Estimation of Warfarin Dosing Using Machine Learning

Gautham Dixit

Abstract

This paper presents a comparison of different machine learning algorithms for predicting optimal Warfarin doses for patients. Warfarin is a commonly prescribed anticoagulant, but its dosing can be challenging due to the high variability from individual to individual, and the risk of adverse events such as excessive bleeding. Traditional dosing algorithms, such as fixed dose and clinical dosing algorithms, have limitations in predicting accurate doses for individual patients. This study evaluates the performance of linear bandits and neural networks in predicting Warfarin doses using a dataset of patient characteristics and genetic information. Results show that both linear bandits and neural networks outperform traditional dosing algorithms in terms of accuracy and consistency. Moreover, further analysis reveals that linear bandits provide a good trade-off between accuracy and interpretability, while neural networks can capture complex relationships between patient features and Warfarin doses. These findings suggest that machine learning algorithms can significantly improve the precision of Warfarin dosing and have the potential to reduce the risk of adverse events in patients.

1. Introduction

Warfarin is a medication that is used as an anticoagulant agent or a blood thinner. It works by inhibiting the syntheses of vitamin K-dependant clotting factors in the liver. Reducing the production of these clotting factors helps the body prevent blood clots from forming and can reduce the risk of serious conditions such as stroke. While very beneficial for its use case, Warfarin does have some side effects and can interact with other medications in ways that can cause harm. It is imperative that the correct dosage of Warfarin be given to each patient to minimize the risk of adverse effects. Too little of a dose may not be enough for a patient to help them with blood clots, and too much of a dose can cause excessive bleeding which could be a potentially fatal outcome.

The dosage of Warfarin to administer varies heavily from patient to patient. A variety of biological factors play a role in determining the required dosage. This includes but is not limited to the patients: age, race, height, weight, current medications that they are taking, and a few genes that play a direct role in how Warfarin is metabolized in the liver. The most prominent of these genes are CYP2C9 and VKORC1. CYP2C9 is a gene that codes for an enzyme involved in the metabolism of Warfarin in the liver and VKORC1 is involved in the synthesis of vitamin K-dependent clotting factors which is what Warfarin inhibits.

The objective of this study is to evaluate the efficacy of machine learning algorithms in accurately predicting the optimal Warfarin dosage for a patient, using a set of patient-specific features. This study investigates four distinct techniques: a fixed baseline, a linear clinical dosing algorithm, a linear UCB bandit algorithm, as well as a simple neural network. This study uses a dataset that contains 5645 examples of patient features as well as their actual recommended dosage of Warfarin. After preprocessing, the dataset was cut down to 5475 examples. Using this dataset, predictions can be made as to which category of dosage should be recommended to each patient. The categories are defined as follows: if the recommended dose < 21, it is categorized as a low dose. If the dose is $>= 21$ and < 49, it is categorized as a medium dose. Finally If the dose is > 49, it is categorized as a high dose. The accuracy of the models will be assessed by examining if the model was able to predict a dosage that falls into the same category as the actual recommended dose. The metrics that will be used are cumulative regret and the percentage of incorrect predictions.

2. Related Works

There are various studies that have also investigated the use of machine learning models to predict Warfarin doses for patients. This section of the paper will briefly examine the approach and results of two of these studies and will highlight some design choices made in these papers that will be used in this study.

2.1. Estimation of Warfarin Dosage with Reinforcement Learning

The paper "Estimation of Warfarin Dosage with Reinforcement Learning" by Arpita Vats (2021) attempts to use Reinforcement learning as well as an online supervised learning

to model the proper dosage of Warfarin for patients. In particular, the paper examines the use of least squares linear regression as well as linear UCB, the latter being one of the algorithms this study will also be using.

2.1.1. Approach

The aim of this paper is to use an upper confidence bounds algorithm on the Warfarin dataset to estimate the correct category of Warfarin dose to give to each individual patient. This paper implements linear UCB which is an algorithm that uses least squares linear regression to estimate the expected reward of each action possible at a given time step and choose the action with the highest expected reward. These actions are called "arms" in the bandit setting. The paper also examines supervised learning in an online setting in order to perform regression on the correct dosages rather than on the observed rewards. These two algorithms are compared to the same baseline algorithms we will be using in this paper including the fixed dose, as well as the linear baseline.

2.1.2. Results

The paper uses two evaluation metrics to measure the performance of all of the algorithms: cumulative regret and the incorrect fraction. The results show that while linear UCB clearly outperformed the two baseline algorithms in both metrics, linear regression was still the best performing algorithm out of them all. The paper also uses reward shaping to improve the accuracy of each of the models however only marginally.

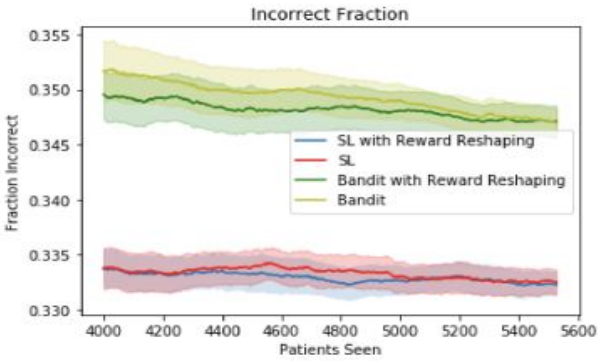

Figure 1. Incorrect fraction percentage of supervised learning algorithm vs linear bandit algorithm. (Vats,2021)

2.2. Optimizing Warfarin Dosing using Deep Reinforcement Learning

The paper "Optimizing Warfarin Dosing using Deep Reinforcement Learning" (Zadeh et al., 2022) proposes to use a deep reinforcement learning based model to predict warfarin doses. The paper uses "Pharmacokinetic/Pharmacodynamic (PK/PD) model of warfarin to simulate dose-responses of virtual patients". It formulates the warfarin dosing problem as an MDP in order to utilize Deep Q-Learning.

2.2.1. Approach

The aim of this paper is to use deep Q-learning to approximate the optimal policy in the given environment. The MDP is formulated as follows:

- State = tuple(INR reading of the current decision point, a history of INR values and dosing decisions, patient biodata)
- Action = tuple(dosage, duration)
- Reward = the euclidean distance between the predicted action (dosage,duration) and the true (dosage,duration).

The paper compares the model that uses this MDP to several baseline models.

2.2.2. Results

The results of the paper show that the Deep Q-Learning model clearly outperformed the baseline protocols they were measured against. The evaluation metric this paper chooses to use is percentage of time in therapeutic range. This is the percentage of time that a patient's International Normalized Ratio (INR) values fall within the therapeutic range, which is typically between 2.0 and 3.0. The higher the percentage of time in this range, the better the model.

protocol sensitivity	Base model	AAA	CAA	PGAA
normal	**92.4% (0.04)**	72.4% (0.18)	73.9% (0.17)	78.5% (0.13)
sensitive	**89.3% (0.11)**	43.8% (0.27)	60.3% (0.26)	68.5% (0.23)
highly sens.	**90.5% (0.08)**	14.2% (0.15)	24.9% (0.21)	59.1% (0.25)
all	**91.3% (0.08)**	60.3% (0.27)	67.3% (0.23)	74.3% (0.18)

Figure 2. Percentage of time in Therapeutic Range of the papers model vs three of the baselines used to compare against. (Zadeh et al., 2022)

3. Approach

This section will discuss the approach of this paper in predicting the accurate dose of Warfarin to give to patients. It will examine four algorithms including two baseline algorithms, the Linear UCB algorithm inspired by the paper

by Arpita Vats, as well as a simple neural network. Preprocessing techniques used to increase model performance will be discussed before diving deeper into each algorithm used.

3.1. Features

From the dataset provided, the features this paper will use for all algorithms except the neural network include: age, height (cm),weight (kg), Cyp2C9 genotypes, race, Amiodarone (Cordarone), VKORC1 genotype, and enzyme inducer status. Enzyme inducer status for each data point = 1 if the patient is using at least one of the following medications: Carbamazepine (Tegretol), Phenytoin (Dilantin), Rifampin or Rifampicin. It is 0 otherwise. These features not only draw inspiration from the features used in Arpita Vats paper but were also selected for their relatively high correlation to the recommended therapeutic dosage.

3.2. Pre-processing

Some pre-processing techniques employed in this study include one hot vector encoding, factorization, and filling/dropping null values. Normalization for the height and weight features were experimented with but ultimately did not play a significant role in the performance of the algorithms. All categorical features were first factorized into numerical values and then encoded into one-hot vectors. Weight and height were imputed with the average values of those columns respectively. Other features in which there were "NA" values present were filled with either 0 or "Unknown" depending on if the feature was numerical or categorical. The age feature was converted from a range, (40-49) for example, to a single value representing the patients age in decades, which would be 4 in the case of the previous example. Finally, the dataset was split into a training and test set for the neural network, where 80 percent of the data was used for training and 20 percent was used for validation.

3.3. Fixed Baseline

The first algorithm used in this paper is the Fixed Baseline algorithm. In this scenario, we predict 35mg/week to all patients and measure the performance. 35mg/week falls exactly in the middle of the medium dosage category ($>=$ 21 and $<$ 49).The goal of this baseline is to see how well assigning the most conservative dose of Warfarin to all patients performs compared to the other algorithms tested in this paper.

3.4. Clinical Dosing Algorithm

The second baseline algorithm this paper examines is the Warfarin Clinical Dosing algorithm. This algorithm uses a predefined set of weights that correspond to 8 features from the dataset. These features include: age in decades, height in cm, weight in kg, Asian race, black or African American, missing race, enzyme Inducer Status, and Amiodarone status. Note that in this baseline algorithm, the two genes, Cyp2C9 and VKORC1 are not used. The predefined weights allow us to construct a linear combination of the features used which gives us the square root of the predicted Warfarin dose. We can then get the squared result and then categorize the resulting value into the low, medium, high dosage categories.

3.5. LinUCB

The first non-baseline algorithm implemented is Linear UCB or the Linear Upper Confidence Bounds algorithm. This algorithm also attempts to construct a linear combination of weights and features to make predictions on Warfarin doses. LinUCB is a bandit algorithm meaning that it tries to construct a model that predicts the expected reward of each arm. An arm is defined as a particular choice or action that can be taken by the algorithm. In the case of this study, there are 3 arms representing the 3 categories we can bucket our predicted doses into. After finding the expected reward for each arm, the algorithm will choose the arm with the highest expected value.
The exact algorithm used is a slight variation of Figure 3 which is the algorithm used in the paper "Contextual Bandits with Linear Payoff Functions" by Wei Chu, Lihong Li, Lev Reyzin, and Robert E. Schapire (2011).

Algorithm 1 LinUCB: UCB with Linear Hypotheses

Inputs: $\alpha \in \mathbb{R}_+, K, d \in \mathbb{N}$
$A \leftarrow I_d$ {The d-by-d identity matirx}
$b \leftarrow \mathbf{0}_d$
for $t = 1, 2, 3, \ldots, T$ **do**
  $\theta_t \leftarrow A^{-1}b$
  Observe K features, $x_{t,1}, x_{t,2}, \cdots, x_{t,K} \in \mathbb{R}^d$
  for $a = 1, 2, \ldots, K$ **do**
    $p_{t,a} \leftarrow \theta_t^\top x_{t,a} + \alpha\sqrt{x_{t,a}^\top A^{-1} x_{t,a}}$ {Computes upper confidence bound}
  end for
  Choose action $a_t = \arg\max_a p_{t,a}$ with ties broken arbitrarily
  Observe payoff $r_t \in \{0, 1\}$
  $A \leftarrow A + x_{t,a_t} x_{t,a_t}^\top$
  $b \leftarrow b + x_{t,a_t} r_t$
end for

Linear UCB algorithm (Chu et al., 2011)

There are a few differences between the Linear UCB algorithm used in this paper and the one shown in Figure 3. The first difference is the presence of a regularization term. When θ is updated, instead of using:

$$\theta \leftarrow A^{-1}b$$

The algorithm will use:

$$\theta[a] \leftarrow (A[a] + \lambda \cdot \mathbf{I}_d + \mathbf{b}_a)^{-1} b[a]$$

Only the θ corresponding to the arm chosen at the time step will be updated. The regularization term $\lambda \cdot \mathbf{I}_d$ is added to help with overfitting by decreasing the variance.
Another difference between the version of the algorithm presented in this paper and the one presented in the paper (Chu et al., 2011), is that instead of using a fixed α, we set α to be a factor of both the number of timesteps as well as the number of times each arm has been pulled. The α term helps us control the exploration vs exploitation trade-off during training. During the prediction portion of the algorithm we set

$$\alpha = c \cdot \sqrt{\frac{\log(t+1)}{n_a}}$$

where c is some constant ($c = 50$ is used in this paper), t is the is the current time step, and n_a is the number of times arm a was pulled. Doing this allows the algorithm to more effectively explore in the early stages of training and then gradually start exploiting the learned values over time. All of this is done to have a stronger control on how much the algorithm explores vs exploits.
Linear UCB requires a reward to be observed in order to make updates. In this setting, we assign a reward of 0 if the arm chosen by algorithm matches the category the actual dosage falls into. Recommending a high dose to a patient that needs a low dose is more of a disastrous outcome than a patient that needs a high dose and is recommended a low dose. Because of this, this study introduces risk sensitivity to the problem. If the required dose is low but the model recommends a high dose, it is penalized more severely and incurs a reward of -20. If the required dose is high but the model recommends a low dose, it incurs a reward of -10. All other rewards for wrong predictions are -1.
Linear UCB has sublinear regret

$$\tilde{O}(\sqrt{KdT})$$

where K is the number of arms, d is the number of features, and T is the time steps.

3.6. Neural Network

We will now examine the use of a neural network to predict Warfarin doses on the provided dataset. A neural network is a type of machine learning algorithm that is inspired by the mechanisms of the human brain. It works by combining a large number of connected nodes which together can process and analyze complex information. One of the benefits of using a neural network is its ability to handle complex and large amounts of data and find relationships between the features and the target values that linear algorithms may not be able to find.

A neural network consists of nodes(neurons) and hidden layers. A neuron in the context of neural networks is a node which receives an input and then performs some mathematical operation on the input to get an output. This mathematical operation is usually a non-linear operation. The function used in each node to calculate the output is called the activation function. The two most common activation functions are the ReLU and the Sigmoid function. Hidden layers are what connect the outputs of one layer to the inputs of the next layer. Each layer has x amount of neurons. and each of these neurons outputs are passed on as inputs to the next hidden layer as long as there is one. When there are no hidden layers left, the final outputs are passed on to some loss function which calculates the loss of the model using the current estimates of the weights, and then makes predictions on the data.

```
self.net = nn.Sequential(
    nn.Linear(8,32),
    nn.ReLU(),
    nn.Linear(32,32),
    nn.ReLU(),
    nn.Linear(32,1))
```

Figure 3. Structure of proposed Neural Network proposed

This paper will utilize a neural network with a single hidden layer. The hidden layer will consist of 32 neurons. These neurons will map 32 inputs into 32 outputs. Finally the 32 outputs from the hidden layer are passed to the output layer where it is converted to a single output. Various other formulations of the neural network architecture were studied but no combination of hidden layers/ number of neurons seemed to affect the performance significantly. The biggest factor when it came to performance were the pre-processing of the data and which features where chosen. The activation function being used is the ReLU activation function which is essentially the hinge loss function.
The features used in this implementation are different than the ones used in the linear bandit algorithm. By reducing the number of features and focusing on the features with the highest correlation to the therapeutic dose, there was less variance and therefore less overfitting which led to higher performance. Instead of encoding the categorical features as one hot vectors, they were simply left as discretized values. This brought the complexity of the problem down. The curse of dimensionality states that as the number of features increases, the amount of data needed to learn the underlying patterns of the problem increases. Limiting the features in this case helps tackle this curse of dimensionality.

4. Results

This study will use two metrics to analyze the performance of the 4 models described previously. These metrics are regret, and the percentage of predictions that were incorrect.

4.1. Baseline algorithms

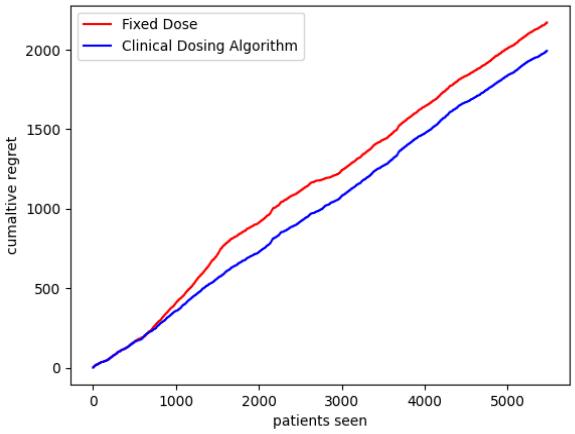

Figure 5. regret of both baseline algorithms

Figure 5 shows that both the Fixed Dose and the Clinical Dosing algorithm have linear regret. The clinical dosing algorithm slightly outperformed the Fixed Dose algorithm in terms of cumulative regret. This is reflected in the correct percentage, or the accuracy of both the algorithms. The fixed dose algorithm predicted 39.61% of examples incorrectly. The Clinical Dosing Algorithm only predicted 36.4% of examples incorrectly.

4.2. LinUCB

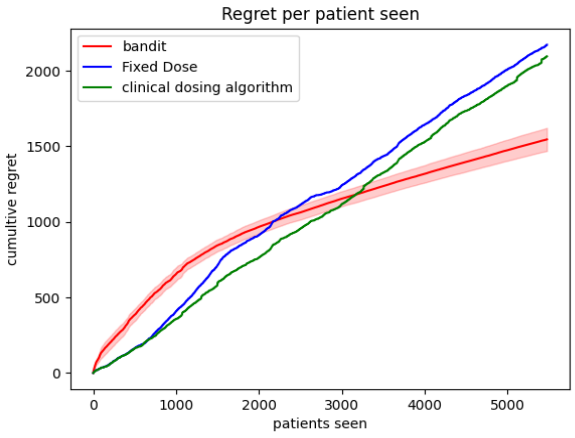

Figure 6. Cumulative regret vs patients seen

Figure 6 shows that while the Fixed Dose and the Clinical Dosing algorithm had linear regret, the Linear UCB bandit algorithm clearly has sub-linear regret. The Linear UCB algorithm was plotted with 95% confidence intervals using the T-distribution. This plot gives us an indication that the Linear UCB algorithm is performing better than the baseline algorithms. The Linear UCB algorithm was run 20 times, each having its own permutation of patients to ensure robustness of the algorithm. The cumulative regret recorded for each run was then averaged to create this plot.

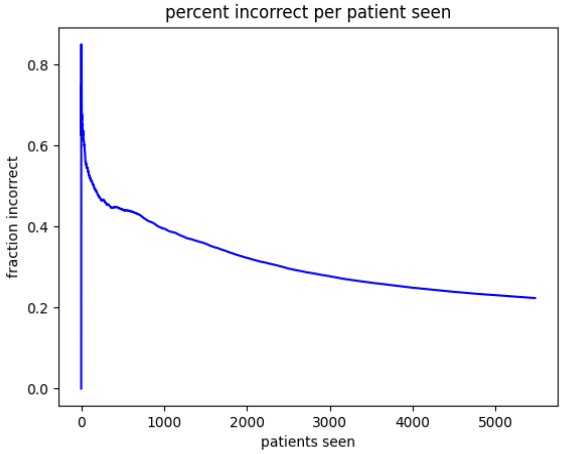

Figure 7. incorrect percentage vs patients seen

Figure 7 shows that fraction of incorrect answers with respect to the number of patients seen continuously decreases as the number of patients seen increases. After achieving sub-optimal results for the Linear UCB algorithm initially, modifying alpha with respect the time step as well as the number of times each arm had been pulled at time step t seemed to increase the performance substantially. Linear UCB was able to predict 68.58% of examples correctly on average. With the maximum percentage of the 20 trials being 77.6%. This means the average incorrect percentage for the 20 trials was 31.42%

4.3. Neural Network

Finally, the paper will examine the results of the neural network model when it comes to loss and validation set accuracy. The algorithm was trained over 3000 epochs and was able to test its performance on a validation set every 10 epochs. The single hidden layer neural network obtained an incorrect percentage of 32.33% on the test set which illustrates that it is the second best performing algorithm of the ones studied in this paper.

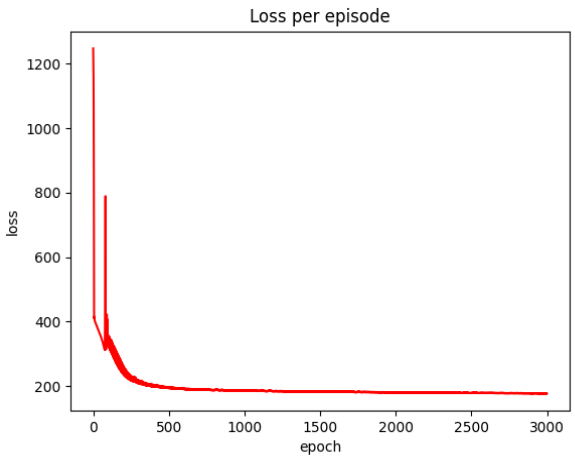

Figure 8. Loss over epochs

Figure 8 is just a sanity check showing that the loss does indeed get smaller and smaller as the model is trained. It appears that 3000 iterations was too much and the same performance can be gained by running at a lower amount of epochs.

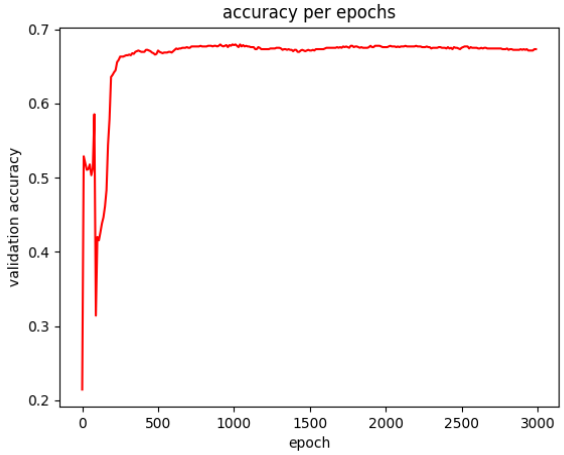

Figure 9. validation set accuracy over epochs

Figure 9 shows that around epoch 300, the network seems to reach the maximum validation accuracy of 67.67%.

Algorithm	FD	CDA	LUCB	NN
% incorrect	39.61%	36.4%	31.42%	32.33%

Table 1. Comparison of algorithm accuracy's

The table above gives a direct comparison to the accuracy of all algorithms tested in this paper. FD is the Fixed Dose algorithm, CDA is the Clinical Dosing algorithm, LUCB is linear UCB, and NN is the neural network. It is clear from the results that while the neural network performs relatively better than both of the baseline algorithms when tested on the validation set, it still does not perform better than the linear UCB algorithm with the slight modifications introduced in this paper.

5. Conclusions

After implementing and analyzing 4 distinct algorithms: Fixed Dose, Clinical Dosing Algorithm, Linear UCB, and a neural network, it is clear from the results that the Linear UCB algorithm is the best option when it comes to predicting the correct category of Warfarin dosage to recommend to patients. While the the neural network does perform better than the baseline algorithms when it comes to accuracy, the Linear Bandit algorithm achieves and even higher accuracy and sub-linear regret. Several modifications were made to Linear UCB in order to improve performance including regularization and imposing more control over the exploration vs exploitation trade-off through the use of a changing α term in the algorithm. After modifying the rewards and tuning the alpha parameter, Linear UCB yielded the best performance. Initially the neural network showed higher promise but after adding risk sensitivity to the reward function and increasing the constant that affects the alpha term significantly, Linear UCB took over as the best algorithm.

The code for this study can be found at:

https://github.com/gauthamdixit/Warfarin-Dosing

6. References

- Vats, A. (2021). Estimation of Warfarin Dosage with Reinforcement Learning. arXiv preprint arXiv:2109.07564.
- Zadeh, S. A., Street, W. N., & Thomas, B. W. (2023). Optimizing warfarin dosing using deep reinforcement learning. Journal of Biomedical Informatics, 137, 104267.
- Chu, W., Li, L., Reyzin, L., & Schapire, R.E. (2011). Contextual Bandits with Linear Payoff Functions. International Conference on Artificial Intelligence and Statistics.