Wydział Matematyki i Nauk Informacyjnych
Politechnika Warszawska

Składowanie danych w systemach Big Data

Analiza wpływu stanu rynku na pojazdy Polaków

Autorzy:
Maciej Pawlikowski,
Yevhenii Vinichenko,
Kacper Kurowski

Prowadzący:
Elżbieta Jowik

Warszawa, January 9, 2023

1 Wstęp

Ten dokument zawiera opis projektu, realizowanego w ramach przedmiotu Składowanie danych w systemach Big Data.

2 Cel projektu

Celem projektu jest stworzenie rozwiązania Big Data do przechowywania, przetwarzania i wizualizacji danych o pojazdach rejestrowanych w Polsce oraz o sytuacji gospodarczej w kraju. Umożliwiając zbadanie zależności pomiędzy dobrobytem Polaków a samochodami, jakie decydują się zakupić.

3 Potencjalne korzyści ze wdrożenia projektu

Nasz projekt będzie umożliwiać producentom i dealerom pojazdów zdefiniowanie wskaźników sygnalizujących zmianę popytu. Wskaźniki te będą pomagały nie tylko w określeniu całościowego popytu, ale też popytu na dany rodzaj samochodu — przykładowo, samochody elektryczne.

Pozwoli to lepiej podejmować decyzje dotyczące zwiększenia/ zmniejszenia produkcji lub dywersyfikacji produktów oferowanych na rynku polskim.

4 Opis zbiorów danych

W tej sekcji skrótowo opisujemy dwa zbiory danych, na podstawie których stworzymy projekt:

- API Centralna Ewidencja Pojazdów i Kierowców (CEPiK),
- Bank Danych Makroekonomicznych

4.1 API dla Centralnej ewidencji pojazdów i kierowców

Centralna Ewidencja Pojazdów i Kierowców (CEPiK) gromadzi dane o pojazdach zarejestrowanych w Polsce oraz o osobach uprawnionych do kierowania pojazdami. Dane są w rzeczywistości odświeżane co tydzień, ale zamierzamy symulować cogodzinne odświeżanie poprzez zamieszczenie nowych danych dziennych z godziny na godzinę. Dane zawierają ponad 60 parametrów w tym:

- marka,
- model,
- rok produkcji,
- data pierwszej rejestracji w kraju,
- poziom-emisji-co2,
- rodzaj-paliwa.

4.2 Bank Danych Makroekonomicznych

BDM jest statystyczną bazą gromadzącą wskaźniki charakteryzujące sytuację gospodarczą i społeczną Polski w skali makro. Baza umożliwia dostęp do długich szeregów czasowych, odświeżanych co miesiąc, dla podstawowych danych makroekonomicznych w różnych obszarach tematycznych. W naszej implementacji najbardziej jesteśmy zainteresowani danymi dotyczącymi stanu gospodarki takimi jak:

- Rynek pracy (bezrobocie)
- Warunki życia ludności (dochód realny, wydatki związane z energią i paliwami)
- Wskaźnik cen (towarów i usług konsumpcyjnych)

5 Planowany stos architektoniczny

Do zrealizowania projektu zamierzamy wykorzystać poniższe technologie.

- Apache HBase — przechowywanie danych związanych ze stanem gospodarki (chcemy mieć dostęp do poszczególnych rekordów)
- Apache Hive i Hadoop — przechowywanie danych o rejestracjach pojazdów (chcemy przetwarzać duże ilości danych nie koniecznie interpretować poszczególne rekordy). Dodatkowo w Hadoop chcemy przechowywać master Data Set z danymi BDM oraz tabele wynikowe używane przy raportowaniu.
- Apache NiFi — automatyzacja przepływu danych
- Apache PySpark — analiza i przetwarzanie danych

Które w naszej implementacji będą pełnić poniższe role:

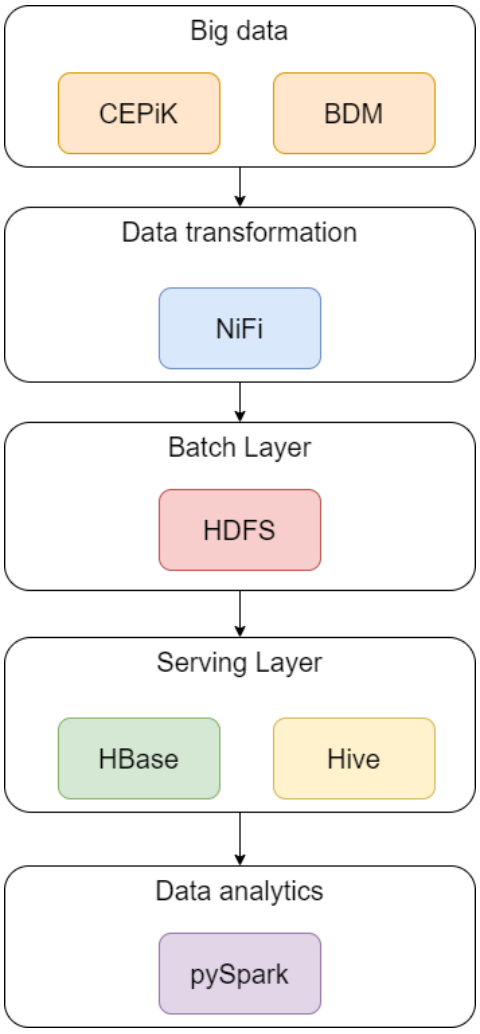


Figure 1: Role w implementacji

Schemat rozwiązania na Rysunku 2 określa, w którym momencie dany program zostanie użyty. Schemat naśladuje architekturę Lambda. Rolę Masterdataset pełnią pliki Parquet oraz Orc, przechowywane w HDFS (Batch layer), z kolei tabele w HBase i Hive odpowiadają widokom z Serving layer. Speed layer w tym przypadku nie jest implementowany, gdyż nie planujemy analizy w czasie rzeczywistym.

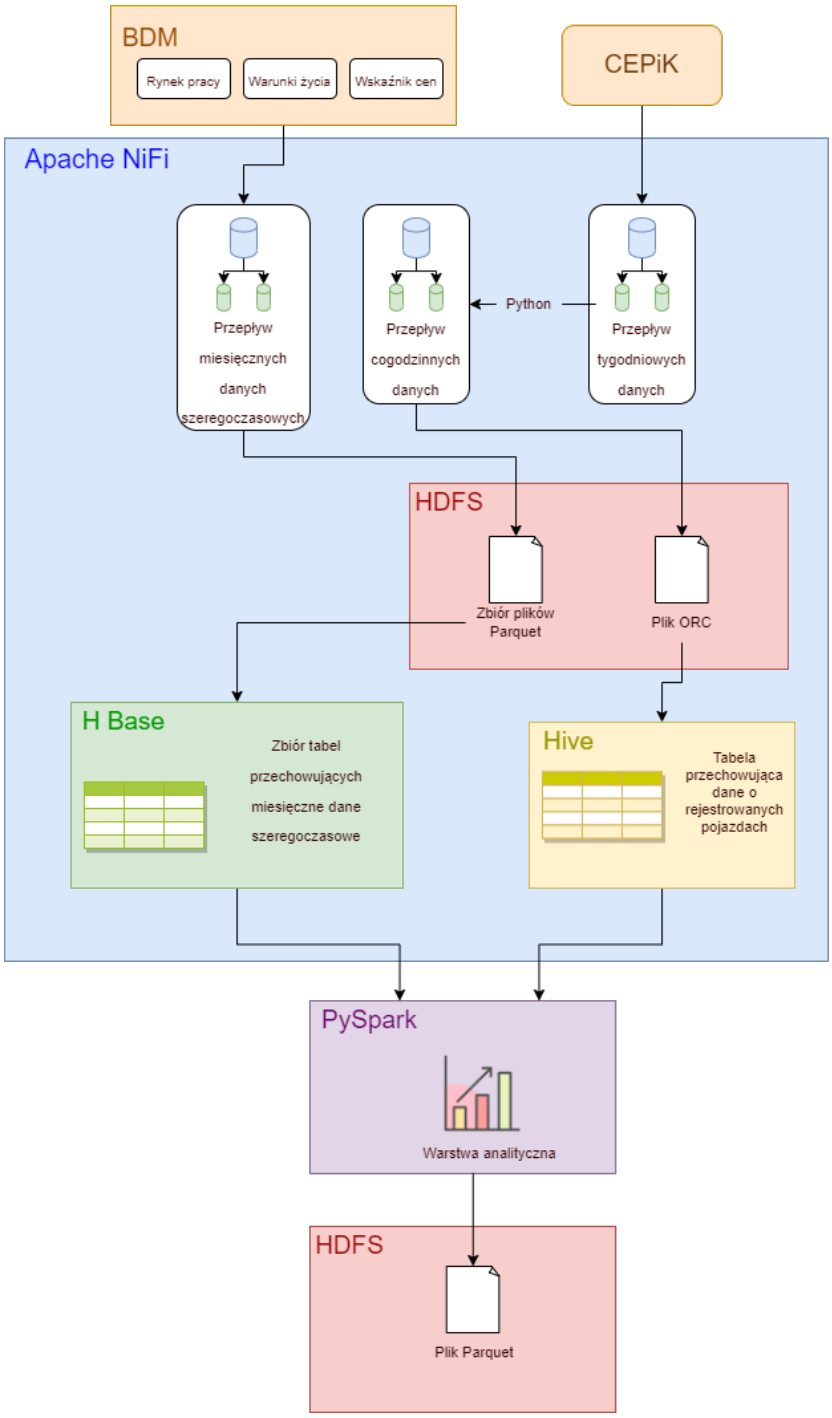


Figure 2: Schemat rozwiązania

5.1 Przepływ miesięcznych danych szeregoczasowych (BDM)

Źródłem danych makroekonomicznych jest zbiór plików .csv, z których każdy zawiera dane opisujące inną dziedzinę życia. W ramach przygotowania ich do późniejszej analizy z pomocą pythona usuwamy z nich polskie znaki i transponujemy je, a z pomocą NIFI zmieniamy ich format na Parquet, w celu lepszej optymalizacji wyszukań, a następnie zapisujemy je w hdfs, skąd przenosimy je do HBase'a.

5.2 Przepływ cogodzinnych danych (CEPiK)

Ponieważ API oferowane przez CEPiK, które pozwalało na pobieranie danych z danego dnia miało ograniczenia przepustowości, a niepełne rekordy podważyłyby prawdziwość naszej analizy(analizujemy między innymi ilość pojazdów zarejestrowanych danego dnia). Dlatego zdecydowaliśmy się skorzystać z API zwracającego plik .zip zawierający wszystkie rekordy dostępne w bazie danych. Samego pobrania pliku jak i jego rozpakowanie dokonujemy z pomocą NiFI, za pomocą którego uruchamiamy również skrypt pythonowy. W skrypcie tym filtrujemy ten plik(by analizować tylko dane z 2019 roku), usuwamy polskie znaki, naprawiamy błędne rekordy oraz tworzymy pomniejsze pliki .csv(każdy zawierający dane z danej doby). Następnie pliki te umieszczamy lokalnie do folderu generated_data skąd NiFi przenosi je później do Batch Layer w HDFS i Serving Layer w Hive.

6 Testy systemu

Testowane zachowanie	Kroki	Oczekiwany wynik
Pobieranie danych z CEPiK	1.Uruchomienie procesorów w NIFI	W folderze /cpik pojawia się plik .csv.
Dzielenie pliku test_full.csv na pomniejsze pliki inny dla każdego dnia.	1.Umieszczamy plik test_full.csv w folderze /home/vagrant.	W folderze /generated_data znajdujemy 365 plików .csv z różnymi datami w nazwie (wszystkie z 2019 roku) oraz każda zawiera te same kolumny co w danych źródłowych.
	2. Uruchamiamy skrypt geenrate_test_data.py.	
	3. Analizujemy zawartość foldera.	
Wstawianie danych do Hive	1.Wstaw plik test1.csv do folderu /generated_data.	W tabeli hive znajduje się 1427 rekordów.
	3. Uruchom skrypt testtable.hql.	
	3. Uruchom procesory w NiFI.	
	4. Sprawdź zawartość tabeli testtable	
Wstawianie nowych danych do istniejącej tabeli Hive	1.Wstaw dwa plik test2.csv i test3.csv do folderu /generated_data.	W tabeli hive znajduje się 4593 rekordów.
	3. Uruchom procesory w NiFI.	
	4. Sprawdź zawartość tabeli testtable	
Symulowanie przepływu danych	1.Zamieszczamy pliki .csv w folderze	Po godzinie 6 plików zostało przeniesionych.
	2. Uruchamiamy procesory w NIFI, w których odstęp między przesłaniami jest zmieniony na 10 min	
	3. Odczekujemy 1 godzinę	
	4. Sprawdzamy zawartość folderu.	
Input validation	1.Start Application.py	GUI doesn't allow to input incorrect values. If value to small or to big is inserted system shouldn't allow to run the visualisation/ start optimization process.
	2. Try changing paramaters to incorect ones (In all modes)	
	3. Run application	

Testowane zachowanie	Kroki	Oczekiwany wynik
Wstawianie danych do HDFS	1. Wczytanie pliku .CSV po początkowym spreparowaniu	W systemie HDFS, w folderze /user/projekt/bdm pojawiają się pliki w formacie .PARQUET
	2.Uruchom procesy w NIFI	
	3.Sprawdź zawartość odpowiedniego folderu w HDFS	
Przeniesienie plików z HDFS do HBASE	1. Uruchom procesy w NIFI	W otrzymywanej tabeli mamy 265 rekordów
	2.Sprawdź zawartość tabeli.	

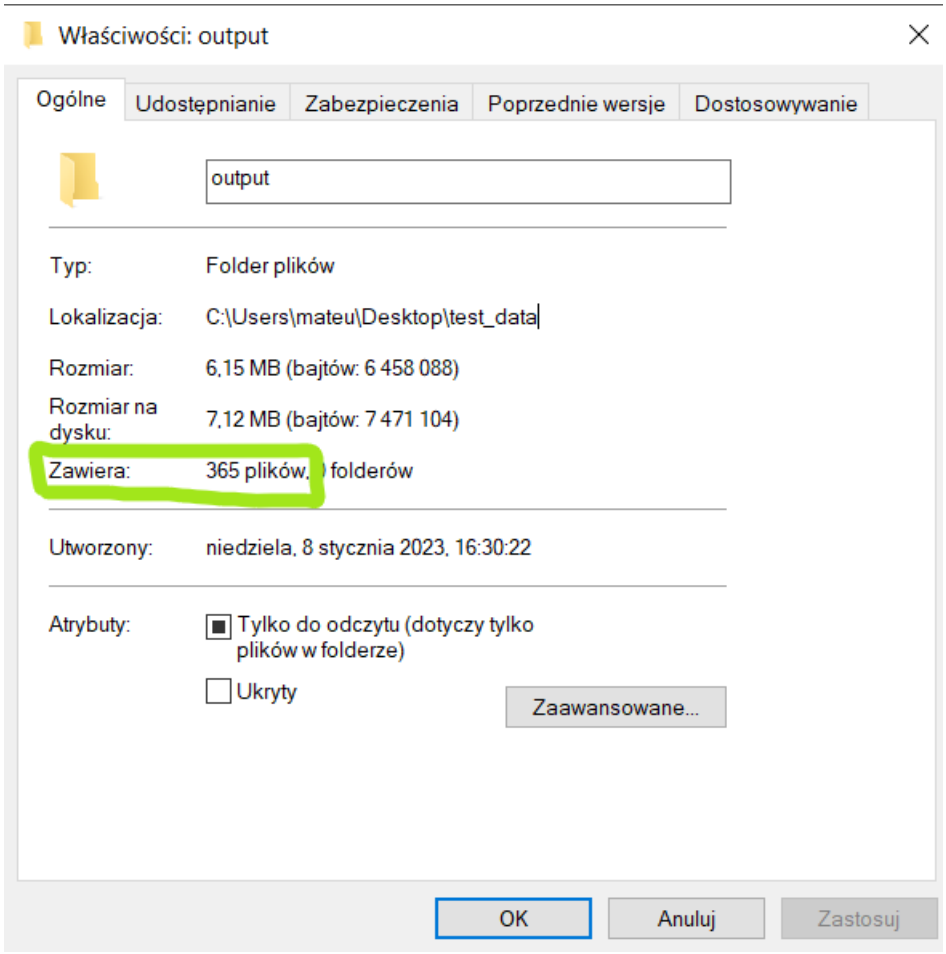
6.1 Wyniki testów

1. Pobieranie danych z CEPiK

/home/vagrant/cpik/

Nazwa	Rozmiar	Zmodyfikowany	Prawa	Właściciel
..		08.01.2023 16:55:52	rwxr-xr-x	vagrant
pojazdy_14_2022-04-1...	1 802 252...	08.01.2023 16:58:01	rw-r--r--	root

2. Dzielenie pliku .csv na pomniejsze pliki inny dla każdego dnia.



pojazdy_2019-01-01	08.01.2023 16:32	Plik wartości oddzi...	2 KB
pojazdy_2019-01-02	08.01.2023 16:32	Plik wartości oddzi...	1 034 KB
pojazdy_2019-01-03	08.01.2023 16:32	Plik wartości oddzi...	1 123 KB
pojazdy_2019-01-04	08.01.2023 16:32	Plik wartości oddzi...	587 KB
pojazdy_2019-01-05	08.01.2023 16:32	Plik wartości oddzi...	2 KB
pojazdy_2019-01-06	08.01.2023 16:32	Plik wartości oddzi...	2 KB
pojazdy_2019-01-07	08.01.2023 16:32	Plik wartości oddzi...	580 KB
pojazdy_2019-01-08	08.01.2023 16:32	Plik wartości oddzi...	516 KB

3. Wstawianie danych do Hive

```
hive> select count(*) from testtable;
```

```
Total MapReduce CPU Time Spent: 0 msec
OK
1427
```

4. Wstawianie nowych danych do istniejącej tabeli Hive

```
hive> select count(*) from testtable;
```

```
Total MapReduce CPU Time Spent: 0 msec
OK
4593
```

5. Przenoszenie pliku .CSV do .PARQUET w HDFS

```
vagrant@node1:~/cpik$ hdfs dfs -ls /user/projekt/bdm_testy
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hadoop-2.7.6/share/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/apache-tez-0.9.1-bin/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
Found 1 items
-rw-r--r--   1 root supergroup      16612 2023-01-08 16:31 /user/projekt/bdm_testy/Rynek_pracy.parquet
```

6. Przenoszenie .PARQUET do HBASE

```
hbase(main):001:0> count 'BDM_test'
275 row(s)
Took 1.6010 seconds
=> 275
```

```
hbase(main):003:0> describe 'BDM_test'
Table BDM_test is ENABLED
BDM_test
COLUMN FAMILIES DESCRIPTION
{NAME => 'rynek_pracy', BLOOMFILTER => 'NONE', IN_MEMORY => 'false', VERSIONS => '3', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_EN
CODING => 'NONE', COMPRESSION => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0', BLOCKCACHE => 'false', BLOCKSIZE => '65536', REPLI
CATION_SCOPE => '0'}

1 row(s)
```

7 Raport

W celu zbadania zależności pomiędzy dobrobytem Polaków, a zakupionymi przez nich samochodami stworzyliśmy raport opierający się o zestaw najciekawszych wizualizacji, które pozwoliły nam wyciągnąć następujące wnioski:

- Zamożność obywateli nie wpływa bezpośrednio na rodzaj paliwa, którym napędzane są pojazdy. Jeżeli korelacja istnieje to jest ona przesunięta o dłuższy przedział czasowy niż pozwala nam nasz zbiór danych.

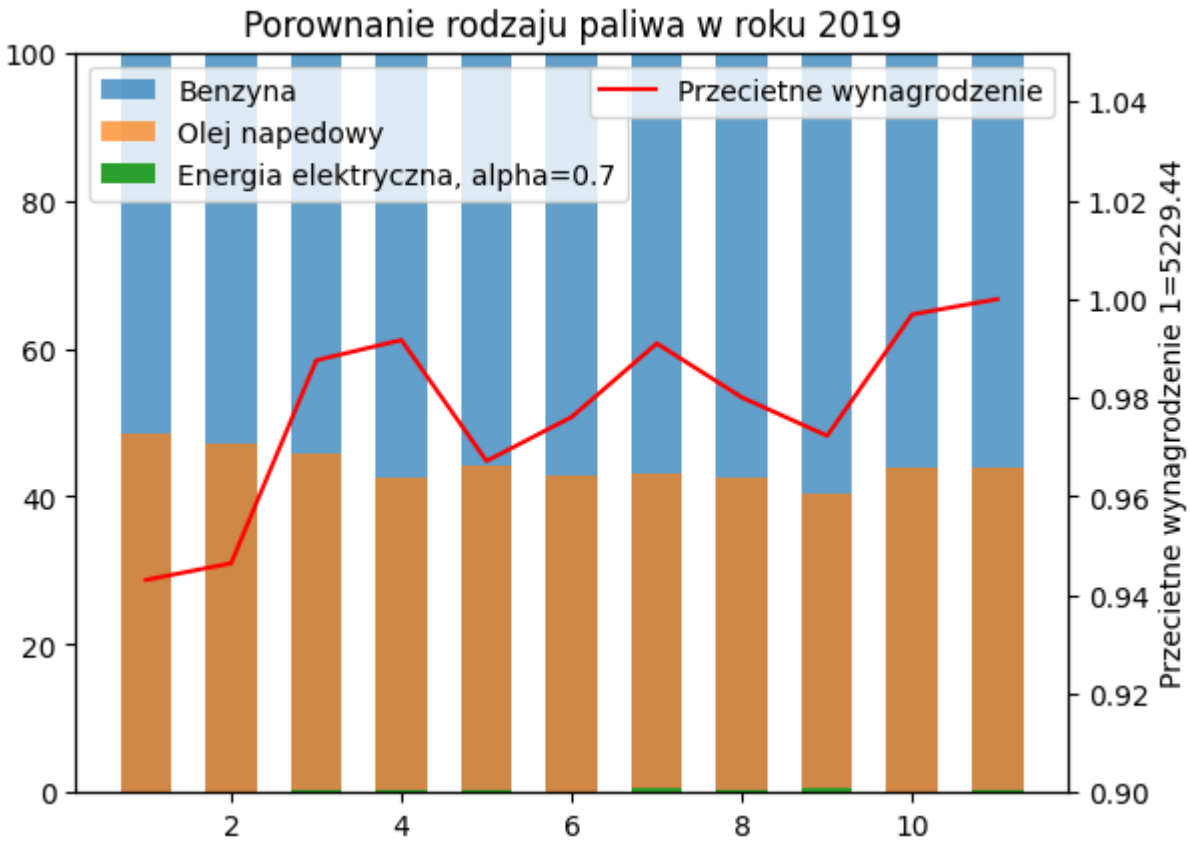


Figure 3: wynagrodzenie, a rodzaj paliwa

- Zarówno ilość samochodów kupowanych, jak i rozkład ich marek nie są zależne od wartości towarów importowanych do kraju. Co ciekawe rozkład marek wybieranych przez polaków wydaje się być bardzo stabilny co jest wykazywane przez wysoką korelację pomiędzy ilościami zakupionych samochodów konkretnych marek.

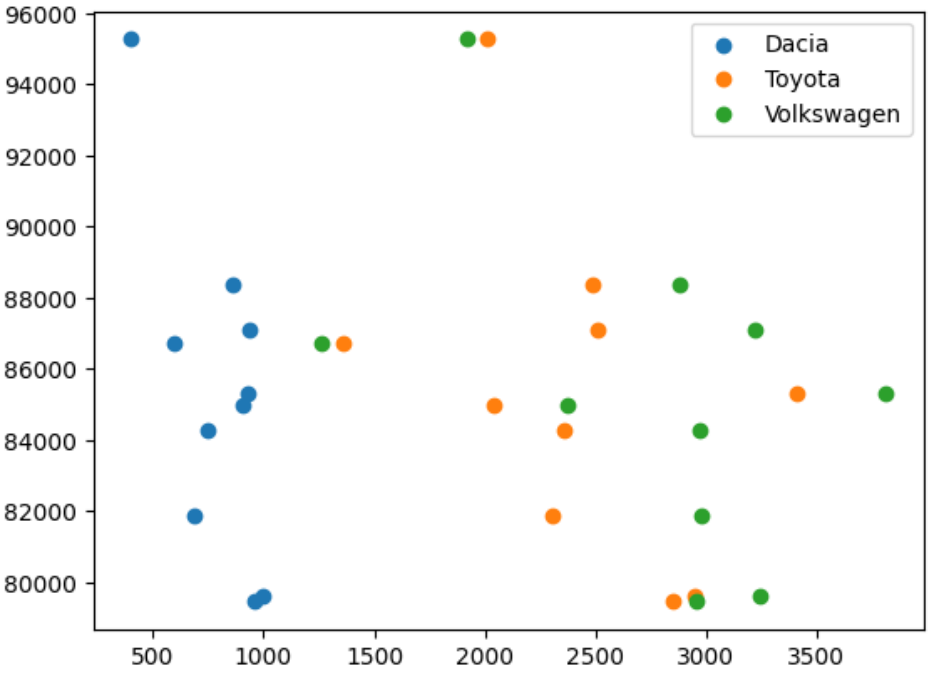


Figure 4: Marki samochodów, a import towarów

]

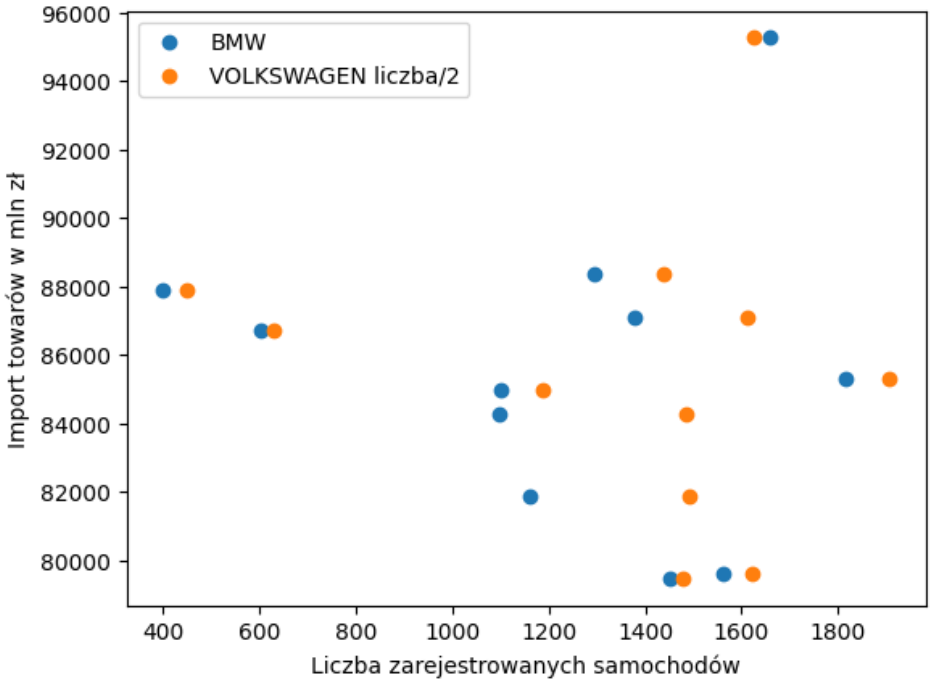


Figure 5: Podobieństwo Volkswagena i BMW

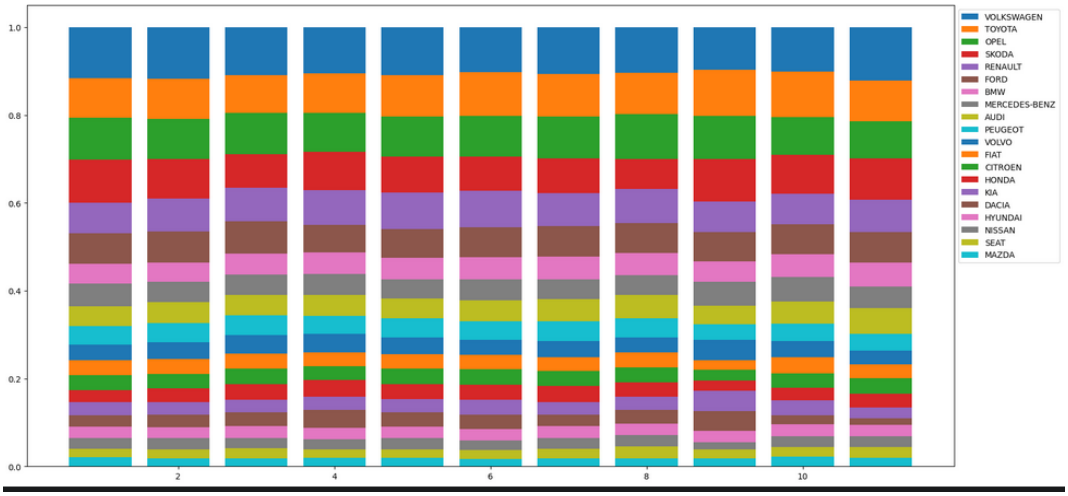


Figure 6: Rozkład marek

Dla pięciu najpopularniejszych marek pojazdów:

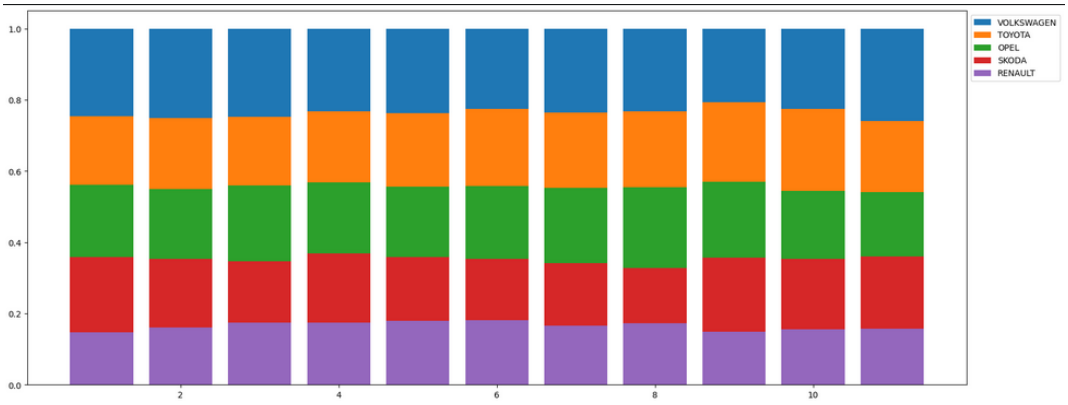


Figure 7: Najpopularniejsze marki

Podsumowując klucz, według którego Polacy kupują samochody (kiedy to robią? jakiej marki pojazd wybierają? ...) nie znajduje się w danych makroekonomicznych i najlepiej byłoby szukać go gdzieś indziej. Natomiast dane, którymi dysponujemy są ciekawe same w sobie. Co podczas poszukiwania zależności między bazami danych zaowocowało paroma ciekawymi znaleziskami:

- Samochody używane przypominają trendy kupowania pojazdów przez młodzież ze wzrostami w wakacje oraz w październiku na początek roku akademickiego.

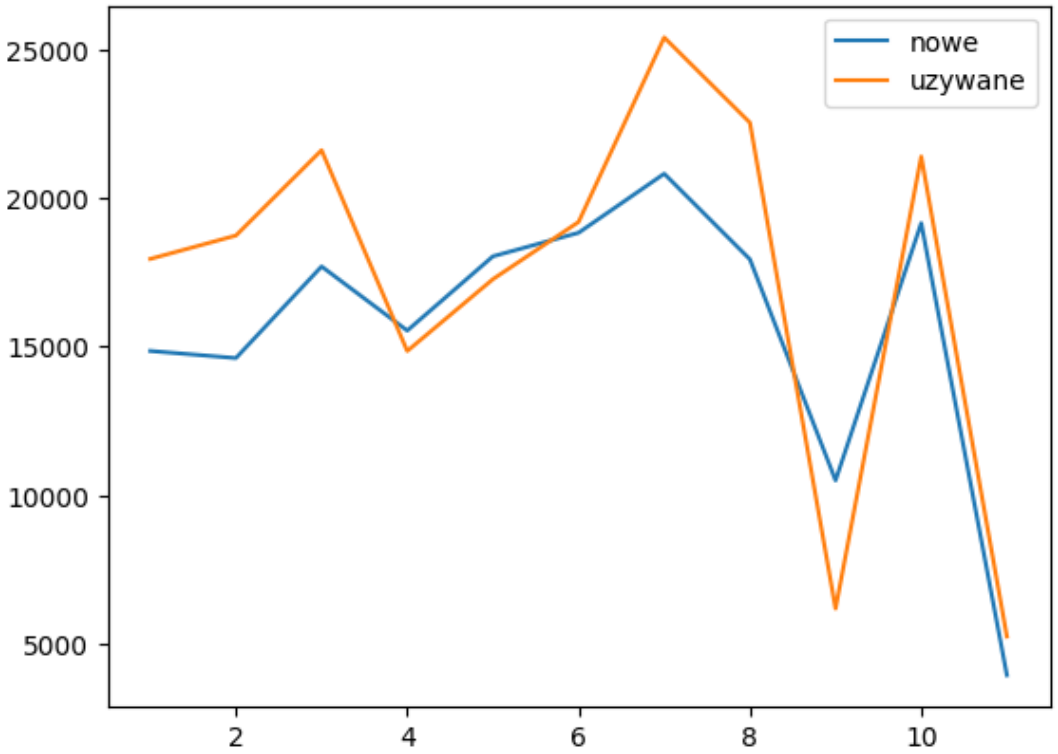


Figure 8: Rozkład nowych i używanych samochodów

8 Podział pracy

Podział pracy przedstawia Tabela 1. Wartości procentowe określają, jaka część danego fragmentu projektu została wykonana przez danego członka zespołu.

Dział projektu	Maciej Pawlikowski	Yevhenii Vinichenko	Kacper Kurowski
Obsługa danych z BDM	0%	0%	100%
Obsługa danych z CEPiK	50%	50%	0%
Symulacja przepływu danych	0%	100%	0%
HBase(serving layer BDM)	0%	0%	100%
Hive(serving layer CEPiK)	100%	0%	0%
Wizualizacje w PySpark	30%	10%	60%
Przetwarzanie danych w NiFi	50%	25%	25%
Pisanie raportu	80%	10%	10%
Robienie prezentacji	20%	60%	20%

Table 1: Tabela podziału pracy w poszczególnych częściach projektu

9 Link do Github'a z kodem

https://github.com/pawlik0wskim/BigData