Wprowadzenie do Przetwarzania Języka Naturalnego

Sprawozdanie z etapu pierwszego

Temat projektu: Przewidywanie oceny na podstawie opinii tekstowej

Mateusz Krakowski

Bartosz Latosek

Mikołaj Bańkowski

19 marca 2024

Spis treści

1 Opis Projektu

1.1 Cel projektu

Celem projektu jest stworzeniu modelu klasyfikacji, który będzie w stanie automatycznie przypisywać liczbę gwiazdek hotelowi, na podstawie analizy recenzji, jaką hotel otrzymał. W tym celu wykorzystamy techniki przetwarzania języka naturalnego (Natural Language Processing) oraz uczenia maszynowego (Machine Learning).

1.2 Założenia projektowe

- Procesowi przetwarzania języka naturalnego, będą poddawane recenzje napisane w języku angielskim,
- Dane mogą pochodzić z różnych platform recenzenckich i zasobów online,
- Testowaniu zostaną poddane trzy różne modele oraz model naiwny, uzyskane wyniki zostaną porównane,
- Projekt realizowany w języku Python

1.3 Kamienie milowe

Na potrzebę projektu niezbędne jest, aby wykonać następujące zadania:

- Zgromadzenie danych zawierających recenzje hotelowe wraz z przypisanymi do nich ocenami w postaci liczby gwiazdek,
- Przygotowanie zbioru danych poprzez poddanie ich procesowi oczyszczania i przetwarzania wstępnego,
- Wybór i implementacja narzędzi lub algorytmów do przetwarzania języka naturalnego,
- Budowa modelu klasyfikacji, który zostanie wytrenowany na podstawie przetworzonych danych,
- Testowanie i ocena modelu za pomocą metryk takich jak dokładność (accuracy), precyzja (precision), czułość (recall) oraz F1-score,

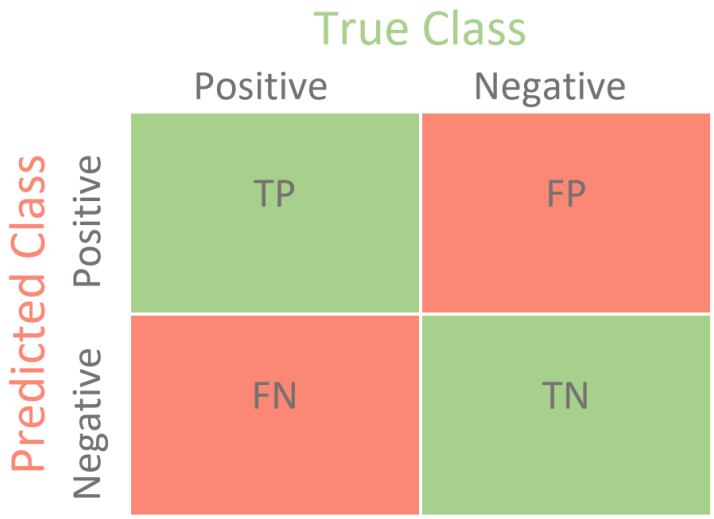
1.4 Zbiory Danych

Na potrzeby projektu przeszukaliśmy publicznie dostępne zbiory danych zawierające recenzje hoteli. Zbiór danych - Hotel Reviews [1] pochodzi z zasobów online dostępnych na platformie Kaggle. W zbiorze danym znajduję się około 50 milionów recenzji, natomiast na potrzeby projektu, będziemy bazować na 50 tys. recenzji. Kluczowymi polami dla nas będzie pole 'rating' zawierającą ocenę numeryczną oraz pole 'text' zawierającą ocenę pisemną hotelu.

1.5 Miary jakości modelu

1.5.1 Macierz pomyłek

Macierz zawierająca 4 wartości mówiące o tym jak model poradził sobie z klasyfikacją danych



Rysunek 1: Macierz Pomyłek

Na podstawie macierzy pomyłek obliczane poniższe miary jakości.

1.5.2 Accuracy

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP}$$

Accuracy niesie informację o tym, jaki procent próbek testowych został poprawnie sklasyfikowany przez model.

1.5.3 Recall(Sensitivity)

$$Recall = Sensitivity = \frac{TP}{TP + FN}$$

Recall mówi nam, jak model radzi sobie z klasyfikowaniem przypadków pozytywnych danej klasy.

1.5.4 Specificity

$$Specificity = \frac{TN}{TN + FP}$$

Specificity mówi nam, jak model radzi sobie z klasyfikowaniem przypadków negatywnych danej klasy.

1.5.5 Precision

$$Precision = \frac{TP}{TP + FP}$$

Precision mówi nam, w jakich proporcjach model klasyfikuje próbki jako pozytywne w zależności od faktycznej klasy próbki.

1.5.6 Miara F1

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall}$$

F1 Score mówi nam, jak dobrze model radzi sobie z klasyfikacją przypadków TP, przy tym minimalizując liczbę przypadków FP i FN.

1.5.7 Support

$$Support = TP + FN$$

Support mówi nam ile w danych ewaluacyjnych jest przypadków danej klasy.

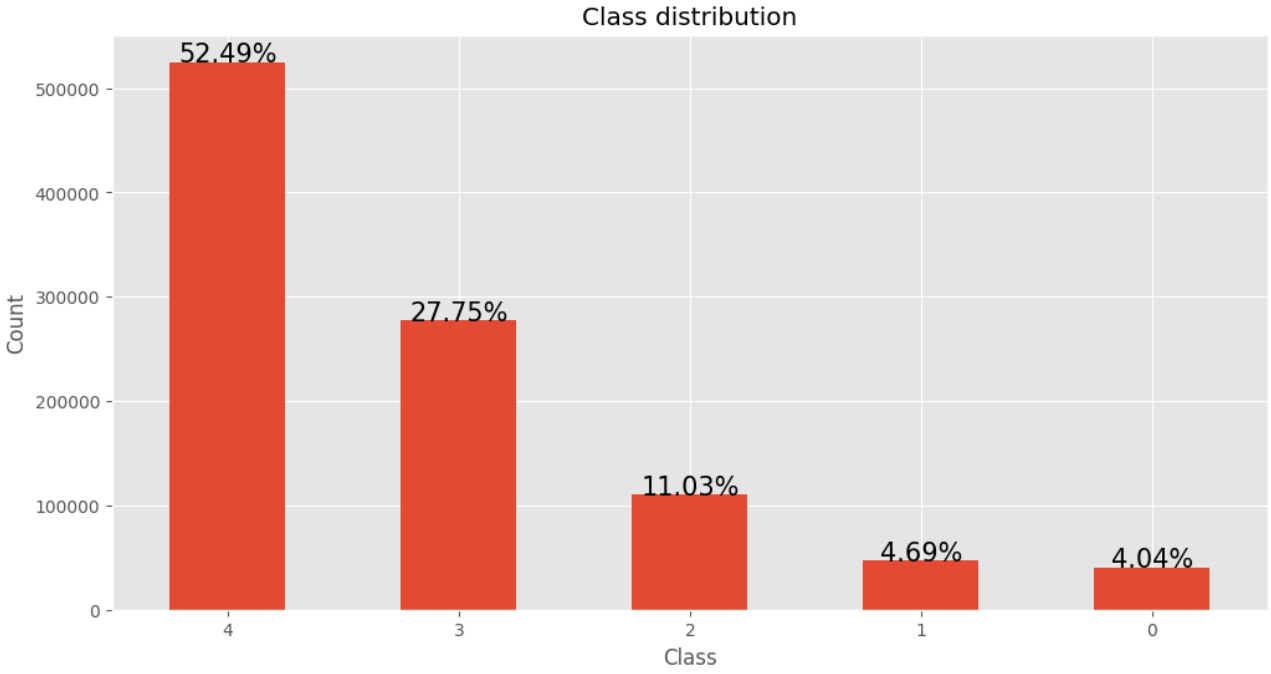
1.6 Założenia odnośnie metryk oceny

Accuracy mimo że mówi nam o celności algorytmu, nie niesie za sobą wystarczającej informacji o jakości modelu. Ważne będzie, aby przy ocenie wziąć pod uwagę Recall oraz Specificity. To, aby ustalić która z tych dwóch metryk jest ważniejsza, musi odpowiedzieć pytanie czy jesteśmy bardziej skłonni dopuścić do klasyfikacji przypadków fałszywie negatywnych, czy fałszywie pozytywnych. Dodatkowo dobrą miarą do porównania modeli będzie miara F1.

2 Implementacja

2.1 Analiza danych

Sekcja ta została stworzona na podstawie pliku data_analysis.ipynb, który znajduje się w katalogu \data\raw. Funkcje odpowiedzialne za przygotowanie danych zostały zaimplementowane w pliku src\data\make_dataset.py.



Rysunek 2: Dystrybucja klas

Mamy do czynienia ze zbiorem 1 miliona rekordów, rekordy składają się z opinii tekstowej oraz oceny dyskretnej w skali 0-4. Każda z klas oceny oznacza liczbę gwiazdek którą otrzymał hotel. Wyliczenie liczby gwiazdek połączonej do danej opinii tekstowej liczymy ze wzoru wartość + 1. Czyli na przykład opinia o klasie z wartością 4 oznacza opinię pięciogwiazdkową.

W zbiorze doszukaliśmy się 38 rekordów, które były duplikatami, pozbyliśmy się ich. Nie odnaleźliśmy żadnych wartości, które oznaczałyby brak opinii tekstowej.

Występują znaczne dysproporcje w kontekście dystrybucji atrybutu klasy. Klasa większościową, którą jest klasa reprezentująca opinię pięciogwiazdkową stanowi ponad 50% całego zbioru. Jest to zarazem plus jak i minus. Z uwagi na tą dysproporcję sensownym pomysłem będzie jako base-line projektu przyjąć model, który jako predykcję za każdym razem zwraca klasę większościową. Możemy się spodziewać, że statystycznie taki model będzie posiadał celność około 52.49% dla zbioru testowego. Dla modeli bardziej skomplikowanych, takich jak transformer RoBERTa, skutki tej dysproporcji muszą zostać zniwelowane poprzez zastosowanie warzenia klas w trakcie nauki. Wprowadzenie tego mechanizmu powinny zmniejszyć przeuczanie się modelu.

2.2 Wykorzystane modele

2.2.1 Model nawiwny

Model naiwny zwraca zawsze klasę której liczba była największa przy zadanym zbiorze uczącym. Z uwagi na to, że grupa większościowa w przyjętym zbiorze danych stanowi 45% zbioru, spodziewamy się 100% miary Accuracy dla klasy oznaczającej opinię 5 gwiazdkową i 0% Accuracy dla pozostałych. Można było również przyjąć model zwracający zawsze wartość losową, ale zdecydowaliśmy się użyć model zwracający klasę większościową po to aby postawić poprzeczkę wyżej dla pozostałych modeli.

Zalety:

- Dobry baseline do porównywania jakości innych modeli.

Wady:

- Oczywista, niska jakość klasyfikatora.

2.2.2 Model transformer NLP RoBERTa

RoBERTa[2], czyli "Robustly Optimized BERT Approach", to zaawansowany model przetwarzania języka naturalnego (NLP) oparty na architekturze Transformer. RoBERTa jest modelem klasyfikacji, który wykorzystuje zaawansowane metody analizy cech do przypisania nowych danych tekstowych do odpowiednich kategorii lub zadań.

Działanie RoBERTa opiera się na przetwarzaniu sekwencji słów w celu zrozumienia kontekstu i znaczenia tekstu. W praktyce, RoBERTa dzięki mechanizmowi atencji[3] analizuje kontekst globalny zdania, uwzględniając relacje między słowami.

RoBERTa korzysta z zaawansowanych technik embeddingów BERT (Bidirectional Encoder Representations from Transformers), które przekształcają słowa na wektory numeryczne, reprezentujące ich semantykę i znaczenie. Te embeddingi są wykorzystywane przez model do uczenia się cech charakterystycznych dla różnych klas lub zadań NLP.

Aby przypisać etykietę lub wykonać inne zadanie NLP, RoBERTa korzysta z ogromnej ilości danych treningowych, w których słowa są powiązane z odpowiednimi etykietami lub informacjami. Podczas procesu uczenia RoBERTa analizuje te dane treningowe, aby nauczyć się, jakie cechy są charakterystyczne dla danej klasy lub zadania, oraz jak przypisać nowe dane tekstowe do tych klas.

Zalety:

- RoBERTa oferuje znacznie lepsze zrozumienie kontekstu i semantyki tekstu dzięki wykorzystaniu architektury Transformer, co przekłada się na wyższą jakość predykcji w złożonych zadaniach NLP.
- Dzięki swojej głębokiej architekturze i trenowaniu na dużych zbiorach danych, RoBERTa może osiągać lepsze wyniki w zadaniach przetwarzania języka naturalnego w porównaniu do tradycyjnych modeli NLP.
- RoBERTa jest skalowalna i może być trenowana na bardzo dużych zbiorach danych, co pozwala na jej adaptację do różnorodnych zastosowań i dziedzin.

Wady:

- RoBERTa wymaga dużej ilości zasobów obliczeniowych i czasu trenowania ze względu na swoją złożoną architekturę i dużą liczbę parametrów, co może być ograniczeniem w przypadku braku procesorów z rdzeniami CUDA.
- Ze względu na swoją skomplikowaną naturę, RoBERTa może być trudna do interpretacji, co może stanowić problem w przypadku zastosowań, gdzie ważna jest transparentność działania modelu.
- Istnieje ryzyko nadmiernego dopasowania (overfitting) do danych treningowych ze względu na dużą pojemność modelu, co może prowadzić do obniżenia wydajności w przypadku nowych, nieznanych danych.

3 Literatura

[1] Diego Antognini and Boi Faltings. Hotelrec: a novel very large-scale hotel recommendation dataset. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 4917–4923, Marseille, France, May 2020. European Language Resources Association.

[2] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019.

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.