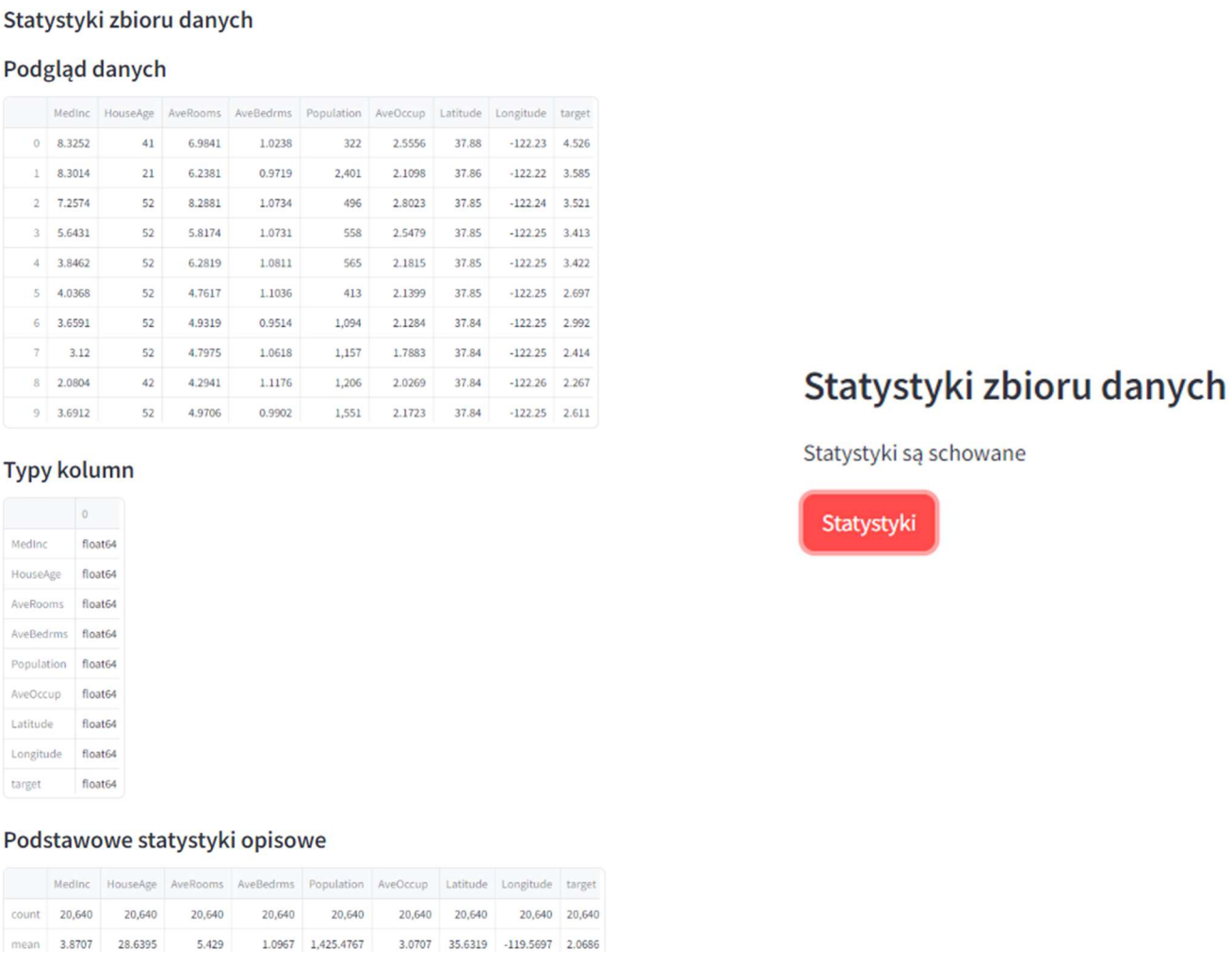
Ćwiczenie: Interaktywna EDA

Stworzona przy użyciu biblioteki Streamlit aplikacja, pozwala przeprowadzić eksploracyjną analizę danych w sposób interaktywny. Dzięki możliwości stworzenia interfejsu użytkownika, dodanie dynamicznych wykresów czy przycisków, zwiększa się przejrzystość – można wyświetlić wybrane wyniki przetwarzania danych, a cały kod pozostaje ukryty. Aplikacja stworzona z pomocą streamlit może zostać zaprezentowana osobom nietechnicznym, które bez konieczności znajomości programowania mogą zobaczyć rezultaty.

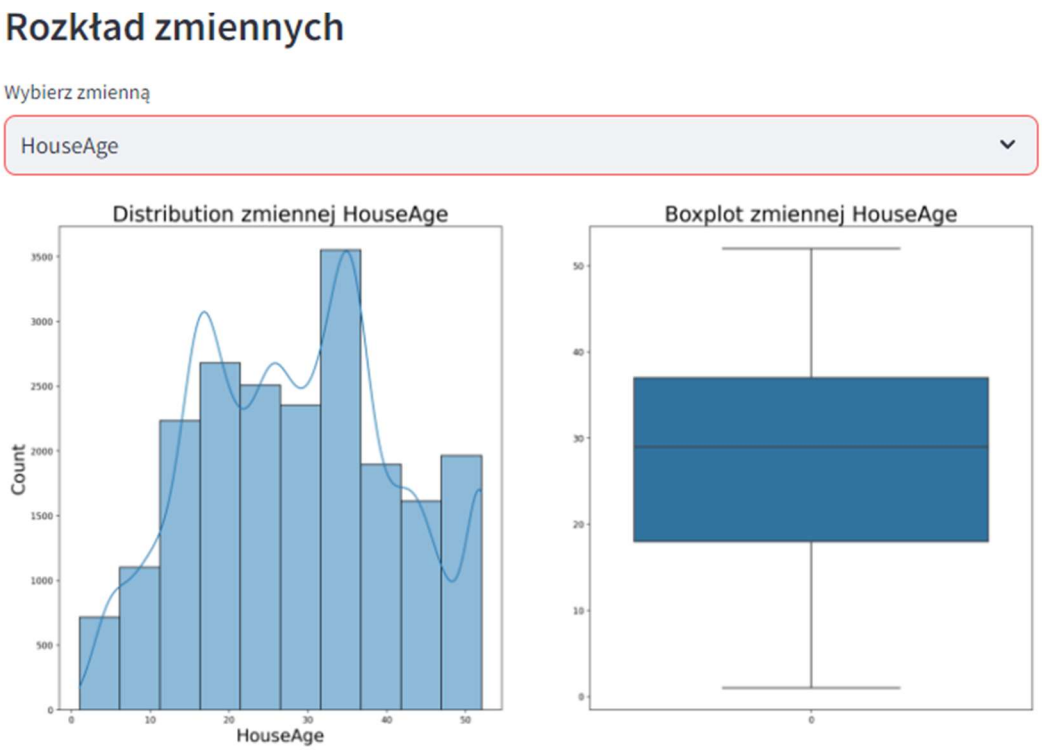
Aplikacja wykonana na potrzeby ćwiczenia jest podzielona na sekcje:

1. Podgląd danych.
2. Rozkład zmiennych.
3. Macierz korelacji.
4. Rozrzut zmiennych.

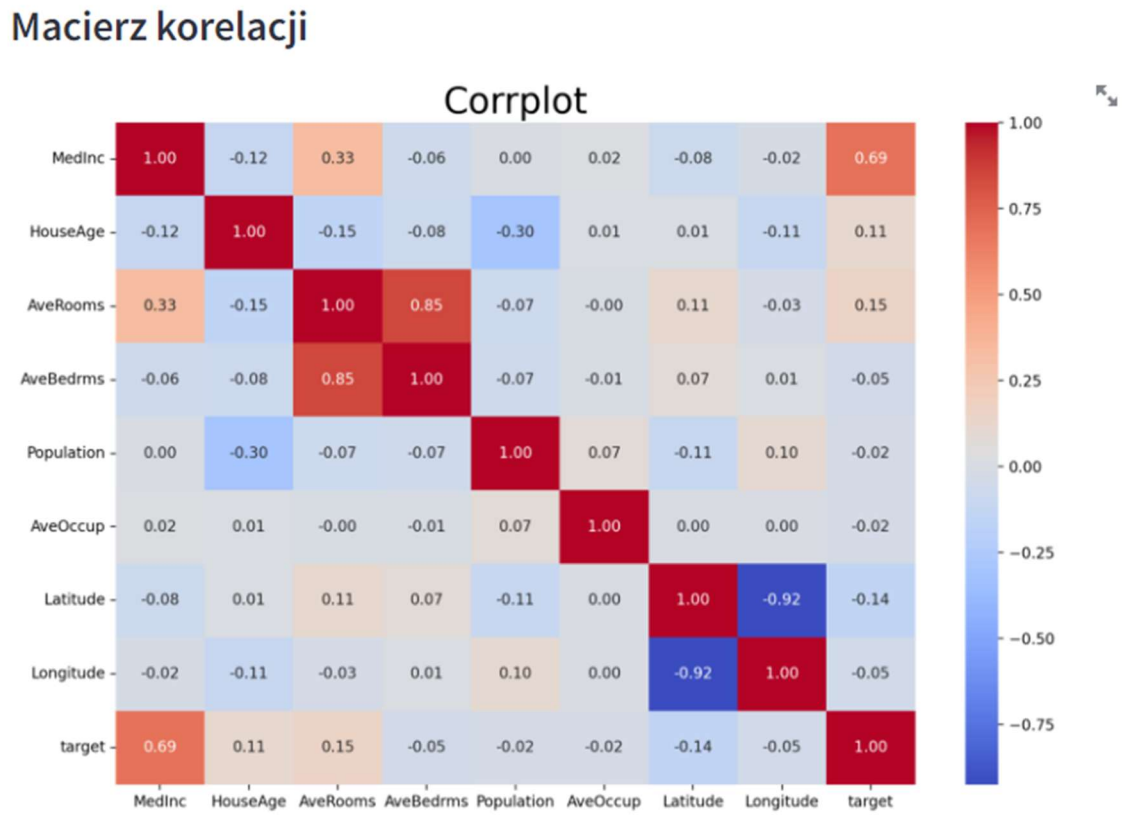
Ad 1. Podgląd danych to sekcja, którą można rozwinąć lub ukryć i zawiera statystyki opisowe, próbkę zbioru danych oraz informacje o nim.



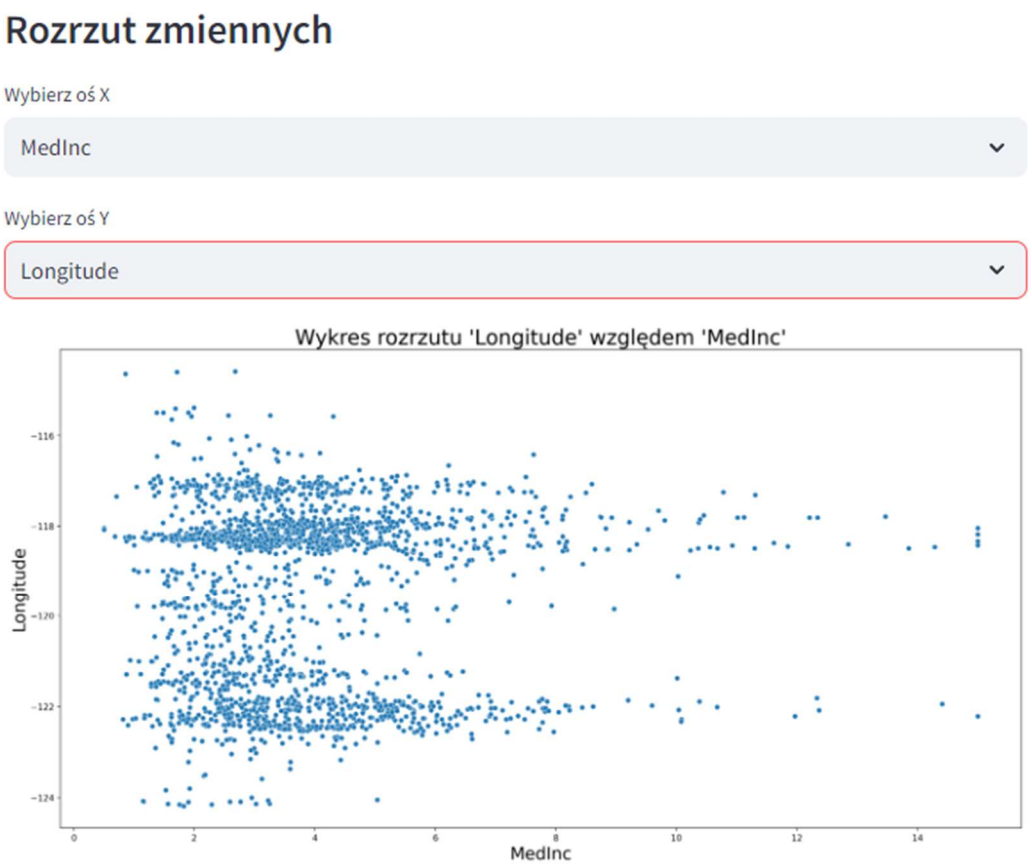
Ad 2. Sekcja rozkład zmiennych zawiera drop-down, na którym można wybrać wyświetlaną zmienną. Wykresy to histogram oraz boxplot.



Ad 3. Sekcja Macierz korelacji to wykres prezentujący jak wygląda wzajemne skorelowanie między zmiennymi.



Ad 4. Sekcja Rozrzut zmiennych zawiera wykres rozrzutu z możliwością wyboru kolumny na osi X oraz Y



Kod który posłużył do wykonania aplikacji:

```
import streamlit as st
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import load_iris
from sklearn.datasets import fetch_california_housing
import matplotlib as mpl


@st.cache
def load_housing_data():
    housing = fetch_california_housing()
    df = pd.DataFrame(data=housing.data, columns=housing.feature_names)

    df['target'] = housing.target
    return df
```

```python
def show_basic_stats(data):
    st.subheader("Podgląd danych")
    st.write(data.head(10))
    st.subheader("Typy kolumn")
    st.write(data.dtypes)
    st.subheader("Podstawowe statystyki opisowe")
    st.write(data.describe())
    st.subheader("Wartości brakujące")
    st.write(data.isnull().sum())


def handle_stats_button(data):
    st.subheader("Statystyki zbioru danych")
    if 'button_clicked' not in st.session_state:
        st.session_state.button_clicked = False

    # Render DataFrame if button is clicked
    if st.session_state.button_clicked:
        show_basic_stats(data)
    else:
        st.markdown("Statystyki są schowane")

    # Button to toggle DataFrame visibility
    if st.button("Statystyki"):
        st.session_state.button_clicked = not st.session_state.button_clicked

# Funkcja do wizualizacji rozkładu zmiennych
def show_distribution_plots(data):
    st.subheader("Rozkład zmiennych")
    mpl.rcParams['axes.labelsize'] = 20
    mpl.rcParams['axes.titlesize'] = 24

    selected_col = st.selectbox("Wybierz zmienną", data.columns)

    if selected_col:
        fig, ax = plt.subplots(1,2,figsize=(20,10))

        col = data[data[selected_col] <= data[selected_col].mean() + 3 *
data[selected_col].std()][selected_col]
        sns.histplot(x=col, bins=10, kde=True, ax=ax[0])
        sns.boxplot(data=col, ax=ax[1])

        ax[0].set_title(f"Distribution zmiennej {selected_col}")
        ax[1].set_title(f"Boxplot zmiennej {selected_col}")

        st.pyplot(fig)

def show_corrplot(data):
    st.subheader("Macierz korelacji")
```

```
    correlation_matrix = data.corr()

    fig, ax = plt.subplots(figsize=(12,8))
    # Create correlation plot using Seaborn
    sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", ax=ax)
    ax.set_title("Corrplot")
    st.pyplot(fig)


def show_scatterplot(data):
    st.subheader("Rozrzut zmiennych")
    data_frac = data.sample(frac=0.1, random_state=42)
    selected_col_x = st.selectbox("Wybierz oś X", data_frac.columns)
    selected_col_y = st.selectbox("Wybierz oś Y", data.columns)
# Plot chart based on selected column
    if selected_col_x and selected_col_y:
        fig, ax = plt.subplots(figsize=(20,10))
        sns.scatterplot(x=data_frac[selected_col_x], y=data_frac[selected_col_y],ax=ax)

        ax.set_title(f"Wykres rozrzutu '{selected_col_y}' względem '{selected_col_x}'")

        st.pyplot(fig)


def main():

    housing_data = load_housing_data()

    handle_stats_button(housing_data)

    show_distribution_plots(housing_data)

    show_corrplot(housing_data)

    show_scatterplot(housing_data)


if __name__ == "__main__":
    main()
```