PLOD: Predictive Learning Optimal Data Discovery

Thomas Hoang
Denison University
USA
hoang_t2@denison.edu

ABSTRACT

Inspired by the BOD: Blindly Optimal Data Discovery method [10], our algorithm inherits the benefits of its predecessor, which addresses the challenge of an unknown utility function and integrates human input for attribute ranking just once, thus avoiding the repetitive and time-consuming loop process. Additionally, our machine learning approach predicts the desired utility function from the data more accurately than BOD. Furthermore, existing methods such as [7] require precise knowledge of the utility function, which is not ideal. In contrast, our PLOD algorithm successfully predicts outcomes based on the data without needing the exact utility function, highlighting the potential of PLOD: Predictive Learning Optimal Data Discovery in contemporary data science and analytics.

PVLDB Reference Format:
Thomas Hoang. PLOD: Predictive Learning Optimal Data Discovery. PVLDB, 18(1): XXX-XXX, 2025.
doi:XX.XX/XXX.XX

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/Thomas12012002/MOD.

1 INTRODUCTION

Getting to know what data can do from large datasets is the main goal of predicting application in many data science applications. Data analysts and scientists want to get to the point where they have valid data candidates in terms of no missing, duplicate, and all are integers. After combining or augmenting all the datasets together vertically as there are sufficient instances in each dataset with different attributes, scientists want to predict what comes after understanding the data. For example, taking predicting housing prices problem which could be predicted based on several factors including location (the house may be near urban center and the criminal free status is low, etc), home values (whether the size of the house is big, medium or small; whether the house is modern style or old money style), the policy of the government in the area (the tax may a big concern for citizens). In order to predict the price of the house, scientists may use their particular domain knowledge to rank the attributes of the datasets and the algorithm will try to match the predicting utility function using machine learning.

Based on certain scenarios, with their domain knowledge, scientists prioritize one attribute over others. For example, an apartment in a city center like New York City (with low criminal status, and the apartment is modern with stylish furniture, along with the building has its own guards and camera surveillance which makes the safety standards high) compared to a house dated from 100 years ago in a middle of nowhere would be an easy bet and a good investment even they want to sell it in the future for a person with enough money to buy the apartment in a city center. But in a case where another apartment in the nearby area in New York City appears to be the same as the first one but in a kinda suburban area with a lower price, whether this is still a good investment after many years? And may other factors like friendly neighborhoods or peaceful surrounding environments could be a good factor for the buyers. These differences in the relatively important attributes model by a utility function define all the attributes and the degree to which they matter to the scientists and analysts when it comes to predicting the housing price markets.

The previous algorithms using machine learning required users to know their exact utility function beforehand which is not an optimal approach for many users. The BOD: Blindly Optimal Data Discovery [10] asks users to rank the attributes and filter out the sufficient tuples in which the number of returned tuples still has some distinguishable tuples that are the result of the predicting utility function does not highly match with the intended-to-figure-out utility function. The PLOD helps solve these stated problems by asking the user to rank the attributes beforehand and apply machine learning techniques to predict the utility function. The algorithm then outputs all the tuples based on the predicted utility function.

The work of algorithms: The algorithm asks user to rank the attributes then narrows down each coefficient in each attribute in the range of $\{0, 1\}$ based on the highest to the lowest ranked attributes. The coefficients will be estimated in each equaled range in the range of $\{0, 1\}$ based on the total number of n attributes. After that, the algorithm estimates the synthetic utility function based on the coefficients obtained from previous estimation. Then the algorithm estimates the real coefficients for the estimated

Proceedings of the VLDB Endowment, Vol. 18, No. 1 ISSN 2150-8097.
doi:XX.XX/XXX.XX

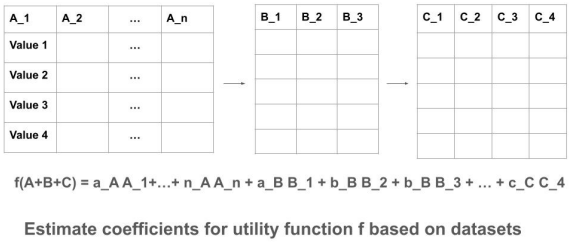

Figure 1: Augmenting tables for applying machine learning to predict the utility function based on the human's ranking for variables

utility function based on synthetic utility function using machine learning. The algorithm then outputs all the subsets that have utility satisfied the predicting utility function.

2 RELATED STUDIES

Machine Learning. The previous algorithms like in the work [7] asks for the user's utility function as an input. Nevertheless, this approach is limited in many cases that user does not understand or realize their utility function. Other works of machine learning techniques such as [1] [2] [21] also assume the functions that have limitations in some cases in housing predicting approaches.

Top-k. Many works such as [6] [12] [13] [22] [24] has been studied and developed with their applications in the filed. Nevertheless, these algorithms asks for the user with utility function which is not applicable in real life in many cases.

Skyline. there have been many developments such as [12] [14] [16] [3] [4] [26] [25] [19] [8] [28] that output a few subsets among many tuples in the datasets. However, the output size is still large that lack of control, meaning there are still many tuples that are out of interested for the user. The advanced work that combines the technique of top-**k** and Skyline algorithms is [17] helps with controlling the output's size.

OLAP-like Aggregation. The work by Gray et al. introduced the Data Cube, a relational aggregation operator that aims to generalize the operations of Group-By, Cross-Tab, and Sub-Total [9]. This work helps make the multidimensional data analysis efficient, which makes it useful for OLAP (Online Analytical Processing). However, like Top-k algorithms, this method assumes the user can define the utility function needed for data aggregation and analysis. The Data Cube provides a mechanism for summarizing data but does not inherently solve the challenge of identifying the most relevant data subsets without explicit utility functions. In detail, this assumption limits its applicability when users cannot precisely define their preferences or utility functions, which often happens in real-world applications.

Probabilistic Approximation. The algorithm introduced by Vitter [27] on random sampling provides an efficient method for probabilistic approximation in large datasets, which is helpful when dealing with large datasets where full data processing is computationally expensive. In detail, the algorithm ensures that each sample is likely to be included, even in scenarios where the total dataset size is unknown in advance. Although efficient in terms of time, similar to Top-k and Skyline methods, this approach still relies on users understanding some forms of utility function. The disadvantage is that While it addresses scalability and efficiency, it does not address users' lack of knowledge about their utility functions.

Multi-objective Optimization. The NSGA-II algorithm by Deb et al. [5] aims for solving multi-objective optimization problems and is effective in scenarios with multiple conflicting objectives and seeks to find solutions that represent trade-offs among these objectives (the Pareto front). However, similar to other algorithms like Skyline, this method can result in many potential solutions, many of which may not align with the user's preferences. Like other optimization and selection techniques, NSGA-II assumes that users can clearly define their objectives and utility functions. In practice, this assumption is often invalid, limiting the algorithm's usability in real-world applications where user preferences are not explicitly known or difficult to articulate.

Blindly Optimal Data Discovery. The advantage of BOD [10] is directly getting to understand the data or the goal of data and this could be achieved without knowing the utility function. However, this approach still outputs subsets that are uninterested in predicting mathematical problems. In order to output better subsets that are highly predicting the outcome based on datasets, we present PLOD which takes advantage of machine learning to help with predicting the actual utility function in order to output the right tuples.

Predictive learning Optimal Data Discovery. The algorithm asks the user to rank each attribute in all the attributes and then estimates the coefficients of the utility function. When obtained the synthetic predicting utility function, the algorithms estimate the actual predicting utility function using machine learning linear regression [15], [20], [23]. Then the algorithm uses the actual predicting utility function to output the predicting tuples.

3 PROBLEM DEFINITION

Given the set of relations, S_i in a total of D relations, $i \in D$, having a total of T tuples for all the relations, we want to get out of a goal of what the datasets can do or we want to predict something based on the datasets applying machine learning algorithm, Linear Regression, without knowing the exact utility function. An example of this could be predicting housing prices, in which it depends on several factors (for

Table 1: Notation and meaning

T	The set of all tuples
D	The number (interger) of all relations
D_i	The relation ith in number interger
d	The total attributes of all the relations combined
d_i	The attribute ith of a relation
S	The relation that is a result after combining all relations
S_i	The relation ith
e	The total coefficients of a relation
e_i	The ith coefficient of a relation
t	a subset of T

example: location, home values, the policy of the governments, etc), which represent several relations. In each of these relations, there are many small factors (for example, for location, we have the factors of near urban, criminal-free, etc). Using machine learning Linear Regression without a previously defined utility function could be a tough problem, we solve this by introducing PLOD: Predictive Learning Optimal Data Discovery.

This paper presents the Predictive Learning Optimal Data Discovery (PLOD) algorithm, an advancement over the previously developed Blindly Optimal Data Discovery (BOD). PLOD eliminates the necessity for users to predefine a utility function, utilizing machine learning to predict this function instead, which can significantly streamline the data analysis process. The method is especially pertinent to large-scale data environments where such predefined knowledge is not practical.

This algorithm asks the scientists with their domains to rank the attributes in each relation then narrows down each value in the range of $\{0, 1\}$ based on the highest to the lowest ranked in each attribute. After obtaining a relation S, we apply machine learning linear regression algorithm to estimate the synthetic utility function. Below is the linear function:

Let S_i for $i \in \mathbb{N}$ denote an ith relation that has n_i attributes such that there are D relations with a total of d attributes. The set T of all tuples that have relations $(S_1, \ldots, S_n)$, in which each relation has the same T tuples, in case there are no missing or duplicate tuples.

Linear function:

- $LINEAR = \{f | f(x) = \sum_{i=1}^{d} f(x_i)$
  {where each $f(x_i)$ is a linear function.}

MATHEMATICAL PROOF OF LINEAR REGRESSION COEFFICIENTS

Based on notable work on linear regression [11], we aim to find the best-fitting line for given datasets after the scientist ranks the attributes. The best-fitting line is the one that minimizes the sum of the squared differences between the observed values and the values predicted by the line. This method is known as the least squares method.

3.1 Model Representation

The linear regression model can be represented as:

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

for $i = 1, 2, \ldots, n$, where:

- y_i is the dependent variable.
- x_i is the independent variable.
- β_0 is the y-intercept.
- β_1 is the slope.
- ϵ_i is the error term.

3.2 Objective

The objective is to find the coefficients β_0 and β_1 that minimize the sum of the squared errors (SSE):

$$SSE = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_i))^2$$

3.3 Derivation of Coefficients

To find the optimal values of β_0 and β_1, we take the partial derivatives of SSE with respect to β_0 and β_1 and set them to zero.

3.3.1 Partial Derivative with Respect to β_0.

$$\frac{\partial}{\partial \beta_0} \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_i))^2 = -2 \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_i))$$

3.3.2 Partial Derivative with Respect to β_1.

$$\frac{\partial}{\partial \beta_1} \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_i))^2 = -2 \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_i)) x_i$$

3.3.3 Solving the Equations. We now have two normal equations:

$$\sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i) = 0$$

$$\sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i) x_i = 0$$

These can be rewritten as:

$$\sum_{i=1}^{n} y_i = n\beta_0 + \beta_1 \sum_{i=1}^{n} x_i$$

$$\sum_{i=1}^{n} y_i x_i = \beta_0 \sum_{i=1}^{n} x_i + \beta_1 \sum_{i=1}^{n} x_i^2$$

Solving for β_0 and β_1, we get:

$$\beta_1 = \frac{n \sum_{i=1}^{n} y_i x_i - \sum_{i=1}^{n} y_i \sum_{i=1}^{n} x_i}{n \sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2}$$

$$\beta_0 = \frac{\sum_{i=1}^{n} y_i - \beta_1 \sum_{i=1}^{n} x_i}{n}$$

3.4 Final Formulas

The formulas for the coefficients are:

$$\beta_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

$$\beta_0 = \bar{y} - \beta_1 \bar{x}$$

where $\bar{x}$ and $\bar{y}$ are the means of x and y, respectively.

This derivation follows the standard approach described in *Introduction to Statistical Learning* [11].

Then the algorithm estimates the real coefficients. After that, based on the estimated real utility function, the algorithm returns all the subsets that satisfy the predicted utility function.

Algorithm 1 Predictive Learning Optimal Data Discovery

Require: The set of relations S_i for $i \in D$ that has T tuples.
Ensure: return a subset t of T.

1: Let $D_i = 1$
2: **while** $D_i \leq D$ **do**
3: **for** d_i in relation D_ith (or S_i) **do**
4: Ask the scientist to rank each d_i in each table S_i based on their domain knowledge
5: If scientist ranks d_i as 1st then $e_i = 1$
6: If scientist ranks d_i as the least then $e_i = 0$
7: Else, calculate e_i based on ranking by dividing the number of attributes in S_i with the rank
8: $D_i += 1$
9: Concatenate S_i horizontally along columns with T tuples.
10: **for** $d_i \in d$ **do**
11: Find the maximum value in d_i, called max_i
12: Scale down all values by dividing max_i
13: Update new S
14: Using Linear Regression Machine Learning to estimate actual coefficients for actual utility function based on synthetic coefficient e
15: Using estimated actual utility function to filter only a subsets t
16: **return** t

4 PLOD: PREDICTIVE LEARNING OPTIMAL DATA DISCOVERY

Line 1 – 8, the algorithm asks the user to rank the attributes and then narrows down each coefficient in each attribute in the range of $\{0, 1\}$ based on the highest to the lowest ranked attributes. The coefficients will be estimated in each equaled range in the range of $\{0, 1\}$ based on the total number of attributes. Line 9 – 13, the algorithm concatenates all the relations and then scales down the values in each cell in the range of $\{0, 1\}$. Line 14 – 15, the algorithm estimates the synthetic utility function based on the coefficients obtained from previous estimations. Then, the algorithm estimates the real coefficients for the estimated utility function based on the synthetic utility function using machine learning. Line 16, the query then outputs all the subsets that have utility satisfied the predicting utility function.

Figure 2: Sample on datasets on actual and estimated utility function

We run a sample of 5000 tuples on a total of 9 attributes in which the values are randomly generated in the range of $\{0, 1\}$. Based on the ranked attributes by scientists, we estimate synthetic utility function then use machine learning linear regression to estimate the actual utility function. We find the utilities based on the actual utility function compare the actual and estimated output as in the figure 2

5 EXPERIMENTAL EVALUATION

Assuming the process of concatenating datasets along the columns is correct with no error from the scientist's domain, it means the data are valid integers, not duplicates, and there are no missing values. Using Google Collaboration on a laptop GL 65 Leopard 10SCXK, an x64-based PC, on Microsoft Windows 11 Home Single Language, we tested the algorithm PLOD with different numbers of tuples, including 30000, 50000, 100000, 150000, and 300000 tuples with 9 attributes in total, as shown in Figure 3 and Figure 4. The values in datasets are generated randomly in $\{0, 1\}$.

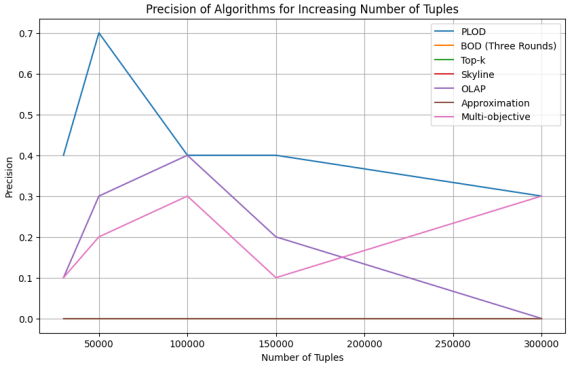

Figure 3: Precision comparison with changes in number of tuples.

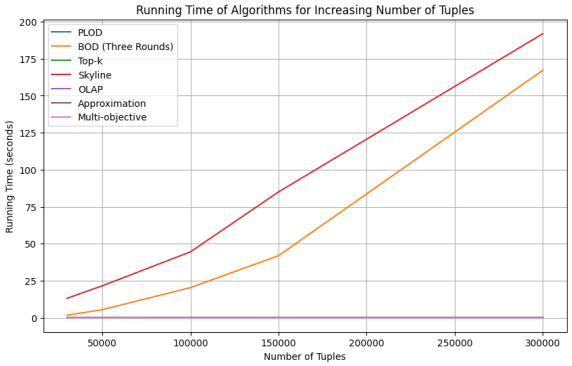

Figure 4: Runtime comparison with changes in number of tuples.

In the precision analysis shown in Figure 3, we observe that the PLOD (Predictive Learning Optimal Data Discovery) algorithm demonstrates higher precision than other algorithms when the number of tuples is smaller. However, as the number of tuples increases, the precision of PLOD decreases slightly but stabilizes around a specific value. In contrast, other algorithms like Skyline and OLAP show more fluctuation in precision, with Skyline notably decreasing as the number of tuples increases.

In Figure 4, we compare the runtime of PLOD with other algorithms as the number of tuples increases. PLOD consistently runs faster than BOD (Blindly Optimal Data Discovery) across all tested dataset sizes. In addition, the runtime of Skyline significantly increases with the number of tuples, making it the slowest among the tested algorithms. On the other hand, PLOD shows a near-linear increase in runtime but remains the most efficient, closely followed by Top-k and OLAP, which also show efficient scaling with the number of tuples.

5.1 Precision and Stability comparison using Boston Housing data [18]

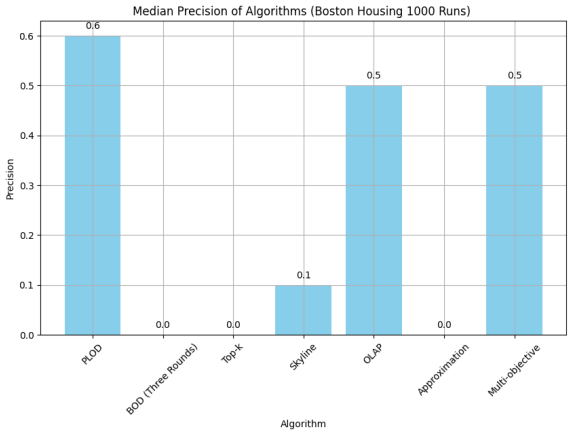

Figure 5: Precision comparison with Boston Housing Data.

The precision comparison is shown in Figure 5 using the Boston Housing dataset. The Predictive Linear Optimization Decision-making (PLOD) algorithm demonstrates the highest precision at 0.6, outperforming others such as OLAP, Approximation, and Multi-objective, each of which achieves a precision of 0.5. However, algorithms like BOD (Three Rounds), Top-k, and Multi-objective show significantly lower precision, indicating their limitations in their application to this dataset.

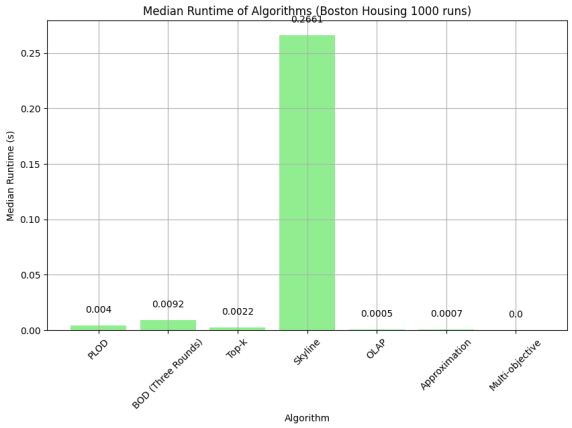

Figure 6: Runtime comparison with Boston Housing Data.

Figure 6 illustrates the runtime performance of these algorithms over 1000 runs. In detail, the Skyline algorithm is the most computationally expensive since its median runtime is approximately 0.2661. In real-time applications, the OLAP and Approximation methods have the fastest execution times, with median runtimes as low as 0.0005 seconds. Thus, this makes them more suitable for scenarios where computational efficiency is important, as opposed to the other methods, which may be much slower.

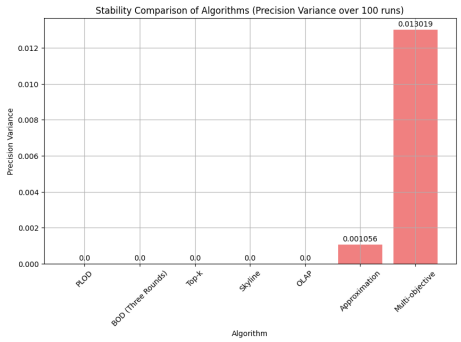

Figure 7: Stability Comparison of Algorithms on Boston Housing Data (Precision Variance over 100 runs).

The stability comparison of algorithms, as illustrated in Figure 7, shows the consistency across multiple executions.

Thus, lower variance indicates higher stability, in this analysis, the PLOD, BOD (Three Rounds), Top-k, Skyline, and OLAP algorithms demonstrated good stability, with near-zero precision variance, indicating their robust performance across different runs. Besides, the Approximation and Multi-objective algorithms demonstrate higher variance, with the Multi-objective algorithm showing the highest instability at a variance of 0.013019. This suggests that these algorithms are more sensitive to changes in the dataset, which causes fluctuations in precision across runs.

6 CONCLUSION

The advantage of the algorithm PLOD: Predictive Learning Optimal Data Discovery is its estimation for highly accurate utility function without asking for the user's utility function and the running time compared to other approaches before it. The open questions for other later findings may be: 1, Can a vector database be applied with machine learning for such an approach like PLOD? 2, If there are other types of data not like integers, can we apply a vector database? or 3, How can we transform these types of data, including text, audio, etc., into the $\{0, 1\}$ range and use machine learning or another approach?

7 CITATIONS

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Z. Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek Gordon Murray, Benoit Steiner, Paul A. Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zhang. 2016. TensorFlow: A system for large-scale machine learning. In *USENIX Symposium on Operating Systems Design and Implementation*. https://api.semanticscholar.org/CorpusID:6287870

[2] Dana Van Aken, Dongsheng Yang, Sebastien Brillard, Ari Fiorino, Bohan Zhang, Christian Billian, and Andrew Pavlo. 2021. An Inquiry into Machine Learning-based Automatic Configuration Tuning Services on Real-World Database Management Systems. *Proc. VLDB Endow.* 14 (2021), 1241–1253. https://api.semanticscholar.org/CorpusID:232328277

[3] C. Chan, H. Jagadish, K. Tan, A. Tung, and Z. Zhang. 2006. Finding k-dominant skylines in high dimensional space. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*.

[4] C. Chan, H. Jagadish, K. Tan, A. Tung, and Z. Zhang. 2006. On high dimensional skylines. In *Advances in Database Technology-EDBT 2006*. Springer, 478–495.

[5] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation* 6, 2 (2002), 182–197.

[6] P. Fraternali, D. Martinenghi, and M. Tagliasacchi. 2012. Top-k bounded diversification. In *Proceedings of the 2012 International Conference on Management of Data*.

[7] Sainyam Galhotra, Yue Gong, and Raul Castro Fernandez. 2023. Metam: Goal-Oriented Data Discovery. *2023 IEEE 39th International Conference on Data Engineering (ICDE)* (2023), 2780–2793. https://api.semanticscholar.org/CorpusID:258187398

[8] M. Goncalves and M. Yidal. 2005. Top-k skyline: a unified approach. In *On the Move to Meaningful Internet System 2005*.

[9] Jim Gray, Adam Bosworth, Andrew Layman, and Hamid Pirahesh. 1996. Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-total. *Data mining and knowledge discovery* 1, 1 (1996), 29–53.

[10] Thomas Hoang. 2024. BOD: Blindly Optimal Data Discovery. *ArXiv* abs/2401.05712 (2024). https://api.semanticscholar.org/CorpusID:266933015

[11] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2013. *An Introduction to Statistical Learning: with Applications in R*. Springer.

[12] Jongwuk Lee, Gae won You, and Seung won Hwang. 2009. Personalized top-k skyline queries in high-dimensional space. *Information Systems* 34, 1 (2009), 45–61.

[13] X. Lian and L. Chen. 2009. Top-k dominating queries in uncertain databases. In *Proceedings of International Conference on Extending Database Technology: Advances in Database Technology*.

[14] X. Lin, Y. Yuan, Q. Zhang, and Y. Zhang. 2007. Selecting stars: The k most representative skyline operator. In *Proceedings of International Conference on Data Engineering*.

[15] Dastan Hussen Maulud and Adnan Mohsin Abdulazeez. 2020. A Review on Linear Regression Comprehensive in Machine Learning. https://api.semanticscholar.org/CorpusID:231748167

[16] D. Mindolin and J. Chomicki. 2009. Discovering relative importance of skyline attributes. In *Proceedings of the VLDB Endowment*.

[17] Kyriakos Mouratidis, Keming Li, and Bo Tang. 2021. Marrying Top-k with Skyline Queries: Relaxing the Preference Input While Producing Output of Controllable Size. In *Proceedings of the 2021 International Conference on Management of Data* (Virtual Event, China) *(SIGMOD/PODS '21)*. Association for Computing Machinery, New York, NY, USA, 1317–1330.

[18] Altavish Nair. 2021. Boston Housing Dataset. https://www.kaggle.com/datasets/altavish/boston-housing-dataset

[19] D. Papadias, Y. Tao, G. Fu, and B. Seeger. 2005. Progressive skyline computation in database systems. In *ACM Transactions on Database Systems (TODS)*, Vol. 30. ACM, 41–82.

[20] Jaykumar Parekh. 2023. House Price Prediction Using Linear Regression Model. *International Journal For Multidisciplinary Research* (2023). https://api.semanticscholar.org/CorpusID:266805860

[21] Andrew Pavlo, Matthew Butrovich, Ananya Joshi, Lin Ma, Prashanth Menon, Dana Van Aken, Lisa Lee, and Ruslan Salakhutdinov. 2022. External vs. Internal: An Essay on Machine Learning Agents for Autonomous Database Management Systems. *IEEE Data Eng. Bull.* 42 (2022), 32–46. https://api.semanticscholar.org/CorpusID:202548908

[22] L. Qin, J. Yu, and L. Chang. 2012. Diversifying top-k results. In *Proceedings of the VLDB Endowment*.

[23] Samkit Saraf. 2021. House Price Prediction Using Linear Regression. *International Journal for Research in Applied Science and Engineering Technology* (2021). https://api.semanticscholar.org/CorpusID:240448667

[24] M. Soliman, I. Ilyas, and K. Chen-Chuan Chang. 2007. Top-k query processing in uncertain databases. In *Proceedings of International Conference on Data Engineering*. IEEE, 896–905.

[25] Y. Tao, L. Ding, and J. Pei. 2009. Distance-based representative skyline. In *Proceedings of International Conference on Data Engineering*.

[26] Y. Tao, X. Xiao, and J. Pei. 2007. Efficient Skyline and Top-k Retrieval in Subspaces. In *TKDE*.

[27] Jeffrey Scott Vitter. 1985. Random sampling with a reservoir. *ACM Transactions on Mathematical Software (TOMS)* 11, 1 (1985), 37–57.

[28] T. Xia, D. Zhang, and Y. Tao. 2008. On skylineing with flexible dominance relation. In *Proceedings of International Conference on Data Engineering*.