

# Studio per la costruzione di un recommender system per lo YELP Open Dataset

Corso di Laurea Magistrale in Informatica

Anno Accademico 2021/2022

Nome del gruppo: V.D.F.

Componenti del gruppo

Vallasciani Giacomo - 0000954881

Di Maria Giuseppe - 0000954089

Fabbri Alessandro - 0000934261

# Contents

<b>1</b>	<b>Introduzione</b>	<b>3</b>
1.1	Descrizione del problema . . . . .	3
1.2	Descrizione della soluzione proposta . . . . .	4
1.3	Suddivisione del lavoro . . . . .	5
<b>2</b>	<b>Metodo proposto</b>	<b>5</b>
<b>3</b>	<b>Risultati sperimentali</b>	<b>9</b>
3.1	Istruzioni per l'esecuzione . . . . .	9
3.2	Tecnologie utilizzate . . . . .	13
3.3	Misurazione delle performance . . . . .	14
3.4	Risultati della configurazione migliore . . . . .	16
3.4.1	Classificazione di una recensione . . . . .	16
3.4.2	Raggruppamento degli utenti in base al comportamento sulla piattaforma . . . . .	17
3.4.3	Raggruppamento dei ristoranti di una città in base a caratteristiche simili . . . . .	21
3.5	Comparazione dei risultati ottenuti . . . . .	22
3.5.1	Classificazione di una recensione come positiva o negativa	22
3.5.2	Raggruppamento degli utenti in base al comportamento sulla piattaforma . . . . .	23
3.5.3	Raggruppamento dei ristoranti in base a città e carat- teristiche . . . . .	25
3.6	Classificazione di una recensione da 1 a 5 stelle . . . . .	27
<b>4</b>	<b>Discussioni e conclusione</b>	<b>29</b>
4.1	Risultati ottenuti in termini di performance . . . . .	29
4.1.1	Classificazione di una recensione come positiva o negativa	29
4.1.2	Raggruppamento degli utenti in base al comportamento sulla piattaforma . . . . .	30
4.1.3	Raggruppamento dei ristoranti in base a città e carat- teristiche simili . . . . .	31
4.2	Classificazione di una recensione da 1 a 5 stelle . . . . .	32
4.3	Confronto con le aspettative e con la letteratura . . . . .	33
4.3.1	Classificazione di una recensione come positiva o negativa	33
4.3.2	Raggruppamento degli utenti in base al comportamento sulla piattaforma . . . . .	33
4.3.3	Raggruppamento dei ristoranti di una città in base a caratteristiche simili . . . . .	33

4.3.4	Classificazione di una recenziona da 1 a 5 stelle . . . .	34
4.4	Limiti della soluzione proposta . . . . .	34
4.5	Lavori futuri . . . . .	35

# 1 Introduzione

## 1.1 Descrizione del problema

Si vuole sviluppare un **recommender system** per la piattaforma YELP utilizzando come dati da analizzare lo **YELP Open Dataset**, ovvero una porzione dell'intero dataset di YELP che l'azienda stessa mette a disposizione al pubblico.

Tale sistema dovrà essere in grado di:

- Consigliare ad un utente la valutazione di una recensione;
- Mostrare ad un utente gli utenti presenti sulla piattaforma divisi in base al loro comportamento (utile per giudicare l'affidabilità delle recensioni dei vari utenti);
- Consigliare ad un utente dei ristoranti in base alla città e alle features scelte.

Un **recommender system** è un software di filtraggio dei contenuti che crea delle raccomandazioni personalizzate specifiche per l'utente così da aiutarlo nelle sue scelte. Viene utilizzato per diversi prodotti, come libri, musica, film, video, notizie e social media.

Per la costruzione di tale sistema solitamente si seguono i seguenti approcci:

- **Collaborative filtering approach:** si creano dei suggerimenti utilizzando la similarità tra gli utenti;
- **Content-based approach:** suggerimenti creati utilizzando i contenuti degli elementi presenti nel dataset, un elemento è costituito dalla sua descrizione, attributi, parole chiave e etichette. Si usa spesso un **classificatore Bayesiano**.
- **Hybrid approach:** combinazione dei due approcci sopra descritti.

Per la costruzione del recommender system sono stati presi in considerazione i seguenti problemi:

1. Classificazione di una recensione come **positiva** o **negativa** (Classificazione binaria).
2. Raggruppamento degli **utenti** in base al loro **comportamento** sulla piattaforma (Clustering).
3. Raggruppamento dei **ristoranti** di una specifica città in base a **caratteristiche simili** (Clustering).

Per il problema numero 1 si è scelto poi di raffinarlo per permettere al recommender system di consigliare, oltre al fatto di essere positiva o negativa, una **valutazione in stelle** della recensione inserita. Trasformandolo così da un problema di classificazione binaria ad un problema di classificazione n-aria (in questo caso valutazioni da 1 a 5 stelle).

## 1.2 Descrizione della soluzione proposta

Iniziamo descrivendo i diversi tipi di approccio utilizzati per ognuno dei problemi:

1. Classificazione: classificazione di una recensione se **positiva** o **negativa**, rating (**in termini di stelle**) - **Content-based**;
2. Raggruppamento degli **utenti** in base al loro comportamento sulla piattaforma - **Collaborative filtering**;
3. Raggruppamento dei **ristoranti** in una specifica città in base a caratteristiche simili - **Hybrid approach**.

Per la ricerca della migliore soluzione per ognuno dei problemi sopra elencati è stato eseguito uno studio tra metodi di classificazione binaria prettamente statistici e tecniche inerenti all'ambito del **NLP** (prettamente per la classificazione, per i problemi di raggruppamento sono stati studiati algoritmi di clustering).

In merito al problema della **classificazione binaria** sono stati presi in esame Logistic Regression, Random Forest [5] e XGBoost [3] ed in seguito sono stati aggiunti al confronto Neural Network e Transformer, come vedremo tutti gli algoritmi presentati mostrano delle ottime performance per la risoluzione di questo problema.

Riguardo ai problemi di **raggruppamento** degli utenti e dei ristoranti invece la scelta è ricaduta sull'utilizzo di algoritmi di clustering, in quanto sono più utili per questo scopo e lavorano anche con set di dati non supervisionati, e gli algoritmi messi a confronto sono DBSCAN [2], KMEANS [7] e KNN.

Per il problema di **classificazione n-aria** invece sono stati messi a confronto il Naive Bayes Classification Algorithm [6], Word2Vec Neural Network [4] e BERT Transformer [1].

Dai risultati presentati in seguito si nota che:

- **Classificazione binaria:** la tecnica che meglio risolve il problema è la **Neural Network**;
- **Classificazione n-aria:** la tecnica che meglio risolve il problema è il **Transformer**.

- **Raggruppamento utenti:** in questo caso non si può determinare quale delle tecniche mostrate sia la migliore in quanto sia DBSCAN che KMEANS hanno dei **Silhouette scores** molto simili, la scelta ricade sulle esigenze dell'implementatore.
- **Raggruppamento ristoranti:** il miglior raggruppamento viene effettuato da KMEANS che ha il **Silhouette score** più alto.

### 1.3 Suddivisione del lavoro

Il Lavoro è stato suddiviso nel seguente modo:

- **Giacomo:**
  1. Supervisione del progetto;
  2. Implementazione riconoscimento recensione positiva o negativa;
  3. Implementazione valutazione in stelle di una recensione;
  4. Stesura relazione;
  5. Code refactoring;
- **Giuseppe:**
  1. Implementazione raggruppamento ristoranti in base a caratteristiche simili;
  2. Implementazione demo;
  3. Stesura relazione;
- **Alessandro**
  1. Implementazione raggruppamento utenti in base al comportamento sulla piattaforma;
  2. Implementazione demo;
  3. Stesura relazione;

## 2 Metodo proposto

Per ogni problema analizzato prima sono stati eseguiti tutti gli algoritmi presentati, in seguito sono stati messi a confronto e si sceglie quello che si rivela migliore in termini di performance. Di seguito verranno descritti brevemente tutti gli algoritmi usati. Come già anticipato, per problema di **classificazione binaria** sono stati messi a confronto i seguenti algoritmi:

- **Logistic Regression** [5]: modello lineare, i modelli lineari sono composti da una o più variabili indipendenti che descrivono una relazione con una variabile di risposta dipendente. La mappatura delle caratteristiche di input qualitative o quantitative su una variabile target che si tenta di prevedere come dati finanziari, biologici o sociologici è nota come apprendimento supervisionato se le etichette sono note. Uno dei modelli statistici lineari più comuni utilizzati per l'analisi discriminante è la regressione logistica;
- **Random Forest** [5]: è un algoritmo di apprendimento basato su un insieme che comprende  $n$  raccolte di alberi decisionali non correlati. Si basa sull'idea dell'aggregazione bootstrap, che è un metodo per il ricampionamento con sostituzione al fine di ridurre la varianza. Random Forest utilizza più alberi per calcolare la media (regressione) o calcolare i voti di maggioranza (classificazione) nei nodi foglia terminali quando si effettua una previsione. Basati sull'idea degli alberi decisionali, i modelli random forest hanno portato a miglioramenti significativi nell'accuratezza delle previsioni rispetto a un singolo albero aumentando il numero di alberi; ogni albero nel set di addestramento viene campionato in modo casuale senza sostituzione. Gli alberi decisionali consistono semplicemente in una struttura ad albero in cui il nodo superiore è considerato la radice dell'albero che è diviso ricorsivamente in una serie di nodi decisionali dalla radice fino al raggiungimento del nodo terminale o del nodo decisionale.
- **XGBoost** [3]: XGBoost è un software avanzato basato su Gradient Tree Boosting in grado di gestire in modo efficiente attività di Machine Learning su larga scala. Per la sua superiorità delle prestazioni e tempo di calcolo e complessità in memoria, è stato ampiamente applicato a una varietà di campi di ricerca, che vanno dalla diagnosi del cancro, analisi della cartella clinica alla valutazione del rischio di credito e metagenomica. Inoltre, grazie alla sua interfaccia Python facile da usare e alla sua natura facilmente spiegabile, è diventato di fatto il metodo di prima scelta per la maggior parte dei problemi di data science.  
Tuttavia le prestazioni di questo modello risultano calare in presenza di dati sbilanciati.
- **Word2Vec Neural Network** [4]: Word2vec comprende e vettorizza il significato delle parole in un documento basandosi sull'ipotesi che parole con significati simili in un dato contesto presentino distanze ravvicinate.

La CNN è una rete neurale utile per estrarre e classificare le features perché può passare valori al livello successivo senza perdere informazioni spaziali.

- **BERT Transformer** [1]: Bidirectional Encoder Representations from Transformers (BERT), è una tecnica di apprendimento automatico basata su Transformer per il pre-training sull'elaborazione del linguaggio naturale (NLP), sviluppata da Google. BERT è stato creato e pubblicato nel 2018 da Jacob Devlin e dai suoi colleghi di Google.

I risultati del confronto indicano che tutti i modelli lavorano abbastanza bene, tuttavia in particolare spiccano per accuracy la **Word2Vec NN** e il **Transformer**, quest'ultimo però è stato omesso dai risultati per questo task in quanto presenta alcune problematiche in fase di test.

Per il problema di raggruppamento degli **utenti** e dei **ristoranti** sono state messe a confronto diverse tecniche di clustering:

- Raggruppamento degli utenti in base al comportamento sulla piattaforma:
  1. **DBSCAN** [2]: DBSCAN è un algoritmo di clustering basato sulla densità. Tale algoritmo ha la caratteristica di trovare qualsiasi classe di forma nel set di dati.  
L'algoritmo DBSCAN è insensibile ai punti di rumore. Può identificare i punti di disturbo ed escluderli dai risultati del clustering. Per ogni classe nel risultato del raggruppamento, la densità all'interno della classe è maggiore della densità ai margini della classe. La densità del punto di disturbo è inferiore a quella del bordo. In base alle caratteristiche di distribuzione dei dati, l'algoritmo utilizza la differenza di densità per identificare diverse regioni di densità e contrassegna i risultati del clustering.
  2. **KMEANS** [7]: k-means clustering è un metodo di quantizzazione vettoriale, originariamente utilizzato per l'elaborazione dei segnali, che mira a partizionare n osservazioni in k cluster in cui ogni osservazione appartiene al cluster con la media più vicina (centri del cluster o centroidi del cluster), fungendo da prototipo per il cluster.
- Raggruppamento dei ristoranti per città e caratteristiche:
  1. **KNN**: approccio di clustering basato sui grafi. KNN viene utilizzato per identificare i k punti più simili attorno a ciascun punto e mediante la fusione condizionale vengono generati cluster.



2. **DBSCAN** [2];
3. **KMEANS** [7].

Per il task aggiuntivo di **classificazione n-aria** sono stati prese in considerazione le seguenti tecniche:

1. **Naive Bayes Classification Algorithm** [6]: Classificatore lineare noto per essere semplice ma molto efficiente. Il modello probabilistico dei classificatori Naive Bayes si basa sul teorema di Bayes e l'aggettivo Naive deriva dal presupposto che le caratteristiche in un set di dati siano mutuamente indipendenti. In pratica, il presupposto dell'indipendenza viene spesso violato, ma i classificatori Naive Bayes tendono ancora a funzionare molto bene sotto questo presupposto irrealistico. Soprattutto per campioni di piccole dimensioni, i classificatori Naive Bayes possono superare le alternative più potenti.
2. **Word2Vec Neural Network** [4];
3. **BERT Transformer** [1].

Per questo task invece, rispetto a quello di **classificazione binaria**, il modello che lavora meglio è proprio il **Transformer**.

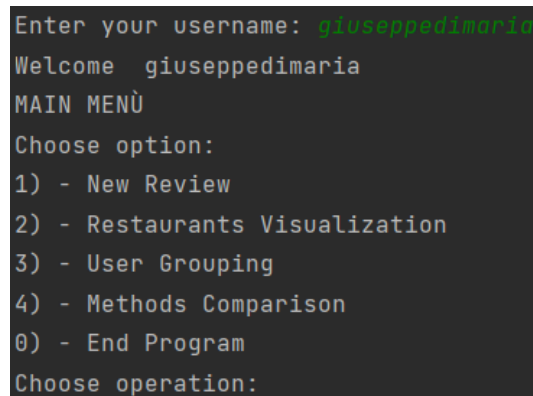
La scelta di tali implementazioni è ricaduta sul fatto che si è voluto effettuare uno studio tra le alternative più utilizzate per risolvere problemi simili e confrontarne poi le performance.

In particolare per i problemi di classificazione sono state scelte le alternative dalle più “semplici” a quelle più “complesse” e che sono di ampio interesse attualmente. Infine, per ogni problema, verrà scelta la configurazione che otterrà i risultati e i punteggi (descritti nel capitolo 3.3) migliori.

## 3 Risultati sperimentali

### 3.1 Istruzioni per l'esecuzione

1. Installare tutte le librerie elencate in 3.2;
2. Installare anche le seguenti:
  - `pip install -U kaleido;`
  - `pip install pyyaml h5py;`
  - `pip install imblearn;`
  - `pip install prettytable.`
3. Clonare il repository da GitLab dal seguente link: [https://gitlab.com/Scianso/ai\\_vdf](https://gitlab.com/Scianso/ai_vdf);
4. Aprire il terminale nella directory “ai\_vdf”;
5. Eseguire **python demo.py** per avviare la demo;
6. Una volta avviata la demo verrà richiesto l'inserimento di uno username ed in seguito verrà visualizzato il menù con le operazioni disponibili.



```
Enter your username: giuseppedimarìa
Welcome giuseppedimarìa
MAIN MENU
Choose option:
1) - New Review
2) - Restaurants Visualization
3) - User Grouping
4) - Methods Comparison
0) - End Program
Choose operation:
```

Figure 1: Inserimento username e elenco azioni disponibili

- 1 - Si procederà con la scelta di una città, poi di un ristorante per il quale si dovrà lasciare una recensione che verrà valutata dal sistema e verrà suggerita una valutazione in stelle.

```

Choose operation: 1
NEW REVIEW
0      Santa Barbara
1      Affton
2      Tucson
3      Philadelphia
4      Brentwood
...
9968    Brentwood
9969      Reno
9970    Saint Louis
9971    Indianapolis
9972    Philadelphia
Name: city, Length: 9973, dtype: object

Choose a city from those listed: Reno

```

Figure 2: Lista delle città

```

Choose a city from those listed: Reno

List of Restaurant in  Reno
15      Romano's Macaroni Grill
123      Reno Bighorns
128      Reno Family Chiropactic
159      Daniel Colombo , MD
168      The UPS Store
...
9769      Pitch Black Printing
9775      Green's Feed
9818      Wendy's
9841      Blaze Pizza
9969    Sierra Silver & Gold Buyers
Name: name, Length: 378, dtype: object
Choose a restaurant in  Reno
Enter the name of the restaurant: Blaze Pizza
You chose,  Blaze Pizza

```

Figure 3: Lista dei ristoranti

```

Write your review:
I loved this Place! The Service was Excellent!
Prediction on review
Review is : positive

Suggested stars
Your review should have ★★★★★ star/stars

```

Figure 4: Output del sistema

- 2 - Si richiederà all'utente di scegliere una città, scegliere un numero arbitrario di caratteristiche dei ristoranti tra quelle presenti nella lista mostrata ed infine verrà restituita una lista di ristoranti con caratteristiche simili a quelle scelte.

```

Choose operation: 2
RESTAURANTS VISUALIZATION

BUSINESS AND CITY
0      Santa Barbara
1      Affton
2      Tucson
3      Philadelphia
4      Brentwood
...
9968   Brentwood
9969   Reno
9970   Saint Louis
9971   Indianapolis
9972   Philadelphia

```

Figure 5: Scelta della città

```

Choose a city from those listed: rome
Select Features
-----+-----
| Index |          FEATURES          |
-----+-----
| 0 | RestaurantsGoodForGroups_False |
| 1 | RestaurantsGoodForGroups_True  |
| 2 | RestaurantsTakeOut_False       |
| 3 | RestaurantsTakeOut_None        |
| 4 | RestaurantsTakeOut_True        |
| 5 | RestaurantsAttire_'casual'      |
| 6 | RestaurantsAttire_'dressy'     |
| 7 | RestaurantsAttire_'formal'     |
| 8 | RestaurantsAttire_u_'casual'   |
| 9 | RestaurantsAttire_u_'dressy'   |
| 10 | RestaurantsAttire_u_'formal'   |

```

Figure 6: Scelta delle caratteristiche

```

Choose features from those listed: 1,2,55,120,222,300,425

Restaurant similar with respect to chosen features
      name  stars
0      La Orquidea      3
1  Kased Brothers Halal      4
2      Shake Shack      3
3    Wolly's Kitchen      5
4      Subway      2
5  Perks Coffee & Cafe      5
6    Ba Ta's Cafe      4
7    La Chilanga      4
8      Mondo      3
9  Daisy Dukes Diner      4
10    City Greens      2
11    Chick-Fil-A      3
12    Midway Pizza      2
13      MoPho      3
14      Subway      3
15  Little Caesars      3
16    Folse Market      2
17      KFC      2
18  Cartozzo's Bakery      2
19    Salad Station      3
20    Burger King      2

```

Figure 7: Risultato

- 3 - Verrà subito restituito il raggruppamento in 6 classi diverse degli utenti.
- 4 - Si passerà ad un nuovo menù che permetterà di eseguire il confronto tra le soluzioni prese in considerazione.

```

MAIN MENÙ
Choose option:
1) - New Review
2) - Restaurants Visualization
3) - User Grouping
4) - Methods Comparison
0) - End Program
Choose operation: 4
CHOOSE TASK
1) - Positive or negative review
2) - User grouping
3) - Restaurant grouping
4) - Give stars to a review
5) - Back
0) - End program
Choose operation:

```

Figure 8: Sotto-menù

## 3.2 Tecnologie utilizzate

Di seguito sono riportate le tecnologie utilizzate per il progetto svolto:

- python 3.7.13+
- pandas 1.3.5
- numpy 1.21.6
- matplotlib 3.2.2
- nltk 3.7
- wordcloud 1.5.0
- scikit-learn 1.0
- keras 1.8.0
- scipy 1.7.3

- kneed 0.7.0
- tensorflow 2.8.2
- Transformers 4.20.1

### 3.3 Misurazione delle performance

Di seguito verranno descritte nel dettaglio le metriche scelte per misurare le performance dei modelli studiati. Prima definiremo cosa sono i **TP**, **TN**, **FP**, **FN**.

- **True Positives (TP)**: questi sono i valori positivi predetti correttamente, il che significa che il valore della classe effettiva è sì e anche il valore della classe prevista è sì. Per esempio, se il valore effettivo della classe indica che questo passeggero è sopravvissuto e la classe prevista ti dice la stessa cosa.
- **True Negatives (TN)**: questi sono i valori negativi predetti correttamente, il che significa che il valore della classe effettiva è no e anche il valore della classe prevista è no. Per esempio, se la classe effettiva dice che questo passeggero non è sopravvissuto e la classe prevista ti dice la stessa cosa.
- **False Positives (FP)**: quando la classe effettiva è no e la classe prevista è sì. Per esempio, se la classe effettiva dice che questo passeggero non è sopravvissuto, ma la classe prevista ti dice che questo passeggero sopravviverà.
- **False Negatives (FN)**: quando la classe effettiva è sì ma la classe prevista in no. Per esempio, se il valore effettivo della classe indica che questo passeggero è sopravvissuto e la classe prevista ti dice che il passeggero morirà.

Per i problemi di classificazione le metriche scelte sono:

- **Precision:** la precisione è il rapporto tra le osservazioni positive previste correttamente e il totale delle osservazioni positive previste. La domanda a cui questa metrica risponde è: di tutti i passeggeri etichettati come sopravvissuti, quanti sono effettivamente sopravvissuti? Elevata precisione corrisponde ad un basso tasso di falsi positivi.

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

- **Recall:** è il rapporto tra le osservazioni positive previste correttamente e tutte le osservazioni nella classe effettiva. La domanda a cui questa metrica risponde è: di tutti i passeggeri che sono sopravvissuti veramente, quanti ne abbiamo etichettato?

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

- **F1-score:** il punteggio F1 è la media ponderata di Precisione e Recall. Pertanto, questo punteggio tiene conto sia dei falsi positivi che dei falsi negativi. Intuitivamente non è facile da capire come l'accuratezza, ma F1 è solitamente più utile dell'accuratezza, specialmente se hai una distribuzione di classi irregolare. La precisione funziona meglio se falsi positivi e falsi negativi hanno un costo simile. Se il costo dei falsi positivi e dei falsi negativi è molto diverso, è meglio guardare sia Precisione che Recall.

$$F1Score = \frac{2 \cdot (Recall \cdot Precision)}{Recall + Precision}$$

Per i problemi di raggruppamento è stato valutato il **Silhouette score**.

Il **Silhouette score** è una misura di quanto un oggetto è simile ad oggetti appartenenti al proprio cluster (coesione) rispetto ad oggetti appartenenti ad altri cluster (separazione). La silhouette varia da -1 a +1, dove un valore alto indica che l'oggetto è ben abbinato al proprio cluster e scarsamente abbinato ai cluster vicini. Se la maggior parte degli oggetti ha un valore elevato, la configurazione del cluster è appropriata. Se molti punti hanno un valore basso o negativo, la configurazione del cluster potrebbe avere troppi o troppo pochi cluster.



### 3.4 Risultati della configurazione migliore

Di seguito verranno presentati i risultati delle configurazioni che hanno ottenuto i migliori punteggi in termini di performance per ogni problema descritto.

La configurazione migliore per ogni problema è stata poi implementata nella demo da dove sono stati presi i risultati mostrati sotto.

#### 3.4.1 Classificazione di una recensione

Si vuole permettere all'utente di inserire una recensione, per un ristorante scelto, fornire automaticamente una valutazione della recensione (positiva o negativa) e consigliare una valutazione in stelle.

```
Write your review:
I loved this place! The service was Excellent!!!
Prediction Review

Review is : positive
Stars Review

Your review should have ★★★★★ star/stars
```

Figure 9: Recensione positiva e stelle consigliate

```
Write your review:
Horrible place!
Prediction Review

Review is : negative
Stars Review

Your review should have ★ star/stars
```

Figure 10: Recensione negativa e stelle consigliate

Gli screenshot sopra mostrano due casi di inserimento di una recensione, il

primo mostra l’inserimento di una recensione positiva con relativo suggerimento di valutazione in stelle (dal testo della recensione il risultato sembra anche corretto), il secondo mostra l’inserimento di una recensione negativa.

### 3.4.2 Raggruppamento degli utenti in base al comportamento sulla piattaforma

Si vuole consentire all’utente di visualizzare gli utenti della piattaforma divisi in 6 categorie diverse (numero di categorie che ha mostrato i migliori punteggi di silhouette), che potrebbe aiutare l’utente a decidere la veridicità di una particolare recensione (in base a quale categoria appartiene l’utente che ha scritto la recensione).

Sample of New or not active users					
	name	useful	fans	average_stars	compliment_hot
2	Mike	399	23	3.73	4
3	Rachelle	109	7	4.04	2
4	John	154	4	3.40	0
5	Charlene	63	4	4.51	4
6	Jennifer	29	1	4.15	0
7	Ronskee	56	9	3.84	15
8	AJ	201	4	3.60	1
10	Joz Joz Joz	2063	116	3.93	319
11	Zuzzi	819	37	3.81	46
12	Nina	1944	75	3.41	24
13	Reis	137	2	3.91	0
15	Michelle	439	22	3.63	33
16	Helen	700	49	3.33	25
18	Joanne	150	7	3.54	1
19	Chang	279	4	3.26	1
20	Mike	71	2	3.87	1

Figure 11: Utenti appena registrati o poco attivi

Sample of Users with low number of votes sent (up to 17500)					
	name	useful	fans	average_stars	compliment_hot
14	Lia	12773	345	3.69	693
25	Shiho	19237	492	3.92	1018
38	Andrea	18545	1002	3.97	1317
77	Kim	27549	825	3.81	841
83	Samantha	6485	186	3.65	104
181	Gary	8683	227	3.27	347
233	Kevin	7875	383	3.95	384
252	Ruth	14973	487	3.22	1180
289	Steph	27139	991	4.23	800
352	Tiffany	10932	323	4.17	304
394	Pete	7563	184	4.09	319
413	Rob	8482	158	3.72	104
435	Andy	14792	1377	3.93	418
492	Karen	16950	558	3.69	357
547	Pegah	6142	231	4.04	76
608	Pasquale	8409	162	3.41	143
611	Andre	7674	184	4.05	627
616	Adam	9232	307	3.82	109
650	Marieille	15105	379	4.10	344
666	Jennifer	23254	828	3.34	766

Figure 12: Utenti con pochi voti “useful” inviati

Sample of Users with medium number of votes sent (up to 45000) an high number of votes receipt					
	name	useful	fans	average_stars	compliment_hot
103	Michelle	29481	308	4.28	12391
518	Anthony	36929	1021	3.24	7632
781	DJ	19181	290	3.58	6360
2783	Toni	39527	247	4.17	10944
3905	John	19176	620	3.77	4684
8462	Angela	28561	737	4.11	9597

Figure 13: Utenti con un buon numero di voti “useful” inviati e molti “compliment” ricevuti

Sample of Users with medium number of votes sent (up to 45000)					
	name	useful	fans	average_stars	compliment_hot
0	Daniel	43091	3138	3.74	1145
1	Jane	14953	1357	3.85	1713
9	Jelena	17331	828	4.06	2177
17	J	14004	316	3.61	1286
26	Monica	12640	804	4.17	1262
39	Yelper	11276	326	3.68	1178
52	Farrah	27350	2073	3.95	1575
59	Joi	27137	944	4.10	1729
86	Archie	13103	319	2.98	1113
153	Miriam	19610	1201	3.84	1991
166	Katy	28015	1118	4.20	1587
188	Matt	15653	351	3.74	3672
201	Chun	14647	433	4.34	1424
203	Patt	9050	145	4.05	1219
222	Colleen	10675	695	3.84	1025
275	Jenny	17333	427	3.62	2497
309	Eric	41470	444	3.94	2133
316	Kevin	12467	483	3.79	2811
353	Damien	15998	1106	3.87	888
364	Sandy	23700	513	4.01	1521

Figure 14: Utenti con un buon numero di voti “useful” inviati

Sample of Users with a huge number of votes sent (up to 200000)					
	name	useful	fans	average_stars	compliment_hot
2585	Bruce	173089	867	3.67	4375
9397	Fox	206296	3493	3.77	4076

Figure 15: Utenti con un enorme numero di voti “useful” inviati

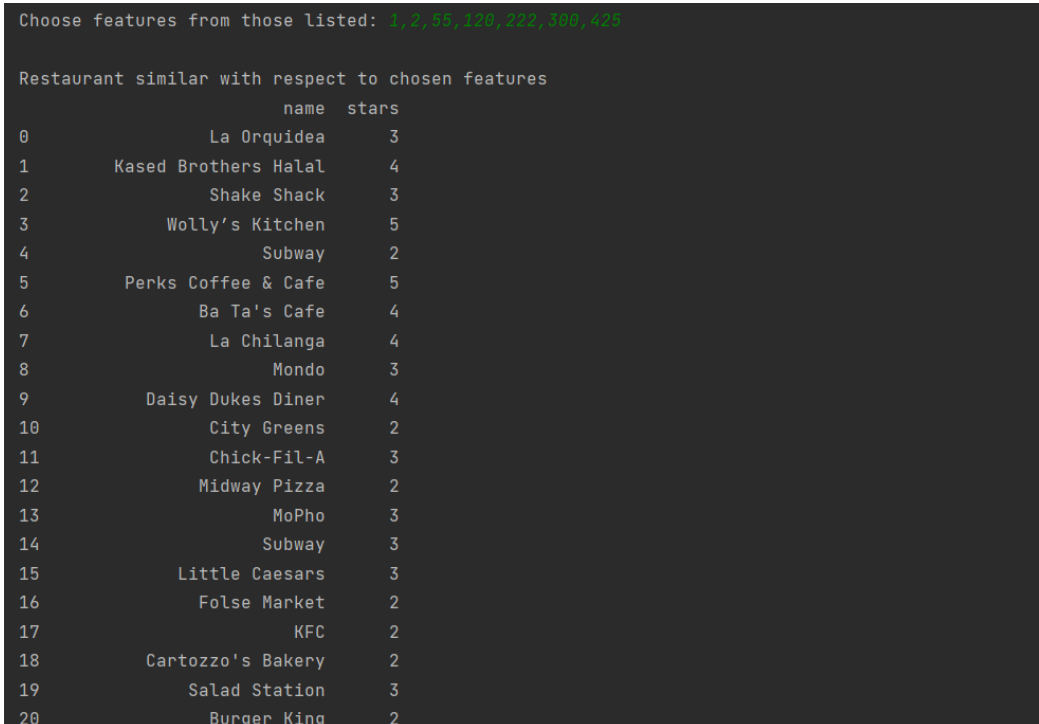
Sample of	Users with a high number of votes sent (up to 100000) and good number of votes receipt				
	name	useful	fans	average_stars	compliment_hot
68	Peter	45810	2388	4.01	2352
106	Dani	64312	412	3.39	5100
173	Ed	73528	2251	3.71	2754
309	Eric	41470	444	3.94	2133
392	Randy	48205	1124	3.77	893
1007	Phil	48424	841	3.57	4059
2154	Michelangelo	65413	1435	4.43	3943
2174	Nijole	57242	921	3.75	331
7032	May	106525	634	4.37	6119
8252	Christy	37564	740	4.19	2624
22557	Dave	76505	539	3.91	1810
22979	Jennifer	63234	655	3.98	1294
23109	J David	59561	615	3.75	2209
24430	Gerald	61899	703	3.68	599
25297	Aaron	70894	726	3.53	5695

Figure 16: Utenti con un alto numero di voti “useful” inviati

### 3.4.3 Raggruppamento dei ristoranti di una città in base a caratteristiche simili

Si vuole consentire all'utente di cercare ristoranti di una città a scelta in base a similitudini tra categorie e servizi offerti dai vari ristoranti.

Come mostrato in precedenza l'utente dovrà scegliere la città dove desidera scegliere un ristorante, scegliere una o più caratteristiche che il ristorante deve avere e gli verranno mostrati i ristoranti con caratteristiche simili a quelle cercate.



```
Choose features from those listed: 1,2,95,120,222,300,425

Restaurant similar with respect to chosen features
  name  stars
0      La Orquidea      3
1  Kased Brothers Halal  4
2      Shake Shack      3
3      Wolly's Kitchen  5
4      Subway           2
5  Perks Coffee & Cafe  5
6      Ba Ta's Cafe     4
7      La Chilanga      4
8      Mondo            3
9      Daisy Dukes Diner  4
10     City Greens      2
11     Chick-Fil-A      3
12     Midway Pizza     2
13     MoPho            3
14     Subway           3
15     Little Caesars   3
16     Folse Market     2
17     KFC              2
18     Cartozzo's Bakery 2
19     Salad Station    3
20     Burger King      2
```

Figure 17: Ristoranti con caratteristiche simili a quelle cercate

## 3.5 Comparazione dei risultati ottenuti

In questa sezione del report si confronteranno i risultati di tutte le alternative implementate.

### 3.5.1 Classificazione di una recensione come positiva o negativa

Per questo problema sono state confrontate 5 tecniche di classificazione diverse per poi scegliere la migliore tra le 5. Tutte e 5 ottengono degli ottimi punteggi di **accuracy**, **precision** e **F1** e sotto sono mostrati i risultati delle predizioni su due recensioni in input, una **positiva** e una **negativa**.

```
Prediction on an input string: There was too noises in the resturant and the chef was very rude!!!
Logistic Regression model: 0
Random Forest model      : 0
XGboost model            : 0
Transformer class.       : 0
NN classification        : 0

-----

Prediction on an input string: Magnificent place, very good food and friendly workers
Logistic Regression model: 1
Random Forest model      : 1
XGboost model            : 1
Transformer class.       : 0
NN classification        : 1
```

Figure 18: Comparazione dei risultati della classificazione

Dal confronto sopra mostrato si nota che 4 delle 5 tecniche di classificazione testate funzionano bene (rispettano le attese) mentre si evidenzia qualche problema per il modello basato su **BERT Transformer**, che come accennato, nonostante elevati valori di precision, recall e F1, lavora male in fase di test classificando ogni recensione come negativa.

### 3.5.2 Raggruppamento degli utenti in base al comportamento sulla piattaforma

Per quanto riguarda il problema di raggruppamento degli utenti mostreremo i risultati dei clustering effettuati in forma di grafico in quanto potrebbe risultare più chiara la differenza tra i due algoritmi testati.

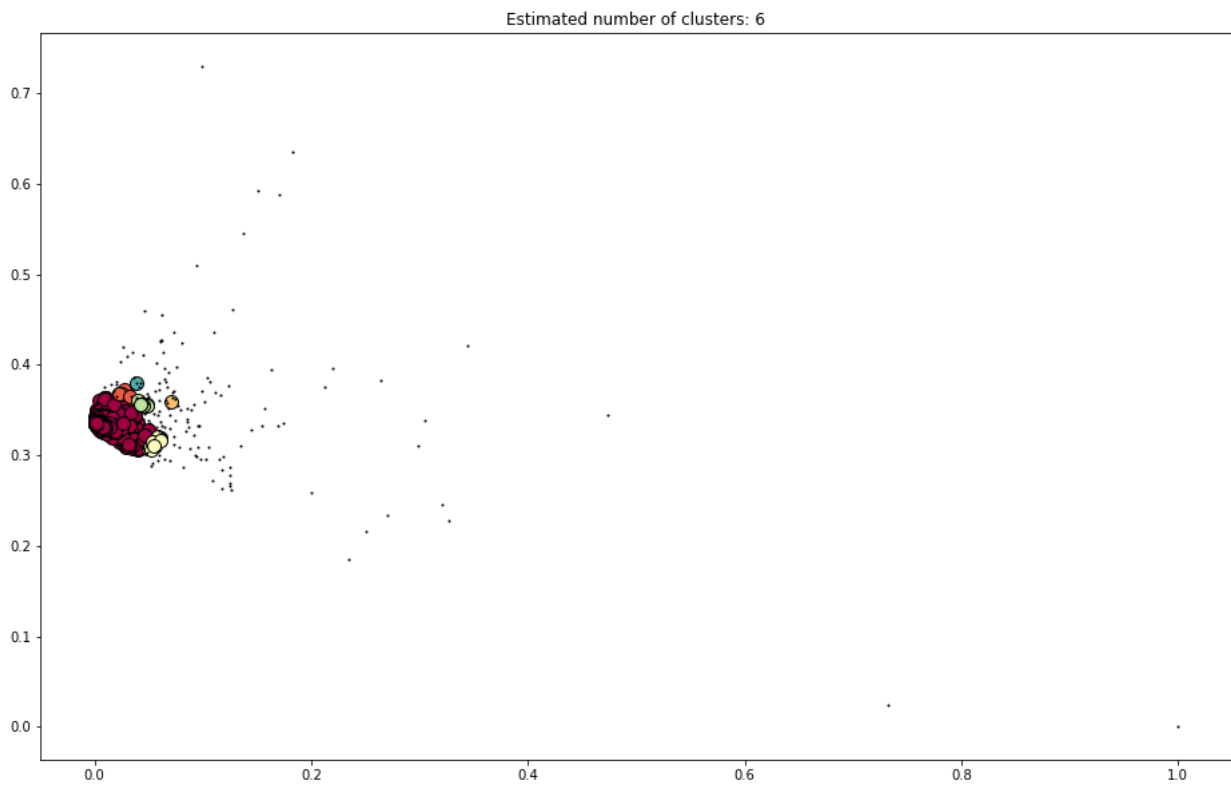


Figure 19: DBSCAN Clustering



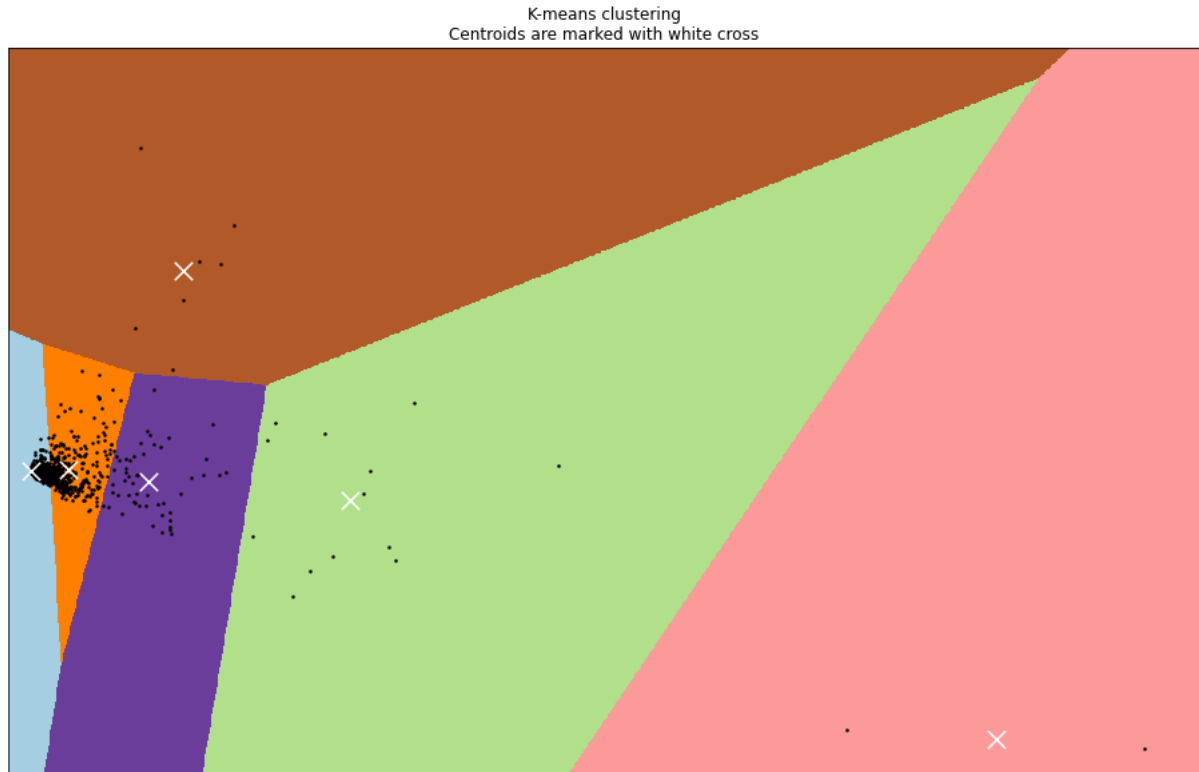


Figure 20: KMEANS Clustering

A parità di numero di cluster generati (6 anche per KMEANS dato che permette di ottenere il silhouette score migliore) si nota subito che per il nostro scopo è meglio utilizzare il clustering ottenuto da KMEANS.

Il clustering ottenuto da DBSCAN (nonostante anche DBSCAN abbia ottenuto un silhouette score molto alto) crea due macro-categorie di utenti (punti rossi e outliers) e altre 4 composte da pochi punti. Si preferisce non utilizzare tale clustering in quanto nei macro-cluster potremmo trovare un range troppo ampio di utenti con un comportamento differente sulla piattaforma.

### 3.5.3 Raggruppamento dei ristoranti in base a città e caratteristiche

Anche per questo problema mostreremo i grafici relativi ai cluster ottenuti dai diversi algoritmi, tuttavia non è stato possibile inserire il grafico relativo al clustering di KNN a causa di problemi con la libreria che ne consentiva il plot. Mostreremo comunque i grafici degli altri due algoritmi confrontati, ovvero **DBSCAN** e **KNN**.

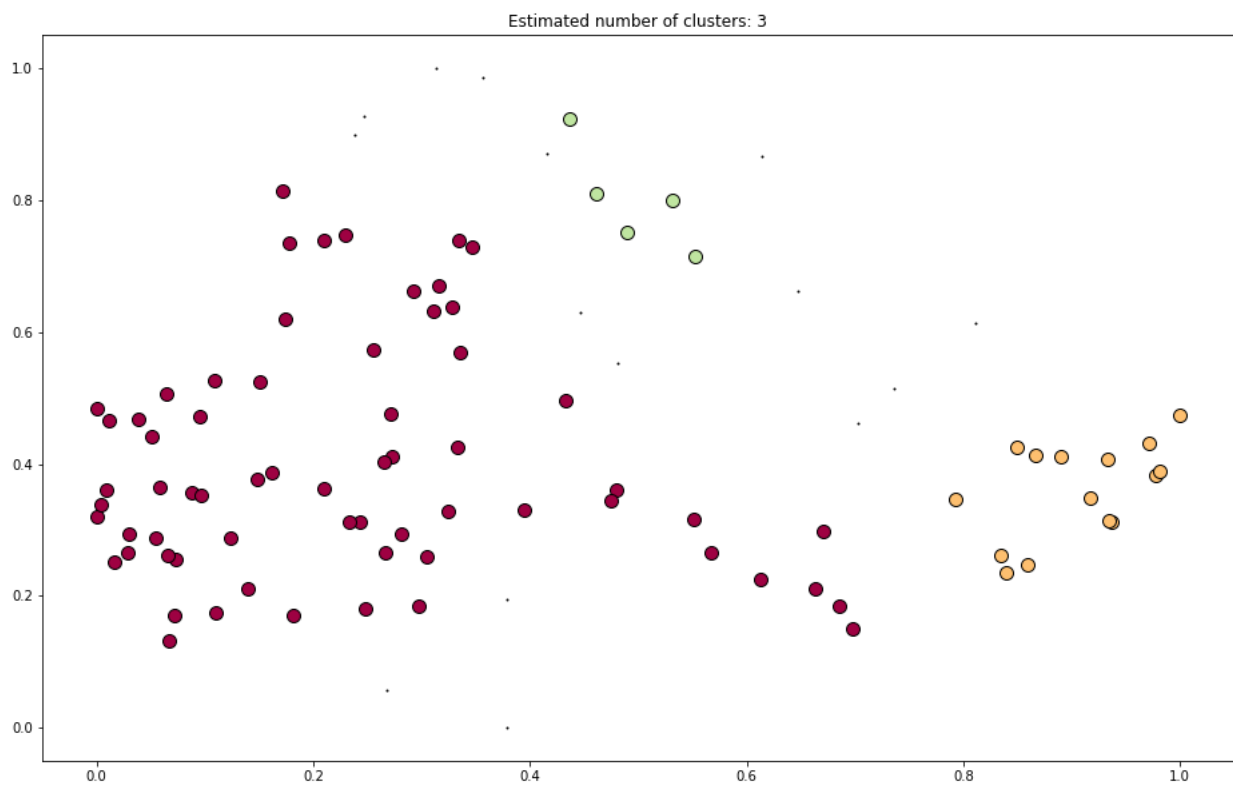


Figure 21: DBSCAN Clustering

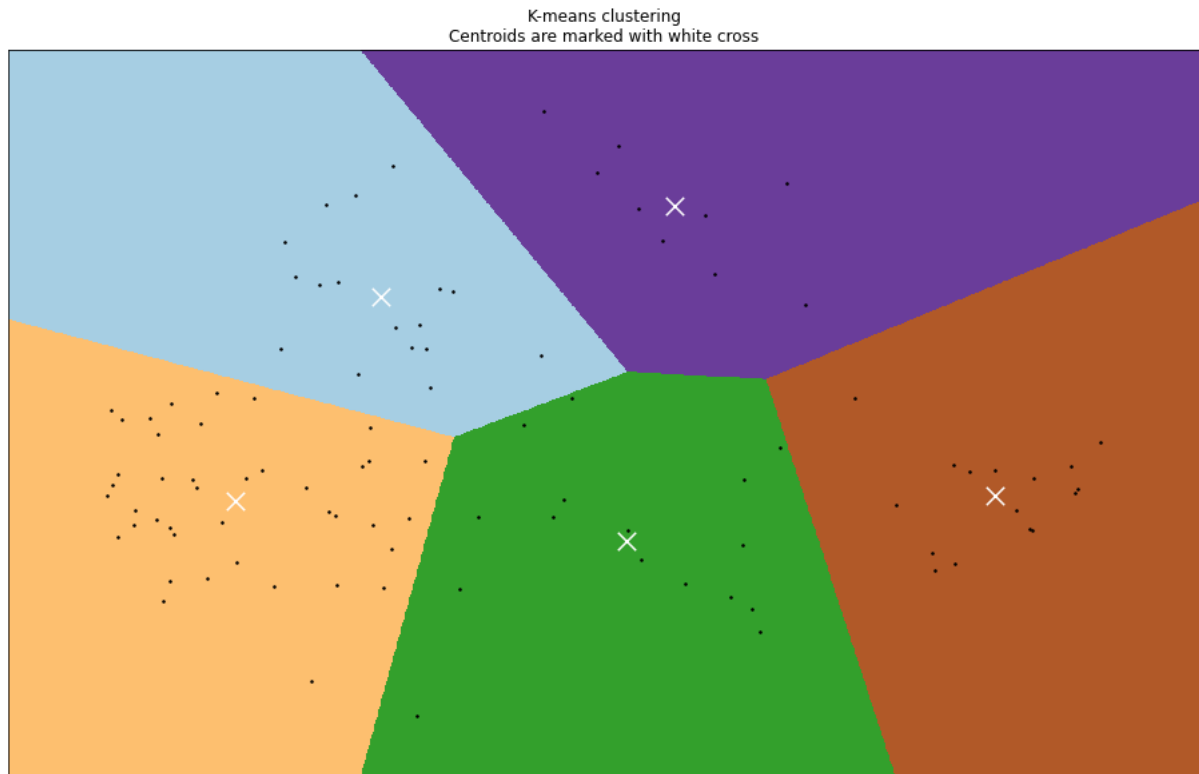


Figure 22: KMEANS Clustering

I grafici mostrati sopra sono relativi al clustering effettuato per la città di **Kenner**.

In questo caso **KNN** non è stato preso in considerazione in quanto è l'algoritmo che ottiene il silhouette score più basso.

La scelta, anche qui, è ricaduta su **KMEANS** dato che è l'algoritmo che ottiene le performance migliori e a livello visivo sembra creare dei cluster più coerenti per il nostro scopo, **DBSCAN** invece, come per il raggruppamento degli utenti, crea dei cluster che includono ristoranti con caratteristiche eterogenee.

### 3.6 Classificazione di una recensione da 1 a 5 stelle

Come già anticipato, questo problema è un raffinamento del problema di classificazione binaria di cui abbiamo già parlato, tuttavia, nonostante all'apparenza sono problemi simili, gli approcci utilizzati per la risoluzione sono molto diversi (2 dei quali sono stati adattati anche alla classificazione binaria per verificarne le performance e sono **Neural Network** e **Transformer**).

```
Prediction on an input string: There was too noises in the resturant and the chef was very rude!!!  
  
Naive Bayes prediction: ★ stars  
-----  
  
Neural Network prediction: ★ stars  
-----  
  
Transformer prediction: ★ stars
```

Figure 23: Comparazione risultati su recensione da 1 stella

```
Prediction on an input string: ordinary restaurant, quite good food, no service. fast-food like  
  
Naive Bayes prediction: ★★★★★ stars  
-----  
  
Neural Network prediction: ★★★★★ stars  
-----  
  
Transformer prediction: ★★ stars
```

Figure 24: Comparazione risultati su recensione da 2 stelle

```
Prediction on an input string: ordinary restaurant, quite good food  
  
Naive Bayes prediction: ★★★★★ stars  
-----  
  
Neural Network prediction: ★★★★★ stars  
-----  
  
Transformer prediction: ★★★★★ stars
```

Figure 25: Comparazione risultati su recensione da 3 stelle

```
Prediction on an input string: fantastic restaurant!!! very good food but services was bad

Naive Bayes prediction: ★★★★★ stars
-----

Neural Network prediction: ★★★★★ stars
-----

Transformer prediction: ★★★★★ stars
```

Figure 26: Comparazione risultati su recensione da 4 stelle

```
Prediction on an input string: fantastic restaurant!!!

Naive Bayes prediction: ★★★★★ stars
-----

Neural Network prediction: ★★★★★ stars
-----

Transformer prediction: ★★★★★ stars
```

Figure 27: Comparazione risultati su recensione da 5 stelle

Dai risultati mostrati si può notare che per recensioni molto positive o molto negative tutti i modelli concordano sulla valutazione in stelle da suggerire mentre per valutazioni intermedie si possono notare alcune differenze, in particolare, **Neural Network** ed in particolare **Naive Bayes Classification** soffrono un po' mentre il **Transformer**, rispetto al caso della classificazione binaria, sembra lavorare molto bene.

Tuttavia c'è da tenere conto del fatto il rating di una recensione dato da un utente non è una valutazione oggettiva e potrebbero esserci nel dataset alcune recensioni il quale rating non rispetta a pieno ciò che c'è scritto nella recensione vera e propria.

Tutto ciò verrà confermato dallo studio dei punteggi ottenuti nel capitolo seguente.

## 4 Discussioni e conclusione

### 4.1 Risultati ottenuti in termini di performance

Di seguito verrà riportato il confronto delle performance per ognuno dei problemi affrontati.

#### 4.1.1 Classificazione di una recensione come positiva o negativa

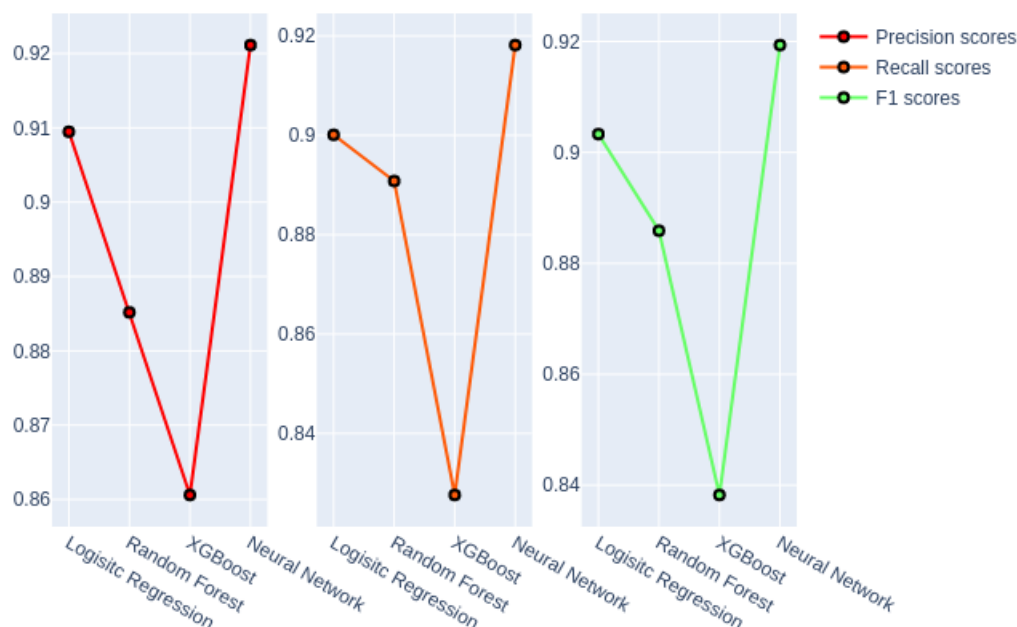


Figure 28: Confronto tra le metriche scelte

Come già anticipato il Transformer è stato omesso dall'analisi delle performance a causa di problemi in fase di test molto probabilmente a causa dell'elevato sbilanciamento del dataset (sono presenti molte più recensioni etichettate come positive che come negative) e dell'utilizzo di una porzione ancora più ristretta del dataset rispetto agli altri modelli (causa StackOverflow) per il training.

Analizzando le performance delle tecniche confrontate si nota che **Logistic Regression** e **Random Forest** hanno ottimi punteggi ma con un leggero

vantaggio per il classificatore basato su Logistic Regression (risultato che non sorprende in quanto il comportamento dei due algoritmi è lo stesso che riscontriamo in [5]), per quanto riguarda **XGBoost** si nota che è l'algoritmo che ottiene le performance peggiori rispetto agli altri (anche se di poco), risultato che tuttavia non coincide con quello mostrato in [3]. Tale perdita di performance può essere dovuta al fatto che il dataset sia sbilanciato.

Il metodo con le migliori performance risulta essere il classificatore basato su **Neural Network** che, come già descritto in [4], risulta essere un buon metodo per processare informazioni che derivano da testi.

#### 4.1.2 Raggruppamento degli utenti in base al comportamento sulla piattaforma

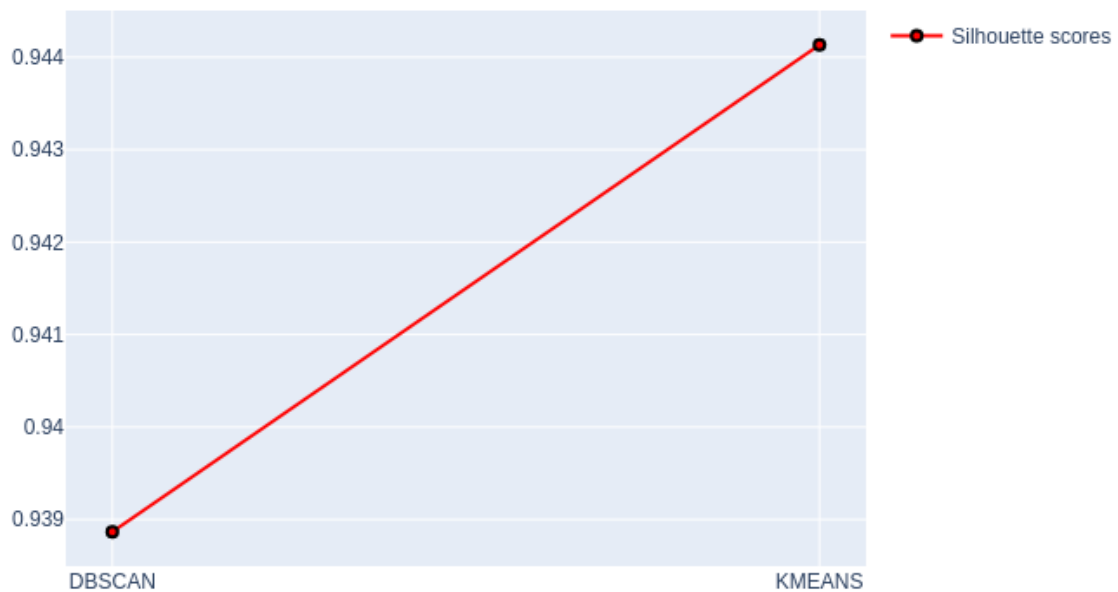


Figure 29: Confronto Silhouette scores

Per questo problema entrambe le tecniche proposte hanno ottenuto ottimi punteggi, tanto che la differenza tra le due è molto piccola, tuttavia l'effettiva

clusterizzazione risulta essere molto differente tra i 2 in quanto sono algoritmi che tengono conto di informazioni completamente diverse nella fase di creazione dei cluster. Risulta difficile comparare i risultati ottenuti con quelli mostrati in [2] e [7].

#### 4.1.3 Raggruppamento dei ristoranti in base a città e caratteristiche simili

Per lo svolgimento di questo task sono stati presi in considerazione i ristoranti della città di **Kenner**.

In questo caso nel confronto è stato inserito anche **KNN** che utilizza come insieme di labels la valutazione in stelle dei ristoranti. I punteggi ottenuti dai tre algoritmi sono i seguenti:

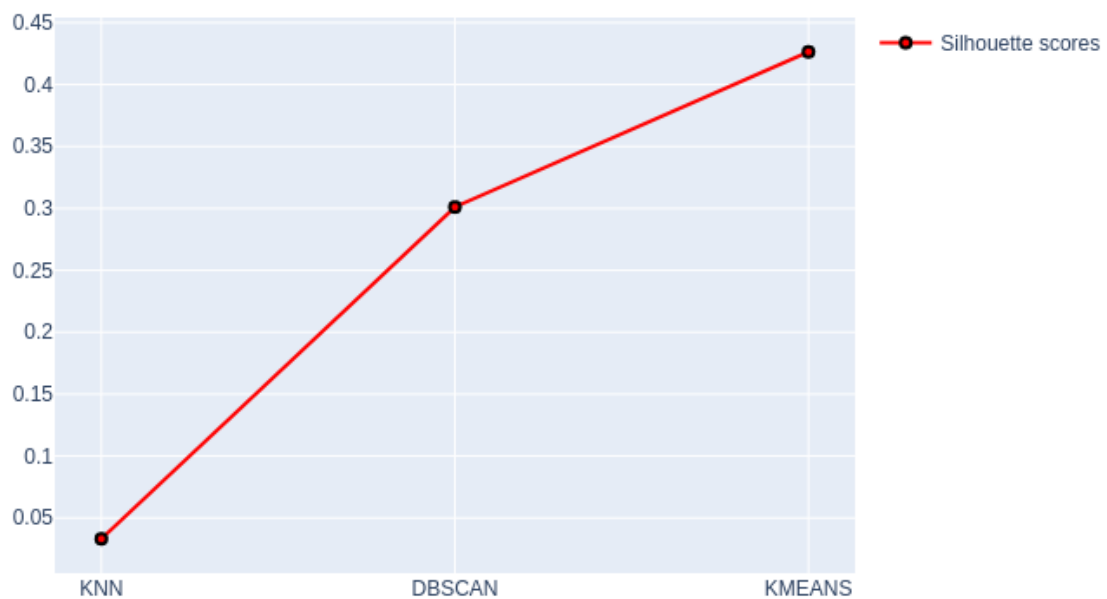


Figure 30: Confronto Silhouette scores

Si nota subito che l'algoritmo con le migliori performance in questo caso è



**KMEANS**, anche in questo caso risulta difficile comparare i risultati con la letteratura presa in considerazione.

## 4.2 Classificazione di una recensione da 1 a 5 stelle

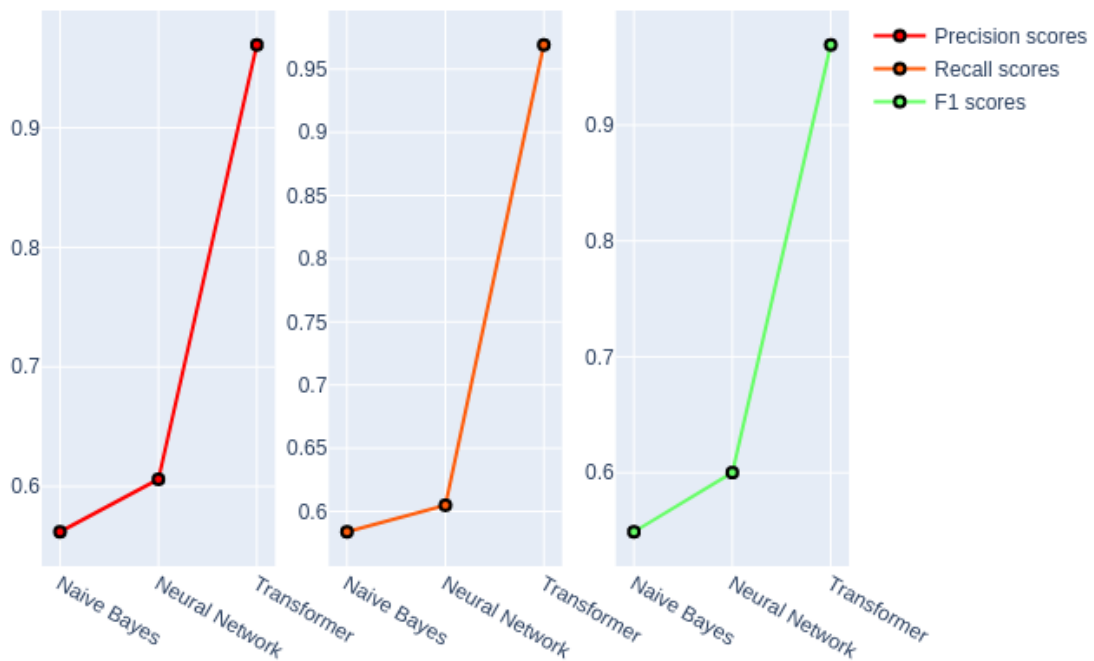


Figure 31: Confronto delle performance tra i 3 algoritmi usati

A differenza del primo caso, qui il **Transformer** è effettivamente il modello che svolge il lavoro nel modo migliore e che ottiene i migliori punteggi. Inoltre, diversamente a come mostrato in [1], qui i punteggi ottenuti sono molto più alti, probabilmente ciò è dovuto al fatto che la classificazione effettuata è più “semplice” rispetto a quella mostrata nell’articolo. Anche nel primo caso bisogna dire i punteggi ottenuti dal Transformer sono stati superiori agli altri algoritmi, ma quel problema in fase di test ci ha fatto preferire l’omissione del transformer dal confronto dei punteggi.

Per quanto riguarda **Naive Bayes Classification** e **Neural Network** i

risultati sono stati inferiori e di molto rispetto alle aspettative e a quanto viene mostrato in [6] e [4] rispettivamente.

### 4.3 Confronto con le aspettative e con la letteratura

#### 4.3.1 Classificazione di una recensione come positiva o negativa

[3], [5] e [4] ci dicono che **XGBoost**, **Logistic Regression**, **Random Forest** e **Word2Vec NN** sono tecniche che permettono di ottenere ottime performance in merito al problema di classificazione binaria su di un testo, di conseguenza ci si aspetta di ottenere performance simili dato che i problemi da noi affrontati hanno degli elementi in comune con quelli descritti nella letteratura. Le aspettative sono state pienamente rispettate da tutti gli algoritmi tranne che per il classificatore basato su **BERT Transformer**, in termini di punteggi ha superato di molto ciò che viene descritto in [1] (comunque stiamo trattando una classificazione più semplice), tuttavia ha mostrato problemi in fase di test.

#### 4.3.2 Raggruppamento degli utenti in base al comportamento sulla piattaforma

[2] e [7] ci dicono che **DBSCAN** e **KMEANS** sono ottimi algoritmi di clustering e unsupervised learning.

Entrambi forniscono ottime performance su insiemi diversi di dati e il loro funzionamento è totalmente differente. È risultato difficile confrontare i risultati ottenuti con quelli mostrati in letteratura ma anche nel nostro caso entrambi gli algoritmi hanno funzionato bene.

La differenza di performance è molto piccola tuttavia la scelta di **KMEANS** è stata aiutata dal clustering ottenuto che sembra attenersi maggiormente al nostro obiettivo.

#### 4.3.3 Raggruppamento dei ristoranti di una città in base a caratteristiche simili

Per questo problema oltre ai due algoritmi presentati per il raggruppamento degli utenti è stato preso in analisi anche **KNN** ma con risultati abbastanza scarsi. Visti i risultati del precedente problema, ci aspettavamo delle performance simili, ma i record trasformati in punti in questo caso presentano delle regioni con densità di punti completamente diverse rispetto al caso precedente.

L'algoritmo che ha ottenuto il **silhouette score** peggiore è **KNN** con un punteggio vicino allo zero (che è basso ma il range ricordiamo va da -1 a 1),

mentre il migliore è **KMEANS** con un punteggio di 0.5 circa. **DBSCAN** si piazza tra i due con un punteggio di 0.3 circa.

Ci si aspettava un risultato migliore da **KNN** ma alla fine si è rivelato essere il peggiore dei 3 algoritmi confrontati.

#### 4.3.4 Classificazione di una recenziona da 1 a 5 stelle

[6], [4] e [1] mostrano come le relative tecniche di classificazione funzionino abbastanza bene, tuttavia le performance ottenute non hanno soddisfatto le attese per quanto riguarda **Naive Bayes** e **Word2vec NN**. Per quanto riguarda il **BERT Transformer** invece si rispecchia quanto descritto in [1] e le aspettative sono state superate di gran lunga con punteggi che sfiorano il 92% di **precision**, **recall** e **f1-score**. Tuttavia, non ci si aspettavano dei risultati così diversi tra **BERT Transformer** e **Word2vec NN**.

### 4.4 Limiti della soluzione proposta

I limiti delle soluzioni proposte sono i seguenti:

- Per quanto riguarda gli algoritmi di classificazione, sono stati allenati su una porzione ristretta del dataset (circa 100 mila recensioni su un totale di circa 2 milioni) a causa dei tempi di allenamento molto lunghi;
- Per il **BERT Transformer**, che in fase di training risulta essere ancora più lento degli altri algoritmi usati per fare classificazione di recensioni, è stato effettuato il training solo su un training set di 10 mila elementi circa, sia per il caso della classificazione binaria che n-aria;
- Per determinare se una recensione è positiva o negativa si potrebbero usare al posto della neural network, o Logistic Regression o XGBoost (il secondo da provare su un dataset bilanciato) che risultano avere punteggi lievemente più scarsi ma risultano essere più veloci in termini di tempo di esecuzione in fase di prediction.
- Per il **BERT Transformer** nel caso della classificazione binaria si è cercato di allenarlo sempre su un insieme ristretto di 10 mila record circa ma bilanciando il numero di recensioni considerate positive e negative ma il problema in fase di test/prediction persisteva.
- Riguardo al problema del raggruppamento degli utenti, il clustering è stato eseguito su un insieme ristretto di 30 mila utenti circa, questo a causa di DBSCAN che con un numero maggiore di dati in input andava in stack overflow.

## 4.5 Lavori futuri

I possibili sviluppi futuri per tale piattaforma potrebbero essere:

1. Risoluzione dei limiti precedentemente evidenziati;
2. Cercare soluzioni migliori o migliorare quelle attuali per quanto riguarda il raggruppamento dei ristoranti;
3. Affinare il suggerimento della valutazione in stelle di una recensione aggiungendo eventualmente le stelle ammezzate (raddoppiando così il numero di classi tra le quali classificare i dati in input);
4. Provare a fornire un indice di gradimento dei ristoranti analizzando i testi presenti nel dataset delle mance (tips), per fare ciò è possibile passare in input ogni recensione presente nel dataset delle mance al **BERT Transformer** per ricavarne un indice da associargli.

## References

- [1] Francisca Adoma Acheampong, Henry Nunoo-Mensah, and Wenyu Chen. “Transformer models for text-based emotion detection: a review of BERT-based approaches”. In: *Artificial Intelligence Review* 54.8 (2021), pp. 5789–5829.
- [2] Dingsheng Deng. “DBSCAN clustering algorithm based on density”. In: *2020 7th International Forum on Electrical Engineering and Automation (IFEEA)*. IEEE. 2020, pp. 949–953.
- [3] JP Haumahu, SDH Permana, and Y Yaddarabullah. “Fake news classification for Indonesian news using Extreme Gradient Boosting (XGBoost)”. In: *IOP Conference Series: Materials Science and Engineering*. Vol. 1098. 5. IOP Publishing. 2021, p. 052081.
- [4] Beakcheol Jang, Inhwan Kim, and Jong Wook Kim. “Word2vec convolutional neural networks for classification of news articles and tweets”. In: *PloS one* 14.8 (2019), e0220976.
- [5] Kaitlin Kirasich, Trace Smith, and Bivin Sadler. “Random forest vs logistic regression: binary classification for heterogeneous datasets”. In: *SMU Data Science Review* 1.3 (2018), p. 9.
- [6] Sebastian Raschka. “Naive bayes and text classification i-introduction and theory”. In: *arXiv preprint arXiv:1410.5329* (2014).
- [7] Kristina P Sinaga and Miin-Shen Yang. “Unsupervised K-means clustering algorithm”. In: *IEEE access* 8 (2020), pp. 80716–80727.