Lecture 5: Adventures in Feature Engineering

Feature Engineering

- Identifying Informative Inputs
- Shaping the Feature to Match the Question
- Being Practical with the Available Information

Where the Rubber Meets the Road

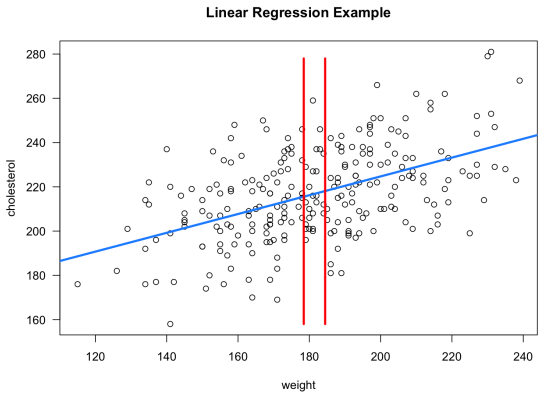
Feature Engineering…

- Is not always straightforward
- Has many practical challenges
- Also has many opportunities!

A Mirror to Data Cleaning

- Feature Engineering often reveals additional data cleaning that needs to be done.
- Every variable has to be reinspected in light of your plans.
- Even perfectly clean, understandable variables may require additional restructuring – or simply more thinking – before they are incorporated into the analyses..

In the Context of Linear Regression

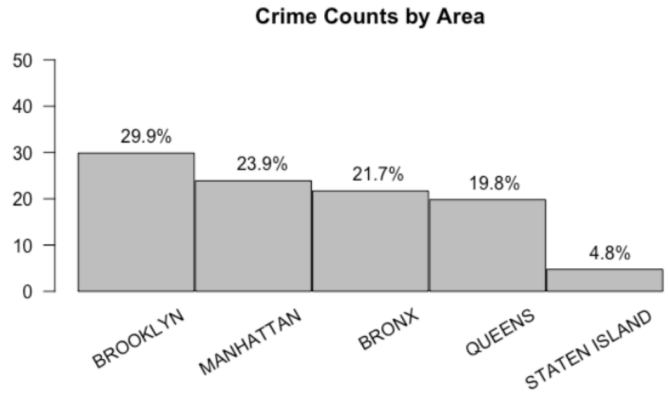


- We are assuming some degree of familiarity with the theory and applications of linear and logistic regression.
- The lessons we'll learn here also apply to other machine learning techniques.

Shallow Dive #1: Crime in NYC

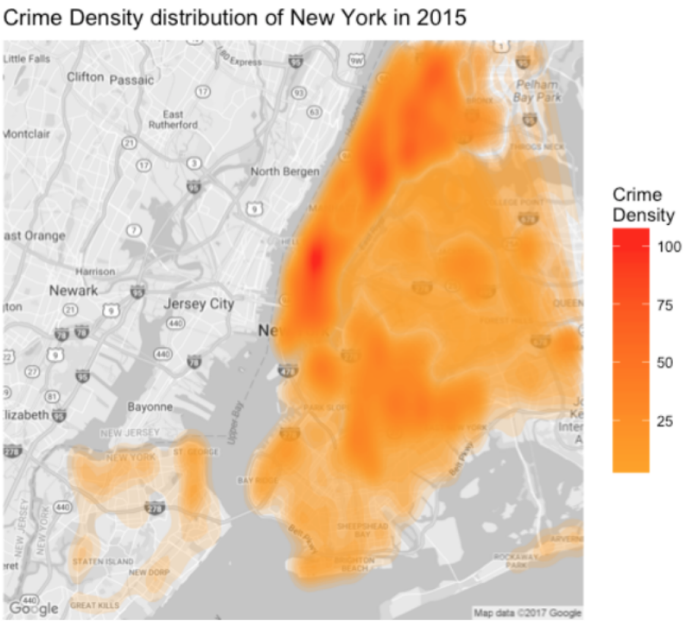
- In a previous class, a group of students put together a project analyzing the patterns of crime in New York City.
- **Data Source**: https://data.cityofnewyork.us/Public-Safety/NYPD-Complaint-Data-Historic/qgea-i56i
- NYC's OpenData portal would be one potential source of data to explore for your own final project: https://data.cityofnewyork.us/browse

Crime Rates by Borough in NYC



- The finding of this group's preliminary report: **Brooklyn** has the most crime, more than any other borough of New York City.

A Heat Map



- This would seem to contradict the notion that Brooklyn has the most crime.

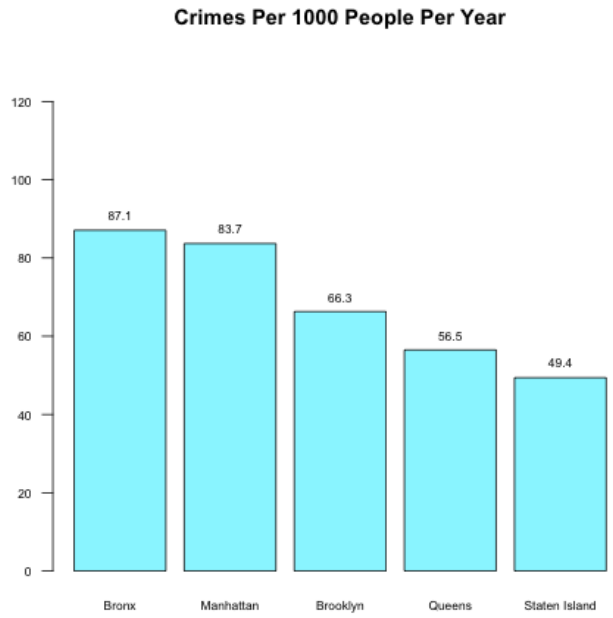
Which One is Right?

- **They both are!** But they are each measuring different things.
- Brooklyn has **the most crime overall**.
- Areas of Manhattan and the Bronx have higher densities of crime.

But They Can’t Both Be Right

- In judging the risks of living in different geographic regions, we don’t really care about the **total volume of crime**. Instead, we are concerned about the **rate of crime**.
- Brooklyn has the most crime of any borough, but **it also has the largest population** of any borough, too.
- Areas of different sizes can be better compared by looking at the per-capita rates.

Normalized Rates



How Do We Spot These Issues?

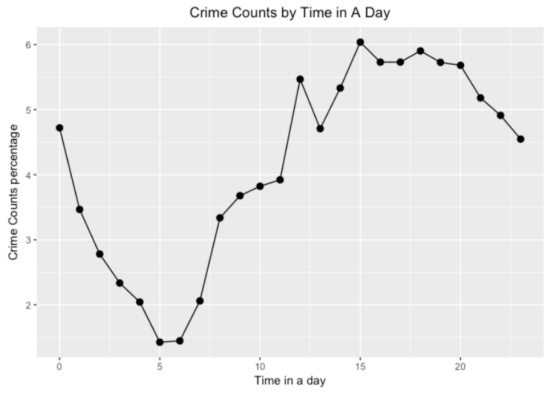
There are a variety of different ways in which one might investigate a claim. Maintaining a healthy skepticism, especially about novel results, is usually a good practice. In this case, I noticed the issue right away due to a few reasons:

- I have some **direct experience** with living and working in all of these geographic areas.
- I was also aware of the **differences in area and population** for these regions. Staten Island especially is small relative to the other boroughs.

What About the Next Disconnection?

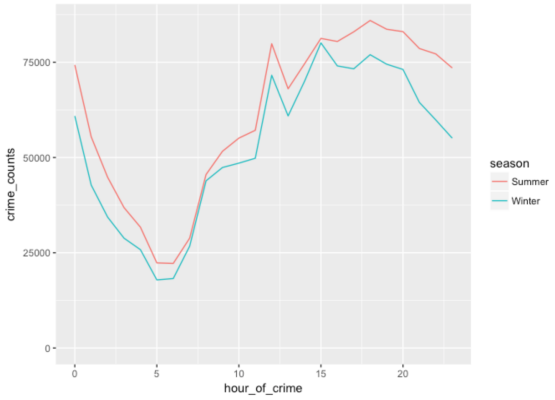
- You never know when you might be producing an erroneous conclusion based on a single graph, table, or result.
- Everything that you produce is worthy of scrutiny. If a result seems strange, it's worthy of investigation. This may be due to a coding mistake or a misinterpretation.
- Asking your colleagues for a review can be especially beneficial. They can spot results that you might have missed in the fog of producing them.
- Don't speed ahead too quickly with a result. It's always better to verify its accuracy in a number of ways before sharing what you've found.

Crime Patterns by Time of Day



- Reported crimes followed an interesting pattern.
- The conventional wisdom that crime is more frequent at night is in line with the observed trends.

An Interesting Split



- Compared to the winter, it would appear that longer days in the summer are associated with later spikes in reported crimes.
- For predicting the patterns of crime, we might conclude from this observation that there may be an **interaction between the time and the seasonality**.
- How should we account for the possibility of an interaction between these variables?

Approaches to Interactions

- The classic form of an interaction would be to include an **interaction term** within the model. For a linear regression model, this may have the form of, for example:

Average Number of Crimes in Hourly Time Period = B_0 **+** B_1 *** Hour of the day +** B_2 *** Season +** B_3 *** (Hour of the day * Season) + …**

- **Alternative approach #1**: Use sunset data to create a categorical variable to reflect whether the time period is at night.

- **Alternative approach #2**: Calculate an **Hours After Sunset** variable that calculates the time relative to the sunset.

These are merely a few examples. You could pursue a variety of alternative approaches.

But What is the Best Feature?

- In the case of predicting the rate of crime, some ways of measuring time and seasonality will lead to more informative results than others.
- How you incorporate the features may depend on the goals of the analysis. A simple feature that's easy to understand might be best for studies that prioritize the interpretation, while a more complicated feature may have more predictive power.
- Without experimentation, it's impossible to know whether one feature works more effectively than the alternative ways to incorporate the information.

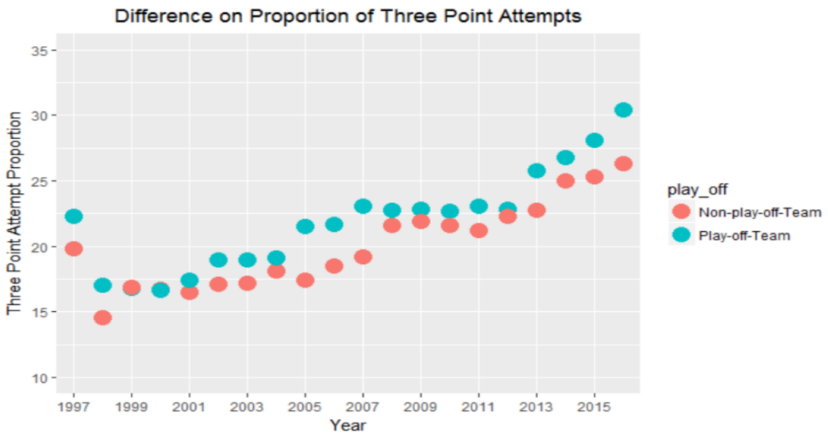
Shallow Dive #2: Basketball

- Analytics have changed the strategy of how basketball is played over time.
- Shot attempts with the highest expected production are increasingly emphasized.
- The **three point** shot – which features greater risks and greater rewards than closer two point shots – was found to be undervalued.

A Research Question

- A project group from an earlier class worked on a simple question: does attempting a greater volume of 3 point shots make a team more successful?
- The outcome: a binary indicator of **whether each team made the playoffs** (roughly the top 16 out of 30 teams) in the season.
- The inputs: 3 point attempts per game and other associated data about each basketball team's performance. Data were gathered from historical NBA records.

Trends in 3 Point Attempts



- Year after year, the successful playoff-bound teams shot more three point attempts on average relative to the less successful teams.
- So imagine my surprise when I saw the following model output:

A Model of Making the Playoffs

Logistic Model - Play Off

	rn	Estimate	Std. Error	z value	Pr(>|z|)	Odds.Ratio	OR.Lower.95	OR.Upper.95
1	(Intercept)	22.062	5.072	4.35	0	3812315654.966	183752.389	79094213238633.5
2	Offensive_Efficiency	0.196	0.031	6.251	0	1.216	1.144	1.293
3	Defensive_Efficiency	-0.29	0.033	-8.673	0	0.749	0.701	0.799
4	Pace	-0.122	0.042	-2.866	0.004	0.886	0.815	0.962
5	Three_Point_Field_Goal_Attempts	0.001	0	3.837	0	1.001	1.001	1.002
6	Two_Point_Field_Goal_Attempts	0	0	-1.622	0.105	1	0.999	1
7	Free_Throw_Attempts	0.002	0.001	4.093	0	1.002	1.001	1.003
8	Personal_Fouls	-0.003	0.001	-3.261	0.001	0.997	0.996	0.999

- At face value, this model would suggest that every additional 10 three point attempts per game would improve a team's chances to make the playoffs by **1%**.

A Variety of Issues

- The Intercept term looks unreasonably high.
- Three Point Attempts and Offensive Efficiency are highly related factors.
- Success and its relationship to three point attempts is relative.

Each of these issues is worthy of some additional discussion.

Issue #1: A High Intercept

- The intercept term for the logistic regression had an estimate of **22.062**, which translated to an astronomical Odds Ratio of **381 million**.
- The starting point for the model (before considering any additional covariates) would be an estimate of **exp(22.062) / (1 + exp(22.062))**, which is effectively a 100% chance of success.
- However, the coefficients of **pace, offensive efficiency, and defensive efficiency**, with typical values of 100-120, would effectively drive the results up or down sufficiently to get reasonable estimates on a team's chances of reaching the playoffs.

Overall, the high intercept was a cause for concern, but its value could be explained relative to some of the other terms.

Issue #2: Offensive Efficiency and Three Point Attempts

- Offensive efficiency – roughly a team's average points scored per 100 posessions – encompasses all of the shots a team takes.
- The information contained in the number of three point attempts is likely to be strongly correlated with the offensive efficiency.
- Recall that three pointers increased because they were found to be efficient ways of scoring points!

By including both offensive efficiency and most of its components – 3 point, 2 pointers, and free throws – we created a situation in which the effect of additional three point attempts was distributed in part to the estimate for offensive efficiency.

Issue #3: How Should We Measure Success?

- From the earlier graph, it's clear that **three point attempts increased over time** for all of the teams.
- However, making the playoffs is a matter of a team's **relative success** against that year's competition.
- What is important is **shooting more three pointers than your competition**, not more than a historical benchmark.

Comparing Factors Over Time

- Because of these historical trends, modern success is not directly comparable with historical success in terms of absolute numbers.
- Instead, it would make more sense to compare the performances across different eras in relative terms.
- For this reason, I would suggest to the team that they measure how each team attempted three pointers in terms of their **standard deviations above average** relative to that year's competition.

Then we could more reasonably answer the research question by estimating **the effect of attempting an additional standard deviation above average** in the number of three pointers relative to other teams on the likelihood that a team would make the playoffs.

Shallow Dive #3: Medicines and Outcomes

- **How effective is a medicine?** One way to examine this question is to compare similar patients who did and did not take the medicine and look at their outcomes over time.
- **The medicine's effect** should show up as a regression estimate that associates reduced outcomes with taking the medicine.
- However, in looking at the data, we had a **specific concern** in mind.

Predicting Adverse Medical Events

- What is the strongest predictor of adverse medical events such as mortality?
- You might guess a variety of factors such as obesity, diabetes, cardiovascular disease, smoking, etc.
- These are good all good guesses! But there is one factor that outpaces everything else.

The Strongest Predictor

- **Going to the hospital** is the strongest predictor of mortality. After all, the vast majority (60%) of patients die in the hospital (https://palliative.stanford.edu/home-hospice-home-care-of-the-dying-patient/where-do-americans-die/).
- Meanwhile, an overwhelming majority of patients out of the hospital are not likely to pass away in the short term.
- If misinterpreted, it would appear as if going to the hospital is the reason why patients pass away.

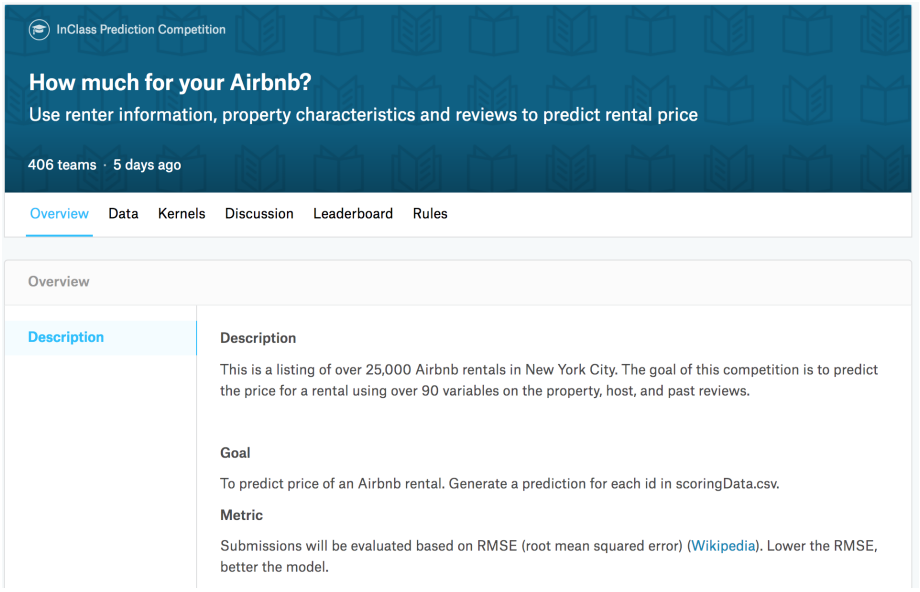
Reverse Causality

- Hospitalization is not the factor that makes patients pass away.
- Instead, hospitalization is a lagging indicator of some other serious medical problem.
- **Correlation does not imply causation**: we should be careful to interpret the results of a regression model and not to unreasonably infer causal relationships.

Reverse Causality and Medicine

- Taking an appropriate medicine **should reduce the risk** of adverse events.
- However, a patient who **should have been taking the medicine all along** might only start doing so when the situation becomes dire.
- Meanwhile, patients who are doing well may no longer need the medicines.
- The combination of these factors could lead to **imbalances in the cohorts** who are and are not taking the medicine. These imbalances may lead us to improperly conclude that a medicine has a harmful effect.

Deep Dive: The AirBnB Data



Goal: Use machine learning models to accurately predict the prices of new listings based on their measured features and historical pricing data.

The Data

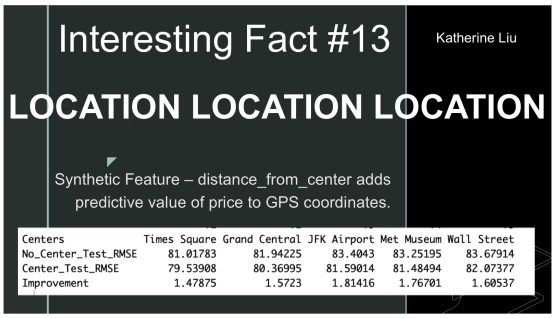
```
library(data.table)
dat <- fread(input = "AirBnB analysisData.csv", sep = ",",
    fill = TRUE)
dim(dat)
```

```
[1] 29142    96
```

```
test <- fread(input = "AirBnB scoringData.csv", sep = ",",
    fill = TRUE)
dim(test)
```

```
[1] 7286   95
```

Examples of Features



A Cozy Studio in Midtown, Close to Everything

Meanwhile, you might be surprised how many listings are described as **cozy**:

```
description.name <- "description"
price.name <- "price"
cozy.name <- "cozy"
pattern.cozy <- "cozy"
dat[, eval(cozy.name) := 0]
w <- grep(pattern = pattern.cozy, x = dat[, tolower(get(description.name))])
dat[w, eval(cozy.name) := 1]
dat[, mean(get(cozy.name))]
```

```
[1] 0.182623
```

```
dat[, .(`Mean Price` = mean(get(price.name))), by = cozy.name]
```

```
   cozy Mean Price
1:    1   115.0840
2:    0   136.9762
```

Based upon my own experience in renting apartments from online listings, **cozy is a nice way of saying the place is small**.

However, Challenges are Everywhere

- Investigating the data
- Troubleshooting models
- Unexpected surprises

The following examples are meant to illustrate some of the challenges the students faced in putting together their predictive models.

Challenge #1: Formulae

A number of students initially thought the competition would be as simple as fitting the following model:

```
mod = lm(formula = "price ~ .", data = dat)
```

The idea of this model is that the **outcome** of **price** should be modeled using **all of the other measured variables** in the data set.

However, there are some issues to think about before diving in.

Issues with the . Notation

- This assumes that you want all of the variables in as predictors in the model, no matter what.
- If variables are added or removed in a new version of the data set, you may not even notice this.
- Likewise, any changes you make to the structure (such as adding a new variable) would also bring in this variable as a new predictor.

A Lack of Accountability

- Any use of the . notation makes an assumption about the full structure of the data.
- This can work in limited contexts, but it is not robust to changes.
- In many ways, using the . notation is like hard-coding a constant. If you refer to a column by its index, e.g. **dat[, 2]**, then what happens if a new version of the data shifts the columns around?

More Issues with a Formula's Notation

- Writing the variable names of a formula by hand: what if you make a typo?
- Handling variable names with spaces: These names must be inside of accent marks: `This Name`
- What if the variable name doesn't exist in the data?
- What if you accidentally include a column name twice?
- What if you want to search for variables based on a pattern?

Creating Better Formulae

```
create.formula <- function(outcome.name, input.names, input.patterns = NA,
    all.data.names = NA, return.as = "character") {
    variable.names.from.patterns <- c()
    if (!is.na(input.patterns[1]) & !is.na(all.data.names[1])) {
        pattern <- paste(input.patterns, collapse = "|")
        variable.names.from.patterns <- all.data.names[grep(pattern = pattern,
            x = all.data.names)]
    }
    all.input.names <- unique(c(input.names, variable.names.from.patterns))
    all.input.names <- all.input.names[all.input.names !=
        outcome.name]
    if (!is.na(all.data.names[1])) {
        all.input.names <- all.input.names[all.input.names %in%
            all.data.names]
    }
    input.names.delineated <- sprintf("`%s`", all.input.names)
    the.formula <- sprintf("`%s` ~ %s", outcome.name, paste(input.names.delineated,
        collapse = " + "))
    if (return.as == "formula") {
        return(as.formula(the.formula))
    }
    if (return.as != "formula") {
        return(the.formula)
    }
}
```

Example #1: Removing a Bogus Name

```
num.bedrooms.name <- "bedrooms"
num.bathrooms.name <- "bathrooms"
bogus.name <- "Column not in the data"

the.formula.1 <- create.formula(outcome.name = price.name,
    input.names = c(num.bedrooms.name, num.bathrooms.name,
        bogus.name), all.data.names = names(dat))
print(the.formula.1)
```

```
[1] "`price` ~ `bedrooms` + `bathrooms`"
```

Example #2: Finding Patterns

```
pattern.review.scores <- "review_scores_"

the.formula.2 <- create.formula(outcome.name = price.name,
    input.names = c(num.bedrooms.name, num.bathrooms.name),
    input.patterns = pattern.review.scores, all.data.names = names(dat))

print(strwrap(x = the.formula.2))
```

```
[1] "`price` ~ `bedrooms` + `bathrooms` + `review_scores_rating` +"
[2] "`review_scores_accuracy` + `review_scores_cleanliness` +"
[3] "`review_scores_checkin` + `review_scores_communication` +"
[4] "`review_scores_location` + `review_scores_value`"
```

Example #3: Column Names with Spaces

```
square.feet.name <- "square_feet"
square.meters.name <- "Square Meters"
dat[, `:=`(eval(square.meters.name), get(square.feet.name)/10)]

the.formula.3 <- create.formula(outcome.name = price.name,
    input.names = c(num.bedrooms.name, num.bathrooms.name,
        square.meters.name), all.data.names = names(dat))

print(the.formula.3)
```

```
[1] "`price` ~ `bedrooms` + `bathrooms` + `Square Meters`"
```

Without placing the names of the variables within accent marks, the formula would generate an error when trying to fit a model.

Example #4: Removing Duplicates

Here we'll accidentally include the number of bedrooms in the formula twice:

```
the.formula.4 <- create.formula(outcome.name = price.name,
    input.names = c(num.bedrooms.name, num.bathrooms.name,
        square.meters.name, num.bedrooms.name), all.data.names = names(dat))

print(the.formula.4)
```

```
[1] "`price` ~ `bedrooms` + `bathrooms` + `Square Meters`"
```

The duplicated names are automatically removed.

Challenge #2: Contrasts

Meanwhile, when the students did try to fit this model:

```
mod = lm(formula = "price ~ .", data = dat)
```

… they were surprised to find that it generated the following error message:

```
> mod = lm(formula = "price ~ .", data = dat)
Error in `contrasts<-`(`*tmp*`, value = contr.funs[1 + isOF[nn]]) :
  contrasts can be applied only to factors with 2 or more levels
```

A Lack of a Contrast

- Some input variables in the data set only have a single value.
- Without variation in an input, a linear model cannot estimate the impact of changes in the input on changes in the outcome.
- Realistically, we'll have to drop these variables from the model.

Identifying Variables that Lack Contrasts

```
length.unique <- function(x) {
    return(length(unique(x)))
}
unique.values.tab <- dat[, .(Variable = names(dat), Num_Unique = as.numeric(lapply(X = .SD,
    FUN = "length.unique")))]
# setorderv(x = unique.values.tab, cols = 'Num_Unique',
# order = 1)
library(DT)
datatable(data = unique.values.tab[Num_Unique == 1, ], rownames = FALSE)
```

Show 10 entries Search:

Variable	Num_Unique
scrape_id	1
experiences_offered	1
thumbnail_url	1
medium_url	1
xl_picture_url	1
host_acceptance_rate	1
has_availability	1
requires_license	1
license	1

Showing 1 to 9 of 9 entries Previous 1 Next

Taking a Look at the Singular Values

```
dat[1, .SD, .SDcols = unique.values.tab[Num_Unique == 1,
    Variable]]
```

```
        scrape_id experiences_offered thumbnail_url medium_url
1: 20180303203649                none          <NA>       <NA>
   xl_picture_url host_acceptance_rate has_availability requires_license
1:           <NA>                  N/A                t                 f
   license
1:      NA
```

Information Not Available

- We have no listing that include **supplementary experiences** (e.g. tours, cultural events, etc.).
- There are **no measured host acceptance rates**.
- **Licence** information, which might impact how professionally the rental is managed, is also not collected.

Each of these factors could plausibly impact the price of a listing, but we have no way of estimating their effect due to a lack of measured data.

Removing Variables

- The earlier approach of counting the unique values can help identify the variables to remove.
- If you simply rely on the error messages that are generated (e.g. by starting with every variable in the model), then you'll only find out about a lack of contrast when you go one at a time.
- Likewise, we may also choose to remove variables that have **too many** categories.

Reducing the Formula

```
reduce.formula <- function(dat, the.initial.formula, max.categories = NA) {
    require(data.table)
    dat <- setDT(dat)

    the.sides <- strsplit(x = the.initial.formula, split = "~")[[1]]
    lhs <- trimws(x = the.sides[1], which = "both")
    lhs.original <- gsub(pattern = "`", replacement = "",
        x = lhs)
    if (!(lhs.original %in% names(dat))) {
        return("Error:  Outcome variable is not in names(dat).")
    }

    the.pieces.untrimmed <- strsplit(x = the.sides[2], split = "+",
        fixed = TRUE)[[1]]
    the.pieces.untrimmed.2 <- gsub(pattern = "`", replacement = "",
        x = the.pieces.untrimmed, fixed = TRUE)
    the.pieces.in.names <- trimws(x = the.pieces.untrimmed.2,
        which = "both")

    the.pieces <- the.pieces.in.names[the.pieces.in.names %in%
        names(dat)]
    num.variables <- length(the.pieces)
    include.pieces <- logical(num.variables)

    for (i in 1:num.variables) {
        unique.values <- dat[, unique(get(the.pieces[i]))]
        num.unique.values <- length(unique.values)
        if (num.unique.values >= 2) {
            include.pieces[i] <- TRUE
        }
        if (!is.na(max.categories)) {
            if (dat[, is.character(get(the.pieces[i])) |
                is.factor(get(the.pieces[i]))] == TRUE) {
                if (num.unique.values > max.categories) {
                  include.pieces[i] <- FALSE
                }
            }
        }
    }
    pieces.rhs <- sprintf("`%s`", the.pieces[include.pieces ==
        TRUE])
    rhs <- paste(pieces.rhs, collapse = " + ")
    the.formula <- sprintf("%s ~ %s", lhs, rhs)
    return(the.formula)
}
```

Example: Removing Variables with a Lack of Contrasts

```
medium.url.name <- "medium_url"
orig.formula.5 <- create.formula(outcome.name = price.name,
    input.names = c(num.bedrooms.name, num.bathrooms.name,
        medium.url.name))
formula.5 <- reduce.formula(dat = dat, the.initial.formula = orig.formula.5)
print(formula.5)
```

```
[1] "`price` ~ `bedrooms` + `bathrooms`"
```

- The **medium_url**, with a single unique value across the full range of data, was removed from the formula.

Challenge #3: Too Many Contrasts

```
datatable(data = unique.values.tab[Num_Unique >= 20000,
    ], rownames = FALSE)
```

Show 10 entries Search:

Variable	Num_Unique
id	29142
listing_url	29142
name	28770
summary	27497
space	21254
description	28805
picture_url	29134
host_id	24731
host_url	24731
host_thumbnail_url	24674

Showing 1 to 10 of 14 entries Previous 1 2 Next

What Generates So Many Contrasts?

- Some variables are unique in every entry – or nearly each one.
- Is this a problem? Not for numeric variables!
- However, categorical variables with too many categories can have major issues.

Categorical Variables in Linear Models

- Indicator coding: A new column (1 or 0) for each value after the base case – and a new coefficient.
- Less Data for each column: A unique indicator (like the listing URL) will base an estimate of the average price difference on a single data point.
- Inferences without a meaning: Even if we can get a coefficient, why are we comparing the differences between the listing URLs on price?

Example: Removing Variables with Too Many Categories

```
orig.formula.6 <- create.formula(outcome.name = price.name,
    input.names = c(num.bedrooms.name, num.bathrooms.name,
        description.name))
formula.6 <- reduce.formula(dat = dat, the.initial.formula = orig.formula.6,
    max.categories = 30)
print(formula.6)
```

```
[1] "`price` ~ `bedrooms` + `bathrooms`"
```

- The description variable, with 28805 free text responses, was removed from consideration.

Tips for Handling Categorical Variables

- Beware of free text responses.
- Consolidate the categories into a smaller, more meaningful collection.
- Extract useful information (like the **cozy** variable) from free text variables like the **description**.

Challenge #4: How Granular Should the Categories Be?

The location data for each listing in New York City is categorized across a variety of variables:

- **Borough:** neighbourhood_group_cleansed
- **Neighbourhood**: neighbourhood_cleansed
- **Zip Code**: zipcode
- **Coordinates**: latitude and longitude

Each of these variables provides a different degree of precision about the information. How do we decide which factors to incorporate?

Considerations for Selecting the Granularity

- **Sample sizes** can greatly impact what can and can not be considered.
- Less granularity provides more context and a larger area for finding similar cases.
- More granularity will provide a greater degree of customization to the specific circumstances.

Issues of Sample Size

- A **categorical label of geography** – like neighborhoods or zip codes – will consider all subjects within a category as reasonably comparable.
- **Categories that are too small** – like zip codes that only have a handful of listings for rentals – become unusable. There need to be enough similar listings within the category to estimate the average of the group.
- **Categories that are too large** – like the boroughs of New York City – will have too many listings that span a wide range of geographic patterns. When the area considered is too wide, the average of the group can be too general to describe some of the subgroups it contains.

How to proceed is highly dependent on the context of the problem. There is no absolute answer for the correct sample size or degree of granularity.

Interpretative Quality Versus Predictive Accuracy

- Less granular information leads to improved interpretations. This can give you a notion of **neighborhood** or **borough**-level effects.
- Knowing the general vicinity of a listing's location can provide insights that are more relatable to people who are making decisions. Seeing that certain neighborhoods or boroughs have higher prices can better line up with a person's understanding.
- Knowing a highly specific location can provide greater accuracy in making predictions. There can be geographic patterns within a neighborhood that influence the price of the rentals in a highly localized manner.

Challenge #5: Other Levels of Granularity

- What is measured already provides different levels of information about the location.
- However, there are other factors to consider.
- Some geographically disconnected neighborhoods have other commonalities.

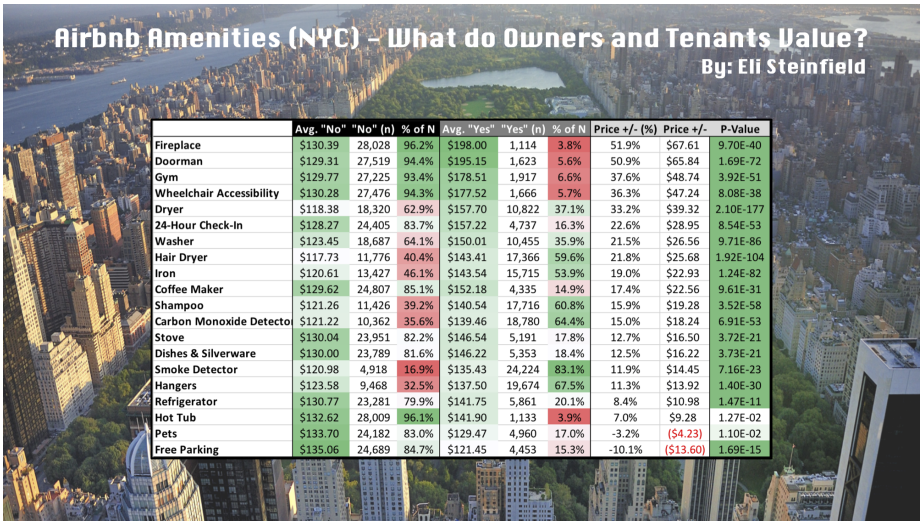
Commonalities in Neighborhoods:

- **Cool neighborhoods**: places that are considered hip to live in.
- **Wealthy neighborhoods**: places that typically have more wealthy residents.
- **Walkable neighborhoods**: locations with a high degree of nearby commerce and population density.
- **Bedroom communities**: locations that are considered quiet and relatively empty during the day because most of the residents commute elsewhere.

You Can Create Your Own Features

- This is an opportunity to utilize your expertise about the domain – or to ask other members of your team for input.
- My idea about classifying rentals as **cozy** or not turned out to be a significant predictor of pricing trends.
- Meanwhile, sharing that idea with my earlier class sparked considerable interest in identifying and investigating other possible features from the free-text responses.

Additional Features



	Avg. "No"	"No" (n)	% of N	Avg. "Yes"	"Yes" (n)	% of N	Price +/- (%)	Price +/-	P-Value
Fireplace	$130.39	28,028	96.2%	$198.00	1,114	3.8%	51.9%	$67.61	9.70E-40
Doorman	$129.31	27,519	94.4%	$195.15	1,623	5.6%	50.9%	$65.84	1.69E-72
Gym	$129.77	27,225	93.4%	$178.51	1,917	6.6%	37.6%	$48.74	3.92E-51
Wheelchair Accessibility	$130.28	27,476	94.3%	$177.52	1,666	5.7%	36.3%	$47.24	8.08E-38
Dryer	$118.38	18,320	62.9%	$157.70	10,822	37.1%	33.2%	$39.32	2.10E-177
24-Hour Check-In	$128.27	24,405	83.7%	$157.22	4,737	16.3%	22.6%	$28.95	8.54E-53
Washer	$123.45	18,687	64.1%	$150.01	10,455	35.9%	21.5%	$26.56	9.71E-86
Hair Dryer	$117.73	11,776	40.4%	$143.41	17,366	59.6%	21.8%	$25.68	1.92E-104
Iron	$120.61	13,427	46.1%	$143.54	15,715	53.9%	19.0%	$22.93	1.24E-82
Coffee Maker	$129.62	24,807	85.1%	$152.18	4,335	14.9%	17.4%	$22.56	9.61E-31
Shampoo	$121.26	11,426	39.2%	$140.54	17,716	60.8%	15.9%	$19.28	3.52E-58
Carbon Monoxide Detecto	$121.22	10,362	35.6%	$139.46	18,780	64.4%	15.0%	$18.24	6.91E-53
Stove	$130.04	23,951	82.2%	$146.54	5,191	17.8%	12.7%	$16.50	3.72E-21
Dishes & Silverware	$130.00	23,789	81.6%	$146.22	5,353	18.4%	12.5%	$16.22	3.73E-21
Smoke Detector	$120.98	4,918	16.9%	$135.43	24,224	83.1%	11.9%	$14.45	7.16E-23
Hangers	$123.58	9,468	32.5%	$137.50	19,674	67.5%	11.3%	$13.92	1.40E-30
Refrigerator	$130.77	23,281	79.9%	$141.75	5,861	20.1%	8.4%	$10.98	1.47E-11
Hot Tub	$132.62	28,009	96.1%	$141.90	1,133	3.9%	7.0%	$9.28	1.27E-02
Pets	$133.70	24,182	83.0%	$129.47	4,960	17.0%	-3.2%	($4.23)	1.10E-02
Free Parking	$135.06	24,689	84.7%	$121.45	4,453	15.3%	-10.1%	($13.60)	1.69E-15

What About Free Parking?

- Listings that offered free parking had an average price of 121.45 dollars, while those not offering parking averaged 135.06.
- Why would **free parking be associated with lower prices**?
- Shouldn't people want to pay for that?

Who is Offering Parking, and Who Isn't?

- Relative to the rest of the United States, New York City offers an unusual degree of population density and public transportation.
- The highest priced listings are in Manhattan or other highly dense areas.
- By contrast, the outlying areas of New York City are more likely to include driveways, but the price of the rentals is lower due to their relative location.

Reverse Causality Strikes Again

- In nearly any location, free parking should either add to the price or at least not detract from it.
- This would be true even in the center of Manhattan.
- However, the confluence of parking and geographic desirability created imbalances in the cohorts that were and were not offering free parking.

If we simply went with the observed data, we would mistakenly conclude that free parking has a negative effect on the price of a listing.

Challenge #6: New Levels in the Testing Data

It would be reasonable to look for neighbourhood-level effects in pricing. Consider the following linear model:

```
neighborhood.name <- "neighbourhood_cleansed"
```

```
the.formula <- create.formula(outcome.name = price.name,
    input.names = neighborhood.name)
mod <- lm(formula = the.formula, data = dat)
pred <- predict(object = mod, newdata = test)
```

While the model was estimated properly, the predictions led to the following error message:

```
> mod <- lm(formula = "price ~ neighbourhood_cleansed", data = dat)
> pred <- predict(object = mod, newdata = test)
Error in model.frame.default(Terms, newdata, na.action = na.action, xlev = obj
ect$xlevels) :
  factor neighbourhood_cleansed has new levels Hollis Hills, Westerleigh
```

The New Level Problem

- For categorical variables, a linear model cannot predict for new levels: there is no coefficient for the new level and no data to base it on!
- Where possible, you can ensure that the training set is reasonably representative of the predictions to be made.
- However, new values emerge over time!

The same advice applies here: consolidating the categories or dropping the variables from the model may be necessary.

Identifying Variables with New Levels

```
find.new.levels <- function(old.dat, new.dat, variable.name) {
    require(data.table)
    setDT(old.dat)
    setDT(new.dat)
    old.levels <- old.dat[, unique(get(variable.name))]
    new.levels <- new.dat[!(get(variable.name) %in% old.levels),
        unique(get(variable.name))]
    return(new.levels)
}
find.new.levels(old.dat = dat, new.dat = test, variable.name = neighborhood.name)
```

```
[1] "Hollis Hills" "Westerleigh"
```

```
zipcode.name <- "zipcode"
find.new.levels(old.dat = dat, new.dat = test, variable.name = zipcode.name)
```

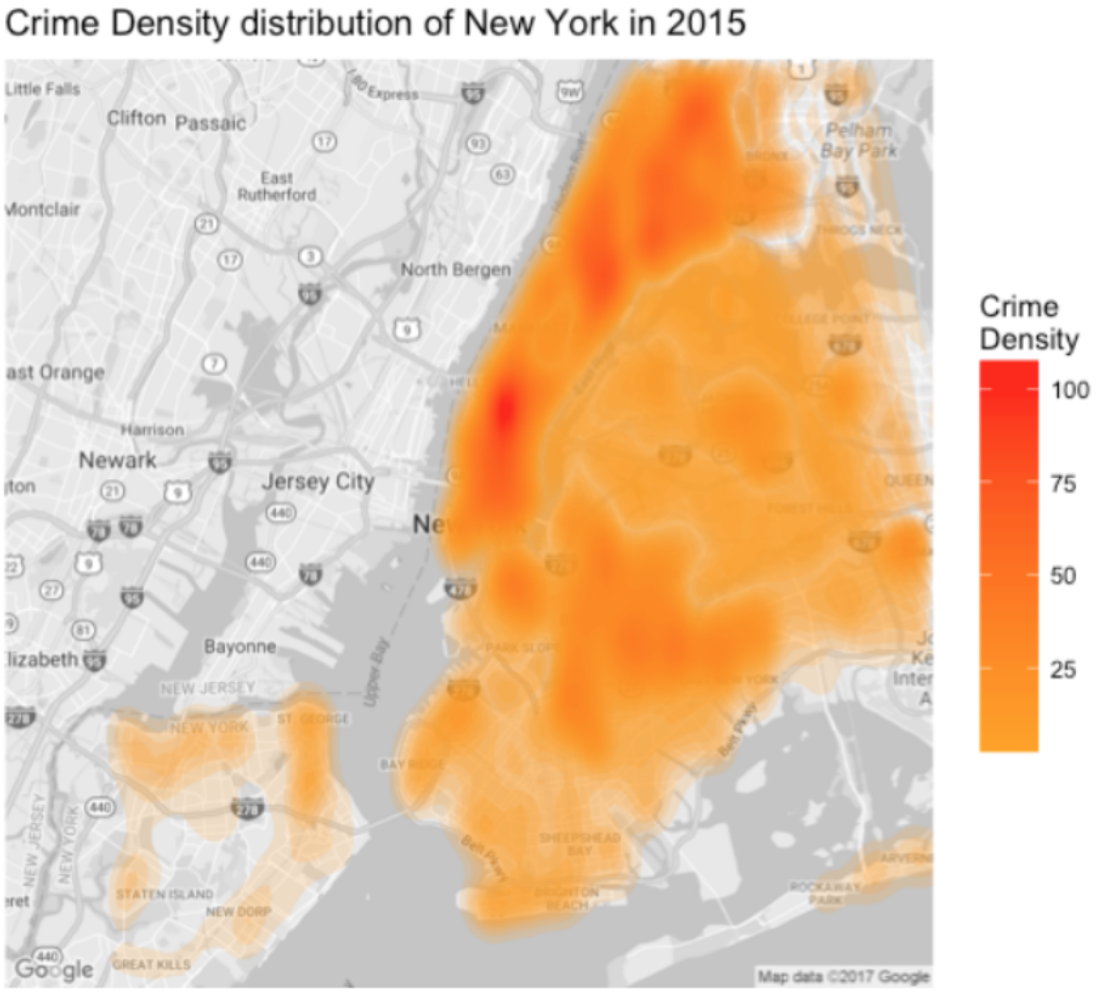
```
[1]    NA 10080 11363 10550
```

Challenge #7: Finding a Better Feature

- Neighbourhoods and Zip Codes have many categories overall and many new levels in the testing set.
- But **location, location, location**! We need to include geography somehow.
- Fortunately, the data also include the **latitude and longitude** of each listing.

Many of the top entries in the competition incorporated latitude and longitude. Especially using advanced tree methods, the models were able to make insightful splits that led to highly specific geographic predictions.

Non-Linear Geographical Effects



Features Driving Methods

- The effect of geography on the price **is highly non-linear** in New York City.
- Investigations that **prioritize accuracy** would favor tree-based models using the most granular information (**latitude** and **longitude**).
- Investigations that **prioritize interpretative qualities** would favor linear models using less granular information (e.g. boroughs or neighborhood groups).

Challenge #8: Number of Bedrooms

- Let's turn our attention to the number of bedrooms in each listing:

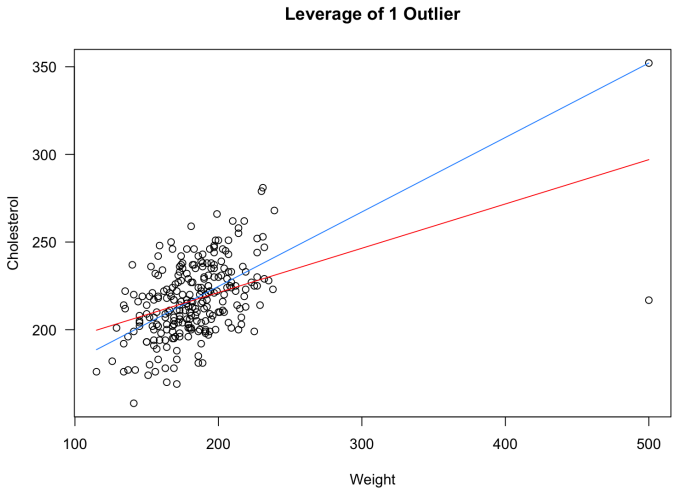
```
dat[, .N, keyby = num.bedrooms.name]
```

```
    bedrooms     N
 1:        0  2756
 2:        1 21062
 3:        2  3790
 4:        3  1165
 5:        4   282
 6:        5    58
 7:        6    14
 8:        7    10
 9:        8     3
10:        9     1
11:       11     1
```

Unusual Listings

- It is relatively rare to have more than a 2-bedroom apartment in many parts of New York City.
- Larger spaces certainly exist, but they are not as comparable to the other listings.
- We may also have the **reverse causality** problem here. An extremely large property with many bedrooms may be in outlying areas of New York City. Lower relative prices in these regions may lead to spurious conclusions about the effect of having more space in a listing.

Leverage Points



- A single outlying value can considerably move the overall line.
- Such a value is called a **leverage point** because it has a disproportionate impact on the overall regression.

Sensitivity Analyses

- How do these unusual listings impact the model?
- One way to investigate is to consider **sensitivity analyses** that impose small changes on the conditions to look for differences in the conclusions.
- These sensitivity analyses are performed **in addition to** the original model that was developed for comparative purposes.

There are a variety of approaches to consider:

Forms of Sensitivity Analyses

- **Capping the Values**: imposing a limit on the maximum number of bedrooms. Any listing with more than 4 bedrooms would have its value reduced to 4.
- **Restricted Attention**: Any unusual listing (e.g. with more than 4 bedrooms) could be dropped from the model.
- **More Specific Comparisons**: Perhaps it is also worth exploring more specific questions. For example, what is the average difference in price between **studios** and **1-bedroom** apartments? Meanwhile, keep in mind that the 0-bedroom listings can include both studios (full occupancy) and couchsurfing (shared occupancy) settings.

Challenge #9: Unusual Properties

- Let's take a look at some of the unusual property types with relatively few listings:

```
property.type.name <- "property_type"
property.counts <- dat[, .N, keyby = property.type.name]
property.counts[N <= 5, ]
```

```
             property_type N
 1:             Aparthotel 4
 2:                   Boat 4
 3:                  Cabin 1
 4:              Camper/RV 3
 5: Casa particular (Cuba) 1
 6:                 Castle 1
 7:                   Cave 1
 8:                 Chalet 1
 9:            Earth house 1
10:                 Island 1
11:                 Resort 3
12:                   Tent 1
13:             Tiny house 1
14:                  Train 1
15:              Treehouse 1
16:                   Yurt 1
```

A Tent, a Cave, a Castle, a Chalet, an Island, an Earth House

- These properties may or may not be serious listings. Maybe we should do an investigation!
- Even if they are valid, these properties are not at all representative of the typical renting experience in New York City.
- Any of the prior options – e.g. recategorizing or removal – are on the table for handling these unusual cases.

Issue #10: Seasonality

- It's reasonable to assume that the price of the listings might change according to **seasonal trends**.
- The summer, holidays, and special events that attract greater degrees of tourism are all likely to see higher prices for the listings.
- It's worth investigating the seasonal impact in the data we have.

Seasonality in the AirBnB Data

```
date.name <- "last_scraped"
dat[, .N, keyby = date.name]
```

```
   last_scraped     N
1:   2018-03-04 13022
2:   2018-03-05 15650
3:   2018-03-06   470
```

- These data were all collected over the course of a 3-day period in March of 2018.
- We won't be able to make any estimates about the seasonal trends from the data we've collected.

But Seasonality is Important

- Our model would likely improve with additional data about seasonality.
- Historical trends might also give us a better sense of how to forecast the future prices of these listings.
- It would make sense to gather a greater variety of data over a longer period of time.

So Let's Actually Fit a Linear Regression Model

- The **lm** command works reasonably well.
- However, it can help to move the output into a more usable format.
- Better yet, computing more summary measures (like confidence intervals) can also help.

Fitting Functions

```
linear.regression.summary <- function(lm.mod, digits = 3, alpha = 0.05) {
  lm.coefs <- as.data.table(summary(lm.mod)$coefficients, keep.rownames = TRUE)
  z <- qnorm(p = 1 - alpha/2, mean = 0, sd = 1)
  lm.coefs[, Coef.Lower := Estimate - z * `Std. Error`]
  lm.coefs[, Coef.Upper := Estimate + z * `Std. Error`]
  return(lm.coefs)
}
round.numerics <- function(x, digits){
  if(is.numeric(x)){
    x <- round(x = x, digits = digits)
  }
  return(x)
}
fit.model <- function(dat, the.formula, digits = 3) {
  mod <- lm(formula = the.formula, data = dat)
  mod.summary <- linear.regression.summary(lm.mod = mod, digits = digits)
  mod.summary.rounded <- mod.summary[, lapply(X = .SD, FUN = "round.numerics", digits = digits)]
  return(mod.summary.rounded)
}
```

A Parsimonious Model

- We'll start with a model meant to generate insights about boroughs:
- Let's restrict our attention to apartment, loft, or condominimium settings.
- We'll also limit the study to units with 4 or fewer bedrooms.

Fitting the Model

```
the.settings <- c("Apartment", "Loft", "Condominimium")
bedroom.limit <- 4
borough.name <- "neighbourhood_group_cleansed"
rating.name <- "review_scores_rating"
x.names <- c(num.bedrooms.name, borough.name, cozy.name,
    rating.name)
the.initial.formula <- create.formula(outcome.name = price.name,
    input.names = x.names, all.data.names = names(dat))
the.formula <- reduce.formula(dat = dat, the.initial.formula = the.initial.formula,
    max.categories = 30)
print(the.formula)
```

```
[1] "`price` ~ `bedrooms` + `neighbourhood_group_cleansed` + `cozy` + `review_scores_rating`"
```

```
mod <- fit.model(dat = dat[get(num.bedrooms.name) <= bedroom.limit &
    get(property.type.name) %in% the.settings], the.formula = the.formula)
```

Model Results

```
library(DT)
datatable(data = mod, rownames = FALSE)
```

Show 10 entries Search:

rn	Estimate	Std. Error	t value	Pr(>|t|)	Coef.Lower	Coef.Upper
(Intercept)	-64.426	7.592	-8.486	0	-79.305	-49.547
bedrooms	63.483	0.83	76.506	0	61.857	65.11
neighbourhood_group_cleansedBrooklyn	27.01	4.627	5.837	0	17.941	36.079
neighbourhood_group_cleansedManhattan	85.274	4.612	18.488	0	76.234	94.314
neighbourhood_group_cleansedQueens	11.618	4.944	2.35	0.019	1.929	21.307
neighbourhood_group_cleansedStaten Island	1.293	11.496	0.112	0.91	-21.238	23.824
cozy	-12.522	1.353	-9.253	0	-15.174	-9.869
review_scores_rating	0.794	0.064	12.373	0	0.668	0.92

Showing 1 to 8 of 8 entries Previous 1 Next

Cleaning the Table

```
old.variable.name <- "rn"
variable.name <- "Variable"
replacement.value.borough <- "Borough: "
setnames(x = mod, old = old.variable.name, new = variable.name)
mod[, `:=`(eval(variable.name), gsub(pattern = borough.name,
    replacement = replacement.value.borough, x = get(variable.name)))]
datatable(data = mod, rownames = FALSE)
```

Show 10 entries Search:

Variable	Estimate	Std. Error	t value	Pr(>|t|)	Coef.Lower	Coef.Upper
(Intercept)	-64.426	7.592	-8.486	0	-79.305	-49.547
bedrooms	63.483	0.83	76.506	0	61.857	65.11
Borough: Brooklyn	27.01	4.627	5.837	0	17.941	36.079
Borough: Manhattan	85.274	4.612	18.488	0	76.234	94.314
Borough: Queens	11.618	4.944	2.35	0.019	1.929	21.307
Borough: Staten Island	1.293	11.496	0.112	0.91	-21.238	23.824
cozy	-12.522	1.353	-9.253	0	-15.174	-9.869
review_scores_rating	0.794	0.064	12.373	0	0.668	0.92

Showing 1 to 8 of 8 entries Previous 1 Next

Some Insights

- On average, each bedroom is associated with an additional 63.48 dollars in price.
- Likewise, **cozy** units are associated with a reduction: -12.52 dollars on average.
- Each additional point of **review_scores_rating** is associated with an average of 0.794 additional dollars in price.
- The Bronx and Staten Island had similar prices, with Queens and Brooklyn being somewhat higher, and Manhattan is considerably higher.

Predictive Power Versus Inference

- The Kaggle competition prioritizes the accuracy of the predictions.
- Borough and neighborhood effects on pricing are interesting – but not as powerful as the exact coordinates.
- How to proceed depends on your priorities.

Products Versus Insights

- This dichotomy also plays out in the different applications that you may work on.
- When the model's prediction is the product, then accuracy is prioritized above all else.
- When the model is used for understanding the relationships between factors – or communicating those relationships to others – then the predictive accuracy of the model needs to be balanced against the quality of its inferences.

Settings Prioritizing Insights

- **Marketing research**: identifying segments of customers who especially like or dislike a product.
- **Medicine**: Examining the impact of clinical factors, treatments, and patient behaviors on the risk of adverse outcomes.
- **Education**: What is the impact of smaller class sizes, changes in the schedule, or new pedagogical methods on the performance of students?

Settings Prioritizing Accuracy

- **Product Recommendations**: Items, pages, or people of interest to a user, especially on digital platforms.
- **Medical Laboratory Tests**: Accurate classification of diagnoses with relatively few mistakes.
- **Decisions with Money on the Line**: Any time a prediction directly leads to a choice that can be evaluated in terms of a future payout, e.g. financial transactions, strategic choices, salary negotiations, etc.

You Can Also Do Both

- Different models can serve different purposes.
- Insights can inform pursuits of greater accuracy and vice versa.
- Identifying good features can improve your results in both regards.

Meeting the Challenges, Mastering the Methods

Understanding these practical challenges can make all the difference in real projects. This is what modeling with real data – messy and imperfect all around – is like. The good news is that, with experience, you'll be better prepared to handle the next setting. And, better yet, each practical challenge reinforces the methodological lessons you've learned!