User Churn prediction for Music Box

The Lifelong Learning Platform of Silicon Valley

Bittiger DS501 Capstone Project
Meng Li

Outline

- Data preprocessing and Cleaning
- Feature Extraction
- Model Fitting and Tuning
- Performance Analysis and Visualization

BITTIGER

Data Preprocessing

- Use shell scripts to extract and clean the data
- Remove some noise patterns by regex such as control characters and IP addresses (113.140.48.146)TM
- Use pandas and sqlite to save the data locally
- Drop redundant feature paid_flag, which is always 0
- Delete incomplete records where song_id or song_name is missing, delete records with negative playtime
- Group the data by song_id and song_name and identify the correct song_length by the mode
- Remove songs with extremely large song_length and join the songs data with the original data

Feature Extraction

- Group the dates from 2017-03-30 to 2017-05-12 to four time windows, each of which is 11-day
- For play data, extract the play frequency, accumulated play time, average play percentage
- For download data, extract the download frequency, number of unique songs downloaded, number of unique singer
- Also calculate play recency and download recency as the days between last play/download date and the first day of last window
- Keep the indicator of device

Churn Definition and Further Cleaning

- Remove records with play frequency that exceeds the 0.999 quantile
- Calculate the average song length times 0.999 quantile of play frequency as the cutoff for large outliers in play time
- For download data, remove records with download frequency that exceeds the 0.999 quantile
- Join download data and play data, fill NA recency with -999; fill other NA values with 0
- Remove inactive users in the first three windows, the define a churn user to be the one that did not play or download a single song in the last time window

Snapshot of training data

	freq_w1	play_time_w1	avg_play_pct_w1	freq_w2	play_time_w2	avg_play_pct_w2	freq_w3	play_time_w3	avg_play_pct_w3	play_recency	...	down_uniq_sin
0	45.0	9027.0	0.949343	114.0	24372.0	0.900190	109.0	24790.0	0.972819	1	...	
1	4.0	184611.0	1.000000	0.0	0.0	0.000000	0.0	0.0	0.000000	33	...	
2	0.0	0.0	0.000000	2.0	102.0	0.325743	0.0	0.0	0.000000	18	...	
3	354.0	13547.0	0.148254	283.0	14781.0	0.202751	333.0	15397.0	0.179196	1	...	
4	13.0	780.0	0.239631	0.0	0.0	0.000000	10.0	1665.0	0.616586	9	...	

- The churn data frame have 572171 records in total with 21 features
- Churn rate is 66.26%
- Use 80% of data as training data, 20% of data as test data

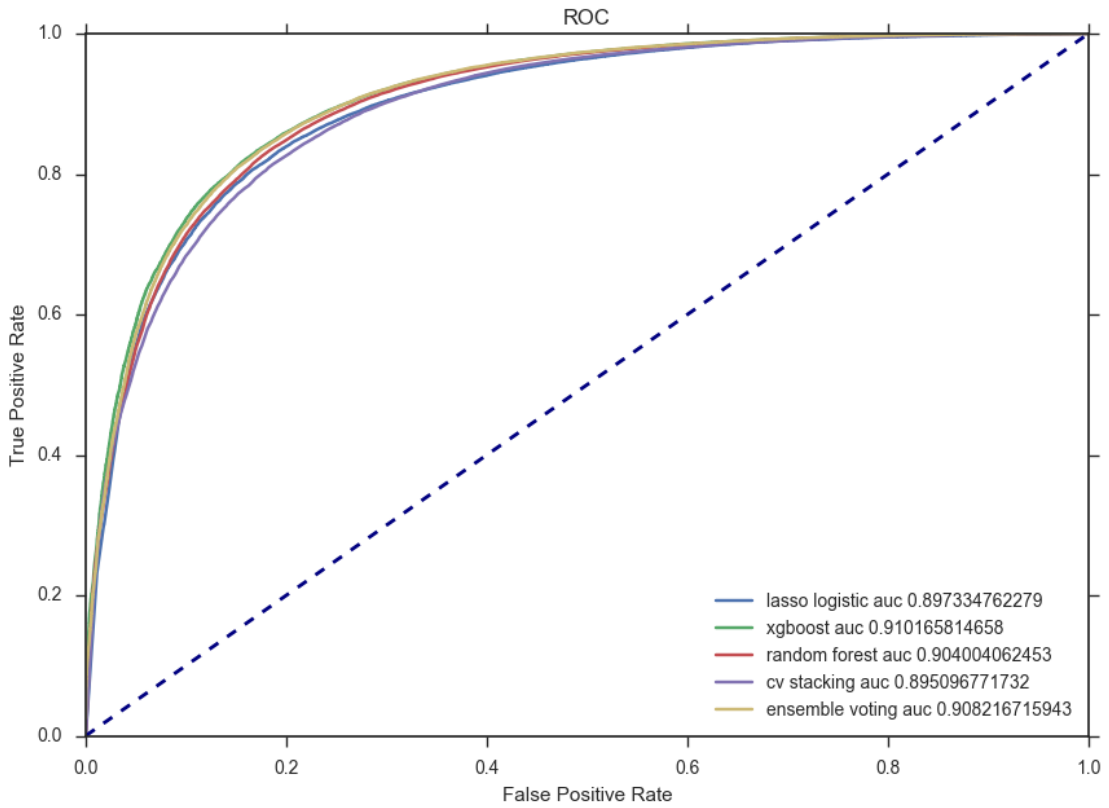
Model results grid search/oob_error

Model	Random Forest	XGBoost	Logistic Regression	SVM(SGD)
optimal parameters setting	max_feature='sqrt' n_estimators=300	learning_rate=0.1 n_estimators=186 max_depth=7 min_child_weight=5 gamma=0 subsample=0.9 colsample_bytree=0.8	alpha=1(lasso) lambda=0.02993902	NA
Test AUC score	0.904000972985	0.910165814658	0.897334762279	<=0.8

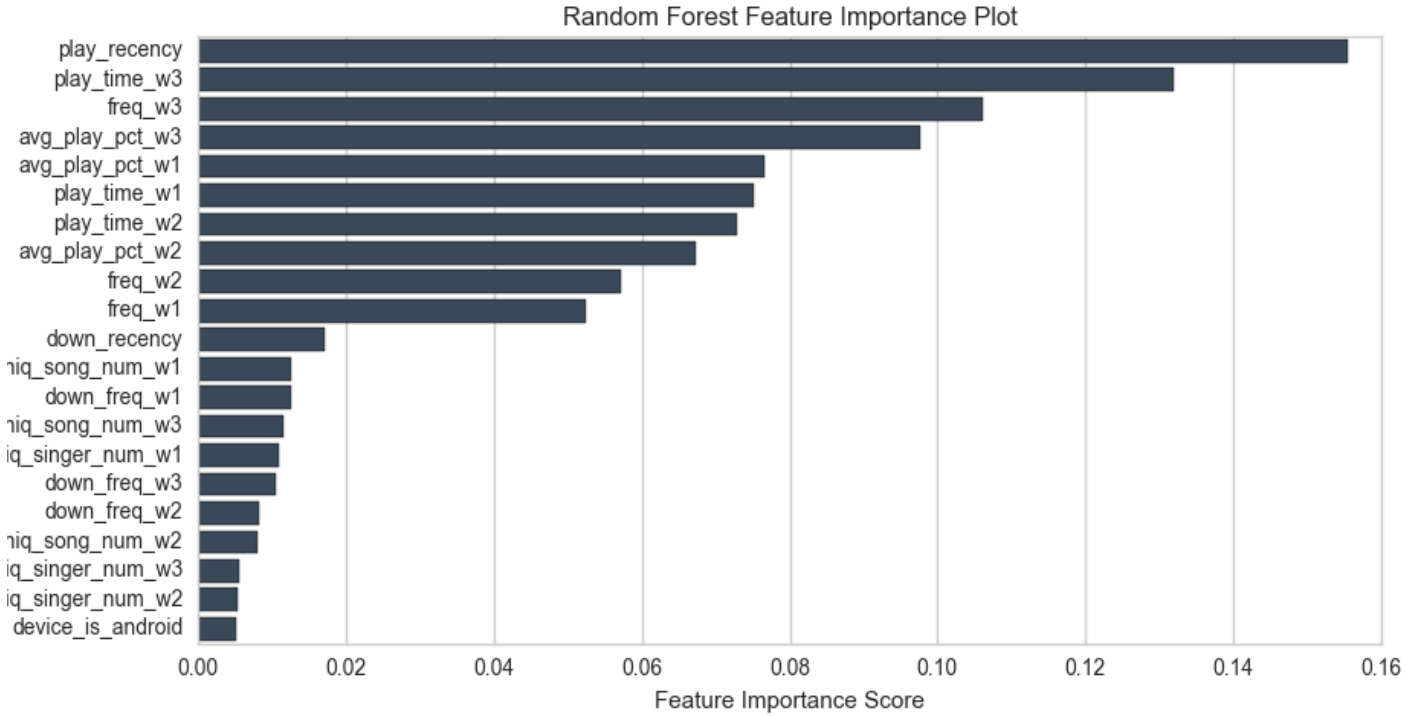
Model results (stacking)

model	best auc score
random forest	0.904000972985
xgboost	0.910165814658
lasso logistic	0.897334762279
Ensemble voting (soft)	0.90821643158
CV stacking	0.89511394605

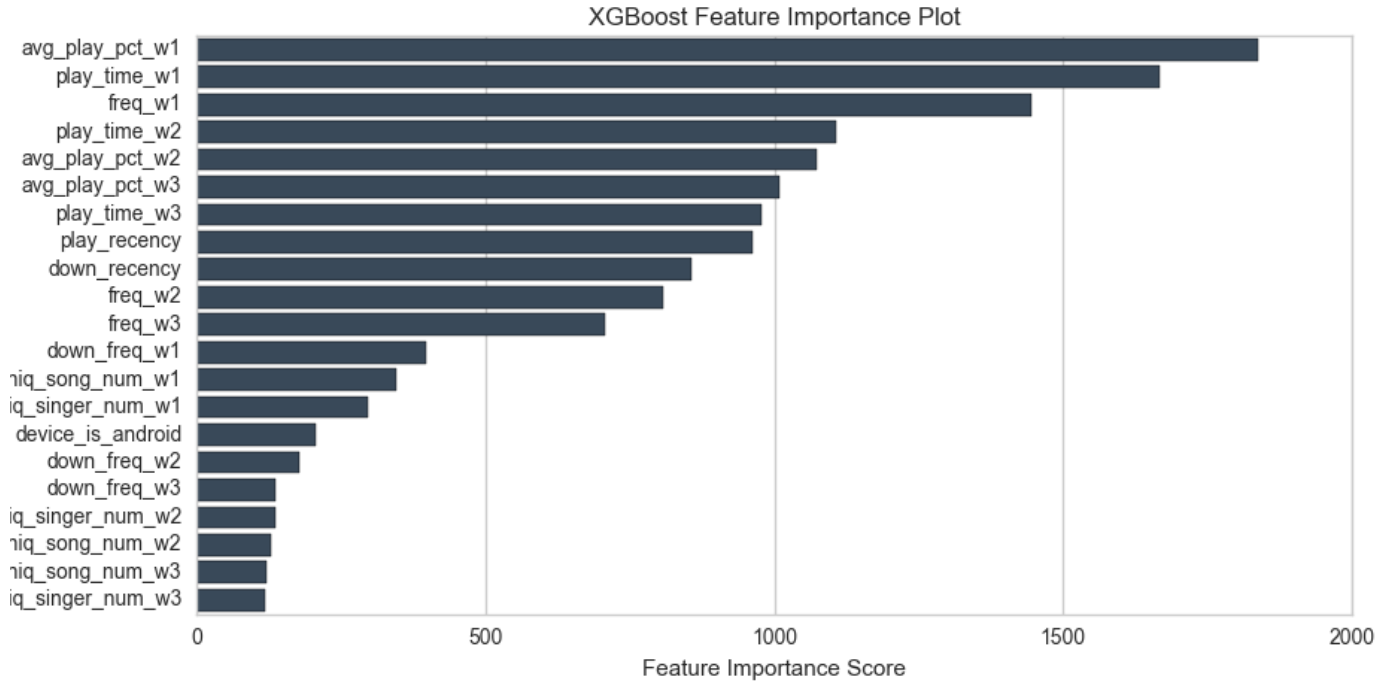
ROC Curve


ROC
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
lasso logistic auc 0.897334762279
xgboost auc 0.910165814658
random forest auc 0.904004062453
cv stacking auc 0.895096771732
ensemble voting auc 0.908216715943

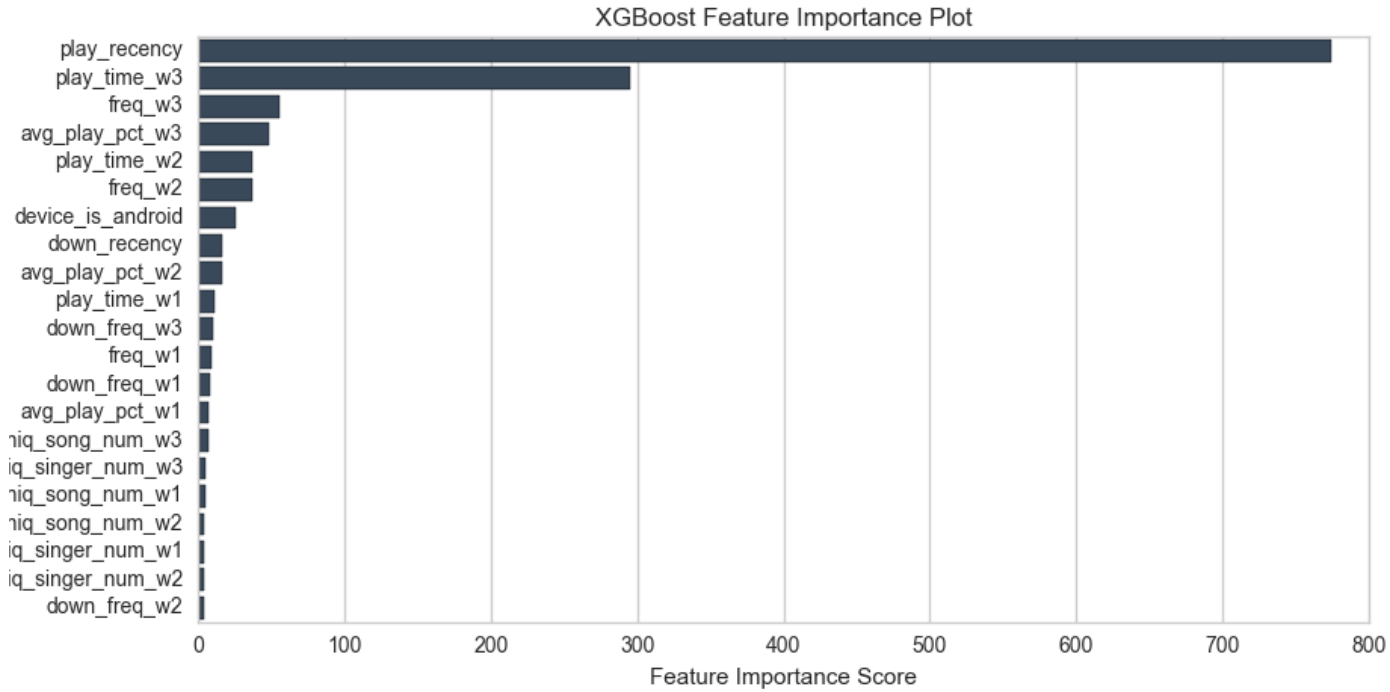
Feature Analysis Random Forest



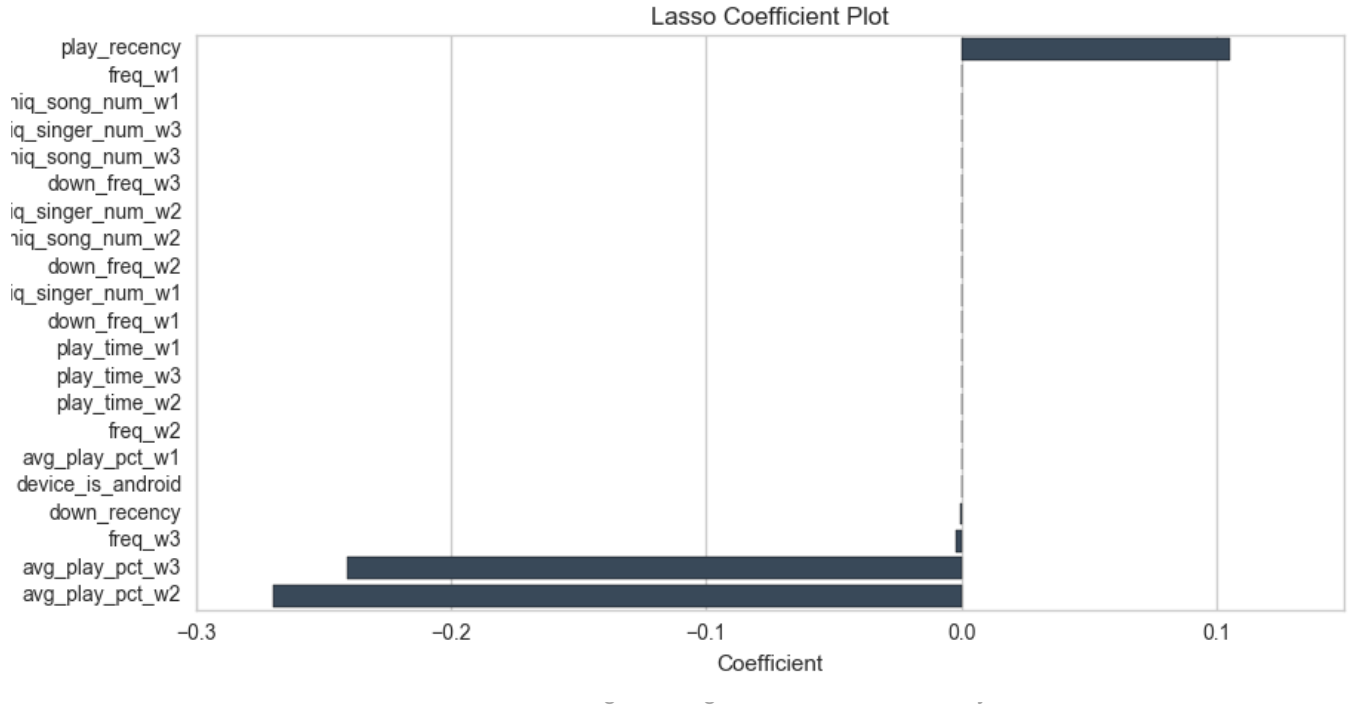
Feature Analysis XGBoost (weight)



Feature Analysis XGBoost (gain)



Feature Analysis Lasso Logistic Regression



Questions?

BITTIGER

The Lifelong Learning Platform of Silicon Valley

Thank you for listening!

BITTIGER

The Lifelong Learning Platform of Silicon Valley