Predicting Cirrhosis Patient Survival From Demographic and Clinical Variables

Alexander Weitzman
Brown University
https://github.com/alexweitzman5/midterm-project

December 2023

0.1 Introduction

Predicting health outcomes allows doctors to make informed treatment decisions, understand the course of an illness, and develop a prognosis. This paper describes a classification model designed to predict cirrhosis patient survival outcomes. It was trained on a dataset of 418 cirrhosis patients at the Mayo Clinic, selected for the clinical trial of the drug D-penicillamine.

Cirrhosis results from prolonged liver damage, is characterized by extensive scarring of the liver, and is often caused by hepatitis or chronic alcohol consumption. Effects from cirrhosis cannot be undone and may lead to liver cancer. But if the disease is caught early, treatment can extend life expectancy. Understanding the local severity of a patient's condition can inform a treatment plan.

The dataset is from Kaggle and includes 312 participants who have completed clinical tests with no missing values. The remainder did not participate in the clinical trial, but instead recorded basic tests and agreed to survival tracking. These data points include missing values. There are 18 features describing age, sex, severity of disease (stage), disease horizon (N_Days), and symptom tests (thirteen individual features). A categorical variable, status, is the target variable, and is broken down into three categories: censored (C), censored due to liver transplantation (CL), and death (D). This study considers a classification problem to predict whether patients will survive institutionalization, require a transplant, or die.

Two previous Kaggle members developed similar models to predict survival outcomes. The first used f1 scores to evaluate predictive power. The best model reached an f1 score of 0.835, on the higher end achieved by this study's models. The developer imputed missing values and trained three types of models with varying hyperparameters. The other attempt also trained three model types and used logloss as its evaluation metric. They incorporated clinical trial drug data, demographic, and clinical variables in order to predict survival outcomes.

0.2 EDA

EDA revealed only 25 patients had status CL. The imbalance is demonstrated in Figure 1 and indicates the model is working off limited samples of CL patients (see Methods)

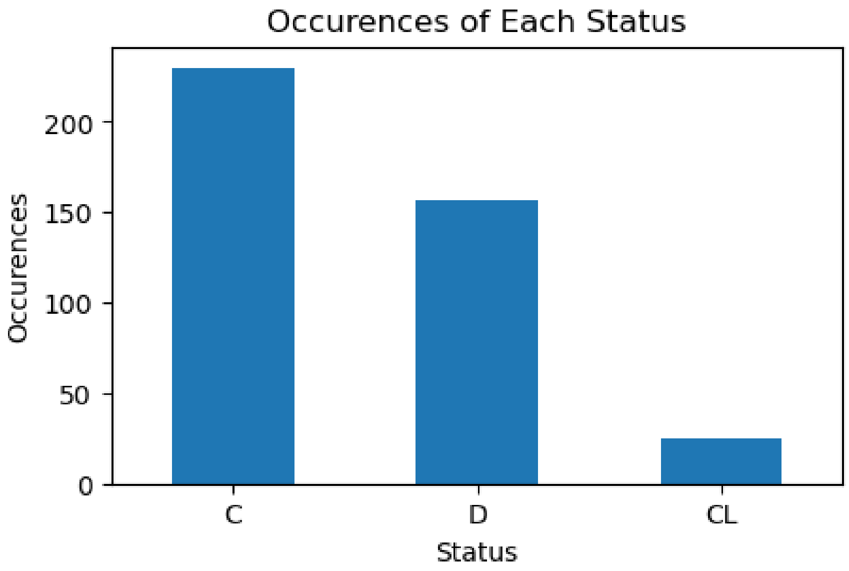
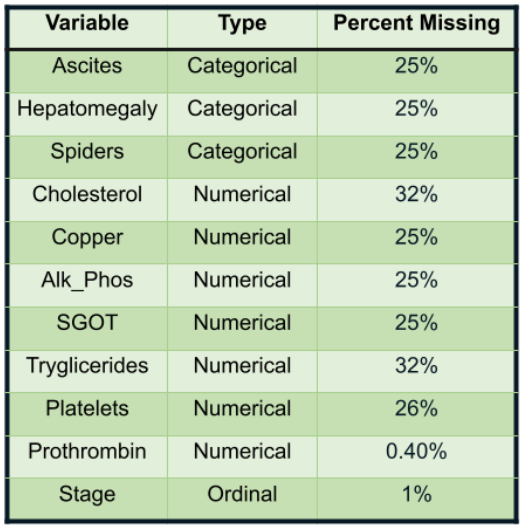


Figure 1: Occurences of target (status): C (censored), CL (censored due to liver transplant), and D (death)

Figure 2 shows the percentage of missing values found in the data, and clustered, balanced missing values reflect omitted tests.



Variable	Type	Percent Missing
Ascites	Categorical	25%
Hepatomegaly	Categorical	25%
Spiders	Categorical	25%
Cholesterol	Numerical	32%
Copper	Numerical	25%
Alk_Phos	Numerical	25%
SGOT	Numerical	25%
Tryglicerides	Numerical	32%
Platelets	Numerical	26%
Prothrombin	Numerical	0.40%
Stage	Ordinal	1%

Figure 2: Table of features and their percentage of missing values

Additionally, stage, a categorical variable indicating disease severity, is disproportionally high, indicating most cases are severe (Figure 3).

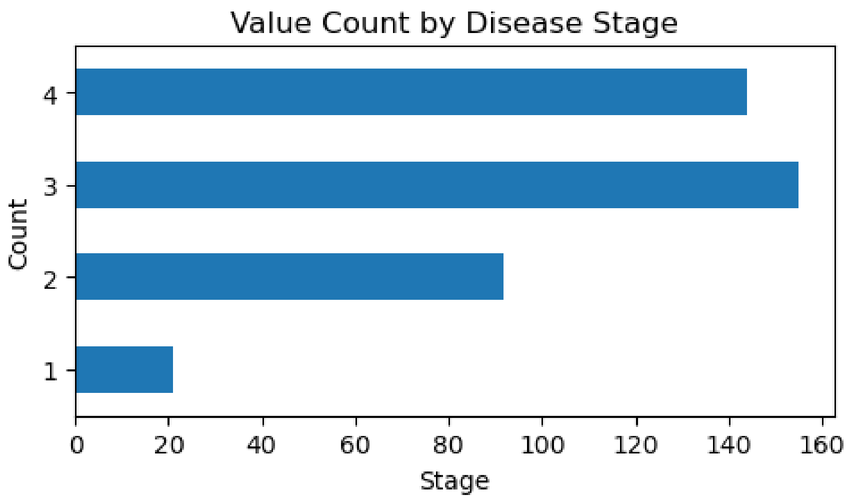


Figure 3: Breaks down disease stage distribution

The data is also 89% female, despite women only accounting for 30-45% of incidence. Overall, our model may be best deployed for women and severe cases, as it is predominantly trained on them.

EDA also revealed data correlation, shown in Figure 4.

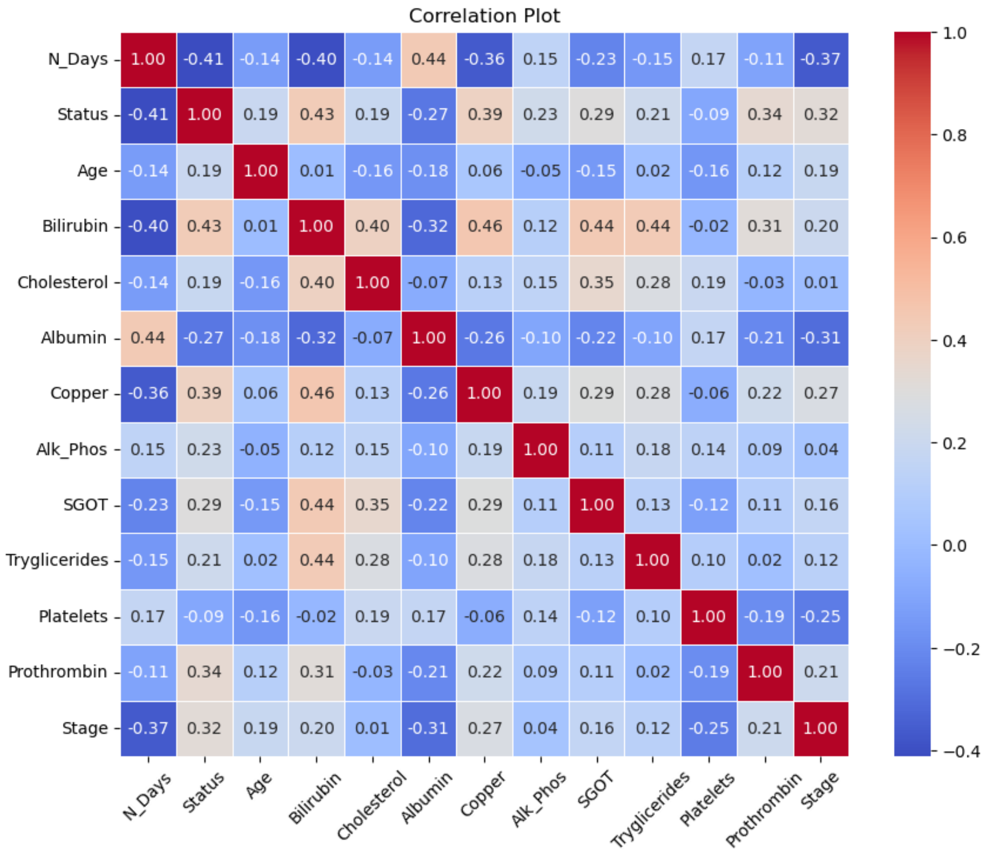


Figure 4: Correlation matrix: includes analyzed numerical features

Note relatively high correlations within symptom variables, and correlation between features and the target variable. Figure 5, which is normalized against death, shows disproportional Ds in the patients with ascites – fluid found in the abdomen – than those without.

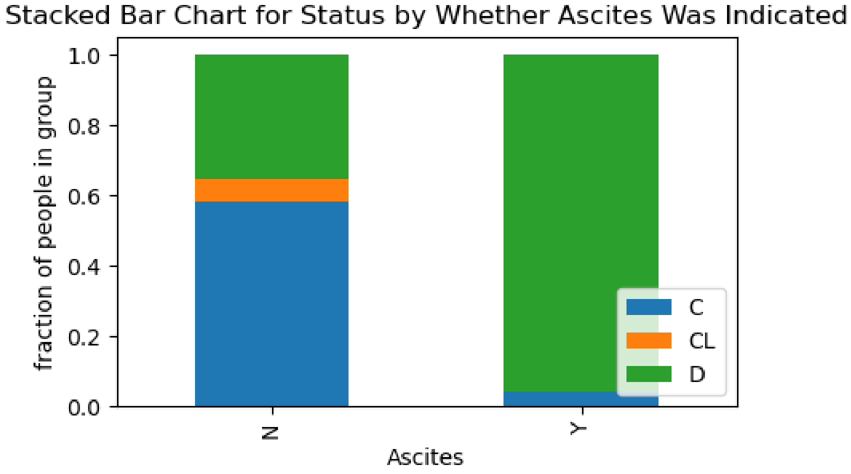


Figure 5: Status portion by ascites classification normalized against D

Figure 6 shows how bilirubin levels mostly fall between 0 and 5, but the ranges are more extended in D than C.

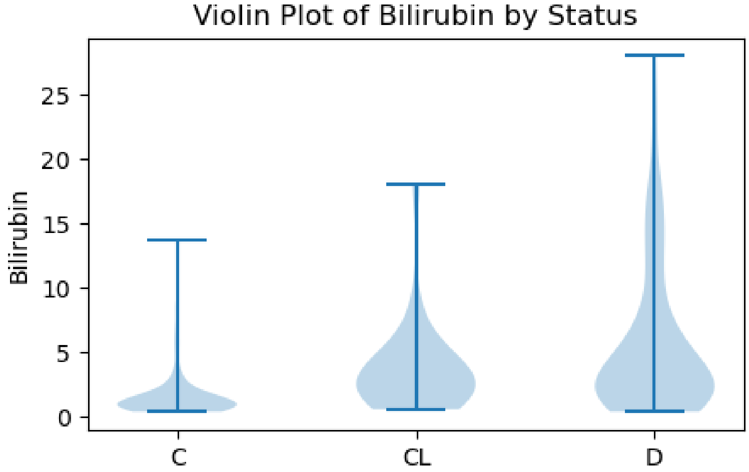


Figure 6: Visualizes distribution of bilirubin by status classification, with error bars

0.3 Methods

Only 6% of data points recorded CL, so our target variable is imbalanced, necessitating StratifiedKFold. Since we used five folds, 15 CLs were included in training sets and five in test sets, ensuring every model was trained and tested on an adequate number of CLs. The feature matrix and y-labels were first inputted into train_test_split with test size set to 20%, before StratifiedKFold was applied. Features were then categorized into ordinal, categorical, and numerical, to be preprocessed separately. Since ordinal and categorical features contained missing values, we used SimpleImputer to treat nans as their own designation, and then applied OrdinalEncoder and OneHotEncoder respectively. Consequently, fit_transformation of the test set and transformation of val and training meant feature options were split into separate columns and total features increased from 18 to 26.

Numerical variables were more complicated. Rather than imputing medical data and assuming test results, we chose to use the reduced feature model. This meant training models on data matched by and then excluding each of the unique missing value paring. Since non-participants were responsible for the majority of missing values, we only needed six datasets with unique missing value patterns to be matched. Creating a function to mutate the preprocessed data and output a list of six lists containing the mutated sets without its missing value columns satisfied this goal. That way, instead of trying to predict results of medical tests, we could train different models on complete datasets that were more indicative of individual patient cases for which the model would be deployed.

This study uses f1 score to evaluate each model. F1 score considers both precision and recall equally, which best fits the use case. High precision avoids false positives, which in practice means administering treatment to those who don't need it. High recall avoids false negatives, which in practice means not treating severe illness adequately. Both can be catastrophic for the patient, so f1 score weights them evenly.

Our dataset was small enough to use GridSearchCV in our CV pipeline to find the best hyperparameters to maximize f1 score. We wrote a generalized function – able to take in any model type as input – to loop through five random states for all above steps and train five corresponding models on each of the six unique feature pairings. The best score at each random state was inputted into a list, so that the final five scores and their corresponding models and reduced datasets could be stored and analyzed.

We used this generalized pipeline for three models – SVC, RandomForestClassifier, and LogisticRegression, and also trained an XGBoost model without using the reduced feature model. Since XGBoost deals with missing values internally, we were able to develop a separate pipeline using all the training data instead. Each model had a unique parameter grid: SVC tuned C and gamma, with five and three candidate values respectively, spaced between 0.1 and 100. RandomForestClassifier's parameter grid spanned max_depth and max_features. Max_depth contained five evenly spaced candidates from 1 to 20 (approximately equaling the number of features by convention), while max_features contained five evenly spaced values from 0.5 to 1. LogisticRegression tuned C and penalty. C contained seven evenly spaced values between 0.001 and 1000, while penalty tried l1, l2, and elasticnet to consider normalization differences. Finally, XGBoost considered max_depths of 3, 5, and 7, and reg_alpha and reg_lamba of five evenly spaced values between 0.01 and 100. Also note this grid sets colsample_bytree and subsample below one, at 0.9 and 0.66 respectively.

Uncertainty in f1 score comes from both splitting and non-deterministic ML methods. Thus, we measure the standard deviation of best scores at each random state to measure uncertainty. Figure 7 shows the fluctuation in f1 scores caused by randomness, with similar averages and standard deviations throughout. Influence of randomness should be relatively high given the low sample size, so the discrepancies are expected.

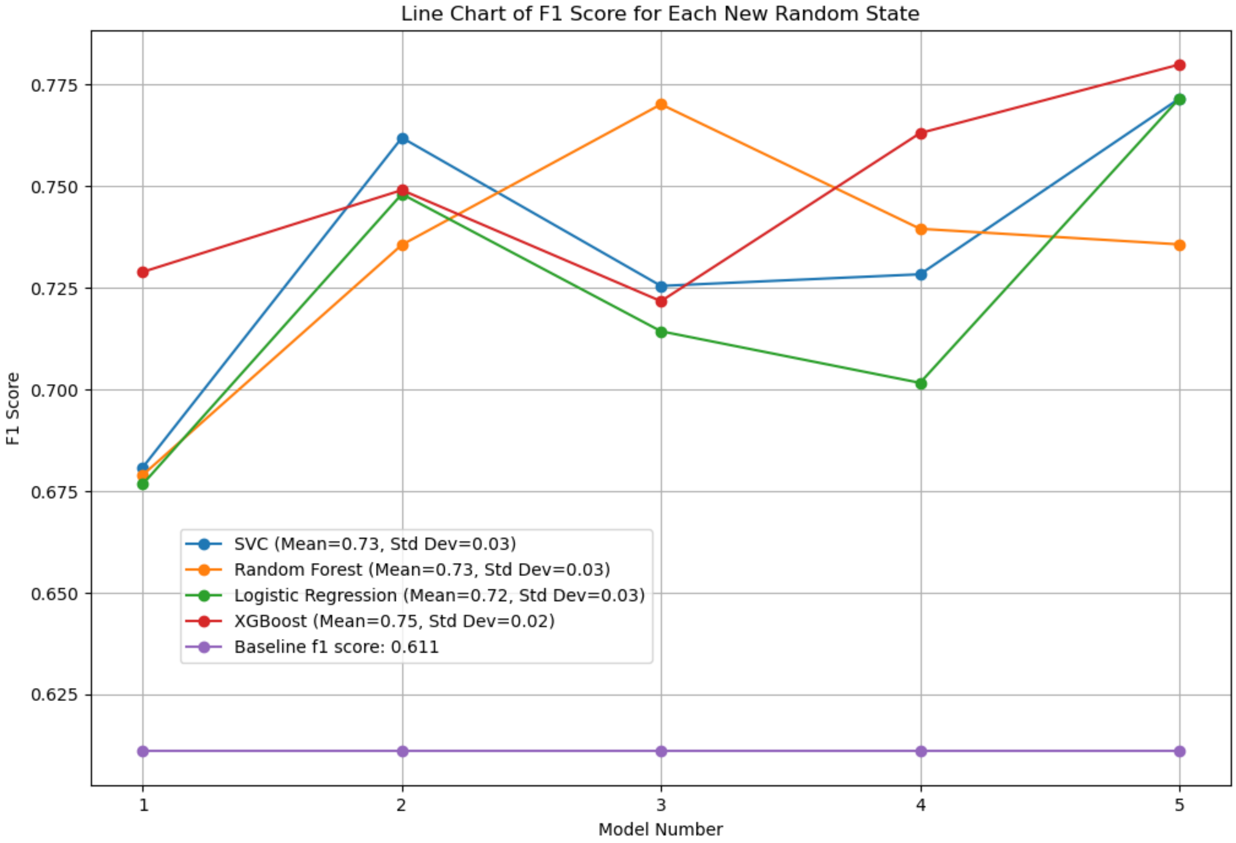


Figure 7: F1 score fluctuation of four models at all five random states against baseline score.

0.4 Results

All four average scores fell well above the baseline. The baseline f1 score, calculated by only choosing the majority class C, is 61.11 and represented in Figure 7 by the purple line. Each model's average f1 score was between 0.72 and 0.75, and the standard deviations of their five scores ranged from 0.02 to 0.03. The gap between average and baseline was as low as 3.66 times LogisticRegression scores' standard deviation, while it was up to 7 for XGBoost.

Figure 8 compares means and standard deviations for all four models. We can see XGBoost had the highest average and smallest standard deviation.

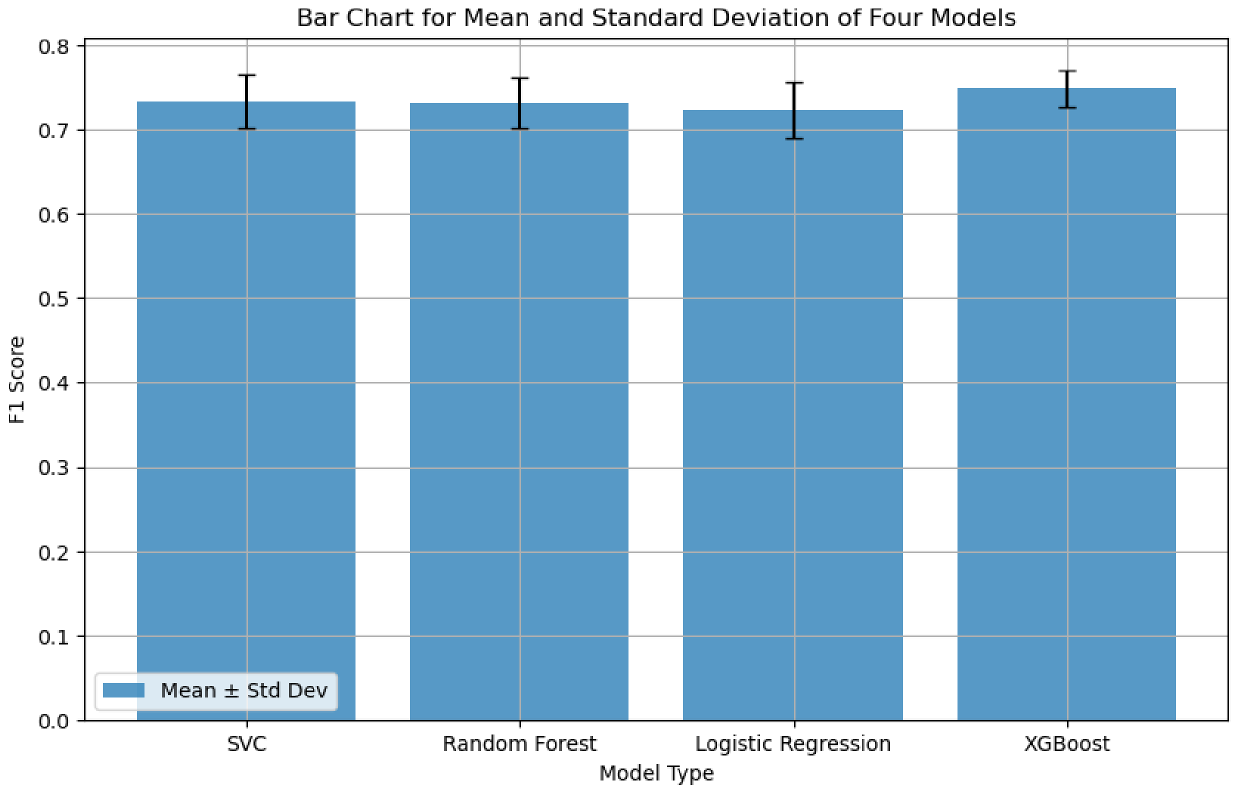


Figure 8: Mean f1 scores across random states with standard deviation error bars

Since XGBoost incorporates trends in missing values, and each model was trained off more data rather than reducing features, This makes sense. The model corresponding to the last random state tried, had the best f1 score at .7799, so we will be using it and the corresponding training and test sets, as our best case. Note the model's hyperparameters were $max\_depth = 7$, $reg\_alpha = 1$, and $reg_lambda = 100$.

As seen in Figure 4, there is high correlation between features. Accordingly, permutation importances are less reliable, though they are shown in Figure 9.

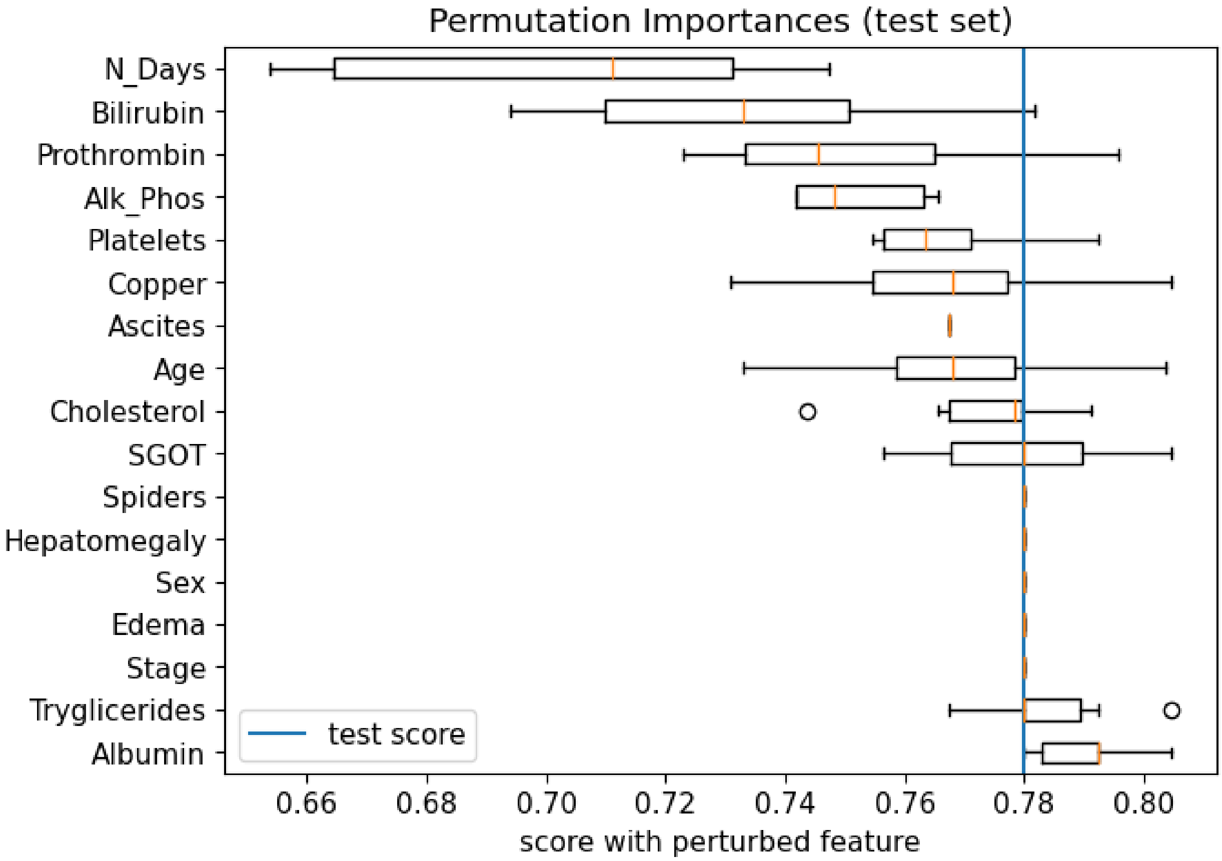


Figure 9: Shuffled feature f1 score mean and standard deviations versus model f1 score

N_Days as well as the test numerical variables are of highest importance, decreasing predictive power by up to 0.06 on average when shuffled. These numerical variables have wider ranges of outcomes and seem to carry the weight of symptom importance given correlation between tests.

Weight, gain, cover, total gain, and total cover are more reliable importance metrics (Figure 10).

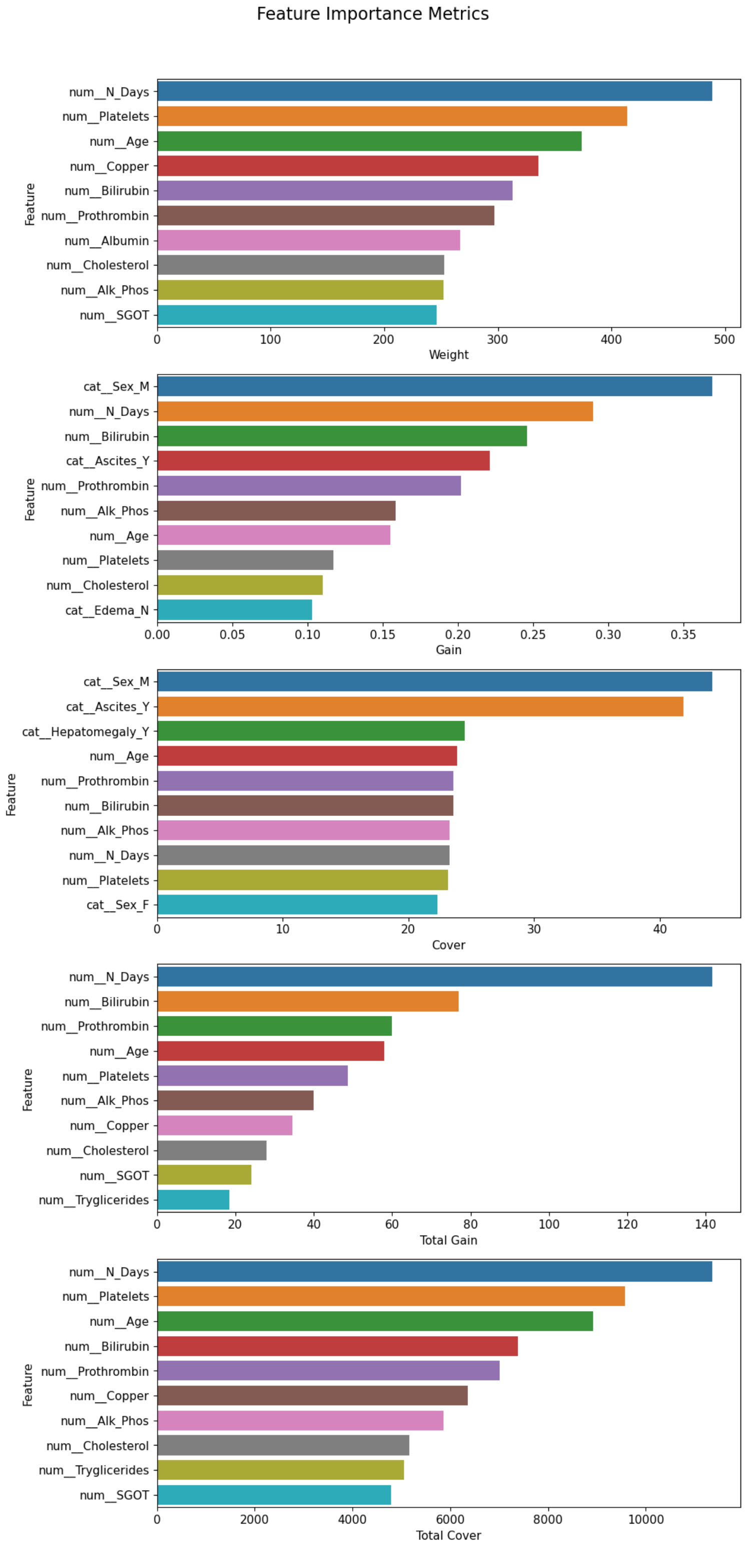


Figure 10: Top 10 values for five XGBoost feature importance metrics

Notice N_Days is also the most important via weight, total gain, and total cover, while being male is most important by gain and cover. High coverage implies many data points are related to sex – which aligns with the overall sex imbalance – and that the feature has high impact in each tree. N_Days' importance seems to come from its relative frequency within each tree. In the context of this problem, it means decisions keep returning to the symptom horizon. We notice differences between metrics, but there are some that are consistent throughout. mirroring permutation importance, N_Days, platelets, age, bilirubin, prothrombin, and alkaline-phosphatase levels are top ten in all five metrics, with age and prothrombin occurring in the top five every time. Overall, doctors can use this information to explain to patients which results are most concerning and understand which to target in treatment. To our surprise, age was consistently an important feature, at magnitudes on par with diagnostic features. It may be that age introduces confounding variables on health outcomes, and future research should consider related variables.

Figure 11 compares three features' local importance.

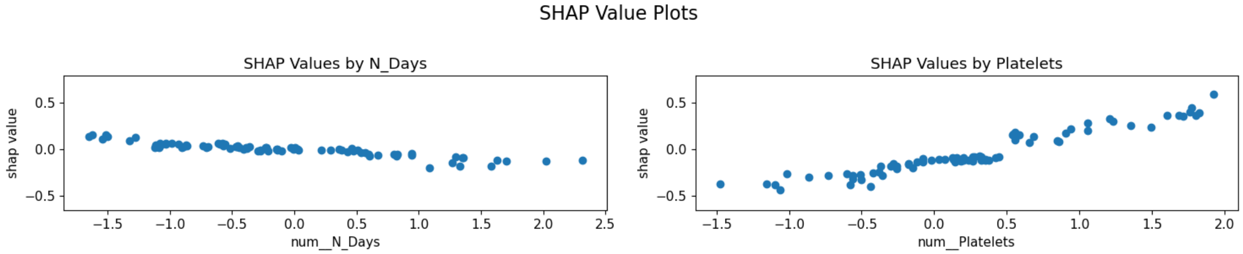


Figure 11: Comparing SHAP values for N_Days and Platelets

N_Days is downward sloping, indicating that extremes have the most positive/negative effect respectively. Cholesterol is upward sloping, so high levels mean a highly positive impact on symptom prediction, while low levels have an equal but opposite effect. Note that the slope for N_Days is smaller than cholesterol, meaning the latter has a larger absolute effect. These trends correspond to global feature importance, though here symptom tests supercede N_Days.

The confusion matrix in Figure 12, considering D as a positive case, shows $recall = 0.75$ and $precision = 0.89$.

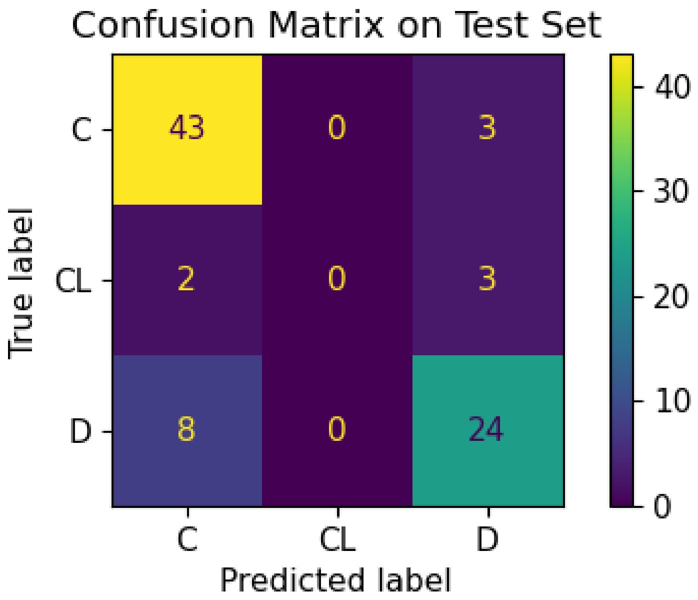


Figure 12: Unscaled confusion matrix comparing predicted and true values

This indicates the model is effective in predicting D while also avoiding false positives and negatives. However, the model never predicts CL which, despite overall imbalance, is worrisome. Future research should address this issue. Currently, the model is better suited to predicting severity overall rather than transplant specifically.

0.5 Outlook

To improve the model's performance we should collect more data. 418 points create high variance, and causes low confidence that the best model is optimal. Collecting more CL points and rebalancing by sex would increase predictability for CL and reliability for males. Additionally, using an evaluation score that weights CL predictions higher would improve performance on these points, with a possible tradeoff of overall accuracy.

Assuming increased sample size, force plots with increasingly confident expected values and feature importances would help doctors explain prognosis and uncertainty. Interpretability is key, and finely tuned estimates could help patients understand the nuances of their disease.

Finally, feature engineering could make the model more interpretable and handle correlations within features. Both Kaggle attempts clustered variables by test type. Combining metrics based on their relative correlation may improve predictive power and remove noise. Bombarding patients with technical tests can be overwhelming, so descriptive test groupings could help.

0.6 References

Data Source:
https://www.kaggle.com/datasets/joebeachcapital/cirrhosis-patient-survival-prediction

Cirrhosis Background:
https://www.mayoclinic.org/diseases-conditions/cirrhosis/symptoms-causes

https://www.ncbi.nlm.nih.gov/pmc/articles

Prior Model Attempts:
https://www.kaggle.com/code/arunklenin/ps3e26-cirrhosis-survial-prediction-multiclass

https://www.kaggle.com/code/ashishkumarak/liver-cirrhosis-survival-prediction-multiclass