Math 505 Lab 7

This lab covers diagnostics for functional form, variable selection, and model validation.

1. Import the data set `Lab7Data.txt`, whose first column is Y and whose other 40 columns are X variables. Split the data into two parts, setting aside the first 1000 rows to fit the best possible model and the last 1000 rows to validate the model.

2. Begin by fitting a `bigmodel` that contains all of the X variables

```
Y=Lab7Data[1:1000,1]
X=Lab7Data[1:1000,2:41]
bigmodel=lm(Y~.,data=X)
```

3. The error term diagnostics indicate nonnormal heteroscedastic error terms and the need for a transformation of Y. Such a transformation would most likely lead to a good model, but we will assume that *no transformation of Y is necessary* and skip this step. At the end of the lab, we will obtain a final model with good diagnostics, including good error term diagnostics.

4. Look at plots of Y vs. $\hat{Y}$ and e vs. $\hat{Y}$ for `bigmodel`. Does curvature appear to be present? We will attempt to model this curvature by transforming the X variables instead of Y.

5. There are many variables in this model, many of which are likely irrelevant to the model. Let's use a stepwise process to narrow down the list of variables.

```
stepmodel=step(bigmodel)
summary(stepmodel)
```

6. We would like to use a best subsets method with delete-d cross-validation to narrow down the list of variables even more, but we can't, since there is a factor variable present (`v34`) with more than two levels.

   (a) How many levels does `v34` have (don't forget about the reference level)? The command `table()` might be helpful.

   (b) Try to group the levels into two categories (hint: consider grouping them on the basis of their coefficients). Create a new version of this variable called `subv34` with only two levels by grouping similar levels together (this is like the problem involving parent's education level on Lab 4).

   (c) Fit a univariate model by regressing Y on `v34`.

   (d) Create a submodel by regressing Y on `subv34`.

   (e) Compare these two models with an F-test. Does it appear that the submodel is adequate?[1]

7. Now that `v34` has been replaced by `subv34`, we can use best subsets with delete-d cross-validation.

   (a) First, create a data frame `Xy` whose first columns are all of the variables from `stepmodel` (except `subv34` instead of `v34`) and whose last column is Y. You can use the command `Xy=as.data.frame(cbind())`.

   (b) Now we can use best subsets. It's usually a good idea to save everything before using best subsets, since it can lock up the program. The tuning parameter of $t = 100$ below is the number of random subsets of d points that are deleted for each model in the algorithm. Large values of t will crash the program, while small values of t are not as effective. For this problem, $t = 100$ seemed to be a good compromise.

```
library(bestglm)
bestmodel=bestglm(Xy,IC="CV",family=gaussian,t=100)
summary(bestmodel$BestModel)
```

8. Now that we have reduced the number of variables with best subsets, let's investigate curvature for each variable.

   (a) Look at plots of Y vs. $\hat{Y}$ and e vs. $\hat{Y}$ for `bestmodel` to determine if curvature appears to be present.

   (b) For each remaining quantitative variable X_j, investigate curvature as follows:

   - Plot e vs. X_j. A trend indicates some type of problem with the functional form for that variable. Lack of trend means that the variable is probably ok.
   - Fit a second model that includes all the terms from `bestmodel` but also includes X_j^2. Then compare this new model to `bestmodel` with an F-test. A low p-value indicates that the second order term should be added. (If second order terms are not capable of resolving the curvature problem, one may need to include higher order terms, such as X_j^3 etc.)

   (c) Fit `model2`, which includes all the terms from `bestmodel` but also includes any needed second order terms.

   (d) Now we will investigate interaction terms. For each pair of remaining variables X_{j_1} and X_{j_2}, the need for an interaction term can be investigated as follows.

   - Plot e vs. $X_{j_1}X_{j_2}$. A trend indicates that the interaction term should probably be added.
   - Fit another model that includes all the terms from `model2` but also includes the interaction term $X_{j_1}X_{j_2}$, and compare the models with an F-test. A low p-value indicates that the interaction term should be added.

   (e) Fit `model3`, which includes all the terms from `model2` and any needed interaction terms. Check `model3` to make sure additional higher order terms or interaction terms do not need to be added. Also, check the error term assumptions of normality and constancy of variance. All of the diagnostics for `model3` should look good, and it is our tentative final model.

9. Finally, let's cross-validate `model3` using the last 1000 rows of the original data set.

   (a) Use `model3` to predict the value of Y_i, for each $i = 1001, \ldots, 2000$. Recall that model prediction uses the equation $\hat{Y} = X\hat{\beta}$. You will use the estimated coefficients you obtained in the previous problem from the first 1000 rows of data, but you will use the last 1000 rows of data for the X values. Here, X represents the model matrix for `model3`, not the entire set of 40 variables, and it will have to reflect second order terms, interaction terms, and the replacement of `v34` with `subv34`.

   (b) Compute the root mean square prediction error

$$RMSPE = \sqrt{\frac{1}{1000}\sum_{i=1001}^{2000}(Y_i - \hat{Y}_i)^2}.$$

   The RMSPE provides an estimate of how well this model will serve to make predictions for future data. The fact that RMSPE ≈ 3 is so close to $\hat{\sigma} \approx 3$ from `model3` is also evidence of a high quality model. Such strong agreement is the result of this data being simulated, and we wouldn't expect such close agreement with a real data set. Still, if RMSPE and $\hat{\sigma}$ aren't close at all, it suggests that the model merely fits the training data well and is not making good predictions for the validation sample.

   (c) Find the RMSPE for `bigmodel`, which regressed Y on all 40 X variables and didn't take any curvature or interactions into account. Doing this manually could be quite annoying, since there are three factor variables that we would have to create dummy variables for. Instead, we can use the `predict.lm` command.

```
Y=Lab7Data[1:1000,1]
X=Lab7Data[1:1000,2:41]

bigmodel=lm(Y~.,data=X)

validY=Lab7Data[1001:2000,1]
validX=Lab7Data[1001:2000,2:41]

bigpredY=predict.lm(bigmodel,newdata=validX)
```

   The last command uses the model `bigmodel` with the new data `validX` to predict values for Y and store them in the variable `bigpredY`. Now RMSPE can be computed using `validY` and `bigpredY`.
   Which model is superior in terms of RMSPE? What would happen if someone simply regressed Y on all 40 X variables without doing any further analysis – would they get a very good model?

   (d) One last thing we can do to validate `model3` is to fit this model to the entire data set. You will notice that the estimated coefficients and $\hat{\sigma}$ do not change very much, and all of the diagnostics still look good.

Notes

[1]The high F-test p-value indicates that the submodel is adequate, so for the remainder of this lab, we will use `subv34` instead of `v34`.