ITEC 621 Predictive Analytics

Addressing Dimensionality Issues in Predictive Models

Prof. J. Alberto Espinosa
Kogod School of Business, IT Department
Last updated November 11, 2016

Curse of Dimensionality

The "curse of dimensionality" (Bellman 1961) is a well-known problem in predictive modeling and it all boils down to a tradeoff between bias and variance. Simpler, low-dimension models (i.e., models with a few predictors) tend to be biased (i.e., omitted variables cause the included variable to be biased) but tend to be more stable in terms of variance. In contrast, more complex, high-dimension models (i.e., models with lots of predictors) tend to be less biased (i.e., more unbiased) but they suffer from higher variance – i.e., if you fit a model with several different random samples of the data you are likely to get very different coefficients each time. Finding the "right" set of variables for a model thus becomes a balancing act. There are many issues with high dimensionality, including overfitting, noise or nuisance (irrelevant) predictors, less interpretability, higher variance, less stable results. Naturally, the best way to compare models is with cross-validation with test data. Less stable, high variance models will yield poorer test cross-validation measures (i.e., Adjusted R^2, AIC, BIC, MSE, etc.).

Addressing High Dimensionality

Suppose that you have identified all possible predictors that your business domain knowledge tells you are relevant for your predictive model. Then, you have two possible extreme models: the **"null model"** (i.e., no variables, just the regression intercept, which is the same as using the mean and variance of the response variable to make predictions) and the **"full model"** which includes all available predictors. The best model will generally fall somewhere in between. Sometimes we identify a minimum set of predictors that must be included in the model because we believe from business experience that they all influence the response variable – let's call this the **"reduced model"**. And let's further assume that you want to test the inclusion of a few more (not all) additional variables – let's call this the **"extended model"**. We now have a series of 4 "nested models" starting with the full model, following with the other sub-models: full model → extended model → reduced model → null model:

Null Model: $Y=\beta_0$
Full Model: $Y=\beta_0+\beta_1X_1+\beta_2X_2+\ldots\ldots+\beta_pX_p$ (with p available predictors)
Reduced and Extended Models: somewhere in between

Dimensionality is addressed by evaluating these models and those in between. We generally evaluate models ranging from the null model to the full model, but there is no reason why we couldn't just focus on models ranging from the reduced to the extended model. Your business knowledge, not statics, should dictate this. For the purposes of this discussion, I will discuss three popular approaches to select the best model between the null and full models:

- **Variable selection** – adding, dropping and testing variables
- **Shrinkage** – keeping all or most variables, but shrinking their coefficients
- **Dimension reduction** – combining highly correlated variables into few components

Variable Selection

There are several approaches for variable selection, but an analyst should start by selecting only those variables in the data set that make sense from a business perspective. The most popular ones are:

- **ANOVA Testing** – Identify a number of interesting model subsets between the null and full models and test them in sequence with an ANOVA Wald's F test. Retain the larger model if the p-value is significant, select the smaller model otherwise.
- **Best Subset** – You can try each possible combination of predictors into all possible 1-variable models, 2-variable models, etc. However, this is computationally infeasible when there are a large number of predictors. For example, 20 predictors yield 2^{10}-1=1,048,576 possible models. Fortunately, **R** functions like ***regsubset( ){leaps}*** to generate the "best" 1-predictor model, 2-predictor model, etc. By default, ***regsubset( )*** only yields 8 models (1 to 8 predictors), but you can change with with the ***nvmax=xx*** argument (see R script). Important: these are NOT the best possible models because ***regsubset( )*** does not generate every possible model, just adequate subsets. Once you have all subsets, select the one with the largest adjusted R2 or lowest MSE, AIC or BIC. Preferably, test this using cross-validation.
- **Step Methods** – Adding and removing variables one at a time. This can be done 3 ways:
  - **Forward Inclusion** – Start with the **Null** (or Reduced) model and try all possible one-predictor models. Retain the one that is most significant. Then add a second predictor one at a time and retain the one that is most significant. Keep adding variables one at a time until no excluded variable is significant at the ***p<0.15*** level when included in the model (note the inclusion p-value can be changed, that is tuned – lower p-values will include fewer variables and vice versa). Test the final models for multicollinearity. You may need to remove high VIF predictors if there is severe multicollinearity. One shortcoming of the Forward Step method is that the starting model is the most biased (because it has a single predictor).
  - **Backward Elimination** – Start with the **Full** (or Extended) model. Then drop the least significant variable in the model. Then drop the next least significant predictor, and so on. Stop when all remaining variables are significant at the ***p<0.15*** level (note the inclusion p-value can be changed, that is tuned – lower p-values will remove more variables and vice versa). Test the final models for multicollinearity. You may need to remove high VIF predictors if there is severe multicollinearity. One shortcoming of the Backward Step is that the starting model is the one with highest variance, dimensionality and multicollinearity.
  - **Stepwise** – Start with either the Null (or Reduced) model and proceed Forward, or with the **Full** (or Extended) model and proceed Backward. If you go Forward, add one variable at a time as described above, but evaluate if any of the included predictors became non-significant at the ***p<0.15*** level. If so, the proceed one step backward and eliminate that predictor. Then continue forward. If you go Backwards, proceed as described above, but evaluate if any of the variables dropped earlier would become significant if added back to the model. If so, then add it back and continue Backwards. Test the final models for multicollinearity. You may need to remove high VIF predictors if there is severe multicollinearity. The shortcomings are similar to Forward and Backward, but Stepwise is usually preferred because the significance of variables tends to change when the model begins to experience multicollinearity.

- You can use either the ***regsubsets( ){leaps}*** or ***step( ){stats)*** functions to run any of the step methods above, which will show you all the models tested. My preference is with the ***step( )*** function because it provides more information on the variables included and excluded. See R script.

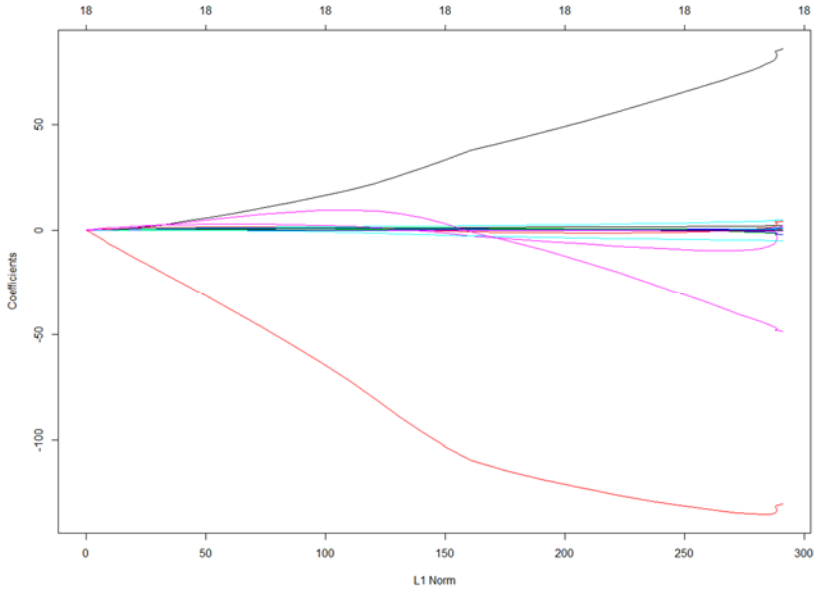
When to use variable selection? *There is no substitute for business domain knowledge. Your variable selection should be driven by your understanding of the business domain of analysis and less so by statistical association of variables, some of which may be spurious. Consequently, variable selection methods are generally considered to be "exploratory". It is perfectly fine to let one of these methods yield an initial model that is statistically sound, and then tweak it based on your business knowledge. It is perfectly OK to settle for a model with lower fit statistics if it makes business sense. In sum, variable selection should be used when there are a number of candidate variables and you want to explore which ones provide best predictive power, and when it is OK to add and remove variables. Be careful to test your final model for multicollinearity.*

Shrinkage or Penalized Models

The variable selection methods described above add or remove variables depending on their p-values. For example, if you use the Backwards method, every time you remove a predictor you are essentially **"shrinking"** its coefficient to 0. But rather than radically shrinking coefficients, why not shrink them a little at a time. Shrinkage methods do exactly that. They are popular methods when you have a business reason to retain all variables, but doing so causes multicollinearity and other dimensionality issues. For example, an OLS predictive model with 500 predictors will most likely suffer from multicollinearity. With the shrinkage method all coefficients are standardized (so they are all in a similar scale). You then start with the **Full** model, which is identical to OLS and start shrinking the coefficients a little at a time using the SSE formula provided in the lecture slides. With extreme shrinkage, all coefficients are "crushed" to 0 and you end up with a model that is identical to the **Null** model. How much shrinkage is done is controlled with a **"tuning"** parameter called the **"shrinkage factor"** λ (. There are 2 popular shrinkage methods:

- **Ridge Regression** – Instead of minimizing the SSE, Ridge minimizes the SEE plus λ times the sum of the squared coefficients (standardized) or $\boldsymbol{SSE + \lambda\,(\beta_1^2 + \beta_2^2 + \cdots + \beta_p^2 + etc.)}$. In essence:
  - $\lambda = 0$ does not do any shrinking, so the resulting model is exactly the Full OLS model $Y = \beta_0+\beta_1X_1+\beta_2X_2+\ldots\ldots+\beta_pX_p$ (with p available predictors)
  - $\lambda = \infty$ "crushes" all coefficients to 0, so the resulting model is exactly the Null model $Y = \beta_0$
  - So, just like the variable selection methods, the best model is somewhere between the Null and Full models. The difference with Ridge is that all models (except when $\lambda = \infty$) include all coefficients, but with shrinking values as λ becomes larger.
  - P-values are no longer relevant because we are retaining all coefficients. What matters is the "importance" of the variable as measured by the size of its standardized coefficient (the larger it is, the more standard deviations affecting the response variable).
  - It is important to note that **"all"** coefficients get shrunk as λ increases. As λ increases, the predictors become less and less important relative to the intercept. When $\lambda = \infty$ all coefficients disappear and the only thing left is the intercept (i.e., the null model), which is exactly at the mean of the response variable (i.e., same as having no regression model).

- o $\ell 2$ ***Norm*** (L2 norm) is a useful statistic in Ridge, equal to the square root of the sum of all standardized squared regression coefficients: $\ell 2 = \text{sqrt}(\beta_1^2 + \beta_2^2 + \ldots + \beta_p^2)$. It is an indicator of the aggregate amount of coefficient shrinkage in the model, for a given λ. When $\lambda = 0$, $\ell 2$ is exactly equal to the square root of the sum of the OLS squared coefficients, and when $\lambda = \infty$, $\ell 2$ is exactly 0 (i.e., λ and $\ell 2$ move in opposite directions). The graph below illustrates how the coefficients shrink as we change λ. Suppose that you fit a Ridge regression model with 100 different λ**'s** ranging from 10^{10} and 10^{-2} using this formula to store all 100 values in a vector named grid in **R**: ***grid = 10^seq(10,-2,length=100)***. You then generate all 100 Ridge regressions for the corresponding 100 values of λ, and store the results in an object called ridge.mod, with this command: ***ridge.mod = glmnet(x,y,alpha=0,lambda=grid)***. You can then plot the resulting coefficients for each value of $\ell 2$ ***Norm*** simply with the function ***plot(ridge.mod)*** (Note: the graph says L1 Norm, but it is actually L2 Norm. Again, λ is 0 at the far right and ∞ at the far left.



- LASSO, is almost identical to Ridge regression, except that LASSO minimizes $SSE + \lambda\,(|\beta_1| + |\beta_2| + \cdots + |\beta_p| + etc.)$. Where the vertical bars denote "absolute values".
  - o Like with Ridge, there is an overall shrinkage statistic called $\ell 1$ ***Norm*** (L1 norm) equal to the sum of the absolute values of the coefficients. Like with Ridge, $\ell 1$ ***Norm*** $= 0$ when $\lambda = \infty$ and $\ell 1$ ***Norm*** = sum of the absolute values of the OLS coefficients when $\lambda = 0$.
  - o Ridge and LASSO are competing similar models. But there is one notable difference between the two. With Ridge, the coefficients never become 0, except when $\lambda = \infty$. In contrast, absolute values have an interesting mathematical property that can actually shrink coefficients to 0 at values $<\infty$. Because coefficients can be removed by LASSO as λ grows, LASSO is situated somewhere in between variable selection (where coefficients are removed, but never shrunk) and Ridge (where coefficients are shrink, but never removed).

When to use shrinkage methods? *Shrinkage methods were not very popular for business applications because they were computationally very costly. With increasing computing power and versatility of*

statistical software like R, shrinkage methods are quickly becoming common place. In contrast to variable selection methods, shrinkage methods are less exploratory. However, you could use it in an exploratory fashion and decide to remove variables with very small coefficients, just to simplify your model. Still, variable selection is more appropriate when you have a small number of predictors (e.g., up to 20 or so) and multicollinearity is not a problem with the final model. Shrinkage methods are more appropriate when you have a very large number of predictors (hundreds or even thousands) and you want to retain all or most of them in the model. The nice thing with shrinkage methods is that you don't need to worry about multicollinearity. But you should be warned that your coefficients will be more biased than the corresponding OLS coefficients (because coefficients have been shrunk artificially). The good news is that with large models, shrinkage methods tend to reduce the model variance substantially, thus yielding more stable models and predictive accuracy. Of course, you can use cross-validation to evaluate the value λ *of that gives you better cross-validation test results than OLS.*

In sum, use shrinkage methods when you have a very large number of predictors and you want to keep all or most of them. Use Ridge is you want to keep all of them and LASSO if you want to keep most of them. No need to worry about p-values or multicollinearity with these methods.

Dimension Reduction Models

These methods actually take advantage of the correlation among predictors that often cause muticollinearity, to combine highly correlated variables into new variables called "components". It is important to note that most dimension reduction models are useless if the predictors have little correlation with each other, but naturally you have little to worry about multicollinearity in such cases. But if the predictors are highly correlated amongst each other we can take advantage of the predictor correlation structure to construct components that have little correlation with each other. This is achieved by creating linear combinations of the variables – e.g., a little bit of X1, plus a little of X2, etc.

For example, if I want to predict the likelihood of you becoming CEO's 10 years after completing your MS program, I would probably use predictors like GPA, attendance, grade in my class, your prior experience, etc. I would suspect that GPA, attendance and your grade in my class will be highly correlated. Your prior experience may have some correlation with these predictors, but probably not as much. Intuitively, I could simply standardize GPA, attendance and your ITEC 621 grade and average them out into a single component we will call "Academic Performance", and keep your prior experience as a separate predictor. However, your overall GPA may have a higher weight on your overall academic performance than attendance, etc. So perhaps I can combine these variables with different weights. Well, that is precisely what dimension reduction does, but in a more systematic way. There are several dimension reduction approaches, but the two most popular ones are Principal Components Regression (PCR) and Partial Least Squares (PLS). These methods are well explained in the lecture slides, so I only offer some further intuition here.

- **Principal Components Regression (PCR)** – when the predictors are uncorrelated any two or three variable plot will look like a sphere of dots without a noticeable alignment of the dots in any direction. This is an indication that correlation is very low and therefore PCR will not help much. However, as you so in those ***ggpairs( )*** plots, when predictors are correlated, the dots tend to align in one direction or another, showing a more elliptical cloud. The higher the correlation the more elongated the ellipse. In these plots, every predictor has its own axis. But if you start rotating the axes and align one of the along the direction of highest variance in the data, that axis is called the

"1st. Principal Component" (PC) and you can specify that axis as a linear combination of all variable (e.g., 32% of X1 plus 22% of X2, etc.). You then take the next axis and place it perpendicular to the 1st. PC and rotate it around it until you find the second direction of most variance. You continue on until you have done the same for all P axes. This is called a "rotation". Notice that you rotated the axes, but the data didn't move, so the correlation structure of the data is unchanged. It is only the perspective from the axes that has changed. But we now have P principal components (for P predictors) and they are all perpendicular to each other (i.e., they are "orthogonal" or uncorrelated). More importantly, the 1st. PC explains the largest portion of the variance in the data, followed by the 2nd., 3rd., and so on.

- PCR uses the correlation matrix of the P predictors and does this rotation for you until it finds all P PC's. Since the PC's are ordered from highest to lowest variance, the first M components (M<P) will explain a large share of the variance in the data. In highly correlated predictor sets, it is not uncommon to have, say 100 somewhat correlated predictors, and then find that the first 5 or 6 PC's explain 70% to 80% of the variance or more.
- The goal of PCR is first to do this rotation for you. Your tuning parameter is M, which is the number of components selected for your model. Instead of modeling a highly multicollinear regression with P predictors, you can build your model with the M PC's that explain a large proportion of the variance in the data. If P>>M, then you will have achieved a high degree of dimension reduction. More importantly, your M PC's are perfectly perpendicular (i.e., orthogonal or independent), so all problems with correlation go away.
- Again, your goal with PCR is to find an optimal value of M to achieve a high degree of dimension reduction, but explaining a high proportion of the variance at the same time. This can only happen if there is high correlation among the predictors, which is when PCR works well.
- Each of the M PC's is composed as a linear combination of all available variables – e.g., $M_1 = L_{11}X_1 + L_{12}X_2 + \ldots\ldots + L_{1P}X_P$ for the first PC, where L_{11}, L_{12} , etc. are the "loadings" for the 1st. PC, which are the fractional weight that each variable has on the PC. These loadings are constructed so that if you square them and add them up the result is 1. Same thing for M_2 and so on.
- So when you pick the optimal M, even if M<<P, all variables will be represented in the M PC's, but with varying weights.

Partial Least Squares (PLS)

- PCR and PLS are competing models. You should select one or the other based on cross-validation test results.
- One disadvantage of PCR is that it is an "unsupervised" method because it never uses the response variable when constructing the PC's. So, it is possible that the components do not correlate highly with the response variable, thus making them not so good as predictors.
- PLS corrects this by doing some further rotation to make the first few components more correlated with the response variable.
- The result is often a model that has better fit statistics and predictive power. It is not uncommon for PLS to outperform PCR. However, the further we rotate PCR axes the farther away we move from the optimal alignment of PC's in the directions of high predictor variance. So, it is possible that in some cases PCR may outperform PLS, thus the importance of cross validation testing.

- The interpretation of PLS is almost identical to PCR, with M components extracted that explain a high proportion of the predictor variance. Again, the goal is to have a small number of M PLS components, relative to the large number of P predictors.

When to use dimension reduction methods? *These methods are more effective when you have a large number of correlated predictors and you don't mind losing the interpretation by not having the original predictors in the model. If you have GPA, attendance and ITEC 621 grade in the model, you know exactly what they mean. If you replace them with a component called "Academic Performance" the coefficients are not as intuitive. Dimension reduction methods perform really well when you have a small data set and a large number of predictors. These are some of the only methods that can actually work even when you have more predictors than data points. For example, survey studies are notorious for having hundreds of items, but you may only have 100 respondents in your data set. In such cases, dimension reduction methods are a must.*

In sum, dimension reduction methods work well for high multicollinear data with many predictors and a relatively small number of observation, although it may perform well with large data sets too. PCR and PLS are truly alternative modeling options and it is recommended to run both models and select the one with better cross validation test results.

Overall Summary

- *Use your **business domain knowledge** so identify and select you candidate predictors*
- *Use **variable selection** if the number of predictors is relatively small (no more than 20 or 30 predictors) and you want to explore which are the best predictors, statistically. Adjust your predictor set using your business domain knowledge. Check your final model for multicollinearity and make corrections if needed.*
- *Use **shrinkage** methods if you have a very large number of predictors and you wish to retain all (Ridge) or most (LASSO) predictors in the model.*
- *Use **dimension reduction** if you have a large number of somewhat correlated predictors, especially (but not necessarily) with small data sets, and interpretation is not an important analysis goal. Try PCR and PLS and select the one with best cross validation test results.*