iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 12 **Issue:** III **Month of publication:** March 2024

DOI: https://doi.org/10.22214/ijraset.2024.59326

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue III Mar 2024- Available at www.ijraset.com

Mathematical Model Applied for Hotel Recommendation System with Machine Learning

B.Shyti[1], A.Stërgu[2], Dh.Valera[3]
[1]*Profesor,* [2]*Assistant Lecturer,* [3]*Lecturer, Department of Mathematics, Faculty of Natural Sciences, University of Elbasan "A.Xhuvani"*

Abstract: ***the purpose of this paper is to promote the value of Albanian tourism creating a Hotel Recommendation System applying a Mathematical Model. The data of one hundred and one hotels, loaded from two excel files in CSV format are implemented in the Spyder IDE and processed by code, using Mathematical and Machine Learning knowledge. The code recommends ten best hotels as output. There is made a combination of Machine Learning methods and techniques with the concepts of probability, statistics, algebra and geometry. The main methodological approach taken in this study is Content-Based Filtering.***

Keywords: ***Recommendation Systems, Machine Learning, Hotels, Tourism, Probability, Statistics,***

I. INTRODUCTION

Albania is a small country, very little known in the world, but with great natural resources, which create the possibility of tourism development. This paper brings an innovation on how to promote the values carried by Albanian tourism and how to make the names of some of the most prestigious hotels on the Albanian coast known, using Machine Learning and mathematical knowledge, such as Bayes' theorem, Conditional Probability, Cosine Similarity, etc.,

This paper serves the readers as a guide to orientate foreign tourists towards the Albanian coast, as well as a methodology that explains the creation of a code through the knowledge of the branches of mathematics and ML.

The rapid progress of technology and the increasing need of people to use it has resulted in a rise of data. The information spreads quickly and in a short time. The necessity of people to handle with this large amount of data prompted the advance of Artificial Intelligence (AI). According to the impacts of AI, the human society is adopting it in all areas of life [1].

Nowadays, people are becoming more and more addicted to technology. They spend most of the time on the phone, iPad, laptop, computer desktop, etc. They can shop there, have access to up-to-date information on events, communicate with each other, booking hotels, etc. So, there is a good idea, to promote tourism through code implemented with the Python programming language.

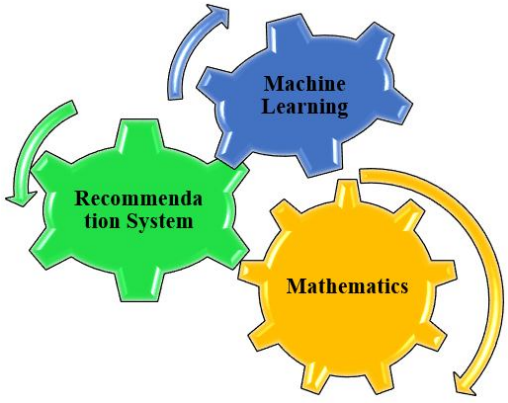


Fig. 1 The relationship between Mathematics, Machine Learning and Recommendation System

This paper presents a Machine-Learning Model for Hotel Recommendation System using knowledge of different areas of mathematics, including: probability and statistics, linear algebra, calculus, geometry. This code is like booking application, that people access to search hotels for holidays. The intelligent machine is placed at the service of human society, thanks to the algorithms, that allow it to process and generate new models based on data receiving from a huge database [2].

©IJRASET: All Rights are Reserved

Machine Learning is a sub-field of Artificial Intelligence, which uses Recommendation Systems to recognize and matches the behavior of users, in order to train algorithms and create different models [3].

Algorithms learn from the activity of users (viewers, likes, dislikes, comments, ratings, etc.) and make hotel recommendations. The users are able to choose hotels they may like, grouped according to the features studied by algorithms. Furthermore, hotels from large datasets can be filtered using machine learning based autonomous tool recommendation movie system [4].

Additionally, Recommendation System can increase the amount of products and provide personalized service support to users by learning their previous behaviors and predicting their current preference for products [5].

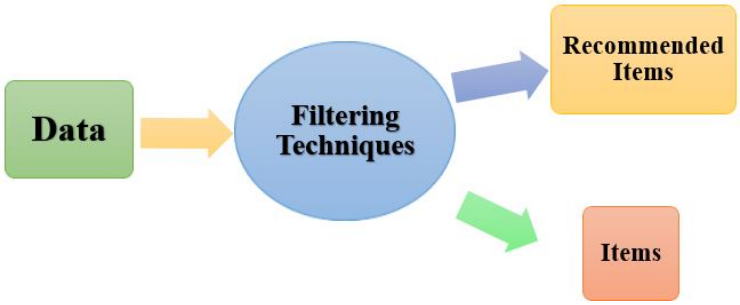


Fig. 2 Three components of Hotel Recommendation System

We have written the code in an open source Interactive Development Environment platform (IDE) called Spyder (https://www.spyder-ide.org/), part of the Anaconda Navigator package (https://docs.anaconda.com/free/anaconda/ide-tutorials/spyder/). The type of Recommendation System that we have used is Content-Based Filtering. It generates recommendations by examining the descriptions of the products the user has rated. Three main components, which go into the creation of a Hotel Recommendation System, are the personal data of the user (interests, history of watching in Netflix, ratings, reviews), item data (features and specifications) and the technique of filtering.

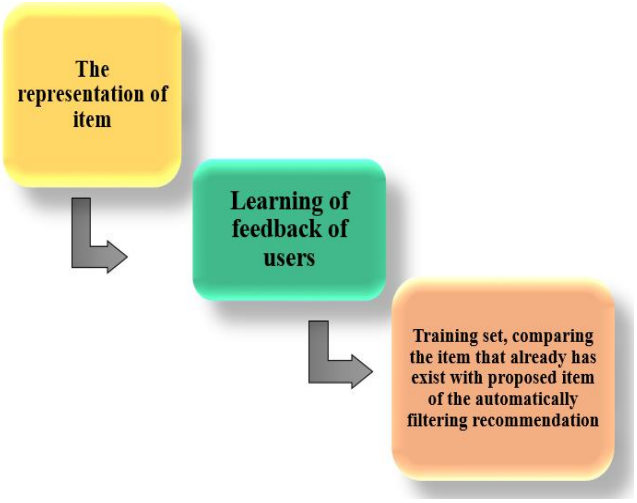


Fig. 3 Example of an unacceptable low-resolution image

There are three common types of Recommended Systems, respectively: “Content-Based Filtering”, “Collaborative Filtering”, and “Hybrid Filtering”.

The Content-Based System placed the algorithm in some steps: the representation of item, learning of feedback of users (explicit or implicit), training set, comparing the item that already has exist with proposed item of the automatically filtering recommendation [6]. The paper is divided into three sections. The first section focuses in the role of Content-

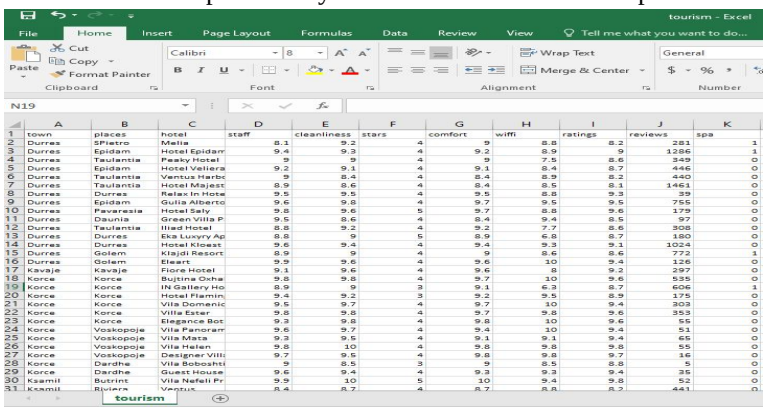
Based Filtering method in Recommendation Systems. In the second section, we present the code written in the Editor and evaluated in the *IPython* console. We explain all the steps for creating this algorithm, using probability knowledge. The third section contains information on the results, conclusion, and benefits of Recommendation Systems. The simplest implementation of this is through a binary feature matrix. Columns can be represented as features, while the row can be an object that either has those features (1) or not (0).

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue III Mar 2024- Available at www.ijraset.com

II. METHODS AND RESULTS

A. *Loading from external dataset*

First of all, we collect data from the page: *https://www.booking.com* and organize them in a dataset called tourism in Excel. Secondly, we store the file of excel in Comma-Separated Value (CSV) format. Thirdly, we implement the dataset with thirty-eight movies. Each hotel has eleven attributes: town, places, hotel, staff, cleanliness, stars, comfort, wiffi, ratings, reviews, spa. There is a screenshot of the "tourism" dataset below. We import five Python libraries. Pandas is an open-source library that provides.



	A	B	C	D	E	F	G	H	I	J	K
1	town	places	hotel	staff	cleanliness	stars	comfort	wiffi	ratings	reviews	spa
2	Durres	SPietro	Melia	8.1	9.2	4	9	8.8	8.2	281	1
3	Durres	Epidam	Hotel Epidam	9.4	9.3	4	9.2	8.9	9	1286	1
4	Durres	Taulantia	Peaky Hotel	9	9	4	9	7.5	8.6	349	0
5	Durres	Epidam	Hotel Vellera	9.2	9.1	4	9.1	8.4	8.7	446	0
6	Durres	Taulantia	Ventus Harbo	9	8.4	4	8.4	8.9	8.2	440	0
7	Durres	Taulantia	Hotel Majest	8.9	8.6	4	8.4	8.5	8.1	1461	0
8	Durres	Durres	Relax In Hote	9.5	9.5	4	9.5	8.8	9.3	39	0
9	Durres	Epidam	Gulia Alberto	9.6	9.8	4	9.7	9.5	9.5	755	0
10	Durres	Pavaresia	Hotel Saly	9.8	9.6	5	9.7	8.8	9.6	179	0
11	Durres	Daunia	Green Villa P	9.5	8.6	4	8.4	9.4	8.5	97	0
12	Durres	Taulantia	Ilias Hotel	8.8	9.2	4	9.2	7.7	8.6	308	0
13	Durres	Durres	Eka Luxyry Ap	8.8	9	5	8.9	6.8	8.7	180	0
14	Durres	Durres	Hotel Kloast	9.6	9.4	4	9.4	9.3	9.1	1024	0
15	Durres	Golem	Klajdi Resort	8.9	9	4	9	8.6	8.6	772	1
16	Durres	Golem	Eleart	9.9	9.6	4	9.6	10	9.4	126	0
17	Kavaje	Kavaje	Fiore Hotel	9.1	9.6	4	9.6	8	9.2	297	0
18	Korce	Korce	Bujtina Oxha	9.8	9.8	4	9.7	10	9.6	535	0
19	Korce	Korce	IN Gallery Ho	8.9	9	3	9.1	6.3	8.7	606	1
20	Korce	Korce	Hotel Flamin	9.4	9.2	3	9.2	9.5	8.9	175	0
21	Korce	Korce	Vila Domenic	9.5	9.7	4	9.7	10	9.4	303	0
22	Korce	Korce	Villa Ester	9.8	9.8	4	9.7	9.8	9.6	353	0
23	Korce	Korce	Elegance Bot	9.3	9.8	4	9.8	10	9.6	55	0
24	Korce	Voskopoje	Vila Panoram	9.6	9.7	4	9.4	10	9.4	51	0
25	Korce	Voskopoje	Vila Mata	9.3	9.5	4	9.1	9.1	9.4	65	0
26	Korce	Voskopoje	Vila Helen	9.8	10	4	9.8	9.8	9.8	55	0
27	Korce	Voskopoje	Designer Villi	9.7	9.5	4	9.8	9.8	9.7	16	0
28	Korce	Dardhe	Vila Boboshti	9	8.5	3	9	8.5	8.8	5	0
29	Korce	Dardhe	Guest House	9.6	9.4	4	9.3	9.3	9.4	35	0
30	Kamil	Butrint	Vila Nafeli Pr	9.9	10	5	10	9.4	9.8	52	0
31	Kamil	Riviera	Ventus	8.4	8.7	4	8.7	8.8	8.2	441	0

Fig. 4 Dataset of the hotels with their values in CSV format of Excel

tools for reading and writing data structures into files and manipulating them. NumPy (Numerical Operations in Python) is the fundamental package for scientific computing in Python [7]. It provides a multidimensional array object and various derived objects (arrays and matrices). *Scikit-learn (Sklearn)* is the most useful and robust library for machine learning in Python.

It provides a selection of efficient tools for machine learning and statistical modeling, including classification, regression, and clustering in Python. *Matplotlib* is a popular visualization library for making 2D plots of arrays and creating static, interactive, and animated visualizations in Python [8].

The code *pandas.read-csv* implement all the data in the code. If we want to apply the string split method on a pandas data-frame based on a condition and print even the data of the column called "places" and "hotel", we use this piece of code:

```
rs_h=pd.read_csv('tourism.csv')
print(rs_h)
rs_h['places'] = rs_h['places'].str.split('|')
rs_h['places'] = rs_h['places'].fillna("").astype('str')
tourism1 = pd.read_csv('tourism.csv', sep=',', encoding='latin-1', usecols=['hotel', 'places'])
print(tourism1)
```

Part of the code implementation is also the bar chat, we can show the distribution of views for 30 hotels

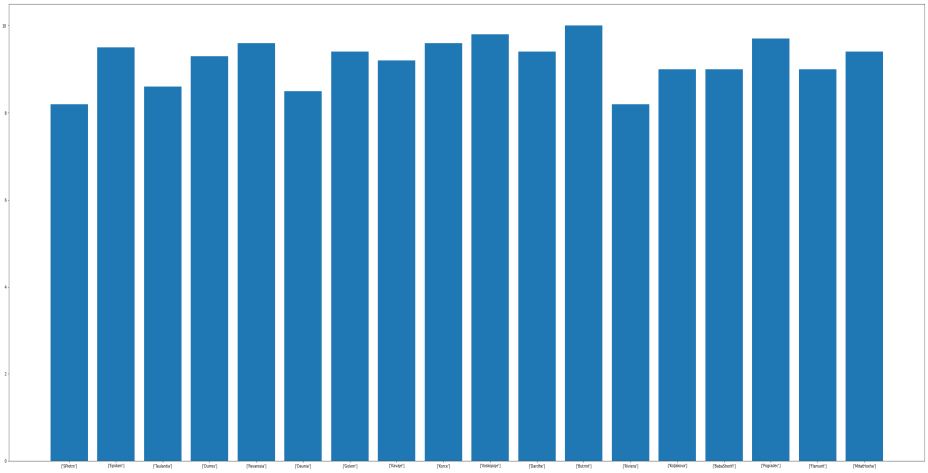


Fig. 5 The bar chart of the distribution of views for 30 hotels

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue III Mar 2024- Available at www.ijraset.com

B. The implementing of TD IDF vectorizer algorithm in the code

Starting with a set of data in a document *(d)* and the set of term *(t)*, we can model each data as a vector *v* in *t* dimensional as follow: *spaceRt*. We import Term Frequency Inverse Document Frequency *(TD-IDF)* from *Sklearn* library. We call the variable *freq-time* and give as a value the object *TdidfVectorizer*. The attributes of the object are used to analyze *words* and *n-gram* characters. This algorithm transforms text to numbers, as long as the prediction of hotels will be made. The piece of code *tfidf-matrix = freq-term.fit-transform (tourism['hotel'])* converts the data into a matrix.

```
tf = TfidfVectorizer(analyzer='word',ngram_range=(1, 1),min_df=0, stop_words='english')
tfidf_matrix = tf.fit_transform(tourism1['places'])
tfidf_matrix.shape
print(tfidf_matrix)
```

There is the mathematical formulation of the measure TFIDF

The term *frequencyfreq(d,t)* will be the number of occurrence of term *t* in the document *d*. The term *frequency matrix (weight)* or *TF(d.t)* measures the associated of a term *t* with respect to the given document *d*. It is defined as zero (0), if the document does not contain the term and nonzero (1) otherwise [10]. *TF*IDF = (the amount of times a term appears in the document)*(a term is common in the collection of documents) = count of t in d / number of words* in d and *DF(t)=occurrence* of *t* in documents. By comparing the word frequency to the total number of times each term appears in the document, one can determine the relative term frequency.

$$TF(d,t) = \begin{cases} 0 & freq(d,t) = 0 \\ 1 + \log(freq(d,t)) & freq(d,t) \neq 0 \end{cases}$$

Another important measure, called inverse document frequency (IDF) in formula. It represents the importance of a term t (scaling factor) and will be reduce if a term t occurs in most of the data.

$$IDF(t) = N/df(t) = N/N(t)$$
$$IDF(t) = log(N/df(t))$$
$$IDF(t) = log(1 + |d|)/(dt)$$
$$IDF\ as\ ID(t) = log[n/(df(t) + 1)])$$

Where, *d* is the document collection, and *dt* is the set of documents containing term *t*. In complete vector space model, *TF* and *IDF* are combined together, which forms the TF-IDF measure. In the dataset "tourism", there are 38 samples, three features (independent data), 2 classes (zero / one) [11].

$$f - IDF(t,d) = TF(t,d) * IDF(t)$$
$$TF - IDF(d,t) = TF(d,t) * IDF(t)$$

C. Implementing Cosine Similarity in the code

The second research question, we need to give an answer is "Which movies are similar to others?" To answer this question, we are going to import the *SciPy* library to provide the function of cosine. There is time to determine the correlation between each hotel's ratings. This function will calculate pairwise similarities for all movies in our vector matrix.

```
cosine_sim = cosine_similarity(tfidf_matrix, tfidf_matrix)
cosine_sim[:4, :4]
print(cosine_sim)
```

The mathematical viewpoint of Cosine Similarity Metric

Let *V = v1,v2,v3, ...,vn andW = w1,w2,w3, ...,wn* be two non-zero vectors. The dot product of vectors is given by the formula *V = v1 ∗w1+v2 ∗w2+...+vn ∗wn.* The magnitude formula of the vector V:

$$|V| = \sqrt{((v_1)^2) + (v_2)^2 + (...) + (v_n)^2}$$

Cosine is a type of angle measurement between two vectors. In this case, the vectors can be the user preference and the item. How far apart they are is what measures the similarity. It is not a measure of magnitude (e.g. Euclidean Distance) but more about orientation (e.g. preferences) between the two vectors.

©IJRASET: All Rights are Reserved

To produce a scalar through inner product multiplication, both vectors must be part of the same inner product space. Mathematically, the similarity of two vectors measured by the cosine of the angle between them. The range of Cosine Similarity is [0, 1] [12]. There is the formula of this geometric concept:

$$similarity(x,y) = \frac{\sum_{i\in I}^{n} xy * (rx,i - r\bar{x}) * (rx,i - r\bar{y})}{\sqrt{(rx,i - r\bar{x})^2 * \sum_{i\in a_{xy}}^{n} (ry,i - r\bar{y})^2}}$$

$$similarity(A,B) = cos(\Theta) = \frac{A * B}{|A| * |B|} = \frac{\sum_{i=1}^{n} a_i b_j}{\sqrt{\sum_{i=1}^{n} a_i * \sum_{i}^{n} b_j}}$$

Each of the segment's point, we subtract from the mean average value of the couple of coordinates. In the formula above, Θ is the angle between two vectors, *A * B* is the dot product, which is equal with the sum of unit vectors; |*A*| is the magnitude of the vector A. |*B*| is the magnitude of the vector B. Combining the similarity calculated from the scoring matrix and the similarity obtained from the commodity type, the calculation of the similarity will be more accurate [13].

D. Comparison between Cosine Similarity and Linear Kernel

We implement data for credits of movies in the Data Frame called movie-credits. Let us make a comparison between the matrices of the two files. The users are similar if their rating vectors are close according to a distance measure. For a similarity function with a range of *[0,1]*, if *s(x,y) = 1*,then *x = y*. This means that the larger the value of the similarity function, the closer the two objects are. Array can be multidimensional. All elements of an array need to be of the same type, e.g. all integers or all floats. They are computationally-memory efficient. There is a set of similar items. We take an item I from the set. All the items consisting of received user ratings. On the other hand, we have the user U, who has not rated anymore. We pick out N items form the similarity list to calculate the rating based on these ratings N. Suppose that we have (u, v) two vectors, where u is the vector for users who like adventure movies (8.6) and v is the vector for users, who like drama movies (8.1). The inverse of the distance, which implies higher similarity, should be the similar factor. We get cosine similarity subtracting the cosine distance from 1 [14].

E. Conditional Independence and Count Vectorization

There are three sets *X, Y, Z*. The variable *X* is conditionally independent of *Y* given *Z*, if and only if the probability distribution governing *X* is independent of the value *Y* given *Z* *∀(i, j,k) P(X = xi,Y = yj,Z = zk) = P(X = xi,Z = zk).*

Naıve Bayes provides a mechanism for using the information in sample data to estimate the conditional probability *P(Y / X)* of each class *Y* given an object *X* [15]. We make accurate predictions and probabilities about the hotels of Albania that are more visited by the tourists and most rated, last years, using conditional probabilities. We predict classes with accuracy and precision using Bayes. The data is presented in a matrix with dimensions [100 rows x 2 columns], by the code as below:

```
titles_hotel = tourism1['hotel']
indices = pd.Series(tourism1.index, index=tourism1['hotel'])
print(indices)
```

We set the research problem using a combination of mathematical and machine-learning knowledge. The Bayesian method is a technique to calculate conditional probability in Machine Learning. Naıve Bayes involves a series of probabilistic computations to find the best-fit classification for a given piece of data, such as thirty-eight movies of different genres. The Bayesian Theorem, given that an event B has occurred, defines the conditional probability of an event *A*. It is true. Also, is a set point in the set when *A* is a sample space and the equation is

$$P(A/B) = (P(B/A) * P(A))/(P(B))$$

The score matrix may be an *n-dimensional* score vector *P (U) = I1, I2 ,..., In.* If the value is 1(one), the user scores the commodity, if the value is 0 (zero), the user scores the un-commodity. Looking carefully at the target users *n-dimensional* score vector, it is simple to count the type of commodities and the number of times to put the statistical results into the list.

In the statement, *(i, n)* variable *i* expresses a type of commodity, and variable *(i, n)* expresses how many times the commodities have scored. Based on the number of scores for specific types in the list, we predict the user's interest in this type of product, according to the number of users scoring commodity. If the target get the commodity is *j*, then *N(j)* represents the scoring number of u on the *j* type and *M* is the total number of users to score commodities. We calculate the probability: *P (u, j) = N(j) / M.*

```
df1=pd.read_csv('tourism.csv')
df2=pd.read_csv('tourism.csv')
print(df2['places'].head(10))
print(df1['cleanliness'].head(10))
tfidf = TfidfVectorizer(stop_words='english', analyzer = 'word')
df2['places'] = df2['places'].fillna('')
tfidf_matrix = tfidf.fit_transform(df2['places'])
tfidf_matrix.shape
print(tfidf_matrix.shape)
count_matrix = pd.DataFrame(tfidf_matrix.toarray(), index=df2['places'].index.tolist())
print(count_matrix)
print(tfidf.get_feature_names_out()[940:990])
cosine_sim = linear_kernel(tfidf_matrix, tfidf_matrix)
indices = pd.Series(df2.index, index=df2['places']).drop_duplicates()
print(cosine_sim)
```

The result will be a matrix with [100 rows x 43 columns]

```
    0   1   2   3   4   5   6  ...  36  37  38  39  40  41  42
0   0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 0.0 0.0 0.0 0.0
1   0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 0.0 0.0 0.0 0.0
2   0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 1.0 0.0 0.0 0.0 0.0 0.0
3   0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 0.0 0.0 0.0 0.0
4   0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 1.0 0.0 0.0 0.0 0.0 0.0
..  ... ... ... ... ... ... ... ... ... ... ... ... ... ... ...
95  0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 0.0 0.0 0.0 0.0
96  0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 0.0 0.0 0.0 0.0
97  0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 0.0 0.0 0.0 0.0
98  0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 0.0 0.0 0.0 0.0
99  0.0 0.0 0.0 0.0 0.0 0.0 0.0 ... 0.0 0.0 0.0 1.0 0.0 0.0 0.0

[100 rows x 43 columns]
[]
[[1. 0. 0. ... 0. 0. 0.]
 [0. 1. 0. ... 0. 0. 0.]
 [0. 0. 1. ... 0. 0. 0.]
 ...
 [0. 0. 0. ... 1. 0. 0.]
 [0. 0. 0. ... 0. 1. 0.]
 [0. 0. 0. ... 0. 0. 1.]]
places
SPietro      0
Epidam       1
Taulantia    2
Epidam       3
Taulantia    4
             ..
Jonufer     95
Radhime     96
```

```
Lungomare    97
Dhermi       98
UjiFtohte    99
Length: 100, dtype: int64
[[1. 0. 0. ... 0. 0. 0.]
 [0. 1. 0. ... 0. 0. 0.]
 [0. 0. 1. ... 0. 0. 0.]
 ...
 [0. 0. 0. ... 1. 0. 0.]
 [0. 0. 0. ... 0. 1. 0.]
 [0. 0. 0. ... 0. 0. 1.]]
```

The first research question is: *What is the probability that the hotel will be chosen by the user is in top ten hotels?*
To response this question, we need to create another CSV file in Excel and then implementing the code, where we take as the main attributes: the values of the colons “hotels” and “stars” from the “tourism.csv” file and “names” with “average” from the second file “touristAL.csv”

```
df1=pd.read_csv('tourism.csv')
df2=pd.read_csv('touristAl.csv')
print(df1)
df1['hotel'] = df1['hotel'].str.split('|')
df1['hotel'] = df1['hotel'].fillna("").astype('str')
tourismAL = pd.read_csv('tourism.csv', sep=',', encoding='latin-1', usecols=['hotel', 'places'])
print(tourismAL)
tf = TfidfVectorizer(analyzer='word',ngram_range=(1, 2),min_df=0, stop_words='english')
tfidf_matrix = tf.fit_transform(tourismAL['hotel'])
tfidf_matrix.shape
print(tfidf_matrix)
survey_data = pd.crosstab(index=df2['names'], columns=df2['average'], margins=True)
survey_data1 = pd.crosstab(index=df2['hotel'], columns=df2['stars'], margins=True)
print(survey_data)
print(survey_data1)
print(survey_data.iloc[1, 0])
print(survey_data1.iloc[0, 3])
print(survey_data1.iloc[2, 3])
print(survey_data1.iloc[0, 3] / survey_data1.iloc[2, 3])
```

The result of the code is:

```
 (0, 195)          1.0
 (1, 36)           0.4335517635073323
 (1, 83)           0.4335517635073323
 (1, 118)          0.4335517635073323
 (1, 249)          0.32320220304868047
 (1, 33)           0.3528403423866311
 (1, 82)           0.4335517635073323
 (1, 114)          0.13847563745183833
 (2, 220)          0.6897346994818647
 (2, 219)          0.6897346994818647
 (2, 114)          0.22029999696169625
 (3, 135)          0.6897346994818647
 (3, 263)          0.6897346994818647
```

(3, 114)	0.22029999696169625
(4, 110)	0.5035343572790704
(4, 265)	0.5035343572790704
(4, 109)	0.5035343572790704
(4, 264)	0.4620539187107895
(4, 114)	0.16082794944493803
(5, 127)	0.6897346994818647
(5, 179)	0.6897346994818647
(5, 114)	0.22029999696169625
(6, 233)	0.7172109695273825
(6, 232)	0.6581281579338739
(6, 114)	0.22907586717975748
:	:
(91, 58)	0.5282045744986511
(91, 236)	0.366793590103216
(91, 114)	0.1687075715410871
(92, 137)	1.0
(93, 125)	0.6897346994818647
(93, 161)	0.6897346994818647
(93, 114)	0.22029999696169625
(94, 178)	0.6043230432231961
(94, 177)	0.6043230432231961
(94, 8)	0.5192179878036876
(95, 114)	1.0
(96, 218)	1.0
(97, 126)	0.6897346994818647
(97, 168)	0.6897346994818647
(97, 114)	0.22029999696169625
(98, 80)	0.6866159700943292
(98, 79)	0.6866159700943292
(98, 114)	0.23899167186922346
(99, 53)	0.4264100403366699
(99, 231)	0.4264100403366699
(99, 52)	0.4264100403366699
(99, 230)	0.4264100403366699
(99, 134)	0.3912829924453172
(99, 249)	0.3178782236381276
(99, 114)	0.1361945841802252

$$P(A/B) = P(A) * P\left(\frac{B}{A}\right) / P(A) * P(B/A) + P(\bar{A}) * P(B/\bar{A})$$

[44 rows x 24 columns]

staff	7.1	7.5	7.6	7.8	7.9	...	9.7	9.8	9.9	10.0	All
names			...								
Alex Bedand Breakfast	0	0	0	0	0	...	0	0	0	0	1
Amos Hotel	0	0	0	0	0	...	0	0	1	0	1
Aphrodite's Garden	0	0	0	0	0	...	1	0	0	0	1
Artemis Beach House	0	0	0	0	0	...	0	0	0	0	1
Azar Luxury Suites	0	0	0	0	0	...	1	0	0	0	1

	...	...	...	...	...	...	...	...	...	...	...	
Vila Panorama	0	0	0	0	0	...	0	0	0	0	1	
Villa Ester	0	0	0	0	0	...	0	1	0	0	1	
Yard Pardise Rooms	0	0	0	0	0	...	0	0	0	0	1	
Yatch Premium Hotel	0	0	0	0	0	...	0	0	0	0	1	
All	1	1	3	1	1	...	6	7	5	4	100	

[100 rows x 26 columns]

Count Vectorization is a way of considering the count of each word in the code [16]. A word with more counts is acceptable. In this paper, we use a combination of two types of Recommendation Systems, called Content-Based Filtering and Collaborative Filtering (CF). We have utilized a large amount of data in the past and predicts which items users will like using CF. It relies on the connection between the user and items, which encoded in a matrix. Each element of the matrix represents a specific user rating on a specific item. The Bayesian Naive Algorithm derived from Bayes' probability theory is one of the crucial algorithms in ML, which we have used, in our paper. Consider a Supervised Learning problem, in which we approximate an unknown target function *f: X→Y* or equivalently *P(Y / X).*We consider *Y* as a Boolean valued random variable and
X as a vector with n-Boolean attributes. *x = (x1,x2,x3, . . . ,xn).* Applying the rules of Bayes [17].

$$P(Y = y_i/X)$$
$$P(Y = y_i/X = x_i) = (P(X = x_k/Y = y_i) * P(Y = y_i))/ \Sigma_j P(X = x_k/Y = y_i)$$

F. Implementing get recommendation

We create the function genre-recommendation with the attribute title, because the code will predict ten best hotels as an output. Initiating the variable mx with the value of *signIndication = pd.Series(tourism.index,* index=tourism['hotel']. A robust method is *enumerate(),*which help cosine-similarity to record the number of times and return as similarity-scores. The next method is sorted which put one by one in a list ten hotels after compared them. In the dataset "tourism" we create a column to each word of the dataset "tourism". If the word is in the title, we give it1, if it is not, and then is zero (0).

```
def hotel_recommendations(hotel):
    idx = indices[hotel]
    sim_scores = list(enumerate(cosine_sim[idx]))
    sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)
    sim_scores = sim_scores[1:21]
    tourism_indices = [i[0] for i in sim_scores]
    return titles_hotel.iloc[tourism_indices]
 print(hotel_recommendations('Casa De Sol').head(20))
```

After we finish implementing the code, the result will be ten hotels:

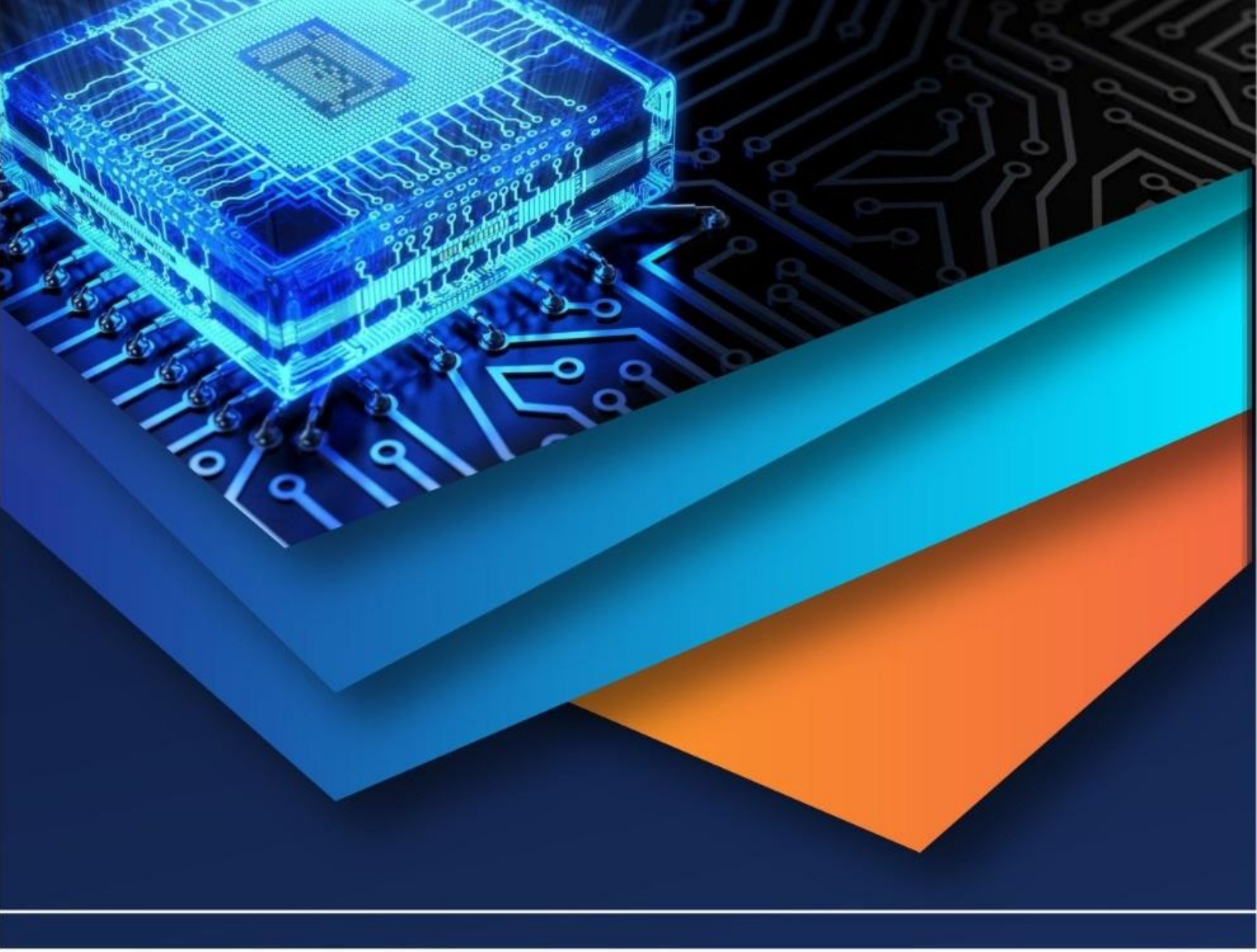
77	Casa De Sol
84	Green Coast Resort and Residence
90	Ulcas Gues tHouse
94	Maison Apartment
0	Melia
1	Hotel Epidam Botique SPA
2	Peaky Hotel
3	Hotel Veliera
4	Ventus Harbor Hotel
5	Hotel Majestic

III. CONCLUSIONS

Through a Mathematical Model we have created a simple Recommendation System as a type using data, based on our knowledge of probability, statistics, linear algebra, calculus, geometry. This model recommends to the user a list with ten hotels of Albania, from one hundred and one in total. We have used content-based filtering to create the code and to choose the movies based on the preferences and ratings of users. This model will help users interact with the page quickly and not take longer to find what they are looking for. With this code, we offer tourists the opportunity to get to know ten Albanian hotels that are at the top of the classification. As we also invite tourists to come and spend their holidays in the tourist spots of Albania. We also promote the values of Albanian tourism abroad.

REFERENCES

[1] H. Herrmann, What's next for responsible artificial intelligence: a way forward through responsible innovation, J.Helyon (2023).
[2] M. C.-T. Tai, What is next for responsible artificial intelligence: a way forward through responsible innovation, Tzu Chi Medical Journal (2020).
[3] L. Jianguo, Research progress of personalized recommendation system. adv. nat. sci., EURASIP Journal onWireless Communications and Networking (2009).
[4] SonuAiren, Movie recommender system using parameter tuning of user and movie neighbourhood via coclustering, Procedia Computer Science (2023).
[5] Q.Zhang, Artificial intelligence in recommender systems, complex intelligent systems, SpringerLink (2020).
[6] F.Hdioud, Multi-criteria recommender systems based on multi-attribute decision making
[7] W. McKinney, Python for data analysis, O'Reilly (2017) 1.
[8] Ch.Hill, Learning scientific programming with python, Cambridge Core (2020)
[9] J.Hunter, Matplotlib user guide,
[10] Tutorial point. simply easy learning, tutorials point (2019) .
[11] R.Manjula, Content based filtering techniques in recommendation system us-ing user preference,
[12] E. Schubert, A triangle inequality for cosine similarity, arxiv (2021)
[13] A. Sh.Tyagi, A machine learning based fashion recommendation system, Research Archive of Rising School (2023) 13.
[14] F. Richi, Introduction to recommend system handbook, book chapter, International Journal of Innovations in Engineering and Technology (IJIET) (2011) 35.
[15] A.Wibawa, Na¨ıve bayes classifier for journal quartile classification, International Journal of Recent contributions from Engineering, Science and IT (2019).
[16] K. Kuter, Math 345 – probability, LibreText (2023)
[17] M. F.Triola, "Bayes'Theorem". "University of Washington".



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)