WUM Projekt 1 - Raport

Aleksandra Wójcik, Jan Opala

Kwiecień 2024

1 Wstęp

1.1 Cel

Celem projektu jest opracowanie rozwiązania problemu klasyfikacji demencji w oparciu o dane zdrowotne pacjentów. Zależy nam na tym, żeby uzyskać model, który z wysoką dokładnością i precyzją stwierdzi, czy pacjent jest chory na demencję, czy nie. Takie rozwiązanie może przydać się w diagnostyce medycznej, a także w profilaktyce, pomagając wyłonić cechy silnie skorelowane z występowaniem demencji i które potencjalnie są za nią odpowiedzialne.

1.2 Źródło

Dane dotyczące pacjentów pochodzą z ramki danych z poniższego źródła:
Dementia Patient Health Dataset

2 Eksploracyjna analiza danych

2.1 Informacje na temat zmiennych

Analizowana ramka danych zawiera 24 kolumny (zmienne) po 1000 wierszy (odpowiadających kolejnym pacjentom). Poszczególne zmienne i ich opisy prezentują się następująco:

- 'Diabetic' - zmienna zerojedynkowa stwierdzająca obecność cukrzycy
- 'AlcoholLevel' - ilość alkoholu w promilach (numeryczna)
- 'HeartRate' - tętno w uderzeniach serca na minutę (numeryczna)
- 'BloodOxygenLevel' - % poziomu tlenu we krwi (między 96% a 99%) (numeryczna)
- 'BodyTemperature' - temperatura ciała w stopniach Celsjusza (numeryczna)
- 'Weight' - masa ciała w kilogramach (numeryczna)

- 'MRI_Delay' - parametr medyczny związany z rezonansem magnetycznym (numeryczna)
- 'Prescription' - przepisany lek na demencję/brak
- 'Dosage in mg' - ilość w miligramach leku na demencję (jeśli dotyczy) (numeryczna)
- 'Age' - wiek (numeryczna)
- 'Education_Level' - poziom edukacji (brak, podstawowe, średnie, wyższe)
- 'Dominant_Hand' - dominująca ręka (left/right)
- 'Gender' - płeć (kobieta/mężczyzna)
- 'Family_History' - czy ktoś z rodziny chorował na demencję (Yes/No)
- 'Smoking_Status' - status palacza (nigdy, kiedyś, obecnie)
- 'APOE_ϵ4' - obecność białka odpowiedzialnego za chorobę Alzheimera (Positive/Negative)
- 'Physical_Activity' - tryb życia pod względem aktywności fizycznej (siedzący, łagodny, umiarkowany)
- 'Depression_Status' - obecność depresji (Yes/No)
- 'Cognitive_Test_Scores' - wynik w testach sprawdzających zdolności poznawcze (skala 0 - 10)
- 'Medication_History' - czy pacjent przyjmował leki na demencję (Yes/No)
- 'Nutrition_Diet' - rodzaj diety (niskowęglowodanowa, zrównoważona, śródziemnomorska)
- 'Sleep_Quality' - jakość snu (poor/good)
- 'Chronic_Health_Conditions' - choroby przewlekłe (choroba serca, nadciśnienie, cukrzyca, brak)
- 'Dementia' - zmienna zerojedynkowa, obecność demencji

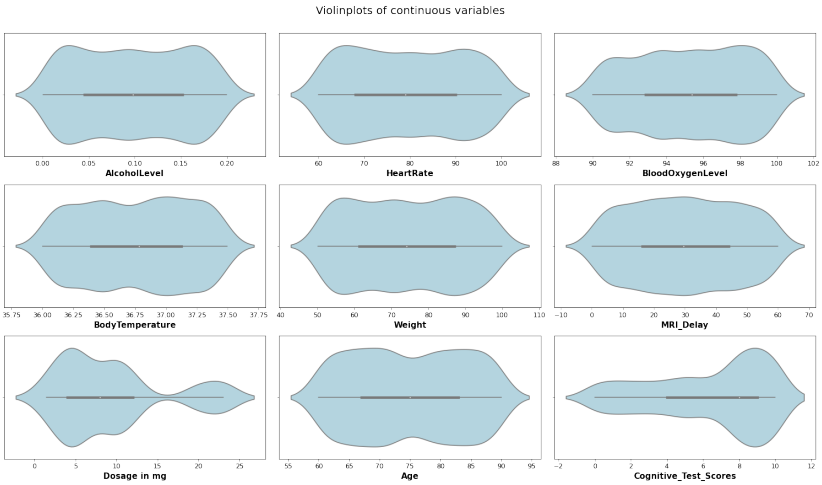
2.2 Braki danych

zmienna	liczba braków danych
Prescription	515
Dosage in mg	515
Chronic_Health_Conditions	179

Kolumny 'Prescription' i 'Dosage in mg' odnoszą się do tego samego, czyli do leków na demencję. Brak danych w tym przypadku oznaczał nieprzymowanie żadnego leku.
W przypadku chorób braki danych pochodzą od pacjentów, którzy nie są przewlekle chorzy.

2.3 Rozkłady zmiennych numerycznych

Sporządziliśmy wykresy skrzypcowe w celu zaobserwowania tego, jak rozkładają się dane zmiennych numerycznych w zbiorze danych.

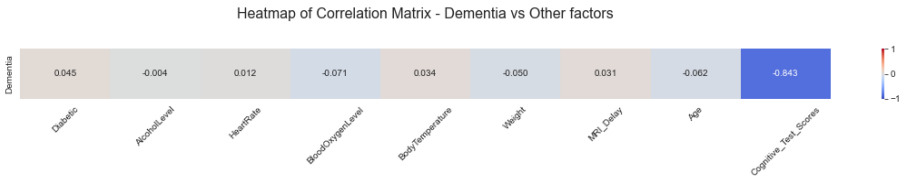


Rysunek 1: Wykresy skrzypcowe przedstawiające rozkłady zmiennych numerycznych

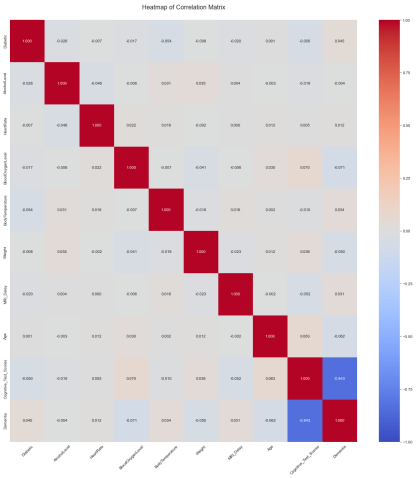
Proste wnioski, jakie można wyciągnąć w oparciu o powyższe wykresy to kształt wykresów rozkładów bliski normalnemu (spłaszczone z brakiem jednoznacznego wierzchołka), w związku z czym przeważają dane bliskie mediany, jest mało wartości skrajnych. Odstępstwa od powyższych reguł są zauważalne dla zmiennych: 'Dosage in mg' (wyraźny szczyt bliski 4 miligramów, lokalne maksimum dla 22 miligramów) oraz 'Cognitive_Test_Scores' (wyraźne maksimum dla 9 punktów na 12 możliwych).

2.4 Korelacje zmiennych

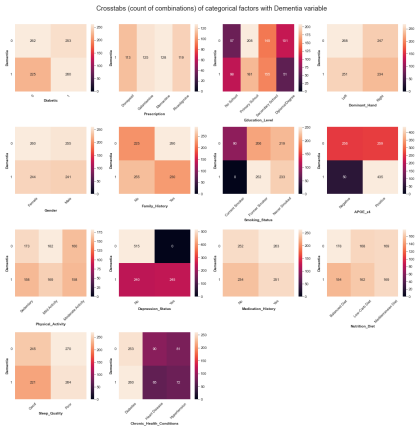
Na etapie eksploracyjnej analizy danych zdecydowaliśmy się określić jakie są korelacje między poszczególnymi zmiennymi i w szczególności między zmiennymi a naszą zmienną przewidywaną (zmienna zerojedynkowa wskazująca na wystąpienie demencji.



Rysunek 2: Heatmapa przedstawiająca korelacje między demencją z zmiennymi numerycznymi



Rysunek 3: Heatmapa przedstawiająca korelacje między zmiennymi numerycznymi



Rysunek 4: Heatmapa przedstawiająca korelacje między demencją a zmiennymi kategorycznymi

W oparciu o heatmapy można dojść do następujących wniosków:

- Istnieje bardzo wysoka korelacja między wystąpieniem demencji a wynikiem testów poznawczych.
- Żaden z aktywnych palaczy nie jest chory na demencję.
- U wszystkich osób chorych na depresję wykazano wystąpienie demencji.

3 Preprocessing i transformacja zmiennych

3.1 Transoformacja zmiennych kategorycznych na numeryczne

W celu wygodnego tworzenia modeli podjęliśmy decyzję o konieczności zamiany wszystkich zmiennych kategorycznych na numeryczne. Żeby tego dokonać, skorzystaliśmy z następującego kodu:

```
dementia_conv = df
dementia_conv['Family_History'] = dementia_conv['
    Family_History'].replace({'Yes' : 1, 'No' : 0})
dementia_conv['Gender'] = dementia_conv['Gender'].replace({'
    Female' : 1, 'Male' : 0})
dementia_conv['Depression_Status'] = dementia_conv['
    Depression_Status'].replace({'Yes' : 1, 'No' : 0})
dementia_conv['Medication_History'] = dementia_conv['
    Medication_History'].replace({'Yes' : 1, 'No' : 0})
dementia_conv['Sleep_Quality'] = dementia_conv['
    Sleep_Quality'].replace({'Good' : 1, 'Poor' : 0})
```

```
dementia_conv['Physical_Activity'] = dementia_conv['
    Physical_Activity'].replace({'Moderate Activity' : 1, '
    Mild Activity' : 0.5, 'Sedentary' : 0})
dementia_conv['Smoking_Status'] = dementia_conv['
    Smoking_Status'].replace({'Current Smoker' : 1, 'Former
    Smoker' : 0.5, 'Never Smoked' : 0})
dementia_conv['APOE_$\epsilon4$'] = dementia_conv['APOE_ 4'
    ].replace({'Positive' : 1, 'Negative' : 0})
dementia_conv['Left_Hand'] = dementia_conv['Dominant_Hand'].
    replace({'Left' : 1, 'Right' : 0})
dementia_conv['Education_Level'] = dementia_conv['
    Education_Level'].replace({'Diploma/Degree' : 1, '
    Secondary School' : 2/3, 'Primary School' : 1/3,  'No
    School' : 0})
```

W powyższym kodzie wprowadziliśmy następujące zmiany:

- 'Education_Level' - poziom edukacji (brak, podstawowe, średnie, wyższe) zamienione na 0 1/3 2/3 1
- 'Gender' - płeć zamieniona na zmienną zaerojedynkową (bycie kobietą)
- 'Family_History' - czy ktoś z rodziny chorował na demencję (Yes/No), zamieniona na zmienną zerojedynkową
- 'Smoking_Status' - status palacza (nigdy, kiedyś, obecnie) na 0 1/2 1
- 'APOE_ϵ4' - obecność białka odpowiedzialnego za chorobę Alzheimera (Positive/Negative) zamienione na zmienną zerojedynkową
- 'Physical_Activity' - tryb życia pod względem aktywności fizycznej (siedzący, łagodny, umiarkowany), zamienione na 0 1/2 1
- 'Depression_Status' - obecność depresji, zamienionę na zerojedynkową
- 'Medication_History' - czy przyjmował leki na demencję (Yes/No) zamieniona na zerojedynkową
- 'Sleep_Quality' - jakość snu poor/good zamieniona na zerojedynkową (1 good, 0 poor)

Innymi słowy, tam gdzie zmienna mówiła o obecności lub braku jakiegoś czynnika (Yes/No) zmienną zamieniliśmy na zerojedynkową, a tam gdzie mieliśmy pewien rodzaj stopniowania, to zmienną zamieniliśmy na numeryczną o równych odstępach (np. $0, \frac{1}{3}, \frac{2}{3}, 1$ itd).

3.2 Dummies

Część złożonych zmiennych (np. występowanie choroby przewlekłej) zamieniliśmy na kilka prostych zmiennych zerojedynkowych. W tym celu posłużyliśmy się funkcjonalnością *get_dummies*:

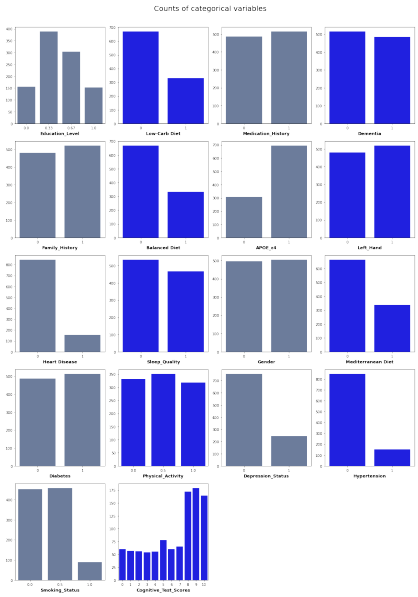
```
dummies = pd.get_dummies(dementia_conv['
    Chronic_Health_Conditions'])
dementia_conv = pd.concat([dementia_conv, dummies], axis=1)
dummies2 = pd.get_dummies(dementia_conv['Nutrition_Diet'])
dementia_conv = pd.concat([dementia_conv, dummies2], axis=1)
dummies3 = pd.get_dummies(dementia_conv['Prescription'])
dementia_conv = pd.concat([dementia_conv, dummies3], axis=1)

for column in dementia_conv.columns:
    if dementia_conv[column].dtype == bool:
        dementia_conv[column] = dementia_conv[column].astype
            (int)

dementia_conv = dementia_conv.drop(columns=['Diabetic', '
    Nutrition_Diet', 'Chronic_Health_Conditions', 'Donepezil'
    , 'Galantamine', 'Memantine', 'Rivastigmine', '
    Dominant_Hand', 'Prescription', 'Dosage in mg'])
```

W rezultacie otrzymaliśmy nowe następujące kolumny w kodzie:

- 'Left_Hand' - zmienna zerojedynkowa (leworęczność)
- 'Diabetes' - zmienna zerojedynkowa - obecność cukrzycy
- 'Heart Disease' - zmienna zerojedynkowa - obecność chorób serca
- 'Hypertension' - zmienna zerojedynkowa - obecność nadciśnienia
- 'None' - zmienna zerojedynkowa - brak chorób
- 'Balanced Diet' - zmienna zerojedynkowa - dieta zbalansowana
- 'LowCarb Diet' - zmienna zerojedynkowa - dieta niskowęglowodanowa
- 'Mediterranean Diet' - zmienna zerojedynkowa - dieta śródziemnomorska

Zmienne zerojedynkowe rodzaju przyjmowanego leku na demencję:

- 'Donepezil',
- 'Galantamine',
- 'Memantine',
- 'Rivastigmine'



Rysunek 5: Histogramy przedstawiające ile razy występuje dana wartość dyskretna dla kolumn zmiennych dyskretnych

3.3 Rozkłady zmiennych dyskretnych

Żeby zaobserwować jak się w zbiorze danych rozkładają zmienne dyskretne (w tym zerojedynkowe oraz te, które pierwotnie były kategoryczne) zdecydowaliśmy się sporządzić odpowiednie histogramy. Z powyższych histogramów możemy wywnioskować, że zbiory różnych płci, zbiory zdrowych/chorych na demencję oraz cukrzyków/niecukrzyków są równoliczne. Możemy również zaobserwować, że większość badanej populacji stanowią osoby z wykształceniem podstawowym i średnim.

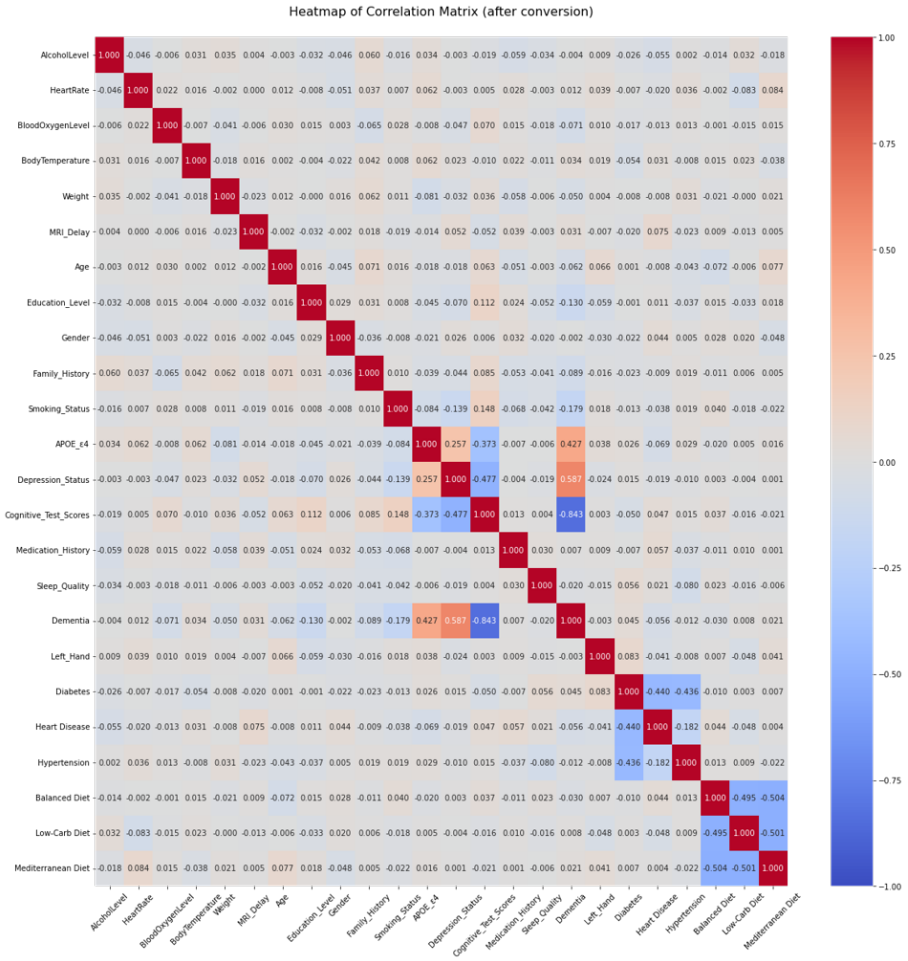
4 Próba redukcji wymiarów

W celu przetestowanie możliwości redukcji wymiarów i sprawdzenia wystąpienia ukrytych czynników, które miałyby wysoką zdolność eksplanacyjną, zdecydowaliśmy się zastosować kilku metod analizy czynnikowej.

4.1 PCA

Podstawową metodą, jaką użyliśmy do redukcji wymiarów, było Principal Component Analysis (PCA).
W tym celu wykonaliśmy analizę składowych w Pythonie:

```
from sklearn.decomposition import PCA
```



Rysunek 6: Macierz korelacji dla wszystkich zmienych po transformacji

```
X = dementia_conv.drop('Dementia', axis=1)
pca = PCA(n_components=4)
X_pca = pca.fit_transform(X)
explained_variance_ratio = pca.explained_variance_ratio_
print("Explained variance ratio for each component:")
for i, variance in enumerate(explained_variance_ratio):
    print(f"Component {i+1}: {variance:.2f}")

total_variance_explained = sum(explained_variance_ratio)
print(f"\nTotal variance explained by selected components: {
    total_variance_explained:.2f}")
X_pca
```

```
components = pca.components_

for i, component in enumerate(components):
    print(f"Kolumny b Źd ce kombinacj  liniow  dla
        komponentu {i+1}:")
    for j, coefficient in enumerate(component):
        print(f"- Kolumna {X.columns[j]}: {coefficient:.2f}"
            )
    print()
```

z otrzymanym outputem:

```
Explained variance ratio for each component:
Component 1: 0.38
Component 2: 0.28
Component 3: 0.20
Component 4: 0.11

Total variance explained by selected components: 0.97
Kolumny b Źd ce kombinacj  liniow  dla komponentu 1:
- Kolumna AlcoholLevel: -0.00
- Kolumna HeartRate: -0.00
- Kolumna BloodOxygenLevel: 0.00
```

Powyższa analiza została powtórzona w R:

```
pca_result <- prcomp(dementia_conv, scale. = TRUE)

cat("Wariancja dla ka dego komponentu:\n")
print(pca_result\$sdev^2)

cat("\nProporcja wariancji wyja Żnionej przez ka dy
    komponent:\n")
print(pca_result\$sdev^2 / sum(pca_result\$sdev^2))

cat("\nKombinacje liniowe dla ka dego komponentu:\n")
print(pca_result\$rotation[, 1:4])
```

I rezultat:

```
Wariancja dla ka dego komponentu:
> print(pca_result$sdev^2)
 [1] 1.2244687 1.1593724 1.0947380 1.0214900 0.9916208
     0.9532264 0.9372628 0.9031486 0.8875372
[10] 0.8271351
> cat("\nProporcja wariancji wyja Żnionej przez ka dy
    komponent:\n")

Proporcja wariancji wyja Żnionej przez ka dy komponent:
> print(pca_result$sdev^2 / sum(pca_result$sdev^2))
 [1] 0.12244687 0.11593724 0.10947380 0.10214900 0.09916208
     0.09532264 0.09372628 0.09031486
```

```
 [9] 0.08875372 0.08271351
> cat("\nKombinacje liniowe dla ka dego komponentu:\n")

Kombinacje liniowe dla ka dego komponentu:
> print(pca_result$rotation[, 1:4])
                                                    PC1
                                                        PC2

                                                        PC3

                                                        PC4
Cognitive_Test_Scores                        0.05141387
    -0.3543524  0.30357657 -0.42118760
Depression_Status                           -0.06203344
    -0.4365842 -0.29248252 -0.42423540
APOE_ 4                                      0.17083207
    -0.1701255 -0.51605864  0.26478197
Smoking_Status                               0.20075529
    0.4075703  0.38412848 -0.07055233
Education_Level                              0.31951341
    -0.2606779 -0.17495803  0.49622982
Family_History                              -0.49485131
    -0.3502366  0.19134763  0.25049214
BloodOxygenLevel                             0.53069895
    -0.1904343  0.19939577 -0.12307360
Age                                         -0.01294494
    -0.4880730  0.30509846  0.02537398
Chronic_Health_ConditionsHeart Disease       0.05958358
    0.1005590 -0.44595605 -0.49060270
Weight                                      -0.54008925
    0.1164523 -0.09736145 -0.04545220
```

Po dwukrotnym przeprowadzeniu analizy głównych składowych można dojść do wniosku, że nie przynosi ona żadnego pożytecznego efektu. Utworzoe czynniki mają bardzo niską wariancję (w odniesieniu do całości patrząc na proporcje) co czyni je bardzo słabe ze względu na zdolności wyjaśniania zjawiska demencji. Być może, demencja nie jest definiowana przez pewne czynniki będące kombinacjami liniowymi istniejących cech, a tychże cech. Przed pochopnym stwierdzeniem takiego wniosku wpierw przeanalizujemy inne metody redukcji wymiarów.

4.2 Inne metody redukcji wymiarów

Zdecydowaliśmy się wykorzystać również inne metody matematyczne (LDA, SVD i Random Forest), ale nie przyniosły one żadnych pożytecznych rezultatów. W rzeczywistości, doszliśmy do tych samych konkluzji co podczas analizy PCA.
Kody, które zostały użyte do metod redukcji wymiarów prezentują się następująco:

4.2.1 LDA

```
from sklearn.discriminant_analysis import
    LinearDiscriminantAnalysis as LDA
y = dementia_conv['Dementia']
lda = LDA(n_components=1)
X_lda = lda.fit_transform(X, y)
X_lda
```

4.2.2 SVD

```
from sklearn.decomposition import TruncatedSVD
svd = TruncatedSVD(n_components=2)
X_svd = svd.fit_transform(X)
X_svd
```

4.2.3 Random Forest

```
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier()
rf.fit(X, y)
feature_importance = rf.feature_importances_
feature_importance
```

Output powyższej funkcji:

```
array([0.00962462, 0.00667218, 0.01125311, 0.01320403,
    0.01035281,
        0.00962958, 0.00986874, 0.00690074, 0.00133112,
            0.00234324,
        0.02130784, 0.07555461, 0.00246376, 0.13829204,
            0.66807925,
        0.00136675, 0.00195628, 0.00128949, 0.00089974,
            0.00129462,
        0.00176716, 0.00085333, 0.00128357, 0.0013008 ,
            0.00111059])
```

W związku z tym mamy do czynienie z bardzo małą istotnością utworzonych czynników. Wynika z tego fakt, że prawdopodobnie żadna z obliczeniowych metod redukcji wymiarów nie przyda się nam przy tworzeniu modelu.

4.3 Rozwiązanie problemu braków danych

W celu pozbycia się braków danych zrezygnowaliśmy z korzystania z kolumny 'Chronic_Health_Conditions' (informacje o chorobach znajdują się w dummies), a także z leków na demencję, ponieważ występują one tylko pośród osób mających demencję.

4.4 Kryterium korelacji

Zdecydowaliśmy się sprawdzić, które zmienne mają najwyższy współczynnik korelacji i potencjalnie wybrać akurat te do naszego modelu.
W celu znalezienia 4 zmiennych najsilniej skorelowanych z demencją wywołaliśmy kod:

```
high_correlation_columns2 = abs(
    selected_correlation_matrix_conv['Dementia']).nlargest(4)
high_correlation_columns2
```

i otrzymaliśmy w efekcie:

```
Cognitive_Test_Scores     0.843247
Depression_Status         0.587006
APOE_$\epsilon4$                    0.427292
Smoking_Status            0.179253
Name: Dementia, dtype: float64
```

W związku z tym zdecydowaliśmy się zrobić nowy zbiór danych, w którym znalazłyby się tylko te 4 kolumny.

5 Wstępne modele

Wstępne modelowanie przeprowadziliśmy zarówno dla zbioru danych z 10 kolumnami o najwyższej korelacji z demencją, jak i zbioru zawężonego do 4 najbardziej skorelowanych kolumn.

Dla obu tych wersji dokonaliśmy podziału zbioru na zbiór treningowy i testowy, a następnie ze zbioru treningowego wydzieliliśmy również wewnętrzny zbiór walidacyjny. Wszystkie utworzone podzbiory zapisywaliśmy do plików. Kod, który to realizował, wyglądał następująco:

```
path = 'C:\\Users\\User\\DataspellProjects\\
    AleksandraWojcikJanOpalaEDA_Projekt1'

X = dementia_conv_high(liczba_kolumn).drop('Dementia', axis
    =1)
y = dementia_conv_high(liczba_kolumn)['Dementia']

X_train_val, X_test, y_train_val, y_test = train_test_split(
    X, y, test_size=0.1, random_state=42, stratify=y)

X_train, X_val, y_train, y_val = train_test_split(
    X_train_val, y_train_val, test_size=0.25, random_state
    =42, stratify=y_train_val)

df_train = pd.concat([X_train, y_train], axis=1)
```

```
df_val = pd.concat([X_val, y_val], axis=1)
df_test = pd.concat([X_test, y_test], axis=1)

df_train.to_csv(path + '\\train.csv', index=False)
df_val.to_csv(path + '\\val.csv', index=False)
df_test.to_csv(path + '\\test.csv', index=False)
```

Zbiory danych nazwane były odpowiednio dla 10 i 4 kolumn: "dementia_conv_high10" i "dementia_conv_high4".

Dla każdego z poniższych prostych klasyfikatorów napisaliśmy podobny blok kodu, który tworzył klasyfikator i go trenował, a następnie generował prognozy dla zestawu danych testowych i wypisywał wyniki oceny wydajności modelu w czterech metrykach (również rysował macierz pomyłek). Kod wyglądał następująco (przykład: DummyClassifier):

```
dc = DummyClassifier(strategy='uniform', random_state=42)
dc.fit(X_train,y_train)
y_pr = dc.predict_proba(X_test)
y_hat = dc.predict(X_test)

accuracy = accuracy_score(y_test, y_hat)
print("Dokladnosc:", accuracy)

precision = precision_score(y_test, y_hat)
print("Precyzja:", precision)

recall = recall_score(y_test, y_hat)
print("Czulosc:", recall)

f1 = f1_score(y_test, y_hat)
print("F1-score:", f1)

conf_matrix = confusion_matrix(y_test, y_hat)
print("Macierz pomylek:\n", conf_matrix)
```

5.1 Zbiór danych 10 zmiennych

5.1.1 Baseline

Dla strategii uniform:

	klasyfikacja pozytywna	klasyfikacja negatywna
stan pozytywny	28	20
stan negatywny	28	24

atrybut metody pomiaru	wartość
Accuracy (dokładność)	0.52
Precision (precyzja)	0.5
Recall (czułość)	0.5833333333333334
F1	0.5384615384615384

Dla strategii stratified:

	klasyfikacja pozytywna	klasyfikacja negatywna
stan pozytywny	29	30
stan negatywny	18	23

atrybut metody pomiaru	wartość
Accuracy (dokładność)	0.41
Precision (precyzja)	0.3829787234042553
Recall (czułość)	0.375
F1	0.37894736842105264

Dla strategii 'most_frequent':

	klasyfikacja pozytywna	klasyfikacja negatywna
stan pozytywny	0	48
stan negatywny	0	52

atrybut metody pomiaru	wartość
Accuracy (dokładność)	0.52
Precision (precyzja)	0.0
Recall (czułość)	0.0
F1	0.0

5.1.2 Regresja logistyczna

Dla penalty 'None':

	klasyfikacja pozytywna	klasyfikacja negatywna
stan pozytywny	48	0
stan negatywny	0	52

atrybut metody pomiaru	wartość
Accuracy (dokładność)	1.0
Precision (precyzja)	1.0
Recall (czułość)	1.0
F1	1.0

Dla penalty 'l1':

	klasyfikacja pozytywna	klasyfikacja negatywna
stan pozytywny	48	0
stan negatywny	0	52

atrybut metody pomiaru	wartość
Accuracy (dokładność)	1.0
Precision (precyzja)	1.0
Recall (czułość)	1.0
F1	1.0

Dla penalty 'l2':

	klasyfikacja pozytywna	klasyfikacja negatywna
stan pozytywny	48	0
stan negatywny	0	52

atrybut metody pomiaru	wartość
Accuracy (dokładność)	1.0
Precision (precyzja)	1.0
Recall (czułość)	1.0
F1	1.0

5.1.3 SVM

	klasyfikacja pozytywna	klasyfikacja negatywna
stan pozytywny	36	12
stan negatywny	0	52

atrybut metody pomiaru	wartość
Accuracy (dokładność)	0.88
Precision (precyzja)	1.0
Recall (czułość)	0.75
F1	0.8571428571428571

5.1.4 Drzewo decyzyjne

	klasyfikacja pozytywna	klasyfikacja negatywna
stan pozytywny	48	0
stan negatywny	0	52

atrybut metody pomiaru	wartość
Accuracy (dokładność)	1.0
Precision (precyzja)	1.0
Recall (czułość)	1.0
F1	1.0

5.1.5 Klasyfikator k sąsiadów

	klasyfikacja pozytywna	klasyfikacja negatywna
stan pozytywny	39	9
stan negatywny	1	51

5.2 Zbiór danych 4 zmiennych

5.2.1 Regresja logistyczna

Dla penalty 'None':

atrybut metody pomiaru	wartość
Accuracy (dokładność)	0.9
Precision (precyzja)	0.975
Recall (czułość)	0.8125
F1	0.8863636363636362

	klasyfikacja pozytywna	klasyfikacja negatywna
stan pozytywny	48	0
stan negatywny	0	52

atrybut metody pomiaru	wartość
Accuracy (dokładność)	1.0
Precision (precyzja)	1.0
Recall (czułość)	1.0
F1	1.0

Dla penalty 'l1':

	klasyfikacja pozytywna	klasyfikacja negatywna
stan pozytywny	48	0
stan negatywny	0	52

atrybut metody pomiaru	wartość
Accuracy (dokładność)	1.0
Precision (precyzja)	1.0
Recall (czułość)	1.0
F1	1.0

Dla penalty 'l2':

	klasyfikacja pozytywna	klasyfikacja negatywna
stan pozytywny	45	3
stan negatywny	0	52

atrybut metody pomiaru	wartość
Accuracy (dokładność)	0.97
Precision (precyzja)	1.0
Recall (czułość)	0.9375
F1	0.967741935483871

5.2.2 SVM

	klasyfikacja pozytywna	klasyfikacja negatywna
stan pozytywny	48	0
stan negatywny	0	52

atrybut metody pomiaru	wartość
Accuracy (dokładność)	1.0
Precision (precyzja)	1.0
Recall (czułość)	1.0
F1	1.0

5.2.3 Klasyfikator k sąsiadów

	klasyfikacja pozytywna	klasyfikacja negatywna
stan pozytywny	48	0
stan negatywny	0	52

atrybut metody pomiaru	wartość
Accuracy (dokładność)	1.0
Precision (precyzja)	1.0
Recall (czułość)	1.0
F1	1.0

6 Modele zaawansowane

Na etapie modelowania zaawansowanego zdecydowaliśmy się wykorzystać metodę automatycznego uczenia maszynowego (AutoML), która pozwala na zautomatyzowanie całego procesu, w tym inżynierię cech, normalizację czy strojenie

hiperparametrów, a także przeprowadza analizę wielu różnych modeli i dobiera najbardziej optymalne rozwiązania. Do naszego zbioru danych użyliśmy pakietu Lightautoml, który stanowi połączenie kilku modeli i technik, wykorzystując algorytmy z popularnych bibliotek, takich jak scikit-learn, XGBoost, CatBoost i LightGBM.

Ustawiliśmy zadanie dla automatyzacji jako klasyfikację binarną. Wykorzystaliśmy narzędzie TabularAutoML, dla którego skonfigurowalismy parametry następująco:
ograniczenie czasu obliczeń do 10 min,
użycie 4 rdzeni CPU,
wykorzystanie konkretnych algorytmów (”linear l2”, ”lgb”, ”lgb tuned”).

Za pomocą funkcji ”fit_predict” wytrenowaliśmy model, używając danych z początkowego zbioru podzielonych na dwa podzbiory - jeden do trenowania, drugi do oceny modelu. Ta funkcja jednocześnie przewiduje wyniki dla tego samego zbioru danych.
Następnie wygenerowaliśmy przewidywania dla drugiego zbioru (nowy zestaw danych) za pomocą funkcji ”predict”.
Dla obu zestawów danych wypisaliśmy wyniki oceny modelu w trzech metrykach: F1, accuracy, Gini.

```
def f1_metric(y_true, y_pred):
    return f1_score(y_true, (y_pred > 0.5).astype(int))

def accuracy_metric(y_true, y_pred):
    return accuracy_score(y_true, (y_pred > 0.5).astype(int)
        )

def gini_metric(y_true, y_pred):
    return 2*roc_auc_score(y_true, y_pred)-1

task = Task('binary', metric = f1_metric)

roles = {'target': 'Dementia'}

automl = TabularAutoML(task = task,
                       timeout = 600,
                       cpu_limit = 4,
                       general_params = {'use_algos': [['
                           linear_l2', 'lgb', 'lgb_tuned']]})

train_data, valid_data = train_test_split(df,
                                          test_size=0.1,
                                          stratify=df['Dementia
                                              '],
```

```
                                        random_state=42)
print('train_data shape = {}, \nvalid_data shape = {}'.
    format(train_data.shape, valid_data.shape))

oof_pred = automl.fit_predict(train_data, roles = roles)

valid_pred = automl.predict(valid_data)

print('OOF accuracy: {}'.format(accuracy_metric(train_data['
    Dementia'].values, oof_pred.data[:, 0])))
print('OOF f1: {}'.format(f1_metric(train_data['Dementia'].
    values, oof_pred.data[:, 0])))
print('OOF gini: {}'.format(gini_metric(train_data['Dementia
    '].values, oof_pred.data[:, 0])))

print('OOF accuracy: {}'.format(accuracy_metric(valid_data['
    Dementia'].values, valid_pred.data[:, 0])))
print('OOF f1: {}'.format(f1_metric(valid_data['Dementia'].
    values, valid_pred.data[:, 0])))
print('OOF gini: {}'.format(gini_metric(valid_data['Dementia
    '].values, valid_pred.data[:, 0])))
```

7 Wpływ zmiennych na predykcję

W celu lepszego zrozumienia modeli sztucznych inteligencji odpowiedzialnych za przewidywanie demenci w oparciu o nasz zbiór danych zdecydowaliśmy się skorzystać z technik wyjaśnialnej sztucznej inteligencji, aby wyłonić najważniejsze cechy, które determinują przypisanie do klasy, a także, żeby zobaczyć odwrotny skumulowany rozkład reszt.

7.1 Stworzenie wyjaśnialnego modelu klasyfikacji

W tym celu skorzystamy z klasyfikatora opartego na sieci neuronowej (multilayer perceptron). W tym celu skorzystamy z następującego kodu:

```
import dalex as dx
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler,
    OneHotEncoder
from sklearn.neural_network import MLPClassifier

classifier = MLPClassifier(hidden_layer_sizes=(150, 100, 50)
    , max_iter=500, random_state=0)

clf = Pipeline(steps=[
```

```
    ('classifier', classifier)
])
clf.fit(X_train4, y_train4)
exp = dx.Explainer(clf, X_train4, y_train4)
```

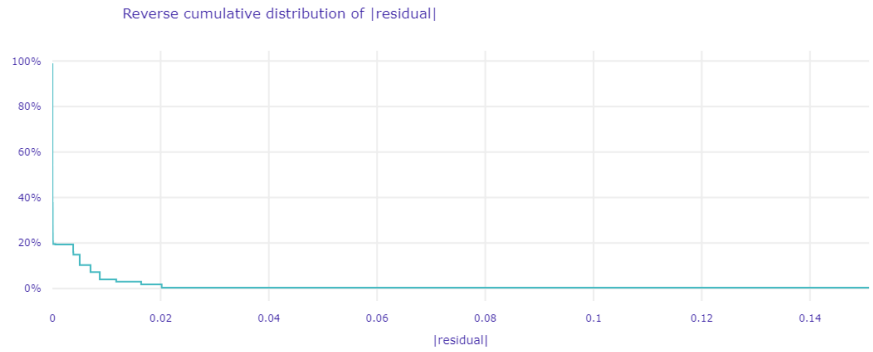
Następnie po wywołaniu kodu:

```
model_performance = exp.model_performance(model_type="
    classification")
model_performance.result
```

sprawdziliśmy wydajność modelu opartego o sieć neuronową. Nie zaskoczyło nas zupełnie, że jest bezbłędny, a jego atrybuty metody pomiaru prezentują się następująco: Powyższą wydajność modelu można zwizualizować pokazując

atrybut metody pomiaru	wartość
Accuracy (dokładność)	1.0
Precision (precyzja)	1.0
Recall (czułość)	1.0
F1	1.0

odwrotny skumuluowany rozkład reszt: Na rysunku możemy zauważyć że sku-



Rysunek 7: Odwrotny skumuluowany rozkład reszt dla MLP klasyfikatora dla zbioru danych 4 zmiennych

mulowany rozkład "szybko" osiąga plateau (zbiega). Jako reszty rozumiemy różnice między wartościami rzeczywistymi a przewidywanymi. Ze względu na to, że one występują tylko na niskich kwintylach, dochodzimy do wniosku, że model bardzo dobrze dopasowuje się do danych.

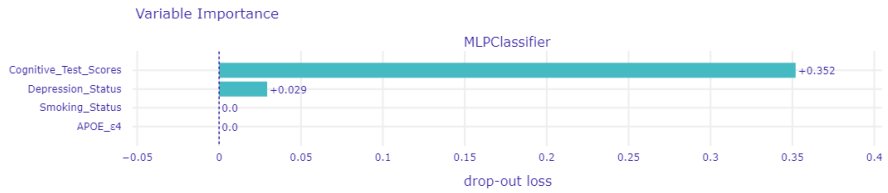
7.2 Istotność zmiennych

Najistotniejszym etapem w wyjaśnianiu modelu uczenia maszynowego jest określenie, które cechy najbardziej istotnie odpowiadają za klasyfikację taką a nie

inną. Wywołujemy w tym celu kod:

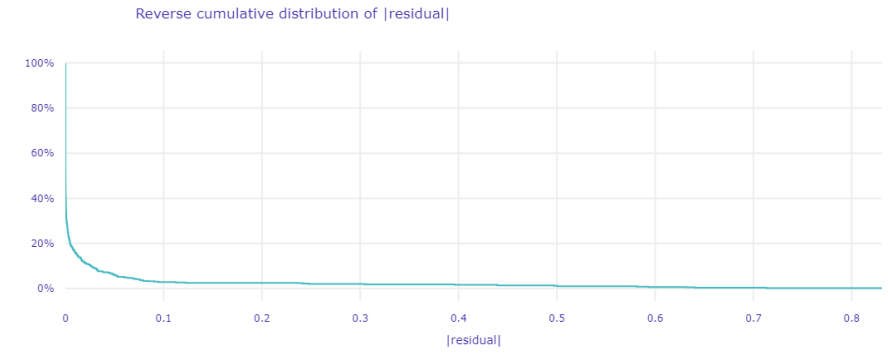
```
variable_importance = exp.model_parts(N=1000, B=15,
    random_state=11)
```

Powyższe wyniki można zwizualizować: Z powyższego wykresu wynika, że zmien-



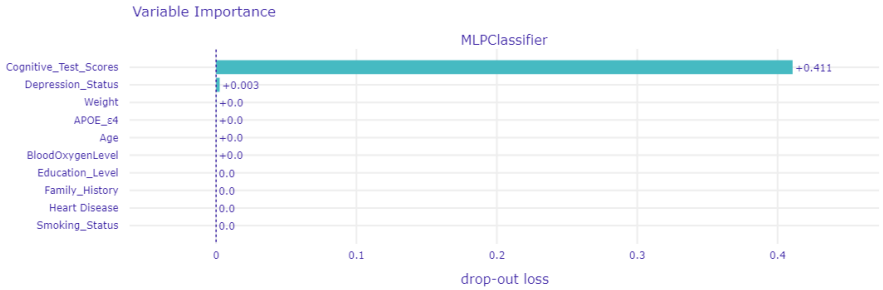
Rysunek 8: Barplot przedstawiający istotność zmiennych dla zbioru 4 zmiennych

na o wyniku testów poznawczych znacząco dominuje model kiedy weźmiemy zbiór wybranych przez nas 4 zmiennych (też dość dużą istotność ma depresja). Zweryfikujmy, czy sytuacja będzie podobna jeśli weźmiemy zbiór danych 10 zmiennych: Jak widzimy powyższy wykres jeszcze płynniej przechodzi do pro-



Rysunek 9: Odwrotny skumuluowany rozkład reszt dla MLP klasyfikatora dla zbioru danych 10 zmiennych

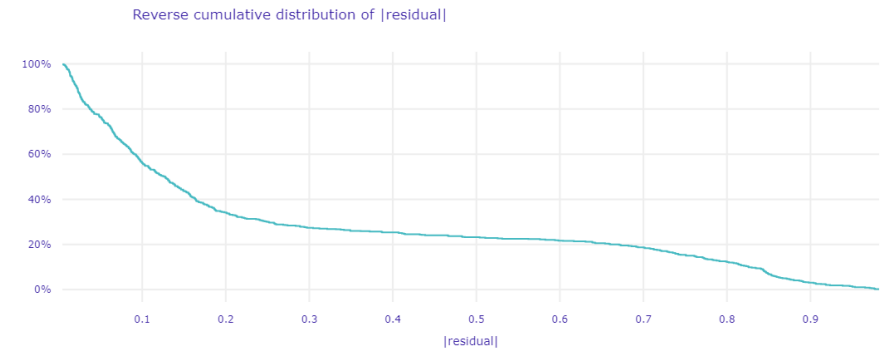
stej linii, zatem dane jeszcze lepiej dopasowują się do modelu. Powyższy wykres jednoznacznie potwierdza hipotezę o dominacji zmiennej wyników testów poznawczych. Oznacza to, że model tak naprawdę jest w stanie wyłącznie w oparciu o wynik tej zmiennej zaklasyfikować pacjenta jako chorego na demencję lub zdrowego. Taki rezultat nas nie zaskakuje, bowiem w rzeczywistości diagnoza demencji opiera się na diagnozie zaburzeń poznawczych u pacjenta.



Rysunek 10: Barplot przedstawiający istotność zmiennych dla zbioru 10 zmiennych

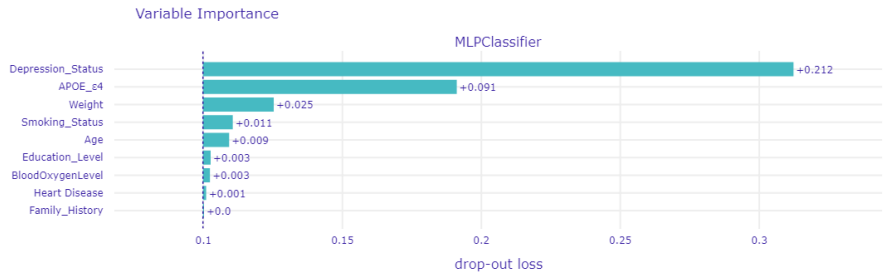
7.3 Usunięcie kolumny dominującej

W celach eksploracyjnych zależało nam na tym, żeby zobaczyć jaka będzie istotność zmiennych kiedy pozbędziemy się dominującej zmiennej mówiącej o wynikach testów poznawczych. Powtórzyliśmy powyższy kod dla zbioru po wykluczenie tej zmiennej, a następnie sporządziliśmy analogiczne wizualizacje (takie jak poprzednio). Widzimy, że bez wyników testów poznawczych, dane bardziej



Rysunek 11: Odwrotny skumuluowany rozkład reszt dla MLP klasyfikatora dla zbioru danych 9 zmiennych (10 bez cognitive test scores)

odbiegają od predykcji. Widoczne są reszty na wysokich kwintylach, a wykres nie osiąga plateau. Po wykluczeniu zmiennej o wynikach testów poznawczych najistotniejszą zmienną okazała się depresja (nic zaskakującego, gdyż na etapie EDA zdaliśmy sobie sprawę, że każda osoba chora na depresję ma demencję). W tym wypadku nie mamy już tak jednoznacznej klasyfikacji w oparciu o jedną zmienną. Widzimy wysoką istotność genu APOE_ε4, co również nas nie zaskakuje ze względu na istniejącą w populacji korelację między występowaniem demencji i choroby Alzheimera. Kilka innych zmiennych (waga, wiek, palenie)



Rysunek 12: Barplot przedstawiający istotność zmiennych dla zbioru 9 zmiennych (10 bez cognitive test scores)

miały również pewną istotność dla modelu.
W ten sposób dochodzimy do konkluzji, że po wykluczeniu zmiennej o wynikach testów poznawczych stworzyliśmy model, który miałby dużo lepsze zastosowanie praktyczne - możemy spróbować przewidzieć demencję bez konieczności przeprowadzania testów i wyłącznie w oparciu o parametry pacjenta.

7.4 Atrybuty metody pomiaru dla nowego zbioru danych

Dla zbioru danych bez wyników testów poznawczych zdecydowaliśmy się przetestować kilka podstawowych modeli:

7.4.1 Baseline dla strategii 'most_frequent'

	klasyfikacja pozytywna	klasyfikacja negatywna
stan pozytywny	0	97
stan negatywny	0	103

atrybut metody pomiaru	wartość
Accuracy (dokładność)	0.515
Precision (precyzja)	0.0
Recall (czułość)	0.0
F1	0.0

7.4.2 Regresja logistyczna

Dla penalty 'None':

	klasyfikacja pozytywna	klasyfikacja negatywna
stan pozytywny	55	42
stan negatywny	15	88

atrybut metody pomiaru	wartość
Accuracy (dokładność)	0.715
Precision (precyzja)	0.786
Recall (czułość)	0.567
F1	0.659

Dla penalty 'l2':

	klasyfikacja pozytywna	klasyfikacja negatywna
stan pozytywny	54	43
stan negatywny	14	89

atrybut metody pomiaru	wartość
Accuracy (dokładność)	0.715
Precision (precyzja)	0.794
Recall (czułość)	0.557
F1	0.655

7.4.3 SVM

	klasyfikacja pozytywna	klasyfikacja negatywna
stan pozytywny	33	64
stan negatywny	39	64

atrybut metody pomiaru	wartość
Accuracy (dokładność)	0.485
Precision (precyzja)	0.458
Recall (czułość)	0.340
F1	0.391

7.4.4 Drzewo decyzyjne

Dla drzewa decyzyjnego atrybuty metody pomiaru wyszły najwyższe, toteż ostatecznie wybraliśmy ten model jako najlepszy dla tego zbioru danych.

	klasyfikacja pozytywna	klasyfikacja negatywna
stan pozytywny	74	23
stan negatywny	22	81

atrybut metody pomiaru	wartość
Accuracy (dokładność)	0.775
Precision (precyzja)	0.771
Recall (czułość)	0.763
F1	0.767

7.4.5 Klasyfikator k sąsiadów

	klasyfikacja pozytywna	klasyfikacja negatywna
stan pozytywny	38	59
stan negatywny	47	56

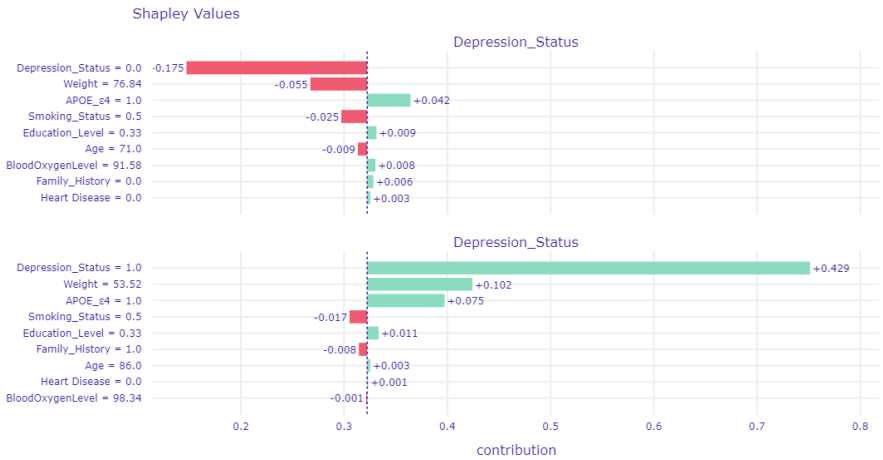
atrybut metody pomiaru	wartość
Accuracy (dokładność)	0.47
Precision (precyzja)	0.447
Recall (czułość)	0.392
F1	0.418

8 Zastosowanie predykcyjne modelu na nowym zbiorze

Spróbujmy zobaczyć jak parametry zdrowotne pojedynczych pacjentów przyczyniają się na istotność w klasyfikacji u nich demencji. W tym celu wybraliśmy dwóch pacjentów - jednego z depresją, jednego bez. Oznaczyliśmy ich jako patient1 i patient2 i opracowaliśmy kod:

```
sh_p1 = exp3.predict_parts(patient1, type="shap", B=50,
    label=patient1.index[0])
sh_p2 = exp3.predict_parts(patient2, type="shap", B=50,
    label=patient2.index[0])
sh_p1.plot(sh_p2)
```

I powstały wykres prezentuje się następująco:



Rysunek 13: Wkład poszczególnych zmiennych w klasyfikację demencji u dwóch wybranych pacjentów wybranych ze zbioru danych

Pokrywa się on z omówieniem istotności zmiennych kiedy wykluczyliśmy wyniki testów poznawczych. Dodatkowo, widzimy że obecność depresji istotnie sugeruje zaklasyfikowanie pacjenta jako chorego na demencję (ze względu na fakt, że każdy chory na depresję miał również demencję według zbioru danych). Tym samym sprowadza to do pytania, czy pozbywając się zmiennej dominującej nie sprowadziliśmy sytuacji do podobnej, tyle że tym razem czynnikiem, który determinuję diagnozę demencji jest wystąpienie depresji. Uważamy, że może to być spowodowane niewielką liczbą danych w zbiorze. Prawdopodobnie dla większego zbioru danych zdolność predykcyjna byłaby lepsza i mniej zdominowana. Istnieje również prawdopodobieństwo, że jest to faktyczne zjawisko występujące w rzeczywistości. Literatura naukowa (druga pozycja w bibliografii) sugeruje, że u osób chorujących na depresję wystąpuje wzmożone ryzyko wystąpienia demencji na starość.

9 Podsumowanie

Podsumowując uzyskane przez nas wyniki, możemy stwierdzić, że demencja jednoznacznie wyznaczana jest przez wyniki testów poznawczych, co powodowało

100% dokładności dla (prawie) każdego modelu.
Przy wycofaniu zmiennej o testach wyników poznawczych nasz model miał naszym zdaniem lepsze zastosowanie w praktyce, kiedy może nie być możliwości przeprowadzenia takiego testu.
Niemniej, fakt, że każda osoba chora na depresję chorowała na demencję, również zaburzał model w stronę klasyfikacji w oparciu o jedną zmienną dominującą.
Nasz zbiór danych był niewielki, rozszerzenie projektu na zbiór o większej liczności miałoby naszym zdaniem większą sensowność i pozwoliłoby skuteczniej przewidywać zjawisko występowania demencji u pacjentów.

10 Walidacja projektu

Podczas instalacji pakietu lightAutoML walidatorzy napotkali problemy techniczne. Kod, który został przez nas przedstawiony, został oecniony jako reprodukowalny i został zweryfikowany przez walidatorów. Walidatorzy wskazali że słusznie potwierdziliśmy nasze przypuszczenia dotyczące wysokiej dokładności modelu i podjeśliśmy próbę zobaczenia, jak wyglądałby ten model bez dominującej zmiennej. Dodatkowo zauważono przydatność projektu w diagnostyce.

11 Bibliografia

Hugo J, Ganguli M. Dementia and cognitive impairment: epidemiology, diagnosis, and treatment. Clin Geriatr Med. 2014 Aug;30(3):421-42. doi: 10.1016/j.cger.2014.04.001. Epub 2014 Jun 12. PMID: 25037289; PMCID: PMC4104432. Kitching D. Depression in dementia. Aust Prescr. 2015 Dec;38(6):209-2011. doi: 10.18773/austprescr.2015.071. Epub 2015 Dec 1. PMID: 26843714; PMCID: PMC4674029.