Sprawozdanie
Podstawy Sztucznej Inteligencji
Perceptron breast cancer
WEKA

Natalia Gadocha 304165
Geoinformatyka III

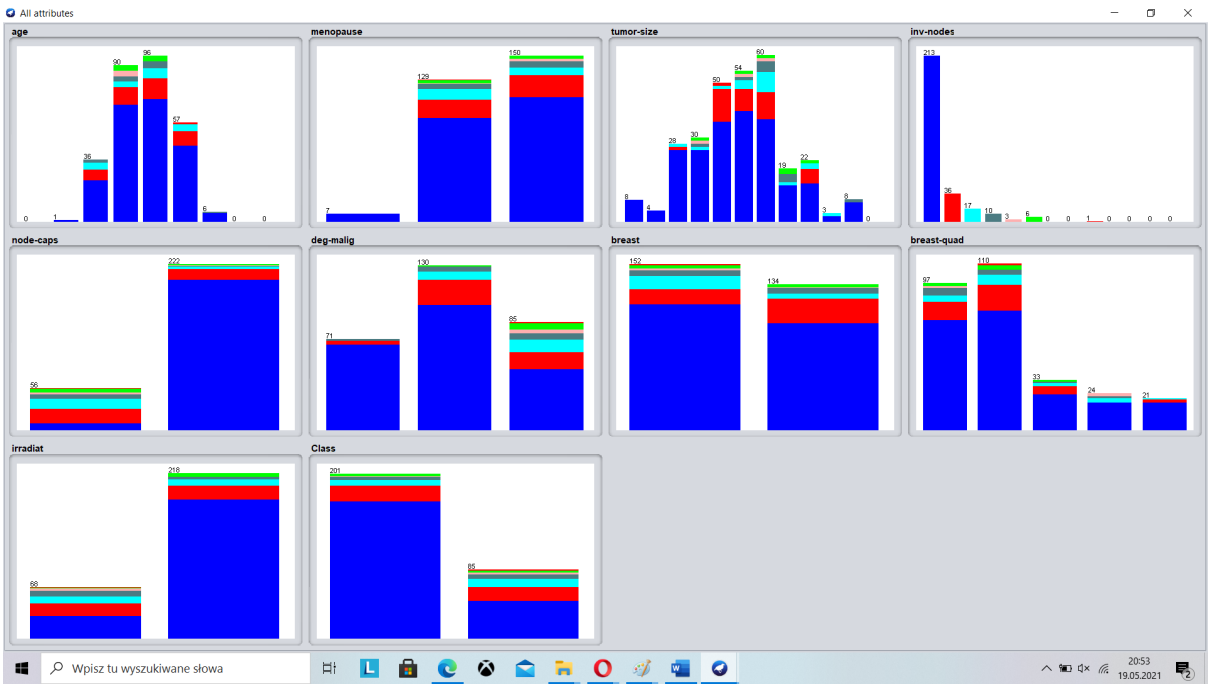
Badane dane

Nasze dane są oparte na informacjach dotyczących zachorowań na raka piersi. Instancje są opisane przez 9 atrybutów i jeden atrybut klasy. Każde z nich posiada 286 instancji. Suma poszczególnych wag wynosi również 286. Wśród wspominanych atrybutów znajdują się: age, menopause, tumor-size, inv-nodes, node-caps, deg-malig, breast, breast-quad, irradiat, class. Atrybut Class składa się z recurrence-events (85 wartości) oraz no-recurrence-events (201 wartości). Wszystkie atrybuty są typu Nominal.

Wartości brakujące występują dla dwóch argumentów:

- node-caps - 3% (8 wartości)
- breast-quad - w przybliżeniu 0% (1 wartość)

Rozkład powyższych atrybutów prezentuje się następująco:



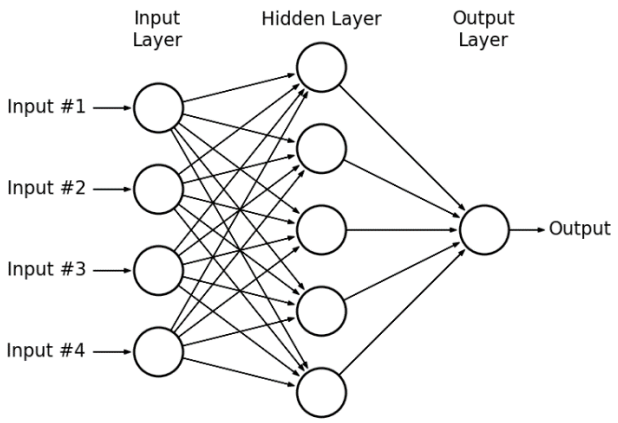
A dodatkowo, nasze atrybuty prezentują się tak:

Attribute Name	Attribute Values
Class	{no-recurrence-events, recurrence-events}
Age	{10-19, 20-29, 30-39, 40-49, 50-59, 60-69, 70-79, 80-89, 90-99}
Menopause	{lt40, ge40, premeno}
Tumor-size	{0-4, 5-9, 10-14, 15-19, 20-24, 25-29, 30-34, 35-39, 40-44, 45-49, 50-54, 55-59}
Inv-nodes	{0-2, 3-5, 6-8, 9-11, 12-14, 15-17, 18-20, 21-23, 24-26, 27-29, 30-32, 33-35, 36-39}
Node-caps	{yes, no}
Deg-malig	{1, 2, 3}
Breast	{left, right}

Trochę o Multilayer Perceptron

Jest to typ sztucznej sieci neuronowej. Składa się zwykle z jednej warstwy wejściowej, kilku warstw ukrytych oraz jednej wyjściowej.
Perceptrony wielowarstwowe wykorzystuje się często do nadzorowania problemów uczenia się. Trening ten polega na dostosowaniu parametrów lub wag i odchyleń modelu w celu zminimalizowania błędów.
MLP jest bardzo pomocny w rozwiązywaniu wielu różnych problemów, które są trudne przy użyciu zwykłych algorytmów opartych na zasadach programowania. Wyróżnia go to, iż jest może być on szkolonym przy pomocy algorytmu wstecznej propagacji błędów.
Jest również bardzo czuły na minima lokalne. Aby ich uniknąć należy powtórzyć kilka razy proces nauki z różnymi punktami startowymi. Od nich bowiem zależy jego trajektoria.

Schematyczne działanie algorytmu przedstawione zostało na poniższym obrazie:



https://www.researchgate.net/figure/A-hypothetical-example-of-Multilayer-Perceptron-Network_fig4_303875065

Przebieg i wyniki analiz

Poniższe dane są wynikiem przeprowadzenia kilkukrotnego przejścia algorytmu i wybranych tych danych, które dały najlepszy wynik o tych samych parametrach. Kolejno będą testowane różne opcje danych ustawień. W kolejnych krokach dla tych samych ustawień, tylko z różnymi parametrami zostaną podane tylko powstałe różnice. Różne wyniki sprawiają, iż może pojawić się ich duża wariancja. Dążyć będziemy do stworzenia jak najlepszego modelu.

Zacznijmy więc od użycia opcji Use training set, żeby poglądowo tylko spojrzeć na uzyskane potencjalne wyniki.

Use training set				
Correctly Classified Instances	276 96.5035 %	a	b	classified as
Incorrectly Classified Instances	10 3.4965 %	197	4	a = no-recurrence-events
Total Number of Instances	286	6	79	b = recurrence-events
Kappa statistic	0.9157			
Precision	0.965			
Mean absolute error	0.0482			
Root mean squared error	0.1567			
Relative absolute error	11.5332 %			
Root relative squared error	34.2946 %			

Wyniki jakie otrzymaliśmy nie są wiarygodne. Otrzymujemy je mocno przejaskrawione w związku z czym nie bierzemy ich pod czujne oko analizy i dalszych badań.

Tą samą procedurę przeprowadzimy również dla cross-validation = 10 oraz percentage split = 70%. A wyniki prezentują się następująco:

Cross-validation = 10				
Correctly Classified Instances	185 64.6853 %	a	b	classified as
Incorrectly Classified Instances	101 35.3147 %	150	51	a = no-recurrence-events
Total Number of Instances	286	50	35	b = recurrence-events
Kappa statistic	0.1575			
Precision	0,648			
Mean absolute error	0.3552			
Root mean squared error	0.5423			
Relative absolute error	84.8811 %			
Root relative squared error	118.654 %			

Percentage-split = 70%				
Correctly Classified Instances	162 72.093 %	a	b	classified as
Incorrectly Classified Instances	124 27.907 %	121	60	a = no-recurrence-events
Total Number of Instances	286	64	41	b = recurrence-events
Kappa statistic	0.3534			
Precision	0,717			
Mean absolute error	0.3075			
Root mean squared error	0.5001			

Relative absolute error	69.8012 %
Root relative squared error	101.1071 %

W tym zestawieniu lepsze wyniki otrzymaliśmy dla użytej opcji cross-validation - ma większą trafność i mniejsze błędy.

Przed jeszcze bardziej dogłębną analizą wypróbujemy działanie dodawanego szumu. Wartość ta jest używana w wece do zapoczątkowania generatora liczb losowych. Zaczniemy od dodania szumu równego 100.

seed = 100				
Correctly Classified Instances	108 37.7622 %	a	b	classified as
Incorrectly Classified Instances	178 62.2378 %	47	154	a = no-recurrence-events
Total Number of Instances	286	24	61	b = recurrence-events
Kappa statistic	-0.0337			
Precision	0,550			
Mean absolute error	0.5107			
Root mean squared error	0.512			
Relative absolute error	122.0612 %			
Root relative squared error	112.0131 %			

Co możemy więc tu dostrzec? Widzimy, iż otrzymujemy tutaj duże błędy. Wartość Kappa jest ujemna. Nasza precyzja oraz liczba poprawnie sklasyfikowanych danych nie są duże. Spróbujmy dodać trochę mniejszą ilość szumu (= 50). Jednakże pomimo swoich wad, szum pozwala uciec z płytszych minimów, chociaż generowanie modeli trwa zdecydowanie dłużej.

Różnicami w uzyskanych wynikach dla dodanego szumu mogą być:

- zwiększa się ilość poprawnie przyporządkowanych instancji (98) oraz delikatnie precyzja (0,498)
- wartość Kappa jest już dodatnia, choć niewielka (0,0295)
- błędy zmniejszają się

Przeciwny efekt otrzymujemy dla danych o zwiększonym szumie. Możemy zatem wnioskować, iż dodanie szumu będzie utrudniało pracę z danymi oraz obniżało wyniki algorytmu.

Kolejnym aspektem jaki poruszymy będzie zmiana czasu do nauki. Zaczniemy od ustawienia wartości 200.

time to training = 200				
Correctly Classified Instances	186 65.035 %	a	b	classified as
Incorrectly Classified Instances	100 34.965 %	154	47	a = no-recurrence-events
Total Number of Instances	286	53	32	b = recurrence-events
Kappa statistic	0.1456			
Precision	0,643			
Mean absolute error	0.3605			
Root mean squared error	0.5395			
Relative absolute error	86.1705 %			
Root relative squared error	118.026 %			

Natomiast jeżeli zwiększymy tą wartość na 700 możemy dostrzec:

- identyczne, choć trochę słabsze poprawne przypasowanie instancji oraz precyzję
- w niewielkim stopniu zwiększone błędy

Podczas zmniejszenia wydzielonego czasu na 100 widzimy, iż:

- wartości poprawnie przyporządkowanych instancji oraz precyzja zwiększyły się (189 oraz 0,653)
- błędy zmniejszyły się

Sprawdźmy teraz jak działa dodawanie ukrytych warstw na działanie algorytmu.

hidden layers = 2				
Correctly Classified Instances	209 73.0769 %	a	b	classified as
Incorrectly Classified Instances	77 26.9231 %	175	26	a = no-recurrence-events
Total Number of Instances	286	51	34	b = recurrence-events
Kappa statistic	0.2957			
Precision	0.713			
Mean absolute error	0.3254			
Root mean squared error	0.4653			
Relative absolute error	77.777 %			
Root relative squared error	101.7919 %			

Natomiast jeżeli będziemy zwiększać liczbę ukrytych warstw - przykład dla siedmiu:

- zmniejsza się ilość poprawnie sklasyfikowanych instancji (191) oraz precyzja (0,662)
- Wszystkie brane pod uwagę błędy zwiększają się

Dla całkowitego braku warstw ukrytych otrzymujemy również skrajne wyniki, których nie możemy brać pod uwagę. Ich błędy są za duże, aby dane były wiarygodne.

Kolejnym czynnikiem, który będziemy badać to batch size określający wielkość partii.

batch size = 20				
Correctly Classified Instances	197 68.8811 %	a	b	classified as
Incorrectly Classified Instances	89 31.1189 %	167	34	a = no-recurrence-events
Total Number of Instances	286	55	30	b = recurrence-events
Kappa statistic	0.1979			
Precision	0,668			
Mean absolute error	0.3406			
Root mean squared error	0.5109			
Relative absolute error	81.4043 %			
Root relative squared error	111.7654 %			

Dla batch size = 150 możemy zaobserwować:

- Taką samą ilość poprawnie przyporządkowanych instancji
- Dokładnie taką samą precyzję
- Brak zmieniających się innych wartości

Podobną sytuację otrzymujemy dla większych wartości batch size. Możemy więc wywnioskować, iż algorytm nie jest bardzo czuły na ten parametr.

Learning rate jest następnym czynnikiem, który będziemy badać. Jest to współczynnik uczenia się algorytmu propagacji wstecznej. Zaczniemy od wartości równej 0.1.

learning rate = 0.1				
Correctly Classified Instances	208 72.7273 %	a	b	classified as
Incorrectly Classified Instances	78 27.2727 %	176	25	a = no-recurrence-events
Total Number of Instances	286	53	32	b = recurrence-events
Kappa statistic	0.2786			
Precision	0,707			
Mean absolute error	0.3524			
Root mean squared error	0.4321			
Relative absolute error	84.2157 %			
Root relative squared error	94.537 %			

Zwiększając stopniowo omawiany parametr otrzymujemy:

- coraz gorsze przyporządkowania instancji, oraz zmniejszającą się precyzję
- coraz większe błędy przekraczające 100% (dla root relative squared error)
- coraz mniejszą wartość Kappa

Kolejnym aspektem jaki omówimy będzie parametr momentum. Mówi nam on o tempie pędu nauki dla algorytmu wstecznej propagacji.

momentum = 0.1				
Correctly Classified Instances	208 72.7273 %	a	b	classified as
Incorrectly Classified Instances	78 27.2727 %	177	24	a = no-recurrence-events
Total Number of Instances	286	54	31	b = recurrence-events
Kappa statistic	0.2731			
Precision	0,706			
Mean absolute error	0.3541			
Root mean squared error	0.4304			
Relative absolute error	84.6393 %			
Root relative squared error	94.156 %			

Jeżeli ustawimy tą wartość na 0.01 możemy dostrzec:

- Delikatnie zwiększoną poprawność sklasyfikowanych instancji (210)
- Precyzja również wrasta (0,714)
- Błędy zmniejszają się średnio o 2 punkty procentowe

Próba zmiany dokładności (wartości decimals) nie przyniosła żadnych dodatkowych efektów. Wartości te powtórzyły się z powyższego przykładu, bez zmian poprawności ani błędów. Podobną sytuację otrzymujemy dla parametru validation threshold, który to mówi o liczbie kolejnych wzrostów błędu dozwolonych dla testów walidacyjnych przed zakończeniem uczenia. Zmiana wartości parametru nie wpływa na uzyskane dane.

Spróbujemy zatem wpłynąć na badany algorytm poprzez ostatni już parametr - Validation set size. Jego wartość ustawmy początkowo na 50. A oto wyniki:

validation set size = 50				
Correctly Classified Instances	200 69.9301 %	a	b	classified as
Incorrectly Classified Instances	86 30.0699 %	193	8	a = no-recurrence-events
Total Number of Instances	286	78	7	b = recurrence-events
Kappa statistic	0.0558			
Precision	0,639			
Mean absolute error	0.4009			
Root mean squared error	0.4453			
Relative absolute error	95.8058 %			
Root relative squared error	97.4243 %			

Próby zwiększenia tej wartości powodowały pogorszenie się uzyskanych wyników. Jednakże po jej zmniejszeniu (na wartość równą 5) dostajemy bardziej poprawne sklasyfikowanie instancji (210, 73.4266%). Nasza precyzja również się zwiększa (0,712), a błędy zmniejszają.

Podsumowanie

Powyższa przeprowadzona analiza ukazuje dokładnie wady i zalety algorytmu Multilayer Perceptron. Dostrzegamy, iż nie jest on na pewno podatny na problemy związane z wymiarem przestrzeni. Inne parametry oraz ich wpływ został opisany na górze.
Sieci neuronowe w trakcie uczenia potrafią od razu skoncentrować się na pewnym podzbiorze przestrzeni wejściowej. Jego zdolność uczenia się na przykładach sprawia, że sztuczne sieci neuronowe są bardzo elastyczne i wydajne. Widzimy również wpływ dodawanego szumu na jego działanie. Algorytm ten był jednakże jednym z najwolniejszych z wcześniej poznanych i użytych w tym programie. Generowanie, powstawanie oraz ponowne tworzenie modelu trwało dość długo.