# Exploring Interpretability Models for Machine Learning in Prioritizing Bona Fide Bacterial Small RNAs

by

**© Carlos Dasaed Salcedo Carreño**

A report submitted to the School of Graduate Studies in partial fulfillment of the requirements for the degree of Master of Science.

Supervisor: Dr. Lourdes Peña-Castillo, Ph.D.

Department of Computer Science

Memorial University

August 2020

St. John's, Newfoundland and Labrador, Canada

# Abstract

While there is a lot of research focused on creating black box machine learning (ML) models with high performance metrics, understanding what these models are learning or how they are arriving at their predictions has received less attention until recently. However, understanding what these black box models are actually learning can be just as important as the model's performance. Here, we applied Interpretability Models (IM) to the random forest (RF) model developed by Eppenhof et al. [1] in the pursuit of new knowledge about bona fide bacterial small RNAs, or sRNAs, and the usefulness of IMs. Sequencing-based studies have generated an overwhelming amount of putative sRNAs available in the literature. Eppenhof et al.'s model is designed to assist researchers in narrowing down putative sRNAs by prioritizing sequences based on the predicted likelihood of them being a bona fide sRNA. We used Partial Dependence Plots (PDP), Local Interpretable Model-agnostic Explanations (LIME), and SHapley Additive Explanations (SHAP) to obtain robust results and assess the agreement between the IMs. In particular, the PDPs showed that the lower the free energy of the predicted secondary structure of a genomic sequence, the higher the likelihood of it being bona fide sRNA. The distance to the opening reading frames (ORF) was also identified as one of the most important features in the identification of bona fide sRNAs. While LIME and SHAP served to confirm findings from the PDPs, they may have also hinted at the existence of a possible bias in the RF model when the distance to the ORFs is within a couple of nucleotides. Thanks to the IMs, not only was Eppenhof et al.'s model explained, but possible improvements may have also been identified, which may lead to new models with better performance metrics.

# Acknowledgements

I would like to thank my project supervisor, Dr. Lourdes Peña-Castillo, Ph.D. of the Departments of Computer Science and Biology at Memorial University of Newfoundland and Labrador. Dr. Peña-Castillo was always there to guide me throughout the project, steering me in the right direction, and helping me overcome obstacles when I got stuck. I could not have asked for a better supervisor. I would also like to thank my wife, Erika Paola Prada Garavito, who served as a second pair of eyes during the project and who provided me a steady coffee supply during various sleepless nights.

# Table of contents

# List of figures

# Glossary

| | |
|---|---|
| DistTerm | **Dist**ance to the closest Rho-independent **Term**inator |
| Distance | Distance to the closest left Opening Reading Frame |
| DownDistance | Distance to the closest right Opening Reading Frame |
| EPS | **E**xplanation **P**roducing **S**ystems |
| IM | **I**nterpretability **M**odel |
| LIME | **L**ocal **I**nterpretable **M**odel-Agnostic **E**xplanations |
| ML | **M**achine **L**earning |
| ORF | **O**pening **R**eading **F**rame |
| OrigRF | **O**riginal **R**andom **F**orest |
| OrigRF_Norm | **Orig**inal **R**andom **F**orest trained with **Norm**alized data |
| OrigRF_scaled | **Orig**inal **R**andom **F**orest trained with **scaled** data |
| PDP | **P**artial **D**ependence **P**lot |
| Pos10wrstsRNAStart | Distance to the closest -10 promoter site predicted in the genomic region |
| RF | **R**andom **F**orest |
| RFE | **R**andom **F**orest **E**xplainer |
| SHAP | **Sh**apley **A**dditive Ex**p**lanations |
| SS | free energy of the sRNA predicted **S**econdary **S**tructure |
| sRNA | **s**mall **R**ibo**N**ucleic **A**cid |
| sameDownStrand | Indicates whether the sRNA is transcribed on the **same strand** as its left ORF |
| sameStrand | Indicates whether the sRNA is transcribed on the **same strand** as its right ORF |
| $\Phi$ | Feature contribution for the SHAP values. |

# Chapter 1

# Introduction

American author Dale Carnegie once said, "When dealing with people, remember you are not dealing with creatures of logic, but with creatures of emotion...", [2] and in a world where Machine Learning (ML) is becoming an additional tool of the medical, financial, and legal systems, amongst others, it may sound irrational not to expect people to demand more answers and deeper explanations of ML models. [3] Unfortunately, while ML is arguably one of the most popular and fastest growing technologies of the twenty first century, many of its models are black boxes, even to developers themselves. [4] When it comes to low risk applications, like a movie recommender system, not knowing the exact intricacies behind the recommendations of the model may not be of particular importance. However, the same cannot be said about high risk models, like medical diagnosis systems, where knowing why a certain diagnosis was predicted and the contributing factors that yielded such result are of high interest to the parties involved. [5] This is also an issue that already transcends simple human curiosity, as laws, such as the General Data Protection Regulation (GDRP) in Europe, already require entities to properly justify why certain automated systems, produce certain results or predictions. [6, 7] Interpreting black box models is not only highly beneficial to end-users, but it also could provide developers and researchers with various benefits and insights into their models. Being able to understand why models generate their predictions can help in the detection of some common problems in ML such as data leakage, overestimation of accuracy, and propagated feedback loops. [8] In an attempt to understand what black box models do, researchers have come up with multiple methods, generally known as Interpretability Models (IM), such as Partial Dependence Plots (PDP) [9, 10], Local Interpretable Model-Agnostic Explanations (LIME) [4], and Shapley Additive Explanations (SHAP) [11]. This paper presents a comparison between these IMs applied to the ML model presented in the work of Eppenhof et al. [1] Since the validation process for sRNAs requires wet lab testing to be done, it is nearly impossible to test for every putative small RNA (sRNA) reported in the literature. The ML model developed by Eppenhof et al. was designed to help scientist in the field of microbiology identify and prioritize possible bona fide bacterial small RNAs so that lab testing can be more selective and efficient. By applying IMs to this ML model, it is possible to obtain new insights into the characteristics of actual sRNAs. [1]

While the term interpretability may be defined in various places, its exact meaning has not been universally agreed upon. [7] In a ML context this term can be summarized as the ability to explain and present in understandable terms to a human a proper justification of why a model generated a given prediction. [12] However, the way in which interpretability is presented to the end-user may vary depending on the model, the IM to be applied, and the scope of the interpretation. For this reason, IMs tend to be classified as model agnostic or model specific, and as local or global. Model agnostic and model specific simply refer to the ML methods to which the interpretability model can be applied. Anchors for example, would be considered model agnostic as it can virtually be applied to any ML model, while Treeinterpreter, as the name suggests, can only be applied to models generated by random forests. [8, 13, 14] Local and global models refer to the scope of the analysis done by the IM, but there are a few models that fit into both of these categories, such as Treeinterpreter. Local models explain the prediction obtained for a particular instance, while global models explain the entire model. [5] It has also been suggested that by combining multiple explanations from a local IM, a general idea of what the model as a whole is doing can be inferred, and hence, a global model can be obtained. [4, 8]

# Chapter 2

# Literature Review

Several different algorithms and frameworks have been developed to tackle the issues of interpretability and assist researchers in gaining deeper insights from their models. [15] Interpretability is not only useful for knowing why models behave the way they do, but also to help the ML specialist better tune their algorithms, correct mistakes and biases, and in the end, generate more powerful solutions. [12, 16, 17] Major tech companies, such as Microsoft, IBM, and Google, are already working on state-of-the-art frameworks to complement their AI technologies with IMs. [18, 19, 20, 21] [18]–[21] These technologies are designed to expand on the usefulness of ML models by allowing developers to implement global, local, model agnostic, and model specific IMs from a single interface and with little modifications to the code. However, big tech companies are not the only alternatives for creating and implementing IMs, and some notable examples can be found in [5], where the author provides a series of openly available R libraries as well as links to well-prepared examples. There are also open source libraries in Python, such as Yellowbrick [22], Eli5 [23], and MLxtend [24], which can be used to build IMs from scratch and tailored to specific needs. Other more specific applications of IMs can also be found in [16], [17], and [25].

## 2.1 Prevalent Models

While there exist several solutions in ML interpretability, Partial Dependency Plots (PDP), LIME, and SHAP provide a good initial overview. It is easy to find these methods, or variations of them, being used as a basis in various research papers (including this one), and they are also present in enterprise solutions. IBM's Watson OpenScale incorporates a LIME python library, Microsoft's Azure ML implements multiple variations of SHAP algorithms, and finally, several corporate solutions offer similar functionalities to those expected from a PDP. [18, 19, 26, 27] PDPs are used to show the effect that variations in one or two features would have on the predicted output in a given ML model, and are particularly useful in finding if the relationship between the features and the output is linear, monotonous, or more complex. An example of a PDP can be seen in Figure 2.1, where the researchers compared the probability of having cervical cancer versus the number of years of a woman using hormonal contraceptives. PDPs are able to accomplish these by marginalizing all other input features, and hence are able to simulate a relationship between the feature of interest, and the output of the model. While PDPs are considered easy to implement, they do have a few major drawbacks: they assume feature independence, and they are only able to realistically use a maximum of two features in a PDP function. [5] A deeper and more detailed mathematical explanation of the formulas behind PDPs can be found in [28], and [10] proposes a more profound analysis of the interpretability offered by PDPs in ML.

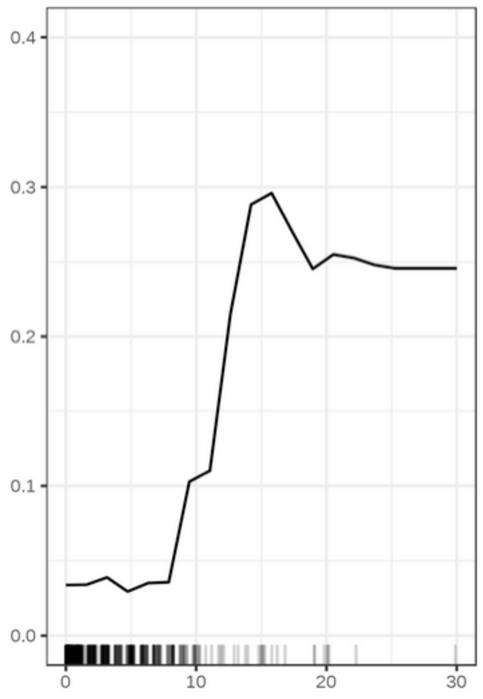


Fig. 2.1: Sample PDP displaying the probability of woman having cervical cancer based on years with hormonal contraceptives. [5] The authors of the image used data from the UCI Cervical cancer (Risk Factors) Data Set found in [29].

Local Interpretable Model-Agnostic Explanations (LIME) is an IM that has been well received by a large community, as its corresponding GitHub repository has over 7400 stars and 1200 forks. [30] While LIME may be primarily used for interpreting local instances, the researchers do mention that by building upon various individual predictions, a representative view of the ML model in question may be obtained. [4] To provide a general idea of LIME, the steps to apply this method to a black box ML model may be summarized as follows:

1. Select the instance to be analyzed
2. Make several slight perturbations to the features of the selected instance such that a whole new dataset that is not too different from the original instance gets created
3. Get the black box model predictions for the new data set

4. Based on the previous steps, select a minimum number of features that yield the highest likelihood of the predicted class being that of the instance in step 1

5. Fit a new weighted interpretable model on the new dataset basing the weights on the proximity between the permuted instances and the original one.

6. Explain the initial prediction for the instance in step 1 using the model obtained in the previous step [5, 31]

While the concepts behind LIME may seem intuitive, the method does have a couple of drawbacks to keep in mind. While there are many parts of the method that are tunable by researchers, it also implies that a lot of trial an error may take place in an effort to find an ideal and precise kernel. Additionally, some of the sampling done by LIME assume Gaussian distributions, or in other words, much like PDPs, feature correlations may be ignored. [5] Finally, LIME may not always guarantee consistent results, and studies have shown that different explanations may be obtained from very similar points. [32]

Even though SHAP may be an interesting and fairly new method in the world of interpretable ML, its main source of inspiration is based on game theory research from the 1950s, and in particular, the work found in [33], proposed by Nobel Prize winner Lloyd Shapley. [34] Lloyd Shapley proposed a method by which each player in a game would be entitled to a share of the winnings based proportionally on their contributions. A simple analogy can be made to explain the SHAP method, by considering the prediction task the game, each feature a player, and the prediction the winnings. This method works by estimating the marginal contribution each feature has on the difference between the predicted outcome and the average prediction. [11] Figure 2.2 illustrates some of the initial steps in the SHAP method, where the Shapley values, or contributions, each feature makes towards the final prediction is represented by the red and blue lines, as each feature is added into the IM. Figure 2.2 also hints at one of the biggest drawbacks of using Shapley values, the order in which the features are added does affect their marginal importance, and hence the reason behind the different values of $\Phi_4$ and $\Phi_5$ in permutation 1 and permutation $n$. The contribution each feature makes in every permutation needs to be added and averaged out to be able to obtain the correct marginal value, which makes this process computationally expensive and in the realm of NP-Hard problems. However, SHAP is able to overcome some of this computational complexity by using linear regression to estimate the Shapley values. [34, 35]

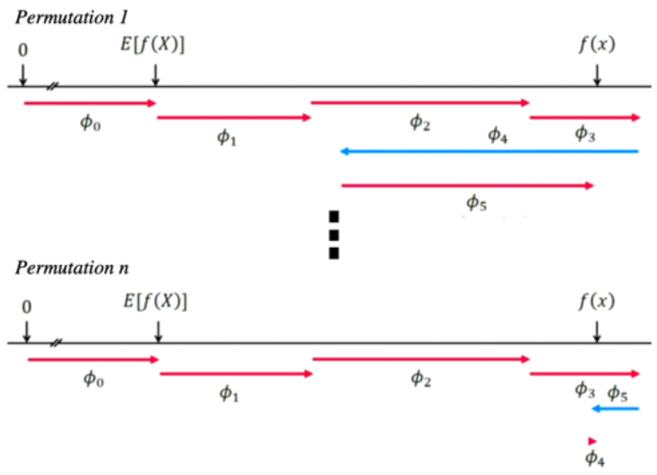


Fig. 2.2: Based on the explanations presented by Scott Lundberg in [34]. E[*f(X)*] is the average prediction expected of a given model, each $\Phi_n$ is a feature, *f(x)* is the prediction obtained for an instance, and the arrows represent the contribution each feature made in moving towards *f(x)* in every permutation. The average contribution for each feature would be its Shapley value, or marginal contribution.

SHAP draws its basis from other models, including LIME itself, but improves on these methods by using mathematically proven theorems and axioms. [34, 35] Figure 2.3 illustrates a comparison the researchers in [11] made between SHAP and other methods, hinting at the consistency and potential of SHAP in the world of machine learning.

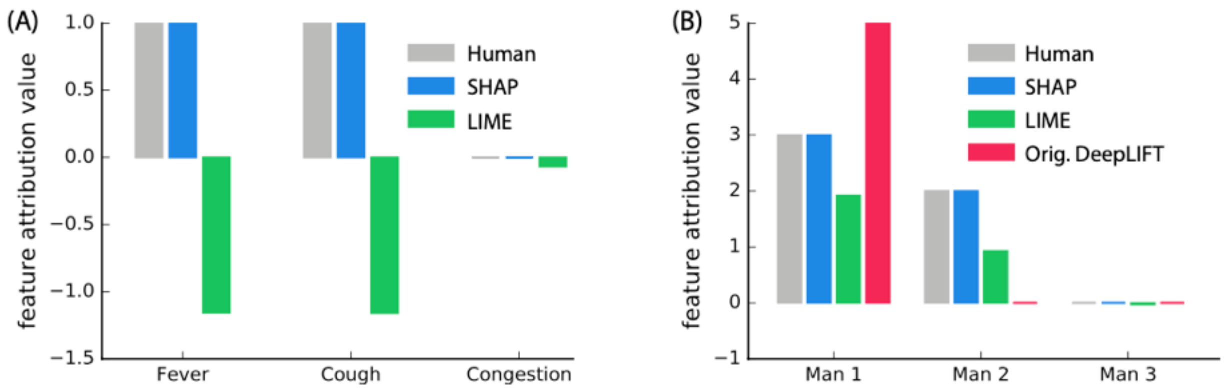
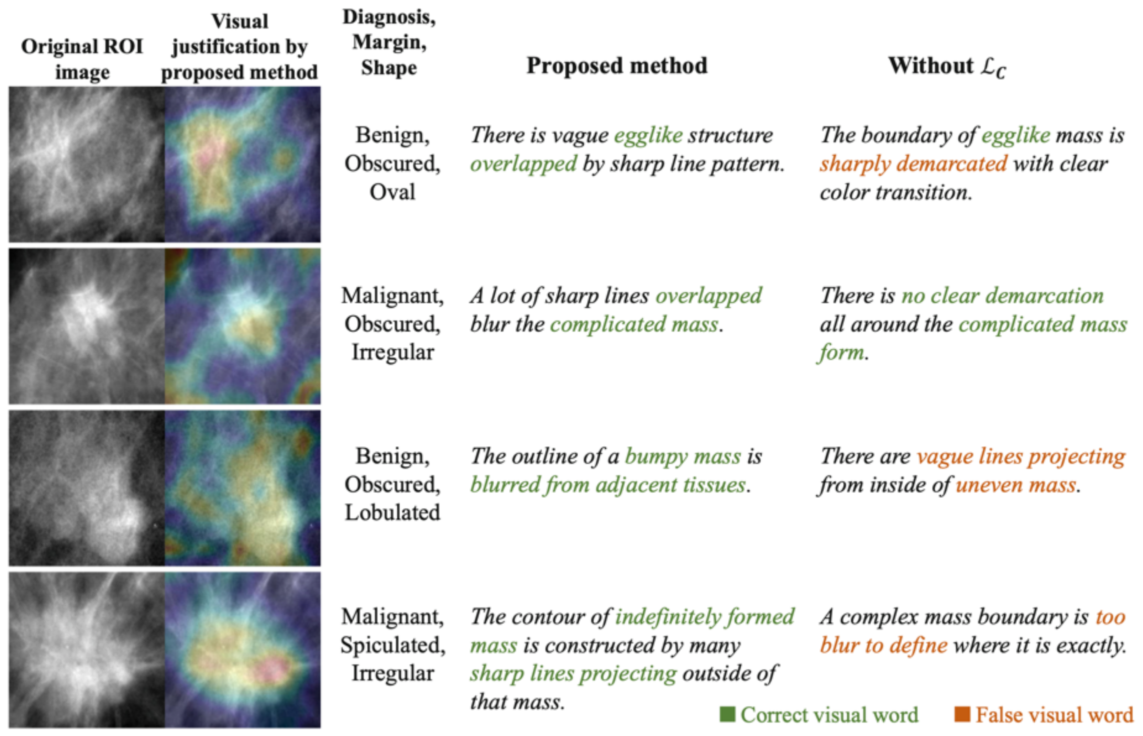


Fig. 2.3: Obtained from [11] CC-by. Shows results obtained from select interpretable models (SHAP, LIME, and Orig. DeepLIFT in the case of B) as compared with results from humans in two separate settings. As the authors in [11] mention, "participants were asked to assign credit for the output (the sickness score or money won) among the inputs (i.e., symptoms or players). We found a much stronger agreement between human explanations and SHAP than with other methods."

In this section, we have only discussed popular methods at their base level; however, a plethora of other algorithms can be found in the literature. SHAP's consistency and robust mathematical theorems and axioms, may make it seem like the superior method of choice. However, only popular methods at their base level have been discussed to this point and not the improvements that other researchers or companies may have done to them. Even the authors of LIME, have improved on their own algorithms with methods such as Anchors [8], and there exists various alternatives to PDPs such as ICE and ALE plots. While LIME, SHAP, and PDPs may not be the only existing IMs, they help in painting a general picture of research being done in ML. [5]

## 2.2 ML Models with Embedded interpretability

There are ML algorithms that do not rely on any of the previously mentioned IMs, and [36] refers to some of these models as "Explanation-Producing Systems", or EPS for short. These types of systems cause one to wonder if interpretability-related algorithms should simply be added as a part of the ML models themselves, or if applying IM post model training would be just as good or better? One way to go about creating these types of systems is to create ML models that have explanations included as part of each input value in their training data set. This allows the models to learn fairly appropriate explanations for their predictions, as evidenced in [37]. These models focus on the use, and sometimes combination of, textual and visual cues to provide a better explanation to the end user. A recent example of such models can be seen in Figure 2.4, which illustrates an improvement over the work done in [38], by including a word constraint model in their algorithm. [39] Another alternative to create EPS is to have a combination of different ML models embedded into a single system, where the models would have different end-goals related to the task at hand, but together, would generate a single output with a reasonable explanation. An example can be seen in Figure 2.4, where the researchers created a system composed of a Restricted Boltzmann Machine (RBM) and a random forest classifier, to better find correlations between imaging data, features, and target variables. [40]



| Diagnosis, Margin, Shape | Proposed method | Without $\mathcal{L}_C$ |
|---|---|---|
| Benign, Obscured, Oval | *There is vague egglike structure overlapped by sharp line pattern.* | *The boundary of egglike mass is sharply demarcated with clear color transition.* |
| Malignant, Obscured, Irregular | *A lot of sharp lines overlapped blur the complicated mass.* | *There is no clear demarcation all around the complicated mass form.* |
| Benign, Obscured, Lobulated | *The outline of a bumpy mass is blurred from adjacent tissues.* | *There are vague lines projecting from inside of uneven mass.* |
| Malignant, Spiculated, Irregular | *The contour of indefinitely formed mass is constructed by many sharp lines projecting outside of that mass.* | *A complex mass boundary is too blur to define where it is exactly.* |

🟩 Correct visual word 🟧 False visual word

Fig. 2.4: Sample results from [39] CC-by., in which they compare their proposed method with another one similarly trained but without their word constraint improvement (Lc).

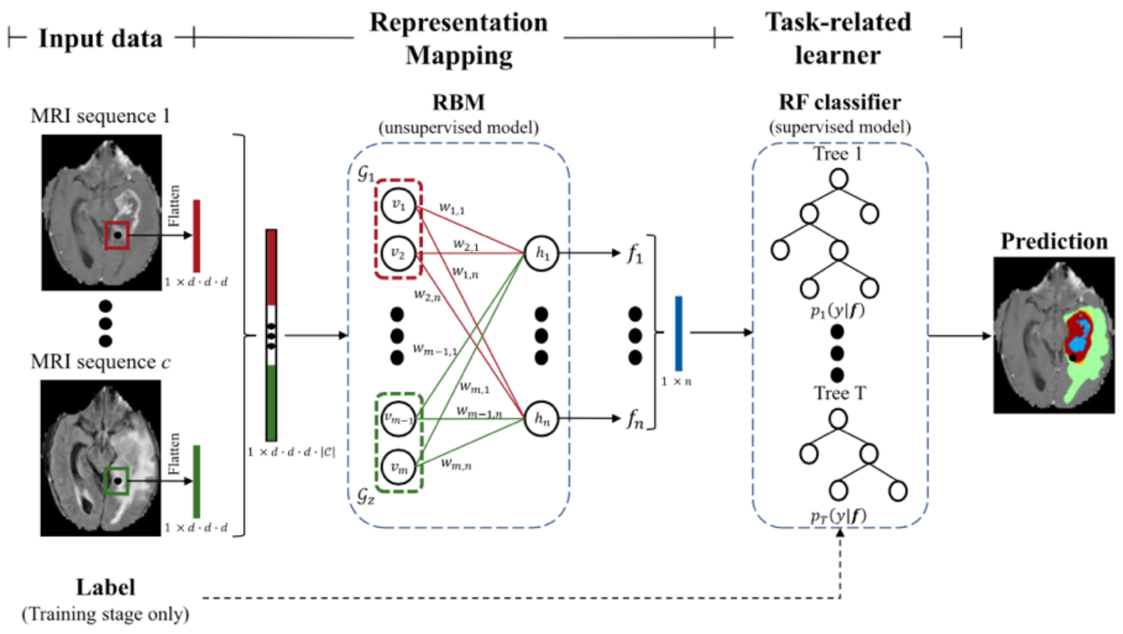


Fig. 2.5: Obtained from [40] CC-by., shows an overview of how the researchers proposed system, and how the combined models would generate a prediction with a visual explanation.

## 2.3 Evaluations of Interpretability Models

For IMs to be useful, they have to be tailored to suit domain experts and end users, as applying a random IM to a ML model does not guarantee that a person will obtain useful explanations. [7, 36] Therefore, defining the criteria for interpretability and the end goal of the IMs is an important step in any IM related research. For example, the researchers in [41], compared the results of LIME, Anchors, and SHAP in a health care setting and used six different metrics to measure each IM: identity, stability, separability, similarity, time, and bias detection. As the author's defined the terms:

"1) Identity: ... if there are two identical instances, then they must have identical explanations.

2) Stability: ... instances belong to the same class must have comparable explanations.

3) Separability: ... if there are two dissimilar instances, then they must have

dissimilar explanations ... means that all the features used in the model are relevant to the prediction.

4) Similarity: ... the more similar the instances to be explained, the closer their explanations should be and vice versa.

5) Time: this metric represents the average time used by the interpretability framework to output an explanation across all the instances in the testing dataset...

6) Bias detection: the ability to detect bias in training data (used to train a machine learning model) from the explanations of instances in the testing dataset.“ [41]

However, each IM performed great in at least one metric, but poorly in another, meaning that the researchers were unable to declare any of the methods (LIME, Anchors, or SHAP) as a clear overall winner. The researchers in [32] focused on the aspect of robustness, which can be seen as a combination between stability and similarity, by testing if minor changes to instances would generate significant changes to the explanations generated by IMs, and demonstrated that for the most part, IMs are not robust enough to deal with counter-examples. Other papers serve to prove that IMs can be effectively applied to ML models, as in [42], but do not necessarily help on improving the understanding of any of the IM metrics mentioned before. Since interpretability is usually defined by the context in which it is used, when new IMs are created, researchers will generally test their methods by creating new ML models with common datasets (like the MNIST digit dataset), and then test their IM on these new ML models against other popular IMs, as seen in [11]. While these studies do present comparisons of various IMs, their focus is mostly on proving that their IMs can yield explanations just as acceptable by humans as other methods, but also disregard other aspects of interpretability, like the metrics used in [32, 41] or the interpretability levels found in [7]. Since there is no consensus for how to measure interpretability, there is not a clear method to properly decide if an explanation, or IM, is better than another. The researchers in [3] proposed a method to address this issue by measuring the information transfer rate at which humans replicate ML model predictions, but they only compared LIME and COVAR in a text classification task. While their results seem promising, more testing is necessary to validate how their method would translate to regression tasks and other types of classification tasks. Their method, much like many others, requires human intervention via crowd sourcing to validate the correctness of an IM's explanations. While human validation seems like an obvious way to go about certifying IM's explanations, humans may be prone to biases themselves, errors, and crowd sourcing itself may be time consuming and expensive, especially if field experts are needed for the validations. [3, 5]

# Chapter 3

# Methodology

The methodology used during the study consisted of a complete empirical analysis of the machine learning model developed by Eppenhof et al. [1], and the insights obtained from the models' results and IMs methods. The model will be referred to as Original Random Forest (OrigRF) to distinguish it from the variations made further in this study. The project was implemented in R, and all the relevant R scripts, notebooks, results, and data sets can be found in the supplementary material of the article. Multiple IMs methods with the same end result were used to provide sanity checks, validation for the generated results, and agreements between the IMs. Based on the testing done, the following steps were followed for the application of IMs to the OrigRF ML model:

1. **Data preprocessing and proxy model training**

2. **Application of global IMs**: To obtain a general view of the model as a whole. PDPs were chosen for this study [43], but in theory, any other global IMs may be used. [5]

3. **Application of local IMs**: To analyze local instances, LIME [44] and SHAP [45] were chosen. This facilitated validating the results from the previous steps and gaining deeper insight into outliers and misclassified instances.

4. **Results comparisons**: Compare the results of the IMs to generate a deeper understanding of the ML model, and the data being analyzed.

The IM_sRNA.Rmd file in the additional documents is organized according to these steps, and it contains a more detailed and in-depth analysis of the libraries, steps, and procedures alongside most of the corresponding R code. Since many of the corporate solutions for ML already offer IMs built-in as part of their packages, a corporate, but free to use software solution was utilized: H2O. [46] Other equivalent solutions from IBM and Microsoft, such as Watson OpenScale [19, 26] and the azureml-interpret package [18] respectively, were considered as well, since they do offer free packages and trials. However, corporate solutions seem to have the particular drawback of requiring the ML models to be created with their packages and not external libraries. In other words, directly loading the OrigRF into these libraries' methods is not be possible, and instead, a new model within each corporate solution would need to be created. While H2O was chosen over the other solutions, this was done out of convenience since the LIME R package [44] is compatible with H2O models by default. The H2O library also has built-in capabilities for providing performance metrics, PDPs, and SHAP values without having to install additional libraries, and allowing all the tests to be ran locally. [47]

## 3.1 Data preprocessing and proxy model training

Even though LIME was the first IM tried during the study prior to any data preprocessing, a couple of LIME's major drawbacks soon became present. The first drawback from LIME, was its inability to properly deal with mixed data types. Since LIME ultimately tries to fit a linear model to generate its explanations, it also means that it must assign weights to the different features present in the datasets. However, if the datasets are composed of a mixture of categorical and numerical values with different scales, it is likely that irrelevant numerical features with large scales can easily outweigh other more important features with lower scales. Additionally, as Zafar et al. mention in [48], "The process of perturbing the points randomly makes LIME a non-deterministic approach, lacking "stability"...". For these reasons, the initial explanations obtained from LIME with the OrigRF were not consistent; the explanations for an instance would drastically change between runs. To solve this issue, two different but similar data preprocessing techniques were applied to the training data used for the OrigRF: scaling and normalization. Afterwards, two new RF models were trained with the scaled and normalized data sets using similar settings and configurations as those used for the OrigRF. In other words, the number of trees was set to four hundred, the mtry (number of variables sampled at each split) was set to two, and only the additional localImp variable was set to true. The localImp variable in the randomForest function determines if the casewise importance measures should be assessed or not, and even though this setting was not included in the OrigRF, it was needed by the Random Forest Explainer (RFE) library. [43] To confirm the validity of the new models, their performance metrics and their testing data predictions, were compared to those of the OrigRF. The new RF models were named OrigRF_scaled and OrigRF_norm according to their training data. Finally, the minimum, maximum, standard deviation, and mean of the training data was used to normalize and scale all of the testing data.

The second drawback in LIME was found with the settings available in the implementation of the R library. The LIME library offers various adjustable settings, such as distance function, kernel width, quantile binning, and the number of quantiles to use. [49] However, these settings are applied the same way to each of the features in the entire data set even if some features follow a normal distribution while others do not. For example, there were two boolean features in the data sets that generated warning messages in the code due to lack of variance, and depending on the instance to be analyzed, were sometimes completely ignored by the LIME IM. Even though the normalized and scaled models were primarily meant to be used with LIME, they were used with SHAP as well. While the need for data preprocessing was not anticipated, as this is normally not required for RF models, having additional models for testing meant that the results would be more robust. Since this study was based around the work of Eppenhof et al., only the data sets included in that paper were used to train the models and test the IMs. The features used by the ML models are those identified by Grüll et al. [50], and can be automatically calculated using the pipeline provided in [1]. Eppenhof et al. describes these features as follows:

“1. free energy of the sRNA predicted secondary structure,

2. distance to the closest -10 promoter site predicted in the genomic region starting 150 nts upstream of the start of the sRNA sequence to the end of the sRNA sequence (if no promoter site is predicted in that region. value of -1,000 is used),

3. distance to the closest predicted Rho-independent terminator in the range of

[0, 1,000] nts (if no terminator is predicted within this distance range a value of 1,000 is used),

4. distance to the closest left ORF, which is in the range of (-$\infty$, 0] nts,

5. a Boolean value (0 or 1) indi cating whether the sRNA is transcribed on the same strand as its left ORF,

6. distance to the closest right ORF, which is in the range of [0, + $\infty$), and

7. a Boolean value indicating whether the sRNA is transcribed on the same strand as its right ORF."[1]

These features are referred to as SS, Pos10wrtsRNAStart, DistTerm, Distance, sameStrand, DownDistance, and sameDownStrand, respectively. The acronym ORF in this context refers to the Opening Reading Frame.

## 3.2 Application of global IMs

Global IMs are meant to provide a general view of a ML model, and depending on the library or method to use, there are various options available. [5] For this study, PDPs were implemented as they are generally easy to interpret and understand. The first PDPs were generated by the H2O library, but the plots contained negative values, or values beyond one. In other words, since the RF model provides the probability of an instance being an sRNA in decimal form, any value that is not between zero and one should not be possible. An explanation for this behavior in the PDPs can be attributed to the H2O library assuming the RF models were for regression, and not classification. To solve this issue, the Random Forest Explainer (RFE) library [43] was used as an alternative to identify key feature interactions and generate new two-dimensional PDPs. Based on the feature interactions identified by the RFE, the PDP R library [51] was also used to create additional PDPs. Both, the RFE and PDP R libraries generated similar plots with only differences in how the results were presented. The RFE library evenly divides the plot area into a specified number of squares provided by the user, and then outputs a single result for each particular square. For this reason, some of the plots generate by RFE might appear to be somewhat fuzzy, but at least possible potential outliers can be identified. On the other hand, the PDP R library averages out the results over a much broader and irregular region generating cleaner looking plots. While both libraries provided similar plots, only the ones from the PDP R library are included in the results section. Based on the feature interactions identified by the RFE library, the following PDPs were generated: DistTerm vs DownDistance, DownDistance vs Distance, Pos10wrtsRNAStart vs DownDistance, Pos10wrtsRNAStart vs Distance, SS vs Distance, SS vs DownDistance, and SS vs DistTerm.

## 3.3 Application of Local IMs and Results Comparisons

The final steps in the methodology consisted of applying LIME and SHAP to the normalized version of the OrigRF model and comparing the results among all the IMs tested in this study to empirically identify new insights about sRNAs and the ML models. Since the datasets present in the work of Eppenhof et al. [1] were used, nearly five thousand testing samples were available for analysis. To narrow down this number, the ten false positive samples for the OrigRF with the highest confidence scores were selected from the SLT2 [52, 53] and LU [54] testing data set. Likewise, the false negatives from the OrigRF with the lowest confidence scores from the same testing data sets were also used, yielding a total of forty samples to be analyzed. A list of these instances can be found in Figure 4.13 in the results section. Since the models created in the preprocessing steps were proven to be identical to the original one, only the LIME explanations from the OrigRF_norm were used. Each sample was run through the LIME and the SHAP IMs at least four times, to guarantee consistency in the explanations. However, complete consistency was only present in the explanations from SHAP, while the ones from LIME would occasionally vary, meaning that a feature's importance and contribution would sometimes change from one run to the next. The current implementation of LIME works in two parts: an explainer and an explain function. The parameters for each of these parts were set through trial and error until consistent and reasonable results were obtained, leading to the following settings:

- Settings for the explainer:
  - **bin_continuous** a was set to TRUE, as it determines if continues variables

should be binned together or not.

- **quantile_bins** was set to FALSE in order to be able to include the sameStrand and sameDownStrand features in the explanations. These features do not have enough variance for the algorithm and therefore would have otherwise been excluded from all the explanations.
- **n_bins** was set to ten as the default value was too low for this experiment and yielded inconsistent results.

- Settings for the explain function:
  - **n_permutations** was set to two thousand as initial tests indicated that the smaller the number, the less consistent the explanations would be in each run.
  - **dist_fun** was set to the default, Gower, as according to the LIME documentation, this particular distance function is capable of dealing with mixed categorical and numerical values. [49]
  - **n_features** and **n_labels**, were set to seven and one respectively, as they determine the number of features to use for an explanation, and the number of labels to explain.

On the other hand, the Shapper library [45] used to generate the SHAP values only required a predict_function to be defined. As mentioned by the library's documentation, "Since the model is a black box, the 'predict_function' is the only interface to access values from the model." [55] The comparisons between LIME and SHAP were done primarily by examining if both IMs agreed that a particular feature contributed positively or negatively in the ML model's prediction. The analysis of the magnitude of the feature contributions in each prediction was based on the feature rankings rather than the actual magnitude value. For example, if both IM's agreed that Distance was the top feature contributing to the model's prediction in a given instance, it was irrelevant if SHAP assigned a value of 0.4 and LIME a value of 0.3 as the information provided to the user was basically the same. After the comparisons between LIME and SHAP were done, the location of each instance was estimated in the PDPs that corresponded to the top contributed features estimated by the IMs. In other words, if a local IM identified SS, Distance, and DownDistance as the top contributing features, then the PDPs containing these features were used to validate if the models were in agreement. This way, the prediction from each instance was properly assessed across the three IMs in question.

# Chapter 4

# Results

While SHAP and LIME are great IMs, all the primary insights were obtained from the PDPs generated by the RFE and PDP R libraries. The PDPs generated for the individual features did not generate at any point, a value above fifty percent, meaning that a single variable is not enough to generate a conclusion. Nonetheless, the individual feature PDPs do show general tendencies for each of the variables, and in turn, do provide some insights into the identification of bona fide bacterial sRNAs. For example, Figure 4.1 shows that having a DownDistance slightly above zero, but under three thousand may increase the chances of finding a bona fide bacterial sRNA. In other words, each of the graphs from figure 4.1 to 4.5 show an ideal range for finding bona fide bacterial sRNAs based on the corresponding feature.

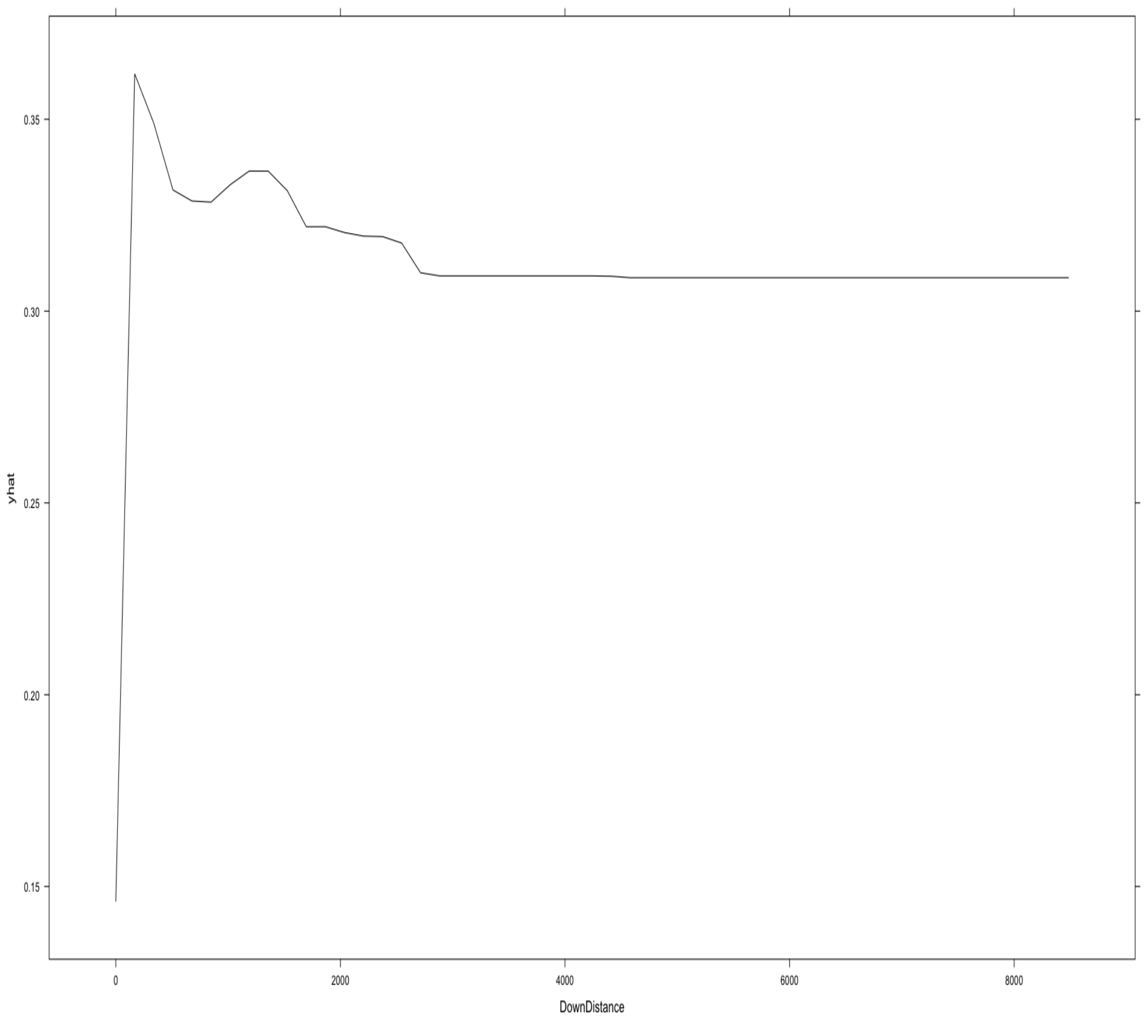


Fig. 4.1: PDP for the distance to the closest right ORF showing that the optimal range for finding bona fide sRNAs is between 0 and just under 3000.

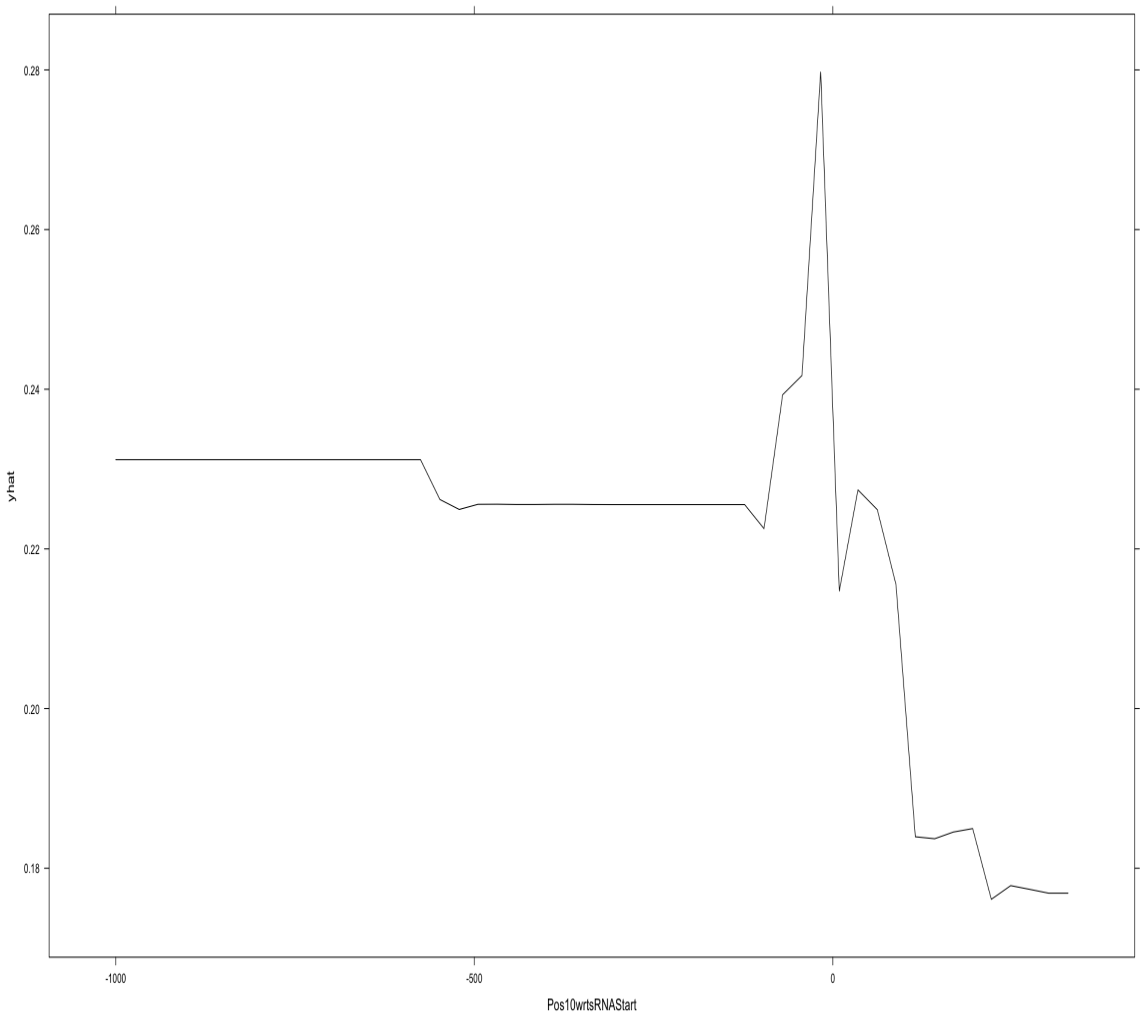


Fig. 4.2: PDP for the distance to the closest -10 promoter site. Based on the graph, the Pos10wrtsRNAStart may be a key defining feature when the values are closest to 0.

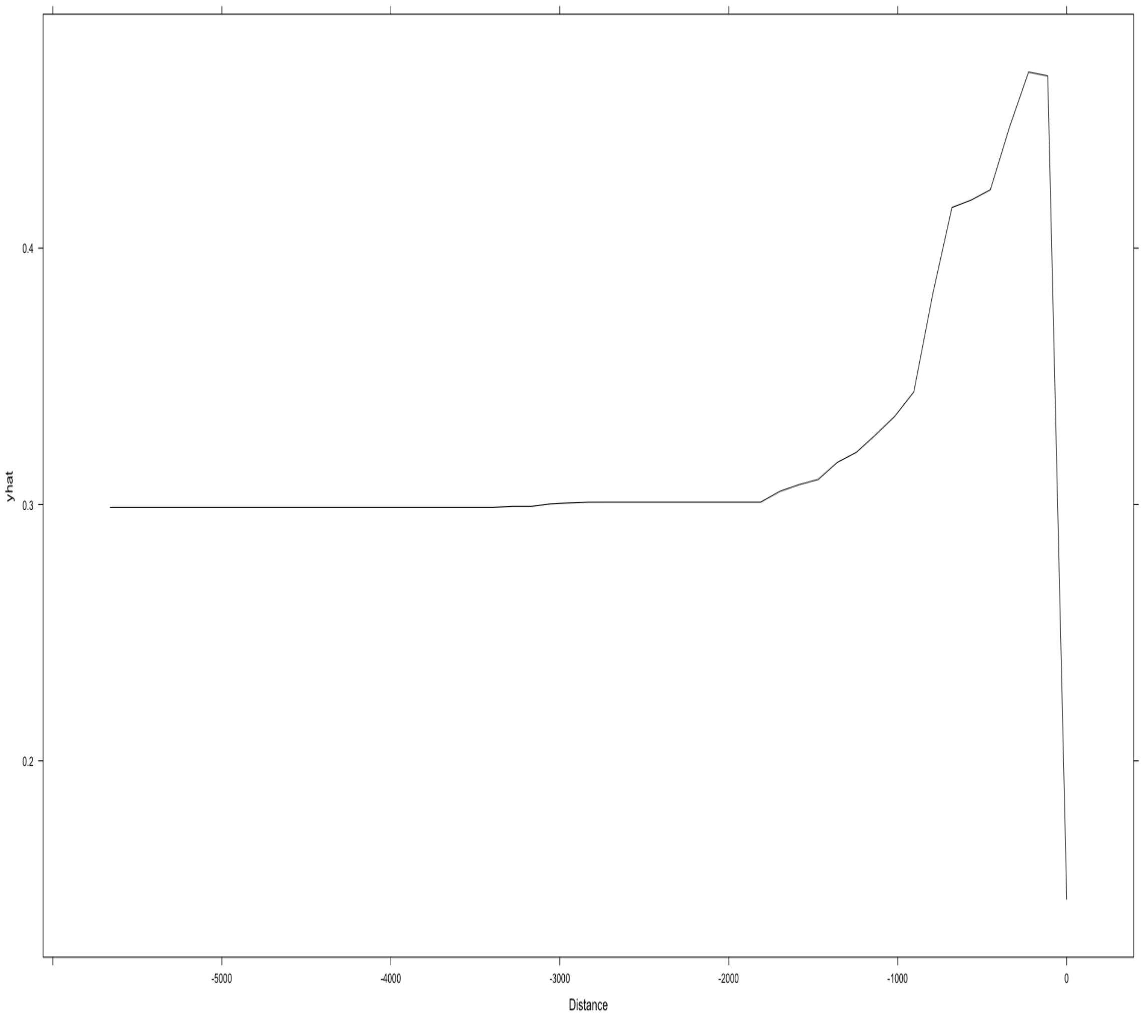


Fig. 4.3: PDP for the distance to the closest left ORF, indicating that the model's confidence in finding a bona fide sRNA is greater when the values are above -2000 and just below 0 nts.

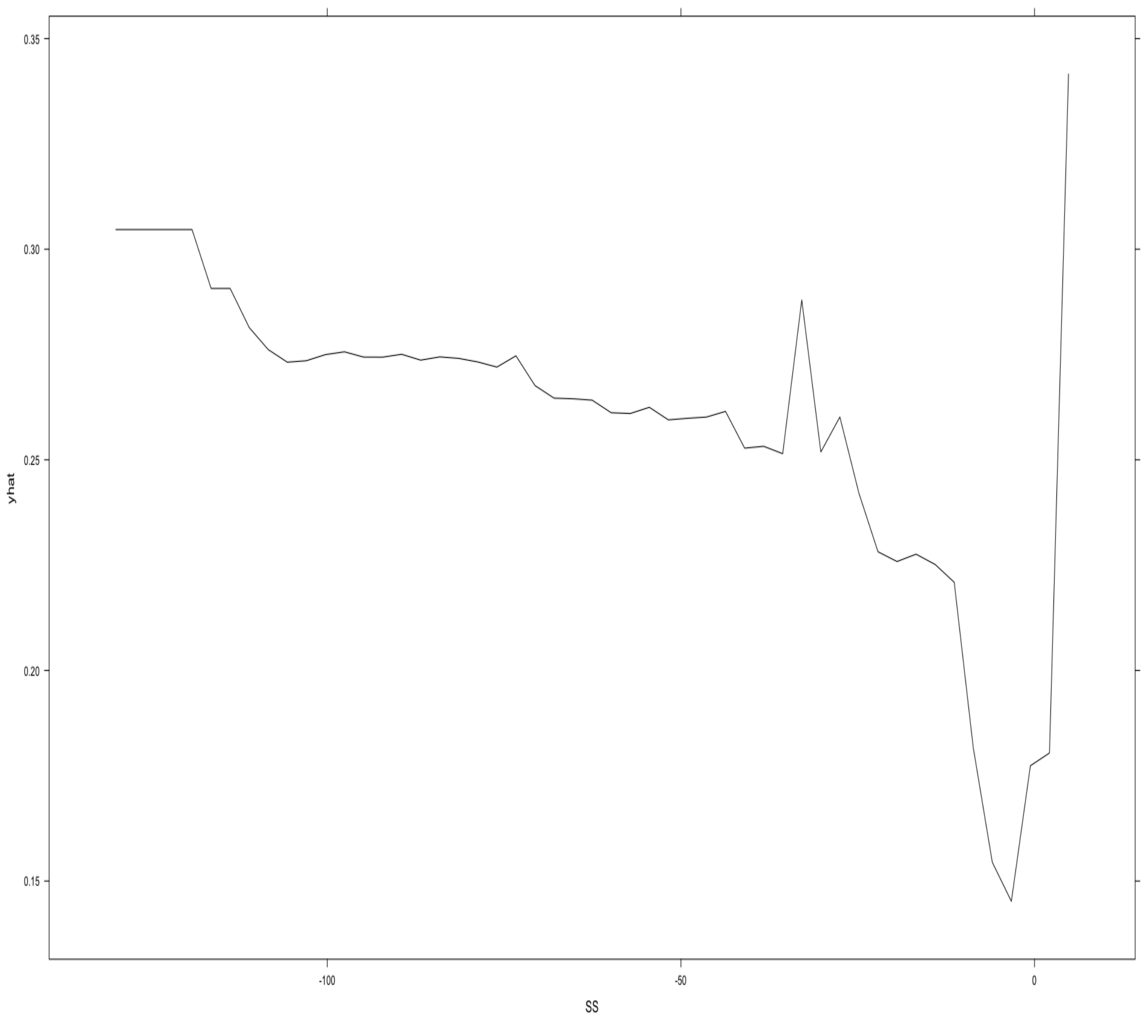


Fig. 4.4: PDP for the free energy of the sRNA predicted secondary structure, signifying that the less energy present in the instance, the higher the chances of finding an sRNA. RF models are able to deal with values outside the normal ranges even when these ranges are nonsensical, and hence, the values at 0 or above in the graph should be ignored.

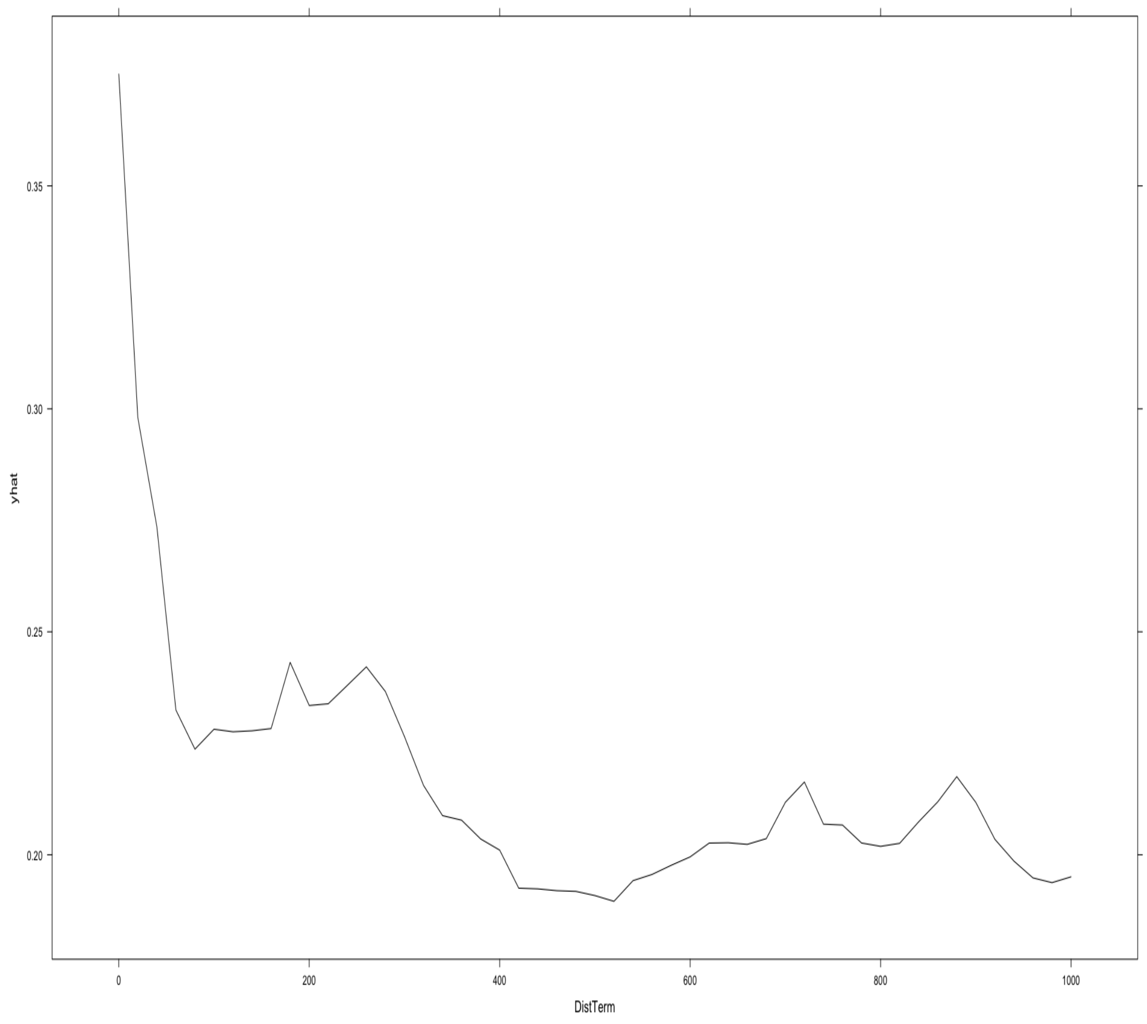


Fig. 4.5: PDP for the distance to the closest predicted Rho-independent terminator, demonstrating that the DistTerm may greatly contribute to the positive identification of bona fide sRNAs when its values are just under 100.

Since the individual feature PDPs are not enough to classify an instance as a bona fide sRNA, it was necessary to create additional two-dimensional PDPs based on key feature interactions. PDPs are limited to at most two variables per plot and with seven features, twenty-one possible unique graphs could have been generated to identify every possible non-repeating combination. Based on figures 4.6, 4.7, and 4.8, these graphs were narrowed down to seven PDP plots: DownDistance vs SS, Distance vs SS, DownDistance vs Pos10wrtsRNAStart, DistTerm vs SS, Distance vs Pos10wrtsRNAStart, DistTerm vs DownDistance, and DownDistance vs Distance. The first PDP plots correspond to the top five feature interactions displayed in Figure 4.8, while the last two are based on the importance rankings of Figures 4.6 and 4.7. In other words, even though the Distance and DownDistance are not in the top ten feature interactions, their importance ranking justified generating their PDP plot. The Distance vs DistTerm PDP plot was also created for the same reason. Even though Figures 4.6 and 4.7 have SS ranked fourth, this feature seems to be present in more interactions than any of the other features. Finally, Figures 4.6 and 4.7 also suggest that whether the sRNA is transcribed on the same strand as its left or right ORF may be irrelevant for the identification of bona fide sRNAs.

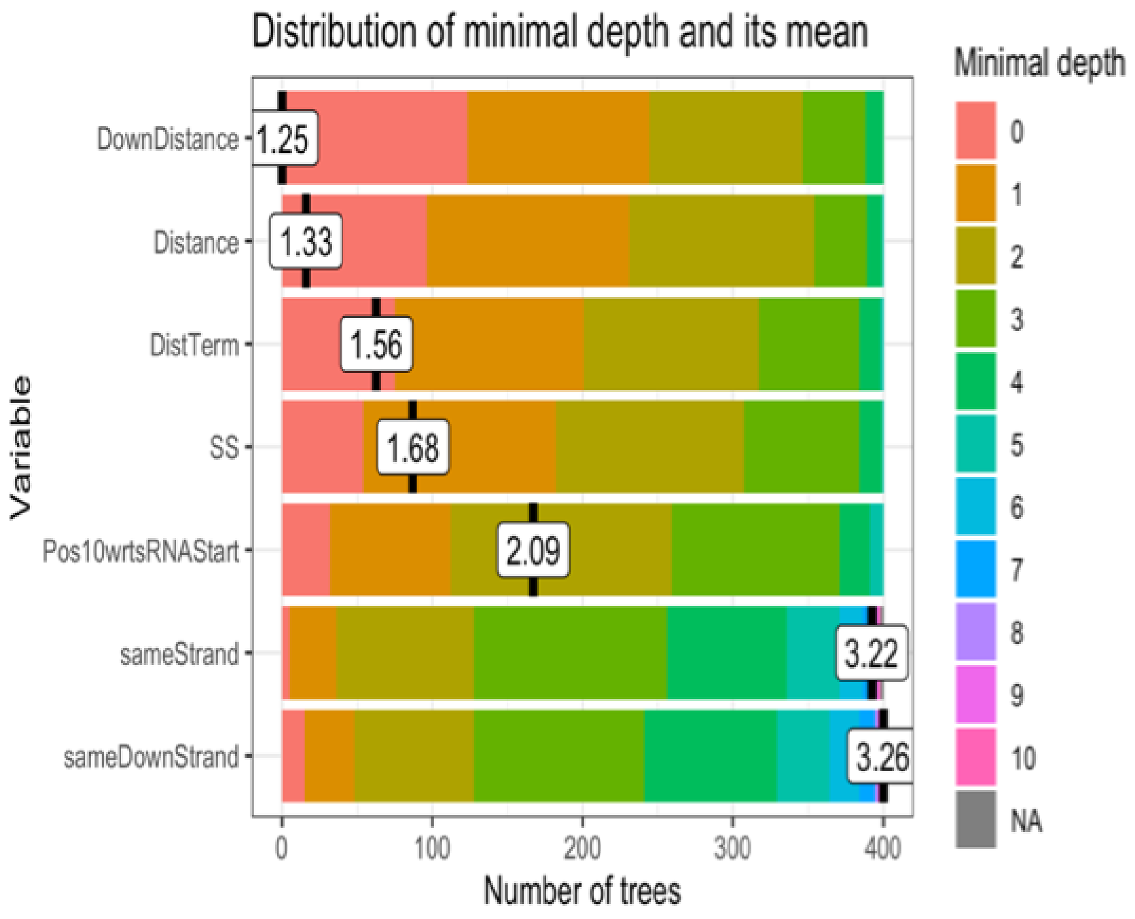


Fig. 4.6: Distribution of minimal depth and its mean for the OrigRF.

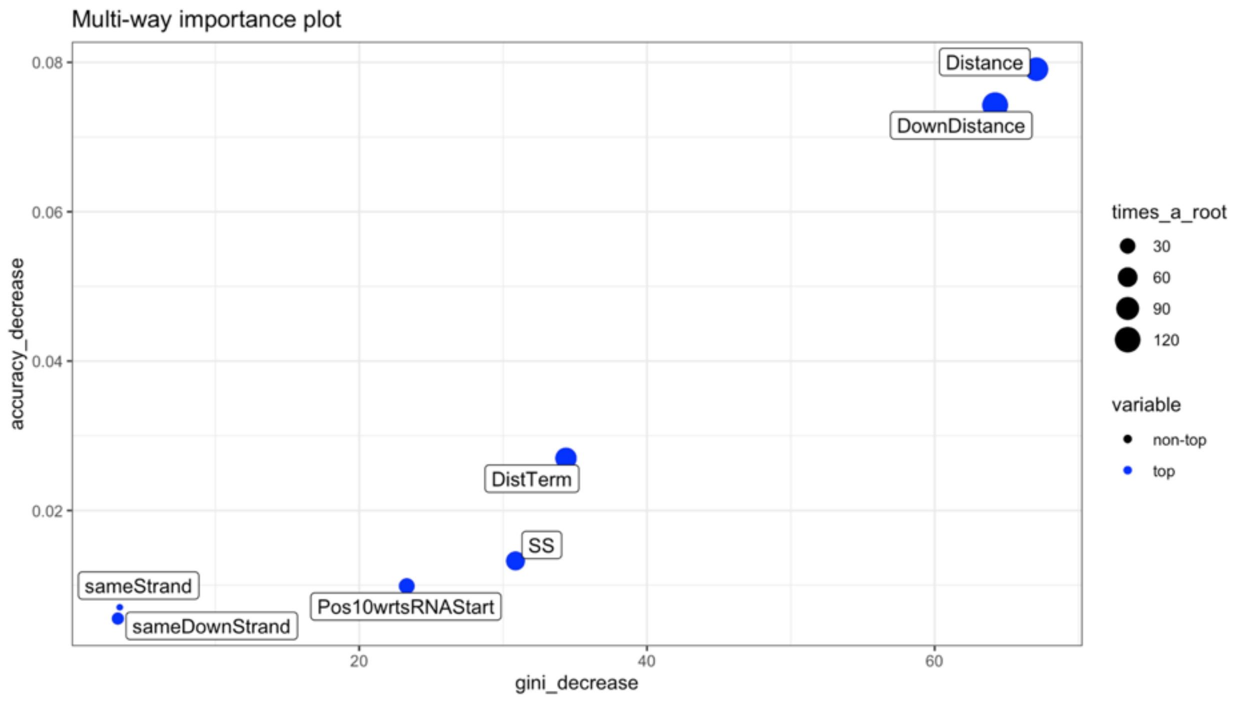


Fig. 4.7: Multi-way importance using the accuracy decrease, Gini decrease, and the times a feature was selected as a root.

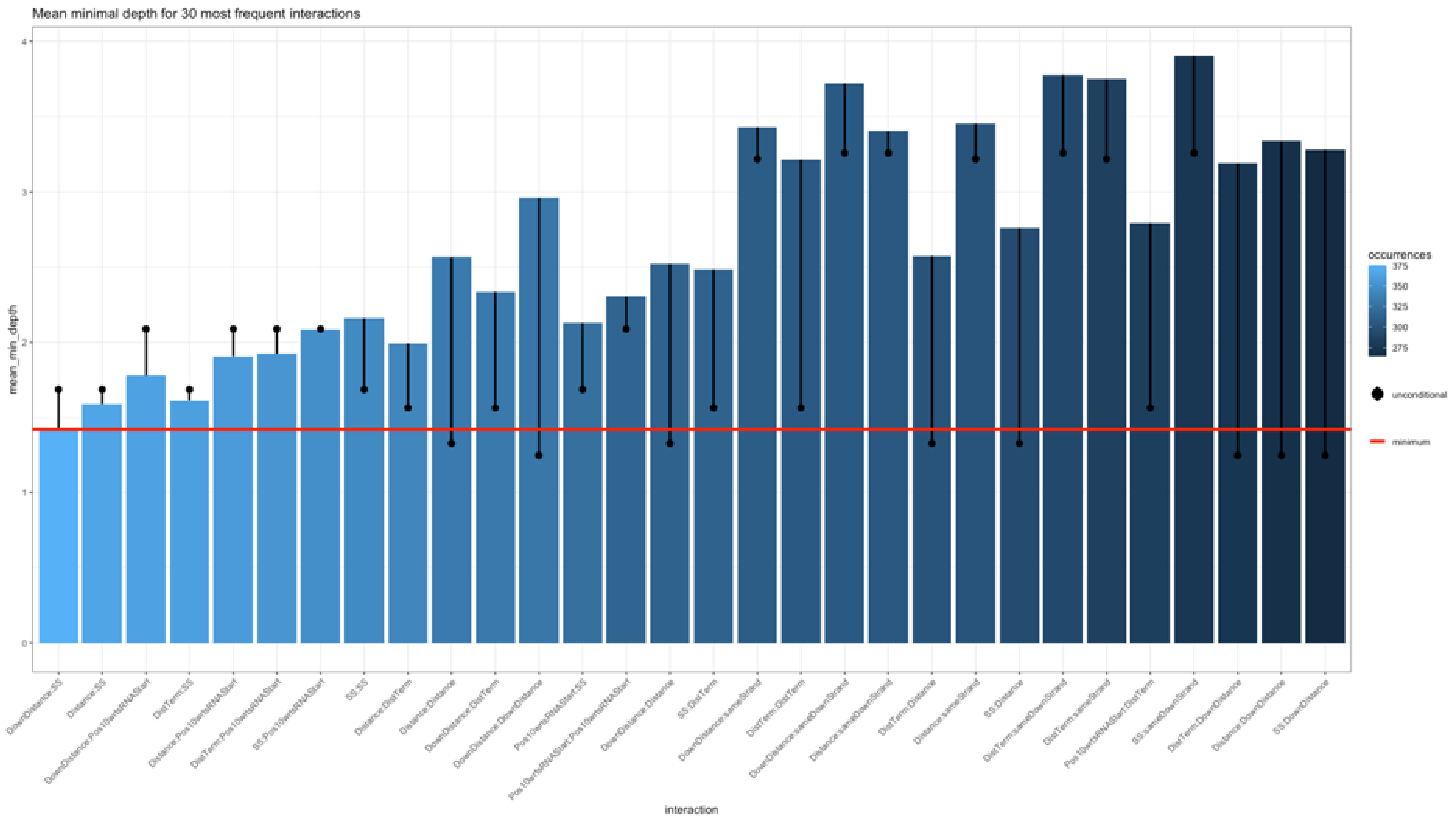


Fig. 4.8: Mean minimal depth for the 30 most frequent interactions organized by decreasing number of occurrences, and where the following top 5 interactions may be identified: DownDistance:SS, Distance:SS, DownDistance:Pos10wrtsRNAStart, DistTerm:SS, and Distance:Pos10wrtsRNAStart.

While only some of the two-dimensional PDPs are presented in this article, all of the PDPs may be found in the supplementary materials. Figure 4.9 contains the DownDistance vs Distance PDP, which supports the plots in Figures 4.1 and 4.3 and further highlights the importance of the distances to the left and right ORFs. Even though Figure 4.4 previously presented a complex polynomial curve, a simpler explanation of how the SS variable contributes to the identification of a bona fide sRNA is provided in Figure 4.10, where the values for the SS contributions seem to be almost perfectly stratified. Nonetheless, both Figure 4.4 and Figure 4.10 support the hypothesis that the lower the free energy of the sRNA predicted secondary structure, the higher the chances of having a bona fide sRNA. Finally, Figure 4.11 corroborates Figure 4.5, and shows that the lower the DistTerm, the better the chances of finding bona fide sRNAs, so long as these values are below 100 nts.

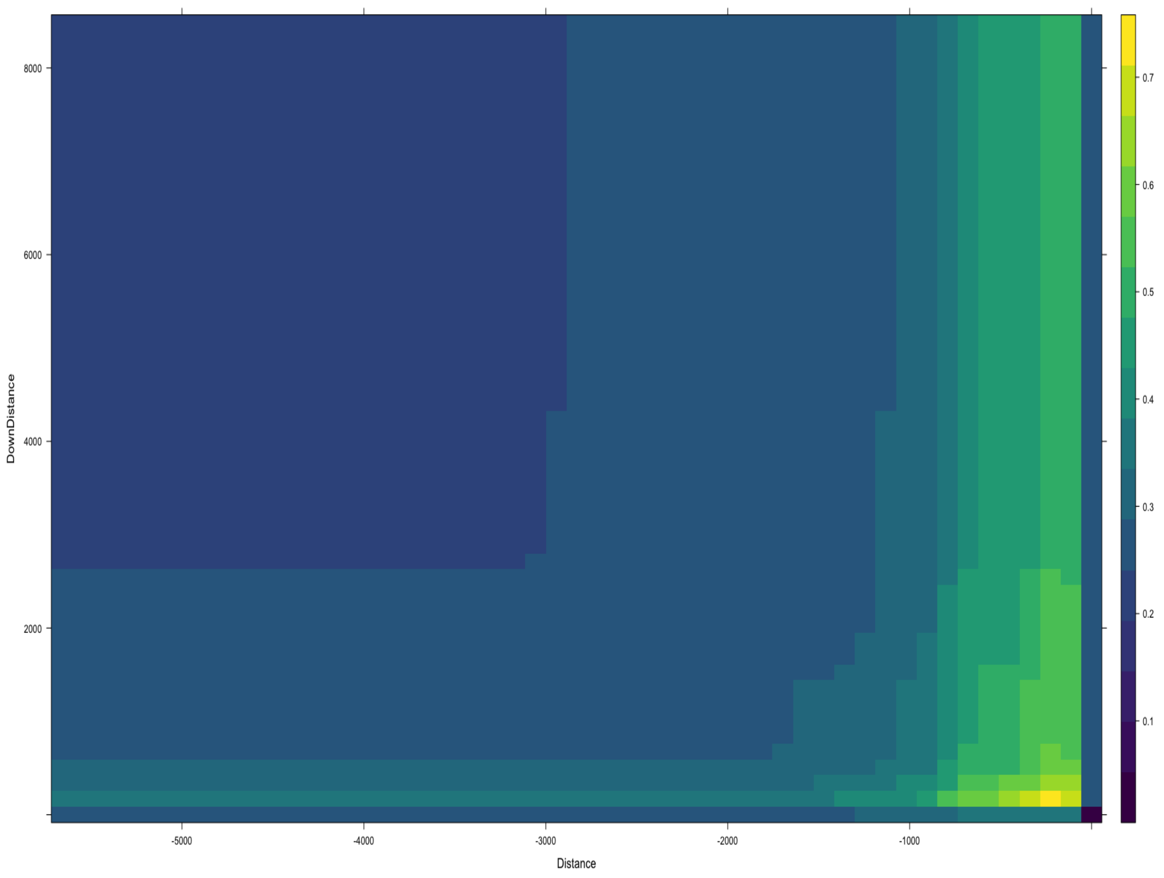


Fig. 4.9: DownDistance vs. Distance PDP showing that while a Distance and DownDistance of 0 yields a very low chance of an instance being an sRNA, instances close to 0 are highly likely to be sRNAs. Color bar indicates the predicted probability of being a bona fide sRNA.

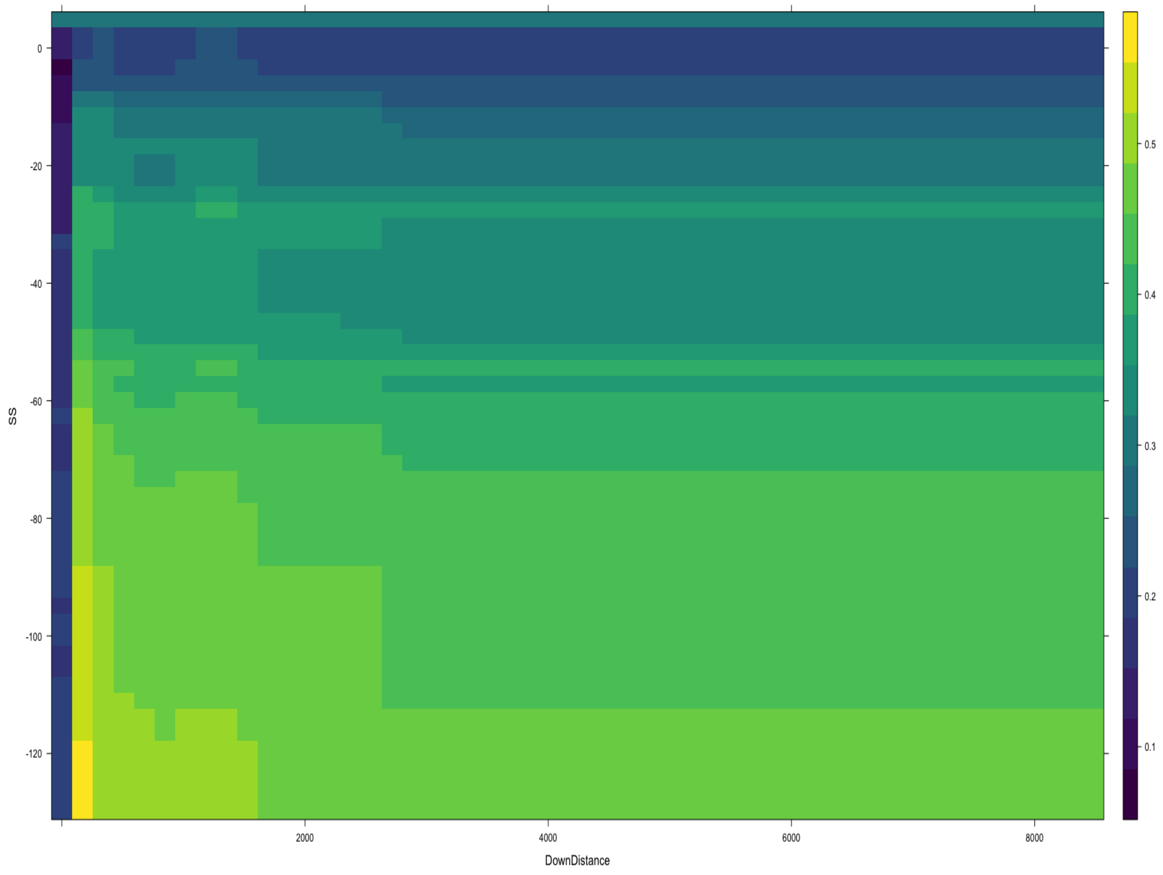


Fig. 4.10: SS vs. DownDistance PDP where distinct levels for the decrease in free energy of the sRNA predicted secondary structure can be somewhat distinguished. Color bar indicates the predicted probability of being a bona fide sRNA.

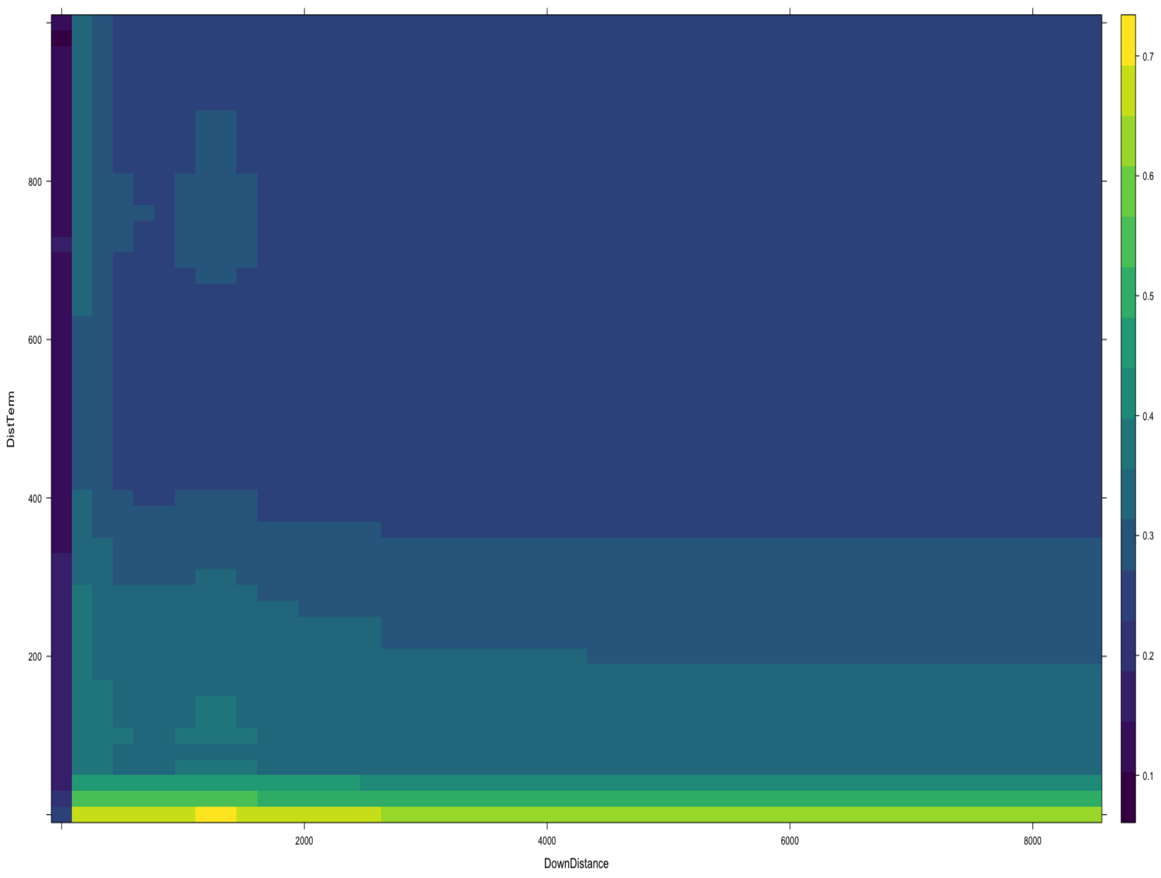


Fig. 4.11: DistTerm vs. DownDistance PDP proving that the distance to the closest predicted Rho-independent terminator is only relevant when its values are under 100. Color bar indicates the predicted probability of being a bona fide sRNA.

LIME and SHAP both provide similar explanations by assigning a value to each feature used by the ML model in every prediction. This value is meant to indicate whether a particular feature contributes or detracts towards the explanation generated by the ML model, meaning values can be positive or negative, and virtually of any magnitude. However, as Figure 4.12 demonstrates, LIME is not always consistent, even when explaining the same instance. The graphs generated by LIME organize the features from highest to lowest absolute value, while the blue bars indicate a positive contribution, and the red bars a negative one. In Figure 4.12, all four plots correspond to the same instance and yet, the first graph has a different ranking for the *DownDistance* feature, and even a completely opposite contribution for *Pos10wrtsRNAStart*. Nonetheless, this does not invalidate LIME's explanations, as there are noticeable consistencies in the explanations when looking into the most contributing factors. For example, the top three contributing factors for the instance in Figure 4.12 always had the same sign, meaning that *DownDistance* was always negative, while *Distance* and *sameStrand* were positive. With this in mind, the main focus in the comparisons between LIME and SHAP were around the sign (positive or negative) of each feature, and the top three factors with highest absolute value in each of the local IMs.

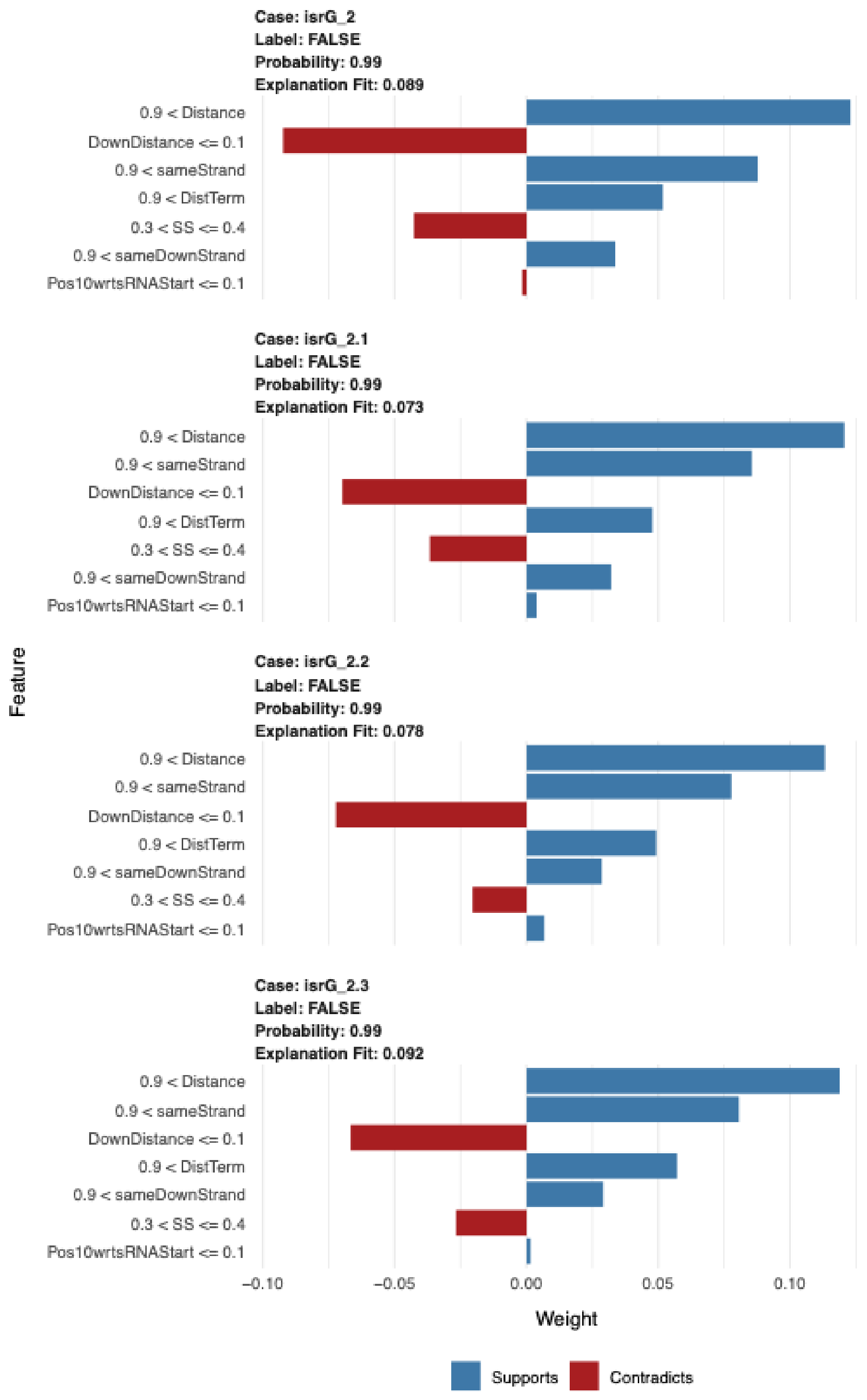
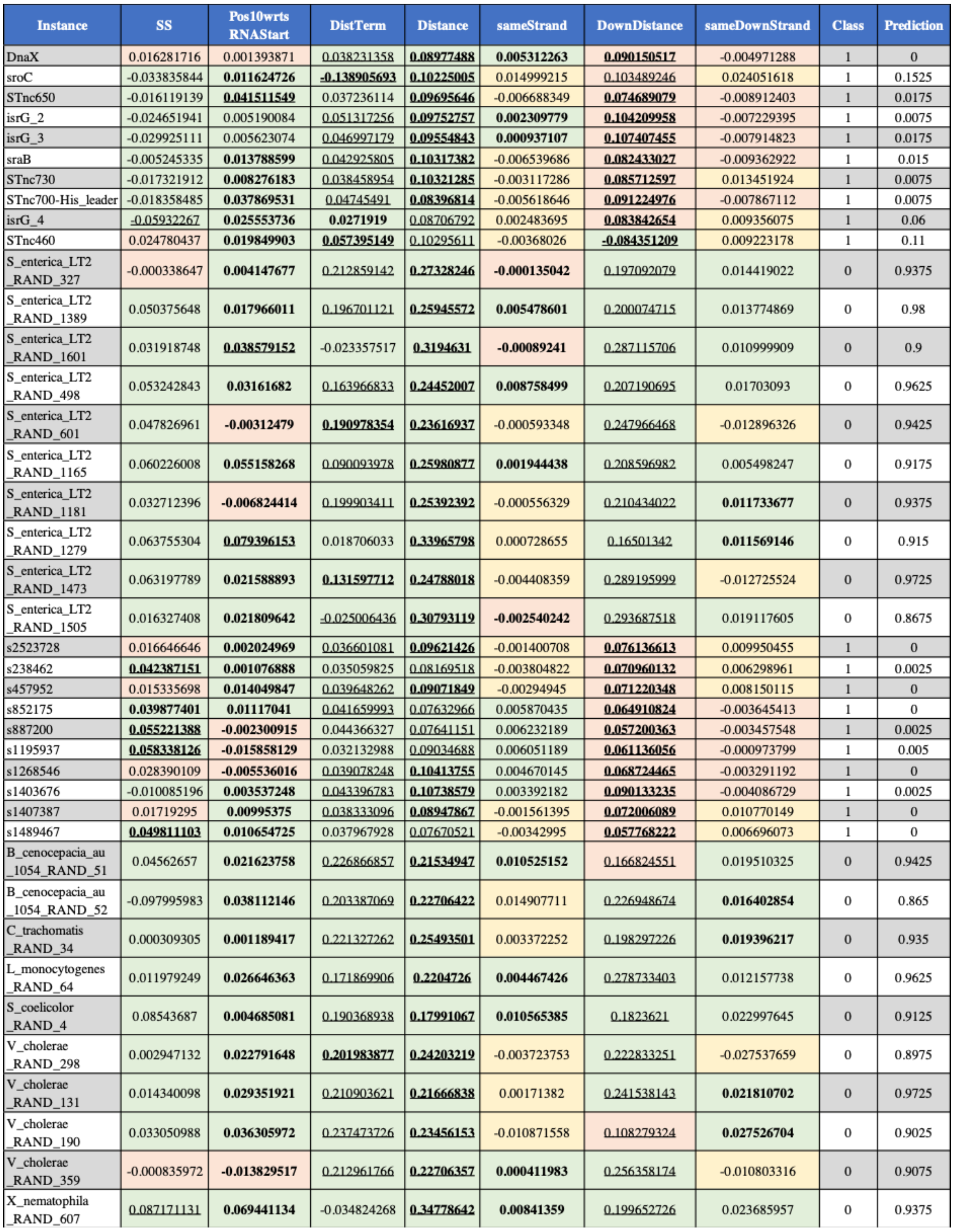


Fig. 4.12: Sample LIME explanations for the IsrG_2 instance. All four cases are the same instance, yet there are obvious differences in the first case as compared to the other three.

The results and comparisons from the local IMs have been summarized in Figure 4.13. The table itself contains the SHAP values for each instance, while the background color in each cell, the bold values, and the underlined numbers indicate the level of agreement with LIME. Cells with a green color indicate that LIME and SHAP both agree in regard to the sign of the contribution (positive or negative), while a red cell would indicate the opposite. Yellow cells indicate features that were not used by LIME for its explanations. All the values that are in bold correspond to the top 3 features with the highest absolute value in contribution as reported by LIME. The underlined values are there to easily identify the top 3 SHAP values based on their absolute value. As an example, one can compare the explanations from Figure 4.12 to the Isrg_2 (fourth row) instance in Figure 4.13. For this instance, both models agreed about the direction, either positive or negative, of the contribution for the *SS, Pos10wrtsRNAStart, DistTerm, Distance* and *sameStrand* parameters, but disagreed about *DownDistance* and *sameDownStrand*. Therefore, the first 5 cells for the IsrG_2 row are colored in green, while the next 2 cells are colored in red. The values in that row for *Distance, sameStrand* and *DownDistance* have also been **bolded** to indicate that those were the features with the highest weights according to LIME. Similarly, the values for *DistTerm, Distance* and *DownDistance* were <u>underlined</u> to better identify the SHAP values with the highest absolute value.



| Instance | SS | Pos10wrts RNAStart | DistTerm | Distance | sameStrand | DownDistance | sameDownStrand | Class | Prediction |
|---|---|---|---|---|---|---|---|---|---|
| DnaX | 0.016281716 | 0.001393871 | 0.038231358 | **0.08977488** | **0.005312263** | **0.090150517** | -0.004971288 | 1 | 0 |
| sroC | -0.033835844 | **0.011624726** | **-0.138905693** | **0.10225005** | 0.014999215 | 0.103489246 | 0.024051618 | 1 | 0.1525 |
| STnc650 | -0.016119139 | **0.041511549** | 0.037236114 | **0.09695646** | -0.006688349 | **0.074689079** | -0.008912403 | 1 | 0.0175 |
| isrG_2 | -0.024651941 | 0.005190084 | 0.051317256 | **0.09752757** | **0.002309779** | **0.104209958** | -0.007229395 | 1 | 0.0075 |
| isrG_3 | -0.029925111 | 0.005623074 | 0.046997179 | **0.09554843** | **0.000937107** | **0.107407455** | -0.007914823 | 1 | 0.0175 |
| sraB | -0.005245335 | **0.013788599** | 0.042925805 | **0.10317382** | -0.006539686 | **0.082433027** | -0.009362922 | 1 | 0.015 |
| STnc730 | -0.017321912 | **0.008276183** | 0.038458954 | **0.10321285** | -0.003117286 | **0.085712597** | 0.013451924 | 1 | 0.0075 |
| STnc700-His_leader | -0.018358485 | **0.037869531** | 0.04745491 | **0.08396814** | -0.005618646 | **0.091224976** | -0.007867112 | 1 | 0.0075 |
| isrG_4 | -0.05932267 | **0.025553736** | **0.0271919** | 0.08706792 | 0.002483695 | **0.083842654** | 0.009356075 | 1 | 0.06 |
| STnc460 | 0.024780437 | **0.019849903** | **0.057395149** | 0.10295611 | -0.00368026 | **-0.084351209** | 0.009223178 | 1 | 0.11 |
| S_enterica_LT2_RAND_327 | -0.000338647 | **0.004147677** | 0.212859142 | **0.27328246** | **-0.000135042** | 0.197092079 | 0.014419022 | 0 | 0.9375 |
| S_enterica_LT2_RAND_1389 | 0.050375648 | **0.017966011** | 0.196701121 | **0.25945572** | **0.005478601** | 0.200074715 | 0.013774869 | 0 | 0.98 |
| S_enterica_LT2_RAND_1601 | 0.031918748 | **0.038579152** | -0.023357517 | **0.3194631** | **-0.00089241** | 0.287115706 | 0.010999909 | 0 | 0.9 |
| S_enterica_LT2_RAND_498 | 0.053242843 | **0.03161682** | 0.163966833 | **0.24452007** | **0.008758499** | 0.207190695 | 0.01703093 | 0 | 0.9625 |
| S_enterica_LT2_RAND_601 | 0.047826961 | **-0.00312479** | **0.190978354** | **0.23616937** | -0.000593348 | 0.247966468 | -0.012896326 | 0 | 0.9425 |
| S_enterica_LT2_RAND_1165 | 0.060226008 | **0.055158268** | 0.090093978 | **0.25980877** | **0.001944438** | 0.208596982 | 0.005498247 | 0 | 0.9175 |
| S_enterica_LT2_RAND_1181 | 0.032712396 | **-0.006824414** | 0.199903411 | **0.25392392** | -0.000556329 | 0.210434022 | **0.011733677** | 0 | 0.9375 |
| S_enterica_LT2_RAND_1279 | 0.063755304 | **0.079396153** | 0.018706033 | **0.33965798** | 0.000728655 | 0.16501342 | **0.011569146** | 0 | 0.915 |
| S_enterica_LT2_RAND_1473 | 0.063197789 | **0.021588893** | **0.131597712** | **0.24788018** | -0.004408359 | 0.289195999 | -0.012725524 | 0 | 0.9725 |
| S_enterica_LT2_RAND_1505 | 0.016327408 | **0.021809642** | -0.025006436 | **0.30793119** | **-0.002540242** | 0.293687518 | 0.019117605 | 0 | 0.8675 |
| s2523728 | 0.016646646 | **0.002024969** | 0.036601081 | **0.09621426** | -0.001400708 | **0.076136613** | 0.009950455 | 1 | 0 |
| s238462 | **0.042387151** | **0.001076888** | 0.035059825 | 0.08169518 | -0.003804822 | **0.070960132** | 0.006298961 | 1 | 0.0025 |
| s457952 | 0.015335698 | **0.014049847** | 0.039648262 | **0.09071849** | -0.00294945 | **0.071220348** | 0.008150115 | 1 | 0 |
| s852175 | **0.039877401** | **0.01117041** | 0.041659993 | 0.07632966 | 0.005870435 | **0.064910824** | -0.003645413 | 1 | 0 |
| s887200 | **0.055221388** | **-0.002300915** | 0.044366327 | 0.07641151 | 0.006232189 | **0.057200363** | -0.003457548 | 1 | 0.0025 |
| s1195937 | **0.058338126** | **-0.015858129** | 0.032132988 | 0.09034688 | 0.006051189 | **0.061136056** | -0.000973799 | 1 | 0.005 |
| s1268546 | 0.028390109 | **-0.005536016** | 0.039078248 | **0.10413755** | 0.004670145 | **0.068724465** | -0.003291192 | 1 | 0 |
| s1403676 | -0.010085196 | **0.003537248** | 0.043396783 | **0.10738579** | 0.003392182 | **0.090133235** | -0.004086729 | 1 | 0.0025 |
| s1407387 | 0.01719295 | **0.00995375** | 0.038333096 | **0.08947867** | -0.001561395 | **0.072006089** | 0.010770149 | 1 | 0 |
| s1489467 | **0.049811103** | **0.010654725** | 0.037967928 | 0.07670521 | -0.00342995 | **0.057768222** | 0.006696073 | 1 | 0 |
| B_cenocepacia_au_1054_RAND_51 | 0.04562657 | **0.021623758** | 0.226866857 | **0.21534947** | **0.010525152** | 0.166824551 | 0.019510325 | 0 | 0.9425 |
| B_cenocepacia_au_1054_RAND_52 | -0.097995983 | **0.038112146** | 0.203387069 | **0.22706422** | 0.014907711 | 0.226948674 | **0.016402854** | 0 | 0.865 |
| C_trachomatis_RAND_34 | 0.000309305 | **0.001189417** | 0.221327262 | **0.25493501** | 0.003372252 | 0.198297226 | **0.019396217** | 0 | 0.935 |
| L_monocytogenes_RAND_64 | 0.011979249 | **0.026646363** | 0.171869906 | **0.2204726** | **0.004467426** | 0.278733403 | 0.012157738 | 0 | 0.9625 |
| S_coelicolor_RAND_4 | 0.08543687 | **0.004685081** | 0.190368938 | **0.17991067** | **0.010565385** | 0.1823621 | 0.022997645 | 0 | 0.9125 |
| V_cholerae_RAND_298 | 0.002947132 | **0.022791648** | **0.201983877** | **0.24203219** | -0.003723753 | 0.222833251 | -0.027537659 | 0 | 0.8975 |
| V_cholerae_RAND_131 | 0.014340098 | **0.029351921** | 0.210903621 | **0.21666838** | 0.00171382 | 0.241538143 | **0.021810702** | 0 | 0.9725 |
| V_cholerae_RAND_190 | 0.033050988 | **0.036305972** | 0.237473726 | **0.23456153** | -0.010871558 | 0.108279324 | **0.027526704** | 0 | 0.9025 |
| V_cholerae_RAND_359 | -0.000835972 | **-0.013829517** | 0.212961766 | **0.22706357** | **0.000411983** | 0.256358174 | -0.010803316 | 0 | 0.9075 |
| X_nematophila_RAND_607 | 0.087171131 | **0.069441134** | -0.034824268 | **0.34778642** | **0.00841359** | 0.199652726 | 0.023685957 | 0 | 0.9375 |

Fig. 4.13: SHAP Values vs. LIME Explanations. The green cells correspond to features that both LIME and SHAP agreed in regard to the sign (positive or negative) of the contribution. The opposite is true for the orange cells, and the yellow cells correspond to features that LIME did not include in its explanation. The values in bold highlight the features with highest absolute value according to LIME. The underlined values are there to easily identify the top 3 features with highest absolute value according to SHAP.

For the most part, based on the individual instances analyzed, most of the IMs agreed with each other. The few exceptions between LIME and SHAP, not including the cases were LIME would omit *sameStrand* and *sameDownStrand*, were predominantly present in the contributions made by the *DownDistance* feature in the false negative instances. However, this inconsistency can easily be explained away by LIME's algorithm and the PDPs in Figures 4.9 and 4.10. In the bottom right corner of Figure 4.9 and the bottom left corner of Figure 4.10, there is a dark blue square indicating that instances in that exact area would be classified as not being sRNAs. Since most of the false negative instances analyzed had a *Distance* and *DownDistance* of 0 nts, they were automatically classified as not being an sRNA. Since LIME would have generated permutations near that area, most of the nearby sectors as shown in Figures 4.9 and 4.10, would have strongly suggested that the instances in question were bona fide sRNAs. For this reason, it is safe to assume that LIME concluded that the *DownDistance* feature actually went against the model's prediction, contrary to the suggestions made by SHAP. LIME and SHAP also agreed that *Distance* and *DownDistance* were among the most contributing factors towards classifying a sequence as a bona fide sRNA. Finally, the only other disagreement between the IMs are in the instances where LIME ranked *sameStrand* or *sameDownStrand* as one of the top contributing factors. Based on the feature importance rankings, the PDPs, and the SHAP values, *sameStrand* and *sameDownStrand* are almost meaningless compared to all the other features. All of the supplementary material may be found in [56].

# Chapter 5

# Conclusions

Different IMs were applied to the RF developed by Eppenhof et al. with the purpose of obtaining new insights about the identification of bona fide bacterial sRNAs. The IMs pointed out that the lower the free energy of the sRNA predicted secondary structure and the smaller the distance to the left and right ORF, the higher the likelihood of finding a bona fide sRNA. Additionally, the use of various IMs proved to be helpful in validating the results even if they did not always completely agree with each other. There is still a lot of potential for future works with this project, considering the number of instances left in the testing data sets that were not analyzed. Additionally, the OrigRF was trained with only 653 instances, even though the testing sample contained over 5000 samples. In other words, even with just the data available in the works of Eppenhof et al., [1] there is great potential to improve the OrigRF by increasing its training data set. It would be interesting to see if a newer and better ML model would generate the same graphs and explanations as the current OrigRF. This project was also limited to 3 LIME, SHAP and PDPs, which means there are plenty of other IMs to try and test out.

# Bibliography

[1] Erik J. J. Eppenhof and Lourdes Peña-Castillo. Prioritizing bona fide bacterial small RNAs with machine learning classifiers. *PeerJ*, 7:e6304, January 2019. Publisher: PeerJ Inc.

[2] Dale Carnegie. When dealing with people, remember you... Dale Carnegie - Forbes Quotes. https://www.forbes.com/quotes/2024/. Library Catalog: www.forbes.com.

[3] Philipp Schmidt and Felix Biessmann. Quantifying Interpretability and Trust in Machine Learning Systems. *arXiv:1901.08558 [cs, stat]*, January 2019. arXiv: 1901.08558.

[4] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. *arXiv:1602.04938 [cs, stat]*, August 2016. arXiv: 1602.04938.

[5] Christoph Molnar. *Interpretable Machine Learning*. 2019. https://christophm.github.io/interpretable-ml-book/.

[6] European Union. EUR-Lex - 32016R0679 - EN - EUR-Lex. Library Catalog: eur-lex.europa.eu.

[7] Abdul Karim, Avinash Mishra, MA Hakim Newton, and Abdul Sattar. Machine Learning Interpretability: A Science rather than a tool. *arXiv:1807.06722 [cs, stat]*, July 2018. arXiv: 1807.06722.

[8] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[9] Wen Patrick Hall Phan, SriSatish Ambati. Ideas on interpreting machine learning, March 2017. Library Catalog: www.oreilly.com.

[10] Gero Szepannek. How Much Can We See? A Note on Quantifying Explainability of Machine Learning Models. *arXiv:1910.13376 [cs, stat]*, December 2019. arXiv: 1910.13376.

[11] Scott Lundberg and Su-In Lee. A Unified Approach to Interpreting Model Predictions. *arXiv:1705.07874 [cs, stat]*, November 2017. arXiv: 1705.07874.

[12] Finale Doshi-Velez and Been Kim. Towards A Rigorous Science of Interpretable Machine Learning. *arXiv:1702.08608 [cs, stat]*, March 2017. arXiv: 1702.08608.

[13] Ando Saabas. treeinterpreter: Package for interpreting scikit-learn's decision tree and random forest predictions. https://github.com/andosa/treeinterpreter.

[14] Interpreting random forests | Diving into data. http://blog.datadive.net/interpreting-random-forests/. Library Catalog: blog.datadive.net.

[15] H2O.ai. Machine Learning Interpretability. https://www.h2o.ai/products-dai-mli/. Library Catalog: www.h2o.ai.

[16] Hugo Yeche, Justin Harrison, and Tess Berthier. UBS: A Dimension-Agnostic Metric for Concept Vector Interpretability Applied to Radiomics. In Kenji Suzuki, Mauricio Reyes, Tanveer Syeda-Mahmood, Ben Glocker, Roland Wiest, Yaniv Gur, Hayit Greenspan, and Anant Madabhushi, editors, *Interpretability of Machine Intelligence in Medical Image Computing and Multimodal Learning for Clinical Decision Support*, Lecture Notes in Computer Science, pages 12–20, Cham, 2019. Springer International Publishing.

[17] Peifei Zhu and Masahiro Ogino. Guideline-Based Additive Explanation for Computer-Aided Diagnosis of Lung Nodules. In Kenji Suzuki, Mauricio Reyes, Tanveer Syeda-Mahmood, Ben Glocker, Roland Wiest, Yaniv Gur, Hayit Greenspan, and Anant Madabhushi, editors, *Interpretability of Machine Intelligence in Medical Image Computing and Multimodal Learning for Clinical Decision Support*, Lecture Notes in Computer Science, pages 39–47, Cham, 2019. Springer International Publishing.

[18] mesameki. Model interpretability in Azure Machine Learning - Azure Machine Learning. https://docs.microsoft.com/en-us/azure/machine-learning/how-to-machine-learning-interpretability. Library Catalog: docs.microsoft.com.

[19] IBM. FAQs. https://cloud.ibm.com/docs/services/ai-openscale?topic=ai-openscale-trainingdata.

[20] Google Inc. Explainable AI. https://cloud.google.com/explainable-ai. Library Catalog: cloud.google.com.

[21] Nick Condry. Meaningful Models: Utilizing Conceptual Structure to Improve Machine Learning Interpretability. *arXiv:1607.00279 [cs, stat]*, July 2016. arXiv: 1607.00279.

[22] Benjamin Bengfort and Rebecca Bilbro. Yellowbrick: Visualizing the Scikit-Learn Model Selection Process. *Journal of Open Source Software*, 4(35):1075, March 2019.

[23] Mikhail Korobov Lopuhin, Konstantin. eli5: Debug machine learning classifiers and explain their predictions. https://github.com/TeamHG-Memex/eli5.

[24] Sebastian Raschka. MLxtend: Providing machine learning and data science utilities and extensions to Python's scientific computing stack. *Journal of Open Source Software*, 3(24):638, April 2018.

[25] Vincent Couteaux, Olivier Nempont, Guillaume Pizaine, and Isabelle Bloch. Towards Interpretability of Segmentation Networks by Analyzing DeepDreams. In Kenji Suzuki, Mauricio Reyes, Tanveer Syeda-Mahmood, Ben Glocker, Roland Wiest, Yaniv Gur, Hayit Greenspan, and Anant Madabhushi, editors, *Interpretability of Machine Intelligence in Medical Image Computing and Multimodal Learning for Clinical Decision Support*, Lecture Notes in Computer Science, pages 56–63, Cham, 2019. Springer International Publishing.

[26] IBM. FAQs. https://cloud.ibm.com/docs/ai-openscale?topic=ai-openscale-wos-faqs.

[27] IBM. Explaining transactions. https://cloud.ibm.com/docs/ai-openscale?topic=ai-openscale-ie-ov.

[28] Jerome H. Friedman. Greedy Function Approximation: A Gradient Boosting Machine. *The Annals of Statistics*, 29(5):1189–1232, 2001. Publisher: Institute of Mathematical Statistics.

[29] UCI Machine Learning Repository: Cervical cancer (Risk Factors) Data Set. https://archive.ics.uci.edu/ml/datasets/Cervical+cancer+%28Risk+Factors%29.

[30] Marco Tulio Correia Ribeiro. marcotcr/lime. https://github.com/marcotcr/lime, April 2020. original-date: 2016-03-15T22:18:10Z.

[31] H2O. Interpretable Machine Learning Using LIME Framework - Kasia Kulma (PhD), Data Scientist, Aviva. https://www.youtube.com/watch?v=CY3t11vuuOM.

[32] David Alvarez-Melis and Tommi S. Jaakkola. On the Robustness of Interpretability Methods. *arXiv:1806.08049 [cs, stat]*, June 2018. arXiv: 1806.08049.

[33] L. S. Shapley. A VALUE FOR N-PERSON GAMES. Technical Report RAND-P-295, RAND CORP SANTA MONICA CA, March 1952.

[34] H2O. Scott Lundberg, Microsoft Research - Explainable Machine Learning with Shapley Values - #H2OWorld. https://www.youtube.com/watch?v=ngOBhhINWb8.

[35] Microsoft Research. Explainable AI for Science and Medicine. https://www.youtube.com/watch?v=B-c8tIgchu0.

[36] Leilani H. Gilpin, David Bau, Ben Z. Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining Explanations: An Overview of Interpretability of Machine Learning. *arXiv:1806.00069 [cs, stat]*, February 2019. arXiv: 1806.00069.

[37] Dong Huk Park, Lisa Anne Hendricks, Zeynep Akata, Anna Rohrbach, Bernt Schiele, Trevor Darrell, and Marcus Rohrbach. Multimodal Explanations: Justifying Decisions and Pointing to the Evidence. *arXiv:1802.08129v1 [cs]*, February 2018. arXiv 1802.08129v1.

[38] Seong Tae Kim, Jae-Hyeok Lee, and Yong Man Ro. Visual evidence for interpreting diagnostic decision of deep neural network in computer-aided diagnosis. In *Medical Imaging 2019: Computer-Aided Diagnosis*, volume 10950, page 109500K. International Society for Optics and Photonics, 2019.

[39] Hyebin Lee, Seong Tae Kim, and Yong Man Ro. Generation of Multimodal Justification Using Visual Word Constraint Model for Explainable Computer-Aided Diagnosis. In Kenji Suzuki, Mauricio Reyes, Tanveer Syeda-Mahmood, Ben Glocker, Roland Wiest, Yaniv Gur, Hayit Greenspan, and Anant Madabhushi, editors, *Interpretability of Machine Intelligence in Medical Image Computing and Multimodal Learning for Clinical Decision Support*, Lecture Notes in Computer Science, pages 21–29, Cham, 2019. Springer International Publishing.

[40] Sérgio Pereira, Raphael Meier, Richard McKinley, Roland Wiest, Victor Alves, Carlos A. Silva, and Mauricio Reyes. Enhancing interpretability of automatically extracted machine learning features: application to a RBM-Random Forest system on brain lesion segmentation. *Medical Image Analysis*, 44:228–244, February 2018.

[41] Radwa El Shawi, Youssef Sherif, Mouaz Al-Mallah, and Sherif Sakr. Interpretability in HealthCare A Comparative Study of Local Machine Learning Interpretability Techniques. In *2019 IEEE 32nd International Symposium on Computer-Based Medical Systems (CBMS)*, pages 275–280, June 2019. ISSN: 2372-9198.

[42] Gajendra Jung Katuwal and Robert Chen. Machine Learning Model Interpretability for Precision Medicine. *arXiv:1610.09045 [q-bio]*, October 2016. arXiv: 1610.09045.

[43] Aleksandra Paluszynska, Przemyslaw Biecek, and Yue Jiang. *randomForestExplainer: Explaining and Visualizing Random Forests in Terms of Variable Importance*, 2019. R package version 0.10.0.

[44] Thomas Lin Pedersen and Michaël Benesty. *lime: Local Interpretable Model-Agnostic Explanations*, 2019. R package version 0.5.1.

[45] Szymon Maksymiuk, Alicja Gosiewska, and Przemyslaw Biecek. *shapper: Wrapper of Python Library 'shap'*, 2019. R package version 0.1.2.

[46] Erin LeDell, Navdeep Gill, Spencer Aiello, Anqi Fu, Arno Candel, Cliff Click, Tom Kraljevic, Tomas Nykodym, Patrick Aboyoun, Michal Kurka, and Michal Malohlava. *h2o: R Interface for the 'H2O' Scalable Machine Learning Platform*, 2020. R package version 3.30.0.4.

[47] H2O.ai. Performance and prediction — h2o 3.30.0.7 documentation. https://docs.h2o.ai/h2o/latest-stable/h2o-docs/performance-and-prediction.html. Library Catalog: www.h2o.ai.

[48] Muhammad Rehman Zafar and Naimul Mefraz Khan. DLIME: A Deterministic Local Interpretable Model-Agnostic Explanations Approach for Computer-Aided Diagnosis Systems. *arXiv:1906.10263 [cs, stat]*, June 2019. arXiv: 1906.10263.

[49] Thomas Lin Pedersen and Michal Benesty. Local interpretable model-agnostic explanations. https://cran.r-project.org/web/packages/lime/lime.pdf, 2019.

[50] Marc P. Grüll, Lourdes Peña-Castillo, Martin E. Mulligan, and Andrew S. Lang. Genome-wide identification and characterization of small RNAs in Rhodobacter capsulatus and identification of small RNAs affected by loss of the response regulator CtrA. *RNA Biology*, 14(7):914–925, July 2017. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/15476286.2017.1306175.

[51] Brandon M. Greenwell. pdp: An R package for constructing partial dependence plots. *The R Journal*, 9(1):421–436, 2017.

[52] Javier Arnedo, Rocío Romero-Zaliz, Igor Zwir, and Coral Del Val. A multi-objective method for robust identification of bacterial small non-coding RNAs. *Bioinformatics*, 30(20):2875–2882, 2014.

[53] Ranjan Kumar Barman, Anirban Mukhopadhyay, and Santasabuj Das. An improved method for identification of small non-coding RNAs in bacteria using support vector machine. *Scientific reports*, 7:46070, 2017.

[54] Xiaojun Lu, Heidi Goodrich-Blair, and Brian Tjaden. Assessing computational tools for the discovery of small RNA genes in bacteria. *RNA*, 17(9):1635–1647, 2011.

[55] Szymon Maksymiuk, Alicja Gosiewska, and Przemyslaw Biecek. Wrapper of python library 'shap'. https://cran.r-project.org/web/packages/shapper/shapper.pdf, 2019.

[56] Carlos Dasaed Salcedo Carreno. Exploring interpretability models for machine learning in prioritizing bona fide bacterial small RNAs supplementary material. https://github.com/dasaed/masters_project_2020, 2019.