STT 811 Semester Project

Examining ICU Healthcare Data Final Report

Angelica Gacis, Anna Jeffries, Chaeyeon Yim

March 30, 2024

Contents

1 Overview

We are examining data collected from hospital ICU patients in order to consider the features that indicate the likelihood of mortality (or lack thereof). We wish to explore the variation of bias as told by different models on differing instances of the same statistical information; namely, the role that a patient's gender or ethnicity might play. The scope of our project is limited, but we wish to situate ourselves between three overarching areas of concern.

1. How much biased information should a model "see" in order to produce best[1] results? And how can that threshold be determined?

2. Based on the pre-existing standards of APACHE II or APACHE III that are used by hospitals today, how could they be improved in order to identify the most critical variables that determine a patient's outcome within the first 24 hours of care in an ICU unit? What demographic information is actually important? Or, from a more interesting aspect, could there fathom-ably be a "bias variable" factored into a given calculation that is informed by certain demographic indicators?

3. More broadly, how can mathematical results, whether via APACHE or statistical models, have the most impact on how medical professionals actually behave and interact with their patients? "Getting in the door" is, perhaps, the largest challenge for unbiased medical care because admissions personnel can (and do) turn away potential patients due to their own biases. Is it possible to shape these highly subjective and volatile environments by way of unbiased models?[2]

All of our work is done in Python.

1.1 The Data

The data. has 91,713 observations with 186 features collected from the first 24 hours of a patient's stay in a hospital ICU. There is no time feature, so we are effectively working in a temporal vacuum. The target variable (`1`, death, or `0`, no death) is highly unbalanced, with one class making up less than 9% of the total. There are also a number of missing values.

It should be noted that our data source provides two different datasets. One is the "training" dataset, which contains our labelled target "hospital_death." The second, however, is unlabelled, e.g., with no target outcome. This means that it is impossible to assess model

[1] "Best" by balancing accuracy and efforts to combat any inherent data bias.

[2] i.e., even if an admissions person would normally turn away a certain patient due to underlying biases, could that incorrect decision be over-ridden/bypassed by a model or something similar that tells the admissions person the patient in question must be admitted?

metrics as there is no "ground truth" against which to compare our predictions. Due to the limited scope of this project, we will not be addressing this unlabelled dataset.

1.2 Our Question

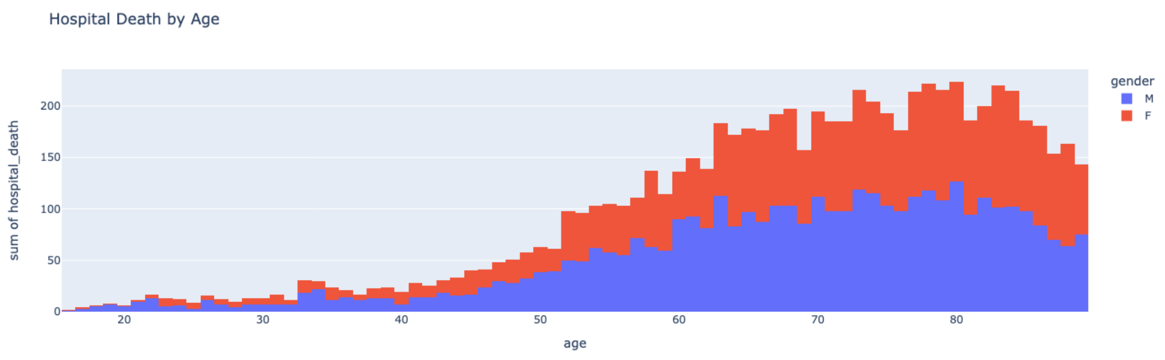
We are seeking to identify and understand the extent of correlation and causation (presuming their existence) within the data and their relationship with the ultimate patient outcome (e.g., death or not-death). Our question is motivated by the discrepancies in professional medical care due to bias and other discriminatory attitudes towards patients.[3]

2 Data Pre-processing

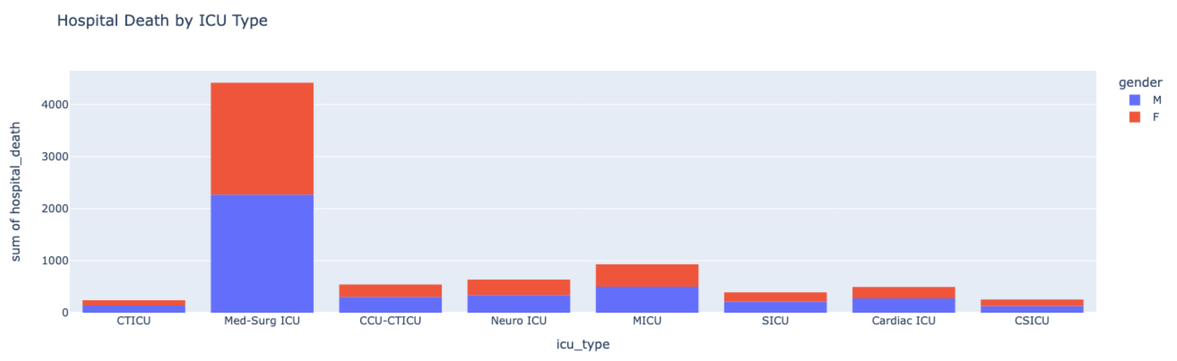
2.1 Exploratory Data Analysis

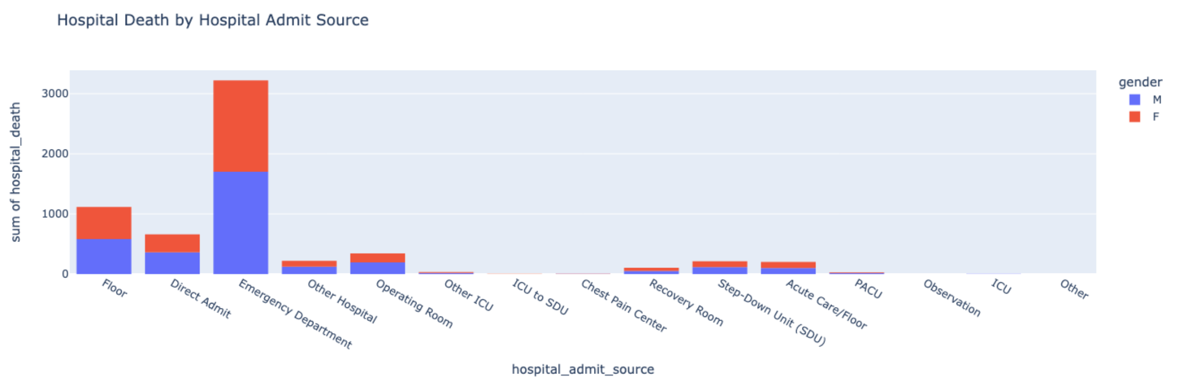
After loading the data into `pandas`, we first looked at its features, their data types, and a minute sample of the dataframe itself. Of the 196 columns, 170 contain floats, eight contain integers, and eight contain objects.

2.2 Visualisations

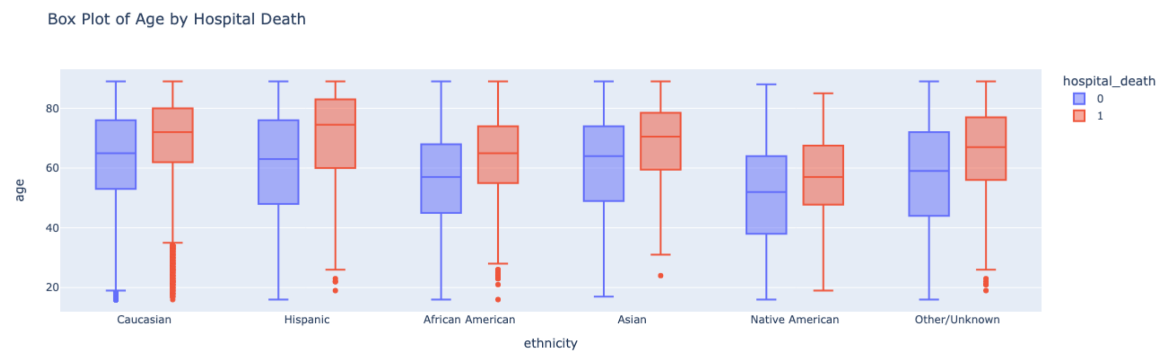


[3]A most pertinent example being this study that was recently published in the *Journal of the American College of Cardiology*. "**Substantial differences** in care are present across the spectrum of acute chest pain management from first contact through to hospital discharge. **Women have higher mortality for STEMI**, but better outcomes for other etiologies of chest pain compared with men" (emphasis added).

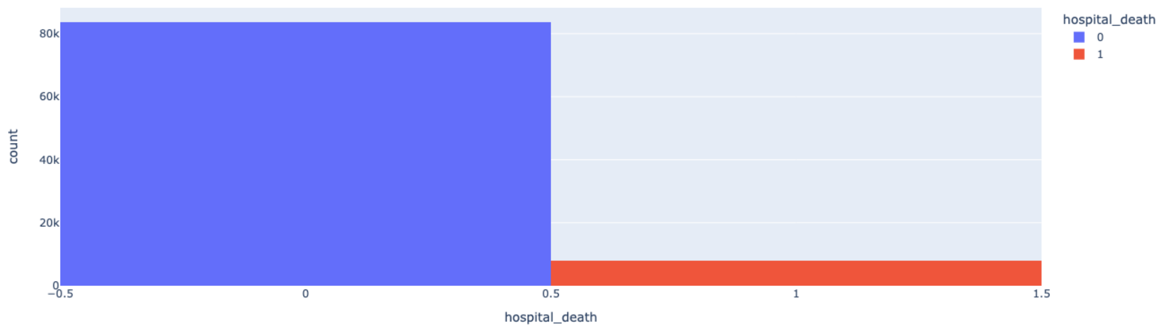




Overall, women appear more likely to die than men.



Native Americans and African-Americans tend to have less probability of death. In contrast, Caucasian, Hispanic, and Asian people tend to have more probability of death.



91.4% of data belongs to `0` (live) and only 8.63% of data belongs to `1` (death).

2.3 Cleaning

As previously stated, there are a number of missing values as revealed by both manual inspection and the results of the `describe` function. For example, the "hospital_id" field is complete, but "bun_apache" (the measurement of blood urea nitrogen) is incomplete with only 72,451 of the 91,713 observations. If every `NaN` were excluded, our dataset would be reduced to about 50 observations, so that was obviously an issue. We opted to deal with missingness *after* we had scaled and normalised the available data.

We decided to use two different methods that we could then compare at the end based on our model results.

1. The first method was to turn every `NaN` into a `0`. This is a crude solution that we realise can also greatly skew the overall dataset because those 0s represent the overwhelming majority of certain features and may thus cause issues. We are aware of this and it is part of our experimental methodology.
2. The second method was to impute the missing values. After trying a variety of imputation methods, we discovered that, overall, they yielded approximately the same results. Therefore, we opted for the simple imputation method of using the mode.[4]

2.4 Feature Engineering, Selection, and Rationale

Of the 196 columns, 170 contain floats, eight contain integers, and eight contain objects. The multitude of *categorical* variables was the first obstacle we encountered. Per the usual

[4]Another approach would be to drop the columns with a majority of `NaN` values prior to imputing. However, this then gets into the murky waters of whether the dropped columns should be determined before or after PCA and how corresponding models would compare.

course, we decided to conduct one-hot-encoding to convert each categorical field. Turning to the numerical fields, we both scaled and normalised their values, as raw values had a wide and varying spread and different value systems are used for different patient features.

As part of our pre-processing, we divided the features according to their medical category (with the aid of the data's accompanying dictionary of feature meanings). That is, each of the 186 features can be classified as either "demographic," "APACHE,"[5] "vital," "labs," or "blood labs."

Of the primary feature categories, we chose to excise nearly all "APACHE" fields and several of the "demographic" ones as well (as it is primarily the demographic variables of gender and ethnicity that we are interested in observing). We then conducted principal component analysis and in-depth exploration of feature correlations and co-linearity.

Our end result was a list of 17 features, as shown in the following table. An explanation of what all features represent, including these 17, may be found in the accompanying CSV document.

- d1_lactate_max
- d1_lactate_min
- h1_lactate_min
- h1_lactate_max
- d1_sysbp_invasive_min
- d1_mbp_invasive_min
- d1_sysbp_noninvasive_min
- h1_albumin_min
- h1_albumin_max
- d1_inr_max
- d1_inr_min
- d1_sysbp_min
- d1_sysbp_max
- d1_albumin_min
- d1_inr_min
- h1_inr_min
- d1_bun_min

Here, we note that, due to the complex nature of the available features and their subject matter, we did not create any new composite variables.

2.4.1 Balancing

As previously illustrated in the last figure of §2.2, the data is significantly unbalanced, with 0 (no death) being the overwhelming majority of observations. In order to account for

[5]Here we note that "APACHE" is an acronym for "Acute Physiology and Chronic Health Evaluations." The "APACHE score" is used as a metric for evaluating the severity of a patient's condition (i.e., differentiating between "mild" and "severe" presentations of a given malady). There are multiple variations of APACHE, primarily APACHE II and APACHE III, both of which appear in this dataset. Notably, APACHE II ranges from 0 to 71 based on lab results whereas APACHE III ranges from 0 to 299 based on both lab results and other contextual and demographic information.

this, the data was processed using the `imbalanced-learn` library. Utilising `SMOTE` and `RandomUnderSampler`, the pipeline produced a more balanced dataframe by oversampling and under-sampling.

2.5 Final Data for Modelling

Ultimately, we created three different dataframes through three separate processes in order to best assess and experiment with how each process impacted the final models.

1. A scaled, normalised dataframe where missing values were imputed using mode. (What we will refer to as the "missing-0" dataset.)
2. A scaled, normalised dataframe where missing values were simply replaced with `0`. (What we will refer to as the "imputed" dataset.)
3. A scaled, normalised dataframe where missing values were not addressed but the dataframe was balanced, as described in §2.4.1. (What we will refer to as the "balanced" dataset.)

3 Modelling

We pursued three different models in order to contrast and compare their results. These three models are:

1. support vector machine (using `svm.SVC` from `sklearn`);
2. xgboost (using the `xgboost` package); and
3. neural network (using `tensorflow.keras`).

With respect to this document's formation, the following sections are organised by *model* rather than by corresponding parts. Results will be synthesised in §4.

3.1 Baseline

There is one true baseline model and one psudo-baseline model against which to compare our final results.

3.1.1 FLAML (Pseudo-Baseline)

We initially applied FLAML to the original, unaltered data to see how it would fair.

Table 1
FLAML's estimators and their configurations.

	lgbm	rf	xgboost	extra_tree	xgb_ld	lrl1
n_estimators	343	14	721	90	30	-
num_leaves	103	-	-	-	-	-
min_child_samples	2	-	-	-	-	-
learning_rate	.0647	-	.0268	-	.4643	-
log_max_bin	10	-	-	-	-	-
colsample_bytree	.8318	-	.7308	-	1	-
reg_alpha	.0020	-	.0009	-	.0009	-
reg_lambda	62.38	-	.3734	-	.0868	-
max_features	-	.0737	-	.1529	-	-
max_leaves	-	6	248	246	-	-
criterion	-	entropy	-	entropy	-	-
min_child_weight	-	-	3.563	-	2.201	-
subsample	-	0	.8274	-	.9661	-
colsample_bylevel	-	-	.3310	-	.9722	-
max_depth	-	-	-	-	4	-
n_estimators	-	-	-	-	-	.25

On cursory review, the estimators behave as one would expect. Namely, `xgboost` fares best overall and `linear_regression` is just.... out there. The FLAML model was validated and resulted in:

- Accuracy = .9232
- F-1 score = .4691

3.1.2 Naïve Bayes (Baseline)

To establish a baseline model and determine a minimum level of performance for our final model, we opted for a more traditional approach and created a Naïve Bayes model. Among several available Naïve Bayes algorithms in scikit-learn, we chose the Gaussian Naïve Bayes algorithm. However, scikit-learn's Naïve Bayes algorithms do not handle missing values in the input data. Therefore, we used the `fillna` method from Pandas to impute the missing values in our dataset, using the values of 0 and 99.

Our baseline model achieved an accuracy between 84% to 86%, a sensitivity between 57% to 67%, and an F-1 score between 21% to 35%, with hyperparameter tuning. Thus, we have set the benchmark as 86.31 accuracy.

Table 2
Baseline model score (Naïve Bayes)

	Accuracy	Precision	Recall	F-1 Score	AUC
FillNA with 99	{'priors': None	'var_smoothing':	1e-09}		
Baseline	.8424	.1862	.5729	.2125	.6891
FillNA with 0	{'priors': None	'var_smoothing':	1e-06}		
Baseline	.8631	.2951	.6656	.3491	.7739

3.2 Support Vector Machine

3.2.1 Training & Validation & Testing

Because of the need to experiment with training and validation on three different datasets, the following describes the pipeline (in name only) process used for each of the three.

First, the dataset in question (one of "imputed," "missing-0," and "balanced") underwent splitting. Separating it into its `X` and `y` parts, `train_test_split` was applied from `sklearn` with 70% for training and 30% for what we will refer to as validation. This done, four models were instantiated.

1. `LinearSVC` (no kernel argument required)
2. `SVC(kernel = 'poly')`
3. `SVC(kernel = 'rbf')`
4. `SVC(kernel = 'sigmoid')`

Each model was subsequently fitted upon the training portion, used for prediction on the validation portion, and then scored using the model's own `score` method (capturing the mean accuracy of its predictions against the "ground truth"). All of the results were recorded and may be found below in §3.2.3.

Next, the pre-existing[6] `train_X` and `test_X` dataframes were filtered so that the resulting, now `train_X_best` and `test_X_best`, contained *only* the 17 "best" features, as listed in §2.4. A *second* set of models were instantiated[7] and fitted, used for prediction, and scored.

Because each model had poorer performance in every single instance of the reduction of features to just 17, these second models were not applied to the true testing dataset. This

[6]That is, the splitting proportions and contents did *not* change as no new `train_test_split` was conducted.

[7]That is, *not* the same models previously trained and tested, but new ones entirely.

choice served the additional purpose of reducing the already extensive computational time required.

3.2.2 Hyperparameter Tuning

Following the assessment of the four varieties of `SVC` kernels, each kind was once again processed using `RandomizedSearchCV`. The parameters for `poly`, `rbf`, and `sigmoid` are as follows:

- `"C": uniform(loc = 0, scale = 3)`
- `"gamma":  uniform(loc = 0, scale = (3/171)) #the final # of feats`
- `n_iter = 10`
- `n_jobs = -1 #running all processors; parallel computing`
- `scoring = ['accuracy', 'f1', 'precision', 'recall']`
- `refit = 'accuracy'`
- `cv = 5`

As mentioned, these parameters only apply to three of the four kernels. `LinearSVC` differs because there is no "gamma" argument. Instead, the second parameter distribution argument was `"penalty":['l1', 'l2']`

The process was the same as with the standard ones. Each model was run on the data with all-inclusive features and a new iteration was run on the data with only 17 features. Once again, only the models trained on all available features were applied to the final true testing dataset.

3.2.3 Results

Each round of testing on a given dataset resulted in 16 results. The net result was 48 scores with accompanying cross-validation parameter tabulations. The *overall* scores are reported below (e.g., of all 10 `RSCV` iterations). "CV" indicates the `RSCV` models.

Table 3
Training and validation results using <u>all</u> numeric features.

	Lin	Poly	RBF	Sig	Lin CV	Poly CV	RBF CV	Sig CV
Imputed	.9224	.9222	.9240	.8745	.9223	.9222	.9249	.9194
Balanced	.8058	.9222	.8169	.7261	.6670	.9263	.9303	.6973
Missing = 0	.9261	.9223	.9287	.8803	.9254	.9263	.8161	.9050

Table 4
Training and validation results using 17 best numeric features.

	Lin	Poly	RBF	Sig	Lin CV	Poly CV	RBF CV	Sig CV
Imputed	.9156	.9158	.9162	.8439	.9153	.9149	.9160	.9089
Balanced	.7424	.9148	.7536	.6112	.7422	.7278	.7505	.7360
Missing = 0	.9204	.9148	.9210	.8503	.9204	.9199	.9207	.9139

According to Table 3, the RBF kernel achieved the highest accuracy with hyperparameter tuning on the balanced dataset. Indeed, overall, the RBF kernel performed better than the other three kernels when all features were used. In contrast, Table 4 shows that the results of hyperparameter tuning were mixed at best, with the only increase occurring with the sigmoid kernel. The RBF kernel, albeit without hyperparameter tuning, still produced the best results on the imputed and missing = 0 datasets.

Below are the details for the best *iteration* of the RBF kernel *with hyperparameter tuning*. The values are the *mean* of the relevant iteration is reported. As one can observe, while overall accuracy dropped on the balanced dataset, recall was significantly better than on the other two datasets while improvement in precision was modest.

Table 5
Parameters and metrics with hyperparameter tuning using all numeric features.

	C	gamma	Accuracy	Precision	Recall	F-1 Score
Imputed	2.019	.0037	.9254	.7671	.2509	.3780
Balanced	1.539	.0056	.8226	.7822	.6488	.7092
Missing = 0	2.019	.0037	.9277	.7736	.2313	.3560

3.2.4 Experimental Results

Finally, the RBF kernel with hyperparameter tuning was trained and tested on all three datasets with three key modifications:

1. Each dataset was treated to the same balancing pipeline that was originally applied to only the balanced dataset. (No change to the already-balanced dataset.)
2. Assisted by `LabelEncoder`, gender and ethnicity were added back into the X portion of each dataset, thus allowing the model to "see" these previously withheld features.
3. All APACHE features were removed using `regex`, reducing the total number of features to 133 (from 171) prior to the addition of gender and ethnicity.

Results recorded indicate the best iteration of each model.

Table 6
Parameters and metrics with hyperparameter tuning using all non-APACHE numeric features with encoded gender and ethnicity.

	C	gamma	Accuracy	Precision	Recall	F-1 Score
Experimental Imputed	1.618	.0039	.8045	.7788	.5823	.6662
Experimental Balanced	2.419	.0060	.7960	.7480	.5857	.6569
Experimental Missing $= 0$	2.798	.0066	.7976	.7524	.5780	.6537

While this more experimental approach is cursory at best, it is significant to observe the generally higher results of precision, recall, and F-1 score compared to the models' predecessors. Of course, this increase comes with a decrease in accuracy. The confusion matrices of these experimental results compared to those of the original three formulations are shown and discussed in §4.1.

3.3 xgboost

3.3.1 Training & Validation

In order to preserve a balanced proportion of both classes, the data was split using stratified `train_test_split`, allocating 70% for training and 30% for what will be referred to as validation. Only the numerical features were used for the XGBoost models.

All of the results were recorded and may be found below in §3.3.3.

3.3.2 Hyperparameter Tuning

The parameter that performed best was the one with the missing values replaced as 0. The two model was then processed with `GridSearchCV`. The best parameters were found to be the following:

- `"gamma" = 1`
- `learn_rate = 0.1`
- `max_depth = 5`
- `reg_lambda = 0`
- `subsample = 0.9`
- `colsample_bytree = 0.5`

The scores from the cross-validation parameter tabulations are reported below.

3.3.3 Results

Table 7
Training and validation results before hyperparameter tuning

	Accuracy	Precision	Recall	F-1 Score	AUC
Imputed	.8198	.7547	.6805	.7157	.8882
Balanced	.8216	.7479	.7012	.7238	.8869
Missing = 0	.8216	.7540	.6901	.7206	.8885

Table 8
Training and validation results after hyperparameter tuning

	Accuracy	Precision	Recall	F-1 Score	AUC
Missing = 0	.8292	.7658	.7024	.7327	.8921

Balancing via SMOTE was also applied to the "imputed" and "missing-0" data. This step was done instead of using the `scale_pos_weight` parameter in the model since using the weights as a way of balancing did not improve the accuracy from previous experimentation of the model. Hence we chose this method of dealing with the imbalanced data for XGBoost modeling. It is no surprise that the "balanced" and "missing-0" data performed almost similarly since the option for XGBoost to deal with missing values by replacing them with 0 is applied in all of the models. However, we still treated the "missing-0" as the best one since it has both the highest accuracy and AUC score. The overall metrics after hyperparameter tuning improved at the very least.

3.4 Neural Network

3.4.1 Training & Validation

Each dataset was split into training and validation sets, with a ratio of 70% to 30%, respectively. For the neural network model, we only used numerical features, totaling 155 columns. We recorded all the results, which can be found in §3.4.3.

3.4.2 Hyperparameter Tuning

By using `BayesianOptimization` with an objective of maximizing accuracy, we were able to identify the optimal hyperparameters for our neural network. Since the neural network does not automatically apply weight to underrepresented classes, we manually examined the weight as follows:

- Weight for `class 0:  0.55`

- Weight for `class 1:  5.79`

From the above manual result, we set the range of weight in the hyper parameter pipeline from 5 to 6 with step of 0.25.

The best combination of hyperparameters was found to be the units in the two dense layers of 288 and 16 each, a droupout rate of 0.3, and a learning rate of 0.001. This configuration resulted in a significant improvement in model performance, achieving 80.17% accuracy, 81.48% recall, and 88.02% F-1 score, a notable improvement over the baseline Naïve Bayes model, which only achieved 86% accuracy, 67% sensitivity, and 35% F-1 score. The optimal parameters were identified as follows:

- `batch_size = 64 # set manually`
- `epochs = 100 # set manually`
- `dense_units1 = 288`
- `dropout_rate1 = 0.3`
- `dense_units2 = 16`
- `dropout_rate2 = 0.3`
- `learning_rate = 0.001`

3.4.3 Results

Table 9
Training and validation results before hyperparameter tuning

	Loss	Accuracy	Precision	Recall	F-1 Score	AUC
Imputed	.3725	.8966	.3637	.1700	.9446	.6631
Balanced	.8237	.3910	.3130	.7190	.3381	.4860
Missing = 0	.5291	.8224	.1280	.1836	.9008	.5325

Table 10
Training and validation results after hyperparameter tuning

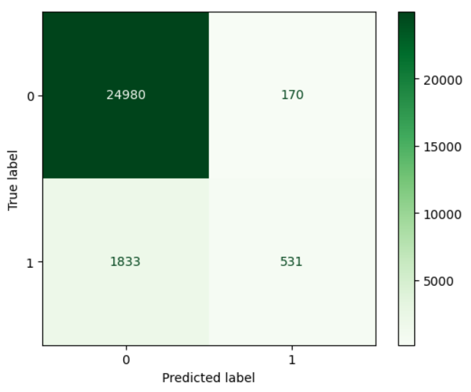
	Loss	Accuracy	Precision	Recall	F-1 Score	AUC
Imputed (Unweighted)	2.3062	.0897	.0897	1.000	.9138	.5949
Imputed (Weighted)	.4201	.8017	.2870	.8148	.8802	.8854

4 Analysis

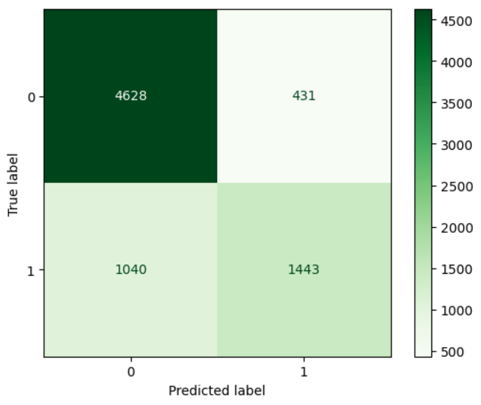
In each of the following sections, the captioned model's results are discussed and interpreted.

4.1 Support Vector Machine

Regarding the performance of the four different kernels available for `SVC`, the varying performance of each is not unexpected. We discount the performance of the polynomial kernel because of how tedious and impractical it is from a calculation and computation perspective, as well as suffering from the drawback of over-fitting. The linear and RBF kernels are the two that had the best performance and, therefore, are generally the two that we are interested in for the purposes of analysis. Comparing the confusion matrices of the RBF kernel below, one may note the change in results between the original datasets and their more-modified counterparts (the latter being described in §3.2.4).

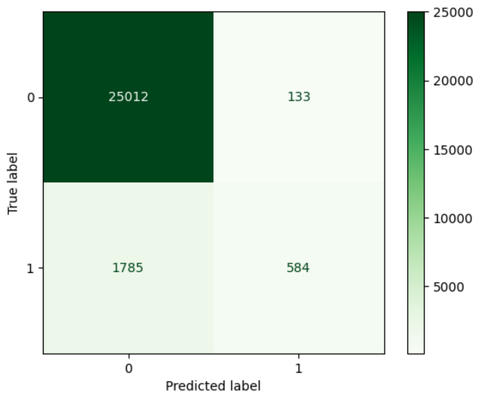


(a) RBF with hyperparameter tuning on the imputed dataset.

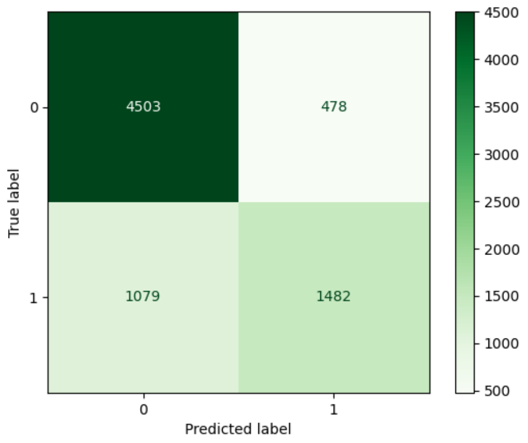


(b) RBF with hyperparameter tuning on the experimental imputed dataset.

Figure 1
The imputed data.



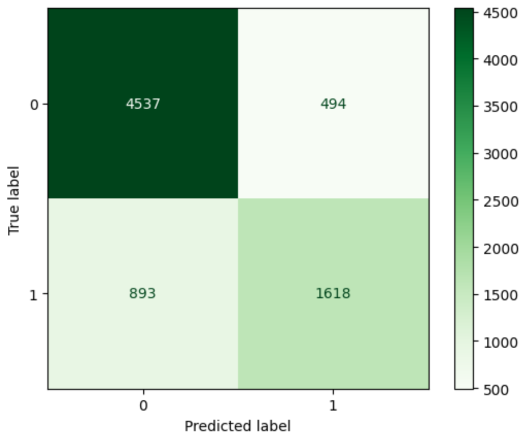
(a) RBF with hyperparameter tuning on the missing = 0 dataset.



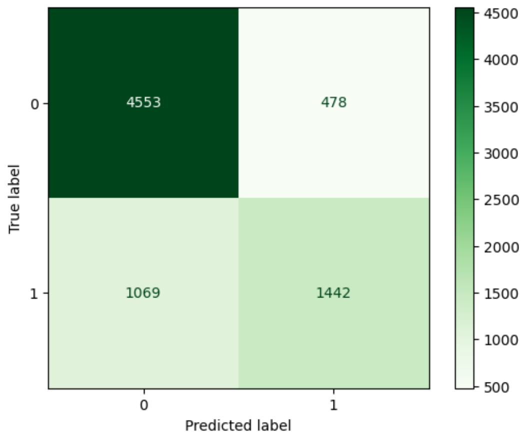
(b) RBF with hyperparameter tuning on the experimental missing = 0 dataset.

Figure 2
The missing = 0 data.

The original results between the imputed and missing = 0 datasets are comparable, and the effects of the experimental datasets are nearly equal in magnitude. As previously shown in §3.2.3, recall and precision both increase, corresponding with the modest reduction in Type II errors (the false negatives in the lower left quadrant) and the increase in Type I errors (the false positives in the upper right quadrant). Additionally, there is an increase in correct predictions for the lesser of the two classes (e.g., death = 1). In contrast, observing the confusion matrices for the balanced dataset reveals that there is a reduction in correct predictions for class 1 but also an increase in Type II errors. However, both of these changes are relatively slight when one considers the entirety of the available data.



(a) RBF with hyperparameter tuning on the balanced dataset.



(b) RBF with hyperparameter tuning on the experimental balanced dataset.

Figure 3
The balanced data.

It is not entirely clear which modifications for the experimental datasets led to the increase in recall and precision. PCA analysis would need to be conducted and reassessment to test the modifications in isolation. More broadly, however, in biostatistics, recall and precision are generally more relevant for day-to-day operations than overall predicting accuracy. In the context of our data, this principle aligns with what we would expect. Understanding patient profiles is critical for medical professionals as healthcare is a dynamic environment. Our target variable of death or not-death regards the ultimate outcome of the combination of patient profiles and according medical intervention, both of which are the primary focus of healthcare workers.

4.2 xgboost

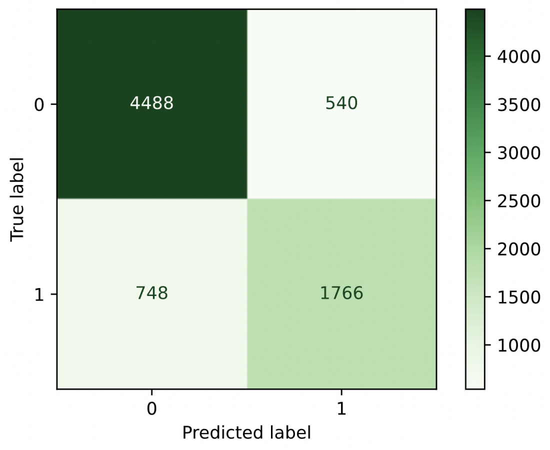


Figure 4
XGBoost with hyperparameter tuning on the missing = 0 dataset.

As can be seen from the confusion matrix, there are less samples due to the combination of under and oversampling performed prior the modeling. In this case, the precision is slightly higher than the recall. However, it may be important to note that this may be riskier for predicting hospital deaths. This result may be due to method of handling the imbalanced data. After applying oversampling and undersampling via SMOTE, the final balance is still around 1:3 where the minority is class 1. Applying the weights in the hyperparameter can increase the recall, but may not necessarily retain or increase the precision which can lead to a lower or unimproved accuracy rate.

4.3 Neural Network

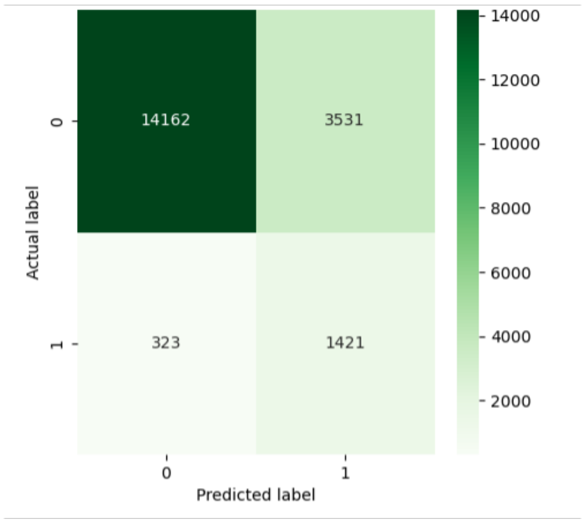


Figure 5
The imputed data.

Based on the confusion matrix, we can observe that there are a relatively high number of false positives when compared to false negatives. This is primarily due to the imbalance in the dataset. However, in the prediction of hospital death, it is of utmost importance to avoid false negatives and prioritise the metric of recall, which is equivalent to sensitivity. This is because it is crucial to correctly identify patients who are at high risk of dying. To address this issue, we applied a weighted method and oversampling to the model. However, oversampling did not show significant improvements compared to the weighted method.

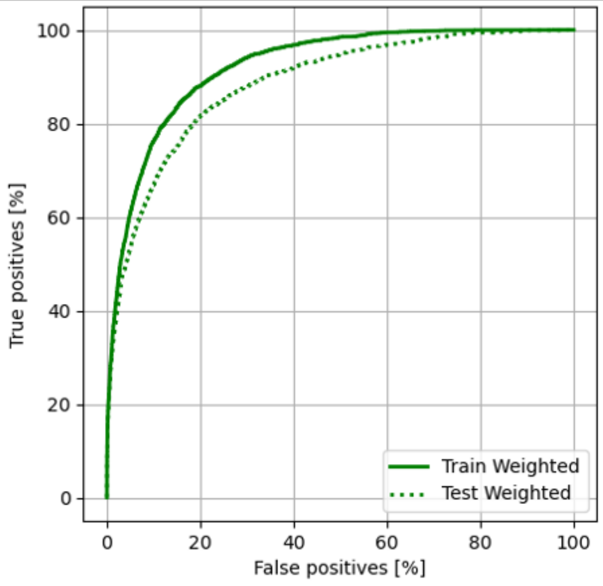


Figure 6
The imputed data.

To evaluate the performance in depth, we examined the ROC curve and found that the area under the curve (AUC) reached 88.54% in the validation dataset ("Test Weighted" in the legend), which indicates a good balance between true positive and false positive.

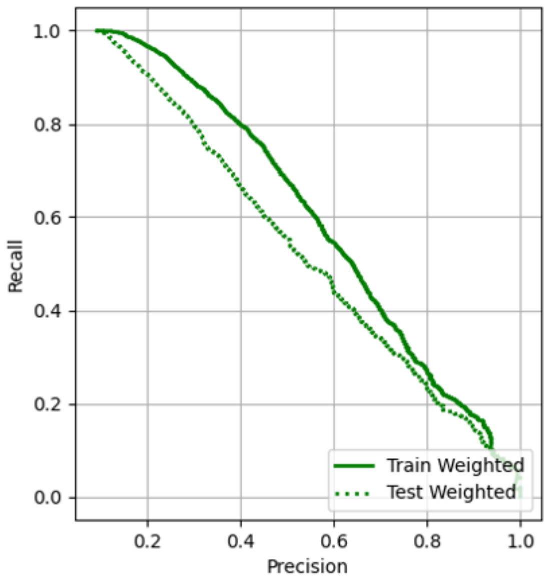


Figure 7
The imputed data.

On the other hand, the precision-recall curve showed that the area under the curve (AUPRC) only reached 55.78%. Although an AUPRC of above 70% is generally considered a good performance, our main goal is to predict hospital death. Therefore, this moderate performance in terms of precision may be acceptable if we can achieve a high F-1 score. With the weighted neural network model, we were able to balance the precision and recall, achieving an 88% F-1 score.

5 Conclusion

5.1 What We Learned

5.1.1 The Problem of Bias

While our original objective was to examine bias, this was, of course, difficult to articulate throughout the convolutions of our project. On the one hand, demographic information should not affect the quality or appropriate level of care a patient should receive. However, on the other hand, in computational terms, the differing occurrences of different health problems for different populations must also be taken into account, though the degree of such is uncertain. Much more data is needed to create a fuller picture, *especially* data for women and minorities. What's more, another layer of complication for bias and gender is the introduction of patients who do not strictly fall within the cisgender binary. However, there may simply not be enough data to adequately assess this particular aspect that it effectively becomes an outlier scenario.

Ideally, we would test more demographic features, particularly ones such as readmission status and ICU admission type. Similarly, it might be a worthwhile pursuit to create models for each population within the data that may then be integrated together to see the net effect of such an approach. Future work would also require an extensive literature review to compile and collate other methods used for such an endeavour and to situate our particular question.

5.1.2 Project Process

While there is still much that could be done and more models made to eke out the nuances within the data, we are quite pleased with our efforts. Throughout the development of this project, the most crucial aspects for our success were coordination and frequent conferencing. While we made efforts to largely follow the same procedure in each of our models, we acquiesced to the fact that replicating the exact same process with three different people and three different models was not something we should strive for.

We hope that the variation of our methods are cogently described within the relevant sections. The components of this project were divided approximately as:

- Anna — FLAML, preparation of 3 datasets, **SVC modelling**, relevant slides and final proofing of the presentation, nearly all aspects of document writing, editing, and formatting, and relevant sections for SVC
- Angelica — EDA, correlations and correspondences, **xgboost modelling**, supplementary statistical insights, relevant presentation slides, and relevant sections for xgboost
- Chaeyeon — EDA, PCA, Naïve Bayes, imputation, scaling, troubleshooting modelling data issues, **neural network modelling**, presentation template creation, relevant presentation slides, and relevant sections of this document for the neural network

5.2 The Bigger Picture

People in STEM (particularly statistics, mathematics, and computer science) oftentimes have the misconception that they are working in a large, disinterested vacuum. and do not consider the underlying nature and structure of the information in question. This is a profound oversight that can have serious consequences. For example, OpenAI's refusal to reveal anything about the model architecture and training procedure for their GPT iterations is of great concern because there is no way to assess shortcomings like bias. More pertinent to our project, however, are the specific concerns about data's human impact, as succinctly described in "On the Dangers of Stochastic Parrots" (Bender et al., 2021)[8].

The people who shape the narrative of digital information today have a responsibility to not only understand the context and consequences of what they do, but to also act on it.[9] Women and minorities are historically at greater risk in terms of healthcare; it is a system problem that is not easily remedied. However, we would like to do everything we can as we move forward in our respective careers to be thoughtful and compassionate towards the people who are too often obscured by our approach to the science of information.

[8]Bender, Emily and Gebru, Timnit. "On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?" (2021) In Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency. *Association for Computing Machinery*, 610–623. https://doi.org/10.1145/3442188.3445922

[9]For example, "Mathematicians urge colleagues to boycott police work in wake of killings."