Implementacja algorytmu regresji logistycznej

Wiktoria Jańska 259709, Maciej Ziulczyk 259685

Politechnika Wrocławska, Wrocław, Polska
259709@student.pwr.edu.pl
259685@student.pwr.edu.pl

Streszczenie W ramach projektu zaimplementowano algorytm regresji logistycznej oraz przeprowadzono porównanie z innymi klasyfikatorami. Celem badania było ocenienie wydajności i dokładności implementacji w porównaniu do innych metod klasyfikacji. Projekt rozpoczęto od implementacji metody regresji logistycznej, a następnie przygotowano zbiór danych, który został udostępniony na platformie Kaggle. Zbiór ten został odpowiednio przetworzony i podzielony na zbiór treningowy oraz testowy przy użyciu walidacji krzyżowej. Przeprowadzono również eksperymenty, takie jak selekcja cech czy szukanie suboptymalnej liczby iteracji, w celu osiągnięcia najlepszej możliwej dokładności. Za pomocą popularnych metryk oraz testów statystycznych oceniono model.

Keywords: Algorytm regresji logistycznej · Klasyfikacja · Implementacja · Analiza danych

1 Wstęp i przegląd literatury

W tym rozdziale wyjaśnimy podstawowe definicje i założenia dotyczące metody regresji logistycznej oraz przedstawimy metody referencyjne.

1.1 Podstawowe założenia

Tematem projektu jest algorytm regresji logistycznej - narzędzia statystycznego, którego celem jest modelowanie wyniku dwumianowego za pomocą jednej lub więcej zmiennych objaśniających. Zakłada ono, że dane mogą być sklasyfikowane poprzez prostą lub płaszczyznę n-wymiarową. Regresja logistyczna sprowadza się do wyliczenia wartości wielomianu pierwszego stopnia o następującej postaci:

$$y = w_1x_1 + w_2x_2 + ... + w_nx_n + b$$

gdzie x jest daną wejściową, w wagą przypisaną do tego parametru, b błędem algorytmu, a n liczbą parametrów wejściowych. Następnym krokiem jest przepuszczenia tak uzyskanego wyniku y przez funkcję logistyczną w celu uzyskania wartości z przedziału (0;1). Po uzyskaniu wartości możemy zaklasyfikować daną wejściową do grupy A lub grupy B.

1.2 Podstawowe definicje

Uczeniem nadzorowanym [11] nazywamy konstruowanie algorytmów, które są w stanie generalizować ogólne wzorce i hipotezy przy użyciu instancji dostarczonych z zewnątrz, w celu przewidzenia losu przyszłych instancji. Jednym z zadań uczenia nadzorowanego jest **klasyfikacja** [7, 9, 5], gdzie

dysponujemy obserwacjami z etykietami (klasami), które przyjmują wartości dyskretne. Celem uczenia jest skonstruowanie klasyfikatora separującego obiekty należące do różnych klas. Klasyfikator konstruowany jest tak, aby możliwe było przewidzenie klas nowych, niesklasyfikowanych obserwacji. Kolejnym z zadań jest **regresja** [7, 9, 5],w którym dysponujemy obserwacjami z odpowiadającymi im etykietami ciągłymi. Celem uczenia jest skontrowanie modelu regresji na podstawie danych. Model konstruowany jest tak, aby możliwe było przewidywanie nowych obserwacji. Klasyfikacja polega na przewidywaniu etykiety, natomiast regresja na przewidywaniu wartości ciągłych. Ważnym pojęciem przy okazji uczenia maszynowego i regresji logistcznej jest również **optymalizacja** [2] oraz **matoda gradientu prostego** [2]. Problem optymalizacji możemy zapisać jako

$$\begin{aligned} &minimize \quad f(x) \\ &subject\ to \quad x \in X \end{aligned}$$

gdzie **X** nazywamy zbiorem rozwiązań dopuszczalnych. Ponadto problem optymalizacji ciągłej to taki problem, w którym optymalizowane zmienne należą do zbioru liczb rzeczywistych, czyli $X \subseteq R^n$. Problem, w którym $X = R^n$ nazywamy problemem bez ograniczeń. **Metoda gradientu prostego** polega na wykonywaniu kroków o założonej długości w kierunku przeciwnym do gradientu funkcji celu w punkcie startowym danej iteracji. Jeżeli wartość funkcji celu w nowo wyznaczonym punkcie jest większa niż w punkcie początkowym, to znaczy, że właśnie wykonany krok był zbyt długi. W takim wypadku należy nowy punkt odrzucić i wykonać krok ponownie, tym razem ze zmniejszoną długością.

1.3 Metody referencyjne

Jedną z popularnych metod klasyfikacji jest **k-najbliższych sąsiadów (KNN)** [3]. Metoda ta polegająca na wybraniu liczby obserwacji (k) leżących najbliżej badanego obiektu oraz przypisanie go do klasy, która wynika z największej liczby pasujących sąsiadów. Kolejną często stosowanym klasyfikatorem jest **naiwny klasyfikator bayesowski** [8] Klasyfikatory Bayesowskie przypisują najbardziej prawdopodobną klasę do danego obiektu, opisanego przez swój wektor cech. Uczenie się takich klasyfikatorów można znacznie uprościć zakładając, że cechy są niezależne od danej klasy, tzn. $P(X|C) = \prod_{i=1}^{n} P(X_i|C)$, gdzie $X = (X_1, ..., X_n)$ jest wektorem cech, a C jest klasą. Warto również wspomnieć o metodzie klasyfikacji, jaką jest **maszyna wektorów nośnych (SVM)** [10], której łównym celem jest rozwiązanie wypukłego problemu optymalizacji w celu uzyskania globalnie optymalnego rozwiązania w teorii, a tym samym przezwyciężenie lokalnego dylematu ekstremalnego innych technik uczenia się maszyn. SVM w swojej podstawowej formie jest liniowym klasyfikatorem binarnym, który identyfikuje pojedynczą granicę między dwiema klasami. Liniowa SVM zakłada, że dane wielowymiarowe są liniowo separowane w przestrzeni wejściowej. Maszyny SVM określają optymalną hiperpłaszczyznę, aby podzielić zbiór danych na dyskretną liczbę predefiniowanych klas przy użyciu danych treningowych. Ostatnią metodą, na którą zwrócimy uwagę jest **decision tree(DT)** [6] Algorytm ten wielokrotnie dzieli dane, tworząc grupy lub klasy. Klasyfikator ten buduje ogromny zestaw danych i nie zatrzymuje się, dopóki nie osiągnie pożądanego celu.

2 Metoda

Jeśli przyjmiemy ważoną sumę danych wejściowych jako wynik, tak jak robimy to w regresji liniowej, wartość może być większa niż 1, natomiast nas interesuje wartość z przedziału od 0 do 1, stąd też na danych wyjściowych (ważona suma danych wejściowych) należy zastosować **funkcję aktywacji**, a dokładniej **funkcję sigmoidalną**, której postać wygląda następująco:

$$\frac{1}{1+e^{-wx+b}} \tag{1}$$

Następnie należy wyznaczyć **gradient** dla funkcji wynikowej. Robimy to obliczając pochodną dla funkcji straty (cost function), która używana jest do zmierzenia, jak bardzo prognoza różni się od etykiet

$$J(w,b) = J(\theta) = \frac{1}{N}\sum_{i=1}^{n}[y^i log(h_\theta(x^i)) + (1-y^i)log(1-h_\theta(x^i)] \tag{2}$$

$$J'(\theta) = \begin{bmatrix} \frac{dJ}{dw} \\ \frac{dJ}{dw} \end{bmatrix} = [...] = \begin{bmatrix} \frac{1}{N}(\hat{y}-y_i)x_i \\ \frac{1}{N}(\hat{y}-y_i) \end{bmatrix} \tag{3}$$

Kolejnym krokiem jest aktualizacja parametrów w oraz b (wagi i błędu)

$$w = w - \alpha * dw \tag{4}$$

$$b = b - \alpha * dw \tag{5}$$

gdzie α to inaczej *learning rate*, czyli parametr który określa wielkość kroku w każdej iteracji, przesuwając się w kierunku minimum funkcji straty.

3 Projekt eksperymentów

W tym rozdziale skupimy się na opisie zbioru danych, na którym będziemy trenować nasz model. Przedstawimy zmienne objaśniające oraz użyte przez nas technologie, metody trenowania i oceny modelu oraz testy statystyczne. Przedstawimy również zaplanowane eskperymenty, które mają na celu uzyskanie jak najlepszych wyników dokładności osiąganych przez nasz model.

3.1 Opis zbioru danych

3.1.1 Celem projektu jest implementacją metody klasyfikacji, bazującej na regresji logistycznej, która jest w stanie przewidzieć, którzy pasażerowie przeżyli katastrofę Titanica. Na potrzebę projektu posłużymy się gotowym zestawem danych, udostępnionym na platformie **Kaggle**. Wybrany zbiór danych zawiera wiele cech, stąd też w naszej pracy posłużymy się **selekcją cech**, która to wybierze te, które mają największy wpływ na wynik klasyfikacji.

Tabela 1: Przykładowe dane z pliku train.csv

PassId	Sur	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Emb
1	0	3	"Braund, Mr. Owen Harris"	male	22	1	0	A/5 21171	7.25		S
2	1	1	"Kink, Mr. Vincenz"	female	38	1	0	PC 17599	71.2833	C85	C
3	1	3	"Heikkinen, Miss. Laina"	female	26	0	0	STON/O2. 3101282	7.925		S
4	1	1	"Kraeff, Mr. Theodor"	female	35	1	0	113803	53.1	C123	S
5	0	3	"Allen, Mr. William Henry"	male	35	0	0	373450	8.05		S
6	0	3	"Moran, Mr. James"	male		0	0	330877	8.4583		Q
7	0	1	"McCarthy, Mr. Timothy J"	male	54	0	0	17463	51.8625	E46	S
8	0	3	"Palsson, Master. Gosta Leonard"	male	2	3	1	349909	21.075		S
9	1	3	"Bing, Mr. Lee"	female	27	0	2	347742	11.1333		S
10	1	2	"Nasser, Mrs. Nicholas (Adele Achem)"	female	14	1	0	237736	30.0708		C
11	1	3	"Sandstrom, Miss. Marguerite Rut"	female	4	1	1	PP 9549	16.7	G6	S
12	1	1	"Bonnell, Miss. Elizabeth"	female	58	0	0	113783	26.55	C103	S
13	0	3	"Saundercock, Mr. William Henry"	male	20	0	0	A/5. 2151	8.05		S
14	0	3	"Andersson, Mr. Anders Johan"	male	39	1	5	347082	31.275		S
15	0	3	"Rugg, Miss. Emily"	female	14	0	0	350406	7.8542		S

3.1.2 Zmienne objaśniające

są to zmienne, które służą nam do predykcji zmiennych objaśnianych (zależnych). W naszym projekcie są to następujące atrybuty:

PassId - Passenger ID, numer identyfikacyjny danego pasażera
Survived (Sur) - określa, czy dany pasażer przeżył (0 = nie, 1 = tak)
Pclass - Passenger Class, klasa biletu pasażera (1 = 1st, 2 = 2nd, 3 = 3rd)
Name - imię i nazwisko pasażera
Sex - płeć pasażera
Age - wiek pasażera
SibSp - rodzeństwo/małżonek/małżonka na pokładzie
Parch - rodzice/dzieci na pokładzie
Ticker - numer biletu
Fare - opłata za rejs
Cabin - numer kabiny
Emb - port, na którym pasażer wszedł na pokład; C = Cherbourg, Q = Queenstown, S = Southampton

Niektóre dane są brakujące, jak np. wiek pasażera o ID = 6. W takim przypadku wykorzystujemy 'SimpleImputer', aby brakujące dane uzupełnić średnią wartością dla wszystkich danych w tej kategorii.

3.2 Wykorzystane technologie i biblioteki

Projekt został zaimplementowany w języku Python przy użyciu środowiska PyCharm. Wykorzystane zostały poniższe biblioteki:

- **numpy** - biblioteka programistyczna posiadająca obsługę wielowymiarowych tabel, macierzy oraz zaawansowanych funkcji matematycznych do ich obsługi

- **scipy** - rozszerzenie biblioteki **numpy** dostarczająca dodatkowcy narzędzi i funkcji do obliczeń naukowych
- **matplotlib** - wszechstronna biblioteka programistyczna do tworzenia statycznych, animowanych i interaktywnych wizualizacji
- **scikit-learn** - biblioteka programistyczna służąca do obsługi uczenia maszynowego. Elementami tej biblioteki są algorytmy klasyfikacji, regresji i klastrowania.
- **pandas** - biblioteka programistyczna służąca do manipulacji oraz analizy danych

3.3 Metoda trenowania i oceny modelu

Walidacja krzyżowa [1] - W naszym projekcie w celu trenowania i oceny modelu wykorzystujemy protokół eksperymentalny, walidację krzyżową, czyli jedną z najczęściej stosowanych metod podziału danych, w celu oszacowania rzeczywistego błędu przewidywania modeli i dostrojenia parametrów modelu. Zbiór danych dzielimy na dwa podzbiory - dane treningowe oraz dane testowe. Dane treningowe stanowią 50% wszystkich danych ze zbioru, natomiast pozostałe 50% stanowią dane testowe. Nasz model predykcji będzie się uczył, używając do tego celu danych treningowych, a efekty tego nauczania widoczne będą podczas analizy danych testowych. Procedura zostanie wykonana przy użyciu 5 powtórzeń oraz 2 foldów.

3.4 Wykorzystane testy statystyczne

Test t Studenta [4] - test statystyczny, do którego wykorzystywany jest rozkład **t**. Podstawowym test polega na sprawdzeniu hipotezy dotyczącej wartości oczekiwanej z rozkładu normalnego, gdy nie znamy jego wariancji.

Model:
$(X_1, ..., X_n)$ jest próbą losową, w której cecha X ma rozkład $N(\mu, \sigma^2)$ oraz nieznana jest wariancja σ^2.

Hipoteza:
$H_0 : \mu = \mu_0$
$H_1 : \mu \neq \mu_0$

Statystyka:

$$T = \frac{\sqrt{n}(\overline{X} - \mu_0)}{S} \tag{6}$$

, gdzie:

$\overline{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$
$S = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(X_i - \overline{X})^2}$

W momencie kiedy H_0 jest prawdziwa, to T ma rozkład Studenta $T(n-1)$

Obszar krytyczny:
$K = \{t :| t |> t_{n-1}(1 - \alpha/2)\}$, gdzie t jest wartością statystyki T

3.5 Metryki oceny wyników

- **Dokładność (accuracy)** - liczba prawidłowych predykcji w stosunku do wszystkich predykcji
  $accuracy = correctPredictions/totalPredictions$

  True Positive (TP) - pozytywna predykcja dla klasy pozytywnej
  True Negative (TN) - pozytywna predykcja dla klasy negatywnej
  False Positive (FP) - negatywna predykcja dla klasy pozytywnej
  False Negative (FN) - negatywna predykcja dla klasy pozytywnej
  $Accurancy = (TP + TN)/(TP + TN + FP + FN)$

- **Precyzja (precision)** - liczba prawidłowych predykcji dla klasy pozytywnej
  $Precision = TP/(TP + FP)$

- **Czułość (recall)** - liczba prawidłowych predykcji dla klasy negatywnej
  $Recall = TP/(TP + FN)$

- **Krzywe ROC (ROC Curve)** - wykres graficzny przedstawiający zdolność diagnostyczną klasyfikatora binarnego

- **F1 Score** - średnia harmoniczna precyzji oraz czułości.

3.6 Zaplanowane eksperymenty

W celu uzyskania jak najlepszej dokładności naszego klasyfikatora przeprowadzone zostały eksperymenty, takie jak znalezienie optymalnej liczby iteracji, selekcja cech oraz porównanie z metodami referencyjnymi. Celem eksperymentu dotyczącego **liczby iteracji** jest określenie jaka liczba iteracji będzie optymalna, by uzyskać jak najlepsze wyniki metryk dla naszego modelu. **Selekcja cech** polega na identyfikacji istotnych cech i eliminacji tych, które nie mają wpływu na wyniki modelu, bądź negatywnie na nie wpływają. Ostatnim eksperymentem jest **porównanie naszej metody regresji logistycznej z innymi metodami klasyfikacji**, którego celem jest stwierdzenie ogólnej oceny naszego klasyfikatora, porównując jego wyniki z wynikami innych metod.

4 Wyniki eksperymentów

W poniższym rozdziale skupimy się na przeprowadzonych eksperymentach oraz ich wynikach. Każdy z wyników eksperymentu jest zapisywany do pliku o odpowiednim rozszerzeniu. Pliku są następnie wczytywane, aby dokonać analizy.

4.1 Liczba iteracji

Jednym z eksperymentów, na które się zdecydowaliśmy, było znalezienie suboptymalnej liczby iteracji, by uzyskać zadowalające wyniki metryk. Przetestowane zostały tylko 4 wartości: [100, 500, 1000, 5000], gdyż wyniki były do siebie bardzo zbliżone i większa ilość iteracji byłaby zbędna.

Tabela 2: Dokładność klasyfikatora w zależności od liczby iteracji

n_iters	dokładność
100	0.780
500	0.780
1000	0.780
5000	0.784

Wyniki dla każdej iteracji są do siebie bardzo zbliżone. Największą dokładność osiągamy dla liczby iteracji równej 5000, jednak przy tak dużej liczbie iteracji nasz program wykonuje się stosunkowo długo, stąd też zdecydowaliśmy się na wybór liczby iteracji równej 100. Jest to najwydajniejsze rozwiązanie, przy którym zachowamy podobną dokładność klasyfikatora.

4.2 Selekcja cech

Kolejnym z eksperymentów, na które się zdecydowaliśmy, była selekcja cech, która miała za zadanie wybór cech, które mają największy wpływ na zmienną wynikową. W eksperymencie zbadaliśmy dokładność regresji logistycznej w zależności od ilości tychże cech. Przetestowane zostały wartości (ilości cech): [5, 10, 15, 20].

Tabela 3: Dokładność klasyfikatora w zależności od ilości wybranych cech przez selekcję cech

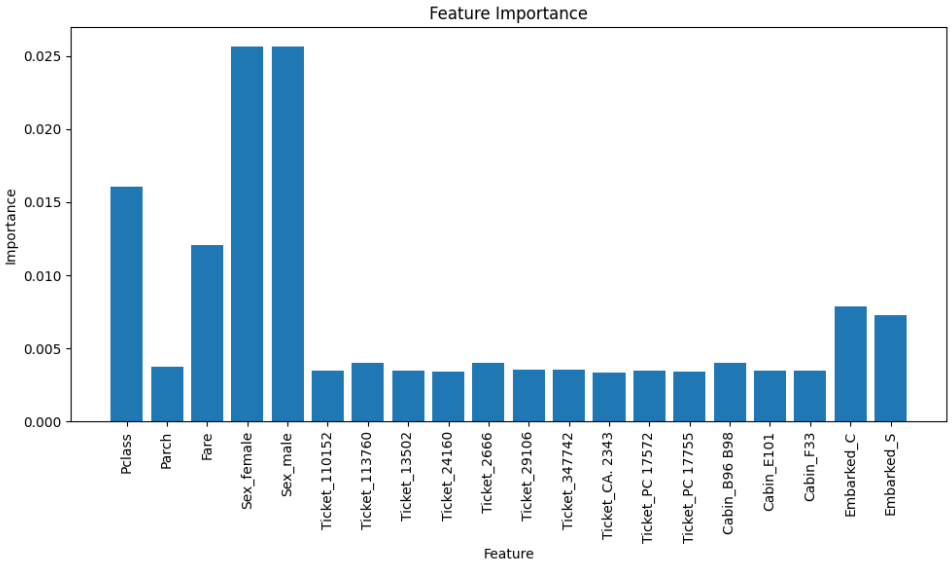
n_features	dokładność
5	0.778
10	0.777
15	0.778
20	0.780

Z wyników można łatwo wywnioskować, że przy liczbie cech równej 20, dokładność klasyfikatora jest największa, więc zdecydowaliśmy się na wykorzystanie tych 20 cech. Eksperyment wyszczególnił poniższe cechy, jako te najbardziej wpływowe:

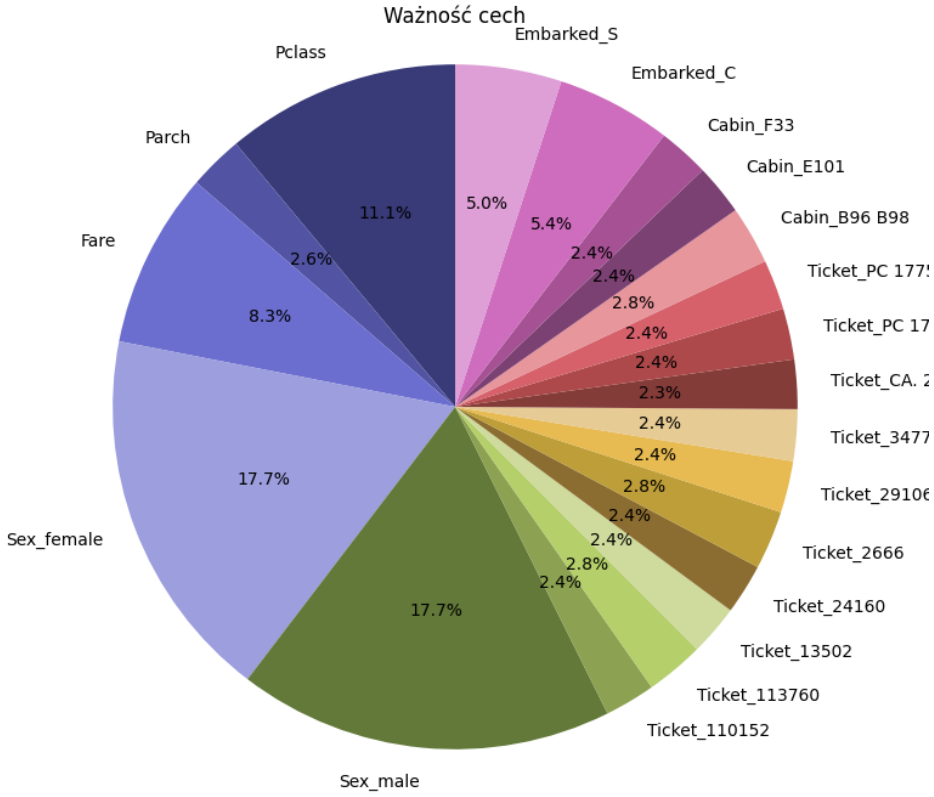
- Sex_female
- Sex_male
- Pclass
- Fare
- Embarked_C
- Embarked_S
- Parch
- Ticket_110152
- Ticket_113760
- Ticket_13502
- Ticket_24160

- Ticket_2666
- Ticket_29106
- Ticket_347742
- Ticket_CA. 2343
- Ticket_PC 17572
- Ticket_PC 17755
- Cabin_B96 B98
- Cabin_E101
- Cabin_F33

Cechy, takie jak 'Ticket' czy 'Cabin' zostały wyodrębnione na poszczególne kategorie ze względu na użycie 'pd.get_dummies', która to metoda przetwarza kolumny zawierające kategorie na zestaw kolumn zawierających wartości binarne (0 lub 1). Tworzy ona tzw. kodowanie "one-hot", w którym dla każdej kategorii tworzy się osobną kolumnę. Nasz model wykorzystał te dwadzieścia cech w celu predykcji zmiennej wynikowej.



Rysunek 1: Wykres słupkowy przedstawiający ważność poszczególnych cech na przeżycie katastrofy Titanica przez jednostkę



Rysunek 2: Wykres kołowy przedstawiający ważność poszczególnych cech na przeżycie katastrofy Titanica przez jednostkę

Z powyższych wykresów możemy wywnioskować, że cechą, która ma największe znaczenie dla przeżycia katastrofy Titanica jest płeć, następnie klasa biletu pasażera oaz opłata za rejs. My zdecydowaliśmy się na wykorzystanie wszystkich wyżej wymienionych cech w celu predykcji, aby zachować jak najlepszą dokładność.

4.3 Porównanie z innymi metodami klasyfikacji

Ważnym aspektem przy ocenie naszego modelu (**LR**) jest porównanie go z innymi klasyfikatorami. Zdecydowaliśmy się na porównanie z następującymi metodami klasyfikacji: K-najbliższych sąsiadów **(kNN)**, maszyna wektorów nośnych **(SVM)**, drzewem decyzyjnym **(DCT)** oraz naiwny klasyfikator bayesowski **(GNB)**.

Tabela 4: Porównanie dokładności modelu regresji logistycznej z innymi klasyfikatorami

classifier	mean score	standard deviation
LR	0.780	0.013
kNN	0.799	0.018
DTC	0.777	0.017
SVM	0.810	0.015
GNB	0.648	0.005

Patrząc na powyższą tabelę wyników możemy przypuszczać, że najmniej skutecznym klasyfikatorem dla naszego zbioru danych jest naiwny klasyfikator bayesowski, gdyż jego dokładność wynosi zaledwie około 65%, natomiast pozostałe klasyfikatory plasują się na podobnym poziomie, tzn. w okolicach 70-80%.

4.4 Ocena modelu

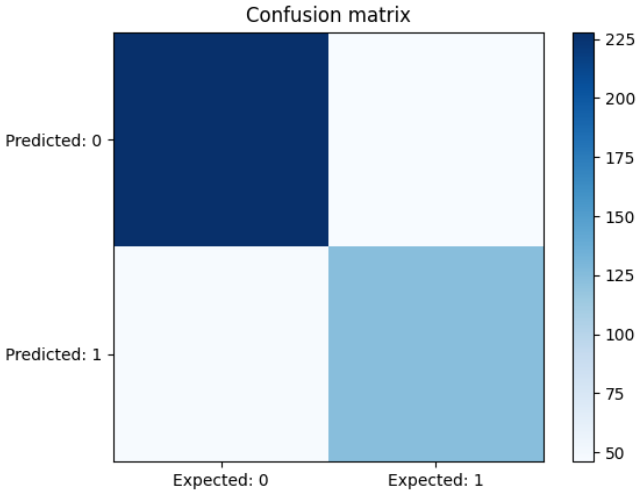
Nasz model oceniliśmy posługując się takimi narzędziami jak **macierz pomyłek (confussion matrix)**, która to pokazuje ilość błędnie i poprawnie sklasyfikowanych danych, a także przy pomocy **raportu klasyfikacji (classification report)**, który tworzy zestawienie metryk w celu oceny poprawności i dokładności danego modelu. Zawiera takie metryki jak Accuracy, Precision, Recall i F1-score dla każdej klasy oraz średnią ważoną tych miar dla wszystkich klas.

Jako że w naszym projekcie wykorzystujemy walidację krzyżową, która dzieli zbiór danych (891 próbek) na dwa podzbiory (446 i 445 próbek), poniższa macierz pomyłek oraz raport klasyfikacji jest wykonany tylko dla ostatniego foldu, stąd też wartości takie jak na przykład "accuracy", nieznacznie różni się od całkowitej średniej dokładności modelu.

Macierz pomyłek dla naszej implementacji regresji logistycznej dla ostatniego foldu (10) wygląda następująco:

$$M = \begin{bmatrix} 228 & 46 \\ 47 & 124 \end{bmatrix} \tag{7}$$

Pierwszy rząd macierzy pomyłek mówi nam o tym, że 228 próbek negatywnych zostało poprawnie sklasyfikowanych jako próbki negatywne (True Negative), natomiast 46 próbek negatywnych zostało niepoprawnie sklasyfikowanych jako próbki pozytywne (False Positive).
Drugi rząd macierzy pomyłek mówi nam o tym, że 47 próbek pozytywnych zostało niepoprawnie sklasyfikowanych jako próbki negatywne (False Negative), natomiast 124 próbek pozytywnych zostało poprawnie sklasyfikowanych jako próbki pozytywne (True Positive).
Na podstawie macierzy pomyłek widzimy, że estymator poprawnie sklasyfikował 352 próbek, a 93 zostało źle sklasyfikowanych.



Rysunek 3: Macierz pomyłek

Raport klasyfikacji dla naszej implementacji regresji logistycznej dla foldu 10 wygląda następująco:

Tabela 5: Raport klasyfikacji dla modelu regresji logistycznej

0/1	precision	recall	f1-score	support
0	0.83	0.83	0.83	274
1	0.73	0.73	0.73	171
accuracy	-	-	0.79	445
macro avg	0.78	0.78	0.78	445
weighted avg	0.79	0.79	0.79	445

Z powyższej tabeli wyników możemy wyciągnąć kilka wniosków:

- Dokładność naszego modelu (accuracy) wynosi 0.79, co oznacza, że 79% wszystkich próbek zostało poprawnie sklasyfikowanych.
- Precyzja (precision) dla klasy 1 wynosi 0.73, co oznacza, że 73% próbek sklasyfikowanych jako klasa 1 została poprawnie przewidziana. Wyższa precyzja oznacza mniejszą liczbę fałszywie przypisanych przypadków do danej klasy.
- Czułość (recall) dla klasy 1 wynosi 0.73, co oznacza, że 73% próbek klasy 1 zostało poprawnie zidentyfikowanych. Wyższa czułość oznacza mniejszą liczbę fałszywie odrzuconych przypadków danej klasy.
- Miara F1 (f1-score) dla klasy 1 wynosi 0.73, co jest średnią harmoniczną precyzji i czułości.
- Wyniki dla klasy 0 są nieco wyższe - precyzja wynosi 0.83, czułość wynosi 0.83, natomiast miara F1 wynosi 0.83.
- Support wskazuje liczbę obserwacji należących do każdej klasy w zbiorze danych testowych. Oznacza to, że dla klasy 0 w zbiorze testowym znajduje się 274 obserwacji, natomiast dla klasy 1 jest to 171.

Dla każdego foldu wartości te są do siebie podobne, stąd też z naszych wyników można wywnioskować, iż nasz model osiąga przyzwoite wyniki klasyfikacji, ale jest lepszy przy rozpoznawaniu obiektów klasy 0, niż klasy 1.

Średnie wartości precyzji i czułości wyglądają następująco:

Tabela 6: Średnie wartości precyzji i czułości dla klasy 0 i klasy 1

class	mean precision	mean recall
0	0.83	0.81
1	0.71	0.73

Oprócz powyższych narzędzi wykorzystaliśmy również **test t-studenta**, aby przeprowadzić porównanie pomiędzy naszym modelem regresji logistycznej a innymi metodami klasyfikacji. Test ten pomaga określić, czy istnieje statystycznie istotna różnica w wynikach między tymi modelami.

Tabela 7: Wyniki po przeprowadzeniu testu t-studenta

classifier	LR	kNN	DCT	SVM	GNB
LR	False	False	False	True	False
kNN	False	False	False	True	False
DCT	True	True	False	True	True
SVM	False	False	False	False	False
GNB	True	True	False	True	False

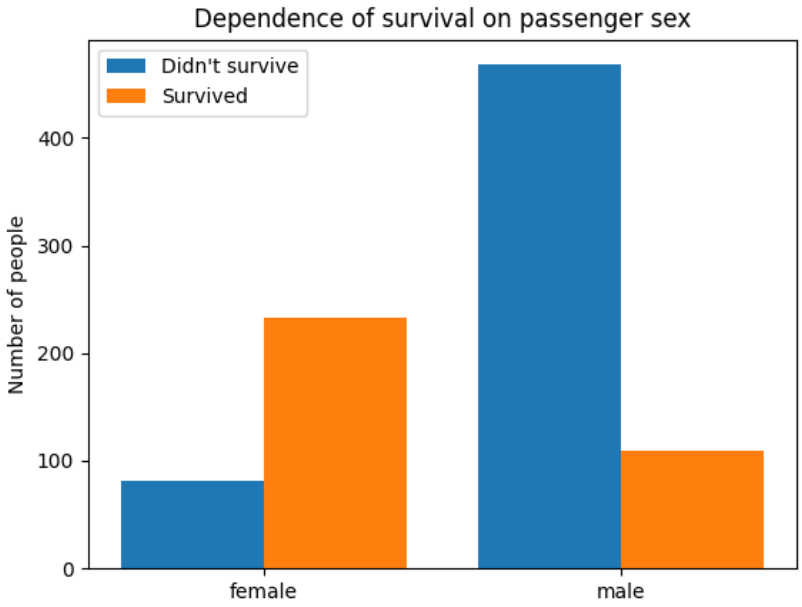
Rysunek 4: Dokładności dla różnych klasyfikatorów z uwzględnieniem testu t-studenta

classifier	mean score	standard deviation
LR (1)	0.780 3,5	0.013
kNN (2)	0.799 1,3,5	0.018
DTC (3)	0.777 5	0.017
SVM (4)	0.810 all	0.015
GNB (5)	0.648 -	0.005

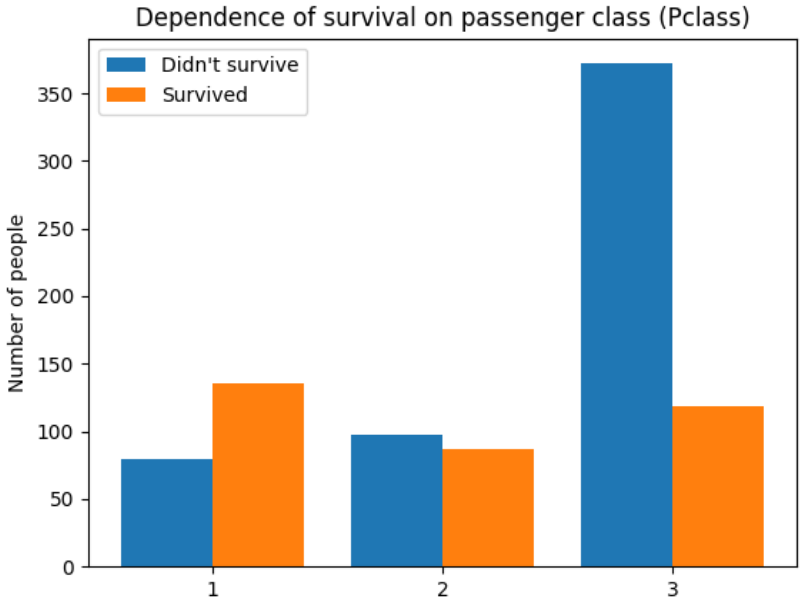
Z powyższej tabeli możemy wysnuć kilka wniosków:

- Klasyfikator LR (acc: 0.780) jest lepszy od klasyfikatora DCT (acc: 0.779) oraz GNB (acc: 0.648), ale jest gorszy od SVM (acc: 0.810) i kNN (acc: 0.799).
- Klasyfikator kNN (acc: 0.799) jest lepszy od klasyfikatora LR (acc: 0.780), DCT (acc: 0.779) oraz GBN (acc: 0.648), ale jest gorszy od SVM (acc: 0.810)
- Klasyfikator DCT (acc: 0.777) jest lepszy od klasyfikatora GNB (acc: 0.648) oraz gorszy od każdego z pozostałych klasyfikatorów
- Klasyfikator SVM (acc: 0.810) jest lepszy od każdego z pozostałych klasyfikatorów
- Klasyfikator GNB (acc: 0.648) jest gorszy od każdego z pozostałych klasyfikatorów

4.5 Graficzne przedstawienie wyników metody regresji logistycznej na naszym zbiorze danych

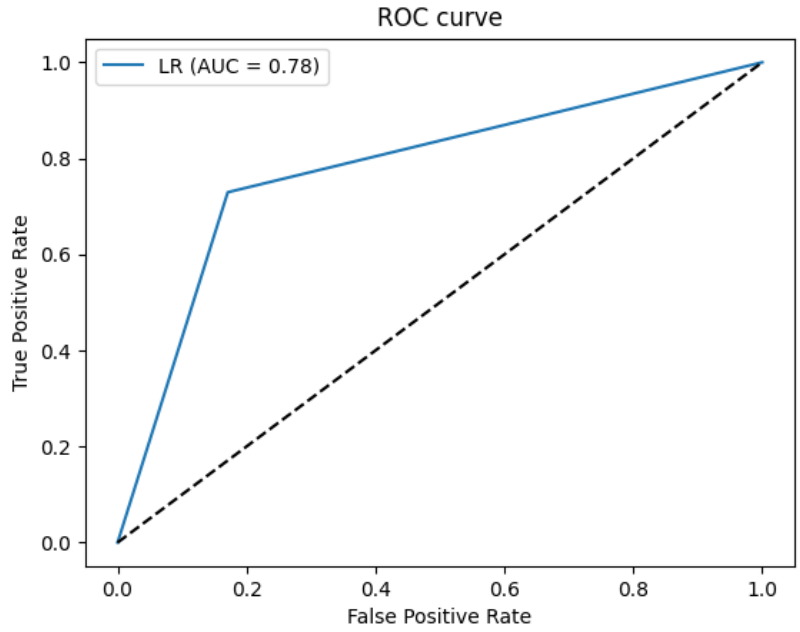


Rysunek 5: Zależność przeżycia katastrofy Titanica od płci pasażera (Sex)



Rysunek 6: Zależność przeżycia katastrofy Titanica od klasy biletu pasażera (Pclass)

Na wykresie przeżywalności w zależności od płci pasażera (Rys.4) możemy zauważyć sporą przewagę przeżywalności jednostki, jeśli jest ona kobietą. Liczba kobiet, które nie przeżyły jest bliska liczbie mężczyzn, którzy przeżyli. Na wykresie przeżywalności w zależności od klasy biletu pasażera (Rys.5) możemy zauważyć, że największa śmiertelność jest obecna dla klasy 3, natomiast najmniejsza śmiertelność obecna jest dla klasy 1. W klasie drugiej przeżywalność i śmiertelność plasują się na podobnym poziomie.



Rysunek 7: Krzywa ROC

AUC dla naszego modelu wynosi 0.78, co zgadza się z miarami wydajności uzyskanymi wcześniej przy użyciu innych metryk (np. accuracy = 0.780). Krzywa ROC dla idealnej metody klasyfikacji binarnej osiąga punkt (0, 1), co oznacza zerową wartość FPR (False Positive Rate) oraz jednocześnie jednostkową wartość TPR (True Positive Rate). W naszym przypadku krzywa ROC osiąga punkt (0.190, 0.725). Krzywa ROC, tak jak nazwa wskazuje, wyglądem powinna przypominać krzywą. W naszym przypadku jest to prosta, która załamuje się w wyżej wymienionym punkcie, ze względu na to, że w naszym projekcie krzywa ROC została wykonana na etykietach, a nie na wsparciach.

5 Wnioski

Nasza implementacja klasyfikatora regresji logistycznej w porównaniu do innych klasyfikatorów osiąga podobne wyniki. Dokładność modelu (accuracy) wynosi 0.780, co jest porównywalne ze średnim wynikiem dokładności dla pozostałych klasyfikatorów. Osiągane dokładności są zadowalające

i wskazują, że model może być użyteczny w praktycznych zastosowaniach. Na podstawie tych informacji możemy śmiało stwierdzić, że nasz model jest dobrym wyborem metody klasyfikacji dla użytego zbioru danych.

Należy jednak pamiętać, że model ten jest odpowiedni w przypadku tego konkretnego zbioru danych. Skuteczność klasyfikacji może różnić się przy podstawieniu innego zbioru danych, ze względu na specyfikę problemu. Aby potwierdzić skuteczność modelu należy przeprowadzić dalsze badania i eksperymenty.

Klasyfikator regresji logistycznej oprócz zadowalających wyników i swojej skuteczności jest również atrakcyjny ze względu na swoją prostotę implementacji i interpretowalność wyników, stąd też w przypadkach, gdzie istotne jest zrozumienie czynników decydujących o klasyfikacji, może być on chętnie stosowany.

Bibliografia

[1] D. Berrar. Cross-validation., 2019.

[2] S. Boyd, S. P. Boyd, and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[3] O. Kramer and O. Kramer. K-nearest neighbors. *Dimensionality reduction with unsupervised nearest neighbors*, pages 13–23, 2013.

[4] R. Magiera. *Modele i metody statystyki matematycznej*. Oficyna Wydawnicza GiS, 2002.

[5] V. Maini and S. Sabri. Machine learning for humans. *Online: https://medium. com/machine-learning-for-humans*, 2017.

[6] M. M. Mijwil and R. A. Abttan. Utilizing the genetic algorithm to pruning the c4. 5 decision tree algorithm. *Asian Journal of Applied Sciences*, 9(1), 2021.

[7] T. M. Mitchell and M. Learning. Mcgraw-hill science. *Engineering/Math*, 1:27, 1997.

[8] I. Rish et al. An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, volume 3, pages 41–46, 2001.

[9] S. Shalev-Shwartz and S. Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[10] M. Sheykhmousa, M. Mahdianpari, H. Ghanbari, F. Mohammadimanesh, P. Ghamisi, and S. Homayouni. Support vector machine versus random forest for remote sensing image classification: A meta-analysis and systematic review. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 13:6308–6325, 2020.

[11] A. Singh, N. Thakur, and A. Sharma. A review of supervised machine learning algorithms. In *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, pages 1310–1315. Ieee, 2016.