Predicting Russian Housing Prices Using House Characteristics and Macroeconomic Data

Kristen Keller

Table of Contents

Abstract

We aimed to predict the prices of Russian houses using information about the individual houses and characteristics of the areas houses were located in. We were given a train dataset with 291 variables describing 30,471 houses that were sold from August 2011 to May 2016 and tasked with predicting the prices of 7,662 houses that were sold during the following months. Our main objective was minimizing the root mean square log error (RMSLE) of our predictions. Prior to creating any models, we put the data through an extensive pre-processing phase where we removed anomalous observations, rescaled variables, imputed missing values, applied dimension reduction techniques, created new features, and dropped unnecessary variables. We then trained XGBoost models and Random Forest models and found that the XGBoost models performed better than the Random Forest models. After an initial peek at the data, we saw that the data from 2014 onwards appeared to be more complete than the data from before 2014. We ran additional models using only data from 2014 and 2015, which performed better than the previous models. In an effort to further improve model performance, we added macroeconomic data describing the state of the Russian economy to our models. When we included all macroeconomic variables in our models, predictive performance decreased, suggesting that there was a lot of noise in these variables. When we added only a few macroeconomic variables we were interested in, predictive performance improved. The best predictions were generated when we averaged predictions from our two best models, emphasizing the importance of combining predictions from multiple models to create our final predictions.

Introduction

Motivation and Objective

Housing costs are of great interest to developers, homeowners, and bankers alike. Before building a home, a developer must be sure the selling price will be high enough to offset building costs. Before purchasing a home a homeowner must know the amount of money they will need to save up, and when the time comes to purchase a home they must be sure that the purchase price is fair. Before a banker lends money to a homeowner, they must be sure that if the homeowner fails to pay their loan and the property get turned over to the bank, it will be worth enough to offset the price of the initial loan. All these people could benefit from a model that reliably predicts the selling price of a home based on characteristics of the home and the state of the economy.

Sberbank, the largest bank in Russia, is currently sponsoring a Kaggle competition that challenges participants to build a model to predict the sale prices of Russian houses. They have provided two datasets. The train dataset contains house prices and can be used to build models to predict future prices. The test dataset contains no prices. The prices for these houses must be predicted using the train data and submitted on the Kaggle website. The submissions for this competition will be evaluated based on the root mean square log error (RMSLE). The formula for calculating RMLSE is

$$RMSLE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}[\log(p_i + 1) - \log(a_i + 1)]^2}$$

where ***n*** is the total number of observations in the dataset, $\mathbf{p_i}$ is the predicted response, and $\mathbf{a_i}$ is the actual response. It should be noted that this measure penalizes under predicted estimates more than over predicted estimates.

Description of Data

The dataset provided for this competition contains information on houses that were on the market in Moscow and the surrounding area between August 20, 2011 and May 30, 2016. The train dataset contains 30,471 observations that were recorded between August 20, 2011 and June

30, 2015 and the test dataset contains 7,662 observations that were recorded between July 1, 2015 and May 30, 2016. Note that there is no overlap between the time that the train and test datasets were collected.

There are 291 variables in the original dataset not counting the price of the house. Some of the variables are unique to each house, whereas others contain information on the neighborhood the house is located in. The variables that are specific to the house include information such as the date that the house was purchased on, the amount of living space, the number of rooms, the floor the house is located on if it is an apartment, and the year the house was built. The neighborhood variables include information on the population and distribution of demographic characteristics, the number of education and healthcare centers in the area, the presence of certain environmental hazards (ex. radioactive waste disposal sites), the distance to certain public facilities, and the average prices of restaurants and cafés in the neighborhood.

In addition to the test and train datasets, a third dataset was provided that contains information on the state of the Russian economy. This dataset contained 2484 observations on 100 variables. There was one record for each date ranging from January 1, 2010 to December 19, 2016. The dataset contains information on the exchange rate between the Russian Ruble and other currencies, inflation, GDP, average rent prices, trade balance, mortgage rates, unemployment, divorce rates, and mortality rates, among other factors. Note the first date in this dataset was more than a year before the first date in the train dataset. We will call this dataset the macroeconomic dataset.

We examined the missing data patterns in the test and train datasets. There were 51 variables in the train dataset that have at least one missing value and 48 in the test dataset. Overall, there were many more missing values in the train dataset. The variables containing information on the build year and the state of the houses were missing in 44.6% and 44.5% of cases in the train dataset but no cases in the test dataset. This suggests there are patterns in missingness by time since the test dataset contains later years than the train dataset. Many of the variables with missing values were related to the cafés in the area or the characteristics of the buildings in the neighborhood (Table 1).

Table 1. Proportion of missing values for each variable in the train and test dataset.

Variable	Proportion Missing (Train)	Proportion Missing (Test)
hospital_beds_raion	0.474	0.112
build_year	0.446	0.034
state	0.445	0.023
cafe_avg_price_500	0.436	0.104
cafe_sum_500_max_price_avg	0.436	0.104
cafe_sum_500_min_price_avg	0.436	0.104
kitch_sq	0.314	-
material	0.314	-
max_floor	0.314	-
num_room	0.314	-
preschool_quota	0.219	0.052
school_quota	0.219	0.052
cafe_avg_price_1000	0.214	0.04
cafe_sum_1000_max_price_avg	0.214	0.04
cafe_sum_1000_min_price_avg	0.214	0.04
life_sq	0.209	0.039
build_count_1921.1945	0.164	0.04
build_count_1946.1970	0.164	0.04
build_count_1971.1995	0.164	0.04
build_count_after_1995	0.164	0.04
build_count_before_1920	0.164	0.04
build_count_block	0.164	0.04
build_count_brick	0.164	0.04
build_count_foam	0.164	0.04
build_count_frame	0.164	0.04
build_count_mix	0.164	0.04
build_count_monolith	0.164	0.04
build_count_panel	0.164	0.04
build_count_slag	0.164	0.04
build_count_wood	0.164	0.04
raion_build_count_with_builddate_info	0.164	0.04
raion_build_count_with_material_info	0.164	0.04
cafe_avg_price_1500	0.138	0.027
cafe_sum_1500_max_price_avg	0.138	0.027
cafe_sum_1500_min_price_avg	0.138	0.027
cafe_avg_price_2000	0.057	0.014
cafe_sum_2000_max_price_avg	0.057	0.014
cafe_sum_2000_min_price_avg	0.057	0.014
cafe_avg_price_3000	0.033	0.006
cafe_sum_3000_max_price_avg	0.033	0.006
cafe_sum_3000_min_price_avg	0.033	0.006
cafe_avg_price_5000	0.01	0.004
cafe_sum_5000_max_price_avg	0.01	0.004
cafe_sum_5000_min_price_avg	0.01	0.004
prom_part_5000	0.006	0.003
floor	0.005	-
ID_railroad_station_walk	0.001	0.001
metro_km_walk	0.001	0.001
metro_min_walk	0.001	0.001
railroad_station_walk_km	0.001	0.001
railroad_station_walk_min	0.001	0.001
green_part_2000	-	0.001
product_type	-	0.001

We calculated what percent of all variables were missing for each house then averaged over observations taken on the same day. We saw that data from 2012 and 2014 had few missing values but data from 2013 had many (Figure 1a). Next, we narrowed in on variables describing individual house and building characteristics (ex. build year, square footage, etc.) and saw that there was a sharp decline in missingness in the beginning of 2014 (Figure 1b). Specifically, maximum floor, material, build year, number of rooms, kitchen square footage, and the state of the house were all missing before 2014. This decline was reflected in the overall missing data pattern, but it did not explain the discrepancy between the number of missing values in 2012 and 2013. We knew that many of the variables related to the cafés in the area had missing values, so we examined the missingness in the café variables. We saw that there were few missing values in 2012 and many in 2013, suggesting that missingness in the café variables was driving the trends seen before (Figure 1c).

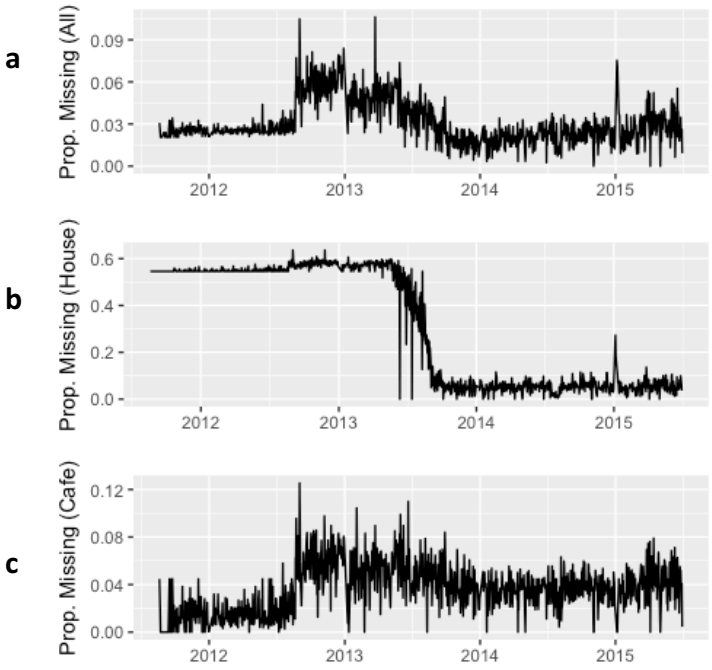
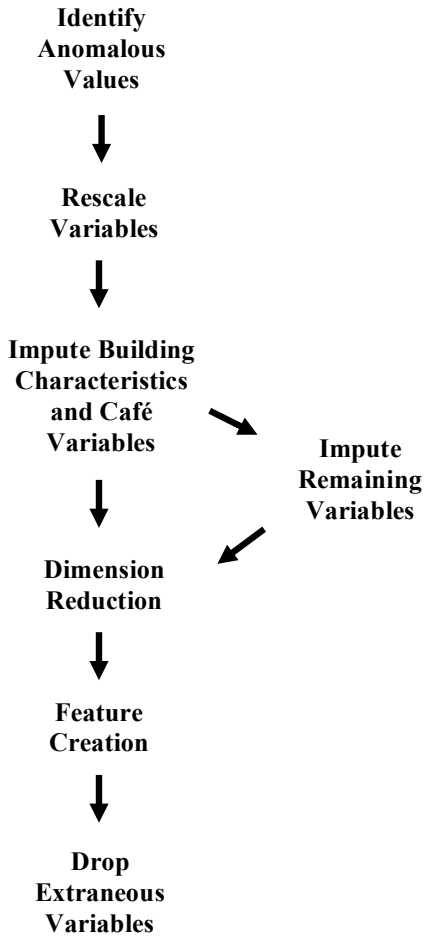


Figure 1. The average proportion of missing variables by date for (a) all variables, (b) variables related to characteristics of an individual house, and (c) café variables.

Data Pre-Processing

Prior to running any models, we took six steps to process our data (Schematic 1). These steps were applied to the test and train dataset but not the macroeconomic dataset. First, anomalous values were identified and replaced with missing values or appropriate substitutions. Anomalous values were mostly found for individual house characteristics. Second, variables that were not on appropriate scales were rescaled and text variables were recoded. Third, missing values were examined. A variable was created to identify houses that were in the same building and missing building characteristics were imputed using this variable. This was particularly useful as data from 2011-2013 did not contain any building characteristics. Next, missing values for the café variables were imputed using MICE in R.



Schematic 1. Illustration of steps that were taken to process data. Steps on the main axis were applied to both the fully imputed dataset and the partially imputed dataset. Steps to the right were applied to only the fully imputed dataset.

At this point, two separate datasets were created. For the first dataset, which we will call the partially imputed dataset, no further imputations were made. For the second dataset, which we will call the fully imputed dataset, imputations were created using MICE for the variables related to the schools in the area as well as the variables related to the individual houses and characteristics of houses in the area. All remaining missing values were imputed with the median for continuous variables or mode for categorical variables. In the fourth step, PCA was used to reduce the dimensionality of the variables that described the cafés in the area as well as the variables that described the number of public facilities in the surrounding area. In the fifth step, additional features were created using features that were already present in the dataset. In the sixth and final step, variables that did not contain useful information were dropped from the dataset.

Identifying Anomalous Values

There were multiple variables in the dataset for which anomalous values were found (Table 2). Most anomalous values were found for variables describing individual house characteristics. The dataset provided for this competition was created by merging data from multiple sources, so it was not surprising that some chunks of the data were more reliable than others.

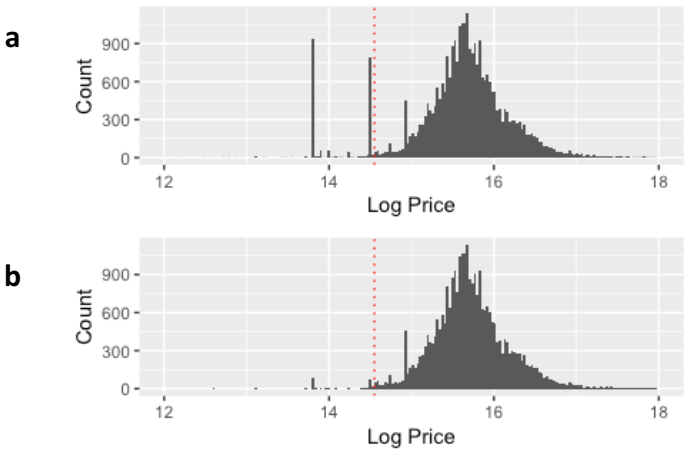


Figure 2. Distribution of the outcome variable (a) before and (b) after anomalous observations were removed.

First, the distribution of the log outcome variable was examined and it appeared there were some home values that were too low to be correct (Figure 2a). It is possible that these values were entered incorrectly. It is also possible that the housing prices were intentionally reported incorrectly for tax purposes or that the properties were purchased for low prices from friends or family members. We examined the distribution of the outcome variable by property type to determine whether the anomalous values were present for both investment properties and owner occupied properties and we saw that they were mostly present for investment properties (Figure 3a). This might support the hypothesis that property values have been purposely underreported to evade taxes.

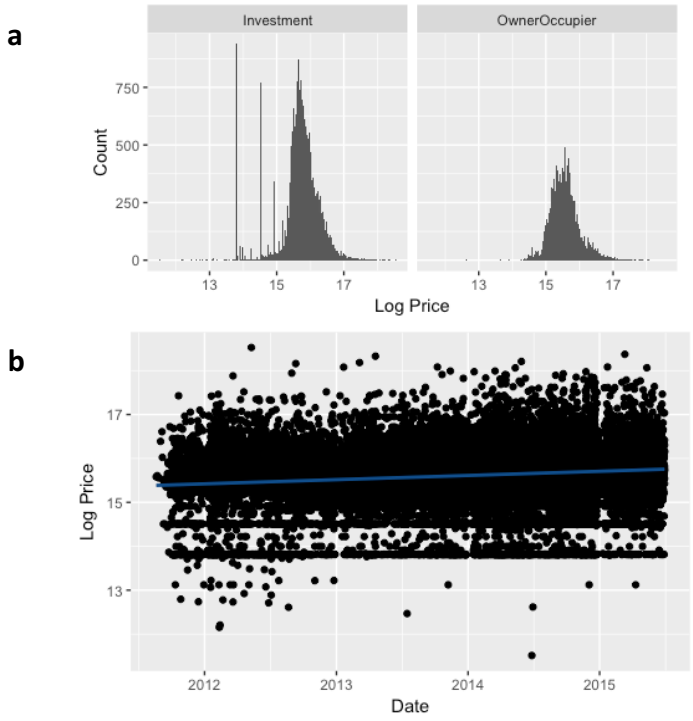


Figure 3. Distributions of the log price by (a) property type and (b) date.

We examined the distribution of the log price values by time to get an idea of how many anomalous values we would expect to show up in the test data. The test data come from the two years following the train data, so we were interested in seeing whether the anomalous values were more prevalent during the beginning of the train data. If the anomalous values were all clustered toward the beginning of the train period then we might expect fewer of these values in the test dataset. The number of anomalous values appeared to be higher on average in the earlier years, suggesting that the test data might have fewer anomalous values, but there were still some anomalous values in the later years (Figure 3b).

When making the decision of whether to remove the anomalous values, we kept in mind the metric that would be used to evaluate the competition. This metric penalizes underestimated responses more than overestimated responses, so removing a few observations with low values and biasing the model to return slightly higher values would not be as bad as keeping the low values when they should have been excluded. We wanted to differentiate the houses that truly had a low price from those with incorrect prices, so we removed the observations that had log prices less than 14.55 and a full square footage that was greater than the 5th percentile. There were 1,877 observations removed in total (Figure 2b).

Table 2. Summary of the actions that were taken to handle anomalous values. All actions were performed on both the test and train dataset.

Variable	Actions
Outcome Variable	
log(price_doc) < 14.55 & full_sq > quantile(full_sq,0.05)	Set to NA
Square Footage Variables	
full_sq ≤ 1, life_sq ≤ 1, kitch_sq ≤ 1	Set to NA
full_sq < 10	Set to NA
life_sq > full_sq & (life_sq – full_sq) < 5	Set life_sq to full_sq
life_sq > full_sq & (life_sq – full_sq) ≥ 5	Set life_sq to NA
kitch_sq > life_sq & (full_sq – life_sq) > life_sq	Set life_sq to NA
kitch_sq > life_sq & (full_sq – life_sq) ≤ life_sq	Set kitch_sq to NA
full_sq > 1000	Set to NA
House Floor	
max_floor < floor	Set max_floor to NA
Build Year	
build_year < 1600, build_year >2018	Set to NA
State of Building	
state > 4	Set to NA
Number of Rooms	
num_room = 0	Set to NA

Next, the variables describing the square footage of the houses were examined. This included variables that represented the total square footage of the house, the living space (excluding balconies, loggias, etc.), and the kitchen. It appeared that 0 and 1 were used to encode missing values, so in cases where the square footage variables were equal to 0 or 1 they were set to NA. Additionally any value of full square footage that was less than 10 was set to NA, as they did not seem realistic. We examined full square footage, living space, and kitchen square footage for consistency. Kitchen area was included as part of living space and living space was included as part of full square footage, so we expected kitchen square footage to be strictly smaller than living space and living space to be strictly smaller than full square footage.

We noted any cases where living space was higher than full square footage. If the difference between living space and full square footage was less than 5 square feet, living space was set equal to full square footage. There were many cases where living space was only 1 unit larger than full square footage. Setting the two variables equal seemed appropriate for these cases. If the difference was greater than or equal to 5, living space was set to NA. We assumed that living space contained the incorrect value because living space appeared to have many more anomalous values than full square footage. We also noted cases where kitchen square footage was higher than living space. We wanted to differentiate cases where living space was too small from cases where kitchen square footage was too large, as both cases appeared to be present in the data. It did not seem possible that less than half of the full square footage of a house could consist of indoor living space, so if the difference between full square footage and living space was greater than the value of living space then living space was set to NA. If the difference was less than the value of living space then kitchen square footage was set to NA. Any remaining values of full square footage that were larger than 1000 were set to NA.

There were a few additional variables that had values that seemed too high or too low to be correct. For example, there were houses with build years of 215 and 20. If the build year was lower than 1600 or higher than 2018, the observed value was replaced with NAs. There were some houses with build years that were a year or two later than the transaction date. These values were retained as the owners may have pre-purchased houses in complexes that were not yet completed. There were also many observations for which the maximum floor in the building, was lower than the floor that that specific house was on. Many of these observations had maximum floor values of 0 or 1. The maximum floor variable had more anomalous values than

the house floor variable, so in cases where the maximum floor was lower than house floor, maximum floor was set to NA. Finally, if the state of an house was greater than 4 it was deleted because the values for this variable were supposed to range from 1 to 4.

Rescaling Variables

There were multiple variables in the dataset that were not on the appropriate scale (Table 3). For example, the variable representing the number of buildings in the neighborhood that were made of brick was given as a count. Expressing the value in this way did not make sense because some neighborhoods had many more buildings than others. If there were 100 brick buildings in a neighborhood, this could mean that virtually all the buildings were made of brick or that less than 10% of the buildings were made of brick. Variables representing the number of buildings in the neighborhood made of a certain material were recoded as proportions using an additional variable that gave the number of buildings in the neighborhood with material information. Similar transformations were applied to the variables describing the number of buildings in the neighborhood built during a certain time period.

Next, we rescaled the variables that described the number of public facilities available in the neighborhood. These variables represented quantities such as the number of healthcare centers, top 20 universities, and shopping centers in the neighborhood. We re-expressed these variables as the number of facilities that were available per 1,000 population as some neighborhoods had many more people than others. We also re-expressed the count variables representing the number of people, male people, or female people of a certain age in the neighborhood as a proportion of population of interest (ie. the male population, female population, or overall population). We next recoded all remaining text variables as numeric. For indicator variables, responses of "yes" were coded as a 1 and responses of "no" were recoded as a 0. The variable representing the specific neighborhood the house was in was recoded as a series of indicators. Each neighborhood that had more than 250 houses represented in the dataset was given a unique indicator. Neighborhoods with fewer than 250 instances were assigned indicators representing whether the neighborhood had <10, 11-100, or 101-250 instances.

Table 3. Summary of actions that were taken to rescale variables. All actions were performed on test and train dataset.

Feature Name	Description	Formula
build_count_[material]	Express number of buildings made of [material] as a proportion of buildings with material data	build_count_[material] / raion_build_count_with_material_info
build_count_[year]	Express number of buildings build during [year] as a proportion of buildings with year data	build_count_[year] / raion_build_count_with_builddate_info
[facility_count]_raion	Express number of public facilities in raion corresponding to [facility_count] per 1,000 population (ex. number of hospital beds per 1,000 population)	([facility_count]_raion/raion_popul)*1000
Various indicator variables	Indicators that were coded as yes/no text variables converted to 0, 1 indicators	NA
sub_area	Created 0, 1 indicators for sub areas with >250 homes represented; indicators for those with ≤10, 11-100, and 101-250 records also included	NA
Demographic age variables	Express counts for then number of people in [age group] as a proportion of total [sex] population, where sex takes on values male, female, or total (existing variables representing total population values such as full_all were not consistent with values of other variables and therefore not used in denominator)	[age group]_[sex] / (young_[sex] + work_[sex] + ekder_[sex])

Missing Values and Dimension Reduction

There were 53 variables in the test and train datasets that contained at least one missing value (Table 1). Some of the algorithms we used to build models could handle missing data internally, whereas others could not. This means we needed two different datasets – one for methods that could handle missing data and one for methods that could not (Table 4). In the first dataset, which was used with algorithms that could not handle missing data internally, all missing values were imputed. In the second dataset, only a few missing values were imputed. We will refer to these datasets as the fully imputed dataset and the partially imputed dataset. We will first describe the imputations that were made for both datasets.

There were many houses in the dataset that appeared to be in the same building, so an apartment identification variable was created to identify apartments that were in the same building. We used the neighborhood the house was located in and the driving distance to the

nearest metro station to create the apartment id variable. The distance to the metro was given to many decimal places, so the neighborhood and metro variables appeared to be enough to uniquely identify a location without requiring the help of other distance variables. The apartment id variable was used to impute building characteristics such as the year the building was built, the maximum number of floors in the building, and the material the building was made of. We only made imputations if there was one unique value associated with an apartment id. These imputations were performed on all datasets because they filled in gaps with known values, rather than randomly generated guesses. This method was particularly useful for transactions that occurred prior to 2014, when many building characteristics were not recorded.

Many of the variables that contained missing values were related to the cafés and restaurants in the surrounding neighborhoods. These were numeric variables that represented metrics such as the number of cafés within distance *d*, the number of cafés within distance *d* with an average bill less than *b* dollars, and the minimum average bill price for cafés within distance *d*. There were 66 total variables related to cafés and restaurants in the neighborhood, 18 of which contained at least one missing value. These values were imputed using MICE in R. These variables were imputed in both the fully imputed dataset and the partially imputed dataset so that dimension reduction techniques could be applied to the café variables.

The following additional imputations were made for the fully imputed dataset. The variables that described the schools in the area were used to create imputations for the school variables that were missing. The same method was used as for the café variables. The variables describing the characteristics of individual houses as well as the characteristics of houses in the area were imputed in the same way. We employed imputation models that used subsets of the variables rather than models that used the entire set of variables to reduce the time required to create imputations. After these models were run, most of the variables with large amounts of missing data had been fully imputed. Any remaining missing values were imputed with the median value for continuous variables and the mode for categorical variables.

There were multiple groups of variables in the dataset that were highly correlated, so we applied dimension reduction techniques. We applied these techniques to the café variables and variables describing the number of public facilities within a certain distance. Specifically, PCA was applied to reduce the dimension of these variables and the top 5 principal components were included as predictors in the model. For both analyses, the principal component that explained

the most variance was highly correlated with the outcome variable. Our goal in applying these dimension reduction techniques was to reduce the run time of our models without greatly impacting their performance by condensing hundreds of variables worth of redundant information into just a few variables.

Table 4. Summary of actions that were taken to impute missing data for the test and train dataset.

Variable	Actions
Café Variables (both datasets) 18 variables containing string 'café'	1) Impute with MICE and predictive mean matching (pmm) using all other variables containing the string 'café'
Building Characteristics (both datasets) build_year, max_floor, material	1) Create apartment ID based on sub_area and distance from metro 2) If there is one unique value for [build_year, max_floor, material] in that apartment, impute all missing values in that apartment with that value
School Variables (fully imputed) school_quota, preschool_quota	1) Impute with pmm using children_preschool, children_school, preschool_education_centers_raion, school_education_centers_raion, school_education_centers_top_20_raion
Build Variables (fully imputed) All house and neighborhood building characteristics	1) Impute with pmm using full_sq, life_sq, floor, max_floor, material, build_year, num_room, kitch_sq, state, and all variables containing the string 'build_count'
Remaining Missing Values (fully imputed)	1) Impute with median for continuous variables or mode for binary variables

Feature Creation and Dropping Variables

Next, we created additional features to help predict the price of a house (Table 5). First, we looked to see whether there were any seasonal patterns in sales price. We plotted the log price by month to see if there was a relationship between the month a house was bought and the price of the house, but did not see any relationship (Figure 4a). We also plotted the log price by weekday to see if there was a relationship between the weekday that the house was bought and

the sale price, but did not see any relationship (Figure 4b). We did not include either of these variables in our final dataset, but we did include a count variable for the number of sales that occurred during the month a house was purchased. This variable might contain information about the state of the housing market at the time of purchase.

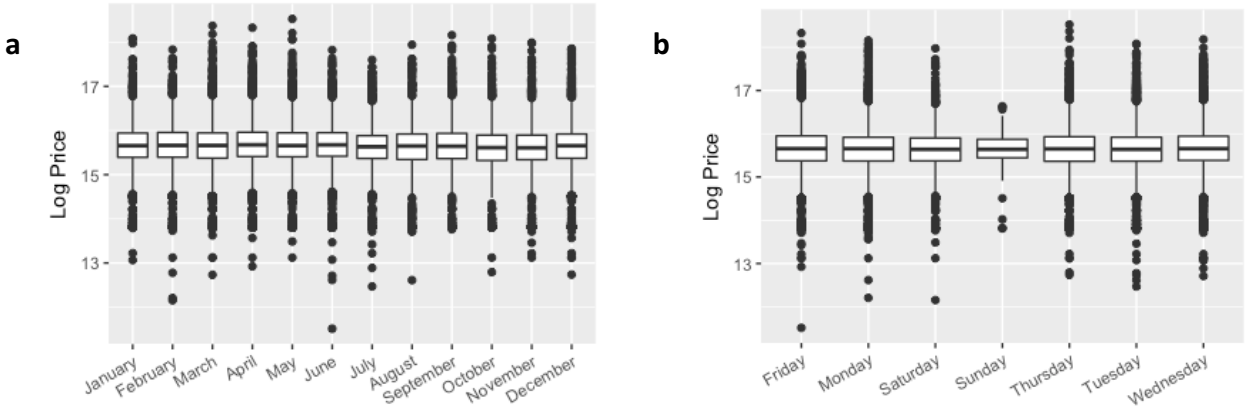


Figure 4. Plots of log price by (a) month and (b) weekday.

We created multiple features using the characteristics of individual houses. First, we created an indicator for whether or not the house was on the top floor and an indicator for whether or not the house was on the bottom floor. Two additional variables were created to describe the size of the buildings the houses were located in. The first was an indicator for whether the building had less than 5 floors and the second was an indicator for whether the building had 30 or more floors. We also included a few features related to the allocation of space in the house. First, we created a variable for the difference between the total square footage and the living space square footage. This represented the amount of space in the house attributable to balconies, loggias, and similar spaces. Additionally, we included a variable that represented the average square footage per room.

We also created a few variables related to the timing of purchase and the age of the house. A variable was created for the age of the building at the time of sale. For older buildings, this variable contained similar information to the build year variable. For recently built buildings, it helped distinguish between houses that were brand new and houses that were a few years old. An additional variable was created to indicate whether each transaction corresponded to the purchase of an already completed home or the pre-purchase of a home in a building that was not yet finished.

Additional features were created to describe characteristics of the buildings houses were located in. First, we created a variable that indicated whether the material used to build the house was a commonly used material in the neighborhood the house was in. Next, building specific features were created using the apartment id variable. We created a count variable representing the number of transactions in the dataset attributable to each building to represent how often houses in each building went on sale. We also included a second version of this variable that was standardized for building size by dividing the previous variable by the number of floors in the apartment. We figured that if a small building appeared in the dataset frequently, this might indicate there were problems with the building. We created an additional variable corresponding to the average square footage for a building.

The remaining variables created described the area that the houses were located in. We created variables describing how many different types of essential public facilities (schools, grocery stores, etc.) and how many different types of recreational facilities (basketball courts, theaters, etc.) were within 2 km, as well as a variable representing how many different types of environmental hazards (nuclear dump sites, chemical industries, etc.) were within 5 km of the property. We also created variables describing the automobile traffic in the area and the number of people per square km to tell us how busy the neighborhood around a house was. Additional variables were included that represented the ratio of preschool aged children to available preschool seats, and the ratio of high school aged children to high school seats. Finally, we created variables related to the demographics of the area, such as the proportion of the population that was male.

After additional features were created, variables that were not needed were dropped from the model. We dropped variables that did not contain information we were interested in, variables that contained redundant information with another variable, and variables that were recoded. Note that we also dropped max floor after we used it to create basic features because it seemed to contain more mistakes than the other house features. The following variables were dropped - max_floor, material, raion_build_count_with_builddate_info, raion_build_count_with_material_info, big_road1_1line, big_road1_2line, apt_id, ecology ,ID_metro, ID_railroad_station_walk, ID_railroad_station_avto, ID_big_road1, ID_big_road2, ID_railroad_terminal, ID_bus_terminal, sub_area, full_all, female_f, male_f.

Table 5. Features that we created using other variables in the dataset.

Feature Name		
top_floor	Indicator for top floor houses	floor = max_floor
bottom_floor	Indicator for bottom floor houses	floor = 1
building_less_5_floors	Indicator for buildings with less than 6 floors	max_floor ≤ 5
building_more_15_floors	Indicator for buildings with more than 19 floors	max_floor ≥ 15
balcony	Non_living space such as balconies, loggias, etc.	full_sq – life_sq
sq_per_room	Square footage per room	life_sq/num_room
num_sales_month	Number of home sales in that month	groupby month : length(id)
school_ratio	Ratio of schoolaged children to seats	children_school/school_quota
preschool_ratio	Ratio of preschool_aged children to seats	children_preschool/school_prequota
apt_id	Identifies apartments in the same building	paste(sub_area, metro_km_avto)
complex_num_sales	Number of records from that complex that are present in the dataset over number of floors	groupby apt_id, build_year, max_floor : length(apt_id)/num_floor
complex_num_sales_stand	Above standardized by building size	complex_num_sales/max_floor
complex_avg_sq	Average square footage of houses in that complex or sdirect area	groupby apt_id, build_year, max_floor : median(full_sq)
traffic_metro	How long it takes to travel certain distances	metro_min_avto/metro_km_avto
prox_score_essentials	How many essential public facilities are within 2 kilometers	+1each if there is a [school, grocery store, department store, university, workplace, public healthcare center, big church] within 2 km
prox_score_recreation	How many recreational facilities are within 2 kilometers	+1each if there is a [pool, ice rink, basketball court, theater, museum, exhibition, fitness center] within 2 km
env_hazard_score	How many environmental hazards are within 5 kilometers	+1 if there is an [incinerator, nuclear reactor, nuclear dump site, oil/chemical industry center] within 5 km
pre_purchase	Indicator for houses that were pre-purchased	build_year < timestamp
material_panel	Indicator for houses made of panel	material = 1
material_brick	Indicator for houses made of brick	material = 2
material_concrete	Indicator for houses made of concrete	material = 4
material_breezeblock	Indicator for houses made of breezeblock	material = 5
popular_material	Indicator for houses made out of a material used for 40%+ of buildings in that raion	NA
population_density	Number of people per sq. meter	raion_popul/area_m
good_ecology	Indicator for at least satisfactory ecology	ecology = good, excellent, satisfactory
poor_ecology	Indicator for poor ecology	ecology = poor
building_age	Age of building (set to 0 if negative)	year(timestamp) - build_year
all_new	Total population of raion	young_all + work_all + ekder_all
female_new	Total female population of raion	young_female + work_female + ekder_female
male_new	Total male population of raion	young_male + work_male + ekder_male
prop_male	Proportion of raion population that is male	male_new / all_new

Methods

Basic Models

Random forest models were run on the fully imputed dataset and evaluated using RMSLE and 5-fold cross validation. For each model, 150 trees were run using the caret package in R and a random seed of 75. A grid search was run using values of 50, 100, 150, and 200 for the number of predictors to consider at each split (mtry) and 500, 200, 100, 50, and 10 for the maximum terminal node size (nodeSize).

XGBoost models were run on the partially imputed dataset and evaluated using RMSLE and 5-fold cross validation. A grid search was run over values of 0.01, 0.1, and 0.3 for the shrinkage parameter (eta), values of 2, 5, and 10 for the tree depth (max_depth), and values of 300 and 500 for the number of trees (nrounds) using the train function in caret with a seed of 43. Default values were used for all other parameters (gamma = 0, colsample_bytree = 1, min_child_weight = 1, subsample = 1). This set of parameters was used for all future models unless otherwise stated. Next, the same set of XGBoost models was then trained on the partially imputed dataset using only observations before January 2, 2014. RMSLE was evaluated on a validation set containing observations after January 1, 2014. The new train dataset contained 25,486 observations and the validation set contained 3,108 observations. A seed of 322 was set prior to running the models.

Weighting Based on Date

The next set of models used the same parameters and validation set as the previous set. For these models, only observations taken after January 1, 2014 were used in the train set. The new train set contained 12,873 observations and the validation set contained the same 3,108 observations as before. A seed of 50 was set prior to running each model. We next compared the features that were commonly split on for a model trained using only the data from 2014 and a model trained using the data from all years (not including the 2015 validation set). The models used in this comparison had a shrinkage parameter of 0.1, tree depth of 5, and 500 trees.

After this, weighted XGBoost models were run on the train dataset containing observations from all years and RMSLE was evaluated using the validation set. We calculated weights for all observations using the following formula:

$$w_i = \ (round(\frac{D_i - D_0}{365}) + 1)^x$$

where D_0 was the first sales date in the dataset D_i was the sales date corresponding to the i^{th} record. We used x values of 1, 2, and 3 to calculate the weights. The same parameters that were used as in previous models and a seed of 50 was set.

Incorporating Macroeconomic Data

Next we merged the macroeconomic dataset with the train and test set by date. We used all macroeconomic variables and house variables to train the model. There were a total of 321 variables included in this model. The same parameters were used as before and a seed of 50 was set. The same validation set was used to evaluate RMSLE. The final model was the same as the previous model, except that only 5 macroeconomic variables were included. These variables were the average mortgage rate (mortgage_rate), average monthly salary (salary), inflation/consumer price index (cpi), quarterly GDP (gdp_quart), and the average rent price for an economical 2-bedroom apartment (rent_price_2room_eco).

Results

We will now discuss the process we went through to find a good model to predict housing prices. Ideally, we would want our final model to combine estimates from multiple individual models to produce predictions. However, for the purpose of this report we will focus on finding a single model that predicts housing prices well.

Basic Models

First, we used random forest models to make predictions using the fully imputed dataset. We performed a grid search to choose optimal values for parameters such as (mtry) and the minimum terminal node size (nodeSize). We evaluated the different parameter combinations using RMSLE and 5-fold cross validation. There were sizable decreases in RMSLE as the maximum terminal node size decreased, suggesting that we should use as small terminal node sizes as possible. There were not large differences in RMSLE when mtry was varied (Figure 5). Based on these results, we would select the model with an mtry of 150 and a node size of 10. The RMSLE for this model was 0.2460.

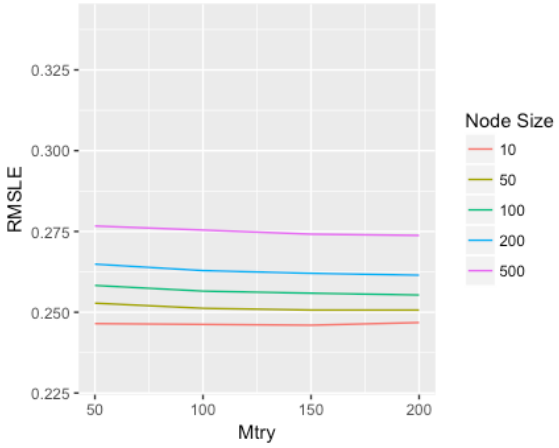


Figure 5. RMSLE calculated using 5 fold cross-validation for 20 different random forest models using all data.

We used XGBoost to make predictions using the partially imputed dataset. A grid search was preformed to select optimal values for the shrinkage parameter, the tree depth, and the number of trees. When the tree depth was small, the models with more trees performed better. As the tree depth increased, the differences in performance between the models with different numbers of trees decreased. The models with a shrinkage parameter of 0.1 generally performed better than the models with other parameters (Figure 6a). Overall, the 500 tree model with a tree depth of 5 and a shrinkage parameter of 0.1 performed best. The RMSLE for this model was 0.2379. These models performed notably better than the random forest models, so we decided to focus primarily on refining our XGBoost model.

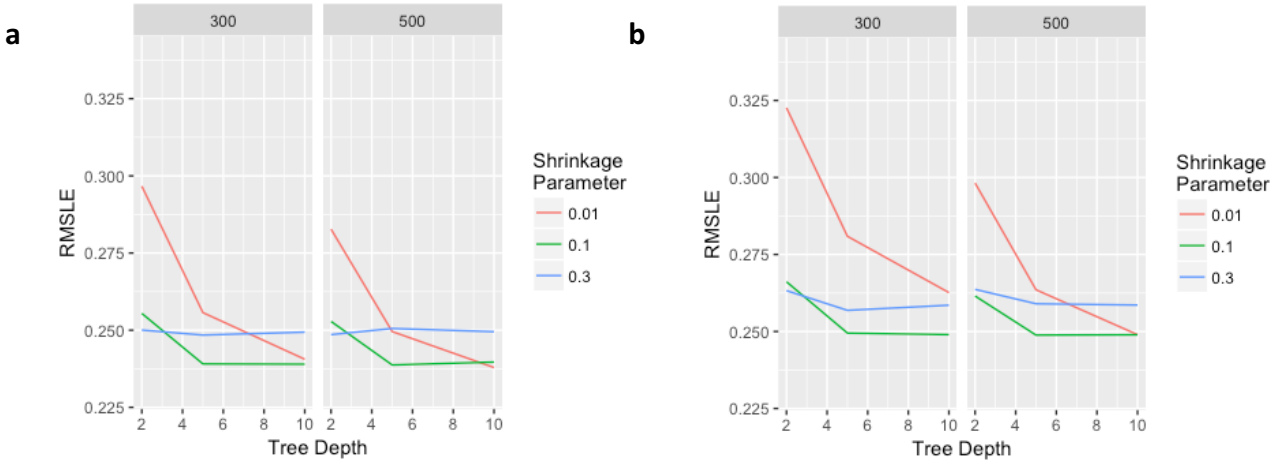


Figure 6. RMSLE calculated using (a) 5-fold cross-validation and (b) the 2015 validation set for 18 different XGBoost models run on all available data.

After running these models, we reevaluated the method we were using to calculate the RMSLE of our models. The test data come from a later time period than the train data and it might be valuable to take this temporal relationship into consideration when evaluating the performance of our models. As such, we decided to evaluate the RMSLE of our models using a validation set containing the data from 2015 rather than 5-fold cross validation. We ran the same XGBoost models as before and evaluated the performance on the validation set. We saw the same overall patterns as before, however the RMSLE values were systematically higher (Figure 6b). The model that preformed best here was the 500 tree model with a max depth of 5 and a shrinkage parameter of 0.1. The RMSLE for this model was 0.2488. From this point forward, we will use a validation holdout set approach to evaluate our RMSLE rather than cross validation.

Weighting Based on Date

When we examined the distribution of data, we saw that the data prior to 2014 did not contain information on characteristics of individual houses such as the number of rooms or build year (Figure 1b). We expected these variables to be among the most important variables for determining the price of a house, so we were concerned about these patterns of missing data. We ran an additional set of models using only the data from 2014 to train the models (and data from 2015 to validate). The models performed better than the models trained using all the data (Figure 7). The best performing model had 500 trees, a tree depth of 5, and a shrinkage parameter of 0.1. The RMSLE for this model was 0.2453.

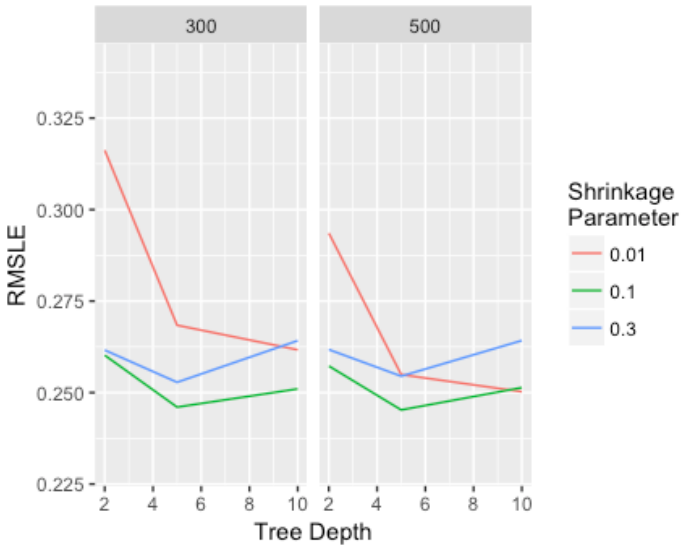


Figure 7. RMSLE calculated using the 2015 validation set for 18 XGBoost models using only data from 2015.

We looked at which variables were being split on most often in the model that used only data from 2014-2015 and the model that used data from all years. We wanted to see if the individual house characteristics were split on more often in the 2014-2015 model. We looked at the 10 features that were split on most frequently in the two models. The 2014-2015 model split on individual house characteristics such as the number of rooms and the state of the house more often than the other model. The model built using all the data split on characteristics that were available prior to 2014 such as the full square footage and the difference between the full square footage and the living space more often (Figure 8).

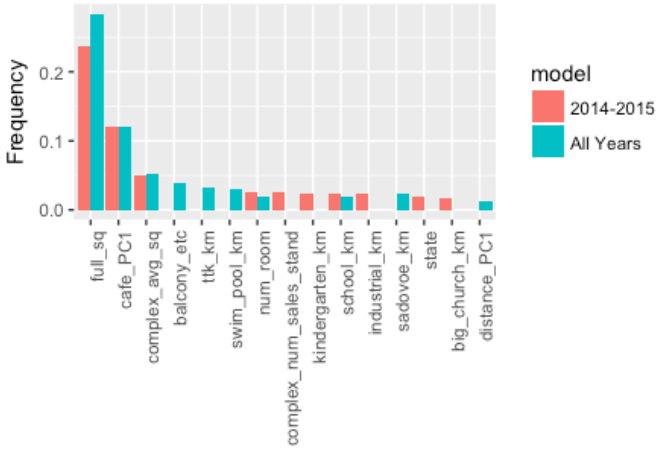


Figure 8. Frequency of splits for the 10 features that were split on most often in the model built on all data and the model built on data from 2014 to 2015.

We next ran weighted XGBoost models in an attempt to put more importance on the observations from later years without completely neglecting the data from years prior to 2014. The weights we used were a function to the number of years that passed from the first date in the dataset to the date of the property sale. Specifically, the weights were proportional to the number of years that had passed to the x^{th} power where x was an integer greater than 0 (see methods for additional information on weights). The models with an x of 1, 2, and 3 did not perform better than the model using only data from 2014-2015 (Figure 9). The best model had 500 trees, a max depth of 5, a shrinkage parameter of 0.1 and an x of 2. The RMSLE was 0.2481. We also used x values of 4, 5, and 6. These models performed slightly better than the previous models, but were still not as good as the model using only 2014-2015 data (Figure S1).

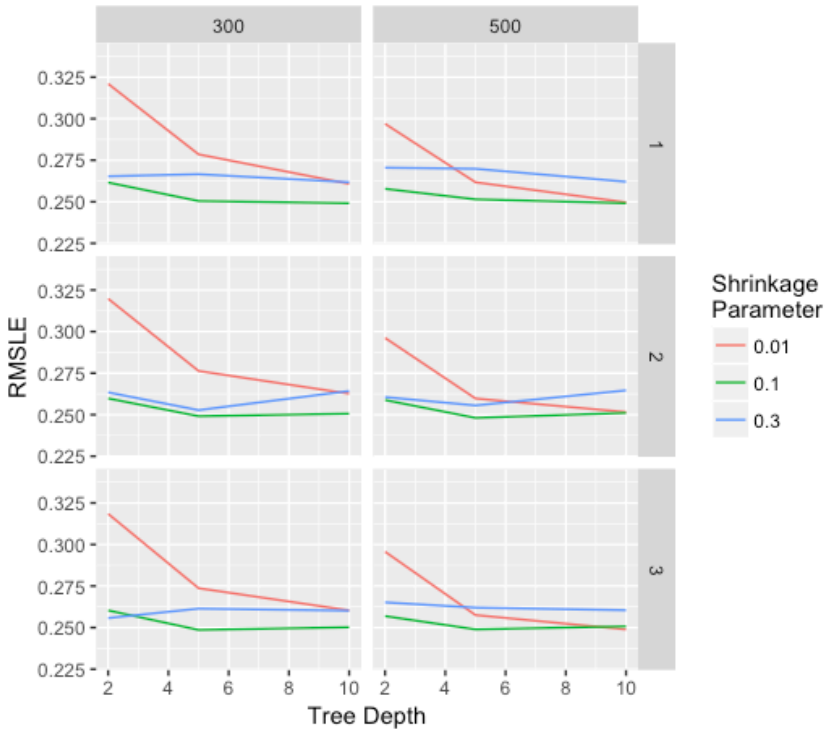


Figure 9. RMSLE calculated using the 2015 validation set for 32 weighted XGBoost models. For each observation, the weight assigned is a function of the number of number of years since the first date in the dataset. A value of 1 represents a linear function, 2 represents a quadratic function, and 3 represents a cubic function.

Incorporating Macroeconomic Data

We next incorporated information from the macroeconomic dataset into our models. Initially, all macroeconomic variables were included. We trained our initial models using data

from all available years. When we included all of the macroeconomic variables, our model performance decreased (Figure 10a). The best model had 500 trees, a max depth of 10, and a shrinkage parameter of 0.01. This model had a RMSLE of 0.2487. Next, we restricted the number of macroeconomic variables we used in our model and only included a subset of the variables. We only included the top 5 variables that we thought would be important. These variables were the average mortgage rate (mortgage_rate), average monthly salary (salary), inflation/consumer price index (cpi), quarterly GDP (gdp_quart), and the average rent price for an (economical) 2-bedroom apartment (rent_price_2room_eco). Adding only these variables to the model improved model performance. The model with the best performance was the model with 300 trees, a shrinkage parameter of 0.1, and a tree depth of 10. The RMSLE for this model was 0.2439. This was our best performing model so far.

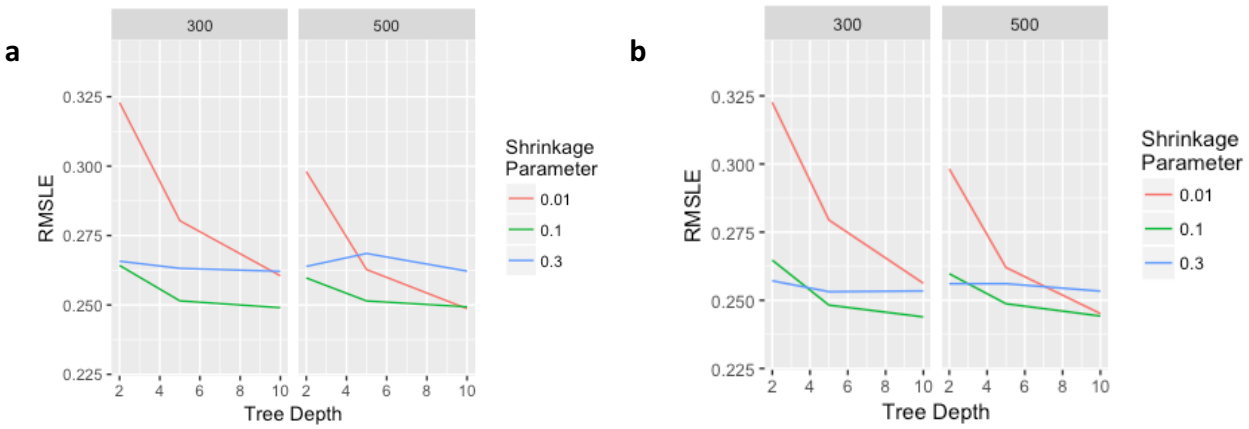


Figure 10. RMSLE calculated using the 2015 validation set for 18 XGBoost models incorporating (a) all macroeconomic data and (b) top 5 macroeconomic variables of interest using data from all years.

We looked for other ways to improve upon the previous model. First, we tried adding 5 additional macroeconomic variables producer price index (ppi), city apartment construction (apartment_build), trade surplus (balance_trade), the US Ruble exchange rate (usdrub), and the average rent for a business class 2 bedroom apartment (rent_price_2room_bus). This failed to improve model performance (Figure S2). Next, we ran a model containing the 5 macroeconomic variables of interest on only data from 2014 and 2015, but this decreased model performance (Figure S3). We tested additional parameter values for tree depth, the shrinkage parameter, and the number of trees but the model with the lowest RMSLE was still the model with 300 trees, a

shrinkage parameter of 0.1, and a tree depth of 10 (Figure S4). We also tried adjusting the proportion of predictors that were considered at each split and the proportion of observations that were sampled for each tree. These values were set to 1 in our previous models. Adjusting these models did not improve model performance (Figure S5, Figure S6).

Combining and Comparing Models

Table 6. Characteristics of the predictions generated by the two best models.

	Proportion
Both Models Off in Same Direction	0.843
Model 1 Prediction Overestimates	0.329
Model 2 Prediction Overestimates	0.387
Model 2 Prediction Closer	0.556

We were interested in comparing the two models that preformed best. The first model, which we will call Model 1, was the model that was built using 2014-2015 data and no macroeconomic variable with 500 trees, a max depth of 5, and a shrinkage parameter of 0.1. The second model, which we will call Model 2, was built using data from all years and 5 macroeconomic variables. We compared the predictions that were made using these models and saw that both of the models tended to underestimate the value of a house. The predictions made by the models were off in the same direction (ie. both models predicted too high or low) about 84% of the time. The predictions produced by Model 2 were closer to the true value slightly more often (Table 6), however the predictions made by Model 1 were closer a lot of the time. We decided to take a simple average of the predictions made by these models and use the averaged predictions as out final predictions. The RMSLE using these predictions was 0.2396, which was lower than any value we had previously seen (Figure 11).

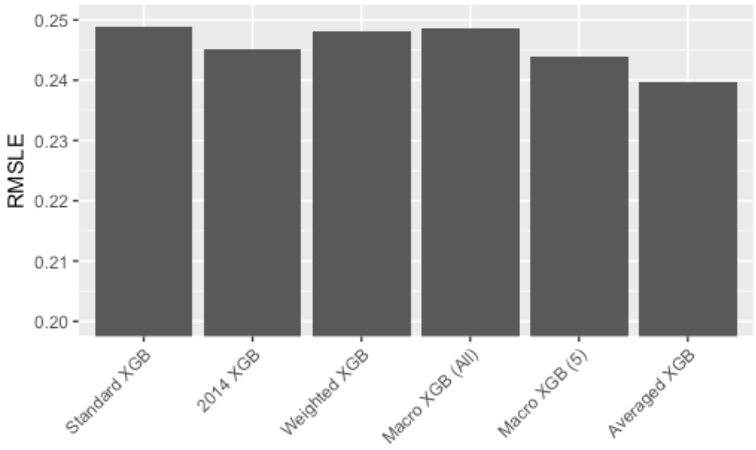


Figure 11. RMSLE for models with the optimal parameters calculated using the validation set from 2015.

Discussion

We built many models in an effort to find an individual model that predicted the price of a Russian house well. We focused primarily of random forest and XGBoost models. Overall, we saw that XGBoost models performed better than random forest models. This may be partially attributable to the fact that XGBoost models handle missing data internally, by learning which side of the split missing values should be sent to from the data. There are many distinctive missing data patterns in this dataset that might have been in some way informative. This information would be masked in the fully imputed dataset the random forest models were trained on. Additionally, it is possible that our imputation models did not produce good imputations. We did not spend much time assessing the quality of imputations made because we knew from the start we would be most interested in algorithms that handle missing data internally. Given more time, we would like to go back and make sure that the imputations we made were reasonable, and even test out different imputation methods such as K Nearest Neighbors.

When examining our data, we noticed that data from before 2014 had a lot more missing values than the rest of the data. Specifically, essential house characteristics such as the number of rooms and the state of the house were missing for all houses prior to 2014. We ran models using only data for transactions during and after 2014 and saw improvements in the fit of our model. We initially thought this could be because the validation data was from later years and house price increased over time so the average price in the 2014-2015 data would be closer to the

average house price in the train data. We decided to look at the variables that were being split on most frequently in the model using 2014-2015 data and the model using data from all years to see if there were any major differences. We saw that some of the housing characteristics only present after 2014, such as the state of the house, were split on often in the model using the more recent data but rarely in the other dataset. There were enough differences between the frequency of splits for key variables of interest that we concluded there were differences between the models that made the 2014 model better beyond just the increase in average price for that dataset.

Our first sets of models did not take into account the variables describing the state of the economy, but rather focused on the variables related to house and area characteristics. When we included the macroeconomic variables in our XGBoost models, we saw an increase in the RMSLE. This may be because the time period we looked at was relatively short. We might need to look at longer periods of time to see meaningful changes in some of the macroeconomic variables. If we put a bunch of noisy variables with random fluctuations in our model, it seems likely the model would pick out a few of these variables and mistake the noise for meaningful variation. This would result in an increase in RMSLE and could mask any predictive power other variables added to the model. When we added only a few relevant macroeconomic variables to the model, predictive performance improved. This supports the hypothesis that the presence of some noisy macroeconomic variables masked the positive effects of other macroeconomic variables.

There were two models that generated predictions with lower RMSLE than the rest. The first model, Model 1, did not contain macroeconomic data and only used data from 2014 and 2015. The second model, Model 2, included only 5 macroeconomic variables that we were interested in a priori and used data from all years. Model1 and Model 2 both made predictions that were too low 60-70% of the time. The fact that the models were consistently underpredicting the sales price of a house is of concern because RMSLE penalizes underprediction more than overprediction. Even if we could produce models that produced residuals of the same magnitude but tended to overestimate prices rather than underestimating them our RMSLE would decrease. We saw that the average housing prices were increasing over time, and we know that the train data comes before the validation data chronologically. This might contribute to the predicted prices for the validation data being too low. We may be able to solve this problem by applying

some transformation to price prior to training our models then transforming the predictions back to the original scale.

We further examined the predictions made by the two models and saw that model 2 generated predictions that were closer to the true values than Model 1 a little more than half of the time. Since each model predicted closer values around half of the time, we decided to average the predictions made by these models to create a new set of predictions. These new predictions had a much lower RMSLE than either of the previous models. This emphasizes the importance of using a final model that combines predictions from multiple individual models.

Future Work

In this report we focused primarily on building individual models with good properties. Our main interest for future work would be building models that combine the predictions made by other models. We could do this in a number of ways, such as using weighted averages of model predictions or by stacking models (ie. using the outputs of individual models as inputs for an overall model). In order to do this, we would want to build a diverse set of base models using different algorithms. We could also build different models on different subsets of the dataset and have some models that incorporate macroeconomic information and some that do not.

We would also be interested in finding more elegant ways to incorporate the macroeconomic data into our models. For starters, it might be good to try to incorporate other subsets of macroeconomic variables into the model. We could also think about averaging macroeconomic variables over longer time periods then incorporating the averaged information into the model. For example, the US/Ruble exchange rate is given on a daily basis, however there is probably a lot of noise due to random fluctuations. Perhaps if we instead looked at the average exchange rate for a month, this variable would be more useful. We might also want to account for lags in the effects of macroeconomic variables, because it takes a while to find and purchase a house after the economy starts improving. We might also be interested in modeling the macroeconomic information separately from the house characteristics then combining the information from the different models. For example, if we created a model using the macroeconomic data to predict the average sale price for a month, a quarter, or a year then we could incorporate this as an additional variable in our model.

There are also many other things that could be done to improve our models. For example, we might be interested in creating additional features. Specifically, it would be useful if we could create a good feature or a smaller set of features to summarize how desirable the locations of individual properties are. As we stated before, we might also be able to improve our models by applying some transformation to stabilize price prior to training our models.

Appendix A: Supplemental Figures

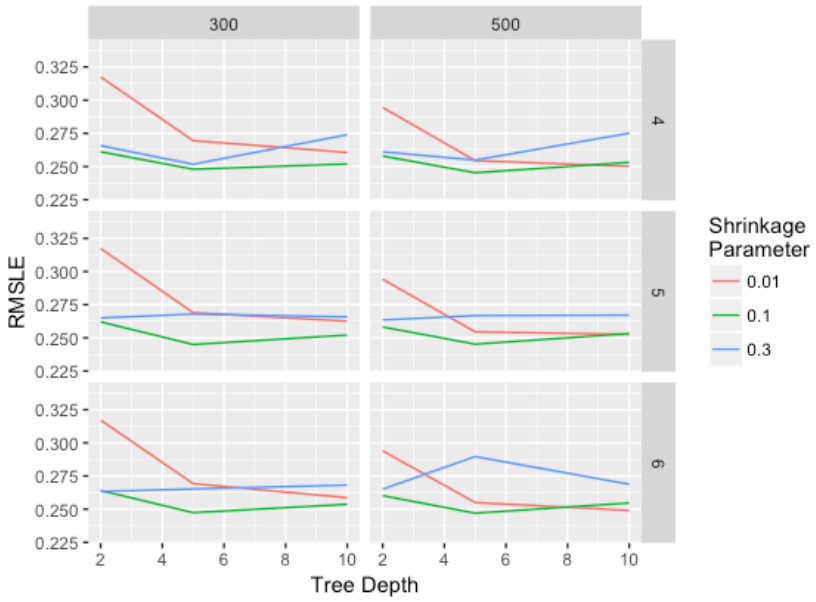


Figure S1. RMSLE calculated using the 2015 validation set for 32 weighted XGBoost models. For each observation, the weight assigned is a function of the number of years that have passed from the first date in the dataset to the date of the current sale. Models were trained using data from all years.

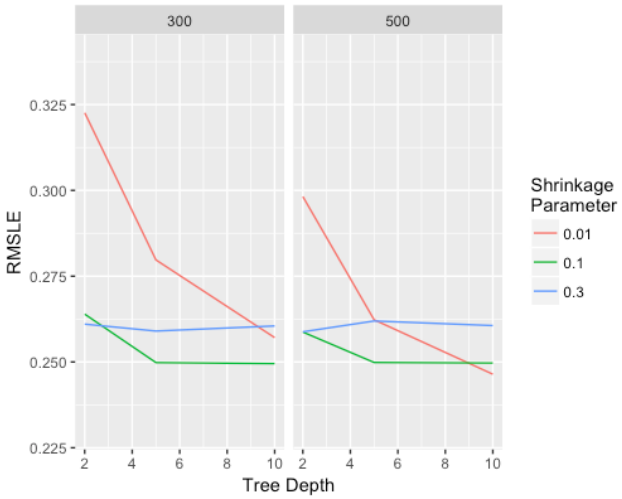


Figure S2. RMSLE calculated using the 2015 validation set for 18 XGBoost models containing the top 10 macroeconomic variables of interest. Models were trained using all data.

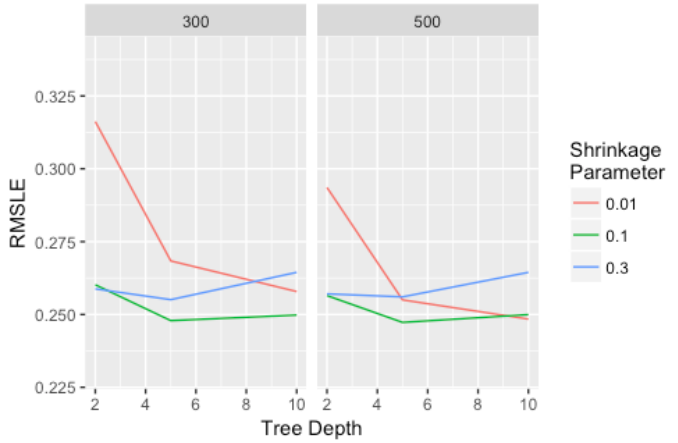


Figure S3. RMSLE calculated using the 2015 validation set for 18 XGBoost models containing the top 5 macroeconomic variables of interest. Models were trained using only data from 2014 and 2015.

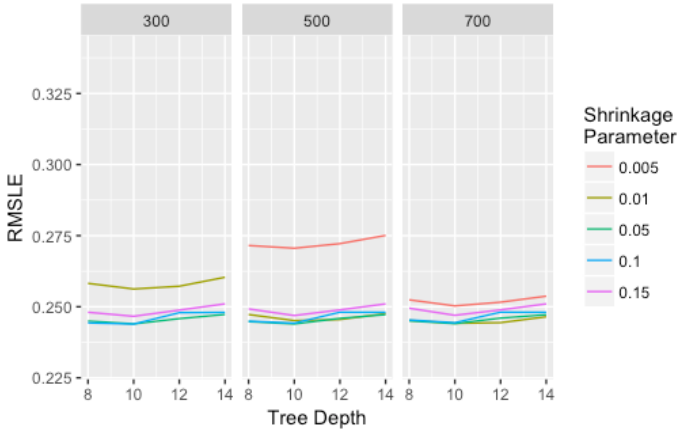


Figure S4. RMSLE calculated using the 2015 validation set for 60 XGBoost models containing the top 5 macroeconomic variables of interest. Additional values for the shrinkage parameter, tree depth, and number of trees were tested. Models were trained using all data. Note: models with 300 trees and a shrinkage parameter of 0.005 had RMSLE values above the upper limit of the y axis for this figure and therefore do not appear.

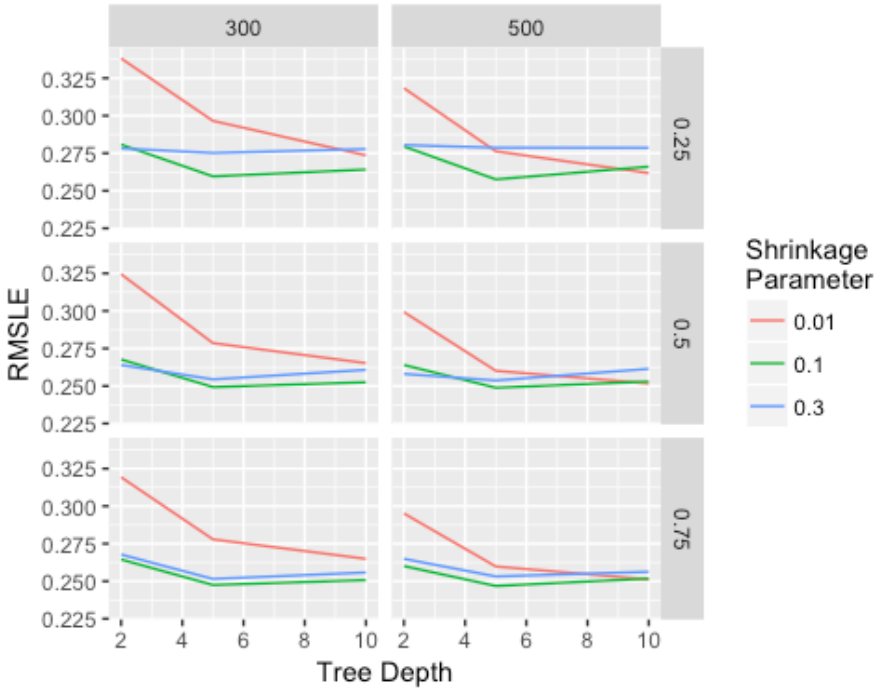


Figure S5. RMSLE calculated using the 2015 validation set for 54 XGBoost models incorporating the top 5 macroeconomic variables of interest using data from all years. The labels along the right of the vertical axis represent the proportion of predictors that were considered at each step.

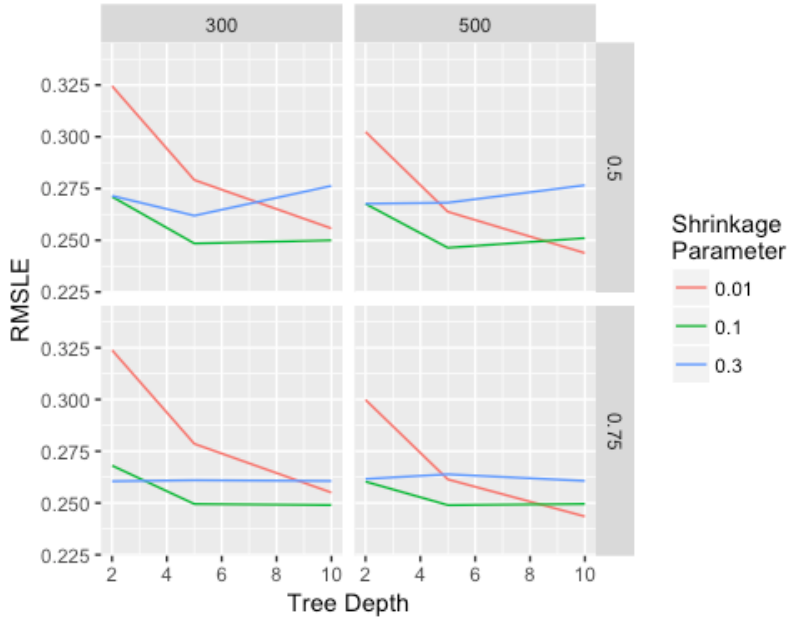


Figure S6. RMSLE calculated using the 2015 validation set for 54 XGBoost models incorporating the top 5 macroeconomic variables of interest using data from all years. The labels along the right of the vertical axis represent proportion of observations that were sampled for each tree.