AI7101 Project Report:

New User Engagement on Zindi

1. Problem Description

The Zindi platform, a prominent online community for data scientists in Africa, faces the challenge of new user engagement. Many new users may register but not actively participate or contribute to the platform. This lack of engagement can lead to a stagnant user base and reduced overall platform activity. From a business perspective, higher engagement often correlates with increased data science competition participation, solution submissions, and community interaction, all of which contribute to Zindi's growth and value proposition.

Our machine learning model aims to predict new user engagement and identify factors that influence it. By understanding which users are at risk of disengaging, Zindi can implement targeted interventions, such as personalized recommendations, introductory tutorials, or mentorship programs, to encourage greater participation. This can lead to a more vibrant community, increased data science talent development, and ultimately, a more successful platform. The ML model can help reduce the cost of acquiring new users by retaining existing ones and increase the quality of the user base by fostering active participation.

1.1 Data Description and Guidelines

The Zindi platform provides a dataset for this challenge, encompassing various aspects of user activity and demographic information. Understanding the structure and content of these tables is crucial for effective feature engineering and model development.

1.1.1 Provided Data Tables

The dataset includes several interconnected tables, each offering unique insights:

- **Users Table:** Contains information about registered users, such as registration date, country, and some other masked features. This table will be key to understanding the user base and merge other tables based on the user IDs from this one.
- **User Activity:** Containing different activities done by each user and their respective time where they happened; each row represents an activity done by a specific user. This table was used as the main indication of the activity of a user.
- **Competitions Participation Table:** Details about competitions that each user joined and the number of submissions each user did in this competition along with the dates of joining this competition.
- **Competitions, Discussions, Blogs, and Jobs Tables:** These tables serve as look-up tables providing metadata about each instance. For example, the Competitions table includes details like start and end dates, while the Discussions table contains information on who created the discussion, creation dates. Similarly, the Blogs, Comments and Jobs tables offer metadata related to their respective content.

1.1.2 Data Guidelines and Considerations

The following guidelines with detailed insights about the data, provided by Zindi in a separate CSV, are important to consider:

- **Test Set Definition:** The model will be evaluated on a test set comprising all users whose accounts were created in month 4. Crucially, there will be no activity data available for these users in month 5, which is a key consideration for feature engineering and target variable creation.
- **Chronological Month Data:** All monthly columns within the dataset are presented in chronological order. However, it is important to note that "January" does not necessarily correspond to "month 1" in this chronological sequence. And it's important to note that the 12th month is followed by the first month although they are passed in data as both being in the same year. So the right order of months would be as follows: “11,12,1,2,3,4,5”
- **Target Variable Creation:** The dataset does not include a pre-defined target column. A target variable must be engineered from the available data before model building. This target should specifically indicate user activity in the second month following each user's signup.
- **Evaluation Metric:** Error metric for this problem is F1-score

2. Dataset Preparation

This section outlines the steps taken to prepare the dataset for machine learning, including encoding, handling missing values, merging respective tables and feature engineering. All the dataset preparation was done in the *merge.py* file in our repo. The specific columns chosen to be merged or used from each table were decided on from our understanding of the problem’s goal and the thorough description given on the platform for the problem.

2.1 Encoding Categorical Variables

Converting categorical features into a numerical format suitable for machine learning models. One-hot encoding was used in all of our encodings as it prevents the model from assuming an ordinal relationship between categories.

- **Activity Grouping:** Different activities were encoded from the User Activity table by summing the occurrences of each activity per user. Some activity names indicated similar actions in different contexts (e.g., `comp_id_1234` and other `comp_id` variations were grouped to count all competitions a user engaged with). Similar grouping was applied to activities related to jobs, blogs, and badges. However, distinct activities that did not require grouping were counted individually.
- **Competition Submissions:** Similar to activity grouping, the number of successful submissions for each user in different competitions was encoded. These submissions were often represented by ranking names, which were treated as categorical features. To avoid implying ordinality where none existed, one-hot encoding was applied to these submission rankings, and the counts of each submission type were aggregated per user.

2.2 Merging Respective Tables

The various tables (Users, User Activity, Competitions Participation, etc.) will be merged based on common identifiers, primarily user IDs. This step is essential to consolidate all relevant information into a single, comprehensive dataset for model training. Careful consideration will be given to the type of merge (e.g., inner, left, outer) to ensure that all necessary data is retained and no critical information is lost. All of the merging was conditioned on that any activity or feature was done in the same month as the user's account creation month.

- The Activity Grouping and Competition Submissions from the previous point was merged with the Users table.
- Comments and Discussions were counted per user and merged as well with the Users table.

2.3 Handling Missing Values

Missing values in the dataset can negatively impact model performance. Various strategies can be employed to address them. In the data provided only one column had missing values which was Country_ID per user; around half of its entries were Nans (47%), thus, we dropped the column.

2.4 Feature Generation

Based on domain intuition and initial data understanding, new features were engineered from existing ones:

- **Chronological Month Mapping:** The provided chronological month sequence (11, 12, 1, 2, 3, 4, 5) was mapped to calendar months based on the inference that two consecutive months with 31 days must correspond to July and August. This led to the following mapping: original month 11 was mapped to April, 12 to May, 1 to June, 2 to July, 3 to August, 4 to September, and 5 to October. This detailed mapping is crucial for accurate temporal analysis and feature engineering.
- **Activity Counts and Date Statistics:** From the User Activity table, three key features were engineered. These features were developed with the primary goal of monitoring user activity since its the main focus of the problem. Incorporating additional features that describe user activity in various ways could provide further benefits :

    - **Count of Activities:** The total number of activities performed by a user within the month their account was created.
    - **Standard Deviation of Activity Dates:** The standard deviation of the dates on which activities were performed, providing an indication of the spread or span of activity days within the creation month.
    - **Number of Unique Activity Days:** The actual number of distinct days within the account creation month on which a user performed at least one activity.

2.5 Target Variable Preprocessing

The target variable, representing user engagement, was not explicitly provided in the problem's dataset. We engineered this target column based on domain intuition and problem understanding. A value of '1' was assigned if a user performed any activity from the User Activity table in the month immediately following their account creation month, and '0' if no activity was recorded during that subsequent month.

2.6 Data Splitting

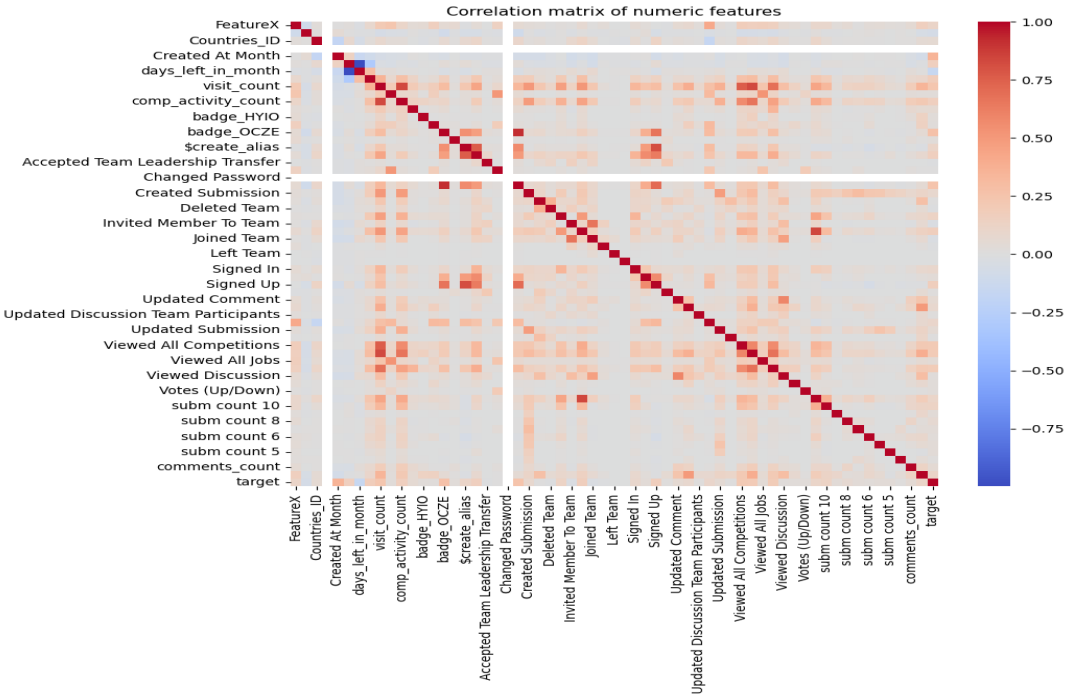
The prepared dataset will be split into training and testing sets. As defined in Section 1.1.2, users whose accounts were created in the 4th month (September, based on our chronological month mapping) will constitute the test set, and no activity data will be available for them in the 5th month. This approach aligns with the competition's evaluation criteria. The remaining data will be used for training.

3. Exploratory Data Analysis (EDA) and Visualization

Exploratory Data Analysis (EDA) will be performed using Seaborn to visualize the data and uncover meaningful patterns. This step is crucial for understanding the data distribution, identifying outliers, and gaining insights that can guide further preprocessing and model selection. Additional analyses were conducted to determine the amount of missing values and the distribution of data. Visualization, including linear and non-linear correlations, was performed to check the direct relationship with the target variable, aid in feature selection, and inform which model will work best with the data.

Findings from EDA:

1) Initial investigations using Pearson correlation during the Exploratory Data Analysis did not reveal significant linear relationships between the features and the target variable. This suggested that a simple linear model might not fully capture the underlying patterns of user engagement.



2) To address this, we employed *mutual_info_classif* from scikit-learn to evaluate feature importance. This method, based on mutual information, estimates the extent to which knowing a feature's value reduces uncertainty about the target label. Mutual information is particularly valuable as it can capture both linear and non-linear dependencies, providing a more comprehensive understanding of feature relevance when linear correlations are weak.This approach, however, did not reveal any strong indications of relationships with highest Mutual information with one of the features being 0.06 which is considered very low, suggesting that the features individually possess little or no univariate signal. In other words, each feature, when considered in isolation, did not demonstrate a strong direct correlation with the target label.

3) Further visual analysis was conducted using Seaborn. The distribution of each feature was plotted for both target classes (engaged and not engaged, represented by '1' and '0' respectively). Box plots were also generated for each feature, allowing for a clear visualization of their spread, central tendency, and the presence of outliers across the two target categories. These visualizations helped to identify features that, while not linearly correlated, showed distinct distributions between engaged and disengaged users, thus indicating their potential importance for the model. Some of them is: visit_count, Confirmed Email, badges and competition counts

4) We checked for missing values and found that the `Country_ID` column had approximately 47% missing values, leading to its removal. Additionally, the `Year` column exhibited zero variance (all values were '1'), rendering it uninformative, and was therefore also dropped.

4. Model Building and Evaluation

The model building process was carefully designed to handle the **imbalanced, binary classification problem** of predicting whether a new user will become engaged in the Zindi platform. The workflow consisted of (1) designing a robust cross-validation procedure, (2) selecting evaluation metrics motivated by the data distribution and business needs, (3) choosing appropriate machine learning models, and (4) tuning hyperparameters to maximize predictive quality.

4.1 Cross-Validation Procedure

The dataset exhibited **class imbalance**, with many more disengaged users (label 0) than engaged users (label 1). Standard random splits could bias performance estimates toward the majority class. To address this, we used **Stratified K-Fold Cross-Validation** with 5 folds. This ensured each fold preserved the same proportion of engaged vs. disengaged users, producing a fairer and more reliable estimate of generalization performance.

4.2 Choice of Quality Measure

The competition objective is to identify at-risk disengaged users early. In this context:

- **Accuracy** would be misleading because predicting all users as disengaged would achieve high accuracy but fail the business goal.
- **Precision** (for the engaged class) reflects how many users flagged as engaged actually are, which matters for targeted interventions.
- **Recall** reflects how many engaged users were successfully identified, which is equally important to avoid missing potential contributors.
- **F1-score** balances precision and recall, making it the most suitable measure.

Accordingly, **per-class precision, recall, and F1-score** were reported for each fold, and the **macro-averaged F1** across classes was used as the main measure of model quality.

4.3 Model Selection and Hyperparameter Tuning

The training pipeline consisted of **several stages of model training and stacking**, progressively refining features and combining model outputs:

1. **Logistic Regression with Standardization**
    - The first stage applied a pipeline of **StandardScaler + Logistic Regression** (class_weight="balanced", max_iter=10000).
    - Out-of-fold predictions were generated and stored as a new feature (lg) for stacking.
    - This provided a simple, interpretable baseline model to capture linear relationships in the data.
2. **CatBoost Stacking Model**
    - A **CatBoost Classifier** (n_estimators=1000, learning_rate≈0.01, depth=7) was then trained, using balanced class weights.
    - Out-of-fold predictions were saved as another stacking feature (cat).
    - Additionally, a weighted average of logistic regression (lg) and CatBoost (cat) probabilities was computed as a meta-feature (weighted_avg).
3. **Data Filtering Step**
    - To refine the dataset, we computed the difference between the true label and CatBoost predictions (diff).
    - Samples with diff <= -0.85 were excluded, effectively removing cases where the model was highly confident but incorrect.
    - This filtering step reduced noise and improved the reliability of the final model.
4. **Final CatBoost Classifier**
    - A deeper and longer-trained CatBoost model (n_estimators=2000, learning_rate≈0.001, depth=9) was trained on the enriched feature set.
    - Early stopping (100 rounds) prevented overfitting, while class-balanced weights ensured fairness across categories.
    - This final model formed the backbone of our solution, combining linear signals (from logistic regression) with complex non-linear interactions (from CatBoost).

4.5 Estimation of Predictive Quality

Using the prepared cross-validation pipeline, we estimated the quality of prediction as follows:

- Each fold reported **per-class precision, recall, and F1-scores** for training and validation sets.
- Metrics were averaged across folds to obtain **reliable estimates of generalization performance**.
- The final CatBoost model consistently achieved strong validation F1-scores, demonstrating balanced predictive power across engaged and disengaged classes.

This procedure ensured that our model choice, quality measure, and hyperparameter settings were fully aligned with the data characteristics and the practical requirements of the engagement prediction problem.

5. Results Analysis and Conclusion

5.1 Analysis of Results

The final CatBoost model, trained within our stacking and filtering pipeline, achieved the following performance across cross-validation folds:

- **Validation Performance**:

	Precision	Recall	F1
Class 0 (Disengaged)	0.9173	0.9004	0.9088
Class 1 (Engaged)	0.7488	0.7851	0.7665

- **Training Performance**:

	Precision	Recall	F1
Class 0 (Disengaged)	0.9260	0.9062	0.9160
Class 1 (Engaged)	0.7652	0.8083	0.7861

Key Findings

1. **Balanced Performance Across Classes**:
   The model performs very well for the majority (disengaged) class, with precision and recall above 0.90. Importantly, performance on the minority (engaged) class remains strong, with F1 around 0.77. This balance indicates the model is not simply defaulting to majority predictions.

2. **Minority Class Detection**:
   Detecting engaged users is more challenging, as expected given class imbalance and noisier signals of early activity. Still, recall of 0.7851 suggests the model successfully identifies nearly 80% of users who will engage, which is highly valuable for targeted interventions.

3. **Generalization**:
   The close alignment of training and validation F1-scores across both classes shows that the model generalizes well, with no strong signs of overfitting. The filtering step (removing high-confidence mispredictions) likely contributed to this robustness.

4. **Interpretability and Practical Use**:
   Logistic Regression features contribute linear interpretability, while CatBoost captures complex non-linear interactions. This makes the pipeline not only effective but also partially interpretable, which is useful for real-world adoption by Zindi administrators.

5.2 Applicability and Impact

The model directly addresses the problem posed at the beginning: **predicting new user engagement to support targeted retention strategies**. By reliably identifying users who are at risk of disengagement (Class 0) and those likely to become active (Class 1), Zindi can:

- **Target early interventions** such as personalized recommendations, welcome tutorials, or mentorship assignments for users flagged as likely engaged.
- **Optimize retention efforts**, avoiding wasted resources on users who are unlikely to respond, while focusing support on those with strong engagement potential.
- **Strengthen community growth** by improving the conversion rate from new registrations to active contributors.

Based on the metrics, the model is expected to capture around **3 out of 4 engaged users correctly (recall ≈ 0.79)**, while maintaining strong reliability in disengaged predictions. In practical terms, this means the platform could successfully intervene with the majority of potentially active users, significantly reducing early-stage churn.

5.3 Overall Conclusion

Our machine learning pipeline — combining Logistic Regression, CatBoost stacking, and robust cross-validation — achieved a **macro F1-score of 0.84 and weighted F1-score of 0.87**, demonstrating both balance and reliability across classes.

The model effectively solves the problem of predicting new user engagement on Zindi. It provides a data-driven mechanism for early intervention, which could **increase active user participation, reduce churn, and enhance the overall vibrancy of the Zindi community**. Future improvements could focus on richer feature engineering, temporal modeling, or further hyperparameter tuning to push recall for the minority (engaged) class even higher.