

Hurtownie i Eksploracja Danych	
Temat	Sieć restauracji
Przygotowali	Piotr Kaczmarczyk, Przemysław Postrach

1. Opis problemu

W ramach projektu opracowaliśmy hurtownie danych wspomagającą prace sieci restauracji “Vege cziken w majo”. Dane pochodzą z systemu transakcji synchronizującego prace całej sieci. Zakupy mogą być dokonywane osobiście w restauracji, w specjalnych kioskach oraz online w aplikacji. Płatności są przyjmowane przez wszystkie dostępne formy min. Gotówkowo, PAYU czy kartą płatniczą. Dzięki dacie wygenerowania paragonu przeprowadzana jest synchronizacja w i kolejność dodawania rekordów w bazie.

W systemie rejestrowane są również dostawy. Aplikacja wysyła informację o miejscu dostawy oraz o pracowniku odpowiedzialnym za przygotowanie i skompletowanie zamówienia. W sieci restauracji produkty są pogrupowane w kategorie i zorganizowane w różne menu. Każda restauracja posiada takie same produkty, kategorie oraz menu.

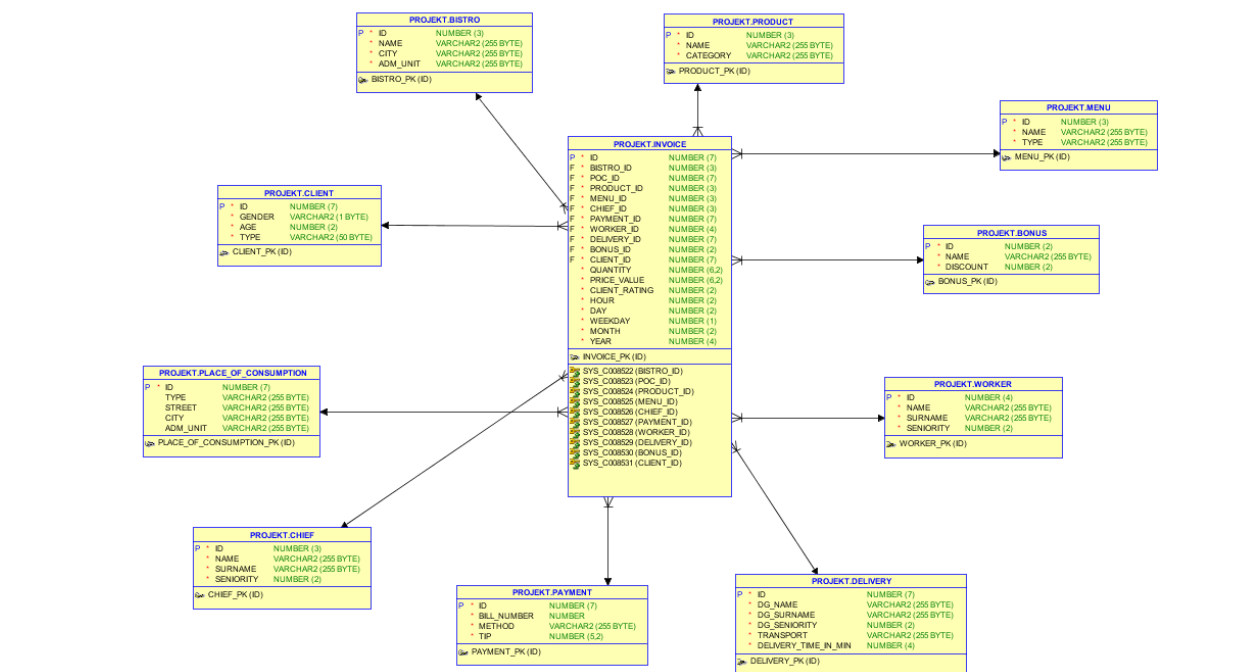
Ponadto sieć oferuje różne bonusy, które można wykorzystać przy zakupach. Dostępne dla wszystkich klientów, nie tylko posiadaczy kart stałego klienta. Z pomocą sztucznej inteligencji system może codziennie generować promocje dla klientów oraz przewidywać jakie promocje będą się najchętniej sprzedawały. Dzięki temu możemy generować popyt i podaż w zależności od stanów magazynowych czy okresów gorszej sprzedaży.

Każdy klient może ocenić każdy posiłek w skali od 1 do 10 przy kasie. Dzięki temu możemy przechowywać informacje o tym, co, kiedy i gdzie klientom podobało się bardziej lub mniej. Dane te są podstawą do analiz danych i predykcji poprzez sieci neuronowe. Najbardziej podstawowym elementem analizy jest sprawdzenie czy dana transakcja ucieszyła klienta. Pozwala to nam przewidzieć co należy poprawić i udoskonalić w naszych restauracjach.

System zarządzania siecią działa w trybie online i na bieżąco dostarcza dane do magazynu. W ramach projektu zaprojektowaliśmy go i symulowaliśmy dane wprowadzane do niego przez system na przestrzeni kilku miesięcy. Hurtownia posiada dane nie tylko o transakcji zakupionych produktach, ale też o pracownikach, klientach, dostępnym menu oraz kiedy transakcja została przeprowadzona.

2. Hurtownia danych

Schemat bazy danych:



Hurtownia danych realizuje klasyczny schemat gwiazdy. Wszystkie tabele posiadają połączenie jeden do wielu z tabelą "Invoice".

Miary:

1. Quantity – waga zakupionego posiłku
2. Price_value – cena
3. Client_rating – ocena klienta
4. Data

Zostały wybrane ze względu na optymalizację zapytań oraz możliwości późniejszej analizy data minerem.

Wymiary:

1. Client – informacje z karty stałego klienta
2. Place_of_consumption – informacje o miejscu konsumpcji
3. Bistro – informacje o restauracji
4. Menu – posiada nazwę i typ
5. Bonus – informacje o zniżce
6. Produkt – informacje o produkcie i jego kategorii
7. Worker – informacje o pracowniku
8. Chief – informacje o szefie kuchni
9. Payment – informacje o płatności
10. Delivery – informacje o dostawie

Wymiary pomagają nam zebrać bardziej szczegółowe informacje o restauracji, klientach oraz posiłkach. Dzięki temu profilowanie klientów będzie sprawniejsze. Będziemy mogli sprawdzić jakie oni mają upodobania oraz co najczęściej się sprzedaje.

3. Instalacja

Instalacja oraz konfiguracja bazy danych przebiegła w skrócony sposób. Wykorzystaliśmy obraz maszyny wirtualnej opartej na WIN 10 z zainstalowanym Oracle sql server oraz sql developer. Została ona nam udostępniona przez prowadzącego.

Należało jedynie nadać uprawnienia dla użytkownika "projekt" aby można było bez przeszkód wykonywać zadania na bazie danych. Zostało to zrobione poprzez egzekucję komendy:

```
GRANT ALL PRIVILEGES TO PROJEKT;
```

Przygotowany obraz posiadał dostępny moduł "Data miner" należało go jedynie pobrać przy pierwszym uruchomieniu bazy. Często można spotkać się z błędami przy jego instalacji, dlatego na postawie miejsca oraz nazwy wykonywanego SQLa należy w odpowiednim miejscu dodać poniższą linijkę:

```
alter session set "_oracle_script" = true;
```

4. Zasilenie hurtowni danymi

Zasilenie hurtowni danymi odbyło się poprzez skrypt zawarty w pliku "createAndLoad.bat".

```
1 | sqlplus %1 @buildDB.sql
2 | cd ctl
3 | for %%f in (*.ctl) do (
4 |     sqlldr CONTROL=%%f log=../logs/%%~nf.log bad=../bads/%%~nf.log skip=1 userid=%1
5 | )
6 | cd..
```

Skrypt odpalamy w konsoli komendom:

```
CreateAndLoad.bat *użytkownik*/*hasło*
```

W pierwszej linii zostaje uruchomiony sql plus i wykonany kod SQL usuwający bazę oraz jej zależności i tworzący tabele na nowo.

Następnie zostają wykonane skrypty ładujące dane do bazy danych. Ze względu na więzy integralności tabela "Invoice" zostaje załadowana na samym końcu. W tym celu nazwa zmieniona została na "ww_Invoice.ctl".

5. Zapytania SQL

Ustawienia bazy danych spowodowało brak obsługi polskich znaków.

Cube:

1. Zapytanie pokazuje która restauracja jest najbardziej rentowna

```
1  --cube
2  SELECT
3      nvl(bis.name, 'ALL') bistro_name,
4      COUNT(inv.id)         invoice_count,
5      SUM(inv.price_value) takings
6  FROM
7      invoice inv
8  JOIN bistro bis ON bis.id = inv.bistro_id
9  GROUP BY
10     CUBE(bis.name)
11  ORDER BY
12     bis.name;
```

	BISTRO_NAME	INVOICE_COUNT	TAKINGS
1	Bistro Bułeczka	30001	15016841
2	Bistro Masełko	29762	14834316
3	Burger & Co	29833	14938826
4	Gruszka	30211	15168908
5	Jabłko	30085	15018433
6	Krówka	30065	14989186
7	Melon i Spółka	29912	15021740
8	Naleśniczek	30126	15063759
9	Smaczne Bistro	30003	15070805
10	Tanio i smacznie	30002	15013368
11	ALL	300000	150136182

2. Zapytanie pokazuje minimalny, maksymalny wiek klienta oraz jaką ocenę wystawił.

```
14  SELECT
15      nvl(bis.name, 'OVERALL RATING') bistro_name,
16      MIN(cli.age) min_age,
17      MAX(cli.age) max_age,
18      ROUND(AVG(cli.age),1) avg_age,
19      ROUND(AVG(INV.client_rating),6) avg_rate
20  FROM
21      invoice inv
22  JOIN bistro bis ON bis.id = inv.bistro_id
23  JOIN client cli on cli.id = inv.client_id
24  GROUP BY
25     CUBE(bis.name)
26  ORDER BY
27     bis.name;
```

	BISTRO_NAME	MIN_AGE	MAX_AGE	AVG_AGE	AVG_RATE
1	Bistro Bułeczka	0	80	40,3	5,502917
2	Bistro Masełko	0	80	39,8	5,528762
3	Burger & Co	0	80	40,1	5,477827
4	Gruszka	0	80	40,1	5,508623
5	Jabłko	0	80	39,9	5,529832
6	Krówka	0	80	40,4	5,530684
7	Melon i Spółka	0	80	39,9	5,514643
8	Naleśniczek	0	80	39,9	5,52068
9	Smaczne Bistro	0	80	40,2	5,485051
10	Tanio i smacznie	0	80	40,1	5,483768
11	OVERALL RATING	0	80	40,1	5,508293

3. Zapytanie pokazuje jakie dostawy zostały wykonane dla danej restauracji.

```

29 SELECT
30     nvl(bis.name, 'OVERALL') bistro_name,
31     nvl(poc.type, 'OVERALL TYPE') TYPE,
32     count(poc.type)
33 FROM
34     invoice inv
35 JOIN bistro bis ON bis.id = inv.bistro_id
36 JOIN place_of_consumption poc ON poc.id = inv.poc_id
37 GROUP BY
38     CUBE(bis.name, poc.type)
39 ORDER BY
40     bis.name, poc.type
41 ;

```

	BISTRO_NAME	TYPE	COUNT(POC.TYPE)
1	Bistro Bułeczka	Dostawa	10021
2	Bistro Bułeczka	Na_miejscu	9819
3	Bistro Bułeczka	Na_wynos	10161
4	Bistro Bułeczka	OVERALL TYPE	30001
5	Bistro Masełko	Dostawa	9781
6	Bistro Masełko	Na_miejscu	9933
7	Bistro Masełko	Na_wynos	10048
8	Bistro Masełko	OVERALL TYPE	29762
9	Burger & Co	Dostawa	9965
10	Burger & Co	Na_miejscu	9869
11	Burger & Co	Na_wynos	9999
12	Burger & Co	OVERALL TYPE	29833
13	Gruszka	Dostawa	9998
14	Gruszka	Na_miejscu	9945
15	Gruszka	Na_wynos	10268
16	Gruszka	OVERALL TYPE	30211
17	Jabłko	Dostawa	9970

Grouping sets:

1. Zapytanie pokazuje kucharza, wykonany posiłek, ocenę klienta oraz ilość przygotowanych posiłków.

```

2 SELECT
3     NVL(TO_CHAR(CHI.NAME), 'ALL') AS "CHEF NAME",
4     NVL(TO_CHAR(CHI.SURNAME), 'ALL') AS "CHEF SURNAME",
5     NVL(TO_CHAR(MN.NAME), 'ALL') AS MENU,
6     CLIENT_RATING AS "CLIENT RATING",
7     "SALES COUNT"
8 FROM
9     (
10        SELECT
11            CHIEF_ID,
12            MENU_ID,
13            NVL(TO_CHAR(INV.CLIENT_RATING), 'ALL') AS CLIENT_RATING,
14            COUNT(*) AS "SALES COUNT"
15        FROM
16            INVOICE INV
17        GROUP BY
18            GROUPING SETS (
19                (INV.CHIEF_ID),
20                (INV.CHIEF_ID, INV.CLIENT_RATING),
21                (INV.CHIEF_ID, INV.MENU_ID),
22                (INV.CLIENT_RATING)
23            )
24        ) MAIN
25 LEFT OUTER JOIN MENU MN ON MAIN.MENU_ID = MN.ID
26 LEFT OUTER JOIN CHIEF CHI ON CHI.ID = MAIN.CHIEF_ID;
27

```

	⚡ CHEF NAME	⚡ CHEF SURNAME	⚡ MENU	⚡ CLIENT RATING	⚡ SALES COUNT
1	Adrianna	Koczara	Kids	ALL	791
2	Adrianna	Koczara	Normalne	ALL	779
3	Adrianna	Koczara	Premium	ALL	721
4	Adrianna	Koczara	Vegan	ALL	730
5	Adrianna	Koczara	ALL	ALL	3021
6	Miś,osz	Radziewicz	Kids	ALL	804
7	Miś,osz	Radziewicz	Normalne	ALL	744
8	Miś,osz	Radziewicz	Premium	ALL	717
9	Miś,osz	Radziewicz	Vegan	ALL	726
10	Miś,osz	Radziewicz	ALL	ALL	2991
11	Dagmara	Kaczka	Kids	ALL	771
12	Dagmara	Kaczka	Normalne	ALL	745
13	Dagmara	Kaczka	Premium	ALL	736
14	Dagmara	Kaczka	Vegan	ALL	726
15	Dagmara	Kaczka	ALL	ALL	2978
16	Dawid	Bogaczyk	Kids	ALL	778
17	Dawid	Bogaczyk	Normalne	ALL	752

2. Zapytanie pokazuje sprzedaż z podziałem na dni tygodnia i godziny.

```
SELECT
    NVL(TO_CHAR(BIS.NAME), 'ALL') AS BISTRO,
    HOUR,
    WEEKDAY,
    SALES
FROM
    (
        SELECT
            BISTRO_ID,
            NVL(TO_CHAR(INV.HOUR), 'ALL') AS "HOUR",
            NVL(TO_CHAR(INV.WEEKDAY), 'ALL') AS "WEEKDAY",
            count(*) sales
        FROM
            INVOICE INV
        GROUP BY
            GROUPING SETS (
                (INV.HOUR),
                (INV.HOUR, INV.WEEKDAY),
                (INV.BISTRO_ID, INV.HOUR, INV.WEEKDAY),
                (INV.HOUR, INV.WEEKDAY)
            )
        ORDER BY
            INV.BISTRO_ID ASC,
            INV.HOUR ASC,
            INV.WEEKDAY ASC
    ) MAIN
LEFT OUTER JOIN BISTRO BIS ON BIS.ID = MAIN.BISTRO_ID;
```

	BISTRO	HOUR	WEEKDAY	SALES
1	Naleśniczek 0	0	1	206
2	Naleśniczek 0	0	2	189
3	Naleśniczek 0	0	3	189
4	Naleśniczek 0	0	4	174
5	Naleśniczek 0	0	5	191
6	Naleśniczek 0	0	6	177
7	Naleśniczek 0	0	7	189
8	Naleśniczek 1	1	1	188
9	Naleśniczek 1	1	2	184
10	Naleśniczek 1	1	3	190
11	Naleśniczek 1	1	4	138
12	Naleśniczek 1	1	5	161
13	Naleśniczek 1	1	6	195
14	Naleśniczek 1	1	7	162
15	Naleśniczek 2	2	1	176
16	Naleśniczek 2	2	2	162
17	Naleśniczek 2	2	3	188

3. Zapytanie pokazuje jaki szef przygotował jaką ilość jedzenia, dodatkowo pokazana została średnia ocena klienta.

```

SELECT
  NVL(TO_CHAR(BIS.NAME), 'ALL') AS BISTRO,
  NVL(TO_CHAR(CHI.NAME), 'ALL') AS "CHEF NAME",
  NVL(TO_CHAR(CHI.SURNAME), 'ALL') AS "CHEF SURNAME",
  CLIENT_RATING AS "CLIENT RATING",
  AVERAGE_QTY AS "AVERAGE QUANTITY"
FROM
  (
    SELECT
      BISTRO_ID,
      CHIEF_ID,
      NVL(TO_CHAR(INV.CLIENT_RATING), 'ALL') AS CLIENT_RATING,
      ROUND(AVG(INV.QUANTITY), 2) AS AVERAGE_QTY
    FROM
      INVOICE INV
    GROUP BY
      GROUPING SETS (
        (INV.BISTRO_ID),
        (INV.CHIEF_ID),
        (INV.CLIENT_RATING),
        (INV.BISTRO_ID, INV.CLIENT_RATING)
      )
    ORDER BY
      INV.BISTRO_ID ASC,
      INV.CHIEF_ID ASC,
      INV.CLIENT_RATING ASC
  ) MAIN
LEFT OUTER JOIN BISTRO BIS ON BIS.ID = MAIN.BISTRO_ID
LEFT OUTER JOIN CHIEF CHI ON CHI.ID = MAIN.CHIEF_ID;

```

	BISTRO	CHEF NAME	CHEF SURNAME	CLIENT RATING	AVERAGE QUANTITY
1	Naleśniczek	ALL	ALL	1	2,96
2	Naleśniczek	ALL	ALL	2	2,98
3	Naleśniczek	ALL	ALL	3	3,01
4	Naleśniczek	ALL	ALL	4	2,99
5	Naleśniczek	ALL	ALL	5	2,98
6	Naleśniczek	ALL	ALL	6	3
7	Naleśniczek	ALL	ALL	7	3,01
8	Naleśniczek	ALL	ALL	8	3
9	Naleśniczek	ALL	ALL	9	3,02
10	Naleśniczek	ALL	ALL	10	3,01
11	Naleśniczek	ALL	ALL	ALL	3
12	Bistro Masełko	ALL	ALL	1	3,03
13	Bistro Masełko	ALL	ALL	2	2,96
14	Bistro Masełko	ALL	ALL	3	2,99
15	Bistro Masełko	ALL	ALL	4	2,98
16	Bistro Masełko	ALL	ALL	5	3,01
17	Bistro Masełko	ALL	ALL	6	3,01

Partition:

1. Zapytanie pokazuje % utargu danej restauracji.

```
SELECT
  BIS.NAME AS BISTRO,
  TAKINGS AS "TAKINGS  % IN YEAR"
FROM
  (
    SELECT
      DISTINCT INV.BISTRO_ID,
      INV.YEAR,
      ROUND (
        (
          100 * sum(PRICE_VALUE) OVER (partition BY INV.BISTRO_ID, INV.YEAR) / SUM(INV.PRICE_VALUE) OVER (partition by
          INV.YEAR)
        ),
        2
      ) TAKINGS
    FROM
      INVOICE INV
    ORDER BY
      INV.YEAR ASC
  ) MAIN
JOIN BISTRO BIS ON BIS.ID = MAIN.BISTRO_ID;
```

	BISTRO	TAKINGS % IN YEAR
1	Naleśniczek	10,39
2	Naleśniczek	9,94
3	Naleśniczek	9,77
4	Naleśniczek	9,89
5	Naleśniczek	9,9
6	Naleśniczek	10,24
7	Naleśniczek	9,88
8	Naleśniczek	9,92
9	Naleśniczek	9,77
10	Naleśniczek	10,31
11	Naleśniczek	9,66
12	Naleśniczek	10,05
13	Naleśniczek	9,41
14	Naleśniczek	9,95
15	Naleśniczek	10,79
16	Naleśniczek	9,8
17	Naleśniczek	10,06

2. Zapytanie pokazuje najczęściej kupowany produkt w menu

```
SELECT
  PROD.NAME AS PRODUCT,
  MN.NAME AS MENU,
  PIM AS "% PRODUCT IN MENU"
FROM
  (
    SELECT
      DISTINCT INV.PRODUCT_ID,
      INV.MENU_ID,
      ROUND(
        (
          100 * COUNT(INV.PRODUCT_ID) OVER (partition BY INV.PRODUCT_ID, INV.MENU_ID) / COUNT(INV.PRODUCT_ID) OVER (
            partition by INV.MENU_ID
          ),
          2
        ) as PIM
      FROM
        INVOICE INV
      ORDER BY
        INV.MENU_ID
    ) MAIN
  JOIN PRODUCT PROD ON PROD.ID = MAIN.PRODUCT_ID
  JOIN MENU MN ON MN.ID = MAIN.MENU_ID;
```

	PRODUCT	MENU	% PRODUCT IN MENU
1	Tea	Kids	4,53
2	Tea	Normalne	4,57
3	Tea	Premium	4,59
4	Tea	Vegan	4,56
5	Water	Kids	4,39
6	Water	Normalne	4,58
7	Water	Premium	4,49
8	Water	Vegan	4,49
9	Mirinda	Kids	4,59
10	Mirinda	Normalne	4,54
11	Mirinda	Premium	4,48
12	Mirinda	Vegan	4,5
13	Coffie	Kids	4,66
14	Coffie	Normalne	4,51
15	Coffie	Premium	4,65
16	Coffie	Vegan	4,58
17	Grass salad	Kids	4,54

3. Zapytanie pokazuje rabaty w danym bistro.

```
SELECT
  BIS.NAME AS BISTRO,
  BN.NAME AS BONUS,
  BIB AS "% BONUS IN BISTRO"
FROM
  (
    SELECT
      DISTINCT INV.BONUS_ID,
      INV.BISTRO_ID,
      ROUND(
        (
          100 * COUNT(INV.BONUS_ID) OVER (partition BY INV.BONUS_ID, INV.BISTRO_ID) / COUNT(INV.BONUS_ID) OVER (
            partition by INV.BISTRO_ID
          ),
          2
        ) as BIB
      FROM
        INVOICE INV
      ORDER BY
        INV.BISTRO_ID
    ) MAIN
  JOIN BISTRO BIS ON MAIN.BISTRO_ID = BIS.ID
  JOIN BONUS BN ON BN.ID = MAIN.BONUS_ID;
```

BISTRO	BONUS	% BONUS IN BISTRO
1 Jabłko	Brak	20,3
2 Bistro Masełko	Brak	20,21
3 Tanio i smacznie	Brak	19,91
4 Smaczne Bistro	Brak	19,95
5 Naleśniczek	Brak	20,37
6 Krówka	Brak	19,76
7 Melon i Spółka	Brak	20,29
8 Burger & Co	Brak	20,27
9 Bistro Bułeczka	Brak	19,86
10 Gruszka	Brak	20,23
11 Burger & Co	Dla studentów	19,86
12 Tanio i smacznie	Dla studentów	20,1
13 Bistro Masełko	Dla studentów	19,96
14 Bistro Bułeczka	Dla studentów	19,86
15 Smaczne Bistro	Dla studentów	19,94
16 Naleśniczek	Dla studentów	20,12
17 Melon i Spółka	Dla studentów	19,83

Rank:

1. Ranking restauracji ze względu na sprzedaż

```

SELECT
    BIS.ID,
    BIS.NAME,
    MAIN.COUNT "SALES COUNT",
    DENSE_RANK() OVER (
        ORDER BY
            MAIN.COUNT DESC
    ) AS RANK
FROM
    (
        SELECT
            INV.BISTRO_ID as ID,
            COUNT(*) AS COUNT
        FROM
            INVOICE INV
        GROUP BY
            INV.BISTRO_ID
    ) MAIN
JOIN BISTRO BIS ON BIS.ID = MAIN.ID;

```

ID	NAME	SALES COUNT	RANK
1	9 Gruszka	30211	1
2	1 Naleśniczek	30126	2
3	8 Jabłko	30085	3
4	4 Krówka	30065	4
5	5 Smaczne Bistro	30003	5
6	6 Tanio i smacznie	30002	6
7	7 Bistro Bułeczka	30001	7
8	10 Melon i Spaghetti	29912	8
9	3 Burger & Co	29833	9
10	2 Bistro Masełko	29762	10

2. Ranking ze względu na oceny klientów

```

SELECT
    BIS.ID,
    BIS.NAME,
    MAIN.AVG_RATING AVG_RATING,
    DENSE_RANK() OVER (
        ORDER BY
            MAIN.AVG_RATING DESC
    ) AS RANK
FROM
    (
        SELECT
            INV.BISTRO_ID as ID,
            ROUND(AVG(INV.CLIENT_RATING), 2) AVG_RATING
        FROM
            INVOICE INV
        GROUP BY
            INV.BISTRO_ID
    ) MAIN
JOIN BISTRO BIS ON BIS.ID = MAIN.ID;

```

ID	NAME	AVG_RATING	RANK
1	4 Krówka	5,53	1
2	2 Bistro Masełko	5,53	1
3	8 Jabłko	5,53	1
4	1 Naleśniczek	5,52	2
5	9 Gruszka	5,51	3
6	10 Melon i Spaghetti	5,51	3
7	7 Bistro Bułeczka	5,5	4
8	5 Smaczne Bistro	5,49	5
9	3 Burger & Co	5,48	6
10	6 Tanio i smacznie	5,48	6

3. Ranking szefów kuchni

```

SELECT
    CHI.ID,
    CHI.NAME,
    CHI.SURNAME,
    MAIN.AVG_RATING,
    DENSE_RANK() OVER (
        ORDER BY
            MAIN.AVG_RATING DESC
    ) AS RANK
FROM
    (
        SELECT
            INV.CHIEF_ID AS ID,
            ROUND(AVG(INV.CLIENT_RATING), 2) AVG_RATING
        FROM
            INVOICE INV
        GROUP BY
            INV.CHIEF_ID
    ) MAIN
JOIN CHIEF CHI ON CHI.ID = MAIN.ID

```

ID	NAME	SURNAME	AVG_RATING	RANK
1	41 Kornelia	Ąwirko	5,64	1
2	38 Wiktor	Gajowiak	5,63	2
3	7 Przemysław	Dolega	5,62	3
4	100 Oskar	Ątyp	5,61	4
5	85 Wojciech	Ropiak	5,61	4
6	35 Marcelina	Macias	5,61	4
7	63 Paweł	Grzelec	5,61	4
8	21 Kaja	Szyc	5,59	5
9	17 Nicole	Uroda	5,59	5
10	54 Adam	Chatys	5,58	6
11	4 Dawid	Bogaczyk	5,58	6
12	93 Bruno	Pachołek	5,58	6
13	19 Aniela	Wasiuk	5,57	7
14	95 Hubert	Steuer	5,56	8
15	62 Marek	Siarkiewicz	5,56	8
16	37 Miłosz	Jajko	5,56	8
17	9 Rafał	Kucharz	5,56	8

Rollup:

1. Zapytanie pokazuje utarg w danych miesięcach i latach.

```
SELECT
    nvl(bis.name, 'ALL') bistro_name,
    inv.client_rating,
    COUNT(inv.id) invoice_count,
    SUM(inv.price_value) takings
FROM
    invoice inv
    JOIN bistro bis ON bis.id = inv.bistro_id
GROUP BY
    rollup(bis.name, inv.client_rating)
ORDER BY
    bis.name;
```

	BISTRO_NAME	CLIENT_RATING	INVOICE_COUNT	TAKINGS
1	Bistro BuÅeczka	6	2944	1455307
2	Bistro BuÅeczka	7	2992	1484551
3	Bistro BuÅeczka	8	2963	1483727
4	Bistro BuÅeczka	9	2992	1507750
5	Bistro BuÅeczka	10	3081	1544382
6	Bistro BuÅeczka	(null)	30001	15016841
7	Bistro BuÅeczka	4	2977	1484444
8	Bistro BuÅeczka	3	2995	1503159
9	Bistro BuÅeczka	2	3049	1520027
10	Bistro BuÅeczka	1	2997	1513848
11	Bistro BuÅeczka	5	3011	1519646
12	Bistro MaseÅko	5	2974	1495707
13	Bistro MaseÅko	4	3016	1495653
14	Bistro MaseÅko	3	2996	1509477
15	Bistro MaseÅko	2	2913	1448826
16	Bistro MaseÅko	1	2868	1419869
17	Bistro MaseÅko	6	3004	1496493

2. Zapytanie zwraca liczbę sprzedaży w danym bistro w danym miesiącu i roku danego produktu.

```
SELECT
    nvl(bis.name, 'OVERALL RATING') bistro_name,
    inv.month,
    ROUND(AVG(INV.client_rating),6) avg_rate
FROM
    invoice inv
JOIN bistro bis ON bis.id = inv.bistro_id
GROUP BY
    rollup(bis.name, inv.month)
ORDER BY
    bis.name, inv.month;
```

	BISTRO_NAME	MONTH	AVG_RATE
1	Bistro Bułeczka	1	5,509031
2	Bistro Bułeczka	2	5,361451
3	Bistro Bułeczka	3	5,430237
4	Bistro Bułeczka	4	5,512393
5	Bistro Bułeczka	5	5,657792
6	Bistro Bułeczka	6	5,572005
7	Bistro Bułeczka	7	5,470754
8	Bistro Bułeczka	8	5,701747
9	Bistro Bułeczka	9	5,456513
10	Bistro Bułeczka	10	5,434194
11	Bistro Bułeczka	11	5,501386
12	Bistro Bułeczka	12	5,428103
13	Bistro Bułeczka	(null)	5,502917
14	Bistro Masełko	1	5,549324
15	Bistro Masełko	2	5,622015
16	Bistro Masełko	3	5,47621
17	Bistro Masełko	4	5,557959

3. Zapytanie zwraca liczbę zamówień danego menu w restauracji

```
SELECT
    nvl(bis.name, 'OVERALL') bistro_name,
    nvl(poc.type, 'OVERALL TYPE') TYPE,
    count(poc.type)
FROM
    invoice inv
JOIN bistro bis ON bis.id = inv.bistro_id
JOIN place_of_consumption poc ON poc.id = inv.poc_id
GROUP BY
    rollup(bis.name, poc.type)
ORDER BY
    bis.name, poc.type
;
```

	BISTRO_NAME	TYPE	COUNT(POC.TYPE)
1	Bistro Bu ^ł ,eczka	Dostawa	10021
2	Bistro Bu ^ł ,eczka	Na_miejscu	9819
3	Bistro Bu ^ł ,eczka	Na_wynos	10161
4	Bistro Bu ^ł ,eczka	OVERALL TYPE	30001
5	Bistro Mase ^ł ,ko	Dostawa	9781
6	Bistro Mase ^ł ,ko	Na_miejscu	9933
7	Bistro Mase ^ł ,ko	Na_wynos	10048
8	Bistro Mase ^ł ,ko	OVERALL TYPE	29762
9	Burger & Co	Dostawa	9965
10	Burger & Co	Na_miejscu	9869
11	Burger & Co	Na_wynos	9999
12	Burger & Co	OVERALL TYPE	29833
13	Gruszka	Dostawa	9998
14	Gruszka	Na_miejscu	9945
15	Gruszka	Na_wynos	10268
16	Gruszka	OVERALL TYPE	30211
17	Jab ^ł ,ko	Dostawa	9970

Window:

1. Zapytanie zwraca przyrosty dochodu z miesiąca na miesiąc

```
SELECT
    YEAR,
    MONTH,
    PV AS "THIS MONTH TAKINGS",
    Sum(PV) over (
        PARTITION BY YEAR
        ORDER BY
            MONTH RANGE BETWEEN unbounded preceding
            AND CURRENT ROW
    ) AS "TAKINGS SUM TO THIS MONTH"
from
(
    SELECT
        MONTH,
        YEAR,
        SUM(PRICE_VALUE) PV
    FROM
        INVOICE
    GROUP BY
        MONTH,
        YEAR
)
ORDER BY
    YEAR,
    MONTH;
```

	YEAR	MONTH	THIS MONTH TAKINGS	TAKINGS SUM TO THIS MONTH
1	1997	1	484083	484083
2	1997	2	461167	945250
3	1997	3	453292	1398542
4	1997	4	440684	1839226
5	1997	5	449444	2288670
6	1997	6	460190	2748860
7	1997	7	481892	3230752
8	1997	8	444516	3675268
9	1997	9	491739	4167007
10	1997	10	437610	4604617
11	1997	11	480326	5084943
12	1997	12	444104	5529047
13	1998	1	442104	442104
14	1998	2	441319	883423
15	1998	3	476546	1359969
16	1998	4	489257	1849226
17	1998	5	481929	2331155

2. Zapytanie zwraca różnicę w ilości sprzedaży między bieżącym miesiącem a poprzednim.

```
SELECT
    YEAR,
    MONTH,
    L_SPRZED AS INVOICES_COUNT,
    NVL(
        L_SPRZED - (
            Lag(L_SPRZED, 1) over (
                ORDER BY
                    YEAR,
                    MONTH
            )
        ),
        0
    ) AS "DIFF BETWEEN MONTHS"
from
(
    SELECT
        MONTH,
        YEAR,
        COUNT(*) L_SPRZED
    FROM
        INVOICE
    GROUP BY
        MONTH,
        YEAR
)
ORDER BY
    YEAR ASC,
    MONTH ASC;
```

	YEAR	MONTH	INVOICES_COUNT	DIFF BETWEEN MONTHS
1	1997	1	992	0
2	1997	2	938	-54
3	1997	3	918	-20
4	1997	4	918	0
5	1997	5	895	-23
6	1997	6	912	17
7	1997	7	965	53
8	1997	8	916	-49
9	1997	9	951	35
10	1997	10	898	-53
11	1997	11	952	54
12	1997	12	921	-31
13	1998	1	909	-12
14	1998	2	884	-25
15	1998	3	937	53
16	1998	4	978	41
17	1998	5	939	-39

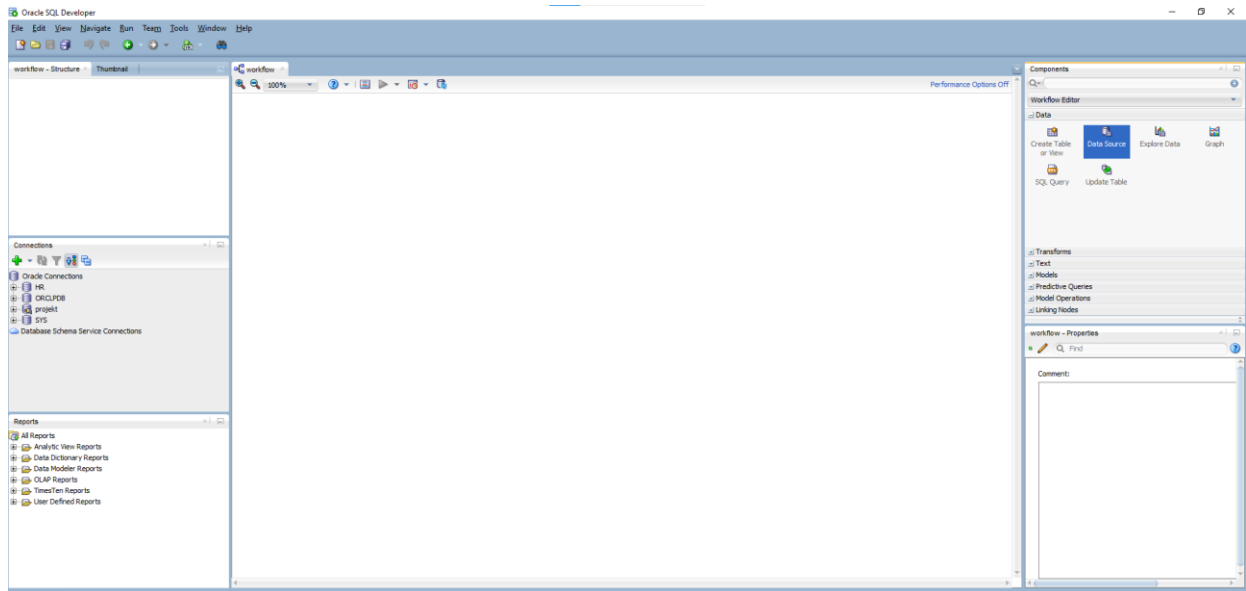
3. Zapytanie zwraca, jak się zmieniała średnia ocen klientów.

```
SELECT
    YEAR,
    MONTH,
    A_RAT AS AVG_RATING,
    NVL(
        A_RAT - (
            Lag(A_RAT, 1) over (
                ORDER BY
                    YEAR,
                    MONTH
            )
        ),
        0
    ) AS "DIFF BETWEEN MONTHS"
from
    (
        SELECT
            MONTH,
            YEAR,
            ROUND(AVG(CLIENT_RATING), 2) A_RAT
        FROM
            INVOICE
        GROUP BY
            MONTH,
            YEAR
    )
ORDER BY
    YEAR ASC,
    MONTH ASC;
```

	YEAR	MONTH	AVG_RATING	DIFF BETWEEN MONTHS
1	1997	1	5,42	0
2	1997	2	5,34	-0,08
3	1997	3	5,48	0,14
4	1997	4	5,57	0,09
5	1997	5	5,38	-0,19
6	1997	6	5,31	-0,07
7	1997	7	5,57	0,26
8	1997	8	5,47	-0,1
9	1997	9	5,46	-0,01
10	1997	10	5,54	0,08
11	1997	11	5,54	0
12	1997	12	5,47	-0,07
13	1998	1	5,32	-0,15
14	1998	2	5,58	0,26
15	1998	3	5,57	-0,01
16	1998	4	5,65	0,08
17	1998	5	5,46	-0,19

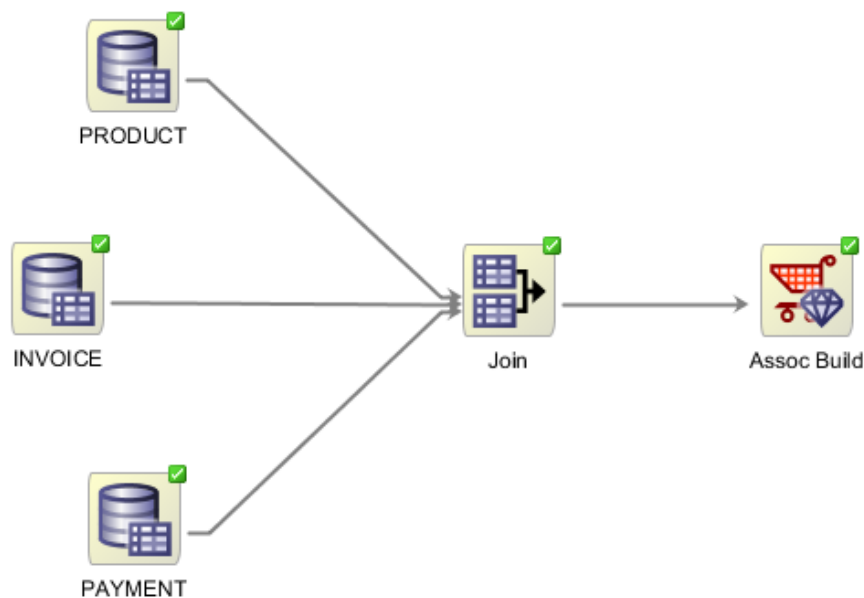
6. Eksploracja danych

Po uruchomieniu data minera należy stworzyć nowy projekt. Dla każdego podejścia eksploracji danych został stworzony osobny workflow. Do zbudowania schematu zostało wykorzystane menu po prawej stronie, z którego możemy tworzyć schematy.

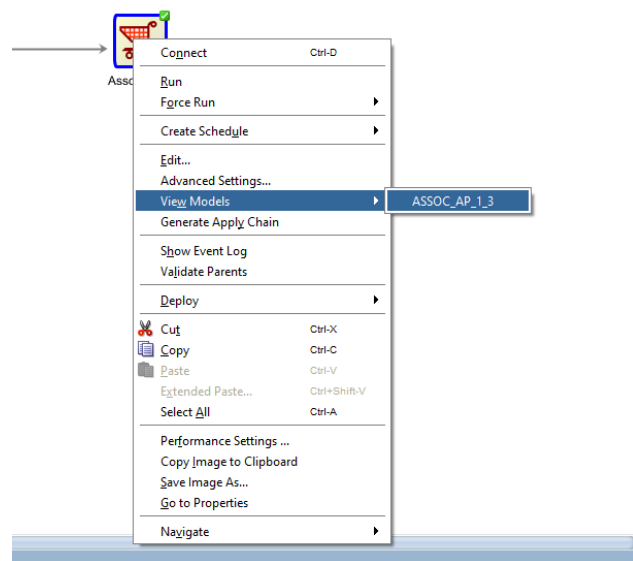


1. Reguły asocjacyjne

Schemat:



Po uruchomieniu modelu możemy podejrzeć wynik. Jest on dostępny po kliknięciu prawym klawiszem na ostatni moduł.



Otrzymany wynik:

Welcome Page workflow ASSOC_AP_1_3

Rules ItemSets Settings Alerts

Sort by: Lift Ascending Descending

Fetch Size: 1 000 Query More

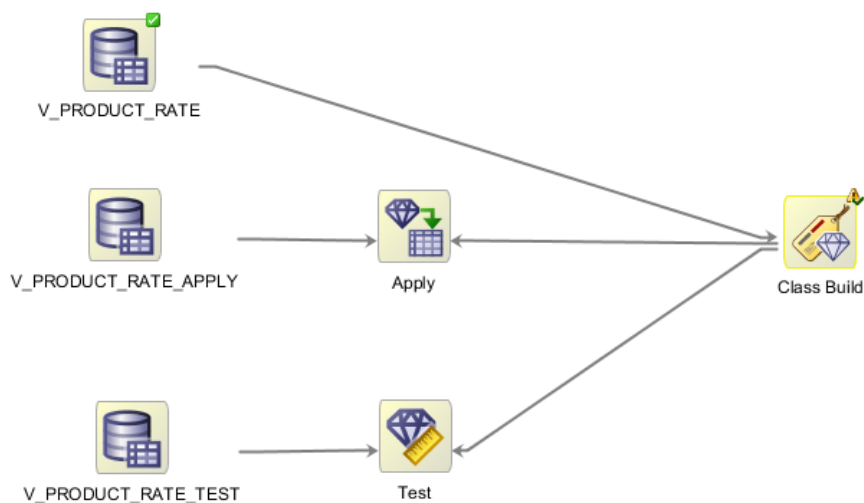
Filter No Filter

Rules: 1 000 out of 34 342

ID	Antecedent	Consequent	Lift	Confidence(%)	Support(%)	Item Count	Antecedent Support(%)	Consequent Support(%)	Reverse Confidence
26 938	Coffie AND Ragu alla Bolognese AND Ragu alla Carbonara	Tea	1,3703	49,1510	2,7673	4	5,6302	35,8692	7,7150
26 106	Coffie AND Mirinda AND Ragu alla Bolognese	Salmon salad	1,3561	48,7319	2,7069	4	5,5547	35,9346	7,5329
26 138	Coffie AND Mirinda AND Ragu alla Carbonara	Tea	1,3466	48,2999	2,7874	4	5,7711	35,8692	7,7711
18 464	Burger AND Salmon salad AND Schabowy	Pasta al basilico pesto	1,3464	48,0427	2,7170	4	5,6553	35,6830	7,6142
26 137	Coffie AND Mirinda AND Tea	Ragu alla Carbonara	1,3462	47,5945	2,7874	4	5,8566	35,3560	7,8839
32 954	Mushroom soup AND Ragu alla Carbonara AND Salmon salad	Tea	1,3433	48,1818	2,6667	4	5,5346	35,8692	7,4344
18 383	Pasta al basilico pesto AND Pierogis AND Tea	Burger	1,3420	48,9517	2,7019	4	5,5195	36,4780	7,4069
6 369	Asparagus soup AND Burger AND Schabowy	Pasta al basilico pesto	1,3417	47,8751	2,7774	4	5,8013	35,6830	7,7834
32 178	Mirinda AND Pizza Margherita AND Ragu alla Carbonara	Tea	1,3412	48,1091	2,7522	4	5,7208	35,8692	7,6729
29 425	Greek salad AND Mirinda AND Stroganov	Salmon salad	1,3376	48,0668	2,7522	4	5,7258	35,9346	7,6589
32 062	Mirinda AND Pierogis AND Ragu alla Bolognese	Salmon salad	1,3361	48,0138	2,7975	4	5,8264	35,9346	7,7849
9 663	Mushroom soup AND Ragu alla Bolognese AND Salmon salad	Asparagus soup	1,3352	47,4978	2,7220	4	5,7308	35,5723	7,6521
34 134	Pizza Margherita AND Ragu alla Carbonara AND Salmon salad	Tea	1,3344	47,8656	2,6516	4	5,5396	35,8692	7,3923
31 837	Mirinda AND Pappa al pomodoro AND Tea	Ragu alla Carbonara	1,3343	47,1747	2,7723	4	5,8767	35,3560	7,8412
10 904	Big beef AND Pappa al pomodoro AND Water	Burger	1,3336	48,6462	2,7119	4	5,5748	36,4780	7,4345
18 448	Burger AND Ragu alla Carbonara AND Schabowy	Pasta al basilico pesto	1,3334	47,5806	2,6717	4	5,6151	35,6830	7,4873
29 340	Greek salad AND Pierogis AND Tea	Mirinda	1,3327	48,4461	2,6667	4	5,5044	36,3522	7,3356
18 047	Mushroom soup AND Pierogis AND Salmon salad	Burger	1,3327	48,6135	2,8226	4	5,8063	36,4780	7,7379
17 537	Burger AND Herring salad AND Schabowy	Pasta al basilico pesto	1,3312	47,5000	2,7723	4	5,8365	35,6830	7,7693
18 964	Caesar salad AND Greek salad AND Herring salad	Chicken breasts	1,3306	48,1621	2,5711	4	5,3384	36,1962	7,1031

2. Klasyfikacja

Schemat został zbudowany w taki sam sposób jak poprzedni, ale z wykorzystaniem innych bloków.



Aby można było podzielić dane na testowe, uczące oraz weryfikujące zostały stworzone 3 widoki dzielące je na równe części.

```
Create or replace view V_PRODUCT_RATE AS
SELECT
    inv.product_id,
    pr.category,
    inv.id,
    CASE
        WHEN inv.client_rating < 6 THEN 'GOOD' ELSE 'BAD'
    END rating
FROM invoice inv
JOIN product pr ON inv.product_id = pr.id
and inv.id < 66000;

Create or replace view V_PRODUCT_RATE_TEST AS
SELECT
    inv.product_id,
    pr.category,
    inv.id,
    CASE
        WHEN inv.client_rating < 6 THEN 'GOOD' ELSE 'BAD'
    END rating
FROM invoice inv
JOIN product pr ON inv.product_id = pr.id
and inv.id between 66000 and 132000;

Create or replace view V_PRODUCT_RATE_APPLY AS
SELECT
    inv.product_id,
    pr.category,
    inv.id,
    CASE
        WHEN inv.client_rating < 6 THEN 'GOOD' ELSE 'BAD'
    END rating
FROM invoice inv
JOIN product pr ON inv.product_id = pr.id
and inv.id between 132000 AND 200000;
```

Po uruchomieniu otrzymujemy wyniki:

Welcome Page

workflow1

CLAS_GLM_1_4

CLAS_SVM_1_4

Details

Coefficients

Compare

Settings

Alerts

Target Value:

BAD

☒ Sort by absolute value

Coefficients: 25 out of 25

Attribute	Value		Standardized Coefficient	Coefficient	Exp(Coefficient)
CATEGORY	Salads		0,43205084	3,74425007	42,27729012
PRODUCT_ID	6		-0,21313256	-3,74571674	0,02361869
PRODUCT_ID	7		-0,21176863	-3,69996128	0,02472448
PRODUCT_ID	5		-0,21121245	-3,70117095	0,02469459
PRODUCT_ID	8		-0,21089501	-3,67304719	0,02539896
PRODUCT_ID	9		-0,20790310	-3,58256627	0,02780425
CATEGORY	Drinks		0,19375497	1,83386551	6,25803044
PRODUCT_ID	2		-0,10234460	-1,79535783	0,16606802
PRODUCT_ID	4		-0,10176306	-1,80628488	0,16426326
PRODUCT_ID	3		-0,09972768	-1,72330736	0,17847489
PRODUCT_ID	1		-0,09932117	-1,75193649	0,17343776
CATEGORY	Soups		-0,07562158	-0,70983849	0,49172361
PRODUCT_ID	12		0,04721853	0,82111751	2,27303856
PRODUCT_ID	10		0,04344697	0,74795848	2,11268253
PRODUCT_ID	11		0,04205931	0,73601423	2,08759823
PRODUCT_ID	13		0,04176458	0,72636867	2,06755898
PRODUCT_ID	17		0,00525285	0,09071987	1,09496223
PRODUCT_ID	20		0,00525169	0,09079485	1,09504433
PRODUCT_ID	22		0,00505279	0,08788705	1,09186479
PRODUCT_ID	15		0,00492990	0,08543240	1,08918793
PRODUCT_ID	18		0,00407580	0,07136719	1,07397551
PRODUCT_ID	21		0,00401958	0,07013757	1,07265574
PRODUCT_ID	19		0,00341055	0,06010776	1,06195097
PRODUCT_ID	14		0,00249992	0,04336539	1,04431941
<Intercept>			0,00000000	-0,06241080	0,93949686

Welcome Page

workflow1

CLAS_GLM_1_4

CLAS_SVM_1_4

CL

Details

Coefficients

Compare

Settings

Alerts

Target Value:

GOOD

☒ Sort by absolute value

Coefficients: 25 out of 25

Attribute	Value	Standardized Coefficient	Coefficient	Exp(Coefficient)
CATEGORY	Salads	<div></div> -0,43205084	-3,74425007	0,02365336
PRODUCT_ID	6	<div></div> 0,21313256	3,74571674	42,33934245
PRODUCT_ID	7	<div></div> 0,21176863	3,69996128	40,44573841
PRODUCT_ID	5	<div></div> 0,21121245	3,70117095	40,49469395
PRODUCT_ID	8	<div></div> 0,21089501	3,67304719	39,37169646
PRODUCT_ID	9	<div></div> 0,20790310	3,58256627	35,96572041
CATEGORY	Drinks	<div></div> -0,19375497	-1,83386551	0,15979468
PRODUCT_ID	2	<div></div> 0,10234460	1,79535783	6,02162907
PRODUCT_ID	4	<div></div> 0,10176306	1,80628488	6,08778852
PRODUCT_ID	3	<div></div> 0,09972768	1,72330736	5,60302909
PRODUCT_ID	1	<div></div> 0,09932117	1,75193649	5,76575718
CATEGORY	Soups	<div></div> 0,07562158	0,70983849	2,03366277
PRODUCT_ID	12	<div></div> -0,04721853	-0,82111751	0,43993974
PRODUCT_ID	10	<div></div> -0,04344697	-0,74795848	0,47333188
PRODUCT_ID	11	<div></div> -0,04205931	-0,73601423	0,47901938
PRODUCT_ID	13	<div></div> -0,04176458	-0,72636867	0,48366214
PRODUCT_ID	17	<div></div> -0,00525285	-0,09071987	0,91327351
PRODUCT_ID	20	<div></div> -0,00525169	-0,09079485	0,91320504
PRODUCT_ID	22	<div></div> -0,00505279	-0,08788705	0,91586431
PRODUCT_ID	15	<div></div> -0,00492990	-0,08543240	0,91811520
PRODUCT_ID	18	<div></div> -0,00407580	-0,07136719	0,93111993
PRODUCT_ID	21	<div></div> -0,00401958	-0,07013757	0,93226556
PRODUCT_ID	19	<div></div> -0,00341055	-0,06010776	0,94166306
PRODUCT_ID	14	<div></div> -0,00249992	-0,04336539	0,95756144
<Intercept>		0,00000000	0,06241080	1,06439952

Welcome Page

workflow1

CLAS_SVM_1_4

CLAS_DT_1_4

CLAS_NB_1_4

Coefficients

Compare

Settings

Alerts

Target Value:

BAD

Sort by absolute value

Fetch Size:

1 000

Query

Coefficients: 27 out of 27

Attribute

Value

Coefficient

PRODUCT_ID

12

1,41599983

PRODUCT_ID

16

-1,29200003

PRODUCT_ID

14

-1,29199982

PRODUCT_ID

19

-1,29199978

PRODUCT_ID

9

1,17999993

PRODUCT_ID

8

1,17999941

PRODUCT_ID

3

1,01599990

PRODUCT_ID

1

1,01599972

PRODUCT_ID

4

-0,98399988

PRODUCT_ID

2

-0,98399979

PRODUCT_ID

6

-0,81999994

PRODUCT_ID

5

-0,81999973

PRODUCT_ID

7

-0,81999971

PRODUCT_ID

20

0,70799973

PRODUCT_ID

17

0,70799973

PRODUCT_ID

22

0,70799970

PRODUCT_ID

15

0,70799967

PRODUCT_ID

18

0,70799932

PRODUCT_ID

21

0,70799927

PRODUCT_ID

13

-0,58399998

PRODUCT_ID

11

-0,58399998

PRODUCT_ID

10

-0,58399988

CATEGORY

Main courses

0,37200006

CATEGORY

Soups

-0,33599998

CATEGORY

Salads

-0,10000004

<Intercept>

-0,07999995

CATEGORY

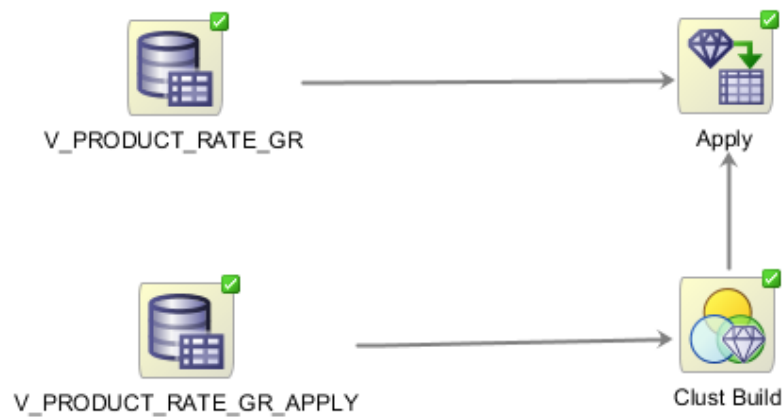
Drinks

0,06399995

Welcome Page workflow1 CLAS_SVM_1_4 CLAS_DT_1_4 CLAS_NB_1_4		
Coefficients Compare Settings Alerts		
Target Value: GOOD	Sort by absolute value	Fetch Size: 1 000 Query
Coefficients: 27 out of 27		
Attribute	Value	Coefficient
PRODUCT_ID	12	-1,41599983
PRODUCT_ID	16	1,29200003
PRODUCT_ID	14	1,29199982
PRODUCT_ID	19	1,29199752
PRODUCT_ID	9	-1,17999993
PRODUCT_ID	8	-1,17999941
PRODUCT_ID	3	-1,41599990
PRODUCT_ID	1	-1,41599972
PRODUCT_ID	4	0,98399988
PRODUCT_ID	2	0,98399979
PRODUCT_ID	6	0,81999994
PRODUCT_ID	5	0,81999973
PRODUCT_ID	7	0,81999971
PRODUCT_ID	20	-0,70799973
PRODUCT_ID	17	-0,70799973
PRODUCT_ID	22	-0,70799970
PRODUCT_ID	15	-0,70799967
PRODUCT_ID	18	-0,70799932
PRODUCT_ID	21	-0,70799927
PRODUCT_ID	13	0,58399998
PRODUCT_ID	11	0,58399996
PRODUCT_ID	10	0,58399987
CATEGORY	Main courses	-0,37200006
CATEGORY	Soups	0,33999998
CATEGORY	Salads	0,10000004
CATEGORY	<Intercept>	0,07999995
CATEGORY	Drinks	-0,06399995

3. Grupowanie

Zostało wykonane podobnie do dwóch poprzednich schematów jednak z innymi blockami.



Otrzymane wyniki:

Welcome Page
workflow1
CLAS_NB_1_4

Probabilities
Compare
Settings
Alerts

Target Value: BAD
Fetch Size: 1000
Query

Probabilities: 1 out of 1
Attribute
Value
Probability(%) for BAD
Attribute

<PRIOR>	NULL	50,00000000
---------	------	-------------

Welcome Page
workflow1
CLAS_DT_1_4
CLAS_NB_1_4

Performance
Performance Matrix
ROC
Lift
Profit

Measure: All Measures
Sort By: Name
Descending

Predictive Confidence (%)

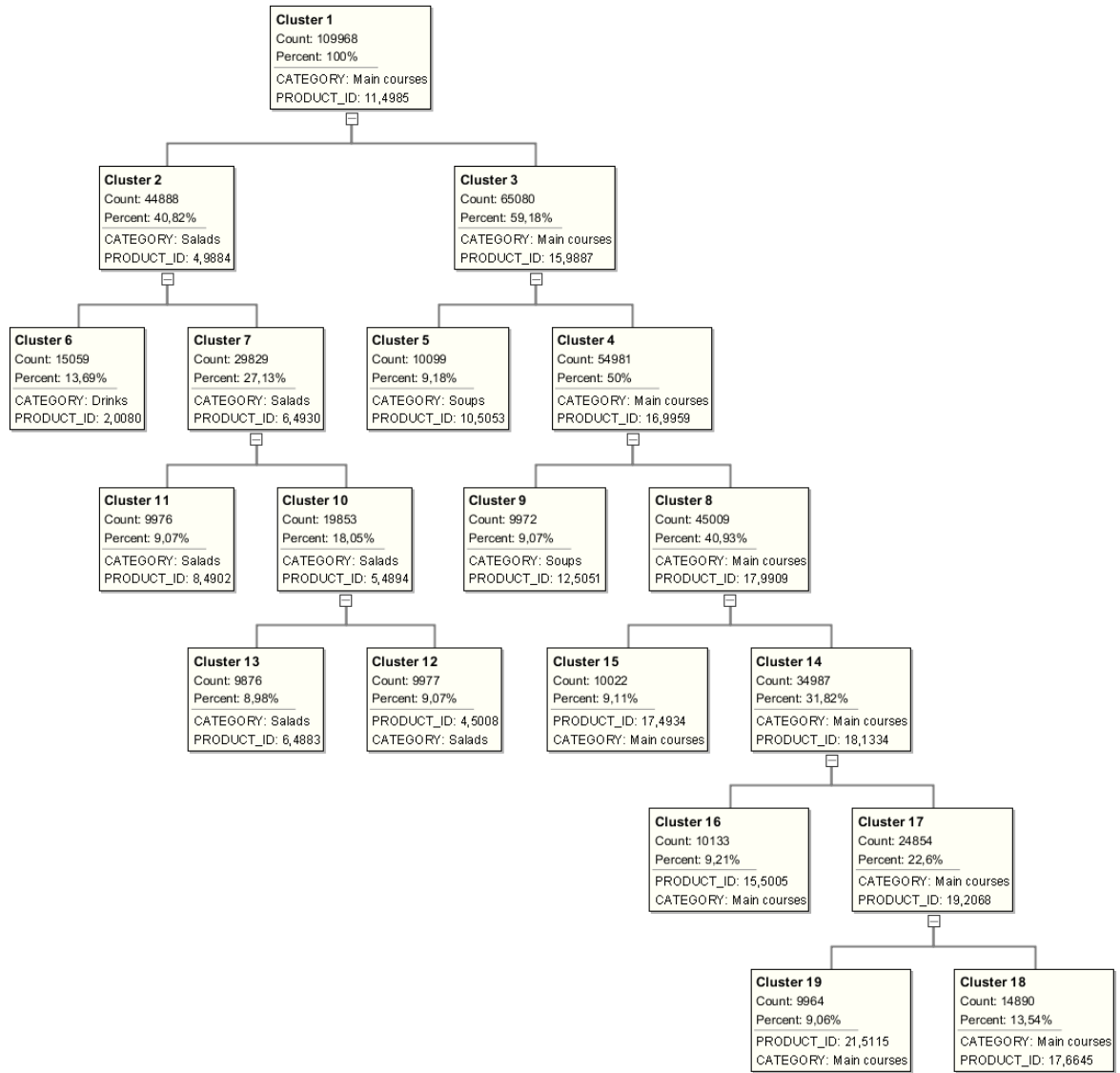
Average Accuracy (%)

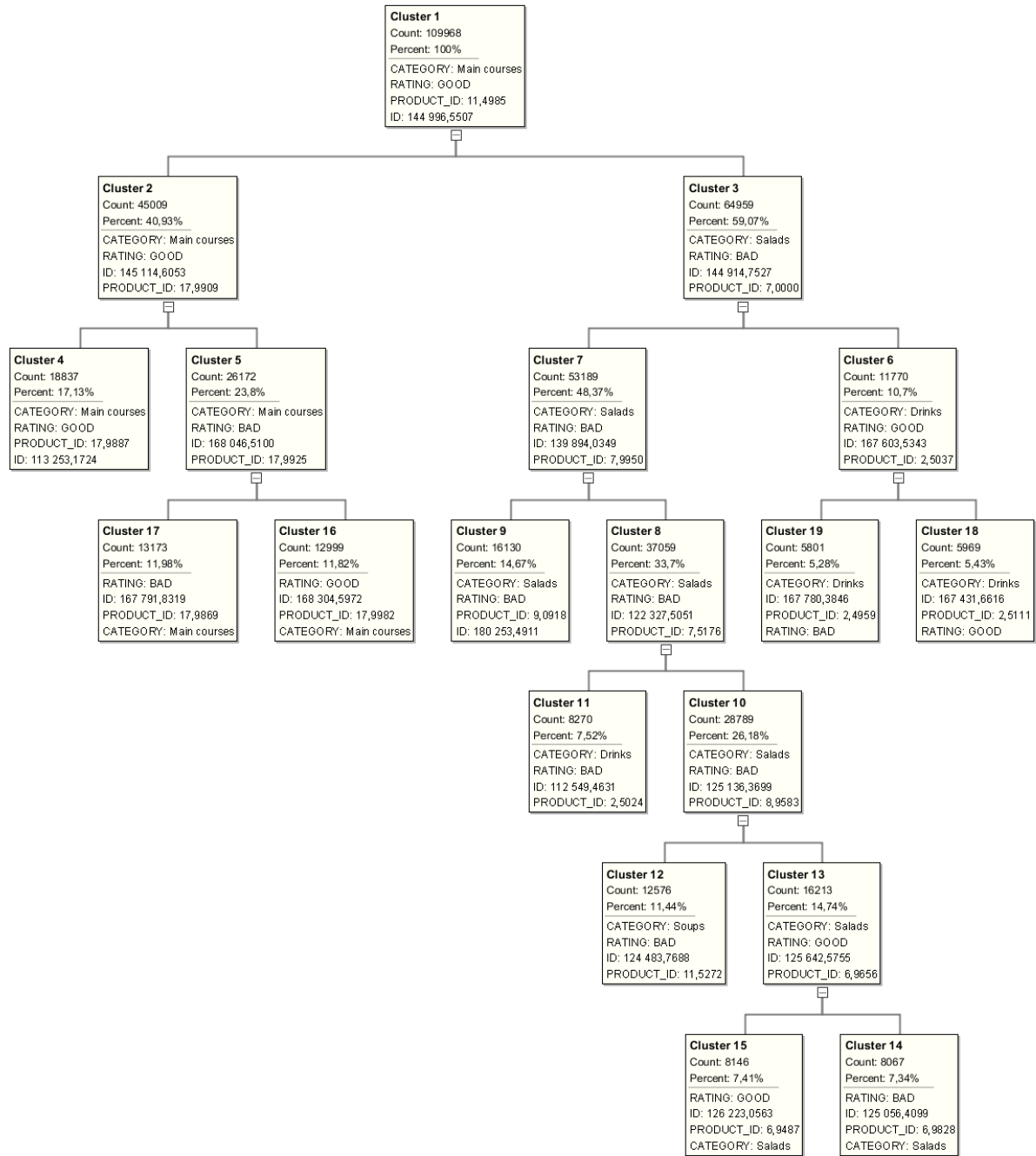
Overall Accuracy (%)

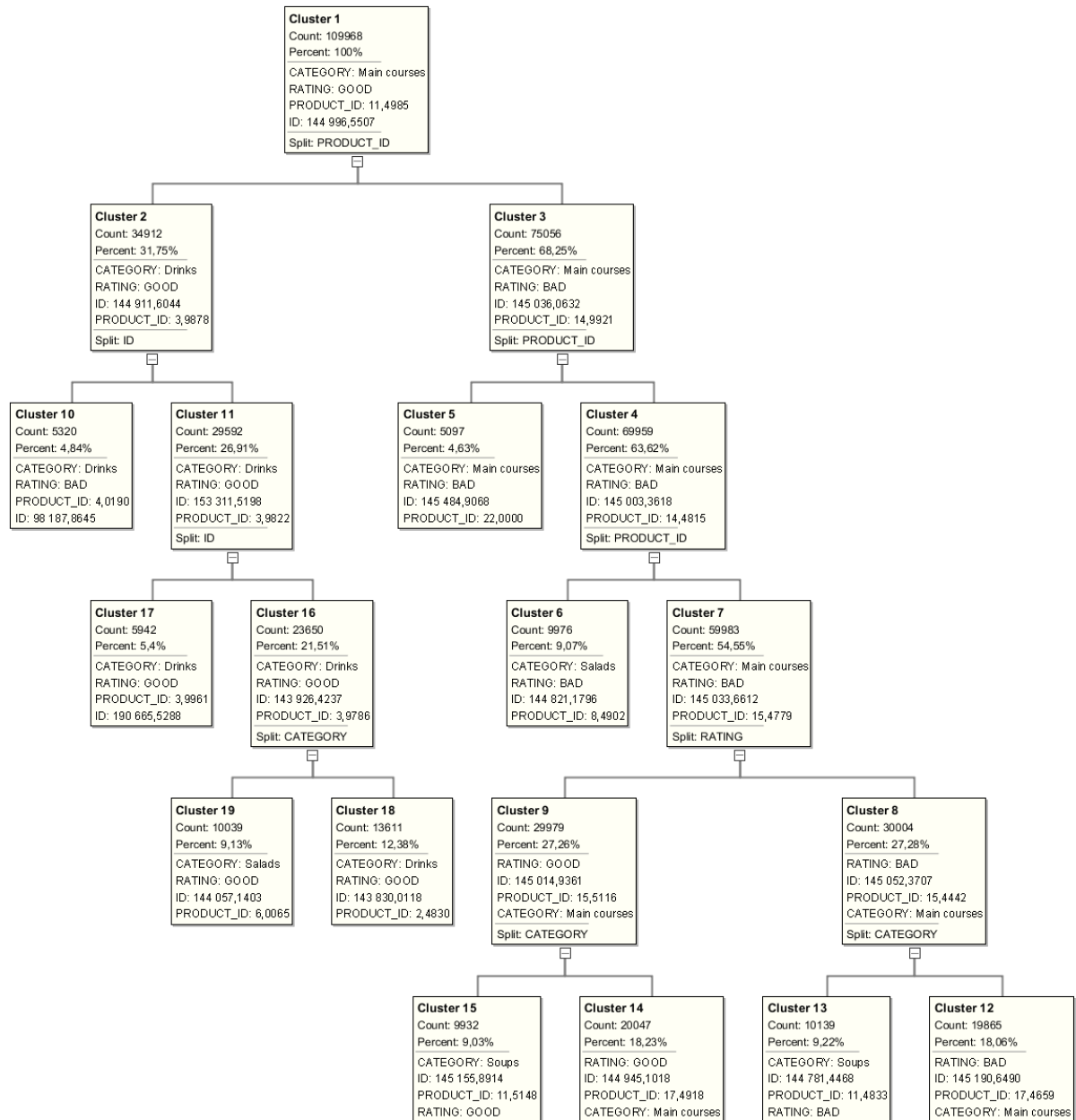
Cost

Models

Name	Predictive Confidence %	Overall Accuracy %	Average Accuracy %	Cost	Algorithm	Creation Date
CLAS_DT_1_4	0,0000	49,8892	50,0000	26 601,63425879	Decision Tree	11.01.23 21:56







7. Wnioski

Projekt pozwolił zbudować hurtownię danych oraz zapoznać się z pracą z nimi. Mogliśmy poznać jak optymalnie stworzyć hurtownię aby gromadzone dane były przydatne dla biznesu.

Stworzone zapytania pozwoliły nam się zaznajomić z zaawansowanymi konceptami jakie możemy spotkać w bazach danych Oracle SQL. Pokazuje nam to jakie możliwości raportowania możemy osiągnąć samym kodem bez zewnętrznych aplikacji. Ze względu na losowość danych nie niosą one zbyt wiele informacji.

Poznany moduł data miner pozwolił nam spojrzeć na to jak dane są przetwarzane oraz jakie zależności można wyłapać. Dzięki temu wiemy, jak tworzyć takie modele oraz używać ich w praktyce.