Predicting House Prices

Quantitative Methods 1 (Tutorial)

Conor, Linette, Lucas, Minh

1 The intuition behind variable selection

To predict the Adjusted Sale Price of houses, we considered three major factors:

- the neighbourhood;
- the size of the house; and
- the quality of the construction.

Figure 1 below summarizes the variables we selected to represent each of those characteristics and their relationship.

```
pairs(train[c(9,12,16,22,24)])
```

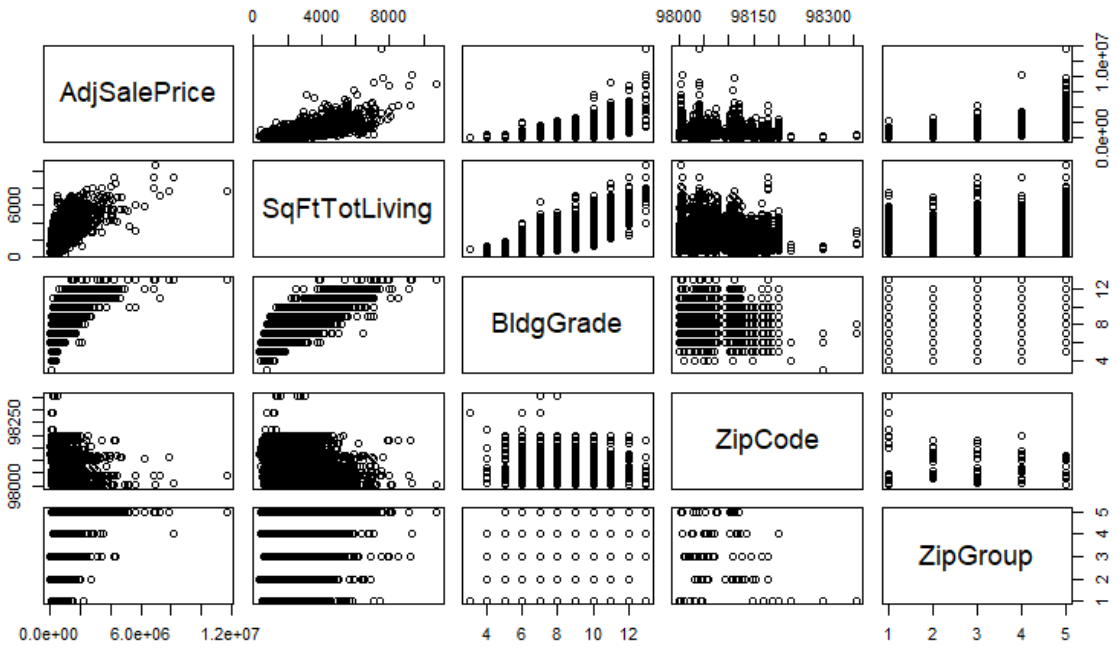

Figure 1: Associations of `AdjSalePrice` with `SqFtTotLiving`, `BldgGrade`, and `ZipCode`

`ZipCode` represents **neighborhood**. We grouped zip codes based on house prices in the variable `ZipGroup` to reflect the socioeconomic profile of each neighborhood.

```
zip_group_pr <- train %>%
  group_by(ZipCode) %>%
  summarise(med_price = median(AdjSalePrice),
            count = n()) %>%
  arrange(med_price) %>%
  mutate(cumul_count = cumsum(count),
         ZipGroup = ntile(cumul_count, 5))

train <- train %>%
  left_join(select(zip_group_pr, ZipCode, ZipGroup), by = "ZipCode")
```

`SqFtTotLiving` represents the **size of the house**. We avoided adding to the model the variables `SqFtLot`, `SqFtFinBasement`, `NbrLivingUnits` `Bathrooms`, and `Bedrooms` because they are also related to the size of the construction. That decision allowed us to reach a more **parsimonious** model, preventing us from loosing degrees of freedom and from incurring in issues related to **collinearity**. The risk of collinearity among those variables is visible in Figure 2 below:

```
pairs(train[c(12,11,13,14,15)])
```

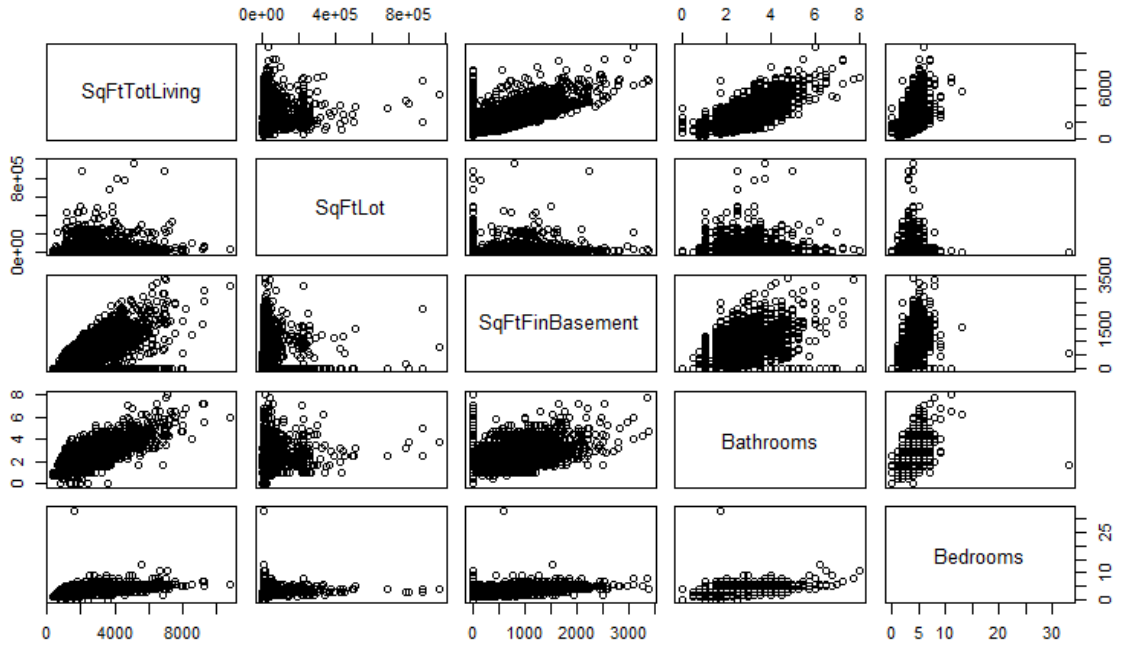

Figure 2: Risk of collinearity among `SqFtLot`, `SqFtFinBasement`, `Bathrooms`, and `Bedrooms`

Lastly, `BldgGrade` represents the **quality of the construction**. For the same reasons stated above (parsimony, preventing the loss of degrees of freedom, and avoiding collinear-

ity issues), we decided not to add `YrBuilt`, `YrRenovated`, and `NewConstruction`, which are also related to the quality of the construction.

2 The intuition behind the interaction effect

To complete the model, we added an interaction effect between `ZipGroup` and `SqFtTotLiving`. The interaction reflects the idea that each additional square foot in the building will affect house prices differently depending on the neighborhood. As a consequence, the slope of `SqFtTotLiving` on `AdjSalePrice` in expensive neighborhoods will be steeper than in popular ones (as can be seen in Figure 3).

```
ggplot(train, aes(SqFtTotLiving, AdjSalePrice, group = ZipGroup)) +
  geom_point(aes(colour = ZipGroup)) +
  geom_smooth(method = "lm", aes(colour = ZipGroup))
```

Figure 3: The effect of `SqFtTotLiving` on `AdjSalePrice` by `ZipGroup`

3 Final model

The effects of `SqFtTotLiving`, `BldgGrade`, `as.factor(ZipGroup)`, and `SqFtTotLiving:ZipGroup` on `AdjSalePrice`

```
mod1 <- lm(AdjSalePrice ~ SqFtTotLiving, data = train)
mod2 <- lm(AdjSalePrice ~ BldgGrade, data = train)
mod3 <- lm(AdjSalePrice ~ ZipGroup, data = train)
mod4 <- lm(AdjSalePrice ~ SqFtTotLiving + BldgGrade + as.factor(ZipGroup) +
    SqFtTotLiving:ZipGroup,
          data = train,
          na.action = na.omit)
```

```
stargazer(mod1, mod2, mod3, mod4)
```

Table 1: The effects of `SqFtTotLiving`, `BldgGrade`, `as.factor(ZipGroup)`, and `SqFtTotLiving:ZipGroup` on `AdjSalePrice`

	Dependent variable:			
	AdjSalePrice			
	(1)	(2)	(3)	(4)
SqFtTotLiving	294.357***			−45.169***
	(2.132)			(5.890)
BldgGrade		221,513.500***		74,339.440***
		(1,695.289)		(2,324.481)
ZipGroup			134,825.800***	
			(1,767.656)	
as.factor(ZipGroup)2				−60,600.140***
				(6,261.020)
as.factor(ZipGroup)3				−112,855.500***
				(7,949.155)
as.factor(ZipGroup)4				−168,392.000***
				(9,795.236)
as.factor(ZipGroup)5				−257,812.900***
				(13,626.880)
SqFtTotLiving:ZipGroup				64.755***
				(1.442)
Constant	−47,126.100***	−1,136,579.000***	138,024.100***	−232,072.400***
	(4,843.068)	(13,173.570)	(6,085.377)	(15,604.750)
Observations	20,340	20,340	20,340	20,340
R^2	0.484	0.456	0.222	0.622
Adjusted R^2	0.484	0.456	0.222	0.621
Residual Std. Error	278,257.400	285,528.100	341,480.700	238,266.300
	(df = 20338)	(df = 20338)	(df = 20338)	(df = 20332)
F Statistic	19,053.750***	17,073.130***	5,817.690***	4,770.375***
	(df = 1; 20338)	(df = 1; 20338)	(df = 1; 20338)	(df = 7; 20332)
Note:				*p<0.1; **p<0.05; ***p<0.01