What makes an Android App popular?

Presented by Xu Han, Data Science ScM

Dec 4, 2019

Github repo

Recap

- While people spend more time using apps, it also becomes a challenge to develop a popular app. The data was downloaded from Kaggle.
- In this project, the goal is to predict app popularity (number of installs) from features such as category, size, current version, rating, paid or free etc.
- Number of installs: '<10k', '<500k', '<5 million', '> 5 million'. Classification.
- Free apps, certain categories (entertainment, food & drinks, video player), apps that make localized versions based on devices (Size Varies, Current Version Varies = 1) usually have higher downloads.
- Number of reviews is strongly correlated with installs.

CV Pipeline General

- Split data in stratified manner into other and test (20%)
- Split other into 5 stratified folds
- Continuous features transformer: Impute the ones with missing values using iterative imputer (no imputer when using XGB), then standard scaler or minmax scaler
- Categorical features transformer: One-hot or ordinal encoder
- Set up **parameter grid**, **preprocesser pipe** and **estimator pipe** to feed into grid search
- Use grid to fit X_other, y_other, then calculate accuracy score for test set
- Call each model 8 times (random state = (i+1)*42 for each call) to calculate average accuracy and standard deviation

CV Pipeline: Random Forest

- Parameter grid
    1. max_depth: 30, 35, 40, 45, 50, 55, 60
    2. min_samples_split: 2, 3, 4, 5
    3. random_state: controlled by function call index i
- Preprocesser: continuous transformer and categorical transformer
- Estimator: RandomForest Classifier

CV Pipeline: SVC

- Parameter grid
    1. C: 1e+3, 1e+4, 1e+5, 1e+6, 1e+7
    2. gamma: 1e-4, 1e-1, 1e+2
    3. PCA n_components = 25
- Preprocesser: same
- Estimator: PCA first, then SVC so that SVC can run faster

CV Pipeline: k-NN

- Parameter grid
    1. n_neighbors: 20, 25, 30, 35, 40, 45
- Preprocesser: same
- Estimator: KNeighborsClassifier

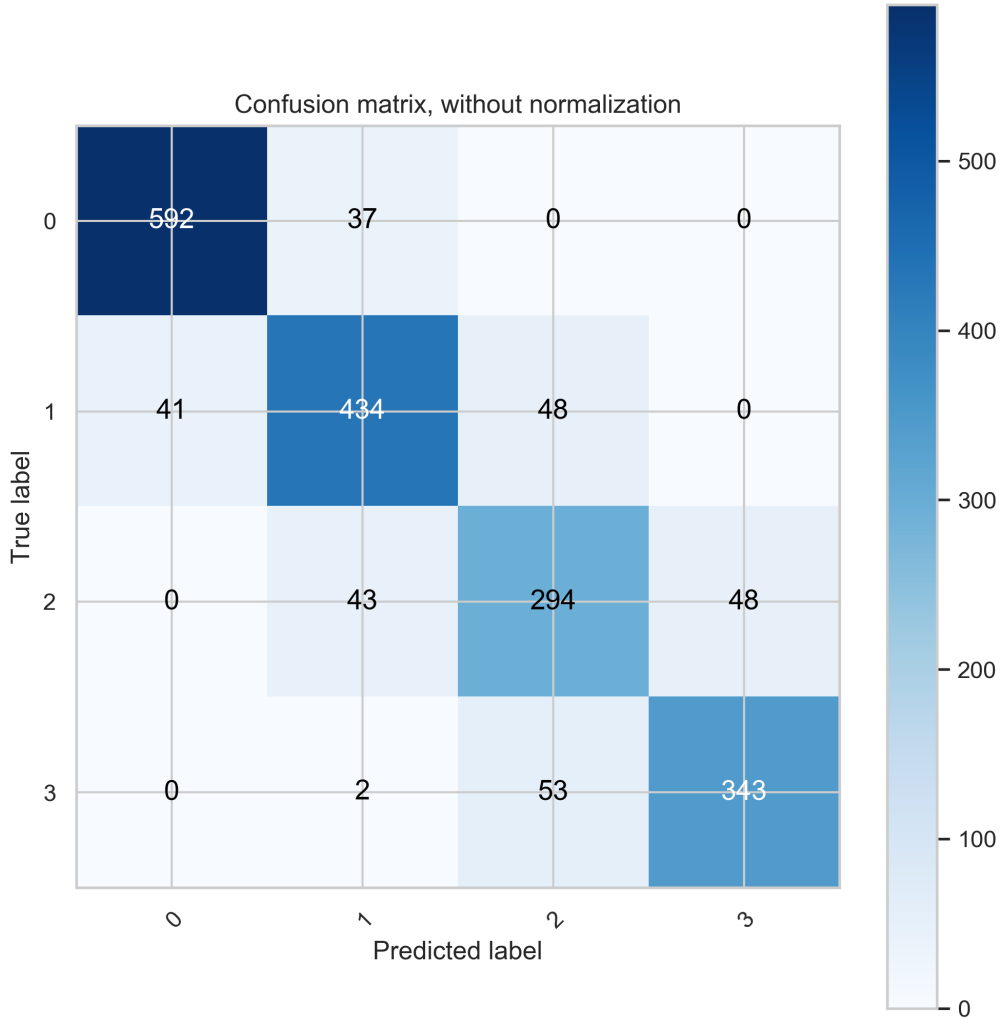
CV Pipeline: XGB

- Parameter grid
  1. learning_rate: 0.03
  2. max_depth: 20, 30, 40, 50
  3. n_estimator: 100
  4. colsample_bytree: 0.75
  5. subsample: 0.68
  6. random state: controlled by function call index i
- Preprocesser: no imputer was used for continuous features, otherwise the same
- Estimator: XGBClassifier

Results

Baseline accuracy: 0.325, percentage of most popular class (download < 10k)

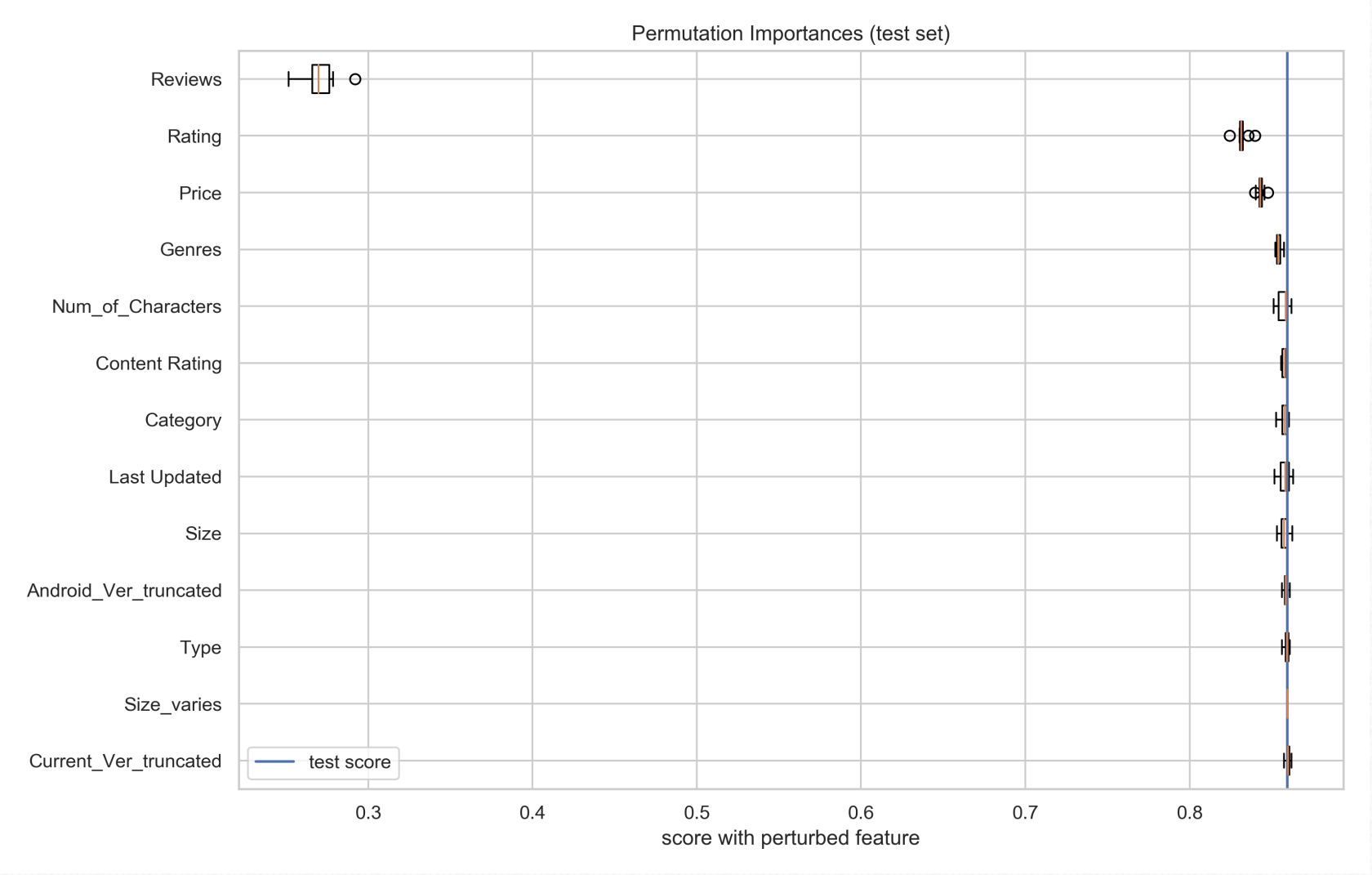
- Random Forest: 0.844 +/- 0.005, max_depth 35-60, min sample split 3-5
- SVC: 0.672 +/- 0.014, C 1e+6 and 1e+7, gamma 1e-4
- k-NN: 0.501 +/- 0.012, n_neighbors 20-45
- XGB: 0.858 +/- 0.002, max_depth 20-50



	precision	recall	f1-score	support
0	0.935229	0.941176	0.938193	629
1	0.841085	0.829828	0.835419	523
2	0.744304	0.763636	0.753846	385
3	0.877238	0.861809	0.869455	398
accuracy			0.859432	1935
macro avg	0.849464	0.849112	0.849228	1935
weighted avg	0.859868	0.859432	0.859598	1935

- From XGB model
- Class 0 (downloads < 10k) has the highest f1-score
- No test point has a predicted class that is more than one level away from real class

Results: feature importance



- Get more reviews
- Aim for higher ratings
- Make it available for free (or cheap)
- Choose popular genres

Outlook

- Weak spot
  1. Number of reviews is not really controllable by developers
  2. I expected Current Version Varies and Size_Varies to have bigger impact
  3. SVC classifiers have very large Cs

Improvements

  1. Try to collect app size information and try reduced feature model
  2. Run XGBoost with imputed data, and impute using different seeds
  2. Run SVC without using PCA, tune again
  3. Try other models such as neural net, logistic regression