Project: Click-Through Rate Prediction

Made by Mahi, Kirti, Tanish, Divyasha

INDRAPRASTHA INSTITUTE *of*
INFORMATION TECHNOLOGY
DELHI

Motivation

- Having found nominal CTR predictions to be fairly accurate, they would result in ads tailored to enhance user engagement and, thus, advertising returns.

- Finally, marketing is informed by data: Nothing beats machine learning in making evidence-based decisions that lead to the execution of marketing strategies that are calibrated for effectiveness.

- Professional growth: This project improves your data science and machine learning skill sets, making one more employable in analytics and marketing.

Literature Review

Click-Through Rate Prediction in Online Advertising:

- **Yanwu Yang and Panyu Zhai (2022)** provide a comprehensive literature review on predicting CTR in online advertising. They highlight the evolution of CTR models, their methodological frameworks, and the significant impact of feature interaction complexity on model performance. Their review covers key advancements in models like logistic regression, factorization machines, and deep learning, outlining their strengths and weaknesses.

Ad Click Prediction: A Comparative Evaluation of Logistic Regression:

- **Niharika Namdev and Nandini Tomar (2023)** analyze the effectiveness of logistic regression for CTR prediction in large-scale applications. Their work emphasizes the simplicity and interpretability of logistic regression while acknowledging the challenges posed by non-linear interactions in CTR prediction.

BARS-CTR: Open Benchmarking for CTR Prediction:

- **Jieming Zhu et al. (2023)** focus on the lack of standardized benchmarks in CTR prediction, which leads to inconsistencies in model evaluation. They propose an open benchmarking framework to standardize the evaluation of CTR models, promoting reproducibility and consistency

Feature Analysis & Data Structure

Overview:

- **Columns:**
  - **Index:** `['id', 'click', 'hour', 'C1', 'banner_pos', 'site_id', 'site_domain', 'site_category', 'app_id', 'app_domain', 'app_category', 'device_id', 'device_ip', 'device_model', 'device_type', 'device_conn_type', 'C14', 'C15', 'C16', 'C17', 'C18', 'C19', 'C20', 'C21', 'month', 'dayofweek', 'day', 'hour_time']`
- **Train Dataset:** (5,000,000, 24)
- **Test Dataset:** (4,577,464, 23)

Target Variable:Click
No Click (0): 83%
Click (1): 17% → **Significant class imbalance**

Features:

- **Numerical:** 13
- **Categorical:** 9
- **Time Range:** 2014-10-21 to 2014-10-30

Observations:

- **Y and Click:** These columns appear identical; after correlation analysis, one can be dropped.
- **Month:** Contains only a single entry, adding no useful information, so it can be dropped.
- **Highly Centered Data:** Features like `banner_pos`, `device_conn_type`, `C20`, `C15`, and `C16` are concentrated around certain values, indicating limited variability.
- **Outliers Detected:** Potential outliers found in `C15`, `C16`, `C19`, and `C21`.
- **Redundant & Correlated Features:**
  - `C14` and `C17` are highly correlated; `C17` was removed after model evaluation.
  - **C20:** Shows nearly 47% of values as `-1`, which is inappropriate for a categorical variable, so it will be removed.
- **Categorical Variables:** All categorical features have a large number of unique values.

Feature Distribution & Preprocessing

Data Preparation:

- Dropped `id` column (unique values with no predictive significance)
- Dropped `hour` (derived into multiple columns)
- Dropped C17 (highly correlated with C14
- Renamed `click` to `y` (target) and `hour_time` to `hour`
- After dropping the `hour` column, transformed `hour_time` into `hour`
- Sampled 10% of data for analysis to manage computational resources, preserving the original target ratio (No Click: 83%, Click: 17%)
- Outlier Handling: Treatment of outliers in C15, C16, C19, and C21
- Feature Selection: Removal of redundant features (C17 due to correlation with C14)

Feature Distribution Patterns:

- **Right-Skewed Features:**
  - `C14`: Multiple peaks (15000-25000 range)
  - `C19`: Peak near 0, long tail
- **Discrete-Like Features:**
  - `C15`, `C18`, `C16`
- **Complex Distributions:**
  - `C17`, `C21`: Multimodal, distinct peaks

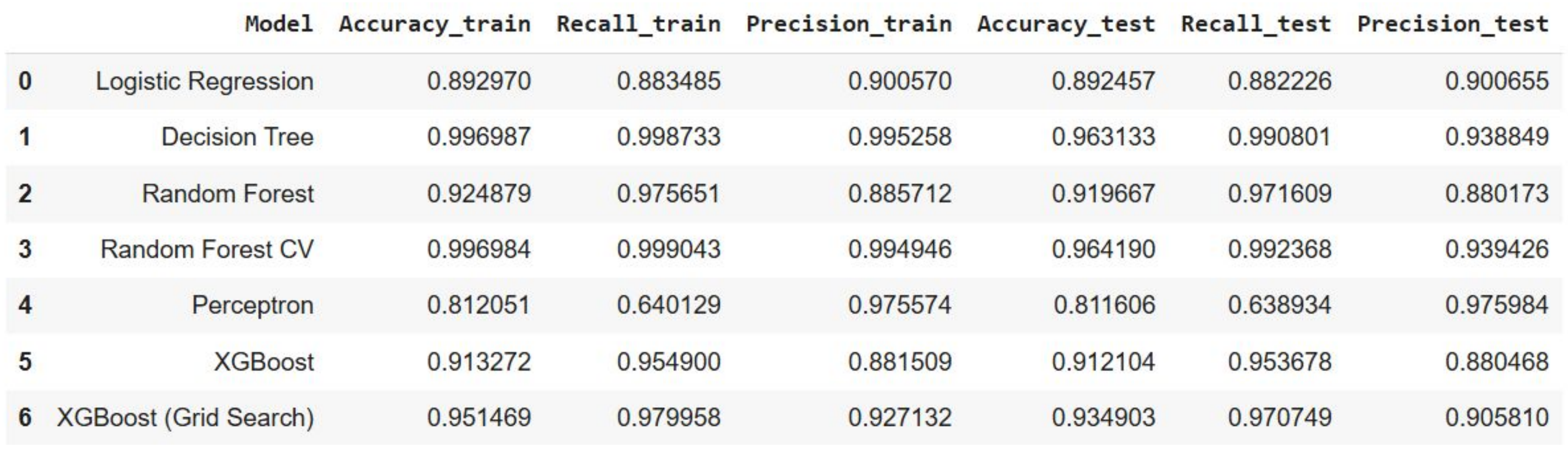
We Have Done:

- **Imbalance Handling:** Used random oversampling for class balancing; used feature selection after evaluating all models
- **Feature Engineering:** Encoded categorical features, grouped sparse categories
- **Data Transformation:** Scaled right-skewed features, handled outliers, normalized numerical features
- **Feature Selection:** Removed redundant/collinear features (`month`, `C14/C17`) after model evaluation, and dropped anomalous `C20`

Methodology

1. **Data Imbalance Handling:**
   - **Random Oversampling:** Addressed class imbalance in the target variable using random oversampling to ensure balanced classes for model training.
2. **Feature Scaling:**
   - **Min-Max Scaling:** Applied Min-Max scaling to normalize the feature values within a range of 0 to 1, improving model convergence and performance.
3. **Model Development:**
   - Developed and evaluated five models:
     - **Logistic Regression:** A baseline model for binary classification.
     - **Decision Tree:** A non-parametric model for capturing non-linear relationships.
     - **Random Forest:** An ensemble method to enhance accuracy and reduce overfitting.
     - **XGB:** A gradient boosting model implemented with grid search optimization for hyperparameters (n_estimators, learning rate, max_depth), improved performance through ensemble learning.
     - **Perceptron:** A linear classifier chosen to establish a baseline and compare against more complex models due to its ability to handle binary classification with minimal computational overhead.

Result and Analysis

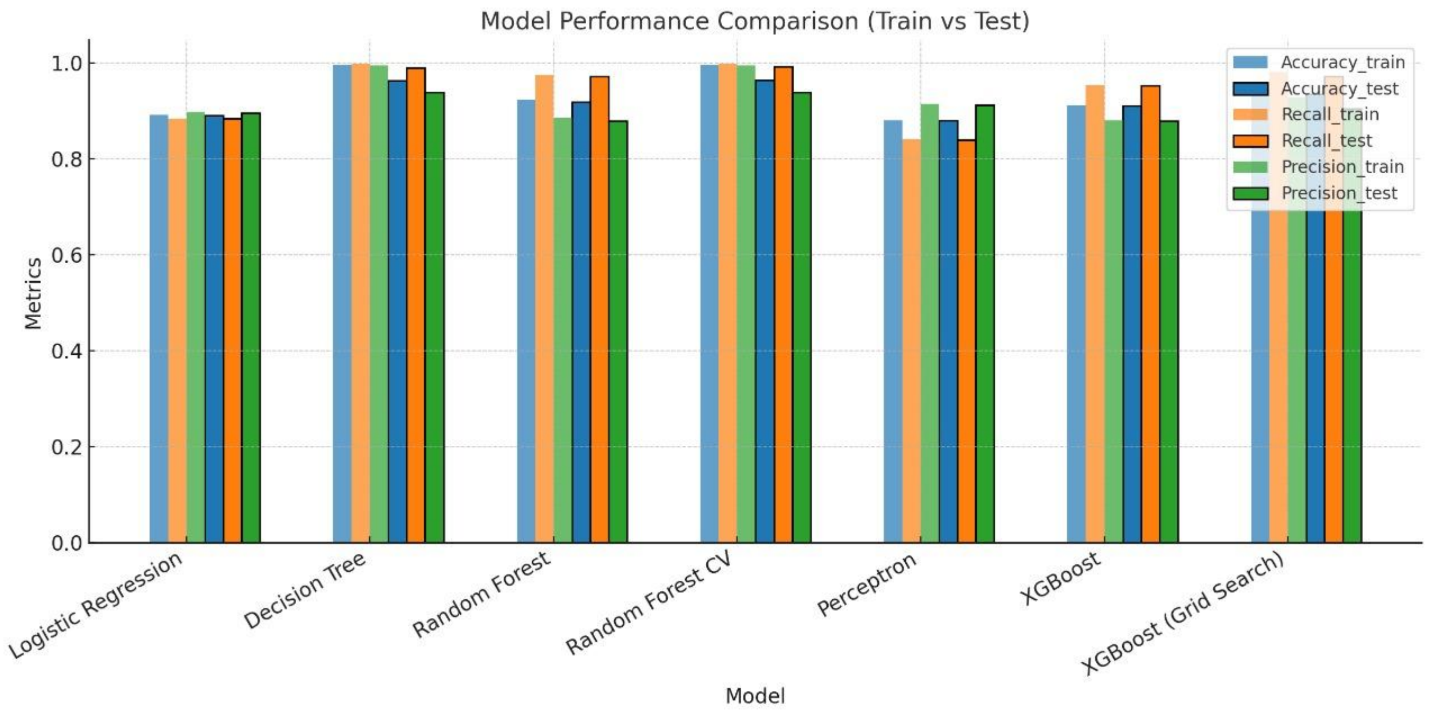


	Model	Accuracy_train	Recall_train	Precision_train	Accuracy_test	Recall_test	Precision_test
0	Logistic Regression	0.892970	0.883485	0.900570	0.892457	0.882226	0.900655
1	Decision Tree	0.996987	0.998733	0.995258	0.963133	0.990801	0.938849
2	Random Forest	0.924879	0.975651	0.885712	0.919667	0.971609	0.880173
3	Random Forest CV	0.996984	0.999043	0.994946	0.964190	0.992368	0.939426
4	Perceptron	0.812051	0.640129	0.975574	0.811606	0.638934	0.975984
5	XGBoost	0.913272	0.954900	0.881509	0.912104	0.953678	0.880468
6	XGBoost (Grid Search)	0.951469	0.979958	0.927132	0.934903	0.970749	0.905810

Observations:

- **Logistic Regression:** Shows balanced performance with decent accuracy and recall on both training and testing sets.
- **Decision Tree:** Achieved high training metrics but significantly lower accuracy and precision on the test set, indicating potential overfitting.
- **Random Forest:** Similar to the Decision Tree, it performed well on the training set but struggled on the test set, highlighting the need for further tuning.
- **Random Forest CV:** Shows excellent performance with high training metrics and maintains strong test metrics indicating good generalization with 5-fold cross-validation.
- **Perceptron:** Demonstrates lower performance compared to other models with moderate training accuracy and notably low recall, suggesting that this simple linear model struggles to capture complex patterns in the data.
- **XGBoost:** Shows balanced performance in training and test sets, with consistent recall scores.
- **XGBoost (Grid Search):** Achieves improved metrics through hyperparameter optimization, showing better generalization than the base XGBoost model and maintaining high recall on test data.

Model performance comparison



Random Forest with Cross-Validation (RF CV) is the optimal model for CTR prediction as it shows the highest test accuracy (0.964190) and recall (0.992368), maintains consistent performance across all metrics, and demonstrates reliable generalization through 5-fold cross-validation, making it most suitable for real-world applications.

Timeline

Timeline proposed at the start:

Week 1-2: Define project objectives and scope; review relevant datasets.

Week 3-4: Perform exploratory data analysis;identify key features.

Week 5-6: Develop initial models.

Week 7-8: Evaluate model performance; conduct comparative analysis with baselines.

Week 9-10: Refine models, Optimise hyperparameters

Week 11-12: Final model validation, complete project documentation

The group has diligently followed the proposed timeline. Objectives and scope have been defined. Multiple datasets were reviewed before choosing the final one. The final dataset posed problems like unbalanced number of clicks etc which were viewed and fixed using random sampling. EDA was performed and initial models were made and performance optimisation was done.

Feature engineering was implemented through target encoding of categorical variables and outlier treatment for specific features (C15, C16, C19, C21). Redundant features were removed after correlation analysis, particularly C17 due to its correlation with C14. Data normalization was performed using MinMaxScaler. The modeling phase progressed from basic models to advanced implementations, including Random Forest with cross-validation, XGBoost with grid search optimization, and a Perceptron model for baseline comparison.

Contributions

Mahi Mann: Model Development, Parameter optimization(performance), Feature Engineering and PPT

Kirti Jain: Exploratory Data Analysis, Model Performance and Report

Tanish Verma: Model Development, Feature Engineering, Model performance and Report

Divyasha Priyadarshini: Model Development, Feature Engineering, Exploratory Data Analysis and PPT