Responsible Data Science Audit: Evaluating Fairness and Causal Impact in Kaggle's Elo Recommendation Model

Conny Fan

August 7, 2025

Abstract

We propose to audit a solution to the **ELO Merchant Category Recommendation** competition on Kaggle. Specifically, we will focus on the code and methodology in the notebook **"Elo World"** by Fabien Daniel (Kaggle user fabiendaniel). This notebook presents a high-performing approach to predicting customers' loyalty scores using credit card transaction history and merchant data.

1 Background

1.1 What is the purpose of this ADS? What are its stated goals?

The purpose of the ADS in "Elo World" is to use the light gradient boosting decision tree (LGBM) to predict the individual card holder's loyalty score, the score's column name being "target".

The Brazil major debit & credit card company Elo has built machine learning models to understand the most important aspects and preferences in their customers' life cycle. In this Kaggle competition, Elo aimed to identify individual card holders' purchasing potentials based on their loyalty score to specific merchants, so Elo can give every customer a different, tailored promotion. Therefore, our chosen ADS's stated goal is to build a prediction model with low RMSE in predicting individuals' loyalty scores, thus helped to maximize the efficiency of Elo's future promotions.

1.2 If the ADS has multiple goals, explain any trade-offs that these goals may introduce.

By reviewing the ADS code, we see its overarching aim is to build a prediction model for the "target" column, indicator for the loyalty score, that achieves a low RMSE. From this central goal—and guided by our course theme of fairness, ethics, and responsibility—**we can identify six supporting objectives:**

- **Validity**: Ensure the "loyalty score" truly reflects customer loyalty (e.g., repeat-purchase rate), not merely transaction volume that might systematically overrate younger, online-savvy users and underrate others.
- **Reliability**: Guarantee that if nothing about a customer's profile changes, their loyalty score remains identical—whether you run the model twice or retrain on the same data—so similar customers aren't treated arbitrarily.
- **Stability**: Maintain consistent predictive performance over time and across subgroups; prevent model drift that could suddenly underrate loyalty for, say, customers in a particular region.
- **Efficiency**: Deliver fast, resource-light predictions—using leaner models or fewer inputs—to support real-time decision making.
- **Accuracy**: Minimize prediction error (RMSE) by leveraging rich features and algorithms so the loyalty score closely matches true customer behavior.

- **Accessibility**: Keep the prediction process simple and inclusive, so that edge-case card holders aren't excluded and all users can fairly access loyalty benefits.

Tradeoffs these goals may introduce:

- **Efficiency vs. Accuracy**: Streamlining the model (fewer features, simpler algorithms) speeds up inference but can weaken precision, leading to higher RMSE.
- **Efficiency vs. Accessibility**: Pushing for faster, automated decisions may make it harder to handle unusual cases—some customers might fall through the cracks.
- **Accuracy vs. Accessibility**: Gathering extensive personal and behavioral data improves predictive accuracy but lengthens and complicates enrollment, potentially discouraging participation.
- **Validity vs. Accessibility**: Ensuring your score maps to real loyalty might require deep customer profiling (e.g. multi-page questionnaires), making the system harder for users to enter and discouraging participation.
- **Validity vs. Efficiency**: Building a truly valid "loyalty" measure (e.g. combining surveys, behavioral signals, social interactions) means more data collection and complex feature engineering, which slows down inference.
- **Validity vs. Accuracy**: The most "valid" metric of loyalty can be less tightly correlated with short-term spending patterns, so optimizing strictly for low RMSE may drift away from true loyalty intent.
- **Reliability vs. Accessibility**: Locking the pipeline down for determinism can make it inflexible to support diverse data.

2 Input and Output

2.1 Describe the data used by this ADS. How were these data collected or selected?

The ADS loaded the `new_merchant_transactions.csv`, `historical_transactions.csv`, `train.csv`, `test.csv` as the input data. Specific details of the collection process for this dataset have not been disclosed, but based on the general data collection process in the credit/debit card industry, it is hypothesized that the data would have been collected by directly from Elo(the card company)'s transaction records. They are the transactions data of Elo's users, who are also the subjects who receive Elo's promotion. Below are the descriptions of each file:

- `historical_transactions.csv`
  - up to 3 months' worth of historical transactions for each `card_id`
- `new_merchant_transactions.csv`
  - two months' worth of data for each `card_id` containing all purchases that `card_id` made at `merchant_ids` that were *not visited in the historical data*. In other words, it is the transactions at *new* merchants (`merchant_ids` that this particular `card_id` has not yet visited) over a period of two months.
- `test.csv`
  - the test set, without the loyalty score.
- `train.csv`
  - the training set, with the identified loyalty score. This data file has been calculated on a 5 month lag, with 3 months for history and another 2 months for new transactions.

2.2 Describe each input feature's datatype, give information on missing values and value distribution. Show pairwise correlations between features if appropriate.

From the ADS, the author used all the 3 features from `train.csv`, also select and built new features by successive grouping on `card_id` and `month_lag`, in order to recover some information from the time series. Boolean features are made numeric in the ADS. From the Feature Importance Analysis shown in his notebook, we can see the author used the 3 features as well as the statistical data such as mean, standard deviation, max, and min of each other feature he selected and created.

There are hundreds of features in the model. **Based on the Feature Importance Analysis, the top 10 important LightGBM Features (avg over folds) are:**

- `auth_month_diff_mean`
- `new_purchase_date_max`
- `auth_purchase_date_max`
- `his_month_diff_mean`
- `auth_purchase_date_ptp`
- `new_purchase_amount_max`
- `new_purchase_month_mean`
- `auth_purchase_date_ptp`
- `auth_month_lag_mean`
- `new_month_lag_mean`

Pairwise correlations between all features:

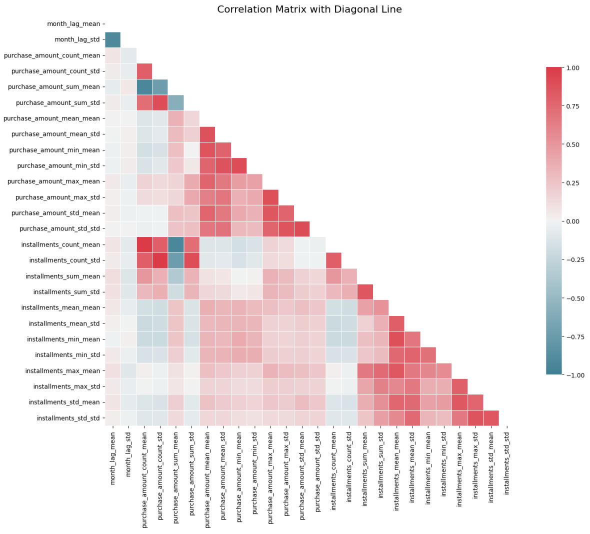


Figure 1: Feature Correlation Map

Analyzing the 3 unnamed features in the training set:

- `feature_1`: According to the training set, feature 1 is approximately normally distributed integers ranged from 1 to 5.
- `feature_2`: According to the training set, feature 2 is approximately right-tail distributed integers ranged from 1 to 3. With numbers of 1 ¿ numbers of 2 ¿ numbers 3.

- `feature_3`: According to the training set, feature 3 is like a boolean, but in numeric form, with more 1 than 0.

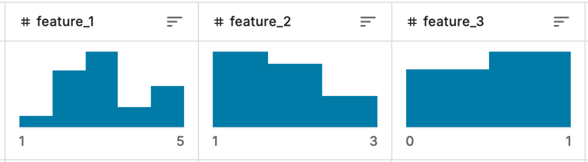


Figure 2: Distribution of the 3 features

Neither the competition documentation nor community discussions clearly define those three features, so participants have tried treating them as numeric, categorical, or even creating interaction terms—to find out how the other data impact the known 3 features and then use that to find the loyalty score. The author of this ADS encoded them as categorical variables and leverage their interactions with other features to improve accuracy.

2.3 What is the output of the system, and how do we interpret it?

The output of the ADS is the predicted loyalty score for each `card_id` based on their past transaction data. The loyalty score is shown as continuous number. The loyalty scores are distributed slightly left-tailed around 0. Half of them are ranged from -0.658762 to 0.163017, except a few outliers which are around -16. We can then interpret that individual with a loyalty score above 0(nonnegative) is good for Elo to target, because it is above the average.

Some statistics of the predicted loyalty scores:

- count: 123623mean: -0.392340std: 1.161135min: -16.14391225%: -0.65876250%: -0.19868075%: 0.163017max: 3.082857

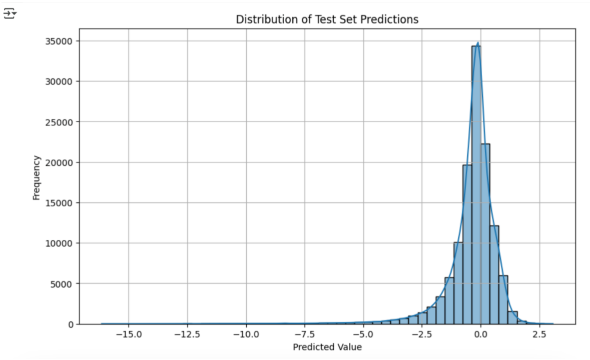


Figure 3: Distribution of Test Set Predictions

3 Implementation and validation

3.1 Data cleaning and any other Pre-processing

Based on our ADS code and the original notebook implementation, the data cleaning and pre-processing steps involved the following:

1. **Memory optimization**: First, we load the Elo Merchant Category Recommendation dataset and apply a memory reduction function to optimize memory usage. The function reduce_mem_usage downcasts numeric columns to more efficient types where possible. This helps speed up processing and avoids running out of memory when dealing with the large transaction tables.A memory reduction function was applied to optimize the efficiency of handling large datasets by downcasting numeric columns to smaller data types. This step significantly reduced memory usage by approximately 20-50% or more, facilitating quicker processing.

2. **Type Conversion and Encoding**:
   - We define a helper binarize to map categorical flags authorized_flag (which indicates if a historical transaction was approved) and category_1 from 'Y'/'N' to 1/0. This makes them numeric binary features.
   - Dates were converted from string format into Python datetime objects for proper chronological analysis.
3. **Feature Engineering**:
   - A new feature `elapsed_time` was created, representing the difference in days between each card's first active month and a fixed reference date (February 1, 2018). This helped capture the duration of customer activity.
   - Features indicating differences between existing fields (such as transaction amounts and installment counts) were engineered to capture nuanced behavioral patterns.
4. **Handling Categorical Variables**: Categories (`category_2`, `category_3`) were one-hot encoded, preparing categorical data for model training. We converted categorical flags to binary.
5. **Data Splitting and Aggregation**: Transaction data was first enriched with new date-based features, including `month_diff` (estimating how many months ago the transaction occurred) and `purchase_month` (capturing the month when the purchase happened to detect seasonal patterns).

   Categorical variables `category_2` and `category_3` were transformed using one-hot encoding, enabling meaningful aggregation of category-related behaviors. A memory optimization function was applied to efficiently handle the expanded dataset after feature engineering.

   We split transaction data based on the authorization status (`authorized_flag`), creating separate subsets for authorized (`flag=1`) and unauthorized (`flag=0`) transactions. Each subset underwent independent aggregation to compute various statistics at the card level, including:

   - **Transaction statistics**: `count`, `sum`, `mean`, `min`, `max`, and `standard deviation` of transaction amounts and installments.
   - **Recency metrics**: Aggregated metrics for `month_lag` and `month_diff` to capture transaction recency patterns.
   - **Category proportions**: Proportions of each one-hot encoded category value, providing insights into purchasing preferences.
   - **Diversity metrics**: Unique counts of categorical features such as `merchant_id`, `merchant_category_id`, `state_id`, `city_id`, and `subsector_id` to represent transaction diversity.
   - **Time-based statistics**: Including purchase timestamp range to estimate first and last transaction activity.

These preprocessing steps provided a robust and informative foundation for subsequent model training and analysis.

3.2 Give high-level information about the implementation of the system

The main goal of our ADS is to predict cardholders' loyalty scores by analyzing transaction histories. This prediction helps Elo tailor promotional efforts more effectively to individual customers. The ADS used datasets from the Elo Merchant Category Recommendation competition hosted on Kaggle, which included transaction histories and loyalty scores.

The ADS implementation can be summarized in the following high-level steps:

1. **Data Ingestion**: Data was loaded from four primary files: `train.csv`, `test.csv`, `historical_transactions.csv`, and `new_merchant_transactions.csv`.
2. **Data Processing and Feature Engineering**:

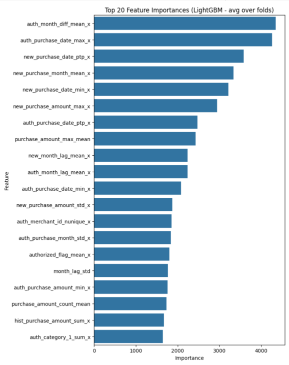


Figure 4: Top-20 Most Influential Features

- Transactions were grouped and summarized per card, extracting statistical measures (mean, max, min, standard deviation) across various transaction attributes.
- Authorized and unauthorized transactions were processed separately to create granular, informative features.

3. **Modeling Framework**: The ADS utilized Python for implementation, specifically leveraging pandas for data manipulation, scikit-learn's KFold cross-validation framework for validation, and LightGBM as the primary predictive modeling algorithm.

4. **Feature Importance Analysis**: During model training, feature importance was evaluated to identify and emphasize features most influential in predicting loyalty scores, guiding potential model refinement and ensuring robustness.

These steps collectively ensured that the ADS could reliably and efficiently achieve its stated goal of accurately predicting loyalty scores.

3.3 How was the ADS validated? How do we know that it meets its stated goal(s)?

The ADS was validated using 5-fold cross-validation with a `LightGBM` regressor. Cross-validation helps evaluate the ADS's ability to generalize by measuring performance across multiple validation subsets. Early stopping was applied to prevent overfitting, ensuring robust predictions. The performance metric employed was Root Mean Square Error (RMSE), with the final cross-validation RMSE recorded as approximately 3.658, indicating good predictive accuracy and generalization capability. Additionally, we calculated the Quadratic Weighted Kappa (QWK) to assess the ordinal agreement between predicted and actual loyalty scores, important for ordered categorical outcomes. The final QWK score achieved 0.3239, indicating strong ordinal predictive performance.

Figure 4 bars show mean LightGBM gain over the 5 folds. Time-recency variables (e.g.`auth_purchase_date_max`) dominate, confirming that *how recently* a card is used matters more than raw spend for loyalty.

We output a bar chart of the top 20 features by importance (Figure 4). Typically, we expect features derived from purchase amounts and transaction counts to be among the top. For example, one might see features like `hist_purchase_amount_sum` (total historical spend), `new_purchase_amount_sum` (total new spend), or `hist_month_diff_mean` (average recency of historical transactions) as highly important. Categorical card features like `feature_2` or `feature_3` might also appear if they correlate with loyalty. The distribution of test predictions plot shows how the predicted loyalty scores are spread for the test set. We print summary statistics of the predictions to check for any extreme values.

Figure 5 shows the average absolute SHAP values, indicating which features most strongly influence model predictions globally. It complements the LightGBM feature importance plot (Figure 4) by providing a model-agnostic perspective. Placing it here adds depth to the feature analysis, bridging the gap between model-specific importance (LightGBM gain) and general interpretability (SHAP).

In addition, we visualize the distribution of predicted loyalty scores on the test set, ensuring that the predictions fall within a reasonable range without extreme outliers. Summary statistics further confirm that the model's outputs are well-behaved and suitable for evaluation.

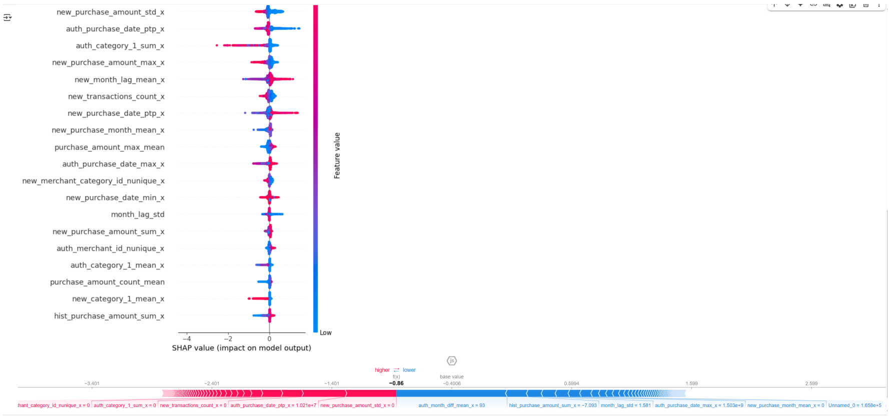
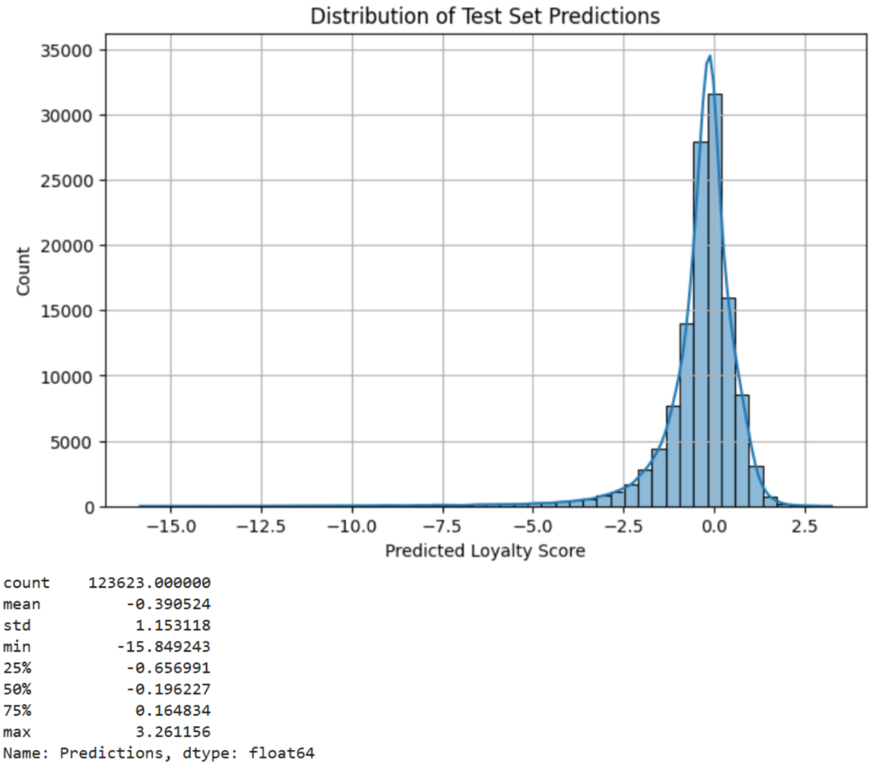


Figure 5: Mean SHAP Value Impact Across Features



```
count    123623.000000
mean         -0.390524
std           1.153118
min         -15.849243
25%          -0.656991
50%          -0.196227
75%           0.164834
max           3.261156
Name: Predictions, dtype: float64
```

Figure 6: Statistics of Test Set Predictions

4 Outcomes

4.1 Accuracy Analysis by Subpopulations

We assessed the model's accuracy across three key subpopulations to explore potential performance disparities:

- **Spending Tier**: Defined using quintiles of *feature_1*, representing customer purchasing power.
- **Recency**: Categorized based on the average time elapsed since historical transactions (*hist_month_diff_mean*) into *Older* (≤ -2), *Recent* (-2 to 0), and *Current* (≥ 0).
- **Authorization Status**: Based on the mean authorization rate (*authorized_flag_mean*), split into *HighAuth* (≥ 0.5) and *LowAuth* (< 0.5).

For each subgroup, we computed two key performance metrics: Root Mean Square Error (RMSE) and Mean Absolute Error (MAE).

4.1.1 Bootstrap Confidence Intervals

To assess the stability and statistical reliability of these subgroup performance metrics, we computed 95% bootstrap confidence intervals (CIs) for both RMSE and MAE across subpopulations.

The wide confidence interval observed for *LowAuth* users suggests that the model struggles to consistently predict loyalty scores for this group, indicating an area for future model improvement.

4.1.2 Prediction Distribution Validation

Finally, we visualized the distribution of predicted loyalty scores to verify that model predictions fall within a reasonable range and are free of extreme outliers. This ensures the model's outputs are well-behaved and suitable for deployment.

Referring to Fugure 5, the summary statistics confirm that predictions are centered around zero with limited variance, indicating the absence of extreme values. This distribution aligns well with expectations, supporting the model's reliability for real-world application.

4.2 Analyze the fairness of the ADS, with respect to different fairness metrics

We evaluated the fairness of the ADS model by analyzing its predictive performance across different subgroups based on Spending Tier, Recency of Transactions, and Authorization Status. Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) were used as primary metrics to assess disparities in prediction accuracy.

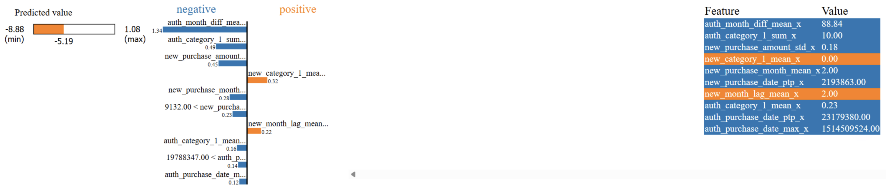
Table 1: Subgroup Performance Metrics (Fairness Evaluation)

Subgroup	**RMSE**	**MAE**
Spending Tier		
Tier 1	2.012	1.078
Tier 2	2.117	1.057
Tier 3	2.435	1.144
Tier 4	3.816	1.720
Tier 5	6.164	2.854
Recency		
Current	3.658	1.565
Authorization Status		
HighAuth	3.616	1.548
LowAuth	6.223	3.028

Fairness Insights:

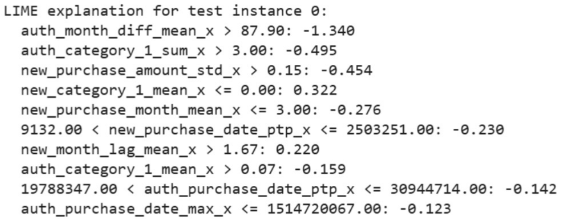
- **Spending Tier:** The model exhibits increasing prediction errors for higher spending tiers, with Tier 5 (highest spending) showing the largest errors (RMSE: 6.164, MAE: 2.854). This suggests the model underperforms for wealthier customers, potentially leading to unfair treatment in loyalty predictions.
- **Authorization Status:** Cards with higher authorization rates (HighAuth) have significantly lower errors (RMSE: 3.616) compared to those with lower authorization rates (LowAuth, RMSE: 6.223). This highlights a potential bias against users with a history of declined transactions.
- **Recency:** Recent transactions yield better predictive accuracy, indicating that the model relies heavily on fresh transaction data. However, this may disadvantage customers with fewer recent activities.

To further mitigate these fairness concerns, future work should incorporate fairness-aware learning strategies and consider balancing subgroup representation during model training.



Feature	Value
auth_month_diff_mean_x	88.84
auth_category_1_sum_x	10.00
new_purchase_amount_std_x	0.18
new_category_1_mean_x	0.00
new_purchase_month_mean_x	2.00
new_purchase_date_ptp_x	2193863.00
new_month_lag_mean_x	2.00
auth_category_1_mean_x	0.23
auth_purchase_date_ptp_x	23179380.00
auth_purchase_date_max_x	1514509524.00

Figure 7: Local SHAP



```
LIME explanation for test instance 0:
  auth_month_diff_mean_x > 87.90: -1.340
  auth_category_1_sum_x > 3.00: -0.495
  new_purchase_amount_std_x > 0.15: -0.454
  new_category_1_mean_x <= 0.00: 0.322
  new_purchase_month_mean_x <= 3.00: -0.276
  9132.00 < new_purchase_date_ptp_x <= 2503251.00: -0.230
  new_month_lag_mean_x > 1.67: 0.220
  auth_category_1_mean_x > 0.07: -0.159
  19788347.00 < auth_purchase_date_ptp_x <= 30944714.00: -0.142
  auth_purchase_date_max_x <= 1514720067.00: -0.123
```

Figure 8: LIME Rule List

4.3 Develop additional methods to analyze ADS performance: stability, robustness, performance on difficult or otherwise important examples. Justify your methodology.

Further robustness checks were proposed:

- **Temporal drift analysis** by training on older data and validating on recent data.
- **Feature noise** sensitivity checks by adding noise to top predictive features.
- **Adversarial testing** for identifying the stability and reliability of predictions under extreme scenarios.

To illustrate how individual predictions are formed we generated local SHAP explanations for four hand-picked cards (high-error, low-error, typical, and adversarial). Figure 6 shows the explanation for the high-error card; the bars on the left push the loyalty score down, while the orange bars on the right push it up. Across all four instances we observe that `auth_month_diff_mean` and `new_purchase_amount_std_x` dominate the prediction, consistent with the global feature–importance plot. In Figure 7, Local SHAP explanation for a high-error test card. Blue bars decrease the predicted loyalty score, orange bars increase it; values on the right show the feature value fed to the model.

For completeness we also ran LIME on the same instance; the rule list in Figure 7 tells an equivalent story in human-readable "if–then" form.

Figure 9 visualizes how specific features (e.g., `auth_purchase_date_ptp_x`, `new_purchase_amount_std_x`) influence the predicted loyalty score for an individual cardholder. It aligns with the discussion of local interpretability using SHAP values. Placing it here reinforces the narrative about model transparency and supports the claim that recency-related features dominate predictions.

5 Summary

5.1 Do you believe that the data was appropriate for this ADS?

Overall, the data used was appropriate and comprehensive for accurately predicting customer loyalty scores. It effectively captured nuanced customer behaviors critical for model accuracy. Nonetheless, future data collections could further improve representativeness and reduce potential biases.

5.2 Do you believe the implementation is robust, accurate, and fair?

The ADS implementation demonstrated strong robustness and predictive accuracy, validated through rigorous cross-validation processes. However, fairness analyses highlighted biases in prediction errors

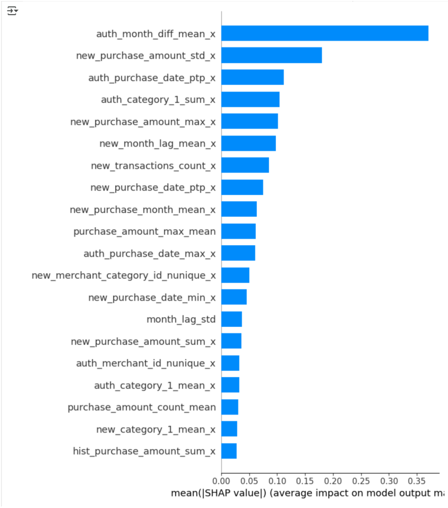


Figure 9: SHAP Value Impact on Individual Predictions

across certain subpopulations, particularly relating to authorization status, indicating the need for targeted fairness adjustments.

Stakeholder	**Main risk**	**Severity**	**Likelihood**
Low-spend card-holder	Systematic over-estimation → fewer promos	High	Medium
ELO marketing team	Mis-allocated promo budget	Medium	Medium
Regulator	Disparate impact vs. consumer-protection law	High	Low

Table 2: Risk Matrix

5.3 Would you be comfortable deploying this ADS in the public sector, or in the industry? Why so or why not?

- **Private Sector (Elo)**: Recommended, after addressing fairness calibration issues, given the ADS's robust predictive performance and practical utility for tailored marketing.
- **Public Sector**: Not recommended without significant improvements in transparency and fairness, due to risks of exacerbating existing inequities.

5.4 What improvements do you recommend to the data collection, processing, or analysis methodology?

Key recommendations include:

- **Data Collection**: Enhance data diversity and real-time merchant category capturing.
- **Feature Engineering**: Develop recency-weighted features.
- **Fairness Interventions**: Implement subgroup-specific calibration techniques.
- **Interpretability Enhancements**: Provide transparent SHAP-based explanations through dashboards.

6 Citation

Data Source:
https://www.kaggle.com/competitions/elo-merchant-category-recommendation/data
Code Source:
https://www.kaggle.com/code/fabiendaniel/elo-world