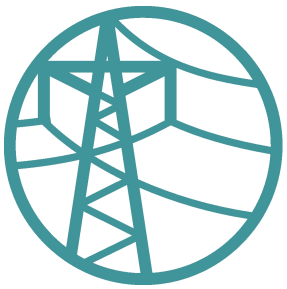


WYDZIAŁ
ELEKTROTECHNIKI
I INFORMATYKI
POLITECHNIKI RZESZOWSKIEJ

Krzysztof Lang

Implementacja wybranych algorytmów wypełniania brakujących wartości, dla strumieni dużych zbiorów danych

Praca dyplomowa magisterska

Opiekun pracy:
dr Michał Piętal

Rzeszów, 2023

Spis treści

1. Wstęp

2. Wprowadzenie do wypełniania brakujących wartości

2.1. Rys historyczny

2.2. Na czym polega wypełnianie brakujących wartości

2.3. Korzyści i zagrożenia

2.4. Perspektywy na przyszłość

3. Omówienie narzędzi i danych

3.1. Python

Do przygotowania programu wykorzystanego do przeprowadzenia badań wybrano język Python. Jest to język wysokiego poziomu, charakteryzujący się prostą składną i wysoką przejrzystością kodu. Programy nie muszą być kompilowane przed uruchomieniem, co znacznie przyśpiesza proces prototypowania i debugowania. Oznacza to też że Python jest wolniejszy od wielu innych języków, jednak w przypadku niniejszej pracy nie ma to znaczenia. Dostępna ogromna ilość gotowych bibliotek służących do obróbki i analizy danych znacząco uprościła przygotowanie programu. Podczas pisania kodu trzymano się dobrych praktyk, stosowano wytyczne zawarte w PEP8.

3.2. Visual Studio Code

Jako środowisko programowania wybrano "Microsoft Visual Studio Code". Jest to darmowy edytor kodu obsługujący wiele języków. Ze względu na otwartość kodu, dostępne jest wiele rozszerzeń do programu, które znacznie ułatwiają tworzenie nawet skomplikowanych projektów. W celu umożliwienia pracy nad programem z wielu urządzeń oraz dla zachowania pełnej historii tworzenia programu wykorzystano integrację "Visual Studio Code"z repozytorium GitHub.

3.3. Biblioteki

3.3.1. Pandas

3.3.2. NumPy

3.3.3. Scikit

3.3.4. EasyGUI

3.4. Źródła danych

3.4.1. Użyte repozytopria danych

Aby wyniki badań niosły ze sobą odpowiednią wartość merytoryczną, potrzebne są odpowiednie zbiory danych na których zostaną przeprowadzone testy W celu znalezienia odpowiednich zbiorów danych, przyjęto następujące założenia:

- zbiór danych musi być wystarczająco duży,
- zbiór danych musi zawierać odpowiedniż ilość atrybutów aby modele decyzyjne

miały do dyspozycji wystarczającą ilość danych uczących,

- atrybuty powinny zawierać różnorodne typy danych w celu przetestowania wypełniania zarówno danych liczbowych (całkowitych i zmiennoprzecinkowych) jak i kategorycznych,

- zbiór danych nie może mieć pustych wartości.

Do wyszukania odpowiednich zbiorów danych wykorzystano narzędzie "Google Dataset Search". Z jego pomocą wybrano 2 zbiory danych z różnych dziedzin. Po uprzedniej ich obróbce zostały wykorzystane do przeprowadzenia testów algorytów wypełniania.

3.4.2. Adult Data Set

Zbiór danych "Adult Data Set"zawiera dane ze spisu ludności przeprowadzonego w roku 1994 w Stanach Zjednoczonych. Jest szeroko wykorzystywany do testowania uczenia maszynowego. Zawiera ponad 30000 rekordów i 15 atrybutów. [1] Opis atrybutów:

- age: wiek spisanej osoby, liczba całkowita,

- workclass: rodzaj zatrudnienia, dane kategoryczne, 8 możliwych wartości,

- fnlwgt: jaka proporcja populacji ma identyczny zestaw pozostałych wartości, liczba całkowita,

- education: osiągnięty poziom edukacji, dane kategoryczne, 16 możliwych wartości,

- education-num: osiągnięty poziom edukacji zakodowany jako liczba całkowita,

- martial-status: status matrymonialny, dane kategoryczne, 7 możliwych wartości,

- occupation: zawód, dane kategoryczne, 14 możliwych wartości,

- relationship: rola w związku, dane kategoryczne, 6 możliwych wartości,

- race: klasyfikacja rasowa, dane kategoryczne, 5 możliwych wartości,

- sex: płeć, dane kategoryczne, 2 możliwe wartości,

- capital-gain: zysk kapitału w zwiazku z inwestycjami, liczba całkowita,

- capital-gain: strata kapitału w zwiazku z inwestycjami, liczba całkowita,
- hours-per-week: ilość godzin pracujących w tygodniu, liczba całkowita,
- native-country: kraj pochodzenia, dane kategoryczne, 41 możliwych wartości,
- attribute: czy osoba zarabia powyżej czy poniżej 50000$ rocznie.

Ten zbiór danych został wybrany ze względu na występowanie zarówno atrybutów liczbowych jak i kategorycznych, zadowalajacą ilość rekordów oraz atrybutów. Ma na celu przetestowanie skuteczności działania algorytmów do wypełniania brakujących miejsc w zbiorach danych z brakami w danych o różnych typach. Nie wymaga dodatkowej obróbki przed rozpoczęciem testów.

3.4.3. Stock Exchange Data

Zbiór danych "Stock Exchange Data"zawiera informacje o cenach akcji na giełdach w różnych krajach w latach 1965-2021. Dane zostały zebrane z "Yahoo Finance", posiadającego dane o giełdzie z wielu lat w wielu krajach. Posiada ponad 100000 rekordów i 9 atrybutów. [2] Opis atrybutów:

- Index: symbol wskazujacy z jakiej giełdy pochodzą dane, dane kategoryczne, 5 możliwych wartości,
- Date: data obserwacji, dane kategoryczne,
- Open: cena akcji podczas otwarcia, liczba wymierna,
- High: najwyższa cena w ciągu dnia, liczba wymierna,
- Low: najniższa cena w ciągu dnia, liczba wymierna,
- Close: cena akcji w momencie zamknięcia, liczba wymierna,
- Adj Close: cena akcji w momencie zamknięcia skorygowana o podziały jak i dywidendy, liczba wymierna,
- Volume: liczba akcji będących przedmiotem obrotu w ciągu dnia sesyjnego, liczba całkowita,
- CloseUSD: cana akcji w momencie zamknięcia wyrażona w dolarach amerykańskich

Ten zbiór danych został wybrany ze względu na bardzo popularną kategorię danych, to jest dane finansowe. Ma na celu przetestowanie skuteczności działania algorytmów w przypadku danych numerycznych, w szegółności liczb wymiernych. W celu lepszego przygotowania do testów zakodowano kolumnę "Data"z wykorzystaniem "label encoding", to jest zamiany danych na postać numeryczną. Usunięto też rekordy posiadające wartość "0"w kolumnie "Volume". Ich duża ilość (ponad 30%) mogła by negatywnie wpłynąć na uczenie modeli decyzyjnych. W wyniku tego zmniejszono liczbę rekordów do ponad 62000, co wciąż jest ilością spełniajacą założenia dla zbiorów danych.

4. Opis przygotowanego programu

4.1. Założenia i realizacja

Założono, że program ma rtealizować 3 zadania:

1) Przygotować dane do wypełniania poprzez sztuczne utworzenie brakujacych wartości.

2) Wypełnić brakujące wartości z wykorzystaniem wybranych algorytmów.

3) Ocenić skuteczność wypełniania w celu porównania algorytmów.

Poszczególne zadania zrealizowano jako osobne moduły.
Przyjęto też następujące założenia:

1) Wykorzystanym językiem ma być Python.

2) Program ma być napisany zgodnie z paradygmatem programowania obiektowego.

3) Poszczególne klasy mają być zawarte w osobnych plikach.

4) Interakcja z programem ma opierać się o prosty interfejs graficzny.

5) Dostęp do wszystkich modułów programu ma być zapewniony z jednego miejsca.

6) Pliki wygenerowane przez jeden moduł mają być przygotowane w sposób umożliwiający wykorzystanie ich przez kolejny. Oprócz odpowiedniego formatowania wewnątrz pliku, oznacza to przyjęcie konwencji nazewnictwa plików opartej o prefiksy i sufiksy.

Program składa się z następujacych plików:

- mgr_main.py: główny plik nie zawierający żadnej klasy, odpowiadający za wybór modułu do uruchomienia, i uruchomienie odpowiedniego modułu po wybraniu,

- mgr_nan_gen"plik zawierajacy klasę "NanGen", odpowiadającą za realizację modułu przygotowujacego plik,

- mgr_fill.py: plik zawierający klasę "Fill", odpowiadającą za realizację modułu wypełniajacego brakujące dane,

- mgr_data.py: plik zawierający klasę "Data", odpowiadającą za wybranie pliku do wypełnienia i przygotowanie do do dalszej obróbki, oraz klasę "PrepareData", odpowiadającą za przygotowanie danych do przekazania silnikowi uczenia maszynowego celem wypełnienia oraz późniejszemu przywróceniu danym ich pierwotnego wyglądu

- mgr_di.py: plik zawierjący klasę DownImpu, odpowiadającą za przygotowanie danych dla algorytmu "Downward Imputation"

- mgr_temp_fill: plik zawierający klasę "TempFill", odpowiadającą za tymczasowe wypełnianie brakujących miejsc, potrzebne podczas przygotowywania danych dla algorytmu "Prostego"

- mgr_acc: plik zawierający klasę "AccuracyTester", odpowiadającą za obliczanie skuteczności wypełniania danych

Wykorzystując narzędzie "auto-py-to-exe", utworzono plik "mgr_suite.exe", pozwalajacy uruchomić program bez konieczności instalowania interpretera Python i potrzebnych bibliotek.

4.2. Działanie programu

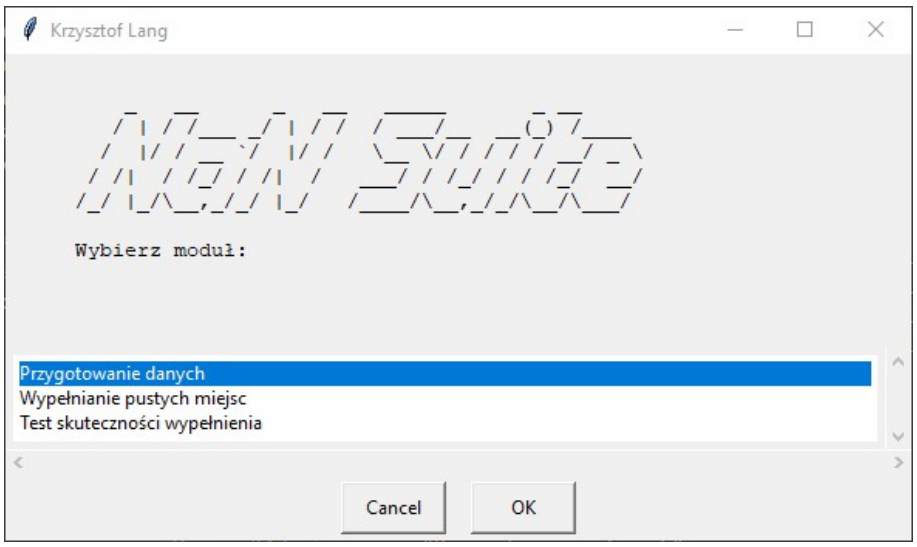
Poniżej zaprezentowano działanie programu na przykładzie pliku data_stock.csv. Zostanie on najpierw przygotowany do testów, następnie brakujace dane zostaną wypełnione z wykorzystaniem jednego z algorytmów, po czym zostanie obliczona dokładność tego wypełnienia.

4.2.1. Uruchomienie

Po uruchomieniu "mgr_suite.exe"wyświetlone zostaje okno służące do wyboru modułu, pokazane na rysunku 4.1.

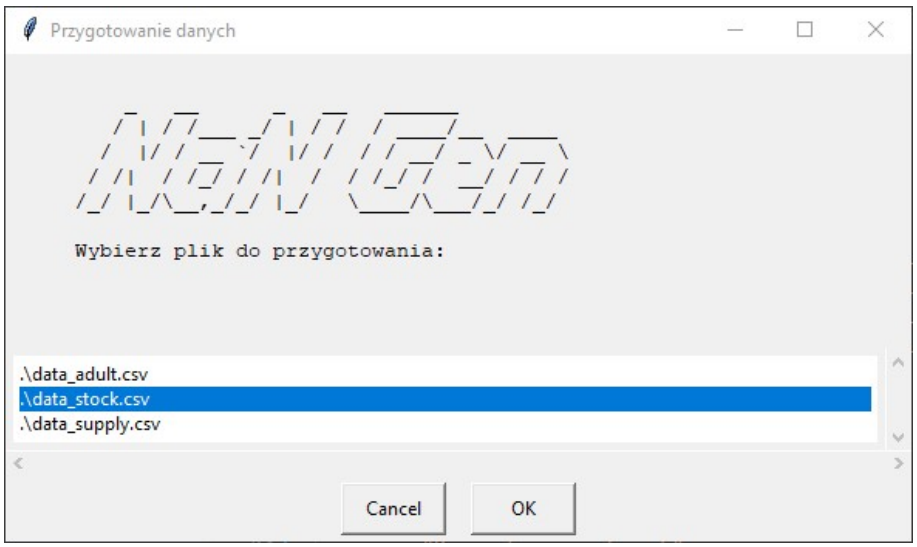
4.2.2. Przygotowanie danych

Pierwszy moduł odpowiada za przygotowanie danych do wypełniania poprzez usunięcie losowych wartości ze zbioru danych. Pierwszym krokiem jest wybranie pliku



Rysunek 4.1: Główne okno programu, pozwalajace na wybór modułu do uruchomienia

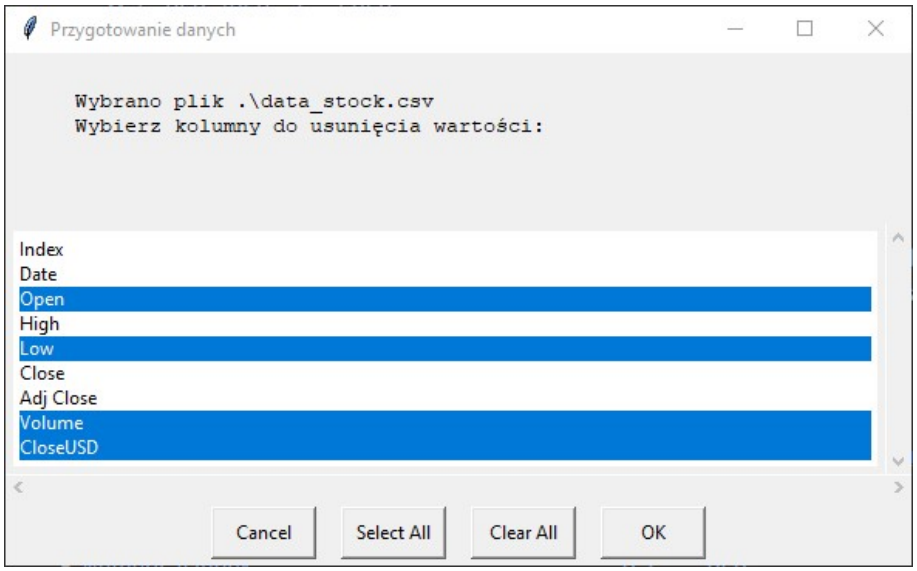
który ma zostać przygotowany z użyciem okna pokazanego na rysunku 4.2. Lista plików generowana jest na podstawie plików znajdujących się w tym samym folderze co uruchamiany program spełniających założony format nazwy. Założono, że pliki które nadają się do wypełnienia mają być zapisane w formacie CSV, natomiast nazwa zaczynać się ma od prefiksu "data_"i nie posiadać żadnych sufiksów.



Rysunek 4.2: Okno wyboru pliku do przygotowania

Następnie wybrane z listy zostają kolumny w których maja zostać usunięte dane. Wyboru dokonuje się z użyciem okna pokazanego na rysunku 4.3. Kolumny do wyboru zaczerpnięte są bezpośrednio z załadowanego wcześniej pliku. Wybrać można dowolną

ilość, lecz zalecane jest poniżej 50%.



Rysunek 4.3: Okno wyboru kolumn

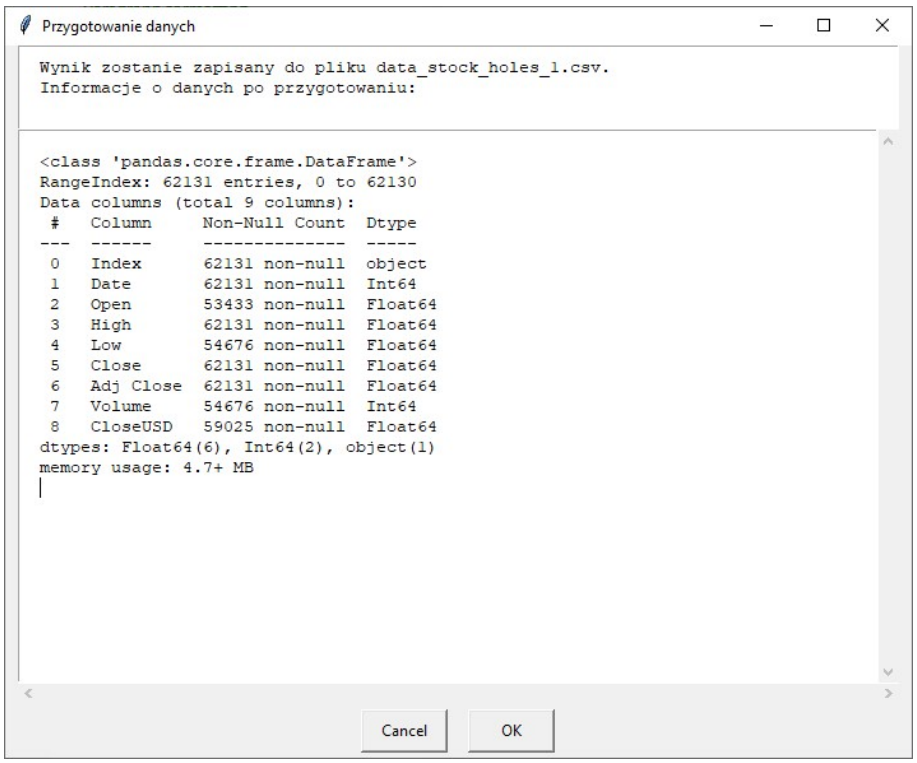
W wybranych kolumnach zostaje usunięte od 5% do 15% wartości - dla każdej kolumny ilość jest losowana. Dodatkowo stworzony zostaje plik przechowujęcy informacje które wartości zostały usunięte. Ta informacja zostaje później wykorzystana do oceny skuteczności wypełnienia zbioru danych. Po zakończeniu usuwania wartości wyświetlone zostaje okno z podsumowaniem jak na rysunku 4.4. Nazwa utworzonego pliku z gotowymi danymi tworzona jest poprzez dodanie sufiksu "_holes_X"do nazwy oryginalnego pliku, gdzie X to kolejna liczba naturalna. Umożliwia to tworzenie plików z danymi usuniętymi z różnych zestawów kolumn bez konieczności ręcznej zmiany ich nazw. Nazwa pliku z informacją które dane zostały usunięte tworzona jest przez dodanie sufiksu "_journal"do nazwy oryginalnego pliku.

4.2.3. Wypełnianie brakujących wartości

Drugi moduł służy do wypełniania brakujacych wartości w zbiorze danych z użyciem wybranego algorytmu.

Najpierw należy wskazać plik który ma zostać wypełniony z użyciem okna wyboru pokazanego na rysunku 4.5. Tak jak wcześniej, lista tworzona jest na podstawie plików w folderze i przyjętej konwencji nazewnictwa plików. Wyświetlane są wyłącznie pliki posiadające sufiks "_holes_X"w nazwie, bez kolejnych sufiksów.

Następnie z użyciem okna jak na rysunku 4.6 wybrany zostaje algorytm który ma zostać wykorzystany do wypełniania brakujących wartości. Wyświetlona zostaje



Rysunek 4.4: Okno z podsumowaniem

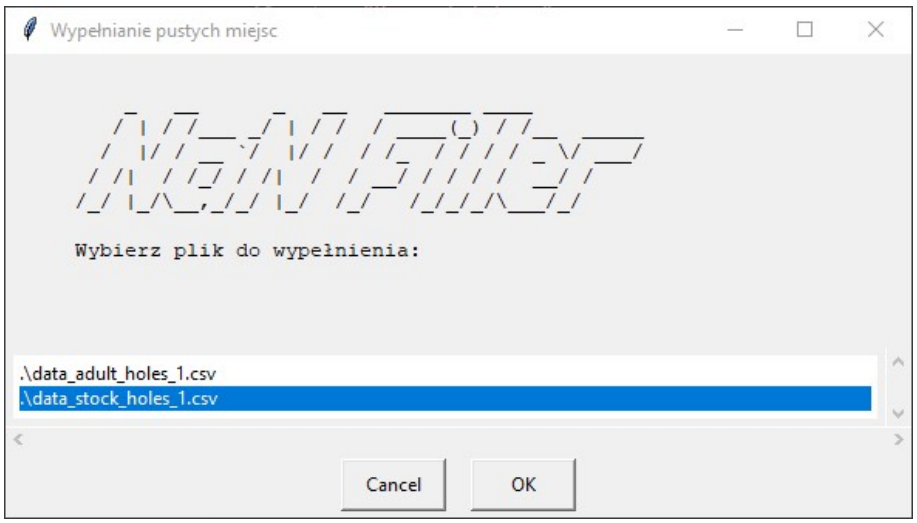
też nazwa wybranego wcześniej pliku w celu uniknięcia błędu wybrania niewłaściwego pliku.

Puste miejsca zostają wypełnione z wykorzystaniem wybranego algorytmu. Dokładny sposób działania algorytmów zostanie opisany w kolejnych rozdziałach. Po zakończeniu wypełniania wyświetlone zostaje podsumowanie jak na rysunku 4.7. Nazwa pliku wynikowego tworzona jest poprzez dodanie sufixu "_filled_Y", gdzie Y to nazwa wybranego algorytmu.

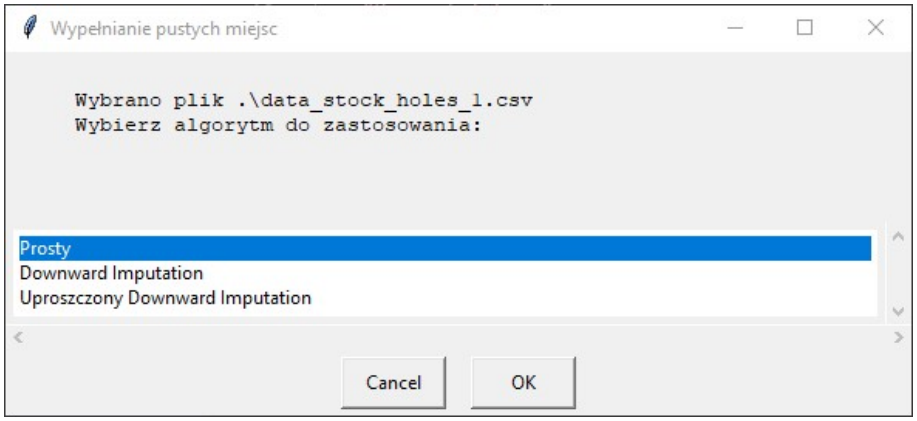
4.2.4. Sprawdzenie skuteczności wypełniania

Ostatni moduł odpowiada za wygenerowanie danych, które można wykorzystać do oceny skuteczności wypełniania brakujących wartości. Jako wystarczające uznano procent skuteczności wypełniania dla danych kategorycznych i liczb całkowitych oraz średnie odchylenie bezwzględne dla wszystkich danych liczbowych. Obie wartości są obliczane dla poszczególnych wypełnionych kolumn.

Jak w przypadku poprzednich modułów, zacząć należy od wyboru pliku który ma zostać poddany analizie. Wyboru dokonuje się z użyciem okna pokazanego na rysunku 4.8. Wyświetlane są tylko pliki zawierające słowo "filled"w nazwie, ponieważ



Rysunek 4.5: Okno wyboru pliku do wypełnienia



Rysunek 4.6: Okno wyboru algorytmu

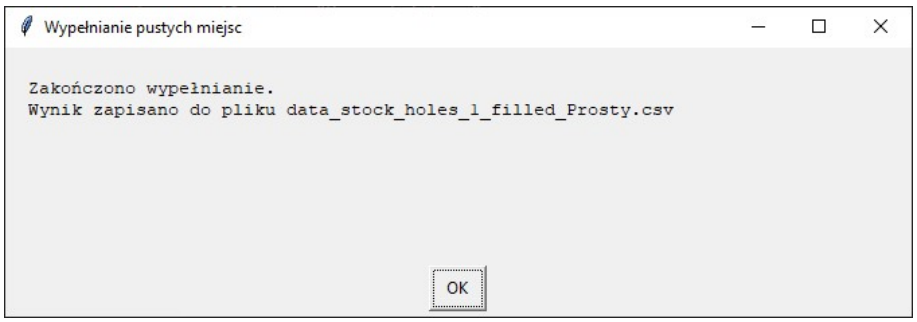
takie pliki posiadają wartości wypełnione za pomocą któregoś algorytmu z użyciem odpowiedniego modułu.

Z użyciem wybranego pliku, oryginalnego pliku z danymi przed usunięciem losowych danych oraz pliku z informacją które dane zostały usunięte a następnie wypełnione, przeprowadzane jest obliczanie skuteczności wypełniania danych.

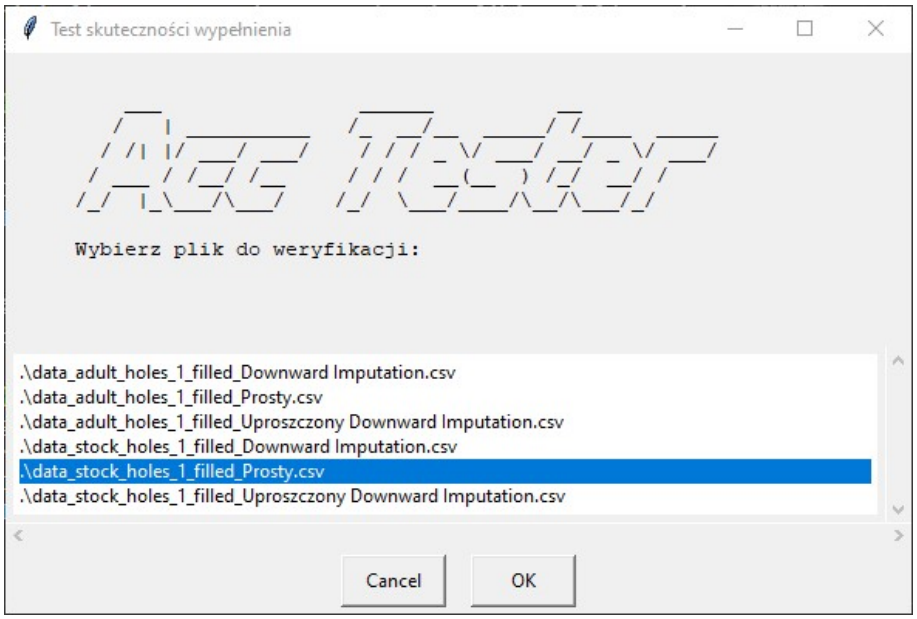
Obliczenie procentowej skuteczności wypełnienia przebiega następująco:

1) Sprawdzenie ilości wypełnianych wartości w danej kolumnie.

2) Zsumowanie ilości poprawnie wypełnionych danych w kolumnie poprzez porównanie wartości o współrzędnych zapisanych w pliku tworzonym podczas przygotowywania danych.



Rysunek 4.7: Okno z podsumowaniem wypełniania



Rysunek 4.8: Okno wyboru pliku do analizy

3) Zastosować wzór 4.1:

$$ACC = \frac{m}{n} \times 100\% \tag{4.1}$$

gdzie: ACC – procentowa skuteczność wypełnienia kolumny, m – ilość poprawnie wypełnionych wartości, n – ilość wypełnionych wartości.

Aby obliczyć średnie odchylenie bezwzględne należy zastosować wzór 4.2:

$$AAD = \frac{\sum_{i=1}^{n} |x_i - \hat{x}_i|}{n} \tag{4.2}$$

gdzie: AAD – średnie odchylenie bezwzględne dla danej kolumny, n – ilość wypełnionych wartości w kolumnie, x_i – wartość wypełnionego i-tego elementu kolumny, $\hat{x}_i$ – oryginalna wartość i-tego elementu kolumny.

Po zakończeniu obliczeń, wyświetlane jest podsumowanie jak na rysunku 4.9, a wynik obliczeń zapisywany jest w dwóch plikach

- wyniki obliczania procentowej skuteczności jest zapisywany w pliku o nazwie tworzonej przez dodanie do nazwy analizowanego pliku sufiksu "_aad",
- wyniki obliczania średniego odchylenia bezwzględnego jest zapisywany w pliku o nazwie tworzonej przez dodanie do nazwy analizowanego pliku sufiksu "_acc".

Rysunek 4.9: Okno z podsumowaniem

5. Implementacja i testy algorytmu

5.1. Opis implementacji

5.1.1. Alg 1

5.1.2. Alg 2

5.1.3. Alg 3

5.2. Napotkane problemy

5.3. Testy algorytmów na wybranych źródłach danych

6. Podsumowanie i wnioski końcowe

Załączniki

Literatura

[1] archive.ics.uci.edu/ml/datasets/adult. Dostęp 26.02.2023.

[2] www.kaggle.com/datasets/mattiuzc/stock-exchange-data. Dostęp 26.02.2023.

POLITECHNIKA RZESZOWSKA im. I. Łukasiewicza Rzeszów, 2023
Wydział Elektrotechniki i Informatyki

STRESZCZENIE PRACY DYPLOMOWEJ MAGISTERSKIEJ

IMPLEMENTACJA WYBRANYCH ALGORYTMÓW WYPEŁNIANIA BRAKUJĄCYCH WARTOŚCI, DLA STRUMIENI DUŻYCH ZBIORÓW DANYCH

Autor: Krzysztof Lang, nr albumu: EF-148853
Opiekun: dr Michał Piętal
Słowa kluczowe: bazy, danych, brakujące, wartości, wypełnianie

Dla poprawnej analizy danych ważna jest ich kompletność. Istnieje wiele sposobów radzenia sobie z brakującymi danymi. Najprostsze metody opierające się między innymi na średniej bądź najczęściej występującej wartości w wielu przypadkach mogą negatywnie wpłynąć na skuteczność analizy. Niniejsza praca ma na celu przeanalizować skuteczność uzupełniania brakujących danych z użyciem bardziej zaawansowanych metod opierających się na wykorzystaniu uczenia maszynowego. Te metody mają na celu wypełnić brakujące dane wartościami dużo bardziej zbliżonymi do rzeczywistych, minimalizując negatywny wpływ na skuteczność późniejszej analizy danych.

RZESZOW UNIVERSITY OF TECHNOLOGY Rzeszow, 2023
Faculty of Electrical and Computer Engineering

MSC THESIS ABSTRACT

IMPLEMETATION OF SELECTED MISSING VALUE FILLING ALGORITHMS FOR LARGE DATA SETS

Author: Krzysztof Lang, nr albumu: EF-148853
Supervisor: Michał Piętal, PhD
Key words: databases, missing, values, filling

For correct data analysis, completeness is important. There are many ways to deal with missing data. The simplest methods based on, among other things, the average or the most frequently occurring value in many cases can negatively affect the effectiveness of the analysis. This paper aims to analyze the effectiveness of filling in missing data using more advanced methods based on the use of machine learning. These methods are designed to fill in missing data with values much closer to the actual data, minimizing the negative impact on the effectiveness of subsequent data analysis.