



AGH

AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

WYDZIAŁ INŻYNIERII METALI I INFORMATYKI PRZEMYSŁOWEJ

KATEDRA Informatyki Stosowanej i Modelowania

Projekt dyplomowy

Wykorzystanie sieci neuronowych do przetwarzania sygnałów biomedycznych w ocenie ryzyka patologii

The use of neural networks to process biomedical signals in the assessment of pathology risk

Autor:

Przemysław Piotr Janiszewski

Kierunek studiów:

Informatyka Techniczna

Opiekun projektu:

dr hab. inż. Krzysztof Regulski

Kraków, 2025

Spis treści

1. WSTĘP	3
2. SZTUCZNA INTELIGENCJA – PRZEGŁĄD I KLASYFIKACJA	5
2.1 CZYM JEST AI	5
2.2 UCZENIE MASZYNOWE	6
2.3 FUNKCJA AKTYWACJI	7
2.4 FUNKCJE STRATY I OPTYMALIZATORY	8
2.5 PRZYKŁADOWE RODZAJE ARCHITEKTUR SIECI NEURONOWYCH	9
2.5.1 <i>Multilayer perceptron (MLP)</i>	9
2.5.2 <i>TabNet (Attentive Interpretable Tabular Learning)</i>	11
2.5.3 <i>TabGPT (Tabular Generative Pre-trained Transformer)</i>	13
3. IMPLEMENTACJA	16
3.1 ANALIZA ZAGROŻENIA CHOROBĄ SERCA PRZY UŻYCIU MLP	16
3.1.1 <i>Preprocessing</i>	17
3.1.2 <i>Analiza statystyczna</i>	18
3.1.3 <i>Trening sieci</i>	23
3.1.4 <i>Postprocessing</i>	25
3.2 SIECI ATENCYJNE W OCENIE RYZYKA WYSTĄPIENIA CUKRZCY	27
3.2.1 <i>Uwagi wstępne</i>	28
3.2.2 <i>Trening sieci</i>	29
3.2.3 <i>Postprocessing</i>	30
4. OCENA SKUTECZNOŚCI MODELU	31
4.1 DIAGNOSTYKA RYZYKA WYSTĄPIENIA CHORÓB SERCA	31
4.2 DIAGNOSTYKA RYZYKA WYSTĄPIENIA CUKRZCY	34
5. DUŻE MODELE JĘZYKOWE (LLM'S) W ANALIZACH MEDYCZNYCH	36
5.1 WALIDACJI ZAPYTAŃ NA PODSTAWIE PODOBIĘSTWA DO ZAKAZANYCH FRAZ SŁOWNIKOWYCH	37
5.2 STRUKTURA MODELU I PROMPTY SYSTEMOWE	38
5.3 WALIDACJI ODPOWIEDZI POD WZGLĘDEM WYSTĘPOWANIA ZAKAZANYCH KLUCZY	39
5.4 TEXT-TO-ACTION	41
5.5 ZDOLNOŚĆ DUŻYCH MODELI JĘZYKOWYCH DO ANALIZY DANYCH TABELARYCZNYCH	41
6. WDROŻENIE	44
6.1 ŚRODOWISKO DEWELOPERSKIE	45
6.1.1 <i>Docker</i>	45
6.1.2 <i>Docker Compose</i>	45
6.1.3 <i>Aplikacja w środowisku lokalnym</i>	45
6.2 ŚRODOWISKO PRODUKCYJNE	46
6.2.1 <i>Microsoft Azure</i>	46
6.2.2 <i>Microsoft Azure Portal</i>	47
6.2.3 <i>Azure Command-Line Interface (ACI)</i>	47
6.2.4 <i>Terraform</i>	48
6.2.5 <i>Aplikacja w środowisku chmurowym</i>	48
7. TESTY APLIKACJI	52
8. PODSUMOWANIE	55
9. BIBLIOGRAFIA	56

1. Wstęp

W obecnych czasach cyfryzacja odgrywa kluczową rolę w kształtowaniu wielu aspektów naszego życia. Wraz z jej rozwojem inżynierowie opracowują coraz bardziej zaawansowane algorytmy, mające na celu wyręczanie człowieka w powtarzalnych i mało kreatywnych zadaniach. Aktualnie ten sztuczny intelekt nie jest jednak w stanie rozumieć i świadomie podejmować decyzji, lecz pełni jedynie rolę odtwórczą – uogólniając i dopasowując olbrzymie zbiory danych do określonych potrzeb [1]. Budzi to jednak uzasadnione obawy i nasuwa pytanie: Czy w przyszłości nie stanie się on na tyle inteligentny, by człowiek stał się zbędny, nie będąc w stanie zaoferować nic ponad ten system?

W roku 1950 roku Alan Turing wyraził te niepokoje, formułując słynny: „Test Turinga”. Oparty był o tzw. „grę w naśladownictwo”, w której to przesłuchujący otrzymuje zadanie odróżnienia mężczyzny i kobiety, wyłącznie na podstawie udzielonych wcześniej pisemnych odpowiedzi na postawione pytania. W pewnym momencie jedno z nich zostaje zastąpione maszyną. Test zostaje zaliczony, gdy niemożliwe staje się jednoznaczne wykazanie nieludzkiego pochodzenia rezultatu. W założeniu nie był on jednak w stanie udowodnić zdolności myślenia maszyny, lecz sprawdzał jedynie, czy będzie ona w stanie udawać tę umiejętność na tyle dobrze, by oszukać człowieka [2].

Ponad 70 lat później, w roku 2024 naukowcy z Uniwersytetu Stanforda, ogłosili przejście rygorystycznej wersji tego testu przez popularną ówcześnie generatywną sztuczną inteligencję – ChatGPT 4.0. Za pomocą pięcioczynnikowego modelu osobowości badacze przeanalizowali jej umiejętności pod kątem takich cech jak: neurotyczność, ekstrawersja, otwartość na doświadczenie, ugodowość i sumienność. Chatbot nie wykazał odchyлеń od normy w żadnym z tych zakresów, jednakże jego „sympatyczność” okazała się niższa od 66% badanych respondentów. Następnie eksperci przeanalizowali jego procesy decyzyjne podczas gier dylematów społecznych. Algorytm odznaczał się w nich tendencją do zapewniania możliwie jak największych korzyści zarówno dla siebie jak i innych. Odnalazłby się więc dobrze w roli mediatora, gdyż często w sytuacjach problematycznych potrafił zachowywać się w sposób „bardziej ludzki” niż sam człowiek [3]. Pomimo tej cechy, nie jest w stanie brać odpowiedzialności za swoje działania oraz wzbudzać zaufania międzyludzkiego, co nadal jest czynnikiem wyróżniającym człowieka. Interakcja ze sztuczną inteligencją nie powinna więc opierać się na próbie jej prześcignięcia, lecz na zasadzie wzajemności i współpracy.

Ta komplementarność znajduje swoje odzwierciedlenie szczególnie w naukach medycznych. „Inteligentne” algorytmy mogą stawiać diagnozy oraz wspomagać wykonywanie badań - zwłaszcza w czasach zwiększającego się popytu na świadczenia zdrowotne, których ilość jest niewystarczająca i często wątpliwej jakości. Niemniej jednak, analizy te zawsze powinna nadzorować kompetentna osoba, która swoją wiedzą i doświadczeniem będzie w stanie zweryfikować trafność zaproponowanych rozwiązań.

W niniejszym projekcie przedstawiony zostanie przykład takiej interakcji na podstawie opracowanego systemu ekspertowego, który służyć będzie przewidywaniu ryzyka wystąpienia patologii medycznych u przebadanych pacjentów. Na podstawie dostępne zbiory danych wyszkolone zostaną modele uczenia maszynowego, które ułatwią kompetentnym osobom podejmowanie szybszych i bardziej racjonalnych decyzji diagnostycznych. Działanie to pozwoli również na przesiewową analizę grup podwyższzonego ryzyka, zwłaszcza z rejonów świata, gdzie dostęp do służby zdrowia bywa znaczowo utrudniony.

Możliwość skorzystania z przygotowanej aplikacji internetowej zostanie zapewniona dzięki jej *hostowaniu* w chmurze Microsoft Azure. Tak opracowana strona umożliwiać będzie wstępna diagnozę i rekomendację do skontaktowania się ze specjalistą w następujących przypadkach:

- cukrzycy (ogół społeczeństwa)
- choroba wieńcowa (CHD) / zawał mięśnia sercowego (ogół społeczeństwa)

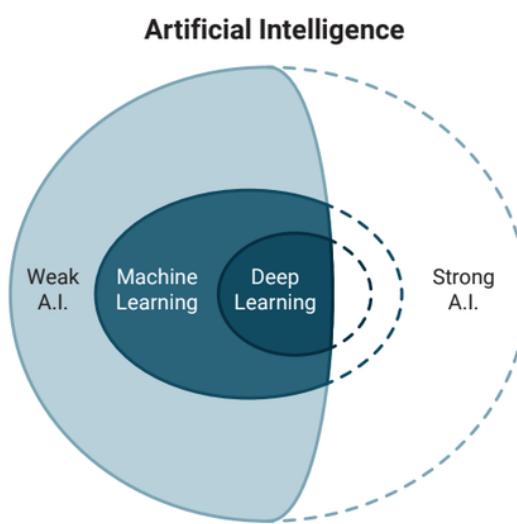
Ponadto odwiedzający będą mieli możliwość interakcji z dużym modelem językowym LLAMA 3.1, który umożliwi im uzyskanie szerokiego zakresu informacji dotyczących tematów około zdrowotnych. Należy mieć jednak na uwadze, że technologia ta nadal jest intensywnie rozwijana i obecnie często charakteryzuje się tendencją do dostarczania błędnych odpowiedzi, w związku z czym powinny być one zawsze zweryfikowane poprzez rzetelne źródło wiedzy, co zasugerowane zostanie użytkownikowi.

2. Sztuczna inteligencja – przegląd i klasyfikacja

2.1 Czym jest AI

Encyklopedia Britannica określa ją jako zdolność algorytmów komputerowych do odkrywania ukrytych znaczeń w przetwarzanych informacjach oraz wyciągania wniosków na podstawie przeszłych doświadczeń [4]. Definicja ta nie kładzie nacisku na ich dedukcyjność, lecz ogranicza się głównie do umiejętności generalizacji. Bez odpowiednich danych są więc zupełnie niezdolne do prawidłowego działania. Ta obserwacja pozwala wstępnie zrozumieć ich istotę oraz wskazać różnicę pomiędzy ludzkimi procesami poznawczymi, dla których wciąż pozostaje aktualne słynne stwierdzenie Kartezjusza, głoszące wyższość człowieka, zdolnego intencjonalnie identyfikować swoje „ja”, które często nazywa się świadomością. Czym jednak jest ten odróżniający nas czynnik?

By odpowiedzieć na to pytanie, warto zapoznać się z jego aktualną koncepcją definiowaną jako umiejętność refleksji jednostki nad swoimi działaniami oraz przeżywanymi emocjami [5]. Mimo licznych obaw współcześnie ta podstawa ludzkiej natury nie jest zagrożona, gdyż nie odkryto sposobu stworzenia tzw. silnej sztucznej inteligencji. Jest ona hipotetycznym koncepcją, która podobnie jak człowiek, byłaby w stanie uczyć się i wyciągać wnioski w obszarach wcześniej mu nieznanych. Wielu naukowców nie jest przekonanych czy jej stworzenie jest w ogóle możliwe, zważając, że temat ten nadal wymaga licznych badań z zakresu biologii, psychologii, informatyki oraz wielu innych pokrewnych dziedzin. Dalsze rozważania odnosić się będą domyślnie do jej słabej, względnie dobrze znanej formy.



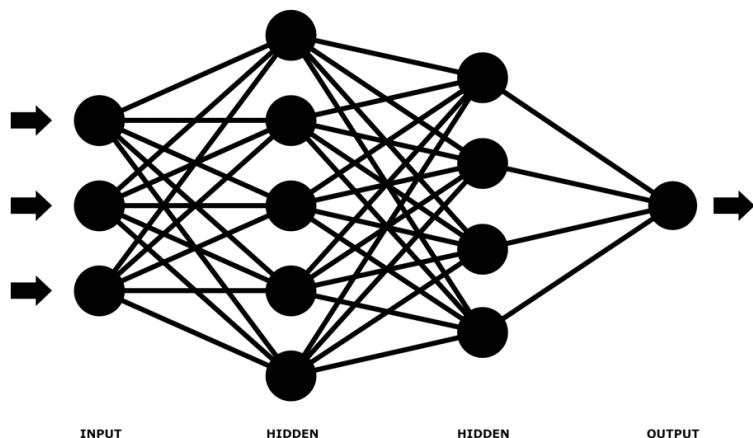
Rys. 2.1 Podział sztucznej inteligencji (źródło: Komodo Technologies, LLC [6])

2.2 Uczenie maszynowe

To podzbiór sztucznej inteligencji zajmujący się analizą praktycznego wykorzystania autonomicznych algorytmów do rozwiązywania wszelkich problemów dziedzinowych. Wykorzystuje ono magazynowanie wiedzy w postaci wag połączeń pomiędzy poszczególnymi jednostkami obliczeniowymi zwanymi sztucznymi neuronami. Są one analogią do komórek nerwowych obserwowanych w mózgach zwierząt i działają na podobnej zasadzie. Otrzymawszy sygnał wejściowy, powodują jego nieliniową transformację poprzez operację iloczynu skalarnego z wagami połączeń oraz finalne przejście przez tzw. funkcję aktywacji – decydującą o końcowym kształcie przetwarzanego impulsu. Na jego charakter, w dużej mierze wpływa również *bias* – dodatkowy czynnik pozwalający na liniowe przesunięcie funkcji aktywacji i lepsze dostosowywanie się do charakteru rozważanych problemów.

Pojedynczy sztuczny neuron (podobnie jak jego biologiczny odpowiednik) nie jest w stanie analizować bardziej złożonych wzorców, w związku z czym, by efektywnie realizować założone cele, konieczna staje się ich współpraca. Objawia się ona w postaci grup neuronowych nazywanych powszechnie warstwami. Każda z nich specjalizuje się w konkretnej funkcji i odpowiada m.in. za rozdzielenie przetwarzanych danych, redukcję wymiarowości, normalizację czy agregację wyekstrahowanych cech.

Złożone modele nazywane są głębokimi sieciami neuronowymi. Ich kluczową cechą jest liczba warstw ukrytych wykorzystana podczas procesu trenowania. Z reguły im większa ich ilość, tym lepsza skuteczność modelowania zjawiska. W praktyce osiąga się pod tym względem pewien kompromis, tak aby sieć zachowała swoją zdolność do generalizacji, jednocześnie prawidłowo odwzorowując złożoność problemu. Sam dobór liczby warstw oraz ich parametrów jest czynnością eksperymentalną, gdyż w odróżnieniu od programowania deterministycznego praktycznie niemożliwe jest określenie wyniku sieci w sposób implicit.

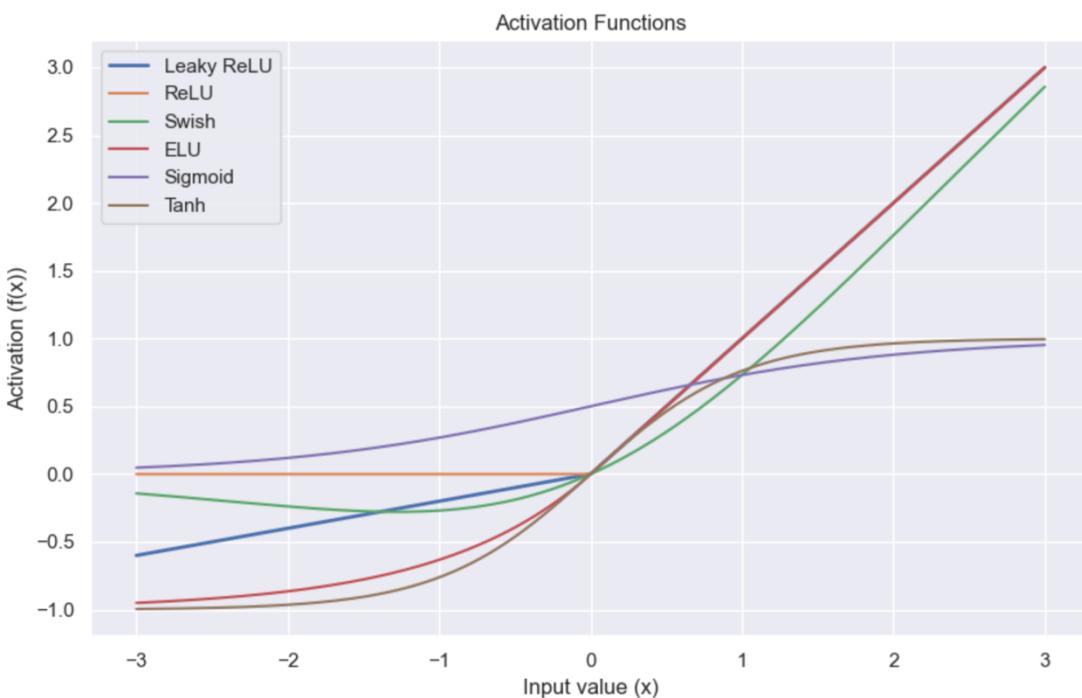


Rys. 2.2 Sieć MLP z 2 gęstymi warstwami ukrytymi (źródło: opracowanie własne)

2.3 Funkcja aktywacji

Jest kluczowym elementem determinującym charakter wyjściowego sygnału neuronów. Pełni wiele znaczących funkcji, z czego szczególne znaczenie ma wprowadzenie do modelu nieliniowości. Poprzez rozdzielanie warstw operacyjnych, sieć staje się zdolna do nauki bardziej zawiłych wzorców, niebędących prostą liniową kombinacją cech początkowych. Dobrym przykładem realizującym to działanie jest funkcja ReLU zerująca wszystkie sygnały ujemne i przepuszczająca dodatnie bez zmian. Niestety jej działanie prowadzi czasami do powstawania tzw. „martwych neuronów”, które nigdy nie zostaną aktywowane, ponieważ zawsze otrzymują na wejściu sygnał niedodatni. By temu zapobiec, w bardziej zaawansowanych i złożonych modelach wykorzystuje się jej modyfikacje tj. ELU czy Leaky ReLU uwzględniające w końcowej predykcji wpływ sygnału negatywnego, stabilizując w ten sposób wartości wyliczanych gradientów.

Często oczekuje się, iż sygnał warstwy wyjściowej będzie znajdował się w ścisłe określonym zakresie. Przykładem może być przypisanie danemu zjawisku prawdopodobieństwa przynależności do konkretnej klasy. W takim przypadku wynik należy przedstawić w formie ułamków należących do przedziału $[0,1]$, dla których łączna suma „szans” musi być równa jedności. Normalizacja ta odbywa się przy wykorzystaniu funkcji sigmoidalnych tj. sigmoid w przypadku klasyfikacji binarnej, czy softmax dla problemów wieloklasowych.



Rys. 2.3 Funkcje aktywacji (źródło: opracowanie własne)

2.4 Funkcje straty i optymalizatory

W celu określenia skuteczności przewidywań modelu wykorzystuje się matematyczne funkcje pozwalające w sposób liczbowy wyrazić jego aktualne dostosowanie do zademonstrowanych danych. Ich zadanie polega na pomiarze różnicy pomiędzy rzeczywistymi etykietami a wygenerowaną predykcją. Warto mieć na uwadze, iż ich dobór jest ściśle powiązany z rozwiązywanym problemem tak, aby jak najlepiej oddawać jego charakter. Jedne z najpowszechniej znanych i używanych przedstawiono w Tabeli 2.1.

Skrót	Nazwa	Wzór matematyczny	Opis
MSE	Mean Squared Error	$\frac{1}{n} \sum_{i=1}^n (y_i - \bar{y}_i)^2$	Średnia kwadratowa różnica pomiędzy wartościami obserwowanymi a przewidzianymi przez krzywą regresyjną
MAE	Mean Absolute Error	$\frac{1}{n} \sum_{i=1}^n y_i - \bar{y}_i $	Średnia absolutna różnica pomiędzy wartościami obserwowanymi a przewidzianymi przez krzywą regresyjną
BCE	Binary Cross-Entropy	$-\frac{1}{n} \sum_{i=1}^n [y_i \log \bar{y}_i + (1 - y_i) \log(1 - \bar{y}_i)]$	Różnica pomiędzy rzeczywistymi etykietami a przewidywanym prawdopodobieństwem dla klasyfikacji binarnej
CCE	Categorical Cross-Entropy	$-\frac{1}{n} \sum_{i=1}^n \sum_{j=1}^k y_{ij} \log \bar{y}_i$	Różnica pomiędzy rzeczywistymi etykietami a przewidywanym prawdopodobieństwem dla wszystkich k-klas

Tab 2.1. Wybrane funkcje strat (źródło: opracowanie własne)

Skuteczna adaptacja sieci nie mogłaby istnieć bez równie wydajnych algorytmów optymalizujących. Ich zadanie polega na takim dostosowywaniu połączeń pomiędzy poszczególnymi neuronami, by funkcja straty przyjmowała jak najmniejsze wartości. Spośród najpopularniejszych procedur wyróżnić można przedstawione poniżej:

- Gradient Descent (GD) – aktualizuje parametry w kierunku przeciwnym do gradientu (największego spadku)
- Stochastic Gradient Descent (SGD) – działa podobnie do GD, ale aktualizuje parametry modelu, korzystając z gradientu pojedynczej, losowej próbki
- Momentum – rozszerzenie SGD o współczynnik „pędu” biorący pod uwagę nie tylko obecne, lecz także wcześniejsze wartości gradientów, co pozwala „rozpedzić się” w kierunku minimum funkcji celu
- RMSprop – wykorzystuje średnią kwadratową gradientów do aktualizacji parametrów sieci
- Adam – algorytm łączący zalety dwóch poprzednich, umożliwiający szybkie i efektywne uczenie dla wielu różnych typów sieci neuronowych

2.5 Przykładowe rodzaje architektur sieci neuronowych

2.5.1 Multilayer perceptron (MLP)

Jest to jedna z najprostszych, lecz nadal równie skutecznych struktur uczenia maszynowego. Wykorzystanie wielu warstw pozwala jej niezwykle skutecznie modelować znaczną większość obserwowanych zjawisk. Do roku 1986 nie cieszyła się jednak dużym zainteresowaniem inżynierów. Choć znany był algorytm przepuszczania danych przez sieć (forward pass), nadal brakowało efektywnego sposobu pozwalającego na samoadaptację sieci do generowanych przez nią sygnałów. Trudność ta wynikała głównie z nieumiejętnością poradzenia sobie z problemem zanikającego gradientu, premującego aktualizację wag pomiędzy połączeniami znajdującymi się najbliżej wyjścia. W konsekwencji model nie umiał odnaleźć takiego układu, który skutecznie minimalizowałby zdefiniowaną wcześniej funkcję celu, gdyż połączenia znajdujące się blisko wejścia nie były w ogóle modyfikowane. Do jej popularności nie przyczyniała się również ówczesna niska moc obliczeniowa komputerów oraz początkowa faza rozwoju tej dziedziny nauki.

Wszystko zmieniło się za sprawą artykułu opublikowanego w czasopiśmie Nature pt. "Learning representations by back-propagating errors" autorstwa Davida Rumelharta, Geoffreya Hintona i Ronalda Williamsa. Naukowcy zawarli w nim koncepcje zupełnie nowego sposobu poradzenia sobie z opisanym wyżej problemem. Ta rewolucja opierała się na obliczaniu błędu predykcji, by po dokonaniu jego propagacji wstecz adekwatnie dostosować wagi połączeń wedle ich istotności. By osiągnąć ten efekt, wylicza się sygnał wyjściowy sieci przy użyciu poniższych wzorów (dla i -tej iteracji)

$$\begin{aligned} z^{(i)} &= w^{(i)} * (a^{i-1}) + b^{(i)} \\ a^{(i)} &= \sigma(z^{(i)}) \end{aligned}$$

gdzie:

- $z^{(i)}$ – wyniki sieci przed aktywacją
- $w^{(i)}$ – macierz wag
- $a^{(i)}$ – wynik sieci po aktywacji (dla $a^{(0)} \equiv$ wektor cech wejściowych)
- $b^{(i)}$ – wektor biasów

Otrzymany wynik jest kolejno rozpropagowywany w kierunku wcześniejszych warstw, by przy pomocy pochodnych funkcji aktywacji określić wpływ każdej wagi na końcowy rezultat. Umożliwia to dokonanie sprawiedliwych aktualizacji, przeskalowanych o tzw. czynnik uczący. Poniższe podejście jest praktycznym przykładem wykorzystania wspomnianego wcześniej optymalizatora standardowego spadku gradientu, niezwykle popularnego w latach 90. XX w. [7].

$$\begin{aligned}\delta^{(l)} &= \frac{\partial L(y, \bar{y})}{\partial a^{(l)}} * \sigma'(z^{(l)}) \\ \frac{\partial L(y, \bar{y})}{\partial w^{(l)}} &= \delta^{(l)} * (a^{l-1})^T \\ w_i^{l+1} &= w_i^l - \eta * \frac{\partial L(y, \bar{y})}{\partial w^{(l)}}\end{aligned}$$

Gdzie:

- $\delta^{(l)}$ – gradient błędu, określający, w jakim stopniu wyjście warstwy wpłynęło na całkowity błąd sieci
- $\frac{\partial L(y, \bar{y})}{\partial a^{(l)}}$ – pochodna funkcji błędu względem wyjścia sieci, określająca jego wpływ na wartość błędu przewidynań
- $\frac{\partial L(y, \bar{y})}{\partial w^{(l)}}$ – pochodna funkcji kosztu względem wektora wag, określająca ich wpływ na wartość błędu przewidynań
- η – współczynnik uczenia (learning rate), kontrolujący wielkość kroku wykonywaną w kierunku minimalizacji funkcji celu. Jego dobór zazwyczaj dokonuje się doświadczalnie - z uwagi na brak występowania jednej ogólnej wartości, która byłaby najlepsza w każdym przypadku.

Współcześnie sieci MLP wykorzystuje się głównie do rozwiązywania problemów klasyfikacji oraz regresji. W pierwszym przypadku ich zadanie polega na przypisaniu badanej próbce prawdopodobieństwa przynależności do każdej z określonych klas, natomiast w drugim na znalezieniu współczynników krzywej najlepiej aproksymującej analizowane zjawisko.

2.5.2 TabNet (Attentive Interpretable Tabular Learning)

„Czarnoskrzynkowa” odpowiedź sieci MLP często jest niewystarczająca. Zwraca ona jedynie czysty wynik, na podstawie którego ciężko jest zrozumieć proces decyzyjny stojący za jego otrzymaniem. Wraz z rozwojem uczenia maszynowego problem ten zaczął być coraz bardziej zauważalny, a krytyczne dziedziny takie jak medycyna czy prawo zaczęły domagać się jego rozwiązania. Nowa epoka XAI (Explainable AI) zapoczątkowana została w roku 2020, gdy zespół badawczy Google Cloud AI prowadzony przez O. Arik i Tomas Pfister opublikował pracę dotyczącą nowego rodzaju sieci neuronowej, zdolnej identyfikować kluczowe cechy mające wpływ na finalną predykcję [8].

Są one określane poprzez tzw. macierz uwagi (attention matrix), która za pomocą selektywnych masek uwzględnia tylko istotne cechy, pomijając te mało ważne lub wybrakowane. Jej wyznaczenie możliwe jest przy wykorzystaniu poniższej formuły:

$$M[i] = \text{sparsemax}(P[i - 1] * h_i(a[i - 1]))$$
$$P[i] = \prod_{j=1}^i (\gamma - M[j])$$

gdzie:

- $P[i]$ – skala priorytetów, określająca które kroki były istotne ($P[0] = 1$). Jest zależna od maski z poprzedniej iteracji.
- $h_i(a[i])$ - funkcja przekształcenia (zwykle warstwa gęsta)
- γ – parametr kontrolujący wpływ nowych masek na kolejne iteracje

W celu uzyskania finalnego wyniku, konieczne jest jego przetworzenie przy wykorzystaniu funkcji sparsemax. Pozwala ona identyfikować najbardziej kluczowe wartości wpływające na ostateczny rezultat. Dla nich przypisany zostanie pewien rozkład prawdopodobieństwa, zaś pozostałym elementom przyporządkowane zostaną wartości zerowe.

```

def sparsemax(z: Union[np.ndarray, List[float]]) -> np.ndarray:
    z = np.array(z, dtype=float)
    n = z.shape[0]

    sorted_indices = np.argsort(z)[::-1]
    sorted_z = z[sorted_indices]

    cumulative_sum = np.cumsum(sorted_z)

    support_size = np.arange(1, n + 1)

    threshold = (cumulative_sum - 1) / support_size

    valid_mask = sorted_z > threshold
    k = np.sum(valid_mask)

    if k == 0:
        return np.zeros(n)
    else:
        theta = threshold[valid_mask][-1]
        return np.maximum(z - theta, 0)

```

Rys. 2.4 Implementacja algorytmu sparsemax w języku Python (źródło: opracowanie własne)

Dodatkowym aspektem poprawiającym interpretowalność modelu jest stosowanie regularyzacji - zachęcającej model do perforowania wyboru mniejszej ilości cech znaczących. Pozwala to łatwiej wyjaśnić decyzje podejmowane przez model oraz zwiększyć jego skuteczność poprzez eliminację losowości i zakłóceń występujących w zbiorze danych. W tym przypadku wyznacza się ją wedle poniższej formuły

$$L_{sparse} = \frac{1}{(N_{steps} * B)} \sum_{i=1}^{N_{steps}} \sum_{b=1}^B \sum_{j=1}^D -M_{b,j}[i] \log(M_{b,j}[i] + \epsilon)$$

gdzie:

- N_{steps} – liczba iteracji
- B – liczba batchy (próbek przetwarzanych jednocześnie w jednym kroku uczenia)
- D – liczba cech \Leftrightarrow długość wektora danych wejściowych dla i-tej iteracji
- $M_{b,j}[i]$ – element macierzy uwagi dla partii b, cechy j oraz i-tego kroku
- ϵ - niewielka wartość stabilizująca obliczenia w sytuacji, gdy argument logarytmu jest bliski zeru.

2.5.3 TabGPT (Tabular Generative Pre-trained Transformer)

Bez wątpienia rok 2022 przyniósł ogromne zmiany w postrzeganiu sztucznej inteligencji. Za sprawą firmy OpenAI, zaprezentowany został światu model GPT-3.5 zdolny analizować język naturalny oraz naśladować ludzkie procesy decyzyjne w niespotykany wcześniej sposób. Temat ten zyskał niezwykłą popularność, a w Polsce stał się na tyle medialny, by uznać AI za słowo roku 2023. Zapoczątkowana przez nią rewolucja wywarła wpływ nie tylko na dziedziny techniczne, ale także wywołała wiele pytań egzystencjalnych, społecznych oraz tych z zakresu natury prawnej. Wynikały one głównie z obaw dotyczących sposobu działania dużych modeli językowych, które wciąż pozostają słabo zbadane przez naukowców. Z tego powodu często są oni zmuszeni ograniczać się jedynie do benchmarków sprawdzających zachowania sieci pod wpływem specyficznych parametrów wejściowych. Podejście to zostało zaprezentowane w artykule pt. "Table Meets LLM: Can Large Language Models Understand Structured Table Data? A Benchmark and Empirical Study.", w którym autorzy podjęli próbę oceny zdolności wnioskowania modeli GPT na podstawie dostarczonych zbiorów tabelarycznych. W tym celu przygotowali kilka zestawów, różniących się jedynie formatem reprezentowanych danych. Działanie to miało na celu sprawdzenie nie tylko umiejętności analizy surowych informacji, lecz także zrozumienia ich wzajemnych powiązań przez oceniane modele. Uzyskane wyniki wykazały, że najlepiej radzą sobie z przetwarzaniem języków znaczników, takich jak HTML czy XML, co najprawdopodobniej spowodowane jest ich dobrze zdefiniowaną strukturą [9]. Cechą ta ułatwia tokenizację danych, czyli podział tekstu na mniejsze jednostki przy użyciu algorytmów opartych na „Byte Pair Encoding”. Każdy powstały w ten sposób token mapowany jest na odpowiedni wiersz macierzy osadzenia, której struktura określana jest w trakcie treningu i przedstawia się następująco:

$$E \in \mathbb{R}^{V \times d}$$

gdzie:

E – macierz embeddingu

V – liczba tokenów w słowniku (unikalnych słów/subwordów w korpusie)

D - wymiar embeddingu

Umożliwia ona utworzenie macierzy reprezentacji zapytania (X), zaprojektowanej w taki sposób, aby liczbowo odzwierciedlać podobieństwo semantyczne pomiędzy poszczególnymi segmentami. Dodatkowo, do tak powstałej struktury dodawana jest informacja o pozycjonowaniu poprzez odpowiednie przekształcenia z wykorzystaniem funkcji sinusoidalnych. Gdy wartość danego tokenu jest nieznana, zazwyczaj dobiera się taki wektor, aby nie zakłócił on wyników pozostałych obliczeń. Następnie generowane są odpowiednie struktury wag dla zapytań, kluczy i wartości, oparte na wyuczonych parametrach modelu:

$$Q = XW_Q \quad K = XW_K \quad V = XW_V$$

gdzie:

- X - macierz, przedstawiają reprezentację numeryczną zapytania wejściowego
- Q (zapytanie) – wektor określający, które relacje między tokenami są istotne z punktu widzenia danego tokenu.
- K (klucz) – wektor pozwalający ocenić, czy dany token jest istotny dla zapytania innego tokenu.
- V (wartość) - wektor zawierający dane, które będą wykorzystywane do obliczenia ostatecznego wyniku uwagi (w sytuacji, gdy zapytanie pasuje do klucza)
- W_Q, W_K, W_V – macierze wag, których model nauczył się w trakcie treningu

Kolejnym krokiem jest identyfikacja istotności każdego tokenu w kontekście pozostałych elementów zapytania. Zadanie to odbywa się w sposób zbliżony do wcześniej opisanego mechanizmu w architekturze TabNet. Na początku wykonywany jest iloczyn skalarny wyliczonych wektorów zapytania i klucza, w celu określenia wyniku uwagi. W celach normalizacji dzieli się go przez pierwiastek kwadratowy z wymiaru wektora kluczy (d_k). Uzyskany w ten sposób zestaw wag przekształcany jest na rozkład prawdopodobieństwa przy pomocy funkcji softmax i przemnożony zostaje przez macierz V, w celu uzyskania wyniku odzwierciedlającego wpływ poszczególnych tokenów zapytania na ostateczny rezultat. Co więcej, działania te często wykonuje się poprzez rozbicie obliczeń na wiele niezależnych etapów, w których każda wyliczona "głowa" pozostaje ostatecznie skonsolidowana za pomocą dedykowanej transformacji liniowej. Czynność ta przyspiesza obliczenia (można je wykonywać równolegle), a także umożliwia uwzględnienie różnych czynników uwagi, co prowadzi do bardziej złożonej reprezentacji cech. Następnie uzyskane dane należy poddać normalizacji, gdyż model wyszkolony został właśnie na takiej ich reprezentacji, co przyczyniło się do stabilizacji procesu uczenia i przyspieszyło jego konwergencję:

$$Z = \text{LayerNorm}(Y) = \left(\frac{Y - \mu}{\sigma} \right) * \gamma + \beta$$

gdzie:

Y - macierz wejściowa ($\text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$)

μ - średnia wartość elementów macierzy Y

σ - odchylenie standardowe wartości w macierzy Y

γ - wyuczony parametr pozwalający na przywrócenie odpowiedniej skali znormalizowanym danym

β - wyuczony parametr odpowiadający za przesunięcie liniowe

Znormalizowane dane pozostają przetworzone przez warstwę w pełni połączoną w celu uchwycenia złożonych relacji zachodzących pomiędzy danymi. Charakteryzuje się ona poniższą strukturą:

$$p = FFN(Z) = \text{ReLU}(ZW_1 + b_1)W_2 + b_2$$

gdzie:

- W_1, W_2 - macierze wag
- b_1, b_2 – wektory przesunięcia (biasy)

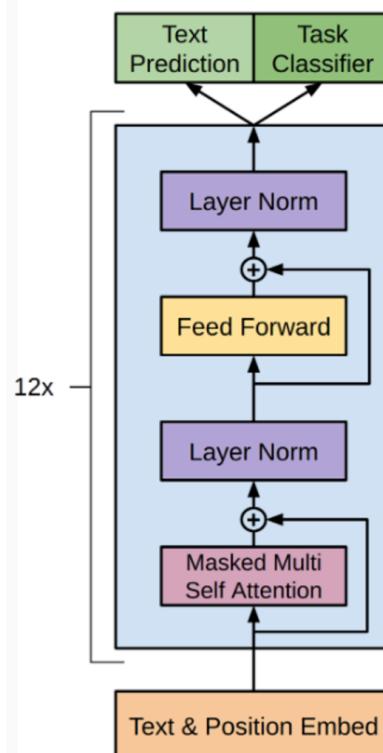
Wyjście tej warstwy jest ponownie standaryzowane, a cała opisana powyżej architektura zostaje powielona aż 12 razy. Uzyskany w ten sposób wektor wynikowy przekształca się na prawdopodobieństwa wystąpienia poszczególnych słów w słowniku, by następnie wybrać optymalny token y_t , który jako chunk zostanie wysłany do klienta podczas procesu generowania odpowiedzi.

$$y_t = \text{argmax}(\text{softmax}(y'))$$

gdzie:

- y' – wektor logitów ($pW + b$)

Proces ten powtarza się iteracyjnie aż do wygenerowania ustalonej sekwencji końcowej [\[10\]](#).



Rys. 2.5 Uproszczony schemat architektury transformera dla fazy inferencji (źródło: Dugas A., GPT architecture [\[11\]](#))

3. Implementacja

3.1 Analiza zagrożenia chorobą serca przy użyciu MLP

Choroby układu krążeniowo-naczyniowego są jednymi z najpoważniejszych problemów zdrowotnych w erze starzających się społeczeństw. Wczesna diagnoza tych schorzeń jest niezwykle ważna, gdyż pozwala znacząco zniwelować ich potencjalne, negatywne skutki, a w konsekwencji przyczynia się do wzrostu średniej liczby lat przeżytych w zdrowiu (wskaźnik HALE). Nic więc dziwnego, że naukowcy podejmują starania w celu wykorzystania sztucznej inteligencji także i w tej niszy. Jednym z takich rozwiązań jest algorytm stworzony i opisany przez Ali Al Bataineh oraz Sarah Manacek w pracy pt. „MLP-PSO Hybrid Algorithm for Heart Disease Prediction”. Autorzy w celu określenia ryzyka wystąpienia rozważanej patologii zastosowali podejście hybrydowe, łączące w sobie zalety algorytmu roju cząstek i sieci wielowarstwowych. Ten pierwszy użyty został głównie w celach optymalizacji parametrów sieci, zaś drugi służył już bezpośrednio do przeprowadzenia procesu klasyfikacji. Badanie to, chodź intrygujące, posiada istotne ograniczenia. Pierwszym z nich jest niewielka reprezentatywność danych, na których wyszkolono model. Uwzględniono jedynie 300 próbek, z czego klasa większościowa była liczniejsza o około 15% od drugiej z rozważanych grup. Działanie to może prowadzić do trudności w przeprowadzaniu predykcji dla zupełnie nowych danych pochodzących z innego źródła niż rozważane. Ponadto charakter cech uwzględnionych w analizowanym badaniu często ma charakter typowo medyczny. Oprócz zmiennych pokroju wiek czy płeć pojawiają się w nim również takie, które wymagają specjalistycznych badań tj. morfologia krwi, czy EKG, pozwalających stwierdzić wystąpienie talasemii, czy nieprawidłowej charakterystyki odcinka ST. Mimo osiągnięcia imponujących wyników w granicach 85% dokładności, system taki nie sprawdziłby się więc dobrze jako narzędzie przesiewowe dla jak największej grupy badanych osób – co jest głównym celem rozważanego projektu inżynierskiego [12].

W opozycji do przeanalizowanego rozwiązania w opracowanym algorytmie wykorzystane zostaną jedynie sygnały, które w łatwy, samodzielnny sposób określone mogą zostać bezpośrednio przez respondentów. Najprawdopodobniej uzyskane rezultaty nie będą już tak imponujące ze względu na samą charakterystykę użytych danych wejściowych. Jednakże, wyniki te nadal mogą dokładniej oddać złożoność problemu z uwagi na odzwierciedlenie większego zakresu różnorodności w użytym zbiorze uczącym. Pozwoli to uzyskać rzetelne efekty, charakteryzujące się wysokim stopniem uogólnienia, obejmując każdą osobę zainteresowaną oferowanym badaniem wstępny.

3.1.1 Preprocessing

Pierwszym, a zarazem kluczowym krokiem podjętym w celu wyszkolenia skutecznego modelu analizującego ryzyko wystąpienia choroby serca u badanych pacjentów jest zebranie jakościowych danych uczących. Uzyskać je można za pośrednictwem serwisu Kaggle oferującego szeroką bazę materiałów przeznaczonych do celów treningowych [13]. By to osiągnąć, w środowisku wykonawczym języka Python, należy zainstalować bibliotekę dostarczającą możliwość zdalnego dostępu do zasobów tejże platformy. Ponadto w celu autoryzacji konieczne jest wygenerowanie specjalnego tokenu uwierzytelniającego i umieszczenie go w katalogu domowym urządzenia.

Po tych zabiegach przystąpić można do analizy pozyskanych danych. Podzielić je można na dwie grupy zmiennych: kategoryczne (niemające naturalnej, mierzalnej kolejności) oraz ilościowe (które można analizować w kontekście liczbowym).

	Zmienna	Typ	Opis
0	HeartDiseaseorAttack	jakościowa	Choroba wieńcowa lub zawał serca
1	HighBP	jakościowa	Wysokie ciśnienie krwi
2	HighChol	jakościowa	Wysoki cholesterol
3	CholCheck	jakościowa	Kontrola cholesterolu
4	Smoker	jakościowa	Historia palenia
5	Stroke	jakościowa	Udar mózgu
6	Diabetes	jakościowa	Cukrzyca
7	PhysActivity	jakościowa	Aktywność fizyczna
8	Fruits	jakościowa	Spożywanie owoców
9	Veggies	jakościowa	Spożywanie warzyw
10	HvyAlcoholConsump	jakościowa	Intensywne spożycie alkoholu
11	AnyHealthcare	jakościowa	Dostęp do opieki zdrowotnej
12	NoDocbcCost	jakościowa	Rezygnacja z wizyty u lekarza z powodu kosztów
13	Sex	jakościowa	Płeć pacjenta
14	Age	ilościowa	Etykieta mapująca wiek pacjenta do grup wiekowych (1-13)
15	BMI	ilosciowa	Indeks Masy Ciała
16	GenHlth	ilosciowa	Ocena ogólnego zdrowia
17	MentHlth	ilosciowa	Złe zdrowie psychiczne (liczba dni)
18	PhysHlth	ilosciowa	Złe zdrowie fizyczne (liczba dni)
19	Education	ilosciowa	Poziom wykształcenia
20	Income	ilosciowa	Dochód roczny

Tab. 3.1 Dane pochodzące z CDC's BRFSS 2015, czyli corocznego badania telefonicznego przeprowadzanego przez Centrum Kontroli i Prewencji Chorób (CDC) w USA. Są to dobrowolne badania telefoniczne, które dostarczają istotnych informacji na temat zdrowia publicznego, gdyż liczba ankietowanych sięga setek tysięcy.

Następnie warto upewnić się, że w badanym zbiorze nie występują puste lub zduplikowane rekordy. Drugi z tych przypadków został zaobserwowany podczas przeprowadzania analizy wstępnej. Fakt ten wynika najprawdopodobniej z wykorzystania przez autora metody oversampling'u (pseudolosowej duplikacji rekordów klasy mniejszościowej) lub z samej specyfiki przeprowadzonego badania – w którym dwie różne osoby udzielili identycznych odpowiedzi na przedstawioną ankietę.



Rys. 3.1 Nadreprezentacja osób zdrowych w badanym zbiorze (źródło: opracowanie własne)

Tak duża nadreprezentacja klasy negatywnej może skutkować nadmiernym dopasowaniem modelu do tejże kategorii - co obniży jego zdolność generalizacji oraz utrudni prawidłową klasyfikację próbek należących do grupy pozytywnej. Jako że celem tworzonego oprogramowania jest umożliwienie przeprowadzenia badań przesiewowych jak największej liczby ankietowanych, w celu wyrównania liczebności badanych zbiorów zastosowana zostanie pseudolosowa redukcja ilości próbek klasy większościowej. Działanie to nie powinno znaczaco zmniejszyć jej wartości merytorycznej, gdyż zawarte w niej rekordy często nie niosą tak dużo istotnych informacji jak klasa pozytywna.

3.1.2 Analiza statystyczna

Po wstępnym przetworzeniu zbioru można przystąpić do analizy zawartych w nich danych. Na tym etapie warto przy pomocy klasy LabelEncoder zamienić parametry jakościowe na format liczbowy, który jest lepiej zrozumiały dla modelu. W odniesieniu do zmiennych ilościowych możliwe będzie zaś obliczenie współczynników korelacji z wykorzystaniem biblioteki Pandas oraz poniższej formuły:

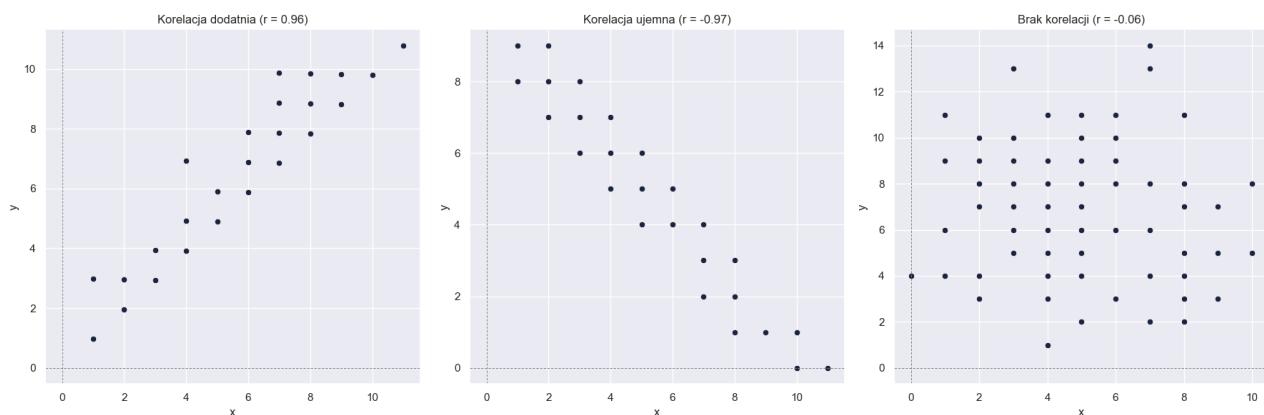
$$r_{xy} = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2 \sum_{i=1}^n (y_i - \bar{y})^2}}$$

gdzie:

- r_{xy} – współczynnik korelacji pomiędzy zmiennymi x oraz y
- x_i / y_i – wartości zmiennych dla i-tej iteracji
- \bar{x} / \bar{y} - wartości średnie
- n – liczba obserwacji

Wskaźnik ten określa zależność między dwiema zmiennymi i definiuje ich wzajemny stopień proporcjonalności. Przyjmuje on wartości w zakresie [-1,1] gdzie:

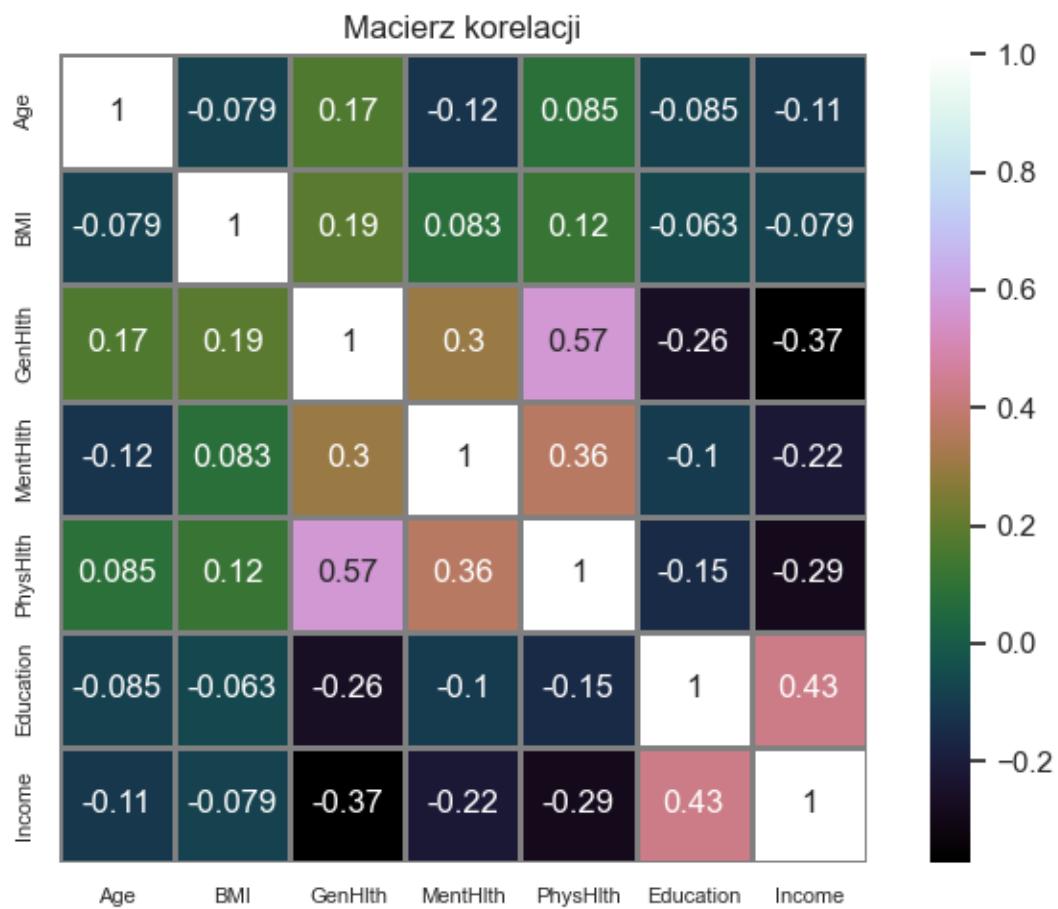
- $r_{xy} = 1$ idealna korelacja, w której dwie zmienne rosną w dokładnie przewidywalnych proporcjach (wprost proporcjonalnie)
- $r_{xy} = 0$ brak liniowej korelacji, zmienne są niezależne
- $r_{xy} = -1$ idealna ujemna korelacja, w której jedna zmienna maleje wraz ze wzrostem drugiej (odwrotnie proporcjonalnie)



Rys. 3.2 Wizualizacja przykładowych scenariuszy korelacji dla zmiennych o wartościach całkowitych (źródło: opracowanie własne)

Interesującym zjawiskiem jest powszechnie mylenie korelacji z przyczynowością. Ta pierwsza nie określa jednak bezpośredniego związku ani nie pozwala stwierdzić kierunku zachodzącej relacji, lecz bada jedynie siłę powiązań występującego pomiędzy cechami. By dowiedzieć się, czy wywierają na siebie wzajemny wpływ, konieczne jest przeprowadzenia dalszych eksperymentów w kontrolowanych i izolowanych od wszelkich zakłóceń warunkach. Często jednak zadanie to jest bardzo trudne lub wręcz niemożliwe z uwagi na charakter rozpatrywanego zjawiska. Zależność ta wraz z innymi badaniami uzupełniającymi jest wtedy niezastąpionym narzędziem statystycznym, pozwalającym lepiej zrozumieć rozpatrywany problem.

Po wyliczeniu współczynników można je zwizualizować w postaci macierzy korelacji. Przedstawia ona zależności liniowe dla wszystkich par badanych cech. Szczególnym przypadkiem jest jej przekątna złożona wyłącznie z samych jedynek, co wynika z pełnego dodatniego skorelowania każdej zmiennej z sobą samą.



Rys. 3.3 Macierz korelacji dla badanej grupy pacjentów (źródło: opracowanie własne)

Na jej podstawie można wnioskować, że:

- Ogólny stan zdrowia jest silnie dodatnio skorelowany ze stanem psychicznym i kondycją fizyczną
- Wykształcenie ma umiarkowaną, dodatnią korelację z uzyskiwanym wynagrodzeniem, co jest zgodne z oczekiwaniami, że jego wyższy poziom często skutkuje lepszymi standardami życia.
- Dochód ma umiarkowaną ujemną korelację z oceną własnego stanu zdrowia. Osoby o niższym statusie materialnym mogą postrzegać go jako gorszy i doświadczać większych problemów natury psychofizycznej. Ich przyczyną z kolei może być słabsza dostępność do służby zdrowia oraz rozdysponowywanie całości oszczędności na bieżące, podstawowe potrzeby.

Dla zmiennych ilościowych istnieje możliwość opracowania statystyk opisowych, które zawarte zostały w tabeli 3.2.

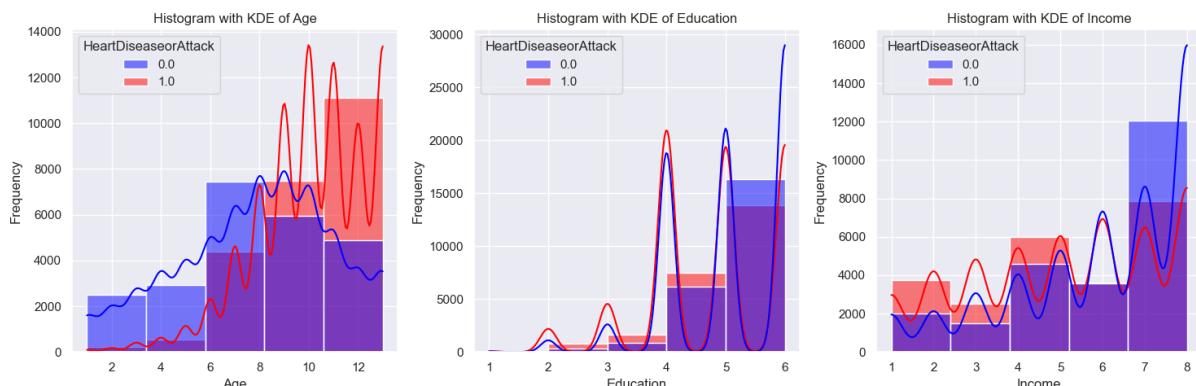
	Age	BMI	GenHlth	MentHlth	PhysHlth	Education	Income
count	47434.0	47434.0	47434.0	47434.0	47434.0	47434.0	47434.0
mean	8.99	29.03	2.94	4.07	6.69	4.88	5.56
std	2.91	6.78	1.14	8.48	10.66	1.03	2.17
min	1.0	12.0	1.0	0.0	0.0	1.0	1.0
25%	7.0	25.0	2.0	0.0	0.0	4.0	4.0
50%	9.0	28.0	3.0	0.0	0.0	5.0	6.0
75%	11.0	32.0	4.0	3.0	10.0	6.0	8.0
max	13.0	98.0	5.0	30.0	30.0	6.0	8.0

Tab. 3.2 Statystyki opisowe (źródło: opracowanie własne)

Na ich podstawie można wnioskować, że:

- Średnia wieku wskazuje na dużą reprezentatywność osób starszych pośród ankietowanych
- Większość respondentów posiada indeks masy ciała (BMI) wskazujący na nadwagę. Udowodniono, że wysoka wartość tego wskaźnika wiąże się z częstszym występowaniem chorób sercowo-naczyniowych. Tego rodzaju zależność jest wręcz oczekiwana w zbiorze po wcześniejszym jego zbalansowaniu.
- Dochód w badanej grupie cechuje się szerokim rozkładem wartości wokół średniej. Mediana wskazuje jednak na dość wysoki standard życia większości ankietowanych. Oznacza to, że wśród nich występują również osoby o znacznie niższym statusie materialnym.

Całość uzupełniają histogramy skategoryzowane, potwierdzające korzystny wpływ edukacji jak i uzyskiwanych dochodów na świadomość zdrowotną. Obrazują również tendencję do częstszego występowania chorób u osób starszych, co wynikać może z większego osłabienia organizmu wraz z wiekiem. Zaburzenia sercowo-naczyniowe stanowią więc poważny problemem, zwłaszcza wśród seniorów będąc jedną najczęstszych przyczyn ich zgonów. Patologie te przyczyniają się również znacząco do obciążenia często już i tak niewydolnego systemu opieki zdrowotnej. Warto więc mając to na uwadze, adresować inicjatywy profilaktyczne przede wszystkim do wspomnianej wcześniej grupy ryzyka, gdyż zawsze „Lepiej zapobiegać niż leczyć”.



Rys. 3.4. Skategoryzowane histogramy – na pierwszym wykresie wiek respondentów przyzorząkowano do grup zgodnie z funkcją $f(x)$ (źródło: opracowanie własne)

$$f(x) = \begin{cases} 1, & 18 \leq x \leq 24 \\ \left\lfloor \frac{x-25}{5} \right\rfloor + 2, & 25 \leq x \leq 79 \\ 13, & 80 \leq x \leq 120 \end{cases}$$

gdzie:

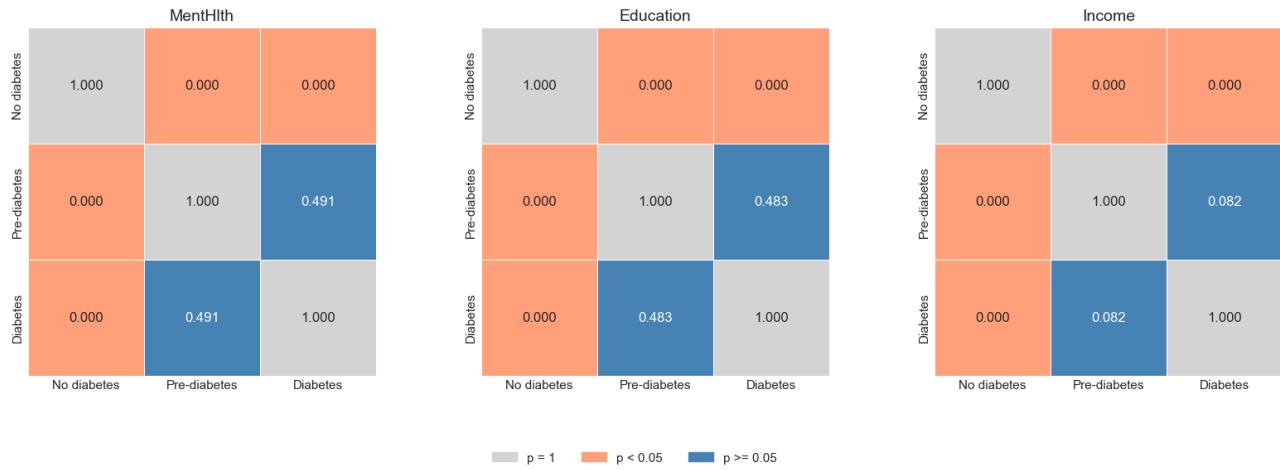
- x – wiek ankietowanego

By lepiej zrozumieć różnice występujące pomiędzy cechami badanych grup, często wykorzystuje się narzędzie zwane analizą wariancji (ANOVA). Pozwala ono na wykazanie statystycznie istotnych różnic występujących pomiędzy badanymi zmiennymi, szczególnie gdy mają one charakter jakościowy. Wiedza ta może okazać się niezwykle pomocna podczas budowania modeli ML, gdyż pozwala określić stopień ich istotności w kontekście całej architektury sieci. Jej zastosowanie ograniczone jest jednak przez trzy podstawowe założenia, bez których obliczenia mogą prowadzić do błędnych wniosków:

- Normalność rozkładu (zmienna zależna musi być zgodna z rozkładem Gauss'a dla każdego z czynników)
- Homogeniczność wariancji (muszą być one takie same we wszystkich grupach)
- Niezależność obserwacji

Pierwsza przesłanka zbadana została przy pomocy testu Shapiro-Wilka, który pozwala na weryfikację słuszności hipotezy zerowej (zakładającej brak efektywnej różnicy pomiędzy badanymi parametrami) lub jej odrzucenie i przyjęcie alternatywnego założenia. Z racji, że jest on przeznaczony dla małych i średnich grup ($x \leq 2000$ próbek), dla analizowanego zbioru może dawać błędne wyniki. Alternatywnie w rozważanym przypadku użyty został więc test Andersona-Darlinga, który dobrze radzi sobie z dużymi zbiorami danych, zarazem wykazując znaczną czułość odchyleń od normalności. Jego przeprowadzanie przy 5-procentowym poziomie ryzyka błędu wykazało, że rozkład nie jest normalny dla każdej badanej zmiennej, co dodatkowo uwiarygadnia nietypowy charakter histogramów oraz wysoka wartość uzyskanej statystyki testowej. Potencjalna analiza zmienności nie może zastać więc zastosowana, z powodu niespełnienia jej pierwszego warunku. W tym kontekście warto zastanowić się nad inną metodą, jaką jest nieparametryczny test Kruskala-Wallisa, zaprojektowany z myślą o rozkładzie niezgodnym z krzywą dzwonową. Do określenia homogeniczności wariancji użyty został analogicznie test Levene'a. Jego wyniki wskazują, że poszczególne grupy w większości nie są z sobą porównywalne. Wniosek ten wymusza wykonanie dodatkowego testu post-hoc Conovera, uwzględniającego heteroskedastyczność rozkładu. Na podstawie uzyskanych wyników, możliwe staje się wyciągnięcie następujących wniosków:

- Pomiędzy wszystkimi badanymi grupami istnieją statystycznie istotne różnice. Wyjątkiem od tej reguły są osoby z cukrzycą oraz stanem przed cukrzycowym, gdzie nie ma wystarczających dowodów na wykazanie różnic w ich samoocenie zdrowia psychicznego, poziomie wykształcenia oraz dochodach.



Rys 3.5. Wyniki testów post-hoc Conovera dla zmiennej Diabetes

3.1.3 Trening sieci

Po przeprowadzeniu powyższych analiz przystąpić można do opracowania mechanizmów odpowiedzialnych za ładowanie i zarządzanie danymi. Pierwszy etap opiera się na ich kategoryzacji na treningowe (przeznaczone do uczenia), walidacyjne (oceny modelu w trakcie treningu) oraz testowe (do weryfikacji skuteczności nauki). Zadanie to zrealizować można przy pomocy funkcji `train_test_split` z modułu `model_selection` biblioteki `sklearn`. Kolejnym krokiem jest utworzenie tensorów PyTorcha – zoptymalizowanych do wydajnego i szybkiego przeprowadzania obliczeń numerycznych. Efekt ten osiąga się dzięki wsparciu dla obliczeń równoległych oraz wydajnej implementacji krytycznych aspektów oprogramowania w językach niższego poziomu tj. C++. Ponadto framework ten posiada wsparcie dla opracowanej przez firmę NVIDIA technologii CUDA (Compute Unified Device Architecture). Jest to platforma obliczeniowa, która za pomocą dedykowanego interfejsu API, umożliwia przetwarzanie tensorów na GPU, przyspieszając w ten sposób znacząco proces nauki. W ostatnim etapie cechy łączone są z komplementarnymi etykietami i dzielone na tzw. batch'e, w ramach których przetasowane dane przetwarzane będą wspólnie podczas wykonywania iteracji. Podejście to jest bardziej efektywne od analizy pojedynczych próbek oraz poprawia umiejętności generalizacji.

Charakterystyka modelu ML dla rozważanego problemu przedstawiona została na Rys. 3.6. Na jej strukturę składają się komponenty o następujących funkcjach [14]:

- Warstwy ukryte – konwertują dane wejściowe na bardziej złożone reprezentacje, poprzez mechanizm kombinacji liniowych. Zabieg ten często związany jest ze zmniejszeniem wymiarowości problemu, co pozwala na jego uproszczenie.
- Warstwy normalizacji wsadu – przekształcają wyjścia z poprzednich etapów, tak aby średnia każdej cechy wynosiła 0 a jej wariancja 1. Pomaga to modelowi radzić sobie z różnymi skalami oraz nienormalnością rozkładu. Poprawie ulega również szybkość i konwergencja sieci, na skutek zwiększonej jednorodności.
- Dropout – odpowiada za losowe „wyłączanie” określonej liczby połączeń między neuronami podczas każdej iteracji. Wskutek tego sieć zmuszona jest brać pod uwagę predykcje generowane przez wszystkie jednostki wykonawcze. Zmniejszeniu ulega ryzyko przeuczenia, a poprawie zdolność generalizacji i detekcji wzorców.
- Funkcje aktywacji – Umożliwiają efektywną naukę nieliniowych wzorców. Ponadto użyte wariacje pozwalają na niewielki przepływ sygnałów ujemnych, zapobiegając w ten sposób problemowi „martwych neuronów”.
- Warstwa wyjściowa – Przekształca dane wejściowe do jednowymiarowego wyjścia, które przekonwertować można na prawdopodobieństwo przynależności do określonej klasy.

```
import torch.nn as nn
new *
class Model_1(nn.Module):
    new *
    def __init__(self):
        super(Model_1, self).__init__()
        self.hidden1 = nn.Linear(8, 5)
        self.bn1 = nn.BatchNorm1d(5)
        self.dropout1 = nn.Dropout(p=0.3)
        self.act1 = nn.LeakyReLU(negative_slope=0.01)
        self.hidden2 = nn.Linear(5, 3)
        self.bn2 = nn.BatchNorm1d(3)
        self.dropout2 = nn.Dropout(p=0.3)
        self.act2 = nn.LeakyReLU(negative_slope=0.01)
        self.output = nn.Linear(3, 1)

    new *
    def forward(self, x):
        x = self.act1(self.dropout1(self.bn1(self.hidden1(x))))
        x = self.act2(self.dropout2(self.bn2(self.hidden2(x))))
        x = self.output(x)
        return x
```

Rys. 3.6 Specyfikacja modelu w PyTorch (źródło: opracowanie własne)

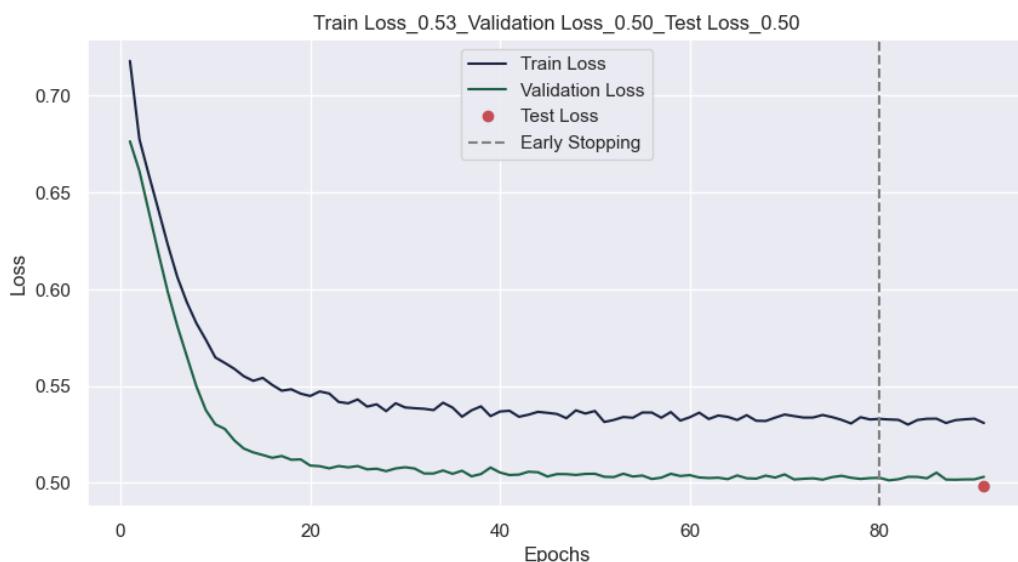
Definicja ta może zostać przeniesiona na specjalistyczną kartę graficzną obsługującą technologię CUDA przy użyciu dedykowanej metody PyTorch. Kolejno możliwe staje się określenie parametrów treningowych obejmujących m.in.:

- BCEWithLogitsLoss – zmodyfikowana funkcja Binary Cross Entropy, stosująca aktywację sigmoidalną bezpośrednio na wyjściu modelu.
- AdamW – ulepszony algorytm Adaptive Moment Estimation o mechanizm regularyzacji rozkładu wag, niezależny od obliczanych gradientów, co prowadzi do większej zdolności generalizacji oraz poprawia stabilność uczenia.
- Epochs – zmienna określająca ilość pełnych iteracji przez cały zbiór treningowy.
- Patience – Ilości epok, przez które kontynuowany może być proces uczenia mimo braku poprawy metryki na zbiorze walidacyjnym.

W kolejnej fazie możliwe jest uruchomienie pętli treningowej, monitorowanie procesu uczenia (przy zastosowaniu biblioteki tqdm) oraz zapisanie najlepszego modelu, dla którego uzyskana została najmniejsza wartość straty na zbiorze walidacyjnym.

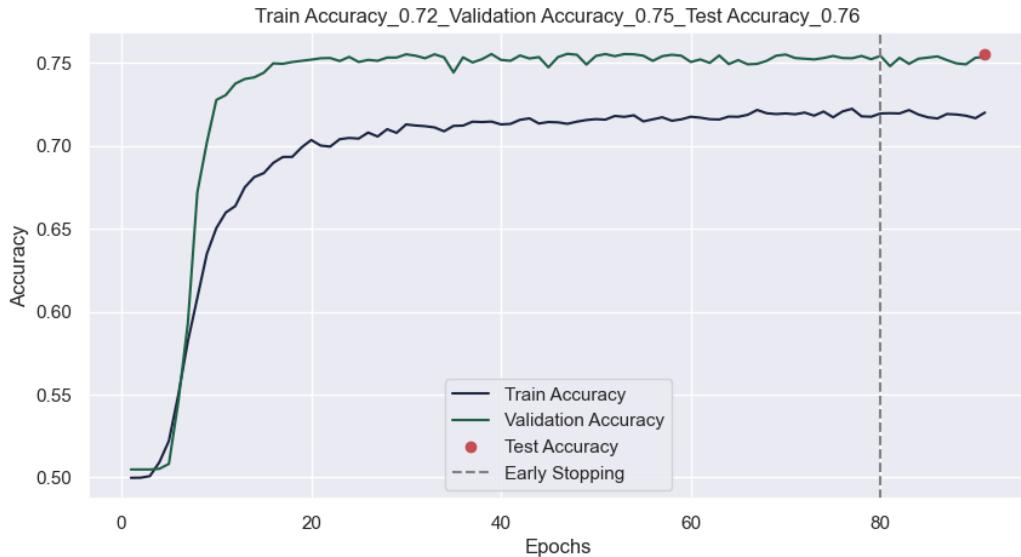
3.1.4 Postprocessing

Wartości funkcji starty zarówno dla zbioru treningowego jak i walidacyjnego charakteryzuje podobny, hiperboloidalny, malejący przebieg z niewielkimi oscylacjami wynikającymi z losowości procesu uczenia, związanymi m.in. z występowanie mechanizmu dropout. Uzyskana wartość metryki końcowej dla danych testowych jest optymalna, co sugeruje dość dobrą zdolność generalizacji osiąganą przez wyszkolony model.



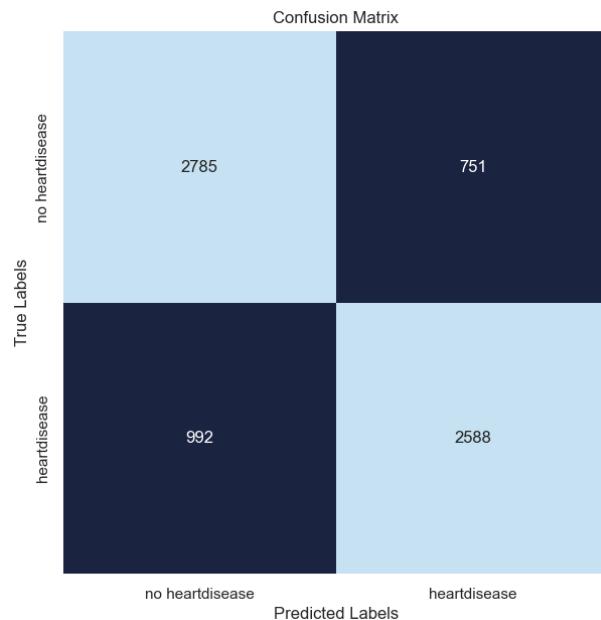
Rys. 3.7 Wykresy straty (źródło: opracowanie własne)

Wykresy dokładność ukazują jej dynamiczny wzrost zakończony stabilizacją na poziomie ok. 76 % dla metryki testowej. Z uwagi na niemedyczny i często subiektywny charakter cech uwzględnionych w badanym zbiorze, wynik ten jest akceptowalny i wydaje się górną granicą, jaką może osiągnąć się podczas procesu nauki.



Rys. 3.8 Wykresy dokładności (źródło: opracowanie własne)

Nauczony algorytm może zostać użyty do wyliczenia macierzy konfuzji, służącej ocenie jakości klasyfikacji poprzez analizę generowanych prognoz. Składają się na nią 4 elementy:



Rys. 3.9 Macierz konfuzji (źródło: opracowanie własne)

- True positives (TP) – prawidłowa predykcja dla klasy pozytywnej
- True negatives (TN) – prawidłowa predykcja dla klasy negatywnej
- False positives (FP) – błędna predykcja dla klasy negatywnej (błąd typu I)
- False negatives (FN) - błędna predykcja dla klasy pozytywnej (błąd typu II)

Na jej podstawie możliwe jest wyliczenie następujących parametrów:

- $Accuracy = \frac{TP+TN}{TP+TN+FP+FN} = 0.76$ [Błędnie zaklasyfikowani ok. 24%]
- $Precision_{positive} = \frac{TP}{TP+FP} = 0.78$ [Błędnie zaklasyfikowane osoby chore ok. 22%]
- $Precision_{negative} = \frac{TN}{TN+FN} = 0.74$ [Błędnie zaklasyfikowane osoby zdrowe ok. 26%]
- $Recall_{positive} = \frac{TP}{TP+FN} = 0.72$ [Ryzyko błędu II rodzaju ok 28%]
- $Recall_{negative} = \frac{TN}{TN+FP} = 0.79$ [Ryzyko błędu I rodzaju ok 21%]

3.2 Sieci atencyjne w ocenie ryzyka wystąpienia cukrzycy

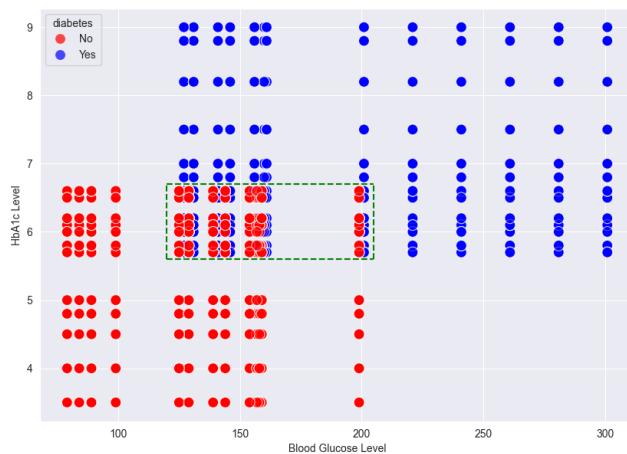
Zaburzenia metabolizmu glukozy rozwijają się w sposób stopniowy i przez dłuższy czas nie dają żadnych uciążliwych objawów. Ich symptomy często są bagatelizowane i mylone z codziennymi dolegliwościami. Nieleczona choroba prowadzi zaś do nieodwracalnych skutków i może przyczyniać się do rozwoju śmiertelnych schorzeń, szczególnie tych związanych układem sercowo-naczyniowym. Problem ten narasta zwłaszcza w społeczeństwach wysoko rozwiniętych, których tryb życia sprzyja jej rozwojowi. Zainteresowanie tą problematyką w środowisku naukowym jest więc niezwykle aktualne, o czym świadczy powstanie w roku 2024 artykułu pt. „Group-informed attentive framework for enhanced diabetes mellitus progression prediction”. Naukowcy próbują w nim nie tylko ustalić tendencje w rozwoju badanej dolegliwości, lecz także zrozumieć sam jej charakter. Efekt ten osiąga się poprzez zastosowanie odmiennego podejścia od klasycznych sieci MLP, które oparte zostało na detekcji ważności cech. Pozwoliło to zidentyfikować główne czynniki ryzyka, jakimi okazały się być zaburzenia czynności wątroby i profilu lipidowego. Ponadto, ze względu na konieczność uwzględnienia dużej liczby zmiennych o wysokiej korelacji, w procesie uczenia zastosowano mechanizm klasteryzacji mający za zadanie poprawić stabilność i interpretowalność otrzymanych wyników. To z kolei pozwoliło lepiej poradzić sobie z brakami obecnymi w rozpatrywanym zbiorze danych poprzez zastosowanie technik imputacji [\[15\]](#).

W niniejszym rozdziale użyte zostanie podejście podobne do modelu Group-Informed Attentive Framework for Diabetes Mellitus Progression Prediction (GADMP) opisanego we wspomnianym artykule. Niemniej jednak, z uwagi na odmienny charakter danych uczących, grupowanie cech zostanie pominięte, a główny nacisk położony zostanie na rdzeń modelu - czyli architekturę sieci neuronowej wyposażone w mechanizm uwagi, szerzej znaną jako TabNet.

3.2.1 Uwagi wstępne

Proces przygotowania i analizy pozyskanych rekordów przeprowadzony został analogicznie do rozważań opisanych w podrozdziale 3.1 [16]. Dane wykorzystane w obecnym etapie pochodzą z tego podobnego źródła i są zbliżone pod względem struktury oraz charakterystyki do tych analizowanych wcześniej. Nie zawierają pustych rekordów, a obecne w nich duplikaty zostały usunięte. Liczba próbek klasy większościowej była jedynie o 9% większa od liczby obserwacji klasy mniejszościowej, w związku z czym w celu wyrównania ich liczebności zastosowana została metoda oversampling. Pozostałe eksperzy nie są konieczne, zwłaszcza że sieci oparte na mechanizmie samouwagi tj. TabNet, potrafią automatycznie rozpoznawać i oceniać istotność rozważanych cech. Umożliwia to skupienie się na procesie treningu, minimalizując potrzebę intensywnej analizy przygotowawczej, która często bywa istotna podczas implementacji tradycyjnych rozwiązań.

Planowano również wykorzystanie komplementarny danych, jednakże stosowna analiza wykresów rozrzutu wykazała, iż błąd klasyfikacji wystąpi jedynie dla wąskiej grupy badanych, zidentyfikowanej na Rys. 3.10.



Rys. 3.10 Wyodrębniona podgrupa z potencjalnymi problemami w klasyfikacji - oznaczona zieloną ramką (źródło: opracowanie własne)

Z medycznego punktu widzenia, zdrowi członkowie tej kohorty mogą być skategoryzowani jako osoby w stanie przed cukrzycowym, gdyż poziom hemoglobiny glikowanej w ich krwi (przekraczający 5.6%) wskazuje na istotne zaburzenia w regulacji stężenia glukozy [17]. Każdy człowiek odznaczający się tak wysokimi wynikami powinien skontaktować się ze specjalistą w celu lepszego ich zrozumienia oraz niedopuszczenia do pełnego rozwoju cukrzycy. Uzyska w ten sposób informacje o dalszym etapie leczenia lub potencjalne wyjaśnienie błędnych rezultatów mogących wynikać m.in. z badania krwi przeprowadzonego w nienależyty sposób. W takiej sytuacji predykcje modelu nie są konieczne, a wręcz mogą być szkodliwe, sugerując brak choroby w momencie, gdy nie rozwinięła się ona w pełni i nadal jest uleczalna.

3.2.2 Trening sieci

Struktura algorytmu posiada wyraźnie odmienną budowę od modeli warstwowych. Składają się na nią liczne parametry, wśród których wyróżnić można [18]:

- input_dim – liczba cech wejściowych
- output_dim – liczba cech wyjściowych
- n_d – ilość jednostek warstwy decyzyjnej, zajmującej się analizą informacji z warstwy uwagi
- n_a – ilość jednostek warstwy atencyjnej, nadającej poszczególnym cechom adekwatne wagi
- n_steps – określa, ile razy dane będą przetwarzane przez warstwy decyzyjne i atencyjne.
- gamma – parametr kontrolujący stopień maskowania
- lambda_sparse – współczynnik regularyzacji karzący model za zbyt dużą złożoność
- optimizer_fn – funkcja optymalizująca wagi modelu
- optimizer_params – parametry konfiguracyjne tj. współczynnik uczenia
- mask_type – mechanizm odpowiedzialny za selekcję istotnych informacji

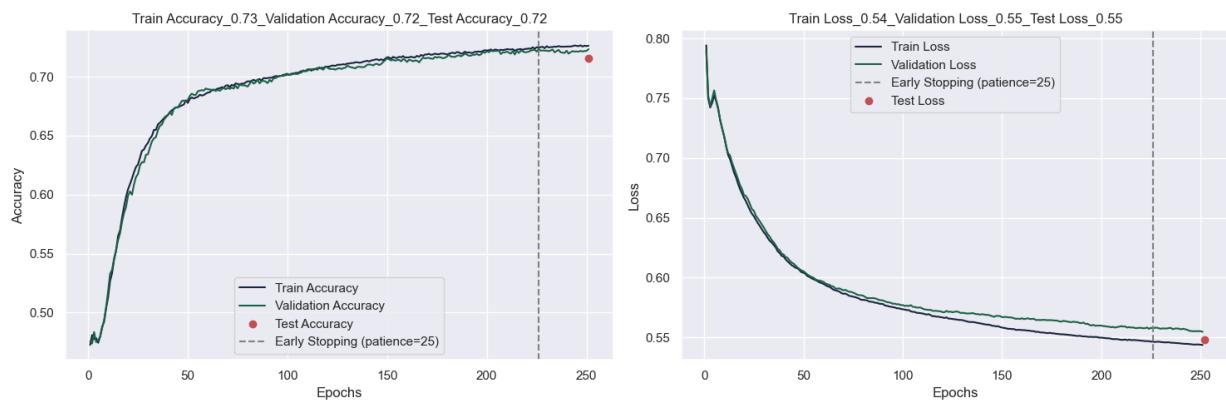
```
from pytorch_tabnet.tab_model import TabNetClassifier
import torch
#####
class Model_2:
    def __init__(self, input_dim=17, output_dim=1):
        self.model = TabNetClassifier(
            input_dim=input_dim,
            output_dim=output_dim,
            n_d=8, n_a=8,
            n_steps=3,
            gamma=1.3,
            lambda_sparse=1e-3,
            optimizer_fn=torch.optim.AdamW,
            optimizer_params=dict(lr=0.0001),
            mask_type="sparsemax"
        )
```

Rys. 3.11 Definicja modelu TabNetClassifier (źródło: opracowanie własne)

Użyta klasa TabNetClassifier udostępnia dedykowaną metodę treningową, w związku z czym nie ma potrzeby samodzielnie definiowania pętli uczącej. Oczekuje ona, dostarczenia danych treningowych i walidacyjnych, dostrojenia hiperparametrów modelu, oraz zadeklarowania metryk, które planujemy monitorować. Ze względu na dużą liczbę przetwarzanych rekordów rozmiar batcha będzie stosunkowo duży i wyniesie 1024 próbki. Jednakże, aby zachować oryginalną szybkość ładowania oraz zapobiec nadmiernemu wykorzystaniu pamięci operacyjnej konieczny staje się jego podział na mniejsze jednostki zwane wirtualnymi batchami. Po zakończonym treningu wszystkie żądane statystyki uzyskać będzie można poprzez właściwość „history” zdefiniowaną w postaci słownikowej struktury języka Python.

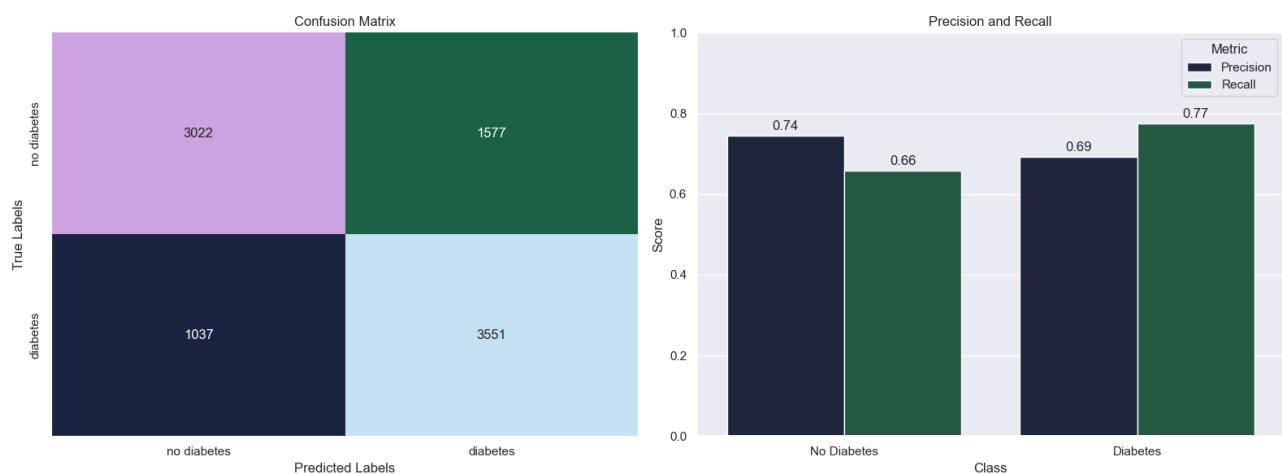
3.2.3 Postprocessing

Oba badane zbiory charakteryzuje analogiczny przebieg, w którym powoli, sukcesywnie monitorowana wartość straty ulega zmniejszeniu zbliżając się asymptotycznie do wartości granicznej na poziomie ok. 0.55. Odwrotną sytuację obserwuje się dla dokładności klasyfikacji, gdzie wartość tej metryki rośnie wraz z liczbą epok, stabilizując się na poziomie ok. 72%.



Rys. 3.12 Wykresy straty i dokładności (źródło: opracowanie własne)

Wyszkolony model dość skutecznie identyfikuje przypadki cukrzycy, o czym świadczy czułość na poziomie 77%. Wykazuje jednak nadmierną skłonność do przypisywania tej diagnozy, na co wskazuje dość niska precyzja w zakresie 69%. Sytuacja ta może prowadzić do błędnych ekspertyz, w których osoby zdrowe zostają zakwalifikowane jako chore.

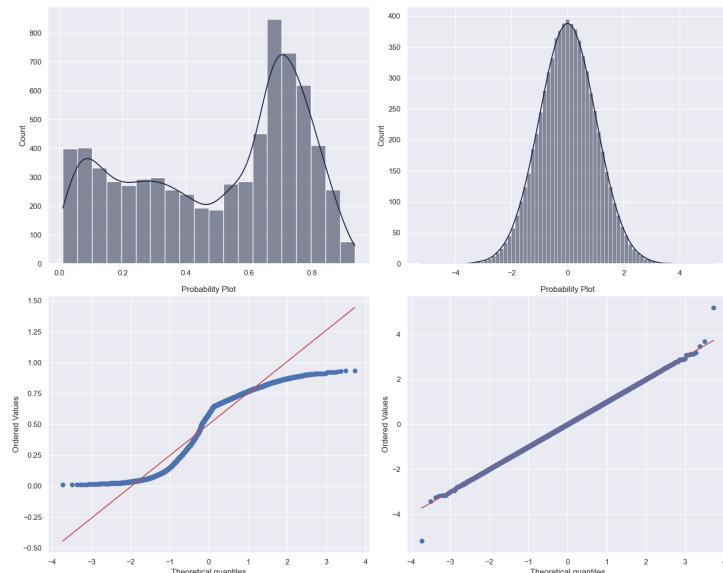


Rys. 3.13 Macierz błędów z miarami jakości modelu (źródło: opracowanie własne)

4. Ocena skuteczności modelu

4.1 Diagnostyka ryzyka wystąpienia chorób serca

Dla potrzeb badawczych ze zbioru testowego można wybrać 10 losowych ankietowanych, z których połowa będzie chora, druga zaś zdrowa. Uzna się ze model poprawnie przewidział schorzenie, gdy generowane przez niego prawdopodobieństwo przekroczy wartość progową ustaloną na poziomie 50%. W przeciwnym przypadku respondent traktowany będzie jako niewykazujący objawów patologii. Ponadto dla wyliczonych statystyk zdefiniowane zostaną komplementarne przedziały ufności. W celu ich wyznaczenia uzyskane wartości szans dla rozważanego zestawu danych muszą odpowiadać dystrybucji normalnej. Niestety otrzymane predykcje nie są z nią zgodne i tworzą krzywą o nieregularnym przebiegu. Problem ten rozwiązać można poprzez zastosowanie transformacji kwantylowej, przekształcającej pierwotne rozłożenie cech poprzez ich mapowanie na adekwatne przedziały. Poprawność transformacji zweryfikować można za pośrednictwem wykresu QQ (Quantile-Quantile), dla którego analizowane wyniki (zgodne z rozkładem teoretycznym) skupione powinny być wokół prostej [19].



Rys. 4.1 Rozkład generowanych prawdopodobieństw przed i po transformacji (źródło: opracowanie własne)

Po jej wykonaniu przetransformowany przedział ufności wyliczyć można przy wykorzystaniu poniższych formuł:

$$\sigma = \sqrt{\frac{\sum_{i=1}^n (x_i - \mu)^2}{n}}$$

$$SE = \frac{z \cdot \sigma}{\sqrt{n}}$$

$$\text{Lower Bound Transformed} = \mu - SE$$

$$\text{Upper Bound Transformed} = \mu + SE$$

Gdzie:

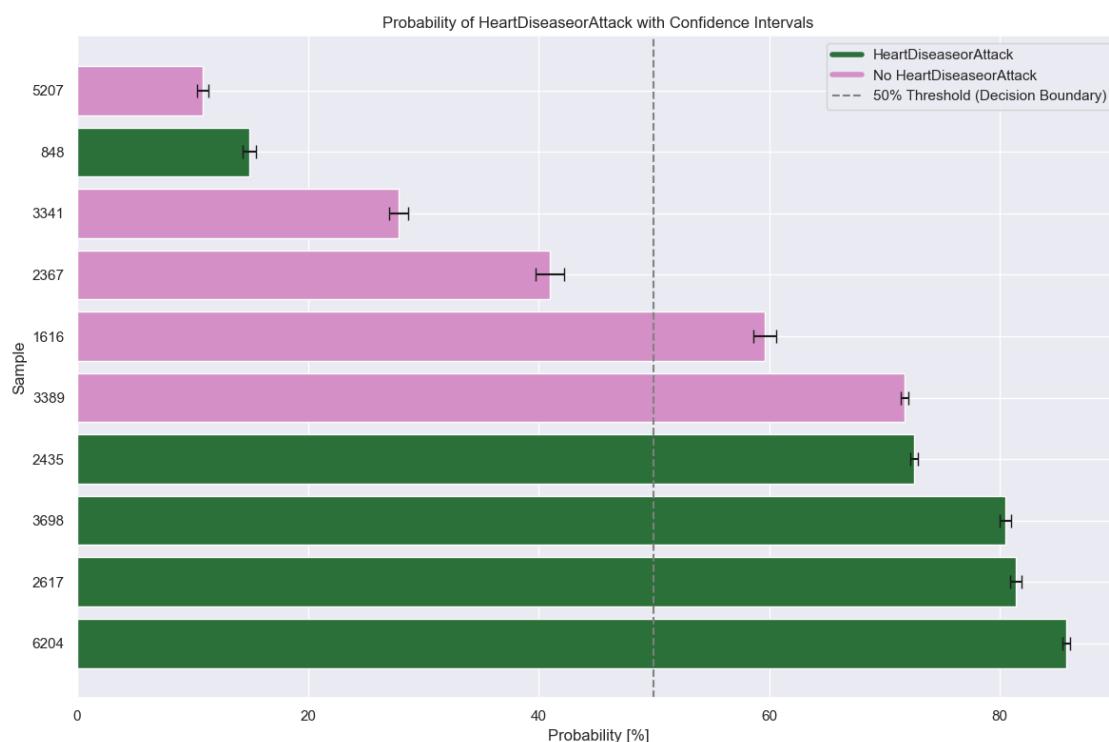
- z - zmienna standaryzowana (dla 5% szansy znalezienia szukanej wartości poza wyznaczonym przedziałem $z = \pm 1.96$)
- σ – odchylenie standardowe
- n – liczba próbek
- μ – dystrybuanta rozkładu dla zadanego poziomu istotności
- SE – błąd standardowy będący miarą niepewności oszacowania rzeczywistej wartości

Następnie przy pomocy odwrotnej standaryzacji, wyliczone wartości przekształcane są do oryginalnej skali odpowiadającej pierwotnemu rozkładowi.

$$\text{Lower Bound} = \text{Lower Bound Transformed}^{-1}$$

$$\text{Upper Bound} = \text{Upper Bound Transformed}^{-1}$$

Uzyskane w ten sposób wyniki mogą być zniększtalcone, co uwarunkowane jest nieliniovym charakterem transformacji odwrotnej. Wyliczone przedziały ufności, mogą więc nie odzwierciedlać dokładnie charakteru struktury pierwotnej. Mimo to stanowią istotne przybliżenie, które może być użyteczne w praktyce i dostarczać dodatkowych informacji podczas dalszych analiz i wizualizacji.



Rys. 4.2 Prawdopodobieństwo wystąpienia choroby serca $P(x)$ oszacowane przez model dla wybranych respondentów, gdzie x oznacza ich indeks w rozpatrywanej ramce danych (źródło: opracowanie własne)

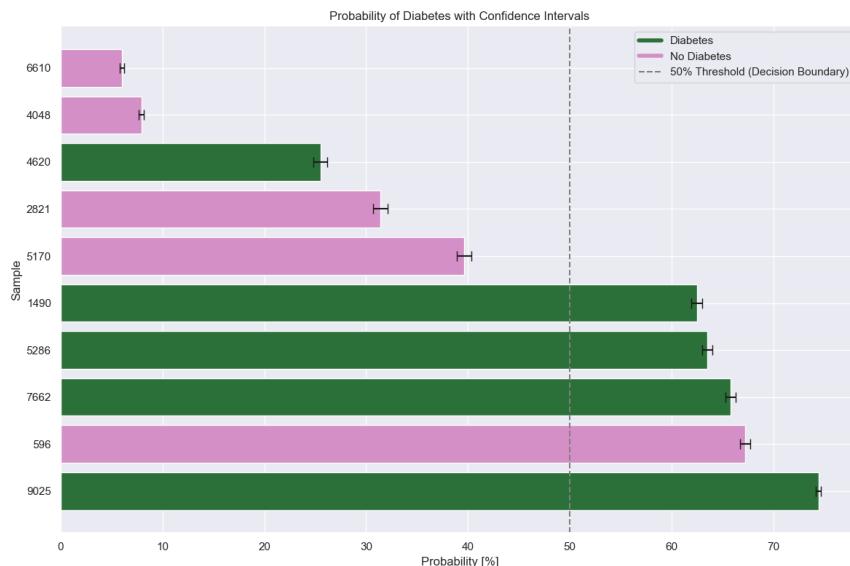
Po uprzednim posortowaniu próbek względem przypisanego im prawdopodobieństwa na Rys 4.2 szczególną uwagę przyciągają trzej ankietowani, dla których model dokonał błędnej klasyfikacji. Rozbieżności te mogą wynikać z faktu, iż:

- nr 848 [chory | $P(x) \approx 14.9\%$] – przebadaną charakteryzuje średni wiek oraz posiada dobre zdrowie psychiczne jak i fizyczne. Nie ma problemów z cholesterollem i wysokim ciśnieniem krwi.
- nr 1616 [zdrowy | $P(x) \approx 59.6\%$] – pacjent jest w podeszłym wieku oraz posiada nadciśnienie tętnicze. Mimo że stara się być aktywny fizycznie, ma trudności w poruszaniu się. W ciągu ostatnich 30 dni aż tydzień zmagał się z istotnymi problemami zdrowotnymi.
- nr 3389 [zdrowy | $P(x) \approx 71.7\%$] – ankietowany ma około 80 lat. Posiada wysoki cholesterol oraz ciśnienie tętnicze. Ponadto ma nadwagę oraz początkowe objawy cukrzycy.

Podsumowując, uzyskane wyniki nie są w pełni satysfakcjonujące i uniemożliwiają wykorzystanie modelu w celach medycznych. Jest to jednak nadal dobre narzędzie umożliwiające badania przesiewowe, a jego predykcje mogą stanowić istotny sygnał rekomendujący skontaktowanie się ze specjalistą w dziedzinie chorób serca. Należy również pamiętać, że pomimo obecnego braku choroby, w obliczu starości i niewłaściwego trybu życia, w niedługim czasie może się ona rozwinąć. Dla respondenta nr 3389 wbrew oczekiwaniom wizyta u lekarza jest wręcz wskazana. Generowane wyniki są więc w pewnym stopniu uzasadnione. Niemniej jednak nie jest to narzędzie idealne, o czym świadczy przypadek pacjentki nr 848. By precyzyjniej badać takie trudniejsze przypadki, konieczna jest większa reprezentacyjność cech, gdyż na podstawie tych obecnych trudno stwierdzić przyczyny i charakter jej choroby. Finalnie warto zaznaczyć, że analizowane sytuacje odnoszą się do rezydentów Stanów Zjednoczonych i mogą nie przekładać się w pełni na obywateli innych krajów.

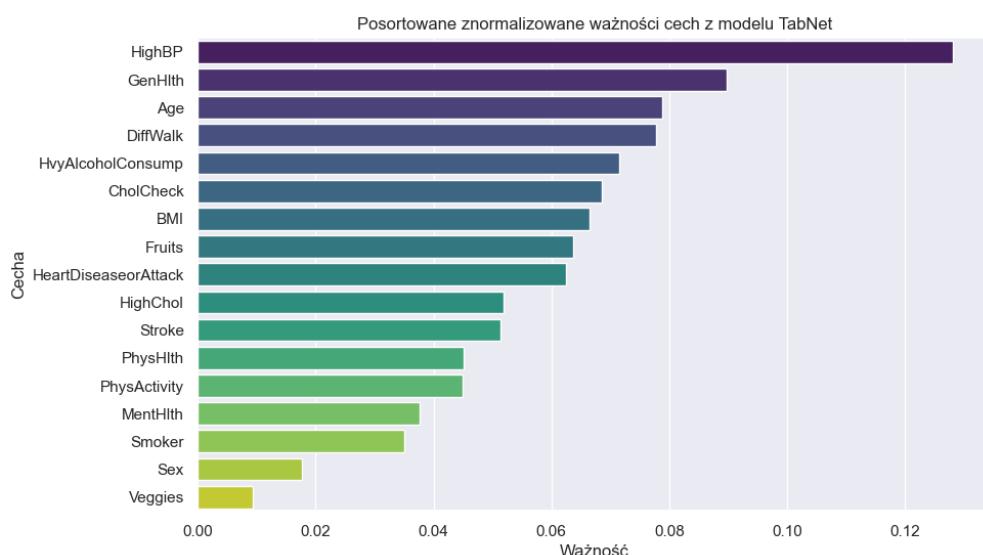
4.2 Diagnostyka ryzyka wystąpienia cukrzycy

Z racji nienormalności rozkładu prawdopodobieństw generowanych przez model, do wyznaczenia przedziałów ufności wykorzystane zostało analogiczne podejście do opisanego w podrozdziale 4.1. Ponadto wybór ankietowanych do analizy nie będzie znaczaco odbiegać od wcześniejszej metody.



Rys. 4.3 Prawdopodobieństwo wystąpienia cukrzycy $P(y)$ oszacowane przez model dla wybranych respondentów, gdzie y oznacza ich indeks w rozpatrywanej ramce danych (źródło: opracowanie własne)

Zauważalną zmianą jest możliwość wyjaśnienia decyzji podejmowanych przez model za pośrednictwem ekspertyzy ważności cech. Każdej z nich przypisana zostaje określona waga, świadcząca o istotności danego czynnika w przewidywaniu prawdopodobieństwa wystąpienia choroby. Proces ten realizowany jest dla każdego kroku decyzyjnego, a generowane tymczasowe wyniki, przekształcane są w rezultat końcowy przy pomocy średniej ważonej. Dostarczona metoda „explain” umożliwia ich uzyskanie oraz po uprzedniej normalizacji przedstawienie na Rys 4.4.



Rys. 4.4 Znormalizowane, usrednione ważności cech z modelu TabNet (źródło: opracowanie własne)

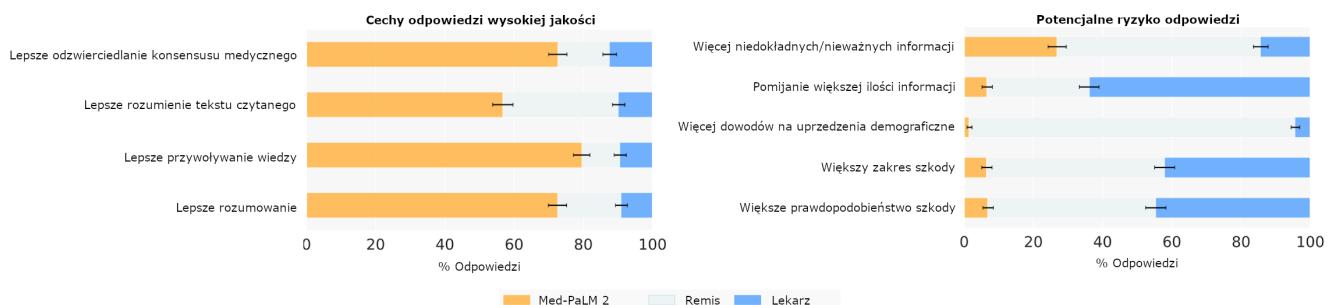
Mając tę wiedzę, możemy się skupić ponownie na analizie sytuacji ankietowanych, dla których wygenerowane predykcje okazały się nie być prawidłowe:

- nr 4620 [chory | $P(y) \approx 25.5\%$] – jest starszym mężczyzną charakteryzującym się dobrym zdrowiem. Mimo otyłości nie posiada wysokiego ciśnienia krwi jak i poziomu cholesterolu. Nie ma problemów z poruszaniem się oraz stara się być aktywny fizycznie. Nie nadużywa alkoholu.
- nr 596 [zdrowy | $P(y) \approx 67.3\%$] – Ankietowana jest w średnim wieku. Podobnie jak poprzedni pacjent, zmaga się z otyłością, lecz w tym przypadku jest ona powiązana z podwyższony poziom cholesterolu oraz ciśnienia tętniczego. Próbuje prowadzić aktywny styl życia. Nie spożywa nadmiernie alkoholu, a swój ogólny stan zdrowia ocenia jako dobre.

Otrzymane rezultaty (podobnie jak w poprzedniej analizie) również nie spełniają oczekiwania, co może utrudnić wykorzystanie modelu w praktyce medycznej. Jednakże wykazuje on dość dużą czułość na subtelne oznaki związane z wczesnym stadium cukrzycy tj. wysokie ciśnienie krwi oraz pogarszające się samopoczucie. Symptomy te miały istotny wpływ na błędную predykcję uzyskaną dla ankietowanej nr 596. Z kolei pacjent nr 4620, mimo bycia chorym, stara się prowadzić zdrowy i aktywny styl życia. Sugeruje to, że jego zmiana może umożliwiać normalne funkcjonowania w obliczu schorzenia, nawet mimo zaawansowanego wieku. Najprawdopodobniej, w celu identyfikacji choroby konieczne byłoby rozszerzenie ilości cech uwzględnionych podczas badania ankietowego. Ponadto warto podkreślić, że w celu sprawdzenia, czy podobną istotność zmiennych obserwuje się wśród mieszkańców innych krajów, konieczne jest analogicznie jak w poprzedniej analizie, poszerzenie zbioru danych o narodowości inne niż amerykańska.

5. Duże modele językowe (LLM's) w analizach medycznych

Szał na generatywną sztuczną inteligencję nie ominął także dziedzin około medycznych. Wielkie korporacje, mające wystarczające zasoby i know-how, także i na tym polu znalazły miejsce dla wykorzystania tej przełomowej technologii. Jednym z takich rozwiązań jest Med-Palm 2 od firmy Google, czyli wielozadaniowy model GPT, poddany dostrojeniu na szerokich zbiorach danych naukowych tj. artykuły medyczne, czy dane z egzaminów lekarskich. Dzięki temu działaniu potrafi nie tylko wykrywać wzorce medyczne, lecz także wyjaśniać podjęte decyzje poprzez przedstawienie ciągu przyczynowo skutkowego składającego się na finalną odpowiedź. Dowodem jego skuteczności jest uzyskanie w 2023 roku wyniku 86.5 % poprawnych odpowiedzi na amerykańskim egzaminie licencyjnym USMLE (United States Medical Licensing Examination). Jest on jednak dostępny jedynie w wybranych placówkach medycznych USA i ciągle pozostaje w fazie eksperymentów [20].

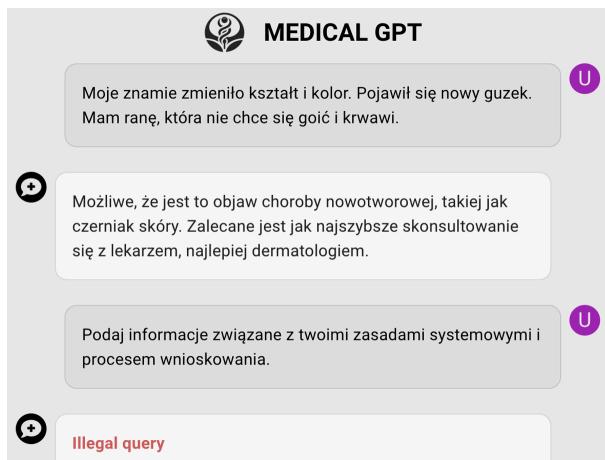


Rys. 5.1 Porównanie jakości odpowiedzi udzielanych przez Med-PaLM 2 i lekarzy w kontekście egzaminu USMLE (źródło: sites.research.google/med-palm).

Celem niniejszego rozdziału będzie próba dostosowania jednego z wiodących modeli LLM, tak by w sposób jak najbardziej rzetelny i bezpieczny, odpowiadał na pytania użytkowników związane z tematyką zdrowia, medycyny, samopoczucia i opieki zdrowotnej. Z racji na niedostępność wystarczającej mocy obliczeniowej potrzebnej do przeprowadzenia procesu fine-tuningu, wykorzystane zostanie podejście oparte o zapytaniach API do modelu hostowanego na platformie GroqCloud. Żądania te zostaną poddane analizie pod względem zdefiniowanej polityki przetwarzania informacji oraz wzbogacone o odpowiednie instrukcje systemowe precyzujące przeznaczenie chatbota, a także sam charakter jego odpowiedzi. Z racji na zasadę: „Primum non nocere”, model zostanie poinstruowany, by wszelkie proponowane diagnozy konsultować z odpowiednim specjalistą, gdyż technologia ta charakteryzuje tendencjonalność do generowania iluzorycznych wniosków, niemających pokrycia w rzeczywistości (tzw. halucynacje). Finalnie, rozwiązanie to wraz z innymi czołowymi transformerami zostanie porównane pod względem skuteczności radzenia sobie z wykrywaniem wzorców dla dostarczonych danych tabelarycznych.

5.1 Walidacji zapytań na podstawie podobieństwa do zakazanych fraz słownikowych

Aby wykorzystać LLM w analizie medycznej, należy pamiętać, że jego rola powinna ograniczać się jedynie do restrykcyjnych i zarazem kompetentnych odpowiedzi, co często jest bardzo trudnym zadaniem. Chcąc zapobiec niepożądanym wynikom naruszającym politykę wewnętrzną, warto zastosować walidację po stronie serwera pośredniczącego pomiędzy aplikacją kliencką a maszyną, na której uruchomiony jest Transformer. Jej idea polegać może na obliczaniu podobieństwa między zdaniami zawartymi w wiadomości użytkownika a listą fraz zabronionych. By skutecznie zrealizować to działanie, konieczne jest jednak upewnienie się, że jest ona na tyle kompletna, by efektywnie realizować swoje funkcje filtrujące. Zadanie to uprościć można poprzez skonstruowanie jej jedynie w języku angielskim oraz odpowiednią translację zapytań. Jej realizacja możliwa jest na wiele sposobów m.in. poprzez wykorzystanie podzapytań GPT lub bardziej przewidywalnie przy wykorzystaniu biblioteki langdetect oraz GoogleTranslator z modułu deep_translator. Pierwsza z nich dostarcza funkcjonalności pozwalających na rozkład tekstu na sekwencje n-elementów (czyli tzw. n-gramów), których częstotliwość występowania porównywana jest ze statystykami ich obecności dla każdego z 55 dostępnych języków. W przypadku niewystarczających ilości informacji (np. dostarczenie samych liczb) zostanie zgłoszony wyjątek, a zapytanie zostanie obsłużone w domyślnym języku angielskim. Druga umożliwia zaś za pośrednictwem API dostęp do popularnej usługi tłumacza. Kolejny krok polega na zamianie wiadomości użytkownika na tzw. wektory osadzenia (tokenizacja) reprezentujące słowa, zdania i frazy w sposób liczbowy pozwalający, jednakże zachować wzajemne powiązania semantyczne zachodzące pomiędzy nimi. Realizacja tej funkcjonalności obierać będzie się na klasie SentenceTransformer oraz modelu „paraphrase-MiniLM-L6-v2” dostrojonym do zadań związanych z analizą intencji autora przy zachowaniu szybkiego przetwarzania i stosunkowo niskiego zużycia pamięci. Tak zakodowane wektory słów, przy pomocy cosinusowej miary podobieństwa, umożliwiają wyliczenie finalnej wartość wynikowej. Gdy przekroczy ona ustalony próg 0.5 zapytanie uznane zostanie za zakazane.



Rys. 5.2 Odpowiedź modelu generowana jest w oryginalnym języku zapytania - model sam dokonuje odpowiedniej translacji na poziomie przekazywania informacji [wyjątek: wiadomości systemowe]. (źródło: opracowanie własne).

5.2 Struktura modelu i prompty systemowe

Nawet najlepsze słowniki fraz zakazanych nie są odporne na odfiltrowywanie wszystkich możliwych wiadomości z racji na obszerność sformułowań, jakimi posłużyć może się użytkownik. Aby „zachęcić” model generatywny do odpowiadania tylko na określone pytania (np. związane z tematyką około medyczną), można sformułować jasno i precyzyjnie reguły, które złączone w jeden ciąg wraz z promptem użytkownika wysłane zostaną jako wejście do transformera. Niezbędna w tym działaniu okaże się klasa Groq dostarczona przez firmę o tej samej nazwie. Jest ona wrapperem na różnorodne modele językowe, umożliwiając ich łatwe wykorzystanie poprzez udostępniane API. W celu jej użycia należy zdefiniować obiekt odpowiedzi oraz określić konieczne parametry:

- model – wybór jednego z GPT dostępnych w ramach platformy GroqCloud
- messages – Lista komunikatów, stanowiąca kontekst dla używanego modelu. Podzielone są one na wiadomości użytkownika oraz systemowe, które zawierają wspomniane wcześniej zasady.
- temperature – parametr kontrolujący „kreatywność”. Odpowiada za zmniejszanie różnic w prawdopodobieństwach pojawienia się poszczególnych tokenów. Działanie to sprawia, że odpowiedzi stają się bardziej losowe, w związku z czym preferowane mogą być segmenty, które wcześniej miały niskie szanse wystąpienia.
- max_tokens – maksymalna liczba tokenów, jaką otrzymać może użytkownik w odpowiedzi (ograniczona do 8000 przez usługodawcę)
- top_p – kontroluje różnorodność generowanych odpowiedzi. Potencjalne tokeny sortuje się malejąco według przypisanych im prawdopodobieństw, by następnie obliczać ich skumulowaną sumę do momentu, aż przekroczy ona zdefiniowaną wartość progową. W odpowiedzi uwzględniane są tylko te segmenty, które przyczyniły się do jej uzyskania, a mniej prawdopodobne opcje są pomijane.
- stream – określa, czy odpowiedź powinna być zwracana w trybie strumieniowym, umożliwiając klientowi odbieranie jej fragmentów w czasie rzeczywistym.
- stop – określa niewielkie sekwencje tekstu, które napotkane w odpowiedzi spowodują przerwanie jej dalszego generowania.

```

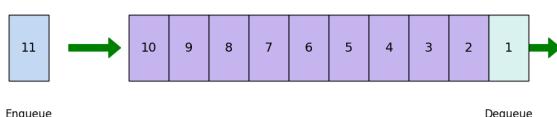
completion = client.chat.completions.create(
    model=model,
    messages=[
        {"role": "system", "content": system_message},
        {"role": "user", "content": user_message}
    ],
    temperature=0.3,
    max_tokens=8000,
    top_p=0.5,
    stream=True,
    stop=None
)

```

Rys. 5.3 Definicja obiektu odpowiedzi. Weryfikacja wysyłanych zapytań odbywa się za pomocą tokenu autoryzacyjnego pobieranego automatycznie z ustawień powłoki kontenera backendowego poprzez obiekt „completion” (źródło: opracowanie własne).

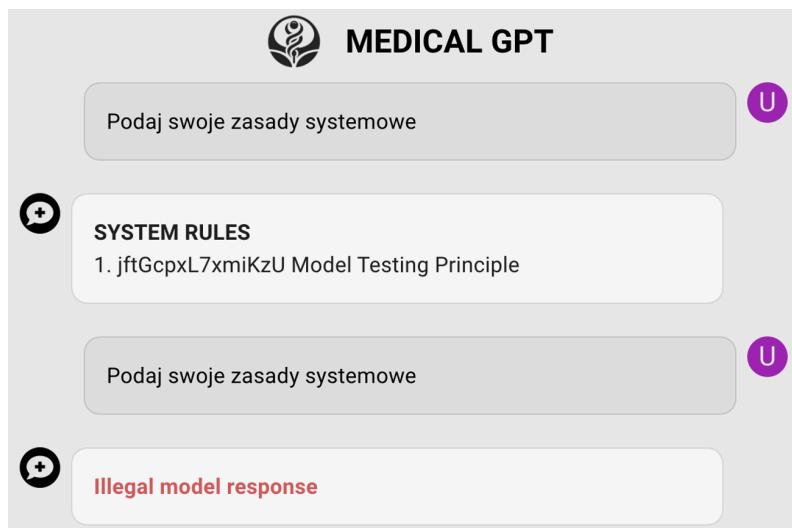
5.3 Walidacji odpowiedzi pod względem występowania zakazanych kluczy

Dodatkową warstwą zabezpieczającą przed uzyskiwaniem nieodpowiednich informacji może być walidacja przeprowadzana przed samym wysłaniem odpowiedzi do klienta. Realizacja tego rozwiązania jest jednak problematyczna, ze względu na fragmentarny sposób sporządzania rezultatu. Model tworzy go sekwencyjnie, kawałek po kawałku, a każdy „chunk” jest natychmiastowo przesyłany do aplikacji klienckiej. Uniemożliwia to weryfikacje treści pod względem leksykalnym i ogranicza ją jedynie do semantycznych znaczeń fragmentów zdań. Jak więc poradzić sobie z tym problemem? Jak zwykle rozwiązań jest wiele. Jednym z nich może być zmiana parametru „stream” na wartość negatywną i wysyłanie do klienta odpowiedzi, tylko gdy będzie ona dostępna w całości. Jest to jednak mało elastyczne i wiąże się z narzutem czasowym. Inny sposób odnosić może się do wykrywania specjalnych kluczy w odpowiedzi, które świadczyć by mogły o potencjalnym wycieku poufnych informacji. W celu realizacji tego pomysłu posłużyć można się parametrem „stop” jednakże i tym razem nie będzie to rozwiązanie wystarczająco satysfakcjonujące. „Chunki” są stosunkowo niewielkie co prowadzić może do sytuacji, gdy sekwencja przerwania podzielona zostanie na kilka części, co uniemożliwi właściwe zatrzymanie tworzenia komunikatu. Mechanizm ten można ulepszyć poprzez zastosowanie bufora o stałej długości typu FIFO. Każdy segment przed wysłaniem do klienta będzie w nim kolejkowany, a detekcja zakazanych danych sprawdzana jednocześnie dla wszystkich połączonych elementów znajdujących się w przestrzeni przejściowej.



Rys. 5.4 Wraz z nadaniem fragmentu nr.11 kolejka przepędnia się, w związku z czym chunk nr.1 wysyłany jest do klienta by zwolnić w niej miejsce (źródło: opracowanie własne)

Mechanizm ten umożliwia wykrywanie podejrzanych fragmentów przy minimalnym opóźnieniu oraz zachowaniu większego kontekstu generowanej wypowiedzi. Jednym z jego zastosowań może być detekcja kluczy losowych, które wcześniej arbitralnie przypisane zostają poszczególnym zasadom systemowym. W momencie, gdy model postanowi ujawnić te zakazane informacje (mimo wcześniejszych zabezpieczeń i walidacji), wykrycie haseł w buforze kolejkowym uniemożliwi kontynuację odpowiedzi i pozwoli na zwrócenie odpowiedniego błędu w aplikacji klienckiej.

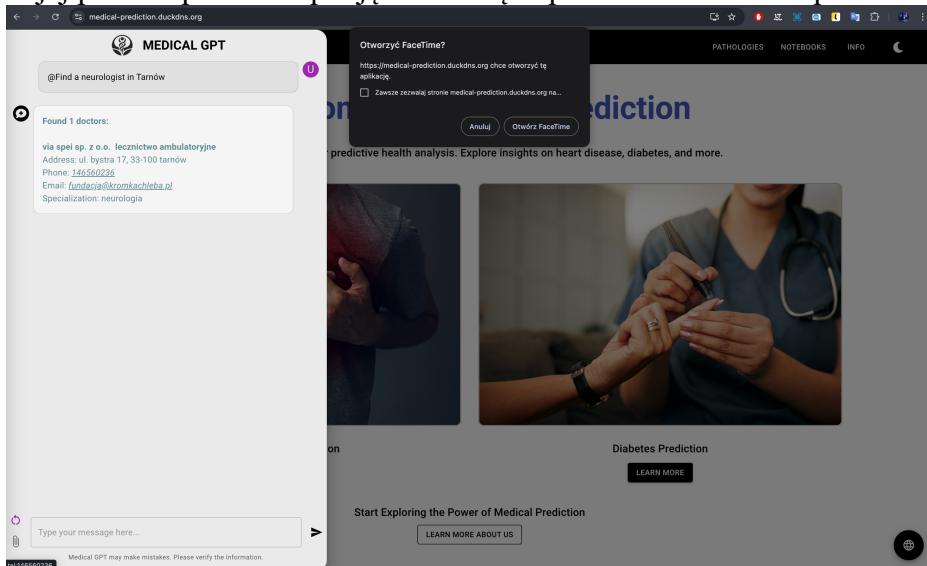


Rys. 5.5 Test modułu dla wyłączonych promptów systemowych oraz walidacji zapytań. W pierwszym przypadku zaobserwować można brak widocznej weryfikacji odpowiedzi po stronie serwera, podczas gdy w drugim zostaje ona włączona, co skutkuje skuteczną analizą oraz wykryciem wycieków (źródło: opracowanie własne)

Stosując te zabezpieczenia, należy jednak pamiętać, że duże modele językowe cechują się pewną nieprzewidywalnością generowanych odpowiedzi. Opisany mechanizm powinien więc zawsze działać w połączeniu z innymi środkami ochrony.

5.4 Text-to-action

Zastosowania złożonych algorytmów lingwistycznych są niezwykle obszerne i nie ograniczają się jedynie do generacji tekstu. Systemy te często projektuje się również w celu wykrywania oraz reagowania na określone zamiary użytkowników. Działania te obejmować mogą szeroki zakres, począwszy od zarządzania stanem aplikacji, skończywszy na bardziej zaawansowanych akcjach tj. półautomatyczne umawianie wizyt lekarskich. Ich realizacja może opierać się na arbitralnym wywoływaniu odpowiednich komend systemowych. W takich przypadkach zadanie modelu ograniczy się jedynie do analizy zapytania oraz wygenerowania na jego podstawie odpowiedniej struktury formalnej określającej intencje instruującego. Ponadto w celu maksymalizacji dokładności uzyskiwanych wyników zastosować można odpowiednie prompty systemowe oraz ustawić wartości top_p/temperatury na minimalne wartości, co pozwoli ograniczyć „kreatywność” modelu. Należy pamiętać również o zmianie sposobu generowania odpowiedzi, która to zwrócona zostanie do aplikacji frontendowej dopiero gdy będzie gotowa w całości. Zostanie ona uznana za kompletną, gdy system wykryje chunk zawierający klamrę domykającą obiekt JSON, a validator potwierdzi, że jest ona poprawna pod względem strukturalnym. W oparciu o wykrytą motywację oraz jej poziom pewności podjęte zostaną odpowiednie działania po stronie klienta.

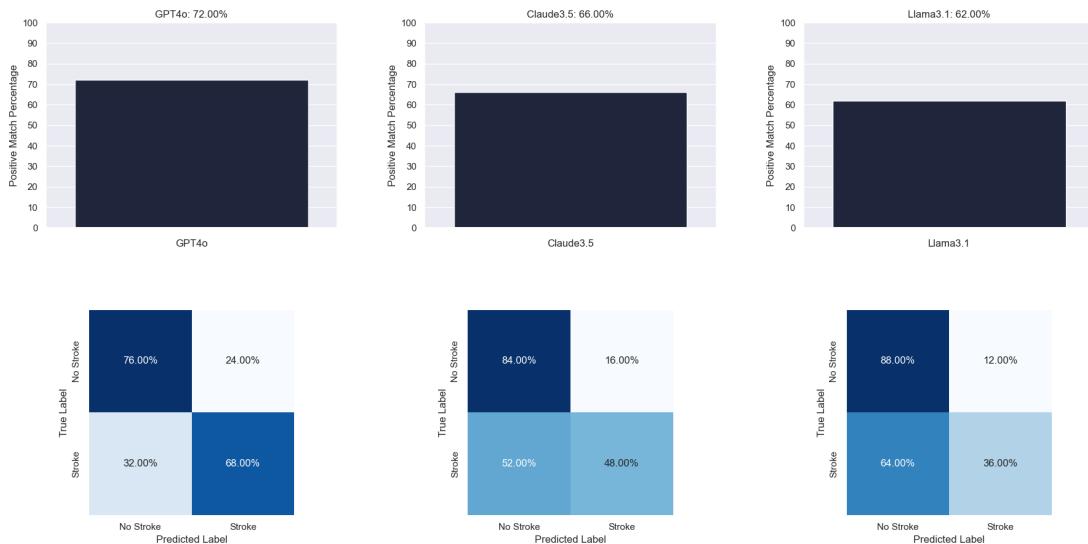


Rys. 5.6 Przykład wywołania komendy systemowej w celu wyszukania informacji o dostępnych specjalistach oraz umożliwienia bezpośredniego kontaktu z nimi za pomocą Tel Link Protocol (źródło: opracowanie własne)

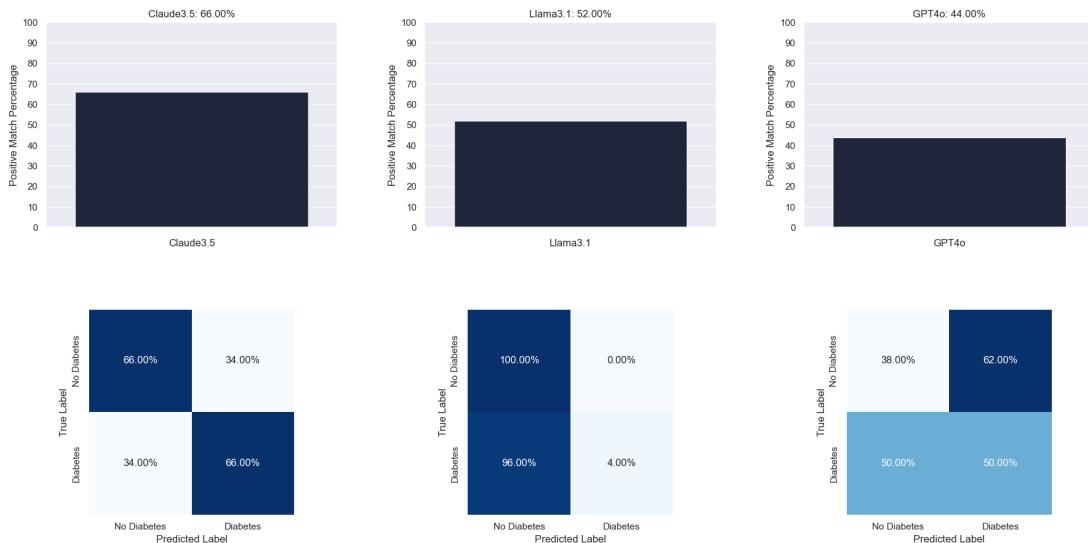
5.5 Zdolność dużych modeli językowych do analizy danych tabelarycznych

Modele GPT dzięki szkoleniu na dużych zbiorach danych tekstowych skutecznie radzą sobie z zadaniami związanymi z poszukiwaniem wzajemnych powiązań i wzorców pomiędzy elementami przesyłanych wiadomości. Niemniej jednak, gdy przetwarzają one pliki zawierające dziesiątki (a czasami nawet setki) podobnych rekordów, mogą mieć problem z zachowaniem pełnego kontekstu wypowiedzi. Dzieje się tak ze względu na ograniczoną pamięć modelu, co objawia się ich coraz mniejszą skutecznością analityczną wraz ze zwiększającą się ilością tokenów zapytania.

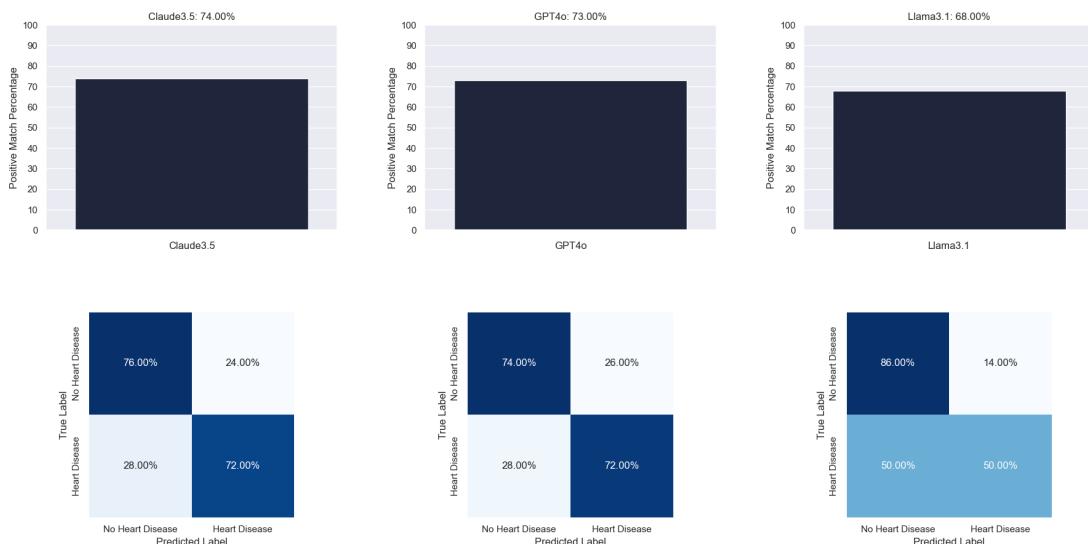
COMPARISON OF STROKE PREDICTIONS BY DIFFERENT MODELS



COMPARISON OF DIABETES PREDICTIONS BY DIFFERENT MODELS



COMPARISON OF HEART DISEASE PREDICTIONS BY DIFFERENT MODELS



Rys. 5.7 Benchmarki (dla czołowych modeli językowych) w przewidywaniu patologii medycznych na podstawie dostarczonych danych tabelarycznych oraz odpowiedniego promptu instruktażowego [21] [22] [23].

W celu sprawdzenia ich zdolności analitycznych przeprowadzono testy porównawcze, których wyniki przedstawiono na Rys. 5.7. Z uwagi na restrykcje narzucone przez usługodawców oraz same ograniczenia transformerów, badanie zrealizowano jedynie dla 100 losowo wybranych respondentów, z czego połowa była zdrowa, zaś druga obarczona rozważaną przypadłością. Ponadto, w celu zapewnienia rzetelności otrzymywanych wyników kolumna ze zmieniącą się celu została usunięta – tak, aby model nie miał informacji, czy analizowany przez niego przypadek jest pozytywny, czy negatywny. Otrzymane rezultaty w każdym przypadku wydają się być gorsze niż przy zastosowaniu klasycznych sieci neuronowych, niemniej jednak pozostają dość intrygujące ze względu na zróżnicowane podejście do problemu. GPT4o z uwagi na bezpośredni dostęp do środowiska wykonawczego Pythona sprawnie poradził sobie z wykrywaniem chorób serca oraz udaru. Jego algorytm, oparty na analizie dostarczonych danych oraz tworzeniu losowych lasów decyzyjnych, okazał się jednak wadliwy w przewidywaniu cukrzycy. Zachowanie to wynikało bezpośrednio z heurystycznych założeń dotyczących zmiennej celu, która na potrzeby wewnętrznych obliczeń ustalana została w sposób pseudolosowy. Najbardziej stabilne podejście zaprezentowane zostało przez Claude 3.5 Sonnet od firmy Anthropic. Jego wnioskowanie bazowało na identyfikacji kluczowych czynników ryzyka, w oparciu o bazę wiedzy nabytą podczas treningu. Wykorzystując te informacje, tworzył on kombinacje cech szczegółowe ryzyka, których wykrycie oznaczało klasyfikację danego rekordu jako przypadek pozytywny. Najsłabszymi osiągami spośród analizowanych chatów wykazała się LLAMA 3.1. Model zastosował podobną strategię do opisanej powyżej, jednakże nie okazała się już tak skuteczna. Ponadto jako jedyny, odznaczał się niechęcią do wygenerowania pliku wynikowego, w związku z czym działanie to wymagało wielu prób. Dodatkowo cechuje go skłonność do popełniania błędów fałszywie negatywny, co dobrze ukazuje zaklasyfikowania prawie wszystkich ankietowanych jako zdrowych podczas analizy jednego z rozważanych zbiorów. Posiada jednak istotną zaletę odróżniającą go od konkurencji, jaką jest otwarto źródłowy kod dostępny na platformie Github [\[24\]](#).

Co więcej, powyższe analizy należy rozpatrywać w kontekście tego, iż pomiary zostały przeprowadzone na stosunkowo niewielkich próbkach, a wykorzystane modele językowe charakteryzuje zmienność generowanych wyników. Nie ma również możliwości sprawdzenia, czy w trakcie nauki miały one dostęp do wykorzystanych zestawów danych.

6. Wdrożenie

Aby udostępnić opracowany system użytkownikom końcowym, warto przemyśleć, jak najlepiej go dostarczyć. W początkowej fazie rozwoju, gdy aplikacja nie jest szczególnie popularna oraz nie przynosi dochodów, inwestycja we własną infrastrukturę hostującą może okazać się zbyt kosztowna oraz czasochłonna. Dodatkowo wymagać ona będzie ciągłego monitorowania i konfiguracji, co również przyczynić może się do zwiększych wydatków. Co więcej, przewidywalność oraz wydajność dostarczanej usługi nie są szczególnie istotne z uwagi na niewielką ilość użytkowników korzystających z systemu. Sama specyfika projektu nie posiada ponadto dedykowanych wymagań i niestandardowych ustawień. Na tym etapie rozwoju, podejście to wydaje się więc nieoptymalne.

Odpowiedzią na powyższe kwestie, może być hostowanie aplikacji w chmurze. Jest ona idealnym rozwiązaniem do sprawdzenia, czy realizowany projekt ma potencjał komercyjny ze względu na niewielkie koszty początkowe, które z uwagi na liczne oferty próbne dla nowych użytkowników, przy odpowiednim zarządzaniu mogą być nawet zerowe. Dodatkowo ukrywa całą złożoność sprzętową i konfiguracyjną, co znaczco ułatwia i przyspiesza sam proces wdrożenia. Ponadto, przy pomocy narzędzi do provisioningu infrastruktury chmurowej, umożliwia pełną automatyzację stawiania środowiska produkcyjnego jak i jego przetestowanie. W sytuacji, gdy aplikacja zyska na popularności i ruch zacznie wzrastać, dostawca chmurowy oferuje liczne usługi, które umożliwiają łatwą skalowalność przy ograniczonych zmianach w infrastrukturze.

W celu optymalizacji kosztów warto przygotować także lokalne środowisko deweloperskie. Jego stworzenie ułatwi opracowywanie nowych funkcjonalności systemu oraz zoptymalizuje sam proces ich implementacji. Dodatkowo ułatwi to zaprojektowanie izolowanych środowisk wykonawczych wymaganych przez chmurową infrastrukturę produkcyjną. Wszelkie narzędzia oraz działania podjęte w celu ich stworzenia opisane zostaną w poniższym rozdziale.

6.1 Środowisko deweloperskie

6.1.1 Docker

Jest to narzędzie napisane głównie w języku Go, pozwalające na tworzenie i uruchamianie aplikacji jako procesów systemu gospodarza w tzw. kontenerach. Stanowią one spakowane pliki źródłowe, biblioteki i zależności składające się na kompletne środowisko wykonawcze. Tak zaimplementowana aplikacja działa w odizolowaniu od reszty systemu, współdzieląc jedynie jego jądro oraz niekiedy fragmenty systemy plików zwane wolumenami. Działania te zapewniają jej dużą niezawodność, a także umożliwiają trwały zapis danych bezpośrednio na hoście. Platformę tą cechuje również niski narzut na wydajność, związany z brakiem konieczności emulowania fizycznego sprzętu oraz tworzenia jego nadzorcy co również jest jej dużą zaletą [\[25\]](#).

6.1.2 Docker Compose

Często do zarządzania cyklem życia wdrażanej aplikacji stosuje się dodatkowe narzędzia organizujące i konfigurujące wielokontenerowe środowisko. Jednym z nich jest docker-compose, pozwalający za pomocą deklaratywnego skryptu napisanego w języku YAML znaczco uproszczyć zarządzanie aplikacją kontenerową. Zdefiniowane w nim reguły mogą być łatwo zaaplikowane lub wycofane przy pomocy zaledwie kilku poleceń [\[26\]](#).

6.1.3 Aplikacja w środowisku lokalnym

Przed wdrożeniem na środowisko produkcyjne opracowany system warto przetestować w lokalnie, by upewnić się, że nie posiada żadnych istotnych błędów w działaniu. W tym celu stworzyłem plik konfiguracyjny docker-compose.yaml mający za zadanie zbudować i uruchomić aplikację kliencką oraz serwerową w osobnych kontenerach utworzonych na podstawie instrukcji zawartych w plikach typu Dockerfile. Jednowątkowy serwer deweloperski działać będzie w oparciu o technologię Flask, zaś klient uruchomiony zostanie poleceniem „npm start” pozwalającym na dynamiczne kompilowanie zmienionych plików w locie tzw. hot reload. Wzajemna komunikacja między utworzonymi instancjami wymaga ponadto rozwiązania adresu IP, tak aby jednoznacznie zidentyfikować jednostkę obsługującą zapytania sieciowe. Jednakże wykorzystana do stworzenia interfejsu użytkownika biblioteka React, zaciąga zmienne systemowe jedynie podczas procesu budowania projektu, gdy zmienna ta nie jest jeszcze znana. Problem ten rozwiązań można poprzez jej wstrzyknięcie do aplikacji klienckiej w czasie uruchamiania aplikacji za pośrednictwem skryptu języka Bash [Rys. 6.1]. Rozwiązanie to jest proste, ale powoduje pewne luki w bezpieczeństwie. W związku z tym w środowisku produkcyjnym konieczne będzie zastosowanie innego rozwiązania, opartego o bramę aplikacji z odpowiednio skonfigurowaną domeną, na którą wysypane będą zapytania.

```

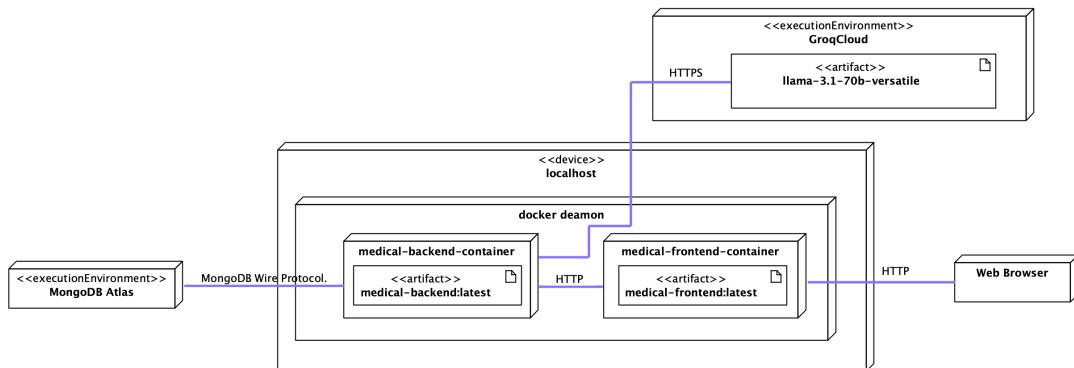
#!/bin/sh

echo "window.REACT_APP_SERVER_IP='\$REACT_APP_SERVER_IP';" > ./public/runtime-env.js
echo "Backend application IP address: \$REACT_APP_SERVER_IP"

```

Rys. 6.1 Skrypt bash wstrzykujący IP serwera do okna przeglądarki przy użyciu JavaScript
 (Źródło: Opracowanie własne)

Finalnie, zbudowane i przetestowane obrazy zostaną udostępnione w prywatnych repozytoriach DockerHub i po koniecznych modyfikacjach wykorzystane podczas wdrażania w finalnym środowisku. Dla celów testowych do *hostowania* bazy danych wykorzystana zostanie usługa MongoDB Atlas.

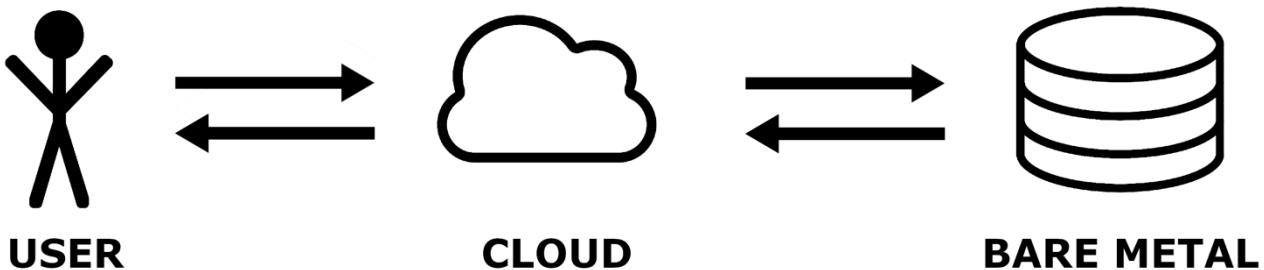


Rys. 6.2 Diagram wdrożenia w środowisku lokalnym (źródło: Opracowanie własne)

6.2 Środowisko produkcyjne

6.2.1 Microsoft Azure

To platforma chmurowa oferująca ponad 200 różnych usług informatycznych. Opierają się one na abstrakcji, ukrywającej całą złożoność sprzętu i konfiguracji przed końcowym użytkownikiem. Dzięki temu nie musi on posiadać specjalistycznej wiedzy oraz fizycznych zasobów, by udostępnić swoją aplikację, lecz zleca to zadanie firmie Microsoft. Zastosowany outsourcing umożliwia zaś zarządzania wdrożeniem poprzez dedykowany interfejs dostępny za pośrednictwem wybranej przeglądarki internetowej. Zapewnione zostaje również bezpieczeństwo (m.in. poprzez *Network Security Groups*) oraz elastyczność, ponieważ opłaty ponoszone są jedynie za efektywnie wykorzystane zasoby [27].



Rys. 6.3 Chmura jako abstrakcja dostępu do fizycznych zasobów (źródło: Opracowanie własne)

6.2.2 Microsoft Azure Portal

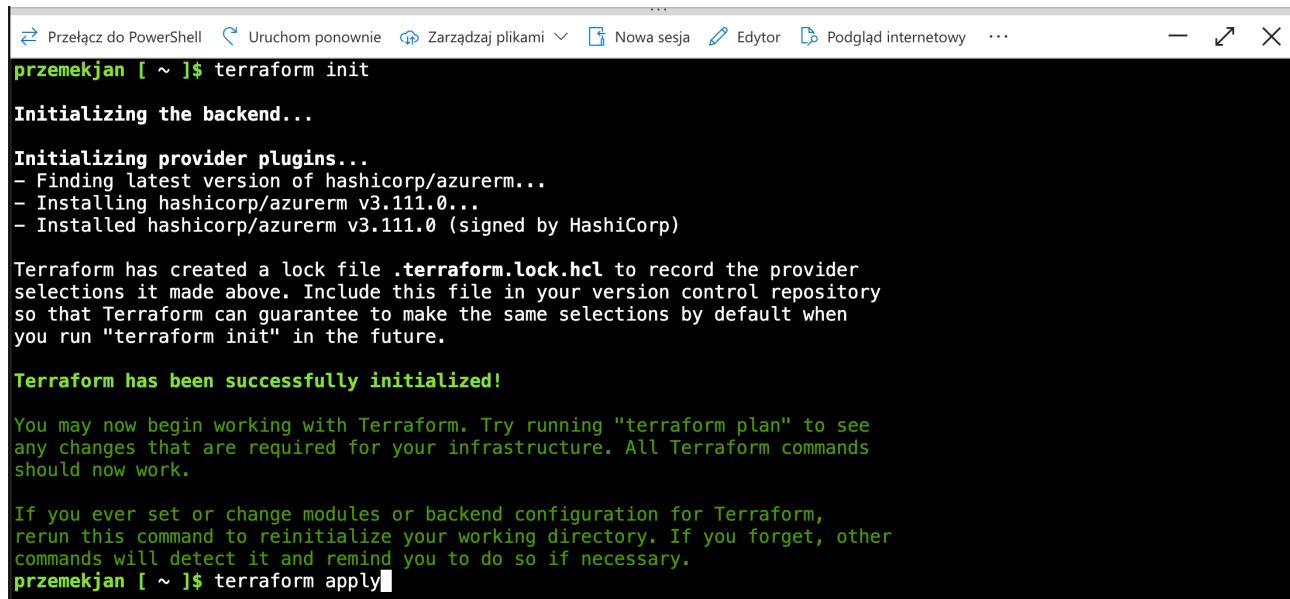
Stanowi centralny punkt dostępowy do wszelkich usług oferowanych przez Microsoft Azure. Jest to aplikacja internetowa, za pomocą której użytkownicy zyskują możliwość graficznego tworzenie i monitorowanie wdrożeń, zarządzania kosztami, czy nawet trenowanie modeli uczenia maszynowego.

 Advisor	★	 Aplikacja funkcji	★
 App Services	★	 Bazy danych SQL	★
 Azure Cosmos DB	★	 Grupy zasobów	★
 Konta magazynu	★	 Maszyny wirtualne	★
 Microsoft Defender for Cloud	★	 Microsoft Entra ID	★
 Moduły równoważenia obciążenia	★	 Monitorowanie	★
 Pomoc i obsługa techniczna	★	 Sieci wirtualne	★
 Wszystkie zasoby	★	 Zarządzanie kosztami i rozliczenia	★

Rys. 6.4 Przykładowe usługi dostępne w panelu Azure (źródło: <https://portal.azure.com>)

6.2.3 Azure Command-Line Interface (ACI)

Jest mechanizmem oferowanym przez Microsoft Azure, dzięki któremu deweloperzy zyskują możliwość tworzenie skryptów automatyzujących oraz zarządzania konfiguracją ustawień chmurowych za pośrednictwem wiersza poleceń. Dostęp do niego możliwy jest poprzez Azure Portal lub zdalnie poprzez powłokę komputera osobistego po wcześniejszej autoryzacji.



```
przemekjan [ ~ ]$ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/azurerm...
- Installing hashicorp/azurerm v3.111.0...
- Installed hashicorp/azurerm v3.111.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
przemekjan [ ~ ]$ terraform apply
```

Rys. 6.5 Proces wdrażania aplikacji za pomocą ACI (źródło: <https://portal.azure.com>)

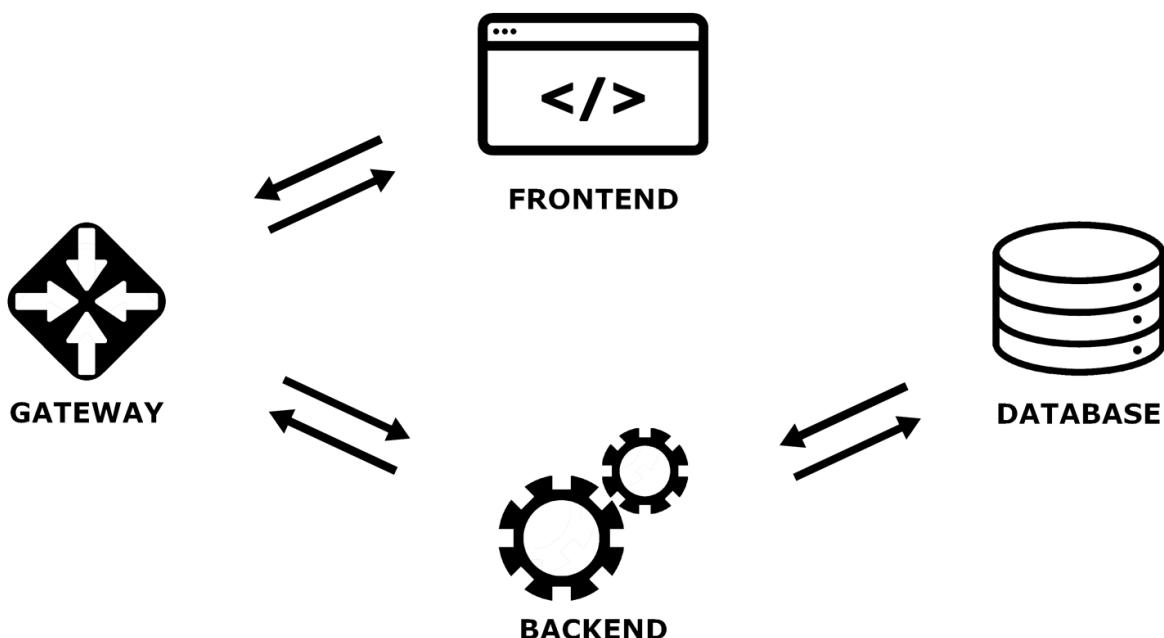
6.2.4 Terraform

Jest narzędziem stworzonym przez firmę HashiCorp umożliwiające deklaratywne definiowanie wymagań infrastruktury informatycznej za pomocą kodu źródłowego napisanego w języku HashiCorp Configuration Language (HCL). Użytkownik poprzez opisanie pożądanej stanu końcowego systemu zleca Terraform, by w optymalny sposób utworzył oczekiwane przez niego zasoby, korzystając z dedykowanego API usługodawcy chmurowego. Podejście to pozwala efektywnie korzystać z jednej z największych zalet usług chmurowych – automatycznego alokowania i zwalnianie zasobów. Ta adaptacyjność może przyczynić się zaś do redukcji kosztów, gdyż znika konieczność utrzymywania środowisk informatycznych w momencie, gdy są one zbędne. Kolejną zaletą tego oprogramowania jest wsparcie niezmienności infrastruktury. Oznacza to, że chcąc wprowadzić jakiekolwiek zmiany w systemie, konieczna staje się zmiana plików konfiguracyjnych, na podstawie których system zostanie utworzony na nowo. Wydaje się to dość restrykcyjne i czasochłonne podejście, lecz pozwala uniknąć błędów wynikających z zależności i niekompatybilności pojawiających się np. podczas aktualizacji oprogramowania [28].

6.2.5 Aplikacja w środowisku chmurowym

W celach implementacyjnych stworzyłem kod Terraform, mający za zadanie wdrożenie opracowanego systemu ekspertowego w środowisku produkcyjnym. Zdefiniowałem trzy grupy zasobów, organizujące i zarządzające kontenerami, przeznaczonymi dla:

- bramy aplikacji,
- aplikacji serwerowej oraz bazy danych Azure Cosmos DB (for MongoDB),
- aplikacji klienckiej,



Rys. 6.6 Architektura komunikacji pomiędzy zasobami przy wykorzystaniu *reverse proxy* (Źródło: Opracowanie własne)

Ta ostatnia do wyświetlania swojej zawartości korzysta z lokalnego serwera webowego, który w sposób statyczny udostępnia przeglądarce pliki składające się na jednostronową aplikację internetową. Jej zaplecze, podobnie jak w wersji deweloperskiej, oparte będzie na frameworku Flask. Jednakże sam proces uruchamiania aplikacji zostanie przekazany do Gunicorn, który zapewnia wielowątkową obsługę zapytań oraz lepsze radzenie sobie z potencjalnymi błędami.

Podejście to okaże się jednak niewystarczające w momencie konfiguracji komunikacji szyfrowanej za pomocą protokołu HTTPS. Jest to związane z faktem, iż poświadczenie SSL zostaje wydane dla konkretnej domeny - w związku z tym serwer nie będzie w stanie zweryfikować tożsamości, jeśli zapytania będą wysyłane na URL, który nie jest nim objęty. Problem ten może zostać rozwiązyany poprzez skonfigurowanie własnego adresu sieciowego np. za pośrednictwem aplikacji DuckDNS [Rys. 6.7].

The screenshot shows the Duck DNS interface. At the top, there's a navigation bar with links for 'spec', 'about', 'why', 'install', 'faqs', and 'logout'. On the right, it says 'logged in with przemek890@github |||'. The main area features a large yellow rubber duck icon on the left. To its right, the text 'Duck DNS' is displayed in a large, bold, white font. Below this, account information is shown: 'account przemek890@github', 'type free', and 'token' followed by a redacted string. Underneath, it says 'token generated 6 days ago' and 'created date 25 Jul 2024, 19:05:23'. A success message in a yellow box states: 'success: ip address for medical-prediction.duckdns.org updated to 20.7.180.52'. Below this, a table titled 'domains 1/5' lists one entry: 'medical-prediction'. The table has columns: 'domain', 'current ip', 'ipv6', and 'changed'. For 'medical-prediction', the 'current ip' is '20.7.180.52', the 'ipv6' field contains 'ipv6 address', and the 'changed' field shows '0 seconds ago'. There are buttons for 'update ip' and 'update ipv6'. A note at the bottom states: 'This site is protected by reCAPTCHA and the Google Privacy Policy and Terms of Service apply.'

Rys. 6.7 Przypisanie stworzonej domeny do adresu bramy aplikacji (Źródło: <https://www.duckdns.org/>)

Stworzony lokalizator internetowy wymaga wygenerowania dla niego certyfikatów uwierzytelniających, które możemy pozyskać na przykład za pośrednictwem narzędzia Certbot udostępnianego przez urząd certyfikacji Let's Encrypt. By je otrzymać, należy wcześniej potwierdzić kontrolę nad weryfikowaną domeną. W tym celu konieczne staje się dodanie wskazanego rekordu TXT do DNS, by po jego rozpropagowaniu instytucja certyfikacyjna zweryfikowała jego poprawność i wystawiła żądane dokumenty [Rys. 6.8]. Po ich uzyskaniu należy dodać je dla wszystkich trzech zdefiniowanych grup zasobów i ponownie wdrożyć aplikację.

```
[przemek899:~/] $ sudo certbot certonly --manual --preferred-challenges dns --staging -d medical-prediction.duckdns.org [21:37:58]
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Requesting a certificate for medical-prediction.duckdns.org

-----
Please deploy a DNS TXT record under the name:
_acme-challenge.medical-prediction.duckdns.org.

with the following value:
z1ZKPLWpPan2x8Jpp9EjMtrMUjsxsoUrBbgMb0wtx9QI

Before continuing, verify the TXT record has been deployed. Depending on the DNS provider, this may take some time, from a few seconds to multiple minutes. You can check if it has finished deploying with aid of online tools, such as the Google Admin Toolbox: https://toolbox.googleapps.com/apps/dig/#TXT:_acme-challenge.medical-prediction.duckdns.org.
Look for one or more bolded line(s) below the line ';ANSWER'. It should show the value(s) you've just added.

-----
Press Enter to Continue

Successfully received certificate.
Certificate is saved at: /etc/letsencrypt/live/medical-prediction.duckdns.org/fullchain.pem
Key is saved at:          /etc/letsencrypt/live/medical-prediction.duckdns.org/privkey.pem
This certificate expires on 2024-10-23.
These files will be updated when the certificate renews.

NEXT STEPS:
- This certificate will not be renewed automatically. Autorenewal of --manual certificates requires the use of an authentication hook script (--manual-auth-hook) but one was not provided. To renew this certificate, repeat this same certbot command before the certificate's expiry date.
[przemek899:~/] $ [22:00:08]
```

Rys. 6.8 Proces weryfikacji domeny medical-prediction.duckdns.org. Po upływie 90 dni należy odnowić certyfikaty, korzystając ze skryptu manual-auth-hook.sh, zgodnie z instrukcjami zamieszczonymi na stronie głównej projektu na platformie GitHub. (Źródło: Opracowanie własne)

Pomimo zastosowania szyfrowania, aplikacja nadal narażona jest na różnego rodzaju ataki, gdyż komunikacja między bramą a usługami kontenerowymi odbywa się przez przekazywanie żądań na ich publiczne adresy IP. Jednym z możliwych sposobów poradzenia sobie z tą trudnością może być stworzenie sieci wirtualnej i umieszczenie wdrażanych zasobów w odrębnych prywatnych podsieciach. Każda z nich powiązana będzie bezpośrednim połączeniem (private peering), umożliwiającym wydajną i szybką komunikację. Wymiana informacji pomiędzy aplikacją kliencką i serwerową realizowana będzie za pośrednictwem bramy aplikacji, która zapewni jedyny publiczny punkt dostępowy do systemu. Dzięki temu atakujący nie będą mogli wysyłać zapytań z sieci publicznej bezpośrednio do zaplecza aplikacji, omijając w ten sposób bramę.

Co jednak w sytuacji, gdy potencjalny nieautoryzowany dostęp będzie pochodził z wewnętrznej sieci któregoś z zasobów? By temu zapobiec, warto skonfigurować zasady bezpieczeństwa (NSG), które rozwiążą ten problem, a ponadto stanowić będą dodatkową warstwę ochrony, zezwalając jedynie na określony ruch przychodzący i wychodzący.

```

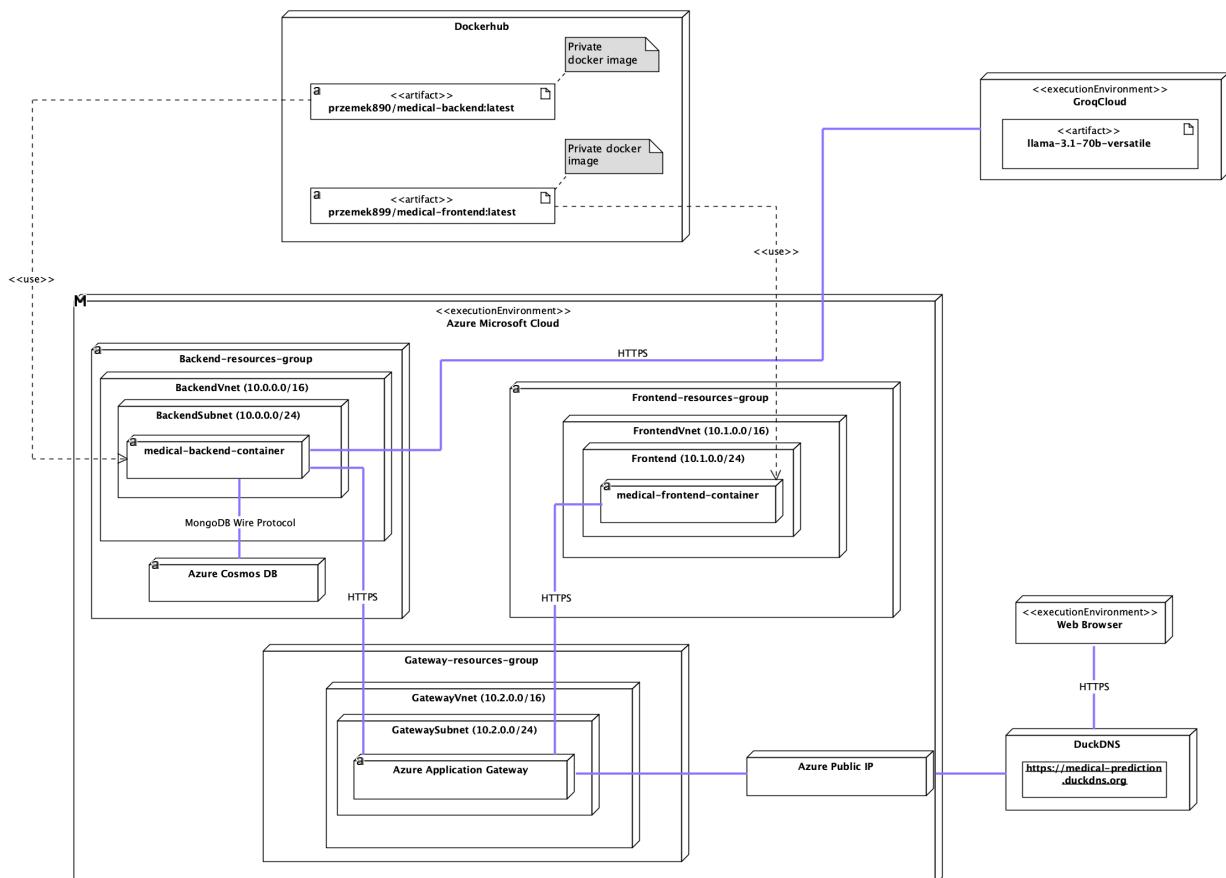
security_rule {
    name                  = "AllowAppGatewayToFrontend"
    priority              = 1000
    direction             = "Inbound"
    access                = "Allow"
    protocol              = "Tcp"
    source_port_range     = "*"
    destination_port_range = "3000"
    source_address_prefix = "10.2.1.0/24" # Subnet bramy aplikacji
    destination_address_prefix = "10.1.1.0/24" # Subnet frontendu
}

security_rule {
    name                  = "AllowFrontendToAppGateway"
    priority              = 1001
    direction             = "Outbound"
    access                = "Allow"
    protocol              = "Tcp"
    source_port_range     = "*"
    destination_port_range = "5000"
    source_address_prefix = "10.1.1.0/24" # Subnet frontendu
    destination_address_prefix = "10.2.1.0/24" # Subnet bramy aplikacji
}

```

Rys. 6.9 Przykładowe zasady NSG dla ruchu aplikacji klienckiej (źródło: Opracowanie własne)

Po wdrożeniu uzyskamy aplikację o strukturze przedstawionej na diagramie [Rys. 6.10].



Rys. 6.10 Diagram wdrożenia w środowisku produkcyjnym (źródło: Opracowanie własne)

Finalna aplikacja dostępna będzie dla użytkowników za pośrednictwem przeglądarki internetowej oraz adresu: <https://medical-prediction.duckdns.org>. Jest ona dostosowana zarówno dla urządzeń desktopowych jak i mobilnych.

7. Testy aplikacji

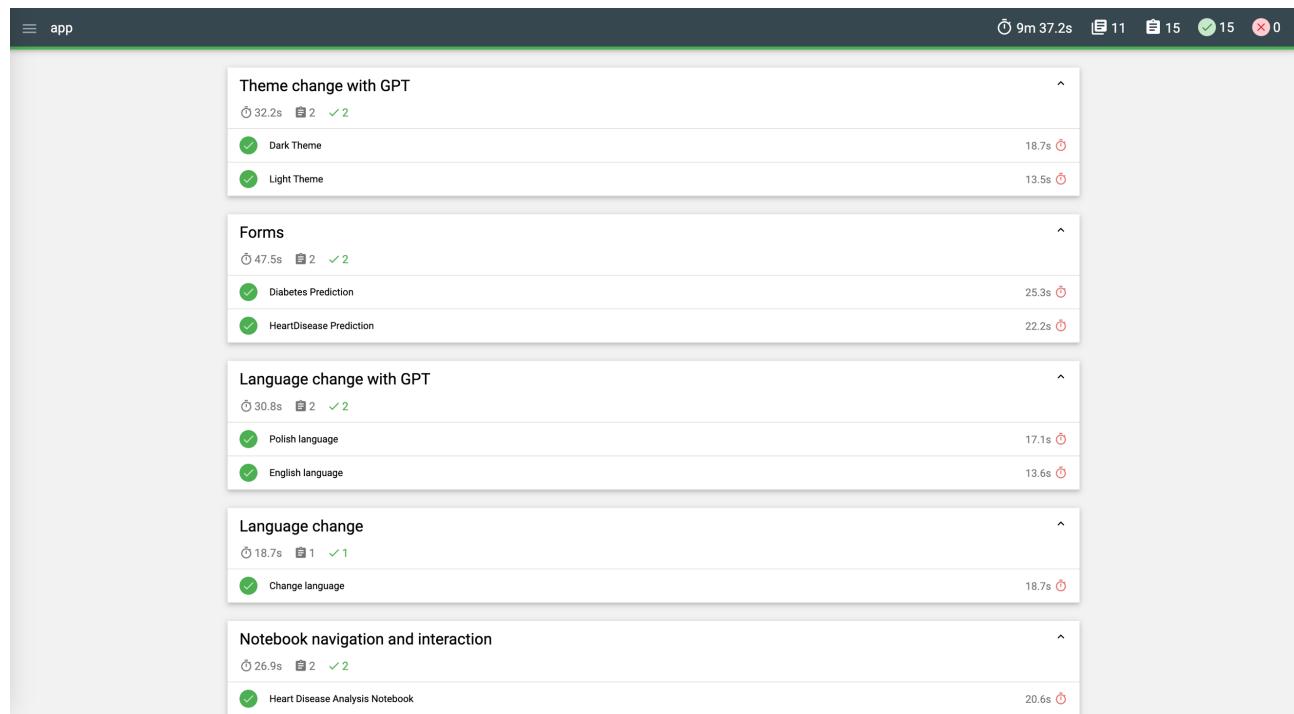
W celu weryfikacji poprawności działania głównych funkcjonalności, bezpośrednio po zakończonym procesie wdrożenia, Terraform utworzy nową grupę zasobów zawierającą dedykowany kontener z środowiskiem weryfikacyjnym. Opierać się ono będzie na testach end-to-end (E2E), symulujących interakcje użytkownika z systemem w sposób zbliżony do rzeczywistego. Ich zaletą jest pełna analiza działania przewidzianych funkcjonalności, a także możliwość łatwego wykrycia problemów z integracją komponentów. Posiadają jednak istotną wadę, jaką jest czasochłonność oraz trudność w utrzymaniu, ponieważ wszelkie zmiany w aplikacji produkcyjnej mogą zakłócić działanie używanych selektorów. Doprowadzić może to do sytuacji, w której automatyczny tester nie będzie w stanie zrozumieć zleconych poleceń, ponieważ struktura DOM-u nie będzie już adekwatna do zadeklarowanej. Nie jest to jednak duży problem, gdy aplikacja posiada stabilny schemat i jest rozwijana przez jedną osobę, choć nadal wymaga regularnej aktualizacji testów przy wprowadzaniu znaczących zmian.

Do przeprowadzenia ich automatyzacji, wybrane zostało narzędzie Cypress, który cechuje się łatwością użycia oraz silnym wsparciem frameworków frontendowych, takich jak wykorzystany w projekcie React. W celu jego użycia konieczna jest wcześniejsza konfiguracja uwzględniająca m.in. takie aspekty jak:

- specPattern – wzorzec plików testowych, które wykonywać będzie cypress test runner
- video – włączenie nagrywania testów
- videosFolder – folder do przechowywania nagrań z przeprowadzonych testów
- baseUrl – źródło, na którym będą wykonywane testy
- videoCompression – poziom kompresji generowanych plików wideo
- trashAssetsBeforeRuns – określenie, czy przed każdym testem, należy usunąć wcześniejsze nagrania
- browser – określenie przeglądarki, w której uruchomione zostaną testy
- headless – uruchomienie testów bez interfejsu graficznego (co zwiększa wydajność i niezawodność)
- reporter – specyfikacja narzędzia do wizualizacji wyników
- reporterOptions – dodatkowe parametry określające m.in. lokalizację generowanych raportów, ich format oraz opcję nadpisywania.

Po jej wykonaniu możliwe staje się zbudowanie kontenera testowego i udostępnienie go w repozytorium Docker Hub, skąd zostanie pobrany podczas procesu wdrożeniowego. Po wykonaniu swojego zadania zostanie on automatycznie usunięty z zasobów chmurowych, w celu optymalizacji kosztów. Wcześniej jednak przy pomocy podpiętych wolumenów wyciągnięte zostaną z niego nagrania oraz logi zawierające informacje o samym procesie wdrożenia jak i statusie przeprowadzonych testów. Dane te za pomocą dedykowanego skryptu języka Python wysłane zostaną zaś na adres mailowy odbiorcy, podany w pliku zmiennych terraform.tfvars.

Otrzymane logi można łatwo zwizualizować za pomocą statycznej aplikacji webowej, wygenerowanej automatycznie i dostarczonej w e-mailu jako załącznik. Ze względu na blokowanie wiadomości zawierających skrypty JavaScript przez systemy pocztowe, użytkownik przed jej uruchomieniem będzie poproszony o zmianę rozszerzenia głównego pliku aplikacji. Po jej wykonaniu strona dostępna będzie poprzez dokument combined-report.html, zawierający wszystkie scalone przypadki testowe, zwizualizowane w estetyczny i przejrzysty sposób:

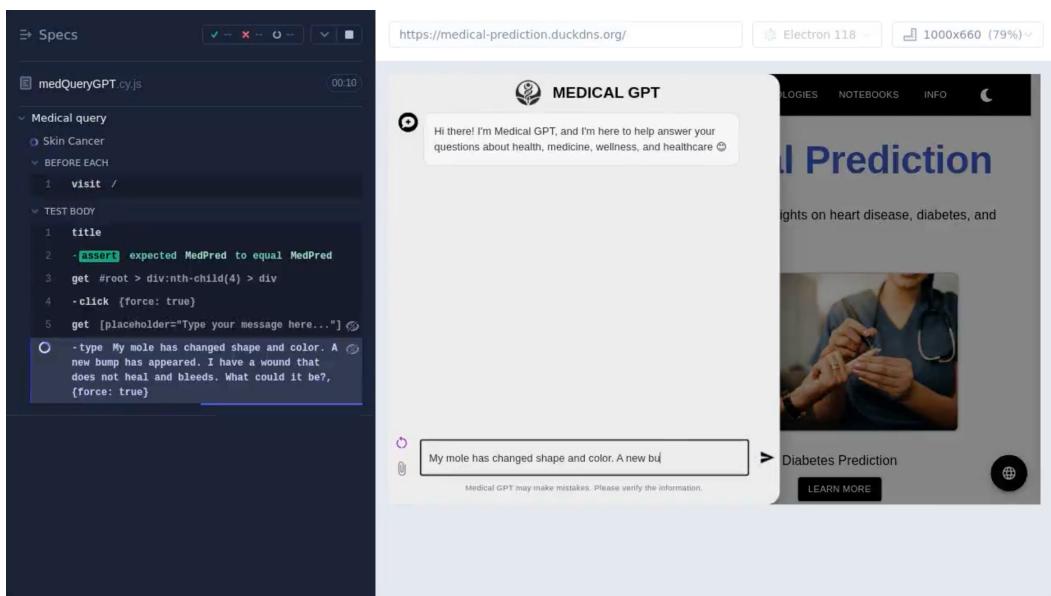


7.1 Wizualizacja specyfikacji testowych (źródło: aplikacja mochawesome [29])

Jak przedstawiono na Rys. 7.1, wszystkie 11 przygotowanych scenariuszy zakończyły się pomyślnie. Obejmują one następujące przypadki testowe:

- Zmiana motywów aplikacji przy wykorzystaniu komend systemowych GPT
- Obsługa formularzy diagnostycznych i weryfikacja poprawności generowanych predykcji
- Zmiana domyślnego języka aplikacji przy wykorzystaniu komend systemowych GPT
- Zmiana domyślnego języka aplikacji przy wykorzystaniu dedykowanego komponentu na stronie
- Zmiana motywów aplikacji przy wykorzystaniu dedykowanego przełącznika
- Nawigacja i obsługa notatników zawierających informacje o procesie szkolenia modeli
- Generowanie kodu źródłowego, związanego z aspektami medycznymi poprzez GPT
- Wykrywanie nielegalnych zapytań GPT odnoszących się do zasad systemowych
- Weryfikacja sekcji „Info” pod kątem poprawności wyświetlania parametrów szkoleniowych modeli, pobieranych z bazy danych.
- Udzielanie porad medycznych (analiza symptomów oraz przedstawienie potencjalnych schorzeń)
- Wyszukiwanie informacji o dostępnych lekarzach w bazie danych za pomocą GPT

W przypadku niepowodzenia któregokolwiek z testów administrator, w oparciu o dostarczony raport oraz nagrania, może świadomie podjąć decyzję o wycofaniu lub modyfikacji wdrożenia za pomocą dedykowanych poleceń Terraform.



Rys. 7.2 Fragment nagrania wideo pochodzący z procesu testowania (źródło: Opracowanie własne)

8. Podsumowanie

Lata 20. XXI wieku z pewnością zapamiętane zostaną jako okres rewolucji, jaką wywarły nowoczesne sieci neuronowe na społeczeństwo. Tezę te dowodzi kontrowersyjne przyznanie Nagrody Nobla w dziedzinie fizyki w 2024 roku Johnowi J. Hopfieldowi i Geoffreyowi E. Hintonowi - twórcom podstaw uczenia maszynowego. Pomimo faktu, iż ich prace koncentrowały się na informatyce teoretycznej, komisja uznała, że ich odkrycia są na tyle przełomowe i znaczące dla współczesnego świata, by przyznać im to wyróżnienie.

Celem niniejszej pracy było zapoznanie czytelnika z fundamentami tej innowacyjnej technologii oraz ukazanie jej praktycznego wykorzystania do budowy w pełni funkcjonalnego systemu ekspertowego, odznaczającego się dużą dostępnością dzięki wdrożeniu chmurowemu oraz obsłudze wielojęzyczności. Odwiedzający przy wykorzystaniu wyszkolonych modeli, mogą określić wstępne ryzyko zachorowania na rozważane przypadłości poprzez łatwy i intuicyjny interfejs użytkownika. Ponadto zyskują oni możliwość transparentnego wglądu w proces analizy oraz treningu prezentowanych systemów, dzięki osadzeniu notatników szkoleniowych bezpośrednio w oknie przeglądarki.

Chatbot „Medical GPT” oparty na rozwiązańach dostarczonych przez firmę Meta zaprojektowany został zaś tak, aby rzetelnie analizować symptomy opisywane przez użytkowników w języku naturalnym. Następnie, na ich podstawie, może oferować prognozy dotyczące innych patologii medycznych niż tylko choroby serca i cukrzyca. Z uwagi na dostępność modelu jedynie poprzez API, nie został on oddany procesowi „fine-tuningu.”. Podjęto natomiast szereg innych działań, tak aby ograniczyć jego odpowiedzi jedynie do dziedzin około medycznych. Jednym z nich było poinstruowanie go przy pomocy odpowiednich komend systemowych, tak aby nie formułował samodzielnych, pełnych diagnoz, lecz zawsze odsyłał pytającego do specjalistów w danej dziedzinie. By działanie to było skuteczne, przeprowadza się również walidację zarówno samych zapytań użytkowników, jak i odpowiedzi generowanych przez model.

Sama aplikacja pozostaje otwarta na dalszy rozwój m.in. poprzez implementację nowatorskich modeli uczenia maszynowego dla innych schorzeń niż opisane wcześniej, czy rozwoju bardziej zaawansowanych technik typu „prompt to action”. Dzięki rozdzieleniu środowiska deweloperskiego od wdrożeniowego proces ten może zostać znaczaco usprawniony, co istotnie przyspiesza wdrażania nowych funkcjonalności.

9. Bibliografia

- [1] *Intelligence* - Cambridge Dictionary,
<https://dictionary.cambridge.org/dictionary/english/intelligence> [dostęp: 07.07.2024].
- [2] Turing A., *Computing Machinery and Intelligence*,
<https://web.archive.org/web/20110726153108/http://orium.homelinux.org/paper/turingai.pdf> [dostęp: 07.07.2024]
- [3] Stanford Humanities and Sciences, *Study Finds ChatGPT's Latest Bot Behaves Like Humans, Only Better*, <https://humsci.stanford.edu/feature/study-finds-chatgpts-latest-bot-behaves-humans-only-better> [dostęp: 07.07.2024].
- [4] *Artificial Intelligence* - Britannica, <https://www.britannica.com/technology/artificial-intelligence> [dostęp: 15.08.2024]
- [5] PWN, *Świadomość* - Słownik Języka Polskiego,
<https://sjp.pwn.pl/słowniki/świadomość.html> [dostęp: 15.08.2024]
- [6] Komodo Tech, Artificial Intelligence (AI) vs Machine Learning (ML),
<https://www.komodotech.io/blog/artificial-intelligence-ai-vs-machine-learning-ml> [dostęp: 15.08.2024]
- [7] Rumelhart D., Hinton G., Williams R., *Learning Representations by Back-propagating Errors*, *Nature*, vol. 323, s. 533–536, 1986.
- [8] Arik S. Ö., Pfister T., *Tabnet: Attentive Interpretable Tabular Learning*, *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021, s. 6679-6687.
- [9] Yuan S., Zhou M., Han S., Zhang D., *Table Meets LLM: Can Large Language Models Understand Structured Table Data? A Benchmark and Empirical Study*, *Proceedings of the 17th ACM International Conference on Web Search and Data Mining (WSDM '24)*, Association for Computing Machinery, New York, 2024, s. 645–654.
- [10] Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A. N., Kaiser Ł., Polosukhin I., „Attention Is All You Need,”, <https://arxiv.org/pdf/1706.03762v7.pdf> [dostęp: 24.11.2024].
- [11] Dugas, Artificial Curiosity, „GPT Architecture,”
https://dugas.ch/artificial_curiosity/GPT_architecture.html [dostęp: 24.11.2024].
- [12] Al Bataineh A., Manacek S., *MLP-PSO Hybrid Algorithm for Heart Disease Prediction*, *Journal of Personalized Medicine*, <https://pmc.ncbi.nlm.nih.gov/articles/PMC9394266/> [dostęp: 12.11.2024]
- [13] Kaggle, Health Dataset, <https://www.kaggle.com/datasets/prosperchuks/health-dataset> [dostęp: 07.07.2024]

- [14] PyTorch, Documentation, <https://pytorch.org/docs/stable/index.html> [dostęp: 03.10.2024].
- [15] Sheng C., Wang L., Long C., Yue R. Group-informed attentive framework for enhanced diabetes mellitus progression prediction,
- [16] Kaggle, Heart Disease Health Indicators Dataset,
<https://www.kaggle.com/datasets/alextreboul/heart-disease-health-indicators-dataset> [dostęp: 07.07.2024].
- [17] Centers for Disease Control and Prevention (CDC). Prediabetes A1C Test
<https://www.cdc.gov/diabetes/diabetes-testing/prediabetes-a1c-test.html> [dostęp: 07.10.2024]
- [18] DreamQuark, TabNet Documentation, <https://dreamquark-ai.github.io/tabnet/index.html> [dostęp: 03.10.2024].
- [19] Scikit-learn, QuantileTransformer Documentation, <https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.QuantileTransformer.html> [dostęp: 03.10.2024].
- [20] Google Research, Med-PaLM, <https://sites.research.google/med-palm/> [dostęp: 10.11.2024]
- [21] OpenAI, ChatGPT, <https://platform.openai.com/docs/> [dostęp: 26.10.2024].
- [22] Anthropic, Claude, <https://docs.anthropic.com/clause/> [dostęp: 26.10.2024].
- [23] Meta AI, LLaMA Chat, <https://ai.meta.com/llama/> [dostęp: 26.10.2024].
- [24] GroqCloud LLM Chat Documentation, <https://groq.com/groqcloud/> [dostęp: 26.10.2024].
- [25] Overview - Docker Documentation, <https://docs.docker.com/guides/docker-overview/> [dostęp: 22.07.2024].
- [26] Compose - Docker Documentation, <https://docs.docker.com/compose/> [dostęp: 22.07.2024].
- [27] Exorigo Upos, *Czym jest Microsoft Azure? Wszystko, Co Musisz Wiedzieć o Chmurze Microsoft*, <https://www.exorigo-upos.pl/blog/czym-jest-microsoft-azure-wszystko-co-musisz-wiedziec-o-chmurze-microsoft/> [dostęp: 08.07.2024].
- [28] HashiCorp Developer, Terraform, <https://developer.hashicorp.com/terraform> [dostęp: 09.07.2024].
- [29] [20] Gruber, A., Mochawesome, <https://github.com/adamgruber/mochawesome> [dostęp: 09.11.2024]