Predicting Realty Prices in Russian Housing Market

Team Jade

Bo Lian, Jade Le-Cascarino, Daniel Rim, Choutine Zhou

Approach

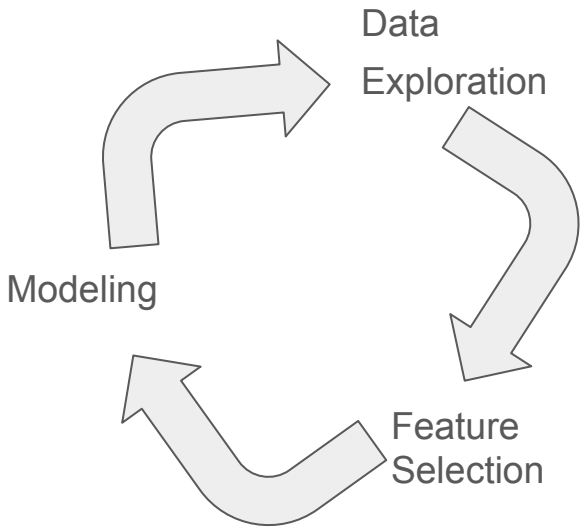
Kaggle Competition : Sberbank Russian Housing Market

Phase 1 Data Exploration

- Major relationships
- Data transformation
  - log(price)
  - Unit price by square feet
- Missing values
- Collinearity

Phase 2 Feature Selection

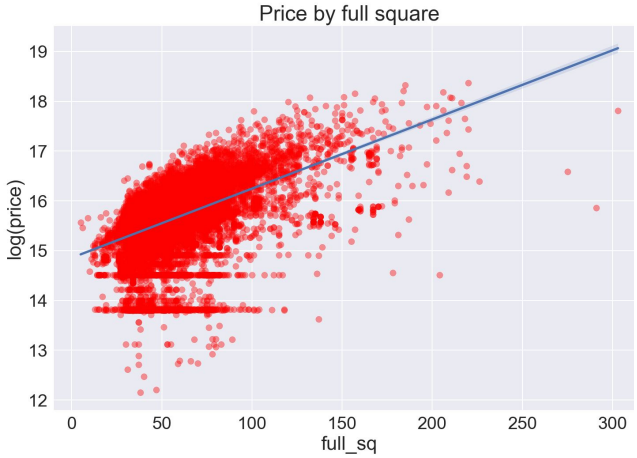
Phase 3 Modeling

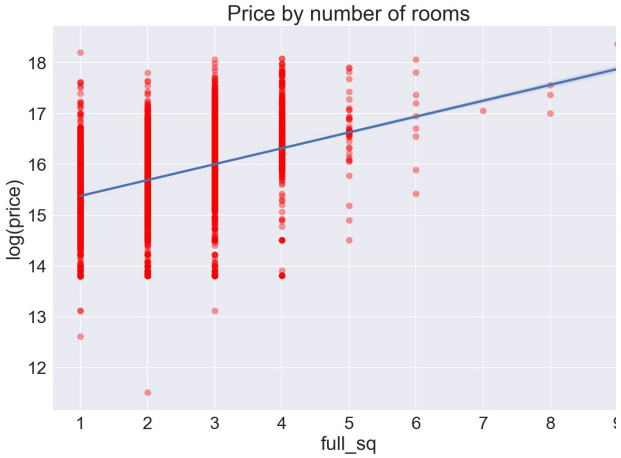


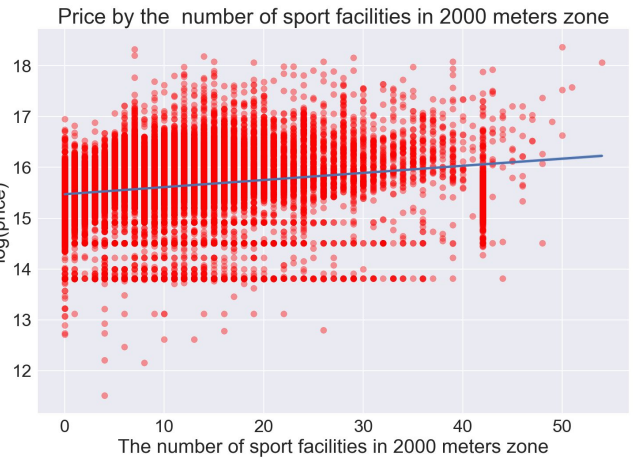
Results

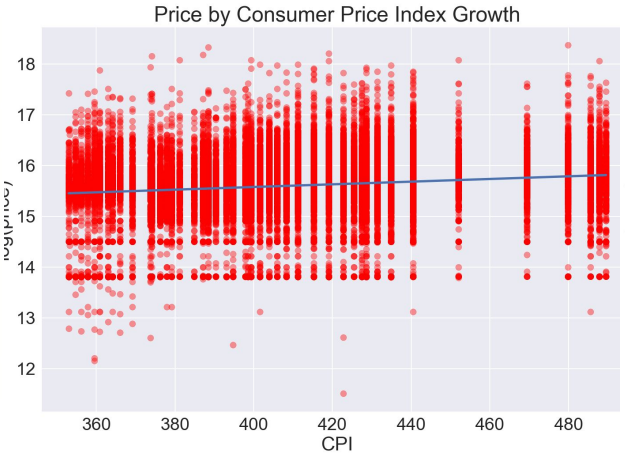
Kaggle Ranking: Top 24% at score: 0.314 without model stacking or ensembling

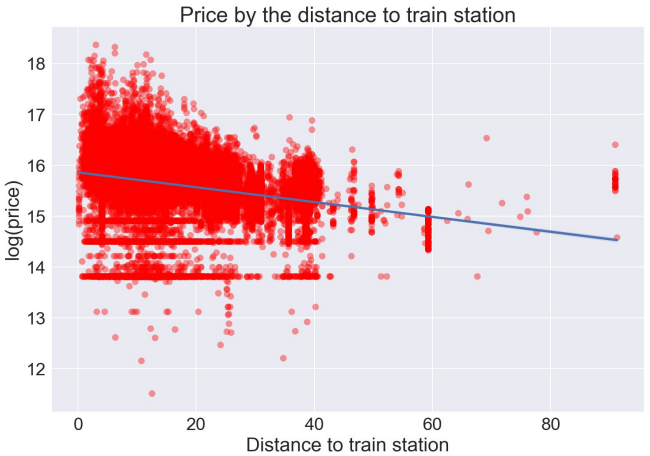
Major predictors used across models



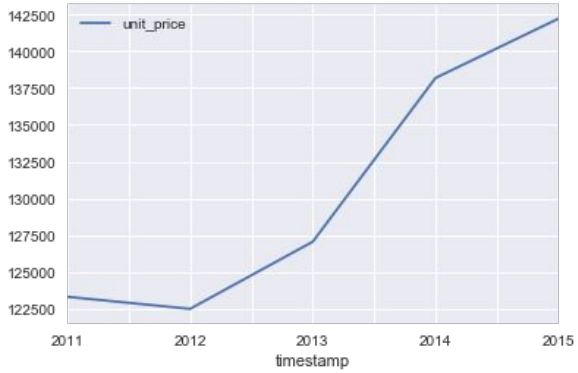


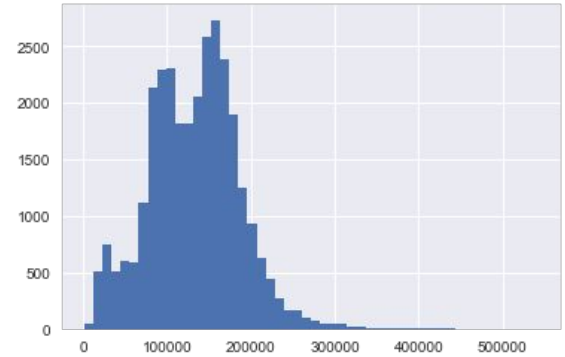


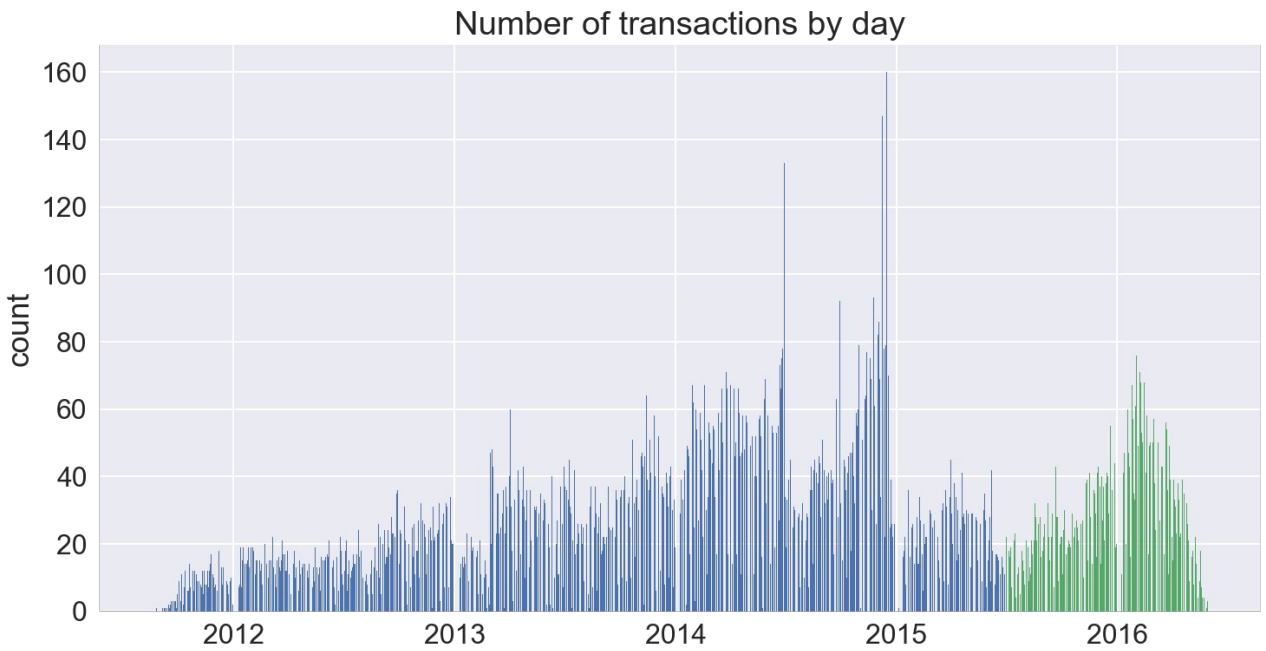




Trend: The unit price changes by year







Feature Selection with Random Forest & Lasso

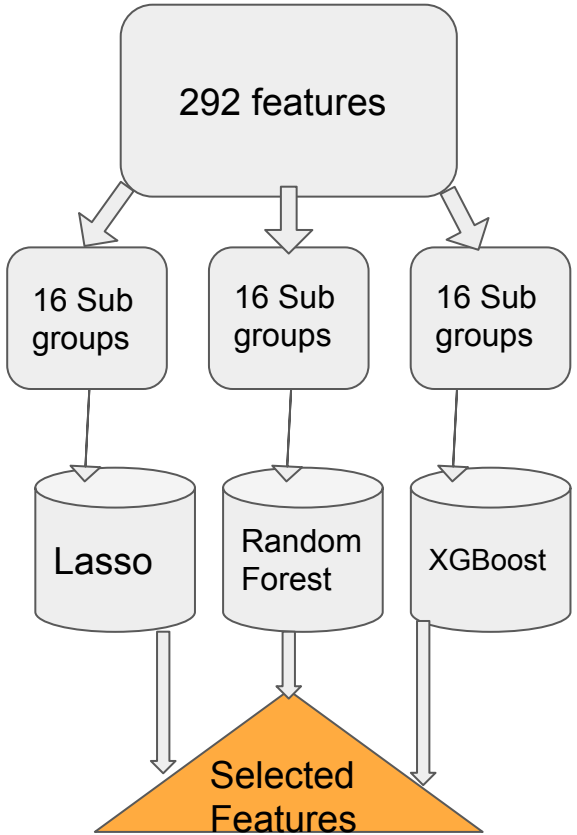
Step 1: Divide features into subgroups (i.e. demographics) => 16 subgroups

Step 2: Run random forests and Lasso on each subgroup

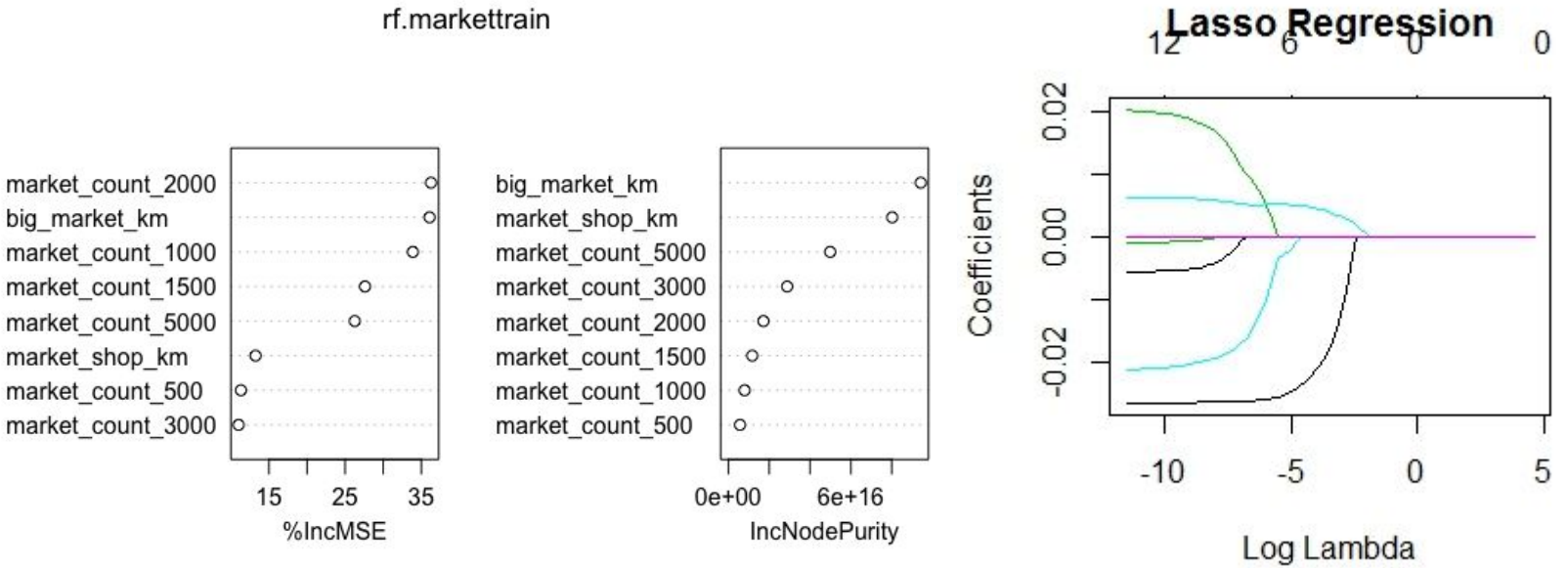
Interpretations: In most groups, LASSO would provide that all the features in the group are significant at the minimum MSE level for lambda parameter

In order to select features, we have chosen parameters that would go to zero slowest as lambda increases

35 features came out to be ideal in this case with MSE of 0.31 or RMSE of 0.56 for training data set and 0.46 on Kaggle's testing set score 0.35



Feature Selection with Random Forest & Lasso (ex)



Interpretations: In most groups, LASSO would provide that all the features in the group are significant at the minimum MSE level for lambda parameter. 35 features came out to be ideal in this case with MSE of 0.31 or RMSE of 0.56 for training data set and 0.46 on Kaggle's testing set score 0.35

Features Selected

Apartment characteristics'Full_sq', 'life_sq', 'floor', 'max_floor', 'material', 'build_year', 'num_room', 'kitch_sq', 'state', 'product_type', 'sub_area',

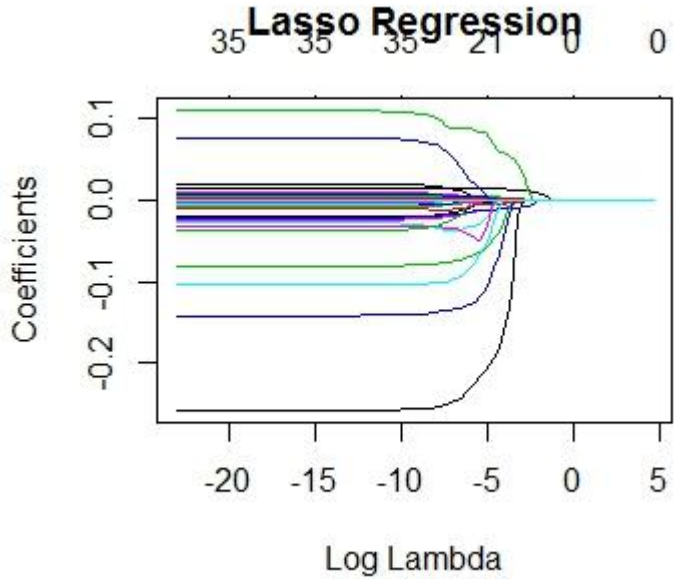
Distance to transportation, services, lifestyle needs. "public_healthcare_km", "additional_education_km", "workplaces_km", "big_church_km", "mosque_km", "big_road2_km", "ttk_km", "basketball_km", "swim_pool_km", "market_shop_km", 'green_zone_km', 'shopping_centers_km', 'public_transport_station_km', 'railroad_1line'

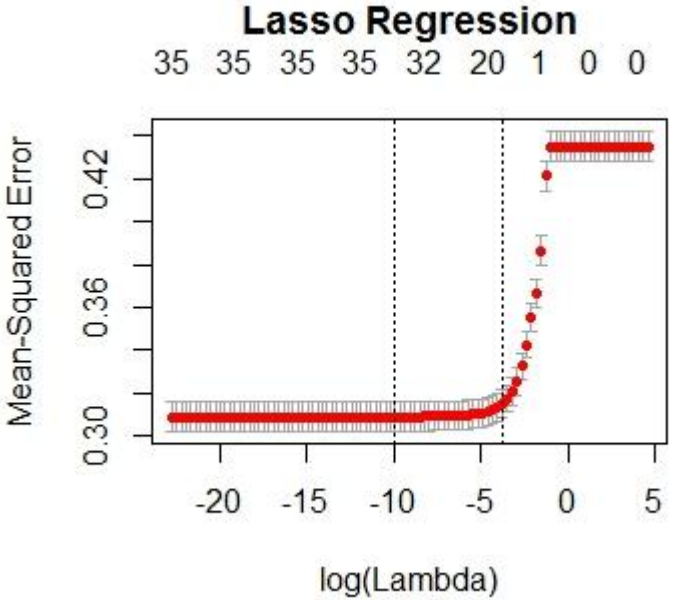
Demographics "Full_all", "male_f", "0_17_female" , "X16_29_female"

Neighborhood Characteristics "Culture_objects_top_25", "market_count_1000", "indust_part", 'children_preschool', 'trc_sqm_5000', 'sport_count_2000', "build_count_monolith"

Raion Characteristics "incineration_raion" ,"oil_chemistry_raion", "sport_objects_raion"

LASSO result





Multiple Linear Regression

We have also tried utilizing Multiple Linear Regression using select features that would make sense in making housing price prediction

For the simplest model, we have used ‘full_sq’(size of the unit), ‘ttk_km’(distance to the Third Ring), and ‘public_transport_station_min_walk’(minutes to walk to public transportation station)

Result was that this simple model gave a superior result to LASSO model with 35 features with RMSE of 0.499 on training set and Kaggle’s score of 0.37535

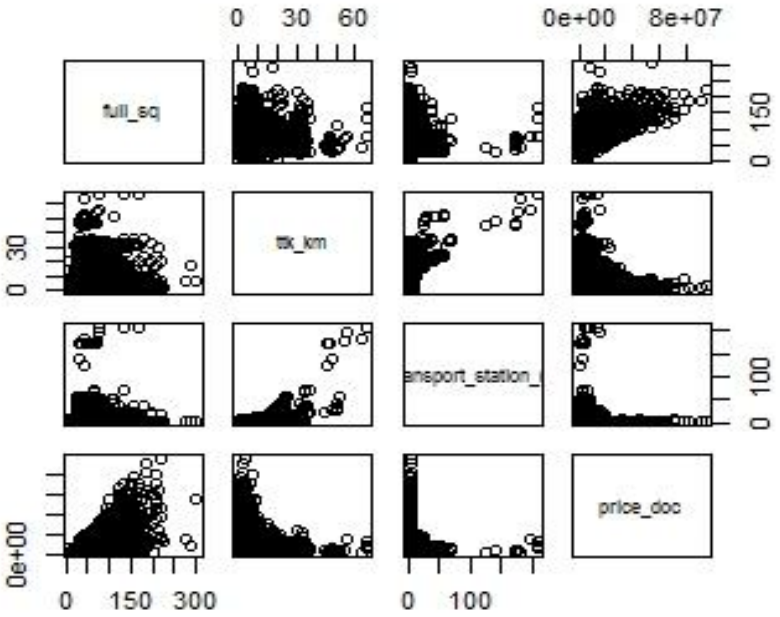
Using 15 features, we were able to lower RMSE a bit further to 0.466 on training set and Kaggle’s score of 0.35189

Macro data may not be as helpful as it is time series data and if year/month are included as independent variable, it would incorporate the time element

Scatterplot of 3 variables

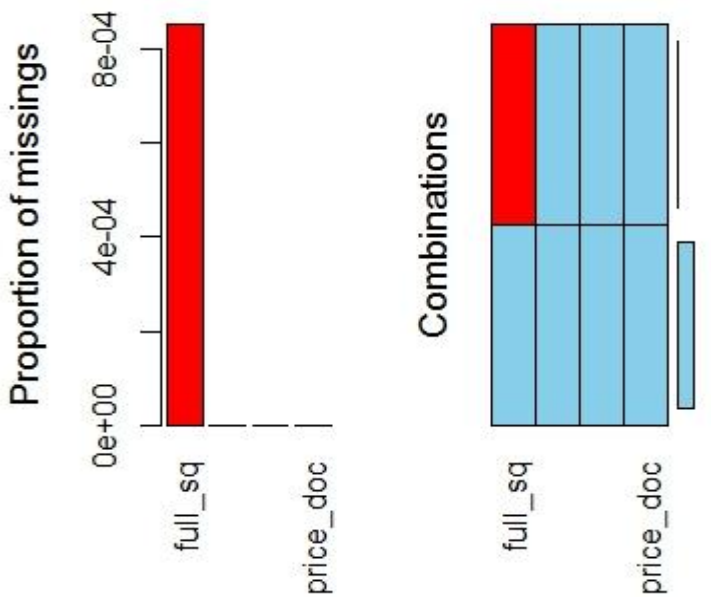
From the Scatterplot it seems that housing price has positive relations with full_sq, and negative relationship with ttk_km and public_transport_station_min_walk

That is, the bigger the house/apartment, price is larger, closer to the third ring, it is more expensive, and closer to public transportation station, the price is higher



Missing Value Inspection

There are hardly any missing values for these features



Multiple Linear Regression Result (3 features)

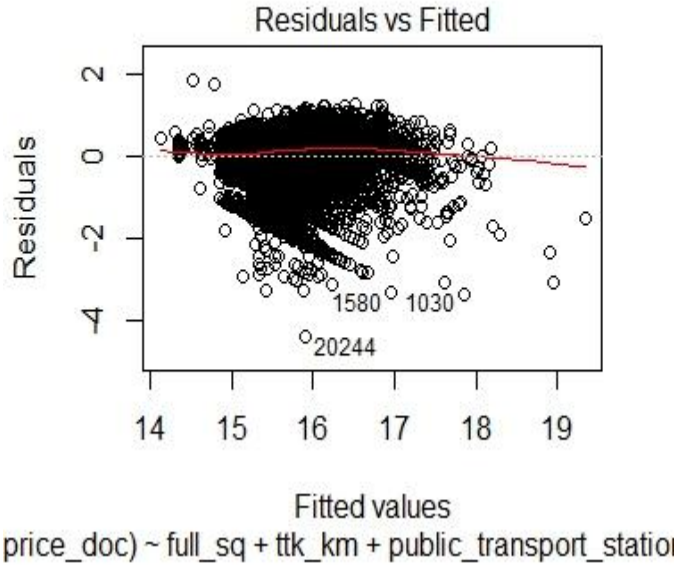
RMSE for training set: 0.499

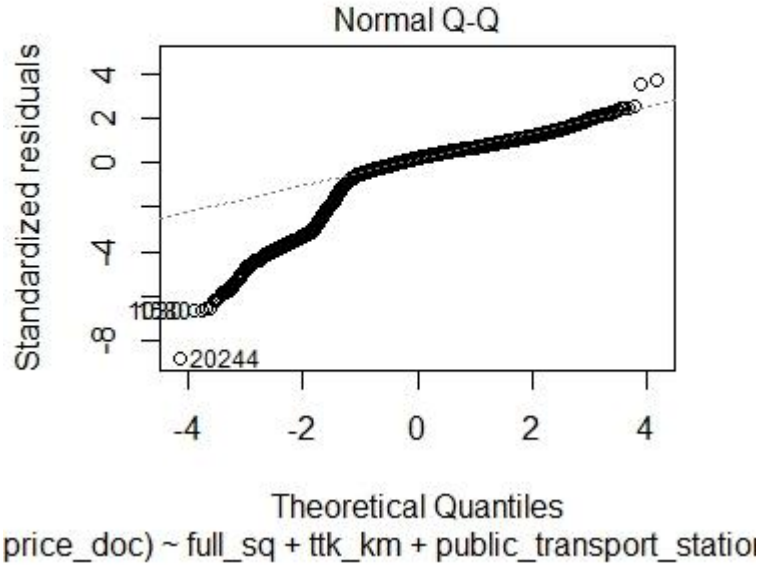
	Estimate	Std. Error
(Intercept)	15.0521835	0.0086749
full_sq	0.0145636	0.0001356
ttk_km	-0.0194311	0.0004101
public_transport_station_min_walk	-0.0016595	0.0002162

	t value	Pr(>|t|)
(Intercept)	1735.136	< 2e-16 ***
full_sq	107.432	< 2e-16 ***
ttk_km	-47.385	< 2e-16 ***
public_transport_station_min_walk	-7.674	1.71e-14 ***

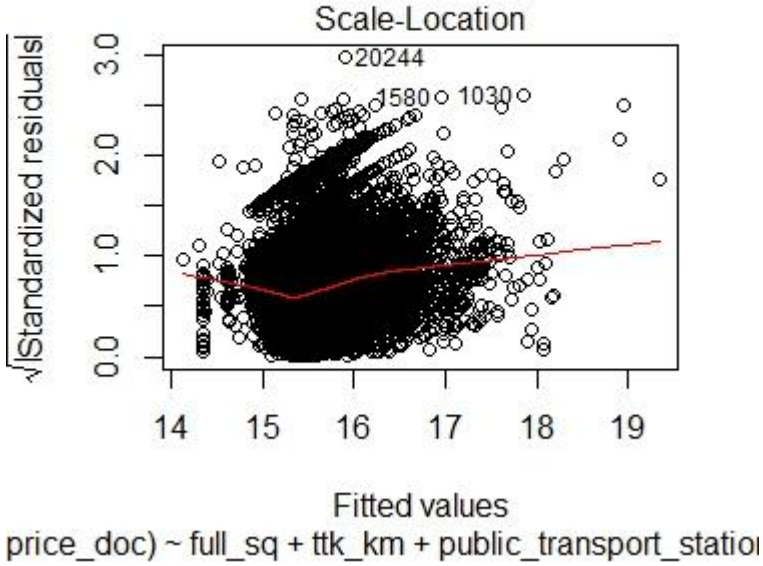
Residual standard error: 0.499 on 30440 degrees of freedom
  (26 observations deleted due to missingness)
Multiple R-squared: 0.3182, Adjusted R-squared: 0.3181
F-statistic: 4735 on 3 and 30440 DF, p-value: < 2.2e-16

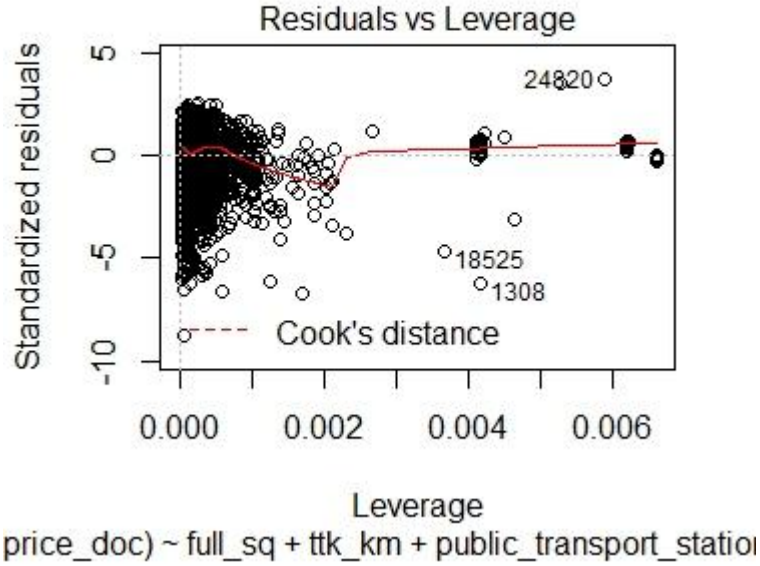
Multiple Linear Regression Assumption Charts





Multiple Linear Regression Assumption Charts

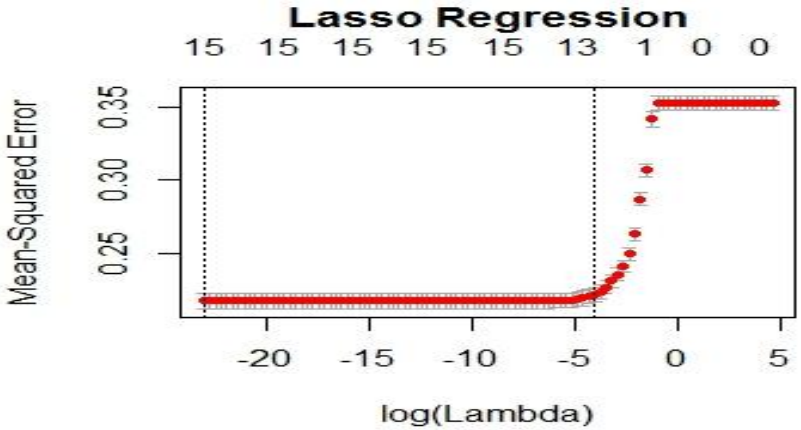




Multiple Linear Regression (15 features)

This version uses more features that would make sense in predicting housing prices(material feature has about 30% of the data missing and those observations were excluded)

LASSO would indicate that this model is not overfitted



Why Use XGBoost?

- Accepts sparse feature format
- Execution Speed
- Insensitive to collinearity
- Flexibility -Customized Error Evaluation
- Model Performance
- Go-to algorithm for Kagglers

XGBoost

Features Selection: 11 main features + 28 selected features +macro features

- Macros: CPI, PPI,gdp_deflator etc.
- Feature Engineering
  - Density = Raion Population /Area Size
  - Month/Weekly Transaction volume Count
  - df['dow'] = df.date.dt.dayofweek (Using datetime library)
  - Relative Floor= Floor / Max. No of Floor
  - Avg. Room Size=living area/ No. of Rooms

XGboost Prediction

Summary of Results

	XGboost				Multiple Linear Regression	
Features (n)	6	43	77	200	3	15
RMSE (Val)	0.43	0.416	0.416	0.415	0.499	0.466
Kaggle Score (RMSLE)	0.334	0.3243 0.314*	0.324	0.343	0.375	0.352

*Unit price as Y, and adjusted by a coefficient to count the bias

Conclusion: What to focus in the real world

- XGBoost is able to give best result for RMSE, Kaggle is addictive, does it tell much?
- LASSO and Random Forest results: < Multiple Linear Regression with 3 key features
- Common sense VS cumbersome models
- Efficient and reasonable study design
- Focus on the several key features
- If we are redoing this project, we will start from the most intuitive and simplest route

In the real world, minimizing time,labor and cost to maximize profit is the universal king:
The big question mark: Worth it or not!!!

Questions

Makridakis Competitions:

"Statistically sophisticated or complex methods do not necessarily provide more accurate forecasts than simpler ones."

Special Thanks to:

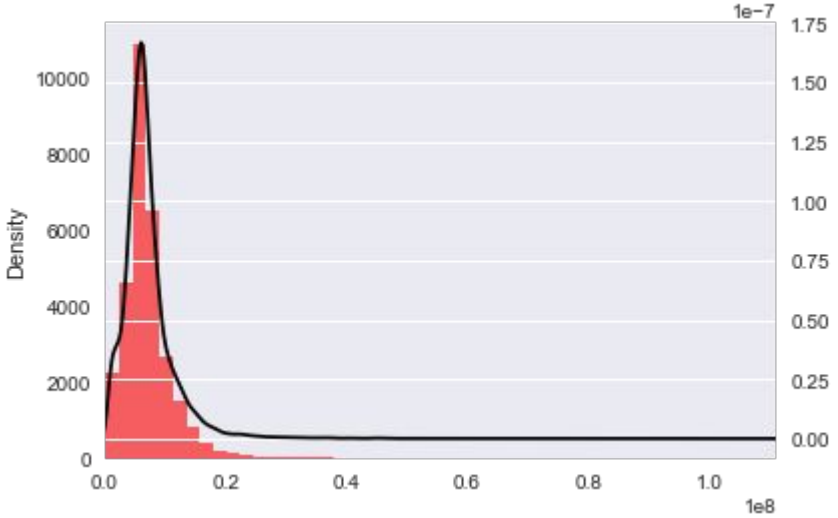
Aiko Liu, Luke Lin, Shu Yan, Zeyu Zhang, Drace Zhan, Thomas Kolasa

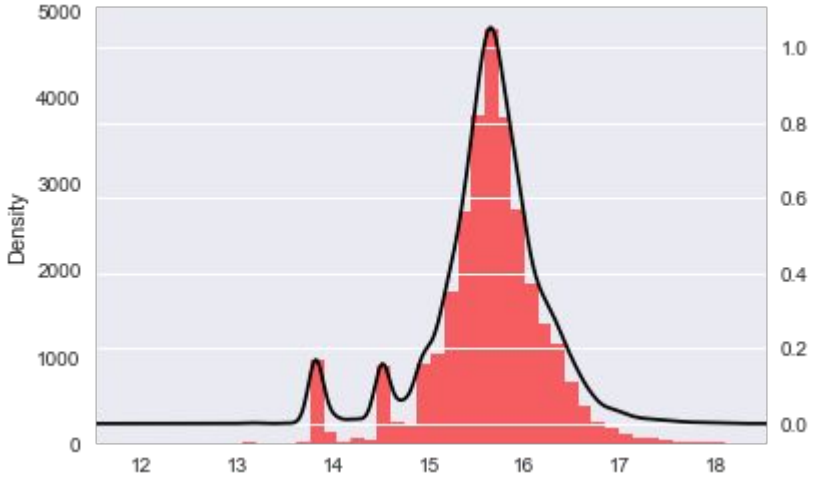
Fellow teammates

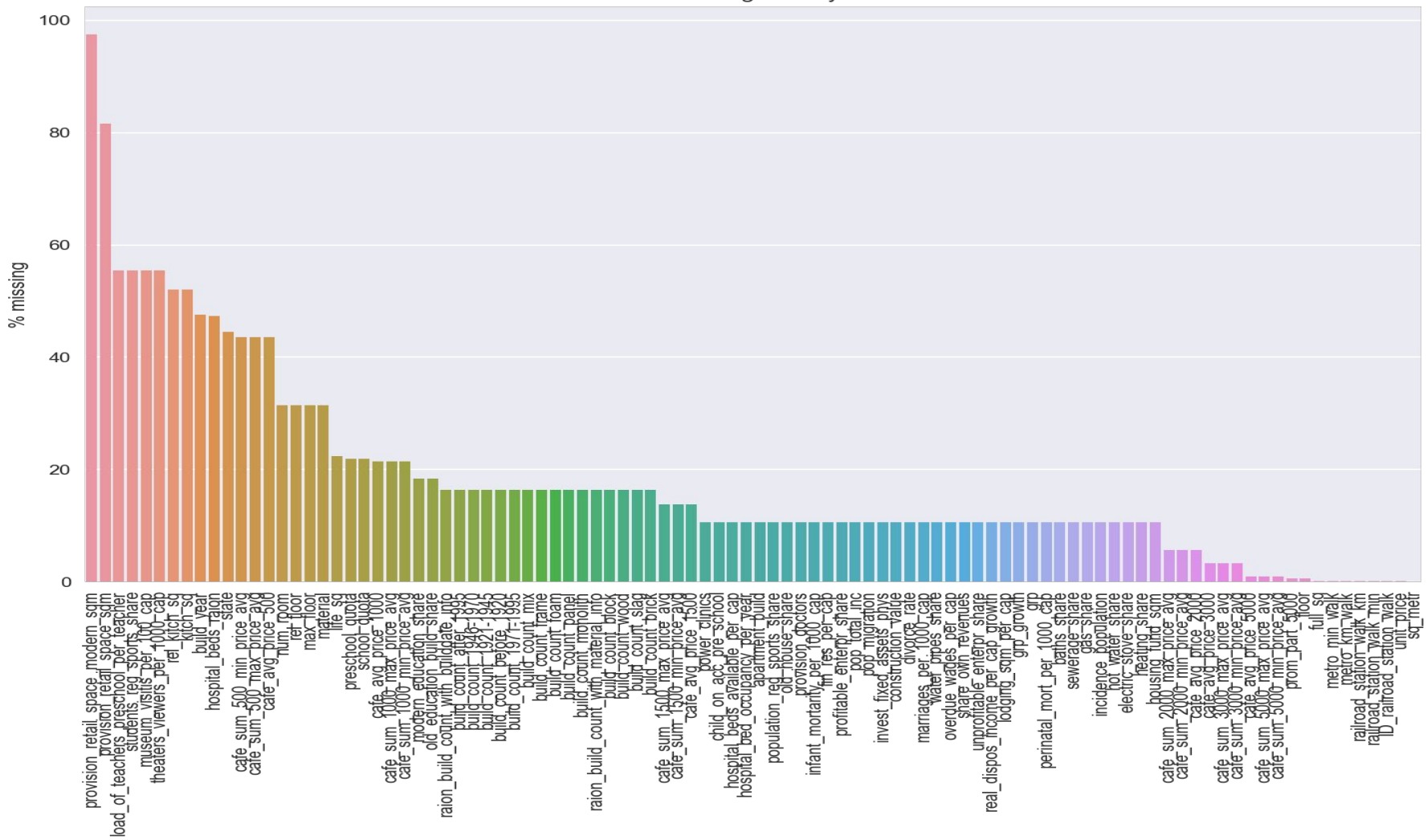
Kaggle for only allowing 5 submissions a day!

Backup slides

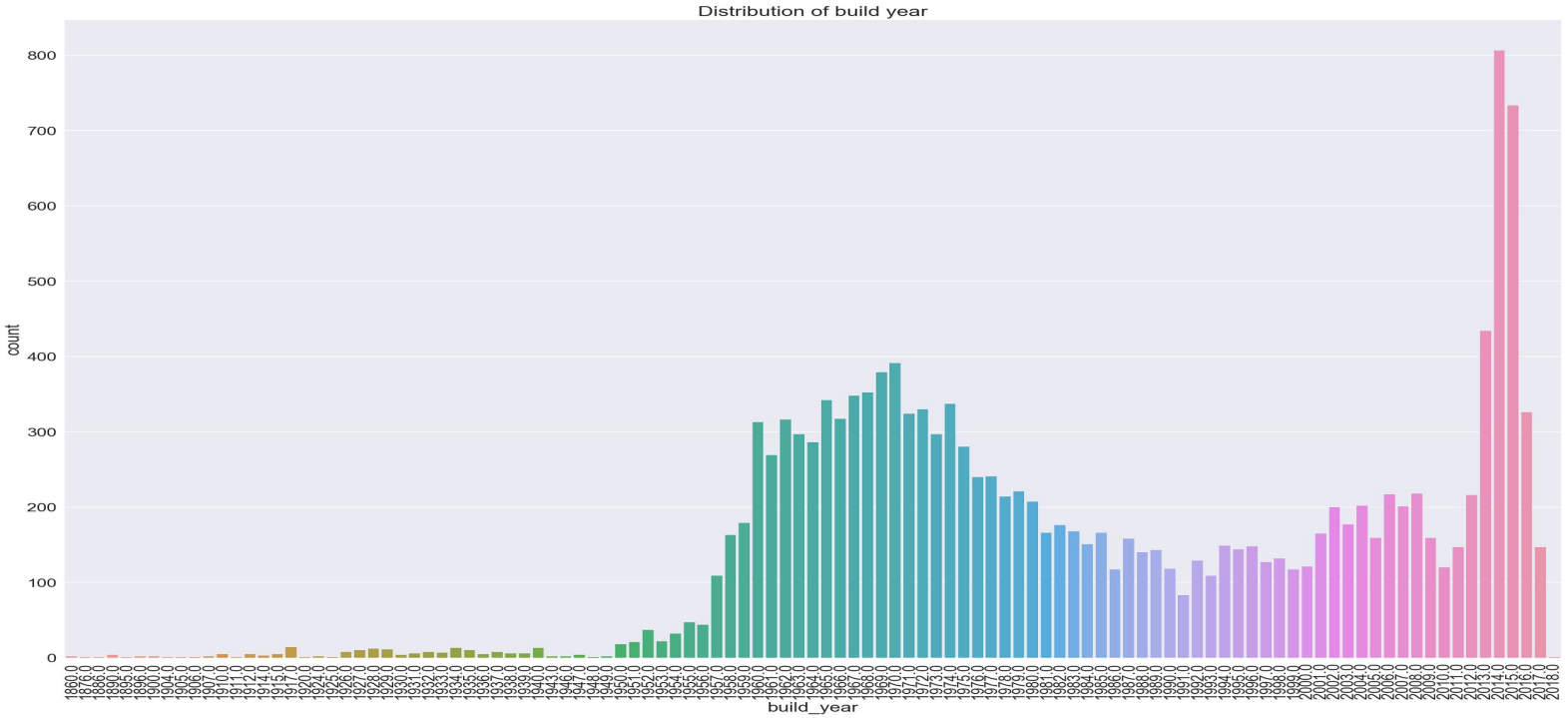
Price log transformation



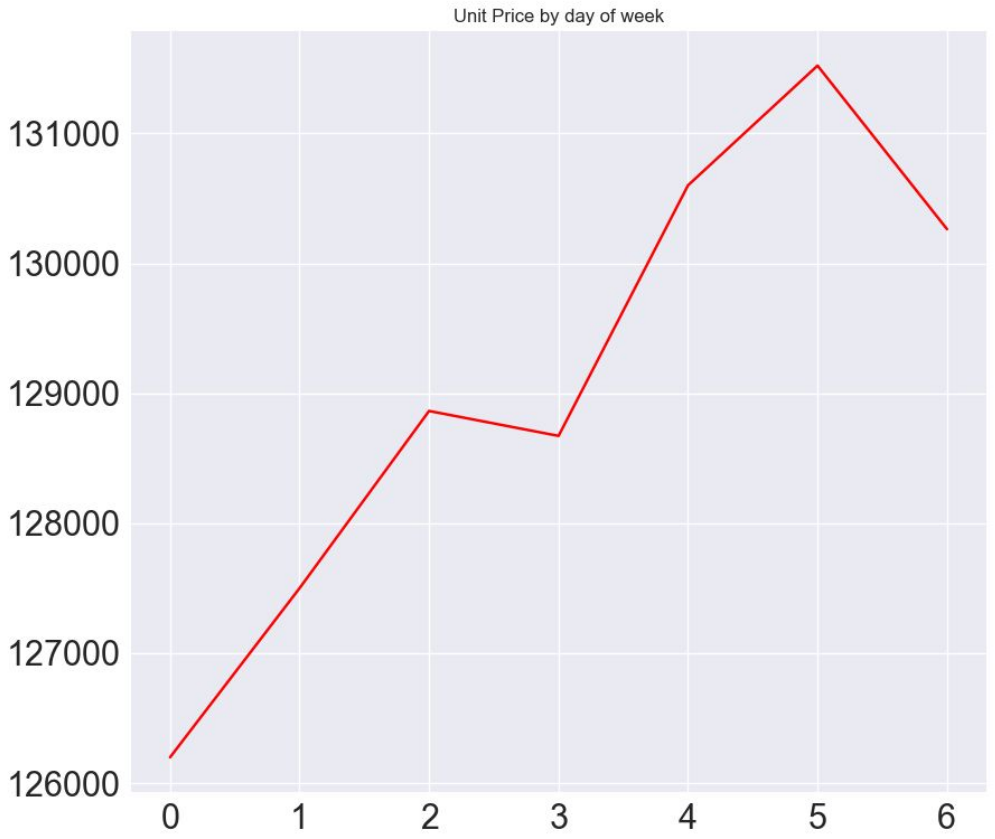



Percent missing data by feature
% missing
100
80
60
40
20
0
provision_retail_space_modern_sqm
provision_retail_space_sqm
load_of_teachers_preschool_per_teacher
students_reg_sports_share
museum_visits_per_100_cap
theaters_viewers_per_1000_cap
rel_kitch_sq
kitch_sq
build_year
hospital_beds_raion
cafe_sum_500_min_price_avg
cafe_sum_500_max_price_avg
cafe_avg_price_500
num_room
ref_floor
max_floor
material
life_sq
preschool_quota
school_quota
cafe_avg_price_1000
cafe_sum_1000_max_price_avg
cafe_sum_1000_min_price_avg
modern_education_share
old_education_build_share
raion_build_count_with_builddate_info
build_count_after_1995
build_count_1946-1970
build_count_1921-1945
build_count_before_1920
build_count_1971-1995
build_count_mix
build_count_frame
build_count_foam
build_count_panel
build_count_monolith
raion_build_count_with_material_info
build_count_block
build_count_wood
build_count_slag
build_count_brick
cafe_sum_1500_max_price_avg
cafe_sum_1500_min_price_avg
cafe_avg_price_1500
power_clinics
child_on_acc_pre_school
hospital_beds_available_per_cap
hospital_bed_occupancy_per_year
apartment_build
population_reg_sports_share
old_house_share
provision_doctors
infant_mortality_per_1000_cap
mil_fes_per_cap
profitable_enterpr_share
pop_total_inc
pop_migration
invest_fixed_assets_phys
construction_value
divorce_rate
marriages_per_1000_cap
water_pipes_share
overdue_wages_per_cap
share_own_revenues
unprofitable_enterpr_share
real_dispos_income_per_cap_growth
loggmg_sqm_per_cap
grp_growth
grp
perinatal_mort_per_1000_cap
baths_share
sewerage_share
gas_share
incidence_population
hot_water_share
electric_stove_share
heating_share
housing_fund_sqm
cafe_sum_2000_max_price_avg
cafe_sum_2000_min_price_avg
cafe_avg_price_2000
cafe_avg_price_3000
cafe_sum_3000_max_price_avg
cafe_sum_3000_min_price_avg
cafe_avg_price_5000
cafe_sum_5000_max_price_avg
cafe_sum_5000_min_price_avg
prom_part_5000
floor
full_sq
metro_min_walk
metro_km_walk
railroad_station_walk_km
railroad_station_walk_min
ID_railroad_station_walk
unit_price
sq_metr

Distribution of build year

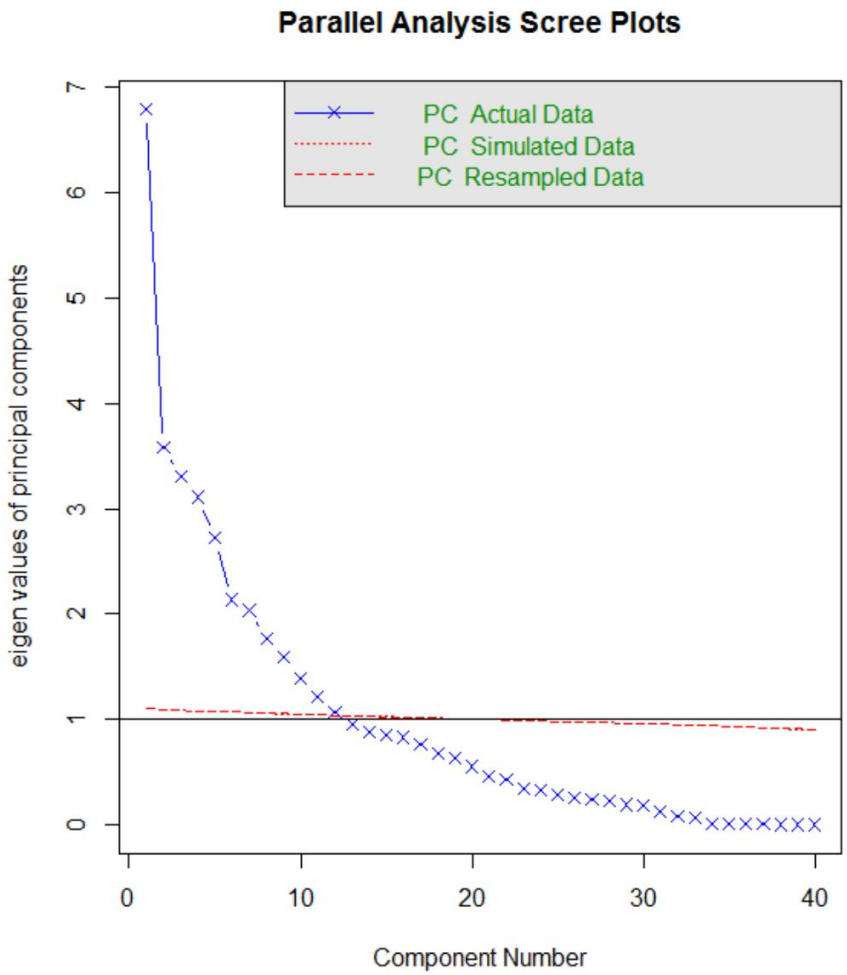


Unit Price by Days of Week



XGBoost Parameters

- Learning Rate : 'eta': 0.02
- Depth of a Tree: 'max_depth': 5
- Subsample Ratio of Training: 'subsample': 0.8
- Subsample Ratio of Columns: 'colsample_bytree': 0.7
- Learning Method: 'objective': 'reg:linear'


Parallel Analysis Scree Plots
PC Actual Data
PC Simulated Data
PC Resampled Data
eigen values of principal components
Component Number
0
1
2
3
4
5
6
7
0
10
20
30
40

```
# Month_year_count
month_year = (trainNew.month + trainNew.year * 100)
month_year_cnt_map = month_year.value_counts().to_dict()
trainNew['month_year_cnt'] = month_year.map(month_year_cnt_map)

# Week_year_count
week_year = (trainNew.date.dt.weekofyear + trainNew.date.dt.year * 100)
week_year_cnt_map = week_year.value_counts().to_dict()
trainNew['week_year_cnt'] = week_year.map(week_year_cnt_map)
```