Insurance Response Prediction Project Report

Eser Inan Arslan

May 2024

Enrian Data Science Challenge

The datasets are considered to be business-sensitive, please do not share them with anyone else.
Guidelines:
● The goal of this challenge is for us to understand your technical/statistical skills and your thinking process.
● Please use Python to complete the tasks.
● Please add comments to your code to see what and how you did.
● Be prepared to present your results and the thought process you followed. We leave the method presentation up to you.
● Feel free to reach out with any questions to us.

Problem:

We work for a major American insurer that wants to sell car insurance to their customers who currently have health insurance. They are launching an outbound marketing campaign and are curious about the success they can achieve with their existing customers based on historical marketing campaigns.

Tasks:

Based on the existing dataset, perform the following tasks.

1. Based on the historical campaign results, build a predictive model that forecasts the likely success of individuals based on their characteristics. The target variable is 'Response' in the given dataset.

2. Given textual descriptions of 10 customers, use the model you've created in the first task to predict the likelihood of marketing campaign success for each individual. Use some form of a Large Language Model to solve this task in a programmatic way that can be scaled to millions of customers if needed.

3. Please compare the distribution of predicted responses with the distribution observed in the original dataset.

Summary of Process

EDA

In the exploratory data analysis (EDA) phase of the Insurance Response Prediction project for Enrian Company, several key observations were made. Firstly, The dataset comprises 381,109 rows and 12 columns, indicating a considerable amount of data with numerous features for analysis.

Here are the some distributions:

Response Distribution:

Class 0: 334,399
Class 1: 46,710

Gender Distribution: The gender distribution in the dataset shows that there are more male entries (206,089) compared to female entries (175,020). This indicates a gender imbalance in the dataset, with a higher representation of males. It's important to consider this gender disparity when analyzing and interpreting any insights or results derived from the dataset. Additionally, exploring how gender might impact the outcomes or patterns in the data could be an interesting avenue for further analysis.

Driving License Distribution: The driving license distribution in the dataset reveals a significant imbalance, with 380,297 entries holding a driving license compared to only 812 entries without a driving license. Such a large difference between the two categories suggests that the vast majority of individuals in the dataset possess a driving license. This could potentially impact any analysis or modeling efforts, especially if driving license status is a significant factor in the target variable or predictive features. It's essential to be aware of this imbalance and its potential implications when interpreting the results or building models based on this dataset.

Previously Insured Distribution: The distribution of previously insured individuals in the dataset shows that there are 206,481 entries classified as "No" for previously insured and 174,628 entries classified as "Yes." This distribution indicates that there is a relatively balanced representation of both previously insured and uninsured individuals in the dataset. However, it's essential to consider how this distribution may impact any analysis or modeling tasks, as the prevalence of previously insured individuals compared to uninsured individuals could influence the behavior of predictive models or the interpretation of results.

Vehicle Age Distribution: The distribution of vehicle age in the dataset reveals the following:

- **1-2 Years:** There are 200,316 entries indicating vehicles aged between 1 and 2 years.
- **< 1 Year:** This category comprises 164,786 entries denoting vehicles less than 1 year old.
- **> 2 Years:** There are 16,007 entries representing vehicles exactly 2 years old.

This distribution provides insight into the age distribution of vehicles in the dataset, which can be valuable for understanding customer demographics and preferences. It's noteworthy that the majority of vehicles fall into the "1-2 Years" category, followed by vehicles less than 1 year old, with fewer vehicles exactly 2 years old.

Vehicle Damage Distribution: The distribution of vehicle damage status in the dataset is as follows:

- Yes: There are 192,413 entries indicating vehicles with damage.
- No: This category comprises 188,696 entries representing vehicles without damage.

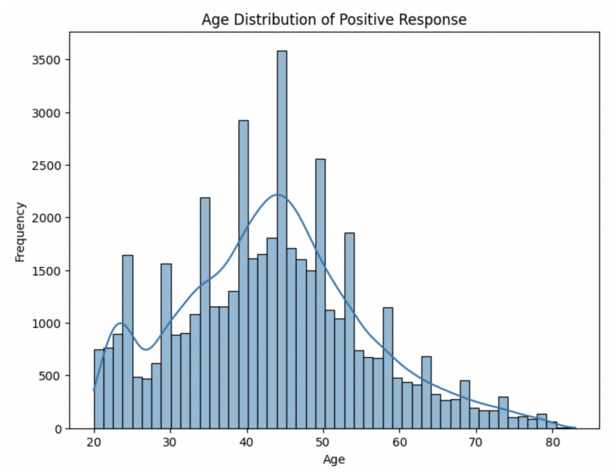
This distribution provides insight into the prevalence of vehicle damage among the dataset entries. Understanding the distribution of vehicle damage can be crucial for insurance companies in assessing risk and determining appropriate pricing and coverage strategies.

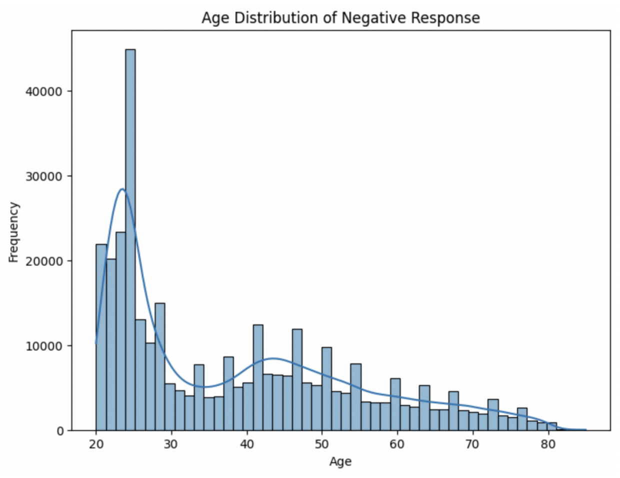
Age Distribution: The age distribution in the dataset is segmented into four bins, each representing a range of ages along with the corresponding number of records:

- **(20.0, 23.311]:** This bin includes 61,677 records.
- **(23.311, 38.823]:** There are 135,087 records in this age range.
- **(38.823, 54.334]:** This bin comprises 111,402 records.
- **(54.334, 85.0]:** The age range of 54.334 to 85.0 includes 66,711 records.

Analyzing age groups can provide valuable insights into response rates, with older age groups potentially showing higher positive response rates. Understanding these age-based trends can inform targeted marketing strategies and product offerings tailored to different age demographics.

Age Distribution by Response: Positive responses exhibit a normal distribution, with the majority falling between 40 and 50 years. Conversely, negative responses skew towards younger ages, particularly between 20 and 30 years.



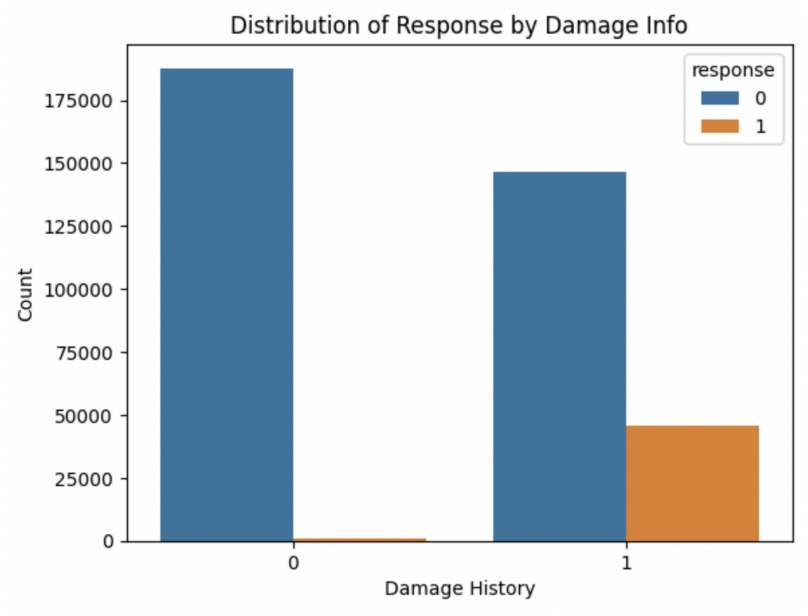


Feature Correlation with Target: The feature correlation analysis identifies several features that have notable correlations with the target variable. These correlations provide insights into which customer attributes may influence their response to marketing campaigns. Understanding these correlations helps in refining marketing strategies to better target customers who are more likely to respond positively, thereby optimizing campaign effectiveness. Features with Absolute Correlation > 0.1:

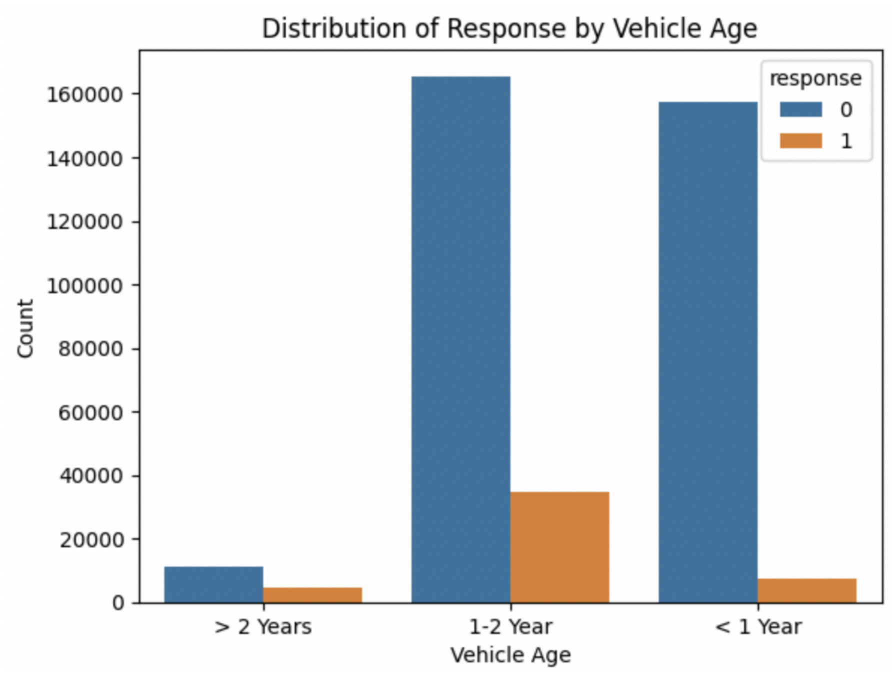
1. is_damage: 0.354
2. vehicle_age: 0.222
3. age: 0.111
4. policy_sales_channel: -0.139
5. previously_insured: -0.341

These features have notable correlations with the target variable.

Impact of Damage History: Cases with damage history tend to have higher positive response rates both numerically and proportionally.



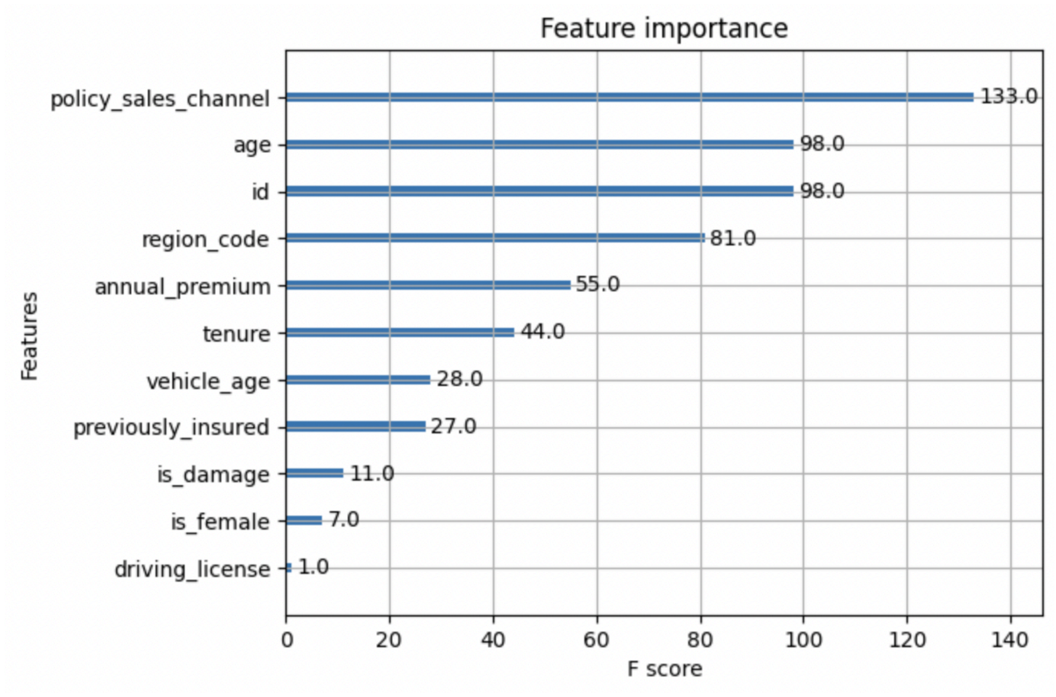
Response Distribution by Vehicle Age: Vehicles aged 2 years and below show a higher proportion of positive responses. However, it's notable that the number of vehicles over 2 years old is significantly lower.



XGB Feature Importance Analysis: The XGBoost feature importance analysis provides valuable insights into the most influential features for predicting campaign response. The analysis ranks the features based on their F scores, which indicate the relative importance of each feature in contributing to the model's predictive performance.

From the results, it's evident that certain features play a more significant role in predicting campaign response than others. Features like 'policy_sales_channel', 'age', 'id', and 'region_code' have relatively high importance scores, suggesting that they strongly influence the model's predictions. On the other hand, features like 'is_female' and 'driving_license' have lower importance scores, indicating they have less impact on the model's predictive performance.

This information can guide marketing strategies by highlighting the key customer attributes that drive campaign success. By focusing on these influential features, marketers can tailor their campaigns more effectively to target customers who are most likely to respond positively, thus maximizing campaign ROI.



Missing Values: The dataset contains no missing values, hence no imputation was necessary.

Overall, the EDA phase provided valuable insights into the dataset's characteristics, distribution, and potential predictors, laying a solid foundation for subsequent analysis and modeling tasks.

PREPROCESS

The preprocessing phase involved several sequential steps to prepare the data for analysis and model training. Firstly, all feature names were standardized by converting them to lowercase, ensuring consistency across the dataset. This step facilitates easier data manipulation and avoids inconsistencies due to variations in capitalization.

Next, categorical features such as 'gender' and 'vehicle_damage' underwent transformation to boolean types for better compatibility with machine learning algorithms. For example, the 'gender' information was converted to a binary indicator 'is_female', simplifying gender representation to 0s and 1s. Similarly, the 'vehicle_damage' feature was transformed into 'is_damage', indicating whether the vehicle had incurred damage or not.

The 'vehicle_age' feature was categorized into three groups based on the age of the vehicles, providing insights into vehicle age-related patterns. These categories were then mapped to numerical values to facilitate numerical computations and analysis.

Another important preprocessing step involved handling missing values and ensuring data consistency. Numerical features such as 'annual_premium', 'vehicle_age', and 'tenure' were standardized and scaled using MinMaxScaler to ensure uniformity in their ranges and prevent features with larger scales from dominating the model training process.

Logical groupings were applied to features like 'previously_insured' to simplify their representation, with the data being classified into binary categories based on logical groupings. This simplification aids in reducing the complexity of the dataset and enhances model interpretability.

In the final stage of preprocessing, the datasets were partitioned into training and testing sets, and necessary adjustments were made to ensure data consistency across different stages of model training and evaluation. Additionally, the 'id' column was removed from the datasets to eliminate redundant information.

Overall, these preprocessing steps are crucial for preparing the data for further analysis and model training, ensuring that the datasets are clean, standardized, and well-structured, which ultimately enhances the effectiveness and interpretability of the machine learning models applied to them.

Final sizes of dataframes :

X_train.shape (304887, 10),
y_train.shape (304887,),
X_test.shape (76222, 10),
y_test.shape (76222,),
z.shape(10,10)

Model Selection

In this analysis, we employed various machine learning algorithms to develop predictive models for the insurance response prediction system. The algorithms utilized include XGBoost Regressor, Random Forest, and Naive Bayes. Additionally, variations of Naive Bayes, such as the isotonic and sigmoid calibrated versions, were also incorporated into the modeling process to explore different probabilistic approaches.

While Recurrent Neural Network (RNN) implementation was considered, its execution time and memory requirements were found to be prohibitively high due to the size of the dataset. Consequently, to avoid over-engineering and ensure practicality, the RNN approach was not included in the final solution.

By leveraging these diverse algorithms, we aimed to explore different modeling techniques and assess their performance in predicting insurance response probabilities. Each algorithm brings its unique strengths and characteristics to the modeling process, contributing to a comprehensive evaluation of predictive capabilities for the given task.

1. XGBoost Regressor

The XGBoost Regressor was chosen due to its versatility and robust performance in various regression tasks. It's particularly well-suited for handling large datasets with high dimensionality.

2. **Naive Bayes**

Naive Bayes is a probabilistic classifier based on Bayes' theorem with the "naive" assumption of independence between features. It calculates the probability of each class given a set of input features and selects the class with the highest probability as the prediction. Despite its simplicity and computational efficiency, Naive Bayes may oversimplify complex relationships in the data, leading to suboptimal performance in certain scenarios.

3. **Calibrated Naive Bayes with Sigmoid Function**

Calibrated Naive Bayes with Sigmoid is an extension of the Naive Bayes algorithm that incorporates probabilistic calibration using the sigmoid function. This calibration technique adjusts the raw output probabilities of the Naive Bayes classifier to better reflect the true likelihood of class membership. By mapping the raw scores to calibrated probabilities, this approach aims to improve the reliability of the model's probability estimates, particularly for imbalanced datasets or when probabilistic outputs are required.

4. **Calibrated Naive Bayes with Isotonic Function**

Similar to the sigmoid calibration method, Calibrated Naive Bayes with Isotonic Regression enhances the probabilistic calibration of Naive Bayes by employing isotonic regression. Isotonic regression is a non-parametric technique that fits a monotonically increasing function to the raw output scores, thereby transforming them into calibrated probabilities. This method is particularly useful for addressing issues of overconfidence or underconfidence in the raw predictions of Naive Bayes, leading to more accurate and well-calibrated probability estimates.

5. **Random Forest Regressor**

Random Forest Regressor is an ensemble learning algorithm that constructs multiple decision trees during training and outputs the average prediction of the individual trees. Each tree in the forest is trained on a random subset of the training data and a random subset of features, promoting model diversity and reducing overfitting. Random Forest is known for its robustness, scalability, and ability to capture complex nonlinear relationships in the data. However, it may suffer from computational inefficiency and lack of interpretability, especially with large datasets.

As a result for algorithm perspectives, each algorithm has its strengths and weaknesses, and the choice of algorithm depends on various factors such as the nature of the data, the complexity of the problem, and computational resources. By leveraging a combination of these algorithms and calibration techniques, we aim to develop a predictive model that optimizes insurance response and enhances return rate effectively.

To optimize model performance and fine-tune algorithm parameters, I implemented the GridSearch pipeline for hyperparameter tuning across selected algorithms. This systematic approach allowed for comprehensive exploration of hyperparameter space, aiding in identifying the most optimal configuration for each algorithm. Despite the computational demands, GridSearch enabled efficient parameter tuning, enhancing the models' predictive capabilities and generalization performance.

Furthermore, I observed varying execution times across different algorithms. While Random Forest, known for its complexity and computational intensity, typically required 100-120 minutes to complete, XGBRegressor takes 10-15 minutes, the Naive Bayes solutions exhibited significantly shorter execution times, completing within minutes. This discrepancy in execution times underscores the trade-off between model complexity and computational efficiency, highlighting the importance of selecting appropriate algorithms based on computational resources and modeling objectives. By leveraging GridSearch for hyperparameter tuning and considering algorithm-specific computational characteristics, I aimed to strike a balance between model performance and computational efficiency, ultimately enhancing the overall effectiveness of the predictive modeling process.

Parameter Tuning

To optimize the performance of each model, I utilized the grid search method for hyperparameter tuning. Grid search allows for an exhaustive search over a predefined hyperparameter space, enabling the identification of the optimal combination of parameters.

Error Measurement

For evaluating the performance of the models in the classification setting, a set of classification error metrics was employed. These metrics provide a comprehensive assessment of the models' performance in predicting insurance responses accurately. The following classification error metrics were utilized:

1. **Precision**: Precision measures the ratio of correctly predicted positive observations to the total predicted positive observations. It provides insight into the model's ability to avoid false positives.
2. **Recall**: Recall, also known as sensitivity, measures the ratio of correctly predicted positive observations to the total actual positive observations. It indicates the model's ability to capture all positive instances.

3. **F1 Score**: The F1 Score is the harmonic mean of precision and recall. It provides a balanced measure of the model's accuracy, considering both false positives and false negatives.
4. **Accuracy**: Accuracy measures the ratio of correctly predicted observations to the total observations. It provides an overall assessment of the model's performance across all classes.

By considering these classification error metrics, we gain a comprehensive understanding of the models' predictive capabilities in the insurance response prediction system, now treated as a classification problem. This approach ensures a thorough evaluation of the models' performance in accurately classifying insurance responses.

Model Conclusion

In addressing the classification problem of insurance response prediction, we explored various machine learning algorithms, including XGBoost Regressor, Random Forest, and different versions of Naive Bias (uncalibrated, isotonic, and sigmoid calibrated). Through rigorous evaluation using precision, recall, F1-score, and accuracy metrics, we gained valuable insights into the performance of each model.

The XGBoost Regressor exhibited an accuracy of 0.87, with a macro-average F1-score of 0.31. Despite its high accuracy in predicting class 0 responses (non-responses), it struggled with class 1 predictions (positive responses), as indicated by its low recall and F1-score for class 1.

Similarly, the Random Forest model achieved an accuracy of 0.88, with a macro-average F1-score of 0.47. While it performed well in predicting class 0 responses, it showed limitations in capturing class 1 responses, resulting in a lower F1-score and recall for this class.

The uncalibrated Naive Bias model demonstrated an accuracy of 0.68, with a macro-average F1-score of 0.60. While it exhibited relatively balanced precision and recall for both classes, it struggled with predicting class 1 responses, as evidenced by its lower precision and F1-score for this class.

The isotonic Naive Bias model achieved an accuracy of 0.87, with a macro-average F1-score of 0.47. Despite its high accuracy in predicting class 0 responses, it showed limitations in predicting class 1 responses, similar to the XGBoost Regressor and Random Forest models.

Lastly, the sigmoid Naive Bias model yielded an accuracy of 0.88, with a macro-average F1-score of 0.47. While it performed well in predicting class 0 responses, it struggled with class 1 predictions, resulting in a lower F1-score and recall for this class.

In conclusion, while all models exhibited strengths in predicting class 0 responses, they faced challenges in accurately predicting class 1 responses. Further optimization and refinement of these models may be necessary to improve their performance in capturing positive insurance responses.

Insurance Response Prediction

Firstly, I delved into the issue of response prediction, focusing on recent customer behavior within %80 of total data. Utilizing machine learning algorithms such as XGB, Random Forest, and Naive Bayes, I computed response probabilities and weighted them based on model accuracy to derive a comprehensive churn score.

$$\text{Probability} = \sum_{Model\ 1}^{Model\ n} (Model\ Score * Model\ Accuracy)$$

For customers, the calculated response scores were classified based on different thresholds. The determined thresholds were 0.5, 0.7, and 0.9.

For the threshold of 0.5:

- Actual class 0 cases: 66,699
  - True class 0 cases: 45,946
  - False class 0 cases: 20,753
- Actual class 1 cases: 9,523
  - True class 1 cases: 8,496
  - False class 1 cases: 1,027
- For the threshold of 0.7:
- Actual class 0 cases: 66,699
  - True class 0 cases: 47,142
  - False class 0 cases: 19,557
- Actual class 1 cases: 9,523
  - True class 1 cases: 8,218
  - False class 1 cases: 1,305

- For the threshold of 0.9:
- Actual class 0 cases: 66,699
    - True class 0 cases: 66,395
    - False class 0 cases: 304
- Actual class 1 cases: 9,523
    - True class 1 cases: 255
    - False class 1 cases: 9,268

These results provide insights into how different threshold values affect the classification of customer response scores, enabling further analysis and decision-making based on the classification outcomes.

For Customer Description Dataset

This solution employs a function called extract_with_chat_model to extract structured features from customer descriptions using the GPT-4 chat model provided by OpenAI. Here's a breakdown of the process:

- **Function Definition:** The extract_with_chat_model function takes a customer description as input and constructs a prompt that outlines the features to be extracted.

- **Prompt Construction:** The function constructs a prompt string that includes the description provided and lists the features to be extracted, such as gender, age, driving license, etc.

- **OpenAI API Query:** The function queries the OpenAI API using the v1/chat/completions endpoint to generate a response based on the prompt and the GPT-4 model. It provides context to the model by simulating a conversation where the user (the function) asks the system (GPT-4) to extract the specified features from the description.

- **Response Parsing:** Once the response is received from the API, the function parses the response text to extract the structured features. It iterates through each line of the response and extracts key-value pairs delimited by colons (':'). The extracted features are then stored in a dictionary, where keys are the lowercase versions of the feature names and values are the extracted values.

- **Dataframe Creation:** The function applies the extract_with_chat_model function to each customer description in a pandas DataFrame (df_desc) using the apply method. It creates a new DataFrame (extracted_df) to store the extracted features along with the corresponding customer IDs and names.

- **Insertion of ID and Name:** The function inserts the customer ID and name columns at the beginning of the extracted_df DataFrame to maintain the association between the extracted features and the respective customers.