Faculty of Engineering
Cairo University

Artificial Intelligence in Medicine - Final Project(Phase II)

10-Year Risk of Death of Individuals from the NHANES I Epidemiology Dataset

---

Submitted to:
Dr. Inas Yassin
Eng. Christeen Adly

Submitted by:

Name	Sec	BN
Mohamed Ahmed Abdelaziz	2	14
Mohamed Khaled Galloul	2	15
Mohamed Abdelkareem Seyam	2	18
Ahmed Mohamed Mohamed	1	7

Cairo University
Please let us know if we need to provide any further details.
mohamed.ahmed997@eng-st.cu.edu.eg
mohamed.attia99@eng-st.cu.edu.eg

Contents

List of Figures

1 Preprocessing

Please refer to phase I for data description [click this line to be redirected to the pdf]

1.1 Handling Missing Data

After Examining the data we find that "Systolic BP" feature has more than 50% of missing values. We will describe different approaches to deal with this problem in section 2.2.

1.2 Balancing the Data

We will use up-sampling using 'SMOTENC' which stands for Synthetic Minority Over-Sampling Technique. SMOTE is performing the same basic task as basic re-sampling (creating new data points for the minority class) but instead of simply duplicating observations, it creates new observations along the lines of a randomly chosen point and its nearest neighbors.

- **Before up-sampling:** number of true targets [6954], number of false targets [1625].
- **After up-sampling:** number of true targets [6954], number of false targets [6954].

1.3 Scaling/Normalizing the Data

We can apply the StandardScaler to our dataset directly to standardize the input variables. We will use the default configuration and scale values to subtract the mean to center them on 0.0 and divide by the standard deviation to give the standard deviation of 1.0.

2 Evaluation Metric

2.1 Concordance index (C-index)

The concordance index (c-index): is a metric to evaluate the predictions made by an algorithm. It is defined as the proportion of concordant pairs divided by the total number of possible evaluation pairs.

$$\text{c-index} = \frac{\text{the proportion of concordant pairs } + 0.5 \text{ x number of ties}}{\text{total number of possible evaluation pairs}}$$

- the concordance index is interested on the order of the predictions, not the predictions themselves.
- This is very different from other evaluation measures such as mean squared error or mean absolute error.

c-index characteristics

- A function of how well the ordered values of the continuous predictor correlate to the corresponding event status.
- For an increasing events status any strictly increasing predictions the concordance index will still be equal to 1.
- The concordance index will be zero for every strictly decreasing predictions.
- In case of Ties in your predictions they are counted as half concordant pairs.
- The concordance index supports right censoring which means that the event of interest has only occurred for a subset of the observations (e.g., 'death of a patient', 'customer unsubscribe' or 'any time to event model').
- Random predictions give a mean concordance index of approximately 0.5. So basically a model that hasn't learned anything give a mean concordance index of 0.5.

2.2 Effect of Preprocessing on c-index with different approaches

2.2.1 data with missing rows dropped (>50% of rows)

We see that our train and validation sets have missing values, but our test set has no missing values.

we will begin with a complete case analysis dropping all of the rows with any missing data. Run the following cell to drop these rows from our train and validation sets.

2.2.2 data after dropping the systolic BP column & Balancing with Up-Sampling.

The two methods above show two Decision Tree models one: with features after balancing and dropping Systolic BP and second: with missing rows dropped (>50% of rows dropped).

The models predicts that the missing rows drop approach achieved c-index score (≈ 0.5), but balanced data achieved c-index score (≈ 0.7) and from that we can deduce two things:

- c-index is indeed a very good evaluation metric for 'the time to an event models', as it show the true model performance which in our simple decision tree classifier is random (model learned nothing). c-index is used as the outcome of interest isn't only whether or not an event occurred, but also when that event occurred.
- Tree seems to be overfitting for the first model: it fits the training data so well that it doesn't generalize to other samples such as the validation set.

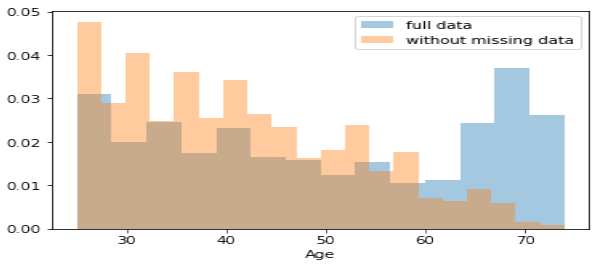
We may now proceed with the balancing approach and dropping the 'Systolic BP' feature, but to make sure we may try mean imputation approach to see how our model would generalize with avoiding complete case analysis.

Later We will also try different classifiers and use Grid Search for Hyper-parameter selection for our best 3 classifiers to improve the c-index score. See Candidate Classification Section

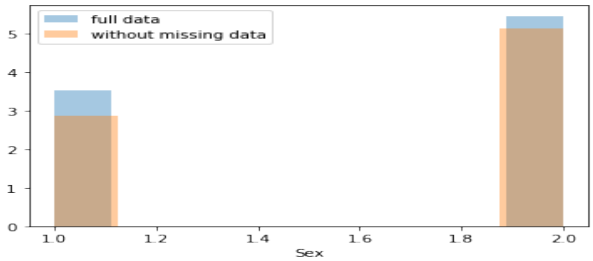
2.2.3 data after imputation (mean imputation)

We will try to improve the test C-Index by filling the missing values and explore if the data is missing at random or not (as we threw away more than half of the data for systolic BP instead of imputing).

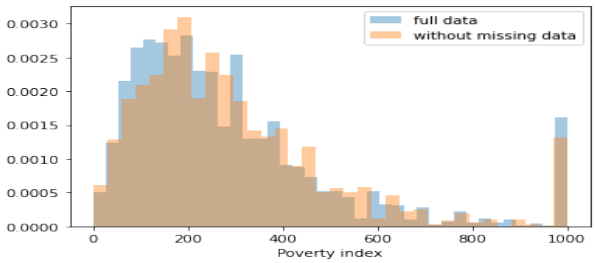
We will plot histograms of the dropped rows against each of the features and compare these to the histograms of the feature in the entire dataset. Try to see if one of the features has a significantly different distribution in the two subsets.



(a) Age Histograms



(b) Sex Histograms



(c) Poverty Index Histograms

Figure 1: histograms of the dropped rows against some of the features compared to the histograms of the features in the entire dataset

Most of the features are distributed similarly whether or not we have discarded rows with missing data. In other words missingness of the data is independent of these features.

If similar distribution was true across all features, then the data would have been said to be missing completely at random (MCAR).

But when considering the age feature, we see that much more data tends to be missing for patients over 65. The reason could be that blood pressure was measured less frequently for old people to avoid placing additional burden on them.

As missing values is related to one or more features, the missing data is said to be missing at random (MAR) and we will replace the missing values with mean substitution approach

- **We should see that avoiding complete case analysis (dropping rows with missing values) allows our model to generalize a bit better.**
- **We Will now proceed with the balanced data and with the systolic BP dropped as it had the best score**

3 Candidate Features

3.1 Permutation Importance

- Feature importance applied after the model has been fitted.
- It selects each feature column one at a time and shuffles/permutate that column randomly. Then it calculates how that affect the prediction/accuracy of the model
- If changing value of a feature column affect the accuracy of the model heavily then the importance of that column feature is higher. In this way, feature importance is calculated.

Weight	Feature
0.1373 ± 0.0169	Age
0.0775 ± 0.0112	Sex
0.0688 ± 0.0114	Serum Albumin
0.0492 ± 0.0078	Poverty index
0.0389 ± 0.0045	Sedimentation rate
0.0334 ± 0.0046	Red blood cells
0.0316 ± 0.0075	Pulse pressure
0.0304 ± 0.0083	TS
0.0251 ± 0.0026	Serum Iron
0.0236 ± 0.0055	White blood cells
0.0208 ± 0.0034	TIBC
0.0205 ± 0.0075	Serum Cholesterol
0.0193 ± 0.0032	Serum Protein
0.0185 ± 0.0048	Diastolic BP
0.0168 ± 0.0022	BMI
0.0160 ± 0.0052	Serum Magnesium
0.0003 ± 0.0010	Race

Figure 2: Features importance ranked by permutation importance

Now we know a lot of inside in our model by using feature importance. We can use this information when collecting dataset and pay more attention to the important features and may discard the features that lower the model performance.

Note:even after removing 5 features (least permutation score) the testing c-index doesn't change as n

3.2 PCA (Dimentionality reduction)

After performing the PCA we can notice that even when choosing high number of principal components the model performance wasn't better than removing the least relevant features from the permutation

Now we will fed our models with the data which was calculated using:

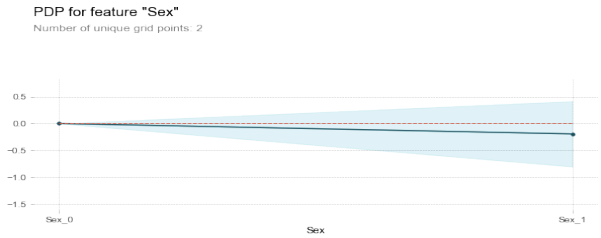
- Data after dropping the systolic BP ($> 50\%$ missing) & Balancing with Up-Sampling.
- Feature Selection using permutation importance

4 How a single feature affect our prediction (PDP)

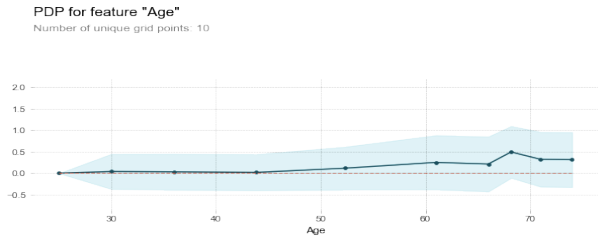
we previously had the most important features of our model. It's a good insight, but we don't know how each feature affecting accuracy (Is sex increasing or decreasing the risk of death?).

Partial Dependance Plot (PDP): shows how each feature will affect the prediction of the model increase or decrease

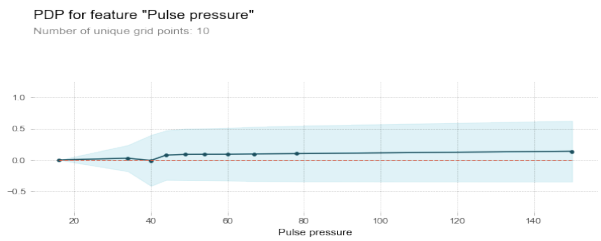
- PDP calculated after a model is fitted.
- Use single row for prediction. Repeatedly alter a variable value to make a series prediction.
- Do that for multiple rows and plot the average prediction on the vertical axes



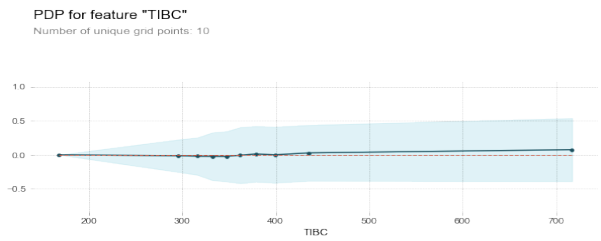
(a) Sex



(b) Age



(c) Pulse Pressure



(d) TIBC

Figure 3: Partial Dependence Plots:The y axis is interpreted as change in the prediction from what it would be predicted at the baseline or leftmost value. A blue shaded area indicates level of confidence (standard deviation).

- **PDP(Sex)** The probability of death risk is decreased in females [poor men! :( ].
- **PDP(Age)** The probability of death risk is decreased if the age is over 60
- **PDP(Pulse pressure)** In the third plot, we see that death risk probability increases after pulse pressure hits 80.
- **PDP(TIBC)** Lastly, we see that death risk probability increases,but not significant, after TIBC hits 410.

5 Why my model predicted that (SHAP Values)

In the previous section, we already have seen how our model works, which features are important and how individuals feature affect the prediction of our model. But now we want to know why an individual prediction is made. Why our model made such a prediction?

SHAP Values (an acronym from SHapley Additive exPlanations) break down a prediction to show the impact of each feature. It explains why a model made a certain prediction. Previously, we see the explanation of the model after training and how each feature will affect the prediction of the model. Now we are going to look at the individual prediction that our model made. Meaning we will now look at our model after it has made a prediction.

How much a prediction is driven by the factor of sex or pulse presure instead of some baseline value of them? And we do these for each feature. The rules for calculating SHAP values is:

$$\text{sum(SHAP values for all features)} = \text{pred for patient - pred for baseline values}$$

- SHAP: how each feature affected the prediction (after prediction).
- It calculates how the value of a feature affected the predicted outcome instead of some baseline value.
- It does it for every feature and shows the contribution of each feature for the outcome.
- SHAP interpret how feature increase or decrease prediction with reference to the PDP and show how consistent your results

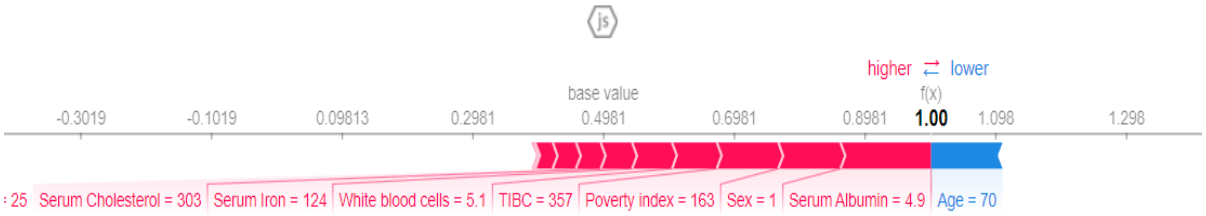


Figure 4: SHAP for a true prediction

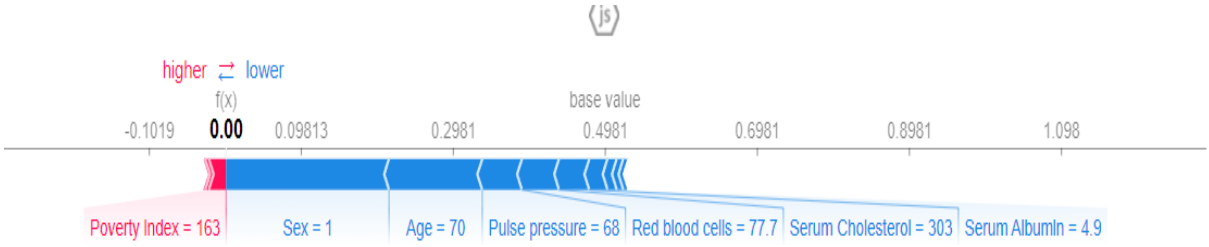


Figure 5: SHAP for a true prediction

From the two shap plots ubove we can see that the most important features we get from permutation importance are the most relevant in the prediction and with shap we can see each feature contribution either it increase or decrease the prediction.

6 Candidate Classification Techniques

We Started fitting numerous models [LogisticRegression, KNN, SVM, DecisionTree, RandomForest, XGBoost, NaiveBayes, HistGradientBoosting] to **choose the best 3 base classifiers** to use in our **ensemble voting technique** after performing **hyper-parameter grid search** on them.

The Best 3 models based on c-index testing score was

- RandomForest: 0.865
- XGBoost: 0.879
- HistGradientBoost: 0.865

6.1 Grid Search and voting classifier

This function helps to loop through predefined hyper-parameters and fit your estimator (model) on your training set.

The grid search of each classifier uses the following hyper-parameters:

- RandomForest: [n-estimators, max-depth, max-features] $\longrightarrow$ 0.863
- XGBoost: [n-estimators, eta, gamma, max-depth, subsample, colsample-bytree] $\longrightarrow$ 0.887
- HistGradientBoost: [max-iter, learning-rate, max-depth, max-leaf-nodes] $\longrightarrow$ 0.869

Voting Classifier $\longrightarrow$ 0.883