NAME - Anagha Uppal Section MW - Homework 11

Model Building

due Nov 17 by 330pm

Question 1

Load in `EX7.BIKE` using the `data` command. This is the DC bike demand dataset we have dealt with many times . Let us build the best descriptive model so that we can compare the average demands between days with particular characteristics and the best predictive model so that we might be able to predict future demands.

```
#Load in EX7.BIKE
data("EX7.BIKE")
```

---

a: In total, there are 4 quantitative predictors, 3 categorical predictors with 2 levels, and 1 categorical variable with 7 levels. What is the effective number of potential predictors k? If we run the ``all possible" procedure (which recall ignores hierarchy) without interactions, how many models are considered? How many are considered if we do have two-way interactions?

```
#R code showing the calculation
k<-4+1+1+1+6
2^k
## [1] 8192
2^(k*(k+1)/2)
## [1] 2.47588e+27
```

---

b: Let us begin by developing a model with no interactions. Run `regsubsets` and `see.models` to find a set of acceptable models. Do not add any additional arguments (like `nvmax`) to either command.

```
#regsubsets
ALL <- regsubsets(Demand~.,data = EX7.BIKE)
summary(EX7.BIKE)
##      Demand              Day      Workingday Holiday      Weather       AvgTemp
EffectiveAvgTemp
##  Min.   : 705   Friday   :57   no :132    no :397   No rain:299   Min.   : 4.40
7   Min.   : 5.083
##  1st Qu.:3626   Monday   :56   yes:278    yes: 13   Rain   :111   1st Qu.:14.83
7   1st Qu.:18.134
##  Median :4726   Saturday :66                                      Median :21.38
8   Median :25.568
##  Mean   :4800   Sunday   :61                                      Mean   :20.93
3   Mean   :24.474
##  3rd Qu.:6200   Thursday :59                                      3rd Qu.:27.00
9   3rd Qu.:30.746
##  Max.   :8714   Tuesday  :55                                      Max.   :35.32
8   Max.   :40.246
##                 Wednesday:56
##   AvgHumidity     AvgWindspeed
##  Min.   :18.79   Min.   : 1.500
##  1st Qu.:51.94   1st Qu.: 9.167
##  Median :62.06   Median :11.855
##  Mean   :62.18   Mean   :12.587
##  3rd Qu.:72.26   3rd Qu.:15.323
##  Max.   :97.04   Max.   :34.000
##
#see.models
see.models(ALL)
## Reporting all models with AIC within 4 of the lowest value
##      AIC NumVars
## 1 -217.4       5
## 2 -217.0       4
## 3 -216.6       6
## 4 -215.4       7
## 5 -214.1       8
##
Terms
## 1                                   DaySunday AvgTemp EffectiveAvgTemp AvgHumid
ity AvgWindspeed
## 2                                             AvgTemp EffectiveAvgTemp AvgHumid
ity AvgWindspeed
## 3                         DayMonday DaySunday AvgTemp EffectiveAvgTemp AvgHumid
ity AvgWindspeed
## 4             DayMonday DaySunday WeatherRain AvgTemp EffectiveAvgTemp AvgHumid
ity AvgWindspeed
## 5  DayMonday DaySunday Holidayyes WeatherRain AvgTemp EffectiveAvgTemp AvgHumid
ity AvgWindspeed
```

b1: What predictors are in the model with the lowest AIC?

Response: 5

---

b2: In this model, you see that the day of the week is only represented by the indicator variable `DaySunday`. What is the implication? In other words, which days of the week does the model imply have the same average demand?

Response: The implication is that all the remaining days of the week (Tuesday-Saturday) have the same average demand. Sunday and Monday are the only days …?

---

b3: It might be nice if we can "get away" with a model that omits the day of the week entirely. Based on the output of `see.models`, can we use a model without day of the week as a predictor? Why or why not?

Response: Yes. The second model has only a slightly higher AIC (well within the acceptable range) and is a simpler model (uses 4 variables as oppossed to 5). This model does not utilize days of the week as a predictor.

---

c: Interactions play an important role in many problems, but as you have seen there are too many possible models to consider. Define `naive` to be the naive model and `full` to be the model with all predictors and two-way interactions. Run `step` two ways (each running using the `"both"` direction): starting with the naive model (call that `S1`), and starting with the full model (call that `S2`). Note: *when you knit this document make sure the argument* `trace=0` *is in* `step` *otherwise the output will be very, very long*.

```
#define naive model
naive <- lm(Demand~1, data = EX7.BIKE)
#define full model
full <- lm(Demand~.^2, data = EX7.BIKE)
S1 <- step(naive,scope=list(lower=naive,upper=full), direction="both", trace=0) #c
omplete command and uncomment
S2 <- step(full,scope=list(lower=naive,upper=full), direction = "both", trace=0)
#complete command and uncomment
```

---

c1: In this case, the two searches do not converge to the same model. Report the AICs of both models (which you can read from the last step of the output of `step`) and comment on which one is "better" than the other (if a definitive statement can be made). The model with the lowest AIC found without interactions from b (on this scale) is 5909. Does incorporating interactions provide a model that is "closer to the truth" than a model without them?

Responses: AIC=5864.81 and AIC=5841.96 (but also much more complex model). Yes, but not by a whole lot.

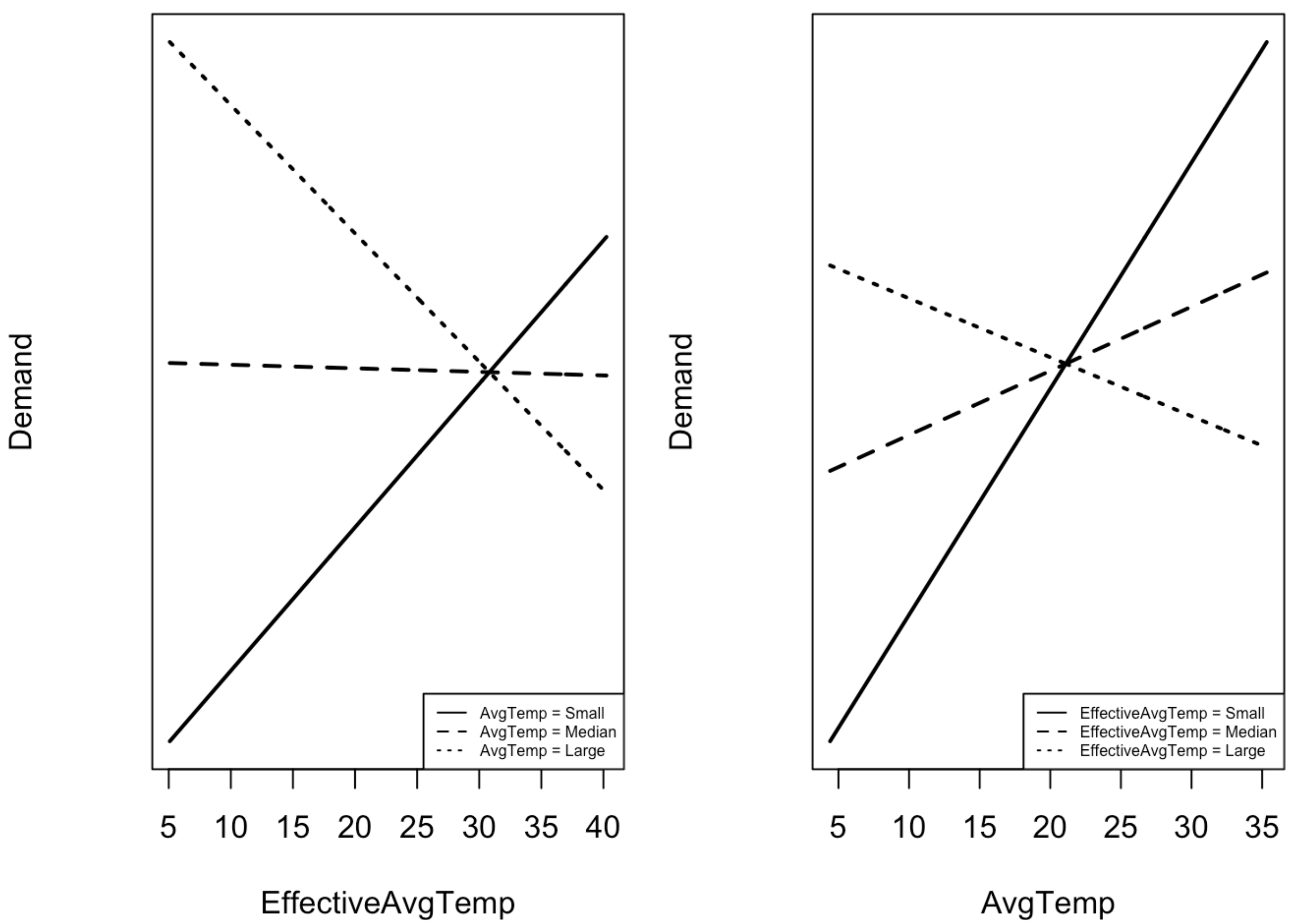
---

c2: In the model selected by a search starting with the naive model, you'll notice that the interaction term for `EffectiveAvgTemp` and `AvgHumidity` is not statistically significant. In general, if we use AICs to compare different models and to choose the final model, should we further eliminate variables that are not statistically significant? Why or why not?

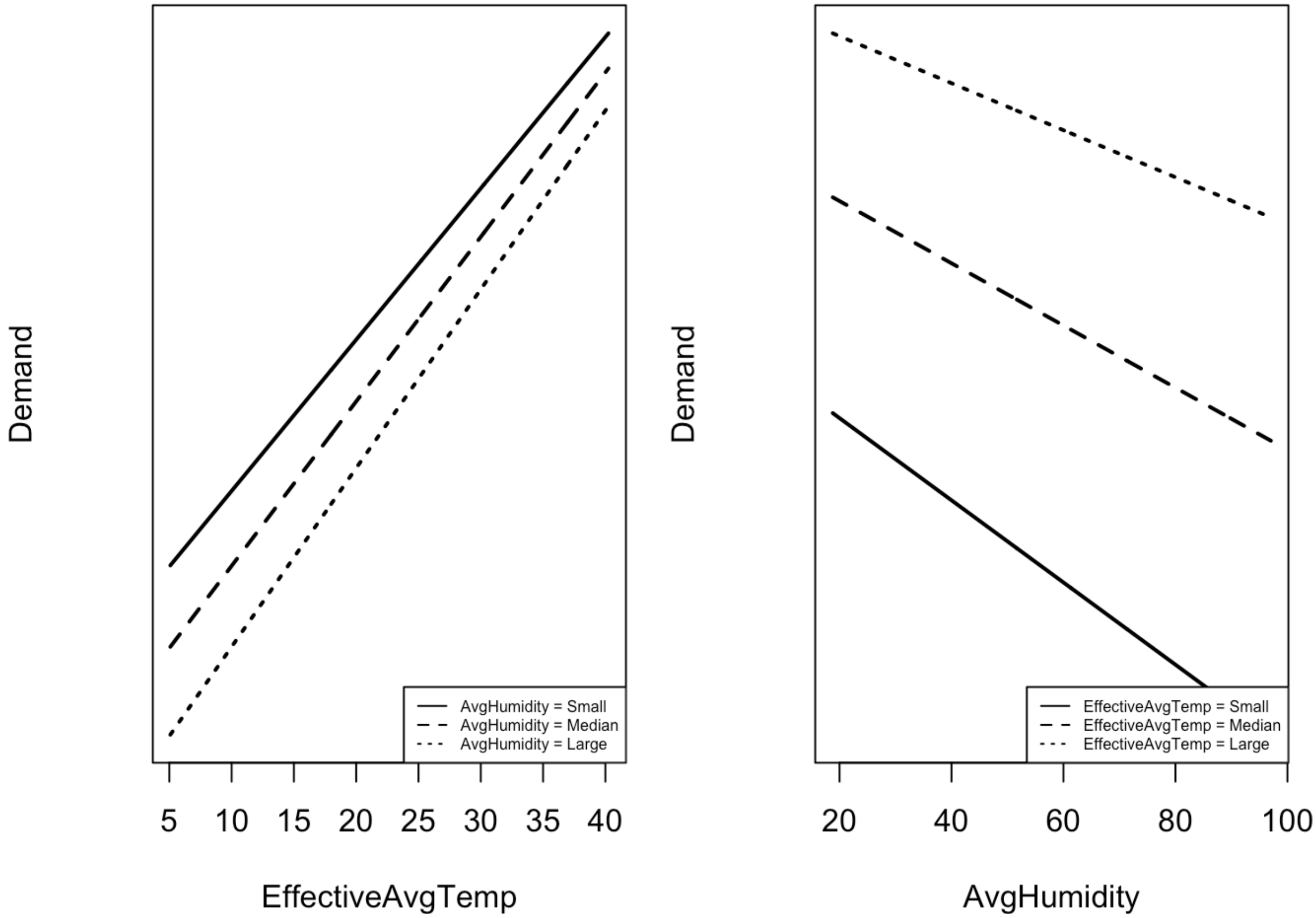
Response: No, we cannot drop variables that are not statistically significant. Statistical significance in this case has nothing to do with whether that particular variable gets us closer to the truth. ***

c3: Imagine the goal of this descriptive model was to answer the question “What does the relationship between Demand and average humidity look like?”. Using the model found by the search starting with the naive model `S1` , run `see.interactions` and comment (since the variable is involved in an interaction, there is no simple way to describe the relationship like “Two days that differ in average humudity by 1% are expect to differ in Demand by …”). Note: it may be useful to add the argument `cex=0.5` so that the legends don’t take up much space.

```
#see.interactions
see.interactions(S1, many=TRUE, cex=0.5)
```




```
##
## Press [enter] to continue to see next set of interactions or q (then Enter) to
quit
```



Response: For lower effective average temperatures, the demand for all levels of average humidity begin very low. As effective average temperature increases, the demand for all levels of average humidity increases until the levels begin to approach equilibrium.

---

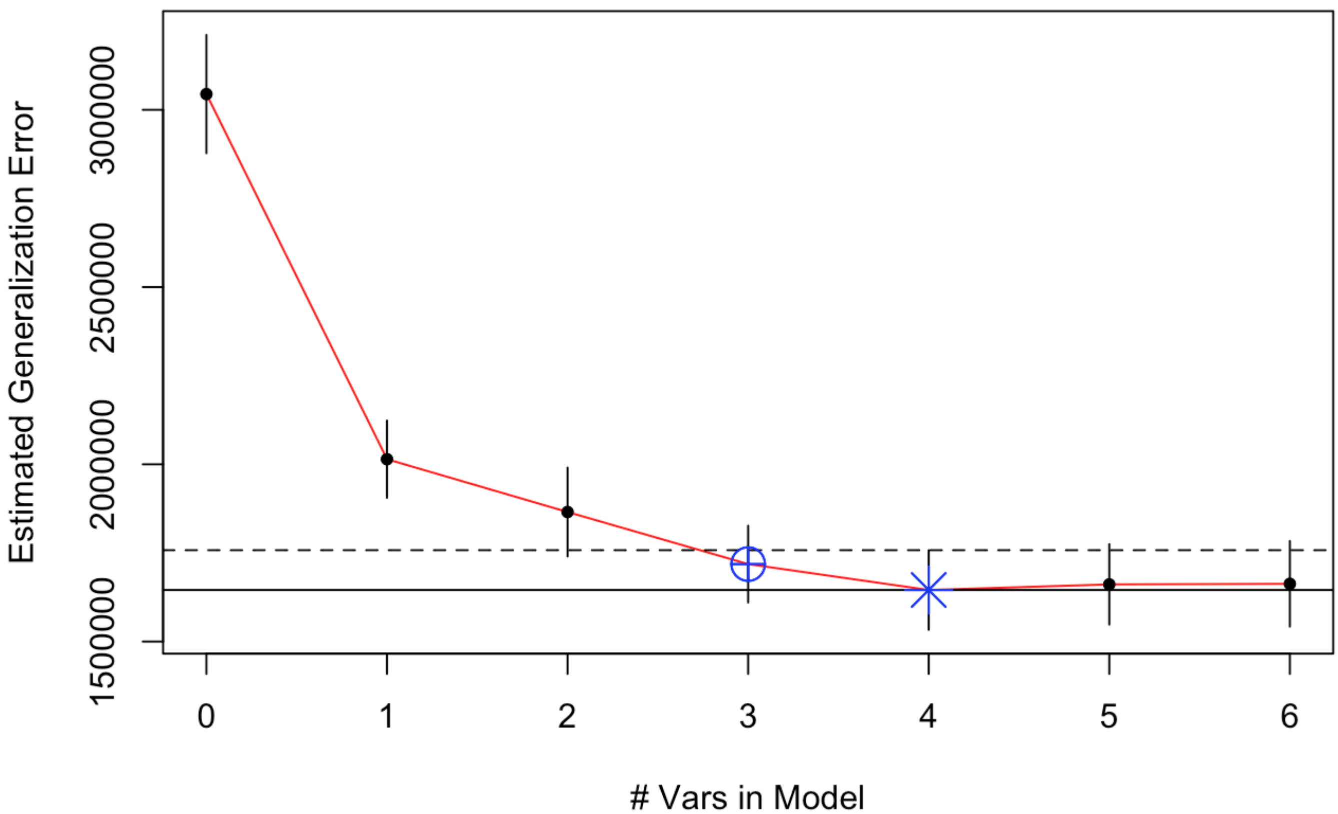
d: The best predictive model may be different than the best descriptive model. To build a predictive model, we need to split the data into training and holdout samples. Do this so that 70% of the original data is in `TRAIN` and the remaining 30% is in `HOLDOUT` . Be sure to `set.seed(320)` immediately before your `sample` command.

```
set.seed(320)
train.rows <- sample(1:nrow(EX7.BIKE),0.7*nrow(EX7.BIKE)) #complete command and un
comment
TRAIN <- EX7.BIKE[train.rows, ] #complete command and uncomment
HOLDOUT <- EX7.BIKE[-train.rows, ] #complete command and uncomment
```

---

d1: Run `build.model` (without considering any interactions) and report the four predictors that appear in the model with the lowest estimated generalization error. Note: be sure to use `seed=320` as an argument. Also, add `prompt=FALSE` so that it will knit.

```
BM <- build.model(formula(S1),data=TRAIN,seed=320,prompt=FALSE)  #complete command and uncomment
##
## Model with lowest estimated generalization error has:
## EffectiveAvgTemp AvgHumidity AvgWindspeed EffectiveAvgTemp.AvgTemp
##
## Model selected with one standard deviation rule has:
## EffectiveAvgTemp AvgHumidity EffectiveAvgTemp.AvgTemp
```

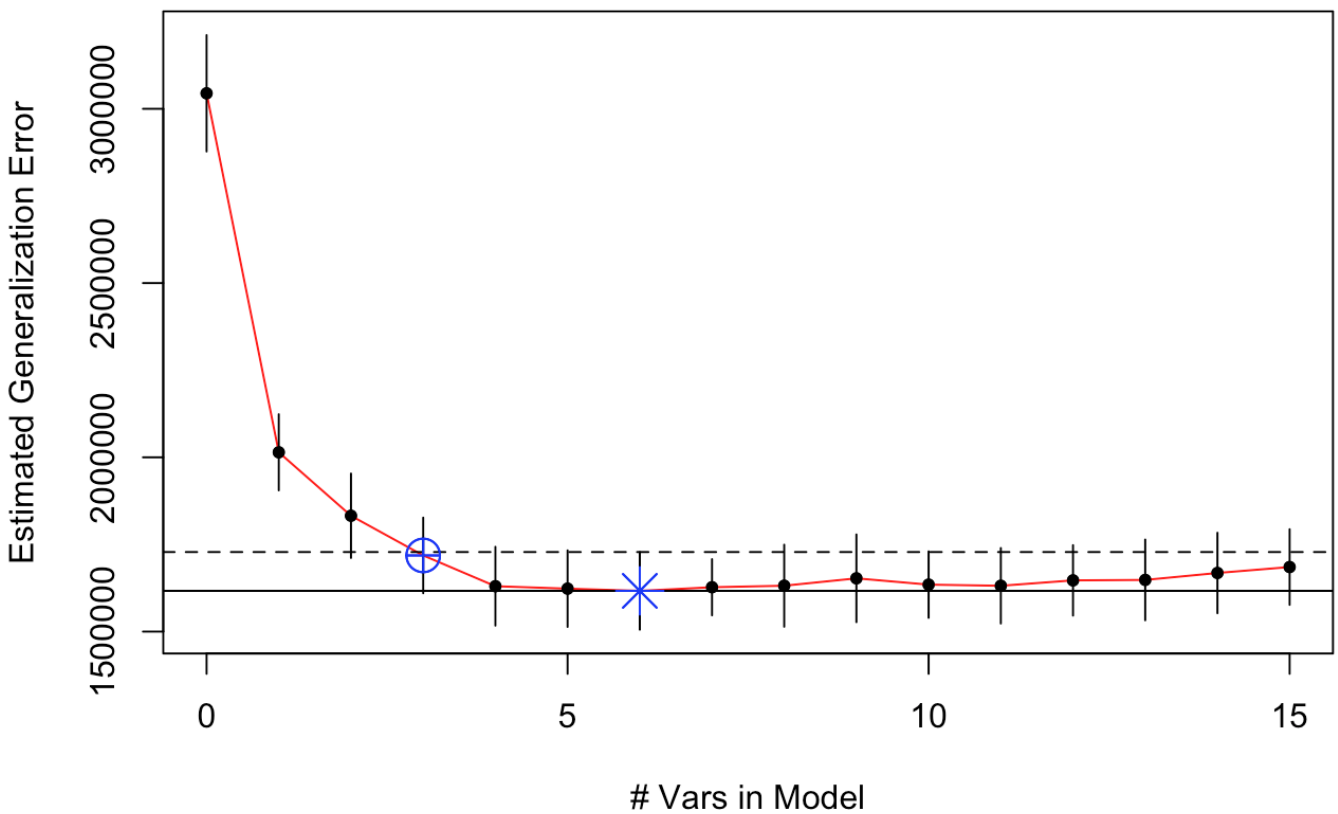


Response: EffectiveAvgTemp AvgHumidity AvgWindspeed EffectiveAvgTemp.AvgTemp

---

d2: Re-run `build.model` by considering as the most complex model the one with those four predictors along with all two-way interactions between them. Again, use `seed=320` and `prompt=FALSE` as arguments. Report the three variables in the model that are selected by the one standard deviation rule. Note: a `.` separating variable names represents an interaction between those variables.

```
BM2 <- build.model(Demand~EffectiveAvgTemp*AvgHumidity*AvgWindspeed*AvgTemp^2, dat
a=TRAIN,seed=320,prompt=FALSE) #complete command and uncomment
##
## Model with lowest estimated generalization error has:
## EffectiveAvgTemp AvgHumidity EffectiveAvgTemp.AvgWindspeed AvgHumidity.AvgWinds
peed EffectiveAvgTemp.AvgTemp EffectiveAvgTemp.AvgWindspeed.AvgTemp
##
## Model selected with one standard deviation rule has:
## EffectiveAvgTemp AvgHumidity EffectiveAvgTemp.AvgTemp
```



Response: EffectiveAvgTemp AvgHumidity EffectiveAvgTemp.AvgTemp

---

d3: Let us see which of the models we have considered actually end up performing the best on the `HOLDOUT` sample. Run the following code the find generalization errors of other models, then fit the model in d2 and find its generalization error. Which model has the lowest RMSE on the holdout sample?

```
#Remove the eval=FALSE when you are ready to knit
M.naive <- lm(Demand~1,data=TRAIN); generalization.error(M.naive,HOLDOUT)
## $RMSE.train
## [1] 1735.166
##
## $RMSE.holdout
## [1] 1816.443
M.full <- lm(Demand~.,data=TRAIN); generalization.error(M.full,HOLDOUT)
## $RMSE.train
## [1] 1321.072
##
## $RMSE.holdout
## [1] 1355.675
M.fullint <- lm(Demand~.^2,data=TRAIN); generalization.error(M.fullint,HOLDOUT)
## Warning in predict.lm(M, newdata = HOLDOUT): prediction from a rank-deficient f
it may be misleading
## $RMSE.train
## [1] 1064.165
##
## $RMSE.holdout
## [1] 1356.673
M.S1 <- lm(formula(S1),data=TRAIN); generalization.error(M.S1,HOLDOUT)
## $RMSE.train
## [1] 1255.16
##
## $RMSE.holdout
## [1] 1272.21
M.S2 <- lm(formula(S2),data=TRAIN); generalization.error(M.S2,HOLDOUT)
## Warning in predict.lm(M, newdata = HOLDOUT): prediction from a rank-deficient f
it may be misleading
## $RMSE.train
## [1] 1111.882
##
## $RMSE.holdout
## [1] 1303.5

M.cv <- lm(Demand~EffectiveAvgTemp+AvgHumidity+AvgWindspeed+EffectiveAvgTemp:AvgTe
mp,data=TRAIN)  #complete command and uncomment
generalization.error(M.cv,HOLDOUT) #complete command and uncomment
## $RMSE.train
## [1] 1262.869
##
## $RMSE.holdout
## [1] 1286.031
```

Response: S1 had the lowest RMSE error, followed closely by CV.

d4: In this case, the model selected using `build.model` with the aim of minimizing generalization error does not have the lowest RMSE on the holdout sample? Should we instead use as our predictive model the one that has the lowest RMSE on the holdout? Explain.

Response: No, you cannot change models at this point. You've already gone through and done the model selection based on the TRAIN and HOLDOUT samples. Fortunately, the difference in error between the two is not very large.

Question 2:

Load in `EX7.CATALOG` using the data command. The goal is to predict whether a customer makes a purchase in the next quarter (column `Buy` , levels are `Yes` and `No`) based on previous buying history.

```
#load in data
data("EX7.CATALOG")
```

---

a. Let us search for a good descriptive model by using the `step` command. Fit the naive model and full model (all predictors and all two-way interactions). Conduct the search by starting with the full model and using `direction="both"` . Note: *remember to add `trace=0` when knitting so that the output is surpressed*. Imagine the goal of the model is to answer the question: "Does the probability of buying increase or decrease with the number of catalogs the customer has received?". Looking at `summary` of the model (on your own, do not include this output), can we answer this question? If so, give the answer. If not, specify which additional characteristic(s) of the customer we must know in order to answer the question.

```
#define naive model
naive2 <- glm(Buy~1, data = EX7.CATALOG, family=binomial)
#define full model
full2 <- glm(Buy~.^2, data = EX7.CATALOG, family=binomial)

#S <- step( ... ,trace=0)   #complete command and uncomment
S <- step(full2,scope=list(lower=naive2,upper=full2), direction = "both", trace=0)
```

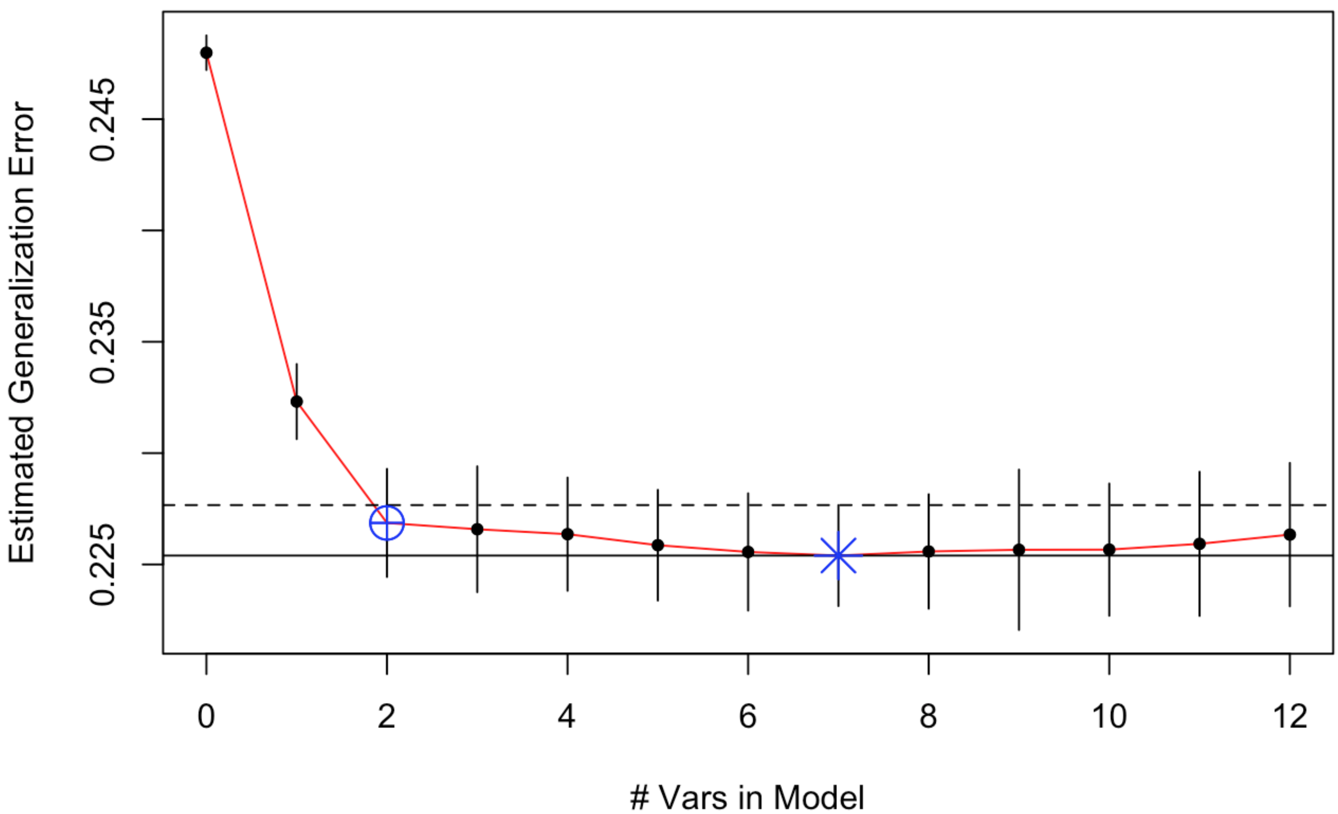
Response: Customers who receive a larger number of catalogs are more likely to buy something in the next quarter.

---

b. Let us build a model that should predict well on new data. Set the seed to 320 and split the data into 60% training and 40% holdout. Run `build.model` (with `seed=320` as an argument) and consider models without interactions.

```
set.seed(320)
#train.rows <-   #complete code and uncomment
train.rows2 <- sample(1:nrow(EX7.CATALOG),0.6*nrow(EX7.CATALOG))
#TRAIN <-        #complete code and uncomment
TRAIN2 <- EX7.CATALOG[train.rows2, ]
#HOLDOUT <-      #complete code and uncomment
HOLDOUT2 <- EX7.CATALOG[-train.rows2, ]

#Build model, with seed=320 as argument
BM3 <- build.model(formula(S),data=TRAIN2,seed=320,prompt=FALSE)
## Morgan-Tatar search since family is non-gaussian.
##
## Model with lowest estimated generalization error has:
## Intercept CatalogsReceived DaysSinceLastPurchase AvgOrderSize LifetimeOrder Per
centQuartersWithPurchase.LifetimeOrder CatalogsReceived.LifetimeOrder DaysSinceLas
tPurchase.AvgOrderSize
##
## Model selected with one standard deviation rule has:
## Intercept DaysSinceLastPurchase PercentQuartersWithPurchase.LifetimeOrder
```



```
formula(S)
## Buy ~ QuartersWithPurchase + PercentQuartersWithPurchase + CatalogsReceived +
##     DaysSinceLastPurchase + AvgOrderSize + LifetimeOrder + QuartersWithPurchas
e:DaysSinceLastPurchase +
##     PercentQuartersWithPurchase:DaysSinceLastPurchase + PercentQuartersWithPurc
hase:LifetimeOrder +
##     CatalogsReceived:LifetimeOrder + DaysSinceLastPurchase:AvgOrderSize +
##     DaysSinceLastPurchase:LifetimeOrder
```

---

c. Compare the misclassifcation rates on the holdout sample for the model found with the search (if you saved that as `S`, you can refit it easily on the training data by doing `S.p <- glm(formula(S),data=TRAIN,family=binomial)`) and for the model with the lowest estimated generalization error from b (which you will have to fit on the training data). Does either beat the misclassification rate of the naive model? If so, which is the better model?

```
M.search <- glm(formula(S),data=TRAIN2,family=binomial) #uncomment and run
generalization.error(M.search,HOLDOUT2) #uncomment and run
## $Confusion.Matrices
## $Confusion.Matrices$Training
##            Predicted No Predicted Yes Total
## Actual No           601           487  1088
## Actual Yes          378           934  1312
## Total               979          1421  2400
##
## $Confusion.Matrices$Holdout
##            Predicted No Predicted Yes Total
## Actual No           409           305   714
## Actual Yes          280           606   886
## Total               689           911  1600
##
##
## $Misclassification.Rates
## $Misclassification.Rates$Training
## [1] 0.3604167
##
## $Misclassification.Rates$Holdout
## [1] 0.365625
#Fit model with lowest estimated generalization error from b on TRAIN
S.p <- glm(Buy~QuartersWithPurchase+PercentQuartersWithPurchase+CatalogsReceived+D
aysSinceLastPurchase+AvgOrderSize+LifetimeOrder+QuartersWithPurchase:DaysSinceLast
Purchase+PercentQuartersWithPurchase:DaysSinceLastPurchase+PercentQuartersWithPurc
hase:LifetimeOrder+CatalogsReceived:LifetimeOrder+DaysSinceLastPurchase:AvgOrderSi
ze+DaysSinceLastPurchase:LifetimeOrder, data=TRAIN2,family=binomial)
generalization.error(S.p,HOLDOUT2)
## $Confusion.Matrices
## $Confusion.Matrices$Training
##            Predicted No Predicted Yes Total
## Actual No           601           487  1088
## Actual Yes          378           934  1312
## Total               979          1421  2400
##
## $Confusion.Matrices$Holdout
##            Predicted No Predicted Yes Total
## Actual No           409           305   714
## Actual Yes          280           606   886
## Total               689           911  1600
##
##
## $Misclassification.Rates
## $Misclassification.Rates$Training
## [1] 0.3604167
##
## $Misclassification.Rates$Holdout
## [1] 0.365625
#get misclassification rate of this model on HOLDOUT
```

```
M.naive2 <- glm(Buy~1,data=TRAIN2, family=binomial); generalization.error(M.naive
2,HOLDOUT2)
## Predicted classes same as naive model (majority class)
## Predicted classes same as naive model (majority class)
## $Confusion.Matrices
## $Confusion.Matrices$Training
##             Predicted Yes
## Actual No            1088
## Actual Yes           1312
##
## $Confusion.Matrices$Holdout
##             Predicted Yes
## Actual No             714
## Actual Yes            886
##
##
## $Misclassification.Rates
## $Misclassification.Rates$Training
## [1] 0.4533333
##
## $Misclassification.Rates$Holdout
## [1] 0.44625
M.full2 <- glm(Buy~.,data=TRAIN2, family=binomial); generalization.error(M.full2,H
OLDOUT2)
## $Confusion.Matrices
## $Confusion.Matrices$Training
##             Predicted No Predicted Yes Total
## Actual No            569           519  1088
## Actual Yes           369           943  1312
## Total                938          1462  2400
##
## $Confusion.Matrices$Holdout
##             Predicted No Predicted Yes Total
## Actual No            383           331   714
## Actual Yes           257           629   886
## Total                640           960  1600
##
##
## $Misclassification.Rates
## $Misclassification.Rates$Training
## [1] 0.37
##
## $Misclassification.Rates$Holdout
## [1] 0.3675
M.fullint2 <- glm(Buy~.^2,data=TRAIN2, family=binomial); generalization.error(M.fu
llint2,HOLDOUT2)
## $Confusion.Matrices
## $Confusion.Matrices$Training
##             Predicted No Predicted Yes Total
## Actual No            615           473  1088
```

```
## Actual Yes          392           920  1312
## Total              1007          1393  2400
##
## $Confusion.Matrices$Holdout
##            Predicted No Predicted Yes Total
## Actual No           407           307   714
## Actual Yes          300           586   886
## Total               707           893  1600
##
##
## $Misclassification.Rates
## $Misclassification.Rates$Training
## [1] 0.3604167
##
## $Misclassification.Rates$Holdout
## [1] 0.379375
M.S <- glm(formula(S),data=TRAIN2, family=binomial); generalization.error(M.S,HOLD
OUT2)
## $Confusion.Matrices
## $Confusion.Matrices$Training
##            Predicted No Predicted Yes Total
## Actual No           601           487  1088
## Actual Yes          378           934  1312
## Total               979          1421  2400
##
## $Confusion.Matrices$Holdout
##            Predicted No Predicted Yes Total
## Actual No           409           305   714
## Actual Yes          280           606   886
## Total               689           911  1600
##
##
## $Misclassification.Rates
## $Misclassification.Rates$Training
## [1] 0.3604167
##
## $Misclassification.Rates$Holdout
## [1] 0.365625
```

Response: The chosen model has the best rate of prediction by a small margin. Yes, it beats the naive model.