Building and interpreting artificial neural network models for biological systems

T. Murlidharan Nair

Indiana University South Bend, 1700 Mishawaka Avenue, South Bend, Indiana, USA, e-mail: mnair@iu.edu

Abstract Biology has become a data driven science largely due to the technological advances that have generated large volumes of data. To extract meaningful information from these data sets requires the use of sophisticated modeling approaches. Towards that, Artificial Neural Network (ANN) based modeling is increasingly playing a very important role. The "black box" nature of ANNs acts as a barrier in providing biological interpretation of the model. Here, basic steps towards building models for biological systems and interpreting them using calliper randomization approach to capture complex information is described.

Keywords: ANN, Calliper randomization, interpreting black-box models

1 Introduction

Biological systems are complex entities involving a myriad of interactions that regulate biological processes in ways that we are only beginning to understand. There has been an acceleration in our understanding of biological systems, and this has largely been due to technological advancements and large-scale initiatives that have generated a plethora of valuable biological data.

Biological data may be broadly categorized as genomic, transcriptomic, and proteomic data (omic data). Genomic data refers to genomic DNA sequences obtained using next-generation sequencing methods. Transcriptomic data refers to data that quantifies transcripts obtained either by RNA sequencing or from microarrays. Proteomic data refers to the distribution of chemical moieties contained in proteins obtained by mass spectrometry. In addition, biological data may also be structural. This includes structures of biological macromolecules (DNA, RNA and proteins) obtained by X-ray crystallography, NMR, or cryo-electron microscopy. All omic data may be obtained in a high-throughput manner, while the throughput of structural data is comparatively lower, several structural genomics projects have contributed to a significant increase in biological structural data.

When studying a particular biological system, the conventional approach has been to propose a hypothesis [1], and then verify the hypothesis using supportive data–often termed a hypothesis-first approach. In the era of high-throughput data, approaching problems by harvesting large-scale data has significant advantages over a conventional hypothesis driven approach. This is because use of a data-driven approach makes it possible to detect things one was not expecting to see [2] . An outstanding example of this is the identification of the fusion of the BCR gene located on chromosome 22 with the ABL gene on chromosome 9, from high-throughput genomic data, even though it was known for decades that chronic myeloid leukemia patients had abnormalities associated with chromosomes 9 and 22 [3]. The precise identification of BCR-ABL gene fusion led to the understanding

of the cause for the deregulated expression of the tyrosine kinase enzyme and to the development of imatinib that inhibits kinase enzyme [4-6]. This example clearly illustrates that, when studying biological systems, the quest has always been to identify the biological entity first and then understand what it does. Biological systems being complex, the task of identifying the entity responsible is never straight forward or simple. This is further complicated by the fact that biological molecules can act pleiotropically, and may simultaneously orchestrate changes in the behaviour of a large cohort of responder molecules to create an altered phenotype. Theoretically, this behaviour can be approximated as molecules exhibiting higher order correlations. Thus analysing high-throughput data to extract such information would help in generating computationally derived hypotheses. Artificial neural networks (ANNs) are a class of machine-learning methods capable of deriving such information. This chapter describes the steps involved in building ANN models for biological systems and interpreting them using a technique called calliper randomization —a technique that helps identify features that are important in imparting knowledge to the ANN during training.

2 Methods

ANNs were initially modelled as mathematical approximations of the biological synapse and were meant to model the human brain[7] . ANNs, however turned out to be very efficient in pattern recognition and have found more application as a pattern recognition machine, than as a means of explaining how the brain

functions. Building an ANN model for a system involves collecting relevant information about the system that can be used to generate a function that approximates its behaviour. ANN models are built by presenting data associated with the system to be modelled to a network of computational units called artificial neural networks (Fig. 1). The network learns the pattern associated with the system being modelled by iteratively updating the weight connections between computational units called neurons. ANNs have been exploited extensively in analysing biological data. One of the main drawbacks of ANN models is that they are black-box models and do not reveal in a readily-interpretable form any information about the system being modelled. However, it is possible to delineate the relative importance of features in imparting knowledge to the network using the calliper randomization approach that we will describe later in this chapter.

2.1 Modeling a biological system using a neural network

Modeling a biological system using an ANN begins by defining the system to be modelled and enumerating the variables that could explain the behaviour of the system. Biological systems being complex entities, it is unlikely that we will be able to account for all the variables that describe it, and hence only partial information is available to model it. For example, imagine the system being modeled involves building a model capable of distinguishing lung cancer types —adenocarcinoma (AC) and squamous cell carcinoma (SCC). While this problem can be approached from different angles, let us consider classifying them using their transcriptomes.

This reduces the problem as the variables are thereby limited to the transcriptomes of the two types of cancers. Their transcriptomes now serve as input features that are experimentally derived. Although not stated explicitly, we are hypothesizing that there exist features within the transcriptome that are capable of distinguishing AC from SCC. The transcriptomes are mapped to classes using an ANN that is indicative of the aforementioned types of cancers. The function/model that approximates this classification is the ANN model. An important first step to building an ANN model is to select a subset of features that is most relevant to be used as input to the neural net. Since transcriptomic data normally contains a very large number of transcripts (>50,000), the process can be very computationally intensive.

2.2 Feature selection

The main goal of feature selection is to distinguish relevant features from irrelevant ones. If the value of a feature in a tumor sample is significantly different from the value of the same feature in a normal sample, then that feature is likely to be relevant. Selecting features from transcriptomes involves testing the differences in expression between many means. This can be conveniently achieved by using multiple comparisons [8]. Comparison of two means may be achieved using routine statistical testing methods like interval estimation or hypothesis testing. Comparisons of several means can be done using ANOVA based methods. A key drawback of this approach is that it does not identify which means were

different; for this, we could use multiple comparison based methods. The multcomp package provides methods to conveniently analyze data using multiple comparison-based methods. Multiple comparison of a subset of features that show significant difference between AC and SCC is shown in Fig. 2.

When several different treatments are involved, it is possible to score each significant comparison and rank features based on their scores [9]. The feature selection approach that we have mentioned here is only one of several such methods [10, 11]. It is noteworthy that the methods used in feature selection need not be too stringent as neural networks are capable of handling noisy data. If the feature selection process is too stringent there exists a possibility of filtering away features that may be associated with the system through second and higher order correlations. The choice of the feature selection method used depends on the system being modeled and should be carefully chosen [12].

2.3 Model building

ANN models can be conveniently built in R using the Stuttgart Neural Network Simulator (RSNNS)[13]. Building a neural network model involves presenting the selected features to an artificial neural network for training. The number of neurons in the input and the output layer is determined by the system being modeled; however, both the number of hidden layers, and the number of neurons in each hidden layer that captures the non-linearity of the system, are rather

arbitrary and need to be optimized for the system under study. Training involves iteratively changing the weight connections between neurons in a manner that will optimally map the input onto the output. In the example being discussed, the input would be expression values of a subset of features that were deemed to be significant by the feature selection process, and the output would be 1 for AC and 0 for SCC. It is important to mention that the data used for developing a neural network model need to be divided into training and test data sets. The weight connections are adjusted based on the training data only. The test data set is used to determine the performance of the model and is never used to adjust the weight connections. A well-trained generalized model should perform optimally not only on the training data set but also on the test data set (Fig. 3). If the network is over-parametrized by using too many hidden layers and neurons, the resulting model will only memorize the training examples and perform poorly on the test data sets.

2.4 Evaluation neural network models

The evaluation of neural network models is done using the *de facto* performance measures of sensitivity and specificity. In the example under discussion, sensitivity measures the correctly classified samples, while specificity measures the proportion of AC classified as SCC and vice versa. It is now routine to depict this information graphically using a receiver operating characteristics (ROC) graph [14-16]. Briefly, ROCs are two-dimensional graphs in which the true positive (TP) rate

(sensitivity or recall) is plotted on the ordinate or Y axis, and the false positive (FP) rate is plotted on the abscissa or X axis. These are defined as follows:

$$TP\ rate = \frac{TP}{TP+FN} \quad (1)$$

$$FP\ rate = \frac{FP}{FP+TN} \quad (2)$$

The ROC curve helps determine the performance of the neural network model and reveals the percent of samples within a particular class that are correctly classified (true positives or "hits") as well as the ones that are incorrectly classified (false positives). In addition, the classification also generates true negatives (TN) and false negatives (FN) or "misses". Since these are complements of the others, they can be ignored when constructing ROC curves. These curves are bow shaped. They rise from the lower left corner, where both percentages are zero, to the upper right corner, where both are one hundred, with a sharp bend for a perfect classifier (Fig. 4).

2.5 Calliper randomization for interpreting neural network models

Neural network models are a "black box". This is a potential serious limitation when applied to systems where it is necessary to interpret the model. Since neural networks are a parallel and distributed system, it is not possible to interpret the weights of the optimized model conveniently. Further, it is also possible to obtain

two different models that have similar performance but whose weights may be completely different. An alternative to this is to perturb the input and evaluate the performance of the model. This approach was inspired by early experiments that were done to determine the principal component involved in carrying genetic information from the complex mixture of cell components [17, 18]. The approach helps provide insight into the system being modeled, assists in evaluating the relative importance of the features that are part of the input space, and derives the principal features among them. This approach, called calliper randomization, was first proposed by the author and applied to the analysis of biological sequences[19]. The approach has been extended to the analysis of microarrays[20], and can be conveniently applied to any learning model. The algorithm is briefly described below.

Algorithm 1 Calliper randomization algorithm

```
procedure CALLIPER RANDOMIZATION
    model ← ANNmodel
    calliperWindow ← required windowSize
    TestDataTmp ← TestData
    numFeatures ← number of features in one TestDataTmp
    For i ← 1 to (numFeatures -calliperWindow) step 1
      predictTest ← predict(model, TestDataTmp)
      calliperError ← determineClassificationError(predictTest)
      TestDataTmp ← TestData
      For j ← i to (i +calliperWindow) step 1
        TestDataTmp[j] ← perturbed data
      end For j
    end For i
```

The calliper plot of perturbed positions versus the calliperError provides a view of the features that are deemed important in imparting knowledge to the neural

network. Those features, which, when perturbed, hamper the performance of the model, in this case by miss-classifying the samples above a particular threshold, (percentage or area under the curve (AUC) of the ROC curve) can be extracted for further functional analysis. Calliper plot for the example of AC and SCC is shown in (Fig. 5).

Features which when perturbed, contribute to miss-classification the most are considered important in imparting knowledge to the network. In figure 5, the features that affect the prediction ability of the classifier by greater than 40% are depicted in larger dark circles. Since in this case a window of features is perturbed, all the features contained in the window are considered important and may be subject to further downstream analysis. Because the expression of genes is affected by other genes, it is possible to identify these by perturbing different subsets of features and evaluating the performance of the model.

Another important notion in using the calliper randomization is that of **directed callipers** —these are features that are known *a priori* to be important for proper functioning of the system under study. In such cases callipers may be placed at specific positions to include known or hypothesized features and those may be selectively perturbed. An example is the Shine/Dalgarno sequences and initiation codons which are known to be important for translation initiation, when building models to identify ribosome binding sites[21]. Selectively perturbing these would hamper the prediction capability of the model, indicating that these are key features of the ribosome binding site. This notion of directed callipers is not limited

to sequence data, and can be extended any type of data for which prior knowledge about features may be available.

3 Summary

Understanding biological systems in terms of the individual components and their function is a complex endeavor, even when a plethora of data is available about the system under study. Using an ANN based modeling approach, it is possible to capture the behavior of the system. However, despite ANN models being able to capture the behavior of the system, they are "Black-box" models and seem removed from being interpretable. This can be circumvented by using a biologically inspired perturbation method called calliper randomization. This method helps delineate the principal features from the complex data sets and may be key functional players biologically. These may be considered computationally derived hypotheses that can then be validated experimentally.

Acknowledgement

I would like to thank IUSB for funding this work. This work is also supported partly by NSF award 1726218.

References

1. Weinberg, R. (2010) Point: Hypotheses first, *Nature.* **464**, 678.
2. Golub, T. (2010) Counterpoint: Data first, *Nature.* **464**, 679.

3. Groffen, J., Stephenson, J. R., Heisterkamp, N., de Klein, A., Bartram, C. R. & Grosveld, G. (1984) Philadelphia chromosomal breakpoints are clustered within a limited region, bcr, on chromosome 22, *Cell.* **36**, 93-9.

4. Nowell, P. C. (1962) The minute chromosome (Phl) in chronic granulocytic leukemia, *Blut.* **8**, 65-6.

5. Nowell, P. C. (2007) Discovery of the Philadelphia chromosome: a personal perspective, *J Clin Invest.* **117**, 2033-5.

6. Salesse, S. & Verfaillie, C. M. (2002) BCR/ABL: from molecular mechanisms of leukemia induction to treatment of chronic myelogenous leukemia, *Oncogene.* **21**, 8547-59.

7. Rumelhart, D. E., Hinton, G. E. & Williams, R. J. (1986) Learning representations by back-propagating errors, *Nature.* **323**, 533-536.

8. Westfall, P. H. (1997) Multiple Testing of General Contrasts Using Logical Constraints and Correlations, *Journal of the American Statistical Association.* **92**, 299-306.

9. Nair, T. M. (2012) Analysis of isoform expression from splicing array using multiple comparisons, *Methods Mol Biol.* **802**, 113-21.

10. Urbanowicz, R. J., Meeker, M., La Cava, W., Olson, R. S. & Moore, J. H. (2018) Relief-based feature selection: Introduction and review, *J Biomed Inform.* **85**, 189-203.

11. Liang, S., Ma, A., Yang, S., Wang, Y. & Ma, Q. (2018) A Review of Matched-pairs Feature Selection Methods for Gene Expression Data Analysis, *Comput Struct Biotechnol J.* **16**, 88-97.

12. Liu, H. & Wong, L. (2003) Data mining tools for biological sequences, *J Bioinform Comput Biol.* **1**, 139-67.

13. Bergmeir, C. & Benítez, J. M. (2012) Neural Networks in R Using the Stuttgart Neural Network Simulator: RSNNS, *Journal of Statistical Software; Vol 1, Issue 7 (2012)*.

14. Swets, J. A., Dawes, R. M. & Monahan, J. (2000) Better decisions through science, *Scientific American.* **283**, 82-7.

15. Fawcett, T. (2006) An introduction to ROC analysis, *Pattern Recognition Letters.* **27**, 861-874.

16. Bewick, V., Cheek, L. & Ball, J. (2004) Statistics review 13: receiver operating characteristic curves, *Crit Care.* **8**, 508-12.

17. Griffith, F. (1928) The Significance of Pneumococcal Types, *J Hyg (Lond).* **27**, 113-59.

18. Avery, O. T., Macleod, C. M. & McCarty, M. (1944) Studies on the Chemical Nature of the Substance Inducing Transformation of Pneumococcal Types : Induction of Transformation by a Desoxyribonucleic Acid Fraction Isolated from Pneumococcus Type Iii, *J Exp Med.* **79**, 137-58.

19. Nair, T. M., Tambe, S. S. & Kulkarni, B. D. (1994) Application of artificial neural networks for prokaryotic transcription terminator prediction, *FEBS Lett.* **346**, 273-7.

20. Nair, T. M. (2018) Statistical and artificial neural network-based analysis to understand complexity and heterogeneity in preeclampsia, *Comput Biol Chem.* **75**, 222-230.

21. Nair, T. M. (1997) Calliper randomization: an artificial neural network based analysis of E. coli ribosome binding sites, *J Biomol Struct Dyn.* **15**, 611-7.

Figure Legends

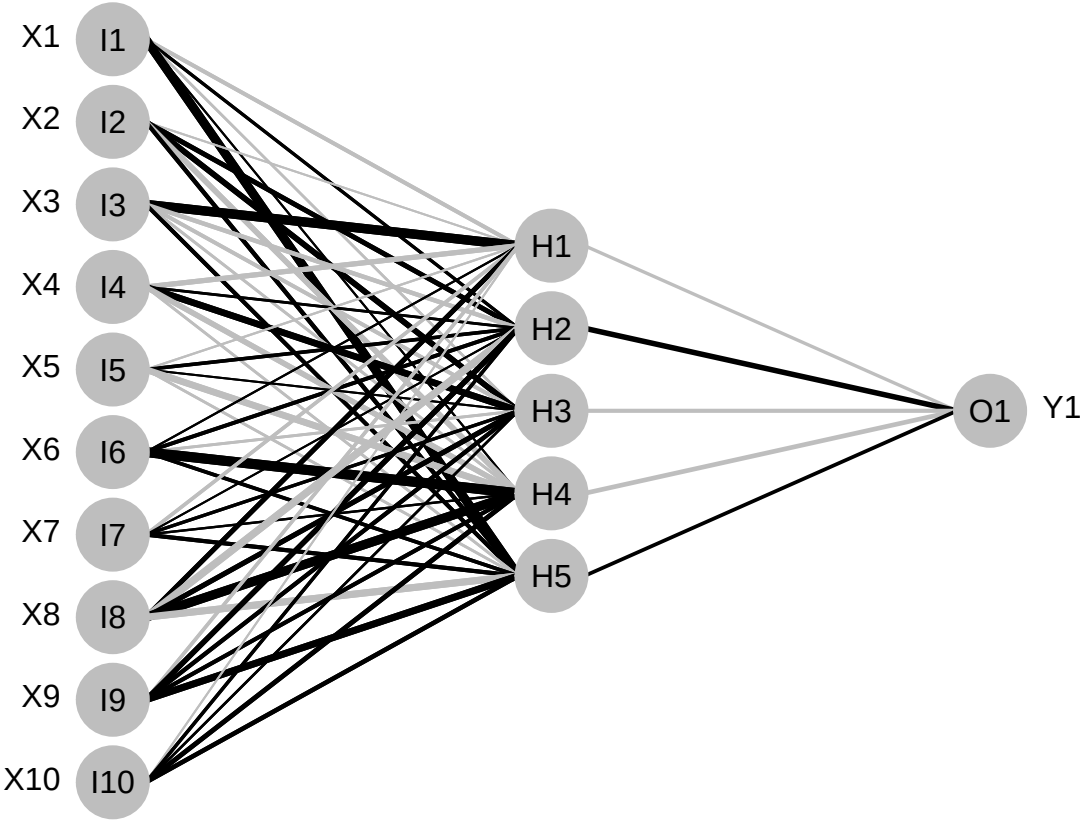
Figure 1. A three layered neural network with ten input neurons, five hidden neurons and one output neuron

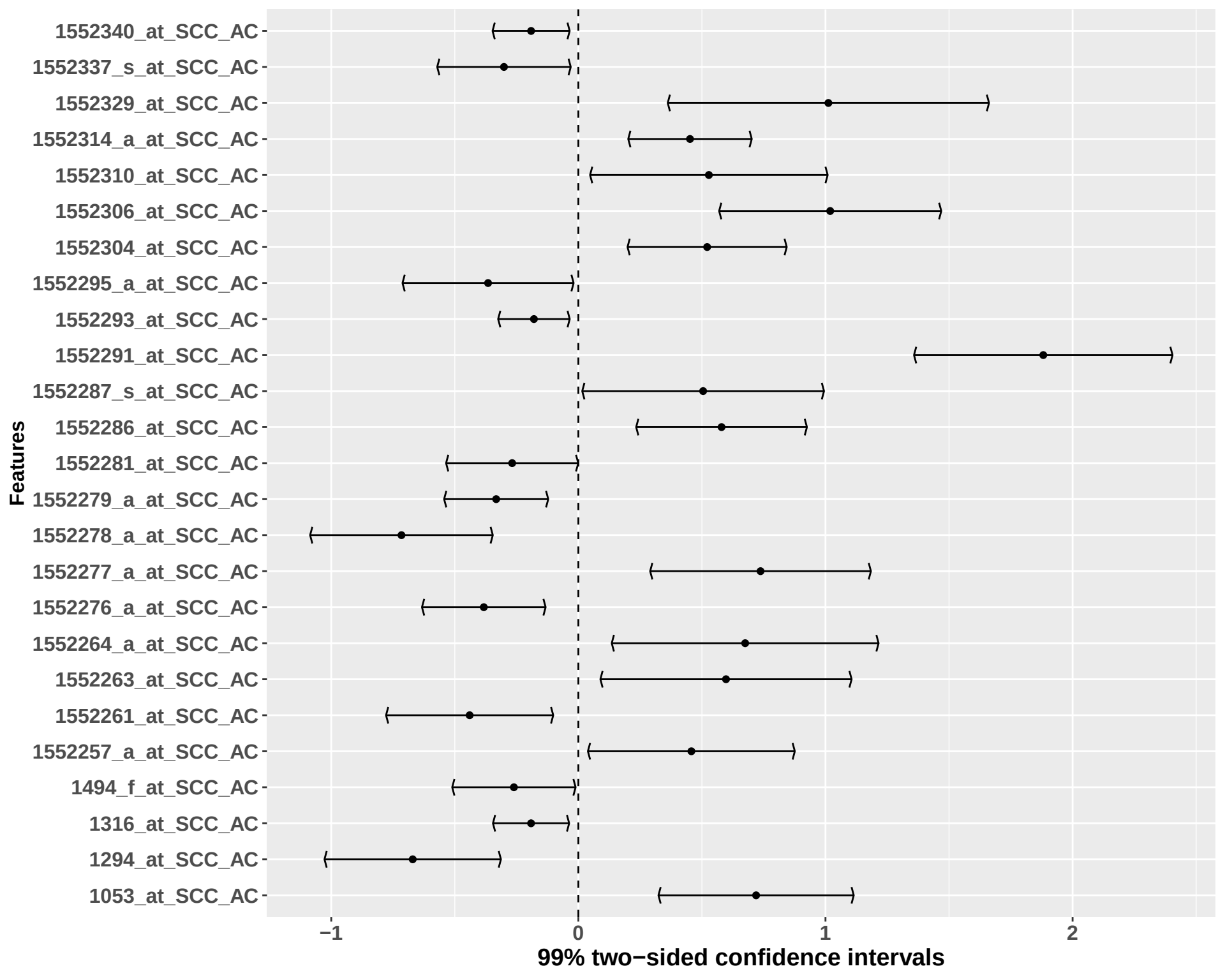
Figure 2. Multiple comparison of expression values for a subset of probes that show significant difference between SCC and AC. The ordinate represents probe ids for SCC and AC that were compared.

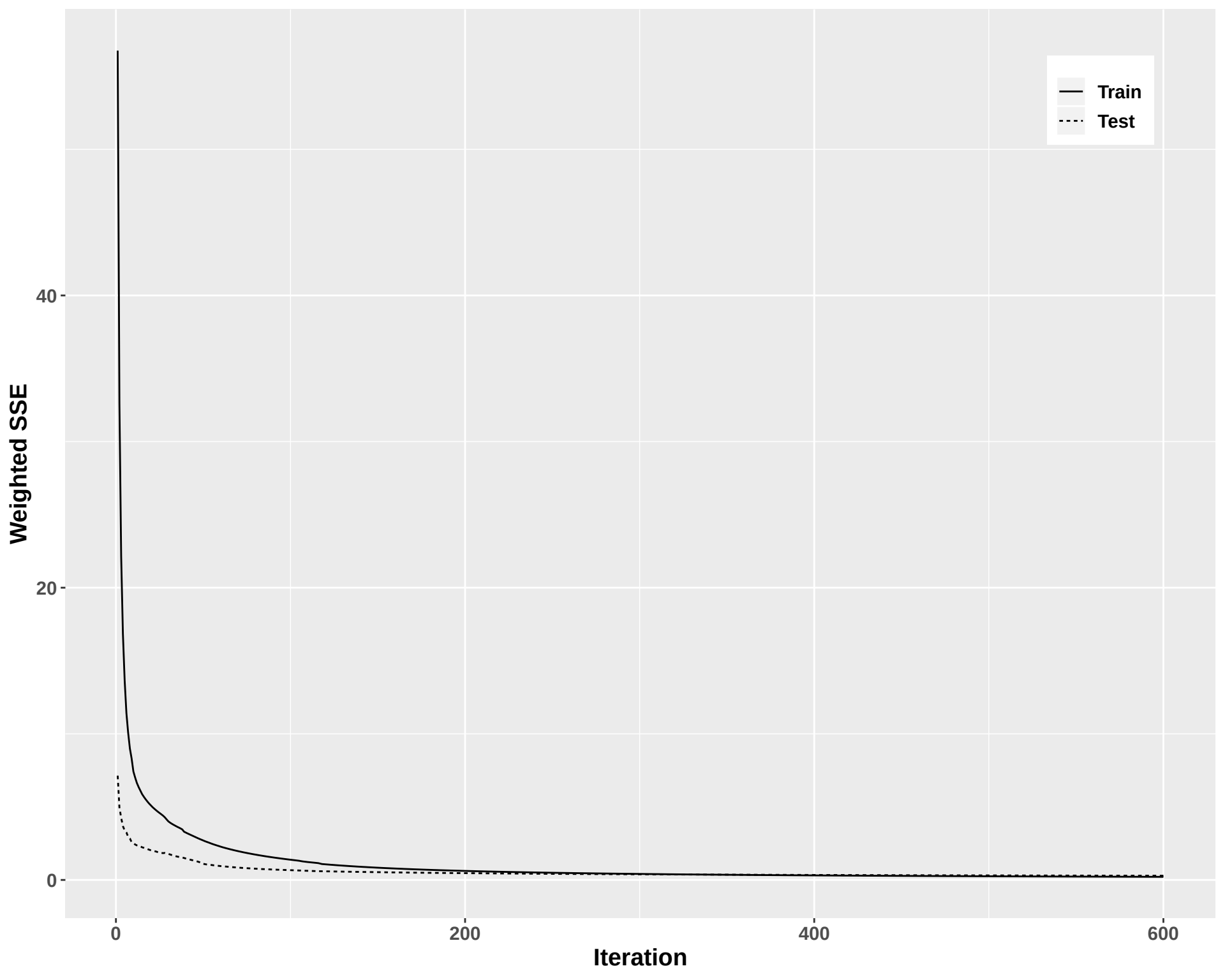
Figure 3. Error profile for the training and test data sets. Only the training data sets is used to change the weights of the neural network

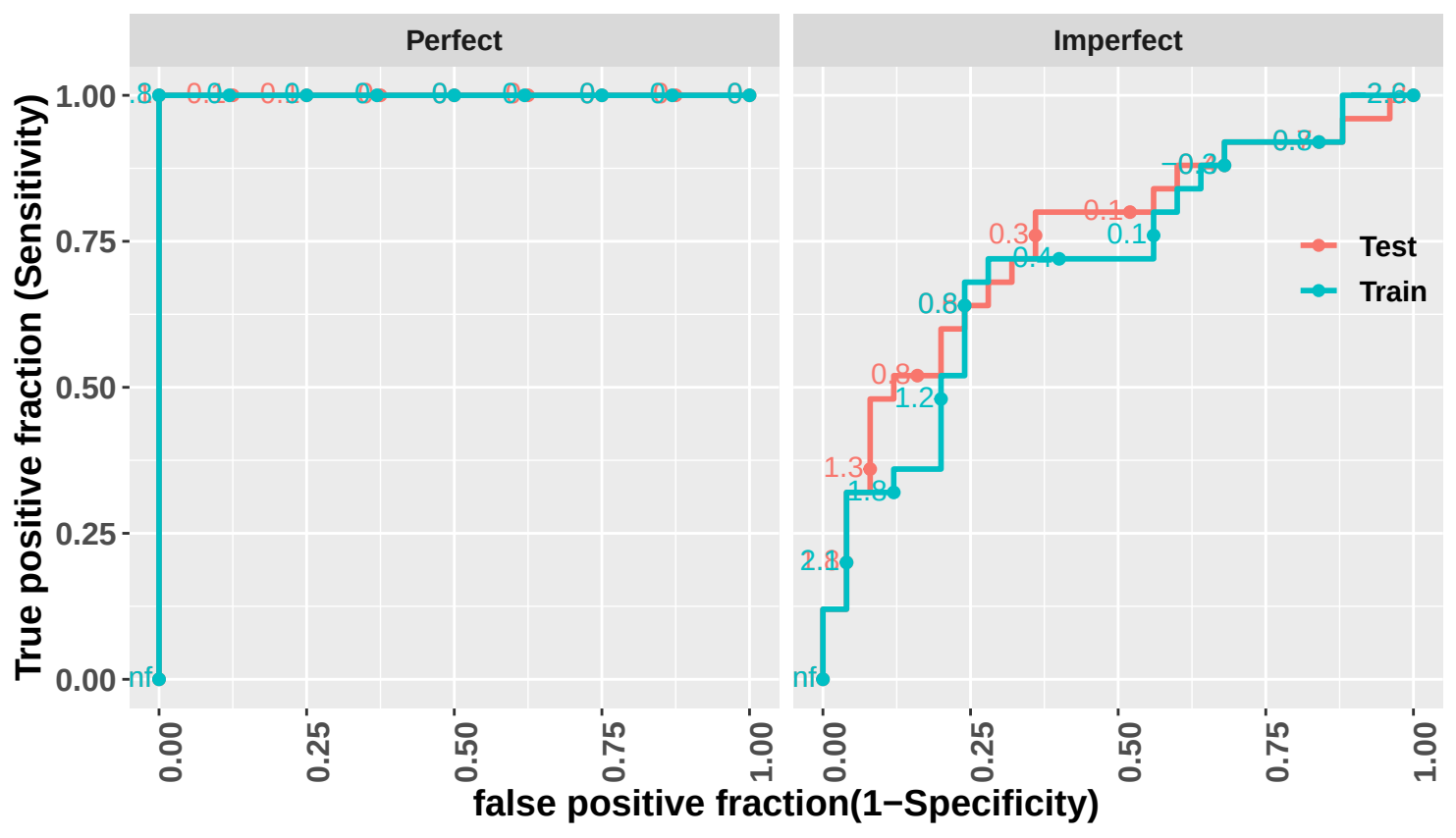
Figure 4. ROC curves depicting the performance of the ANN model for a perfect (left) and imperfect (classifier) right.

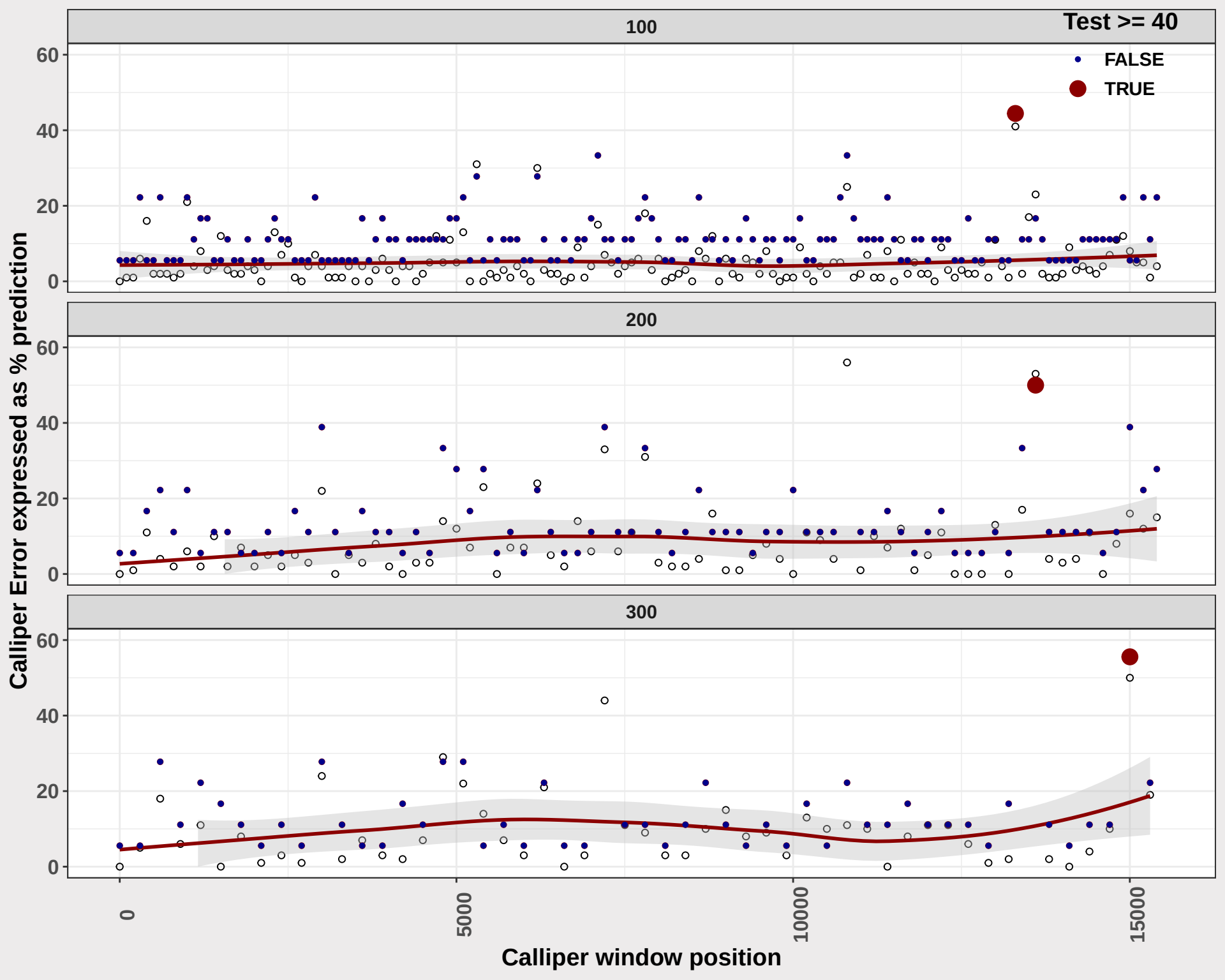
Figure 5. Calliper error as obtained using the neural network model for classifying AC and SCC


X1
I1
X2
I2
X3
I3
X4
I4
X5
I5
X6
I6
X7
I7
X8
I8
X9
I9
X10
I10
H1
H2
H3
H4
H5
O1
Y1


1552340_at_SCC_AC
1552337_s_at_SCC_AC
1552329_at_SCC_AC
1552314_a_at_SCC_AC
1552310_at_SCC_AC
1552306_at_SCC_AC
1552304_at_SCC_AC
1552295_a_at_SCC_AC
1552293_at_SCC_AC
1552291_at_SCC_AC
1552287_s_at_SCC_AC
1552286_at_SCC_AC
1552281_at_SCC_AC
1552279_a_at_SCC_AC
1552278_a_at_SCC_AC
1552277_a_at_SCC_AC
1552276_a_at_SCC_AC
1552264_a_at_SCC_AC
1552263_at_SCC_AC
1552261_at_SCC_AC
1552257_a_at_SCC_AC
1494_f_at_SCC_AC
1316_at_SCC_AC
1294_at_SCC_AC
1053_at_SCC_AC
Features
−1
0
1
2
99% two−sided confidence intervals


Train
Test
Weighted SSE
40
20
0
0
200
400
600
Iteration


Perfect
Imperfect
True positive fraction (Sensitivity)
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.00
false positive fraction(1−Specificity)
Test
Train


Test >= 40
FALSE
TRUE
100
200
300
Calliper Error expressed as % prediction
Calliper window position
0
5000
10000
15000
0
20
40
60