GRUPA:

Rafał Zaborowski
Konrad Pawlenko
Andrzej Zamora
Hubert Waśniewski

Zadanie

W wybranym języku zaimplementuj prosty niespersonalizowany system rekomendacji, który wykorzystuje wybrany algorytm z dziedziny data mining (np. grupowanie, klasyfikacja, analiza regresji, reguły asocjacyjne, i in.).

Zostaliśmy przypisani do algorytmu klasyfikacja

Nasz algorytm działa w następujący sposób:

1. Wczytuję dane
2. Wybieramy osobę dla której będą obliczane rekomendacje
3. Na podstawie KNN dostajemy użytkowników którzy są najbardziej podobni do wybranej osoby (ilość użytkowników zależy od zmiennej ‘`neighbors_count`’)

```
# Wczytanie danych z pliku CSV
data = pd.read_csv('plik.csv')

X = data.drop(columns=['Unnamed: 0'])
X = X.applymap(lambda x: x.strip() if isinstance(x, str) else x)
X = X.replace('x', 0)

# Zamiana wartości na numeryczne
X = X.apply(pd.to_numeric)

knn_model = NearestNeighbors(n_neighbors=2, metric='euclidean')
knn_model.fit(X)

# Przykładowa osoba dla której chcemy uzyskać rekomendacje
person_index = 0
sample_person = X.iloc[person_index].values.reshape(1, -1)

# Znalezienie najbliższych sąsiadów
neighbors_count = 4
distances, indices = knn_model.kneighbors(sample_person, neighbors_count + 1)

new_data = data.iloc[indices[0]].copy()
```

4. Następnie obliczamy czy daną potrawę warto polecić na podstawie podobnych do nas osób

Algorytm działa w taki sposób:

Jeżeli danie zostało ocenione na większą ocenę niż 4 przez osobę dla której szukamy rekomendacje, dany produkt polecamy, jeżeli oceniała na mniejszą ocenę niż 4 to nie rekomendujemy dany produkt, natomiast jeżeli nie oceniła danego produktu to jest liczona średnia ocen danego produktu od podobnych użytkowników, i jeżeli jest większa ta srednia niż 4 to polecamy. Jeżeli żadna z osób nie oceniła danego produktu wtedy dajemy informacje “BRAK_DANYCH”

5. Wynik algorytmu zapisujemy do csv

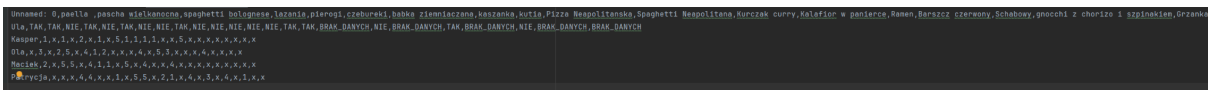
```
column_count = new_data.shape[1]
for column_index in range(column_count):
    if column_index == 0:
        continue

    column = new_data.iloc[:, column_index]
    if pd.to_numeric(column.iloc[0], errors='coerce') >= 4:
        new_data.iloc[0, column_index] = 'TAK'
    elif pd.to_numeric(column.iloc[0], errors='coerce') < 4:
        new_data.iloc[0, column_index] = 'NIE'
    else:
        number_of_people_rating = 0
        rating_sum = 0
        for neighbor_index in range(neighbors_count):
            rating = pd.to_numeric(column.iloc[neighbor_index + 1], errors='coerce')
            if not pd.isnull(rating):
                number_of_people_rating += 1
                rating_sum += rating

        if number_of_people_rating == 0:
            new_data.iloc[0, column_index] = 'BRAK_DANYCH'
        elif rating_sum / number_of_people_rating >= 4:
            new_data.iloc[0, column_index] = 'TAK'
        else:
            new_data.iloc[0, column_index] = 'NIE'

new_data.to_csv('wyniki_rekomendacji.csv', sep=',', index=False)
```

CSV:



Wnioski

Algorytm ten jest dość prosty i ma problemy z rekomendacjami gdy użytkownicy o podobnych upodobaniach nie ocenili takich produktów jak osoba dla której szukamy

rekomendacji. Dodatkowo warto posiadać dość duży zbiór aby skuteczniej odnajdywać osoby podobne. A także sam algorytm nie jest idealny ponieważ skuteczność rekomendacji to około 60% w naszym przypadku. Wykluczając produkty z rezultatem “BRAK_DANYCH”