**Task 1: Recommend New Products**

For task one we were given the job of creating recommendations for new products to be added to the Blackwell line of goods based on a detailed list of 17 potential items. This involved creating an Excel workbook of spreadsheets to catalogue each item, perform some basic differential equations on weighted attributes, and ultimately establish which potential items among various key product categories was the best fit for the company. For me it became abundantly clear from the get-go that having a proper plan for laying out the Excel workbook was key because calculating multiple attributes across 17 products and the existing category merchandise can very quickly become confusing. My recommendation for anyone trying a similar project is to **create a design process before even starting to work** in Excel. This will save a lot of time and frustration from having to redo any possible calculations because you got yourself over your head by trying to be fancy. Often it seems like the simplest approach is the best. For me it worked best to give each of the 17 potential products their own worksheets. Maybe someone else would have just grouped the items by their categories and done all the calculations in one worksheet, but I thought at the time that that would be too much effort to read and understand and I thought my audience would as well. Having a design process really helped for me in keeping focused on whatever specific task was at hand no matter where I was at in the assignment.

This project did not cause any undue issues for me to complete it. I think in a way it was fun to do the work and plug the numbers in to create the Euclidian distances needed to establish the best picks for each project category. Even after the majority of the work in Excel was done and we transitioned to working in Weka to finalize the best results, using a k-nearest neighbor classifier, were not difficult. I think the challenge came from the job being very iterative, requiring the need to sometimes reevaluate the data in Excel and reconsider the weights placed on different attributes. And for that it really came down to trial and error as any gut feelings I had about some of the values, such as the different star ratings, came up to being unimportant. I found the true answers were in results for the calculations being done in Excel and Weka and not my experience or feelings about the different pieces of merchandise. The important lessoned learned here was to **be patient**. I remember hearing from one of my teammates “Why do we have to do any work in Weka?” as if the work we did in Excel was so conclusive that nothing more needed to be done, but after some debate what we found that if wanted a clear representation of the top five products to recommend the corporate office we would need a program to calculate that for us. Sure we could have added a worksheet to the Excel workbook and contrasted and compared the Euclidian distances between each potential product categories sales volumes, but it would not have been as easy and direct as inputting our data into Weka and using the program to help find the top five results for us based on sales revenue.

**Task 2: Get Started With R**

Task two saw us wade into the R programming language – installing the software, reading the introduction to R chapter in our book, and redoing the sales volume projections from task one. Starting the project we installed the R programming core software and RStudio files, which was straight forward enough for me as someone who has worked in an office environment for a number of years and knows how to run through software installation wizards in Windows. For anyone who did have an issue I imagine that, that would stem from installing RStudio first before the core R program and to fix that it is really as straight forward as trial and error of going back and uninstalling the software and trying again. I imagine for those people who had an issue it was a good lesson in reading over the course work to find the solution.

Next we read and worked through chapter 14 of the *Predictive Analytics for Dummies* book. This largely involved reading the chapter and mirroring the R code in the book as the author first created a regression model to predict results from a data set revolving around car models, then made a classification model based on types of wheat. This part of the assignment was fairly simple and without issue. I could see someone who has trouble following directions to the minutest detail having issues and needing additional help, but luckily not for me. I did find the tutorial in the book to be helpful in that it required me to actually type in the code and see the different functions produce the desired results. It would have been much less impactful if I had just read the chapter alone and not remade the code found in the book. So there needs to be continued emphasis on people recreating the code and not just reading the book.

The final part of this assignment was to demonstrate our growing knowledge of R programming by mimicking the work we did in task one of finding the highest revenue items among the potential products for Blackwell by using R programming instead of Excel and Weka. Personally I found this exercise to be very challenging because I am very visually orientated and programming for whatever reason has always been a challenge to me. Thankfully I started early on this assignment, which gave me extra time to run over the code and work out the kinks. For example unlike in chapter 14 where we used functions such as mape and mapeTable for this assignment we had to do something completely different. This lead to me needing extra time to work out why there was an issue with my code as I kept trying to use the mape function over and over again. I think in the future when someone does a similar assignment it would be helpful for them to know where the work diverts from the examples in the book especially since they are so new to the material. And if anyone needs a word of advice when starting out learning programming – **start early**. The worst thing anyone could do is delay getting into their course work for R or possibly any programming language as there are any number of issues someone may encounter. I cannot count how many times I could not run code just because a package in RStudio failed to stay active. Over time I learned to make it a habit to check my package selections to ensure that they were not the cause of my issues. And forming good habits all comes down to giving oneself time to develop them, so don’t delay, start early!

I really cannot say much more about the specifics for coding in the assignment. The major challenge I remember from programming in this case was having code not work exactly how it was supposed to and that was largely fixed by asking for help when needed and looking online for examples of existing code. So really for me it comes down to someone putting their nose to the grindstone and working on developing the code until it provides the desired results. At least that is how it worked for me this early in my learning how to program in R. One thing I will say about the process of starting out programming is to ask for help sooner rather than later. For me there were a number of times when I would just spend hours and hours trying to get code to work instead of just stopping at the realization that I did not know what to do. So for those new programmers who get stuck - **reach out for help**. I would even say make a process for asking for help. For example if you have spent more than 30 minutes trying to get a function to work and you have tried searching through all your resources for assistance, then that is a good time to ask for help. And once you ask for help do not stop working on the assignment. Instead I would take a break and come back to the work and take another crack at it. Who knows maybe taking some time away from programming will provide a fresh perspective.

**Task 3: Predict Which Brand of Products Customers Prefer**

With task three it was in a way a continuation of the learning we did in task two for R programming and learning how to make classification and regression analysis models. The big difference for this assignment was that we were introduced to the caret package, which provides access to both types of models in a relatively intuitive coding set. Once we familiarized ourselves with caret we moved on to addressing the issue of finding the most probable results for a customer survey data set. The issue at hand was that the data set was missing the customers’ answers for favorite brand, which was divided in outcome between Acer and Sony. To resolve this we created equally stratified training and test sets of data from a fully detailed customer survey in R and applied them to both k-nearest neighbor and Random Forest classifier models to not only get accurate predictions for which brand customers would prefer based on their other responses, but to see which of the two classifiers would ultimately be our favorite from all the testing experience we would have from them. After a lot of testing of the two classifiers, using the test set I created, I decided to go with Random Forest. The results I got proved to be slightly more accurate than what I found through kNN and it was easier to program. From finishing the testing of the classifiers it was all of matter of creating the predictions for the customers’ brand answers and delivering them to management for review.

This assignment for me was definitely more challenging than the coding work we did in task two. One reason for that being the case was that it required more research on my part to find examples of code online to help understand better on how to create the kNN and Random Forest classifiers in R. In task two there were some points when I had to do research online for R code examples, but not nearly to the depth that this assignment required for learning to comprehend the caret package and the methodology for running a test set that was missing a column of values. I wonder if it would have been helpful if some examples of code could have been provided, but then again it might have been for the best that we did not get examples on how to do this material. Once we finished this assignment I came to the conclusion it was better to have to do our own research and find resources that we personally found useful. Maybe that was not the intention of the lesson, but to me if someone asked me what to do when presented with similar work I would tell them to **start by researching helpful R programming sites** on the Internet. In my case when I was working on this project I found a number of sites that I would go to for one-off issues, which were just little problems that would come up, such as determining the best data types to use when working with a kNN classifier; most of the time, though, I would use sites such as Quora and Stackexchange to find help. They tended to be the ones I returned to most often and had the widest audience of responders to people’s questions about coding in R.

One thing I really liked and do not want to see go away is the introduction of the coding pipeline for this assignment. The straightforward concept of keeping a constantly amended and improving code set in document form for this project was integral to getting the work done on time and minimizing confusion for code placement as this assignment became more complicated by the addition of more and more functions. I have had previous experience with using coding pipelines by more primitive methods of using programs like Microsoft Word or Notepad to keep records of code I was using at the time.

With this assignment things were different in that I found that developing code required me to often go back and edit what I had written because the syntax was wrong for any number of reasons (incorrectly entered, outdated coding methods, function variable references were wrong, etc.). And for some of my teammates I have seen this be a significant hurdle, which I understand in that programming is a huge foreign concept to a lot of people. For me I think what I have learned from this assignment is that **programming is a craft** – an iterative process, one that requires people to work on their code a number of times in order to develop the correct results. I imagine in time it will become easier as the code and concepts behind them become engrained in how we program, but at least for now starting out there needs to be a repetitiveness to how we do things in order to learn how to do them better.