University of Cambridge

L42: Machine Learning and Algorithms for Data Mining

MPhil in Advanced Computer Science

Histological classification of tumours using somatic mutation data

Author: Ioana Bica (ib354)
Word count: 2491

February 23, 2018

Histological classification of tumours using somatic mutation data

Abstract

Being able to understand the cause and development of cancer is one of the main focuses of biomedical research nowadays. Efforts in this direction have led to the creation of the COSMIC database which contains a comprehensive collection of somatic mutations from The Cancer Genome Atlas and other small studies. This project investigates whether somatic mutations alone can be used to obtain a primary histological classification for tumours. In order to achieve this, Support Vector Machine, Decision Trees, Random Forests and Neural Networks are compared on how well they perform such a classification. The results indicate that the Neural Network achieves the best performance on the histological classification of tumours. Additionally, the impact of the number of gene features used for the classification task is also explored.

1 Introduction

Somatic mutations characterize alterations in the DNA structure that can lead to diseases such as cancer. The Catalogue Of Somatic Mutations In Cancer (COSMIC database) [1] can provide us with significant information about genes that have suffered such mutations in various tumours examined by The Cancer Genome Atlas (TCGA) [2] or by small studies. More specifically, the Cosmic database makes available for use a dataset of tumour samples that are annotated by their mutated genes.

Based on this data, Amar *et al.* [3] have proved that somatic mutations alone can reliably predict cancer subtypes. However, the authors manually label each sample with the corresponding Disease Ontology terms that describe its phenotype. As it is not clear how the mapping can be done without biological knowledge, this project explores instead whether somatic mutations can predict the primary histology of tumours. This histological classification [4] is readily available form the COSMIC database.

In order to investigate whether the somatic mutations can offer enough information about the histological classification, we perform a comparative analysis on the performance of the following types of classifiers: Support Vector Machines (SVMs), Decision Trees (DTs), Random Forests (RFs), Neural Networks (NNs). The classifiers will be used for making predictions about 25 different histological types using as features the mutations encountered for the genes.

The chosen methods are powerful classification techniques that can explore non-linear relationships in the input data and that have also been successfully used in analysing somatic mutation data [5]. For instance, Le Morvan *et al.* [6] achieve state-of-the-art performance on survival prediction and on patient stratification on clinically relevant subtypes using whole-exome somatic mutation profiles and a method relying on SVMs. In addition, RF were successfully used by Marquard *et al.* [7] to identify the tissues of origin of metastatic tumours using somatic mutations. Moreover, Yuan et al. [8] worked on DeepGene, a NNs based model that identifies complex associations between various cancer types and somatic point mutations. This related work indicates that there is an increasing interest in characterising the amount of information we can obtain about tumours using the somatic mutation data.

The following sections will go through the data preprocessing performed on the COSMIC dataset (§2) and the description of the methods used (§3). Then, the experiments performed will be described and their outcomes interpreted (§4).

2 Data processing and histological classification of tumours

The somatic mutation data was downloaded from the COSMIC database. In particular, we used the data table for the genome-wide screens that include whole exome sequencing, where each row describes one mutation in a specific patient. While the data table consists of complete information about the mutations, for the purpose of this project the following columns were used in the classification task:

- Gene name
- Sample name
- Primary histology

A pre-processing step involved filtering, such that the rows that were not 'Confirmed somatic variant' [3] and that did not have the primary histology specified were removed. Additional processing involved selecting the genes that have the highest number of mutations. For most of the experiments, 100 genes were used.

This resulted in a dataset of 16162 patients, where for each patient, the input feature vector consists of the number of times each of the selected genes is mutated. To assess if the somatic mutation information is suitable for classifying the histology, we used the binary relevance method. For each different histological type, we built a dataset for binary classification where the label for each patient indicates if the primary histology of their tumour matches the histological type we want to classify. We obtained 25 such different datasets, one for each different histological type available in the COSMIC database.

3 Methods

This section gives a detailed description of the classifiers used to predict histological types. Their hyperparameters are optimised using cross-validation and grid search. However, as it is not feasible to run such an exhaustive search for all of the hyperparameter settings, the ones that have the highest impact on the overall performance were chosen.

For each histological classification, the dataset has the form:

$$\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n} \tag{1}$$

where $\mathbf{x}_i = [x_1 \ldots x_p]^T$ is the input feature vector for patient i, and $y_i \in \{0, 1\}$ is the label indicating whether the input $\mathbf{x}_i$ is part of that histological classification or not.

3.1 Support Vector Machines

An SVM learns a maximum-margin hyperplane that separates the data points into the two classes, such that the margin between the hyperplane and the closest data point is maximized [9]. However, it is not always the case that the points are linearly separable. Therefore, the SVM maximizes a soft margin and has a hyperparameter C that regulates the total amount by which individual data points can be on the wrong side of the margin or of the hyperplane. Optimising the hyperparameter C represents a bias-variance trade-off; C will be optimised in the range $C \in 2^{[-5,10]}$.

To perform non-linear classification, the kernel trick needs to be used with the SVM, such that the input is mapped into a high dimensional feature space. This way, the SVM can learn the separating hyperplane in this high dimensional feature space. We will consider two possible choices of kernel functions $K(\mathbf{x_i}, \mathbf{x_j})$ which compute the similarity of data points as follows:

- Linear kernel:
$$K(\mathbf{x_i}, \mathbf{x_j}) = \langle \mathbf{x_i}, \mathbf{x_j} \rangle \tag{2}$$

- Radial Basis Function kernel:
$$K(\mathbf{x_i}, \mathbf{x_j}) = \exp(-\gamma \|\mathbf{x_i} - \mathbf{x_j}\|^2) \tag{3}$$
where γ is a hyper parameter which we will optimise in the range $\gamma \in 2^{[-15,5]}$.

3.2 Decision Trees

DTs are binary trees that subdivide the input feature space into non-overlapping regions and perform classification by learning decision rules, associated with each node in the tree, from the data features [10].

The algorithm for building a decision tree follows a top-down approach. This way, the feature space is recursively partitioned to group the data points with the same label together. At each node m, consisting of the partition of the training data $\mathcal{D}_m$, the algorithm optimises the split parameter $\theta = (x_j, t_j)$ specifying an input feature x_j and threshold for that feature t_j. θ is selected to minimise the impurity of the resulting child nodes consisting of partitions of the data obtained by splitting $\mathcal{D}_m$ such that the data points for which $x_j < t_j$ are associated with the left child, while the ones for which $x_j > t_j$ are associated with the right child.

For node q, let R_q be the region in the feature space that is specifies and N_q be the number of data points in this region, then $p_{qk} = \frac{1}{N_m} \sum_{\mathbf{x_i} \in R_m} \mathbb{I}(y_i = k)$ represents the proportion of data points in class k in region R_q, where $k \in \{0, 1\}$. $\mathcal{D}_q$ is the partition of the training data in node q.

We will optimise through cross-validation the impurity measure from the following options:

- Gini:
$$H(\mathcal{D}_q) = \sum_k p_{mk}(1 - p_{mk}) \tag{4}$$

- Cross-entropy:
$$H(\mathcal{D}_q) = \sum_k p_{mk} \log(p_{mk}) \tag{5}$$

3.3 Random Forests

RFs are an ensemble learning method that builds multiple DTs during training and, in order to perform classification, they return the average of the outputs computed by the decision trees [11]. RFs are trained using a technique called bagging that learns a pre-determined number of trees with specific properties. By bagging repeatedly, the training algorithm randomly samples a subset $\mathcal{D}_b$ of $\mathcal{D}$ and fits a DT on $\mathcal{D}_b$, where for each node split only a subset of the input features are used.

The number of trees used is optimised in the range $t \in [10, 100]$ and we also optimise the impurity measure from Gini or cross-entropy.

3.4 Neural Networks

An NN defines some function $f(\mathbf{x}; \mathbf{w}, \mathbf{b})$, dependent on its architecture, where $\mathbf{x}$ is the input vector and $\mathbf{w}$ and $\mathbf{b}$ are the weights and biases in the network respectively. We will use a multi-layer perceptron, a NN architecture where the artificial neurons are arranged in fully connected layers, as illustrated in Figure 1. The input layer consists of input features, while the neurons in the output layer represent the predictions for the possible classes. Each neuron in a hidden layer computes a weighted sum of the outputs from the previous layer, adds a bias, applies an activation function and outputs the result which will represent the input to the neurons in the next layer. The architecture chosen for this project consists of three fully connected hidden layers with the following sequence in the number of neurons in each layer $256 \rightarrow 512 \rightarrow 256$.

The ReLU activation function [12] for the neurons in the fully connected layers is used and the logistic function is applied to the output layer. He initialization [13] is used for the weights,

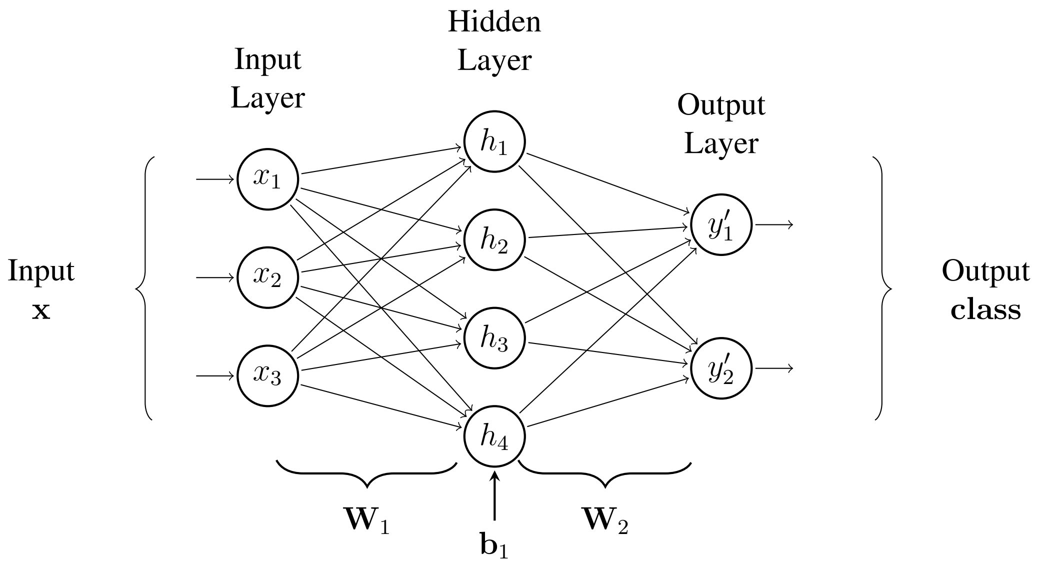


Figure 1: Structure and data flow of a multilayer perceptron with a single hidden layer. The weight matrices $\mathbf{W}_1$ and $\mathbf{W}_2$ are associated with the connections between the layers, while the vector $\mathbf{b}_1$ contains the bias of the neurons in the hidden layer.

while the biases are initialised to zeroes. The Adam Stochastic Gradient Descent optimizer [14] with a batch size of 256 is applied to optimise the weights $\mathbf{w}$ and biases $\mathbf{b}$ such that the binary cross-entropy loss function is minimised.

Regularization is also needed to avoid over-fitting. Batch normalisation addresses the problem of internal covariate shift by normalising the input values to every layer across each mini-batch [15]. Additionally, dropout [16] is a regularisation technique applied during training and involves randomly removing neurons from the network. The reason for using dropout is to ensure that the network does not rely on a small number of neurons. The dropout probability p is selected in the range $p \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$.

4 Evaluation

4.1 Experimental set-up

The project was developed in Python using scikit-learn [17] to implement the SVM, DT and RF and to perform grid search for the hyperparameters. The NN was implemented using the Keras API of TensorFlow [18].

One challenge in assessing the classifiers' performance is given by the class imbalance. Stratified nested k-fold cross-validation was used to determine the best hyperparameters for the models (inner cross-validation) and to evaluate their performance (outer cross-validation). The inner cross-validation consisted of a grid search over the parameters specified for each classifier in section §3. Stratified cross-validation ensures that the folds maintain the proportion of the two classes.

4.2 Metrics

Moreover, due to the class imbalance, the accuracy alone is insufficient in assessing the performance. Additional evaluation metrics were chosen to obtain a better indication of the discriminative ability of the models. For a given binary classifier, the performance metrics are computed using the number of true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN).

The metrics used to evaluate the classifiers are:

- **Accuracy**: proportion of examples correctly classified

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \tag{6}$$

- **Precision**: proportion of actual positive examples among all positive examples predicted

$$\text{Precision} = \frac{TP}{TP + FP} \tag{7}$$

- **Sensitivity**: proportion of positive examples correctly classified

$$\text{Sensitivity} = \frac{TP}{TP + FN} \tag{8}$$

- **F1-score**: harmonic mean of precision and sensitivity

$$F1 = \frac{2TP}{2TP + FP + FN} \tag{9}$$

- **ROC-AUC**: area under the receiver operating characteristics (ROC) curve (ROC-AUC). A ROC curve plots the true positive rate, the sensitivity, against the false positive rate at different threshold values. The ROC-AUC measures the probability that a positive example will receive a higher confidence in its positivity than a negative example.

4.3 Well-classified histological types

The ROC-AUC was used as an initial metric to identify how many histological types were well-classified based on the somatic mutation data. This metric was also used by Amar *et al.* in evaluating the classification of Disease Ontology terms. For each classifier, we computed the number of histological types for which the ROC-AUC > 0.7 and the results are highlighted in Figure 2. It is noticeable that the NN performs the best, by being able to obtain ROC-AUC > 0.7 for 20 histological type. The other classifiers achieve this performance on a significantly smaller number of histological type.

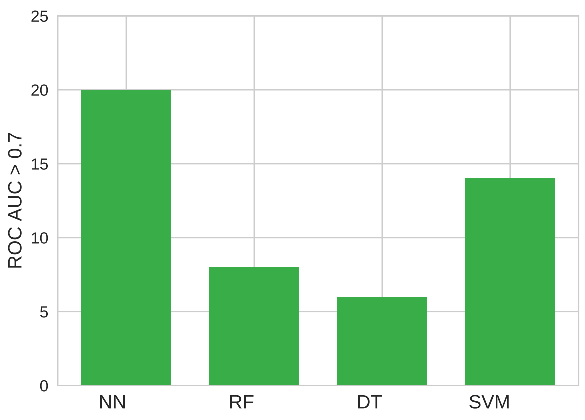


Figure 2: Number of histological types for which each classifier achieves ROC-AUC > 0.7.

To understand whether there were important differences between classifiers in the prediction of each histological type we plotted a heatmap of the ROC-AUC metric for each one of them in Figure 3. Similar colours indicate similar values for the ROC-AUC, and it can be observed that the NN, RF, DT give comparable scores for the different histological type, while the SVM behaves differently for the 10th, 14th and 21st histological types, which correspond to cranio-pharyngioma, haematopoietic neoplasm and pancreatic intraepithelial neoplasia respectively.

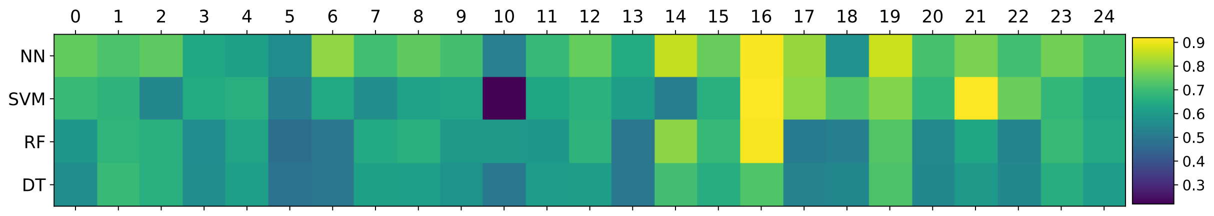


Figure 3: Heatmap of the ROC-AUC scores obtained for each classifier for the histological types. The different histological types are identified by their id at the top of the figure. A lighter colour indicates a higher value for the ROC-AUC.

Although the ROC-AUC score seems to indicate that 20 histological types can be well classified by the NN, we considered it necessary to measure performance with a different metric as well. Hence, for each classifier, we computed the number of histological types for which the precision > 0.7. Figure 4 shows that the results in this situation are considerably different. The NN achieves precision > 0.7 for only 8 histological types. However, it is interesting to notice that for this metric, the RF achieves the second best classification compared to the previous situation when the SVM ranked second.

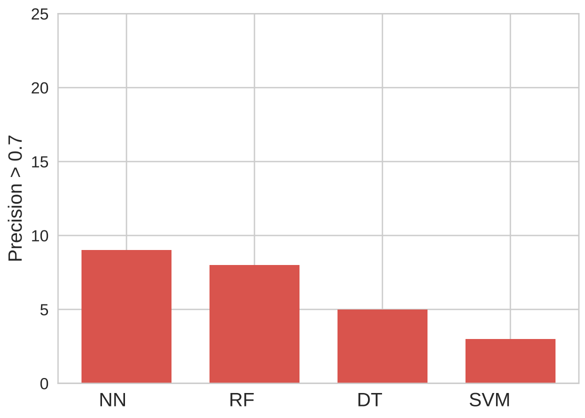


Figure 4: Number of histological types for which each classifier achieves precision > 0.7.

The results indicate that the ROC-AUC measure alone may be misleading in quantifying how well the histological types are classified.

4.4 Results on carcinoma and malignant melanoma

We now explore the results of all evaluation metrics on two individual histological types: carcinoma and malignant melanoma. These types of cancer were selected because their datasets have different properties: the dataset for carcinoma consists of 4262 negative examples and 119000 positive ones, while the dataset for malignant melanoma has 15421 negative examples and 741 positive ones. The results are illustrated in Tables 1 and 2.

Metric	DT	RF	SVM	NN
Accuracy	0.65 ± 0.01	0.70 ± 0.01	0.73 ± 0.01	**0.75** ± 0.01
Precision	0.77 ± 0.03	0.77 ± 0.03	0.74 ± 0.03	**0.78** ± 0.02
Sensitivity	0.77 ± 0.04	0.86 ± 0.04	**0.98** ± 0.03	0.92 ± 0.01
F1 - score	0.77 ± 0.03	0.81 ± 0.04	0.84 ± 0.04	**0.85** ± 0.02
ROC AUC	0.62 ± 0.01	0.67 ± 0.01	0.62 ± 0.02	**0.75** ± 0.01

Table 1: Mean results obtained for each model after 10-fold outer cross-validation for the carcinoma and the standard error in the results.

Metric	DT	RF	SVM	NN
Accuracy	0.94 ± 0.01	**0.97** ± 0.01	**0.97** ± 0.01	**0.97** ± 0.01
Precision	0.43 ± 0.04	0.75 ± 0.03	**0.85** ± 0.03	**0.85** ± 0.02
Sensitivity	0.40 ± 0.04	0.36 ± 0.04	0.40 ± 0.06	**0.46** ± 0.05
F1 - score	0.41 ± 0.03	0.49 ± 0.04	0.52 ± 0.05	**0.60** ± 0.02
ROC AUC	0.73 ± 0.01	0.91 ± 0.01	**0.92** ± 0.02	0.91 ± 0.02

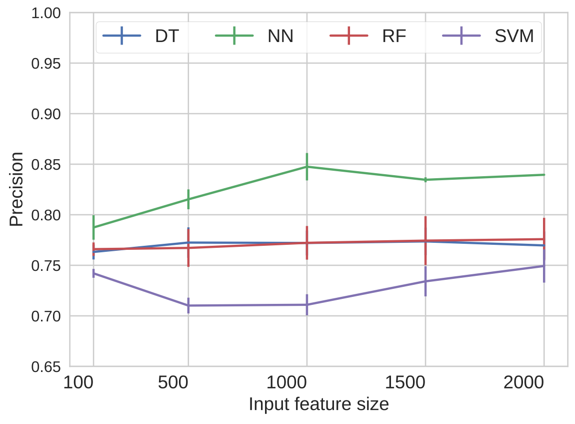
Table 2: Mean results obtained for each model after 10-fold outer cross-validation for malignant melanoma and the standard error in the results.

We can observe that, as expected, for carcinoma, where the positive class is better represented in the dataset the sensitivity is much higher than for the malignant melanoma. Regarding the comparison between the classifier, the NN and SVM achieve the best performance. It is also noteworthy that the DT perform much worse than all of the other methods, due to the fact that it is not able to generalise well.

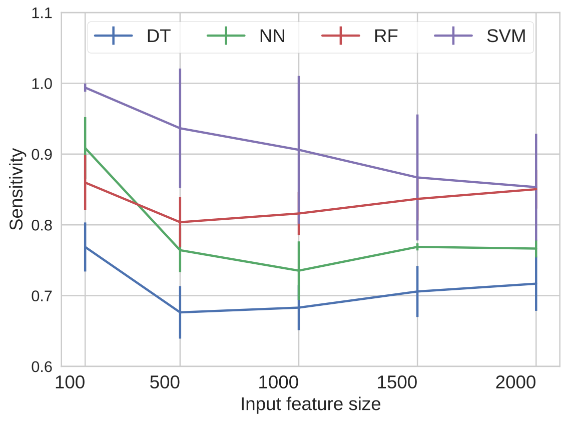
4.5 Performance with varying number of gene features

For the results obtained in sections 4.3 and 4.4 we used the 100 most mutated genes from the COSMIC database as features for classification. However, it was important to investigate whether the number of features used significantly affected performance. Consequently, we repeated the experiments for classifying carcinoma with the following values for the number of most mutated genes selected $\{100, 500, 1000, 1500, 2000\}$. The results obtained for the precision, sensitivity and ROC-AUC are exemplified in Figure 5.

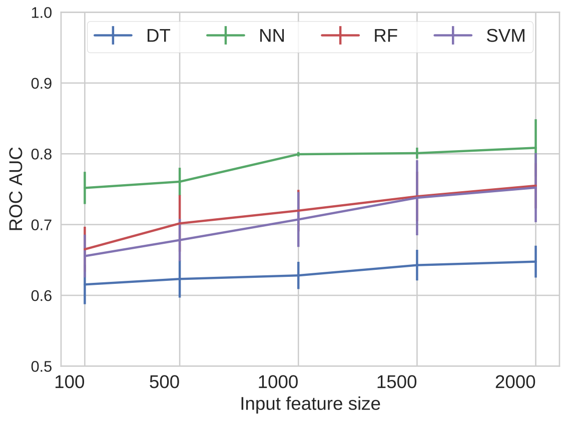
It is noteworthy that sensitivity decreases by increasing the number of features, while the ROC-AUC increases. On the other hand, the precision does not change significantly.



(a) Precision



(b) Sensitivity



(c) ROC-AUC

Figure 5: Mean results for precision, sensitivity and ROC-AUC and the standard deviation of the results obtained by varying the number of input features used for classification for each model.

One of the main issues encountered with varying the feature size was the increase in the time the classifiers would take to fit the training dataset. As it can be observed from Figure 6, the SVM is affected the most by this, followed by the RF.

Considering the results presented in this section, choosing to only use 100 gene features for classification was a good trade-off between obtaining valid results and being able to run the algorithms and perform the sweep across the wide range of values of the hyper-parameters in an acceptable amount of time.

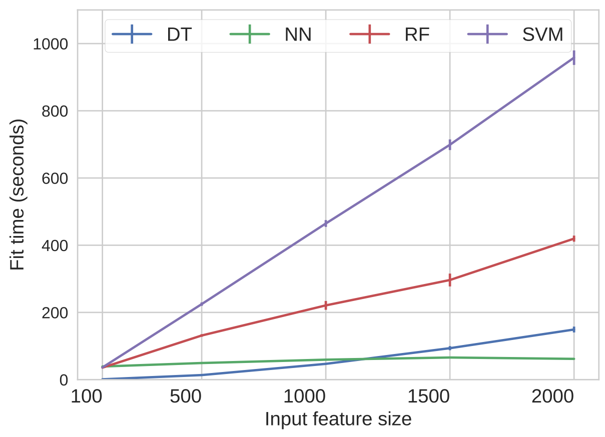


Figure 6: Mean time needed for training each classifier and the standard deviation of the time by varying the number of input features used for classification.

5 Discussion of hyperparameters

In addition to the quantitative results obtained, it was also interesting to analyse the hyperparameters for which each classifier obtained the best performance in the majority of folds while performing nested cross-validation:

- SVM: RBF kernel, $C = 2$ and $\gamma = 2^{-7}$
- DT: cross-entropy impurity measure
- RF: Gini inpurity measure and 95 tree estimators
- NN: dropout probability $p = 0.2$

The hyperparameters chosen for the SVM suggest that, in order to avoid over-fitting, the cross-validation method picks small values for C and γ that result in a separating hyperplane with large margins, thus allowing some amount of misclassification. Moreover, the impurity measure used differs between DT and RF, and the large number of tree estimators needed by the RF indicate that using a DT for classification is expected to perform much worse, a fact also validated by the experiments.

6 Conclusion

This report has investigated how somatic mutation data can be used to perform a histological classification of tumours using the following methods: SVMs, DTs, RFs and NNs. The results obtained show that the NN can achieve ROC-AUC > 0.7 on the classification of 20 histological types and precision > 0.7 on the classification of only 8 histological types. The other methods perform significantly worse for the ROC-AUC metric and comparable on the precision metric. Moreover, the additional experiments done to asses if the number of input genes affects performance show what using only 100 genes with somatic mutations can give sufficient insight in the performance of the classifiers.

References

[1] S. A. Forbes, D. Beare, H. Boutselakis, S. Bamford, N. Bindal, J. Tate, C. G. Cole, S. Ward, E. Dawson, L. Ponting, *et al.*, "Cosmic: somatic cancer genetics at high-resolution," *Nucleic acids research*, vol. 45, no. D1, pp. D777–D783, 2016.

[2] J. N. Weinstein, E. A. Collisson, G. B. Mills, K. R. M. Shaw, B. A. Ozenberger, K. Ellrott, I. Shmulevich, C. Sander, J. M. Stuart, C. G. A. R. Network, *et al.*, "The cancer genome atlas pan-cancer analysis project," *Nature genetics*, vol. 45, no. 10, p. 1113, 2013.

[3] D. Amar, S. Izraeli, and R. Shamir, "Utilizing somatic mutation data from numerous studies for cancer research: proof of concept and applications," *Oncogene*, vol. 36, no. 24, p. 3375, 2017.

[4] L. Sobin, "The international histological classification of tumours," *Bulletin of the World Health Organization*, vol. 59, no. 6, p. 813, 1981.

[5] H. Carter, S. Chen, L. Isik, S. Tyekucheva, V. E. Velculescu, K. W. Kinzler, B. Vogelstein, and R. Karchin, "Cancer-specific high-throughput annotation of somatic mutations: computational prediction of driver missense mutations," *Cancer research*, vol. 69, no. 16, pp. 6660–6667, 2009.

[6] M. Le Morvan, A. Zinovyev, and J.-P. Vert, "Netnorm: Capturing cancer-relevant information in somatic exome mutation data with gene networks for cancer stratification and prognosis," *PLoS computational biology*, vol. 13, no. 6, p. e1005573, 2017.

[7] A. M. Marquard, N. J. Birkbak, C. E. Thomas, F. Favero, M. Krzystanek, C. Lefebvre, C. Ferté, M. Jamal-Hanjani, G. A. Wilson, S. Shafi, *et al.*, "Tumortracer: a method to identify the tissue of origin from the somatic mutations of a tumor specimen," *BMC medical genomics*, vol. 8, no. 1, p. 58, 2015.

[8] Y. Yuan, Y. Shi, C. Li, J. Kim, W. Cai, Z. Han, and D. D. Feng, "Deepgene: an advanced cancer type classifier based on deep learning and somatic point mutations," *BMC bioinformatics*, vol. 17, no. 17, p. 476, 2016.

[9] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[10] J. R. Quinlan, "Induction of decision trees," *Machine learning*, vol. 1, no. 1, pp. 81–106, 1986.

[11] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[12] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814, 2010.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

[14] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[15] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[16] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, 2014.

[17] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[18] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.