Prediction of Heart Disease and Stroke using Unbalanced Datasets

I analyzed two different data sets, one for heart disease (https://www.kaggle.com/yashnaik12/heart-patients) and one for strokes (https://www.kaggle.com/fedesoriano/stroke-prediction-dataset). For the heart disease data set the target attribute is 'TenYearCHD' which stands for 10-year risk of coronary heart disease which is 1 if the patient has heart disease without ten years or 0 if they did not. For the stroke data set the target attribute is ‘stroke’ which is 1 if the patient had a stroke or 0 if they did not. Both of these data sets are unbalanced. They both have a significantly higher number of data points for the 0 class of the target attribute.

The questions I was interested in answering were:

- How well do different models do at predicting risk for heart disease and stroke from these datasets?
- Since these data sets are very unbalanced, can the results be improved by undersampling, oversampling, or adjusting the decision threshold?

Undersampling and oversampling are techniques that are useful for dealing with unbalanced data sets. Adjusting the decisions threshold can also be used, but is also generally useful if the costs associated with false positives and false negatives are different.

While undersampling, oversampling, and adjusting the decision threshold did improve the fit of models on these data sets, the models are still far from perfect.

Data Cleansing and Preparation

There was some missing data in both sets. The number of values missing from each column in each data set is shown in Tables 1 and 2. Of particular note are the 388 values missing from the glucose column of the heart disease data set and the 201 values missing from the BMI column of the stroke data set. Missing numerical data was filled in with the median for that attribute and missing categorical data was filled in with the most common category for that attribute.

Numerical data was then scaled by subtracting the mean and dividing by the standard deviation, this helps with fitting the neural network. Categorical data with only two choices is encoding as either zero or one, while categorical data with more than two choices is encoding using one-hot encoding.

The stroke data set had a column ‘id’ which was dropped because it’s just an arbitrary id number and would not actually be useful for modeling.

For both data sets I created an undersampled and an oversampled version of the data. The undersampled version was created by randomly selecting a subset from the over-represented class, the 0 class of the target variable, so that is the same size as the under-represented class. The oversampled version was created by randomly sampling with replacement from the under-represented class until it was the same size as the over-represented class.

Data Exploration

I made a pair-plot of all the numerical columns of the heart disease data set, shown in Figure 1. All the pairs of attributes look uncorrelated, except 'sysBP' and 'diaBP'. These are the two different numbers given when measure blood pressure, so it makes sense that they would be correlated. Most of the columns look normally distributed, except 'cigsPerDay' which has a large peak at zero and 'age' and 'heartRate' which have some unusual peaks and valleys in them. Using the normaltest() function however reveals that these columns are mostly likely not normally distributed ($p < .05$ means to reject the null hypothesis that the data comes from a normal distribution). Next I plotted 'age' with the bins set to have a width of one, shown in Figure 2. The unusual valleys disappeared after doing this, I think the first plot had the bins at a non-integer width, so some of the bins had two integers and others had only one which made the plot look unusual. I did the same thing with 'heartRate', shown in Figure 3. The peaks appear to be at multiples of 5. To check this I plotted a histogram of 'heartRate' modulo 10, shown in Figure 4. There are clearly more data points with a remainder of 0 or 5 modulo 10 than other numbers, I'm guessing that some of the 'heartRate' measurements were rounded to the nearest multiple of 5 before being recorded. I tried seeing if rounding the rest of the data would make 'heartRate' normal or improve the fit of the models in the next section but it didn't.

Similarly, I made a pair-plot of all the numerical columns of the stroke data set, shown in Figure 5. Two of the pairs appear to have slight quadratic correlations, moderate values of 'age' tend to have higher values of 'bmi' and moderate values of 'avg_glucose_level' tend to have lower values of 'bmi'. Additionally, 'avg_glucose_level' is bimodal. All three of the numerical columns fail the normaltest() and are unlikely to be normally distributed.

Modeling

I choose three different models to compare for this project: Naive Bayes, Logistic Regression, and Neural Network. These are chosen mostly because they're simple and I've used them a lot. I don't expect the naive Bayes model to work that well since the input data isn't normal and there's a mix of numerical and categorical data, but it's an extremely simple model and takes very little work to fit, so on the off chance it can make good predictions it's usually worth trying.

To select the best model I will being using the F-Score to evaluate them, but it will be a modified version of the F-Score called F_β were β is a parameter which can be adjusted to give more value to either recall or precision. I want to give more weight to recall because false negatives are worse than false positive. I ended up setting β at 1.5 because setting it at 2 made some of the models just predict almost all of the data as having heart disease/stroke.

For the neural network I needed to decide what size of hidden layers and how many of them to use. I trained a number of different neural networks on the heart disease data set and compared them, shown in Table 3. For the heart disease data either one hidden layer with 20 nodes or 2 hidden layers with 15 nodes each seemed to work well, more than that and it appears to start being overfit, it continues to work better on the training data but the same or worse on the test data. I chose to use one hidden layer with 20 nodes, hoping that it will take less time to fit. I repeated the same process with the stroke data, with results shown in Table 4, and decided to use one hidden layer with 25 nodes.

The models were also optimized by adjusting the decision threshold. Normally these models return a probability of each data point to be in each class of the target attribute and they are assigned to the class with the higher probability. For these data sets, each model returns a probability that a given data point will have heart disease or a stroke and if that probability is greater than .5 it assigns it to that class, otherwise it assigns it to the other class. Another way to assign data points would be to pick a different number to be the cutoff. Lowering this value would increase the number of true positives but also increase the number of false positives; the reverse is true if you increase the cutoff. For each model the value of this cutoff was selected that would maximize the value of F_β for the training data.

The results of the models are shown in Tables 5 through 8. Tables 5 and 6 show the results on the heart disease data and Tables 7 and 8 show the results on the stroke data. The first Table in each pair shows the results without adjusting the decision threshold and the second shows the results with.

For the heart disease data set, the models fit without using undersampling, oversampling, or adjusting the decision threshold , the ones labeled as ‘base’ in Table 5, perform the worst with both their precision and F_β-score very low. Similar models fit on the stroke data set, the ones labeled as ‘base’ in Table 7, also performed poorly, though the naïve Bayes did manage to get a recall of 1, but it’s extremely low precision means it’s F_β-score was still pretty low.

For the heart disease data set the best results were from logistics regression with adjusting the decision threshold but without undersampling or over sampling followed closely by logistics regression with undersampling but without adjust the decision threshold.

For the stroke data set the best results were from logistics regression with undersampling but without adjust the decision threshold followed by naïve Bayes with adjusting the decisions threshold but without undersampling or oversampling.

Changing the decision threshold tends to improve the models when not using undersampling or oversampling, but is significantly less useful or can even make the model worse when already using one of these techniques.

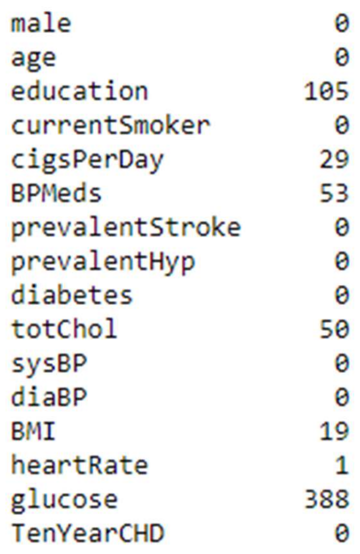
Recommendations:

While undersampling, oversampling, and adjusting the decision threshold did improve the fit of models on these data sets, the models are still far from perfect. I would be hesitant to use them for actually trying to predict heart disease or strokes.

If one were to continue this project they could try looking at other models. I also used the most basic forms of undersampling and oversampling; someone could look into using other more sophisticated forms of these techniques.

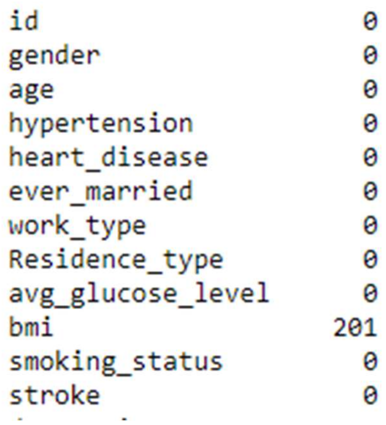
Another way to look at the results might be to conclude that determining whether or not someone will get heart disease or have a stroke is inherently difficult and that we should all consider eating better and exercising more in hopes of reducing that risk.

Tables



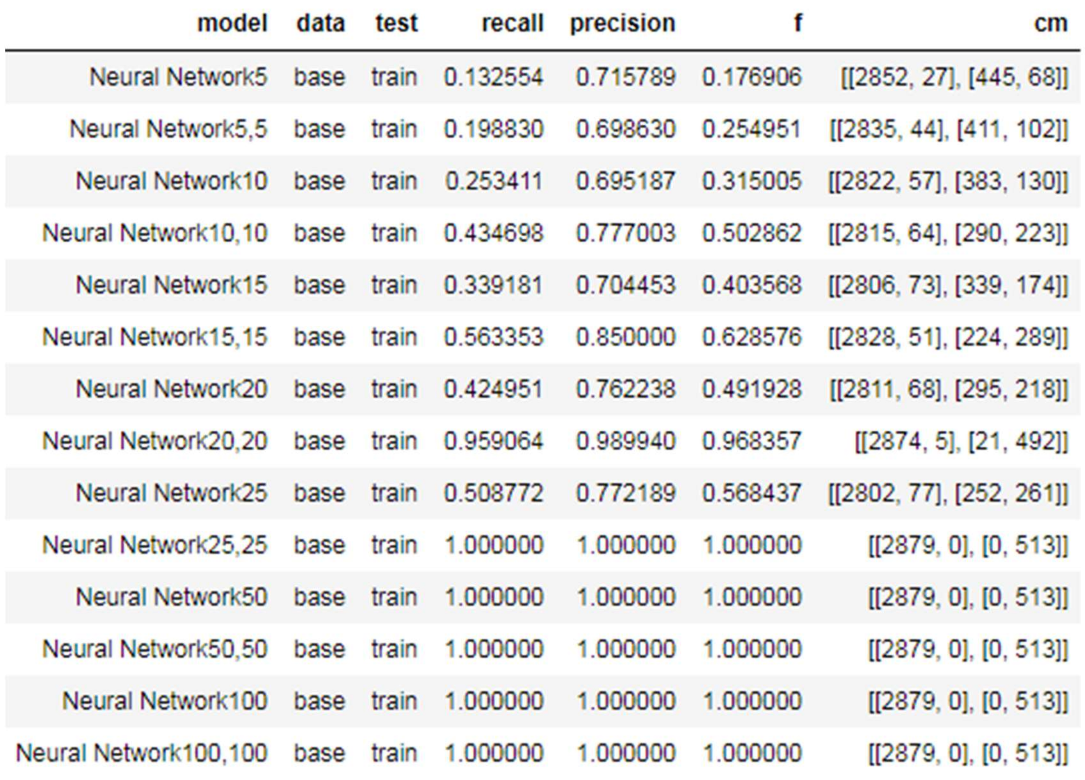
```
male                 0
age                  0
education          105
currentSmoker        0
cigsPerDay          29
BPMeds              53
prevalentStroke      0
prevalentHyp         0
diabetes             0
totChol             50
sysBP                0
diaBP                0
BMI                 19
heartRate            1
glucose            388
TenYearCHD           0
```

Table 1: Number of missing values in each column of heart disease data set



```
id                     0
gender                 0
age                    0
hypertension           0
heart_disease          0
ever_married           0
work_type              0
Residence_type         0
avg_glucose_level      0
bmi                  201
smoking_status         0
stroke                 0
```

Table 2: Number of missing values in each column of stroke data set



model	data	test	recall	precision	f	cm
Neural Network5	base	train	0.132554	0.715789	0.176906	[[2852, 27], [445, 68]]
Neural Network5,5	base	train	0.198830	0.698630	0.254951	[[2835, 44], [411, 102]]
Neural Network10	base	train	0.253411	0.695187	0.315005	[[2822, 57], [383, 130]]
Neural Network10,10	base	train	0.434698	0.777003	0.502862	[[2815, 64], [290, 223]]
Neural Network15	base	train	0.339181	0.704453	0.403568	[[2806, 73], [339, 174]]
Neural Network15,15	base	train	0.563353	0.850000	0.628576	[[2828, 51], [224, 289]]
Neural Network20	base	train	0.424951	0.762238	0.491928	[[2811, 68], [295, 218]]
Neural Network20,20	base	train	0.959064	0.989940	0.968357	[[2874, 5], [21, 492]]
Neural Network25	base	train	0.508772	0.772189	0.568437	[[2802, 77], [252, 261]]
Neural Network25,25	base	train	1.000000	1.000000	1.000000	[[2879, 0], [0, 513]]
Neural Network50	base	train	1.000000	1.000000	1.000000	[[2879, 0], [0, 513]]
Neural Network50,50	base	train	1.000000	1.000000	1.000000	[[2879, 0], [0, 513]]
Neural Network100	base	train	1.000000	1.000000	1.000000	[[2879, 0], [0, 513]]
Neural Network100,100	base	train	1.000000	1.000000	1.000000	[[2879, 0], [0, 513]]

Table 3: Results of training various neural networks on the heart disease data set

model	data	test	recall	precision	f	cm
Neural Network5	base	train	0.062176	0.571429	0.085667	[[3886, 9], [181, 12]]
Neural Network5,5	base	train	0.093264	0.818182	0.128219	[[3891, 4], [175, 18]]
Neural Network10	base	train	0.134715	0.650000	0.178176	[[3881, 14], [167, 26]]
Neural Network10,10	base	train	0.461140	0.809091	0.531465	[[3874, 21], [104, 89]]
Neural Network15	base	train	0.388601	0.742574	0.455395	[[3869, 26], [118, 75]]
Neural Network15,15	base	train	0.984456	0.994764	0.987605	[[3894, 1], [3, 190]]
Neural Network20	base	train	0.393782	0.775510	0.464068	[[3873, 22], [117, 76]]
Neural Network20,20	base	train	1.000000	1.000000	1.000000	[[3895, 0], [0, 193]]
Neural Network25	base	train	0.647668	0.856164	0.700129	[[3874, 21], [68, 125]]
Neural Network25,25	base	train	1.000000	1.000000	1.000000	[[3895, 0], [0, 193]]
Neural Network50	base	train	1.000000	1.000000	1.000000	[[3895, 0], [0, 193]]
Neural Network50,50	base	train	1.000000	1.000000	1.000000	[[3895, 0], [0, 193]]
Neural Network100	base	train	1.000000	1.000000	1.000000	[[3895, 0], [0, 193]]
Neural Network100,100	base	train	1.000000	1.000000	1.000000	[[3895, 0], [0, 193]]

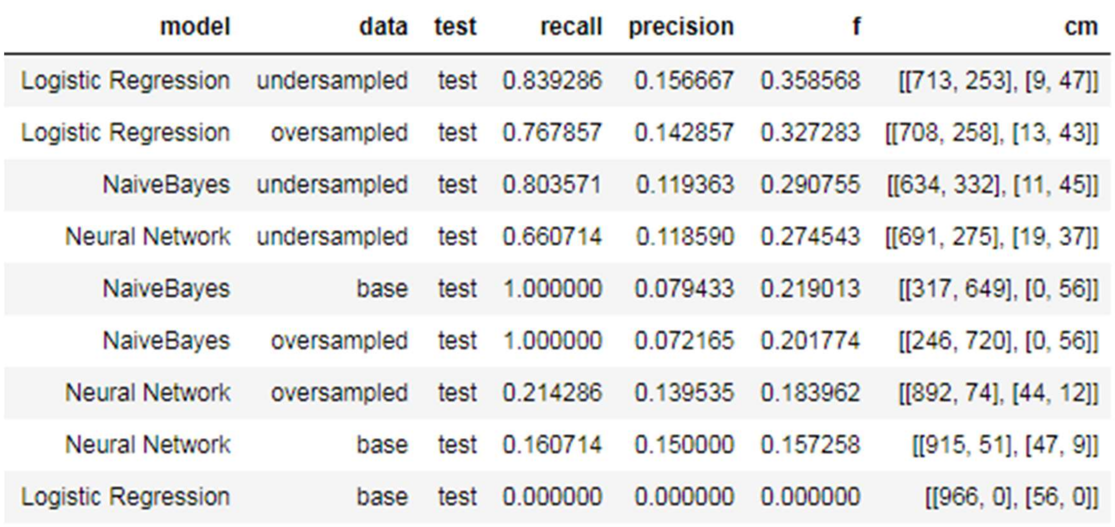
Table 4: Results of training various neural networks on the stroke data set

model	data	test	recall	precision	f	cm
Logistic Regression	undersampled	test	0.687023	0.263930	0.460087	[[466, 251], [41, 90]]
Logistic Regression	oversampled	test	0.671756	0.262687	0.454148	[[470, 247], [43, 88]]
Neural Network	oversampled	test	0.511450	0.270161	0.401198	[[536, 181], [64, 67]]
NaiveBayes	oversampled	test	0.366412	0.358209	0.363848	[[631, 86], [83, 48]]
NaiveBayes	undersampled	test	0.358779	0.350746	0.356268	[[630, 87], [84, 47]]
Neural Network	undersampled	test	0.564885	0.190722	0.352252	[[403, 314], [57, 74]]
Neural Network	base	test	0.229008	0.370370	0.259481	[[666, 51], [101, 30]]
NaiveBayes	base	test	0.213740	0.437500	0.253659	[[681, 36], [103, 28]]
Logistic Regression	base	test	0.106870	0.875000	0.146420	[[715, 2], [117, 14]]

Table 5: Results of models on heart disease data set without adjusting decision threshold, sorted by f_beta

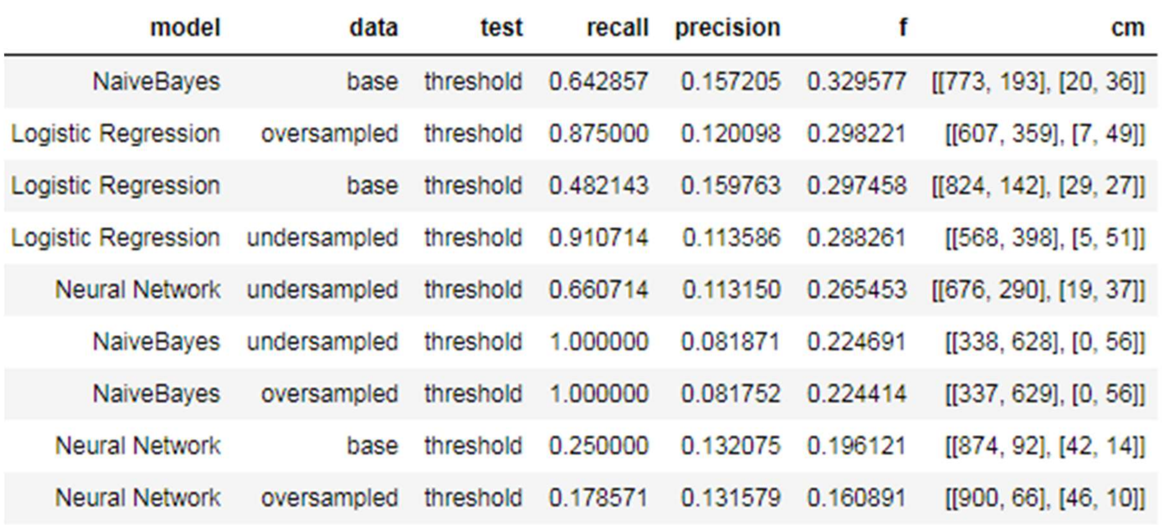
model	data	test	recall	precision	f	cm
Logistic Regression	base	threshold	0.664122	0.281553	0.468323	[[495, 222], [44, 87]]
NaiveBayes	base	threshold	0.748092	0.230588	0.442515	[[390, 327], [33, 98]]
Logistic Regression	oversampled	threshold	0.977099	0.175103	0.405557	[[114, 603], [3, 128]]
Logistic Regression	undersampled	threshold	0.977099	0.171812	0.400096	[[100, 617], [3, 128]]
NaiveBayes	oversampled	threshold	0.954198	0.172176	0.397992	[[116, 601], [6, 125]]
NaiveBayes	undersampled	threshold	1.000000	0.160343	0.382955	[[31, 686], [0, 131]]
Neural Network	oversampled	threshold	0.557252	0.223926	0.382199	[[464, 253], [58, 73]]
Neural Network	undersampled	threshold	0.572519	0.188442	0.351859	[[394, 323], [56, 75]]
Neural Network	base	threshold	0.389313	0.280220	0.347666	[[586, 131], [80, 51]]

Table 6: Results of models on heart disease data set with adjusting decision threshold, sorted by f_beta



model	data	test	recall	precision	f	cm
Logistic Regression	undersampled	test	0.839286	0.156667	0.358568	[[713, 253], [9, 47]]
Logistic Regression	oversampled	test	0.767857	0.142857	0.327283	[[708, 258], [13, 43]]
NaiveBayes	undersampled	test	0.803571	0.119363	0.290755	[[634, 332], [11, 45]]
Neural Network	undersampled	test	0.660714	0.118590	0.274543	[[691, 275], [19, 37]]
NaiveBayes	base	test	1.000000	0.079433	0.219013	[[317, 649], [0, 56]]
NaiveBayes	oversampled	test	1.000000	0.072165	0.201774	[[246, 720], [0, 56]]
Neural Network	oversampled	test	0.214286	0.139535	0.183962	[[892, 74], [44, 12]]
Neural Network	base	test	0.160714	0.150000	0.157258	[[915, 51], [47, 9]]
Logistic Regression	base	test	0.000000	0.000000	0.000000	[[966, 0], [56, 0]]

Table 7: Results of models on stroke disease data set without adjusting decision threshold, sorted by f_beta



model	data	test	recall	precision	f	cm
NaiveBayes	base	threshold	0.642857	0.157205	0.329577	[[773, 193], [20, 36]]
Logistic Regression	oversampled	threshold	0.875000	0.120098	0.298221	[[607, 359], [7, 49]]
Logistic Regression	base	threshold	0.482143	0.159763	0.297458	[[824, 142], [29, 27]]
Logistic Regression	undersampled	threshold	0.910714	0.113586	0.288261	[[568, 398], [5, 51]]
Neural Network	undersampled	threshold	0.660714	0.113150	0.265453	[[676, 290], [19, 37]]
NaiveBayes	undersampled	threshold	1.000000	0.081871	0.224691	[[338, 628], [0, 56]]
NaiveBayes	oversampled	threshold	1.000000	0.081752	0.224414	[[337, 629], [0, 56]]
Neural Network	base	threshold	0.250000	0.132075	0.196121	[[874, 92], [42, 14]]
Neural Network	oversampled	threshold	0.178571	0.131579	0.160891	[[900, 66], [46, 10]]

Table 8: Results of models on stroke data set with adjusting decision threshold, sorted by f_beta

Figures

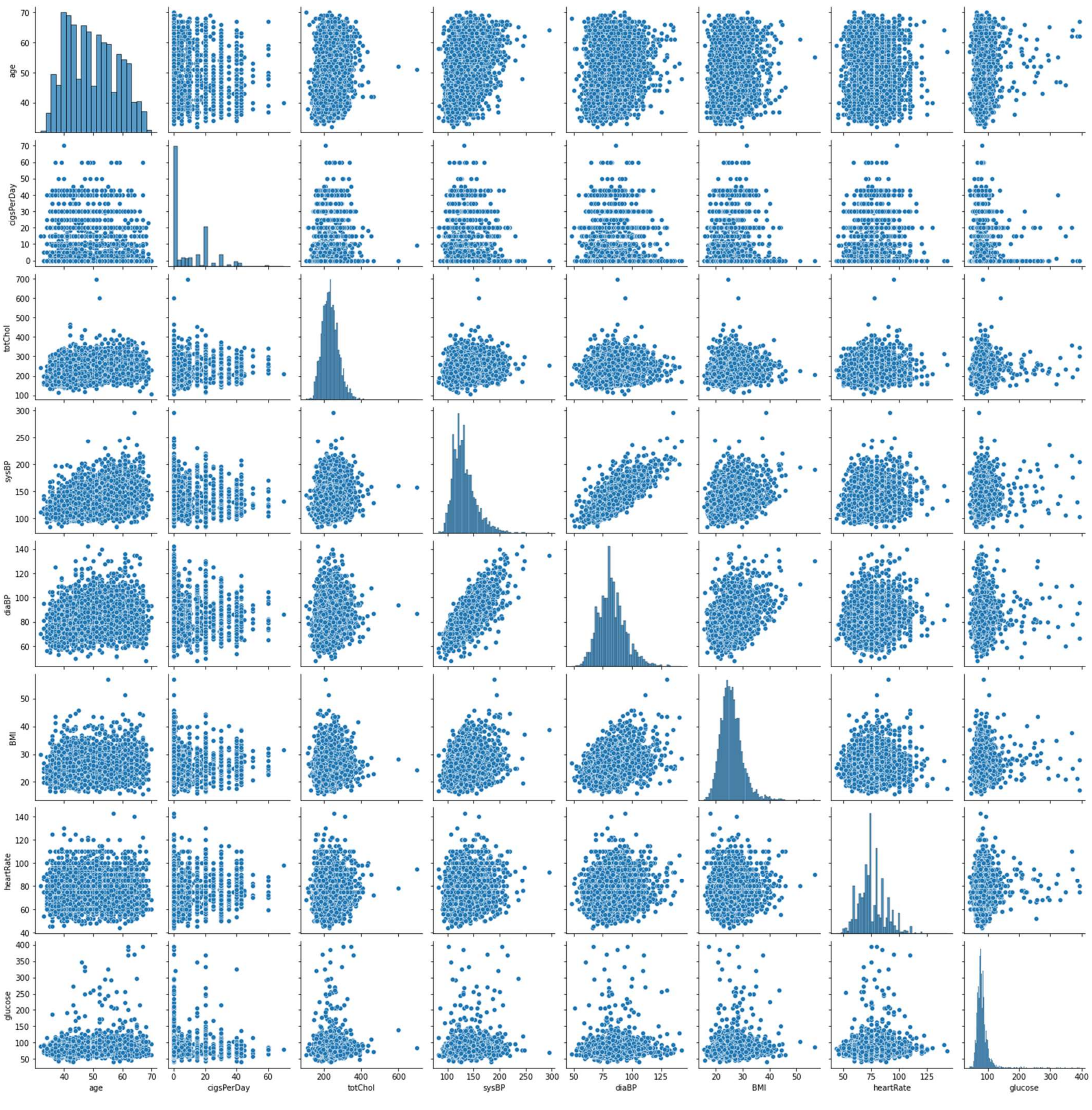


Figure 1: Pair plot of all the numerical data attributes for heart disease data set

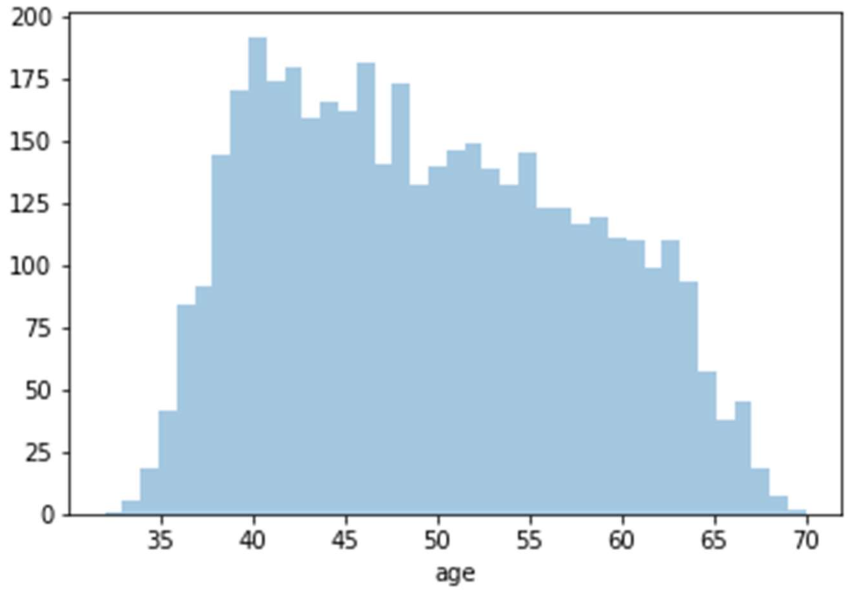


Figure 2: Plot of ‘age’ from heart disease data set with bins set to width 1

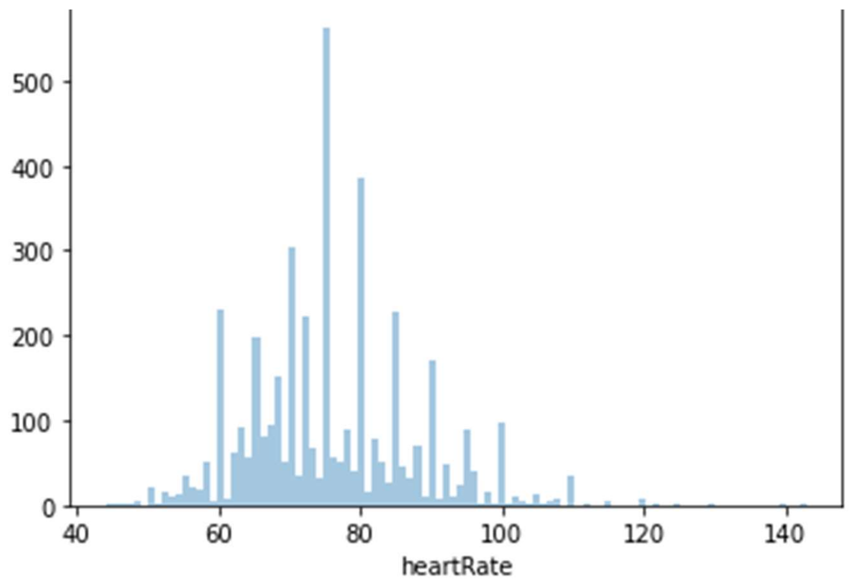


Figure 3: Plot of ‘heartRate’ from heart disease data set, notice the unusual peaks

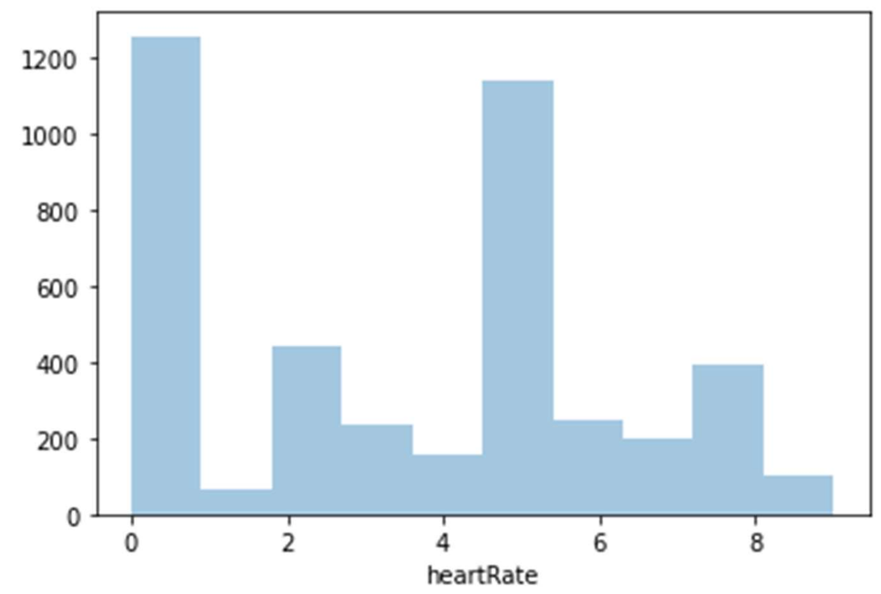


Figure 4: Plot of ‘heartRate’ modulo 10 form heart disease data set, notice 0 and 5 are most common

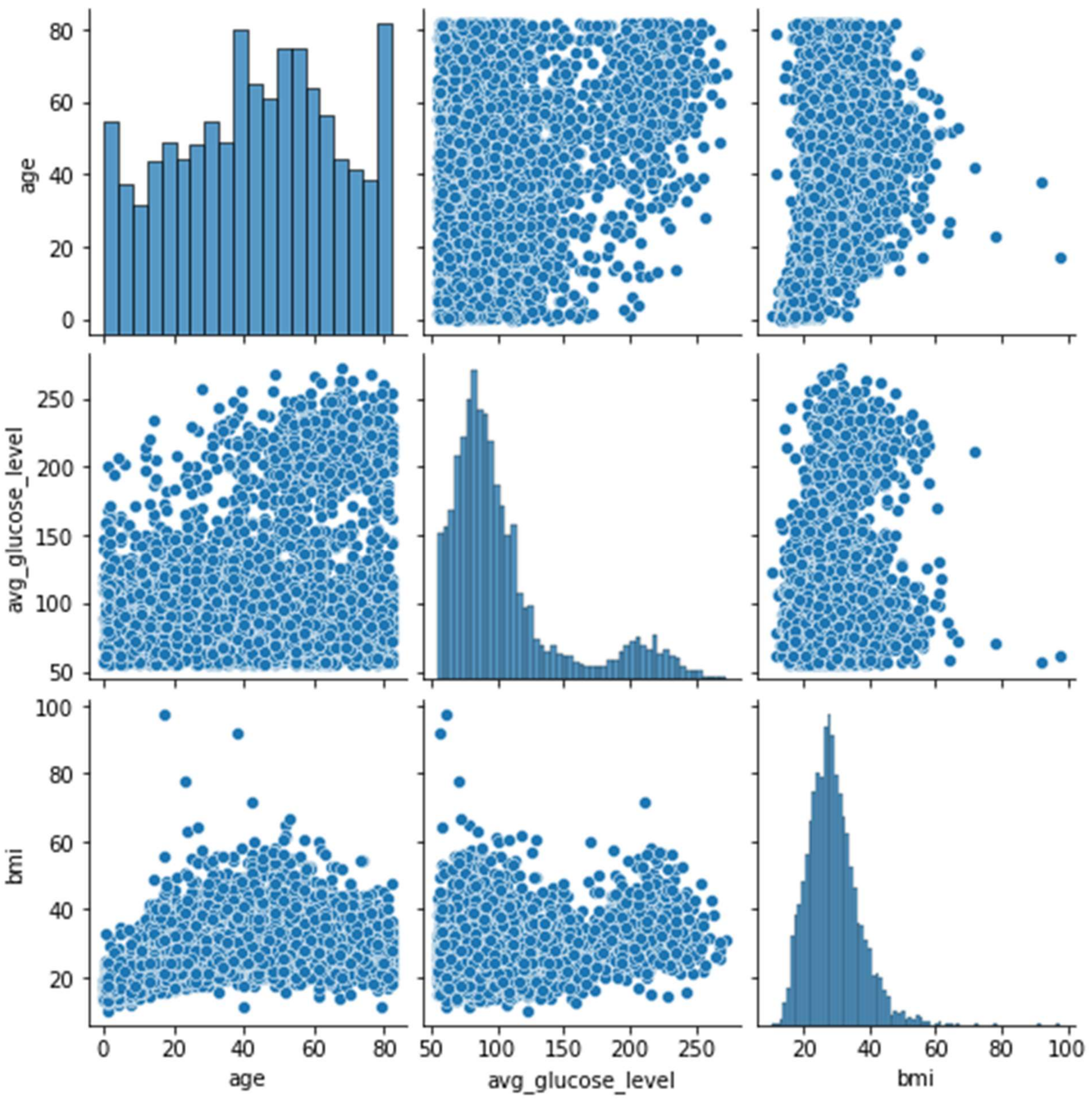


Figure 5: Pair plot of all the numerical data attributes for stroke set