UNSUPERVISED MACHINE LEARNING

PROJECT

SUBMITTED TO

SVKM'S NMIMS (DEEMED- TO- BE UNIVERSITY)

IN PARTIAL FULFILLMENT FOR THE DEGREE OF

MASTER OF SCIENCE

IN

DATA SCIENCE

BY

KHUSHI PARTE
ROLL NO – A016; SAP ID – 86092400016
&
AANCHAL BAFNA
ROLL NO – A004; SAP ID – 86092400004

Directing Customers to Subscription Through App Behaviour Analysis

1.Problem Statement:

In today's highly competitive FinTech environment, retaining users and converting them into paying customers is essential for long-term success. The FinTech company in this project launched a mobile application offering a 24-hour free trial to attract new users. During this trial period, the app captures detailed behavioral data such as login time, types of screens visited, session duration, and whether the user enrolled after the trial. Despite this data being available, a large number of users still fail to convert into paid subscribers. The core challenge lies in the lack of clear segmentation of users based on their in-app behavior. Without this segmentation, it becomes difficult to identify patterns that lead to successful conversions or to flag users who are likely to drop off. As a result, the company struggles to gain actionable insights that could help improve user engagement and subscription rates.

2. Solutions

The primary aim of our project is to uncover meaningful patterns in user behavior during the app's 24-hour free trial period, using unsupervised machine learning techniques. By clustering users based on their in-app activities, the goal is to help the business understand how different users engage with the app, which behavioral patterns lead to successful conversions, and where potential drop-offs occur. These insights can then be used to tailor the user experience, improve retention strategies, and implement personalized marketing campaigns that cater to the needs of specific user groups. To achieve this, we applied **K-Means Clustering** to segment users based on behavioral features, **DBSCAN** to detect irregular usage and validate cluster quality, and **Principal Component Analysis (PCA)** to reduce dimensionality and allow visual interpretation of user groupings.

The two main objectives of our study are:

- To segment users into meaningful groups based on their behavioral patterns using unsupervised learning.
- To identify user traits that are closely linked to conversion or drop-off, aiding in targeted engagement strategies.

3. Methodology
3.1 Datasets:

Dataset 1: FineTech_appData.csv

This was the **primary dataset** used for the clustering analysis. It contains user-level behavioral data collected during the 24-hour free trial period. Key columns included:

1. Size: Thousands of rows (user records), 12 columns.
2. Key columns:
    - **hour**: First app open time.
    - **screen_list**: Comma-separated screen names visited by the user.
    - **num_screens**: Number of screens visited.
    - **enrolled**: 1 if the user enrolled; 0 otherwise.
    - **enrolled_date**: Timestamp of enrollment.
    - **difference**: Derived feature — time (in hours) between first app open and enrollment.

We used this dataset to extract meaningful features like screen counts, frequency of feature access (e.g., loan, credit), time of login, and whether the user converted, which served as the foundation for clustering.

Dataset 2: top_screens.csv

This supporting dataset contained a **curated list of top screens** in the app (e.g., Loan1, Credit3, Offers, etc.) which were considered important touchpoints in the user journey. It was used to:

- Contains names of the top app screens most frequently visited.
- Used to transform the screen_list into meaningful features.
- Screens grouped into categories:
    - **Loan Screens**
    - **Credit Screens**
    - **CC (Credit Card) Screens**
    - **Savings Screens**

By integrating this file, we were able to engineer richer features that reflected how users interacted with different parts of the app, which improved the quality of clustering.

3.2 Data Preprocessing & Feature Extraction:

1. Datetime Parsing:

- first_open and enrolled_date were parsed using dateutil.parser to get accurate time formats.

2. Feature Engineering:

- Calculated difference (in hours) = time between app open and enrollment.
- Users with difference > 48 hours were treated as **non-enrolled**, updating the enrolled flag accordingly.

3. Screen List Transformation:

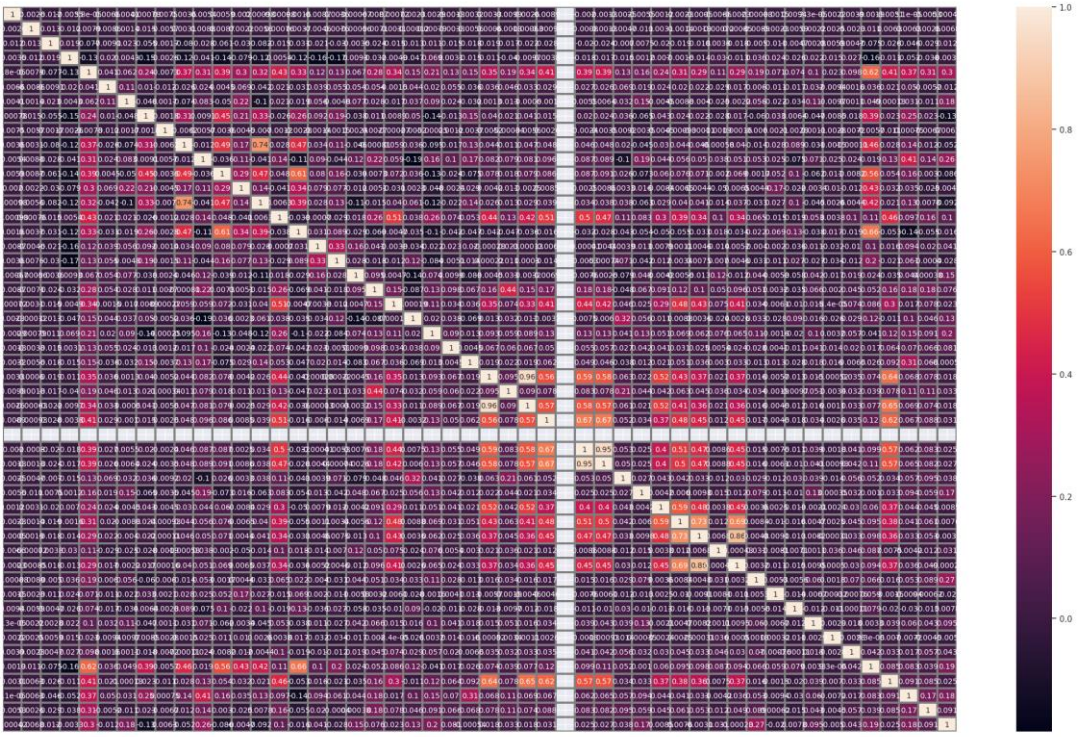
- Converted the long string in screen_list into individual binary columns using top_screens.csv.
- For each top screen (e.g., Loan2, Credit1, Savings2), added a binary column indicating if that screen was visited.
- Removed the screen from the list once matched — reducing redundancy.
- Grouped screens to form aggregate features:
    - Credit Count, Savings Count, Loan Count, CC Count.

4. Data Cleaning:

- Dropped irrelevant columns like screen_list and enrolled_date.
- Ensured no missing or duplicate values remained.

5. Feature Scaling:

- Applied StandardScaler to normalize numeric features and avoid bias in clustering.



This heatmap displays the correlation matrix of all features in the dataset. Most features show low correlation (closer to 0), indicating minimal multicollinearity, while a few screen-related features form small clusters with moderate positive correlations, suggesting grouped user behaviour patterns.

3.3 Methods & Techniques Used:

A] K-Means Clustering

We started with **K-Means Clustering** because it's a straightforward and commonly used method to group similar users. It works by dividing users into a fixed number of clusters based on their behavior like screen visits, conversion status, and time spent in the app.To decide how many clusters to use, we applied the **Elbow Method**, which helps identify the point where adding more clusters doesn't really improve the results. Based on that, we found a suitable number of clusters and ran K-Means on the processed data.The output grouped users in meaningful ways like one cluster had highly engaged users who explored many features, while another had quick converters. This helped us understand different user types and think about how to improve their app experience.

B] DBSCAN (Density-Based Clustering)

While K-Means worked well, it has some downsides — like it doesn't handle outliers (unusual user behavior) very well, and it assumes that clusters are evenly shaped. So, we also used **DBSCAN**, which is another clustering method that works quite differently. DBSCAN looks for users who are closely packed together in terms of behavior and automatically separates out those who don't fit anywhere these are considered outliers. So, even though the main analysis was done using K-Means, DBSCAN gave an extra layer of understanding and helped validate the overall clustering approach.

C] PCA (Principal Component Analysis)

After transforming the data like converting screen visits into separate columns the dataset had too many dimensions to easily visualize or work with. To make it easier to understand and present, we used **PCA**, which reduces the number of features while keeping the important information. Using PCA, we brought the data down to just two dimensions, which made it possible to **plot the user clusters visually**. This really helped in seeing how well the users were grouped, especially when comparing the clusters from K-Means and DBSCAN.It was super helpful not just for checking if the clustering made sense, but also for showing the results in a more intuitive way. The visualizations made it easy to spot which clusters were tight, which were spread out, and if there were any overlaps.

4. Evaluation Matric: Silhouette Score

K-Means Clustering:
The Average Silhouette Score for K-Means came out to approximately **0.42**, suggesting moderate cluster cohesion and reasonable separation between groups. While not perfect, this score indicates that the model was able to segment users into somewhat distinct groups based on behaviour. Visual inspection of the PCA plots also supported this structure.

DBSCAN:
DBSCAN gave a lower Silhouette Score, around **0.18**, primarily due to the presence of many noise points and overlapping clusters. However, this was expected since DBSCAN is not designed for creating balanced clusters like K-Means. Instead, its strength lies in detecting **outliers** and highlighting **irregular usage behavior**, which K-Means might miss.

K-Means was more effective for structured segmentation and general behaviour grouping where as **DBSCAN** added value by identifying unusual patterns and outlier users that don't fit the typical usage profile.

5. Results:

5.1 Customer Behavior Clustering

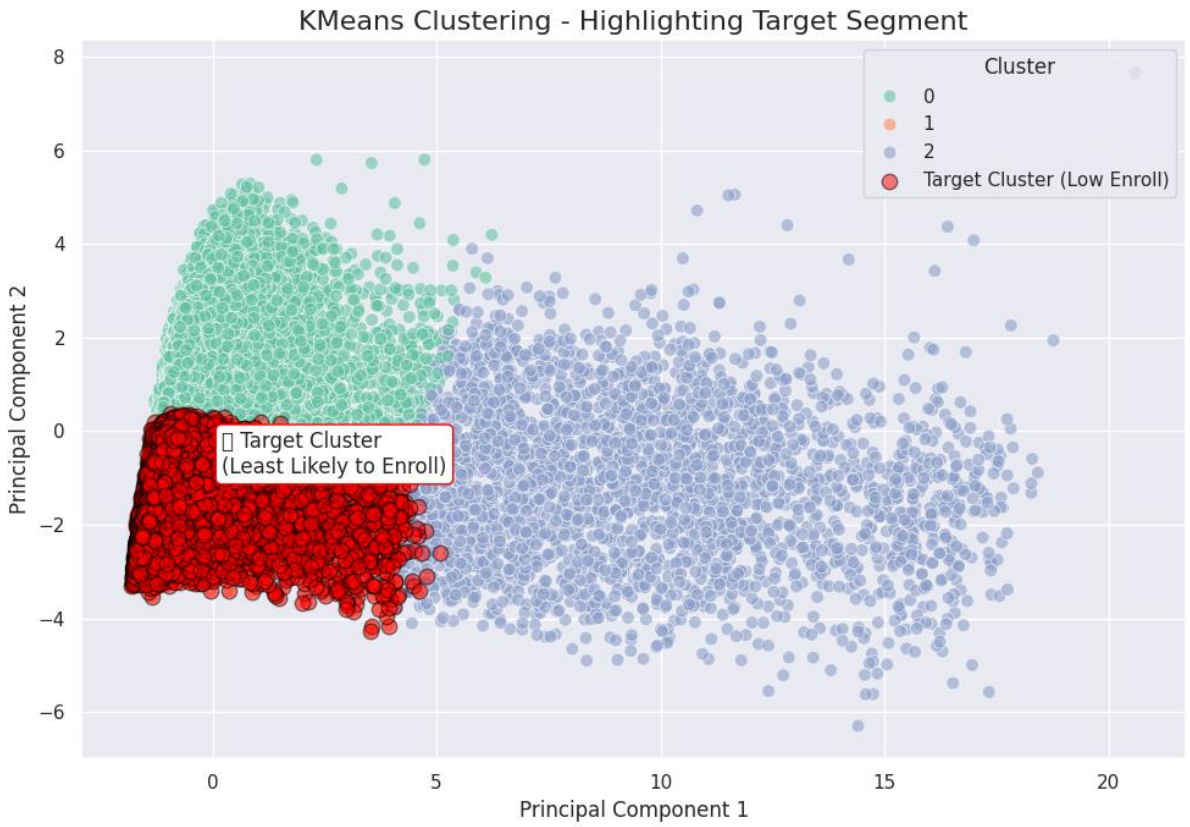
K-Means Clustering: The users are clearly segmented into 3 distinct clusters with minimal overlap. This indicates that K-Means was effective in identifying groups of users with similar in-app behaviours during the trial period. These clusters can be further analysed to understand behaviour patterns leading to conversions or drop-offs.

DBSCAN Clustering: DBSCAN resulted in a high number of small clusters, with many points marked as noise (label -1). This suggests that DBSCAN struggled to form meaningful clusters due to the dense and overlapping nature of the data, although it is effective for spotting outliers.

K-Means provided more interpretable and business-actionable segmentation compared to DBSCAN, which was more suitable for identifying anomalies rather than structured user groups.

5.2 KMeans Clustering- Highlighted Target Segment

This plot shows the output of KMeans clustering on customer behavior data reduced using PCA (Principal Component Analysis). Each point represents a user, colored by the cluster they belong to:



- **Cluster 1 (Red - Highlighted as Target Cluster):** This cluster is specifically marked as the **"Least Likely to Enroll"** group. It's dense, tightly packed in the lower-left PCA space, and has distinct behavioral patterns compared to others.
- **Cluster 0 (Green) and Cluster 2 (Blue):** These represent other user segments, likely with higher engagement or conversion behavior. Cluster 2 (Blue) seems to be the largest and most spread out, indicating varied but potentially more active behaviours.

6. Conclusion:

The project effectively utilized unsupervised learning techniques, specifically K-Means and DBSCAN, to segment app users based on their behavioral patterns. Through this approach, low-engagement users were identified as a distinct cluster, while key behavioral traits associated with higher subscription likelihood were also uncovered. Principal Component Analysis (PCA) was used to visualize the clusters, providing clear and actionable insights for strategic decision-making. These findings empower the business to implement personalized offers for low-engagement users and design targeted strategies to convert curious users, ultimately improving overall conversion rates and enhancing user engagement.