Classifying Head and Neck Cancer Tumour Stages Using Machine Learning Classifiers

Luuk Bolhuis (4392116), Arthur van Nieuw Amerongen (4545257)
Fenne van der Zwaard (4434978), Bart Keulen (4431413)

Link to GitHub repository: https://github.com/bjkeulen/TM10007_PROJECT

Introduction

Head and neck cancers have an annual incidence of 550.000 cases worldwide and a mortality rate of 300.000 deaths per year. The five-year overall survival of head and neck squamous cell carcinomas (HNSCC) is about 40-50%. HNSCC covers 90% of all head and neck cancers. About one-third of patients present with early-stage disease (T1-2). Differentiation between early- (T1-2) and late-stage (T3-4) cancers is crucial to give accurate treatment. Early-stage cancers have a very favourable prognosis. Surgery or radiation generate high cure rates. [1]

The tumour stages can be differentiated according to their phenotypic differences. Phenotype can be visualised using medical imaging. First, the macroscopic tumour is defined, either by a segmentation method or by an experienced radiologist. Quantitative features are extracted from the previously defined tumour region. These features involve descriptors of intensity distribution, spatial relationships between the various intensity levels, texture heterogeneity patterns, descriptors of shape and of the relations of the tumour with the surrounding tissue. [2] Machine learning can use these features to train a model which can thereafter identify HNSCC as being of stage T1-2 or T3-4 based on features extracted from a CT scan. The aim of this project is to develop a predictive model for differentiation between the two tumour stages with an accuracy of at least 70%.

Methods

A description of the dataset

The dataset contains data from 113 patients, described by 160 features. For each patient, the tumour stage is determined as early (T1-2) or late (T3-4). Image traits are extracted into features, such as convexity, compactness of the tissue, volume, CT grey level, and CT grey level variance. All of these features contain information, but not all information is relevant for differentiation between T1-2 and T3-4. Also, the dataset contains more features than samples, which will most likely lead to problematic phenomena, referred to as the curse of dimensionality. On the one hand, this means that the amount of data is too sparse for analyses that take into account all features. A classifier trained on only a few samples and many features will likely not generalize well to a test dataset. On the other hand, this means that most likely some features are not as informative as others, bringing noise into the classifier and thus reducing the performance. Concluding, a type of feature reduction should be applied in order to eliminate uninformative features and reduce the dimensionality.

Pre-processing of the data

From visual inspection, there is no missing data (NaN or None) seen in the dataset. However, code was still added to the script to remove features with missing data in case missing data has been missed. Some cells did have an extremely high or low value of the same value, from which was concluded that corrupt data was always set to a certain default value. If all default

values would be considered as corrupt data, true values could be misunderstood for corrupt data as well, leading to the removal of true data. Therefore, it is decided not to remove possible corrupt data.

From a Shapiro-Wilk test on all features separately, it was concluded that not all features are distributed normally. Therefore, robust scaling was used. Thereby, robust scaling is less sensitive to outliers. Robust scaling does not only remove the median, but it also scales the data to the quantile range. Lastly, the labels of the tumour stadia (T1-2 and T3-4) were changed into labels 0 and 1, respectively.

Cross-validations

To train the classifiers, a stratified 10-fold cross-validation with shuffle was used, splitting the data in a design set and a test set. Within the first fold of the cross-validation, the hyperparameters of several classifiers were estimated using another cross-validation, splitting the design data in a train set and a validation set. This will be discussed further in the section 'Hyperparameter estimation'. These two cross-validations will be referred to as the outer and inner cross-validation, respectively.

Feature reduction

As said, there are more features than samples. Therefore, several methods for reducing the number of features are applied to see which method or combination of methods works best on the data. The methods used were recursive feature elimination (RFE) with cross-validation, generic univariate testing (UT) using an F-test and L1 or LASSO regularization with a slack of 0.1, 0.5 and 1.0, all making use of a statistical test. Subsequently, the cumulative variance of the resulting features was calculated in the first fold of the outer cross-validation. The number of features accounting for 95% of the variance of the data was then used as the number of features to keep for a principal component analysis (PCA) for all folds. Furthermore, the number of features for UT was set to 20 and the minimum amount of features of RFE to 20 as well. The method yielding the highest accuracy was chosen.

In this way, the most predictive features for the tumour stage were chosen after which the dimensionality is further reduced using feature transformation. With the variance analysis for the PCA, the number of features was determined using a solid and clear analysis. This would have been more difficult if the methods were applied in the reversed order, since it is more difficult or not even possible to determine or set this hyperparameter for RFE, L1 or UT.

Classifiers and hyperparameter estimation

Several classifiers are trained and tested. The linear discriminant analysis (LDA), quadratic discriminant analysis (QDA), Gaussian naive Bayes, logistic regression and stochastic gradient descent (SGD) classifiers are tested without hyperparameter assessment.

The optimal hyperparameters for the data were estimated for the k-nearest neighbours (kNN), random forest, ridge regression and support vector machine (SVM) with polynomial, linear, radial basis function (RBF) and sigmoid kernel classifiers. For this estimation, a grid search was performed with stratified 5-fold cross-validation with shuffle. The hyperparameters per classifier and its evaluated parameters in the grid search can be found in Table 1. Because almost all values in a wide range are tested per parameter, a randomised grid search would not add much value. Per classifier, only the best-performing estimator was selected and used for testing on the test data. These best performing hyperparameters from the first fold were extracted and set as the hyperparameters of the classifiers for all the folds of the outer cross-validation.

Table 1. Classifiers for which the hyperparameter was estimated including the evaluated parameters in the grid search.

Classifier	Hyperparameter	Evaluated values
kNN	Number of neighbours (k)	1 to 30 in steps of 1
Random forest	Number of trees (n)	5 to 200 in steps of 5
Ridge regression	Slack^{-1} (a)	10^{-10} to 10^{-1} in 200 steps
SVM, polynomial kernel	Degree of polynomial kernel (d)	1 to 10 in steps of 1
	Slack (C)	10^{-2} to 10 in 200 steps
SVM, linear kernel	Slack (C)	10^{-2} to 10 in 200 steps
SVM, rbf kernel	Slack (C)	10^{-2} to 10 in 200 steps
SVM, sigmoid kernel	Slack (C)	10^{-2} to 10 in 200 steps

Evaluation of performance

The different classifiers were evaluated by calculation of the accuracy, area under the curve (AUC), F1-score, precision and recall. During every fold from the outer cross-validation the different performance parameters of all classifiers were calculated. Subsequently, the performances of the different folds of the outer cross-validation were concatenated in one data frame. In this data frame, the mean and standard deviation of the performance parameters were calculated per classifier. The standard deviation is used to calculate the standard error. The mean and standard error values were used to plot a bar plot of the mean accuracy and its confidence interval for all classifiers. We chose to visualize the accuracy because beforehand we determined that an accuracy of at least 70% would be sufficient for a classifier for this data. Based on this bar plot and its values, an evaluation can be made on which classifier is the best classifier.

Furthermore, a learning curve of the best scoring classifier is plotted. This is done to evaluate whether the complexity of the dataset and the classifier align. It is an extra visual check to evaluate whether the classifier suits the dataset.

Results

Feature reduction method

During the design of the script, it turned out that the number of features used when applying RFE was highly variable, which is not desirable for strong conclusions. Because of this, it was decided that RFE was not a suitable method to use for this purpose using this dataset. Therefore, no results from the method using RFE are published here.

Figure 1 shows the accuracy and its 95% CI of the UT and L1 with a slack of 0.1, 0.5 and 1.0. The accuracy of the univariate testing method was highest throughout the classifiers. Therefore, the univariate testing method will be used to identify the best classifier.

Best classifier

Table 2 shows the mean values of the calculated performance parameters of all classifiers. The optimal hyperparameters are included in the names of the classifiers. The mean accuracy of both the LDA and Ridge regression (a = 21.711) is 74.17%. Their standard errors (SE) are comparable as well, namely 3.74% and 3.45%, respectively. When looking at the other performance parameters, it can be concluded that LDA has a higher AUC than the Ridge regression (81.28% compared to 74.00%) and that all other performance parameters have exactly the same values. For this classifier, the learning curve is plotted in Figure 2.

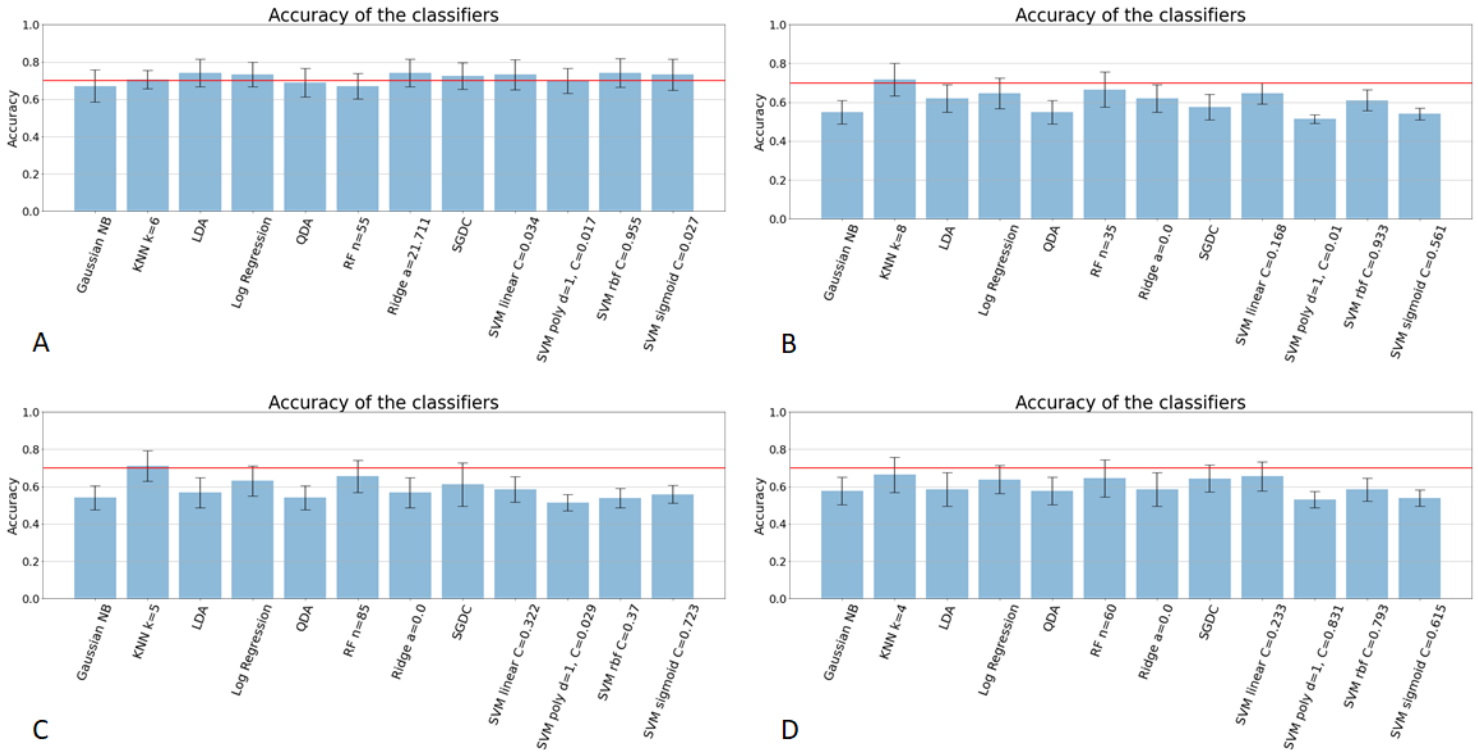

Figure 1. *The four bar plots show the mean accuracy and its confidence interval (error bars) of the univariate testing method and L1 regularisation with a slack of 0.1, 0.5 and 1.0 in figures a, b, c and d, respectively.*

Table 2. *Mean of performance parameters.*

	Accuracy	AUC	F1 score	Precision	Recall
Gaussian Naïve Bayes	0.671212	0.775000	0.627077	0.727381	0.583333
kNN k=6	0.706061	0.790833	0.663232	0.748333	0.630000
LDA	0.741667	0.812778	0.736255	0.742381	0.743333
Logistic regression	0.732576	0.824444	0.727366	0.722381	0.743333
QDA	0.689394	0.796111	0.681570	0.699762	0.676667
RF n=55	0.670455	0.759444	0.629495	0.678333	0.593333
Ridge a=21.711	0.741667	0.740000	0.736255	0.742381	0.743333
SGDC	0.725758	0.718333	0.683651	0.725833	0.680000
SVM linear C=0.034	0.732576	0.730000	0.731127	0.732857	0.743333
SVM polynomial d=1, C=0.017	0.697727	0.698333	0.596854	0.843333	0.496667
SVM rbf C=0.955	0.740909	0.736667	0.715256	0.766429	0.686667
SVM sigmoid C=0.027	0.731818	0.731667	0.711010	0.750476	0.690000

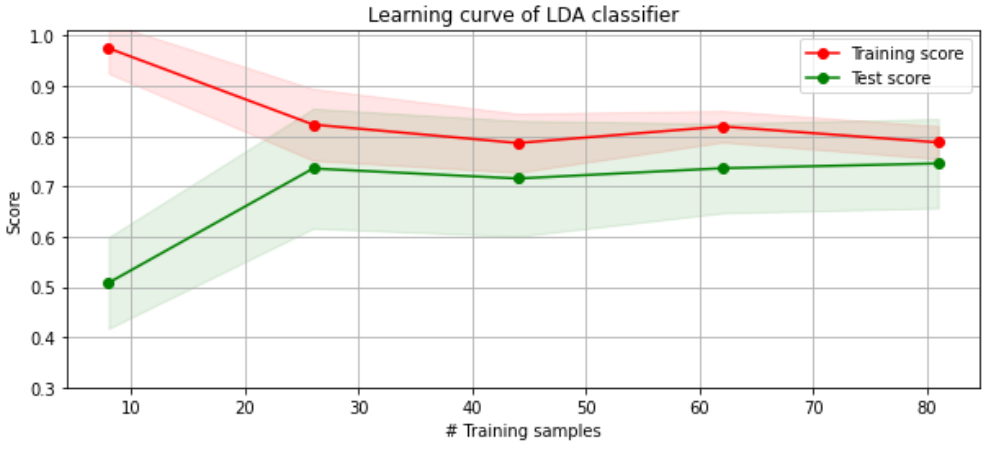

Figure 2. *Learning curve of LDA classifier*.

Discussion

In this study, multiple classifiers were trained with different feature reduction methods to determine the tumour stage (early/late) of HNSCC. When comparing the accuracy of UT and L1 regularization, UT yielded a higher mean accuracy overall. Using UT for feature reduction, LDA and Ridge regression (a=21.711) both showed the highest accuracy of 74.17% with SE of 3.74% and 3.45%, respectively. The other performance parameters were all equal as well, except the AUC which was higher with the LDA classifier. Therefore, LDA was chosen as the most optimal classifier. From the learning curve, it can be concluded that the complexity of the model aligns with the complexity of the dataset. The scores converge well, no over- or underfitting occurred.

Reliable classification of the tumour stage enables physicians to determine the optimal treatment and prognosis of the patient. As a consequence, the treatment can be adjusted as such to treat patients more personally, possibly resulting in more effective care. That said, an accuracy of 74.17% is not perfect and accuracy is not the only parameter of importance for a predictive tool. A higher accuracy will thus result in even more effective care. However, it needs to be said that no conclusions can be drawn regarding the performance of the tool compared to the golden standard (i.e. radiologists) since no comparison was made. It is recommended to implement a comparison with the golden standard in future research.

To improve the performance of the classifier, more research is needed. Most importantly, more data is needed. This classifier was tested on only 113 samples, which limits the complexity of the classifier and the amount of testing that can be done. More data will thus make it possible to increase the complexity of the classifier and therewith possibly the performance as well. Next to that, many more hyperparameters can be adjusted to create a better fit to the data, which was not possible during this study due to the limited time available.

A limitation of the method applied is that no extensive analyses were done on the complexity and shape of the data. Instead, several classifiers were applied to see which one would fit the data best. It is recommended to perform extensive analyses on the data. This way it may be possible to find classifiers with a high chance of yielding good results beforehand, which could increase the performance and save computational power as well.

References

1. Union for International Cancer Control. (2014). LOCALLY ADVANCED SQUAMOUS CARCINOMA OF THE HEAD AND NECK. Available from: https://www.who.int/selection_medicines/committees/expert/20/applications/HeadNeck.pdf. Accessed on: 07-04-2020.
2. Lambin P et al. Radiomics: extracting more information from medical images using advanced feature analysis. Eur J Cancer. 2012 Mar;48(4):441-6.

Reflection

General

Due to the circumstances and corona consequences, this project went differently from the projects we have worked on in the past. Just like the teachers, we had to be flexible and creative in finding a way to communicate, work together, work efficiently and still deliver a qualitative good final product in the end.

The lectures of this course were scheduled on Tuesdays and Thursdays. After every lecture, a meeting was planned via Microsoft Teams. In this meeting, firstly, everyone updated the other group members on the tasks they had worked on. Sometimes this was only an announcement and the task could be ticked off right away, but in other cases, this led to group discussions. In these group discussions, we discussed the problems we encountered and tried to find solutions to these problems together. If we could not come to a proper solution, together we formulated a question which would then be posted on Slack. Every meeting we also discussed the lecture of the day and assessed which topics from the lecture to implement in our project. The previous tasks, group discussions and lecture discussions always led to more tasks. These tasks were formulated and divided over the group members. At the end of the next meeting, a time and date were set for the next meeting. The time before the next meeting could then be used for everyone to work on their tasks. We believe that this planning and communication strategy worked very well, given the circumstances. Of course, we would have rather met in real life and worked on the project together around the same table at the university. But given the circumstances, we believe this is the best we could have done and everyone is very content with the way our planning and communication functioned.

As can be derived from the planning described above, the tasks were divided every meeting. Generally, the tasks were divided equally and everyone received a task to work on before the next meeting. As with every group project, group members have their strengths and their weaknesses. Naturally, group members picked the tasks related to their strengths. This way, the project went efficiently. However, this did not mean that group members only did things they were good at and did not learn anything about the more difficult subjects. The group meetings were used as a way to ask questions to each other if one would not understand a subject, a solution or reasoning. This way, we learned from each other and everyone could participate actively in all group discussions.

To conclude, we had to search to find our way in communication and planning. But all in all, we believe we have succeeded. We believe that the group members worked well together, that there was a good atmosphere and that the project process went well.

Luuk

At the start of this course, I found it hard to cope with the new situation with the coronavirus. I had to adjust my study rhythm to this new situation. Following lectures online worked fine, but engaging myself to do the exercises was a bit harder in the first week.

I am glad we had two meetings a week with the group, after the lectures and the exercises. These sessions were sometimes a bit inefficient because not everyone knew what was the best way to get things done and therefore discussions were sometimes a bit too long. The other group members encouraged me to regain a rhythm. The collaboration then worked fine.

During the course, I started focussing on the report and worked out the choices we made, although we repeatedly differed our strategy. While Arthur and Bart did most of the coding, Fenne and I kept track of the overall goal. Because I've not focussed so much on the coding, sometimes I felt I missed out on giving input in the discussion. Although, in the final part of this course I fully understood the code and the choices we've made. In the last week, we all worked on the report and I am satisfied with how it all worked out given the circumstances. In general, I am satisfied with the end results and the classifier we've built and delivered.

Arthur

Personally, I am satisfied with the results we achieved during this project. Although we had to work hard for these results, a good chemistry remained between all teammates. As mentioned before in the general reflection, the physical distance between each other did not stand in the way of working effectively together. However, I would like to add to this that at times we tended to discuss issues extensively without gaining good progress. Assigning a group leader would have helped in staying oriented to the end goals.

With regard to my own contribution, I had a major share in designing the cross-validation. I made the distinction between the outer and the inner cross-validation and estimated how small the folds could become without being left with too few samples.
Also, I implemented regularization into the design process. L1 regularization was used for feature selection but did not make it in the final result as it was inferior to multivariate testing. L2 regularization was implemented by adding the RidgeClassifier and by tuning the classifiers that offered L2 penalty. Lastly, together with Bart, I designed a strategy to find the best number of components for PCA.

I must say that along the design process, my teammates were always ready to help when encountering issues. Only a few parts were designed solely by one person, as every detail was discussed during meetings. I consider this to be a positive thing.

Fenne

Generally, I believe that the collaboration between the group members went very well. Of course, it took some time and flexibility from everyone in order to come up with a communication strategy and planning. However, we eventually did find our way and came up with a good plan.

The different group members complemented each other and everyone could put their strengths into practice and learn from the other group members. Arthur and Bart were very quick and inventive with programming and searching for new functions we could use to achieve our goals. I am glad that we had our frequent group discussions, so everyone could still catch up with the work of the other group members. I focussed on keeping track of the planning and task division. I also believe that I had quite some input on asking critical questions with regard

to our code and the application of the lecture content on our project. This led to some mistakes from being made. Some of my coding tasks were making the bar plot and structuring the code, so the code is written logically and efficient and can be understood by anyone that would have a look into it.

In the end, we worked together on the report and the finishing touch of the code. This went not very efficiently, as writing and discussing parts of a report together is not recommended to do so via video calls. I believe that we lost some time on group discussions which we could have used to spend on our individual tasks.

Bart

In general, the collaboration went smoothly and the meetings were productive, with a lot of in-depth discussions about our method in which everyone contributed equally. During some meetings, it would have been more effective to split the group though. Working with four people on one problem is not the most efficient way of working. Those moments would have been better spent on other tasks while one or two team members would focus on the problem. In the end, I believe that because of these elaborate meetings, everyone knows what we did and why we did it.

The roles were clearly defined and everyone did as agreed upon, despite the fact that no one really took up the role of leader. My role was mostly writing large parts of the code. Arthur mostly programmed as well, while Fenne and Luuk programmed less and were more focussed on the structure of the code and the report. The roles, however, might have been a bit out of balance. Although we did discuss everything we did and all code was explained to each other, I have the feeling that Fenne and Luuk might not have programmed as much as they have wanted. In hindsight, I should have brought this up and asked them.

In the last few days, we all worked on the report together and finished the code, resulting in two end products I am quite satisfied about.