Predicting costs

Chris Eshleman

10/20/2019

Q: What's the cross-validation being used? Q: Can I relax the penalty I'm applying via lasso? YES - THERE'S SOMETHING TO THIS - Flesh it out!

The agency estimates project costs internally. It may be able to use statistics to improve those predictions.

Model variation in bids, withholding the two most recent quarters of observations. Then test those models on the withheld observations. How well do they help with predictions?

Methods and Data.

1. Basic (manually constructed linear model),
2. Lasso (penalized regression - machine learning).

The project-level data comes from internal agency cost estimation.
The economic data is just quarterly stuff from the usual suspects and is specific to national and regional economic and labor market conditions.

Load some programming tools that are commonly used for this analysis. Not all of these packages will be used, and at some point it'll be worth backing up and cleaning the list.

Overview.

This effort bridges exploration of the agency's data with potential predictors from exogenous sources. The key idea is that if economic indicators (numbers) can add measurable value to the agency's cost estimation methods, it can help set the agency up for better informed next steps. Those next steps are yet to be defined.

```
#![Here's what we have in mind.](Estimator_Data.png)
```

The measures of "accuracy" are, for now, bivariate correlation. We discuss below why it's not yet time for fancier metrics, but the process above represents a first cut at trying to add a little more statistical value to the process.

Data comes from a few sources and needs to be merged. Two usual suspects are the agency's internal project information and a set of economic indicators specific to Greater New York (18 counties on both sides of the Hudson River).

Add economic data.

Add permits and steel prices.

The City of New York's database on permitting covers comercial and residential activity.

I'd like to ask have robust data on the rest of the region, including (and namely) Jersey City, but it's weaker than the City's, which by itself is a decent barometer of construction activity in greater New York.

Prices of construction materials and labor also figure into the agency's internal cost estimation, and I'll use steel prices for now. Future models might try and rope in other pricing data points, but the estimators are generally already taking prices into account when setting their numbers so this is likely of second- and third-order importance but the modeling selection algorithms may suggest using them.

The Engineering News-Record tracks and aggregates construction cost data through an index. Use that to cover prices.

```
cci = read.csv("./cci.csv")
names(cci) = tolower(names(cci))
cci$avg. = NULL
cci = gather(cci, month, cci, jan:dec, factor_key=TRUE)
cci$month = gsub("(^[[:alpha:]])", "\\U\\1", cci$month, perl=TRUE)
cci$month = as.Date(paste(cci$month,"01",cci$year, sep="-"), format="%b-%d-%Y")
cci = cci[month(cci$month) == 2 | month(cci$month) == 5 | month(cci$month) == 8 | month(cci$month) == 1

cci$Quarter = paste("Q",quarter(cci$month), sep="")
cci$Q = paste(year(cci$month),cci$Quarter,sep="-")
cci$Quarter = NULL
cci$month = NULL
cci$year=NULL

bids = merge(bids, cci, by = "Q", all.x=TRUE)

rm(cci)
```

Munge

Stats software treats different variables in different ways depending on individual formatting. It's worth taking a look at the data structure.

I'll need to tell the software to reformat some of the variables, namely the economic indicators.

One variable of interest is the bidding process. Institutional discussions and earlier modeling suggests the bidding process may influence the bids. Limits placed on the range of bidders, for example, could, on average and holding other things constant, increase the average (and lowest qualifying) bid - this is basic microeconomics. I'll simplify the bidding format variable by making it binary: "public" for projects without significant constraints and "other" for ones, such as projects closed to firms not deemed "small business enterprises," that aren't.

There's room to also eventually include the names (anonymized is fine) of each project estimator to help modeling. Past work has suggested there isn't major causal variation between estimators — they generally do a pretty equivalent job in estimating bids. But having their names included nonetheless may prove to offer some control value. We can leave that to future modeling.

Restating objective.

We want to understand whether and how we might help the agency estimate the actual cost of a project. That's invariably going to be represented by the low qualifying bid, and our starting point is the estimate coming from the Engineering Department.

From here on we'll define "accuracy" as the ratio of dollars estimated over dollars bid. So a "1" would mean the engineering team nailed it, a "0.94" would mean they estimated 94 cents for every 1 dollar in the low bid, et cetera.

I think there's a pesky outlier in there - a project that, for a strange reason (unexplainable), was way off. Suggest removing it.

We may want to think of project size categorically.

```
bids$decile = decile(vector = bids$Low.Bid)
bids$decile = ordered(bids$decile, levels = 1:10)
```

Back up data.

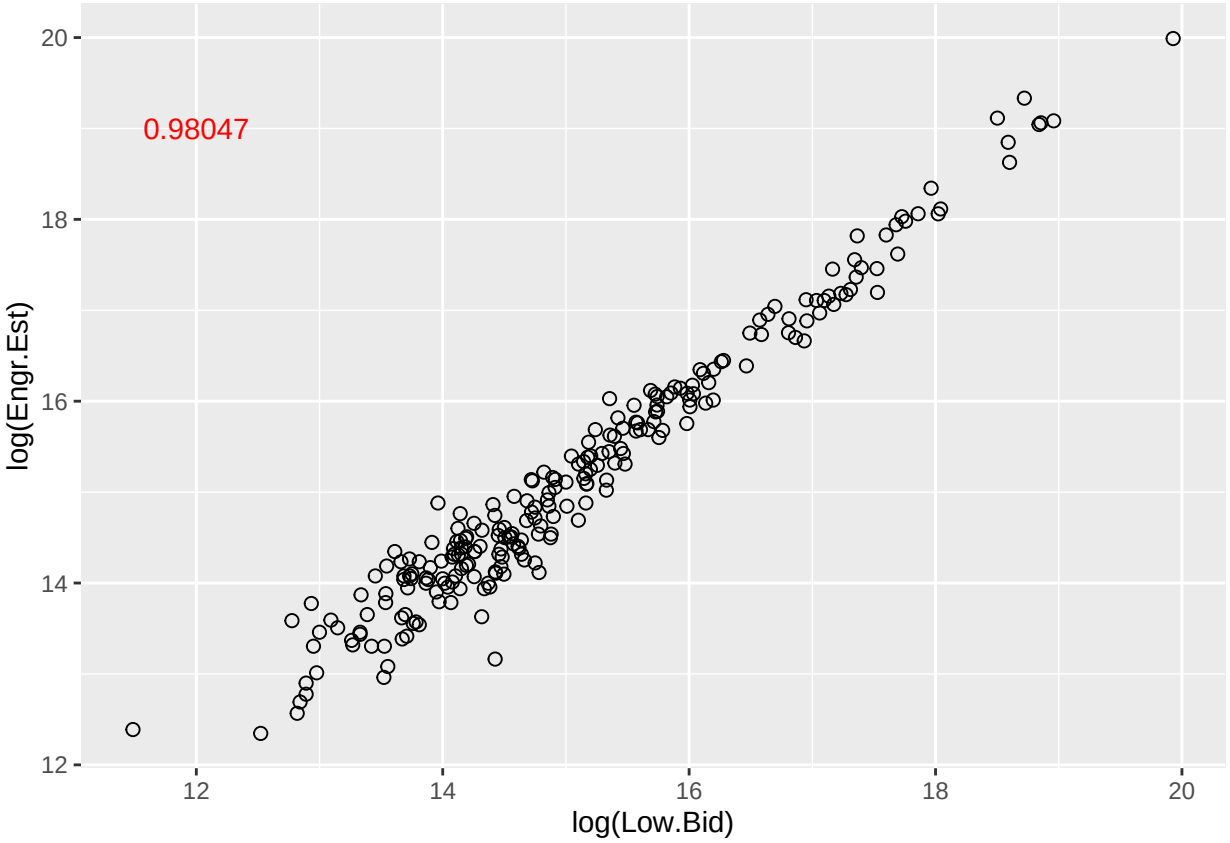
```
bid = bids # backup my data frame
```

Analysis

Now the data is prepped. So split it into the training / test sets we talked about at the start. The bids start in 2015.

Motivation.

Why do we think developing a conrolled multivariate (complicated) model will be worth it? Well, the average gap is $2.5 million, or 20%, off of our estimates. What's the raw (uncontrolled) bivariate relationship between engineering estimates and low bids?




```
##
## Call:
## lm(formula = bids$Low.Bid ~ bids$Engr.Est)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -71323683   -537467   -148673    402475  54486508
```

```
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.474e+05  5.180e+05   0.671    0.503
## bids$Engr.Est 8.273e-01  1.080e-02  76.580   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7543000 on 236 degrees of freedom
## Multiple R-squared:  0.9613, Adjusted R-squared:  0.9612
## F-statistic:  5864 on 1 and 236 DF,  p-value: < 2.2e-16
```

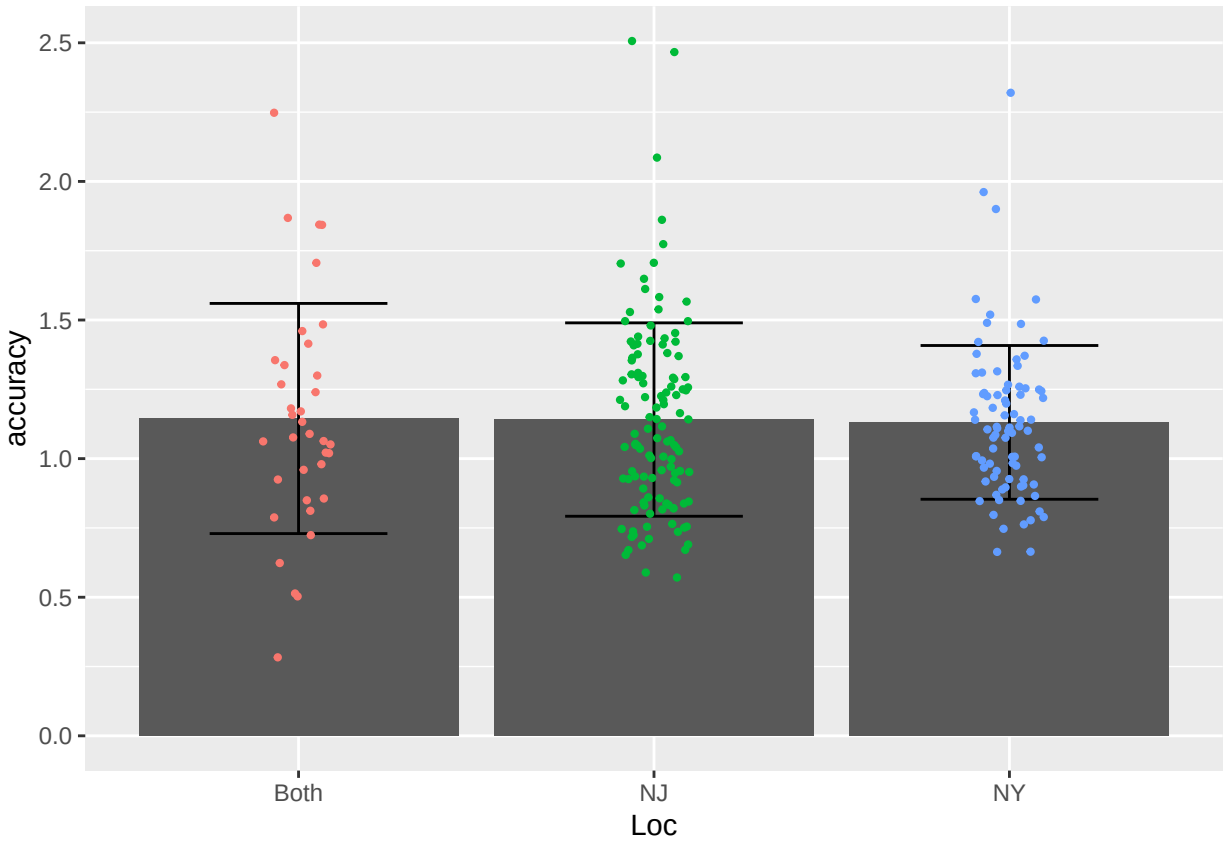
(The logarithm is just to distribute it across the plot (one of the observations is an outlier).)

The in-house engineers arguably do a pretty good job - they're guesses predict more than 98% of the variation in low bids.

Some of the remaining variation can be explained with some guesswork.

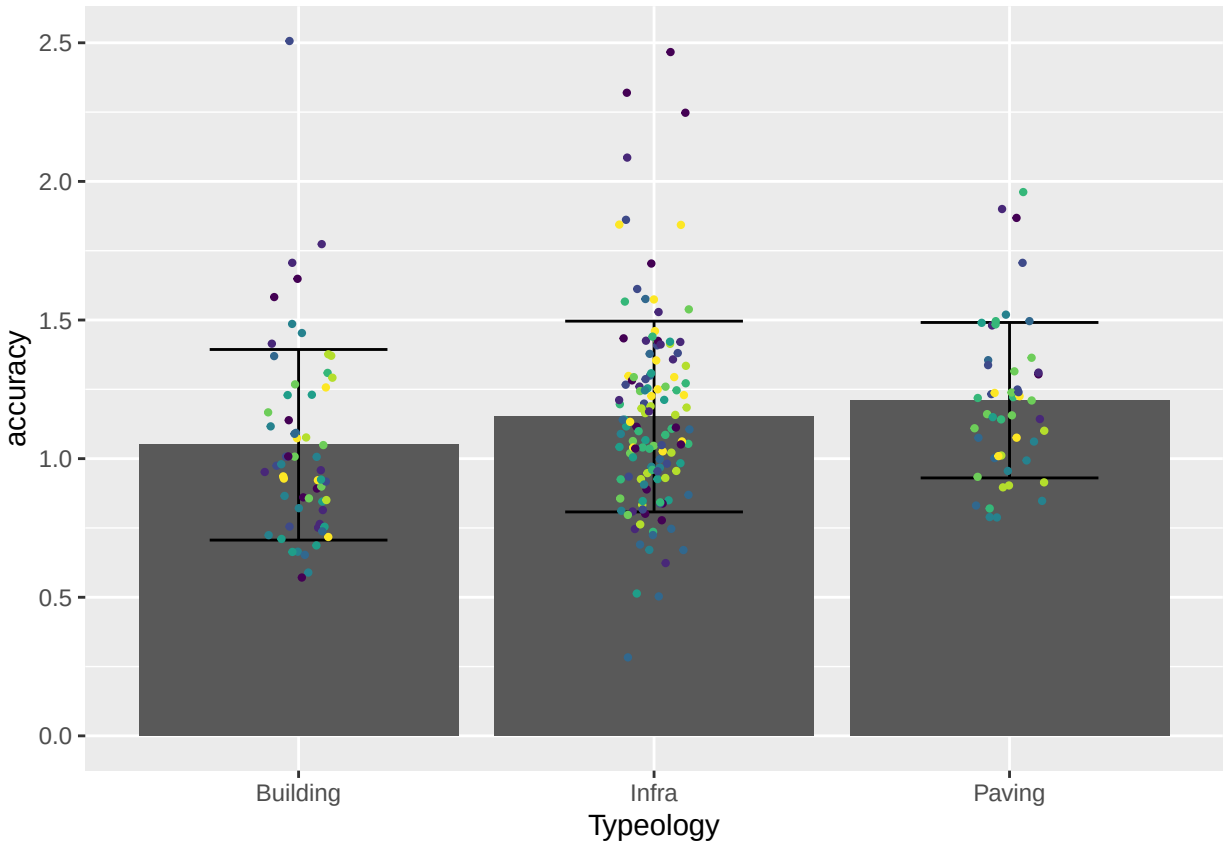
Location seems to have some predictive power. Not a ton, but a little. Basically, projects that somehow span the Hudson River wind up costing more, on average, than ones plunked squarely in either New York or New Jersey. What's the raw (uncontrolled) relationship between bid accuracy and location?

```
##
## Call:
## lm(formula = bids$accuracy ~ bids$Loc)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.86160 -0.21976 -0.03235  0.15669  1.36557
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.144748   0.056128  20.395   <2e-16 ***
## bids$LocNJ  -0.003834   0.064121  -0.060    0.952
## bids$LocNY  -0.014020   0.067086  -0.209    0.835
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3368 on 235 degrees of freedom
## Multiple R-squared:  0.0002656,  Adjusted R-squared:  -0.008243
## F-statistic: 0.03121 on 2 and 235 DF,  p-value: 0.9693
```



The signal is stronger regarding the type of project, which has an identifiable (if yet uncontrolled) relationship with estimating accuracy. What's the raw (uncontrolled) relationship between estimation accuracy and the type of project?

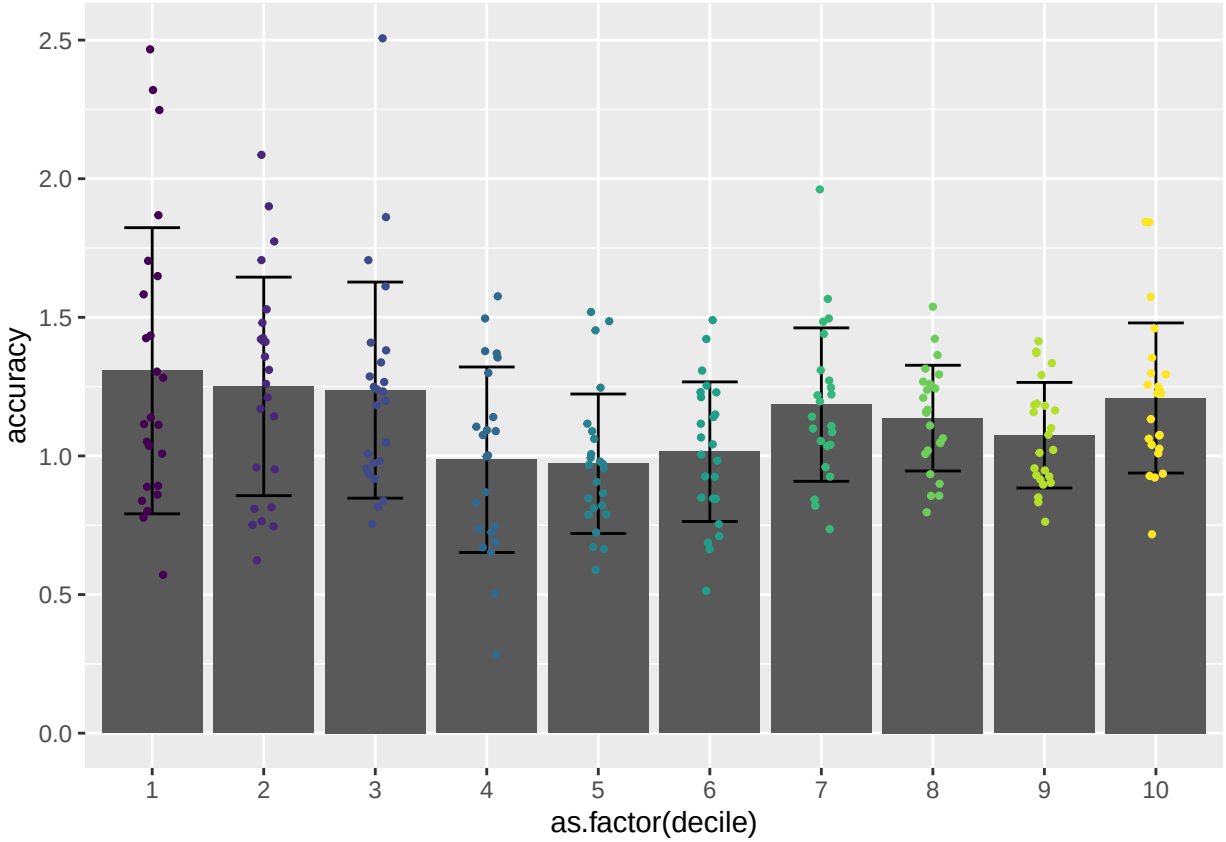
```
##
## Call:
## lm(formula = bids$accuracy ~ as.factor(bids$Typeology))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.86873 -0.21486 -0.04378  0.14739  1.45652
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     1.04997    0.04251  24.701   <2e-16 ***
## as.factor(bids$Typeology)Infra  0.10192    0.05165   1.973   0.0496 *
## as.factor(bids$Typeology)Paving 0.16086    0.06369   2.526   0.0122 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.332 on 235 degrees of freedom
## Multiple R-squared:  0.02841,    Adjusted R-squared:  0.02014
## F-statistic: 3.435 on 2 and 235 DF,  p-value: 0.03384
```



We see signals that projects of medium size (in dollar terms) may be less evasive than much larger or smaller projects. What's the relationship between low bids and accuracy, when we start considering the size of low bids?

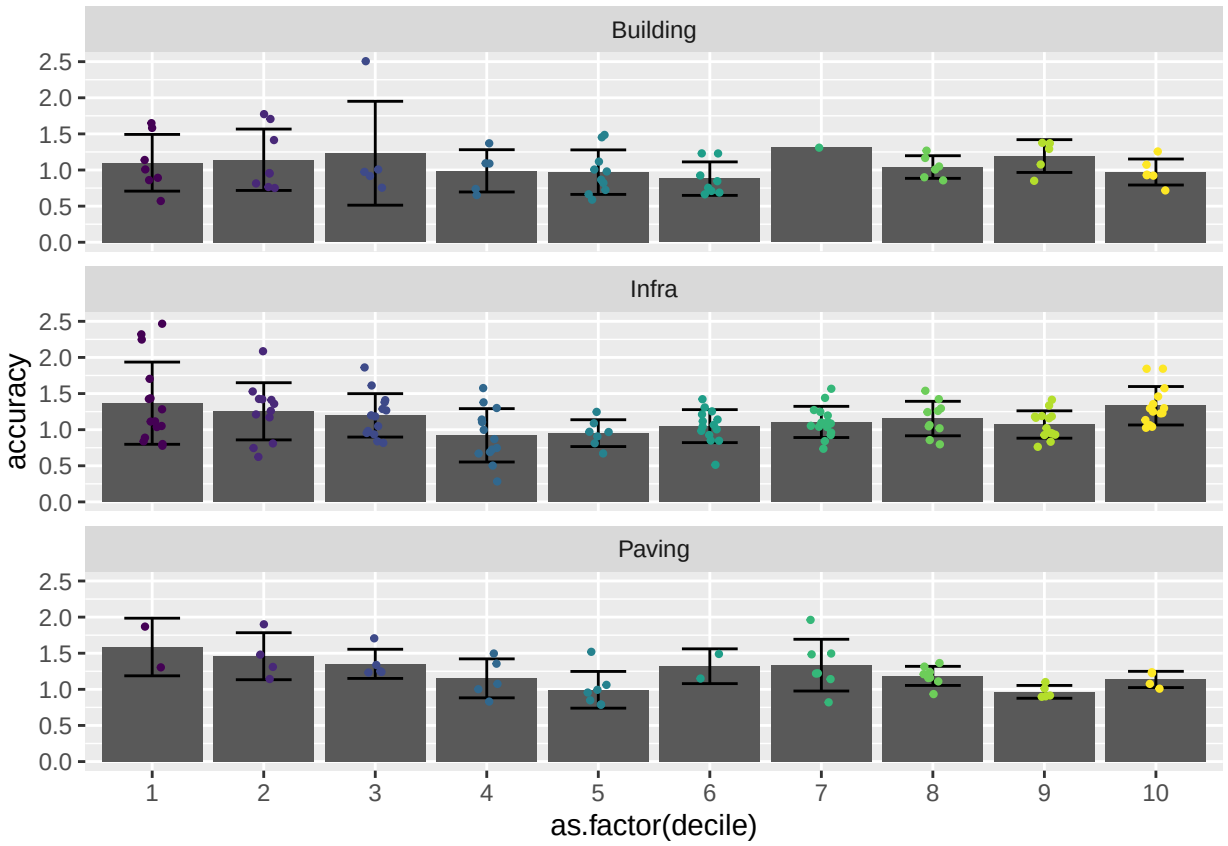
```
##
## Call:
## lm(formula = bids$accuracy ~ as.factor(bids$decile))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.73608 -0.21870 -0.00375  0.14509  1.26902
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 1.13746    0.02086  54.525  < 2e-16 ***
## as.factor(bids$decile).L   -0.10907    0.06576  -1.659 0.098584 .
## as.factor(bids$decile).Q    0.25337    0.06588   3.846 0.000156 ***
## as.factor(bids$decile).C   -0.05198    0.06600  -0.788 0.431769
## as.factor(bids$decile)^4   -0.07375    0.06569  -1.123 0.262789
## as.factor(bids$decile)^5    0.15122    0.06613   2.287 0.023125 *
## as.factor(bids$decile)^6    0.11047    0.06584   1.678 0.094749 .
## as.factor(bids$decile)^7   -0.07075    0.06586  -1.074 0.283788
## as.factor(bids$decile)^8    0.01371    0.06646   0.206 0.836764
## as.factor(bids$decile)^9   -0.06060    0.06610  -0.917 0.360172
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3218 on 228 degrees of freedom
```

```
## Multiple R-squared:  0.1144, Adjusted R-squared:  0.07948
## F-statistic: 3.274 on 9 and 228 DF,  p-value: 0.0009076
```




```
ggplot(bids, aes(x = as.factor(decile), y = accuracy)) +
  stat_summary(fun.y = "mean", geom = "col") +
  stat_summary(fun.data = mean_sdl, geom = "errorbar", width = .5, fun.args = list(mult = 1)) +
  geom_jitter(size=.8,position=position_jitter(width=.1, height=0), aes(color = as.factor(decile))) +
  theme(legend.position="none") +
  facet_wrap( ~ Typeology, ncol=1)
```

```
## Warning: Removed 1 rows containing missing values (geom_errorbar).
```



Visualizing and testing the data iteratively like this has offered some initial insight into what might be accounting for variation in accuracy.

Modeling and prediction

Try and use exogenous covariates to predict an alternative engineering estimate, without using the low bid information, that might be closer to the low bid. Call it "expected low bid" or something so we can remember what we're trying to get.

A. Base model (manual selection)

Interpretation: specification was manual and intuitive.

Note: ensure the "accuracy" variable we calculated earlier is dropped before modeling or I'll be introducing some dual (reverse) causality, which will confuse the models.

Note 2: when a number appears in the output without context, it is likely an information criterion (and AIC), which may or may not provide value post-modeling.

First remove accuracy.

```
train$accuracy=NULL
test$accuracy=NULL
```

Run model.

```
##
## Call:
```

```
## lm(formula = log(Low.Bid) ~ Engr.Est + Employment.in.construction +
##     Format + Typeology + permits_1 + cci + decile, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.56254 -0.10914 -0.00102  0.13129  0.47090
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 1.515e+01  5.806e-01  26.098  < 2e-16 ***
## Engr.Est                    6.448e-09  4.895e-10  13.174  < 2e-16 ***
## Employment.in.construction -2.427e-03  5.129e-03  -0.473   0.6367
## FormatPublic                4.832e-02  3.325e-02   1.453   0.1478
## TypeologyInfra             -2.578e-03  3.824e-02  -0.067   0.9463
## TypeologyPaving             6.331e-02  4.617e-02   1.371   0.1719
## permits_1                  -2.690e-06  3.165e-06  -0.850   0.3963
## cci                         7.551e-05  1.894e-04   0.399   0.6905
## decile.L                    3.895e+00  5.808e-02  67.054  < 2e-16 ***
## decile.Q                    5.361e-01  5.616e-02   9.545  < 2e-16 ***
## decile.C                    4.285e-01  5.219e-02   8.212 2.95e-14 ***
## decile^4                   -1.307e-01  4.877e-02  -2.680   0.0080 **
## decile^5                   -1.758e-02  4.821e-02  -0.365   0.7158
## decile^6                   -1.026e-01  4.739e-02  -2.165   0.0316 *
## decile^7                   -7.983e-02  4.817e-02  -1.657   0.0991 .
## decile^8                   -2.104e-02  4.843e-02  -0.435   0.6644
## decile^9                   -5.398e-03  4.780e-02  -0.113   0.9102
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2175 on 195 degrees of freedom
## Multiple R-squared:  0.9787, Adjusted R-squared:  0.977
## F-statistic: 560.7 on 16 and 195 DF,  p-value: < 2.2e-16
```

The equation above throws extra information at the engineering estimate and tries to predict the low qualifying bid. If the result is noticably closer to the low qualifying bid than the original estiamte, you can use the delta as a post-estimation fudge factor to adjust your final estimate.

The numbers calculated above include a few metrics to use in comparing the three estimates against the objective data point at the low bid, which is what we're trying to predict. Ghose three estimates I'm talking about are: 1. The original, raw engineering cost estimate, 2. The first alternative, where we built a model by hand to try and use a few more data points to enhance the original estimate, and; 3. The second alternative, a kitchen sink model that throws even more data points at the question. This followed an effort to use a penalized regression to identify the best covariates, but that penalization algorithm actually suggested there isn't much we can do to enhance the original estimate. (Note: this will prove prescient.)

I'll build a table near the very end of this script that summarize the metrics I'm using to understand how well these modeling efforts work. The metrics will be: A. A basic t-test to understand whether there's even a statistically significant difference between the estimate I'm getting and the enhanced estimate I'm modeling with it, B. A correlation between the two numbers, to try and understand the magnitude of that difference (if we can trust it really exists), C. Two measures of the predictive modeling power of the models, an adjusted R-squared and the mean squared error (MSE). Both are common metrics of power. The first can be viewed discretely for each model but the second only provides a relative measure between models.

What is the summary of the predicted values? How does it compare to the summary of low bids?

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   13.04   14.07   14.67   15.05   15.88   20.37
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
##     97300   1294754   2544498  12557218   7008666 451841280
```

Looking at the two summaries above, what's more accurate, the original engineering estimate or the first enhanced prediction? Neither, really. In fact the estimate and prediction aren't even (statistically) significantly different. Maybe something more robust can come with a little creativity.

(Note: the model should control to prevent negative values. To be done next time.)

It might have been worth trying with the log of prices, only because the statistical fit becomes multiplicative instead of linear, but that produced similar results.

B. Stepwise selection

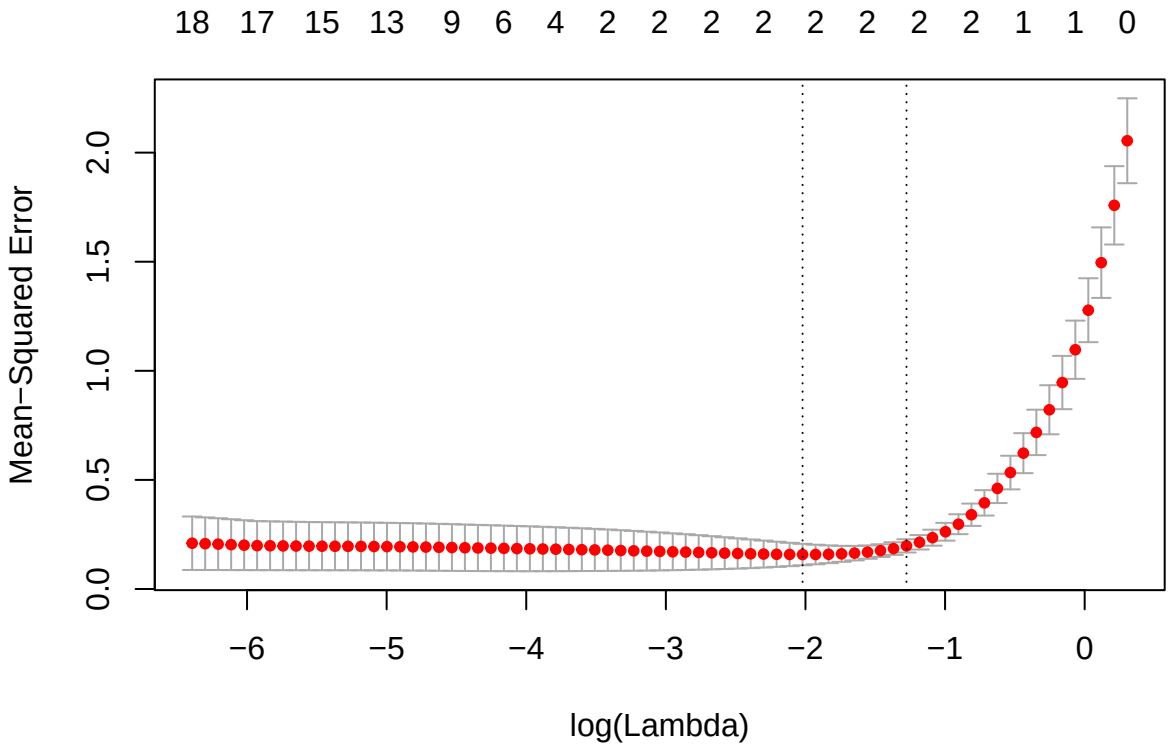
Do a backward stepwise selection only because early runs indicate there's little room to add much predictive power, so recognize that the default process backward will favor leaving variables in more than forward.

COMING SOON.

C. Machine learning

Strip the bids data of unusable stuff, then set controls and run. The package I'm using, glmnet, requires a little extra preparation.

The output below represents the algorithm's effort to look for variatbles that might be dependable in adding predictive power to the original engineering estimate.




```
## 46 x 1 sparse Matrix of class "dgCMatrix"
##                                                      1
## (Intercept)                                 13.118418586407863
## Format                                               .
## Engr.Est                                     0.000000004527796
```

```
## Low.Bid                                          .
## Loc                                              .
## LD                                               .
## Typeology                                        .
## Consumer.price.index                             .
## Employment.in.communications                     .
## Employment.in.construction                       .
## Employment.in.education.and.health               .
## Employment.in.financial.and.business.services    .
## Employment.in.financial.services                 .
## Employment.in.government                         .
## Employment.in.other.services                     .
## Employment.in.production.industries              .
## Employment.in.professional.services              .
## Employment.in.real.estate                        .
## Employment.in.retail                             .
## Employment.in.transport.services                 .
## Employment.in.wholesale                          .
## Output.in.communications                         .
## Output.in.construction                           .
## Output.in.financial.services                     .
## Output.in.government                             .
## Output.in.retail                                 .
## Output.in.education.and.health                   .
## Output.in.financial.and.business.services        .
## Output.in.other.services                         .
## Output.in.production.industries                  .
## Output.in.professional.services                  .
## Output.in.real.estate                            .
## Output.in.transport.services                     .
## Output.in.wholesale                              .
## Personal.disposable.income..nominal              .
## Personal.disposable.income..real                 .
## Personal.income..nominal                         .
## Retail.sales..nominal                            .
## Retail.sales..real                               .
## Total.employment                                 .
## Total.office.based.employment                    .
## Total.output                                     .
## Total.population                                 .
## permits_1                                        .
## cci                                              .
## decile                                0.344327195724216
## 46 x 1 sparse Matrix of class "dgCMatrix"
##                                                        1
## (Intercept)                           12.892952531070996
## Format                                           .
## Engr.Est                              0.000000006884421
## Low.Bid                                          .
## Loc                                              .
## LD                                               .
## Typeology                                        .
## Consumer.price.index                             .
```

```
## Employment.in.communications                    .
## Employment.in.construction                      .
## Employment.in.education.and.health              .
## Employment.in.financial.and.business.services   .
## Employment.in.financial.services                .
## Employment.in.government                        .
## Employment.in.other.services                    .
## Employment.in.production.industries             .
## Employment.in.professional.services             .
## Employment.in.real.estate                       .
## Employment.in.retail                            .
## Employment.in.transport.services                .
## Employment.in.wholesale                         .
## Output.in.communications                        .
## Output.in.construction                          .
## Output.in.financial.services                    .
## Output.in.government                            .
## Output.in.retail                                .
## Output.in.education.and.health                  .
## Output.in.financial.and.business.services       .
## Output.in.other.services                        .
## Output.in.production.industries                 .
## Output.in.professional.services                 .
## Output.in.real.estate                           .
## Output.in.transport.services                    .
## Output.in.wholesale                             .
## Personal.disposable.income..nominal             .
## Personal.disposable.income..real                .
## Personal.income..nominal                        .
## Retail.sales..nominal                           .
## Retail.sales..real                              .
## Total.employment                                .
## Total.office.based.employment                   .
## Total.output                                    .
## Total.population                                .
## permits_1                                       .
## cci                                             .
## decile                                          0.379630668836210
```

ISL suggests "a large value of s corresponds to [lambda]=0 … if s is sufficiently large, …. the … estimates will be the same as the least squares estimates."

The algorithm suggests using anything beyond the engineer's estimate itself to better predict the lowest qualifying good adds more uncertainty (in the form of noise that's tough to explain) than it adds value. (The "penalty" associated with adding variables is greater than the extra predictive power they bring.) The exception is project size, and that would provide a chance to look closer.

```
options(scipen=999)
estimate_c = lasso$fitted.values #predict(compare,train2)
mse_c = round(mse(test$Low.Bid,lasso$fitted.values),0)
#accuracy_c = cor(test$Low.Bid,lasso$fitted.values)
#adjr_c = round(summary(lasso)$adj.r.squared,3)
#ttest_c = t.test(lasso$fitted.values,test$Engr.Est)$p.value
```

Accuracy

Fascinating. I first tried calculating a metric that captures accuracy (the relationship between the estimate and the actual). We wanted to use a weighted average of the ratio between estimate and actuals. Something like this …

accuracy_a = weighted.mean((train$Engr.Est/train$Low.Bid),train$Low.Bid)

… applied across all three prisms (uncontrolled and the two models).

But that calculation just results in 1 (and exactly 1) for both models. So we tried something different, summing the estimates and (separately) the actuals, and then dividing those totals. But that also gave me 1 because the average estimate and the average actual are exactly the same, down to the dollar.

How can we be estimating the value that well, on average? The model must be optimized with respect to the average dependent value.

But this is a question for another time. For now I'll just need to lean on other metrics as indications of accuracy:

Well, the correlations between estimates and actuals do show signals of tightening up slightly. But how robust is this, really? Are the numbers really different? And how would the two models perform on fresh data?

Compare new estimates to original estimates.

No statistically significance differences to be found in either case.

Validate

Run on the withheld data and check.

We've already got the models (both of them): "base" and "compare".

Results for the broader model:

```
##
##  Welch Two Sample t-test
##
## data:  estimate_c2 and test2$Engr.Est
## t = -0.39287, df = 44.093, p-value = 0.6963
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -34182829  23029485
## sample estimates:
## mean of x mean of y
##  20644981  26221653
```

Compare

This is a table comparing performance of: 1. The raw estimates, 2. The manually-developed model, and, 3. The informed model along some key measures. The p-values (below 0.1 is strong) and correlation (closer to 1 is strong) can be considered discretely for the two models (and, for correlation, the raw estimate), as they reflect how strongly predicted bids and actual bids are related, and the adjusted R-squared (closest to

1 is best) and mean squared error (MSE; lower is best) data points offer technical insight into how well each model performs …

For the training exercise (data from 2015-Q1 2019):

Results:

```
#t(results_train)
```

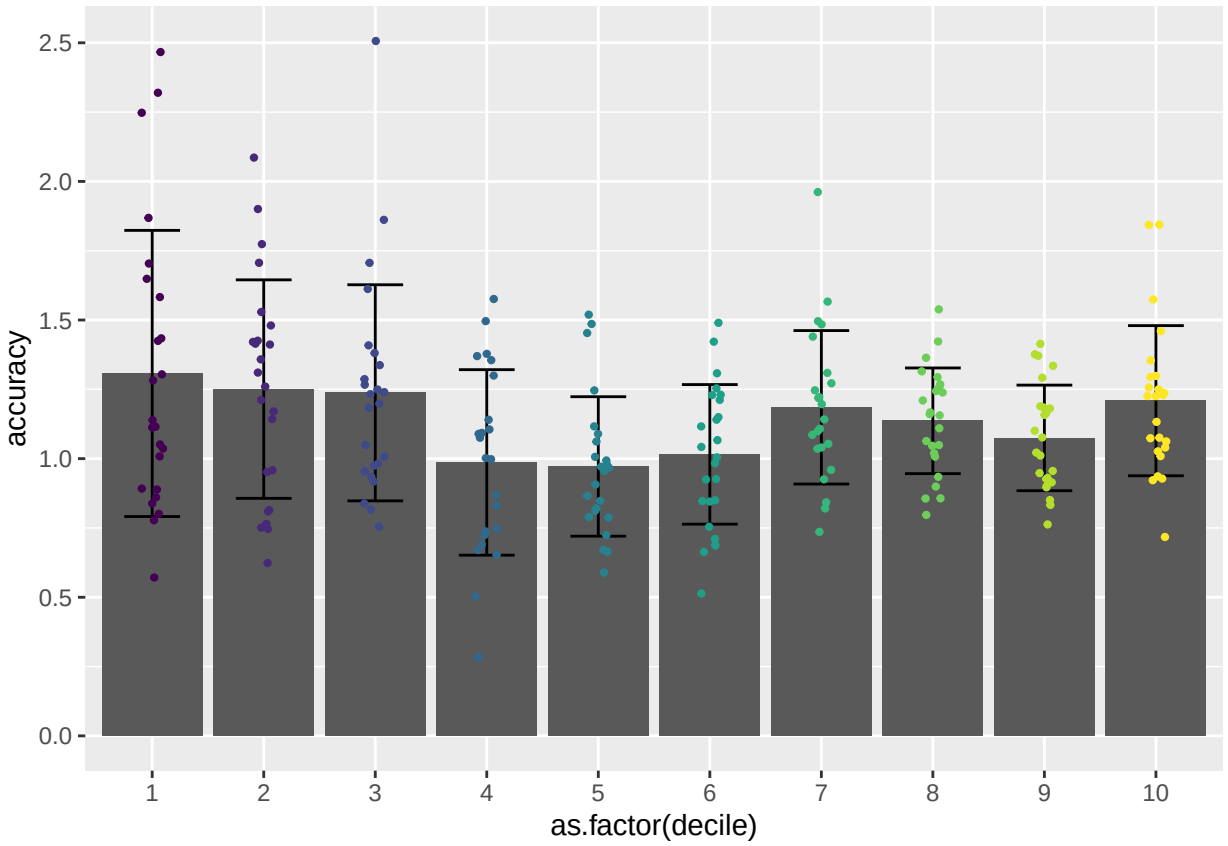
For the test (data withheld, from Q2 and Q3 2019):

Discussion.

The model we selected myself doesn't do much better than the engineer's estimate. And the model with automated variable selection via lasso doesn't really do any better than the one by hand.

The big insight here seems to be the impact project size has on accuracy.

```
##
## Call:
## lm(formula = bids$accuracy ~ as.factor(bids$decile))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.73608 -0.21870 -0.00375  0.14509  1.26902
##
## Coefficients:
##                           Estimate Std. Error t value             Pr(>|t|)
## (Intercept)                1.13746    0.02086  54.525 < 0.0000000000000002
## as.factor(bids$decile).L  -0.10907    0.06576  -1.659             0.098584
## as.factor(bids$decile).Q   0.25337    0.06588   3.846             0.000156
## as.factor(bids$decile).C  -0.05198    0.06600  -0.788             0.431769
## as.factor(bids$decile)^4  -0.07375    0.06569  -1.123             0.262789
## as.factor(bids$decile)^5   0.15122    0.06613   2.287             0.023125
## as.factor(bids$decile)^6   0.11047    0.06584   1.678             0.094749
## as.factor(bids$decile)^7  -0.07075    0.06586  -1.074             0.283788
## as.factor(bids$decile)^8   0.01371    0.06646   0.206             0.836764
## as.factor(bids$decile)^9  -0.06060    0.06610  -0.917             0.360172
##
## (Intercept)               ***
## as.factor(bids$decile).L  .
## as.factor(bids$decile).Q  ***
## as.factor(bids$decile).C
## as.factor(bids$decile)^4
## as.factor(bids$decile)^5  *
## as.factor(bids$decile)^6  .
## as.factor(bids$decile)^7
## as.factor(bids$decile)^8
## as.factor(bids$decile)^9
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3218 on 228 degrees of freedom
## Multiple R-squared:  0.1144, Adjusted R-squared:  0.07948
## F-statistic: 3.274 on 9 and 228 DF,  p-value: 0.0009076
```


accuracy
2.5
2.0
1.5
1.0
0.5
0.0
1
2
3
4
5
6
7
8
9
10
as.factor(decile)