Risk Predictor - Cardiovascular Diseases

Rohit Bansal(rbansal3) Alpesh Darji (adarji2) Kathan Al Jewary (ksa2) Paul Nel (paulnel2)

***Abstract*—This electronic document is the final report for a project to develop a risk predictor for the incidence of heart disease using on some basic health metrics based on publicly available heart disease data set. We developed a predictive model based on the combined set of data from all four the UCI Heart Disease datasets. This model is implemented using PySpark and the associated ML Machine Learning libraries to leverage off the advantages presented by these Apache cloud infrastructure tools. Due to challenges with error rates on single regression and classification models on the chosen dataset, we developed a two-stage predictive model that classifies samples as "low', "medium" and "high" risk based on the specified feature set with acceptable levels of accuracy.**

Keywords—machine learning, map reduce, cardiovascular,

I. Background & Motivation

According to WHO, Cardiovascular diseases (CVD) take the lives of 17.9 million people every year, 31% of global deaths. This is the leading cause of human deaths in the world. Often the old cliché of "prevention better than cure" is very much applicable when it comes to CVD. There are multiple factors which trigger these diseases and often there are multiple indicators before the one gets into grip of these diseases. There are no tools publicly available which could track these indicators and create a risk profile for an individual before the individual is in grip of one of these diseases.

Our idea is to indicate the risk level for a user based on the user data. This risk level will indicate the urgency of taking precautionary measures. As part of this project, we will analyze the CVD dataset to find the leading cause of these diseases using ML algorithms. Once our ML model is trained, we will predict the risk level of an individual by analyzing the individuals corresponding dataset. This tool will be available on cloud so that users can use it to find the risk level corresponding to CVD.

II. Data Set

We will use the Heart Disease Data Set available from the UCI Machine Learning Repository [1]. This data set contains four distinct databases from Budapest, Switzerland and Cleveland respectively. A total of 75 attributes are described in these data sets including metrics such as age, gender, smoking history amongst others. We will also try considering any other publicly available datasets and include them if time permits.

III. Proposed Outcome

We propose to build a simple predictor that will provide a user their risk of heart disease based on basic health metrics such as age, gender etc. This predictor will make use of machine learning techniques to provide a user with the risk score within a defined certainty.

Methodology

We will apply machine learning algorithm(s) on the publicly available dataset to train our model. This way our model will be able to establish good understanding of the underlying data. To predict any individual's risk level for a heart disease, we will gather an individual's vitals and perform a predictive analysis using the model we developed. The work is divided in four major areas research, development, validation and reporting. Since all the team members are capable of performing work in all the four specified areas, we will attempt to allocate work in such a way that everyone gets a chance to work on all four domains.

IV. Timelines

Milestone 1: Due Mar 9

Data Collection and Manipulation

By this milestone we should be done with gathering and processing data for selecting and training a machine learning model. We will ensure that we are able to host the data in a cloud environment and that we are able to work with it effectively and efficiently. Concepts learned regarding cloud environment, hosted instances and virtualization, will be extremely useful in achieving this milestone.

Milestone 2: ~~Due Mar 30~~ Due Apr 10

ML Model Development

By this milestone we expect that our ML algorithm will be effective and mature. The data fed to the ML model should provide enough correlation for the model to predict an outcome. We hope to refine our model during this phase and attempt to implement concepts like server hosting, big data are presented during the CCA course.

Milestone 3: Due Apr 27

Real Time Prediction Using ML Model

By this milestone we should be able to analyze test data and predict result using our ML model in a reasonable time. To achieve this, the concepts concerning parallel processing, Hadoop Map-Reduce will be utilized to the fullest. By this time we will have project almost ready. Final piece (Milestone 4) would be to add more documentation, finalize project report, setup instructions and build presentation.

Milestone 4: Due Apr 30

Project Submission along with Final Report & Presentation

Our team will work together to distribute work regarding preparation of project report and presentation. In this phase code is documented and setup instructions are clear enough for anyone to follow.

V. Resources

We propose to develop the tools using Python and for machine learning we propose to use publicly available libraries such as sklearn or similar.

i) Language: pySpark
ii) Cloud Technologies Used: IaaS, PaaS, Map-reduce, Hadoop, Hortonworks
iii) Algorithms & Concept material: Well-Known Machine Learning (ML) Concepts
iv) Code Management System: Github
v) Libraries: Scikit-Learn, scipy, pandas, numpy

VI. Project Progress Report

We made significant progress in our project goals. The following section includes brief updates regarding progress, challenges, and changes made. It also outlines the prototype of our final project outcome.

VII. Changes

There was a change in second milestone date for the project. We found that building and training the model using PySpark MLib takes more time and requires knowledge presented week 12 of course. This delay will not propagate to the third milestone as real-time prediction is almost instantaneous once the trained model is available. As a result, the delay in second milestone does not affect third milestone and overall the project is on track.

VIII. Progress

Milestone 1: Due Mar 9 (completed)

Data Collection and Manipulation

We used the Heart Disease Data Set from the UCI Machine Learning Repository for this project. This data set consists of four distinct data sets (i) Cleveland, (ii) Hungary, (iii) Switzerland and (iv) VA data sets. After some research we found that the Cleveland data set is the most complete. It has been used for some other machine learning research as well.

We have therefore opted to create two different datasets, one for Cleveland only and the second one a combination of all four datasets but using only the most common attributes across the four sets. Both data sets were initially cleaned to remove all samples that contained erroneous data. The sets were then inspected for outliers based on comparison between standardized residuals and leverage, as well as Cook's distance.

A. Cleveland Data Set

For the Cleveland data set, the following attributes were used (refer to UCI reference for a detailed explanation):

1. #3 (age) 2. #4 (sex) 3. #9 (cp)
4. #10 (trestbps) 5. #12 (chol) 6. #16 (fbs)
7. #19 (restecg) 8. #32 (thalach) 9. #38 (exang)
10. #40 (oldpeak) 11. #41 (slope) 12. #44 (ca)
13. #51 (thal) 14. #58 (num) (the predicted attribute)

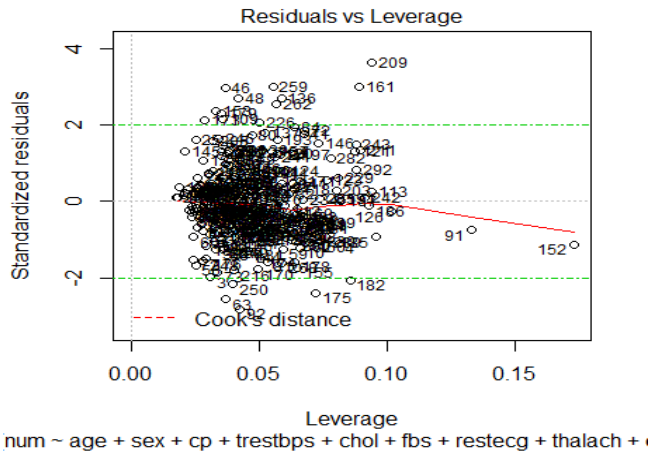

Figure 1Residuals vs Leverage for complete Cleveland data set

Preparation of the data consisted of two parts. Firstly, all rows containing erroneous values were removed. This reduced the data sample count from 303 to 296 which was then used to construct a linear regression model against all attributes using R. Figure 1 above shows the resulting plot with labelled data points. Data point 209 was identified as a possible outlier due to higher than expected deviation from the mean and points 91 and 152 due to relatively high leverage and Cooks' distance. Figure 2 shows the resulting plot after these three data points had been removed.

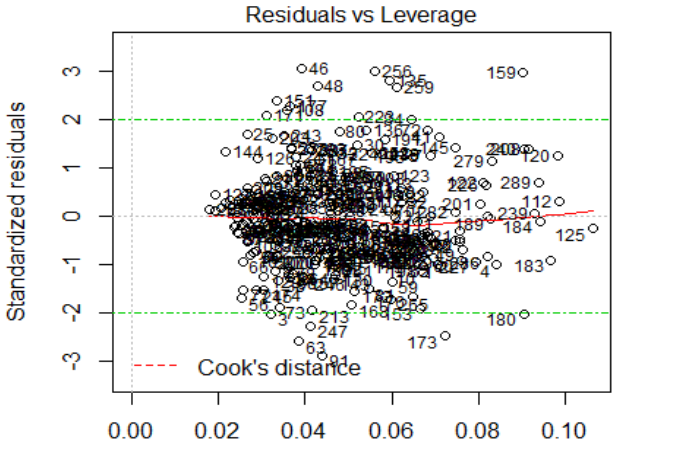

Figure 2 Residuals vs Leverage for reduced Cleveland data set

B. Combined Data Set

Combining data sets proved to be somewhat of a challenge due to the high number of missing attributes in data sets (ii) – (iv) when compared with (i). Evaluating all attributes across all four data sets, the final subset decided upon included:

1. #3 (age) 2. #4 (sex) 3. #9 (cp)
7. #19 (restecg) 8. #32 (thalach) 9. #38 (exang)
14. #58 (num) (the predicted attribute)

Since these attributes will be the ones measured and recorded immediately on admission of a patient to the hospital it therefore suggests that this is a logical grouping of attributes.

In addition to consolidating the attributes across the four different data sets, we also decided to add an additional attribute, *set_id*, that identified the dataset per sample. Since each data set represents a different geographical area, we will test to see if this has a significant outcome in predictions. Combining these data sets resulted in a total sample count of 920. In total, 57 samples were removed that contained at least one erroneous attribute value resulting in an interim data set of 863 samples. We constructed a linear regression model in R to determine any obvious outliers resulting in the plot shown as figure 3.

Figure 3 suggests point 212 as a possible outlier and was removed. The result is presented in Figure 4.

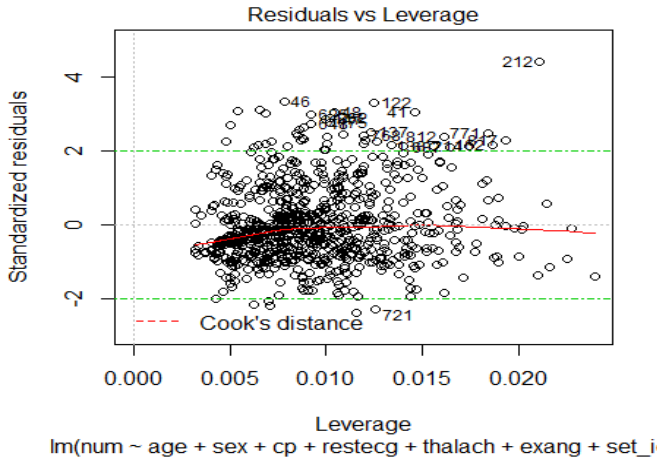

Figure 3 Residuals vs Leverage for complete Combined data set

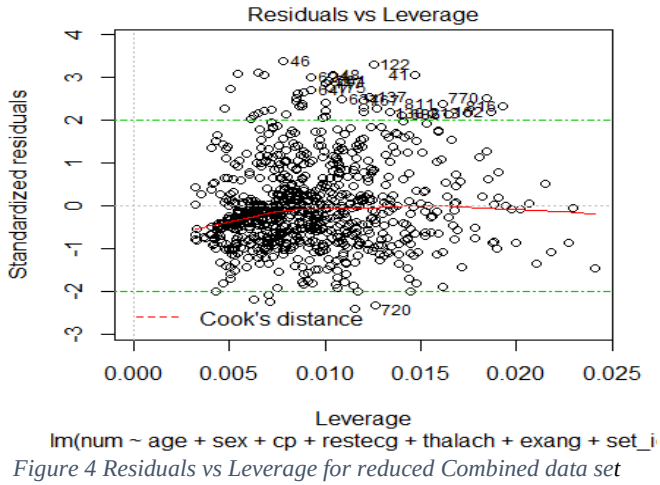

Figure 4 Residuals vs Leverage for reduced Combined data set

Milestone 2: ~~Due Mar 30~~ Due Apr 10 (in Progress)

ML Model Development

After thorough research we found that linear regression will be the best model for our use case. We have implemented the ML model using python-scikit learn to test the accuracy of the predictions and results look promising. We are working further to optimize the prediction accuracy.

IX. Challenges & Solution

In early phase of project bootstrap team faced following challenges:

Data Availability: Finding enough data on which the ML model could be trained for predicative analysis of risk of developing a Heart Disease turned out to be a challenge. We searched on various open source platforms like Google Big Data & Kaggle to explore the data in various categories which could fit our criteria.

Data Quality: Finding reliable data which can be trusted and is backed by authorities or recognized organizations was also a challenge. After some research, we found data from few reliable data sources. We detected data outliers using R and performed some data cleanup to make data usable**.**

ML Algorithm Understanding: we had to develop deep understanding of various Machine Learning Algorithm suitable for the project. To solve this challenge we did a lot of research. Also, we went through ML course content offered by UIUC and found that linear regression technique could be first step to get started.

Remote Team: Our team is scattered all over the globe. We have one person working from Europe while on another has travel job. To meet this challenge, we emphasized on team collaboration by:

- Doing weekly sprints (every Saturday 10AM)
- Utilizing online collaboration tools (like git, zoom, git bot on slack) to the fullest
- Pushing project related communication updates in real time by enabling git bot on our slack channel.

X. Prototype

After having thorough discussions team decided on the following prototype:

Step 1: Preprocess and data cleanup

```
>> python preProcess.py
```

In this batch processing step application would cleanup data and formatted data will be used to train the prediction model.

Step 2: Reading user input and providing prediction result

Console:

```
>> python riskAnalysis.py input.json
```

Step3: predict the risk level based on user input using trained model

Console:

```
Computing and matching your data …
Output of analysis:
riskResult: {
outcome: “MEDIUM”
}
```

The input.json file will have user input in following format

```
userData: {
age: 30,
        sex: “M”,
        cholestrol: 135.65,
        sugar: 153,
        ecg: 23.45
```

```
    ….
}
```

XI. Technologies and Tools

We implemented the project using PySpark and its associated ML machine learning libraries. All datasets where converted to RDD's for processing within the PySpark environment. The specific PySpark ML modules used are listed elsewhere in this paper.

XII. Methodology

Since the Cleveland Dataset is the only dataset that has been used for Machine Learning according to UCI [1], we decided to use the Combined Dataset for our testing since it presents different challenges.

Using PySpark's ML machine learning libraries, we explored various classification and regression models to determine the best fit for predicting the expected risk level of heart disease given typical features similar to those presented in the dataset.

We initially implemented the following models on the cleaned combine dataset using PySpark ML Libraries:

1. Linear Regression
2. Logistic Regression
3. Random Forest
4. Naïve Bayes
5. Linear Support Vector Classifier

However, as will be discussed in more detail in subsequent sections of this paper, we experienced relatively low model accuracies with most of these models. We believe this is largely due the high bias created by a large number of the records having zero-labels.

To rectify this bias, we generated two separate representations of the data. The first dataset contained all original data points with labels of either 1 or 0 indicating presence or absence ("low" risk) of heart disease respectively. We named this dataset as *combined_hd_absence*. Secondly, we extracted the data set *combined_hd_level* which contained all the records indicating a presence of heart disease (i.e. all labels > 0) and re-categorized these labels as follows:

new_label = 1 if *old_label* = 1 ("medium" risk)
new_label = 0 if *old_label* > 1 ("high" risk)

This approach allowed us to do a two-stage classification. Firstly, we used a Linear Support Vector Classifier on the *combined_hd_absence* dataset to train a model to distinguish between the presence/absence of heart disease. Secondly, we used the *combined_hd_level* dataset to train a model to distinguish between a "medium" and "high" risk of heart disease.

When implementing this strategy on new data, we would first test it against the first model and report "low" risk if it predicts 0. If it predicts 1 we would test it against the second model and report either "medium" or "high" risk depending on a 1 or 0 prediction.

In summary, the final methodology can be summarized step-wise as follows:

i) Clean and transform data, identify and remove outliers as described earlier in this paper
ii) Transform the Combined dataset to replace all `nan` values with zeros and convert to *svm* format.
iii) Produce two distinct data sets. (a) A dataset with all data points with all labels > 0 set to 1. This dataset is used to determine the presence of heart disease. (b) A dataset with all records reporting a label of > 0. All labels > 1 are set to 0.
iv) Train a Linear SVC model on dataset (a) and save the model using PySpark library
v) Train a One-vs-Rest with Logistic Regression on dataset (b) and save the model using PySpark library.
vi) To Predict the risk of Heart Disease, load both the model saved earlier.
vii) Predict the risk of heart disease by firstly applying the Linear SVC model. If 0 is predicted, report the sample as "Low Risk". If 1 is predicted, apply the One-vs-Rest model. If 1 is predicted, report the sample as "Medium Risk" otherwise report the sample as "High Risk"

A map of our code implementation of the above procedure is given in Table 1.

Table 1 Map of methodology steps to code modules

Step	Description	Module
(i)	Clean, Transform, Outlier Identification	`outliers.R`
(ii)	Conversion to *svm* format	`ConvertCSVtoSVM.py`
(ii)	Split datasets	`ConvertCSVtoSVM.py`
(iv)	Train Linear SVC Model	`SparkModelFit.py`
(v)	Train One-vs-Rest Model	`SparkModelFit.py`
(vi)	Predict sample risk level	`SparkModelPredict.py`

Other regression and classification models that were tested but not implemented are also included in the final code repository [2].

XIII. Results

Table 2 presents the accuracy achieved on the original multi-label (0, 1, 2, 3, 4) dataset for each of the regression and classification models that we initially tested. These errors were found to be too high to be useful. Based on these findings, it was therefore decided to follow a two-stage model approach by first classing whether heart disease was absent ("low" risk) and if not to classify "medium" or "high" risk levels.

Table 2 Model error for base data

Model	pyspark.ml Module	Error
Linear Regression	LinearRegression	0.8612*
Logistic Regression	LogisticRegression	0.4286
Random Forest	RandomForestClassifier	0.4521
Naïve Bayes	NaiveBayes	0.4681

*Root Mean Squared Error

After testing the various models, we opted for a Linear Support Vector Classifier to do the first pass classification for the presence/absence of heart disease since it showed the best performance. The error on the *combined_hd_absence* (based on an 80/20 train/test split) was 0.1638.

We subsequently trained a One-vs-Rest classifier with Logistic Regression (also using a 80/20 train/test split) on the *combined_hd_level* dataset. Here we achieved an error of 0.2653.

With these two trained models it was now possible to predict the risk of heart disease in a patient as "low", "medium" or "high" based on features present in the combined data set.

We have demonstrated how this could be implemented through a prediction function (*SparkModelPredict.py*) which reads a number of random samples and produces the expected results in terms of risk level predictions. The results for these 9 random samples are presented in Table 3.

Table 3 Sample output of 9 random datapoints

ID	Risk Level
100	low
200	low
300	high
400	medium
500	medium
600	high
700	high
800	medium
900	medium

XIV. Discussion

Our decision to use the combined data set as opposed to the more popular Cleveland dataset implied that we had significantly fewer features per data sample. A large amount of the features in the Cleveland data are not present in a consistent enough manner across data points in any of the other three datasets. This made initial model fitting very difficult and largely too inaccurate to be of much use with normal single pass regression or classification models.

In addition to this, we also found that the bias was significant with a large number of data points reporting labels of 0 and fewer data points reporting 1, 2, 3 and 4.

We managed to address both these challenges through splitting the risk prediction into two stages with an initial prediction only looking at the presence of heart disease. This dealt effectively with the bias and increased the accuracy through changing a multiple classification problem into a binary one.

This concept was transferred to the second stage where the model was again reduced to a binary prediction problem thereby increasing the accuracy of our final result.

While we believe there is still potential for optimizing the model we present here (as discussed in the next section), the results are promising and of acceptable accuracy levels.

We found PySpark is the best technology to implement this. Having a project highly dependent on ML algorithms, we had to use a tool which could support ML algorithm. PySpark libraries were very useful for that. Further, the prediction process is a two-stage process. Thus, we needed a technology which could support multiple ML models working at the same time running independently while providing high scalability. PySpark suited this requirement as well.

XV. Conclusion

The model we developed through this project not only explored the structures and tools available through Apache Spark and specifically the ML machine learning libraries but also allowed us to produce a useful model for predicting the risk level in patients of heart disease given a certain set of features.

While the model is not excessively complex, our implementation demonstrates that it is capable of running on any arbitrary Spark cluster. This enables us to harness the ability of the infrastructure to run the tool anywhere, be that on Hadoop, Mesos, Kubernetes or even as a standalone tool [2]. It therefore implements the essence of the content of the theory presented in the Course Work

We further believe we addressed the initial challenges around accuracy in an innovative way by solving a high bias and low accuracy problem through the implementation of the two-stage predictive model described in this paper.

Future development can focus on the following areas:

i) refining the accuracy of predictions through optimizing the models used or applying different models

ii) refining the prediction in the second stage according to the more detailed levels of 1, 2, 3 and 4 presented in the data.

XVI. Related Work

We found an article by Jonathan Baldie [4] in which the author talks about disease prediction tool. However, there is no implementation provided. Also, our research concluded that there is no implementation that scales. Our implementation will utilize Spark ML pipeline capability which can run on any arbitrary Spark cluster.

XVII. Division of Work

Table 4 presents a breakdown of the division of work between the team members on this project.

Table 4 Division of Work

Team Member	Tasks
Rohit Bansal(rbansal3)	Assistance with finding a dataset and defining a problem/solution Drafting of Progress and Inception reports Development of final regression and classification as well as two-stage model approach implementation of various ML model in PySpark Environment to compare the performance of those models Contribution to all reports not specifically listed
Alpesh Darji (adarji2)	Assistance with finding a dataset and defining a problem/solution Contribution to all reports not specifically listed
Kathan Al Jewary (ksa2)	Assistance with finding a dataset and defining a problem/solution Development of project's PySpark ML environment and implementation of initial Linear Regression Models Development of final regression and classification as well as two-stage model approach Contribution to all reports not specifically listed
Paul Nel (paulnel2)	Assistance with finding a dataset and defining a problem/solution Initial Data Cleaning and outlier detection Testing and review of two-stage model approach and finalization of predict and Naïve Bayes functions. Drafting of final sections of Project Report Contribution to all reports not specifically listed

XVIII. REFERENCES

[1] University of California, Irvine, "UCI Machine Learning Repository," [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Heart+Disease.

[2] R. Bansal, A. Darji, K. Al Jewary and P. Nel, "Project Git Repository," [Online]. Available: https://github.com/rohitbansal83/CS498CCA2019.

[3] Apache, "Apache Spark," [Online]. Available: https://spark.apache.org/.

[4] J. Baldie, "Predicting Heart Disease Diagnoses with Machine Learning," [Online]. Available: https://medium.com/@jonbaldie/predicting-heart-disease-diagnoses-with-machine-learning-2e1a8f5213f8.