Politechnika Śląska

Wydział Automatyki, Elektroniki I Informatyki

TECHNOLOGIE SIECIOWE

PROJEKT

Przewidywanie cukrzycy na podstawie danych medycznych.

Autorzy	Szczepańczyk Katarzyna Gajczak Kamil

Academic year	2023/2024
Kierunek	Informatyka
Stopień	S2
Semestr	1
Prowadząca	dr inż. Monika Nycz

---

1 KRÓTKI WSTĘP I WPROWADZENIE DO TEMATU

Cukrzyca to powszechna choroba przewlekła dotykająca milionów ludzi na całym świecie. Wczesne wykrywanie i dokładne prognozowanie cukrzycy są kluczowe dla zapobiegania powikłaniom i zapewnienia terminowej interwencji. Dzięki szybkiemu postępowi w dziedzinie sztucznej inteligencji i uczenia maszynowego, naukowcy zaczęli wykorzystywać moc obliczeniową sieci neuronowych do tworzenia modeli predykcyjnych opartych na danych medycznych. Celem tego projektu jest wykorzystanie obszernego zestawu danych obejmującego różne cechy pacjentów, takie jak liczba ciąż, poziom glukozy, ciśnienie krwi, grubość skóry, poziom insuliny, wskaźnik masy ciała (BMI), funkcja dziedziczenia cukrzycy, wiek oraz obecność lub brak cukrzycy, w celu zbudowania efektywnego modelu predykcyjnego z wykorzystaniem sieci neuronowych.

Głównym celem projektu jest opracowanie solidnego modelu predykcji cukrzycy przy użyciu sieci neuronowych. Analizując dane medyczne, model będzie dążył do identyfikacji osób o wysokim ryzyku zachorowania na cukrzycę. Terminowe wykrycie osób zagrożonych pozwala na wdrożenie środków zapobiegawczych, takich jak zalecenie odpowiednich zmian stylu życia i opracowanie spersonalizowanych planów leczenia, co ostatecznie zmniejsza wpływ powikłań związanych z cukrzycą.

Projekt skupia się również na optymalizacji wydajności modelu poprzez dostrajanie hiperparametrów, wdrażanie technik regularyzacji oraz eksplorację metod wyboru cech. Te kroki zapewnią, że opracowany model predykcji cukrzycy jest dokładny i niezawodny. Integracja takiego modelu w praktyce klinicznej może znacznie poprawić proces podejmowania decyzji przez pracowników służby zdrowia, prowadząc do lepszych wyników leczenia pacjentów i redukcji powikłań związanych z cukrzycą.

2 ANALIZA ZADANIA

POTENCJALNE PODEJŚCIA DO ROZWIĄZANIA PROBLEMU

Zadanie polega na przewidywaniu obecności lub braku cukrzycy na podstawie cech pacjentów. Zbiór danych zawiera informacje takie jak liczba ciąż, poziom glukozy, ciśnienie krwi, grubość skóry, poziom insuliny, BMI, funkcję rodowodu cukrzycowego i wiek. Jest to problem klasyfikacji binarnej, w którym model musi nauczyć się wzorców i zależności między tymi atrybutami a wynikiem, aby dokładnie przewidzieć obecność lub brak cukrzycy.

Podejścia do rozwiązania problemu

- **Feedforward Neural Networks (FNNs)**: Użycie wielowarstwowych perceptronów (MLP), które składają się z wielu warstw połączonych węzłów, stosujących nieliniowe funkcje aktywacji do sumy ważonej swoich wejść. Wymagają starannego rozważenia architektury sieci, w tym liczby warstw ukrytych, liczby węzłów na warstwę i wyboru funkcji aktywacji.
- **Rekurencyjne Sieci Neuronowe (RNNs)**: Efektywne w przetwarzaniu danych sekwencyjnych, sieci RNN utrzymują wewnętrzną pamięć, co pozwala im uchwycić zależności czasowe między wejściami. LSTM i GRU są popularnymi wariantami RNNs, które mogą obsługiwać długoterminowe zależności i łagodzić problem zanikającego gradientu.

- **Splotowe Sieci Neuronowe (CNN)**: Choć pierwotnie zaprojektowane do analizy obrazów, sieci CNN mogą być adaptowane do danych tabelarycznych, takich jak zbiór danych dotyczący cukrzycy. Stosując konwolucje jednowymiarowe, sieci CNN mogą wyodrębniać istotne cechy z wejść atrybutów pacjentów.
- **Metody Zespołowe**: Metody zespołowe, takie jak Random Forests lub Gradient Boosting, mogą być używane do łączenia wielu modeli dla poprawy wydajności predykcji. Te metody wykorzystują wielu słabych uczących, aby utworzyć silniejszy model ogólny.
- **Selekcja i inżynieria cech**: Przed treningiem modelu ważne jest wykonanie selekcji i inżynierii cech w celu identyfikacji najbardziej znaczących atrybutów i potencjalnych nowych cech. Techniki selekcji cech, takie jak analiza korelacji czy rekurencyjna eliminacja cech, mogą pomóc w identyfikacji atrybutów o największym wpływie na zadanie predykcji cukrzycy.
- **Regularyzacja i strojenie hiperparametrów**: Techniki regularyzacji, takie jak regularyzacja L1 lub L2, mogą pomóc w zapobieganiu nadmiernemu dopasowaniu i poprawie generalizacji modeli. Strojenie hiperparametrów jest również kluczowe dla optymalizacji wydajności modelu, a techniki takie jak grid search lub random search mogą być używane do systematycznego eksplorowania różnych kombinacji hiperparametrów.

Eksploracja różnych podejść, ich kombinacja oraz dostrajanie parametrów modeli pozwala na rozwinięcie dokładnego systemu predykcji cukrzycy. Wybór najodpowiedniejszego podejścia zależy od charakterystyki zbioru danych, zasobów obliczeniowych oraz pożądanej interpretowalności modelu.

SZCZEGÓŁOWY OPIS WYBRANYCH METOD

Feedforward Neural Network

Feedforward Neural Network (FNN), znana również jako multilayer perceptron (MLP), to rodzaj sztucznej sieci neuronowej szeroko stosowanej do różnych zadań uczenia maszynowego, w tym klasyfikacji i regresji. Nazywa się "feedforward", ponieważ przepływ informacji przez sieć odbywa się w jednym kierunku, od warstwy wejściowej do warstwy wyjściowej, bez żadnych pętli, czy połączeń zwrotnych.

Inżynieria Cech

Kod wprowadza inżynierię cech, tworząc dwie dodatkowe cechy interakcji, mianowicie interaction_1 i interaction_2, które są obliczane przez mnożenie wybranych cech wejściowych. Ten krok inżynierii cech ma na celu uchwycenie potencjalnych interakcji między atrybutami wejściowymi i potencjalne poprawienie wydajności predykcyjnej modelu.

PRZYGOTOWANIE DANYCH

Ładowanie danych Kod używa funkcji np.loadtxt, aby załadować dane z pliku CSV o nazwie "diabetes_2.csv". Dane są ładowane do tablicy NumPy o nazwie dataset.

- **Wizualizacja danych:** Kod zawiera krok wizualizacji danych w celu uzyskania wglądu w rozkład zmiennej wynikowej (cukrzyca lub brak cukrzycy). Używa matplotlib.pyplot do stworzenia wykresu słupkowego przedstawiającego liczbę wystąpień dla każdej kategorii wyników. Ta wizualizacja pomaga zrozumieć rozkład zbioru danych i częstość występowania przypadków cukrzycy.
- **Analiza korelacji:** Kod oblicza macierz korelacji (corr_matrix) za pomocą np.corrcoef, aby zmierzyć korelację par między atrybutami wejściowymi w zbiorze danych. Następnie używa seaborn i matplotlib.pyplot do stworzenia wizualizacji heatmap macierzy korelacji. Ten krok pomaga zidentyfikować istotne korelacje między atrybutami wejściowymi, co może dać pogląd w relacje w zbiorze danych.
- **Podział danych**: Kod dzieli zbiór danych na cechy wejściowe (X) i odpowiadającą im zmienną docelową (Y). Cechy wejściowe (X) są wyodrębniane z tablicy dataset za pomocą slicing, a zmienna docelowa (Y) jest wyodrębniana jako ostatnia kolumna tablicy dataset.
- **Inżynieria cech**: Kod wykonuje inżynierię cech, tworząc dwiedodatkowe cechy interakcji: interaction_1 i interaction_2. Te cechy interakcji są generowane przez mnożenie wybranych cech wejściowych. Wynikające cechy interakcji są następnie dołączane do macierzy cech wejściowych (X) za pomocą np.column_stack. Ten krok ma na celu uchwycenie potencjalnych interakcji między istniejącymi atrybutami wejściowymi, co może poprawić wydajność predykcyjną modelu.
- **Normalizacja danych:** Kod stosuje normalizację danych do cech wejściowych (X) używając normalizacji z-score. Oblicza średnią i odchylenie standardowe cech wejściowych, a następnie kalibruje cechy, odejmując średnią i dzieląc przez odchylenie standardowe. Ten krok normalizacji zapewnia, że wszystkie cechy wejściowe są w podobnej skali, zapobiegając dominacji niektórych cech w procesie uczenia.
- **Podział trening-test**: Kod dzieli przetworzone dane na zestawy treningowe i testowe. Używa zmiennej split_index do określenia indeksu, przy którym podzielić dane. Pierwsza połowa przetworzonych danych jest przypisana do zestawu treningowego (X_train i

y_train), podczas gdy druga połowa jest przypisana do zestawu testowego (X_test i y_test). Ten podział pozwala na ocenę wydajności modelu na nieznanych danych.

MOŻLIWE/DOSTĘPNE ZBIORY DANYCH

- **National Health and Nutrition Examination Survey (NHANES):** NHANES to szeroko zakrojone badanie przeprowadzane przez Centers for Disease Control and Prevention (CDC) w USA. Dostarcza ono kompleksowych danych dotyczących zdrowia i odżywiania na podstawie reprezentatywnej próbki populacji USA. Zbiór danych zawiera informacje na temat różnych stanów zdrowotnych, w tym cukrzycy, a także dane demograficzne, ocenę stylu życia i pomiary kliniczne.
- **UCI Machine Learning Repository:** UCI Machine Learning Repository zawiera szeroki zakres zbiorów danych odpowiednich do zadań uczenia maszynowego. Jednym z takich zbiorów danych jest "Pima Indians Diabetes Database," który jest powszechnie używany do predykcji cukrzycy. Zawiera on informacje medyczne i demograficzne kobiet z plemienia Pima, w tym poziomy glukozy, ciśnienie krwi, grubość skóry i wyniki dotyczące cukrzycy.
- **Elektroniczne Bazy Danych Zapisów Zdrowotnych (EHR):** Bazy danych EHR, dostępne w instytucjach opieki zdrowotnej, zawierają obszerne informacje o pacjentach, w tym historię medyczną, diagnozy, leczenia i wyniki badań laboratoryjnych. Zbiory te dostarczają mnóstwo informacji, które mogą być wykorzystane do predykcji cukrzycy. Jednakże, dostęp i użycie zbiorów danych EHR może wymagać odpowiednich zezwoleń i procedur dotyczących przetwarzania danych.

OPIS WYBRANEGO ZBIORU DANYCH

Ten zbiór danych pochodzi z National Institute of Diabetes and Digestive and Kidney Diseases. Celem zbioru danych jest diagnostyczne przewidywanie, czy pacjent ma cukrzycę, na podstawie określonych pomiarów diagnostycznych zawartych w zbiorze. Na wybór tych przypadków z większej bazy danych zostało nałożone kilka ograniczeń. W szczególności, wszyscy pacjenci w tym zbiorze to kobiety w wieku co najmniej 21 lat pochodzące z plemienia Pima.

W zbiorze danych w pliku (.csv) można znaleźć kilka zmiennych, z których niektóre są zmiennymi niezależnymi (kilka zmiennych predykcyjnych medycznych), a tylko jedna jest zmienną zależną (wynik).

Zbiór danych uzyskany z witryny Kaggle.com dostarcza cennych informacji do przewidywania cukrzycy na podstawie danych medycznych. Składa się z kilku atrybutów, które są powszechnie związane z czynnikami ryzyka cukrzycy. Te atrybuty to:

- **Liczba ciąż**: Ten atrybut wskazuje, ile razy pacjentka była w ciąży.
- **Poziom glukozy**: Ten atrybut reprezentuje stężenie glukozy we krwi pacjentki mierzone w miligramach na decylitr (mg/dL).
- **Ciśnienie krwi**: Ten atrybut reprezentuje ciśnienie krwi pacjentki mierzone w milimetrach słupa rtęci (mmHg).
- **Grubość skóry**: Ten atrybut oznacza grubość skóry pacjentki mierzoną w milimetrach (mm).
- **Poziom insuliny**: Ten atrybut reprezentuje poziom insuliny pacjentki mierzony w mili-jednostkach międzynarodowych na litr (mIU/L).

- **BMI (Body Mass Index)**: Ten atrybut wskazuje wskaźnik masy ciała pacjentki, obliczany jako waga w kilogramach podzielona przez kwadrat wzrostu w metrach (kg/m^2).
- **Funkcja dziedziczności cukrzycy**: Ten atrybut dostarcza miary ryzyka dziedziczenia cukrzycy na podstawie historii rodzinnej pacjentki.
- **Wiek**: Ten atrybut reprezentuje wiek pacjentki w latach.
- **Wynik**: Ten atrybut służy jako zmienna docelowa i wskazuje, czy pacjentka ma cukrzycę (1) lub nie ma cukrzycy (0).

PREZENTACJA WYBRANYCH NARZĘDZI

- **Numpy**: NumPy jest biblioteką języka Python używaną do różnych operacji na tablicach oraz obliczeń matematycznych w kodzie.
- **TensorFlow i Keras**: Te frameworki języka Python zostały wykorzystane do budowania i trenowania sieci neuronowych.
- **Matplotlib**: Jest to biblioteka języka Python, która dostarcza funkcje do tworzenia wykresów i wizualizacji.
- **Seaborn**: Jest to bilbioteka języka Python, używana do tworzenia wizualizacji takich jak heatmapy macierzy korelacji.
- **Python**: Język programowania używany do implementacji całego projektu.
- **Jupyter Notebook**: Używany jako interaktywne środowisko deweloperskie.

3 SPECYFIKACJA WEWNĘTRZNA I ZEWNĘTRZNA PROGRAMU

Oto kilka najciekawszych części naszego programu:

```
# Podział na dane wejściowe i wyjściowe
X = dataset[:, :8]
y = dataset[:, 8]

# Dodanie cech interakcyjnych
interaction_1 = X[:, 0] * X[:, 1]
interaction_2 = X[:, 2] * X[:, 3]
X = np.column_stack((X, interaction_1, interaction_2))

# Normalizacja danych wejściowych
X = (X - np.mean(X, axis=0)) / np.std(X, axis=0)

# Podział na zbiór uczący i testowy
split_index = int(len(dataset) * 2 / 3)
X_train, y_train = X[:split_index], y[:split_index]
X_test, y_test = X[split_index:], y[split_index:]

# Definicja modelu sieci neuronowej
model = Sequential()
model.add(BatchNormalization(input_shape=(X.shape[1],)))
model.add(Dense(20, input_dim=X.shape[1], activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(15, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(10, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(5, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(1, activation='sigmoid'))
```

Rysunek 1. Przygotowanie danych i przygotowanie modelu.

```
# Kompilacja modelu
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

# Trenowanie modelu

history = model.fit(X_train, y_train, epochs=100, batch_size=10, validation_split=0.2)

# Testowanie modelu
_, accuracy = model.evaluate(X_test, y_test)
print('Accuracy: %.2f' % (accuracy*100))
```

Rysunek 2. Trenowanie i testowanie modelu.

INTERFEJS UŻYTKOWNIKA / GUI / KONSOLA

Jedną z głównych zalet użycia Jupyter Notebook w projekcie jest usprawnienie procesu tworzenia i zwiększenie możliwości eksploracji danych. Jupyter Notebook oferuje kilka korzyści:

- **Efektywne trenowanie modeli**: Jupyter Notebook pozwala na ponowne użycie już wytrenowanych modeli i zapisanych parametrów, co oszczędza czas i zasoby obliczeniowe, szczególnie przy dużych zbiorach danych lub złożonych modelach.
- **Interaktywna eksploracja danych**: Zapewnia interaktywne środowisko umożliwiające wykonywanie komórek kodu na żądanie.
- **Integracja z bibliotekami wizualizacji danych**: Jupyter Notebook bezproblemowo integruje się z bibliotekami takimi jak Matplotlib, Seaborn i Plotly, ułatwiając tworzenie wykresów. Funkcja inline pozwala na natychmiastowe wyświetlanie wizualizacji w notatniku. Ułatwia to obserwację trendów i korelacji w danych.

4 EKSPERYMENTY

ZMIENNE PARAMETRY/WARUNKI I ICH ZNACZENIE

Podczas eksperymentów skupiliśmy się na modyfikacji kilku parametrów i warunków, aby zoptymalizować dokładność naszego modelu, biorąc pod uwagę niewielki zbiór danych. Zmienne parametry i warunki obejmowały:

- **Typy i liczba warstw**: Testowaliśmy różne typy warstw, takie jak warstwy gęste, i zmienialiśmy ich liczbę. Pozwoliło to znaleźć optymalną równowagę między złożonością a wydajnością modelu.
- **Liczba neuronów w każdej warstwie gęstej**: Regulowaliśmy liczbę neuronów, aby sprawdzić, jak wpływa to na zdolność modelu do wychwytywania złożonych wzorców w danych.
- **Parametr Dropout**: Manipulowaliśmy ustawieniami dropoutu, żeby kontrolować, jak bardzo ograniczamy przetrenowanie modelu.
- **Liczba epok**: Regulowaliśmy liczbę epok, aby zapobiec zarówno niedouczeniu, jak i przeuczeniu, analizując wykresy dokładności i strat w celu znalezienia optymalnej liczby epok.
- **Podział zbioru danych**: Eksperymentowaliśmy z różnymi podziałami zbioru danych na zestawy treningowe i testowe, aby zapewnić wystarczającą ilość danych do treningu, zachowując jednocześnie dane do testowania.
- **Format danych wejściowych**: Badaliśmy różne sposoby formatowania i wstępnego przetwarzania danych wejściowych, w tym techniki inżynierii cech i normalizacji danych, aby zwiększyć zdolność modelu do wydobywania znaczących wzorców.

Poprzez manipulowanie i optymalizację tych parametrów i warunków, dążyliśmy do osiągnięcia jak najwyższej dokładności naszego modelu.

JAKIE WYNIKI ZAOBSERWOWANO I DLACZEGO

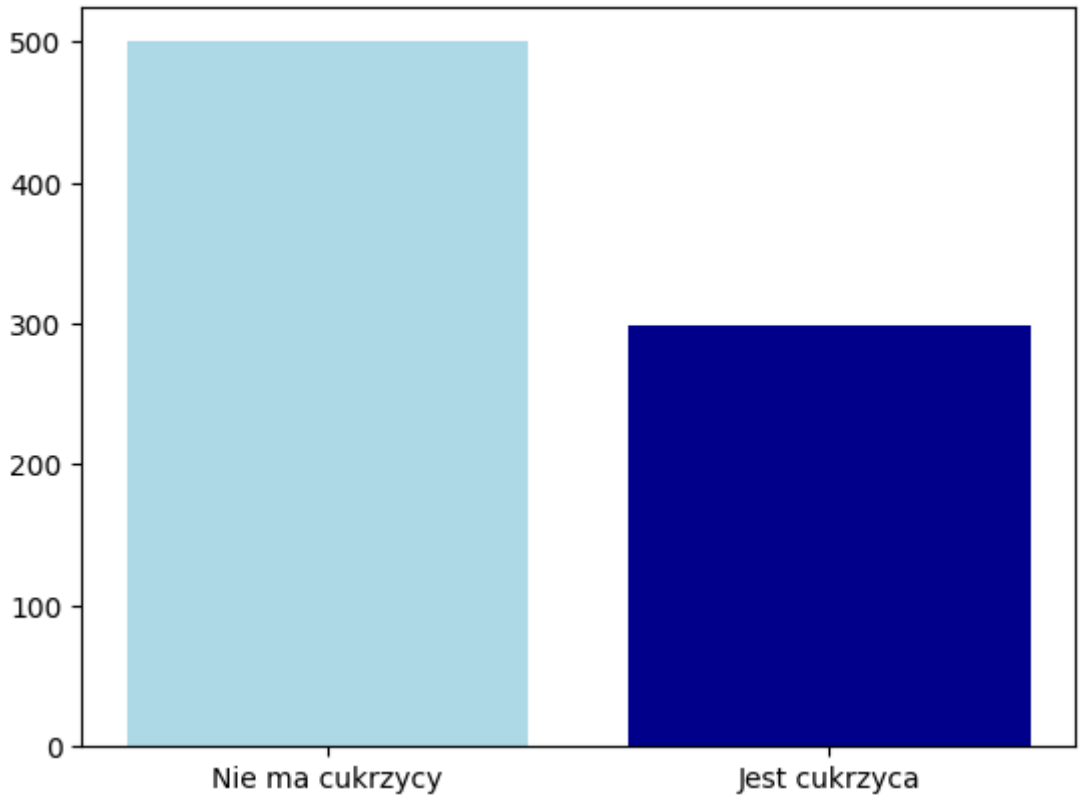
Podczas testowania różnych struktur modelu i wariantów danych wejściowych okazało się, że zwiększenie liczby warstw, epok lub neuronów w każdej warstwie gęstej nie zawsze gwarantuje wyższą dokładność. Poprawy dokładności nie były wyłącznie zależne od tych parametrów.

Zamiast bezmyślnie zwiększać złożoność modelu, przyjęliśmy bardziej systematyczne podejście, eksperymentując z różnymi kombinacjami parametrów, takimi jak architektura sieci neuronowej, funkcje aktywacji, współczynniki dropout i tempo uczenia. Dzięki starannemu dostrojeniu tych parametrów osiągnęliśmy znaczące poprawy dokładności. Iteracyjne testy i eksperymenty pozwoliły znaleźć optymalną konfigurację.

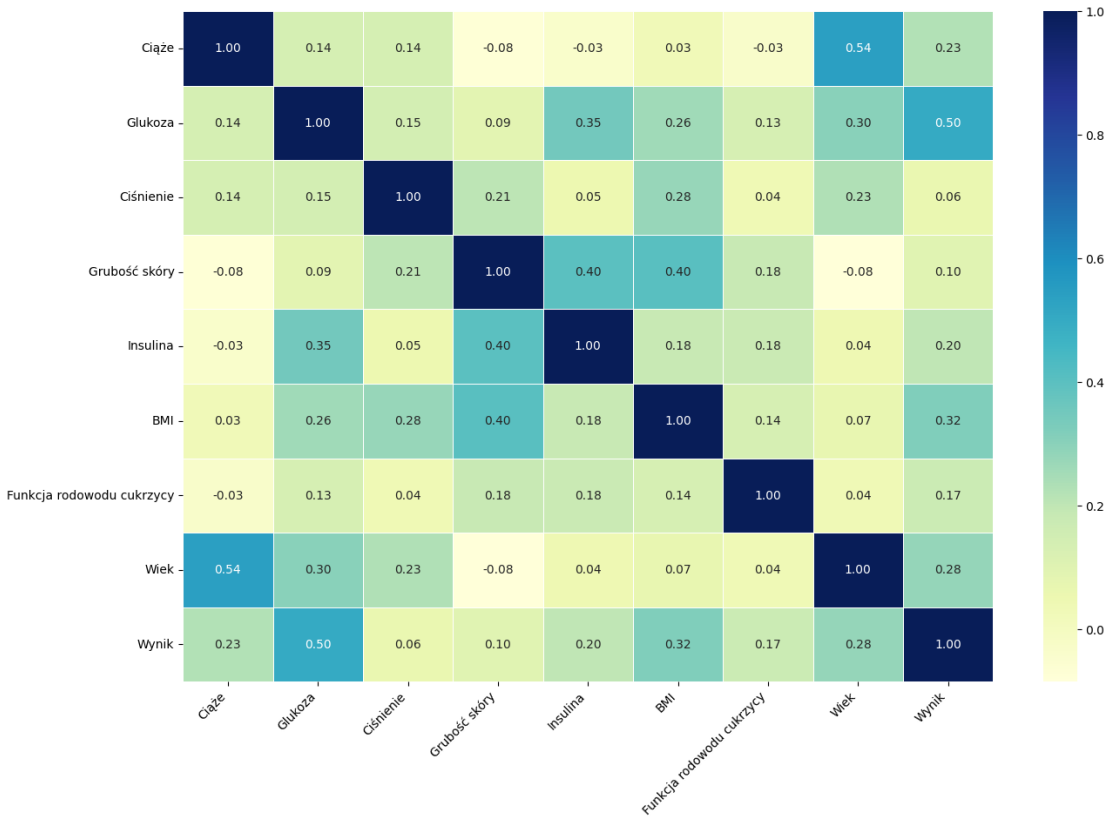
METODA PREZENTACJI WYNIKÓW – OPIS DIAGRAMÓW

W celu skutecznej prezentacji wyników użyjemy kilku diagramów:

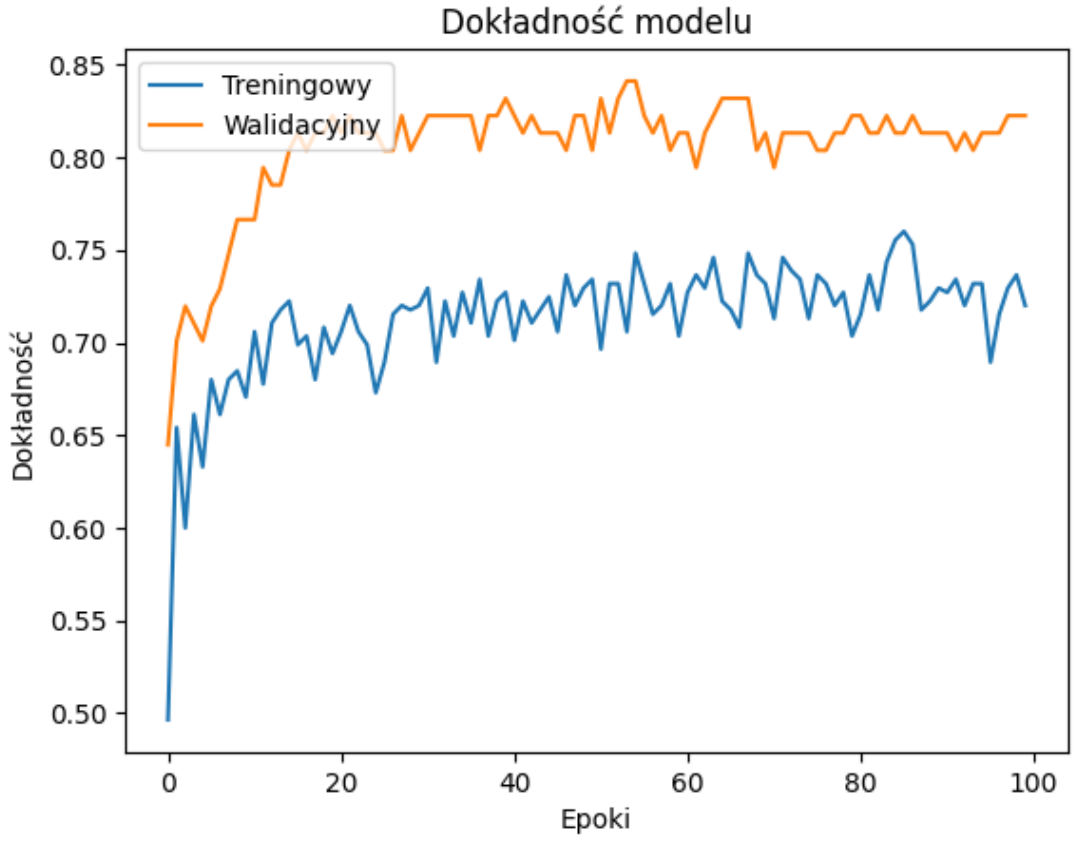
- **Mapa korelacji**: Przedstawia wizualnie korelację między różnymi cechami w zestawie danych, co pomaga zrozumieć związki między zmiennymi i ich wpływ na predykcję cukrzycy. Mapa korelacji jest zazwyczaj wyświetlana jako mapa cieplna, gdzie każda komórka wskazuje siłę i kierunek korelacji.
- **Diagram wyników**: Ilustruje rozkład wyników w zestawie danych, pokazując liczbę przypadków sklasyfikowanych jako cukrzyca i brak cukrzycy. Umożliwia ocenę równowagi zbioru danych.
- **Diagram dokładności modelu**: Przedstawia dokładność modelu na przestrzeni epok treningowych, co pozwala obserwować, jak dokładność ewoluuje w trakcie procesu treningowego.
- **Diagram strat modelu**: Wyświetla straty modelu na przestrzeni epok treningowych. Funkcja strat mierzy różnicę między przewidywanymi a rzeczywistymi wynikami i służy jako metryka optymalizacyjna podczas treningu modelu.



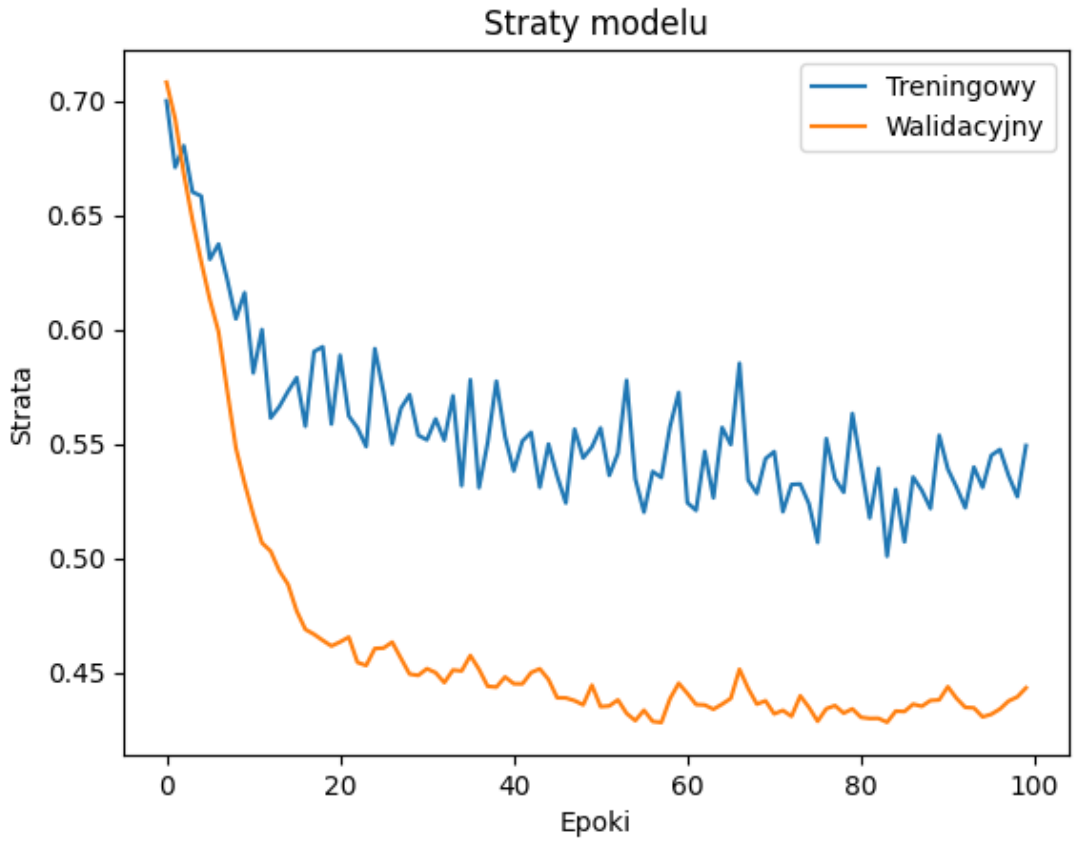
Rysunek 3. Diagram danych wyjściowych.



Rysunek 4. Mapa korelacji.



Rysunek 5. Diagram dokładności modelu.



Rysunek 6. Diagram strat modelu.

PREZENTACJA EKSPERYMENTÓW

W diagramie wyników zauważyliśmy, że liczba osób bez cukrzycy jest znacznie większa niż z cukrzycą. Ta nierównowaga klas może wpłynąć na wydajność modelu, a zwiększenie liczby rekordów mogłoby poprawić dokładność algorytmu. Ważne jest uwzględnienie nierównowagi klas, aby model nie był stronniczy wobec większościowej klasy.

Analizując mapę korelacji, zidentyfikowaliśmy cechy o ograniczonym wpływie na wynik cukrzycy, jak "grubość skóry", która często miała wartość 0, co sugeruje brak pomiaru dla tych osób. Usunięcie takich cech może uprościć model i poprawić jego wydajność.

Patrząc na diagramy dokładności i strat, zaobserwowaliśmy pozytywny trend w dokładności modelu z każdą epoką. Diagram strat pokazuje konwergencję funkcji strat, co oznacza, że model zbliża się do optymalnego rozwiązania. Liczba epok była odpowiednio ustawiona, gdyż dokładność wzrasta, a następnie się wyrównuje, co sugeruje brak przeuczenia.

Te wyniki dostarczają cennych informacji o wydajności i zachowaniu naszego modelu. Uwzględnienie nierównowagi klas i selekcji cech, wraz z obserwowanym zachowaniem uczenia się i odpowiednią liczbą epok, przyczynia się do zrozumienia i oceny dokładności oraz ogólnej wydajności modelu.

5 PODSUMOWANIE

OGÓLNE WNIOSKI

Naszym celem było opracowanie modelu predykcji cukrzycy na podstawie danych medycznych z użyciem sieci neuronowych. Przeprowadziliśmy analizy i eksperymenty, co doprowadziło do kilku wniosków:

- **Zbiór danych:** Wystąpiła nierównowaga klas, z większą liczbą osób nie chorych na cukrzycę. Może to wpłynąć na wydajność modelu, a zwiększenie liczby rekordów mogłoby poprawić dokładność.
- **Wybór cech:** Niektóre cechy, jak „grubość skóry", miały ograniczony wpływ na wyniki. Usunięcie takich cech może uprościć model i poprawić jego wydajność.
- **Wydajność modelu:** Diagramy dokładności i strat pokazały, że model uczył się i poprawiał z każdą epoką.
- **Liczba epok:** Diagram dokładności sugerował, że liczba epok była dobrze dobrana, co zapobiegło przeuczeniu.
- **Narzędzia i frameworki:** Użyliśmy Pythona oraz bibliotek TensorFlow i Keras do budowy i trenowania modelu.

Nasze wyniki podkreślają znaczenie przetwarzania danych, wyboru cech oraz oceny modelu w celu wyboru odpowiednich parametrów.

MOŻLIWE ULEPSZENIA

Pomimo obiecujących wyników, istnieje kilka sposobów na dalszą poprawę modelu:

- **Zwiększenie rozmiaru zbioru danych:** Może to pomóc zniwelować nierównowagę klas.
- **Adresowanie nierównowagi klas:** Techniki takie jak nadpróbkowanie, podpróbkowanie lub SMOTE mogą poprawić dokładność predykcji.
- **Zaawansowane architektury:** Eksploracja bardziej zaawansowanych sieci neuronowych, jak RNN, CNN czy modele oparte na transformatorach, może przynieść lepsze wyniki.
- **Optymalizacja hiperparametrów:** Dokładniejsze poszukiwanie optymalnych hiperparametrów.
- **Inżynieria cech:** Tworzenie nowych cech lub przekształcanie istniejących może poprawić zdolność modelu do wychwytywania istotnych informacji.
- **Metody zespołowe:** Łączenie wielu modeli może zwiększyć dokładność predykcji.
- **Interpretowalność:** Badanie technik zwiększających interpretowalność modelu może pomóc zrozumieć, co wpływa na jego predykcje.

6 BIBLIOGRAFIA – LISTA ŹRÓDEŁ UŻYTYCH PODCZAS PRACY NAD PROJEKTEM

- https://dmsjournal.biomedcentral.com/articles/10.1186/s13098-021-00767-9?fbclid=IwAR1rypcyq7KFlXPI363z9n1pkRyczw4STBZ_K9OwaPf6eVCq1h1jnz2SdQU
- https://www.researchgate.net/publication/347091823_Diabetes_Prediction_Using_Machine_Learning
- https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9318204/?fbclid=IwAR3TZi08mXJhHh5vxeJsEkxAzBQoVSl-vTQtdSpFN3JSXkdsTtHyDUjZJLM#B27-sensors-22-05304
- https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database
- https://medium.com/botsupply/a-beginners-guide-to-deep-learning-5ee814cf7706
- https://www.learndatasci.com/glossary/binary-classification/

7 LINK DO GITHUB'A Z KODEM ŹRÓDŁOWYM

https://github.com/kszczepanczyk/diabetes-prediction