Synthesis of data using existing methods

David Pugh, July 2018

As part of the Synthetic Data projects we will look at some existing data synthesis techniques and see how and when they are used to create synthetic data. In this brief report, we focus on a key set of algorithms used to balance datasets for classification.

Addressing the imbalance

A dataset is imbalanced if the classification categories are approximately equally represented. Many real-world datasets are imbalanced, comprising of predominantly 'normal' examples with only a small percentage of 'abnormal' examples. It is these minority classes that are often of most interest. e.g., detecting instances such as fraud or rare cancers.

Machine learning algorithms have trouble learning when one class dominates the other. They can learn to class all inputs as the majority class and still achieve high scores. The evaluation of algorithm performance using predictive accuracy alone in the case of imbalanced datasets is not appropriate. Often applications require a high rate of detection of the minority class and allows a small error rate in the majority class, often viewed in a confusion matrix. The Receiver Operating Characteristic (ROC) curve is a standard technique for summarising classification performance over a range of trade-off between true positive (TP) and false positive (FP) error rates. The Area Under the Curve (RUC) and ROC convex hull are traditional performance metrics for a ROC curve.

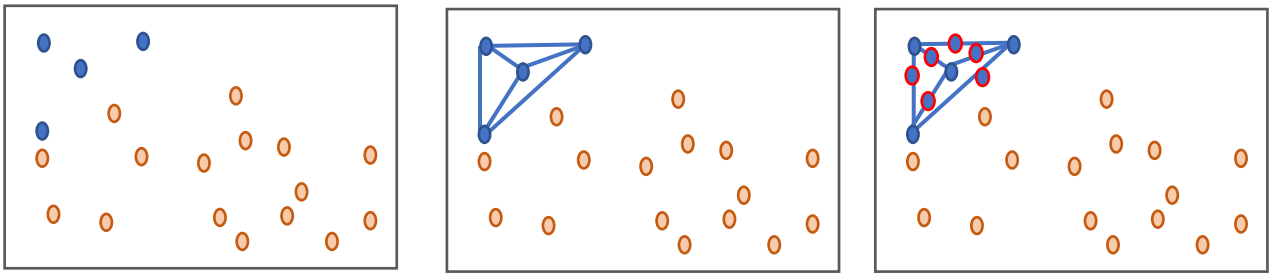
There are a number of approaches to addressing class imbalance and increase sensitivity to the minority class:

- Synthesis of new minority class instances
- Over-sampling of minority class
- Under-sampling of majority class
- Adjust the cost function to make misclassification of the minority instances more important that misclassification of majority instances

In this summary, we will focus on the first approach, looking at some well used techniques to create data to boost the number of minority classes.

Techniques for minority synthesis

Synthetic Minority Over Sampling – SMOTE – synthesises new minority instances between existing (real) minority instances. The technique was proposed in 2002 (Chawla, Bowyer, Hall, & Kegelmeyer, 2002) an is now an established method, with over 85 extensions of the basic method reported in specialised literature (Fernandez, Garcia, Herrera, & Chawla, 2018). It is available in several commercial and open source software packages. A way to visualise how the basic concept work is to imagine drawing a line between two existing instances. SMOTE then creates new synthetic instances somewhere on these lines.



In the above example, we start with an imbalance of 4 blue vs 16 orange instances. After synthesising it is now 10 blue vs 16 orange instances, with the blue instances dominating within the ranges typical for the blue values. This is a key aspect of the technique –over-sampling with replacement focussed data in very specific regions in the decision region for the minority class. It does not replicate data in the general region of the minority instances, but on the exact locations. As such it can cause models to overfit to the data. In SMOTE synthetic data is used rather than replacement. The minority classes are oversampled by taking each minority class sample and introducing synthetic examples along the segments joining any/all of the *k* minority class nearest neighbours. This forces the decision region of the minority class to be more general, leading to better generalising decision trees. Considering a sample x_i, a new sample x_{new} will be generated considering its *k* nearest neighbours. The steps are

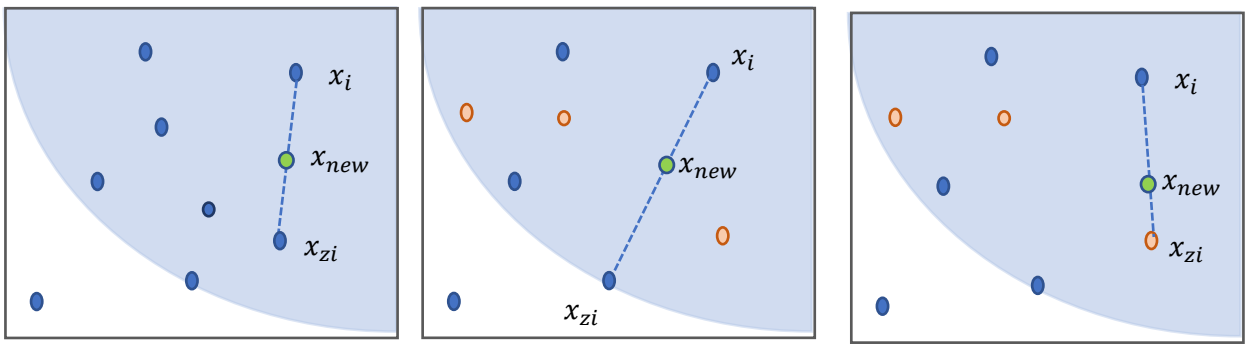
- take the difference between the feature vector sample x_i under consideration and its nearest neighbour x_{zi}
- multiple the difference by a random number γ between 0 and 1 and add this to the sample vector to generate a new sample x_{new}

$$x_{new} = x_i + \gamma(x_{zi} - x_i)$$

A variation of this approach is Adaptive Synthetic (ADASYN) sampling method. This operates in a similar way as regular SMOTE, except the number of samples generated for each $x_i$is proportional to the number of samples which are not from the same class of $x_i$in a given neighbourhood. SMOTE will connect inliers and outliers in the data, while ADASYN can focus solely on outliers. This can sometimes lead to suboptimal decision functions. To help address this SMOTE has different implementation options to generate samples (in fact over 85 different extensions to the regular SMOTE methods have been proposed). For example, in the SciKit Learn imbalanced-learn library these methods - SVM, Borderline1 and Borederline2 - focus on samples near the border of the decision function and will generate samples in the opposite direction of the nearest neighbour class.

The ADASYN and the SMOTE variants differ in the way they select the samples x_i ahead of generating new ones.

Sampling method	**Choice of**
SMOTE regular	Randomly pick up all possible x_i
SMOTE borderline1 SMOTE borderline2	Classifies each sample x_i to be 1) noise (all nearest neighbours from a different class), 2) danger (at least half the nearest neighbours are from same class) or 3) safe (all nearest neighbours are of the same class). Borderline SMOTE operates only on the danger samples. For Borderline1, x_{zi}will belong to a class different from that of x_i For Borderline2, x_{zi}will belong to any class
SMOTE SVM	Uses an SVM classifier to find support vectors and generate samples using them.
ADASYN	Similar to regular SMOTE, except the number of samples generated for each x_i is proportional to the number of samples which are not from the same class that $x_i$in a given neighbourhood.



Examples of regular SMOTE (left), Borderline 2 (middle) and Borderline 1 (right) for k=6 – the difference is how the x_i value is selected before finding x_{zi} and generating x_{new}. For regular SMOTE x_i is randomly selected from all minority class examples. For borderline, we only pick samples that are classed as 'in danger' – i.e., at least half the nearest neighbours are from the opposite class. For borderline 1, x_{zi} is from a different class to x_i, for borderline 2 it can be from any class.
For ADASYN on our right-hand example, we would generate n samples, where n is proportional to 3 (the number of samples not from the same class as x_i).

Application for General Synthetic Data Generation

Can SMOTE be used to generate synthetic data? This was assessed using the following approach:

- Take an existing dataset with *n* entries, make imbalanced (by replicating the dataset and assigning a target class) or generate a dataset with different features and an imbalance of 2:1 (this results in a generated dataset with the same number of samples at the original)
- Run SMOTE (all variants) to generate new data samples (*n* new samples)
- Remove new samples from new dataset
- Compare against original dataset – perform a correlation matrix on all the original data, all the generated data and then subtract the two correlation matrices. The resulting correlation matrix should be all zero for well-matched generated data.
- Use the Bhattacharyya distance to measure the similarity between the original and generated probability distributions

This was implemented in Python using the imbalanced-learn module. The following datasets were used:

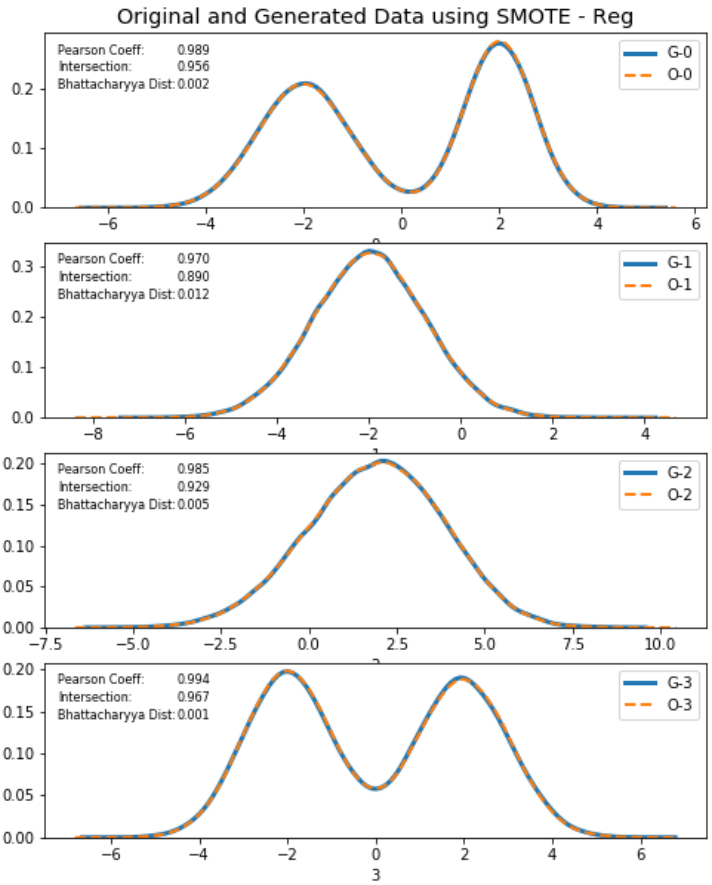
1. Randomly generated data
2. ICS Adult Census data
3. LSOA Atlas Census data
4. SciKit Learn Cancer data

Quality of the generated data was assessed using:

- Pearson R correlation, intersect and Bhattacharyya distance between the original and generated distributions for each feature. These measure how closely the synthetic data for each individual feature matches the real data.
- Cross correlation matrix difference – subtraction of the cross correlation matrix of the real dataset and the generated dataset. If the two datasets are well matched this will be close to zero for the whole matrix.

Generated Data

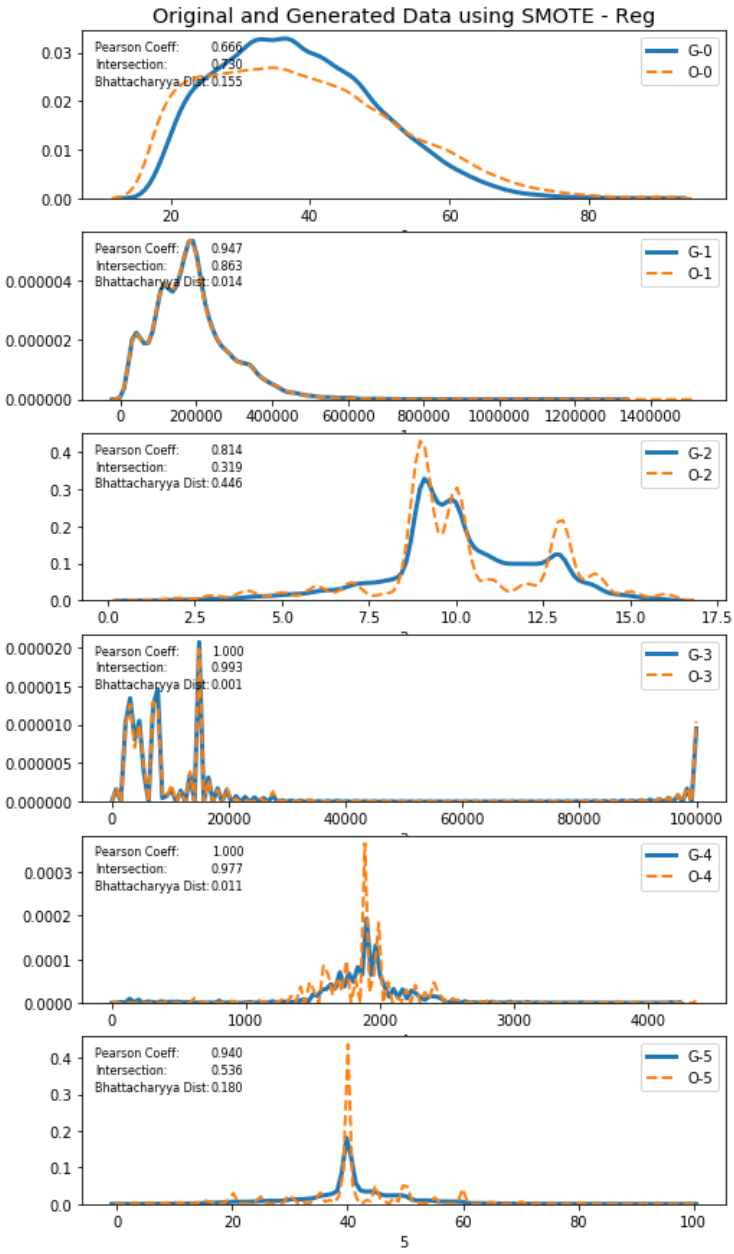
Used the SciKit Learn *make_classification* method to create a dataset of 100,000 samples, 4 features and a target class ratio of 1:9.



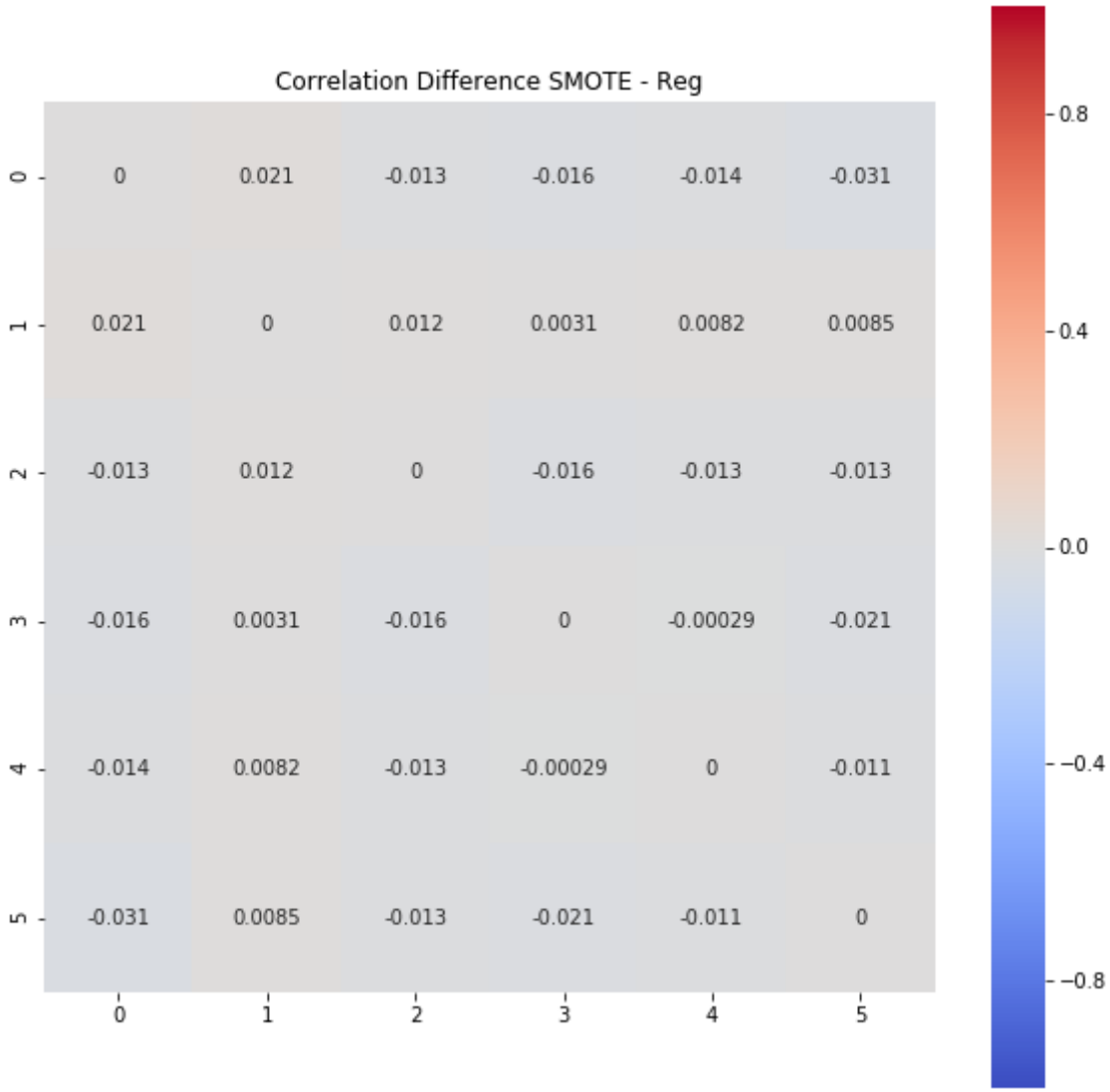
Method	Feature	Pearson R	Intersection	Bhattacharyya Distance
ADASYN	0	0.987	0.938	0.004
	1	0.972	0.892	0.012
	2	0.959	0.882	0.013
	3	0.999	0.985	0.000
SMOTE	0	0.989	0.956	0.002
	1	0.970	0.890	0.012
	2	0.985	0.929	0.005
	3	0.994	0.967	0.001
SMOTE B1	0	0.989	0.956	0.002
	1	0.970	0.890	0.012
	2	0.985	0.929	0.005
	3	0.994	0.967	0.001
SMOTE B2	0	0.994	0.961	0.002
	1	0.982	0.914	0.006
	2	0.993	0.949	0.003
	3	0.923	0.880	0.015

ICS Adult Census Data

The 6 numeric features were used with 32,560 entries. The original and generates data for this more complex data set is shown below for the regular SMOTE approach. This was typical for all variations. For some distributions the generated data matches the original closely but it has a tendency to smooth and generalise the data if it is particularly noisy and can distort some asymmetrical distributions. We see this reflected the quality measures, particularly the intersect and Bhattacharyya distance. The difference in cross correlation matrix figure of merit is also good for this data.



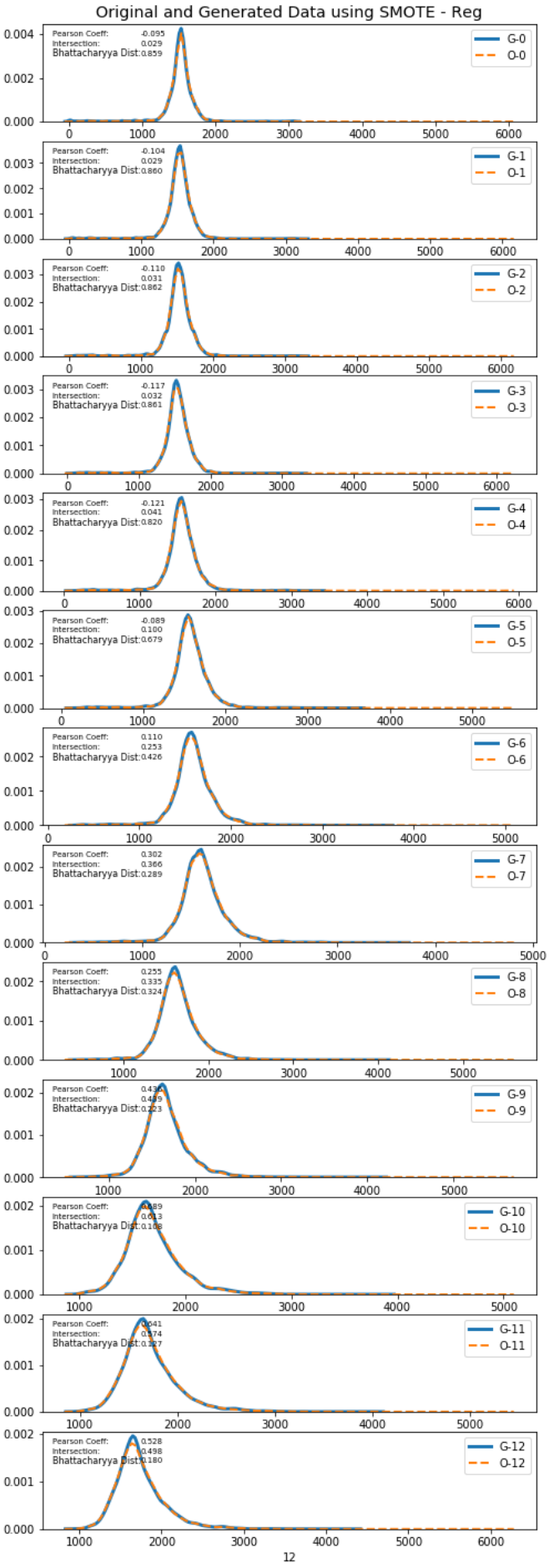
Method	Feature	Pearson R	Intersection	Bhattacharyya Distance
ADASYN	0	0.692	0.741	0.149
	1	0.992	0.947	0.004
	2	0.813	0.318	0.445
	3	1.000	0.994	0.001
	4	1.000	0.967	0.018
	5	0.100	0.967	0.425
SMOTE	0	0.666	0.730	0.155
	1	0.947	0.863	0.014
	2	0.814	0.319	0.446
	3	1.000	0.993	0.001
	4	1.000	0.977	0.011
	5	0.940	0.536	0.180
SMOTE B1	0	0.666	0.730	0.155
	1	0.947	0.863	0.014
	2	0.814	0.319	0.446
	3	1.000	0.993	0.001
	4	1.000	0.977	0.011
	5	0.940	0.536	0.011
SMOTE B2	0	0.701	0.745	0.147
	1	0.952	0.870	0.012
	2	0.571	0.240	0.548
	3	1.000	0.993	0.001
	4	1.000	0.975	0.013
	5	0.936	0.528	0.194

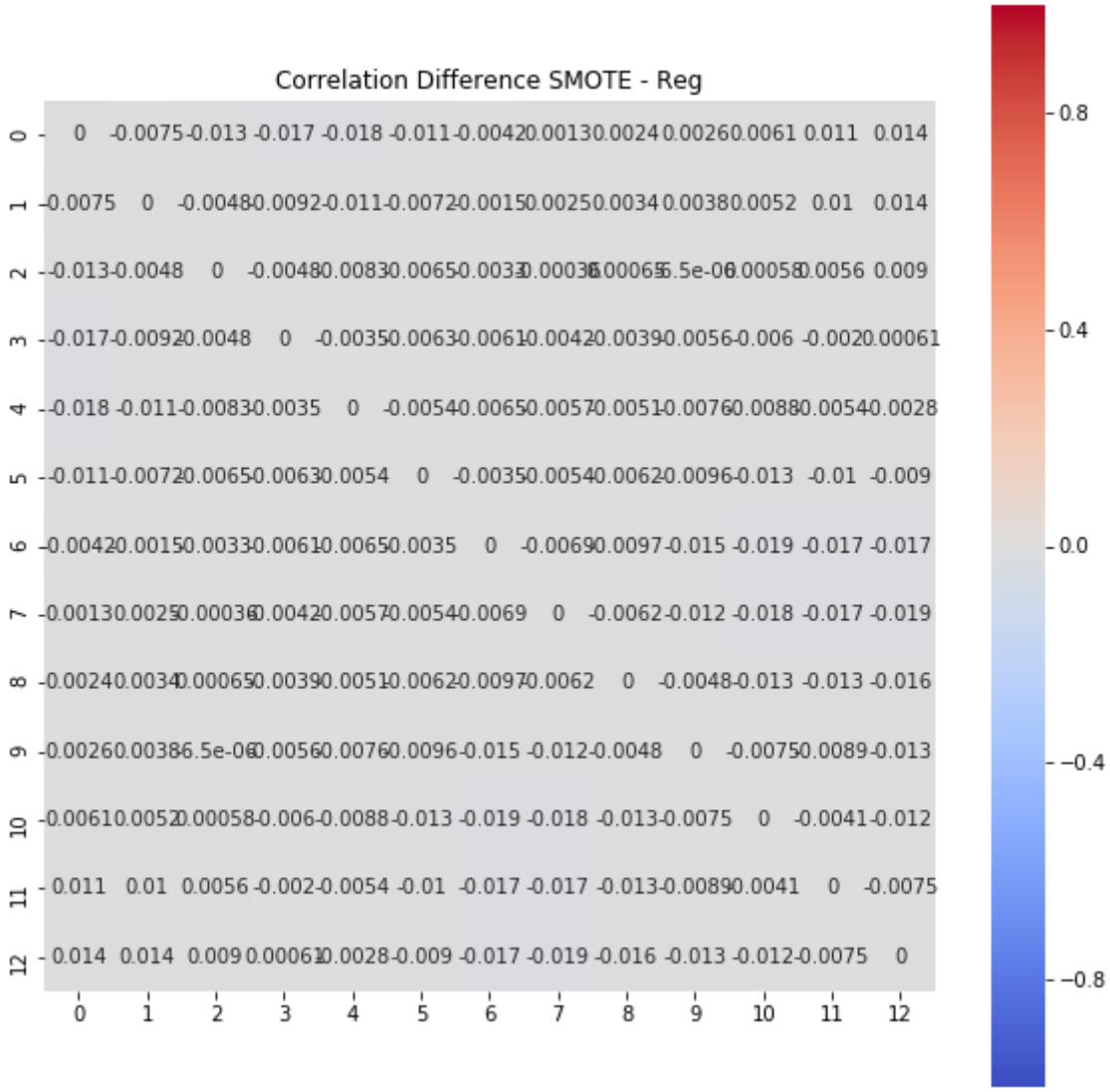

Correlation Difference SMOTE - Reg
0
1
2
3
4
5
0 0.021 -0.013 -0.016 -0.014 -0.031
0.021 0 0.012 0.0031 0.0082 0.0085
-0.013 0.012 0 -0.016 -0.013 -0.013
-0.016 0.0031 -0.016 0 -0.00029 -0.021
-0.014 0.0082 -0.013 -0.00029 0 -0.011
-0.031 0.0085 -0.013 -0.021 -0.011 0
0
1
2
3
4
5
0.8
0.4
0.0
−0.4
−0.8

LSOA Census Data

First 13 numerical features of the LSOA datatset were taken and synthesised. We see a good match between the original and generated datasets.

Method	Feature	Pearson R	Intersection	Bhattacharyya Distance
ADASYN	0	-0.095	0.030	0.858
	1	-0.103	0.028	0.859
	2	-0.109	0.031	0.852
	3	-0.107	0.035	0.833
	4	-0.113	0.050	0.787
	5	-0.082	0.115	0.661
	6	0.025	0.204	0.501
	7	0.176	0.294	0.368
	8	0.145	0.277	0.397
	9	0.491	0.467	0.201
	10	0.736	0.646	0.093
	11	0.685	0.606	0.113
	12	0.555	0.512	0.168
SMOTE	0	-0.095	0.029	0.859
	1	-0.104	0.029	0.860
	2	-0.110	0.031	0.862
	3	-0.117	0.032	0.861
	4	-0.121	0.041	0.820
	5	-0.089	0.100	0.679
	6	0.110	0.253	0.426
	7	0.302	0.366	0.289
	8	0.255	0.335	0.324
	9	0.436	0.439	0.223
	10	0.689	0.613	0.108
	11	0.641	0.574	0.127
	12	0.528	0.498	0.180
SMOTE B1	0	-0.095	0.029	0.859
	1	-0.104	0.029	0.860
	2	-0.110	0.031	0.862
	3	-0.117	0.032	0.861
	4	-0.121	0.041	0.820
	5	-0.089	0.100	0.679
	6	0.110	0.253	0.426
	7	0.302	0.366	0.289
	8	0.255	0.335	0.324
	9	0.436	0.439	0.223
	10	0.689	0.613	0.108
	11	0.641	0.574	0.127
	12	0.528	0.498	0.180
SMOTE B2	0	-0.081	0.049	0.806
	1	-0.082	0.058	0.781
	2	-0.082	0.069	0.754
	3	-0.006	0.149	0.581
	4	0.657	0.570	0.131
	5	0.909	0.785	0.034
	6	0.985	0.914	0.007
	7	0.997	0.955	0.004
	8	0.988	0.925	0.007
	9	0.989	0.937	0.004
	10	0.988	0.933	0.004
	11	0.988	0.932	0.006
	12	-0.095	0.029	0.859


Original and Generated Data using SMOTE - Reg
Pearson Coeff: -0.095
Intersection: 0.029
Bhattacharyya Dist: 0.859
G-0
O-0
Pearson Coeff: -0.104
Intersection: 0.029
Bhattacharyya Dist: 0.860
G-1
O-1
Pearson Coeff: -0.110
Intersection: 0.031
Bhattacharyya Dist: 0.862
G-2
O-2
Pearson Coeff: -0.117
Intersection: 0.032
Bhattacharyya Dist: 0.861
G-3
O-3
Pearson Coeff: -0.121
Intersection: 0.041
Bhattacharyya Dist: 0.820
G-4
O-4
Pearson Coeff: -0.089
Intersection: 0.100
Bhattacharyya Dist: 0.679
G-5
O-5
Pearson Coeff: 0.110
Intersection: 0.253
Bhattacharyya Dist: 0.426
G-6
O-6
Pearson Coeff: 0.302
Intersection: 0.366
Bhattacharyya Dist: 0.289
G-7
O-7
Pearson Coeff: 0.255
Intersection: 0.335
Bhattacharyya Dist: 0.324
G-8
O-8
G-9
O-9
G-10
O-10
G-11
O-11
Pearson Coeff: 0.528
Intersection: 0.498
Bhattacharyya Dist: 0.180
G-12
O-12
12


Correlation Difference SMOTE - Reg
0 0 -0.0075 -0.013 -0.017 -0.018 -0.011 -0.0042 0.0013 0.0024 0.0026 0.0061 0.011 0.014
1 -0.0075 0 -0.0048 -0.0092 -0.011 -0.0072 -0.0015 0.0025 0.0034 0.0038 0.0052 0.01 0.014
2 -0.013 -0.0048 0 -0.0048 0.0083 0.0065 -0.0033 0.00036 0.00065 -6.5e-06 0.00058 0.0056 0.009
3 -0.017 -0.0092 0.0048 0 -0.0035 0.0063 0.0061 0.0042 0.0039 -0.0056 -0.006 -0.002 0.00061
4 -0.018 -0.011 -0.0083 -0.0035 0 -0.0054 0.0065 0.0057 0.0051 -0.0076 0.0088 0.0054 0.0028
5 -0.011 -0.0072 0.0065 0.0063 0.0054 0 -0.0035 0.0054 0.0062 -0.0096 -0.013 -0.01 -0.009
6 -0.0042 0.0015 0.0033 -0.0061 0.0065 0.0035 0 -0.0069 0.0097 -0.015 -0.019 -0.017 -0.017
7 -0.0013 0.0025 0.00036 0.0042 0.0057 0.0054 0.0069 0 -0.0062 -0.012 -0.018 -0.017 -0.019
8 -0.0024 0.0034 0.00065 0.0039 0.0051 0.0062 0.0097 0.0062 0 -0.0048 -0.013 -0.013 -0.016
9 -0.0026 0.0038 -6.5e-06 0.0056 0.0076 0.0096 -0.015 -0.012 -0.0048 0 -0.0075 0.0089 -0.013
10 -0.0061 0.0052 0.00058 -0.006 -0.0088 -0.013 -0.019 -0.018 -0.013 -0.0075 0 -0.0041 -0.012
11 0.011 0.01 0.0056 -0.002 -0.0054 -0.01 -0.017 -0.017 -0.013 -0.0089 0.0041 0 -0.0075
12 0.014 0.014 0.009 0.00061 0.0028 -0.009 -0.017 -0.019 -0.016 -0.013 -0.012 -0.0075 0
0 1 2 3 4 5 6 7 8 9 10 11 12
0.8
0.4
0.0
−0.4
−0.8

Possible Next Actions

- Try more datasets
- Run classification models on the original and generated data to see if models produce similar scores
- Tidy up code for separating the original and generated data – make more robust
- Change imbalance_learn code so it automatically segregates generated and original datasets – alter the SMOTE API from an imbalanced dataset technique to a generic synthetic data approach for easier use in this application