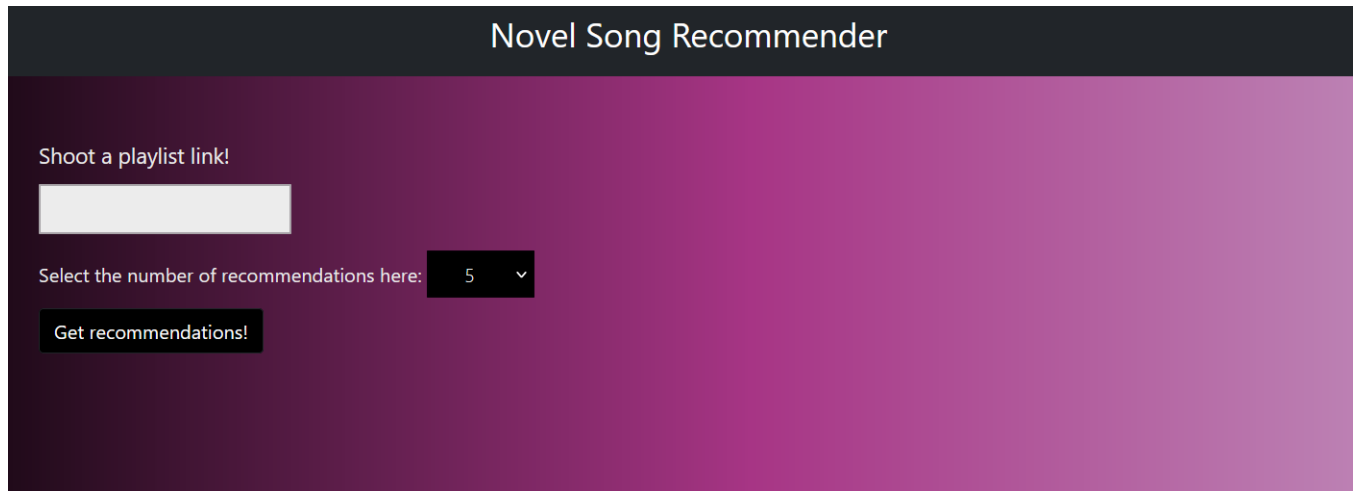


# Novel Song Recommender

*Music is the moonlight in the gloomy night of life*

Ansh Sachdeva  
ansh.sachdeva@colorado.edu  
University of Colorado  
Boulder, Colorado, USA

Shivam Vats  
shivam.vats@colorado.edu  
University of Colorado  
Boulder, Colorado, USA



**Figure 1.** User interface for recommendation system

## Abstract

In every area of industry, whether it is retail, telecommunications, online platforms, etc., recommendation systems have become quite important. The music industry is one such area where this system is extensively utilized. Based on the qualities of the music that has been heard in the past, the music industry is capable of predicting and then presenting suitable songs to its consumers. In this project, we have used a sample dataset of songs collected using Spotify API to analyze user and song similarities in order to suggest new songs to users based on their listening history. Additionally, we created a flask application that allows users to input their playlists and receive more music recommendations in return.

**Keywords:** datasets, content-based filtering, term frequency-inverse document frequency, Cosine Similarity, Cluster Analysis, Sentimental Analysis

## 1 Introduction

The internet has grown to be a major source for obtaining multi-media information, including movies, text, songs, and other media, as a result of the daily growth in the number of users on online platforms. People consider music and movies to be important aspects of their lives, and they frequently

listen to music and watch movies on various platforms like Spotify and Apple Music, thus various people occasionally believe that it is difficult to choose from a vast collection of music. The availability of digital music is currently growing quickly thanks to music streaming services that can be accessed through mobile devices. Providers of musical services require a productive way to manage songs and assist their consumers in discovering music by making pertinent and correct recommendations.

Recommendation systems are important in today's world and how organizations are utilizing its power to maximize user engagement on their platform. Here we will examine how recommendation systems have reduced the time-consuming processes that used to be performed by a person. Recommendation systems utilize an automated information filtering process to suggest a product that a user prefers. It focuses on finding and delivering information that the user is likely to find useful or interesting. By filtering the data source and providing the users with pertinent information, it helps users.

Collaborative filtering and content-based filtering are two methods for obtaining the results in order to achieve accurate recommendations. While collaborative has been selected because it produces the best results when it collects data on user and item and does vectorization. However, because it is well-liked and simple to set up at first, content-based filtering is also widely employed.

**Collaborative Filtering System:** The features of the objects do not need to be specified in the alliterative. Every user and item has a feature vector or embedding that describes them. It generates embedding on its own for both users and goods. Both users and items are embedded in the same embedding space. It notes which goods a specific user enjoys as well as the items that other users who share his or her preferences recommend to that user. It gathers user feedback on various goods and utilizes it to make recommendations. The three subcategories of combinatorial filtering are memory-based, model-based, and hybrid combinatorial filtering.

**Content-Based Filtering System:** Items are recommended by a content-based filtering system based on their characteristics and the similarity of their components to those of the other metrics or features. A content-based recommendation engine uses data that users usually provide, while interacting with the application such as reviews/ratings or by clicking songs. A user profile is created using this data and is then used to recommend options to the user. In order to compute and identify the song that is most comparable to the user's input, our recommendation algorithm makes advantage of all song characteristics, including genre(pop, classic, hip-hop, etc.), audio characteristics (acousticsness, danceability, loudness, etc.), and the metadata. Each song in our corpus will have a similarity score with every other song in our dataset, and we will calculate the similarity value using cosine similarity which basically explains the similarity between two vectors which is songs. The value for similarity ranges from -1 to 1, -1 being the opposite and 1 being the most similar.

## 2 Related Work

Apart from content-based filtering collaborative filtering is widely used in personalization systems. J. Ben Schafer et. al explains how Collaborative filtering evaluates items through the opinions of other people and how it brings together the opinions of large "interconnected" communities on the web, supporting filtering of substantial quantities of data.

There is a large body of work which uses the Spotify playlist records to analyze songs data and build deep learning models over the baseline methodologies of content based and collaborative filtering. Aaron Van Den Oord et. al. optimizes

the performance of collaborative filtering by tackling the problem of cold start, that is, absence of data for producing recommendations by leveraging deep learning techniques.

## 3 Data Collection, Modelling and Cleaning

### 3.1 Data Collection

After contemplation about various data sources available for songs corpus we decided to leverage [Spotify Web API](#) to maintain the authenticity and accuracy of data. To use the web API in python environment, we leveraged the package named [Spotipy](#).

While it provides a set of handy functions for corresponding web API methods, there exists bottlenecks that shall not be overlooked. Spotify Web API implements throttling that was the major roadblock while performing data collection. To overcome that problem, we developed scripts with a special purpose of performing queries in batches. This enabled us to collect the data efficiently without hitting the throttling limits.

### 3.2 Data Modelling

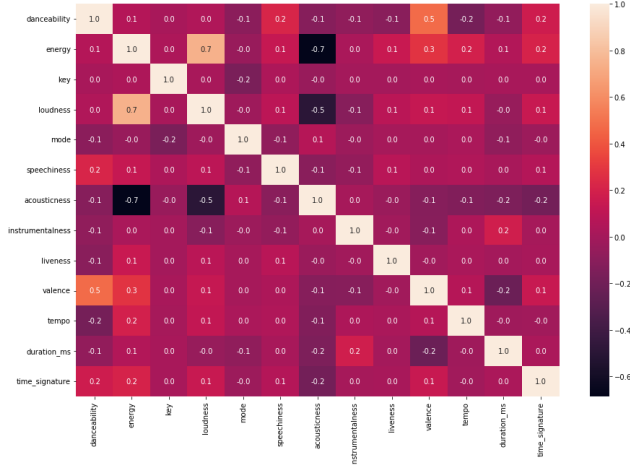
The data is a collection of three different entities: tracks, artist and audio features. Data for each entity is fetched using different methods. After collection, the data is stored in Postgresql. Leveraging the relational database management system provided by Postgresql, each entity is modelled to create relations among each other. Modelling the data gave an upper hand to perform multivariate analysis. Although Postgresql is a self sufficient tool to perform data analyses, we used Tableau in conjunction. This is because Tableau is easier and faster to use and create dashboards and visualizations.

### 3.3 Data Cleaning

Data cleaning went hand-in-hand during the data collection and data modelling phases. The data returned by Spotify API was in format of strings, hence, we transformed it into required data types before storing it in relational database (Postgresql). Additionally, some entries of an entity were missing and it affected the relations in the entity existed. For example, due to some missing records in the artists entity, the relation created with the join of artist and track resulted in some null entries. To tackle the same, we performed the relevant imputations on entities.

## 4 Analysis of Audio Features

One of the most challenging parts of developing a recommendation system is that it creates unseen predictions. Hence



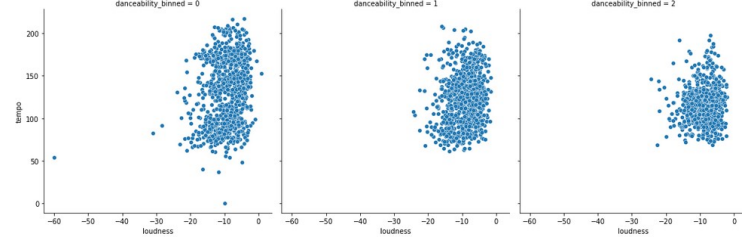
**Figure 2.** Heatmap representing correlation of audio features

it becomes difficult to evaluate its performance. To avoid the problem of under-performance, it is necessary that the features upon which the system is built are true representatives of the users' preferences. To corroborate the choice of features, we have performed exploratory data analysis in form of cluster analysis and bi-variate analysis leveraging Seaborn, Matplotlib and Tableau. Before diving into the analysis, it is wise to observe how the numerical features are related to each other using the measure of covariance.

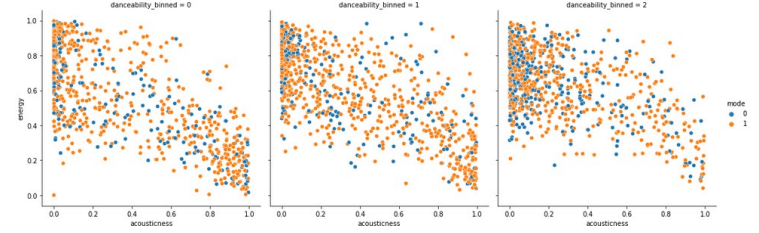
#### 4.1 Feature Relationships

We can observe high correlation among multiple features and focused upon those features in the further analysis. In one of the instances of our analysis, we discovered a pattern in "danceability", "tempo" and "loudness". With increase in the measure of "danceability", the "tempo" and "loudness" of songs followed a positive trend in their respective measures. Using the Gaussian kernel density estimate, we found that these features are not normally distributed, but skewed in either direction. Hence, directly visualization will not reflect their true nature. To overcome this hurdle, we used stratified sampling in order to get a uniform number of entries of each sub-level in a feature. Here is the illustration of the same pattern using the Seaborn library:

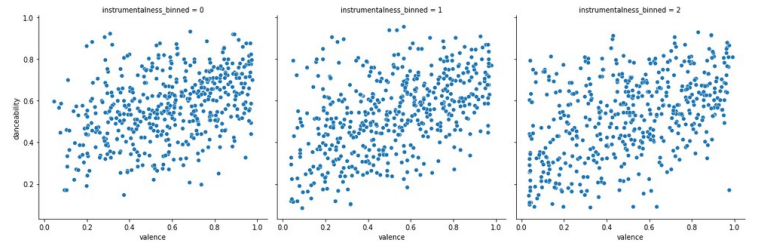
You shall observe how the cluster of points are getting cluttered towards higher "tempo" and "loudness" as the measure of "danceability" is increasing. This is one of the few patterns which we observed. For the rest of the patterns we've used the same sampling technique to maintain uniformity. Other patterns that we identified to be significant were between the features "energy", "acousticness", "valence"



**Figure 3.** Tempo vs. Loudness with respect to Danceability



**Figure 4.** Energy vs. Acousticness with respect to Danceability

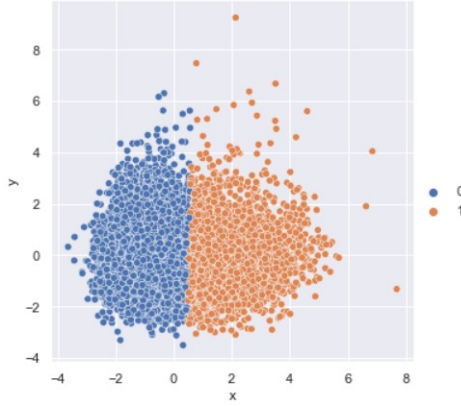


**Figure 5.** Danceability vs. Valence with respect to Instrumentalness

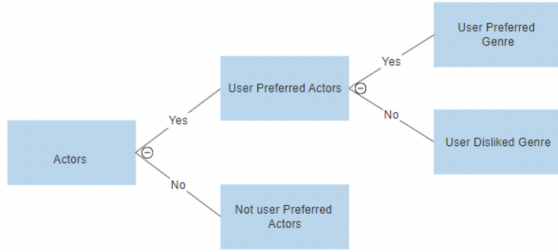
and "instrumentalness":

#### 4.2 Clustering

Clustering is one of the most used techniques in the domain of exploratory data analysis and unsupervised learning. According to Mahamed G.H. Omran et. al. Clustering is the process of grouping similar objects into different groups, or more precisely, the partitioning of a data set into subsets, so that the data in each subset according to some defined distance measure. We leveraged different clustering algorithms such as KMeans, KMedoids and DBSCAN(density-based spatial clustering of applications with noise) to unveil patterns among features. To visualize the results of clustering, we transformed the data into two dimensions using the technique of Principal Component Analysis(PCA). The transformed did not exhibit any cluster segments that could have been a subject of analysis, but it formed a single big cluster.



**Figure 6.** KMedoids classification on transformed data



**Figure 7.** Content Based Filtering process

## 5 Content Based Filtering

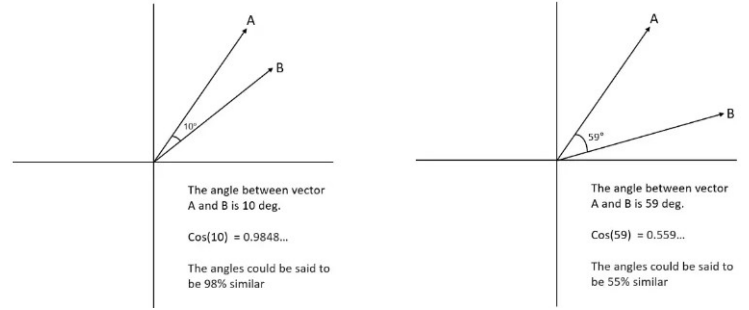
A recommendation engine that is based on content leverages information that users typically supply while interacting with the application, such as minutes listened, favourite genres, artists, tracks, etc. This information is used to build a user profile that will be used to present alternatives to the user. Our recommendation system utilizes all song attributes, audio characteristics, track titles, artist names, and the meta-data in order to compute and select the music that is most similar to the user's input. When we receive user-submitted songs, our model compiles a list of the features they contain, performs Count Vectorization, TFIDF, and generates a score. This score is then applied to a fresh batch of songs to produce recommendations based on the similarity scores between those songs and those in the corpus.

### 5.1 Cosine Similarity

Cosine Similarity is a method to capture similarity between instances of an entity. The cosine similarity is the cosine of the angle between vectors. The vectors are typically non-zero

$$S_C(A, B) := \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}}$$

**Figure 8.** Cosine Similarity Formula



**Figure 9.** Cosine Similarity intuition

and are within an inner product space. The cosine similarity is described mathematically as the division between the dot product of vectors and the product of the euclidean norms or magnitude of each vector.

We implemented the cosine similarity metric on the songs present in the user's playlist with the song corpus. Let me explain how two vectors are said to be similar in comparison to others.

### 5.2 Text Blob

Natural language processing is a broad field that is frequently used to textual data or sentiment analysis. It is clear that with more research in NLP area, a great deal of analysis and text computation can be accomplished with ease. NLP has many real-world applications and has a wide range of potential study topics, from simple tasks like text parsing to complicated ones like performing sentiment analysis on a dataset.

There are numerous packages and libraries provided in Python that can be used to determine the sentiments of text. To determine whether a song is positive, negative, or neutral, we performed sentimental analysis on the track names using TextBlob. Sentiment analysis can assist us ascertain the type and tone of the music as well as acquire important data about the songs.

We used subjectivity and polarity to categorize the sentiment as positive, negative, or neutral. With -1 signifying

a negative tone and 1 denoting a positive tone, TextBlob returns the polarity value from -1 to 1. 0 is a neutral value. Hidden emotions are also uncovered by TextBlob's semantic analysis. The subjectivity scale goes from 0 to 1. Subjectivity is the degree to which a text, in this case the track name, contains both opinion and other information. When provided an input or text, the TextBlob's sentiment algorithm generates results with polarity and subjectivity scores. A statement is given a subjectivity score between 0 and 1, where 0 denotes an objective observation and 1 denotes a subjective assertion.

For instance, when we analyzed polarity and subjectivity for the track "Good Vibrations (Mono)," the results showed that the polarity was positive and the subjectivity was high, both of which are easily understood as indicating a good tone in the track.

However, for a song called "Smells Like Teen Spirit," the polarity was Neutral and the subjectivity was 0.2 (low). This might have happened because the term "Spirit" seems to give the song a negative tone.

### 5.3 Term Frequency and Inter-Document Frequency

A statistical measure called TF-IDF (term frequency-inverse document frequency) assesses a word's relevance to a dataset within a collection of datasets. To do this, multiply the inverse document frequency of the word over a group of datasets by the number of times it appears in a dataset. It is highly useful for evaluating words in machine learning algorithms for Natural Language Processing and has a wide range of applications, with text analysis being the most essential one

TF-IDF is calculated for each word in a document by multiplying two separate metrics:

**Term Frequency:** It is determined by performing a simple word count within a document. The length of a document or the frequency of the word that appears the most frequently in a document are other ways to modify frequency.

**Inverse Document Frequency:** This indicates how prevalent or rare a word is within the full collection of documents. A word is more common the nearer it is to 0. The logarithm of the total number of documents is divided by the number of documents containing a term.

By creating a TF-IDF vectoriser, which created values correspondingly describing the importance of the genres for various tracks in the dataset, we used TF-IDF on genre and other track attributes.

$$w_{i,j} = tf_{i,j} \times \log \left( \frac{N}{df_i} \right)$$

$tf_{i,j}$  = number of occurrences of  $i$  in  $j$   
 $df_i$  = number of documents containing  $i$   
 $N$  = total number of documents

**Figure 10.** TF-IDF Formula

By comparing "the words used in a sentence vs words used in the total document" or the TF-IDF, we can determine:

- How significant a word is in a sentence.
- The key phrases that appear more frequently in a paper
- Aids us in ignoring misspelled words

## 6 Collaborative Filtering

A popular recommendation algorithm called collaborative filtering (CF) based its forecasts and suggestions on the evaluations or actions of other system users. The underlying premise of this approach is that it is possible to pick and combine the opinions of other users in a way that yields a precise estimation of the active user's choice. They use the intuitive assumption that if people concur on the value or applicability of some items, they will likely concur on other items as well. For example, if a group of users has a similar taste in music as User 1, User 1 will receive recommendations for songs they haven't heard.

We can obtain an accurate estimation or recommendation by applying a variety of similarity functions while computing the recommendation using collaborative filtering

- Pearson correlation.
- Constrained Pearson correlation.
- Spearman rank correlation

The term "collaborative" in collaborative filtering refers to the likelihood that a user would receive recommendations from another user whose listening habits are similar to user1's, so that it makes use of the power of accurately recommending songs by collaborating two or more users.



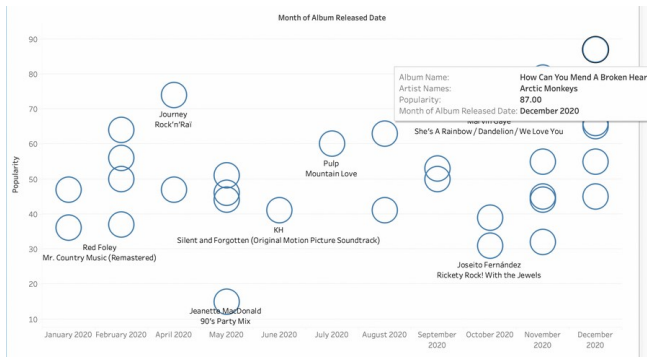


Figure 11. Popular songs in Year 2020

## 7 Postgresql

Postgresql is one of the most used relational database management system in data science and machine learning applications. This is because of its capabilities to support online analytical processing (OLAP) as well as online transaction processing (OLTP). In addition, it provides support for processing unstructured data. We've used it as our main repository for data storage and analysis to determine the connections between various entities, such as between artists and songs and tracks and their features. With Postgresql, we were able to connect straight to Tableau and ensure that our data input for visualization and additional analysis flowed smoothly. The parallel computing feature of Postgresql allowed us to run "big" queries while doing exploratory data analysis.

## 8 Tableau

Tableau is an effective data visualization solution that helps individuals find answers more quickly and discover unexpected insights. We gained deep knowledge about the trends among the artists, tracks in relation to their properties like genres, audio attributes, etc. from the data's visual depiction. We used cluster analysis using tableau as our visualization tool to find hidden patterns, relationships between track parameters (such as loudness increasing as the song's energy grows), and information about the most popular artists over time. The reason for using Tableau apart from Seaborn is its capabilities of complex data blending, computations and dashboarding features.

## 9 Flask

Flask[reference here] is a framework that uses Python language with easy to understand code writing. Flask provides a library and a collection of codes that can be used to build websites, without the need to do everything from scratch. Because of its simple features, the flask will be lighter and not dependent on many external libraries that need attention.

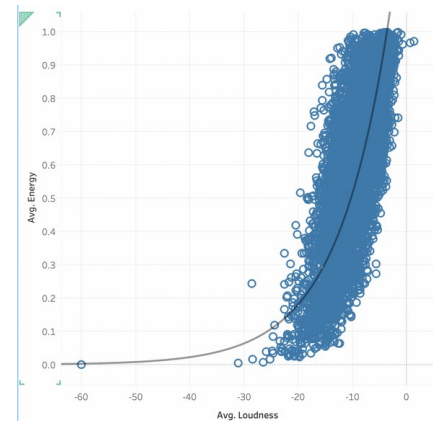


Figure 12. Avg Energy Vs Avg Loudness

Flask makes use of the Werkzeug WSGI ToolKit and the Jinja Template Engine. Template files contain all the Jinja templates, including HTML pages, whereas static files contain all the static code required for websites, such as CSS code, JavaScript code, and image files. The reason we used Flask is because of its incomparable flexibility. Flask enables us to diversify the structure of our application. Since we'll be continuing to add new features to the application and we're not sure of the specific libraries we'll be using in future, opting to perform web development will not be a big obstacle to come over given Flask's wide range of micro-services and stack of its intern libraries.

## 10 Future Work

There are multiple developments possible that will enhance the user experience and boosts the relevance of recommendations. The first and foremost improvement we envision is data. We currently use data from a large playlist containing around 10000 records of a english language. This corpus can be extended to include equal number of songs from different genres. Other than genres, songs from multiple languages shall be included to improve robustness of recommendations. Apart from songs data, inclusion of users' data will empower us to use collaborative filtering in conjunction to the content based filtering. Further, we seek to add more features in the user interface of the application, where user shall be having the power to filter the type of recommendation they are looking for. As an example, we shall add a "language" switch, to filter out all the recommendations that are not same as the user's input, making the application more personalized.

## 11 References

- [1] Muhamed Omran, Andreas Engelbrecht, Ayed A. Salman, 2007. An Overview of clustering methods. *International Journal of Computer Science & Information Technology (IJCSIT)* Vol 8
- [2] Schafer, J Ben and Frankowski, Dan and Herlocker, Jon and Sen, Shilad, 2007. Collaborative filtering recommender systems. *The adaptive web*, Springer
- [3] Van den Oord, Aaron, Sander Dieleman, and Benjamin Schrauwen. "Deep content-based music recommendation." *Advances in neural information processing systems* 26 (2013).
- [4] M. R. Mufid, A. Basofi, M. U. H. Al Rasyid, I. F. Rochimansyah and A. rokhim, "Design an MVC Model using Python for Flask Framework Development," *2019 International Electronics Symposium (IES)*, 2019, pp. 214-219, doi: 10.1109/ELECSYM.2019.8901656.
- [5] P. Nagarnaik and A. Thomas, "Survey on recommendation system methods," *2015 2nd International Conference on Electronics and Communication Systems (ICECS)*, 2015, pp. 1603-1608, doi: 10.1109/ECS.2015.7124857.
- [6] Florian Endel, Harald Piringer. Data Wrangling: Making data useful again. *IFAC-PapersOnLine*, Volume 48, Issue 1 2015, Pages 111-112
- [7] Kim, SW., Gil, JM. Research paper classification systems based on TF-IDF and LDA schemes. *Hum. Cent. Comput. Inf. Sci.* **9**, 30 (2019). <https://doi.org/10.1186/s13673-019-0192-7>
- [8] Qaiser, Shahzad & Ali, Ramsha. (2018). Text Mining: Use of TF-IDF to Examine the Relevance of Words to Documents. *International Journal of Computer Applications*. 181. 10.5120/ijca2018917395.