Optimizing E-commerce Efficiency: A Data Mining Approach for Profitability

Garvit Jairath
Purdue University
gjairath@purdue.edu

***Abstract*—E-commerce businesses face the challenge of optimizing efficiency and profitability in a dynamic market. Leveraging data mining techniques can provide valuable insights and solutions to address this challenge.**

I. Introduction

In this project, I am addressing a data mining problem within the e-commerce domain. The primary challenge is to optimize overall business efficiency and profitability by leveraging data mining techniques. The specific problems I am focusing on include optimized inventory management to forecast customer purchase to build a reliable recommendation system.

The successful resolution of these data mining problems holds significant potential for positive impact. Optimized inventory management ensures resources are used efficiently and enhances targeted approaches, maximizing the impact of promotional efforts. In essence, solving these problems leads to a streamlined and more profitable e-commerce operation, positively affecting both the business and its customers.

II. Problem Formulation

To enhance overall business efficiency and profitability in e-commerce, I formulate the following data mining problems:

1) KMeans Clustering (4)
2) Decision Tree (5)
3) Random Forest (6)
4) SVM (7)

III. Dataset and Statistics

I utilize the E-Commerce dataset available on UCI, accessible at the following link: https://archive.ics.uci.edu/dataset/352/online+retail (2). This dataset contains several bits of information like invoice numbers, description of items, quantities of such items, unit price and country it was located in and more.

A. Brief Analysis

The dataset has eight columns with different types, all explained using the Pandas Library (1). The ”InvoiceNo” feature is a unique identifier for each sale. If it starts with ”C,” it means the item is canceled, something that will be handled in data pre-processing. Other features are easy to understand, and we can calculate ”sales” by multiplying quantity with unit price.

Column	Dtype
InvoiceNo	object
StockCode	object
Description	object
Quantity	int64
InvoiceDate	object
UnitPrice	float64
CustomerID	float64
Country	object

TABLE I: Overview of the Dataset Columns (Using Pandas)

Column	Count	Unique	Top
InvoiceNo	25015	1166	537434
StockCode	25015	2540	85123A
Description	24904	2478	WHITE HANGING HEART
Country	25015	19	United Kingdom

TABLE II: Data Summary

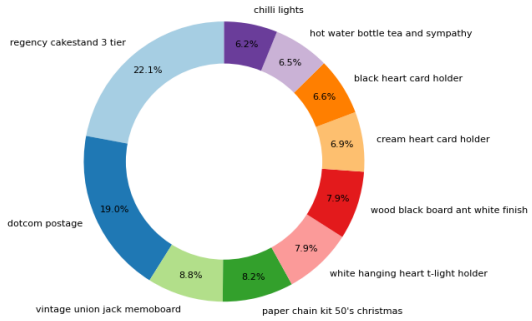


Fig. 1: Top Products By Sales

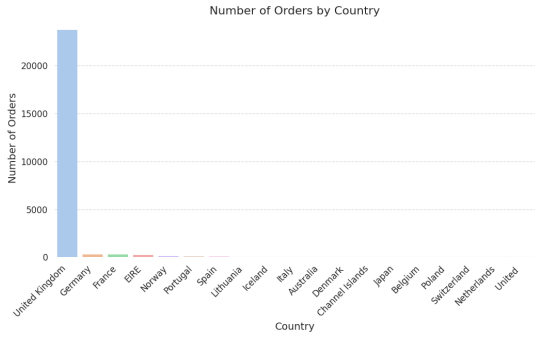


Fig. 2: Top Countries By Orders/Sales

With Pandas, we can find null values in the data pre-

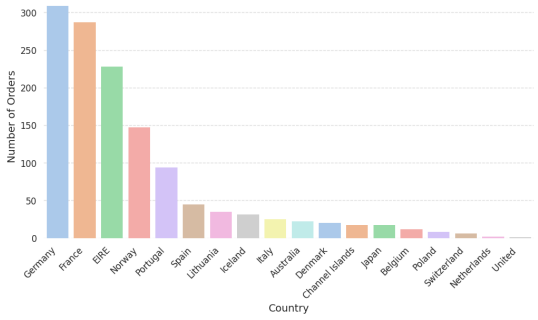


Fig. 3: Top Countries By Orders/Sales (Minus UK)

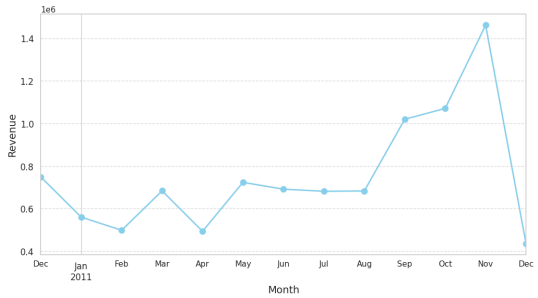


Fig. 4: Monthly Revenues

processing step and also use the library to explore and visualize the data. With this, it was found that most items are sold are in categories like home-decorations and clothing. Since the data was collected mostly in the UK, to acquire a more complete picture, I check sales by countries by excluding the UK. Germany comes second, followed by France. The noticeable difference in the number of orders is another issue that will be tackled during the data pre-preprocessing step.

Finally, the monthly revenues per month suggests that sales peak during November, the obvious reason for this is due to the holidays. Many responsible shoppers remember to buy gifts early. Following this month is a linear decline all the way to December, by which time its too late to buy gifts. This is further confirmed by the point in August. Although this might be confusing for anyone in the US, August is summer time in most European countries. Not only this, August is a month of many world-renown festivals and is probably another explanation of the seasonal high leading to November. It's very likely that people are buying new summer clothes to fit into, or are conforming to new fashion trends or are perhaps redoing their interior design since they have more time given that summer is usually more relaxed in terms of work as well. It tapers off from September to October, this is probably because by then people are more concerned with unwinding their vacations rather than buying new clothes or home decoration (as they happen to be the most popular product categories). In-fact, I might be overdoing this but I think its really interesting to note that alot of these spikes have very intuitive reasons. For instance, halfway through February and March there is a bump, the most obvious answer is Valentines day. It can be observed through items like "black heart" or "cream heart" cards that are found on the pie chart above. The reason I believe seasonal trends are sufficient to explain buying conditions is simple. Imagine if we were to remove all special occasions, then shoppers would have no reason to shop but for essential items. That is exactly what the baseline is around Jan 2011, I assume that is the bare-minimum retail operations that are being conducted. There is further evidence supporting my argument in May, June and July. Given the lack of some external force, there will never be any movement. It's simple physics, seasonal trends and other occasions dictate sale spikes and they are quite significant in nature. Again, the conclusions I've made above are reasonable. I can provide an unreasonable conclusion to offer perspective. If I claim that these seasonal trends are responsible for sales volumes, then it's difficult to explain why the US, given a larger population and consumer driven economy, has such low sales volumes in the graph. Hence the corollary, in this case, does not hold. This is because most of the data was simply collected around the European regions as mentioned by the UCI description of the data-set.

B. Data Pre-processing

The following methodology outlines steps taken to clean the data:

1) **Missing Values** First I checked how many rows were missing in the data and found CustomerID and Description were the only missing them, all these rows were removed, since its hard to tell who purchased what and it's essential for building a product recommendation/clustering system.
2) **Duplicate Values** The simplest thing to do is to just remove any excess rows, rather than doing some form of encoding since any incorrect assumption might result in bias. We can always obtain more realistic and reliable data points that are not duplicates or missing, rather than changing existing ones with a low confidence level or some form of uncertainty.
3) **Cancelled Transactions** Upon reviewing the dataset, I found something that wasn't mentioned in the UCI dataset, any invoice beginning with the letter "C" is a cancelled or refunded transaction. Since the percentage is not high relative to the non-cancelled transactions, it can be useful for the recommendation system to use these labels to not suggest items that are more likely to be cancelled. As a result of which these are kept.
4) **Price Deviations** There are some prices that are negative, some description columns with outlying values and stock codes that are sentences, all of these are removed. Even if all of them are combined, they represent a very small fraction of the dataset and offer no realistic picture of a transaction.

C. Evaluation Metrics

1. KMeans Clustering Parameters

- Number of Clusters: Range(3, 10)
- Initialization Method of Clusters: Smart or k-means++
- Number of times run with different seeds: 30
- Train/Test Split: 75%
- Method to determine optimal number of clusters: Silhouette Score tolerance to split into 4 clusters.
- Evaluation Metric: Silhouette Score

2. Decision Tree Parameters

- Criterion: entropy, gini
- Max Features: sqrt(total features)
- Train/Test Split: 75%
- Minimum number of samples to be at a leaf: 1
- Max Depth: None
- Evaluation Metric: Cross Validation

3. Random Forest Parameters

- Criterion: entropy, gini
- Max Features: sqrt(total features)
- Train/Test Split: 75%
- Number of Estimators: 40 to 100 in increments of 20.
- Max Depth: 4 to 6 in increments of 2.
- Method to find optimal hyper-parmeters: Gridsearch
- Evaluation Metric: Cross Validation

4. Support Vector Machines (SVM)

- Loss Function: Hinge (Default as per sklearn)
- Train/Test Split: 75%
- Maximum iterations: 100
- Value for C: $\log_{10}$ spaced values from 10^{-4} to 10^{4}
- Method to find optimal hyper-parmeters: Gridsearch
- Evaluation Metric: Cross Validation

D. Experimental Setup

I conducted the experiments using Google Colab as my computing environment. The key libraries utilized for algorithm implementation, data manipulation, and analysis are:

- **Scikit-Learn:** (3) Employed for implementing machine learning algorithms, grid search, and model evaluation.
- **Pandas:** (1) Utilized for data manipulation and analysis, facilitating efficient handling of data-sets.

For grid search and parameter tuning, I followed the examples and guidelines provided by Scikit-Learn's official documentation. The experiments involve the exploration of various hyperparameters using the Scikit-Learn API within the Colab environment.

Default values are used for parameters not explicitly mentioned in the above section.

IV. Algorithms

A. KMeans Clustering

Explanation: KMeans is a clustering algorithm that partitions data into distinct groups based on similarity. In e-commerce, it can be used to segment customers into clusters with similar purchasing behavior.

Justification: Asides from the fact that all algorithms here were discussed in class, KMeans is an efficient and simple clustering algorithm that can offer insights into the underlying data. Not only this, it can serve as a baseline for all future modelling endeavors. KMeans clustering can offer some form of groupings of customer segments that can allow for similar users in that product category to get product recommendations that they may not have, but another user in that same cluster might. The algorithm involves iterative steps of assigning data points to clusters and updating cluster centroids, facilitating the identification of customer segments for targeted marketing.

Input: Customer data from the e-commerce dataset, including purchasing behavior, preferences, and related features.

Expected Output: Segment customers into clusters based on similarities in their purchasing behavior or product preferences. From that point, we can either use similar items in similar clusters or categorical products from clusters to derive item assignments, and evaluate the clusters using intrinsic metrics. The latter is an important point since all operations revolve around inside the cluster.

B. Decision Tree (DT)

Explanation: Decision Trees are chosen for e-commerce due to their interpretability, ability to handle non-linear patterns, and transparent feature importance ranking. Their adaptability to mixed data types and scalability make them suitable for large datasets. In recommendation systems, Decision Trees provide logical rules for personalized suggestions.

Justification: Firstly, this algorithm is covered in depth in class. Secondly, Decision Trees excel in capturing complex decision-making processes, which is crucial for e-commerce scenarios with different user behaviors and preferences. Their ability to break down decisions into a series of simple, interpretable steps aligns well with the need for transparency in providing personalized product recommendations. Additionally, the hierarchical structure of Decision Trees makes it easy to identify key features influencing recommendations, aiding businesses in refining their strategies.

Input: Customer data from the e-commerce dataset

Expected Output: A Decision Tree model that could using the Gini-index, create meaningful splits that can help create personalized recommendations for users that can help drive product sales.

C. Random Forest

Explanation: Random Forests, an ensemble learning method, build multiple Decision Trees and merge their outputs to enhance overall performance. In e-commerce, this approach mitigates overfitting, increases accuracy, and provides robust predictions. The diversity among the constituent trees allows the model to capture a broader range of patterns in user behavior.

Justification: While Decision Trees offer transparency and interpretability, Random Forests go a step further by reducing overfitting present in individual trees. The aggregation of predictions from multiple trees helps improve the model's

generalization capabilities, making it well-suited for the dynamic and evolving nature of e-commerce data. Additionally, the feature randomness introduced during tree construction enhances the model's adaptability to various scenarios, making it a useful tool for producing reliable recommendations.

Input: Customer data from the e-commerce dataset

Expected Output: A trained Random Forest model composed of multiple Decision Trees, each utilizing the Gini-index to create meaningful splits based on different subsets of the provided features.

D. Support Vector Machines (SVM)

Explanation: Support Vector Machines (SVM) is a powerful supervised learning algorithm that aims to find the optimal hyperplane that separates different classes in the input space. In the realm of e-commerce, SVM can effectively classify users based on various features, providing a clear decision boundary for personalized recommendation strategies. SVM is particularly advantageous in handling complex, non-linear relationships within the data.

Justification: While Random Forests excel in ensemble learning and capturing diverse patterns, SVM stands out for its ability to handle intricate decision boundaries. In the context of e-commerce, where user behavior may exhibit non-linear relationships, SVM can effectively discern and classify different user segments. This makes it a valuable tool for generating accurate predictions and enhancing the overall recommendation system.

Expected Output: A trained SVM model that establishes an optimal hyperplane to effectively classify users based on the provided features. This decision boundary allows for the identification of distinct user segments, allowing for the creation of personalized recommendations in the dynamic e-commerce environment.

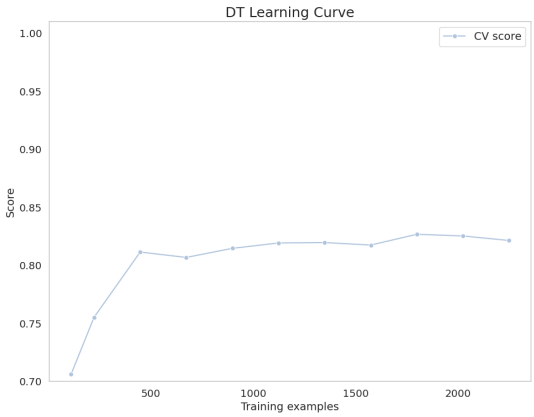
V. Results and Discussion

A. KMeans Algorithm

After doing the split and fitting the KMeans algorithm to the parameters as described above, I found with the use of the silhouette scores that number of clusters equal to four gave the best performance. After using the cluster values to see what the top-k most occurring items were, I found that there were several repeated items in many of the clusters. The fact that there is such a low silhouette score indicates that this is clearly not the best method to run on the data-set. The KMeans performs poorly in both aspects, speaking strictly from an evaluation perspective and from an analytical perspective. This is surprising to me since it goes against what I assumed would be an optimal way to approach the problem, since my plan was to cluster products in different categories and return the top-k most similar suggestions based on user purchasing trends. This is similar to collaborative filtering, where users are given similar recommendations based on similar preferences (as the data-set seemed to conform to such simplicity). The silhouette score here was a mere 0.24. Although this isn't much, using the elbow method, an optimal number of clusters were discovered. My idea was to take all of the similar ranked product categories and place customers that had purchased them into similar clusters. This algorithm cannot be used for recommendation due to the significant overlap in these clusters but it does offer insight into the underlying behavior of the data-set and the nature of the product categories present inside of it.

B. Decision Tree

Decision Tree outperformed KMeans and SVM, achieving an impressive 84.49% score on the testing dataset. Its success lies in its ability to capture complex non-linear patterns, provide transparent decision rules, and adaptation to diverse data types. The feature importance ranking further aids in understanding influential factors. Unlike KMeans and SVM, Decision Trees make minimal assumptions about data distribution, ensuring flexibility. Efficient scalability and effective handling of imbalanced datasets contribute to its superior performance in this classification task. Since from KMeans it was found that there is an overlap in clusters, Decision Tree is able to forgo the strong assumptions KMeans makes to find clear splits at each nodes, giving us with logical rules that can make for quite a robust recommendation system and those rules can be further interpreted in the form of some forensic analysis to help the business in question to understand and therefore conduct its operations better.



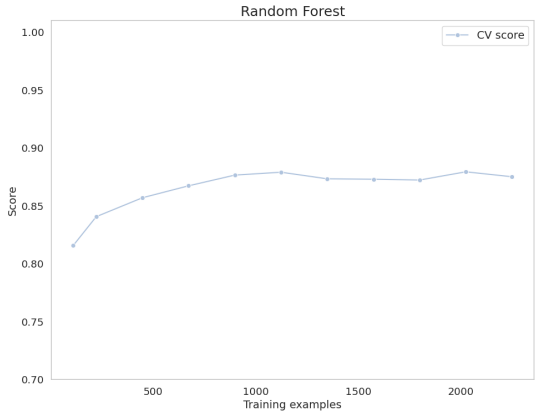
C. Support Vector Machines (SVM)

After doing a 75% split on the training and testing data, we can now run via five fold cross validation on the SVM algorithm. Upon running, the prediction score was found to be 63.25%. Although this isn't much, it indicates better performance than its KMeans counterpart. SVM outperformed KMeans due to its supervised nature, optimized decision boundaries, consideration of class imbalances, and parameter tuning opportunities. SVM's focus on classification aligns better with the task's evaluation metric, contributing to its superior performance in comparison. SVM's lower performance compared to Decision Tree (84.49%) may stem from its inherent assumption of linear decision boundaries, limiting its ability to capture complex non-linear patterns present in the data. The

complexity of SVM and its sensitivity to hyper-parameters could lead to sub-optimal tuning for this specific data-set. Additionally, SVM's less interpretable decision boundaries might hinder an intuitive understanding of the underlying patterns, while Decision Tree's transparent rules provide much more clearer insights.

D. Random Forest

Random Forest dominated all of the algorithms with a score of 89.31%. The ensemble of Decision Trees in Random Forest leverages the strength of multiple models, surpassing the individual capabilities of single decision trees used in isolation, as seen in the Decision Tree. While Decision Tree is susceptible to over-fitting, Random Forest excels by aggregating predictions, reducing over-fitting risks compared to a single Decision Tree. SVM's rigid linear decision boundaries may not adapt well to complex data. Random Forest's introduction of feature randomness enables adaptability, overcoming SVM's limitations. Random Forest's ability to handle mixed data types and imbalances addresses limitations seen in KMeans and SVM, making it more versatile in real-world scenarios.



VI. Bonus

A. Comparison of three or more algorithms on same dataset

The comparison was discussed in Section-V. Here I will speak more on what types of data the algorithms work on. For KMeans, the low silhouette score and the presence of repeated items in the top-k clusters suggest that the inherent cluster structure within the data may not align well with the assumptions made by the KMeans algorithm. KMeans relies on the assumption that the clusters are spherical and equally sized, which might not always be the case. There can be several reasons for this; Firstly, there is geographical bias in the dataset, as most of it is from the UK (Either you remove other countries, or introduce more of them). Secondly, during the data analysis stage, I noticed several items were being repeated. Although a human can tell the difference, it's possible KMeans struggles to tell the subtle nuanced differences between some of these products and their categories (so you need more products that are different in the same category, or more categories). Thirdly, the most frequent items in the data-set, as seen during the data analysis stage, all belonged relatively to the same categories all of which have overlapping traits. Speaking strictly from a data-science perspective, KMeans simply cannot form well separated distinct boundaries in such cases. However, some solutions can still be used if it's essential to only use KMeans here; The first could be feature-engineering, more features can be incorporated to stop the domination of geographical and categorical biases (tough to do without more data, will have to use existing data to form features that are not arbitrary in nature thus there is a finite cap on what can be done on this data-set). Secondly, techniques like adjusting weights for each feature can be used to tackle the data imbalance that is present in the data-set. Although KMeans is easy and efficient to implement, it's sensitive to clusters of varying sizes and the afromentioned data imbalances. To propose solutions to improve SVM, different kernel functions (polynomial, quadratic etc) can be used. Also, a more rigorous grid search can also be done if a stronger computational cluster is available. The number of features or feature importance can also be scaled up or down to improve performance. SVM is efficient in a higher dimensional space, but is sensitive with noise. Although the data was cleaned and pre-processed, it still retains a low score given the imbalance that is present in the data-set and the learning barriers that come alongside the formulation of a good kernel function. For the decision tree, better pruning methods can be employed. This conclusion is derived from the fact that it lags behind the random forest and that is probably due to some form of over-fitting. Furthermore, more fine tuning of hyper parameters can also be done to cover a larger space of parameters than I did due to the limitations posed by the computational cluster. Lastly, the only changes that can be made would be to tackle the data-set itself. That is, improvements will mostly revolve around the underlying training data like the collection of better data samples.

B. Challenges faced and advanced techniques employed

The data-set clearly suffers from imbalance when it comes to geographical biases and also categorical biases. The most obvious technique to remedy this is to collect more data from different places. However, that is an expensive task. The second solution is to distribute the data more uniformly across each feature category. In my opinion, the best solution is to standardize or assign a weight to each feature that can effectively highlight the nuances of these biases that are present. This will allow algorithms like KMeans or SVM to not get biased while making its decision boundaries. Although this may not be the only challenge present in the data-set, the results I have obtained clearly indicate it is one of them. To address this, I tried a few advanced techniques. For example, I attempted to use z-scores and anomaly detection to tackle some of the data outliers. I also attempted to standardize geographical biases present in the data-set. Upon doing this, the silhouette score only saw a slight increase of 0.15. This trend was present across all algorithms as they all saw only a minor increase in their respective scores. After doing more

exploration, I found out the issue that is unavoidable in this data-set is simply the fact that there aren't enough rich features. For example, I attempted to make a column called "season" to address seasonal purchases. As mentioned in before sections, seasonal trends dictate, to a large extent, spikes in purchasing power. However, modelling the weight of this feature in the model was a very complex task, as I initially assigned the column a boolean value. This boolean value did nothing but create a bias towards seasonal trends and ignored the baseline retail operations. Afterwards, I gave it an arbitary integer value which was still not enough to capture its importance reliably. This is because there is no accurate way for me to quantify which month has what importance. Not only this, but there is no realistic way I can keep track of every seasonal trend known in European countries not to mention other countries. Therefore, it's very difficult to scale.

As a result, I think that there is no other way to circumnavigate the bias present in the data-set other than performing three tasks. First, more data needs to be collected from different places. Even if it has to be from the UK, at-least it must be from different cities. Second, the data needs to collected from different shops/stores to handle categorical bias. Lastly, the data needs to be collected in "time deltas". In other words, the shopper must be questioned whether this is a seasonal purchase or a normal purchase, so we can accurately determine the count or frequency of spikes during seasonal trends which vary from place to place and cannot simply be given any arbitrary number. This is an excellent solution as it takes away the burden of assumption from the data-analyst and gives it to the shopper instead. Although the assignment of an arbitrary number might give us temporary results and make the analyst happy in the short term, the results will not be reliable or robust enough to form actually meaningful recommendations. Also, this comes at the cost of scalability as it is not feasible to keep on applying some numerical value to seasons, not to mention keeping track of every season or special occasion.

C. Further comparison with case-study

Illustrating my argument further, I draw parallels with Target's renowned recommendation system. Target's predictive success in identifying a girl's pregnancy (8) was rooted in an extensive analysis of purchasing history. Target collected all sorts of "varied" features. Customer demographics, response to coupons sent, number of coupons ignored, purchase during seasonal trends, frequency of purchases and in which category, was the item under discount/sale etc. Not only did this data allow for a robust clustering of distinct customer groups, it also allowed for very meaningful draws or parallels between products and product categories. It implicitly creates a strong boundary between items of similar and dissimilar types, and customers of similar and dissimilar preferences. In simple terms, 90% of the job was basically done during the data collection step. Literally any algorithm can work on this since the data available is just so good.

Although the algorithmic choices and tuning of its respective parameters is just as important as the data collection step, the results will not be as specific or as precise if we were to have a more rich source of data available to us. This is precisely why Target was able to predict simply through a purchase of a scented candle **and/or** the discontinuous purchase of something like alcohol or other unhealthy items, a change in this specific users lifestyle. This form of pattern recognition requires good data-engineering but moreover requires a rich source of data.

It would be interesting to note, the entire data-set that I used above, would only be a tiny subset of the original subset of the data-set that Target probably possess. Not to mention that Target's entire data collection is automated so scaling it is even easier. Online purchases have meta-data associated with users and physical purchases are tracked through the credit-card companies. This is why they incentivise making an account on their platform by offering discounts and in some cases do not even provide an option to purchase an item without the ownership of one. This offers a glimpse into the potential of rich data and of the value of the people who can actually acquire good data.

In conclusion, the quantity of data is not as important as its quality. In the data-set above, although many attempts were made from simple to advanced techniques to attempt to sort and classify/cluster the data as efficiently as possible, the underlying dependence on the data cannot be eliminated, it can only be compensated.

REFERENCES

[1] pandas-dev/pandas: Pandas The pandas development team feb 2020 Zenodo latest 10.5281/zenodo.3509134 https://doi.org/10.5281/zenodo.3509134

[2] Online Retail. (2015). UCI Machine Learning Repository. https://doi.org/10.24432/C5BW33.

[3] Scikit-learn: Machine Learning in Python scikit-learn Pedregosa, F. and Varoquaux, G. and Gramfort, A. and Michel, V. and Thirion, B. and Grisel, O. and Blondel, M. and Prettenhofer, P. and Weiss, R. et al, E. Journal of Machine Learning Research volume 12 pages 2825–2830 2011

[4] Jin, X., Han, J. (2011). K-Means Clustering. In: Sammut, C., Webb, G.I. (eds) Encyclopedia of Machine Learning. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-30164-8_425

[5] Wu, X., Kumar, V., Quinlan, J. R., Ghosh, J., Yang, Q., Motoda, H., ... others. (2008). Top 10 algorithms in data mining. Knowledge and Information Systems, 14(1), 1–37.

[6] Ho, T. K. (1995). Random decision forests. In Proceedings of 3rd international conference on document analysis and recognition (Vol. 1, pp. 278–282).

[7] Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine Learning, 20(3), 273–297.

[8] Kashmirhill Forbes How target figured out a teen was pregnant before her father did 2012