HeartML: Predicting Co-Incidence of Diabetes and Heart Disease

Santosh Murugan
smurugan@stanford.edu

David Tran
dtran24@stanford.edu

Abstract

Heart disease (1^{st}) and Diabetes (7^{th}) are two of the most common causes of death in the United States [1]. Not only does co-incidence of the two disease often lead to medical complications and poor patient outcomes, but also contributes to drastically increased healthcare spending. Using only cardiovascular physiological parameters, we implement and evaluate several machine learning approaches to predict the whether a patient has diabetes and/or heart disease. We find that our KNN classifier achieves 80 percent accuracy, which exceeds both the baseline and physician oracles. We finally discuss opportunities to extend our work.

1 Introduction

Heart disease and diabetes mellitus (across Type I and Type II) are both in the top 10 leading causes of death in the United States [2]. Although both diseases correlate strongly with age, the prevalence of both diseases (both individually and simultaneously) amongst younger populations remains non-trivial. In many cases, particularly amongst younger populations, the presence of either disease in a given patient is generally only discovered following an adverse medical event (e.g. syncope due to low blood sugar, angina, etc.), which generally requires treatment in an emergency room setting with expensive monitoring devices. Thus, the ability to preemptively predict whether a given patient has heart disease, diabetes, or both, using only a limited set of physiological parameters, has tremendous value.

Current medical knowledge stipulates that a patient has diabetes if their fasting blood glucose level is greater than or equal to 126 mg/dl, and is prediabetic if this same marker is between 100-125 mg/dl. Determining whether a patient has heart disease (e.g. Coronary Artery Disease) relies on a much greater scope of factors and thus requires a more holistic evaluation. According to current medical practices, knowing whether a patient has heart disease is not sufficient to label a patient diabetic, and vice versa. However, with the rise of machine learning, we are able to uncover relationships between diseases which may not be visible or known to human medical practitioners.

In this paper, we investigate to what extent it is possible to predict whether a patient has: 1) both heart disease and diabetes, 2) only heart disease without diabetes, 3) only diabetes without heart disease, or 4) neither heart disease nor diabetes. We first begin with a brief review of related works in this space. Then, using a well-known dataset of physiological (cardiovascular) parameters, we employ several machine learning approaches (including Logistic Regression, K-Nearest Neighbors, Support Vector Machines with and without Stochastic Gradient Descent) to predict the co-incidence of these two diseases. We further discuss several forms of error analysis (including F1, mean and weighted accuracy), and present the results as a series of confusion matrices. We finally conclude with a discussion of the limitations of our work and opportunities to extend the project in the future.

[Project Code can be accessed at https://github.com/santoshmurugan/HeartML]

2 Literature Review

We researched prior approaches which utilized machine learning for predicting heart disease and diabetes individually. In our review, we were unable to find any prior works which predicted both simultaneously. This is a strong indicator of the novelty of our work.

2.1 Cardiovascular Risk

In this section, we highlight three studies which used ML approaches to predict cardiovascular disease risk.

Stephen Weng et al. [3] used machine-learning techniques, such as random forest, logistic regression, gradient boosting machines, and neural networks, to try to accurately predict cardiovascular risk. The study has approximately 380k instances from United Kingdom practitioners. Results show the four aforementioned machine-learning techniques performed better than established prediction methods. Neural networks performed the best out of all the algorithms.

Jaymin Patel et al. [4] set out to build a heart prediction system without any prior knowledge and with the intention of scalability. Three decision tree algorithms are used for comparison – J48 algorithm, logistic model tree algorithm, and random forest decision tree algorithm. Results show algorithm performance ordering, from best to worst, is J48, logistic model tree, then random forest.

Murugesan Mr et al. [5] use decision trees, Naive Bayes, J48 decision tree, K-nearest neighbors, and support vector machines to predict heart disease. The dataset used contains approximately 600 records and 14 features, like age, cholesterol, and thalach. Support vector machines performed the best based on precision, true positive rate, false positive rate, and recall. Results show all the methods achieve > 95 percent true positive rate, prediction, and recall.

2.2 Diabetes Prediction

Quan Zou et al. [6] proposed diabetes prediction using machine learning techniques such as decision tree, random forest, and neural networks. The data includes hospital data from Luzhou, China and diabetic data on Pima Indians. Decision tree, random forests, and neural networks perform similarly, but random forests perform significantly better than the other methods given certain conditions.

Similarly, Goutham Swapna et al. [7] aim to do early detection of diabetes using long short-term memory (LSTM), convolutional neural networks (CNNs), and its combinations to analyze heart rate variability (HRV) data. The data was obtained by collecting electrocardiogram signals from 20 participants for 10 minutes. As a result, 1000 instances were available for every participant. The highest prediction score obtained was 95.7%, the highest score published of any "automated diabetes detection with HRV as input data." However, unlike our study, this paper seeks only to label whether a patient has diabetes (not co-incidence), as HRV is not always sufficient to diagnose many forms of heart disease.

3 Methods

3.1 Dataset

Our dataset (https://www.kaggle.com/ronitf/heart-disease-uci) represents an international collaboration betweeen physicians from the Hungarian Institute of Cardiology, University Hospitals of Zurich and Basel in Switzerland, and the V.A. Medical Center in Long Beach, U.S.A. Abbreviated from a study run by the Cleveland Clinic on heart disease parameters, this is currently one of the most popular and well-known datasets on Kaggle. The data represents 13 physiological parameters and one physician label (i.e. heart disease), for each of 300 patients. Examples of physiological parameters include whether the patient has exercise-induced angina, age, sex, prior history of heart surgery, fasting blood sugar > 120 mg/dl, resting blood pressure, and more. While the original task was to solely predict heart disease from these physiological parameters, we modified the dataset to remove the "fasting blood sugar" column from the training parameters and instead used it as a label for whether the patient is or is not diabetic [or close]. In this way, we ended up with 12 cardiovascular training parameters, and a tuple of labels for each patient. The tuple format is as follows: the first item indicates whether a patient has heart disease (1 if so, 0 if not), and the second item indicates whether a patient has diabetes (1 if so, 0 if not). To more explicitly delineate this format, (1,1) indicates that a patient has both heart disease and diabetes; (1,0) indicates that a patient only has heart disease; (0,1) indicates that a patient only has diabetes; and (0,0) indicates that a patient has neither heart disease nor diabetes. Minor implementation note: because many of the algorithmic implementations required integer labels (as opposed to tuples), we mapped each tuple to an integer ((1,1): 1, (1,0): 2, (0,1): 3, (0,0): 4). In reporting results, we then reversed this mapping and reported in the tuple format. This does not affect the fidelity of the results, only the method of implementation for the algorithms.

3.2 Baselines

To begin, we decided to use a simple rule-based approach (involving no ML) to determine whether a given patient has some combination of the two diseases. After seeing the phenomenally poor performance of this baseline, we then implemented a slightly more sophisticated baseline which employed basic machine learning.

3.2.1 Rule-Based Approach

As mentioned in the introduction to this paper, both diabetes and heart disease generally correlate directly with age, although there are many exceptions. As a simple baseline, we made the following tuple predictions: (0,0) if age ≤ 25; (0,1) if $25 < \text{age} < 50$; (1,1) if age ≥ 50.

3.2.2 Simple ML Approach

Following the the rule-based approach, we decided to extend the set of baseline features to include both age and sex, given that both are generally well-accepted in the medical community to be key influences on cardiovascular and diabetes risk. To do so, we implemented a logistic regression solver using scikit-learn, and evaluated how well the solver could predict label tuples using just age and sex alone. Furthermore, to increase the sophistication of the baseline, we also implemented 7-fold validation, which provided sufficient repetitions to obtain a tighter estimate of model performance, while also satisfying run-time constraints.

3.3 Oracle

For the oracle, we asked a physician certified by the American Board of Internal Medicine (ABIM) to predict the appropriate tuple of labels for heart disease and diabetes, given access to the same 12 cardiovascular parameters. The physician was able to predict heart disease with 100 percent accuracy (especially given that there are certain parameters, such as exercise-induced angina, which are very clear indicators of heart disease). However, because there were no other diabetes markers, the physician advised us that he was unable to predict with greater than 50 percent accuracy (i.e. random chance) whether a patient had diabetes. In this case, the physician suggested crudely predicting that every patient with heart disease also had diabetes, and every patient without heart disease did not have diabetes, yielding an overall accuracy of exactly 50 percent.

3.4 Algorithms

We began by focusing on three key algorithms: Logistic Regression, K-Nearest Neighbors, a "Modified Huber" SVM ("SGDClassifier") optimized with stochastic gradient descent, and a normal Support Vector Machine implementation. The Modified Huber loss function works in such a way that does not excessively penalize outliers relative to non-outliers, which thus provides a different perspective than a loss function such as squared loss. All four of these algorithms are supervised learning algorithms, which is appropriate for this classification task because we have four distinct labels.

All four of the algorithms were implemented via the scikit-learn library for Python, with individual arguments for each classifier selected manually. Number of iterations was set at 10,000,000 because the initial setting of 10,000 often led to issues with model convergence. Even with the higher number of iterations, not only did the SVM implementation not converge, but also failed to yield results within the first 48 hours of runtime, and was thus abandoned in favor of measuring the other three algorithms.

3.5 Train-Test Splits

In the first iteration of implementations, we began by splitting the 300 example dataset into a training set (290) and test set (10). Because the dataset is relatively small, however, we decided to implement K-fold validation to obtain a truer estimate of each models' performance. After iterating over several options for K, we settled on K=7 folds, which allowed a satisfactory tradeoff between runtime constraints and precise performance measurement. We then averaged the

After encountering a problem in which folds would occasionally be lacking one class altogether (e.g. no (1,1)'s in the fold), we designed a solution wherein one example of each class would be held out of the initial 300 person dataset - thus leaving 296 examples which could be used for training/testing. These 4 examples were then manually inserted into each fold's test set so that we could evaluate the performance of the model on at least one example of each class.

3.6 Advanced Work

While we began with using all twelve features in the dataset to predict a 2-dimensional tuple of labels, it occurred to us that our models might be overfitting in training due to the 12-D training set. To investigate whether model performance could be further boosted, we iterated over n=2 to n=12 features to determine which sets of n features yielded the best results. As an example, in the n=2 case, we might end up with ["exang", "age"] being the set of two best features in terms of highest mean accuracy. Whereas in an n=5 case, we might end up with ["exang", "age", "sex", "thal", "cholesterol"], which is the set of five features such that the highest test accuracy is obtained. This process was repeated for each of the four models, with runtime for all four under 10 hours.

4 Results

4.1 Baseline

Between the two baselines, the simple ML approach fared far better than the rule-based approach. The rule-based approach yielded 7.6 percent accuracy. With 7-fold validation, the simple ML approach yielded 55.7 percent accuracy - which is approximately 7x higher.

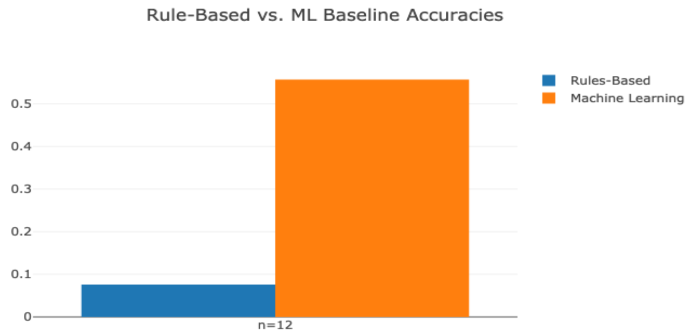


Figure 1: Comparison of Baseline Accuracies

4.2 290/10 Split - Accuracy

With the standard 290/10 (train:test) split, we obtained accuracy results for each of the four algorithmic implementations.

Table 1: Defined Split vs. K-Fold Validation

Algorithm	Defined Split	KFV
LogR	.500	.658
KNN	0.357	0.802
SGD	0.428	0.420
SVM	.429	N/A

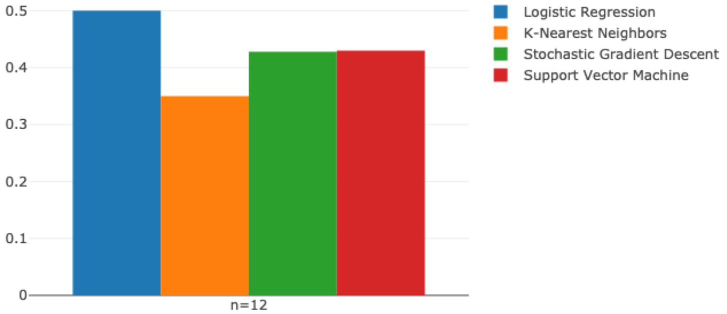


Figure 2: 290/10 Split Accuracy Results

4.3 Improved Results Analysis

Following the statistics with K-Fold Validation, we decided to dive deeper and introduce additional metrics to better understand our models' performance. These additional metrics are well accepted in the ML community. In addition to mean accuracy (across 7-folds), we also have Precision (which is defined to be the ratio of $\frac{\text{True Positives}}{\text{True Positives + False Negatives}}$), F1 Score ($\frac{2*\text{Precision}*\text{Recall}}{\text{Precision + Recall}}$), and a Weighted Average of each class. F1 is a particularly important metric for this dataset because it tolerates class imbalance to some extent. The Weighted Average metric was inspired by the notion that certain classes are more important to predict well than others. For example, (1,1) is more important to predict than (0,0) because the former leads to more healthcare expenditure and worse patient outcomes than the latter, which is a simple false alarm. Thus, as a proof of concept, each class was weighted according to its corresponding annual healthcare expenditure in the United States.

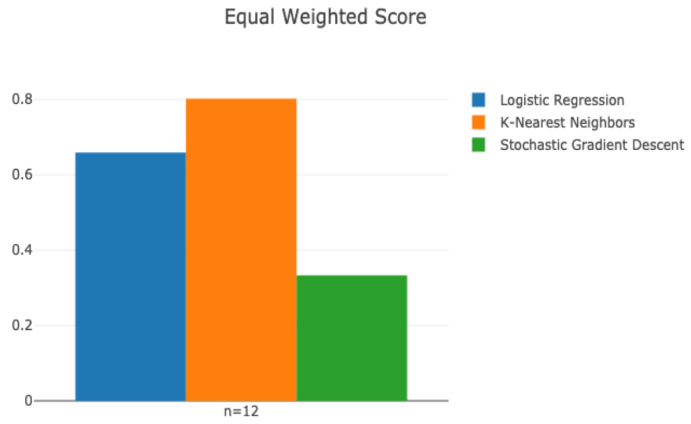


Figure 3: K-Fold Equal Weighted Scores

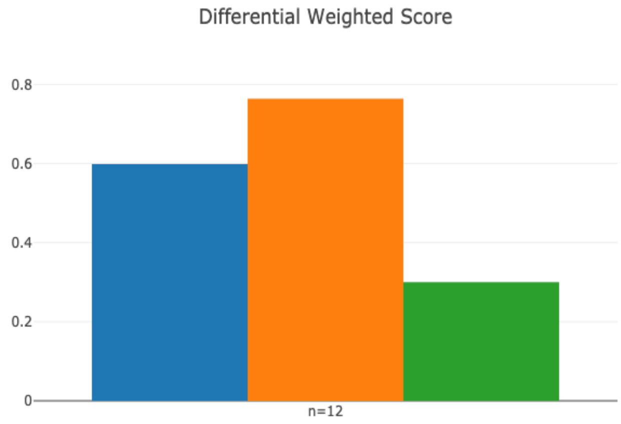


Figure 4: K-Fold Weighted Average Scores

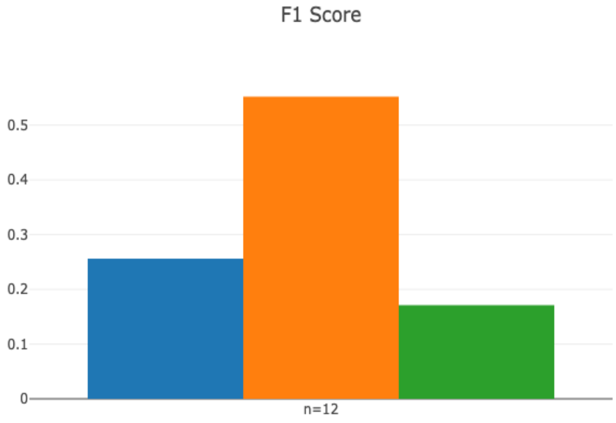


Figure 5: K-Fold F1 Scores

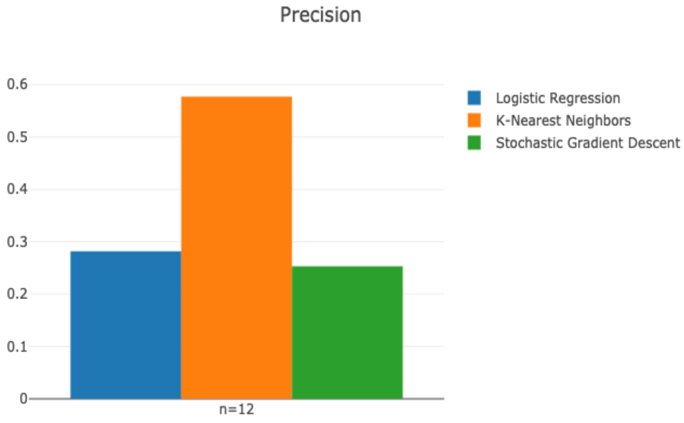


Figure 6: K-Fold Precision Scores

4.4 Advanced Work/Feature Combination Results

For each model, we iterated over all possible subsets of features to identify best performance. The best results were generally obtained for subsets containing 8 features (except for KNN, for which n=7 worked best) Figures 7-10 each highlight a different performance metric for the n=2, n=8 (or 7 for KNN), and n=12 subsets for each model.

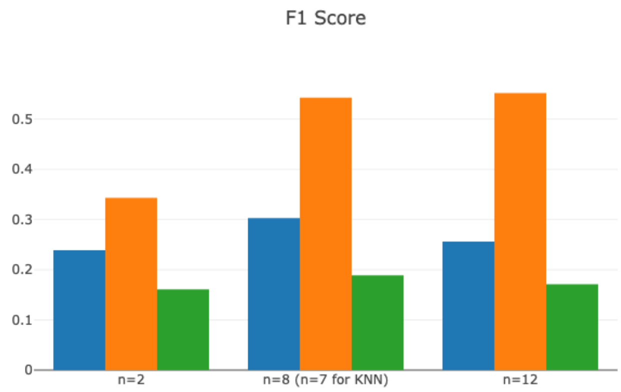


Figure 7: Models' F1 Scores with Varied Size Subsets

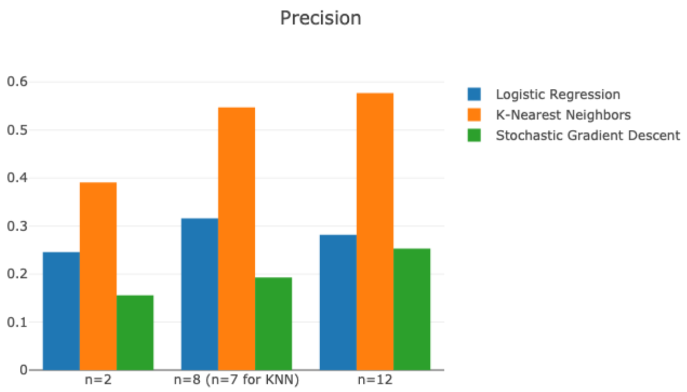


Figure 8: Models' Precision Scores with Varied Size Subsets

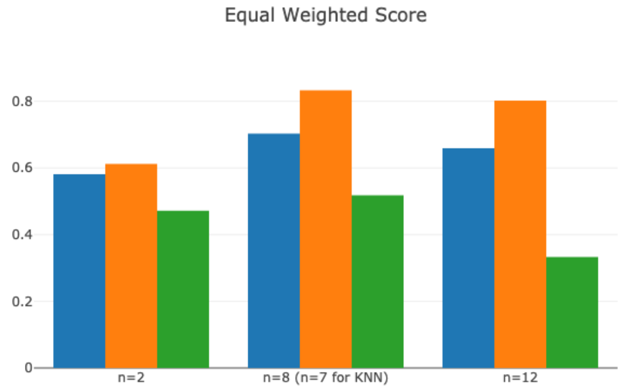


Figure 9: Models' Average Accuracy with Varied Size Subsets

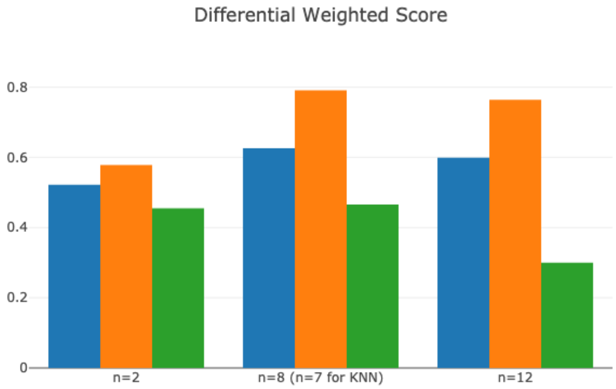
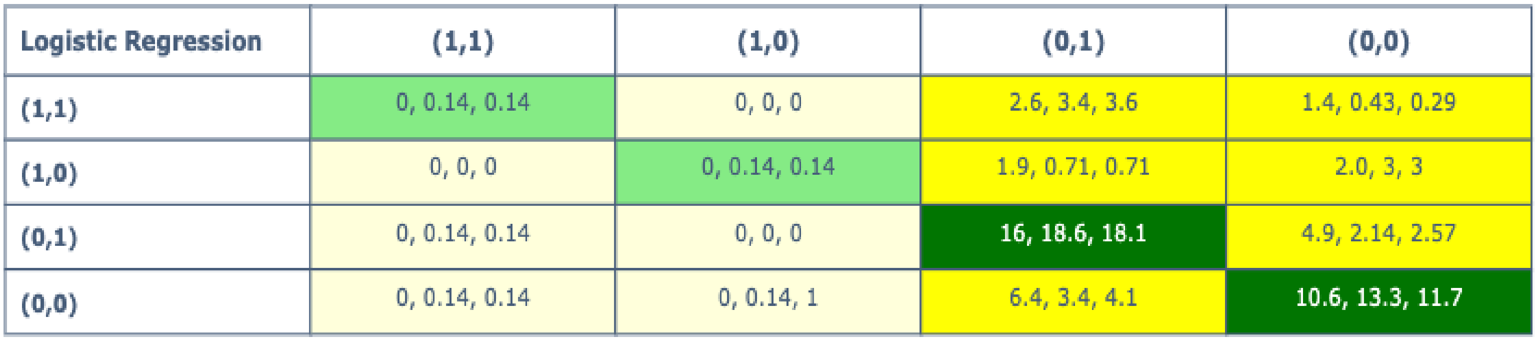


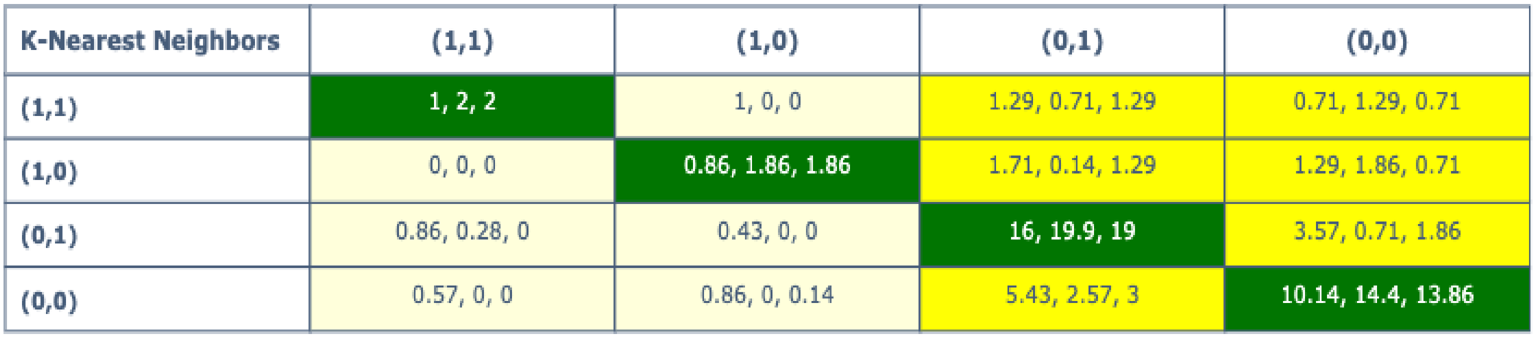
Figure 10: Models' Weighted Accuracy with Varied Size Subsets

We also assembled confusion matrices for each of the three models. Columns represent the actual class; rows represent the predicted class. Each cell entry represents the number of classifications for (n=2, n=8 (7 if KNN), n=12 features). For example, using KNN with only two features, of the [on average] 3.15 instances of (1,0)'s in each fold, the model predicted [on average] 0.86 instances correctly. Increasingly darker shades of green indicates highly accurate classification; yellow indicates instances of substantial misclassification.



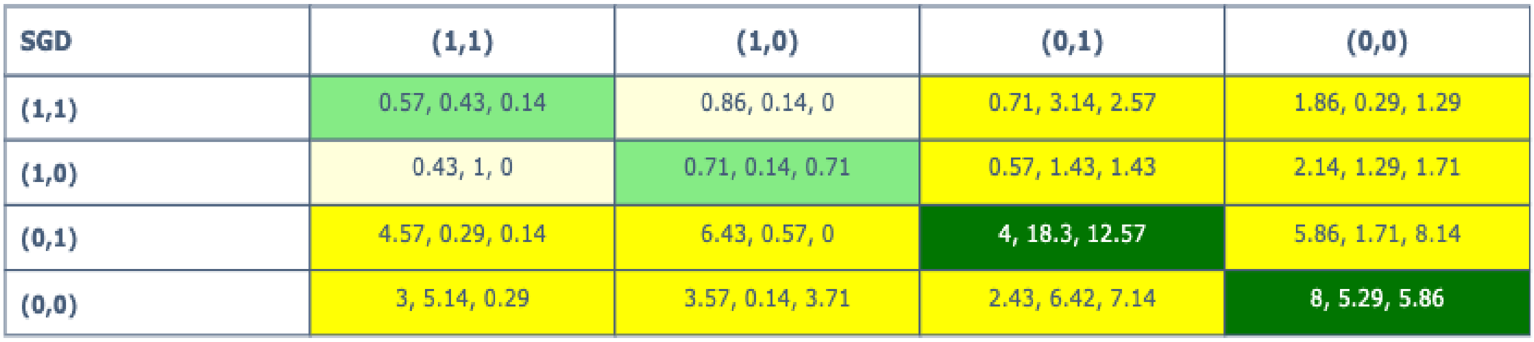
Logistic Regression	(1,1)	(1,0)	(0,1)	(0,0)
(1,1)	0, 0.14, 0.14	0, 0, 0	2.6, 3.4, 3.6	1.4, 0.43, 0.29
(1,0)	0, 0, 0	0, 0.14, 0.14	1.9, 0.71, 0.71	2.0, 3, 3
(0,1)	0, 0.14, 0.14	0, 0, 0	16, 18.6, 18.1	4.9, 2.14, 2.57
(0,0)	0, 0.14, 0.14	0, 0.14, 1	6.4, 3.4, 4.1	10.6, 13.3, 11.7

Figure 11: Confusion Matrix for Logistic Regression (n=2, n=8, n=12)



K-Nearest Neighbors	(1,1)	(1,0)	(0,1)	(0,0)
(1,1)	1, 2, 2	1, 0, 0	1.29, 0.71, 1.29	0.71, 1.29, 0.71
(1,0)	0, 0, 0	0.86, 1.86, 1.86	1.71, 0.14, 1.29	1.29, 1.86, 0.71
(0,1)	0.86, 0.28, 0	0.43, 0, 0	16, 19.9, 19	3.57, 0.71, 1.86
(0,0)	0.57, 0, 0	0.86, 0, 0.14	5.43, 2.57, 3	10.14, 14.4, 13.86

Figure 12: Confusion Matrix for KNN (n=2, n=7, n=12)



SGD	(1,1)	(1,0)	(0,1)	(0,0)
(1,1)	0.57, 0.43, 0.14	0.86, 0.14, 0	0.71, 3.14, 2.57	1.86, 0.29, 1.29
(1,0)	0.43, 1, 0	0.71, 0.14, 0.71	0.57, 1.43, 1.43	2.14, 1.29, 1.71
(0,1)	4.57, 0.29, 0.14	6.43, 0.57, 0	4, 18.3, 12.57	5.86, 1.71, 8.14
(0,0)	3, 5.14, 0.29	3.57, 0.14, 3.71	2.43, 6.42, 7.14	8, 5.29, 5.86

Figure 13: Confusion Matrix for SGDClassifier (n=2, n=8, n=12)

5 Conclusions

5.1 Discussion

In this study, we have implemented four different supervised learning methods on a cardiovascular dataset of 300 patients. We obtained accuracy values for these four algorithms with a simple 290/10 training split, as well as with a 7-fold cross-validation implementation. We further quantify model performance via precision, F1 (including precision and recall), weighted average (according to U.S. healthcare spending ratios), in addition to mean accuracy, on all possible subsets of features.

The best performance was obtained from subsets of 7-8 "best" features, even without an explicit regularization penalty (standard practice is to use a stricter penalty on models which use more features).

With regard to the benchmarks, we exceed the first baseline (rule-based) by > 10x in mean accuracy, and the second baseline (simple ML approach) by almost 25 percent. More importantly, we exceed the physician oracle by close to the same amount, which is extremely remarkable.

It is possible that a cardiac specialist/researcher might be able to shed more light on whether certain factors could lead to a better than 50 percent chance of predicting diabetes given the heart disease diagnosis, but it is unlikely. It is additionally possible that our prediction metrics might not hold over a larger dataset, but again, because we used k-fold validation, this is unlikely.

In a few cases, the models did not fully converge (even with 10 million iterations), but still yielded reasonably high accuracy. Given extra time or compute power, we hypothesize that the models would perform even better than is reported in this paper.

5.2 Future Directions

Given the results we have seen in this paper, we have many potential exciting avenues for future work.

First, we can attempt to implement more advanced architectures such as deep learning approaches. Deep learning approaches have become extremely popular in recent years, and have led to dramatic improvements in a variety of tasks and domains (e.g. Deep Q Learning). Although these approaches might require access to extra compute power, the tradeoff (i.e. higher accuracy) makes this option worth exploring.

Second, we can attempt to augment the data set to improve our performance metrics. In the current dataset, class imbalance is not a dealbreaking problem, but neither is it optimal (especially for metrics other than F1). It would be interesting to synthesize a dataset with better class balance and with more training examples. Moreover, the original dataset (obtained from the Cleveland Clinic before Kaggle pre-processing) included approximately 70 different cardiovascular parameters- as opposed to the subset of 14 which were included in the Kaggle version. Running the same analyses (especially trying all possible subsets of the features) with these new datasets would significantly increase runtime, but could also boost performance and other metrics.

Third, while we chose to limit the task in this project to predicting the co-incidence of only two diseases, it is well-accepted in the medical community that having even more simultaneously occurring diseases can lead to greater risk of complications, which leads to worsened patient outcomes and greater healthcare expenditures. If we could extend this method to predict the co-incidence of more than two diseases, this could greatly increase the value of this tool.

6 Contributions

Santosh and David participated equally. All coding work (including implementations and results analysis) was shared equally among Santosh and David.

References

[1] “Diabetes.” Mayo Clinic, Mayo Foundation for Medical Education and Research, 8 Aug. 2018, www.mayoclinic.org/diseases-conditions/diabetes/diagnosis-treatment/drc-20371451.

[2] “The Top 10 Causes of Death.” World Health Organization, World Health Organization, www.who.int/news-room/fact-sheets/detail/the-top-10-causes-of-death.

[3] Weng SF, Reps J, Kai J, Garibaldi JM, Qureshi N (2017) Can machine-learning improve cardiovascular risk prediction using routine clinical data?. PLOS ONE 12(4): e0174944. https://doi.org/10.1371/journal.pone.0174944

[4] Patel, Jaymin Tejalupadhyay, Samir Patel, Samir. (2016). Heart Disease prediction using Machine learning and Data Mining Technique. 10.090592/IJCSC.2016.018.

[5] Mr, Murugesan Elankeerthana, R. (2018). Support vector machine the most fruitful algorithm for prognosticating heart disorder. International Journal of Engineering Technology. 7. 48. 10.14419/ijet.v7i2.26.12533.

[6] Zou, Quan et al. “Predicting Diabetes Mellitus With Machine Learning Techniques.” Frontiers in genetics vol. 9 515. 6 Nov. 2018, doi:10.3389/fgene.2018.00515

[7] Swapna G., Vinayakumar R., Soman K.P (2018) Diabetes detection using deep learning algorithms. ICT Express 4(4): https://doi.org /10.1016/j.icte.2018.10.005.