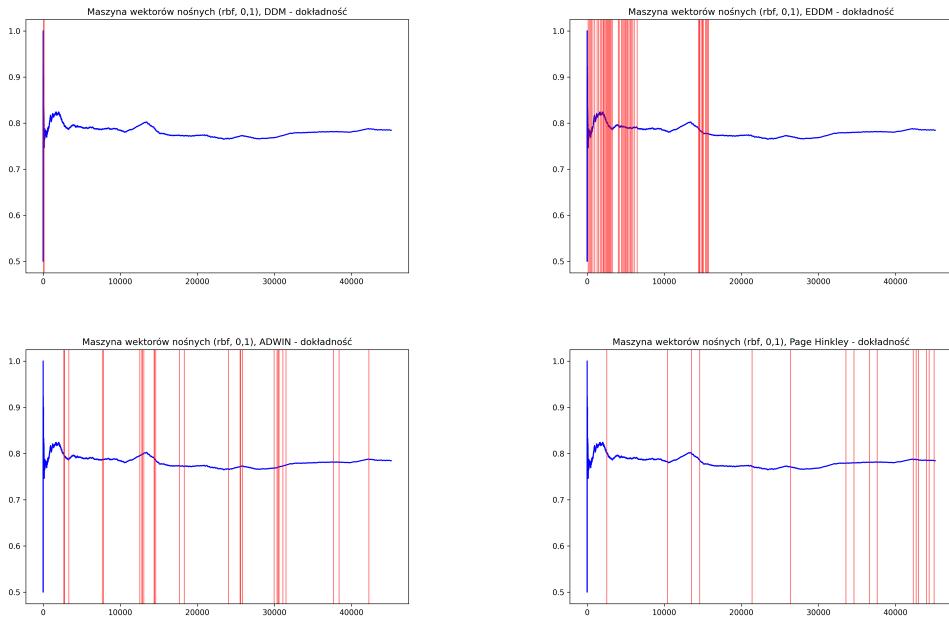
Rysunek 10: Precyzaja dla maszyny wektorów nośnych - metoda wielomianowa, regularyzacja 1



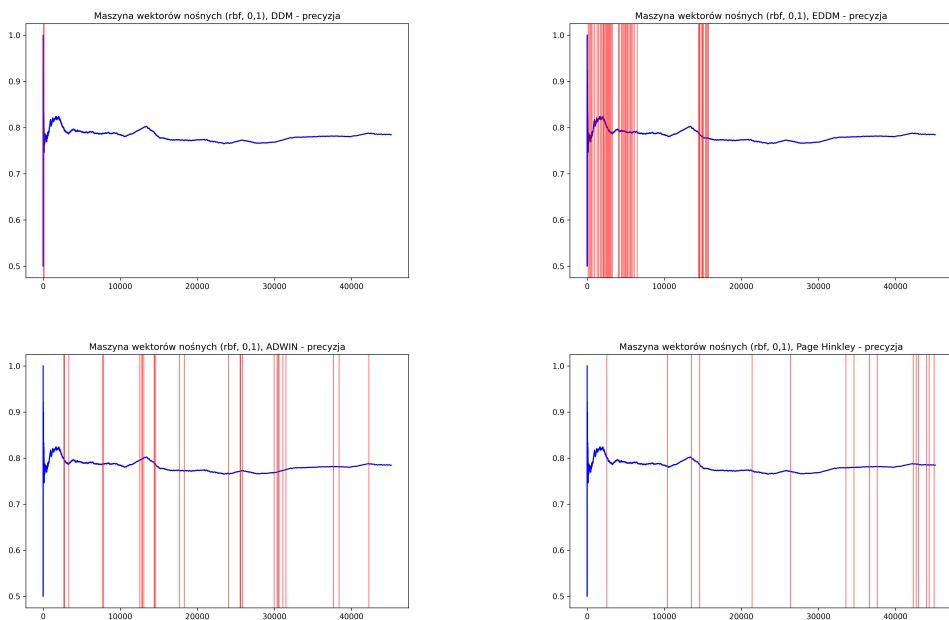
Rysunek 11: Czułość dla maszyny wektorów nośnych - metoda wielomianowa, regularyzacja 1



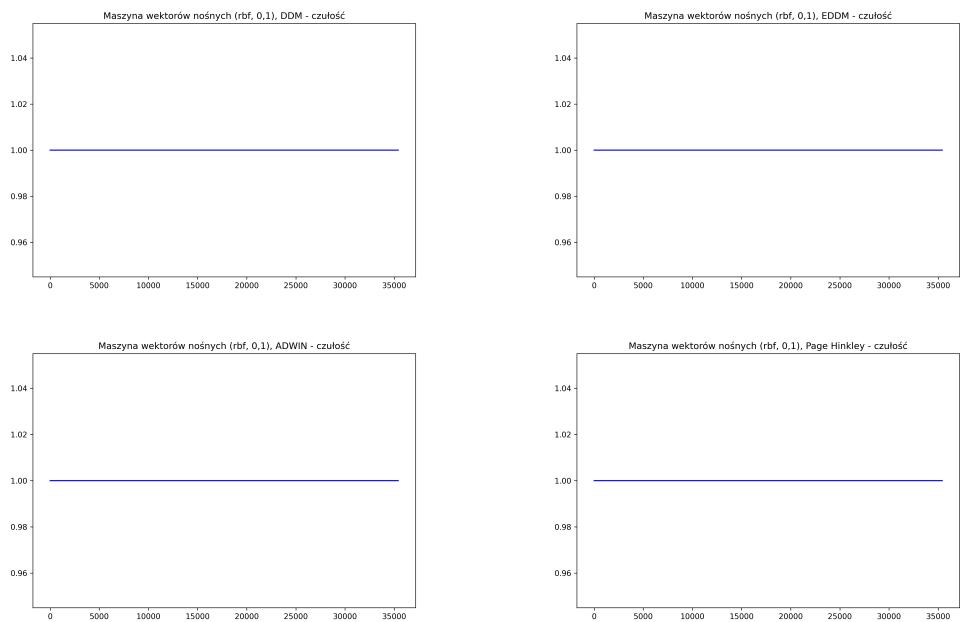
Rysunek 12: Specyficzność dla maszyny wektorów nośnych - metoda wielomianowa, regularyzacja 1



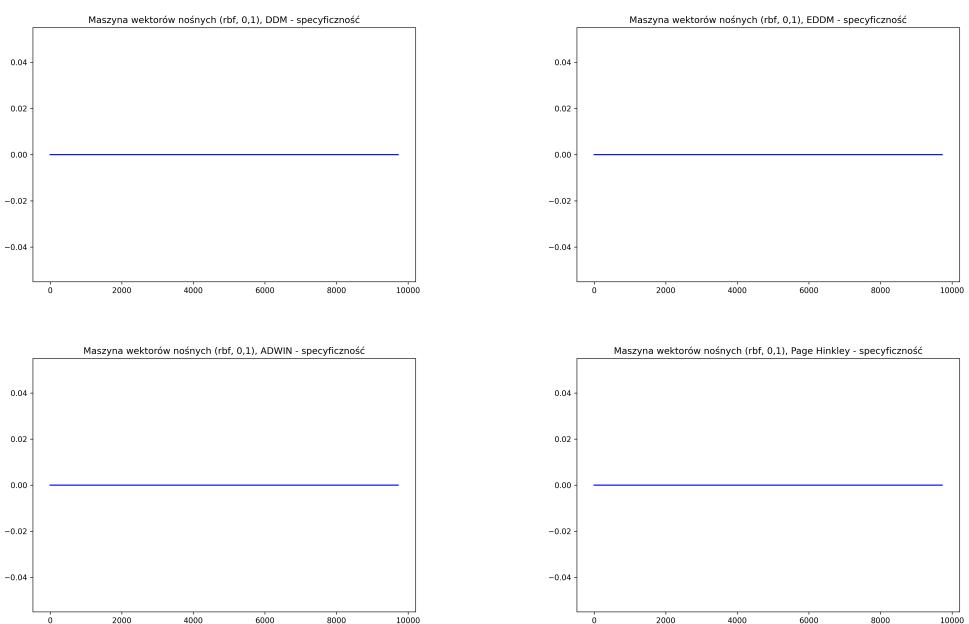
Rysunek 13: Dokładność dla maszyny wektorów nośnych - metoda radialna, regularyzacja 0.1



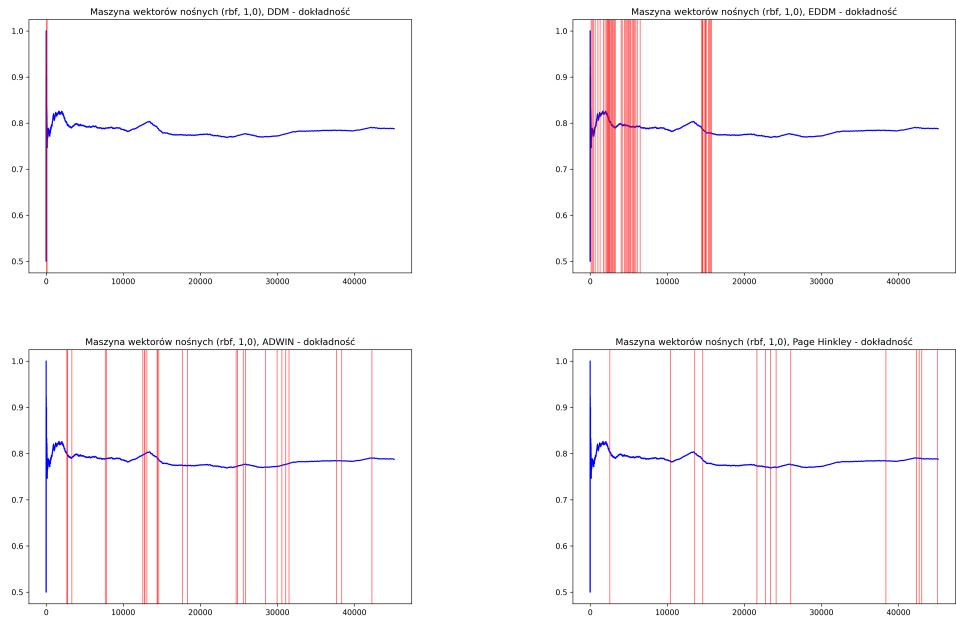
Rysunek 14: Precyza dla maszyny wektorów nośnych - metoda radialna, regularyzacja 0.1



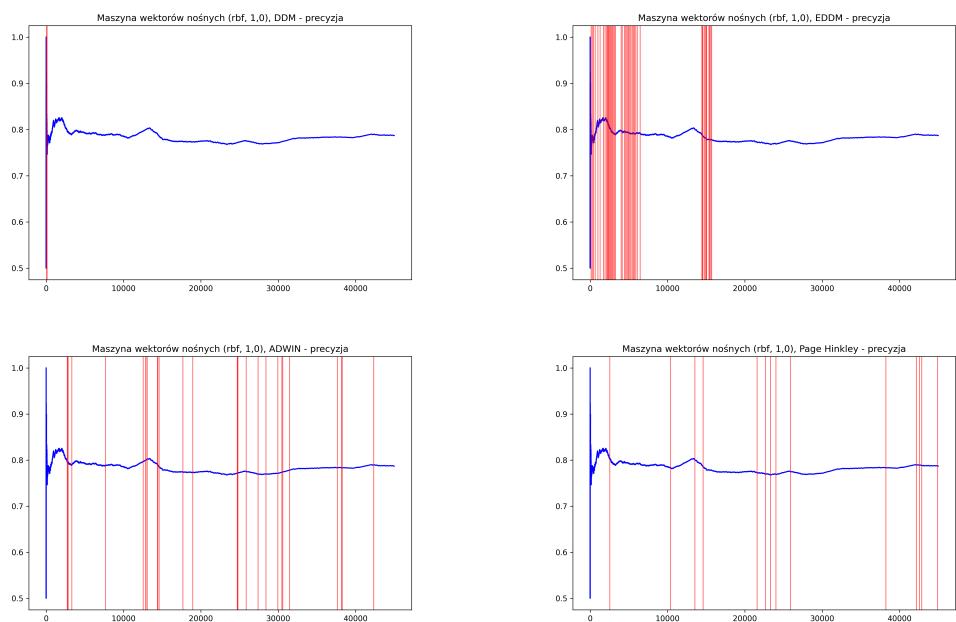
Rysunek 15: Czułość dla maszyny wektorów nośnych - metoda radialna, regularyzacja 0.1



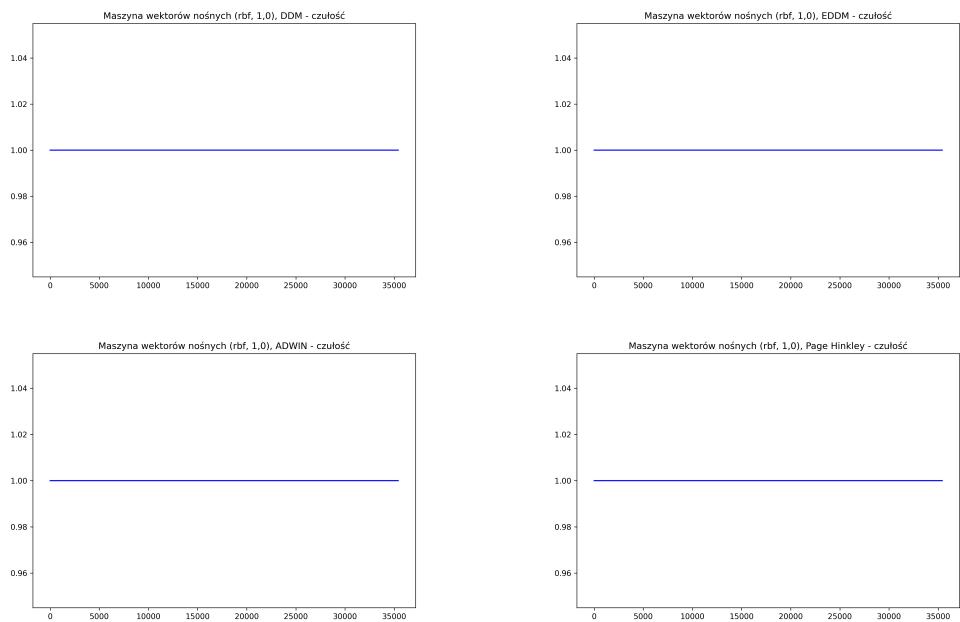
Rysunek 16: Specyficzność dla maszyny wektorów nośnych - metoda radialna, regularyzacja 0.1



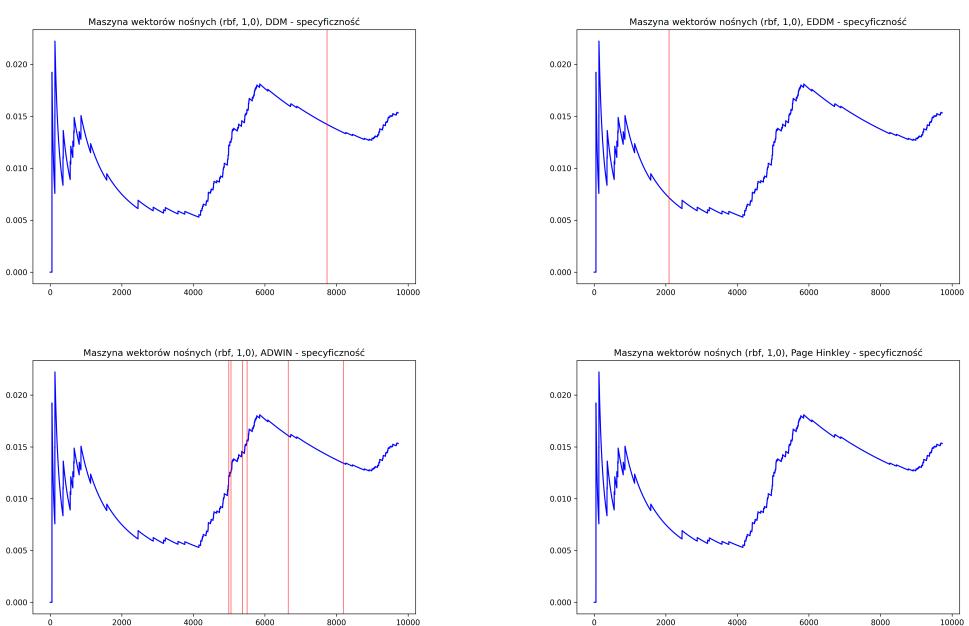
Rysunek 17: Dokładność dla maszyny wektorów nośnych - metoda radialna, regularyzacja 1



Rysunek 18: Precyza dla maszyny wektorów nośnych - metoda radialna, regularyzacja 1

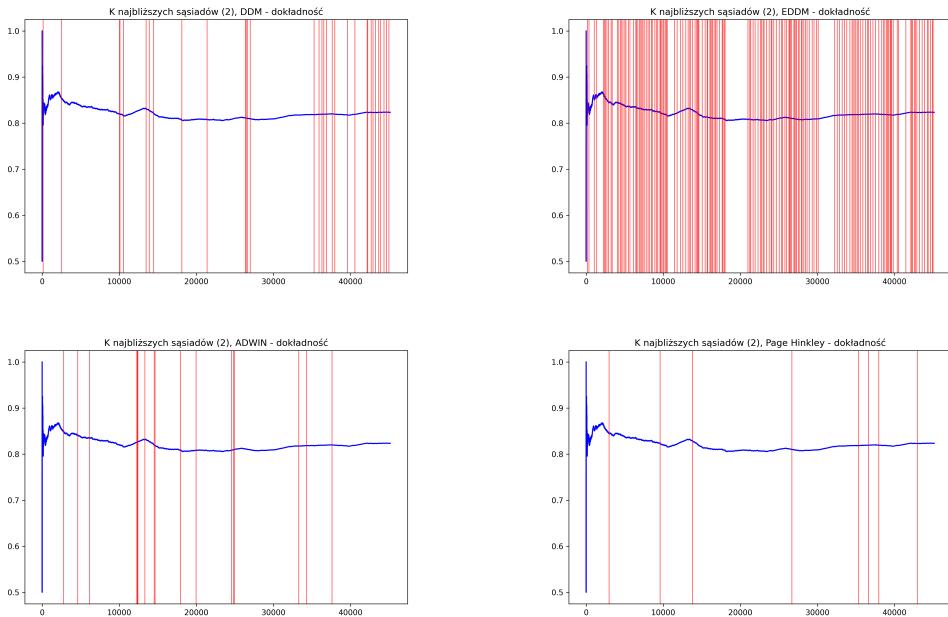


Rysunek 19: Czułość dla maszyny wektorów nośnych - metoda radialna, regularyzacja 1

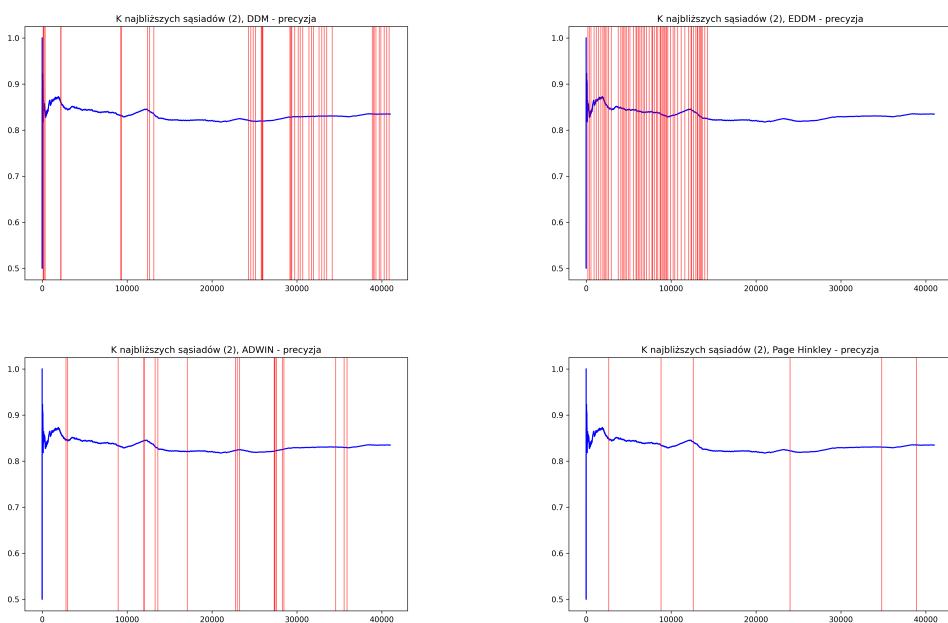


Rysunek 20: Specyficzność dla maszyny wektorów nośnych - metoda radialna, regularyzacja 1

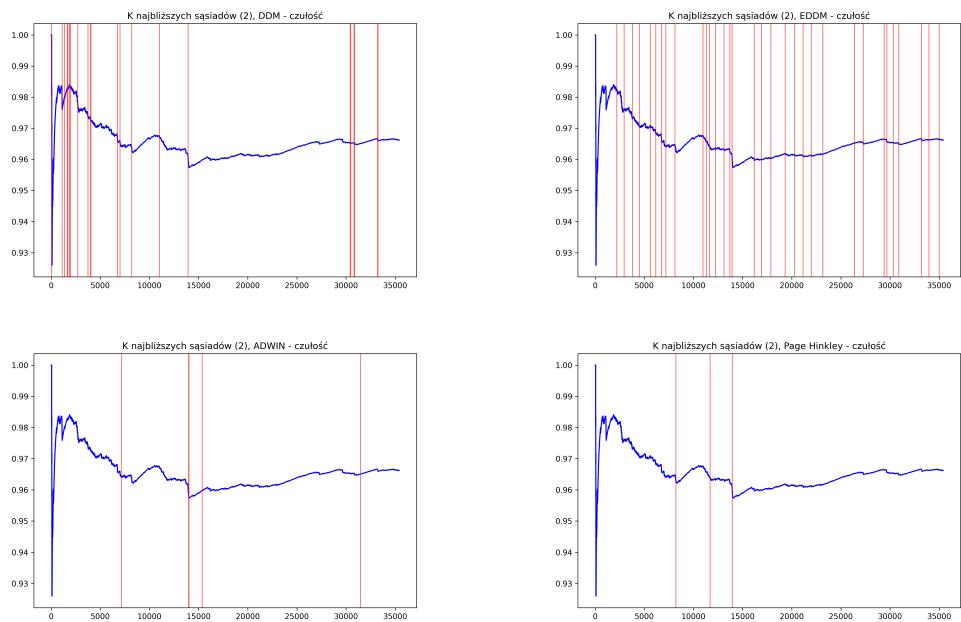
4.3. Klasyfikator k-najbliższych sąsiadów



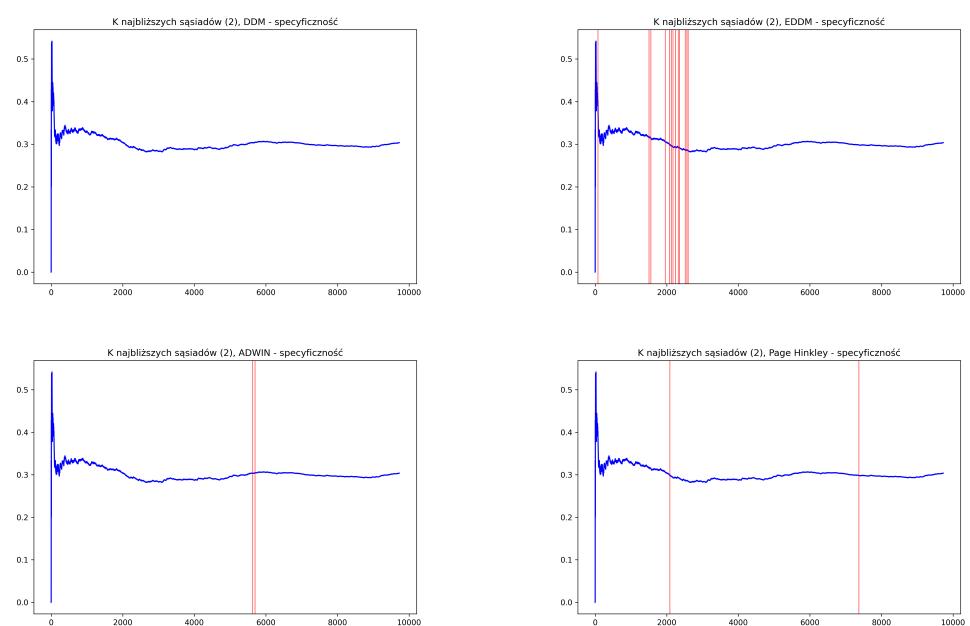
Rysunek 21: Dokładność dla klasyfikatora k-najbliższych sąsiadów - 2 sąsiadów



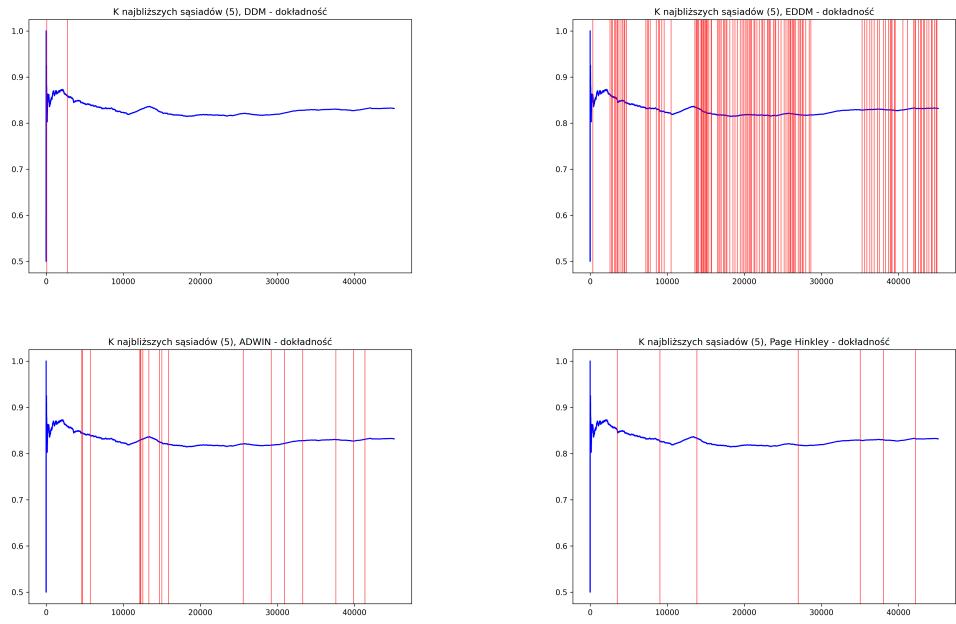
Rysunek 22: Precyza dla klasyfikatora k-najbliższych sąsiadów - 2 sąsiadów



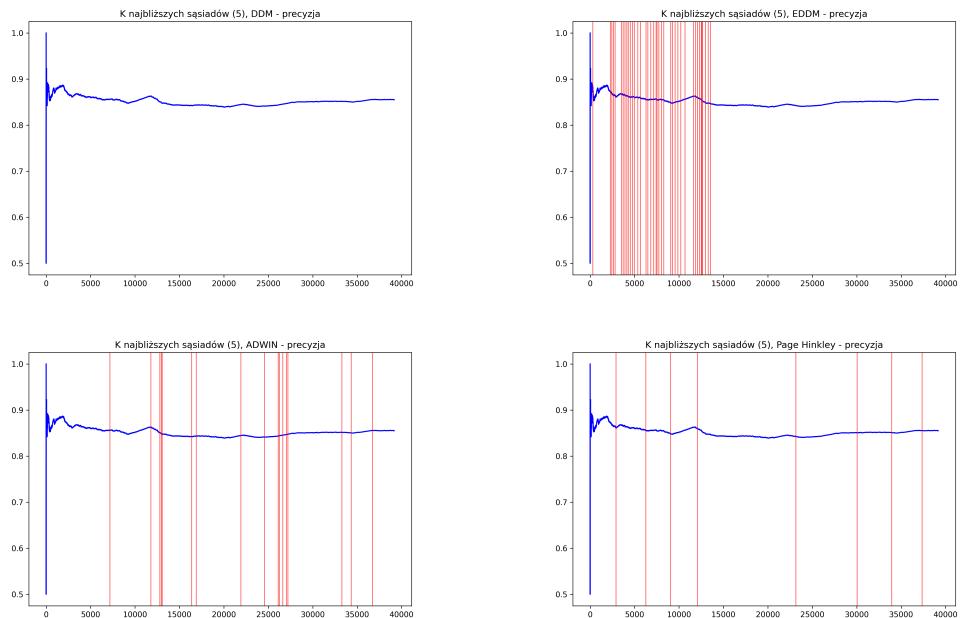
Rysunek 23: Czułość dla klasyfikatora k-najbliższych sąsiadów - 2 sąsiadów



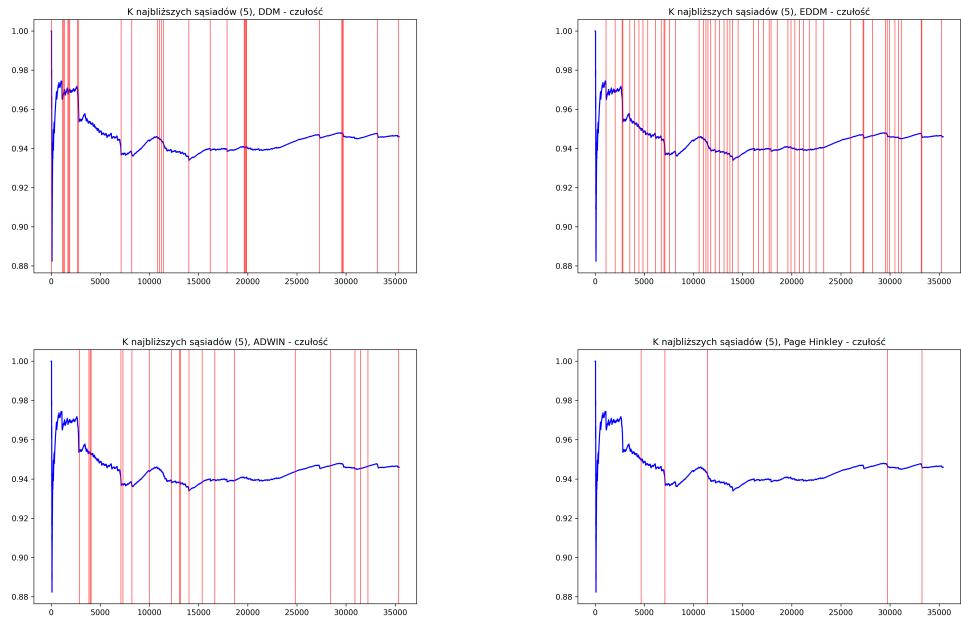
Rysunek 24: Specyficzność dla klasyfikatora k-najbliższych sąsiadów - 2 sąsiadów



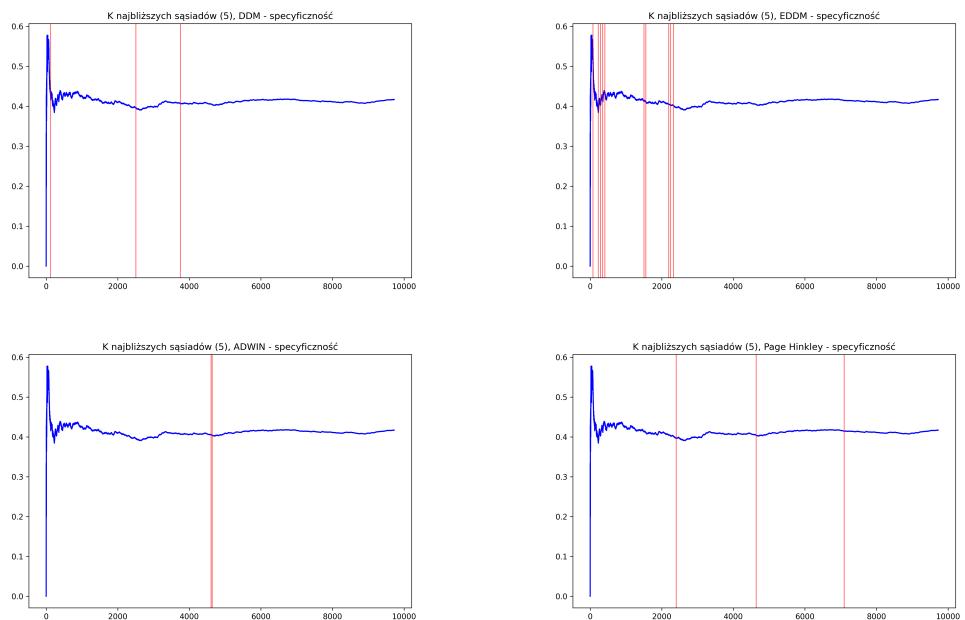
Rysunek 25: Dokładność dla klasyfikatora k-najbliższych sąsiadów - 5 sąsiadów



Rysunek 26: Precyza dla klasyfikatora k-najbliższych sąsiadów - 5 sąsiadów

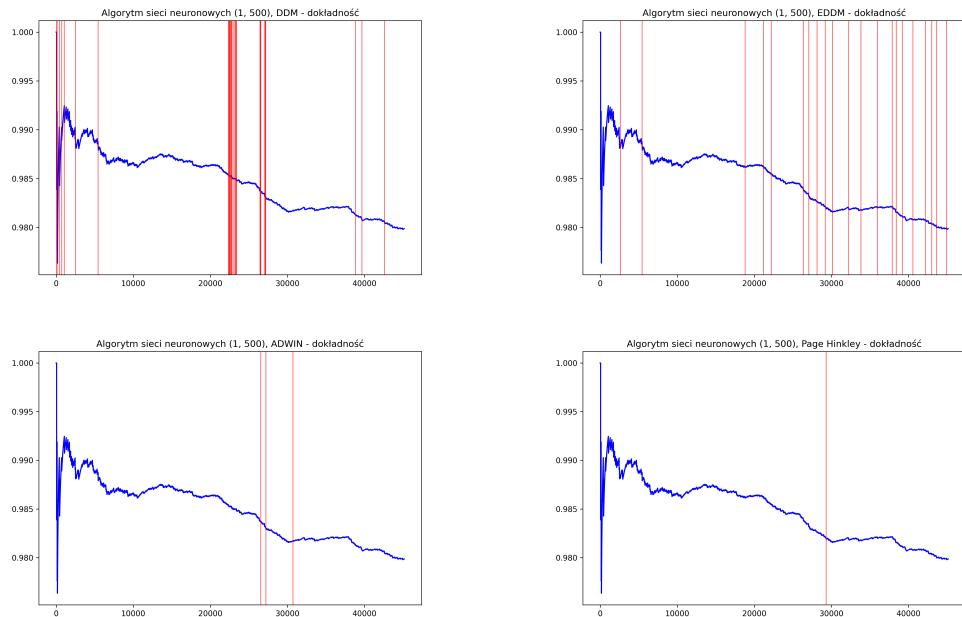


Rysunek 27: Czułość dla klasyfikatora k-najbliższych sąsiadów - 5 sąsiadów

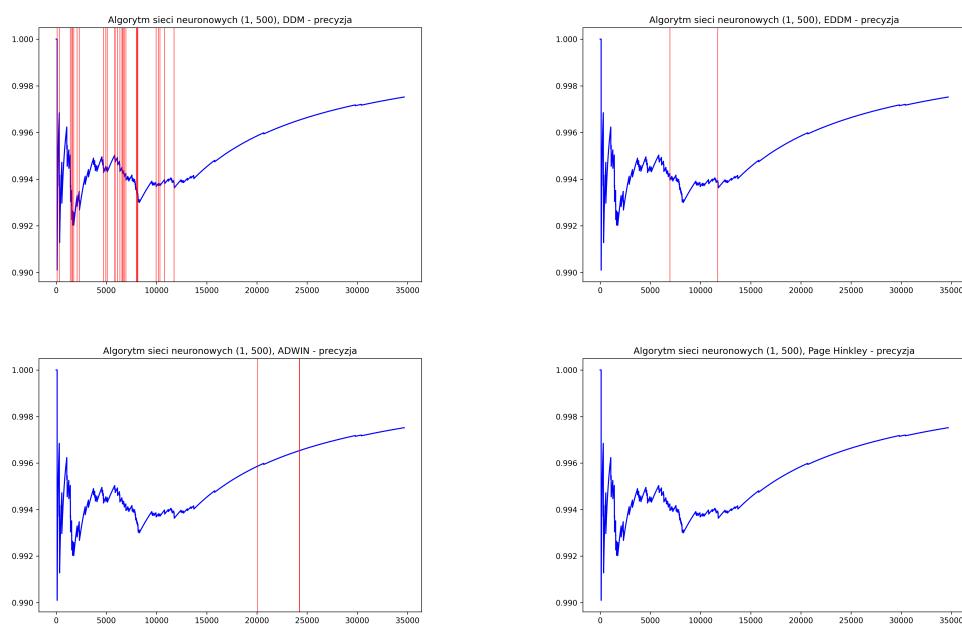


Rysunek 28: Specyficzność dla klasyfikatora k-najbliższych sąsiadów - 5 sąsiadów

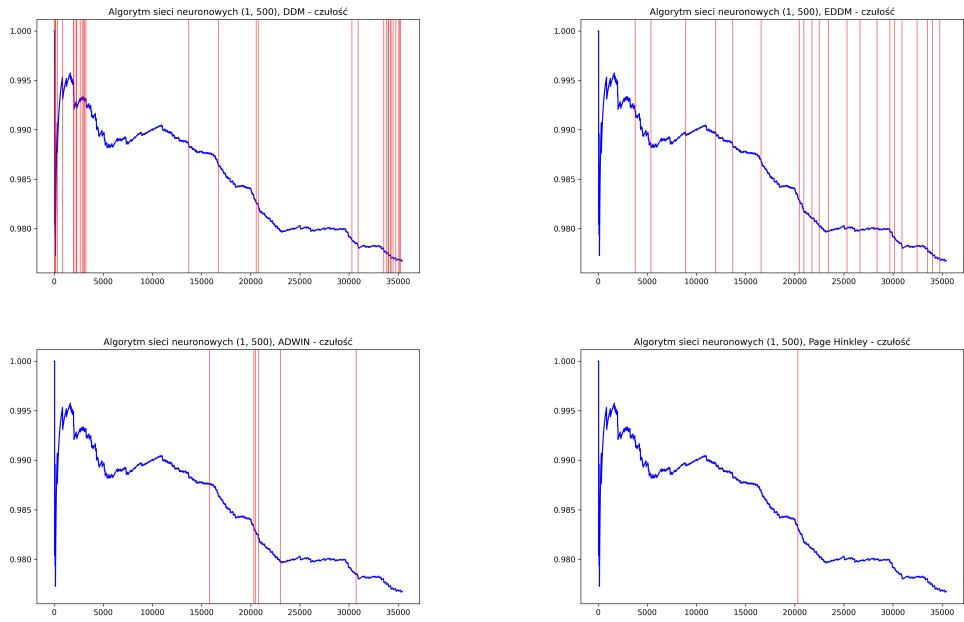
4.4. Algorytm sztucznych sieci neuronowych



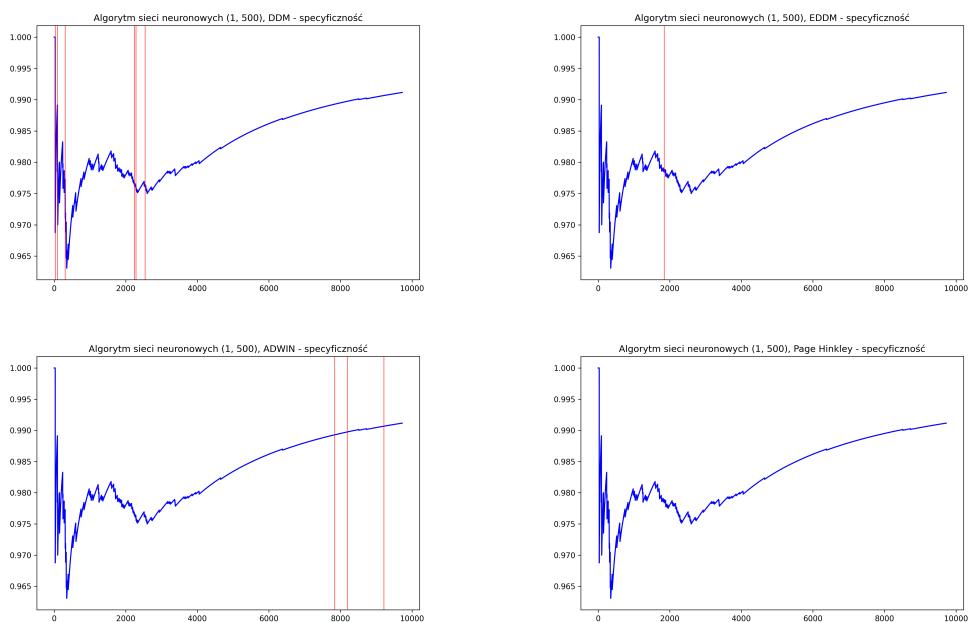
Rysunek 29: Dokładność sztucznej sieci neuronowej - 1 ukryta warstwa, 500 iteracji



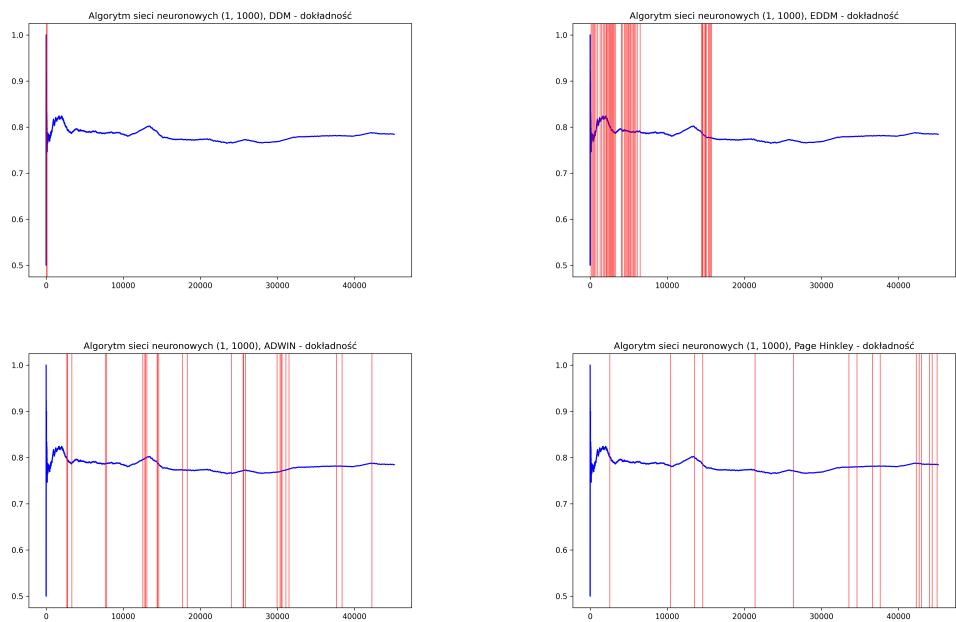
Rysunek 30: Precyza sztucznej sieci neuronowej - 1 ukryta warstwa, 500 iteracji



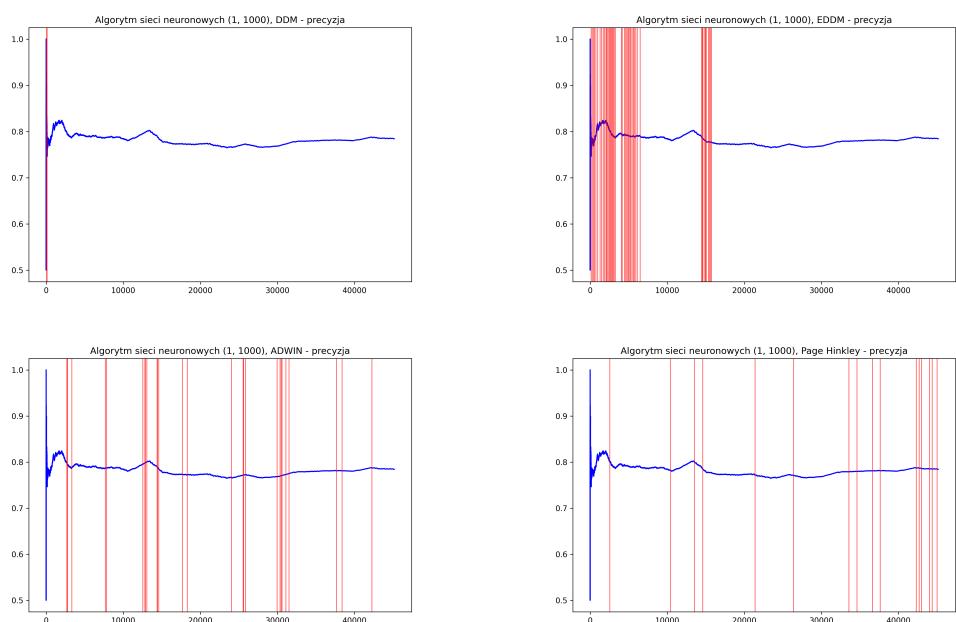
Rysunek 31: Czułość sztucznej sieci neuronowej - 1 ukryta warstwa, 500 iteracji



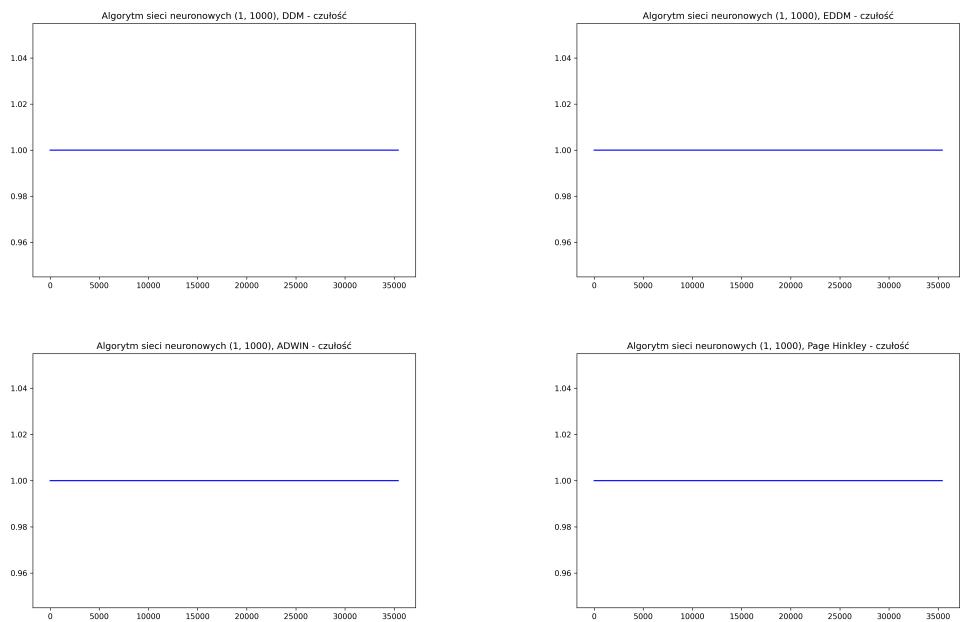
Rysunek 32: Specyficzność sztucznej sieci neuronowej - 1 ukryta warstwa, 500 iteracji



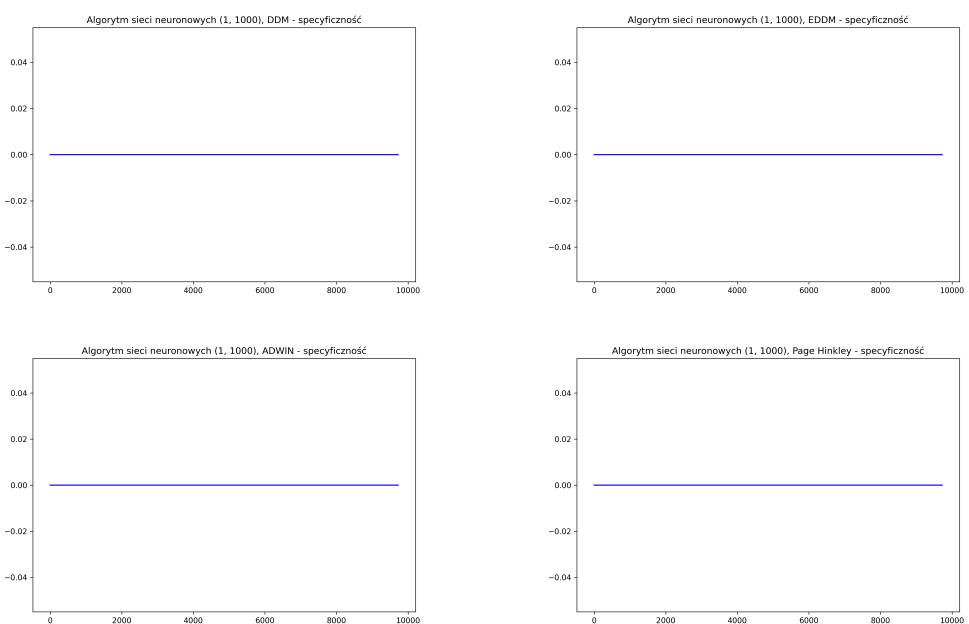
Rysunek 33: Dokładność sztucznej sieci neuronowej - 1 ukryta warstwa, 1000 iteracji



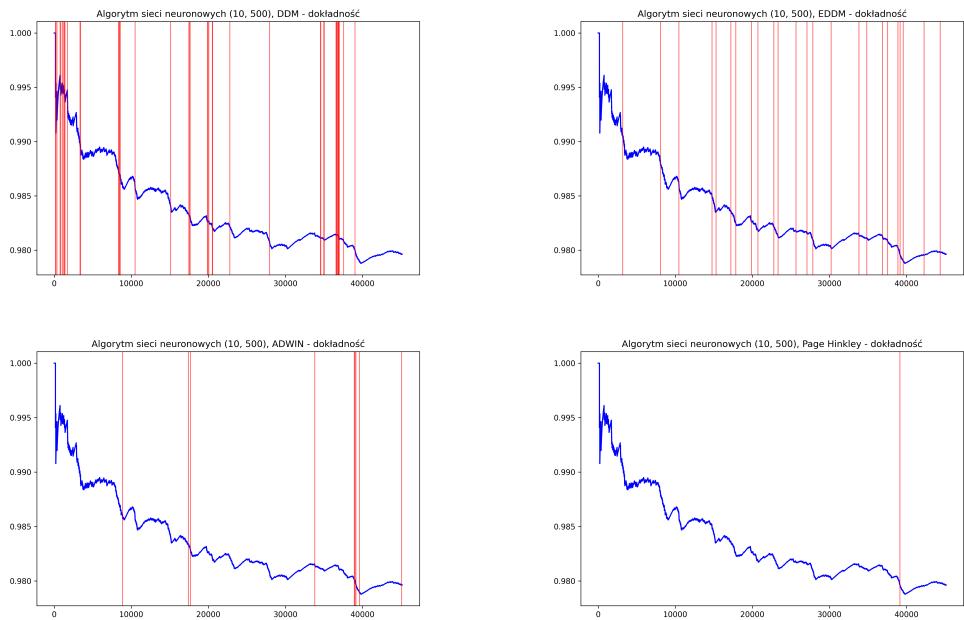
Rysunek 34: Precyza sztucznej sieci neuronowej - 1 ukryta warstwa, 1000 iteracji



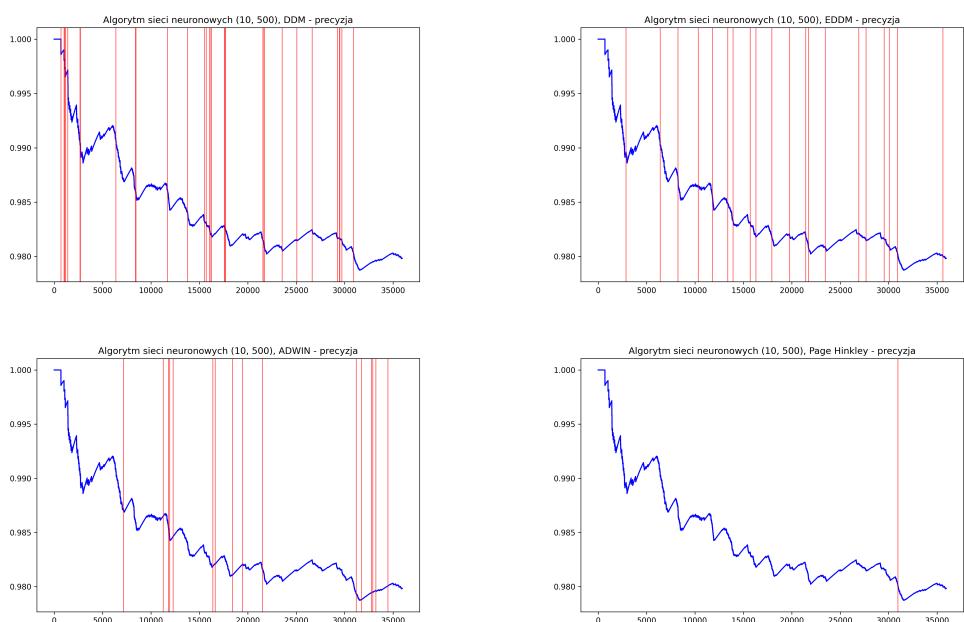
Rysunek 35: Czułość sztucznej sieci neuronowej - 1 ukryta warstwa, 1000 iteracji



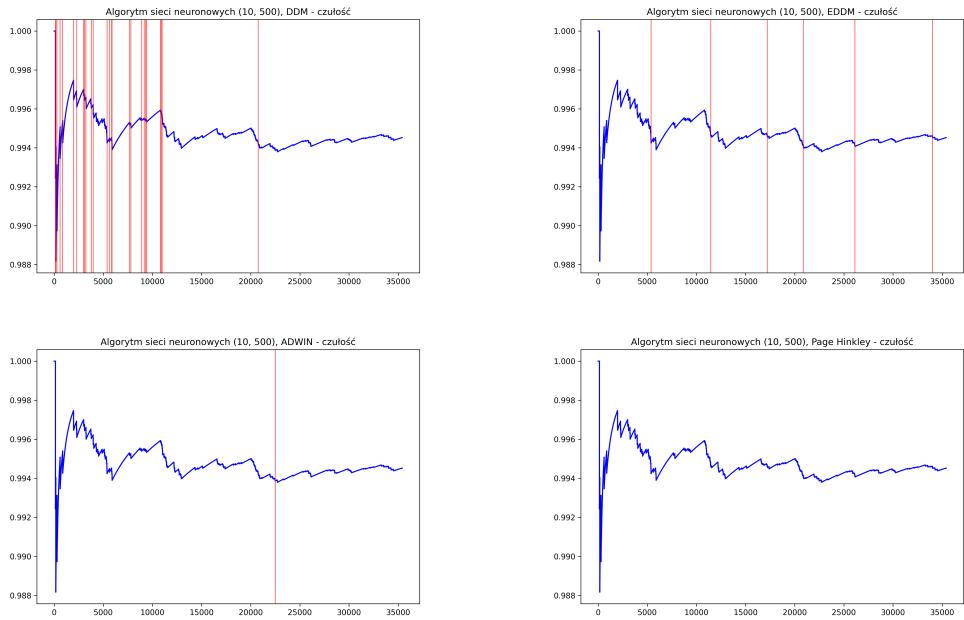
Rysunek 36: Specyficzność sztucznej sieci neuronowej - 1 ukryta warstwa, 1000 iteracji



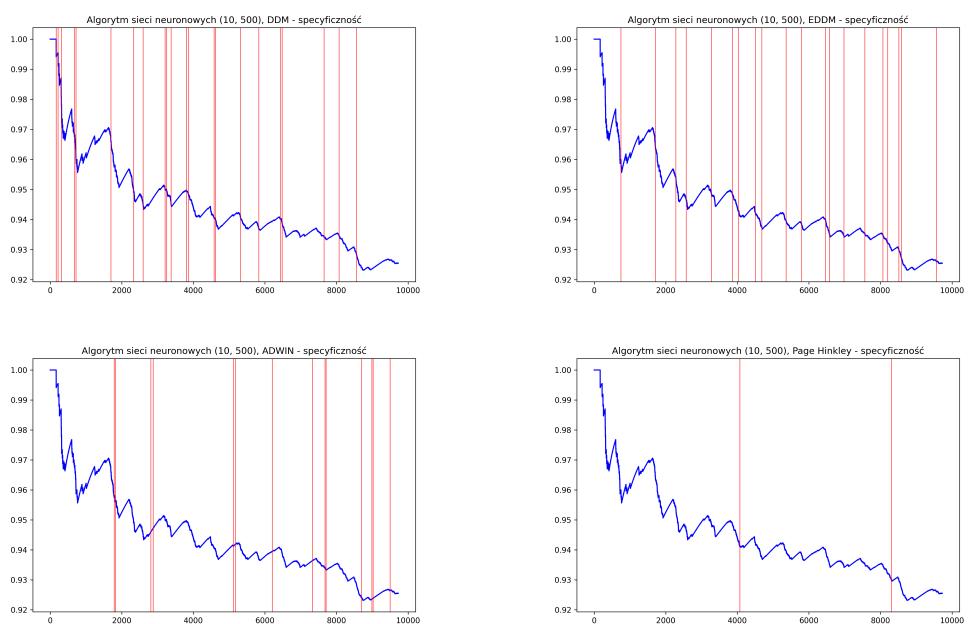
Rysunek 37: Dokładność sztucznej sieci neuronowej - 10 ukrytych warstw, 500 iteracji



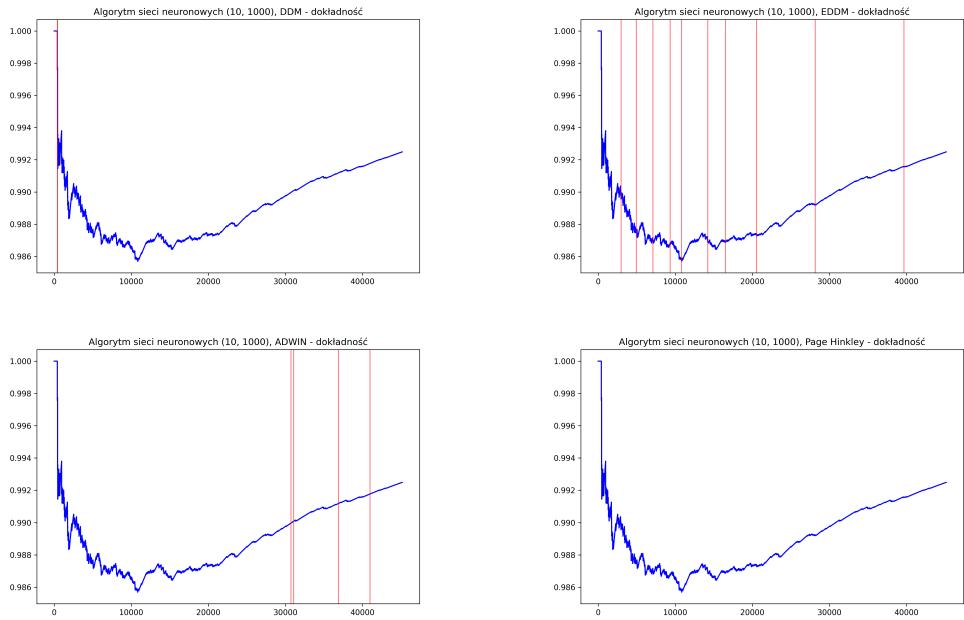
Rysunek 38: Precyzja sztucznej sieci neuronowej - 10 ukrytych warstw, 500 iteracji



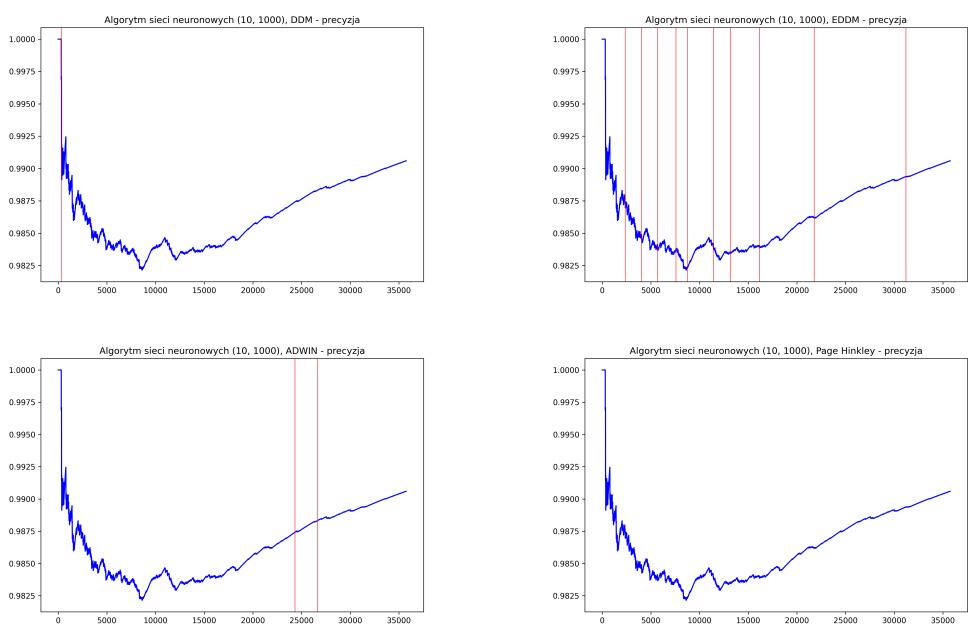
Rysunek 39: Czułość sztucznej sieci neuronowej - 10 ukrytych warstw, 500 iteracji



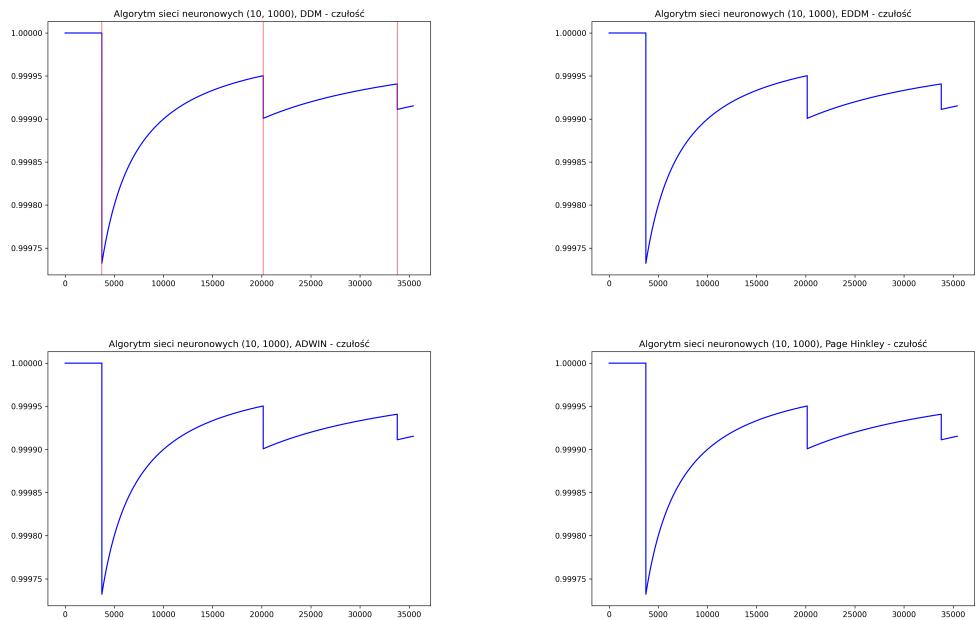
Rysunek 40: Specyficzność sztucznej sieci neuronowej - 10 ukrytych warstw, 500 iteracji



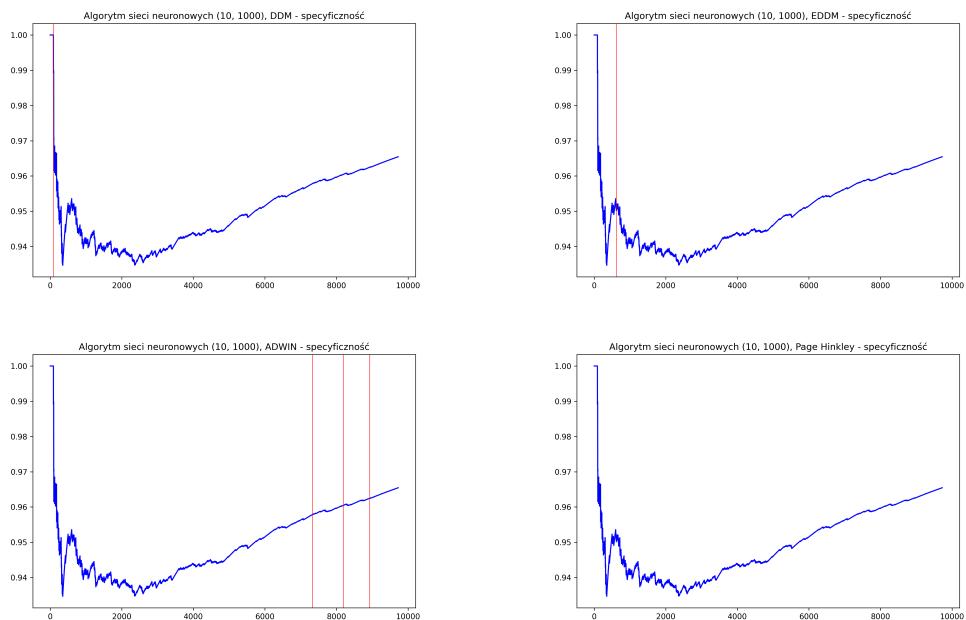
Rysunek 41: Dokładność sztucznej sieci neuronowej - 10 ukrytych warstw, 1000 iteracji



Rysunek 42: Precyza sztucznej sieci neuronowej - 10 ukrytych warstw, 1000 iteracji



Rysunek 43: Czułość sztucznej sieci neuronowej - 10 ukrytych warstw, 1000 iteracji



Rysunek 44: Specyficzność sztucznej sieci neuronowej - 10 ukrytych warstw, 1000 iteracji

5. Wnioski

1. Algorytmy DDM oraz EDDM zazwyczaj wykrywają dryft dużo częściej niż algorytm ADWIN, który zaś wykrywa dryft zauważalnie częściej od algorytmu Page Hinkley.
2. Niezależnie od wybranych parametrów, w przypadku gdy klasyfikator cechuje się bezbłenną klasyfikacją, żaden z algorytmów nie wykrywa dryftów. W przypadku bardzo małych zmian (np. 0.025%) algorytm DDM potrafi wykryć dryft, przez co jest on bardzo wrażliwy na pomyłki przy klasyfikacji.
3. Naiwny klasyfikator Bayesa:
 - Algorytm Page Hinkley wykrył dryft wyłącznie przy zastosowaniu miary czułości.
 - Poza DDM algorytmy przy mierze precyzji bardzo rzadko wykrywają dryft.
 - DDM sprawdził się w wykrywaniu spadków specyficzności – wykres charakteryzuje się skokowymi zmianami, z którymi algorytm, wedle definicji, dobrze sobie radzi.
4. Maszyna wektorów nośnych:
 - Parametr regularyzacji nie wpływa na wykrycie dryftu w przypadku jądra radialnego - dla regularyzacji 0,1 algorytm zachowuje się podobnie także w przypadku jądra wielomianego, dopiero przy regularyzacji 1 widzimy zmiany w mięsach wykrycia dryftu.
 - Algorytm DDM dla miary dokładności wykrywa dryft jedynie na początku w przypadku większości parametrów przebiegu klasyfikacji.
 - Algorytm EDDM dla miar dokładności oraz precyzji wykrywa dryft zbyt często, zaś dla miary czułości - wcale.
 - Algorytm ADWIN przy mierze specyficzności wykrywa dryft dużo częściej niż w inne algorytmy, co nie zdarza się w przypadku innych klasyfikatorów.
5. Klasyfikator k-najbliższych sąsiadów:
 - Liczba najbliższych sąsiadów nie ma znaczącego wpływu na wykrycie dryftu dla większość algorytmów.
 - Algorytm DDM dużo częściej wykrywa dryft dla 2 sąsiadów w przypadku miar dokładności oraz precyzji.
 - Algorytm EDDM zwraca bardzo dużo punktów potencjalnych dryftów, natomiast w żaden sposób nie są one odzwierciedlane na wykresie. Może to świadczyć o tym, że nie jest on optymalnym wyborem dla tego klasyfikatora.

6. Sztuczna sieć neuronowa:
 - Algorytmy wykrywają zauważalnie rzadziej dryft dla 10 warstw ukrytych oraz 1000 iteracji. Konfiguracja ta wykazała najlepsze wyniki we wszystkich wskaźnikach, dlatego brak dryftu jest tu logicznym efektem.
 - Algorytm DDM na Rys. 43 pokazowo prezentuje teoretyczne działanie algorytmów detekcji dryftu
 - Dla 1 warstwy ukrytej 1000 iteracji okazało się być zbyt dużą liczbą – wystąpiło tutaj zjawisko przeuczenia, stąd skrajne wartości czułości i specyficzności.
7. Sieć neuronowa okazała się najtrajniejszą metodą klasyfikacji dla wybranego przez nas zbioru danych.
8. Nie da się jednoznacznie wybrać najlepszej metody detekcji dryftu. Każdy z nich sprawdzał się dobrze w określonych konfiguracjach. Dobór metody wydaje się więc być uzależniony od wybranego klasyfikatora oraz użytej miary.