**Overview**

- **Task:** build a music recommendation system based on user behaviors and song features.
- **We built models to predict the chances of a user listening to a song repetitively after the first observable listening event within a time window, providing a binary prediction.**

**Data Set**

- 3,076 users, 113,750 of songs, and over 700 thousands records on user-user interactions.
- Text data only, no audio features.
- Balanced Class labels (50.9% Positive, 49.1% Negative).
- 70% as training set, 15% as validation set and 15% as test set.
- **Challenge:** Large Amount of Categorical Features (with high cardinality)

**Feature Engineering**

- **Data Cleaning:** Inspect and remove a small portion of rows missing important features or even label
- **Feature Generation:** Explicitly inject time index as feature
- **K-Modes Clustering:** Cluster Categorical Data, Generate Song Features and UI features.
- **MCA:** Multiple Correspondence Analysis to develop feature association between user and songs.
- **One-Hot Encoder:** For Categorical Features, using Label encoder and one hot encoder.
- **Standardization:** For numerical Data, standardized to have mean 0 and variance 1.

**Methodology**

**Diverse Gradient Boosting Decision Tree**

- **XGBoost:**
  - Sparsity-aware algorithm for sparse data
  - Weighted quantile sketch for approximate tree learning

- **CatBoost:**
  - Ordered boosting for Categorical Features

- **LightGBM:**
  - Gradient-based One-Side Sampling: Exclude a significant proportion of data instances with small gradients, and only use the rest to capture the most important information gain.
  - Exclusive Feature Bundling: bundle mutually exclusive features to reduce the number of features.
  - Efficient Leaf-Wise Search

**Other Explorations**

1. Linear SVM
2. Logistic Regression
3. Decision Tree
4. Fully-Connected Neural Network

**Fine Tuning**

1. Randomized Search Cross Validation

**Results**

From our experiments, we can see that **boosting models fit our dataset best.** This is probably because our data is full of categorical features, and boosting methods can handle the categorical features as well as the sparsity of the data very well. Our best single model is achieved by XGBoost with the ROC-AUC Score of 0.859. LightGBM also has a comparable result of 0.853. Catboost doesn't perform as well as the other 2 methods, with the score of 0.779.

Neural Network doesn't perform well on our dataset due to the sparsity. After adding the clustering features, we observe a significant improvement of 10%. Dropout Layer also improves the performance by 2%.

**Discussion**

Different Boosting Models have different weights for features due to the algorithm emphasis.

For LightGBM, the top three most important features are the index of user’s listening record that includes timestamp information, msno (user id), and user registration time. This intuitively suggests that user identity decides if they listen the song once again. We then see that artist name ranking one of the top five features, way more important than genre, language, lyricist, etc. This is again intuitive in the sense that artist name carries more specific "genre" information.

**Future work**

- **Modeling**
  - Improved Random forest
  - Stacking with intentionally diversified models

- **Data & Feature Engineering**
  - Categorical feature cardinality reduction or conversion to numerical values - Data transformation for heavy tail features
  - Outlier detection and aggressive removal for untypical user that fails multiple model predictions