LOGISTIC CLASSIFIER FOR TISSUE MICROARRAY (TMA) ANALYSIS

Laurynas Tamulevicius

University of Glasgow

1. Introduction and motivation

There have been many attempts recently to get deeper insights into cancer structure, evolution and treatment using machine learning techniques. This report will be based on research conducted by (Beck, et al., 2011). The research is focusing on assessment of prognosis and treatment response in cancer using microscopic image analysis. The method is widely used in the world due to its feasibility. However, the existing grading techniques utilise only a few image features and most of them define epithelial anomalies whereas valuable information can be also inferred from cancer stroma as showed the paper (Beck, et al., 2011). Furthermore, the cancer grading systems vary by physician, hence patient prognosis becomes very subjective and not as sophisticated as if it was done by the machine. The paper is describing the alternative grading system, which is based on image feature extraction and their evaluation by the machine learning algorithm. Training and evaluation of the classifier is achieved using wide collection of tissue images, which are divided into smaller chunks with clear stromal or epithelial features. These pieces are called superpixels (Figure 1). In order to identify stromal and epithelial areas in the superpixels paper authors developed L_1-regularized logistic regression driven classifier and managed to achieve 89% classification accuracy on held-out data. The task is typical supervised learning problem as the target (true) data set is provided and was prepared by physicians, who evaluated the cancer tissue samples by hand.

Figure 1 Image broken into superpixels (Beck, et al., 2011)

Regarding future directions, image processing and classification is very important for online learning and interactive learning of computer vision models (Buhmann & Fuchs, 2011). The digital images of tissue samples will be available for everyone online in near future and hence efficient and trustworthy classification technique is essential. The digitized images will be also accessible to patients who could track their medical treatment and progress of disease. One of the most impressive futuristic idea is the real-time cancer detection on cellular level utilizing a fiber-optic fluorescence microscope using a consumer-grade camera (Shin, Pierce, Gillenwater, Williams, & Richards Kortum, 2010). The application would require a stable, fast and easy to use image classification framework. Such an inexpensive and portable application could be used in low-resource settings.

2. Background

2.1. Logistic regression

The logistic regression model uses the same approach as linear regression

$$f(\boldsymbol{x_{new}}, \boldsymbol{w}) = \boldsymbol{w^T x_{new}}$$

Only now the outcome function must be bounded by the sigmoid function as follows

$$h(f(\boldsymbol{x_{new}}, \boldsymbol{w})) = \frac{1}{1 + \exp(-f(\boldsymbol{x_{new}}, \boldsymbol{w}))}$$

Sigmoid function maps values to the range of [0,1]. In theory, any other smoothly increasing functions that map values to range of [0,1] can be used, but the reason we picked sigmoid is the easiness of its derivative calculation and simplistic result

$$h'(x) = h(x)(1 - h(x))$$

Which will be very convenient later when we have to optimize the cost function. (Ng, 2012)

To clarify, we use $\boldsymbol{w}$ to represent our model parameters and $\boldsymbol{x_{new}}$ is the given data we work with. Having defined our model, we want to optimize the parameters and to do that we must write down the cost function.

We will be implementing the MAP (maximum a posteriori) solution which is very similar to maximum likelihood method only MAP includes the prior effect. Let's start from the Bayesian rule

$$p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{t}) = \frac{p(\boldsymbol{t}|\boldsymbol{X}, \boldsymbol{w})p(\boldsymbol{w})}{p(\boldsymbol{t}|\boldsymbol{X})}$$

We can see that posterior (on the LHS) is directly proportional to product of likelihood and prior.

$$p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{t}) \propto p(\boldsymbol{t}|\boldsymbol{X}, \boldsymbol{w})p(\boldsymbol{w})$$

Thus, we can define function which will consider likelihood and prior effects.

$$g(\boldsymbol{w}; \boldsymbol{X}, \boldsymbol{t}, \sigma^2) = p(\boldsymbol{t}|\boldsymbol{X}, \boldsymbol{w})p(\boldsymbol{w}) \propto p(\boldsymbol{w}|\boldsymbol{X}, \boldsymbol{t}, \sigma^2)$$

Here the σ^2 terms appears among arguments because we picked prior to be Gaussian:

$$p(\boldsymbol{w}) = \prod_{d=1}^{D} N(\mu, \sigma^2)$$

In addition, we neglect the marginal likelihood in the denominator as it only normalizes the nominator. Furthermore, we need to find the likelihood function based on our model. To start with, let's define possible outcomes:

$$P(t = 1|x_{new}, w\ ) = h(y)$$
$$P(t = 0|x_{new}, w\ ) = 1 - h(y)$$

Where $y = f(x_{new}, w)$. Value 1 represent stromal element and 0 – epithelial. Note that we consider single values and not vectors in the above case. We have defined all required terms to build a likelihood function hence let's aggregate them:

$$L(\boldsymbol{w}) = \prod_{i=1}^{N} h(y_i)^{t_i} * \big(1 - h(y_i)\big)^{1-t_i}$$

Where N is number of samples and $\boldsymbol{t}$ is set of true values. To simplify maximization problem, we calculate loglikelihood instead.

$$\ell(\mathbf{w}) = \log(L(\boldsymbol{w})) = \sum_{i=1}^{N} \mathrm{t_i} * \log\big(h(y_i)\big) + (1 - t_i) * \log(1 - h(y_i))$$

To finalize the function, we need to add the log of Gaussian prior term

$$\begin{aligned} g(\boldsymbol{w}) = \ & \ell(\mathbf{w}) + \log\big(p(\boldsymbol{w})\big) \\ & = \sum_{i=1}^{N} \mathrm{t_i} * \log\big(h(y_i)\big) + (1 - t_i) * \log\big(1 - h(y_i)\big) + \sum_{i=1}^{N} \log\left(\frac{1}{\sqrt{2\pi\sigma}}\right) - \frac{(\boldsymbol{w} - \mu)^2}{2\sigma^2} \\ & = \sum_{i=1}^{N} \mathrm{t_i} * \log\big(h(y_i)\big) + (1 - t_i) * \log\big(1 - h(y_i)\big) - \frac{1}{2}\boldsymbol{w^T w} \end{aligned}$$

To simplify the function, we picked Gaussian parameters to be: $\mu = 0,\ \sigma = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$.

2.2. Stochastic gradient descent:

Unfortunately, we cannot solve $\frac{\partial g(\boldsymbol{w};\boldsymbol{X},\boldsymbol{t},\sigma^2)}{\partial \boldsymbol{w}} = \mathbf{0}$ analytically, therefore the numerical optimization will be utilized. In the implementation, I negated function to minimize rather than maximize it, hence the function can be called cost function now. To find the most optimal $\boldsymbol{w}$ values the classifier performs gradient descent algorithm which aims to find the minima of the function. The method works as follows:

1. We initialize $\boldsymbol{w}$ values to zero.
2. We update $\boldsymbol{w}$ based on the rule: $\boldsymbol{w} = \boldsymbol{w} - \alpha \frac{\partial}{\partial \boldsymbol{w}} g(\boldsymbol{w})$
3. Repeat until the change in cost (calculated using new parameters) is lower than convergence threshold.

Where α is learning rate, which defines how fast or slow the algorithm progresses towards the most optimal parameters. (Ng, 2012). The implementation has preset $\alpha = 0.001$ as analysis showed reasonable convergence time using this value.

Let's consider how the gradient of the cost function is calculated:

$$\frac{\partial}{\partial \boldsymbol{w}} g(\boldsymbol{w}) = \left(\boldsymbol{t} \frac{1}{h(\boldsymbol{y})} - (1 - \boldsymbol{t}) \frac{1}{1 - h(\boldsymbol{y})}\right) \frac{\partial}{\partial \boldsymbol{w}} h(\boldsymbol{y})$$
$$= \left(\boldsymbol{t} \frac{1}{h(\boldsymbol{y})} - (1 - \boldsymbol{t}) \frac{1}{1 - h(\boldsymbol{y})}\right) h(\boldsymbol{y})\big(1 - h(\boldsymbol{y})\big) \frac{\partial}{\partial \boldsymbol{w}} \boldsymbol{y}$$

Substituting $\boldsymbol{y} = f(\boldsymbol{x_{new}}, \boldsymbol{w}) = \boldsymbol{w^T x_{new}}$ we get:

$$\frac{\partial}{\partial \boldsymbol{w}} g(\boldsymbol{w}) = \left[t \left(1 - h\big(\boldsymbol{w^T x_{new}}\big)\right) - (1 - \boldsymbol{t}) h\big(\boldsymbol{w^T x_{new}}\big)\right] \boldsymbol{x_{new}} = \left(\boldsymbol{t} - h\big(\boldsymbol{w^T x_{new}}\big)\right) \boldsymbol{x_{new}}$$

Therefore, step update expression can be simplified to:

$$\boldsymbol{w} = \boldsymbol{w} - \alpha \boldsymbol{x_{new}} \left(\boldsymbol{t} - h\big(\boldsymbol{w^T x_{new}}\big)\right)$$

Finally, once the results converge and the most optimal $\boldsymbol{w}$ values are estimated we can simply plug them into our model. Finally, given any set of input data we can predict whether the superpixel is stromal or epithelial.

2.3. Performance evaluation metrics

2.3.1. Accuracy and 0/1 loss

Both accuracy ad 0/1 loss describe how well the model predicted the values. Accuracy shows the ratio of the number of correctly classified instances over the total number of instances:

$$\frac{1}{N} \sum_{n=1}^{N} \delta(t_n = t_n^*)$$

Where t_n is the set of true values and t_n^* is the set of predicted values. Whereas, the 0/1 loss is

$$\frac{1}{N} \sum_{n=1}^{N} \delta(t_n \neq t_n^*)$$

However, these metrics does not show the class imbalance.

2.3.2. Sensitivity and Specificity

Our model has a variable which can be in two possible states stromal or epithelial. We could define our classification outcomes using True Positive (TP - true value is epithelial and we predicted epithelial), True Negative (TN - true value is stromal and we predicted the same), False Positive (FP - true value is epithelial but we predicted stromal) and False Negative (FN - true value is stromal and we predicted epithelial). Having these definitions, we can write down ratios

$$S_e = \frac{TP}{TP + FN}$$

Which is sensitivity and defines proportion of epithelial superpixels that we classify as epithelial. The similar metric is specificity which shows the ratio of stromal superpixels that we classify as stromal.

$$S_p = \frac{TN}{TN + FP}$$

We want to maximise these ratios, though they are inversely proportional, therefore we have to consider the meaning of data in order to determine which ratio has to be maximized.

2.3.3. Confusion matrix

Is a simple matrix which sums up the TN, TP, FN and FP cases:

$$C = \begin{matrix} TP & FP \\ FN & TN \end{matrix}$$

Column normalization gives us the sensitivity and specificity.

2.3.4. AUC

It is the area on the ROC curve which shows how sensitivity and specificity changes with changing discrimination thresholds. The desire is to make $AUC = 1$.

2.3.5. Cross validation

It is a method to evaluate the performance of model by chunking the data and training the model on one part of it and testing it on the other. The cross validation is usually defined by choosing the N folds. The data is divided into N pieces, next, the model is built on N-1 chunks and evaluated on remaining one. The testing chunk is iteratively being changed by other data chunk from the training data set N times.

3. Model and Implementation

To begin with, before implementing the classifier it is very important to determine the best approach. To do that, first we tested the provided dataset with Weka software, which is suite for machine learning developed at the University of Waikato, New Zealand (University of Waikato, n.d.). The software provides Java API along with a nice GUI. The GUI was used to evaluate the performance of the possible classifiers for this problem.

The data had to be converted to specific format (.arff (Attribute-Relation File Format)), which then was uploaded to Weka pre-processing engine. The dependent variable (EpiOrStroma) value was converted from numeric type to nominal, which indicates the possible variable states. In our case, it was zero (epithelial) or one (stromal). Next, we selected the classifier and in order to determine the most optimal one for the provided data we tried a couple. The table below summarises the results:

Classifier	CV folds	ROC area	Accuracy (%)	FP	FN	TP	TN
Random Forest	10	0.968	90.2769	438	338	3529	3676
Naïve Bayes	10	0.911	85.9416	554	568	3299	3560
Logistic	10	0.967	90.4398	417	346	3521	3697

As you can see from the table we used a 10-fold cross validation which is the common practice in machine learning (Wikipedia, 2017). Furthermore, the ROC area was slightly smaller for Naïve Bayes classifier whereas Random Forest and Logistic algorithms had approximately the same area which was close to value of one (i.e. the maximum desired value). Accuracy plays a huge role here, as it defines the ratio of correctly identified instances over the total instances. We can clearly see that Logistic classifier is slightly better than Random Forest. Lastly, we need to decide whether we want to maximise the True Positive rate or False Positive rate. As we mentioned in the metrics section, the False Positive represents event when true value is epithelial but we predicted stromal. In other words, we predicted cancerous tissues to be healthy hence, we want to minimize FP counts or maximise the specificity (i.e. false positive rate).

$$S_p^{RandomForest} = \frac{TN}{TN + FP} = \frac{3676}{3676 + 438} = 0.894$$

$$S_p^{NaiveBayes} = \frac{TN}{TN + FP} = \frac{3560}{3560 + 554} = 0.865$$

$$S_p^{Logistic} = \frac{TN}{TN + FP} = \frac{3697}{3697 + 417} = 0.899$$

Therefore, it becomes obvious from the metrics above that the best choice is the logistic classifier. In addition, the paper (Beck, et al., 2011) utilized L_1regularized logistic regression which belongs to the family of logistic classifiers. According to, them: “Standard algorithms for solving convex optimization problems do not scale well enough to handle the large datasets encountered in many practical setting […] we propose an efficient algorithm for L1 regularized logistic regression”. All of that combined, we decided to proceed with the implementation of logistic classifier.

The implementation of the algorithm strictly follows the mathematical definition from the background section. The tuning parameters will be discussed in the following chapter.

4. Analysis and results

The implementation of the model works as predicted and we managed to optimise the model parameters using gradient descent algorithm. The main issue was to choose the right attributes as initial dataset provides 112 different metrics extracted from superpixels. To solve this problem, we brute-forced all the possible attribute range combinations. This means that to build a model we were using a sequence of consecutive attribute combinations starting from the 1^{st} ending at the 112^{th}. There were 6105 range combinations in total. To reduce the calculation time, we used 5-fold cross validation and increasing learning rate along with the convergence threshold (used in stochastic gradient descent algorithm to calculate numeric approximation of the derivative) to 0.1. The latter model simplifications allowed reasonable 0.52s per combination calculation time. The mean accuracy was determined by averaging all 5 values found during cross validation. Therefore, the maximum accuracy was found to be 87.18% for attributes in range of 52 to 111. Further tuning was performed on the best set of attributes by increasing cross validation fold number to 10 and decreasing learning and convergence rates to 0.001, which in theory should determine the model parameters more precisely and therefore identify more instances correctly. However, the latter tuning decreased the mean to 85.12%. The main reason for that could be overfitting, which means that the parameters got tuned only for that training dataset and therefore performed very poorly on unseen testing data.

5. Conclusion and future work

In conclusion, the determined accuracy (87.18%) is a few percent lower than the one achieved by the (Beck, et al., 2011) who got 89% accuracy using specialised L_1regularized logistic regression algorithm. The result is behind the one (90.44%) provided by Weka engine. We have achieved a specificity of 0.873 which shows a great performance regarding the false-positive ratio and is below the specificity predicted by Weka framework.

There are multiple ways to improve the existing model and increase performance score by, for instance, using a different method for parameter estimation (Tfolkman, n.d.). Just to name a few alternatives, BFGS (L-BFGS) which is function optimization technique using the quasi-Newton method of Broyden, Fletcher, Goldfarb, and Shanno (BFGS) or conjugate gradient algorithm based on Ribiere and Polak (Nocedal & Wright, 1999). Among the advantages, we can find learning-rate -free methods, the efficiency in terms of time and computing memory. Unfortunately, these methods are more complex than simple stochastic gradient descent.

The combinations of single attributes could be tested. This would require to explore the whole combination set and might be time consuming, though this could provide the most optimal set of parameters.

References

Beck, A. H., Sangoi, A. R., Leung, S., Marinelli, R. J., Nielsen, T. O., Vijver, M. J., . . . Koller, D. (2011). Systematic Analysis of Breast Cancer Morphology. *Science Translational Medicine*.

Buhmann, J. M., & Fuchs, T. J. (2011). Computational Pathology: Challenges and Promises for Tissue Analysis. *Computerized Medical Imaging and Graphics*, 515-530.

Ng, A. (2012). *Data Science Repo.* Retrieved from https://datajobs.com/: https://datajobs.com/data-science-repo/Generalized-Linear-Models-[Andrew-Ng].pdf

Nocedal, J., & Wright, S. J. (1999). *Numerical Optimization.* Springer.

Shin, D., Pierce, M. C., Gillenwater, A. M., Williams, M. D., & Richards Kortum, R. R. (2010). A Fiber-Optic Fluorescence Microscope Using a Consumer-Grade Digital Camera for In Vivo Cellular Imaging. *PLoS ONE*.

Tfolkman. (n.d.). *Logistic Regression and Gradient Descent*. Retrieved from http://nbviewer.jupyter.org: http://nbviewer.jupyter.org/github/tfolkman/learningwithdata/blob/master/Logistic%20Gradient%20Descent.ipynb

University of Waikato. (n.d.). *Weka 3: Data Mining Software in Java*. Retrieved from http://www.cs.waikato.ac.nz: http://www.cs.waikato.ac.nz/ml/weka/

Wikipedia. (2017). *Cross-validation*. Retrieved from https://en.wikipedia.org/: https://en.wikipedia.org/wiki/Cross-validation_(statistics)