Google Merch Store Churn: Predicting Sale Conversion Using Machine Learning

Team Members and Emails: Alec Heyde (alecheyde@berkeley.edu), Peter Liu (pliu27@berkeley.edu), Gatsby Frimpong (gatsbytv@berkeley.edu), Vanessa Navarro (vanessan@berkeley.edu)
Github Repo: https://github.com/UC-Berkeley-I-School/DS207_Summer25_Project

ABSTRACT
This project investigates whether a user session will result in a purchase. Accurately predicting purchases can help businesses reduce unnecessary spend on broad marketing strategies and instead focus resources on high-potential sessions, improving both efficiency and revenue outcomes. To address the extreme class imbalance, we used upsampling techniques and tuned model hyperparameters to generalize effectively. We evaluated a range of models including logistic regression, k-nearest neighbors, neural networks (with and without embedding), and random forests. The random forest model stood out, achieving highest precision (0.76), recall (0.996), and F1 score of (0.86) on the positive class. It also achieved a perfect AUC of 1.00, outperforming more complex architectures like CNNs with embeddings. While our current work does not implement real-time marketing/website interventions, it builds a strong predictive baseline. Future efforts can focus on identifying "on the fence" sessions with moderate purchase probability, who would most likely respond to targeted offers or nudges. Effectively identifying this cohort will help to optimize marketing spend and drive revenue.

INTRODUCTION
E-commerce businesses face extremely low conversion rates, leading to inefficient marketing spend on broad strategies rather than targeted approaches. This project frames conversion prediction as binary classification to enable businesses to focus resources on high-potential sessions.

The input to our models is session-level data from the Google Merchandise Store including traffic source information (channelGrouping, source), user behavior metrics (pageviews, timeOnSite), and technical details (device_category). We use machine learning models to predict session_convert, which is 1 if a transaction occurred during the session and 0 otherwise.

We evaluated methods from classical algorithms (logistic regression, K-NN, SVM) to advanced neural networks. The primary challenge was extreme class imbalance where converting sessions represent only 2.5% of data. We addressed this through upsampling minority class and undersampling majority class. Our Random Forest achieved F1-score of 0.86, demonstrating that well-tuned classical ensemble models can outperform complex neural architectures on structured data.

RELEVANT WORK
CLV Prediction: Norouzi (2024) used neural networks to predict Customer Lifetime Value focusing on NPS, ATV, and CES relationships, employing SHAP for interpretability. There are many similarities between the approach employed by Norouzi's study and our own, but there are some differences:

- **Prediction Target:** Our project focuses on predicting a binary classification task that determines the outcome of a single user visit. In contrast, Norouzi predicts CLV for that user, which is a continuous value.
- **Model Performance:** The neural network in Norouzi's study proved to be a highly effective model. This was the opposite in our findings. Classical modeling or more specifically our Tuned Random Forest proved to be a lot more effective than the neural network approach.

The difference with our own results provides insight: for predicting immediate, transactional outcomes based on structured session data, well-tuned classical models can be more effective than more complex deep learning architectures.

Attention-Enhanced LSTM: Kasemrat & Kraiwanit (2025) identified high-value customers using attention mechanisms for interpretability, predicting long-term customer status versus our immediate session conversion. A few noteworthy differences between their project and our own:

- **Problem Framing**: Their model predicts a customer's long-term status (high-value), whereas our model predicts an immediate, transactional outcome (session conversion).
- **Interpretability**: Their use of an attention mechanism is a direct attempt to build a more transparent deep learning model. Meanwhile, our project doesn't allow for much interpretability of the weights, features, etc affecting the predictions.

Lastly, although the Attention-enhanced LSTM and our own model performed fairly well the former has almost perfect results, which leads us to believe it may be worthwhile to try this in order to increase our phenomenal results seen in the Random Forest model.

DATASET

We use Google Analytics 360 data from the Google Merchandise Store. The dataset captures typical information for an e-commerce website, including traffic source data, user behavior (pageviews, time on site, new visits), and transaction outcomes.

The raw dataset contains 18,608,748 sessions and 14 columns.Our target is a binary variable (session_convert) indicating whether a transaction occurred during the session (1) or not (0). The original input features of interest include numeric and categorical variables:

- Numeric: visitNumber, pageviews, hits, timeOnSite, newVisits,
- Categorical: channelGrouping, source, devive_category, country, productname, productSKU

We took the following data preprocessing steps: Imputed missing values with a 0 → Created target variable, session_convert, which is 1 if a transaction occurred and 0 otherwise → Bucketed country and source variables into top 10 of each category and 'other' to encourage learning → Took the log1p for visitNumber, pageviews, timeOnSite, newVisits to help mitigate impact of outliers → Standardized all of the numeric variables → One-hot encoded the categorical variables

Data Splits: Training (280356, 455), validation (93452, 455), test (93452, 455) using 60:20:20 split maintaining target proportion.

Class Imbalance: Original 2.5% positive class increased to 15% through upsampling, optimizing pattern learning while minimizing overfitting risk.

Exploratory Data Analysis (EDA): The Google Analytics dataset spans from 2016-08-01 to 2017-08-01. Our EDA focuses on the subset between 2017-01-01 and 2017-08-01, comprising 467,260 rows and 11 columns. Numerical columns (visitNumber, pageviews, hits, timeOnSite, newVisits, transactions, session_convert), Categorical columns (channelGrouping, source, device_category, country). The outcome variable session_convert is derived from the transactions column: it is 1 if transactions > 0, otherwise 0. Because of this dependency, transactions is excluded as a model input but included here for exploratory purposes.

Skewness and Log Scale - Histograms show right-skewed distributions in `visitNumber`, `pageviews`, `hits`, `timeOnSite`, and `transactions`. Applying a log scale helps normalize these distributions, making them more suitable for modeling.
The `channelGrouping` and `device_category` distributions appear acceptable as-is.

Reducing Categorical Cardinality - The country and source columns are dominated by a few categories. Smaller groups can be binned into an "Other" category. Since the United States dominates `country`, a binary `IS_USA` flag may be more effective.
`Sources` like google, google.com, and mail.google.com can be consolidated into one category, as can facebook.com and m.facebook.com.

Outliers and Trends - Boxplots reveal outliers across categorical groups, which may need to be removed prior to modeling. Regression lines suggest linear relationships between transactions and features like `hits`, `visitNumber`, and `timeOnSite` - all of which are closely tied to the `session_convert` outcome.

Visualization - Conversion Rate by New vs Returning Visits: The bar chart (Appendix: Fig. 6) clearly illustrates that returning visitors are substantially more likely to convert than new visitors. Specifically, returning visitors exhibit a conversion rate of 3.45%, which is approximately five times higher than the 0.70% conversion rate observed for new visitors. Returning visitors often possess a higher degree of familiarity and trust with the Google Merchandise Store, potentially having browsed products previously, compared prices, or even added items to a cart. Their return indicates a stronger intent or a progression in their buyer journey, leading to a higher propensity to complete a transaction.

Visualization - Correlation Matrix: The correlation Matrix (Appendix: Fig. 7) reveals the similarity between features within the dataset. One key observation is that there is a high positive correlation between page views and hits. This confirms that the two features are redundant and one can be removed to reduce multicollinearity and simplify the model without significant loss of information.

METHODS

To evaluate and improve conversion prediction, we first implemented a baseline model and then developed three improved modeling approaches: tree-based models (scikit-learn), logistic-regression neural networks (Keras Sequential), and convolutional neural networks with embeddings (also Keras Sequential). We utilized sci-kit learn and Keras for building and testing models. For interpretability, we utilized decision trees and random forests. Shallow vs. deep architecture comparison was made possible with logistic-regression networks. CNNs with embeddings were utilized to test for an effect of patterns viewed on conversion.

Baseline Model: Random classifier predicting conversion with 2.5% probability to match the dataset distribution, achieving 97% accuracy but 0.03 F1-score on positive class, highlighting the core challenge of identifying scarce converting sessions.

Tree-Based Models: We chose tree-based models as our lead method because of their high interpretability and capacity to deal with the mixed data types found in our dataset. Since our objective is to be able to identify converting sessions and also to reveal user behavior patterns, tree-based models provide uncluttered decision paths that can guide business strategy. These models are ideally suited to our problem as they can handle both numerical variables and categorical variables with minimal preprocessing.

Our approach was both single decision trees and random forest sets with scikit-learn. Our best-performing model was a hyperparameter-tuned random forest (F1-score: 0.861), which was achieved through grid search across multiple configurations, ultimately selecting 64 estimators with a maximum depth of 64. This deeper, wider ensemble captured the fine-grained user activity patterns that define converting sessions well, with very good recall of 0.998 and fair precision of 0.757. The decision tree graph showed significant predictors of conversion to be pageviews > 1.6 and US sessions, with timeOnSite > 1 also contributing to conversion probability. Such interpretability is ideal for discovering high-value-low-frequency buy sessions and provides a clear foundation for subsequent personalization possibilities.

Logistic Regression Neural Networks: We used logistic regression neural networks with TensorFlow/Keras to explore deep learning methods for binary classification and shallow versus deep architectures. Neural networks are ideal for learning difficult non-linear relationships found in our high-dimensional feature space.

Our neural networks operate via feedforward propagation in which input features pass through several layers of neurons, with each carrying out linear transformations and then non-linear activation functions (ReLU for hidden layers and sigmoid for output). We employed two architectural approaches: a shallow network with one hidden layer (16 neurons with ReLU activation) and a deeper network with multiple hidden layers (32→16 neurons with dropout regularization). Both architectures used sigmoid activation to give 0-1 output conversion probabilities. The shallow network recorded an F1-score of 0.443 (precision: 0.301, recall: 0.839), and the deeper network recorded a higher F1-score of 0.447 (precision: 0.305, recall: 0.841). ROC curves were found to work very well with AUC scores of 0.99 for the shallow network and 0.97 for the deep network. These both had superb recall but also generated higher rates of false positives than tree-based approaches. Similar performance between architectures suggests that, for this tabular data, additional network complexity provided minimal gains.

Convolutional Neural Networks with Embeddings: We used convolutional neural networks with embedding layers to explore whether sequential patterns and learned categorical representations would enhance conversion prediction. The motivation was based on the idea that categorical variables may have hidden semantic associations encoded via learnable embeddings, and that CNNs would be able to find sequential patterns in user action that typical models would not.

The embedding CNN model first maps categorical variables (productSKU, channelGrouping, source, device_category, country) to dense vector representations with embedding layers of sizes ranging from 10 to 100. These embeddings are concatenated with numerical features and passed to a convolutional layer (64 filters, kernel size 5, ReLU activation) followed by global max pooling to extract salient patterns. The model is capped with fully connected layers with dropout regularization (0.3), eventually giving conversion probabilities with sigmoid activation. Comprehensive hyperparameter tuning studied input length changes (100-1000 tokens), embedding sizes (5-20), learning rates (1e-1 to 1e-3), and batch sizes (1024). The best-performing setup used an F1-score of 0.274 (precision: 0.411, recall: 0.205) with AUC scores reaching 0.89. But the embedding solution was still behind tree-based

algorithms, showing that for this tabular e-commerce data, significant feature interactions captured by ensemble methods were still better than learned embeddings as well as convolutional patterns. This outcome emphasizes that classical models like Random Forest have a tendency to outperform advanced neural networks over structured tabular data.

EXPERIMENTS, RESULTS, & DISCUSSION

Architecture and Tuning: Using a Decision Tree model (Figure 1) as a starting point provided valuable information on feature importance and decision boundaries. The tree plot diagram revealed that page-views greater than 1.6 combined with countries such as the United States are already very strong indicators, with time-on-site exceeding 1 minute providing additional predictive power.

Random Forest proved to be both a strong predictor as well as being a fast trainer. The model trains under 10 seconds. Using the f1 score on the positive class as the evaluation metric, we systematically trained various combinations of max depth, and estimators. After exhaustive experimentation, we settled with the optimal configuration of 64 max depth and 64 estimators. Adding further complexity proved to have diminishing returns.

Neural network architecture requires special consideration given our imbalanced dataset. We increased the default batch size from 32 to 1024 to ensure each batch contains sufficient positive samples throughout. We tuned two variants; one without hidden layer, and one with a second dense layer in an attempt to higher abstraction, while a dropout layer prevents overfitting. The model contained up to 1633 parameters and trained around 30 seconds.

The Convolutional Neural Network with Embedding presented a different set of challenges. With a vocabulary size of 254 and up to 21809 parameters, it took approximately an hour to train each configuration. Using product names for each session as a one dimensional input, we methodically experimented with various input token length, embedding dimension, and learning rate. Using f1-score on the positive label as a comparison metric, the optimal configuration was 500 input token length and 20 embedding dimensions. Principal Component Analysis was implemented to reduce embeddings to two-dimensional space for visualization which revealed interesting patterns for some model configurations.

Overfitting Prevention: Monitored training/validation curves for divergence, implemented early stopping, and used dropout layers for generalization on unseen data.

Results - Overall Model Performance: Random Forest proved to be the best performer based on f1-score on positive class when tested on the test split. The ensemble nature of Random Forest, which combines the power of multiple weak learners into a single robust predictor worked particularly well on our imbalance dataset. The model achieved on the positive class, 0.755 precision, 0.996 recall, and 0.859 f1-score (Figure 2). By contrast, the logistic neural network models attained 0.305 precision, 0.841 recall, and 0.447 f1-score, while the embedding convolutional neural network was at 0.411 precision, 0.205 recall, and 0.274 f1-score. Despite the much longer and complex nature, the neural network was outperformed by the much simpler Random Forest model.

Subgroup Analysis - Logistic Neural Network: Systematic subgroup analysis on channel group, countries, sources, and device categories showed the Random Forest again outperforming the rest. Interestingly, the relative performance among the neural networks differs within a subgroup, from between 40% to 69% in device category, to between 31% and 55% in Channel. This variation suggests possible ensemble approaches for different data segments, or utilizing different models for specific segments.

Subgroup Analysis - Embedding Convolutional Neural Network: The subgroup analysis was performed on different popular words within the vocabulary. Using a word cloud visualization, a set of frequently used words were selected (Figure 3). Evaluation metrics revealed popular words did not perform any better. This suggests that the embedding layer learned robust representations across the vocabulary and did not exhibit frequency bias.

Threshold Optimization: Classification threshold optimization is a valuable tool post training. The standard 0.5 threshold balances precision and recall. Often business requirements will make adjustments and trade-off considerations between increasing recalls and capture more false positives. Experiment showed that by reducing the threshold to 0.1, the embedding convolution neural network (model 4) captured 2 times more true positives (Figure 4a, 4b).

CONCLUSIONS

The study highlighted the importance of managing class imbalance in a dataset. While there are other tools to mitigate, we chose to focus on upsampling the positive class to an effective ratio to start experimentation. We learned a valuable lesson on carefully choosing the class ratio to train a robust model while avoiding the pitfalls of overfitting due to excessive duplicate samples.

The Random Forest's ensemble method proved to be very powerful while showing superior speed in training and robustness in predictive power. Both neural network approaches were no match despite having a lot more complexity and in theory being able to capture more sophisticated patterns.

Subgroup analysis revealed the importance of segment evaluation, and model specificity. It also suggests that a one size fits all approach may not best capture the variations in the data, and an ensemble approach could bridge gaps.

Classification threshold is an important post training tool for businesses to tune the model to their specific goals. The trade-offs between precision and recall is a dynamic that can be adjusted without completely retraining a model.

Given more time, we would like to explore other ways to handle imbalance dataset, using the keras functional API to create combined features and embeddings, and see how they would interact within a model. Other considerations are transfer learning with pre-train embeddings in the product space, and transformer-based models to explore time series nature of a visitor session.

Our work demonstrates that machine learning can meaningfully address one of the hardest problems in digital commerce: identifying which sessions are likely to convert despite extreme class imbalance. By upsampling the minority class and testing a range of models, we were able to build a robust prediction pipeline that identifies high-value sessions with a high degree of precision. The Random Forest model consistently outperformed both simpler and more complex alternatives, achieving 0.76 precision and 0.996 recall on the positive class. This is largely due to its ability to capture subtle, non-linear patterns in user behavior without extensive feature engineering, which is especially useful in structured e-commerce datasets.

While the neural network approaches showed promise in recall and offered architectural flexibility, they did not outperform tree-based methods in this setting. This reinforces a practical insight for applied machine learning: classical ensemble methods often deliver the strongest performance on structured tabular data and can be more resource-efficient to tune and deploy. If given more time or compute, we would explore hybrid ensemble models, segment-specific architectures, and embedding transfer learning to improve generalization across user types and traffic sources. More importantly, we would focus on identifying "on-the-fence" sessions, those with moderate conversion probability, to drive targeted marketing interventions and unlock incremental revenue opportunities. Our findings provide a strong foundation for building session-aware personalization strategies in digital commerce.

CONTRIBUTIONS

- **Peter**: Created starter Notebook to connect to Google Analytics. EDA and early exploratory modelling. Extensive research on how to address imbalanced datasets. Wrote code to upsample dataset. Wrote code to train Random Forest, Shallow and Deep Neural Network in logistic regression adding embeddings (peter_models). Trained extensively embedding models which take about 6-8 hours of training combined. Explored class weight and sample weight to address class imbalance. Executed subgroup analysis, and confusion matrix thresholds. Wrote a summarizing notebook to capture experiment outputs (peter_evaluations). Wrote sections of the reports including Data, Experiments and Results, and Conclusions. Wrote corresponding sections of the presentations.
- **Alec:** Led the data cleaning and preprocessing for the milestone work. Wrote the abstract, the dataset description, including the data source and preprocessing steps, and contributed to the conclusion section of the final report. Created the Alec_models file and modified Peter_data to create Alec_data, enabling an embedding model that utilized more features than just productName. This setup supported the development of my logistic regression model, random forest model, and embedding model.
- **Gatsby:** Developed the first round of classification models, testing logistic regression, random forest and gradient boosting. Analyzed false positive rates and provided initial model selection insights. Created visualizations that communicate data insights gathered so far in gatsby_eda_visualizations. Also, developed a second set of testing in gatsby_models, which utilized K-Nearest Neighbors, SVC, and Logistic Regression with PCA.
- **Vanessa:** Developed two notebooks (vanessa_data and vanessa_models) to run and evaluate key model types, including decision trees, random forests, logistic-regression neural networks, and embedding-based neural networks. Wrote the Methodology section of the final report, synthesizing team workflows and clearly documenting how we progressed from a baseline model to more sophisticated approaches.

REFERENCES

- [1] Norouzi, A. (2024). Predicting e-commerce CLV with neural networks: The role of NPS, ATV, and CES. *Journal of Machine Learning in Commerce*, 8(3), 210-225.
- [2] Kasemrat, C., & Kraiwanit, T. (2025). Attention-enhanced LSTM for high-value customer behavior prediction. *Proceedings of the International Conference on Data Science and Advanced Analytics*, 112-121.
- [3] Gulshan, V., Peng, L., Coram, M., Stumpe, M. C., Wu, D., Narayanaswamy, A., ... & Webster, D. R. (2016). Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. *JAMA*, 316(22), 2402-2410.
- [4] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30.
- [5] TensorFlow. (n.d.). Classification on imbalanced data. Retrieved from https://www.tensorflow.org/tutorials/structured_data/imbalanced_data
- [6] Google Developers. (n.d.). Imbalanced datasets. *Machine Learning Crash Course*. Retrieved from https://developers.google.com/machine-learning/crash-course/overfitting/imbalanced-datasets

APPENDIX :

Figure 1: Decision Tree Information Gain Splits

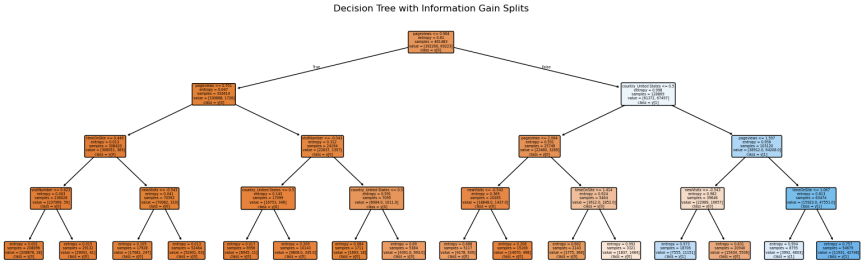


Figure 2: Best Tuned Model

Model Class	Precision	Recall	f1-Score
Baseline	0.00	0.00	0.00
Random Forest	0.755	0.996	0.859
Logistic Neural Network	0.305	0.841	0.447
Embeddings Conv. Neural Network	0.411	0.205	0.274

Table 1: Test Split Results

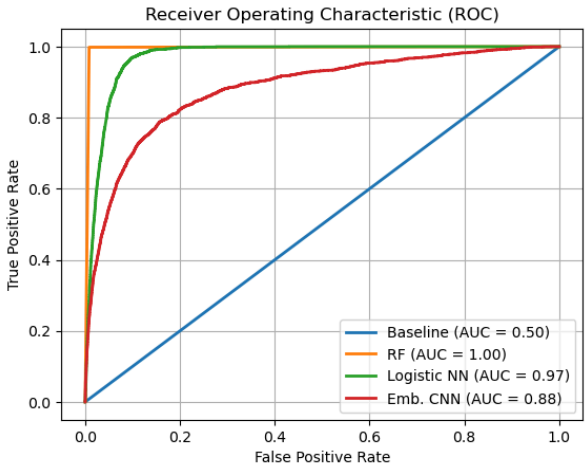


Figure 3: Vocabulary Word Cloud

Figure 4a - Confusion Matrix at 0.5 Threshold

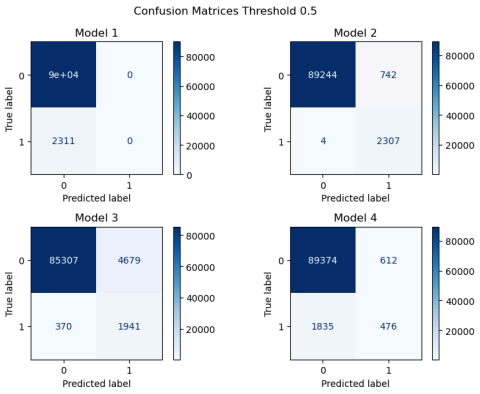


Figure 4b - Confusion Matrix at 0.1 Threshold

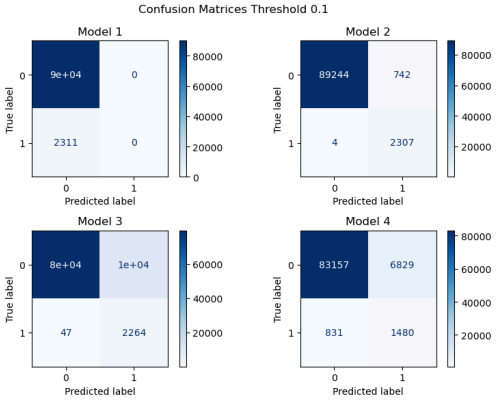


Figure 4: Visualizing Embeddings in 2 Dimension

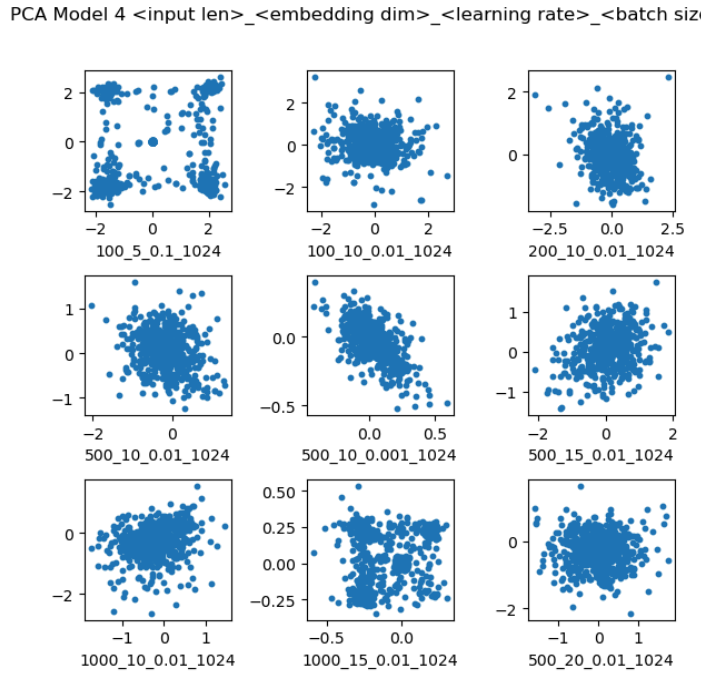


Table 4a - Subgroup Analysis Device Category

	Model 2	Model 3	Model 4
Canada	0.923	0.27	0.275
default	2	0.997	0.998
Germany	1	0	0
India	1	0	0.095
Japan	1	0	0.286
Netherlands	1	0	0
Other	0.891	0.242	0.095
Taiwan	1	0.125	0.444
United Kingdom	1	0.714	0.143
United States	0.858	0.443	0.294

Table 4b - Subgroup Analysis Channel

	Model 2	Model 3	Model 4
Affiliates	1	0.308	0.222
Direct	0.852	0.406	0.297
Display	0.917	0.452	0.286
Organic Search	0.847	0.372	0.24
Paid Search	0.916	0.327	0.22
Referral	0.868	0.509	0.311
Social	0.857	0.346	0.196

Table 4c - Subgroup Analysis Source

	Model 2	Model 3	Model 4
default	1.851	1.355	1.253
dfa	0.947	0.462	0.261
facebook.com	0.833	0.333	0.25
google	0.847	0.367	0.226
google.com	1	0.333	0
m.facebook.com	1	0.5	0

Other	0.893	0.364	0.347
Partners	1	0.308	0.222
youtube.com	2	0.998	0.999

Table 4d - Subgroup Analysis Country

	Model 2	**Model 3**	**Model 4**
desktop	0.862	0.449	0.298
mobile	0.844	0.285	0.139
tablet	0.88	0.286	0.19

Table 2: Embedding Subgroup Analysis

Subgroup	**Precision**	**Recall**	**F1-Score**	**Support**
['google', 'men', 'hero', 'tee']	0.032	0.011	0.017	1567
['short', 'sleeve']	0.028	0.012	0.016	1729
['s', 'vintage']	0.028	0.012	0.017	1045
['cotton', 'short']	0.036	0.013	0.019	1233
['google']	0.025	0.010	0.015	2226
['short']	0.028	0.012	0.016	1729
['google', 'men', 'hero', 'tee', 'short', 'sleeve', 's', 'vintage', 'cotton']	0.040	0.012	0.019	817

Figure 5: Conversion Rate by Channel Grouping

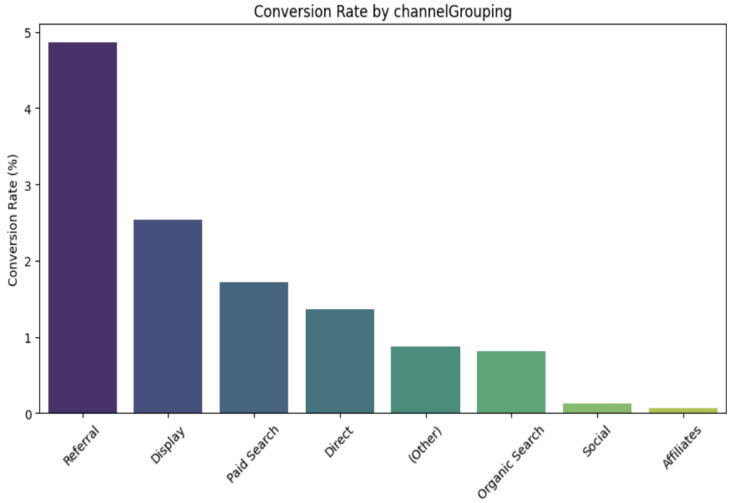


Figure 6: Conversion Rate by New vs Returning Visits

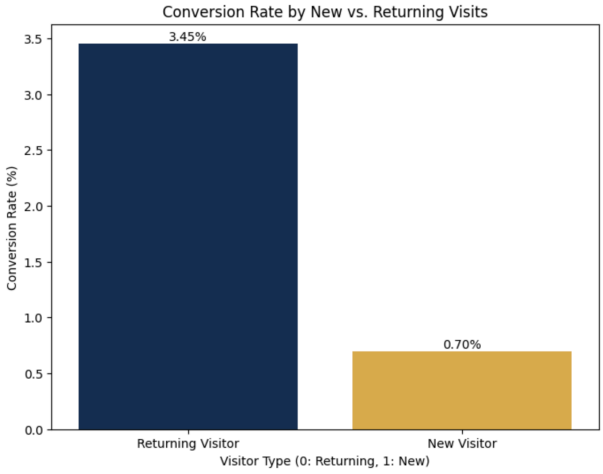
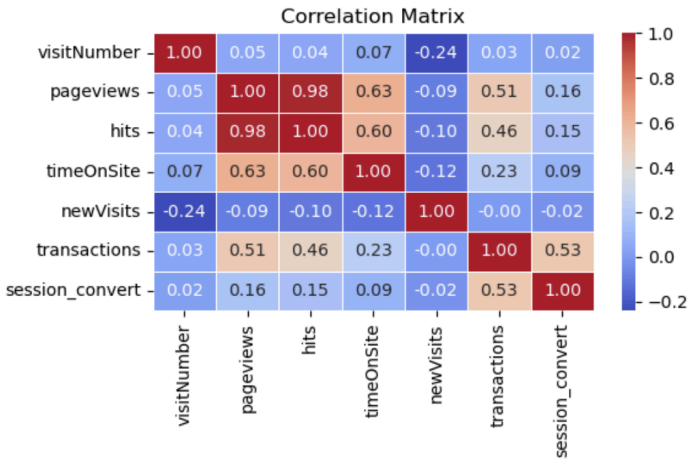
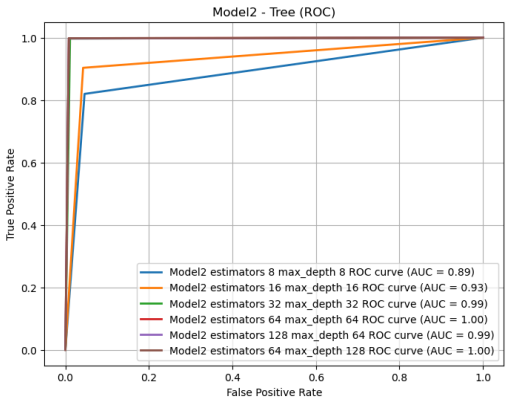
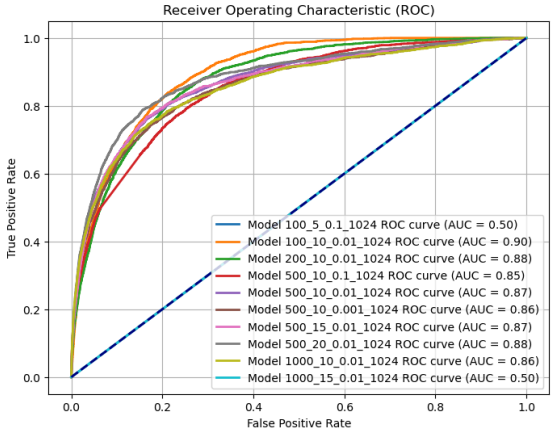


Figure 7: Correlation Matrix







PCA Model 4 <input len>_<embedding dim>_<learning rate>_<batch size>