# IUM – projekt, Etap 2

**Piotr Grabowski**

**Temat 9:**
*Czasami chcielibyśmy wiedzieć, co najczęściej chwalą i krytykują nasi goście.*

# Wprowadzenie

W tym etapie, na podstawie dostarczonych danych, zaimplementowano dwa modele: model bazowy oraz bardziej zaawansowany model docelowy, a także przygotowano mikroserwis umożliwiający ich wykorzystanie i porównanie. Przypominając etap1:

Problem został sformułowany jako połączenie dwóch zadań modelowych: grupowania fragmentów recenzji do aspektów oraz klasyfikacji ich sentymentu. Celem biznesowym to jest natomiast stworzenie automatycznej analizie recenzji tekstowych w celu identyfikacji najczęściej chwalonych i krytykowanych aspektów pobytu w lokalach firmy Nocarz.

# Preprocessing danych (part2_preprocess_data.ipynb)

Dla obu modeli wykonano identyczny wstępny preprocessing danych.

Wybrano trzy kluczowe kolumny posiadajace istotne dane:

- listing_id
- comments
- date

Usunięto rekordy z brakującymi wartościami w polach listing_id lub comments. Nie zrobiono tego dla date, bo mimo, że wartości z tej kolumny są przydatne to się są konieczne do modelu.
Usunięto zduplikowane komentarze na podstawie pary (listing_id, comments).
Usunięto również bardzo krótkie komentarze, ponieważ opinie typu „nice local” nie wnoszą wartościowych informacji o aspektach pobytu. Jako próg minimalnej długości wybrano 20 znaków. Decyzja ta została poparta analizą rozkładu długości komentarzy – średnia długość wynosiła około 275 znaków, a mediana 209 znaków. Komentarze krótsze niż 20 znaków stanowiły bardzo niewielki procent zbioru, więc ich usunięcie nie wpływało istotnie na pokrycie danych.
Zastosowano minimalne czyszczenie i normalizację tekstu (konwersję do małych liter,usunięcie linków, normalizację białych znaków)

Recenzje są wielojęzyczne, dlatego przeprowadzono detekcję języka. Przetestowano dwa podejścia: langdetect oraz fastText. Model fastText okazał się znacząco szybszy, a jednocześnie równie skuteczny w rozpoznawaniu języków, dlatego został wybrany do dalszego przetwarzania.

Dzięki temu modelowi przeanalizowano rozkład języków w zbiorze danych. Ponieważ język angielski stanowił ponad 50% wszystkich recenzji, dla modelu bazowego utworzono osobny dataset zawierający wyłącznie komentarze anglojęzyczne. Dodatkowo przygotowano dataset obejmujący 10

najczęściej występujących języków, którego pokrycie danych przekraczało 95%, oraz wariant zawierający wszystkie języki (jako zapasowy, ale nie użyty ostatecznie).

# Model bazowy (part2_modeling_basic.ipynb)

Model bazowy został ograniczony do języka angielskiego w celu uproszczenia pipeline'u oraz umożliwienia zastosowania bardzo szybkich, klasycznych metod NLP.
Komentarze zostały podzielone na pojedyncze zdania przy użyciu reguł opartych na znakach interpunkcyjnych.
Usunięto zdania bardzo krótkie i nieinformatywne, a jako próg przyjęto 10 znaków.
Podział na zdania wynika z przyjętego założenia, że jedno zdanie opisuje aspekt pobytu, który może zostać jednoznacznie sklasyfikowany - jest to główne założenie w obu modelach.

## Reprezentacja tekstu

Dla modelu bazowego wybrano reprezentację tekstu opartą na TF-IDF, ponieważ jest bardzo szybka obliczeniowo, łatwa do interpretacji. Ograniczeniem jest brak reprezentacji semantycznej – metoda opiera się wyłącznie na częstości występowania słów kluczowych, bez uwzględnienia ich znaczenia. Będzie to zatem dobry punkt odniesienia do modelu docelowego.

Parametry TF-IDF dobrano na podstawie eksperymentów na mniejszych próbkach danych:

- max_features = 25000
- ngram_range = (1, 2)
- min_df = 5

## Grupowanie

Do grupowania użyto podstawowego algorytmu K-Means. Ze względu na mniejszą liczbę danych niż w modelu docelowym zastosowano standardową wersję algorytmu (a nie MiniBatchKMeans).

Kluczowym parametrem była liczba klastrów. Na podstawie testów wybrano K = 22. Jakość klastrów oceniano jakościowo, analizując próbki 10-15 zdań z każdego klastra i ręcznie przypisując im znaczenie semantyczne. Przy każdej testowanej liczbie klastrów cześć trzeba było odrzucić jako źle pogrupowane/nic nie znaczące.

Ostatecznie klastry zostały zmapowane do 12  aspektów biznesowych (11 + 1 ogólny)

Ostatecznie zostawiono: Total sentences kept: 2,611,459 (62.5%)

Ostatecznie aspekty to:

overall_positive
host_communication
apartment_quality
host_arrival_services
transport_proximity
apartment_amenities

location_quality
neighborhood_services
environmental_conditions
equipment_supplies
space_location_features
infrastructure_accessibility

### Klasyfikacja

Do klasyfikacji sentymentu dla każdego komentarza użyto algorytmu VADER, ponieważ jest ekstremalnie szybki, nie wymaga użycia GPU i może być dobrym punktem odniesienia modelu docelowego. Ograniczeniem jego jest natomiast niższa dokładność oraz brak głębokiego rozumienia kontekstu. Progi klasyfikacji wzięto standarowe:

- compound ≥ 0.05 → positive
- compound ≤ -0.05 → negative
- w przeciwnym przypadku → neutral

Rozkład sentymentów to:

sentiment
positive  0.757152
neutral  0.192300
negative  0.050547

# Model zaawansowany (part2_modeling.ipynb)

Podobnie jak dla modelu bazowego, na początku wykonano krótki preprocessing danych.
Komentarze zostały podzielone na pojedyncze zdania przy użyciu reguł opartych na znakach interpunkcyjnych.
Usunięto zdania bardzo krótkie i nieinformatywne, a jako próg przyjęto 10 znaków.
Podział na zdania wynika z przyjętego założenia, że jedno zdanie opisuje aspekt pobytu, który może zostać jednoznacznie sklasyfikowany - jest to główne założenie w obu modelach.
Dla modelu docelowego wykorzystano dane wielojęzyczne. Wybrano dane zawierającej 10 najczęściej występujących języków (to >95% wszystkich komentarzy)
Ze względu na bardzo dużą liczbę zdań, na etapie testów i do doboru większości parametrów wykorzystywano podzbiory danych (np. 30% wszystkich komentarzy). Przy 6 milionach jest jest to reprezentatywna proóbka.
W finalnej wersji modelu użyto już wszystkich danych.

### Reprezentacja tekstu

Ze zdan stworzono ich reprezentacje semantyczne przy użyciu modeli typu Sentence-BERT.
Rozważane modele embeddingowe obejmowały:

(wielojęzyczne)

- intfloat/multilingual-e5-base

- intfloat/multilingual-e5-large
- intfloat/multilingual-e5-small
- paraphrase-multilingual-MiniLM-L12-v2

(tylko ang)

- all-mpnet-base-v2
- all-MiniLM-L6-v2

Modele paraphrase-MiniLM-L12-v2 oraz all-MiniLM-L6-v2 zostały odrzucony przez słabszą separację semantyczną klastrów (któe stworzono w kolejnej częsci). Wybrano model e5base, gdyż czas stworzenia reprezentacji dla large był o wiele dłuższy, a base daje zadowająco dobre wyniki. Ostatecznie wybrano:

- intfloat/multilingual-e5-base dla danych top-10 języków,
- all-mpnet-base-v2 dla danych tylko w języku angielskim.

## Grupowanie

Do grupowania rozważano pojdeścia:

- K-Means / MiniBatchKMeans
- BERTopic
- HDBSCAN

Zostały najpierw przetestowane na mniejszym podzbiorach danych.
HDBSCAN został przetestowany, jednak ze względu na bardzo długi czas działania został odrzucony, gdyż dla wszstkich danych procesowanie zajełoby za długo.
Metody porównywano dla 30% z datasetu dla tylko angielskich oraz dla wszystkich komentarzy.
Porównanie ilościowe jest poniżej:

Top10 języków

| Method | Time | Silhouette | Davies-Bouldin | Calinski-Harabasz |
|---|---|---|---|---|
| K-Means | 1734.66 | 0.0230 | 4.2587 | 2120.98 |
| BERTopic | 2003.84 | 0.0204 | 4.5557 | 1624.38 |

**B**rak klastrów szumowych (noise = 0%).

Tylko język angielski

| Method | Time | Silhouette | Davies-Bouldin | Calinski-Harabasz |
|---|---|---|---|---|
| K-Means | 1738.17 | 0.0381 | 3.6423 | 4052.99 |
| BERTopic | 918.40 | 0.0178 | 3.2466 | 1104.59 |

W tym wariancie BERTopic generował około 44% klastrów szumowych, co niekorzystnie świadczy o modelu.

Dla obu rodzajów danych widzimy, żę leszpe statystki ma K-Means. Dodatkowo dokonywano porównania jakościowego. Z grupowanych klastrów brano po ~10zdań i ręcznie sprawdzano sensowność podziało. Testowanano również różną wartość klastrów początkowych (K) i wartości w okolicach 18-20 były najlepsze dla 30% danych, oznacza to, że dla danych docelowych będzie maksymalnie to o kilka więcej

Dodatkowym plusem tej metody jest możliwość użycia MiniBatchKMeans dla pełnego zbioru danych, co znacząco przyspiesza obliczenia przy niewielkiej utracie jakości (przy ustawieniu dobrych parametrów).

Ostatecznie wybrano MiniBatchKMeans z K = 22 i parametrami:

- max_iter = 300
- reassignment_ratio = 0.01
- liczba epok: 5

Metryki jakości liczone na próbce 100k zdań

- Silhouette: ~0.25–0.30
- Davies-Bouldin: < 2.0

Klastry zostały  ręcznie przeanalizowane, oraz zmergowane, jeśli reprezentowały podobne tematy i oczyszczone z klastrów szumowych (ogólna/niewartościowa informacja).

Ostatecznie uzyskano 12 (11 + 1 overall) końcowych klastrów/aspektów dla pełnego, wielojęzycznego zbioru danych.

overall_positive
cleanliness
host_and_property
walkability
host_communication

dining_entertainment
location_close_center
arrival_check_in_experience
transport_connectivity
apartment_appearance
apartment_capacity
atmosphere_ambiance

### Klasyfikacja

Do klasyfikacji zdań wybrano:

- cardiffnlp/twitter-xlm-roberta-base-sentiment (dane wielojęzyczne)
- twitter-roberta-base-sentiment-latest (dane angielskie).

Modele te zostały wybrane, ponieważ są w miarę nowe, osiągają dobre wyniki, zwracają bezpośrednie etykiety positive/neutral/negative, a nie score. Dodatkowo model wielojęzyczny obsługuje wszystkie języki, które są w danych top10.

Rozważano także inne modele, ale problemem było znalezienie modeli trenowanych stricte na danych hotelowych, dlatego wybrano modele ogólnego przeznaczenia. Dodatkowo niektóre mają ocene liczbową, którą trudnej sklasyfikować na pozytywny/negatywny. Alternatywy któe rozważano to:

Tylko język angielski:

- distilbert-base-uncased-finetuned-sst-2
- siebert/sentiment-roberta-large-english
- j-hartmann/emotion-english-distilroberta-base
- nlptown/bert-base-multilingual-uncased-sentiment
- lxyuan/distilbert-base-multilingual-cased-sentiments-student

Otzrymano taki podział dla ostatecznego modelu:
sentiment
positive 0.750749
neutral 0.169533
negative 0.079718

# Pivot

Na koniec, dla obu modeli dokonano zagregowania wyników w celu efektywnego wykorzystania ich do mikroserwisu i testów A/B. Dane zagregowano poprzez grupowanie według tej kolejności:

1. listing_id
2. aspect
3. sentiment
4. date

Dodatkowo dla tej kombinacji liczona jest liczba zdań (komentarzy).

Dane przekształcono do tabeli przestawnej, gdzie

- kolumny odpowiadają typom sentymentu (positive, neutral, negative),
- wartościami są liczby wystąpień danego sentymentu.

Obliczono dodatkowe metryki:

- score = liczba opinii pozytywnych − liczba opinii negatywnych,
- total_mentions = łączna liczba wzmianek o danym aspekcie.

i stworzono funkcje, która dla zadanego listing_id zwraca:

- Top K najlepiej ocenianych aspektów (najwyższy score),
- Top K najgorzej ocenianych aspektów (najniższy score).

Jest to końcowy punkt obu modeli, który spełnia cel biznesowy.

# Testy (part2_testing.ipynb)

Przeprowadzono testy ręczne testy modelu poleagające na wylosowaniu puli zdań dla obu modeli i sprawdzenia przypisanych im klastrów oraz sentymentowi.W testach porównywano za równo te same zdania dla obu modeli jak i inne zdania. Cały proces budowy modelu kończył się tutaj i był kilkukrotnie powtarzany (iteracyjnie) aż do osiągniecia zadowalających wyników. Z uwagi na bardzo duża liczbę zdań powtarzono dla kilku seedów. Najczęstym miejscem do powrotu była konieczność zmiany nazw klastrów (lub drobna zmiana w postaci dodania któregoś do noise lub mergowania klastrów).

Ostatecznym testem akceptacyjnym było wybranie po 100 zdań dla każdego z modeli ocenienie wyników, które są następujące:

Model bazowy:

- Strict: 50.0% (50/100)

- Relaxed: 55.0% (55/100)

Sentiment accuracy: 98.0% (98/100)

Model docelowy:

- Strict: 60.0% (60/100)

- Relaxed: 84.0% (84/100)

Sentiment accuracy: 95.0% (95/100)

W ocenie aspektu użyto 3 stopniowej oceny: dobre (strict) , w miarę dobre dopasowanie i złe dopasowanie.

Relaxed oznacza dobre + w miarę dobre dopasowanie - nie zawsze łatwo ocenić czy pokrywa sie w 100%.

# Mikroserwis (folder microservice)

Mirkoserwis został zrealizowany w sposób przezroczysty dla klienta, a jednocześnie umożliwienie porównania dwóch modeli (bazowego i zaawansowanego) w ramach eksperymentu A/B.

Udostępnia endpoint /predict, który przyjmuje jedynie:

- listing_id – identyfikator lokalu,
- top_k – liczba zwracanych aspektów.

Klient nie wybiera modelu i nie otrzymuje informacji, czy użyto wariantu A (baseline) czy B (advanced).
Wybór modelu odbywa się po stronie serwera w sposób deterministyczny na podstawie hasha listing_id, co zapewnia:

- spójność wyników (ten sam lokal to zawsze ten sam wariant),
- równomierny podział danych (~50/50),
- pełną transparentność dla użytkownika końcowego.

przykładowy prompt (na windows):

curl.exe -X POST http://localhost:8080/predict -H "Content-Type: application/json" -d '{\"listing_id\": 10719987, \"top_k\": 3}'

a odpowiedź to:


```
{
  "listing_id": 10719987,
  "top_k": 3,
  "top_aspects": [
    {
      "aspect": "overall_positive",
      "score": 304.0,
      "positive": 317,
      "neutral": 31,
      "negative": 13,
      "total_mentions": 361
    },
    {
      "aspect": "host_communication",
      "score": 160.0,
      "positive": 173,
      "neutral": 39,
      "negative": 13,
      "total_mentions": 225
    },
    {
      "aspect": "apartment_quality",
      "score": 76.0,
      "positive": 76,
      "neutral": 11,
      "negative": 0,
      "total_mentions": 87
    }
  ],
  "bottom_aspects": [
    {
      "aspect": "infrastructure_accessibility",
      "score": -79.0,
      "positive": 3,
      "neutral": 17,
      "negative": 82,
      "total_mentions": 102
    },
    {
      "aspect": "equipment_supplies",
      "score": -55.0,
      "positive": 5,
      "neutral": 18,
      "negative": 60,
      "total_mentions": 83
    },
    {
      "aspect": "environmental_conditions",
      "score": -17.0,
      "negative": 41,
      "total_mentions": 102
    }
  ],
  "chart_url": "/predict/chart?listing_id=10719987",
  "timestamp": "2026-01-17T21:59:06.380997"
```

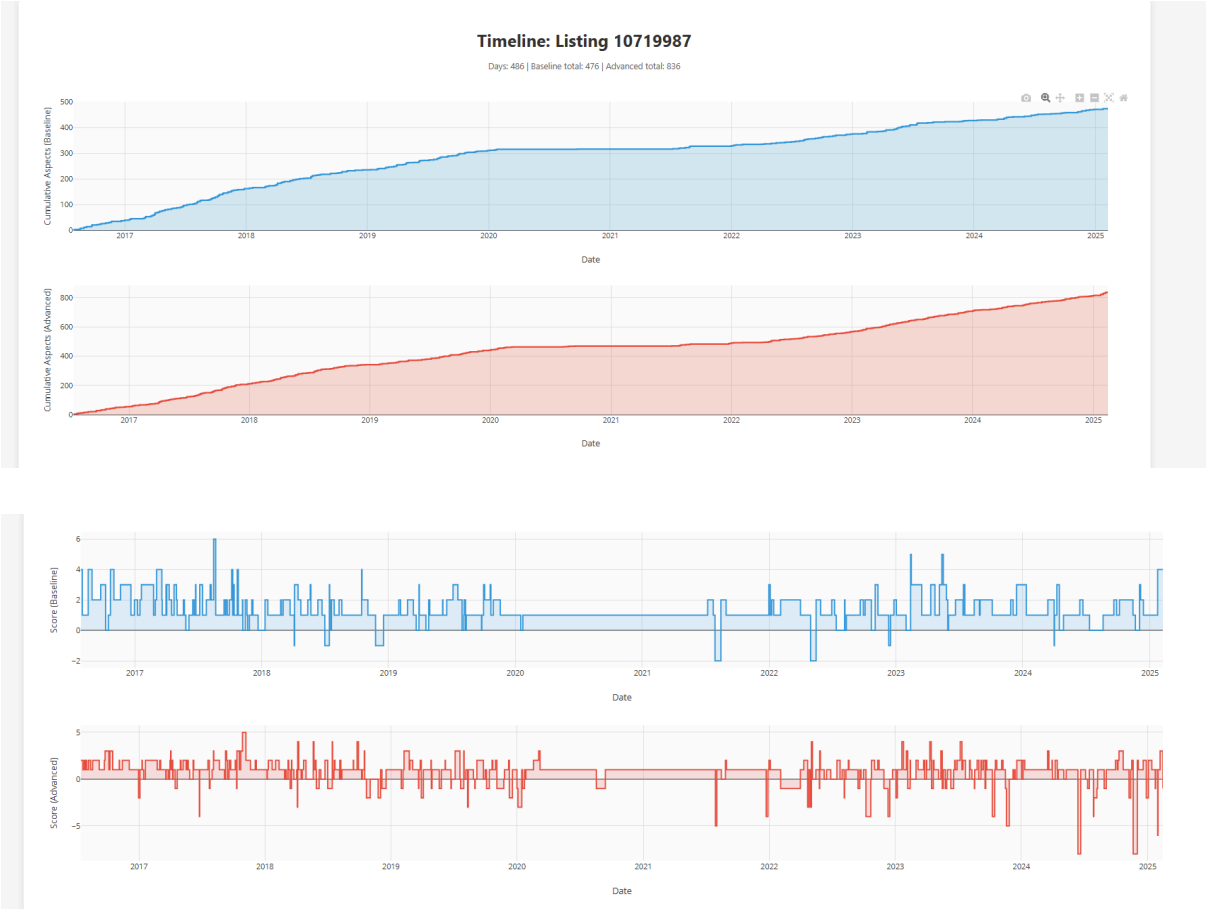
Odpowiedź zawiera szczegółowe statystki dla top k (tutaj 3) aspektów wraz z ich dokładną oceną oraz statystykami.

Zaimplemnowano również dodatek w postaci możliwości przeglądania wykresów dla obu modeli dla każdego lokalu. W odpowiedzi pojawia się

"chart_url": "/predict/chart?listing_id=10719987",

(jest to opcjonalne, dodatkowe porównanie, które nie spełnia przezroczystości celowo)

Po wejściu widzimy wykres score oraz liczby opinii w czasie dla obydwu modeli. Pozwala to sprawdzić, jak ocena dla danego listing_id ewoluuje w czasie.




Mikroserwis oparty jest o Flask i składa się z:

- warstwy API (app.py),
- modułu A/B testingu (przypisywanie wariantów, logowanie),
- menedżera modeli (odczyt wyników z plików CSV).

Modele nie są liczone online, bo serwis działa na przetworzonych danych, co zapewnia bardzo krótki czas odpowiedzi.

### Eksperyment A/B

Zaimplementowano pełny mechanizm A/B testingu, warinat A to model podstawowy ( model_baseline.csv), warinat B to model docelowy: (model_advanced2.csv)

Każde wywołanie endpointu /predict:

1. automatycznie przypisuje wariant,
2. generuje odpowiedź,
3. zapisuje interakcję do logów.

Dodatkowo każda interakcja zapisywana jest do pliku ab_log.csv i zawiera:

- timestamp,
- listing_id,
- wariant (A/B),
- zwrócone aspekty (top i bottom),
- wartości score.

Stworzono róznież skrypt (generate_ab_data.py), który realizuje eksperyment A/B poprzez:

- wczytywanie dostępnych listing_id z obu modeli,
  losuje do 50 identyfikatorów,
- dla każdego wysyła żądanie POST /predict,
- generuje wpisy w ab_log.csv.
- Następnie dane są te podsumowywane w skrypcie (ab_evaluation.ipynb)

# Podsumowanie

W projekcie uwało się stworzyć 2 modele, któe pozwalają na automatyczna analize recenzji tekstowych w celu identyfikacji najczęściej chwalonych i krytykowanych aspektów pobytu. Oba modele bardzo dobrze poradziły sobie z oceną sentymentu (oba modele >95%) oraz w miarę dobrze z grupowanie aspektów - przy czym modele docelowy poradził sobie znacznie lepiej (ma dużo więcej predykcji, które są co najmniej częściowo trafne). Dla obu modeli trzeba było niestety stworzyć również grupę overall (było bardzo dużo opinii, które ciężko było pogrupować). Jednakże przy ilości danych jakie mieliśmy, nawet ignorując tą grupę podstało 12 konkretnych aspektów, które można wykorzystać w celach biznesowych.

W tworzeniu modeli najtrudniejsze okazała się cześć grupowania, dla której próbowano wiele metod. Mimo to wiele razy trzeba było iterować się spowrotem przy testach, które nie dawały satysfakcjonujących wyników.

Zaimplementowano również mikroserwis, który pozwala na klientowi otrzywać JSON'a z top chwalonymi i najgorzej ocenamymi aspektami, działającego w formie mechanizmu A/B dla obu modeli. Dodatkowo dodano wykres ilości recenzji i oceny w czasie, aby klient mógł ocenić czy aplikacja pozytywnie wpłynęła na oceny.