Predictive Models for Heart Disease or Attack

Wenyuan Chen, Yu Huang, Jiadong Chen

1 EDA

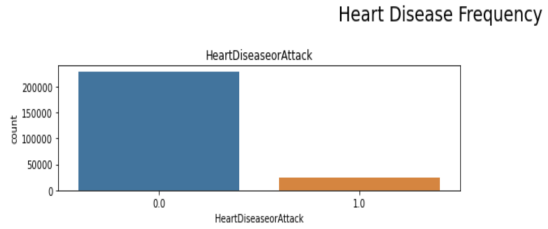
The dataset we used contains health information from the Behavior Risk Factor Surveillance System (BRFSS) in 2015 and was collected and cleaned by user Alex Teboul on Kaggle.This dataset has 253680 observations and each has 22 attributes: {'HeartDiseaseorAttack', 'HighBP', 'HighChol', 'CholCheck', 'BMI', 'Smoker', 'Stroke', 'Diabetes', 'PhysActivity', 'Fruits', 'Veggies', 'HvyAlcoholConsump', 'AnyHealthcare', 'NoDocbcCost', 'GenHlth', 'MentHlth', 'PhysHlth', 'DiffWalk', 'Sex', 'Age', 'Education', 'Income'}, among which 18 attributes are ordinal variables and 4 are discrete variables.

Attribute "HeartDiseaseorAttack" indicates whether the patient had Heart Disease or Attack (1 means yes and 0 otherwise). Our group tends to develop a model to predict whether one would encounter heart disease given his or her health condition and personal information. As heart disease is the leading cause of death in the United States, if one knowing oneself is in high risk of heart disease can always take preventative treatment. Thus, Attribute "HeartDiseaseorAttack" is our project's dependent variable while the rest can be the independent variable.

Single Variable Exploratory Data Analysis

During the basic statistical analysis, we found out that there aren't any missing data points in our dataset and for each attribute the mean value is always less than the median value, indicating that for each attribute, the data distribution is skewed to the left. We also checked the mean, standard deviation, outliers and other basic statistics, but it didn't tell us too much information so we decided to not include statistics here.

```
0.0    229787
1.0     23893
Name: HeartDiseaseorAttack, dtype: int64
```

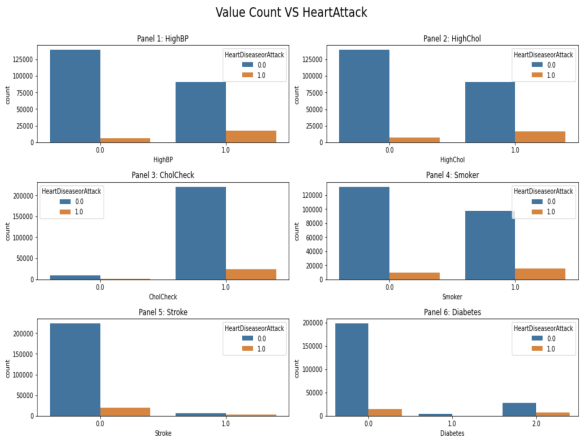


(Figure 1)

Our dataset has 23893 positive data points and 229787 negative ones, which indicates that many people don't have heart disease compared to those who have in our dataset. In other words, our dataset is imbalanced and we need to take this into account when developing models. Also, the data imbalance might explain the data skewness above. Since there are so many attributes that we can use to develop our models, to prevent overfitting, we did bivariable exploratory data analysis to find out the most important features.

Bivariable Exploratory Data Analysis

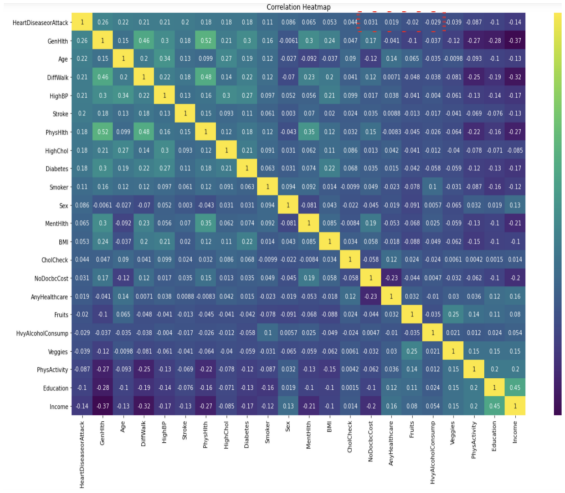
Below are some bivariable exploratory analyses, and we found there might be some risk factors affecting heart disease. For example, in figure 2 panel 1 and 2 below, people who had heart disease are more associated with high blood pressure and high cholesterol. Thus, we see there seems to be some correlations between variables. Thus, we dig in to check for correlations to see what necessary attributes to keep and what unrelated variables to drop for our model development.



(Figure 2)

From the heatmap in figure 3 , we can see that attributes red boxed in figure 3 aren't as important as the other attributes in predicting heart disease. Those attributes almost have no correlation with having a heart attack by having low correlation coefficients. Thus we can consider dropping those attributes when developing methods.

Correlations with Heatmap



(Figure 3)

Given the exploratory analysis, we know that our dataset is imbalanced and there are indeed some attributes that can be indicative of heart disease. We are interested in developing a predictive model to capture those correlations and predict outcomes.

2 Predictive Task

Our predictive task will be a binary classification about whether a person will have heart disease or heart attack based on the features given in the dataset. According to the result from our EDA, we noticed that the dataset is very biased with way more negative than positive results. Based on the correlation map we created during EDA, we found certain attributes that are not closely related to the result. The other attributes are well formatted so that we don't need to do extra preprocessing. Since the dataset is biased, we have to be careful about the prediction of getting a high score by simply giving negative. Also, we will have low tolerance for False Negative since it's prediction for heart conditions. To validate our models, besides the accuracy of our models, another major evaluation we will be looking at is *FNR (False Negative Rate)*.

$$FNR = \frac{FN}{Actual\ Positive} = \frac{FN}{TP + FN}$$

		Test Result		
		Positive	Negative	
True condition	disease	True Positive	False Negative	Sensitivity (% of disease cases identified)
	no disease	False Positive	True Negative	Specificity (% of no disease identified)
	Prevalence (disease/total)	False Positive Rate FP/(Positives)	False Negative Rate FN/(Negatives)	Accuracy (TP+TN)/total

3 Methodology

The task itself is a simple binary classification question, the features are all numeric and binary encoded. The difficulty of this task is that our dataset is biased. Only around 10.4% of patients actually have had heart diseases or heart attacks. As for the models to handle this task, we need a non-biased classifier and we can somehow lower the FNR. Before creating the feature matrix, we drop 4 attributes from the dataset that are not very helpful for prediction based on convention and the correlation map(figure3): 'AnyHealthcare', 'NoDocbcCost', 'Education', 'Income'.

3.1 Logistic Regression

Baseline:
LogisticRegression(fit_intercept=False)

We start by using the basic logistic regression classifier. Since logistic regression is a probit model, it would be easier to deal with an unbalanced training dataset. Although manually balancing the training dataset is also a solution, we think more samples would help train the model better. We randomly split the dataset into 70% training and 30% testing.

Optimize:
LogisticRegression(fit_intercept=True, class_weight='balanced')

Regarding the fact that our training set is biased, we want to change some parameters of the classifier. 'Fit_intercept = True' will add a constant to the decision function and 'class_weight = balanced' can help solve the biased dataset issue.

3.2 Factorization Machine Classifier

We try the model als.FMClassification from fastFM, in order to capture the interactions between features. Here we only include features that have high correlation with 'HeartDiseaseorAttack', namely, 'GenHealth',

'Age', 'DiffWalk', 'HighBP', 'Stroke', 'PhysHlth', 'HighChol', 'Diabetes', 'Smoker' to prevent from overfitting. And we split the data into training data and temp_data by the ratio of 70:30, where temp_data is then splitted into validation set and test set by the ratio of 50:50. Overall the training, validation, and test data are randomly splitted by 70:15:15, using train_test_split from model_selection. We then create sparse matrices, one-hot encode all the features included, and train our model; with the parameters that perform the best on validation set (n_interations = 1000, rank = 5), we apply the model on our test data and reach the result of accuracy score being 0.9093, along with the false negative rate being 0.9218 (Table 1) which is too high to be considered a reasonable model in predicting heart disease. We then dive into the predicting results and find the model tends to give relatively more zeros. Because our dataset is biased, with the number of instances with no 'heart disease or attack' being approximately ten times more than the other, our model will be less confident and thus less likely to predict ones than predicting zeros.

3.3 KNN Classifier

We also build our model using KNN Classification with two different weights. The training, validation, and test data are again randomly splitted by the ratio of 70:15:15. KNN Classifier helps determine the group that a data point belongs to by looking at the points around it. For example, if the majority of the points around a given data point are in the group 'heart disease or heart attack', then it is likely the given data point itself will belong to that group. We initially try uniform weights, and that is - all points in each neighborhood are weighted equally, which gives us 0.8809 accuracy and 0.9709 FNR, the highest FNR among all the models we use. To decrease FNR, we try weights = 'distance', which weights points by the inverse of their distance. Briefly, the closer neighbors of a given data point will have greater influence than further neighbors, and for data points that are less similar, we give them less weight when making decisions. And that model gives us a 0.8791 accuracy score, with 0.9693 FNR (Table 1), which drops only a bit from uniform-weights KNN. As we dig into KNN Classification, we realize that KNN is sensitive to imbalanced datasets, and it becomes biased towards the majority of instances of the training space. Thus in our dataset, which has a positive to negative ratio as 1:10 (both training and testing), conventional KNN does not perform well, as it will tend to predict more zeros than ones.

3.4 Balanced Random Forest Classifier

To better address the bias towards negative samples, we apply a balanced random forest classifier from imbalanced-learn, which randomly under-samples each bootstrap sample to balance the original distribution. For each iteration in a random forest, the algorithm helps draw a bootstrap sample from the minority class, and then randomly draw the same number of instances, with replacement, from the majority class. Each classification tree in the random forest is constructed by samples that have a positive to negative ratio of 50:50. Then aggregate the predictions of the ensemble and make the final prediction. We tune the model through our validation set and find the best n_estimators, i.e. the number of trees in the random forest, 700, and then try our model upon test data, reaching a final accuracy score of 0.7268, with FNR being 0.1719. Thus we successfully drop our FNR by a significant amount.

We then attempt to optimize our Balanced Random Forest Classifier, trying to minimize the false negative rate as much as possible, as we do not want necessary treatment to be delayed or even missed. With the insights

from the research paper *Using Random Forest to Learn Imbalanced Data,* by Professor. Chao Chen from UC Berkeley, we realize that tuning the cutoffs of the BRF classifier might be useful. Conventionally, the default cutoff within BRF is 0.5 and that means, within the random forest, when the proportion of classification trees giving label 1 is more than or equal to 50 percent, we give final decision 1. Here in our scenario, since we want to predict more ones when necessary, we can decrease our cutoff when predicting 1 and raise the cutoff of predicting 0. We then decide to set our cutoffs to be (0.6, 0.4), and because the 'cutoff" parameter is no longer a built-in parameter of Balanced Random Forest, we manually apply the threshold upon brf.predict_prob() which returns the class probability of given test data. Intuitively, we alter the model to give us label 1 whenever 40 percent of trees agree on doing so, and it will predict 0 only when it is very confident (more than or equal to 60 percent) in doing so. With n_estimators=700, and new cutoffs we set, we apply the mode on our test data, and get FNR = 0.1107, the lowest among all.

Note, however, though the false negative rate is dropped with the help of the BRF classifier, the accuracy score has dropped by a significant amount as well. And that is because under-sampling the majority class may result in loss of useful information, as a large part of the majority class would not be used when building up classification trees in our random forest.

4 Literature

The dataset we used was collected from a survey sent out by CDC in 2015 and cleaned by user Alex Teboul on Kaggle. There is no search done on this specific dataset yet. However, a similar research done on predicting type 2 diabetes using the same type of survey by CDC from 2014. As one of the complications of diabetes is heart disease, we think this research is related to our project. The research uses data of 138146 participants and 20467 participants have type 2 diabetes.

Similar to what we did, the research also built several machine learning models including support vector machine, decision tree, logistic regression, random forest, neural network, and Gaussian Naive Bayes classifiers, some of which we also used. Their dataset is also imbalanced with more people without type 2 diabetes compared to those with type 2 diabetes, which is the same situation we faced. The state-of-the-art method they employed to avoid bias resulted from imbalanced data is Synthetic Minority Over-sampling Technique (SMOTE). Given the nature of the problem between the research and our project is the same, our models' prediction behaviors are also similar: there is always a tradeoff between accuracy and false negative rate and we all want to minimize the false negative rate and thus optimize the sensitivity of our model.

Moreover, the research confirms risk factors for type 2 diabetes such as age and BMI. Our model- Balanced Random Forest Classifier- that works the best also confirms some well known risk factors for heart disease such as high blood pressure, high blood cholesterol and diabetes (age and BMI). Since heart disease is one of diabetes' complications, our model is able to confirm diabete as well as age and BMI as one of the risk factors for heart disease , which indicates that our project agrees on the literature's conclusion.

5 Results

Linear Regression:

The predicted result on the test set of the **baseline model** has an accuracy of 0.9046 and FNR 0.8936. This result indicates that we have good overall accuracy but we missed most

positives. As a prediction model for potential heart disease, such a high FNR is not acceptable. The **optimized model** has very different performance since we adjust the parameters to handle the biased training set. The result of the test is 0.7496 accuracy and 0.2098 FNR. This model trades off some accuracy to get significantly lower FNR. Although we predict more false positives with this version, it's acceptable since there is no harm to warn patients of possible heart conditions. According to the Coefficient of features in the decision function, the attribute that has the highest positive impact on the result is 'stroke' and the one that has the most negative impact on the result is 'CholCheck'.

Factorization Machine, KNN:

The prediction result of these two models are: FM(Accuracy: 0.9093, FNR: 0.9218), KNN uniform (Accuracy: 0.8809, FNR: 0.9707), KNN distance (Accuracy: 0.8791, FNR: 0.9693). We are experimenting through different models mentioned in other literatures to see if they are suitable for the task. As expected, since those two models are both sensitive to biased datasets, the FNR are all very high and thus not our ideal choice.

Balanced Random Forest:

This is another model that can automatically under-sample the majority class to balance different classes in the training set. The result of the baseline is 0.7268 accuracy and 0.1719 FNR. To further drop the FNR, we manually select the cutoff to be (0.4, 0.6) and successfully drop the FNR to 0.1107. In this model, the most significant features that impact the result are 'BMI' and 'Age'.

All results are displayed in Table 1 below.

To conclude, our project aims to develop a model with the highest detection rate, indicated by the low FNR because such a model can provide a good initial screening result for any potential patient so that they can get further examination and even treatment from the hospital before it's too late.

Table Summary for Models Classifying Heart Disease

Model used	Accuracy	FNR	Sensitivity (1-FNR)	Specificity (TNR)
LogisticRegression (Baseline)	0.9046	0.8936	0.1064	0.9877
LogisticRegression (Optimized)	0.7496	0.2098	0.7902	0.7454
Factorization Machine Classifier	0.9093	0.9218	0.0782	0.9939
KNN Classifier (weights='uniform')	0.8809	0.9707	0.0293	0.9712
KNN Classifier (weights = 'distance')	0.8791	0.9693	0.0307	0.9705

Balanced Random Forest Classifier	0.7268	0.1719	0.8281	0.7161
Balanced Random Forest Classifier (Optimized)	0.6534	0.1107	0.8893	0.6282

(Table 1)

6 References:

- Xie, Z. (2019, September 19). *Building risk prediction models for type 2 diabetes using Machine Learning Techniques*. Centers for Disease Control and Prevention. Retrieved November 30, 2021, from https://www.cdc.gov/pcd/issues/2019/19_0109.htm.

- Chen, Liaw and Breiman. "*Using Random Forest to Learn Imbalanced Data*". 2004, from https://statistics.berkeley.edu/tech-reports/666.

- Teboul, A. (2021, November 2). *Heart disease health indicators dataset*. Kaggle. Retrieved November 30, 2021, from https://www.kaggle.com/alexteboul/heart-disease-health-indicators-dataset