Musik in bayern

ISSN: 0937-583x Volume 89, Issue 6 (June -2024)

https://musikinbayern.com

DOI https://doi.org/10.15463/gfbm-mib-2023-*255*

Clustering and Beyond: The Many Faces of Unsupervised Learning

Dr. G. Savitha[1], Assistant Professor,
M. Poomani[2], Assistant Professor,
J. Muthuselvi[3] Assistant Professor
Department of Computer Science, FSH, SRMIST, Ramapuram

To Cite this Article

Dr. G. Savitha," Clustering and Beyond: The Many Faces of Unsupervised Learning"

Musik In Bayern, Vol. 89, Issue 6, June 2024, pp225-236

Article Info

Received: 25-04-2024 Revised: 2-05-2024 Accepted: 29-05-2024 Published: 12-06-2024

Abstract

Unsupervised machine learning works with data that does not have predetermined classifications or results. Unsupervised learning algorithms function by finding patterns, structures, and relationships within the data itself, as opposed to supervised learning, which uses labelled datasets to train algorithms. This method works especially well in situations where data labelling is neither feasible nor practicable. Clustering, which puts related data points together, and dimensionality reduction, which reduces the amount of features in data while maintaining its fundamental properties, are important approaches in unsupervised learning. Unsupervised learning has use in many domains, such as picture compression, anomaly detection, and market segmentation. Even with all of its potential, unsupervised learning is difficult to evaluate and comprehend because there are no ground truth labels to help with this process. Nevertheless, in the larger field of data science and artificial intelligence, it continues to be an important area of study and application.

Keywords: Unsupervised Learning, PCA, Chevron (CVX) and ExxonMobil (XOM), K-Means, Screeplot

1. Introduction

Unsupervised learning is sort of machine learning where no predefined label or outcome can guide the behavior. Supervised learning depends on labelled datasets in which the right output is already known, while unsupervised algorithms like K-Means Clustering discover patterns, groups and relationships in data with no previous knowledge of what constitutes correct versus incorrect. While this is true, it can be useful when labelling data is expensive, takes far too much time or just isn't possible.

Clustering & dimensionality reduction are the two major techniques of unsupervised learning. Clustering algorithms such as K-means, hierarchical clustering partition the dataset into distinct groups based on similarity that allows for natural groupings of data. Principal Component

Analysis (PCA) or t-Distributed Stochastic Neighbour embedding (t-SNE) dimensionality reduction algorithms remove the redundant information and reduce the features in the data set, keeping its intrinsic nature that simplifies visualization & analysis of data.

There are many examples of unsupervised learning use case across various sectors. For businesses, it can mean market segmentation to create potential client groupings or segments that correlate well with purchase behaviour. It finds abnormal patterns that could point to fraud or system errors in anomaly detection. This makes it less complex, thereby making image data easier to store and transmit as image compression solutions.

Unsupervised Learning Doesn't Come All Roses. TextChangedompiler_ Error Annotated labels are not available to enable performance evaluation of the algorithms and understanding outputs. Moreover, finding the right number of clusters or dimensions is sometimes a hit-or-miss process that requires domain knowledge.

The term *unsupervised learning* refers to statistical methods that extract meaning from data without training a model on labeled data (data where an outcome of interest is known).

2. Unsupervised Learning

2.1 Principal Components Analysis (PCA)

Key Terms for Principal Components Analysis

- ***Principal component-*** A linear combination of the predictor variables.
- ***Loadings-*** The weights that transform the predictors into the components. *Synonym*
    - Weights
- ***Screeplot-*** A plot of the variances of the components, showing the relative importance of the components, either as explained variance or as proportion of explained variance.

For two variables, X_1 and X_2, there are two principal components Zi (i = 1 or 2):

$$Zi=w_{i,1}X_1+wi_{,2}X_2$$

The component loadings are denoted by the weights (wi,1,wi,2). These give rise to the primary components from the initial variables. The linear combination that best explains the entire variation is the first principal component, or Z1. Z2, the second principal component, is oriented perpendicularly to the first and attempts to account for the majority of the residual variation. (If there were more parts, they would all be perpendicular to one another.)

Example:

The stock price returns for Chevron (CVX) and ExxonMobil (XOM) are subjected to a PCA in this manner:

- Bring in the required library imports
- Take the pertinent information out of the DataFrame
- Establish the number of components in the PCA model at first
- Match the data to the PCA model

DOI https://doi.org/10.15463/gfbm-mib-2023-*255*

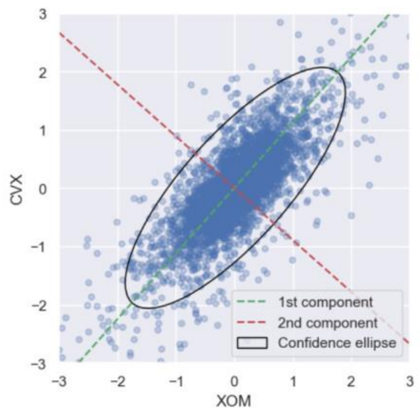
- Extract the PCA loadings and format them

	CVX	XOM
0	-0.747101	-0.664711
1	-0.664711	0.747101

The weights for CVX and XOM for the first principal component are –0.747 and –0.665, and for the second principal component they are -0.665 and 0.747. How to interpret this? The first principal component is essentially an average of CVX and XOM, reflecting the correlation between the two energy companies. The second principal component measures when the stock prices of CVX and XOM diverge.

It is instructive to plot the principal components with the data.

The dashed lines show the direction of the two principal components: the first one is along the long axis of the ellipse, and the second one is along the short axis. You can see that a majority of the variability in the two stock returns is explained by the first principal component. This makes sense since energy stock prices tend to move as a group.



Computing the Principal Components

Going from two variables to more variables is straightforward:

1. In creating the first principal component, PCA arrives at the linear combination of predictor variables that maximizes the percent of total variance explained.
2. This linear combination then becomes the first “new” predictor, Z_1.
3. PCA repeats this process, using the same variables with different weights, to create a second new predictor, Z_2. The weighting is done such that Z_1 and Z_2 are uncorrelated.
4. The process continues until you have as many new variables, or components, Z_1 as original variables X_1.
5. Choose to retain as many components as are needed to account for most of the variance.
6. The result so far is a set of weights for each component. The final step is to convert the original data into new principal component scores by applying the weights to the original values. These new scores can then be used as the reduced set of predictor variables.

Interpreting Principal Components

ISSN: 0937-583x Volume 89, Issue 6 (June -2024)

https://musikinbayern.com

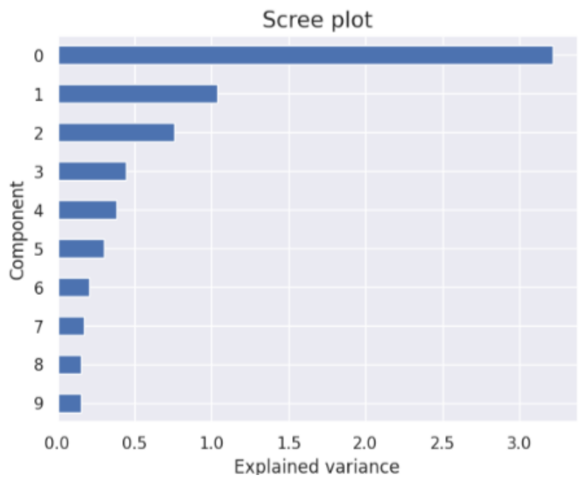
DOI https://doi.org/10.15463/gfbm-mib-2023-*255*

Use *screeplot* to visualize the relative importance of principal components (the name derives from the resemblance of the plot to a scree slope; here, the y-axis is the eigenvalue).

Below code shows an example for a few top companies in the S&P 500:

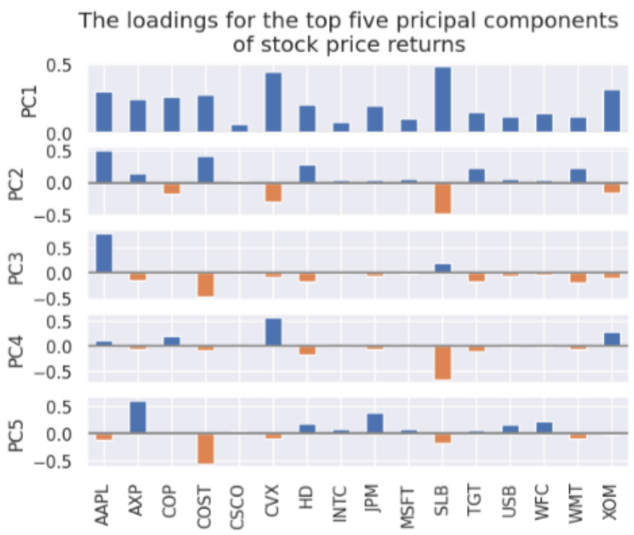
It can be especially revealing to plot the weights of the top principal components.

The loadings for the first principal component have the same sign: this is typical for data in which all the columns share a common factor (in this case, the overall stock market trend). The second component captures the price changes of energy stocks as compared to the other stocks. The third component is primarily a contrast in the movements of Apple and CostCo. The fourth component contrasts the movements of Schlumberger (SLB) to the other energy stocks. Finally, the fifth component is mostly dominated by financial companies.



Correspondence Analysis

PCA cannot be used for categorical data; however, a somewhat related technique is *correspondence analysis*. The goal is to recognize associations between categories, or between categorical features.



2.2 K-Means Clustering

ISSN: 0937-583x Volume 89, Issue 6 (June -2024)

https://musikinbayern.com

DOI https://doi.org/10.15463/gfbm-mib-2023-*255*

Key Terms for K-Means Clustering

- ***Cluster***- A group of records that are similar.
- ***Cluster mean***- The vector of variable means for the records in a cluster.
- ***K***- The number of clusters.

K-means divides the data into *K* clusters by minimizing the sum of the squared distances of each record to the *mean* of its assigned cluster. This is referred to as the *within-cluster sum of squares or within-cluster SS*.

It is easy to find the detailed algorithm of *K*-Means online, I will not put here. It is important that the variables need to be scaled (e.g. standardize) before grouping. We tell the model the importance of each variable thorough scaling. If all variables are standardize scaled, they show same significance in the model.

K-Means Clustering can be used to engineer new feature for more sophisticated learning model.

Example

Suppose we want to divide daily stock returns into four groups. K-means clustering can be used to separate the data into the best groupings. Note that daily stock returns are reported in a fashion that is, in effect, standardized, so we do not need to normalize the data. The following finds four clusters based on two variables—the daily stock returns for ExxonMobil (XOM) and Chevron (CVX):

Algorithm:

- Import the necessary libraries.
- Filter the DataFrame for the specified date range and columns.
- Initialize the K-means model specifying the number of clusters and a random state for reproducibility.
- Fit the K-means model to the data.
- Assign the cluster labels to a new column in the DataFrame.
- Display the first few rows of the updated DataFrame.

	XOM	CVX	cluster
2011-01-03	0.736805	0.240681	0
2011-01-04	0.168668	-0.584516	2
2011-01-05	0.026631	0.446985	0
2011-01-06	0.248558	-0.919751	2
2011-01-07	0.337329	0.180511	0

The first 5 records are assigned to either cluster 0 or cluster 3. Below codes show the center (means) of the clusters.

ISSN: 0937-583x Volume 89, Issue 6 (June -2024)

https://musikinbayern.com

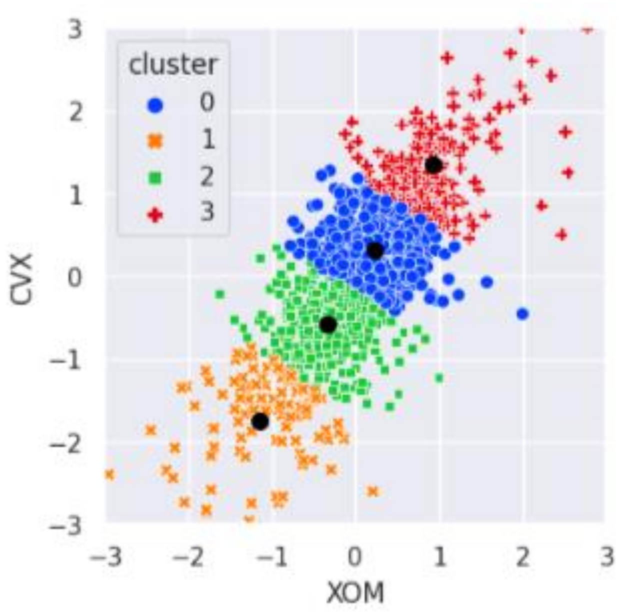
DOI https://doi.org/10.15463/gfbm-mib-2023-255

	XOM	CVX
0	0.231540	0.316965
1	-1.144397	-1.757796
2	-0.330814	-0.574398
3	0.927032	1.346412

Clusters 1 and 3 represent "down" markets, while clusters 0 and 2 represent "up" markets.

Selecting the Number of Clusters

A common approach, called the elbow method, is to identify when the set of clusters explains "most" of the variance in the data. Adding new clusters beyond this set contributes relatively little in the variance explained. The elbow is the point where the cumulative variance explained flattens out after rising steeply, hence the name of the method.



In this case, there is no clear "elbow". In general, there is no single rule that will reliably guide how many clusters to produce.

2.3 Hierarchical Clustering

Key Terms for Hierarchical Clustering

- ***Dendrogram***- A visual representation of the records and the hierarchy of clusters to which they belong.
- ***Distance***- A measure of how close one record is to another.
- ***Dissimilarity***- A measure of how close one cluster is to another.

Hierarchical clustering works on a data set with n*n* records and p*p* variables and is based on two basic building blocks:

Musik in bayern

ISSN: 0937-583x Volume 89, Issue 6 (June -2024)

https://musikinbayern.com DOI https://doi.org/10.15463/gfbm-mib-2023-*255*

- A distance metric di,j$d_{i,j}$ to measure the distance between two records ii and jj.
- A dissimilarity metric DA,B$D_{A,B}$ to measure the difference between two clusters AA and BB based on the distances di,j$d_{i,j}$ between the members of each cluster.

For applications involving numeric data, the most importance choice is the dissimilarity metric. Hierarchical clustering starts by setting each record as its own cluster and iterates to combine the least dissimilar clusters.

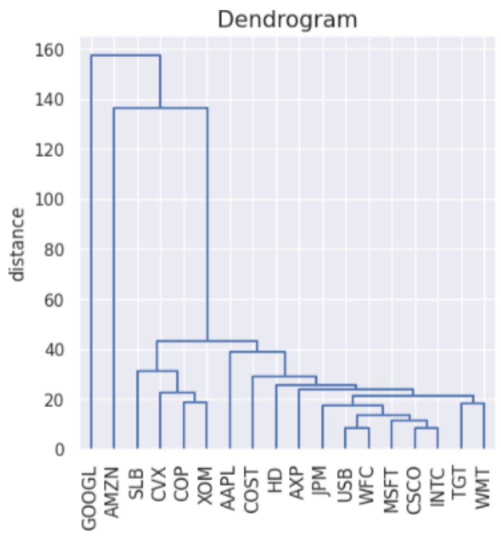
Because hierarchical clustering operates on pairwise distance of all points, the computational cost is high especially with large data sets. As a result, most of the applications of hierarchical clustering are focused on relatively small data sets.

Example

The following applies hierarchical clustering to the stock returns for a set of companies.

Clustering algorithms will cluster the records (rows) of a data frame. Since we want to cluster the companies, we need to *transpose* the data frame and put the stocks along the rows and the dates along the columns.

The scipy package offers a number of different methods for hierarchical clustering in the scipy.cluster.hierarchy module. Here we use the linkage function with the "complete" method:



The result is shown above (note that we are now plotting companies that are similar to one another, not days). The leaves of the tree correspond to the records. The length of the branch in the tree indicates the degree of dissimilarity between corresponding clusters. The returns for Google and Amazon are quite dissimilar to one another and to the returns for the other stocks. The oil stocks (SLB, CVX, XOM, COP) are in their own cluster, Apple (AAPL) is by itself, and the rest are similar to one another.

In contrast to K-means, it is not necessary to prespecify the number of clusters. Graphically, you can identify different numbers of clusters with a horizontal line that slides up or down; a cluster is defined wherever the horizontal line intersects the vertical lines.

ISSN: 0937-583x Volume 89, Issue 6 (June -2024)

https://musikinbayern.com

DOI https://doi.org/10.15463/gfbm-mib-2023-255

The number of clusters to extract is set to 4, and you can see that Google and Amazon each belong to their own cluster. The oil stocks all belong to another cluster. The remaining stocks are in the fourth cluster.

The Agglomerative Algorithm

The main steps of the agglomerative algorithm are:

1. Create an initial set of clusters with each cluster consisting of a single record for all records in the data.
2. Compute the dissimilarity D(Ck,Cℓ) between all pairs of clusters k,ℓ
3. Merge the two clusters Ck and Cℓ that are least dissimilar as measured by D(Ck,Cℓ).
4. If we have more than one cluster remaining, return to step 2. Otherwise, we are done.

Measures of Dissimilartity

There are four common measures of dissimilarity.

1. *Complete linkage*
   D(A,B)=max d(ai,bj) for all pairs i,j$D(A,B)$=max $d(ai,bj)$ for all pairs i,j
   - This method tends to produce clusters with members that are similar.
2. *Single linkage*
   D(A,B)=min d(ai,bj) for all pairs i,j$D(A,B)$=min $d(ai,bj)$ for all pairs i,j
   - This is a “greedy” method and produces clusters that can contain quite disparate elements.
3. *Average linkage*
   - It is the average of all distance pairs and represents a compromise between the single and complete linkage methods.
4. *Minimum variance*
   - Also referred to as *Ward’s* method, is similar to K-means since it minimizes the within-cluster sum of squares.

Below code applies hierarchical clustering using the four measures to the ExxonMobil and Chevron stock returns. For each measure, four clusters are retained.

The results are strikingly different: the single linkage measure assigns almost all of the points to a single cluster. Except for the minimum variance method (ward), all measures end up with at least one cluster with just a few outlying points. The minimum variance method is the most similar to the K-means cluster.

2.4 Model-Based Clustering

Simply speaking, model-based clustering is fitting statistical models (e.g. normal distribution) to the data, similar to peak fittings widely used in engineering tasks.

The most widely used model_based clustering methods rest on the *multivariate normal* distribution.

A multivariate normal distribution can be defined by the means of all the variables μμ (center of the distribution) and the covariance matrix ΣΣ (variance and covariance between variables).

Mistures of Normals

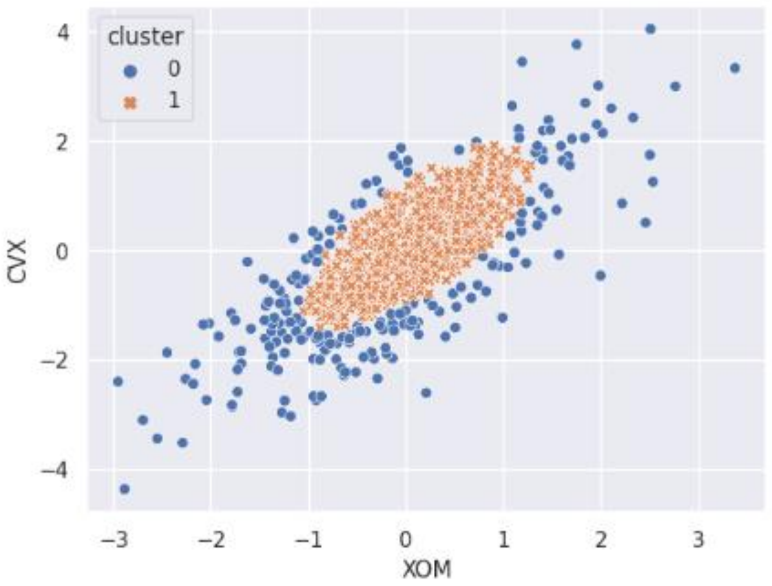
Musik in bayern

ISSN: 0937-583x Volume 89, Issue 6 (June -2024)

https://musikinbayern.com DOI https://doi.org/10.15463/gfbm-mib-2023-*255*

The key idea behind model-based clustering is that each record is assumed to be distributed as one of *K* multivariate normal distributions, where *K* is the number of clusters. Each distribution has a different mean μ*μ* and covariance matrix ΣΣ.

Let's apply model-based clustering to the stock return data we previously analyzed using *K*-means and hierarchical clustering:



There are two clusters: one cluster in the middle of the data, and a second cluster in the outer edge of the data. This is very different from the clusters obtained using K-means and hierarchical clustering, which find clusters that are compact.

Selecting the Number of Clusters

We can select the number of clusters by minimizing *Bayesian Information Criteria (BIC)*. BIC works by selecting the best-fitting model with a penalty for the number of parameters in the model. In the case of model-based clustering, adding more clusters will always improve the fit at the expense of introducing additional parameters in the model.

This plot is similar to the elbow plot used to identify the number of clusters to choose for K-means, except the value being plotted is BIC instead of inertia. One big difference is that instead of one line, there are 4 lines here! This is because we are actually fitting 4 different models (different ways to parameterize the convariance matrix ΣΣ for fitting a model) for each cluster size, then choose the best-fitting model.
In this example, according to BIC, "full" model gives the best fit using two components.

2.5 Scaling and Categorical Variables

Key Terms for Scaling Data

- ***Scaling***
  Squashing or expanding data, usually to bring multiple variables to the same scale.
  - ***Normalization-*** One method of scaling—subtracting the mean and dividing by the standard deviation.

- *Synonym*- Standardization

- ***Gower's distance***- A scaling algorithm applied to mixed numeric and categorical data to bring all variables to a 0–1 range.

Unsupervised learning techniques generally require that the data be appropriately scaled. Otherwise the results will be dominated by the variables with large values.

Categorical Data and Gower's Distance

In the case of categorical data, you must convert it to numeric data, either by ranking (for an ordered factor) or by encoding as a set of binary (dummy) variables. If the data consists of mixed continuous and binary variables, you will usually want to scale the variables so that the ranges are similar.

One popular method is to use *Gower's distance*. The basic idea behind Gower's distance is to apply a different distance metric to each variable depending on the type of data:

- For numeric variables and ordered factors, distance is calculated as the absolute value of the difference between two records (*Manhattan distance*).
- Categorical variables, the distance is 1 if the categories between two records are different, and the distance is 0 if the categories are the same.

Gower's distance is computed as follows:

1. Compute the distance $d_{i,}$ for all pairs of variables i and j for each record.
2. Scale each pair $d_{i,}$ so the minimum is 0 and the maximum is 1.
3. Add the pairwise scaled distances between variables together, using either a simple or a weighted mean, to create the distance matrix.

4. Conclusion:

In conclusion, unsupervised learning is a fantastic method for identifying structure and patterns in data which provides an important usage area in situations where labeled data is non-existent or very scarce. State-of-the-art improvements in algorithms and computational resources have been driving their applications, they are becoming indispensable research topic in machine learning & AI developments.

5. References:

1. Cortes, C., &amp;Vapnik, V. (1995). Support-Vector Networks. Machine Learning, 20(3), 273-297.

2. Joachims, T. (1998). Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In European Conference on Machine Learning (ECML) (pp. 137-142).

3. Li, H., &amp; Church, K. (2003). A Study of Smoothing Methods for Language Models Applied to Ad Hoc Information Retrieval. In Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 334-341).

Musik in bayern

ISSN: 0937-583x Volume 89, Issue 6 (June -2024)

https://musikinbayern.com DOI https://doi.org/10.15463/gfbm-mib-2023-*255*

4. Metzler, D., &amp; Croft, W. B. (2005). A Markov Random Field Model for Term Dependencies. In Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 472-479).

5. LeCun, Y., Bottou, L., Bengio, Y.,&amp;Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278-2324.

6. Hinton, G. E., Osindero, S., &amp;Teh, Y. W. (2006). A fast learning algorithm for deep belief nets. Neural computation, 18(7), 1527-1554.

7. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., &amp; Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems (pp. 3111-3119).

8. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., &amp;Polosukhin, I. (2017). Attention is all you need. In Advances in neural information processing systems (pp. 30- 38).

9. Devlin, J., Chang, M. W., Lee, K., &amp;Toutanova, K. (2018). BERT: Pre- training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

10. Chen, T., &amp;Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 785-794).

11. JiafengGuo, Yixing Fan, Qingyao Ai, W. Bruce Croft &quot;A Deep Relevance Matching Model for Ad-hoc Retrieval&quot;, SIGIR 2016.

12. Jiaxin Mao, Wei-Cheng Chang, Po-Sen Huang, Alan W. Biermann, Alexander J. Smola, Chih-Jen Lin,&quot;DeepRank: A New Deep Architecture for Relevance Ranking in Information Retrieval&quot;, WSDM 2016.

13. JiafengGuo, Yixing Fan, Qingyao Ai, W. Bruce Croft, &quot;DRMM: A Deep Ranking Model for Information Retrieval&quot;, , SIGIR 2016

14. Xuepeng Yang, Qingyao Ai, W. Bruce Croft, &quot;KNRM: A Position-Aware Neural IR Model for Web Search&quot;,: SIGIR 2017

15. Jimmy Lin, Matt Crane, Andrew Trotman, Jamie Callan, IshanChattopadhyaya, John Foley, Grant Ingersoll, Craig Macdonald, SebastianoVigna , &quot;Anserini: Enabling the Use of Lucene for Information Retrieval Research&quot;, SIGIR 2019

16. Rodrigo Nogueira, Wei Yang, Jimmy Lin , &quot;BERT for Ad Hoc Document Retrieval: Strengths and Weaknesses&quot;, SIGIR 2019

17. Qingyao Ai, ChenyanXiong, Stephen Robertson, Michael Bendersky,&quot;Unifying Relevance and Diversity in Embedding-based Retrieval" , SIGIR 2020

18. HamedZamani, PooyaMoradi, W. Bruce Croft, Erik Learned-Miller &quot;Learning to Rank with Selection Bias in Personal Search&quot; SIGIR 2018

19. JingweiXu, Hanxiao Liu, Shuai Zhang, Hao Ma, Hongyan Li &quot;Deep Reinforcement Learning for Page-wise Recommendations&quot; WWW 2019 20. Jun Yan, Ning Liu, Gang

Musik in bayern

ISSN: 0937-583x Volume 89, Issue 6 (June -2024)

https://musikinbayern.com

DOI https://doi.org/10.15463/gfbm-mib-2023-*255*

Wang, JianfengGao, Xiaoming Jin, ZaiqingNie, W. Bruce Croft &quot;Interactive Learning to Rank: A Reinforcement Learning Approach&quot; SIGIR 2011