Milagro & Harriman Capital

Memorandum

To: Harriman Capital
From: Kathleen Ortiz (BVA)

Key Takeaway: Our model's estimate a projected annual profit of $33.22-$40.12M, depending on modeling decisions and techniques. Decisions made based on these models should consider both the model's robustness (high Out-of-sample R^2) and projected evaluation.

Model 1 *(Kathleen's Original Linear Regression):*
Evaluation Estimate: $40,016,174, **Out-of-sample R^2:** 0.79, **In-sample R^2**: 0.72
Description: Kathleen's original linear regression included aggregate income, square footage, college graduates and percent of commuters with over an hour trip.

Model 2 *(Saturated Linear Regression):*
Evaluation Estimate: $33,549,052, **Out-of-sample R^2:** 0.80, **In-sample R^2**: 0.95
Description: This model was a standard linear regression that included every variable in the training data set. This provides a good introductory analysis to potentially significant variables, at the cost of correlated independent variables biasing estimates. Based on the correlation plot provided in Appendix A, we eliminated correlated variables in follow on regressions.

Model 3 *(Manually Built Linear Regression):*
Evaluation Estimate: $40,119,868, **Out-of-sample R^2:** 0.77, **In-sample R^2**: 0.95
Description: This model expanded on Kathleen's original regression but includes retail labor costs, competing stores, and a control for if the store is on an intersection. The former two variables are included based on their statistical significance and the latter for its domain specific importance.

Model 4 *(Forward Stepwise Linear Regression):*
Evaluation Estimate: $33,222,056, **Out-of-sample R^2:** 0.86, **In-sample R^2**: 0.95
Description: This model started with a single variable and iteratively added one variable at a time, if it reduced the model's error based on cross-validation, to ensure we are not overfitting the model to our training data. The model selected eleven variables, described in Appendix B.

Model 5 *(Lasso Regularized Regression):*
Evaluation Estimate: $40,016,174, **Out-of-sample R^2:** 0.81, **In-sample R^2**: 0.95
Description: This modeling technique is made specifically to eliminate correlated variables from the model, as well as provide robust out-of-sample performance by penalizing for large coefficients. Appendix C provides a thorough description of the model.

Appendix A

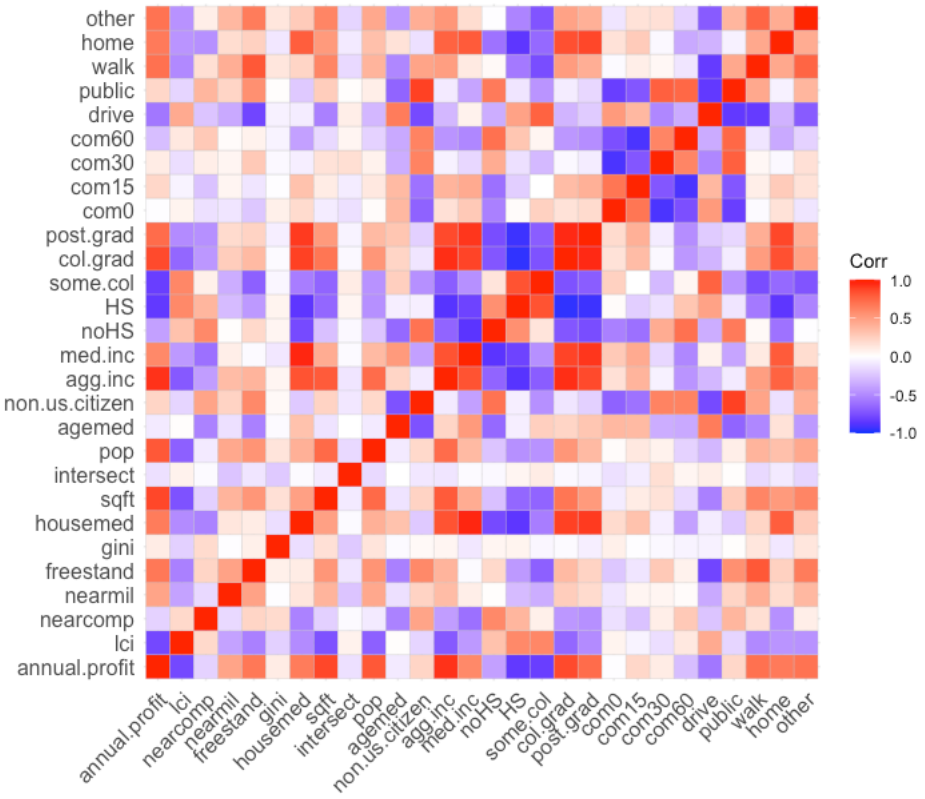

Figure 1A: Correlation Plot

Based on the above figure, median income and college graduates is positively correlated. This makes sense because a bachelor's degree should provide an increase in salary. Additionally, having free standing buildings is negatively correlated with driving because they are both likely correlated with urban areas. In urban areas, you are much less likely to drive to work and buildings are often in a complex. Lastly, non-US citizens is positively correlated with public transportation, likely because it is difficult to obtain a driver's license as a non-citizen or it also may be another yet another mutual correlation to urban areas.

Appendix B

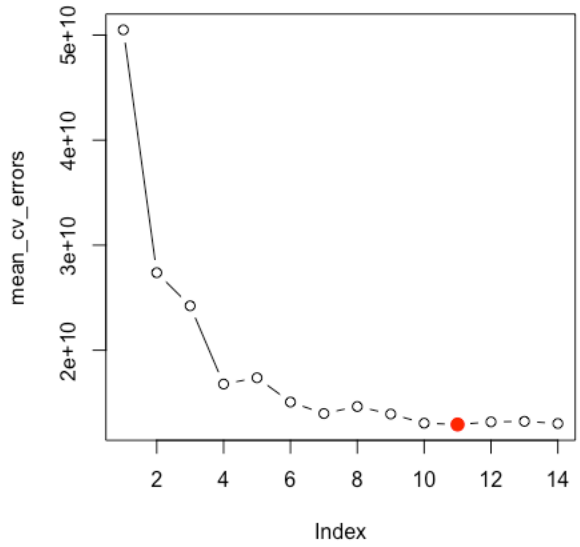

Figure 1B: Forward Selection Error Minimization using Cross-Validation

The above Figure demonstrates the optimal number of variables to minimize error, characterized by mean squared error, is eleven. The model selected the following variables based on forward selection with cross validation and estimated their respective coefficients:

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  6.015e+05  8.046e+04   7.476 5.83e-13 ***
lci         -1.449e+04  3.974e+03  -3.647 0.000304 ***
nearcomp     1.890e+04  2.867e+03   6.593 1.52e-10 ***
nearmil      1.585e+03  4.613e+02   3.436 0.000658 ***
freestand    2.232e+05  1.973e+04  11.309  < 2e-16 ***
sqft         1.585e+02  3.327e+01   4.764 2.76e-06 ***
pop          5.074e+01  4.708e+00  10.778  < 2e-16 ***
agg.inc      2.279e-03  8.256e-05  27.610  < 2e-16 ***
col.grad     4.001e+05  6.194e+04   6.459 3.40e-10 ***
drive       -5.600e+05  6.595e+04  -8.492 5.31e-16 ***
public      -2.008e+05  8.562e+04  -2.346 0.019529 *
home        -1.034e+06  1.993e+05  -5.187 3.57e-07 ***
```

Appendix C

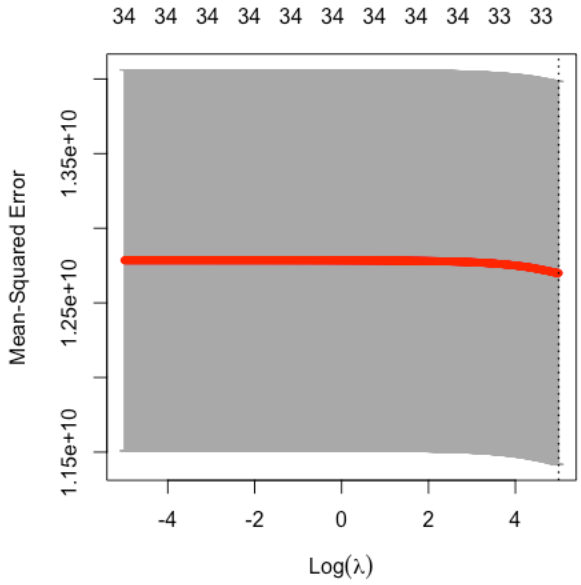

Figure 1C: Lambda Parameter Tuning

The above figure demonstrates the optimization of our coefficient penalizing term, lambda. Using 10-fold cross-validation 100 times at each value, we sought to minimize the error by selecting the optimal lambda. This graph demonstrates the value selected was five, which in turn will decrease our coefficient estimates. The model selected the following variables and estimated their associated coefficients:

(Intercept)	502131.35	intersect	5872.58
store.number	16.69	pop	57.03
lci	-15123.61	agemed	2037.65
nearcomp	18170.89	non.us.citizen	-64650.87
nearmil	1483.47	agg.inc	0.00
freestand	216764.61	med.inc	0.05
gini	7406.96	noHS	-20234.50
housemed	30.96	HS	-181221.60
stateAZ	-8413.15	some.col	.
stateCA	-14247.16	col.grad	287154.74
stateKS	4405.49	post.grad	-135508.13
stateNM	8969.87	com0	57894.14
stateNV	.	com15	-2768.95
stateTX	6119.59	com30	5787.85
stateUT	32770.52	com60	-48581.05
stateWY	25885.89	drive	-498945.03
sqft	152.16	public	.
		walk	133662.54
		home	-1118707.15
		other	260096.44

Appendix D - Code

```
library(tidyverse)
library(miscTools)
library(Metrics)
library(ggcorrplot)
library(caret)
library(glmnet)
library(dplyr)
library(ggplot2)
library(tidyr)
library(lars)
library(leaps)
library(gbm)
library(MASS)
library(caret)
library(leaps)
library(Rcpp)
library(ISLR)
library(leaps)
library(Metrics)
library(miscTools)
library(glmnet)
library(arsenal)


#1.
setwd("/Users/bennetthellman/Desktop/OneDrive - Massachusetts Institute of Technology/Edge/HWs/Pset1")
train = read.csv("train_data.csv")
test = read.csv("test_data.csv")
sites = rbind(train,test)
sites.const = read.csv("site_const_data.csv")
model.1 = lm(annual.profit~ . - store.number, data=sites)
summary(model.1)
newpred.1 = predict(model.1,newdata=sites.const)
value <- sum(newpred.1)
R2 <- summary(model.1)$r.squared


#############################################################################
#2.
num_vars<-train%>%dplyr::select(-c(store.number, state))
#%>%convert(num(nearcomp, freestand, sqft, intersect, pop, ))
corr <- round(cor(num_vars), 3)
ggcorrplot(cor(corr))
select(sites, -"state")



#############################################################################
#3.
#Kathleen's original model
```

```
og_mod<-lm(annual.profit~ agg.inc + sqft + col.grad + com60, data=train)
og_mod_pred = predict(og_mod,newdata=test)
new_pred = predict(og_mod, newdata = sites.const)
value <- sum(new_pred)
train.mean <- mean(train$annual.profit)
SSE <- sum((og_mod_pred - test$annual.profit)^2)
SST <- sum((train.mean - test$annual.profit)^2)
OSR2 <- 1 - SSE/SST
R2 <- summary(og_mod)$r.squared
value
OSR2
R2



#saturated model
sat_mod<-lm(annual.profit~ ., data=train)
sat_mod_pred = predict(sat_mod,newdata=test)
new_pred = predict(sat_mod, newdata = sites.const)
value <- sum(new_pred)
train.mean <- mean(train$annual.profit)
SSE <- sum((sat_mod_pred - test$annual.profit)^2)
SST <- sum((train.mean - test$annual.profit)^2)
OSR2 <- 1 - SSE/SST
R2 <- summary(sat_mod)$r.squared
value
OSR2
R2

#Self-Built Model
bmod<-lm(annual.profit~ agg.inc + sqft + col.grad + com60, data=train)
summary(bmod)
bmod<-lm(annual.profit~ agg.inc + sqft + col.grad + com60 + lci, data=train)
summary(bmod)
bmod<-lm(annual.profit~ agg.inc + sqft + col.grad  + com60 + lci + nearcomp, data=train)
summary(bmod)
bmod<-lm(annual.profit~ agg.inc + sqft + col.grad  + com60 + lci + nearcomp + intersect, data=train)
summary(bmod)


bmod_pred_train = predict(bmod,newdata=train)
bmod_pred = predict(bmod,newdata=test)
new_pred = predict(bmod, newdata = sites.const)
value <- sum(new_pred)
train.mean <- mean(train$annual.profit)
SSE <- sum((bmod_pred - test$annual.profit)^2)
SST <- sum((train.mean - test$annual.profit)^2)
OSR2 <- 1 - SSE/SST
R2 <- summary(b_mod)$r.squared
value
OSR2
R2
```

```
#############################################################################
#4
set.seed(15072)
train.control <- trainControl(method = "cv", number = 10)
n.predictors <- ncol(train) - 1

step.model <- train(annual.profit ~., data = train,
                    method = "leapForward",
                    nvmax = n.predictors,
                    trControl = train.control
)
step.model$results



predict.regsubsets = function(object,newdata,id,...){
  form = as.formula(object$call[[2]]) # Extract the formula used when we called regsubsets()
  mat = model.matrix(form,newdata)    # Build the model matrix
  coefi = coef(object,id=id)          # Extract the coefficiants of the ith model
  xvars = names(coefi)                # Pull out the names of the predictors used in the ith model
  mat[,xvars]%*%coefi                 # Make predictions using matrix multiplication
}

# Assign each observation to a single fold
k=10
folds = sample(1:k, nrow(train), replace = TRUE)
v = 14
# Create a matrix to store the results of our upcoming calculations
cv_errors = matrix(NA, k, v, dimnames = list(NULL, paste(1:v)))
for(j in 1:k){

  # The perform best subset selection on the full dataset, minus the jth fold
  best_fit = regsubsets(annual.profit~., data = train[folds!=j,], nvmax=v)

  # Inner loop iterates over each size i
  for(i in 1:v){

    # Predict the values of the current fold from the "best subset" model on i predictors
    pred = predict(best_fit, train[folds==j,], id=i)

    # Calculate the MSE, store it in the matrix we created above
    cv_errors[j,i] = mean((train$annual.profit[folds==j]-pred)^2)
  }
}

# Take the mean of over all folds for each model size
mean_cv_errors = apply(cv_errors, 2, mean)

# Find the model size with the smallest cross-validation error
```

```
min = which.min(mean_cv_errors)

# Plot the cross-validation error for each model size, highlight the min
plot(mean_cv_errors, type='b')
points(min, mean_cv_errors[min][1], col = "red", cex = 2, pch = 20)

reg_best = regsubsets(annual.profit~., data = train, nvmax = 19)
round(coef(reg_best, 11), 2)


fcv_mod <-lm(annual.profit~  lci + nearcomp + nearmil + freestand + sqft + pop + agg.inc + col.grad + drive + public +
home, data=train)
summary(fcv_mod)
fcv_pred_train = predict(fcv_mod,newdata=train)
fcv_pred = predict(fcv_mod,newdata=test)
new_pred = predict(fcv_mod, newdata = sites.const)
value <- sum(new_pred)
train.mean <- mean(train$annual.profit)
SSE <- sum((fcv_pred - test$annual.profit)^2)
SST <- sum((train.mean - test$annual.profit)^2)
OSR2 <- 1 - SSE/SST
value
OSR2
rSquared(train$annual.profit, resid = train$annual.profit-fcv_pred_train)


##############################################################################
#5
x.sites = model.matrix(Kathleen.Previous.Prediction ~ . - 1,
                 data=sites.const)
x.train = model.matrix(annual.profit ~ . - 1  ,
                 data=train) # The "-1" just mean that we are excluding the constant term (that is, the intercept)
x.train<-x.train[,-13]
x.test<-x.test[,-13]

#had to delete because OK is not in the sites.const dataset
y.train = train$annual.profit # Here, we are only including the dependent variable.
x.test = model.matrix(annual.profit ~ . - 1,
                 data=test)
y.test = test$annual.profit
lambdas.lasso <- exp(seq(5, -5, -.01))
cv.lasso <- cv.glmnet(x.train,
                 y.train,alpha=1,
                 lambda=lambdas.lasso,
                 nfolds=10,
                 ncv =100)
cv.lasso
plot(cv.lasso)
lasso.lambda.cv <- cv.lasso$lambda.min
lasso.lambda.1SE.cv <- cv.lasso$lambda.1se

lasso.final <- glmnet(x.train,y.train,alpha=1,lambda=lasso.lambda.cv)
summary(lasso.final)
```

```
new_pred = predict(lasso.final, x.sites)
value <- sum(new_pred)
pred.test.final.train <- predict(lasso.final,x.train)
pred.test.final <- predict(lasso.final,x.test)
R2.lasso.final <- 1-sum((pred.test.final.train-train$annual.profit)^2)/sum((mean(train$annual.profit)-
train$annual.profit)^2)
OSR2.lasso.final <- 1-sum((pred.test.final-test$annual.profit)^2)/sum((mean(train$annual.profit)-
test$annual.profit)^2)
round(coefficients(lasso.final), 2)
OSR2.lasso.final
R2.lasso.final
value
```