# Predicting Customer Reorder Behavior in Supply Chains

Shubham Banavalikar, Mikel Calderon, Ben Robbins and Alex TenPas

***Abstract*—Customer retention is substantially more cost-effective than customer acquisition, making it a critical objective for retail businesses. To support retention efforts, we developed a predictive model that determines whether a customer will make a repeat purchase based solely on their first order. Using the Kaggle dataset DataCo SMART SUPPLY CHAIN FOR BIG DATA ANALYSIS, which contains 180,519 product-level transactions from an international shipping company, we engineered features and filtered the data to 6,503 first-purchase records. These were split into training (3,901), validation (1,301), and test (1,301) sets. We evaluated a Random Forest, a Gradient Boosted Tree, and a feed-forward Neural Network against a majority-class baseline. Our best-performing model, a stacked ensemble of a neural network and gradient boosting, achieved an accuracy of 85% and an AUC of 0.85, outperforming the 64% accuracy baseline. While further validation is needed to assess generalizability across domains, these results suggest that meaningful repeat-purchase predictions can be made from first-purchase data alone.**

## I. Introduction

Retaining existing customers is far less costly than acquiring new ones, making repeat-purchase prediction a critical strategic capability for retail businesses. A reliable model can help marketing teams target potential repeat customers with timely promotions, increasing overall retention.

In this project, we investigate whether it is possible to predict repeat purchases based solely on a customer's first order, without the benefit of historical purchase data. This "one-shot" prediction approach could be applied in both online and brick-and-mortar settings.

**Inputs and Outputs:** The input to our algorithm is a set of 26 engineered and standardized features describing a customer's first order, including order-level metrics, product category, and shipping details. The output is a binary prediction indicating whether the customer will make a repeat purchase within 90 days.

## II. Dataset

**Source:** Data was sourced from Kaggle-DataCo SMART SUPPLY CHAIN FOR BIG DATA ANALYSIS dataset, containing 180,519 product-level transaction records from an international shipping company. Each record corresponds to an item purchased in a customer order. The dataset includes 53 features covering order details, customer attributes, product characteristics, shipping and delivery metrics, and geographic data. Since multiple rows can correspond to the same order or customer, additional aggregation and feature engineering were required.

**Data preprocessing:** We engineered two new features:

1. The difference between actual and scheduled shipping days.
2. A binary `repeat_customer` flag, defined as having made a repeat purchase within 90 days of the first order.

Highly collinear features were identified and reduced to avoid redundancy. After filtering to only first-purchase records, the dataset contained 6,503 rows. The data was stratified and split into training (60%), validation (20%), and test (20%) sets. Three numerical variables were log transformed, five numerical variables were standardized via z-score normalization, and six categorical columns, including three numerical categories with discrete non-continuous values, were one-hot encoded. Final shapes: `X_train` (3,901 × 26), `X_val` (1,301 × 26), `X_test` (1,301 × 26).

**Exploratory Data Analysis (EDA):** In Fig. 1, a heatmap was generated to identify which variables exhibit collinearity among the chosen columns. For example, 'Sales per customer' and 'Order Item Product Price' are highly collinear, and one of the collinear features was dropped before model training.

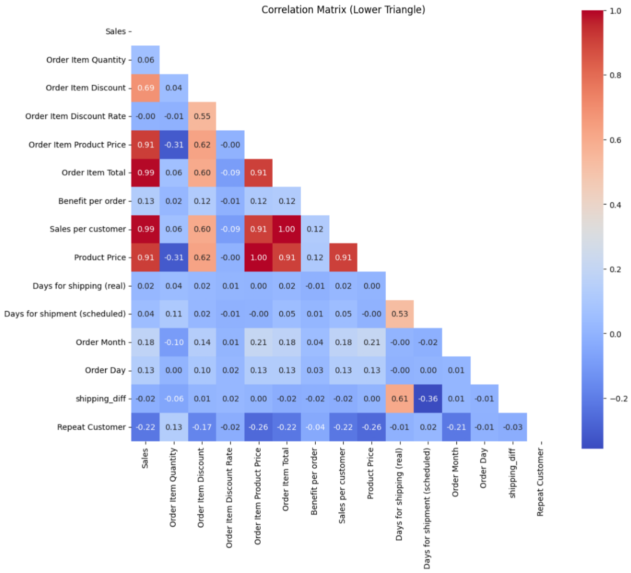


Fig. 1: Heatmap of Standardized Features

In Fig. 2, histograms were used to visualize the skewness and identify any features that contained outliers. The 'Order Profit Per Order' distribution shows an extreme left skew, with most orders having a profit near the mean. Both 'Sales

**Team members and contact information:** *Shubham Banavalikar* (shubhambb@ischool.berkeley.edu), *Mikel Calderon* (mikelcal@berkeley.edu), *Ben Robbins* (ben.robbins@ischool.berkeley.edu), *Alex TenPas* (alextenpas@berkeley.edu)

per customer' and 'Order Item Product Price' show a strong right skew, with the distributions looking very similar. This similarity points to high collinearity between them. These both indicate that most customers purchase generally cheaper products and fewer on average. The remaining columns show weak skewness, but are more centrally located around their mean. Where appropriate, a log transformation was applied.

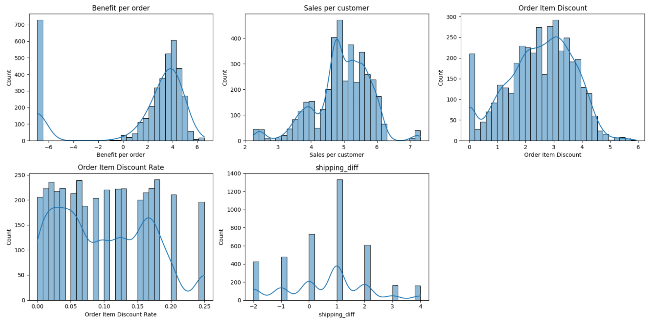


Fig. 2: Histograms of Standardized Features

In Fig. 3, the order profit per order over time shows the standard deviation varies between 1 and -5 standard deviations. The data collected in 2017 is more volatile, suggesting that sales before 2017 were relatively stable.

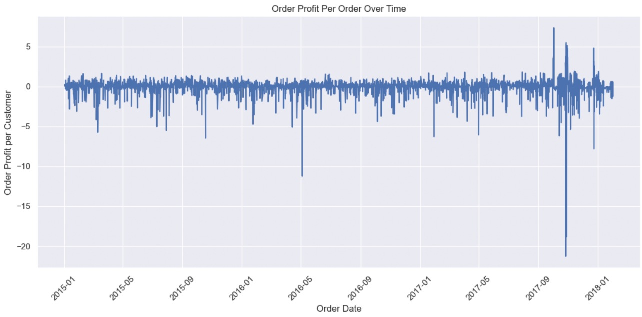


Fig. 3: Order Profit Per Order Over Time

**Data Challenges:** The raw dataset exhibited heavy class imbalance (~95% repeat customers) due to transaction-level representation and multiple orders per customer. Restricting to first orders reduced the imbalance to ~70/30. Applying a 90-day window for repeat classification aligned the target definition with standard business cycles. Additional challenges included selecting nominal columns useful for modeling and removing features with excessive cardinality (e.g., `Country`, 164 unique values) and handling high collinearity.

## III. Related Work

Prior research has addressed repeat-purchase prediction using both traditional machine learning and deep learning techniques.

Lianzhai & Hariyanto (2024) studied online retail behavior following promotional events for online retailers, labeling customers as repeat or non-repeat based on follow-up purchases. Their dataset included demographic and behavioral signals such as age, gender, and clickstream data from the merchant. They evaluated XGBoost, LightGBM, and logistic regression. In their research they found that LightGBM could reasonably predict the repeat customers as the most effective with a test AUC of 0.7411.

In *Deep Temporal Features to Predict Repeat Buyers,* Anand et al. (2015) approached the problem by leveraging historical data of each customer, using Long Short-Term Memory (LSTM) networks to model potential sequential purchasing patterns. They combined deep temporal features with quantile regression to improve accuracy, though results were reported using MSE, making direct comparison to Lianzhai and Hariayanto's work difficult.

While both studies demonstrate it is possible to predict repeat customers given sufficient historical or behavioral data, our approach investigates a more rigid constraint: predicting repeat purchases from first-order information alone. This constraint increases applicability to settings where customer histories are unavailable, such as first-time in-store shoppers, and not limited to the online space.

## IV. Methods

We evaluated five modeling strategies:

1. **Baseline Majority Class Classifier:** Predicted all customers as repeat purchasers, establishing a reference point for accuracy and AUC.
2. **Logistic Regression:** Implemented in both scikit-learn and TensorFlow (single dense layer with sigmoid activation), optimized using stochastic gradient descent and binary cross-entropy loss.
3. **Random Forest:** An ensemble of decision trees with bootstrap aggregation, tuned for number of estimators, maximum depth, and minimum samples per split.
4. **Gradient Boosted Trees:** Implemented using `HistGradientBoostingClassifier`, optimized for learning rate, maximum depth, and number of iterations.
5. **Feed-forward Neural Network:** A multi-layer perceptron with three dense layers, dropout regularization, and mixed tanh/ReLU activations, compiled with Adam optimizer and binary cross-entropy loss. Early stopping was applied to prevent overfitting.

Given the class imbalance, we trained versions of the neural network and logistic regression models with class weights proportional to inverse class frequency. Finally, we created a **stacked ensemble** of the Gradient Boosted model and the L2-regularized neural network to leverage complementary strengths in recall and precision.

## V. Experiments and Results

Starting with the baseline majority-class classifier, we achieved 64% accuracy and an AUC of 0.50, indicating that it was unable to distinguish between the classes. No further experiments were needed for this model. For the Logistic regression model, we tried several improvements on the TensorFlow version using Keras Tuner; however, there were no significant improvements. For the scikit-learn version of the LR model, we tested a variety of max iterations, starting with the standard 1000, 5000, and ultimately deciding on 50, since our model stopped after ~40 iterations and there was no visible improvement or degradation. All models were trained using our 60/20/20 split, 60% for training, 20% for validation, with 20% held out for testing once we identified our best-performing model.

For our FNN model, we experimented with several combinations of units for each layer, as well as different activation methods. One pattern we noticed for this model was that, instead of slowly reducing the number of units between layers, we achieved more consistent results with a sharp drop of units, e.g., 128 > 64, 64 > 16. We also experimented with different dropout values, from 0.1 to 0.6. We found that an offset drop also resulted in an improvement in the overall validation loss and accuracy. The last parameter we experimented with was the default learning rate by lowering it just enough to slow down overfitting (typically for Adam, it is 0.001). Validation and test scores remained closely aligned after tuning, indicating minimal overfitting. After tuning, the feed-forward neural network reached ~85% accuracy and ~0.84 AUC with class weighting and dropout regularization. Applying class weights to the LR and FNN models yielded a modest AUC gain (+0.01) for both neural and logistic regression models.

For our Random Forest and Gradient Boosted Trees, we tested max depth, number of estimators, as well as minimum and maximum nodes to split. After setting our best performing parameters, both achieved >83% accuracy and ~0.85 AUC. See Fig. 4 for a plot of the AUC for each of these models.

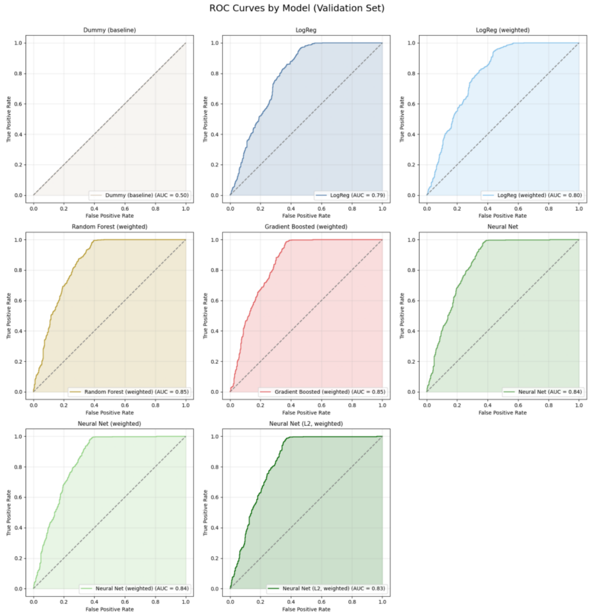
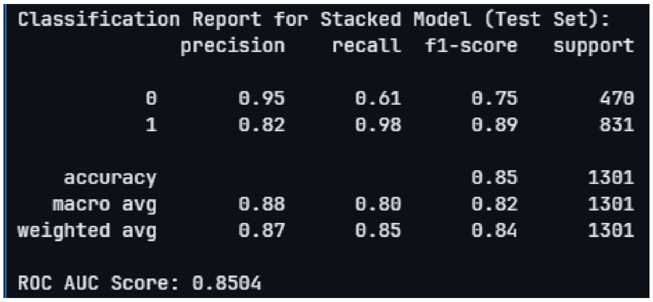


Fig. 4: AUC Scores for distinct ML Models

The stacked ensemble of the Gradient Boosted model and neural network was not extensively tuned, since we used the parameters that yielded the best results for both models. After fitting and training, the model matched the neural network's 85% accuracy and 0.85 AUC but offered more balanced class-wise precision and recall. While the ensemble did not improve overall AUC, it provided more consistent subgroup performance. The test data results for this model can be seen in @img-stacked-report below.



```
Classification Report for Stacked Model (Test Set):
              precision    recall  f1-score   support

           0       0.95      0.61      0.75       470
           1       0.82      0.98      0.89       831

    accuracy                           0.85      1301
   macro avg       0.88      0.80      0.82      1301
weighted avg       0.87      0.85      0.84      1301

ROC AUC Score: 0.8504
```

Fig. 5: Performance of Stacked ensemble against Test data

## VI. Segment Analysis

We examined subgroup performance using the Gradient Boosted model's feature importances as shown in Fig. 6. "Pacific Asia" emerged as the top market predictor, followed by several product category indicators.

- Fan Shop: ~25% representation in validation and test sets; consistently associated with higher repeat-purchase likelihood.
- Book Shop and Pet Shop: <6% representation; lower reliability due to small sample size.
- Regional AUC: Highest in Pacific Asia (~0.93 test), strong in Europe (~0.86), moderate in LATAM (~0.62), and lowest in USCA (~0.69).
- Sales per Customer: Repeat customers generally spend less per transaction (negative z-scores) than non-repeat customers, supporting the pattern that frequent, moderate spenders drive higher repeat rates.

Given the rarity of some categories, their misclassification has less overall impact than errors in high-frequency segments like Fan Shop. Detailed AUC-by-segment values are provided in the following figure.

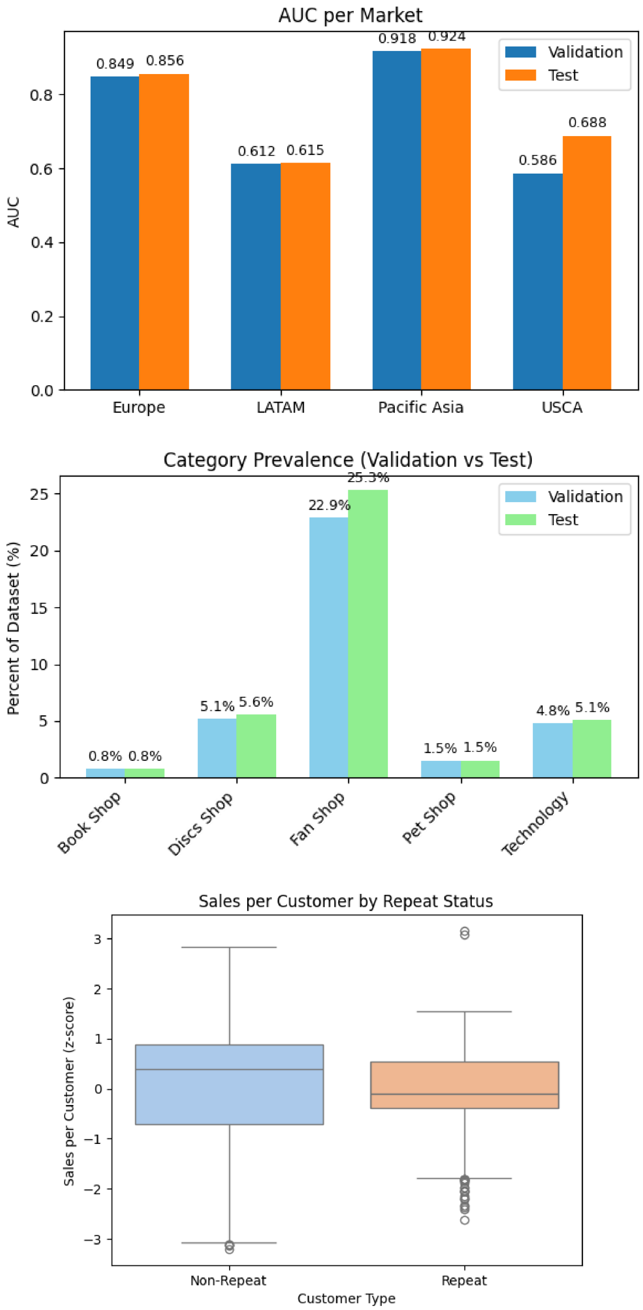


Fig. 6: Segment Analysis Results. Top: AUC performance by market region, showing highest performance in Pacific Asia (~0.93) and Europe (~0.86), with moderate performance in LATAM (~0.62) and USCA (~0.69). Middle: Prevalence of categorical product categories, highlighting Fan Shop as the most represented category (~25%) followed by other product segments with smaller representation. Bottom: Sales distribution comparison between repeat and non-repeat customers, demonstrating that repeat customers generally exhibit lower per-transaction spending (negative z-scores) compared to non-repeat customers.

## VII. Conclusion

Our experiments show that repeat-purchase prediction from first-order data is feasible, with models achieving up to 0.85 AUC. The L2-regularized neural network and Gradient Boosted model performed best, and the stacked ensemble provided the most balanced precision and recall. We therefore recommend the ensemble for deployment, as its stability across subgroups offsets the lack of overall AUC improvement.

Limitations include the modest dataset size after filtering to first orders and the absence of demographic or behavioral features. Future work should explore incorporating additional data sources, applying explainable AI techniques (e.g., SHAP, LIME), and testing advanced architectures like attention-based models for improved generalization.

## VIII. Contributions

- **Shubham Banavalikar:** I worked on a preliminary approach involving customer-level features and helped with some data pre-processing steps (such as how to classify a repeat customer in a quantifiable way). Additionally, I did model experimentation for logistic regression, neural network, and decision trees (random forest and gradient-boosted decision tree). Finally, I helped with the methodology section.

- **Mikel Calderon:** Setting up a GitHub repository, authored Experiments & Results, Segment Analysis, and Conclusion sections. Proofread/edited the final report and set the contents in Quarto format before submission.

- **Ben Robbins:** I re-read the papers and wrote the related work section. I also wrote the abstract and, as we all did, created a model to test (I created a gradient boosted tree using XGBoost but it was over-fitting even after some fine-tuning so we went with another member's gradient boosted tree for the final model that had better engineered features). I also added the references section and made the powerpoint outline.

- **Alex TenPas:** Pushed data preprocessing notebook to GitHub repository with EDA. Wrote data preprocessing and EDA sections. Created a NN model and checked how standardized and PCA affected model performance.

- **All:** We all created our own model(s) which we uploaded to github. We all

## IX. References

Anand, G., Kazmi, A., Malhotra, P., & Vig, L. (2015). Deep temporal features to predict repeat buyers. Retrieved from https://www.researchgate.net/publication/292972291

Lianzhai, D., & Hariyanto, D. T. (2024). Machine learning modeling for forecasting repeat purchases in online shopping. Retrieved from https://journal.irpi.or.id/index.php/malcom/article/view/1388/643

# X. Appendix

## A. Loss and Accuracy Plots for Neural Network Models

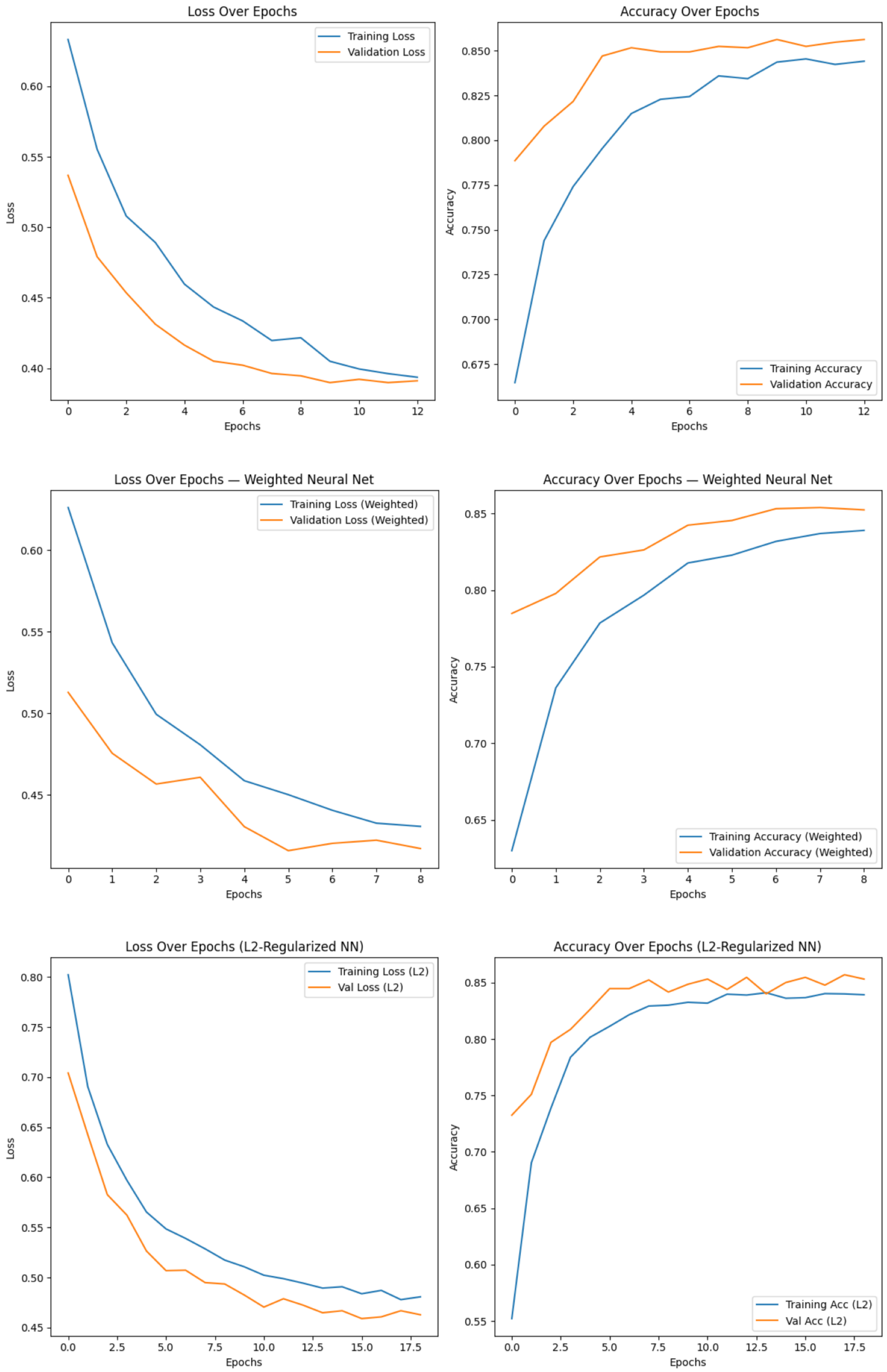


Fig. 7: Neural Network Model Comparison. Top: Standard Neural Network performance showing baseline model metrics. Middle: Weighted Neural Network (wnn) demonstrating the effect of class weighting on model performance. Bottom: L2-regularized Neural Network (l2-nn) showing improved generalization through regularization techniques.

### B. Model Confusion Matrices Comparison

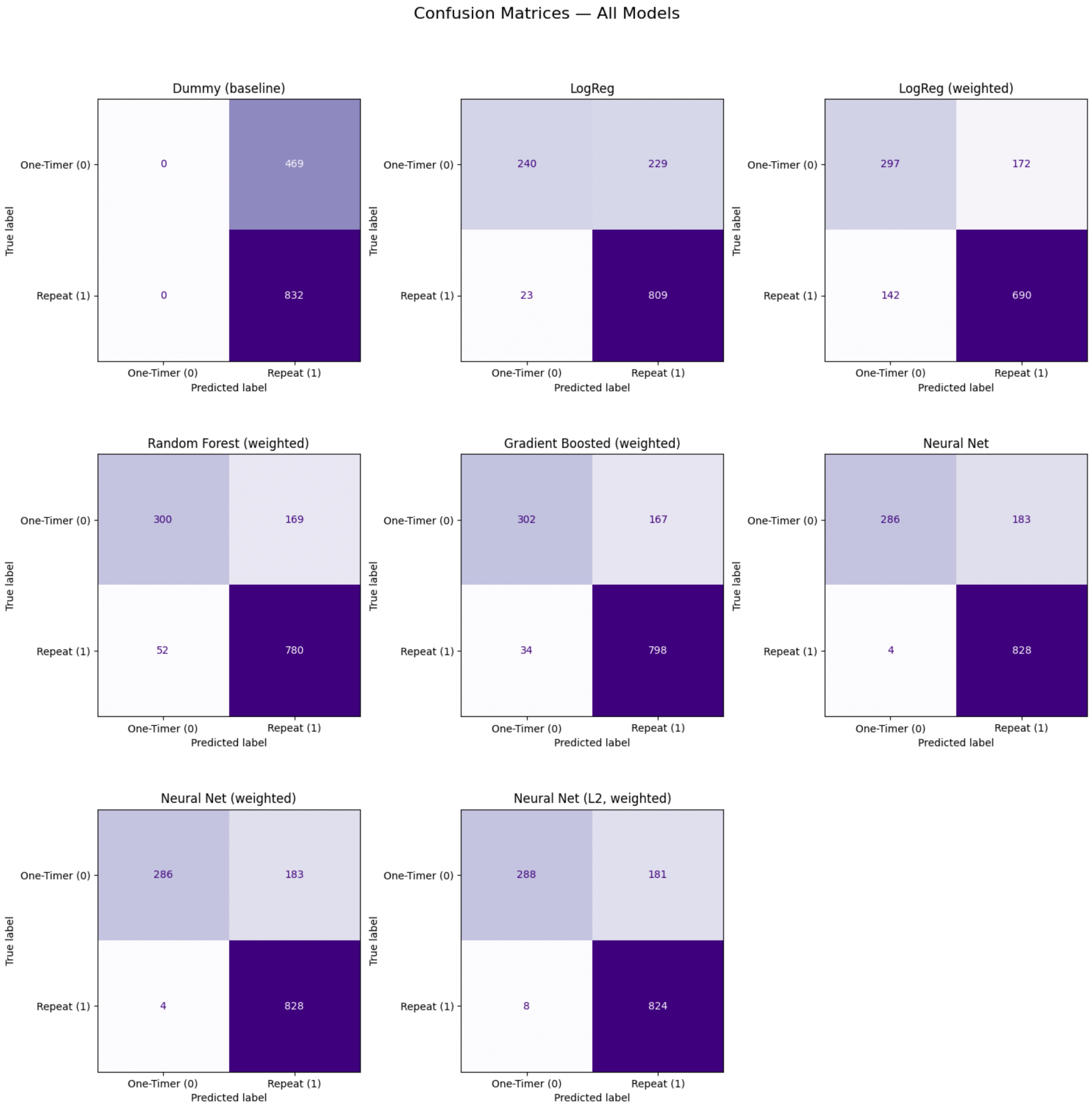


Fig. 8: Comparison of confusion matrix by model

Table I: Validation Metrics by Model for Both Classes

| Model | precision(0) | precision(1) | precision(macro avg) | precision(weighted avg) | recall(0) | recall(1) | recall(macro avg) | recall(weighted avg) | f1-score(0) | f1-score(1) | f1-score(macro avg) | f1-score(weighted avg) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Neural Net (weighted) | 0.986200 | 0.819000 | 0.902600 | 0.879300 | 0.609800 | 0.995200 | 0.802500 | 0.856300 | 0.753600 | 0.898500 | 0.826100 | 0.846300 |
| Neural Net | 0.986200 | 0.819000 | 0.902600 | 0.879300 | 0.609800 | 0.995200 | 0.802500 | 0.856300 | 0.753600 | 0.898500 | 0.826100 | 0.846300 |
| Neural Net (L2, weighted) | 0.973000 | 0.819900 | 0.896400 | 0.875100 | 0.614100 | 0.990400 | 0.802200 | 0.854700 | 0.752900 | 0.897100 | 0.825000 | 0.845100 |
| Gradient Boosted (weighted) | 0.898800 | 0.826900 | 0.862900 | 0.852900 | 0.643900 | 0.959100 | 0.801500 | 0.845500 | 0.750300 | 0.888100 | 0.819200 | 0.838500 |
| Random Forest (weighted) | 0.852300 | 0.821900 | 0.837100 | 0.832900 | 0.639700 | 0.937500 | 0.788600 | 0.830100 | 0.730800 | 0.875900 | 0.803400 | 0.823600 |
| LogReg | 0.912500 | 0.779400 | 0.846000 | 0.827400 | 0.511700 | 0.972400 | 0.742000 | 0.806300 | 0.655700 | 0.865200 | 0.760500 | 0.789700 |
| LogReg (weighted) | 0.676500 | 0.800500 | 0.738500 | 0.755800 | 0.633300 | 0.829300 | 0.731300 | 0.758600 | 0.654200 | 0.814600 | 0.734400 | 0.756800 |
| Dummy (baseline) | 0.000000 | 0.639500 | 0.319800 | 0.409000 | 0.000000 | 1.000000 | 0.500000 | 0.639500 | 0.000000 | 0.780100 | 0.390100 | 0.498900 |