Work Summary Report

for

Special Topics (4340) Project: Servo Medela

Version 1.0 approved

Prepared by Nathan S. Robinson
Computing Science Department
Sam Houston State University
nsr004@shsu.edu

Supervisor: Dr. Shashidhar

August 3, 2018

Table of Contents

1. Introduction

1.1 Purpose

This is the work report for the Servo Medela Project to-date. This report is intended to facilitate a summary used for the evaluation of work performed. It will outline the process employed to study and apply basic machine learning concepts to real-world questions.

It will explain the purpose and features of the software, the inputs and output of the system, the interfaces of the software, what the software will do and the constraints under which it must operate.

1.2 Document Conventions

IEEE Report standards were used to create this document.

1.3 Audience

The intended audience is my Special Topics Supervisor: Dr. Shashidhar, a professor at the Computer Science Department of Sam Houston State University. Other readers would include academics or professionals interested in using or expanding this project.

2. Research

2.1 Independent study of basic machine learning (ML) concepts

2.1.1 Read approximately 40 documentation articles

- ML base models
- Data preprocessing
- Feature selection
- Generating ensembles
- I-Python Notebooks
- Kaggle competitions
- and more…

2.2 Completed Stanford Online ML Course by Dr. Andrew Ng

11-week course requiring several lectures, readings, quizzes, practice work, and application project centered on machine learning principles

https://www.coursera.org/learn/machine-learning

2.2.1 Topics Covered (in order):

- Linear Regression with One Variable
- Linear Algebra Review
- Linear Regression with Multiple Variables
- Octave/Matlab Tutorial
- Regularization
- Neural Networks: Representation
- Neural Networks: Learning
- Advice for Applying Machine Learning
- Machine Learning System Design
- Support Vector Machines
- Unsupervised Learning
- Dimensionality Reduction
- Anomaly Detection
- Recommender Systems
- Large Scale Machine Learning
- Application Example: Photo OCR

2.3 Read several academic papers on heart disease and contributing factors

Factors Related to Cardiovascular Disease Risk Reduction:
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2248795/

Framingham Contribution to Cardiovascular Disease:
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4966216/

Prevalence of Uncontrolled Risk Factors for Cardiovascular Disease:
https://www.cdc.gov/nchs/data/databriefs/db103.pdf

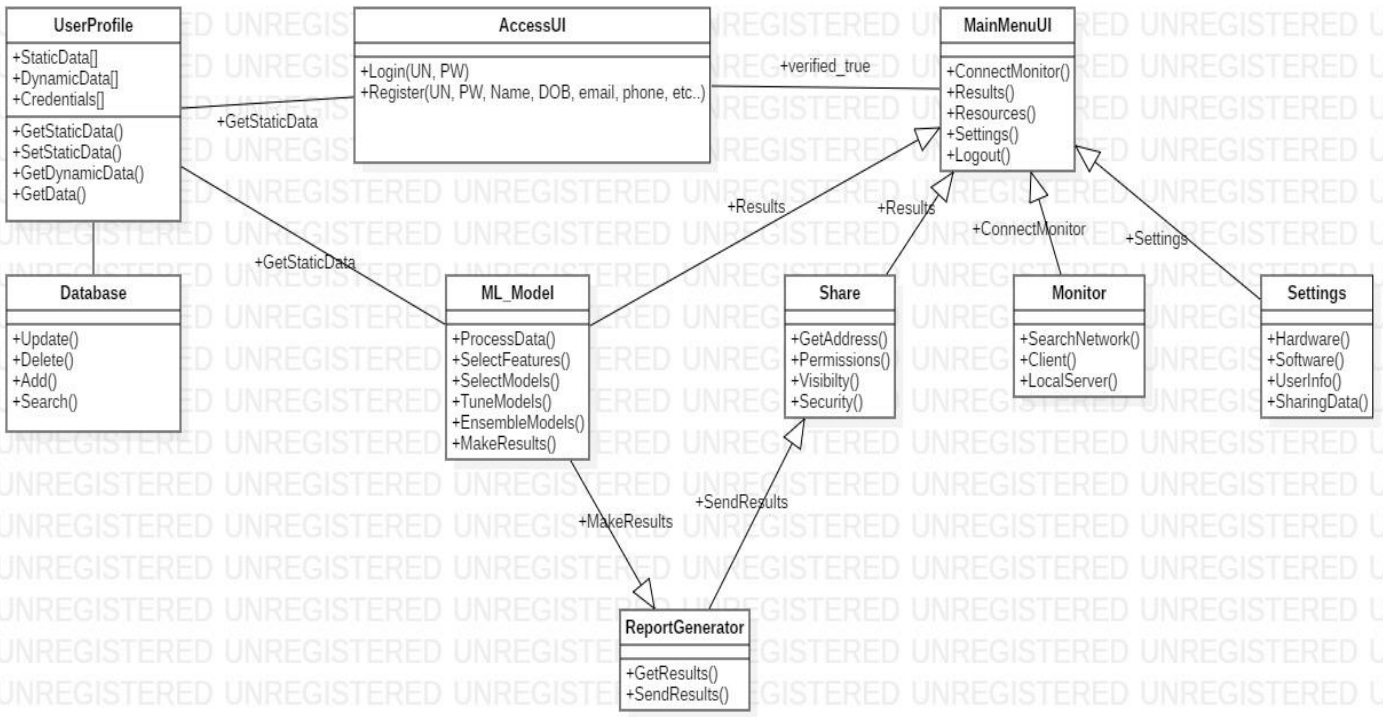
Prevention of Cardiovascular Disease
http://www.who.int/cardiovascular_diseases/guidelines/Full%20text.pdf

Primary prevention and risk factor reduction in coronary heart disease mortality
https://www.bmj.com/content/352/bmj.i721

Cardiovascular Risk Factors
https://www.world-heart-federation.org/resources/risk-factors/?cats=29

3. Specification and Design

3.1 Drafted an object-oriented design for a software system capable of real-time data-collection, analysis, and feedback for high-risk cardiac patients



3.2 Wrote a short design document to accompany the diagram:

Servo Medela is Latin for "Health Watch". This software system is meant to make use of the powerful tools of machine learning to improve the health of at-risk cardiac patients. Servo Medela uses a mobile application to connect users, mainly patients, doctors, and other authorized entities, to these analytical tools. This software system consists of three main components:

- Front-end Application: What the user can is allowed to interact with via the UI platform menus
- Machine Learning Back-end: What receives live data, processes it, automatically trains a ML model, and produces predictions, data visualizations, data stats, and other results
- SQL Database: Where all patient data is hashed, securely stored, and remains available for processing

Object oriented design is the best option for the implementation of this software system. The best way to define this system is by using an object model, via object grouping and relations, a dynamic model, using timing diagrams of live data transfer, and a functional model, using a dynamic data flow diagram. These three perspectives fully encapsulate the design of the software.

The object model chunks the system into relatable, digestible chunks. These chunks are known as objects. Objects provide unique functionality that improve many design metrics. Cohesion is on par with other designs because of the narrow, functional focus of each object.

Coupling occurs but is minimized by limiting the number of subordinates for the control-interface objects. Multiple and single inheritance allows for a well-factored design with a clear hierarchy of control in higher objects down to detail management in lower objects, as well as limited fan-out and fan-in, reducing duplicity in implementation.

Visibility allows the easy customization of the scope of control and effect among objects. The dynamic model supports timing diagrams that describe the how and when of the many scenarios that occur between the front-end UI, the back-end processing, and the database. This clearly demonstrates the flow of control among these three parts. Lastly, we have a working functional model by referencing the functional design diagram. This makes the relationships between many objects, and their operations, less ambiguous.

No other design besides the OOD full captures all the facets of the Servo Medela. The OOD supports excellent design metric results and detail level representation of the system. The implementation of the OOD is also best supported by modern production level languages, available development tools, and the available skills of the development team.

4. Implementation

4.1 Completed a Kaggle Notebook as a foundation for the machine learning backend for this application

4.2 Using a verified dataset of cardiac patients, demonstrated:

- Data Visualization and Analysis
- Data Preprocessing: imputation, standardization, and stratification
- Feature Selection
- Model Selection
- K-Fold Cross Validation
- Parameter Tuning
- Ensemble Generation

5. Testing

5.1 After each module was added to the notebook, a structured walkthrough was preformed to check for performance and structure errors

5.1.1 The notebook allows for each segment to run separately but does not encourage the object-oriented tools the python offers (class, def, etc...), so I exported to a python script to test hierarchical and design metrics

6. Analysis

6.1 Throughout the notebook there is explanation of methods used and analysis of results, as well as a conclusion summary at the end