WHITE PAPER

Nearest Neighbor Methodology using PROC DISTANCE

Faisal Dosani
2 FOUR THREE INC

ABSTRACT
When faced with difficult and complex decisions, analysts tend to revert to past experiences in order to troubleshoot, and come up with solutions. This is the underlying concept of the nearest neighbor (KNN) approach a type of Memory based reasoning (MBR).

This paper explores the nearest neighbor methodology and implementing it in SAS. We will first explore what the methodology is and the techniques and tips to helping solve common business problems.

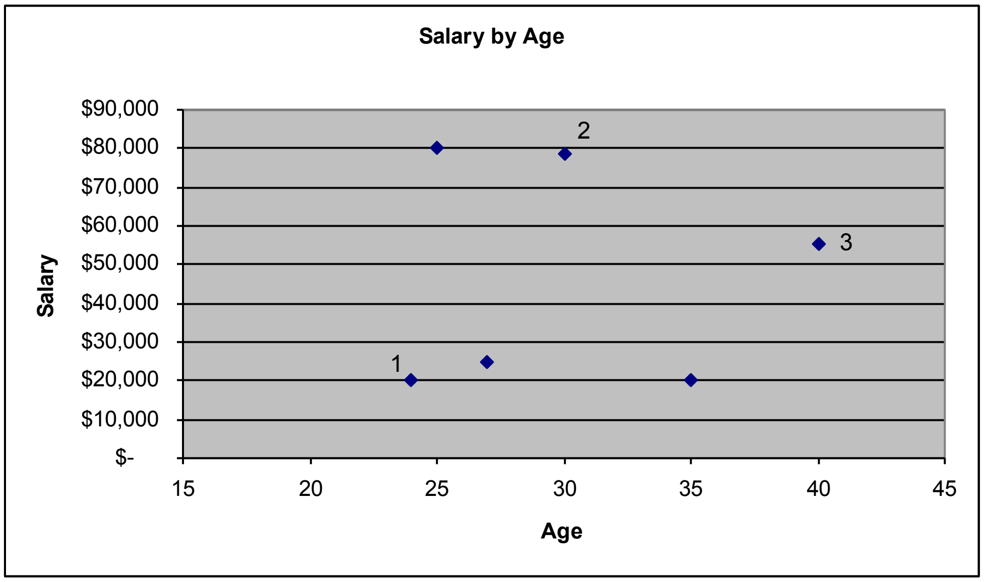
INTRODUCTION

The nearest neighbor approach has a wide range of applications from finance, medicine, fraud, to marketing. If we have information on past experiences we can use this to help find patterns and similarities to possible future events or classifications. A simple example of this is recognizing a particular accent. Someone from the London, England probably sounds closer to someone from Manchester, England, than a person from Toronto, Ontario. This goes back to an old adage: “Birds of a feather flock together”.

Memory based reasoning is integrated in our daily lives probably without us really even consciously thinking too much about it. From having partners with similar values and priorities to watching a movie because a friend with similar taste recommended it. Unlike a regression, which uses an actual formula to estimate, MBR techniques have the answers encoded within them; our goal is to decode them in an efficient and proper manner.

This idea of finding like things can have a powerful business context too. Whether it is detecting fraud on an account, or conducting a test on a new inbound calling strategy where you need to select a control group, there are many interesting and useful applications that translate into the real world.

THE BASICS



A simple graph of neighbors

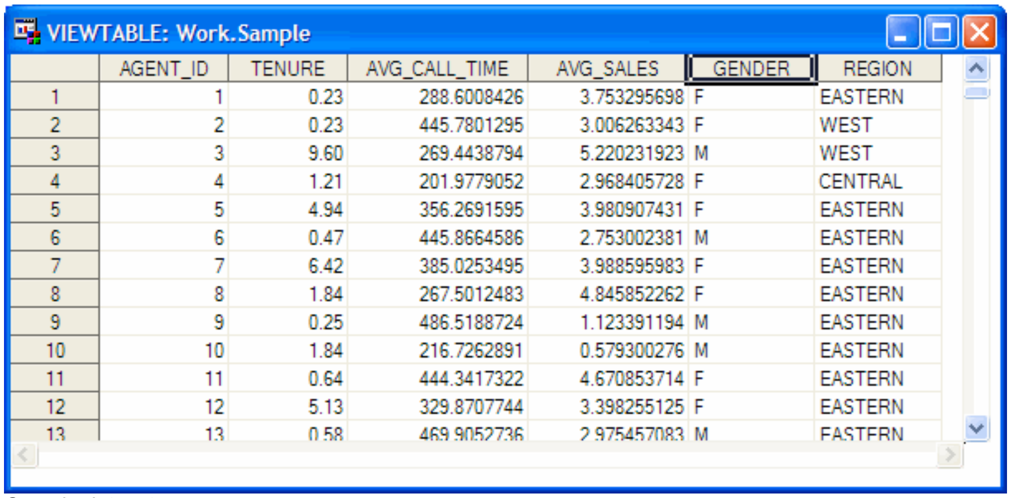
The above illustration represents a simple plot of the different salaries by age. Each of the blue points has logical nearest neighbor. The two points on the bottom left hand side will intuitively stand out as the two closest neighbors in the entire graph. The basic concept is to find the shortest distance between the points of interest. This notion of distance is what will drive our implementation of the near neighbor.

When we talk about distance we normally imagine something in meters, feet, kilometers or miles. But distance can related to essentially any continuous value. Take for example salary where the difference between someone making $90,000 a year verses $100,000 a year is $10,000. Our distance measurement would be $10,000.

This is relatively easy when we deal with single isolated variables because the scales for such variables are the same. What happens when we want to use multiple variables in combination? This is where things can become more complicated and require some planning on how to deal with our data. Going back to Salary as an example that can have a very wide range, age on the other hand usually doesn't go higher than 100. Conversely there are categorical variables and do not translate easily into a numeric value such as gender.

For this paper we will examine a pilot we are conducting for a call center. We want to provide a tool to help improve the performance of our agents. Before we spend money on the tool we would like to first run a pilot for 6 months to see improvements in sales. We pick 20 agents and give them the tool, and then see if their call center statistics improve, but how do we know if the tool is making the difference? It could have been that the chosen agents are generally more experienced. Perhaps the organization had a great year with immense growth as a whole. Perhaps it was purely random, and the tool played no role. We can not be 100% sure it is the tool or some external effect or randomness.

To help solve our dilemma we need to select a control group. This control group is meant to "look" like our test population and serves as fair comparison against our selected agents. If we properly select this control group, external changes that occur will be captured in both, and what we should be left with is the benefit of the tool. Nearest neighbor can help us select a group of control agents to whom we can compare against. Below is the sample of the dataset that we will be using, followed by the code used to generate it.


VIEWTABLE: Work.Sample

	AGENT_ID	TENURE	AVG_CALL_TIME	AVG_SALES	GENDER	REGION
1	1	0.23	288.6008426	3.753295698	F	EASTERN
2	2	0.23	445.7801295	3.006263343	F	WEST
3	3	9.60	269.4438794	5.220231923	M	WEST
4	4	1.21	201.9779052	2.968405728	F	CENTRAL
5	5	4.94	356.2691595	3.980907431	F	EASTERN
6	6	0.47	445.8664586	2.753002381	M	EASTERN
7	7	6.42	385.0253495	3.988595983	F	EASTERN
8	8	1.84	267.5012483	4.845852262	F	EASTERN
9	9	0.25	486.5188724	1.123391194	M	EASTERN
10	10	1.84	216.7262891	0.579300276	M	EASTERN
11	11	0.64	444.3417322	4.670853714	F	EASTERN
12	12	5.13	329.8707744	3.398255125	F	EASTERN
13	13	0.58	469.9052736	2.975457083	M	EASTERN

Sample dataset

```
DATA SAMPLE(DROP=REGRAND GENRAND);
format TENURE 5.2;
format REGION $20.;
        do AGENT_ID=1 TO 1000;
           /*TENURE WILL BE DISTRIBUTED AS LOG NORMAL*/
             TENURE = EXP(RANNOR(0));
           /*AVG_CALL_TIME WILL BE DISTRIBUTED AS UNIFORM BETWEEN 200-500 SECONDS*/
             AVG_CALL_TIME = RANUNI(0)*300 + 200;
           /*AVG SALES WILL BE DISTRIBUTED AS UNIFORM BETWEEN 0.5 - 5 PRODCUTS /
SERVICES*/
             AVG_SALES = RANUNI(0)*5 + 0.5;

           /*GENDER WILL BE M OR F*/
           GENRAND=RANUNI(0);
           IF GENRAND >= 0.5 THEN GENDER='M'; ELSE GENDER='F';

           /*REGION WILL BE WEST(20%), CENTRAL(10%), EASTERN(70%)*/
           REGRAND=RANUNI(0);
           IF REGRAND <= 0.2 THEN REGION='WEST';
           ELSE IF 0.2< REGRAND <= 0.3 THEN REGION='CENTRAL';
           ELSE REGION='EASTERN';
        output;
        end;
RUN;
```
SAS code to generate sample data

ANALYZING THE DATA

Before we can proceed we need to analyze our data. In our case this is a bit easier since we have simulated it for the paper. In the real world you would want to check distributions, make sure there are no missing, or null values. And even potentially transform data to make it behave in a certain way. The more you understand your data the more success you will have down the road in implementing the nearest neighbor methodology. Below is a table of our variables and information about them.

Variable Name	Numeric or Catagorical	Distribution	Description
Agent_Id	Numeric	N/A	This is a unique identifier for each agent.
Tenure	Numeric	Log Normal	Represents in the number of years how long the agent has been with the company.
Avg_Call_Time	Numeric	Uniform (200-500 Seconds)	Represents in the number of seconds on average the agent spends on each call
Avg_Sales	Numeric	Uniform (0.5 – 5 products)	Represents in the number of products and services on average the agent has booked on each call
Gender	Categorical	50/50 (M/F)	Gender of the Agent
Region	Categorical	20/10/70 (WEST/CENTRAL/EASTERN)	The call center region where the agent operates out of.

Now that we have an understanding of the data we need to think about the best way to convert this information into some sort of distance variable that we can use. One thing that may pop out is that our variables have different scales and inherently mean different things. If two agents have a difference of 5 in Tenure and a difference of 5 in Average Call Time, is this equivalent? These are different metrics, and 5 years of tenure doesn't equal 5 seconds in call time. Different scales and different metrics between variables can result in certain variables having more influence over others when calculating distance. We would essentially be giving the average call time the same weight as tenure, but we know they mean different things and have very different scales. We need a way to help standardize the data so we are actually talking about magnitudes rather than absolute values.

STANDARDIZING THE DATA

There are several techniques to standardize data; with the most common one is converting the values to z-scores. The central limit theorem that states that when you take independent random samples the sample mean will take on a normal distribution. A normally distributed variable should fall within 1 standard deviation 34.1% of the time. So the majority (68.2%) should fall within -1 and 1 standard deviations. The main take away is that we need to standardize our continuous variables and the best way to do this, is by creating a z-score with a mean of 0 and a standard deviation of 1. We can use the STANDARDIZE procedure in SAS to help with this.

```
PROC STANDARD DATA=SAMPLE MEAN=0 STD=1 OUT=SAMPLE_Z;
       VAR TENURE AVG_CALL_TIME AVG_SALES;
RUN;
```

SAS code to standardize data

The act of standardizing allows us to represent the different magnitudes of the variables on a similar scale. This is really important, as we don't want one to dominate another just based on its scale alone.

CATEGORICAL DATA

Categorical or nominal variables are a little less intuitive to deal with than their numeric counterparts. How do you quantify a male vs female or regions for that matter? There are some techniques that we can use to create values that will help in calculating distance. The key factor here is how many categories there are in your variable. If there are only 2, like in gender for example we can just simply create a binary variable of 1 or 0 to represent the values. It would be the equivalent of saying 2 males are similar to one another, and that 2 females are also similar to one another. A male and female are dissimilar though.

If our variable has more than 2 categories we can use dummy variables represent similarity and dissimilarity. There are a couple of techniques in doing this but the real key concept is here is consistency. Pick a method and stick with it for all like variables.

In our dataset we have region that has 3 possible values: West, Central, or East. We can represent these 3 values with 2 dummy variables. If we had 5 possible values it would be 4 dummy variables. The rule here is N-1. Where N is the number of values you want to represent. The tables bellow can help visualize what we are trying to achieve.

Gender	GENDER_IND
Male	0
Female	1

Region	REGION_IND1	REGION_IND2
WEST	1	0
CENTRAL	0	1
EAST	0	0

This type of classification will help us in determining if 2 items are similar or dissimilar, which will feed into our distance calculation. With the numeric values we dealt with previously we were saying that 10 is more than 5, but with this type of classification we are not saying the exact same thing. We are classifying similarity and dissimilarity. Below is the code to create our indicator variables:

```
DATA SAMPLE_FINAL;
SET SAMPLE_Z;

/*GENDER WILL BE 1 FOR FEMALE AND 0 FOR MALE*/
GENDER_IND=(GENDER='F');

IF REGION='WEST' THEN DO;
       REGION_IND1 = 1;
       REGION_IND2 = 0;
END;
ELSE IF REGION='CENTRAL' THEN DO;
       REGION_IND1 = 0;
       REGION_IND2 = 1;
END;
ELSE DO;
       REGION_IND1 = 0;
       REGION_IND2 = 0;
END;

/*WE NEED TO CREATE A CHAR VERSION FOR PROC DISTANCE*/
A_ID = left(trim(put(AGENT_ID,4.)));

RUN;
```
SAS code to deal with categorical variables

In certain situations it could make a lot more sense to give some weight if we have some extra insights into the data. For region we might want to give more preference to those in the WEST and CENTRAL regions so we could potentially give a further distance to those in the EAST with some coefficient to compensate.

Some other techniques for managing binary, nominal, ordinal, and various categorical include a simple matching coefficient, or a normalized rank transformation. Check out some of the references at the end for further information on different techniques and theories. The key is to experiment and figure out what works best for you.

DISTANCE

Now that we have our data in the correct format, and standardized to our linking we can start to think about how to calculate the distance in order to find out our closest neighbors. There are several ways to measure distance, we will look into the most common type: Euclidean.

Euclidean distance is the shortest distance from any 2 points. You might remember calculating this in high school math using Pythagoras theorem.

The mathematical formula for Euclidean distance is as follows for any point ($X_1,X_{2,..,}X_n$) to ($Y_1,Y_{2,..,}\ Y_n$):

$$d_{xy} = \sqrt{\sum_{i=1}^{n} \left(x_i - y_i\right)^2}$$

$$\begin{tikzcd} (X_1,X_{2,..,}X_n) \arrow[r] & (Y_1,Y_{2,..,}\ Y_n) \end{tikzcd}$$

Looking back at our first graph of salary by age lets calculate the distance for some of the points and see the affects of standardizing versus not.

Observation	X variable	Y variable	Formula	Distance to point 1 (d_{xy})
1	24	20000		
2	30	78544	$d_{xy} = \sqrt{(24-30)^2 + (20000-78544)^2}$	58544.00
3	40	55555	$d_{xy} = \sqrt{(24-40)^2 + (20000-55555)^2}$	35555.00

The above example shows that 1 and 3 are much closer than 1 and 2. This is mainly because of the scale of salary and how it can over influence the distance calculation. If we standardize the data we expect to see slightly different results.

Observation	Standardized X variable	Standardized Y variable	Formula	Distance to point 1 (d_{xy})
1	-0.99	-0.93		
2	-0.03	1.12	$d_{xy} = \sqrt{(-0.99 - -0.03)^2 + (-0.93-1.12)^2}$	2.26
3	1.58	0.32	$d_{xy} = \sqrt{(-0.99-1.58)^2 + (-0.93-0.32)^2}$	2.85

After standardizing we see that our original assumption was incorrect. 1 and 2 are actually closer than 1 and 3. We have evened the influence that both salary and age play into our distance calculation. This illustrates the importance of standardizing our data.

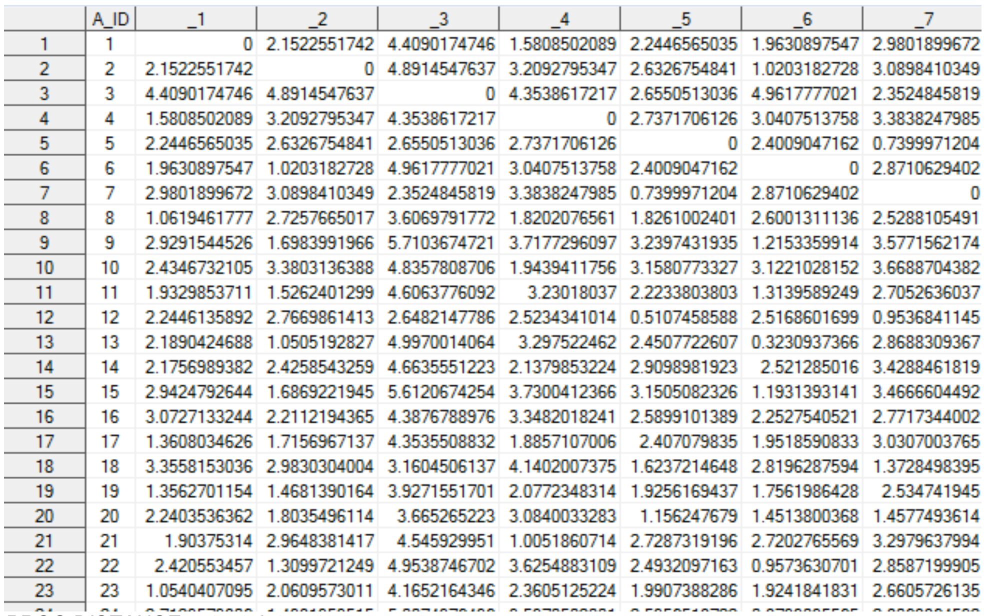
We have shown with a simple formula how you would go about calculating distance, but luckily for us PROC DISTANCE can do the heavy lifting. The procedure can take in a dataset and compute all the distances for us. You can specific the type of distance calculation you would like it to use with the default being Euclidean. There are other types of distance measurements such as City block. Depending on the intent you can set it up to use the many different methodologies. The basic code looks like this:

```
PROC DISTANCE DATA=SAMPLE_FINAL OUT=SAMPLE_DISTANCE SHAPE=SQUARE NOSTD;
ID A_ID;
VAR INTERVAL (TENURE AVG_CALL_TIME AVG_SALES GENDER_IND REGION_IND1 REGION_IND2);
RUN;
```

SAS code to calculate distance

- DATA: This is our final standardized datasets
- OUT: Is the dataset name of the distance matrix that will be created
- SHAPE: This denotes how the matrix will look. The matrix will be symmetrical as the distances from A to B and B to A will all be outputted. The default is Triangle which is without the duplication
- NOSTD: Turns off the standardization features. We have already done this and thus not necessary
- ID: will be the identification variable you want to output along with the distances.
- VAR: These are the variables that we wish to calculate distances for. You normally subset the variables into types. Interval being all your continuous numeric types. You can also have ordinal, nominal etc. Some types only work with certain distance formulas. Nominal for example doesn't work with the Euclidean method, which is why we placed our indicators into the interval section.

The output generated will look like the following:



	A_ID	_1	_2	_3	_4	_5	_6	_7
1	1	0	2.1522551742	4.4090174746	1.5808502089	2.2446565035	1.9630897547	2.9801899672
2	2	2.1522551742	0	4.8914547637	3.2092795347	2.6326754841	1.0203182728	3.0898410349
3	3	4.4090174746	4.8914547637	0	4.3538617217	2.6550513036	4.9617777021	2.3524845819
4	4	1.5808502089	3.2092795347	4.3538617217	0	2.7371706126	3.0407513758	3.3838247985
5	5	2.2446565035	2.6326754841	2.6550513036	2.7371706126	0	2.4009047162	0.7399971204
6	6	1.9630897547	1.0203182728	4.9617777021	3.0407513758	2.4009047162	0	2.8710629402
7	7	2.9801899672	3.0898410349	2.3524845819	3.3838247985	0.7399971204	2.8710629402	0
8	8	1.0619461777	2.7257665017	3.6069791772	1.8202076561	1.8261002401	2.6001311136	2.5288105491
9	9	2.9291544526	1.6983991966	5.7103674721	3.7177296097	3.2397431935	1.2153359914	3.5771562174
10	10	2.4346732105	3.3803136388	4.8357808706	1.9439411756	3.1580773327	3.1221028152	3.6688704382
11	11	1.9329853711	1.5262401299	4.6063776092	3.23018037	2.2233803803	1.3139589249	2.7052636037
12	12	2.2446135892	2.7669861413	2.6482147786	2.5234341014	0.5107458588	2.5168601699	0.9536841145
13	13	2.1890424688	1.0505192827	4.9970014064	3.297522462	2.4507722607	0.3230937366	2.8688309367
14	14	2.1756989382	2.4258543259	4.6635551223	2.1379853224	2.9098981923	2.521285016	3.4288461819
15	15	2.9424792644	1.6869221945	5.6120674254	3.7300412366	3.1505082326	1.1931393141	3.4666604492
16	16	3.0727133244	2.2112194365	4.3876788976	3.3482018241	2.5899101389	2.2527540521	2.7717344002
17	17	1.3608034626	1.7156967137	4.3535508832	1.8857107006	2.407079835	1.9518590833	3.0307003765
18	18	3.3558153036	2.9830304004	3.1604506137	4.1402007375	1.6237214648	2.8196287594	1.3728498395
19	19	1.3562701154	1.4681390164	3.9271551701	2.0772348314	1.9256169437	1.7561986428	2.534741945
20	20	2.2403536362	1.8035496114	3.665265223	3.0840033283	1.156247679	1.4513800368	1.4577493614
21	21	1.90375314	2.9648381417	4.545929951	1.0051860714	2.7287319196	2.7202765569	3.2979637994
22	22	2.420553457	1.3099721249	4.9538746702	3.6254883109	2.4932097163	0.9573630701	2.8587199905
23	23	1.0540407095	2.0609573011	4.1652164346	2.3605125224	1.9907388286	1.9241841831	2.6605726135

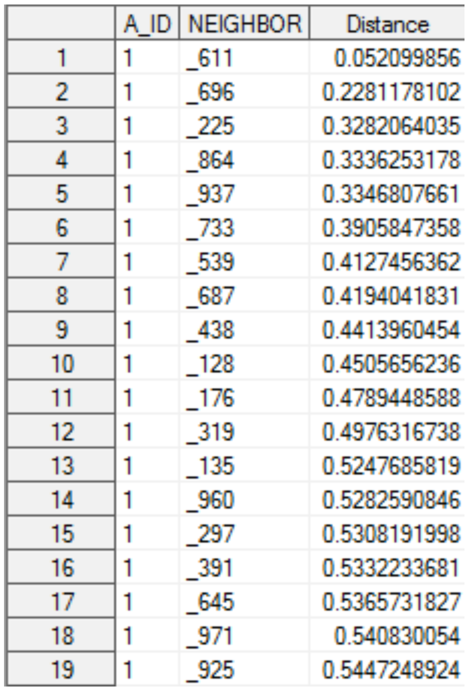
PROC DISTANCE output dataset

The hard part is now all done; we have a matrix representing all our observations and their distances from one another. We can further transpose and sort the matrix to generate a list for further analysis. You might notice that we transformed the AGENT_ID into A_ID. The reason for this is that the ID needs to be a character field rather than a numeric, which is what we originally simulated.

With a few more steps of transposing and sorting we can get to the final step where we have a list of nearest neighbors for each of our agents. We transpose so we can go from a square matrix into a column-based format, and the sorting just shows us the closest agents relative to any given agent.

```
PROC TRANSPOSE DATA = SAMPLE_DISTANCE
OUT = SAMPLE_DISTANCE_LIST
(where=(A_ID ne substr(NEIGHBOR,2))
rename = (col1=Distance))
NAME = NEIGHBOR PREFIX = COL;
BY A_ID NOTSORTED;
RUN;


PROC SORT DATA=SAMPLE_DISTANCE_LIST;
BY A_ID Distance;
RUN;
```



	A_ID	NEIGHBOR	Distance
1	1	_611	0.052099856
2	1	_696	0.2281178102
3	1	_225	0.3282064035
4	1	_864	0.3336253178
5	1	_937	0.3346807661
6	1	_733	0.3905847358
7	1	_539	0.4127456362
8	1	_687	0.4194041831
9	1	_438	0.4413960454
10	1	_128	0.4505656236
11	1	_176	0.4789448588
12	1	_319	0.4976316738
13	1	_135	0.5247685819
14	1	_960	0.5282590846
15	1	_297	0.5308191998
16	1	_391	0.5332233681
17	1	_645	0.5365731827
18	1	_971	0.540830054
19	1	_925	0.5447248924

PROC TRANSPOSE and SORT code / output dataset

In the PROC TRANSPOSE we want to omit all the records where the A_ID is also the neighbor. We have the where clause for this along with renaming and additional options to help the procedure deal with numbers stored as strings (A_ID). We are now able to pick our control group. If agent 1 was part of our test group we know that agent 611 is very similar to him or her, and would be a good candidate in our control population.

If used correctly nearest neighbor can be a powerful data mining technique in selecting control groups, performing classifications, or clustering similar observations together. In a recent R&D session with a client the idea of using this to isolate outliers was also proposed and in theory would be relevant. In our context we were looking for similarity, but this could easily be used to isolate dissimilar observations as well.

CONCLUSION

The nearest neighbor methodology can be a powerful data mining technique used to solve difficult and complex problems. We can pick like groups, create clusters, and even potentially discover multivariate outliers. With a few relatively simple steps we can standardize our data, calculate the distances on a multidimensional space, and find observations which are similar or dissimilar. As with any realm of data sciences there is often a component of art where we need to think of intent and a full understand business problems before applying a technical solution towards it.

REFERENCES

SAS Institute Inc. 2012. Base SAS® 9.3 Procedures Guide, Second Edition. Cary, NC: SAS Institute Inc

sasCommunity.org. 2011. “Finding nearest neighbors” http://www.sascommunity.org/wiki/Finding_nearest_neighbors (November 23, 2013).

Teknomo, Kardi. 2006. “Similarity Measurement” http://people.revoledu.com/kardi/tutorial/Similarity/index.html (November 23, 2013).

Greenacre, Michael. 2008. “Measures of distance between samples: Euclidean” http://www.econ.upf.edu/~michael/stanford/maeb4.pdf (November 23, 2013).

AKNOWLEDGEMENTS
I would like to thank the following people for their insights and help in writing this paper:

- Keith Marcus
- Ozgur Cetiner
- And Anne Conrad.

CONTACT
Your suggestions and questions are welcomed. Please contact the author at:

Faisal Dosani
faisal@2four3.com
http://www.2four3.com