Praca Domowa 3

Julia Girtler

Wstęp

Zadanie polegało na implementacji algorytmu k najbliższych sąsiadów. Metoda została zaimplementowana według instrukcji podanej w opisie pracy domowej.

Sprawdzenie poprawności działania metody

Aby sprawdzić poprawność meotdy została stowrzona pętla, która powtarzała się 3 razy. W każdej pętli tworzona była losowa macierz o losowym wymiarze (przy czym liczba kolumn =2) i wektor etykiet - w naszym przypadku wektor zer i jedynek, ale wektory o innej liczbeności etykiet również zadziałają.

Sprawdzone zostało czy przy obraniu parametru $k = 1$ (ilość najbliższych sąsiadów) i tożsamych próbach uczących i testowych, wektor etykiet zostanie idealnie odtworzony.

Dla wylosowanych trzech macierzy sprawdzona została metryka $L1, L2, L\infty$. Powstał nowy wektor w którym przechowane zostało 9 zmiennych boolowych: “True” - gdy oczekiwany wektor etykiet byl identycznościowy z wektorem obliczonym za pomocą zaimplementowanej funkcji $knn()$, “False” - gdy wektory różniły się conajmniej na jednym miejscu.

Jeśli w wektorze zmiennych boolowych wszystkie wartości były “True” to zwracany był napis: “Zaimplementowany algorytm odtwarza idealnie wekotr etykiet w każdym przykładzie.”

Każda próba przynosiła oczekiwane efekty.

Badanie dokładności dla różnych wartośći k i różnych metryk

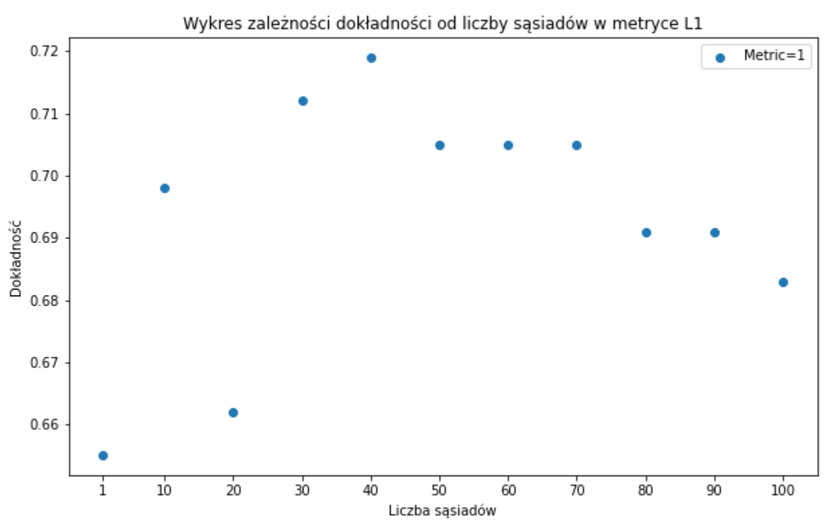
Do przetestowania zaimplementowanej metody zostały użyte dane wykorzytsywane na 8. laboratoriach - SAheart.

Dane SAheart to zestaw danych dotyczący choroby serca w Republice Południowej Afryki. Zawiera on informacje na temat pacjentów i pewnych czynników ryzyka związanych z chorobą serca.

Dane zostały odpowiednio obrobione, podzielone na: y - kolumnę zaiwerającą informację o wystąpieniu choroby (nasz wektor etykiet) i X- pozostałe koluny.

Powstał zbiór treningowy i testowym (test_size = 0,3), a następnie X_train oraz X_test zostały przeskalowane.

Metryka	Średnia dokładność	Średnie ROC AUC score
$L1$	$0,693$	$0,585$
$L2$	$0,681$	$0,574$
$L\infty$	$0,657$	$0,539$



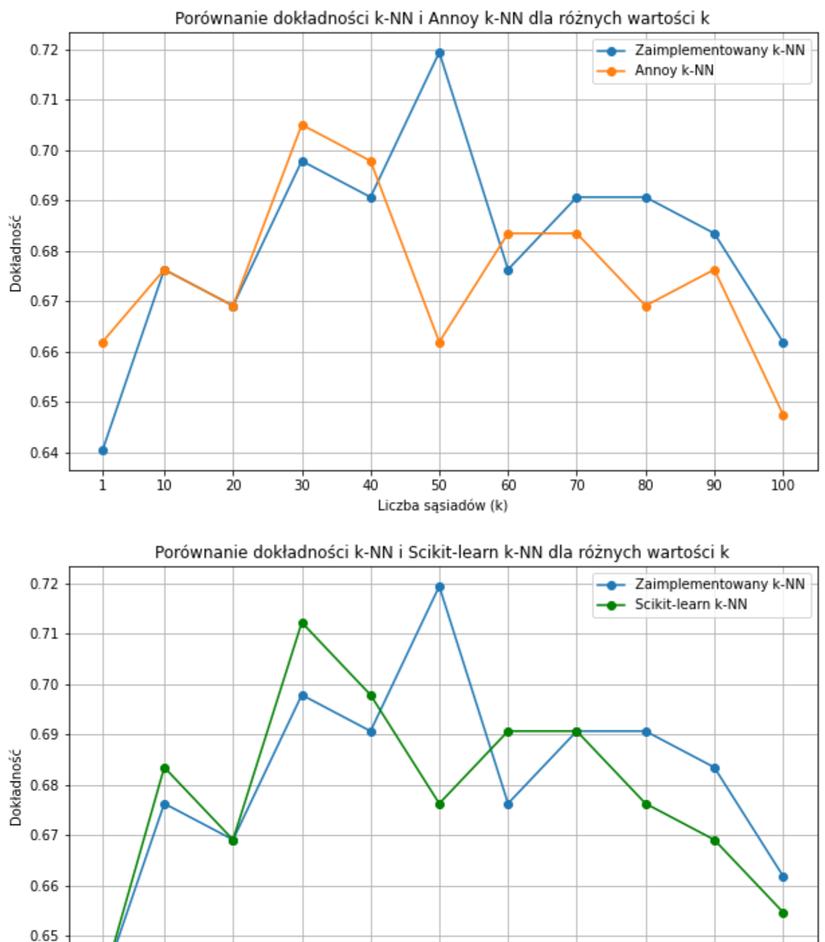
Wnioski

- Dla metryki $L1$ największa dokładność była dla liczby sąsiadów wynoszącej 40, $L2$ - 50, $L\infty$ -30. Są to wartości do siebie zbliżone, możemy więc przypuszczać że dla liczby sąsiadów w okolicy 40 algorytm radzi sobie najlepiej.
- Niskie wyniki dokładności w każdym przypadku były dla liczby sąsiadów równej 1, co może oznaczać, że przy tak małym parametrze k model ulega przeuczeniu.
- Dokładność spada od pewnego momentu przy zwiększaniu liczby sąsiadów.
- Najlepsze średnie wyniki dokładności i roc auc score są otrzymywane przy użyciu metryki $L1$, niewiele gorsze są dla metryki $L2$, co sugeruje korzystanie z nich w celu uzyskania najlepszych predykcji.
- Najgorzej wypada metryka $L\infty$

Porównanie metody z gotowymi implementacjami

W celu oceny skuteczności i poprawności zaimplementowanego algorytmu zostały wykorzystane gotowe metody: Annoy (GitHub - spotify/annoy: Approximate Nearest Neighbors in C++/Python optimized for memory usage and loading/saving to disk) oraz KNeighborsClassifier z pakiety scikit-learn.

W obu przypadkach dla funkcji knn obraliśmy parametr p = 2, ponieważ w pierwszej części pracy domowej dawał nam on wysokie wartości dokładności.



Wnioski (zadanie dodatkowe)

- dla k większych od 50 zaimplementowany algorytm radził sobie lepiej niż gotowe metody
- w obu przypadkach najwyższa dokładność została uzyskana dla zaimplementowanego algorytmu
- można przypuszczać, że po znalezieniu takiej wartości k, że knn() daje nam największą dokładność najbardziej korzystne będzie korzystanie z niego, a nie gotowych metod