Google Play App

Predict the rating on Google Play Store Apps for the Android market with machine learning

Xiongfeng Wang, Brown DSI, 10/16/2020
https://github.com/XiongfengWang/1030project

Introduction

- Predict the rating for Google Play Store Apps:
- Rating affect App's success and visibility.
- Good prediction on new Apps is beneficial.
- Regression: rating scales from 1 to 5 with decile level between each integer.
- Dataset: Kaggle https://www.kaggle.com/lava18/google-play-store-apps

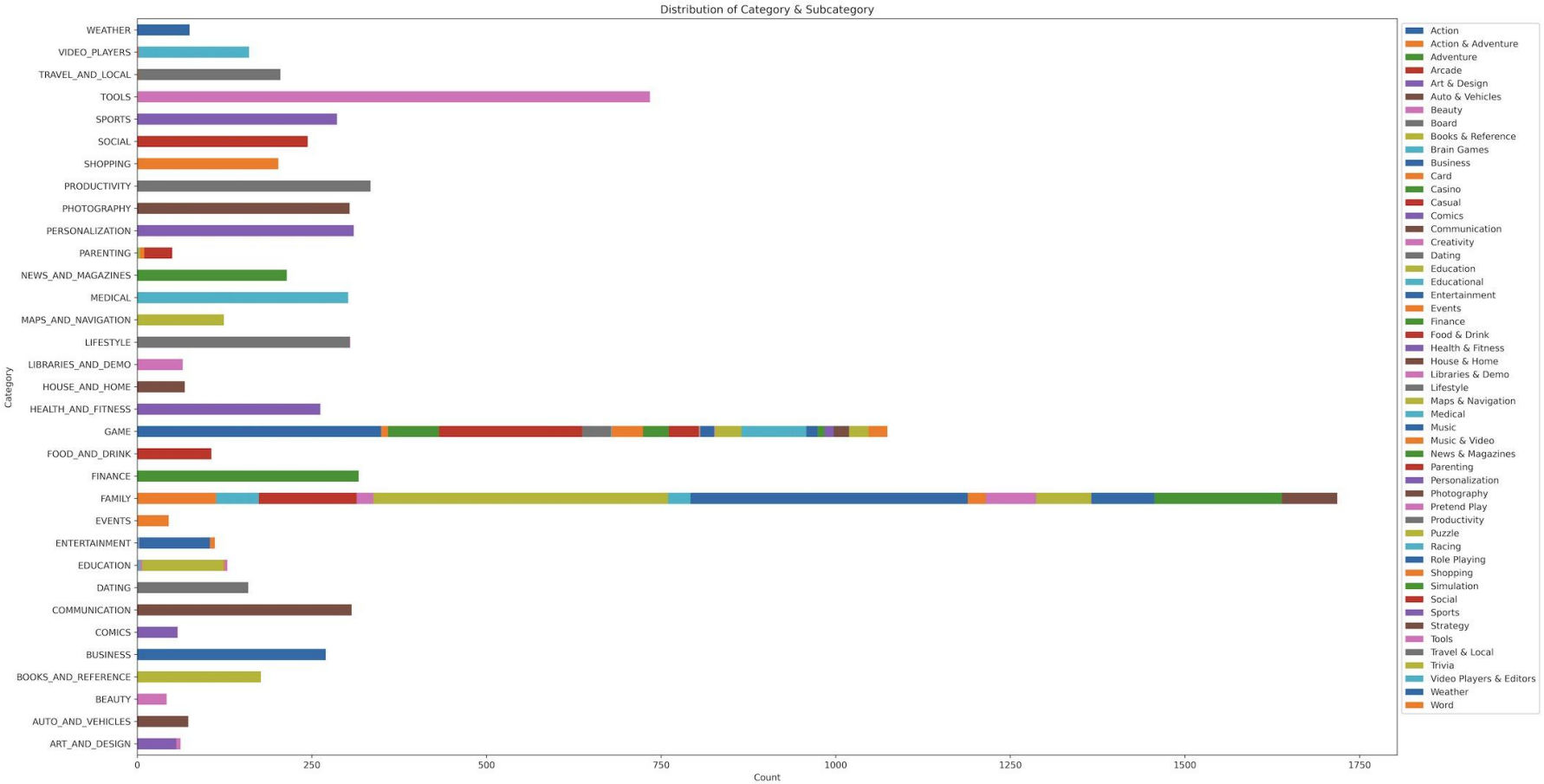
Dataset Over View

- 10,358 samples, 13 columns
- Drop duplicate samples
- Drop samples missing rating
- Drop ‘almost duplicate’
- 8,211 samples, 10 features (exclude App, Current Version, Ratings)
- missing values
- Group structure
- Imbalance

EDA

Features	Reviews	Numerical	‘Varies with device’ kB, MB to Byte
	Price	Numerical	Remove $ sign
	Last Updated	Numerical	Transfer to Days

Group Structure: Category and Genres



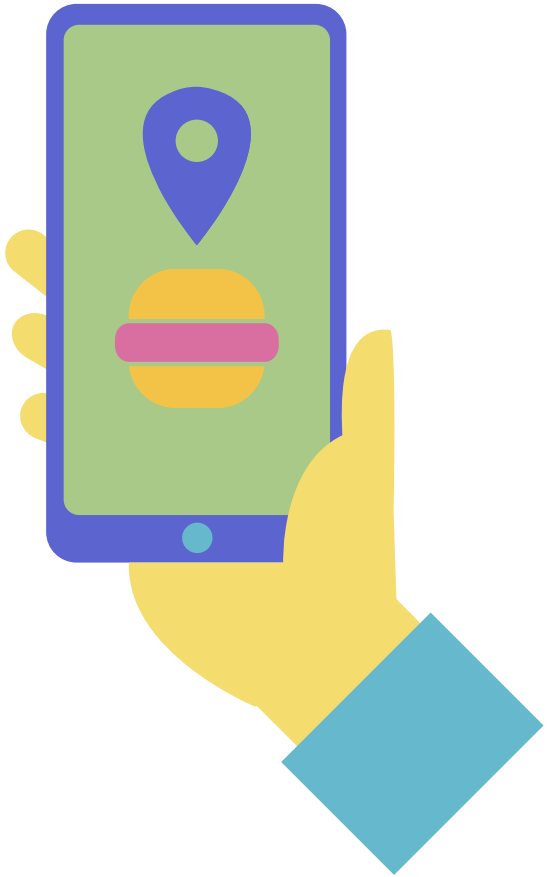
Preprocessing

Group Structure: not I.I.D.

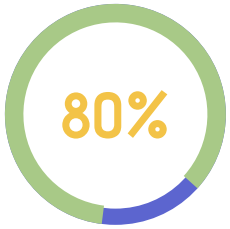
Not time series

Imbalance

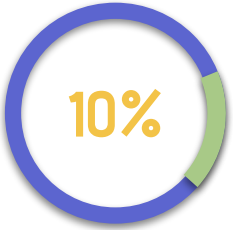
GroupShuffleSplit



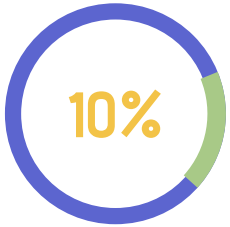
How to split



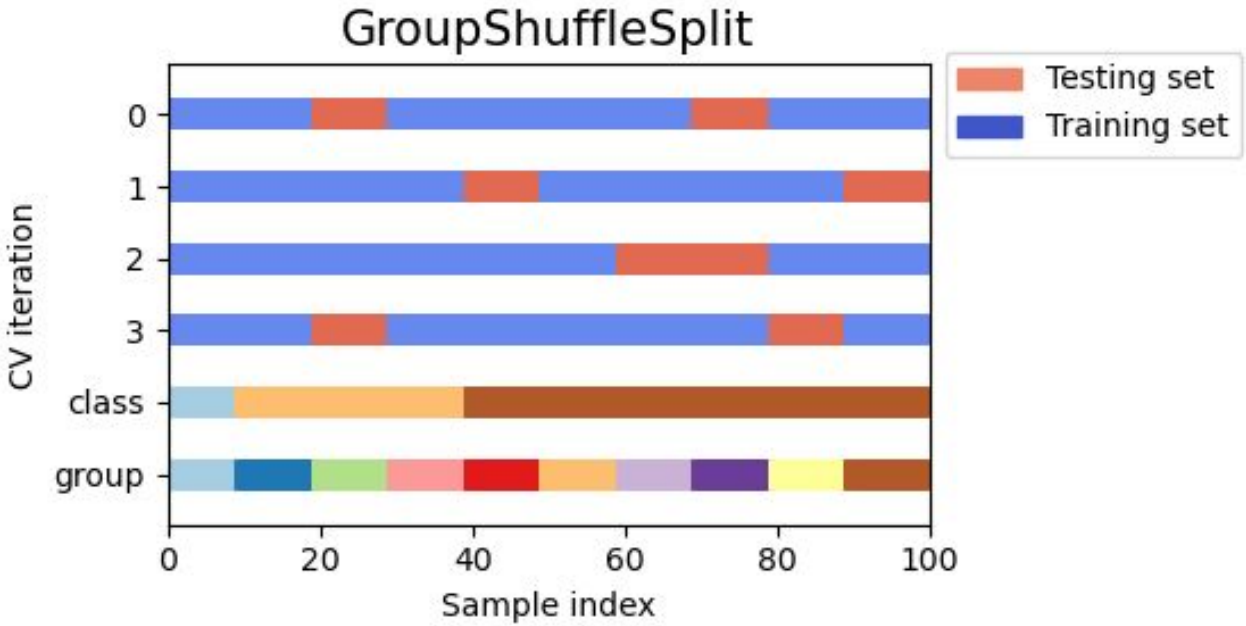
Train



Validation



Test



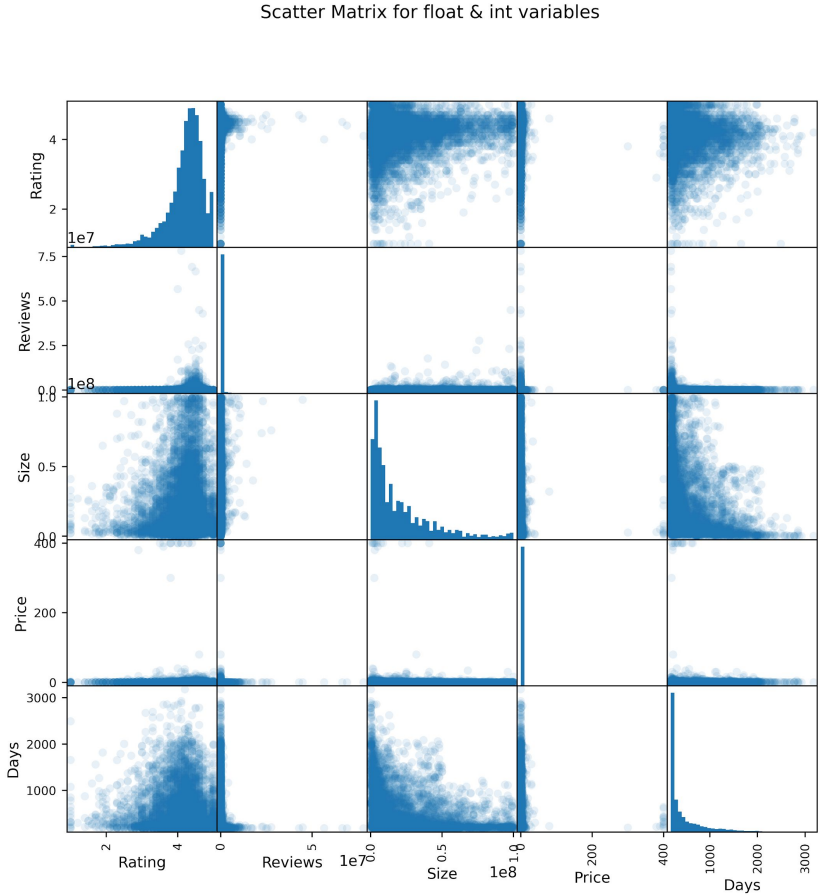
Missing Data

4 in Android Ver:

Assign ‘None’

1171 in Size (MAR):

Iterative imputation



Encoding

	Variable	Classification	Encoder
Key	App		
Features	Category	Categorical	OneHotEncoder
	Reviews	Numerical	StandardScaler
	Size	Numerical	StandardScaler
	Installs	Categorical	OrdinalEncoder
	Type	Categorical	OneHotEncoder
	Price	Numerical	StandardScaler
	Content Rating	Categorical	OneHotEncoder

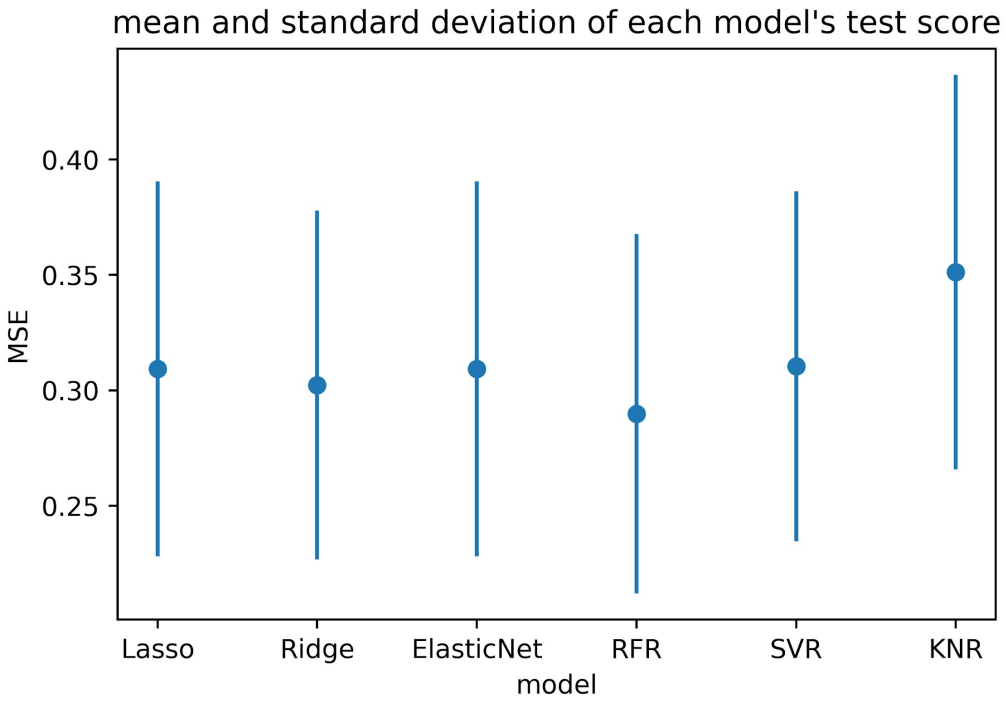
	Variable	Classification	Encoder
Features	Genres	Categorical	OneHotEncoder
	Last Updated	Numerical	
	Current Ver	Categorical	
	Android Ver	Categorical	OneHotEncoder
	Subcategory	Categorical	
	Days	Numerical	StandardScaler
Target	Rating	Numerical	

Cross Validate

Model	Hyperparameter(s)	Values to try
Lasso	alpha	[1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1]
Ridge	alpha	[1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1]
Elastic Net	alpha	[1e-10, 1e-5, 1e-3, 1e-1]
	l1_ratio	[0.1, 0.3, 0.5]
Random Forest	max_features	[1, 5, 10, 30, 50, 100]
	max_depth	[0.1, 0.2, 0.3, 0.4, 0.5, 0.6]
SVM	gamma	np.logspace(-5, 5, 11)
	C	np.logspace(-5, 5, 11)
KNeighbor	n_neighbors	np.linspace(10, 200, 20)
	weights	['uniform', 'distance']

Returns

Baseline model:
linear regression
MSE = 2.65686
RMSE = 1.63



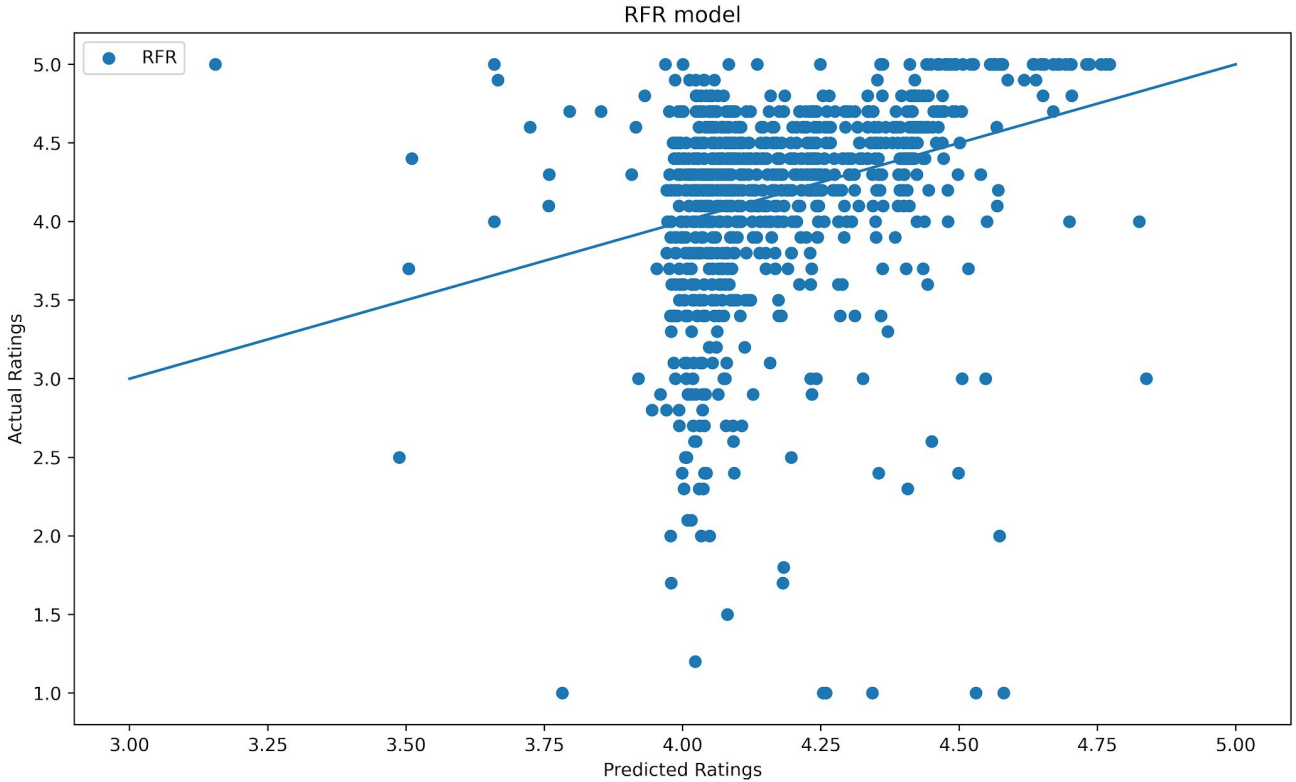
Returns

Random Forest Regression gives the best model with diverse hyperparameters.

np.mean(test_scores) = 0.28, rooted = 0.53

[RandomForestRegressor(max_depth=50, max_features=0.2),
RandomForestRegressor(max_depth=10, max_features=0.6),
RandomForestRegressor(max_depth=50, max_features=0.5),
RandomForestRegressor(max_depth=10, max_features=0.6),
RandomForestRegressor(max_depth=10, max_features=0.6),
RandomForestRegressor(max_depth=50, max_features=0.2),
RandomForestRegressor(max_depth=10, max_features=0.5),
RandomForestRegressor(max_depth=30, max_features=0.3),
RandomForestRegressor(max_depth=30, max_features=0.2),
RandomForestRegressor(max_depth=10, max_features=0.5)]

Returns


RFR model
RFR
Actual Ratings
Predicted Ratings
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
3.00
3.25
3.50
3.75
4.00
4.25
4.50
4.75
5.00

Outlooks

Remove samples with missing value (14%) instead of imputation

Try XGBoost

One App may have several samples with different name (‘Basket Manager 2016 Free)’, we can even the weight for these samples.

Try different Group Structure

Q&A

Thanks!

CREDITS: This presentation template was created by **Slidesgo**, including icons by **Flaticon**, and infographics & images by **Freepik**.

Please keep this slide for attribution.