Final Report

Joshua Zhao
Brown University Data Science Institute
Website: https://github.com/anony3141mous/data1030project

Content Table

Submitted By:	**Joshua Zhao**
Submitted On:	**12/14/2024**

1. Introduction

1.1 Motivation

The goal of this project is to implement a machine learning model on the Ad Click Prediction dataset [1] (sourced from Kaggle.com) to predict whether a user will click on an online advertisement based on user demographics, user device information, and certain properties of the ad. The applications of this model include improving ad targeting, increasing click rates, and enhancing overall marketing efficiency.

1.2 Dataset Description

The Ad Click Prediction dataset consists of data on user interactions with display ads, including both user and ad features. It contains information such as user demographics, ad attributes, device type, and the context in which the ad was shown. The dataset includes labeled instances indicating whether a user clicked on the ad or not. The goal is to predict the likelihood of a user clicking on an ad based on these features.

1.3 Previous Work

There has been previous work done on this model with varied evaluation metrics and correspondingly different results. The model with the highest accuracy score of 87% utilized a decision tree, likely due to its ability to capture non-linear relationships [2]. A different model attempted various algorithms and achieved a final accuracy score of 72% from the random forest algorithm [3]. A model with a similar approach yielded a different optimal accuracy score of 75% from the XG Boost algorithm [4].

2. Exploratory Data Analysis

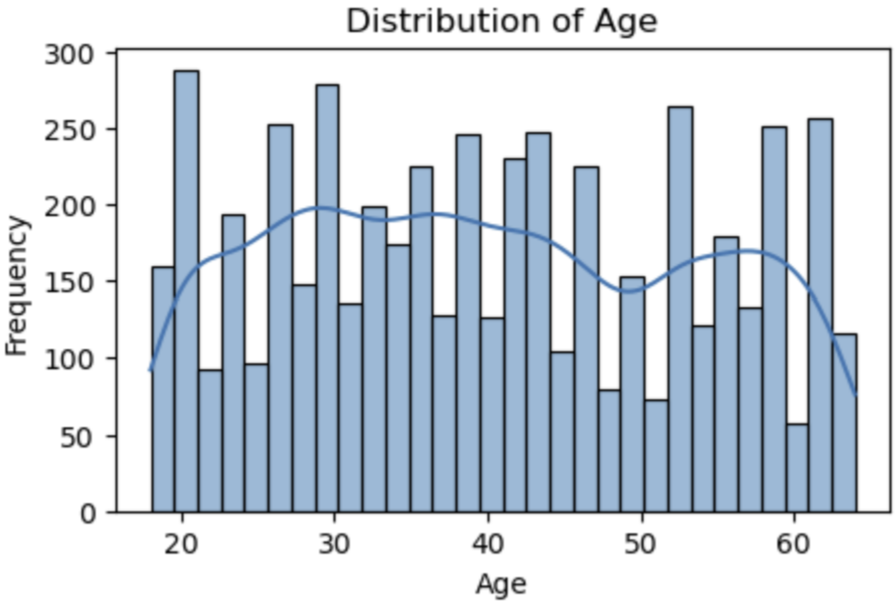
2.1 Overview

This dataset contains 10,000 rows and 9 columns, with the following attributes: id, full_name, age, gender, device_type, ad_position, browsing_history, time_of_day, and click. Click is the target variable, which measures whether a user clicks on an ad. The id and full name are excluded from the feature matrix because they do not provide meaningful information about user behavior. The feature matrix consists of the other 6 features. There are 6,500 instances where the user clicked on the ad (click = 1), and 3,500 instances where the user did not click (click = 0). The data is imbalanced, with more positive clicks than non-clicks.

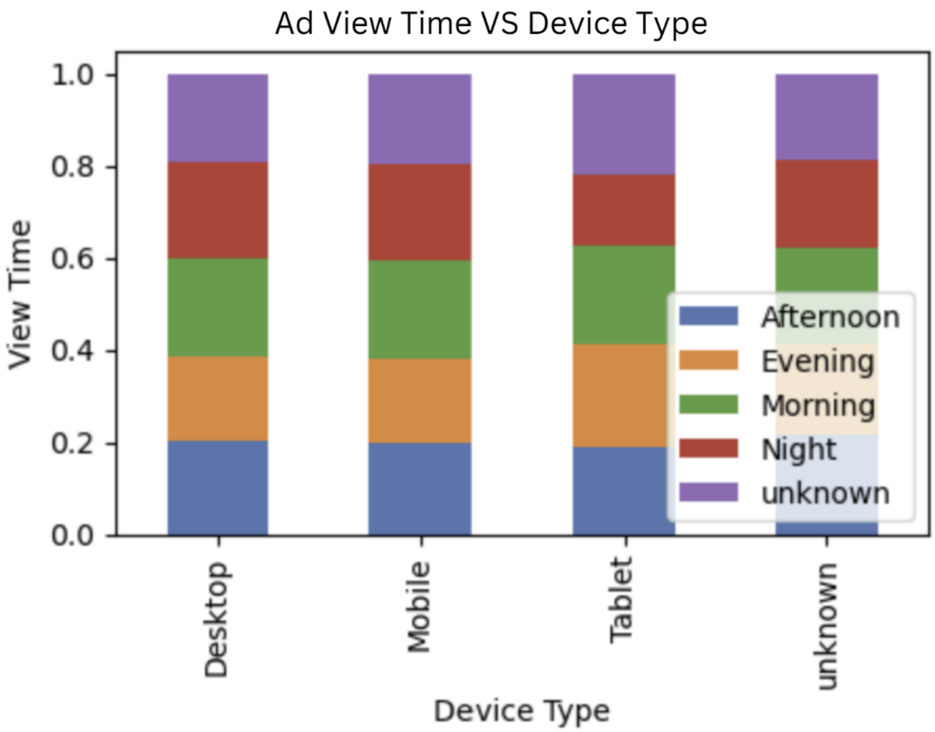
2.2 Missing Data

Each column in the feature matrix contains a significant portion of missing values. The age column has 47.66% missing values; the gender column has 46.93% missing values; The device_type, ad_position, browsing_history, and time_of_day columns each have 20% missing values. All features are treated as categorical values because of the high percentage of missing values. The missing values are treated as its own class, and additional analysis will be performed in model interpretation to filter the meaningful features that contribute to model predictions.

2.3 Visualizations



Age Distribution: the above histogram of the age column reveals a varied distribution with a slight skew, indicating a mix of younger and older users.



View Time of Ads in Relation to Device Type: the above stacked bar chart shows the distribution of time_of_day across different device_type categories. The chart illustrates how the proportion of ad views varies throughout the day for each device type (Desktop, Mobile, Tablet, and Unknown). Each bar represents a device, with segments color-coded for different times of day (Afternoon, Evening, Morning, Night, and Unknown). The normalized data ensures the bars reflect proportions rather than raw counts, offering insight into the relative share of views at each time of day per device. This allows for a comparison of user behavior based on device and time of day.

3. Methods

3.1 Splitting Strategy

The dataset is spitted into the feature matrix and target variable. Because of the decent amount of data points, the conventional 60/20/20 portions are used for training, testing, and validation.

OneHotEncoder is used for encoding, and categorical features (gender, device_type, ad_position, browsing_history, time_of_day) are identified for one-hot encoding, converting these categorical variables into a format suitable for machine learning models. All missing data values are treated as “NA”. The age column is also included to compensate for the high percentage of missing values, though it will be more optimal to be treated as a continuous feature or be treated piecewise as a categorical given that more time and/or additional computational resources are available.

3.2 ML Pipeline

3.2.1. Evaluation Metric

The F-1 score is used as the evaluation metric because it provides a balanced measure of accuracy and recall, specifically useful when the dataset is imbalanced. Since it is cheap to act on positive predictions, a precision score can be considered. Simultaneously, since it is preferable to ensure that as many true positives are identified, a recall score fits such a demand. A baseline F1 score of 0.788 is computed by predicting all instances as "clicked".

3.2.2. Model selection

The four algorithms chosen are random forest classifier (RFC), XGBoost (XGB), support vector classifier (SVC), and K-nearest neighbors (KNN). Each model was chosen based on its different properties in handling data.

RFC and XGB are chosen for their high accuracy and robustness in handling complex data with many features, making them suitable for ad click prediction where interactions and non-linearities exist. SVC is chosen for its ability to model non-linear decision boundaries, particularly useful when the decision boundary between classes is complex. KNN is included as a simple, baseline model that can provide a performance benchmark and works well for smaller or less complex datasets.

3.2.3. Uncertainty from Splitting

For each model, the data is split using 5 random states to account for variability in model performance. This helps measure how stable the model's performance is and reduce the bias introduced by a single random split.

A Stratified K-Fold is used for cross validation. Each fold in cross-validation ensures that the distribution of target classes is preserved, providing more reliable estimates of performance metrics by averaging over several validation splits.

To measure the uncertainty from splitting, the standard deviation of both validation scores and test scores is computed across all random states. This provides an estimate of how much the model's performance varies with different splits.

3.2.4 Hyperparameter Tuning

SVC: **C** and **Gamma**: they directly influence the model's ability to generalize. Choosing optimal values for these parameters is essential to avoid both underfitting (too simple model) and overfitting (too complex model). Testing these different values helps to strike the right balance for the given data.

KNN: **n_neighbors**: N_neighbors determines the number of neighbors to consider when making a prediction for a given data point and directly influences the model's performance.

RFC: **max_depth** and **max_features**: max_depth controls the maximum depth of each decision tree and restricts how deep the trees can grow, thus limiting their complexity to balance overfitting and underfitting. max_features controls the number of features that each decision tree can consider when splitting a node and specifies the fraction of the total number of features to randomly sample for each tree. Tuning max_features balances biases and variance and improves the generalizability of the results.

XGB: **reg_alpha**, **reg_lambda**, and **max_depth**: reg_alpha and reg_lambda hyperparameters control the L1 and L2 regularizaions, respectively, and tuning them balances overfitting and underfitting. Similar to RFC, max_depth in XGB max_depth controls the maximum depth of the decision trees, and a tuned max_depth improves model generalizability.

Below is a table of the range of values of hyperparameters tuned for each model.

Model	Hyperparameters Tuned and Values Chosen
SVC	C: 0.01, 0.1, 1, 10, 100 Gamma: 0.001, 0.01, 0.1, 1, 10
KNN	n_neighbors: 3, 5, 7, 10, 15, 20
RFC	max_depth: 1, 3, 10, 30, 100 max_features: 0.25, 0.5, 0.75, 1
XGB	Alpha: 0.01, 0.1, 1 Lambda: 0.01, 0.1, 1 max_depth: 1, 3, 10, 30, 50

4. Results

4.1 F1 Score from each Model

A table of the F1 scores from each model and its standard deviation is provided below.

Model	F1 Score	Standard Deviation
SVC	0.791	0.005
KNN	0.792	0.007
RFC	0.802	0.006
XGB	0.841	0.016

Note: baseline score is 0.788

XGB is the most predictive model with an F1 score of 0.841, which is 3.3 standard deviations from the baseline score.

4.2 Feature Importance and Discussion

To assess feature importance, all features are permuted with nr_runs = 7. The top three features that contribute to the predictions are device_type_Mobile, gender_Male, and device_type_Tablet with SHAP values 0.37, 0.21, 0.19, respectively.

In the context of the ad-click prediction modeling, It is not surprising that Mobile device type is the most important feature, as mobile devices are often the most used for browsing and clicking on ads. The widespread use of smartphones for online activity, especially on social media, makes this a natural top feature. The high positive SHAP value infers that mobile users are more likely to click on ads compared to users on other devices. This conclusion aligns with general trends in digital marketing and the increasing mobile-first behavior of consumers.

The gender male is an unexpected feature that contributes a lot to predictions. Due to the sensitive and debated nature of the gender variable, especially in the context of advertising, the model showing males are more likely to click ads may be a reflection of the data distribution of the sample dataset and may also be a user behavioral pattern. This feature may also infer that there exists potential bias in the model.

The device type Tablet is also unexpected as people tend to use tablets less compared to desktop or mobile. However, the dataset from Kaggle was created a few years ago, and people then might had a different preference in digital device. If tablet users are predicted to click more often, it could suggest that the content or format of ads is better suited to tablets. This could also be a sign that certain user segments are under-targeted by advertisers, assuming the tablet usage group isn't fully exploited in marketing strategies. It's also worth noting that, compared to mobile, tablet devices may have larger screens, which might be better for ad engagement in some cases.

5. Outlooks

5.1 Alternative Splitting Strategy

With additional time and computation resources, certain improvements can be made for this model. Most importantly, as discussed above in the splitting section, the feature Age could be considered a continuous variable as opposed to a categorical variable, the action of which was taken to facilitate the processing of missing data. Changing this variable property leads to major changes in the ML algorithm, and it was concluded that by the due date of this project, a compile-able model could not be constructed. However, the outlines of such a model are provided here. The most significant change will be the addition of a reduced feature section to handle missing data in addition to using XGBoost. The choice of the four algorithms mentioned in section 3.2.2 will remain the same, as these models will also handle the adjusted datasets. Correspondingly, different sets of hyperparameters will be chosen to exhaust the computational capabilities of the device available for training. The limitation of computation leads to the next potential improvement.

5.2 Widening the Range of Hyperparameters

From training results based on the range of hyperparameters in section 3.2.4, overfitting was never observed, meaning a wider range of hyperparameters, especially larger hyperparameter values, could have been included given more computational power. This situation is a promising indication that the model could handle more complex configurations without the risk of overfitting, suggesting that the chosen hyperparameter space was not restrictive enough to exploit the full capacity of the models. Expanding the range of hyperparameters, particularly for models like Random Forest or XGBoost, will explore values for parameters such as max_depth, learning_rate, n_estimators, and regularization terms (e.g., reg_alpha and reg_lambda). These larger values could help the model learn more complex patterns and potentially increase predictive accuracy.

6. References

[1] Marius, C. (2024, September 7). *AD click prediction dataset*. Kaggle. https://www.kaggle.com/datasets/marius2303/ad-click-prediction-dataset/

[2] Dogukan. (2024, September 14th). *Ad Click Prediction | Max 87%.* Kaggle.

https://www.kaggle.com/code/dogukantabak/ad-click-prediction-max-87

[3] AhmedAshraf299. (2024, September 11). *Ad_click_dataset using randomforest ACC 72*. Kaggle. https://www.kaggle.com/code/ahmedashraf299/ad-click-dataset-using-randomforest-acc-72

[4] NouranmAhmoudd. (2024, September 10). *Ad prediction using all models acuuracy 75%.* Kaggle. https://www.kaggle.com/code/nouranmahmoudd/ad-prediction-using-all-models-acuuracy-75