Machine Learning: Autumn Semester 2015
Instructor: Prof. Joachim M. Buhmann

Course Project 3

Cell Nucleus Classification and Feature Engineering for Renal Cell Carcinoma (RCC)

1 Introduction

Cancer tissue analysis by hand is currently labour intensive and non-robust, it consists of several consecutive estimation and classification steps, and classification is one of the main steps that can be automated by machine learning techniques.

Renal cell carcinoma (RCC) is a wide-spread cancer, so the classification of RCC cell nucleus is important. What is the general procedure of the cell nucleus classification of RCC? First of all, one needs the Tissue Micro Arrays (TMA) as an important preparation technique for the quantitative analysis of RCC. The TMA glass plates carry small round tissue spots of prospective cancerous tissue samples with a thickness of one cell layer for each spot. One TMA image contains many nuclei, each of the nucleus was segmented using graphcut (the segmentation data is already provided).

To get the training label (whether one nucleus is malignant or benign), TMA images were independently labeled by two pathologists. From 1633 extracted patches of size 78x78 pixels centered at labeled nuclei, the labels of 1272 from the two pathologists agree with each other. We use these 1272 well-labeled nuclei (890 benign and 382 malignant) as our training data.

What we have now is 1272 labelled nuclei image patches (78x78), the segmentation data and shape data, we already know the labelling from pathologists is unreliable, we want to automate this labelling procedure as image classification task. One extremely important step for this classification task is the feature extraction, usually the visual features are better than the raw pixel feature. For each nucleus, various features can be extracted, and sometimes the classification performance can be heavily affected by the features used, so one important part of this project would be about the feature engineering.

To make the project incremental and exploratory at the same time, we set the task like this:

Task: We give the design matrix consisting of two kinds of features already extracted by us (696 features in total, will be explained in Section 2), the design matrix has 70% of the data-label pairs (ratio of positive-negative kept), this design matrix will give you some feeling about designing initial classifier.

Except for the design matrix, we also release all the raw data of the 1272 nuclei. The raw data of one nucleus consists of three images (78x78): raw image patch, segmentation mask and shape. With all the raw data, you can try the feature engineering techniques to boost the accuracy: 1) Explore other features you can imagine; or 2) Apply feature importance analysis to the features, and do feature selection accordingly. You should write down the details about techniques you used in the report, and the accuracy improvement because of the techniques.

The project is hosted on Kaggle and you get invited using this link: `https://kaggle.com/join/AedQHmSGWxxWYZcX`

2 Data set description

2.1 Input

70% (890) of the data-label pairs will be given as the design matrix (the ratio of positive and negative samples kept as the same in the test dataset). The design matrix consists of two types of features (696 columns), the meaning of the columns are explained in Table 1. The last column of the design matrix is the label: "1" represents "malignant" and "2" represents "benign".

Columns	Meaning
1	nucleus ID
$2 \sim 681$	PHOG features
$682 \sim 697$	1D signature
698	label

Table 1: Features inside the design matrix.

Explanation of the features:

- PHOG: pyramid histograms of oriented gradients. Calculated over a level 3 pyramid on the gray-scale patches ([1]).
- 1D signature: Is a 1-D functional representation of a 2-D boundary. (see Section 11.2.3 in [2] for detail, this book [2] and the corresponding toolbox give good description for visual features extraction. You can find electrical version of this book online.). As feature, a 16-bin histogram of the signature is generated.

To make the project exploratory, we also give you the raw data: raw image patches, segmentation data and shape data corresponding to all the 100% (1272) training samples. One example about the raw data is in Figure 2.1.

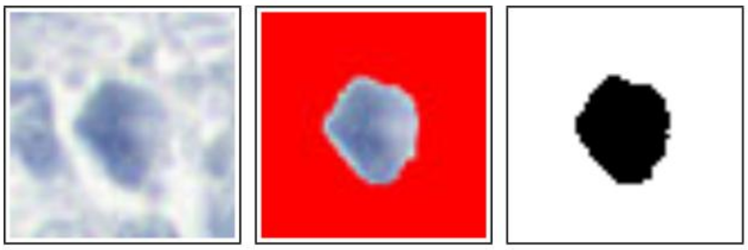

Figure 1: Left: raw image patch; Middle: segmentation mask; Right: shape

You can use these raw pixel data to explore any new features you like.

2.2 Output

You are asked to build a model that predicts the type of nuclei: "1" represents "malignant" and "2" represents "benign".

2.3 Design Matrix

This data is formatted as a comma-separated values (CSV) file in which each line corresponds to an observation. Each observation consists of 698 values: the meaning is in Table 1. The training set is in the file "train.csv". You can also regenerate it using the exemplar program inside the handout files.

2.4 Validation and Test Sets

Both validation and test set contain stages that have not been labeled yet. Your task is to predict the type for a given feature vector. You will be given samples with their respective extracted 696 feature vectors and that provide the information neccessary to perform automatic calssification and you are asked to predict the type of nuclei ("1" represents "malignant" and "2" represents "benign") for each of those samples.

The data sets are given in the file "test_validate.csv". (Often the validation and test set are provided separately. As it was for the first project, since Kaggle has only a single submission, the validation and test set are combined here.)

The formatting of the prediction file is as follows:

- Same line format as the training set ("train.csv") except that the nuclei type is not given.
- **Required output:** a file that contains the predictions. Make sure to use the same ids as in the "test_validate.csv" file and column names as shown in the "example_solution_handin.csv" file. Kaggle won't be able to score your submission if you forget the column headers!

After making the submission Kaggle will compare your solution to the ground truth and compute the error. From the submitted data you will get only feedback on half of your predictions. This feedback corresponds to the validation data set and will be listed in Kaggle's "public leaderboard". The final score and grading will be made from the other half of your prediction in a "private leaderboard" (this corresponds to the test data set). During the competiton, you will see only the results in the "public leaderboard".

To generate the submission file the following code snippet might be helpful. It shows you how to write your prediction with the necessary column headers to a file:

```
labels = [2;1]; %<<< Compute your prediction here
ids = [1;2];       %<<< Use the ids from the prediction file

% Generate a single array with the Ids and delays.
res = horzcat(ids, labels);

% Convert the array to a table and add column headers.
table = array2table(res, 'VariableNames', {'Id', 'Label'});

% Write the obtained table to a CSV file.
writetable(table, 'your_prediction.csv');
```

Final Submission: Please submit your final submission in this way:

1) Create a zip file with your code, report and TeamName.csv (your final submission) and

2) Send it to iml2015@inf.ethz.ch with subject ML- Submission Project 3 - "TeamName"

3 Evaluation and Grading

Each submission (upload of a prediction file for a given data set) will be ranked according to the Categorisation Accuracy (see "http://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html") of the predictions. Since we have the measured actual stage for each sample in the validation set and test set we can calculate the response.

We compare the cost of the submission to a baseline accuracy which is 76%. Hitting this baseline corresponds to a grade of 4. The required condition to get a 6 will be determined in combination with your report.

3.1 Report handin

In order to check the exploratory task you did, you need to provide a brief report that explains how you obtained your results, what kind of feature engineering tasks you have tried, and the corresponding improvement. One report per team is enough.

We include a template for LATEX in the file "report.tex". If you do not want to use LATEX, please use the similar sections as shown in "report.pdf". Upload a zip file with the report (as a PDF file) along with your code or parameters/screenshots of the tools you used. For further instructions refer to the report template.

We might ask you to show us what you did, so please keep the necessary files until the end of the semester.

3.2 Deadline

You will be able to submit predictions starting from **Dec. 2, 2015** until **Dec. 21, 2015, 11:59:00 PM UTC**.

4 Questions

If you have questions regarding the project, please ask during the tutorial, or contact Julian Viereck (jviereck@student.ethz.ch) and David Tedaldi (dtedaldi@student.ethz.ch) via email and use "[ML]" in your email subject. For example "[ML] Question about ...".

References

[1] Anna Bosch, Andrew Zisserman, and Xavier Munoz. Representing shape with a spatial pyramid kernel. In *Proceedings of the 6th ACM international conference on Image and video retrieval*, pages 401–408. ACM, 2007.

[2] Rafael C Gonzalez, Richard Eugene Woods, and Steven L Eddins. *Digital image processing using MATLAB*. Pearson Education India, 2004.