Zadanie projektowe nr 2

Porównanie klasyfikatorów na bazie „COVID-19, diagnoza”

Maciej Chudziński 253748

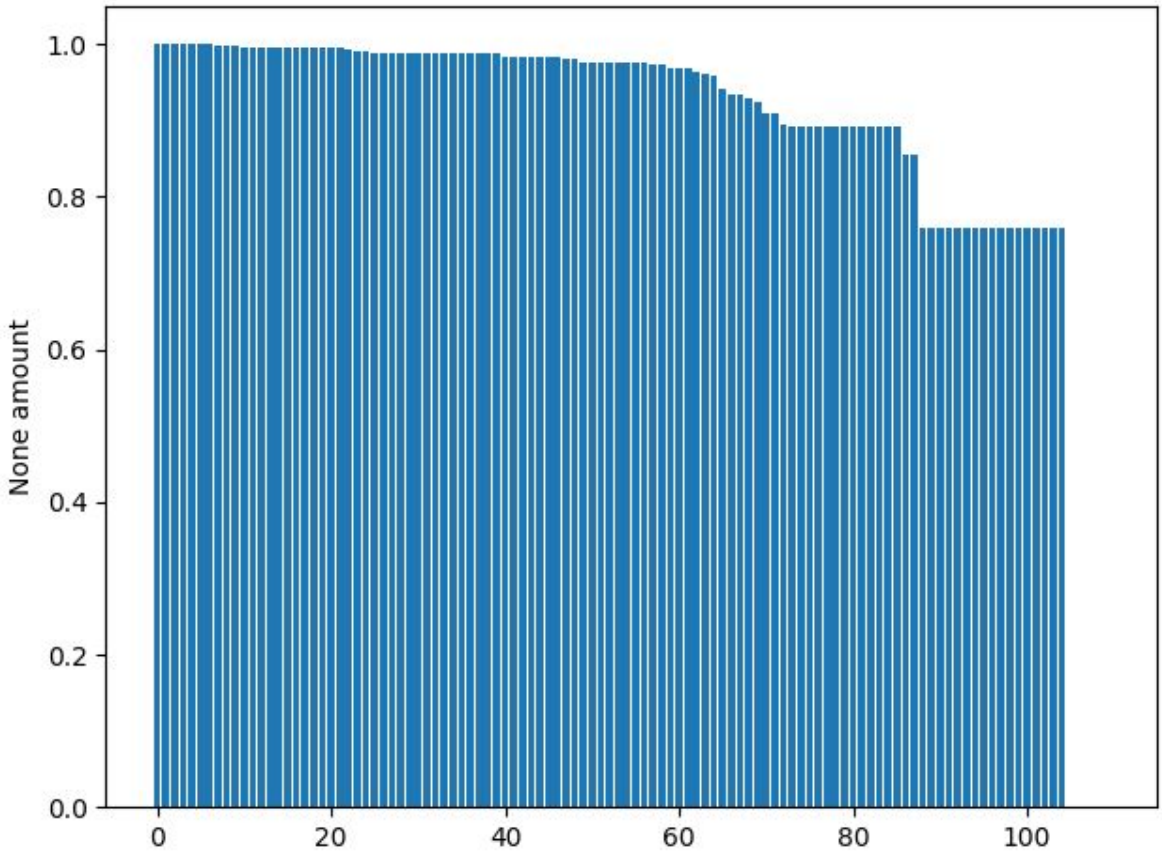
https://github.com/mchudz97/Project-2-IO/

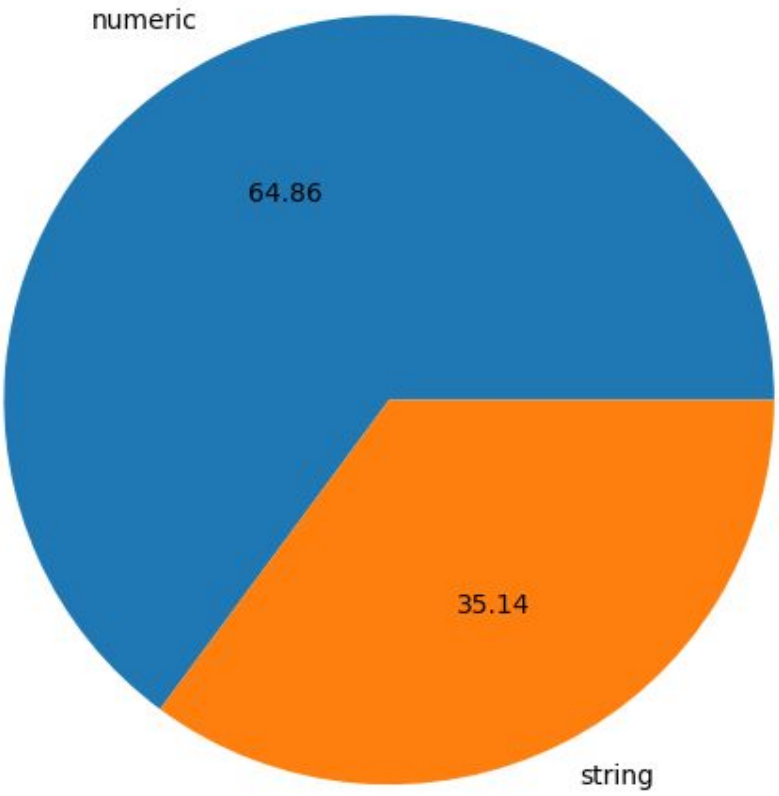
Wstęp

1. Baza danych

Baza danych zawiera 110 kolumn (bez indeksu) oraz 5644 rekordów. Kolumny zawierają dane numeryczne oraz ciągi znaków.

Największym problemem bazy jest bardzo niski poziom zagęszczenia danych. Jeśli wyselektujemy kolumny, które mają powyżej 20% danych to z 110 pozostanie nam ich 22, w czym tylko jedna (wiek pacjenta) będzie numeryczna. Na poniższym wykresie przedstawiono braki w danych względem kolumn.





Wykres typu danych w bazie.

1.1. Obróbka bazy danych

- Podział bazy:
  Całą bazę rozdzieliłem na dwie bazy zawierające dane numeryczne i ciągi znaków

```
NUMERICS = ['int16', 'int32', 'int64', 'float16', 'float32', 'float64']


def return_numerical_columns(df: DataFrame, result_column: DataFrame,  exclude: [] = None):
    df_num = df.select_dtypes(include=NUMERICS)
    if exclude:
        for ex in exclude:
            df_num = df_num.drop(ex, axis=1)

    df_num = df_num.join(result_column)
    return df_num


def return_str_columns(df: DataFrame, result_column: DataFrame, include: [] = None):
    return df.drop(return_numerical_columns(df, result_column, exclude=include).columns.values, axis=1)\
        .join(result_column)
```

- Eliminacja kolumn:

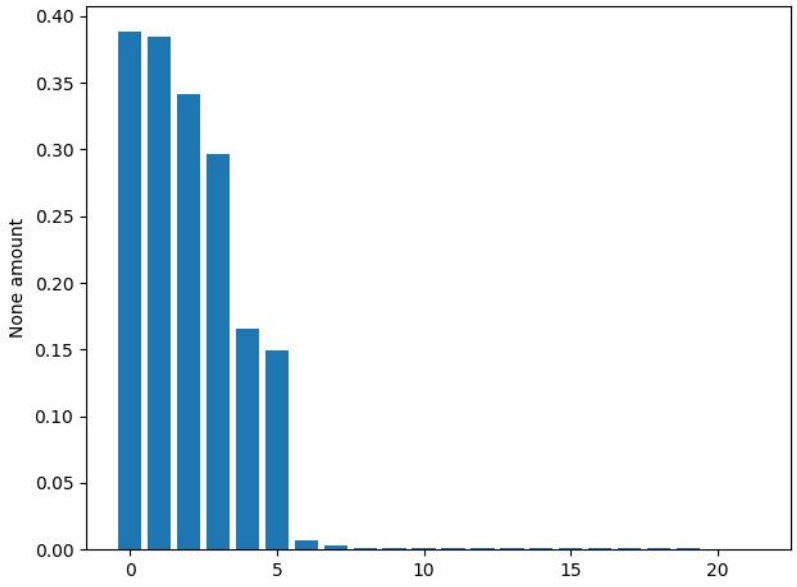
  Podczas badania algorytmów została usunięta część kolumn. Selekcja odbywała się na podstawie zagęszczenia danych. Do projektu użyłem kolumny posiadające minimum 5% danych (45 kolumn).

```python
def reduce_null_columns(data: pd.DataFrame, max_percentage_of_nulls: float):
    df = data.copy()
    print(f'Reduced from {df.shape[1]}')
    for column in df:
        if df[column].isna().sum() / df[column].shape[0] >= max_percentage_of_nulls:
            df = df.drop(column, axis=1)

    print(f'to {df.shape[1]} columns')

    return df
```

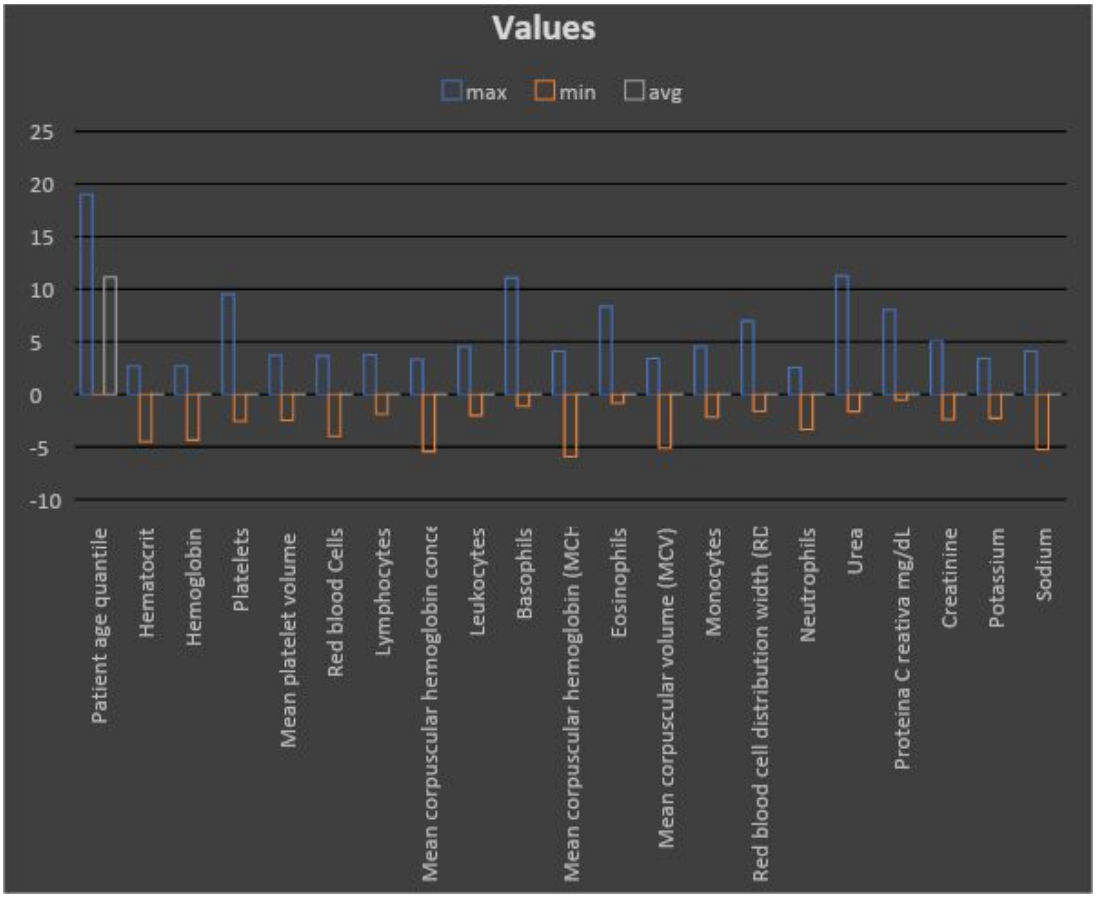
- Eliminacja rekordów:

  Dla bazy numerycznej usunięte zostały rekordy zawierające mniej niż 20% danych.

```python
def select_richest_rows(df: pd.DataFrame, min_percent_of_data: float):
    return df.loc[df.notna().sum(1)/df.shape[1] >= min_percent_of_data]
```



Wykres ilości pustych pól po operacji eliminacji rekordów



Informacje na temat kolumn numerycznych po eliminacji rekordów

- Uzupełnienie brakujących danych:
    - Dla bazy numerycznej puste pola zostały zastąpione medianami dla określonego wyniku testu na występowanie wirusa

```
def fill_nulls_with_medians(numeric_df: pd.DataFrame, y_name: str):
    df_num = numeric_df.copy()
    for col in numeric_df.columns.values:
        if col == y_name:
            continue
        if len(numeric_df.groupby(y_name)[col]) >= 2:
            df_num[col] = numeric_df[col]\
                .fillna(numeric_df.groupby(y_name)[col].transform('mean'))
        else:
            df_num[col] = numeric_df[col].fillna(numeric_df[col].median())

    return df_num
```

    - Dla kolumn nienumerycznych puste pola zostały zastąpione przez „not_tested”

1.2. Efekty obróbki

- Baza numeryczna: 22 kolumny (21 numerycznych + y), 603 rekordy
- Baza pod reguły asocjacyjne: 24, 5644 rekordy

2. Ewaluacja klasyfikatorów

2.1. Użyte algorytmy:

- Naive Bayes
- Drzewa decyzyjne
- k Najbliższych sąsiadów (3n, 5n, 8n)
- Sieci neuronowe (Perceptron i Multilayer Perceptron)
- Random Forest
- Support Vector Machines

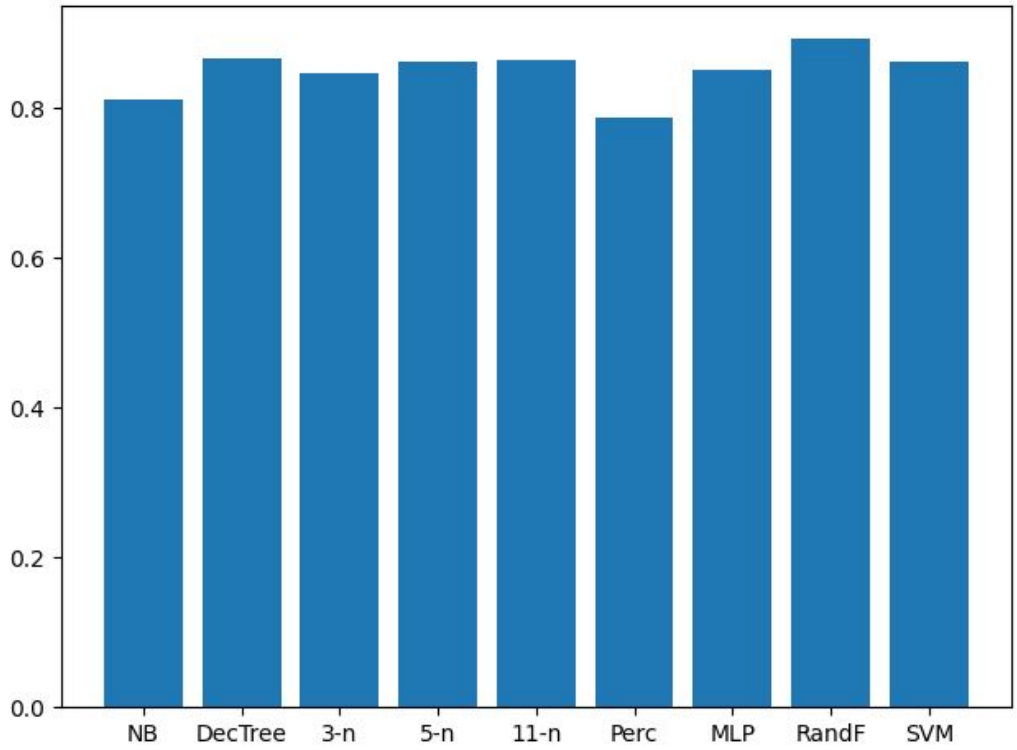
2.1.1. Random Forest

Jest to las zbudowany z losowych drzew decyzyjnych, uśredniający ich wyniki.
Jest bardziej odporny na przeuczenie i bardziej stabilny od drzew decyzyjnych.

2.1.2. SVM

Jest to klasyfikator, który rozdziela klasy za pomocą liniowej hiperpłaszczyzny o możliwie najszerszym zakresie.

2.2. Dokładność

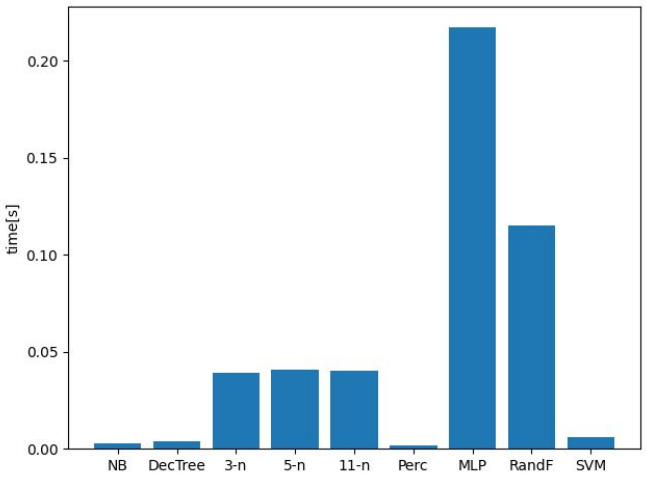


2.3. Macierze błędu

		Naive Bayes		Decision Tree		3-neighbors		5-neighbors		11-neighbors	
		prawdziwa		prawdziwa		prawdziwa		prawdziwa		prawdziwa	
		negative	positive	negative	positive	negative	positive	negative	positive	negative	positive
przewidywana	negative	287	19	320	30	318	36	326	38	334	45
	positive	56	36	23	25	19	19	17	17	9	10

		Perceptron		Multilayer Perceptron		Random Forest		SVM	
		prawdziwa		prawdziwa		prawdziwa		prawdziwa	
		negative	positive	negative	positive	negative	positive	negative	positive
przewidywana	negative	288	30	335	51	339	39	343	55
	positive	55	25	8	4	4	16	0	0

2.4. Czasy



3. Reguły asocjacyjne

Badając bazę natknąłem się na ciekawą regułę.

antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
(Rhinovirus/Enterovirus_detected)	(SARS-Cov-2 exam result_negative)	0.067151	0.901134	0.066088	0.984169	1.092145	0.005576	6.245039

Weryfikując wykryte wirusy spostrzegłem, że Rhinowirus ma taki sam objaw co Covid-19. Cytując:

Rynowirusy odpowiadają za około połowę przeziębień każdego roku oraz są przyczyną większości chorób układu oddechowego w okresie wiosennym i jesiennym.

Biorąc to pod uwagę, można wnioskować, że przez te objawy, niektóre osoby mogły być mylnie uznawane za zarażone koronawirusem.