Optimizing Financial Advisor Matching: A Data-Driven Approach for Enhancing Policy Conversions

Zheyuan Lai, Yuxin Liu, Xiyao Ma, Jingyu Shi, Yuhan Wang*
Faculty of Science, National University of Singapore

February 2025

1 Introduction

The objective of this datathon is to develop a data-driven model that recommends the most suitable financial advisors to customers, optimizing the matching process to enhance policy adoption. By leveraging advanced machine learning techniques, the model aims to improve the efficiency and effectiveness of advisor assignments, ensuring that customers receive tailored recommendations based on their unique profiles. This, in turn, contributes to higher policy conversion rates, improved customer satisfaction, and increased revenue growth for the company.

To address challenges such as high-dimensional data and agent-customer compatibility, we employ a multi-stage approach integrating clustering, classification, and ranking models. Financial advisors are first grouped into representative clusters, followed by customer classification into these clusters. Finally, agents within each cluster are ranked based on relevant success factors.

Beyond predictive accuracy, our model integrates fairness and ethical considerations as a core part of evaluation, ensuring that recommendations are equitable, unbiased, and do not disproportionately favor specific demographic groups. By incorporating fairness metrics, we aim to promote responsible AI deployment in financial advisory services, aligning with Singlife's commitment to customer-centric, ethical, and data-driven decision-making.

*Listed in alphabetical order; All authors contribute equally;
Correspondence to: zheyuan_lai@u.nus.edu

2 Dataset Overview

2.1 Data exploration

Our initial data analysis revealed several key patterns influencing customer-agent interactions and policy choices:

1. Agent-Customer Gender Preference: Customers strongly prefer agents of the same gender. This emphasizes the importance of incorporating gender into the matching process, which can be show by Figure 1.

	pct_SX1_male	pct_SX2_female
pct_SX1_male	1	-0.971223705
pct_SX2_female	-0.971223705	1

Figure 1: Distribution of customer and agent gender pairings.

2. Product Premium Variability: Significant variability in product premiums suggests differing risk profiles and a potential need for agent specialization based on product characteristics, see Figure 2.

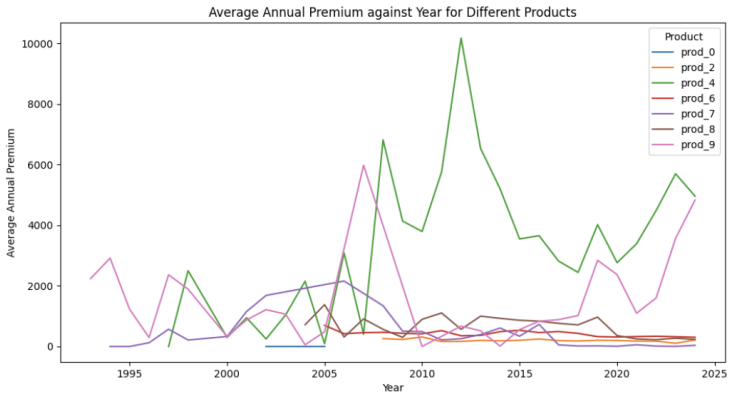

Figure 2: Variability in premium ranges over years.

3. Age and Product Selection: A strong correlation exists between customer age and product selection, indicating the necessity of age-based recommendations.

4. Family Size and Product Choice: Family size influences product choice, with larger families showing a preference for specific product groups, likely reflecting their unique financial needs.

5. Economic Status and Product Selection (Inconclusive): While economic status varies across product groups, a clear, consistent trend is not immediately apparent, requiring further investigation.

These findings are crucial for developing an effective agent-client matching model and informing business strategies.

2.2 Key Challenges

The dataset presents several challenges that must be addressed to develop an effective agent-client matching model.

High dimensionality of agent and customer data. The dataset contains a large number of features, making it difficult to identify key patterns and relationships. Proper dimensionality reduction and feature selection techniques are required to enhance interpretability and model efficiency.

Imbalance in agent-to-customer ratio. With a 3:1 customer-to-agent ratio, agent features are highly dispersed. This necessitates clustering techniques to group agents based on similar characteristics, allowing for a more structured and meaningful recommendation process.

Excessive number of variables. Even after merging datasets, the total number of variables remains high, with over 60 features. Selecting the most informative features while minimizing redundancy is crucial for improving model performance and reducing computational complexity.

Lack of negative examples. Since all policies in the dataset were successfully converted, there are no explicit negative examples to differentiate successful and unsuccessful matches. An alternative variable or proxy metric must be identified to construct a balanced classification problem.

Absence of a clear match quality metric. While the dataset records existing client-agent assignments, it lacks an explicit measure of match effectiveness. This requires the development of custom evaluation criteria to assess agent suitability and optimize the recommendation system.

Ensuring fair and unbiased recommendations. The dataset contains demographic attributes such as gender, marital status, and economic status, which may introduce biases into the agent-client matching process. To prevent unfair advantages or disadvantages, we need to carefully assess model outputs and implement fairness-aware adjustments, ensuring that recommendations remain equitable across different demographic groups.

Addressing these challenges is essential for improving model accuracy, ensuring interpretability, and refining the agent-client matching process.

2.3 Data Preprocessing

To ensure data consistency, reduce computational overhead, and prepare the dataset for machine learning applications, we applied a structured data preprocessing pipeline.

Merging Multiple CSV Files. The dataset was originally provided in three separate files: `agent_data.csv`, `client_data.csv`, and `policy_data.csv`. Processing them individually led to challenges such as inconsistent indexing, missing cross-references, and increased computational complexity. To streamline

data integration and maintain integrity, these files were merged into a single comprehensive dataset, `merged_data.csv`.

Handling Duplicates and Missing Values. Duplicate rows were removed, blank values were replaced with NaN, and missing data was handled appropriately by dropping incomplete records where necessary.

Categorical Data Encoding. Non-ordered categorical variables were one-hot encoded, while ordinal variables were encoded using ordinal encoding to preserve their inherent order.

Date Parsing and Age Calculation. Date columns were converted to datetime format, and client age (`cltage`) was computed based on birthdate and policy inception date.

Agent Expertise Encoding. The `agent_product_expertise` field, initially a list of product categories, was transformed into binary indicators, with each product expertise represented as a separate column.

ID Transformation and Aggregation. Aggregated agent-level statistics were computed, including average `economic_status`, `household_size`, and `family_size`, ensuring that meaningful summaries of agent attributes were incorporated into the dataset.

Redundant Column Removal. Irrelevant or highly correlated columns were dropped to reduce dimensionality and improve computational efficiency.

Feature Engineering. Additional derived features were created, including customer age group classifications, missing product indicator adjustments, and a policy stability metric (`net_indicator`) to better capture policy success rates.

Final Data Export. The cleaned and processed dataset was saved for further modeling and analysis.

This preprocessing workflow ensures data consistency, computational efficiency, and optimal readiness for machine learning applications.

3 Methodology

We propose two approaches for agent-client matching: a structured clustering-classification-ranking pipeline and a representation learning-based method. The first clusters agents using GMM, classifies clients via Random Forest, and ranks advisors within clusters using XGBoost. The second employs transformer-based embeddings, refining agent-client matching through contrastive learning and cosine similarity. While the first ensures interpretability and efficiency, the second captures latent relationships and adapts to incomplete data. The overall structure of our models is summarized in Figure 3.

3.1 Agent-Client Matching via Clustering, Classification, and Ranking

To effectively match customers with the most suitable financial advisors, we implement a three-stage methodology: agent clustering, client classification, and

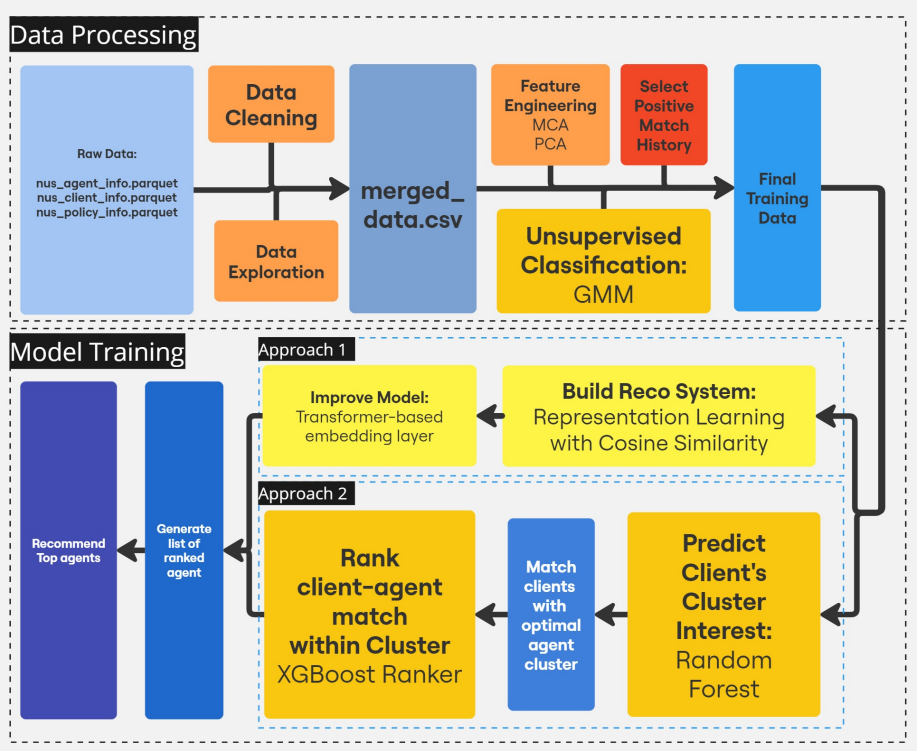

Figure 3: Model Structure

agent ranking. This structured approach enhances recommendation accuracy while addressing the complexity of high-dimensional data and agent sparsity.

First, we select training data based on the conditions annual premium greater than zero, then the policy history which were not lapsed nor expired. This functions to filter training data which were relatively successful, which ensures we identify patterns where clients and agents match well.

Next, we apply **Multiple Correspondence Analysis (MCA)** to reduce dimensionality and extract meaningful latent factors from categorical data. MCA transforms high-dimensional categorical variables into a lower-dimensional representation, improving the interpretability and efficiency of subsequent clustering. Given a dataset with p categorical variables, MCA finds principal components by decomposing the Burt matrix:

$$B = Z^T Z, \tag{1}$$

where Z is the standardized indicator matrix. The resulting principal components serve as input features for clustering.

Then, we apply a **Gaussian Mixture Model (GMM)** to cluster financial advisors based on their feature similarities. Given the high variability among agents, GMM is preferred over K-Means as it allows clusters to take elliptical forms and provides **soft assignments**. Mathematically, the model assumes agent features $\mathbf{x}$ follow a mixture of Gaussian distributions:

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x} \mid \mu_k, \Sigma_k), \tag{2}$$

where π_k represents the weight of the k-th cluster, and $\mathcal{N}(\mathbf{x} \mid \mu_k, \Sigma_k)$ is a multivariate Gaussian distribution with mean μ_k and covariance Σ_k. The **Expectation Maximization (EM)** algorithm optimizes these parameters by maximizing the likelihood of observed agent data.

Once agents are clustered, we classify clients into one of these agent clusters using a **Random Forest** classifier. This model is selected due to its robustness in handling categorical and numerical data while mitigating overfitting. Given a feature vector $\mathbf{x}$ for a client, the model predicts the most suitable agent cluster y:

$$\hat{y} = f_{\text{RF}}(\mathbf{x}), \tag{3}$$

where the final prediction is determined through an ensemble of decision trees using majority voting.

Following this, an **XGBoost Ranker** ranks the agents within the assigned cluster based on factors such as past conversion success and policy inforce rate. The ranking model optimizes a **pairwise ranking objective**, minimizing the loss function:

$$L = \sum_{(i,j)\in D} \log(1 + e^{-(s_i - s_j)}), \tag{4}$$

where s_i and s_j represent ranking scores of two agents, and D denotes the set of agent pairs to be ordered. The predicted scores are used to rank advisors within each cluster, ensuring that the best agent is recommended for each client.

This **three-stage approach** efficiently reduces computational complexity while enhancing prediction accuracy. Clustering organizes agents into meaningful groups, classification matches clients to these groups, and ranking refines the final recommendation to optimize business outcomes.

3.2 Representation Learning-Based Agent-Client Matching

To address the absence of labeled "good match" data, we propose a methodology that learns representations of agents and customers, enabling meaningful matching through learned embeddings. The approach consists of an unsupervised baseline using cosine similarity, a transformer-based embedding model, and a supervised fine-tuning phase incorporating contrastive learning. This framework enhances matching effectiveness while allowing for adaptation in cases where customer data is incomplete.

Initially, agent and customer data are preprocessed through feature engineering, one-hot encoding of categorical variables, and imputation of missing values. The baseline approach then ranks agents for each client using cosine similarity over the processed feature vectors. Given customer and agent representations $\mathbf{v}_c$ and $\mathbf{v}_a$, the similarity score is computed as:

$$\text{similarity}(c, a) = \frac{\mathbf{v}_c \cdot \mathbf{v}_a}{\|\mathbf{v}_c\| \|\mathbf{v}_a\|}. \tag{5}$$

Agents are ranked by similarity scores, with top-ranked advisors recommended to customers. While effective as a baseline, this method does not capture complex relationships between agent and customer attributes.

To improve upon this, we employ a **transformer-based embedding model** that learns meaningful feature representations. The model follows an encoder architecture where input features are transformed through an initial linear layer, followed by N transformer encoder layers and an output projection. Given input vectors $\mathbf{x}_{\text{in}}$, the forward pass is defined as:

$$\mathbf{h}_0 = \text{Linear}(\mathbf{x}_{\text{in}}), \tag{6}$$

$$\mathbf{h}_l = \text{TransformerEncoderLayer}(\mathbf{h}_{l-1}), \quad l = 1, \ldots, N, \tag{7}$$

$$\mathbf{h}_{\text{out}} = \text{Linear}(\text{Squeeze}(\mathbf{h}_N)). \tag{8}$$

Separate models are trained for agents and customers, producing d_{model}-dimensional embeddings. Once embeddings are obtained, agent-client matching follows the same cosine similarity approach as the baseline but now leverages learned representations rather than raw feature vectors.

To further refine the learned representations, we introduce a **supervised fine-tuning** stage using contrastive learning. Since labeled positive agent-client

matches are unavailable, we use a proxy for positive pairs by selecting the top 30% most similar agent-client pairs from the full dataset. The reason why we chose 30% is due to limited GPU sources we will dicuss this further in the future work section. This allows the model to learn an implicit notion of "good matches" through contrastive loss:

$$\mathcal{L} = \frac{1}{2}\left(\mathcal{L}_{c\to a} + \mathcal{L}_{a\to c}\right), \tag{9}$$

where the loss terms encourage higher similarity for positive pairs and lower similarity for negatives:

$$\mathcal{L}_{c\to a} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{\exp(\text{sim}(\mathbf{c}_i, \mathbf{a}_i)/\tau)}{\sum_{j=1}^{N} \exp(\text{sim}(\mathbf{c}_i, \mathbf{a}_j)/\tau)}, \tag{10}$$

$$\mathcal{L}_{a\to c} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{\exp(\text{sim}(\mathbf{a}_i, \mathbf{c}_i)/\tau)}{\sum_{j=1}^{N} \exp(\text{sim}(\mathbf{a}_j, \mathbf{c}_i)/\tau)}. \tag{11}$$

Here, $\mathbf{c}_i$ and $\mathbf{a}_i$ are customer and agent embeddings, sim denotes cosine similarity, and τ is a temperature parameter controlling distribution sharpness. The model is optimized using the Adam optimizer and trained iteratively with mini-batches, improving its ability to discern effective agent-client matches. Figure 4 shows the training loss during different epochs of Adam, and the decreasing trend indicates that the model is learning to better distinguish between positive and negative customer-agent pairs.

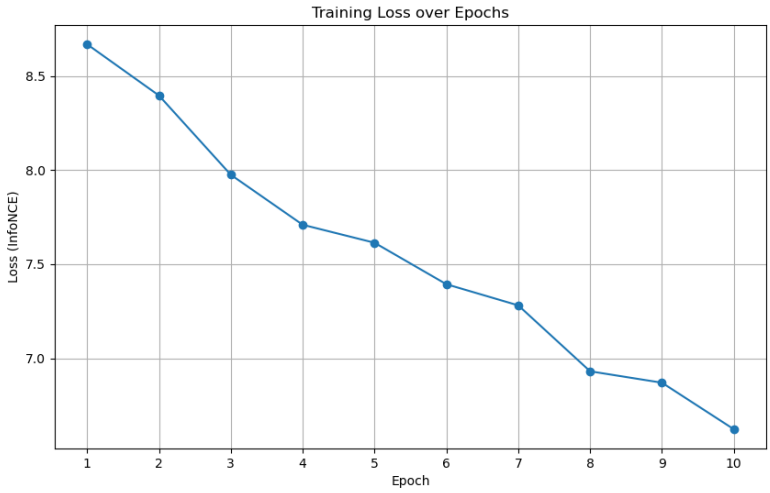

Figure 4: Training loss (InfoNCE) over epochs.

This method is further extended to handle cases where customer data is incomplete. When full policy purchase records are unavailable, feature engineering is used to construct customer representations dynamically. Instead of relying on stored embeddings, the customer vector is computed on-the-fly using pre-trained transformer encoders, allowing real-time adaptation while maintaining consistency with the trained model. The same cosine similarity ranking mechanism is then applied.

By combining unsupervised feature-based ranking, learned transformer embeddings, and contrastive learning, this approach enables a scalable, data-driven framework for financial advisor recommendations. The flexibility to handle incomplete customer information further enhances its applicability in real-world scenarios.

Due to limited time and hardware sources, in the following sections, we shall only discuss the model 1.

4 Evaluation Metrics

To evaluate the effectiveness of our model in matching customers with suitable financial advisors, we assess performance across three distinct stages: clustering, classification, and ranking. Each stage utilizes relevant evaluation metrics to ensure a comprehensive assessment of model quality.

4.1 Clustering Metrics

To assess the quality of agent clustering using the Gaussian Mixture Model (GMM), we employ the following metrics:

Silhouette Score: Measures cluster cohesion and separation. Given a dataset with n samples, the silhouette score for a single sample i is computed as:

$$S(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}, \tag{12}$$

where $a(i)$ is the mean intra-cluster distance (average distance to other points within the same cluster), and $b(i)$ is the mean nearest-cluster distance (average distance to points in the closest different cluster). The overall silhouette score is the mean of $S(i)$ over all data points.

Davies-Bouldin Index: Measures clustering compactness and separation, computed as:

$$DB = \frac{1}{K} \sum_{i=1}^{K} \max_{j \neq i} \frac{\sigma_i + \sigma_j}{d_{ij}}, \tag{13}$$

where K is the number of clusters, σ_i is the average distance between each point in cluster i and the cluster centroid, and d_{ij} is the distance between centroids of clusters i and j. Lower values indicate better clustering.

Log-Likelihood: Evaluates how well the Gaussian Mixture Model (GMM) fits the data. Given parameters Θ and data points $\mathbf{x}_1, \ldots, \mathbf{x}_n$, the log-likelihood is:

$$\log P(\mathbf{x}_1, \ldots, \mathbf{x}_n \mid \Theta) = \sum_{i=1}^{n} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_i \mid \mu_k, \Sigma_k), \tag{14}$$

where π_k, μ_k, and Σ_k are the mixture weights, means, and covariance matrices of the k-th Gaussian component, respectively.

4.2 Classification Metrics

For evaluating the Random Forest classifier that assigns clients to agent clusters, we use standard classification metrics:

Accuracy: The proportion of correctly classified samples, defined as:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}, \tag{15}$$

where TP, TN, FP, and FN represent the counts of true positives, true negatives, false positives, and false negatives, respectively.

Precision: The proportion of correctly classified positive samples among all predicted positive samples:

$$\text{Precision} = \frac{TP}{TP + FP}. \tag{16}$$

Recall: The proportion of correctly classified positive samples among all actual positive samples:

$$\text{Recall} = \frac{TP}{TP + FN}. \tag{17}$$

F1-score: The harmonic mean of precision and recall, providing a balanced assessment of classification performance:

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{18}$$

4.3 Ranking Metrics

For evaluating the XGBoost Ranker that ranks agents within clusters, we use ranking-specific metrics:

Normalized Discounted Cumulative Gain (NDCG@5): Measures ranking quality by considering relevance and order. The discounted cumulative gain (DCG) at position p is:

$$DCG_p = \sum_{i=1}^{p} \frac{rel_i}{\log_2(i+1)}, \tag{19}$$

where rel_i is the relevance score of the i-th ranked item. NDCG normalizes DCG by the ideal DCG (IDCG), ensuring values range from 0 to 1:

$$NDCG_p = \frac{DCG_p}{IDCG_p}. \tag{20}$$

Mean Reciprocal Rank (MRR): Evaluates ranking effectiveness based on the rank of the first relevant agent. Given queries Q, MRR is computed as:

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i}, \tag{21}$$

where rank_i is the rank position of the first relevant result for query i. Higher values indicate better rankings.

These metrics collectively provide a rigorous assessment of model performance, ensuring robustness in clustering, classification, and ranking tasks.

5 Results & Analysis

We use the methods in Section 4 to benchmark the performance of the model in Section 3.1. The evaluation results are presented in three stages: clustering, classification, and ranking.

5.1 Clustering Performance

The clustering evaluation metrics indicate the effectiveness of the Gaussian Mixture Model (GMM) in grouping financial advisors. The **Silhouette Score** of -0.0189 suggests poor cluster cohesion and separation, indicating that clusters may overlap significantly. The **Davies-Bouldin Index** of 9.0730, which measures the ratio of within-cluster dispersion to between-cluster separation, is relatively high, implying poorly defined clusters. However, the **Log-Likelihood** of 5.2830 suggests that the GMM model adequately fits the underlying data distribution, although the clustering structure may not be well-formed.

5.2 Classification Performance

The classification model, implemented using a Random Forest classifier, demonstrates strong predictive performance. The achieved **Accuracy** of 80.67% indicates that the model correctly assigns clients to agent clusters in a majority

of cases. The **Precision** of 81.20% suggests that when the model predicts a particular cluster, it is correct in most cases. Meanwhile, the **Recall** of 80.67% indicates that a significant proportion of clients are correctly assigned to their true clusters. The **F1-score** of 79.37% balances precision and recall, confirming robust classification performance.

5.3 Ranking Performance

The final ranking of agents within clusters, conducted using the XGBoost Ranker, yields mixed results. The **Normalized Discounted Cumulative Gain (NDCG@5)** is 1.0, indicating that the model ranks the most relevant advisors at the top positions with perfect ordering within the top five recommendations. However, the **Mean Reciprocal Rank (MRR)** is 0.0033, suggesting that the first relevant advisor appears at a very low rank on average. This discrepancy indicates that while the top-ranked advisors are highly relevant, the overall ranking distribution might not be optimal for all cases.

5.4 Discussion

The clustering stage presents challenges in defining well-separated agent groups, as indicated by the negative Silhouette Score and high Davies-Bouldin Index. This could be due to high intra-cluster variance among agents, necessitating alternative clustering methods or refined feature selection.

The classification model performs well, with high accuracy and balanced precision-recall trade-offs, demonstrating its reliability in matching clients to agent clusters. This suggests that the extracted features and clustering assignments contain meaningful patterns that the Random Forest model effectively captures.

Finally, the ranking model achieves a perfect NDCG@5 score, ensuring high-quality recommendations within the top positions. However, the low MRR score suggests that relevant advisors may not always be placed at the very top of the list. Future improvements could involve refining the ranking loss function or incorporating additional features that better capture agent success probabilities.

Overall, while the model achieves promising results in classification and ranking, further refinements in clustering may enhance overall performance and improve advisor-client matching outcomes.

6 Insights

Our proposed methodology presents a **robust, scalable, and interpretable** solution for optimizing agent-client matching in financial advisory services. By integrating multiple modeling techniques, it balances **interpretability, accuracy, and adaptability**, ultimately driving significant improvements in **business performance and operational efficiency**. Our approach ensures

transparency, making it highly valuable for **financial decision-making and regulatory compliance** at Singlife.

6.1 Insights from the XGBoost Ranker

Our ranking score for the XGBoost Ranker is defined as:

$$\text{Ranking Score} = \frac{\text{Number of Converted Policies}}{\text{Percentage of Inforce Policies} + 1e - 5} \tag{22}$$

Here, pct_inforce (percentage of currently in-force policies) is used as a proxy for an agent's workload. This ranking score prioritizes agents with high conversion success while discounting those who are already heavily engaged, ensuring that agents who efficiently convert clients without being overloaded are ranked higher.

The strong performance of our model supports this assumption, suggesting that maintaining an optimal workload balance is crucial for agent effectiveness. Agents with excessive ongoing commitments may experience reduced efficiency in converting new clients, highlighting the importance of workload management in sustaining high performance.

From a business perspective, companies can enhance profitability and efficiency by dynamically allocating workload to distribute clients more effectively among agents. Real-time monitoring of agent workload can prevent oversaturation and ensure consistently high conversion rates. Additionally, incentive structures and operational support can be adjusted to help agents sustain peak performance without excessive burden.

6.2 Ethical Considerations, Fairness, and Interpretability

Our model prioritizes **fairness and transparency**, proactively addressing biases in agent recommendations.

While gender and race are included as ranking factors, they do not drive discriminatory outcomes. Their inclusion allows the model to reflect customer preferences and behavioral trends, ensuring recommendations are tailored yet equitable. To prevent **unintended biases**, fairness-aware ranking adjustments are implemented to ensure no demographic group is systematically disadvantaged.

Unlike black-box models like neural networks, our approach is fully interpretable, allowing financial professionals and compliance teams to review, audit, and refine recommendations as needed. This transparency ensures that agent assignments align with business objectives, ethical standards, and regulatory compliance requirements. Furthermore, the ability to double-check and validate model outputs provides a crucial layer of human oversight, reinforcing trust in AI-driven decision-making and ensuring that technology serves as a reliable and responsible tool for financial services.

By combining machine learning, interpretability, fairness considerations, and business intelligence, this methodology empowers Singlife to drive smarter agent-client matching, improve operational efficiency, and unlock new opportunities for business growth.

7 Conclusion & Future Work

In this study, we developed a multi-stage approach to match clients with the most suitable financial advisors using a combination of Multiple Correspondence Analysis (MCA), Gaussian Mixture Model (GMM) clustering, Random Forest classification, and XGBoost ranking. The evaluation metrics demonstrated strong classification performance and high-quality rankings within clusters, though clustering performance remained suboptimal.

Despite the promising results, several areas require improvement. Due to limited computational resources (GPU) and time constraints, we have not fully fine-tuned our models to achieve optimal performance. Future work should involve hyperparameter tuning and further feature engineering to enhance clustering accuracy and ranking effectiveness.

Another critical area for improvement involves handling different types of client data inputs. We formulated two key assumptions: (1) having access to both the client information from the client datasheet and their historical policy purchase records, and (2) having access only to the client datasheet without purchase history. In the latter case, we need to develop transformation techniques to convert client variables into a format compatible with our model, ensuring reliable recommendations even in the absence of prior purchase data.

Future research should focus on refining these assumptions by exploring advanced techniques such as deep learning-based embeddings for categorical variables, improved clustering methodologies, and personalized ranking models. By addressing these challenges, we can further enhance the model's ability to provide accurate and meaningful financial advisor recommendations, ultimately improving client-advisor engagement and policy conversion rates.