Estimating the Required Dosage of Warfarin

Fengjun Yang, Chee Ching Ong
fyang3@stanford.edu, ccong@stanford.edu
Department of Aero/Astro, Stanford University

Summary

Warfarin is a prescription drug used to treat blood clot-related symptoms. We trained models to predict warfarin dosage class for patients, based on given features of the patients. The best model was logistic regression with L1 regularization, and achieved 76.9% Accuracy, 71.3% Sensitivity and 81.2% Specificity.

Data & Features

- The dataset was obtained from the International Warfarin Pharmacogenetics Consortium [1].
- 5528 examples, 62 features

Demographic	Gender, Race, Ethnicity, Age
Background Information	Height, Weight, Indication for Warfarin Treatment, Existing Conditions (e.g. Diabetes), Ongoing Medications (e.g. Aspirin, Antibiotics), International Normalized Ratio (INR)
Genotypic	Cyp2C9 genotypes, VKORC1 genotypes

Figure 1: Feature Examples

- Dosage divided into two classes: small (dose less than 30mg/week) and large (dose greater than 30mg/week). This classification is consistent with the pre-processing done in [2]. 30mg/week is also the mean therapeutic dosage for the patients in the dataset.
- Data split randomly into training (80%) and test (20%) sets.
- 10-fold cross validation

Evaluation Metrics

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$
$$Sensitivity = \frac{TP}{TP+FN}$$
$$Specificity = \frac{TN}{TN+FP}$$

Overall performance metric P:

$$P = 0.8(Specificity) + 0.2(Sensitivity)$$

We are more concerned with overdose as it leads to excessive bleeding. Therefore, minimizing false positives (maximizing Specificity) is deemed more important than minimizing false negatives (maximizing Sensitivity)

Models

- **Linear Regression**
  - Model : $h_\theta(x) = \theta^T x$
  - Objective w/ L1 Regularization: $\sum_{i=1}^{n}(y^{(i)} - \theta^T x^{(i)}) + \lambda\|\theta\|_1,\ \lambda > 0$
  - Objective w/ L2 Regularization: $\sum_{i=1}^{n}(y^{(i)} - \theta^T x^{(i)}) + \lambda\|\theta\|_2^2,\ \lambda > 0$
- **Logistic Regression**
  - Model : $h_\theta(x) = g(\theta^T x) = \frac{1}{(1+\exp\{-\theta^T x\})}$
  - Log Likelihood $l(\theta) = \sum_{i=1}^{n} y^{(i)} log h_\theta(x^{(i)}) + (1-y^{(i)}) log(1-h_\theta(x^{(i)}))$
  - Objective w/ L1 Regularization: $J(\theta) = -\frac{1}{n} l(\theta) + \lambda\|\theta\|_1,\ \lambda > 0$
  - Objective w/ L2 Regularization: $J(\theta) = -\frac{1}{n} l(\theta) + \lambda\|\theta\|_2^2,\ \lambda > 0$
- **k-Nearest-Neighbors (KNN)**: In k-nearest-neighbors, the algorithm stores the data points in the training set. To classify a previously unseen point, the algorithm finds the k nearest training points around the queried point, and takes a simple majority vote of these k points.
- **Support Vector Machine(SVM)**
  - Optimization problem for finding the optimal margin classifier.

$$min_{w,b}\ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i$$
$$s.t.\ y^{(i)}(w^T x^{(i)} + b) \geq 1 - \xi_i,\ i = 1...m$$
$$\xi_i \geq 0$$

  - Kernel Function: We chose Radial Basis Function (RBF) for its ability to map the attributes to an infinite feature space.

$$K(x,z) = \exp\{(-\gamma\|x - x'\|^2)\},\ \gamma > 0$$

  - Hyperparameters : γ controls the influence of a single training example, while C controls the regularization strength.
- **Multi-layer Perceptrons (MLP)**
  - Activation: Relu $a(z) = \max(0, z)$
  - Classification loss:

$$l(\theta) = \sum_{i=1}^{n} y^{(i)} log h_\theta(x^{(i)}) + (1-y^{(i)}) log(1-h_\theta(x^{(i)}))$$

  - Regression loss:

$$l(\theta) = \sum_{i=1}^{n}(y^{(i)} - h_\theta(x^{(i)}))^2$$

  - Early Stopping on a randomly sampled subset containing 10% of the training set

Baseline Performance

- Ridge Regression

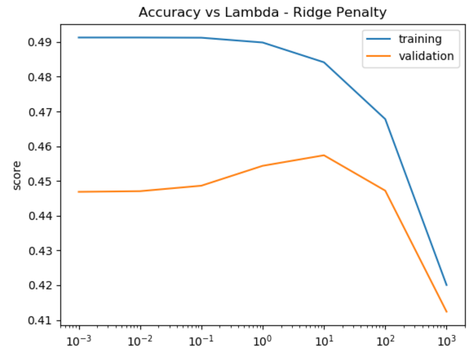

Figure 2: Ridge Regression Results

  - As we increased the penalty, the training score decreased. This is expected as the training fit would be the highest for the zero penalty case.
  - Optimal $\lambda = 10$. Optimum R^2 score of about 0.46 on the validation set.
- Logistic Regression

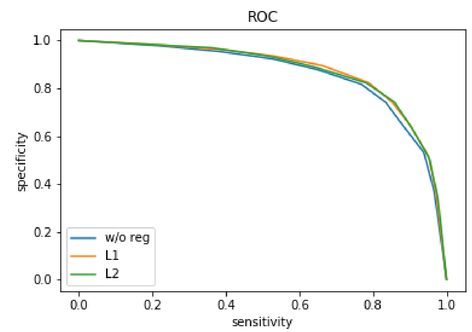

Figure 3: Regularized Logistic Regression Results

  - L1 regularization gives the highest AUC of 0.872.
  - Regularized models perform better than baseline.

References

[1] I. W. P. Consortium. Estimation of the warfarin dose with clinical and pharmacogenetic data. *New England Journal of Medicine*, 360(8):753–764, 2009.

[2] A. Sharabiani, A. Bress, E. Douzali, and H. Darabi. Revisiting warfarin dosing using machine learning techniques. *Computational and mathematical methods in medicine*, 2015, 2015.

Results

- **Model Comparison**

Model	Acc	Sensitivity	Specificity	P
Linear Reg	0.766	0.823	0.715	0.734
Ridge Reg	0.773	0.827	0.728	0.748
Lasso Reg	0.773	0.829	0.726	0.747
MLP Reg	0.786	0.797	0.779	0.783
Log Reg	0.795	0.768	0.817	0.807
Log Reg (L2)	0.805	0.781	0.823	0.815
Log Reg (L1)	0.808	0.789	0.823	0.816
KNN	0.740	0.742	0.738	0.739
SVM Clf	0.793	0.765	0.814	0.804
MLP Clf	0.793	0.752	0.825	0.810
GPC	0.789	0.753	0.817	0.804

Table 1: Comparison of Model Performance on Validation Set

- **Benchmarking Test Set Results** [2]

Model	Best Model = Log Reg (L1)	Sharabiani's
Accuracy	0.769	0.66
Sensitivity	0.713	0.63
Specificity	0.812	0.73

Table 2: Benchmarking of Model Performance.

Discussion

- Due to the large feature space, regularization helps to improve the model performance.
- On this dataset, simple models work better than complicated models. Specifically, we note that the performance of logistic regression is better than that of the multi-layer perceptron. Theoretically speaking, this should not be the case. We attribute this to the noise in the dataset leading to the neural network over-fitting the training set and thus failing to generalize.
- We also note that the performance of logistic regression is better than that of linear regression, which is not surprising as logistic regression is more robust to outliers than regression in classification task.

Future Work

- An immediate step for future work is to improve the performance of neural networks. We plan to apply methods such as dropout to better cope with the over-fitting issue.
- Another interesting direction is to run unsupervised learning algorithms on this dataset to extract useful features, and see how that can improve the regression performance.