Predicting Time until Kidney Transplant Complication using Machine Learning

Shounak Ray[1], Caitlin Kunchur[2], and Rhea Mitr[3]

[1,2,3]Department of Computer Science, Stanford University

1 Introduction

There is a growing gap between the number of patients who need a life-saving organ and the supply of organs available for transplantation. As of March 2022, roughly 17 people die every day on the organ transplant waiting list, with a new person added to the waiting list every 9 minutes. This problem is especially dire in the case of kidney transplants: there were 90,483 patients on the waiting list in 2021 and only 24,670 of them received a transplant[1].

This significant shortage in kidneys is exacerbated when patients receive a kidney transplant that fails shortly afterwards. Despite very prevalent transplant failures, doctors still struggle to effectively determine whether a candidate on the transplant list should receive a kidney; that is, guess the time after which a candidate would encounter a complication. If a doctor decides to offer a transplant to a patient who rejects the kidney shortly after the operation, the patient is back on the waiting list and the doctor's decision was relatively fruitless. The converse is also true, where the outcome is a successful transplant that enhanced the recipients health for an extended period. This nuanced dilemma prevents doctors from proactively and optimally deciding when to offer transplants to their patients, if at all. Virtually all patients who undergo kidney transplants will inevitably undergo some form of transplant complication (ranging in severity) given enough time, and so, it is important for doctors to predict this time until complication (TIC) so they can preemptively adjust treatment plans for patients[2].

Our pipeline relies on extensive pre-processing of Stanford Health Care's electronic health records (SHC-EHR) for kidney transplant patients (with patient demographics, medication history, labs, and diagnostic/procedure reports. We then use Multi-variate Linear Regression (MLR) and an Artificial Neural Network (ANN) to output a predicted continuous *time* until complication (continuous). We also predict the post-transplant complication acuteness, ranging from hyper-acute to chronic rejection. Physicians have stated that five main categories of acuteness are most helpful in guiding treatment plans[3], and these categories of acuteness are determined by the time interval until complication. Therefore, we then use Logistic Regression (LGR) and a semi-supervised Label Propagation model (LBP) to predict the discrete *time-interval* until complication.

2 Related Work

Compared with predicting waiting list time and chance of rejection for a donor-recipient pair, predicting precise TIC has not been as thoroughly explored. Existing studies have evaluated the predictive power of ensemble learning algorithms to predict chance of long-term graft survival in kidney transplant recipients[4] and ANNs have been used to predict exact type of rejection (hyperacute, acute, subacute, and chronic) [5]. In addition to predicting long-term transplant survival via regression, we aim to uniquely classify acuteness via the relatively unexplored semi-supervised classification method of LBP, uniquely suitable for datasets containing unlabeled patients yet to experience rejection. Decruyenaere et al.[6] compared performance of other predictive models for delayed graft function, finding that out of LDA, QDA, linear SVM, and radial SVM, only a linear SVM had a higher discriminative capacity than logistic regression. This study also optimized performance using recursive feature selection, a practice we found crucial to implement for handling noise and sparsity in our own predictive models.

Time-to-event models such as Cox regression have been implemented to predict transplant survival[7], but strong assumptions include a constant hazard ratio over time and that feature changes produce proportional hazard changes regardless of time elapsed. In the context of transplant survival, these assumptions are not robust, as there is increased hazard during the peritransplant period (up to 90 days post transplant). Pinsky et al. address this issue by integrating a Kaplan-Meier method to calculate early graft survival and then fitting a parametric survival model with a Weibull functional form for long-term survival[8]. This two-part consideration is unique from previous survival prediction models.

In addition to simply predicting TIC, we also aim to shed light on the most crucial medical features used to predict TIC and identify characteristic subgroups for which these prognostic models struggle to predict TIC.

3 Dataset and Features

We obtained five datasets concerning SHC kidney-transplant recipients: medications, procedures, labs, diagnoses, and demographics. These datasets did not follow the same schema and each was quite sparse, so significant pre-processing was conducted to produce a model-ready version. The medications, diagnoses, procedures, and labs datasets were longitudinal with multiple entries for a single patient since 1993. Only in the demographics dataset did each row correspond to a distinct patient. Given our objective to determine TIC based on all the patient's medical characteristics – not just their demographics, for example – we developed a framework to combine *all* the disconnected datasets into a form with a single row per patient (Figure 1).

For a given patient, we transformed: the medications data to show the number of days they were on a set of medications falling in common therapeutic classes, diagnoses data to show the number of each ICD-10 diagnoses they received, lab data to reflect their number of abnormally high/low test results, and procedures data to show the number of each procedure they underwent, and virtually no-change to the demographics data. Once the datasets were transformed such that each row represented the medical information of a single patient, they were inner-joined together on a shared identifier, resulting in aggregated information for 1836 patients across 189 features. Moving forward, note that, the term *complication* is defined as naturally-occuring kidney rejection (encoded by standard ICD10 codes). For deceased patients that never had a natural rejection, their complication is their death itself. During this transformation process, we only consider the medical history of patients leading up to a complication. This was because any medical history post-transplant may be biased and mis-representative of patient health that led to the complication itself. We expect the predictive models to find associations between recipient health prior to a complication and the time until they receive a complication. This additional temporal constraint further increased sparsity, so feature selection (FS) was implemented to maximize model performance. Data was standardized[1], since some models assumed that the data followed a Gaussian distribution and aided FS, as will be discussed.

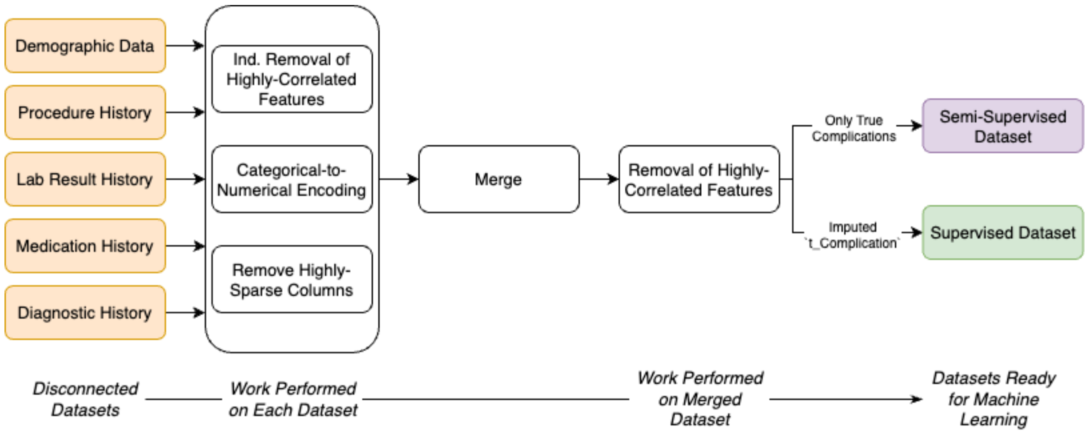

Demographic Data
Procedure History
Lab Result History
Medication History
Diagnostic History
Ind. Removal of Highly-Correlated Features
Categorical-to-Numerical Encoding
Remove Highly-Sparse Columns
Merge
Removal of Highly-Correlated Features
Only True Complications
Imputed `t_Complication`
Semi-Supervised Dataset
Supervised Dataset
Disconnected Datasets
Work Performed on Each Dataset
Work Performed on Merged Dataset
Datasets Ready for Machine Learning

Figure 1: Series of transformations applied to merge distinct patient datasets into model-ready versions. Major independent transformations on each dataset included 1) removing highly-correlated features 2) categorical-feature encoding and 3) removing very sparse columns to reduce noise. After merging the sources, we construct a dataset for upcoming supervised and semi-supervised learning. The semi-supervised dataset only has TIC for patients that had a naturally-occurring complication, including death. The supervised dataset also uses imputed TIC values for living patients that never had a natural rejection: $100 - $ Age at Transplant. That is, we assume natural death at 100, which serves as the "Age at Complication" for the patient. This is intentional overestimate with the purpose of retaining as many samples as possible. So, "Age at Complication" minus "Age at Transplant" yields TIC.

Next, to diminish the bias of structural multicollinearity on the modeling process, we constructed a correlation matrix and removed highly cross-correlated features as per a defined threshold of about $|r| = 0.7$. To further promote the utility of the dataset and lower sparsity of the merged dataset, we also removed extremely sparse columns that had too many NaN values, as per some threshold, and imputed other NaN values for some columns (such as "Smoking Hx" and "BMI").

4 Methods

We used two experimental frameworks – namely, supervised regression and semi-supervised classification – to predict the time from transplant receipt until complication and acuteness of the post-transplant complications, respectively.

4.1 Regression: Supervised Framework

For the supervised framework, using 10-fold cross validation we constructed a baseline MLR model and a more complex regression ANN to improve testing performance. We investigated train-test differences in standard error and RMSE to avoid overfitting and measured the correlation coefficient r as a measure of model fit. For initial setup of the ANN, we used the computationally efficient stochastic gradient-based optimizer Adam[9] and ReLu as our activation function. Adam was chosen over quasi-Newton method Limited-memory BFGS and pure stochastic gradient descent due to its suitability for noisy and sparse data. From there, we conducted our own quantitative analysis to determine ideal train-test split and rank hyperparameter configurations (number of hidden layers, neurons per hidden layer, and learning rate) that simultaneously maximizes r and decreased RMSE and S. We also qualitatively compared width vs depth configurations via heatmapping to more easily identify possible trends. We validated our determined configuration candidates with the existing `GridSearch`[10] algorithm, to more efficiently and exhaustively search over all hyperparameter combinations to converge on the single highest performing candidate.

4.2 Classification: Supervised and Semi-Supervised Framework

LGR was our baseline, first run on the entire supervised dataset with all 197 features. Upon further FS analyses, which will be discussed, the baseline performance was superior after 85% of the features were removed leaving only 29 in our reduced dataset – let it be $\mathcal{R}$. We then proceeded to only use $\mathcal{R}$ in our subsequent classification methods. We also experimented with L1 vs. L2 regularization and train-test split hyperparameters.

Under the classification framework, used quantiles of 20% to discretize TIC into 5 distinct time intervals representing the 5 categories of post-transplant complication acuteness. Given this, we constructed a baseline LGR model and a more

[1]Every feature i was transformed s.t. $\mu_i = 0$ and $\sigma_i = 1$

complex LBP model to predict acuteness of kidney transplant complication.

For our semi-supervised approach, we used the LBP algorithm[11] to predict time interval until complication using the labeled data (patients where we know the TIC) and unlabeled data (all other patients). LBP is a graph-based transductive method that is particularly well suited in our case since our processed dataset has significantly more unlabeled examples (1264 unlabeled examples) compared to labeled examples (572 labeled examples).

LBP works by constructing a connected graph with weighted edges between different datapoints, based on Euclidean distance. A random walk is then performed from every unlabeled datapoint to get the probability distribution of reaching a labeled datapoint. Larger weights for edges between closer datapoints will increase the probability of propagating that label. This random walk continues until the algorithm has explored all possible paths with no change in the probabilities. Estimated labels are then assigned to the originally unlabeled datapoints using the probability distributions found by the algorithm. The labels of the originally labeled points should not change significantly because those labels are more fixed. We experimented with two built-in kernel methods for this model: RBF (generates a fully-connected graph and is more memory-intensive, yet performs better) and KNN (more memory-friendly with lower run times).

5 Experiments/Results/Discussion

5.1 Baseline: Multivariate Linear Regression

For the MLR model, the primary metrics to evaluate performance on were the correlation coefficient r, coefficient of multiple determination r^2, standard error S, and RMSE. Our goal was to ensure maximum generalizability of the MLR model. We began by investigating which train-test split was optimal[2], which was a 70-30 split. As seen in the left-most column of Table 1, the high difference between RMSE_{test} and RMSE_{train} was an indicator that the seed model overfit[12] . Since our initial feature space contained 189 features across only 1836 instances, we hypothesized that the model was overfitting when learning a high number of features. Given that each feature was standardized and no two features exhibited moderate-to-high correlation[3], we sorted the absolute value of the normalized feature weights $|w_{feature}|$ determined by our initial MLR model. To test our hypothesis that a high-dimensional feature space contributed to overfitting, we determined the training and testing fit and errors as we incrementally removed unimportant features – those with the lowest $|w_{feature}|$ score. After conducting this FS analysis, we noticed that removing 85% of features maximizes fit and minimizes error. On $\mathcal{R}$, we identified that a train-test split of 60% was best, yielding significantly better results. The metrics at each "discovery stage" are shown in Table 1 below.

	Complete Dataset 70-30 Split		**Reduced Dataset $\mathcal{R}$ 85% Removed**		**Reduced Dataset $\mathcal{R}$ 60-40 Split**	
	Train	*Test*	*Train*	*Test*	*Train*	*Test*
r	0.626	**0.391**	0.562	**0.461**	0.545	**0.465**
S	0.035	0.055	0.041	0.070	0.046	0.056
RMSE	**0.613**	**0.940**	**0.661**	**0.831**	**0.702**	**0.760**
Outcome	Seed model		Removed unimportant columns		Removed unimportant columns and found best train-test split	

Table 1: Shows the evolution of performance metrics as unimportant features were removed and different train-test splits were explored. On the best configuration (right-most column), the final training r-value and error metrics are marginally worse compared to the naïvest configuration (left-most column). **However**, notice the test RMSE value in the best configuration is lower than the test RMSE value in the naïvest configuration. Furthermore, with respect to the naïve configuration, the 1) *higher test r-value* and 2) the *closer train/test RMSE values* on the best configuration indicates superior generalization.

As shown above, FS notably increased model performance and generalizability. On $\mathcal{R}$, the top features in the linear regression dataset included "Abnormally high protein creatinine levels," "Unspecified Hyperparathyroidism," "Chronic pulmonary edema", and "HLA typing procedures."

5.2 Neural Network Regression

After running on the reduced dataset, we observed optimal performance with a 60-40 split and when optimizing loss with Adam, which revealed fast convergence (within 300 iterations) and the least overfitting as measured by RMSE train-test difference. A learning rate between 0.0005 and 0.008 yielded optimal test r-value. We then evaluated performance as a function of neurons per hidden layer (over a range of 20-60) and number of hidden layers (1-5). Methods for directly selecting the ideal width and depth of a general NN are debated on, but we chose these starting ranges based on literature[13] that generally advises width to be greater than but less than twice the input layer size.

Heatmaps generated to track r, RMSE, and train-test RMSE difference (Figure 2) allowed for visual selection of candidates that maximized test r-value, minimized test RMSE, and also achieved the lowest RMSE difference. We noticed that increased depth of the network and hence complexity, increases overfitting (higher error difference). On average, increasing width per hidden layer beyond 40 neurons decreased model performance. Using these results as a starting point, we refined our search to quantitatively converge on the ideal candidate of 30 neurons for a single hidden layer. For 30 hidden neurons, r-train and r-test were 0.572 and 0.510, respectively. Test RMSE was 0.171 and the RMSE difference was 0.019. While performance stands improvement and the low model train r indicates underfitting, on all accounts, the ANN initialized

[2]Henceforth, optimality is generally defined as maximizing fit (r-value) and minimizing error (S and RMSE)
[3]This is defined as $r > 0.5$

with these hyperperameters achieved higher performance than our baseline MLR model.

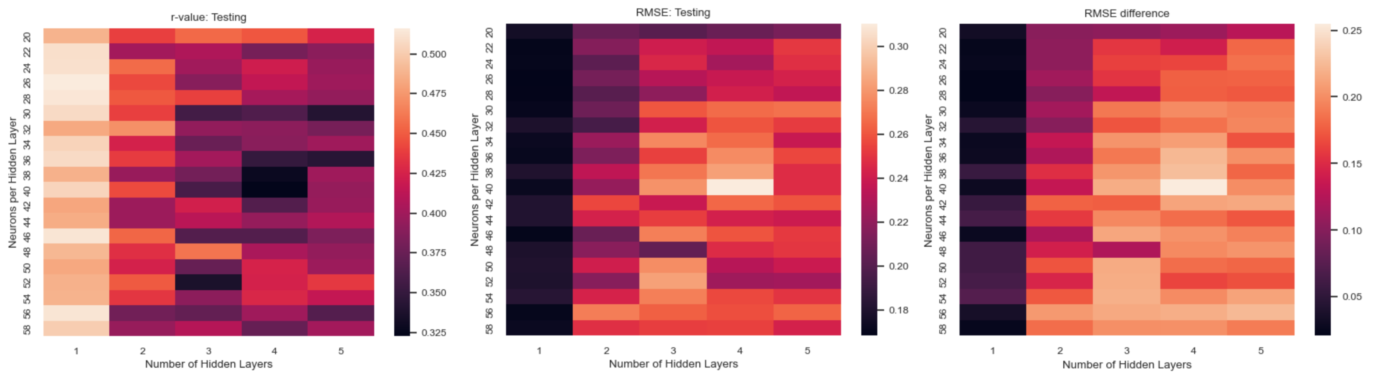


Figure 2: Qualitative heatmap analysis to identify the optimal width per hidden layer and neural network depth. Shown are the results for testing R and RMSE test-train difference.

To further validate the selection, we implemented `GridSearchCV`, which returned the Adam optimizer, a configuration of 21 neurons for a single hidden layer, and a learning rate (α) of 0.001. While the optimal α differs from our method, this chosen architecture is still ranked highly in our findings. Differences in optimal solutions may result from implementation details, such as `GridSearchCV`s allowance of refitting with the best parameters based on only a single scoring metric.

On the optimized ANN, we sorted the medians (for robustness) of the absolute value of the weights between each input and the hidden layer neurons to determine which features the ANN prioritized in predicting TIC. The ANN seemed to largely value the same features as the optimal MLR model did, but also prioritized "Ultrasound Procedures of the Abdomen and Retroperitoneum" and "Acute postprocedural pain."

5.3 Logistic Regression

For our two classification models LGR and LBP, our primary metrics for error analysis were accuracy, precision, recall, and F1-score. For both models, we found that $\mathcal{R}$ (Section 5.1) led to improved overall accuracy and better precision and recall for each class (corresponding to time interval until complication) when compared to the original dataset.

For LGR, we found that L1 regularization consistently led to better performance metrics because the shrinking of coefficients to 0 further helped reduce the number of unimportant features. We also experimented with the train-test split, and identified that a 60-40 train-test split leads to the best accuracy, precision and recall on the test data. To check for overfitting, we compared these results for the test data to the results on the training data, and found that a 60-40 train-test split also helped minimize the difference in performance metrics between the test and train data.

LGR Model with L1 Regularization and 60-40 Train-Test Split						
	Test Data			**Train Data**		
Classes	**Precision**	**Recall**	**F1-score**	**Precision**	**Recall**	**F1-score**
1	0.41	0.61	0.49	0.44	0.72	0.55
2	0.33	0.30	0.31	0.39	0.26	0.32
3	0.36	0.09	0.14	0.50	0.10	0.17
4	0.29	0.13	0.18	0.37	0.19	0.25
5	0.39	0.72	0.50	0.39	0.77	0.52
	Accuracy = 37.118%			**Accuracy = 41.108%**		

Confusion Matrix: Test Data						
		Predicted Labels				
		1	**2**	**3**	**4**	**5**
True Labels	**1**	28	10	1	4	3
	2	19	14	2	1	10
	3	12	11	4	3	15
	4	8	5	3	6	24
	5	2	3	1	7	33

Table 2: Classification report and confusion matrix for best LGR model that mitigates overfitting. Since the class labels were constructed using quantiles of 20% based on the data, the class labels 1,2,3,4, and 5 correspond to the following time intervals: 0 - 32.7 months, 32.7 - 48.9 months, 48.9 - 59.9 months, 59.9 - 69.3 months, and 69.3 - 92.5 months.

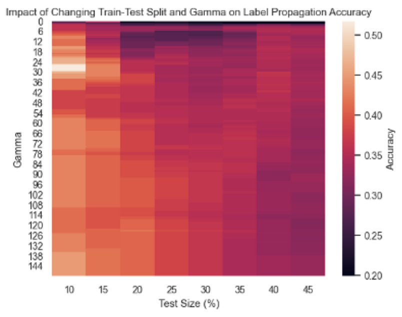
From Table 2, we see that our best LGR model had the highest precision and recall on classes 1 and 5, and performed worse for classes 2-4. The better precision for class 1 (0 - 32.7 months) and class 5 (69.3 - 92.5 months) indicates that when our model predicts a post-transplant complication as either hyperactute or very chronic, it is correct a higher percentage of the time for those extreme classes, allowing doctors to use our model's predictions as more reliable information for very acute or very chronic complications. However, in our clinical setting, recall, or the true positive rate, is a more important metric because we want to avoid situations where our model underpredicts acuteness. We see that the recall for class 1 (0.61) and class 5 (0.72) is significantly higher than for intermediate classes. We also see from our confusion matrix that for true class label of 1 or 5, our model predicts the correct class much more frequently than the other classes. A possible reason for our model's poorer performance on intermediate classes may be the hazy medical distinction between some of the intermediate gradation between acute and chronic complications [3].

5.4 Semi-Supervised Label Propagation

Experimenting with both the RBF and KNN kernel methods described earlier, RBF was ultimately chosen since it substantially and consistently outperformed the KNN kernel. Therefore, the main hyperparameter for our LBP model was γ where $\gamma > 0$, which we see here in the RBF kernel function: $K(x, x') = e^{-\gamma||x-x'||^2}$.

We adjusted γ for the RBF kernel and also the train-test split, and performed qualitative analysis using heatmaps to determine that $\gamma = 30$ and a 85-15 split led to the best model accuracy (43.023%) We also robustly verified the 85-15 train-test split across 10 different runs and test set samples.

From the heatmap in Figure 3, we see that model accuracy is improved with more intermediate γ values and when the test size for the split is smaller. Given the implementation of our LBP model, this may be because a smaller test size means more labeled training examples: this allows LBP to construct a graph with more labeled nodes and better propagate these known labels forward to nearby unlabeled nodes, improving the prediction quality.



LBP with 85-15 Train-Test Split and Gamma = 30

Classification Report

Classes	Precision	Recall	F1-score
1	0.57	0.67	0.62
2	0.50	0.47	0.48
3	0.30	0.18	0.22
4	0.27	0.35	0.31
5	0.47	0.47	0.47
	Accuracy = 43.023%		

Confusion Matrix

		Predicted Labels				
		1	2	3	4	5
True Labels	1	12	1	1	4	0
	2	3	8	2	1	3
	3	2	4	3	5	3
	4	3	2	3	6	3
	5	1	1	1	6	8

Figure 3: Left: Heatmap analysis to identify the optimal train-test split and gamma value to generate the best LBP performance (accuracy) Right: Classification report and confusion matrix for best LBP model on test data

From the table in Figure 3, we see that similar to LGR, our model has highest precision and recall for classes 1 and 5, which map to very acute and very chronic complications. However, unlike LGR, our LBP performed relatively better in terms of recall for classes 2, 3, and 4, with values of 0.47, 0.18, and 0.35, noticeably better than the corresponding recall values in our LGR model (0.30, 0.09, and 0.13, respectively). Furthermore, LBP's accuracy (43.023%) is better than LGR's accuracy of 37.118%.

6 Conclusion

Creating $\mathcal{R}$ after performing FS by analyzing feature weights on the original, standardized dataset 1) narrowed down which features the MLR model found most informative and 2) allowed us to test whether our other models also benefited from $\mathcal{R}$. Conducting FS with respect to the MLR feature weights led to an interesting outcome: patient demographics (ethnicity, smoking history, etc.), most medication frequency data, and most lab results were almost insignificant. Rather, 29 out of 197 features persisted through FS. Diagnoses relating to "endocrine, nutritional and metabolic diseases", "injury, poisoning and certain other consequences of external causes", and "Diseases of the respiratory system" seemed to be the most important categories for *all* the models, not just the baseline. On the other hand, the frequency "Ultrasound Procedures of the Abdomen and Retroperitoneum", "Hematology and Coagulation Procedures," and "Enipuncture and Transfusion Procedures" appear as most helpful procedural predictors for TIC. The consistency across the independent FS processes for the MLR and ANN models promotes model explainability. Not only does this shows that these systems are valuing the same information about a patient when predicting TIC, but is also consistent with literature[14] .

After feature selection, the optimal NN configuration showed better performance than the baseline MLR, attributable to its higher complexity and ability to capture nonlinear relationships. As discussed earlier, thirty neurons lays in the recommended range to avoid both under- and overfitting, with a single hidden layer suitable for most practical problems, as long as there is a continuous mapping from one finite space to another[13]. However, performance for the NN requires great improvement before usability can be considered. Several limitations of our current searching method must be considered. Our searching implementation assumes the same number of neurons per additional hidden layer. L2 regularization was also the built-in method for the sklearn MLP Regressor method, but L1 is more suitable for handling sparse data. Also, further feature pruning beyond the initial reduction could change performance. Explainability of the neural network beyond the predicted time until complication primarily lay in ranking feature weights after fitting.

Our LBP model performed better than our baseline LGR (accuracy and recall). This is potentially because LBP can train on a larger dataset (uses both labeled and unlabeled data to find underlying relationships between examples), while LGR was trained on only labeled data (smaller dataset). However, both classification models, although well-intended, achieved worse performance than our regression models. As described before, accurately predicting time interval until complication (acuteness) relies on clear distinction between the gradation levels between hyperacute and long-term chronic complications, which is not frequently the case. To identify patient groups with higher model misclassification, we compared the feature-by-feature distribution of misclassified patients to that of all patients and qualitatively determined notable anomalies. Our model seems to struggle to understand patient with a high number of "Venipunctures," "Most recent hemoglobin A1c," "Vitamin B-12," and "Immunofluorescence antibody stain procedure" – which have not been previously discussed in the existing literature. However, our explanation of precision/recall earlier in the discussion section highlights model interpretability, which is especially important for physicians when making transplant decisions.
Our LBP model performed better than our baseline LGR (accuracy and recall). This is potentially because LBP can train on a larger dataset (uses both labeled and unlabeled data to find underlying relationships between examples), while LGR was trained on only labeled data (smaller dataset). However, both classification models, although well-intended, achieved worse performance than our regression models. As described before, accurately predicting time interval until

complication (acuteness) relies on clear distinction between the gradation levels between hyperacute and long-term chronic complications, which is not frequently the case. To identify patient groups with higher model misclassification, we compared the feature-by-feature distribution of misclassified patients to that of all patients and qualitatively determined notable anomalies. Our model seems to struggle to understand patient with a high number of "Venipunctures," "Most recent hemoglobin A1c," "Vitamin B-12," and "Immunofluorescence antibody stain procedure" – which have not been previously discussed in the existing literature. However, our explanation of precision/recall earlier in the discussion section highlights model interpretability, which is especially important for physicians when making transplant decisions.

Future work includes tasking the neural network with classification of rejection acuteness to compare against the LBP model and testing ensemble learning methods (e.g. random forest classification, gradient boosting). A significant drawback of our research is the lack of access to kidney donor data; incorporating both patient and donor data would increase prediction accuracy and provide a more contextual suggestion for whether a transplant candidate should receive a kidney.

7 Contributions

Shounak suggested an initial experimental and analytical framework/architecture for the project, performed data pre-processing, and implemented the baseline multivariate regression model. Caitlin obtained EHR database access, performed preliminary filtering to build a kidney transplant patient cohort for pre-processing, and implemented the ANN regression model. Rhea implemented the classification baseline model of LGR and the semi-supervised LBP model. All members worked on interpretation of results, analysis, and drafting of the manuscript. All code can be found in this repository: `github.com/ShounakRay/predict-kidney-complication`.

8 References

[1] Health Resources and Services Administration. (2022, March). Organ donation statistics.
Retrieved from http://www.organdonor.gov/learn/organ-donation-statistics

[2] Salamin, Pauline et al. "Predictive Factors of Surgical Complications in the First Year Following Kidney Transplantation." Annals of vascular surgery vol. 83 (2022): 142-151. doi:10.1016/j.avsg.2021.08.031

[3] Naik, Ruchi H. and Saed H. Shawar. "Renal Transplantation Rejection." StatPearls, StatPearls Publishing, 23 May 2022.

[4] Yoo, K.D., Noh, J., Lee, H. et al. A Machine Learning Approach Using Survival Statistics to Predict Graft Survival in Kidney Transplant Recipients: A Multicenter Cohort Study. Sci Rep 7, 8904 (2017). https://doi.org/10.1038/s41598-017-08008-8

[5] Petrovsky, N., Tam, S., Brusic, V., Russ, G., Socha-Hernandez, L., Bajic, V. (2002). Use of Artificial Neural Networks in Improving Renal Transplantation Outcomes. Graft, 5, 6–13. doi:10.1177/15221620222237391

[6] Decruyenaere, A., Decruyenaere, P., Peeters, P. et al. Prediction of delayed graft function after kidney transplantation: comparison between logistic regression and machine learning methods. BMC Med Inform Decis Mak 15, 83 (2015). https://doi.org/10.1186/s12911-015-0206-y

[7] Oztekin, A., Delen, D., Kong, Z. (2009). Predicting the graft survival for heart–lung transplantation patients: An integrated data mining methodology. International Journal of Medical Informatics, 78(12), e84–e96.
doi:10.1016/j.ijmedinf.2009.04.007

[8] Pinsky BW, Lentine KL, Ercole PR, Salvalaggio PR, Burroughs TE, Schnitzler MA. (2012). Predicting long-term graft survival in adult kidney transplant recipients. Saudi J Kidney Dis Transpl, 23:693-700

[9] Kingma, D. P., Ba, J. (2014). Adam: A Method for Stochastic Optimization. doi:10.48550/ARXIV.1412.6980

[10] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., . . . Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825–2830.

[11] Zhu, X., and Ghahramani, Z. (2002). Learning from Labeled and Unlabeled Data with Label Propagation. Carnegie Mellon University.

[12] Cawley, G. C., Talbot, N. L. C. (2010). On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation. Journal of Machine Learning Research, 11(70), 2079–2107.

[13] Heaton, J. (2008). Chapter 5: Feedforward Backpropagation Neural Networks. In Introduction to neural networks for Java (2nd ed.) Heaton Research, Inc.

[14] Oweira, Hani et al. "Risk Factors of Rejection in Renal Transplant Recipients: A Narrative Review." Journal of clinical medicine vol. 11,5 1392. 3 Mar. 2022, doi:10.3390/jcm11051392