Machine Learning - Homework 2 Report

Lukas Johannes Ruettgers (2023372403)

October 11, 2023

1 Objective

The experiment's objective is to examine and evaluate the performance of a logistic regression model on a multi-feature dataset. The dataset originates from https://www.kaggle.com/c/widsdatathon2020/data and contains medical information of patients at the Intensive Care Unit (ICU) of US-American hospitals. The patients x_i were classified into two groups depending on their survival s_i during their stay at the ICU.

The logistic regression model is supposed to estimate the probability of survival ($s_i = 1$) or death ($s_i = 0$) respectively for any patient x_i represented by the features consistent with the schema of the given dataset. As modeling function, logistic regression avails to the sigmoid function $\vartheta(x) = \frac{1}{1-e^{-x}}$, since its curve resembles the naturally sharp threshold at which feature values representing patients from the one group transition to feature values representing the other. Furthermore, $\vartheta(x)$ is facicely usable as a probability density function over a binary domain, since $\vartheta(-x) = \frac{1}{1+e^{-(-x)}} = \frac{1}{1+e^{x}} = \frac{e^{-x}}{1+e^{-x}} = 1 - \frac{1}{1+e^{-x}} = 1 - \vartheta(x)$, where the third equality was obtained by an extension of both the numerator and denumerator with e^{-x}. If we therefore define y_i as a random variable over the domain $\{-1, 1\}$ which obeys $y_i = \begin{cases} 1 & , s_i = 1 \\ -1 & , s_i = 0 \end{cases}$, $\vartheta(x)$ satisfies all mathematical conditions for a probability density function for y_i.

However, $\vartheta(x)$ only serves as an approximate to the unknown true probability density function $\varphi(x)$ to which y_i obeys. The logistic regression model assumes a linear contribution of each feature $f_j, 1 \leq j \leq k$ to the exponent term in $\vartheta(x)$ and consequently assigns each f_j one parameter w_j. Including a bias w_0, we therefore obtain the weight vector $w^T = \begin{pmatrix} w_0 & w_1 & \cdots & w_k \end{pmatrix}$ and the feature fector $x_i^T = \begin{pmatrix} 1 & x_{i,1} & \cdots & x_{i,k} \end{pmatrix}$, where $x_{i,1}, \ldots, x_{i,k}$ are the features of patient i. The value $\vartheta(w^T x_i)$ can therefore be regarded as the probability $P(y_i = 1 \mid x_i)$ and owing to $\vartheta(-1) = 1 - \vartheta(1)$, $\vartheta(y_i \cdot w^T x_i)$ equates the probability that s_i would have been correctly sampled from the probability distribution $\vartheta(x)$.

2 Experiment Setup

2.1 Preprocessing

The features comprised binary, numeric and integer values. While the binary values were solely converted into integers, the values of numeric and integer features were all normalization to $[0, 1]$, since their values varied largely in size and units. This was achieved by linearly rescaling each feature value x to $\frac{x - x_{min}}{x_{max} - x_{min}}$, where x_{min} and x_{max} are the maximum and minimum values of the respective feature.

Conforming to the domain of y_i, each 0 in the labels was replaced by -1.

2.2 Training

We choose to minimize the negative log likelihood of the model and hence obtain the loss

$$E(w) = \frac{1}{n} \sum_{i=1}^{N} \ln(1 + exp^{-y_i \cdot w^T x_i}),$$

which results in the gradient

$$\nabla E(w) = -\frac{1}{n} \sum_{i=1}^{N} \frac{y_i \cdot x_i}{1 + exp^{y_i \cdot w^T x_i}}.$$

The weights are updated using gradient descent with a learning rate η, where the learning rates $\eta = 0.1$ and 0.001 were probed for the experiment. The gradient descent method was programmed to terminate when either the loss function $E(w)$ undercuts an error threshold $\varepsilon = 0.01$ or a maximum amount of iterations was reached, which was chosen to be $300n$ for each cross-validation iteration and $3000n$ or $6000n$ for the normal training procedure.

Additionally, both a stochastic and a deterministic sample choice was investigated to compare their influence on the training and testing error. While the stochastic algorithm chooses the next unit x_i randomly from the sample dataset, the deterministic algorithm repeatedly iterates through the entire testing dataset.

2.3 Cross Validation

Besides training the model with the entire training dataset, the bias of the models was assessed via 10-fold cross validation (CV). The training data set was split into ten equally sized intervals and the 10-fold CV error was computed as the mean accuracy of the predictions of the trained models on their respective test intervals.

2.4 Implementation Details

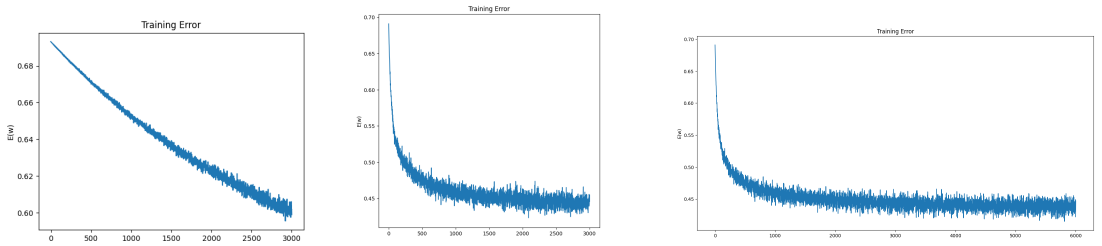
The entire code was independently written in Python 3.12.0 and only avails to four packages. Firstly, it invokes the `csv` package to process the format of the dataset, uses the log and exp functions by the `math` package, generates random integers with the `random` package and finally imports the `matplotlib.pyplot` package to plot figures.

To avoid overflows, the calculation of $\vartheta(x), E(w)$ and $\nabla E(w)$ relies on different, but mathematically equivalent formulas for different ranges of the input.

3 Results

3.1 Training

The training dataset contained 5000 samples. Regarding the learning rate η, 0.1 was found to be a reasonable value that on the one hand requires relatively little iterations until convergence and on the other hand maintains moderate oscillations in the training error curve (Fig. 1c). While $3000n$ iterations suffice for the model trained with $\eta = 0.1$ to converge, when trained with $\eta = 0.001$, it is still far from its minimum. The continuously steap slope (Fig. 1a) suggests that the gradient unambiguously points to a more suitable region in the parameter space, which however is not reached in time due to the small learning rate. Even at convergence, the models were only able to reach an accuracy of at most 80% on the training data set, suggesting that the objective function might not capture all the dynamics to which the data obeys.



(a) $\eta = 0.001, i = 3000n$ (b) $\eta = 0.1, i = 3000n$ (c) $\eta = 0.1, i = 6000n$

Figure 1: Training error curves for gradient descent with randomized sample order.

3.2 Cross Validation

The 10-fold cross validation yielded similar results for the deterministic and randomized sample choice and did just decrease slightly when switching from learning rate $\eta = 0.1$ to $\eta = 0.001$. While the best results were obtained by the randomized model with $\eta = 0.1$, whose CV error equated 0.2374 (Fig. 2), the CV error of the other models also resided around 0.25. This error is not significantly larger than the error on the training data set, which suggests that the model does not suffer from signifcant overfitting.

Iteration	TP	FP	FN	TN	Error
1	207	43	70	180	0.226
2	196	54	62	188	0.232
3	193	60	61	186	0.242
4	176	64	65	195	0.258
5	186	61	61	192	0.244
6	191	47	62	200	0.218
7	192	53	61	194	0.228
8	199	50	69	182	0.238
9	162	63	67	208	0.260
10	215	59	55	171	0.228
Total	1917	554	633	1896	0.2374

Figure 2: Cross validation results on the model with learning rate $\eta = 0.01$ and randomized sample choice. The training set was divided into 10 intervals and for each iteration, the model performed $i = 300n$ gradient descent updates for training. TP = True Positives, FN = False Negatives, FP = False Positives, TN = True Negatives.

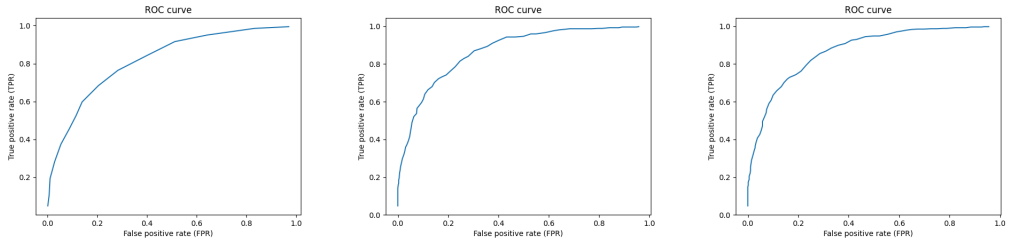
3.3 Testing

The test dataset contains 1097 sample patients, among which 547 died and 550 survived. Surprisingly, the model with deterministic sample order achieves an almost identical performance in comparison to its randomized counterpart. Both achieve an accuracy of around 0.78 (Fig. 3). Over and above, it's training error curve decreases smoothly and does not exhibit any periodic pattern or oscillations (Fig. 5a) such as the randomized model does across various learning rates. Regarding its similarity with the training error curve of the randomized counterpart model (Fig. 1b) and the high amount of gradient descent iterations, this suggests that the model does not profit from permuting the order of the feature vectors. One reason could be that feature vectors with similar values do not form clusters in the data but are spread across the dataset which avoids strong oscillations between two locally optimal gradient directions.

Model Parameters	TP	FP	FN	TN	Accuracy
RAND, $\eta = 0.1$, $i = 6000n$	**434**	127	**113**	423	**0.7812**
DET, $\eta = 0.1$, $i = 3000n$	427	124	120	426	0.7776
RAND, $\eta = 0.001$, $i = 3000n$	374	**113**	173	**437**	0.7393

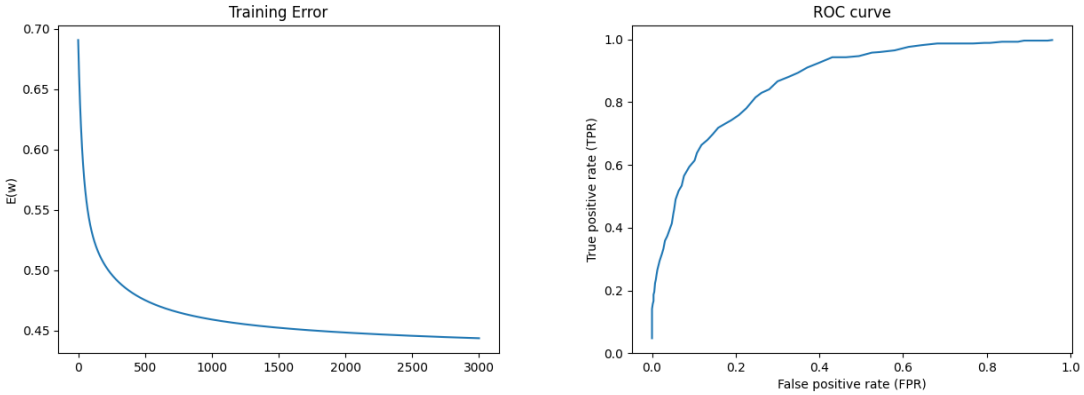
Figure 3: Test results of the investigated models. Models are categorized by whether their sampling order was randomized (RAND) or deterministic (DET), their learning rate η and the number of gradient descent iterations i.

The ROC curve of the randomized 6000-step model (Fig. 4c) indicates that the bias weight w_0 to which the model converged is reasonably balanced and achieves a sound tradeoff between positive and negative predictive values. The weaker performance of the $\eta = 0.001$ model is also reflected in its ROC curve (Fig. 4a), which is less steep and implies both worse false positive rates (FPR) and true positive rates (TPR). Again, the ROC curve of the deterministic model (Fig. 5b) and its randomized counterpart (Fig. 4b) are almost indistinguishable to the human eye.



(a) $\eta = 0.001, i = 3000n$ (b) $\eta = 0.1, i = 3000n$ (c) $\eta = 0.1, i = 6000n$

Figure 4: ROC curves for gradient descent with randomized sample order.



(a) Training error curve (b) ROC curve

Figure 5: ROC curves for gradient descent with deterministic sample order, learning rate $\eta = 0.1$ and $i = 3000$ outer training iterations. In each outer training iteration, the samples were selected from the first to the last in order of their occurence in the dataset for loss update, resulting in $3000n$ total gradient descent operations.

All in all, though the model does not achieve an outstanding accuracy on the given data in the training phase, it successfully maintains its accuracy when transitioning from the training set to the test set, which suggests that it does not suffer from strong biases in the data.