Team 17

Introduction

The goal of our project is to provide a novel data visualization tool for movie recommendation. Traditionally, movie recommendation is done using collaborative filtering in which movies are recommended based on what other similar users watch. Our approach deviates from this by providing information on the content-based similarity of various movies, utilizing attributes such as popularity and budget. We expect this tool to be valuable as it allows users to maintain privacy over what movies they have watched and provides less biased results.

Problem Statement

We aim to create a movie recommendation tool with visualizations of relative similarities between movies. In this visualization tool, we will show the relative similarities of the movie content per genre. In addition, for a given movie, we provide visualizations of the most similar movies.

Survey

The first group of papers we provided a survey of movie recommendation approaches. Iliopoulou et al. classifies movie recommendation systems into collaborative-filtering and content-based filtering. This is related to our project because our project will involve recommending movies to users without knowledge of their movie preferences, limiting us to content-based approaches. Their results from both approaches highlight that K-nearest neighbors is a viable approach towards the problem [5]. Jayalakshmi et al. provides a survey of the research being done in movie recommender systems and provides areas of improvement across different methodologies. The paper suggests that clustering movies into groups based on distance to the centroid of the data distribution will provide rich visualizations [6]. Katarya et al. proposes a recommendation system built upon an optimization algorithm called cuckoo search. This paper is useful because it provides an alternate algorithm that could inspire some features of our project [7]. The second group of papers we analyzed involved principal components analysis for dimensionality reduction of data. While paper [9] had a heavy reliance on a user rating matrix, references [3] and [15] provided applicable research into utilizing the PCA algorithm. Βοζαλής et al. highlights tuning algorithms for PCA to optimize recommendations [3], while Sun et al. uses PCA alongside deep learning algorithms on historic watching data to provide a recommendation [15]. This suggests the possibility of using one movie to generate recommendations. Additionally, we dove into the second step of our analysis of using the 3D feature vector from PCA to perform KNN for movie recommendation by using its embeddings. This is useful for our project because it helps illustrate how to extract the necessary features for clustering. We will improve and modify the linearity of the Euclidean distance used in the paper to account for bias [18]. We were also interested in exploring how the k-nearest neighbor classification performs on a heterogenous dataset that is made of both numerical and categorical features. This is advantageous for our project because we were thinking about seeing how we can use the title and specific critic reviews in recommending movies. We plan on improving on this by placing a weighted emphasis on numerical compared to categorical features [2]. We also looked at different approaches to KNN using developer recommendation and expertise ranking to retrieve better results for developer recommendation for bug resolution. This paper is useful because it provides a more accurate, intuitive approach to ranking similarity versus traditional multi-label classification methods. We can improve on how some of the meta-fields are the same among developers in this paper introducing bias in their findings [16] . We explored the K-Means Clustering to appropriately classify movies to their appropriate genres. This is applicable to our project since after the preprocessing we can create a matrix to display the statistic on how useful a word is and run K-means on the matrix. One limitation to this paper is the hard-clustering on genres used which forces movies into 5 groups, where we are looking for a more fluid approach to grouping movies [13]. Paper [17] also conducts clustering analysis on movie recommendation data and shows similar results. We also looked into a paper that uses K-means clustering and the elbow method to identify the best customer profile cluster. This paper is useful to us since typically k-means struggles to find the ideal number of clusters, however, this can be mitigated by using the elbow method to find the best number of clusters on the k-means. We would improve upon the customer segmentation aspect by using the NLTK rather than raw data mining. [11] Paper 14 extended the K-means algorithm to analyze cliques of movies, which is an interesting visualization.We also explored two algorithms like K-nearest-neighbor and K-means clustering performed in the context of movie recommendations. This is directly applicable to our project with the exception of the interactive capabilities [12]. One mechanism which content-based recommender systems use is natural language processing (NLP). Farinella et al. uses tf-idf vectorization and LSA semantic analysis on movie description to analyze semantic similarities in the text. This is highly applicable to our project as we can use NLP on the plot keywords or the movie descriptions from the tMDB dataset to generate meaningful numeric features prior to clustering [1]. After processing the text to construct a numeric feature vector for each movie, Li et al. highlights a hierarchical clustering method which can be used on a distance matrix that can recommend movies based on various movie features rather than comparisons to other individual's recommendations [8]. Additionally, we looked into a visualization technique called "t-SNE" to pair with PCA to visualize high-dimensional data. This technique is relevant to our project since we want to study how this technique could be used to visualize the high-dimensional movie data we are analyzing [10].

Key Innovations:

Our first key innovation is that we use **PCA** to map features associated with the movie, as opposed to user-movie relationships, to a smaller latent embedding that is then used downstream for both movie clustering and movie recommendation. Then we perform **K Means** clustering on the latent embeddings stratified according to genre of movie to obtain a genre-specific grouping of movies that can be explored. We use **UMAP** as a rich visualization tool for the feature embeddings representing the groupings of movies along genre. We also provide a tool for movie recommendation that uses **KNN** to predict the top ten most aligned movies based on the obtained PCA embedding. These are validated by showing the rating on aggregate of all the movies relative to actual rating of the searched movie. Our novel approach uses a combination of these four tools on a javascript-based **D3 interactive visualization** to allow users to see movies clustered by their corresponding features and search for a movie to get recommendations of similar content.

Methods:

Our approach consists of performing multiple machine learning algorithms coupled with a visualization tool for the results. Specifically, we will perform K Nearest Neighbors and K-Means algorithms on the TMDB movie dataset. We will also perform Principal Component Analysis on the data before running the aforementioned algorithms. The goal of our approach is to generate the movies that are "nearest" to a certain movie or in the same cluster as it. Our approach to visualization will leverage the predictions of our machine learning models. Specifically, when a user types in a movie they will be able to see the recommended movies. This two-pronged approach combines the mathematical power of machine learning with the ease of visualization to provide a seamless interactive experience for a user.

Our approach is new because it will recommend movies specifically on the basis of movie contents and attributes. This is in contrast to existing movie recommendation systems which are done using collaborative filtering [5], which makes automatic predictions about the movies that a user may be interested in by collecting preferences or taste information from many users. Our approach is also unique in that we operate on the assumption that we are recommending movies without access to knowledge of user preferences towards movies. Thus we are operating on more limited assumptions than existing movie recommendation methods.

Our composition of techniques is also new in that we leverage PCA to map a large set of features associated with the movie to a smaller latent embedding that is then used downstream for both movie clustering and movie recommendation per a given prompt. The use of PCA has been studied on collaborative-filtering based approaches with identifying the principal components on a user rating matrix [3, 15] but has not been studied before in previous content-based approaches [6]. We will perform KMeans movie clustering using the embeddings obtained from PCA, thus getting more meaningful clusters around more important features in the data. We also perform K Nearest Neighbors prediction to obtain the top 10 recommended movies validated by the estimated rating of the movie in comparison to the actual IMDB rating.

Principal Component Analysis is a dimensionality reduction technique, meaning it finds what features or attributes of the data are most important / related. For example, two features of the data could have the same effect on the final predictions. If this is the case, then having both features is redundant. The effect of PCA is that we could go from say 30 features to 10 or so. Having less features will increase the effectiveness of the other algorithms. The K-Nearest-Neighbors algorithm will use the PCA cleaned data to compute "distances" between different movies. The algorithm will represent a movie as a point in space and find its nearest neighbors.The end result of K-NN is that we will have a list of similar movies for each movie. The K-Means algorithm is similar except it generates different clusters based on the data. Each cluster has a prototype or centroid which we update on every iteration. For each movie, we compute the distance to each cluster and update the centroid of a cluster as the average of all its constituents. Eventually, we will end up with each movie being a member of one of the clusters. For example, a user can enter a movie, then see its nearest neighbors and also its cluster information.

The use of content-based filtering instead of collaborative filtering provides less biased results because no data from other users is required and recommendations are based on matching characteristics to other movies. Initially, Principal Component Analysis would benefit us by improving performance at a low cost, as well as producing uncorrelated features of data. PCA can help solve our problem of determining which features change together and are correlated by calculating a similarity matrix in Euclidean space. For our purposes similar features between movies that result in identical results would reduce those features, which would improve accuracy for other algorithms used including K-NN. K-NN can help solve our problem by finding the appropriate neighbors by filtering through the dimensionally reduced features to conduct predictive classification on individual data points. Through this method, we will be able to find similar movies surrounding the respective centroid data points and be able to group them into genres based on similarities. Along with this, K-Means would help solve our unsupervised learning problem because we do not have a single target to predict, but instead we try to group similar features and groups. Intuitively, after every iteration of K-means the sum of distances to the chosen center is reduced, so we are guaranteed to have a group of movies that can converge to a certain cluster depending on the user's selected movie genre. The implicit use of Euclidean pairwise distances to the centroid best fits our problem because the sum of pairwise squared Euclidean distances divided by the number of points gives us a convergence that can be specialized to various clusters.

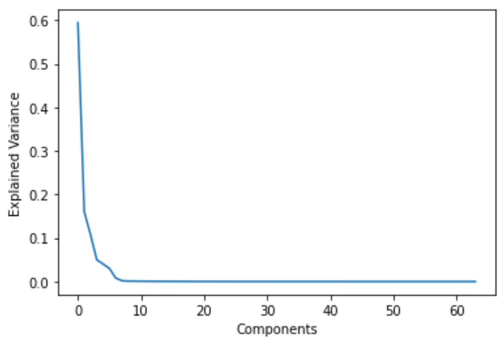
Our approach also lends itself to a novel visualization approach where we take the movie dataset transformed along the principal components according to our PCA transformation and visualize that embedding. The method we use is UMAP which has the capacity to visualize the latent embedding from PCA which is of dimensionality greater than three onto a two-dimensional plane which we color the nodes according to our K Means clustering outputs. UMAP has the benefit of being able to visualize large datasets quickly and better preserve the data's structure as compared to t-SNE, commonly used in visualization of high-dimensional data [10]. We build out the interactive visualization using d3 in javascript and allow the user to hover over the UMAP visualization to see the top 10 recommended movies for corresponding highlighted selection. Our method proposes a novel process for visualizing movie data which uses cluster labels obtained from K Means stratified along movie genres to color a mapping of our latent embeddings into a smaller 2-dimensional space done by UMAP. Alongside the movie cluster visualization by genre, we also have a recommendation tool that allows users to input their favorite movie title and the visualization showcases the top 10 closest movies. Our visualization showcases the key criterion as to why our K Nearest Neighbors recommendation displayed the predicted movie when showing the movie recommendations. This adds interpretability to our approach and allows the user to understand why the algorithm recommended those movies.

Experiments / Evaluation

The data engineering experiments aim to construct the best feature embeddings which can be used for movie similarity and movie clustering. The data clustering experiments seek to identify clusters of movies according to genre that best group together movies with similar attributes. The recommendation experiments attempt to find the top ten most similar movies to recommend based on the metadata and predicted rating of the movie. The visualization experiments aim to build interactive graphs that enable the user to best understand the relative similarities between all of the movies in the dataset.

The data that we got was the TMDB 5000 Movie Dataset, which we downloaded from Kaggle. This dataset contains two CSV files. The first CSV, called movies, is a file containing specific information about each movie including the budget, genre, keywords, popularity, production company, etc. There are a total of 20 unique columns in this dataset and there are 5000 movies that this information is provided for. The second csv, called credits, information revolving around the credits of the movie including who the cast included and the crew members that were a part of each movie. The credits csv file also had 5000 different movies that matched up with the 5000 movies found in the first csv file. The size of the movies csv on disk is 5.7 MB and the size of the credits csv on disk is 40 MB. The dataset is not temporal as there are movies in the dataset that span multiple different time periods.

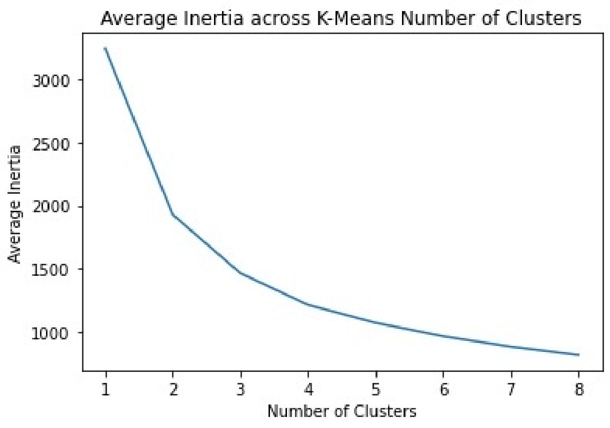
While we were able to download the dataset there was still a lot of data cleaning that had to be done, to turn the data into a format that could be easily parsed by our machine learning and visualization tools. First we loaded the two csv into data frames and converted the JSON to strings for five different columns: genres, keywords, production_companies, cast, and crew as these columns had a json list of elements for every cell. Once this was completed for both csv files we merged the two dataframes based on the id in order to get a merged csv file that had all the columns we needed. We filtered the columns based on the columns that we wanted and also printed out the columns that our final dataframe had as well as the data type for each column to understand what kind of types we would need to read in for our data analysis and visualization. We evaluated our approach by implementing PCA on our preprocessed movies dataset using 64 features ranging from budget and popularity to language and genre. We plotted a curve showing the explained variance relative to the number of principal components we select in our PCA transformation. These results are captured in the curve below which shows that using above the top 7 / 8 components we are able to explain almost all the variance shown across the features in the dataset.



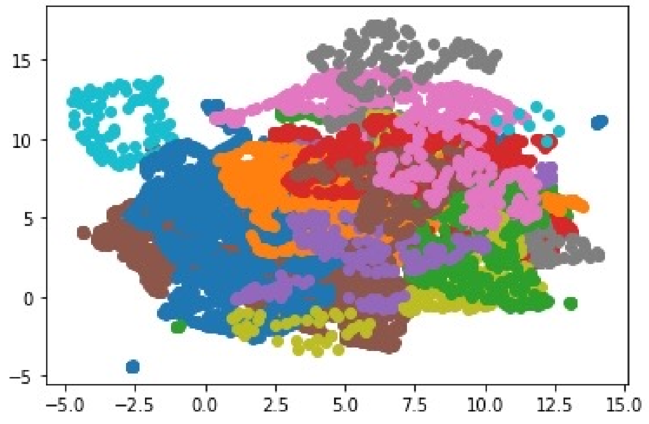
We can use the elbow method along this curve and thus choose to select the top 8 components to be used in our PCA transformation. Our method of using PCA allows us to learn a meaningful embedding of our large set of features across all our movies into a compact representation. This is as opposed to other approaches which cluster across all features and thus weigh them all equally or do not weigh by the most important features (i.e. the principal components).

We can then use the lower level representation obtained from our PCA transformation in our K Nearest Neighbors recommendation system. We evaluate this approach by first implementing a euclidean distance that represents the distance between movies searched in our prompt to the user and then evaluate across our dataset to find the top 10 nearest neighbors. This method is different from other methods since our system is user-agnostic and does not require prior information about user preferences. Rather, we can just prompt the user for a movie they previously watched or are interested in and we can use the mapping obtained along the most important features from PCA to identify the most similar movies in terms of important metrics like genre and rating. We also show what our KNN predicts is the rating on aggregate of all the movies relative to actual rating of the searched movie and as opposed to previous methods, we seek to show there is strong alignment between the predicted rating and actual rating.

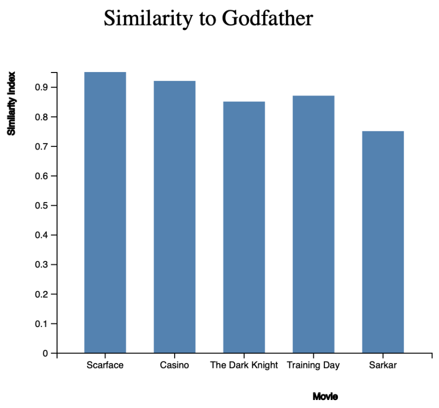
We performed the data clustering experiment using K Means to obtain groupings of movies based on genre for the movie dataset. As part of this, we used the elbow method to plot the inertia values for different values of k. From our plot we saw that the inertia value seems to level off at around 8. The inertia is defined as the squared distance between each point and the centroid of the cluster. Using the elbow method, we want to identify a low number of clusters while also having low average inertia across genres. Thus we identified 4 as a good number of clusters using the elbow method. We saw that there was leveling off as K increases: we could add more clusters but this will not give us the information we want even though the error seemingly decreases. One important thing to note is that the K means clusters were calculated using the embeddings obtained from PCA rather than the original data. It is known that in higher dimensions, Euclidean distances become inflated. Therefore, by using the mapping of embeddings obtained by PCA which looks at the features that have the highest variance in the data, the data is more separable and thus can produce a better downstream visualization.



We showcase how using 4 clusters for each genre in a K means clustering of the movie data produces a rich visualization in the plot on the left. This visualization is produced using UMAP which is able to project high-dimensional data to a low-dimensional visualization while preserving the global structure of the data (i.e. the relative distinctions between different classes). These benefits of UMAP are clearly shown in the visualization as there are distinct regions where each subclass exists that are clearly separable by genre. This plot colors each point according to the cluster obtained per genre.



The data that we got was the TMDB 5000 Movie Dataset, which we downloaded from Kaggle. This dataset contains two CSV files. The first CSV, called movies, is a file containing specific information about each movie including the budget, genre, keywords, popularity, production company, etc. There are a total of 20 unique columns in this dataset and there are 5000 movies that this information is provided for. The second csv, called credits, information revolving around the credits of the movie including who the cast included and the crew members that were a part of each movie.evalu



In our interactive visualizations, users can search along their favorite genre and this 4-coloring of all movies is shown on the left using UMAP. The user can hover across the different clusters of movies to see each individual movie and a bar chart is shown with the top 10 most similar movies. The user is also shown the distance (i.e. how close the movie is to the selected movie). The figure on the left which was generated from a d3 Javascript visualization of the movies by genre. In this visualization, the movies can be displayed as points where

relative distances are correlated with similarity. In this experiment, the top ten movies were shown on the x-axis and the similarity score (distance) was used as the y-axis.
Another aspect of our interactive visualization is a search feature that allows the user to search a movie within the database and get a selection of the top 5 most similar movies. The figure on the left is generated from a d3 Javascript visualization. In the visualization, the 5 movies with the highest similarity index to the chosen movie are shown in the bar chart. The similarity index is determined from the distance calculated by KNN. Listed on the y-axis is the similarity index and on the x-axis are the 5 movies. We can see that the recommendations produced by KNN are highly similar to those of the movie the user searched for, showing that using a content-based approach finds movies aligned by genre and rating that would interest the user.

Evaluation

The main question our paper seeks to answer is: given a movie can we find movies that are most similar to it? We also delve into this question by looking within different genres. Hence, this question forms the basis of most of our experiments. Through our machine learning methods K-NN and K-means, we can attempt to answer this question to the best of our ability. The visualization tool utilizes the results of running these algorithms to display the most similar movies for a given movie.
Our testbed consists of multiple components. Within our machine learning experiments, we needed to test the errors for different values of k. For example, in K-means different clusters lead to different error rates, and at some point the error rate levels off and additional clusters are not necessary. Thus, it was necessary to conduct machine learning specific tests to ensure best results for visualization. The final component was verifying results according to our visualization tools. This consists of ensuring that the visualizations match the results from our machine learning experiments. It also consists of verifying results manually while using the visualization tool.

Conclusion and Discussion

All team members have contributed equally to the project.

The main idea of our project is to provide a recommendation system for movies through a series of machine learning algorithms and visualization techniques. The power of the project comes from the coupled nature of the two. The machine learning aspect allows us to search through the huge dataset of movies and attributes and come up with formulaic similarities between movies. The visualization component facilitates the display process and makes the results marketable. This is not uncommon for other projects. However, our project discusses new machine learning techniques to produce different results. Given the breadth of machine learning, our project provides a new angle for movie recommendation systems. It also goes more in depth by providing similarities within genres, which essentially finds the most similar movies possible.

The limitation of such a project is that to a non computer science person, the final product may seem similar to other movie recommendation systems. For example, for a user the result may seem the same as another project that takes in a movie and spits out similar movies. The user does not know the complex algorithms necessary to obtain such results. This is why we chose unique visualization techniques to showcase how similar movies are. We can thus attempt to convey the inner workings of our project and separate ourselves from other systems.

The future implications of our work come from our understanding of the relationship between machine learning and visualization. Machine learning is one of the most popular and upcoming fields, but many people still don’t know how to use it or integrate it into other technologies. Our project showed how one can leverage technological visualization techniques to present machine learning results in the simplest ways possible. We can “blackbox” machine learning results and use them to create unique visualization experiences. This is powerful and doing so could increase knowledge and understanding of machine learning. Our visualization showcases the clusters and distances for different movies while abstracting away the many for loops and libraries that allowed us to find those clusters. Developers and common people may be better able to understand what machine learning can do for them.

References

[1] A D Alfarizy et al 2017 IOP Conf. Ser.: Earth Environ. Sci. 58 012032

[2] Ali, N., Neagu, D. & Trundle, P. Evaluation of k-nearest neighbour classifier performance for heterogeneous data sets. SN Appl. Sci. 1, 1559 (2019). https://doi.org/10.1007/s42452-019-1356-9

[3] Βοζαλής, Εμμανουήλ & Vozalis, Manolis & Μαργαρίτης, Κωνσταντίνος & Margaritis, Konstantinos G.. (2008). A Recommender System using Principal Component Analysis.

[4] Farinella, T., Bergamaschi, S., & Po, L. (2012). A Non-intrusive Movie Recommendation System. OTM Conferences.

[5] Iliopoulou, K., Kanavos, A., Ilias, A., Makris, C., & Vonitsanos, G. (2020). Improving Movie Recommendation Systems Filtering by Exploiting User-Based Reviews and Movie Synopses. Artificial Intelligence Applications and Innovations. AIAI 2020 IFIP WG 12.5 International Workshops: MHDW 2020 and 5G-PINE 2020, Neos Marmaras, Greece, June 5–7, 2020, Proceedings, 585, 187–199. https://doi.org/10.1007/978-3-030-49190-1_17

[6] Jayalakshmi S, Ganesh N, Čep R, Senthil Murugan J. Movie Recommender Systems: Concepts, Methods, Challenges, and Future Directions. Sensors (Basel). 2022 Jun 29;22(13):4904. doi: 10.3390/s22134904. PMID: 35808398; PMCID: PMC9269752.

[7] Katarya, Rahul, and Om Prakash Verma. “An Effective Collaborative Movie Recommender System with Cuckoo Search.” Egyptian Informatics Journal, Elsevier, 9 Nov. 2016, https://www.sciencedirect.com/science/article/pii/S1110866516300470.

[8] Li, B., Liao, Y., & Qin, Z. (2014). Precomputed clustering for movie recommendation system in real time. Journal of Applied Mathematics. https://link.gale.com/apps/doc/A424531585/AONE?u=googlescholar&sid=googleScholar&xid=fd22fbc4

[9] L. Kozma, A. Ilin and T. Raiko, "Binary principal component analysis in the Netflix collaborative filtering task," 2009 IEEE International Workshop on Machine Learning for Signal Processing, 2009, pp. 1-6, doi: 10.1109/MLSP.2009.5306186.

[10] Maaten, Laurens van der and Geoffrey E. Hinton. “Visualizing Data using t-SNE.” Journal of Machine Learning Research 9 (2008): 2579-2605.

[11] M A Syakur et al 2018 IOP Conf. Ser.: Mater. Sci. Eng. 336 012017

[12] Movie Recommender System Using K-Means Clustering and K-Nearest Neighbor. https://www.researchgate.net/publication/334763301_Movie_Recommender_System_Using_K-Means_Clustering_AND_K-Nearest_Neighbor.

[13] Shah, D., & Motiani, S. (2013). Movie Classification Using k-Means and Hierarchical Clustering An analysis of clustering algorithms on movie scripts.

[14] Vilakone, P., Park, DS., Xinchang, K. et al. An Efficient movie recommendation algorithm based on improved k-clique. Hum. Cent. Comput. Inf. Sci. 8, 38 (2018). https://doi.org/10.1186/s13673-018-0161-6

[15] W. Sun, J. Jiang, Y. Huang, J. Li and M. Zhang, "An Integrated PCA-DAEGCN Model for Movie Recommendation in the Social Internet of Things," in IEEE Internet of Things Journal, vol. 9, no. 12, pp. 9410-9418, 15 June15, 2022, doi: 10.1109/JIOT.2021.3111614.

[16] W. Wu, W. Zhang, Y. Yang and Q. Wang, "DREX: Developer Recommendation with K-Nearest-Neighbor Search and Expertise Ranking," 2011 18th Asia-Pacific Software Engineering Conference, 2011, pp. 389-396, doi: 10.1109/APSEC.2011.15.

[17] Xu, B., Chen, C., & Yang, J. H. (2022). Application of Cluster Analysis Technology in Visualization Research of Movie Review Data. Computational intelligence and neuroscience, 2022, 7756896. https://doi.org/10.1155/2022/7756896

[18] Zhang Z. Introduction to machine learning: k-nearest neighbors. Ann Transl Med. 2016 Jun;4(11):218. doi: 10.21037/atm.2016.03.37. PMID: 27386492; PMCID: PMC4916348.