Explainability Techniques For Neural Networks Applied to Gene-Disease Associations Discovery

Percorso di Eccellenza

Gianluca De Carlo (mat. 1805894)

A.Y. 2021/2022

Contents

1 Introduction

Group of well known genes, called disease genes, are responsible of causing complex diseases. Identifying the associations between diseases and their causal genes is a critical problem concerning human health; indeed, understanding the mechanisms of diseases, can result in their better diagnosis, treatment, and prevention.

During the recent years, many powerful techniques have been developed to find disease-gene associations using different approaches, like text mining of a specific disease's literature or by employing Machine Learning (ML) algorithms, exploiting a wide variety of models, training datasets and feature extraction mechanisms.

Many efforts are devoted to the discovery of genes involved with specific diseases, and powerful techniques are used to detect numerous candidate genes. However, the experimental validation of candidate genes is an expensive and resource-consuming task. Therefore, it is important to develop computational approaches able to focus expensive laboratory tests only on the most promising candidates.

The results of a ML method applied to the **Gene-Disease association** (GDA) problem will be hand in to scientist working in laboratories where the associations are validated, and usually these scientist are not experts in machine learning. So that costly experiments are performed on the results of a ML algorithm, it is necessary that the scientists are able to trust the model. To achieve such goal, it is required to understand how the model reached a certain decision, but most deep learning models are designed without considering later interpretability of their results; this leads into treating them as black boxes.

Without understanding the relationships behind the inputs and the predictions, ML models cannot be understood and fully trusted, which prevents their applications in critical areas [1].

Explainable AI is a recent research area, where the goal is to dive deep into a non-interpretable model, to understand why it reached a determined prediction. Employing explainability has two major benefits: explain the reasons behind a model's prediction even to non-ML experts, making the model more trustworthy, and debugging the learning process of the model.

The use of explainability in this work is two-folded:

- Provide an insight on the model's predictions. By exploiting the explanation's results, it can be possible to analyze the features that are most important for a particular disease; moreover, explainability will be used to understand how different models learn on the same dataset.
- Explainability methods are then used as a tool to perform **dimensionality reduction**. **Curse of dimensionality** is a widespread problem in Data Science. By using the explanations, it is possible to uncover if there are some features that are not relevant for the purpose of classification, hence, that can be removed, reducing the size of each vector in the dataset, and consequently, bringing the dataset to a lower-dimensional space.

2 NeDBIT Features

The **NeDBIT** features, introduced by Paola Stolfi *et al.* [2], are used to perform the experiments in this work. The authors of NIAPU treated GDA as a Positive-Unlabeled (PU) learning problem, where the known positive genes (P) are labeled as such, while the remaining part of the dataset is unlabeled (U). The set of unlabelled instances is assumed to be a contaminated set of negative instances. In the specific case of gene discovery, this contamination is given by the possibility of the negative instances to contain not yet discovered positive genes.

The study has been performed on two datasets: Human protein-protein interactions (PPI) from BioGRID and GDAs from DisGeNET. With a focus on five pathologies: malignant neoplasm of breast (disease ID C0006142, 1074 genes), schizophrenia (C0036341, 883 genes), liver cirrhosis (C0023893, 774 genes), colorectal carcinoma (C0009402, 702 genes) and malignant neoplasm of prostate (C0376358, 616 genes).

The features used to characterize each gene are: degree, ring, NetRank, NetShort, HeatDiff, InfoDiff.

Where the **degree** of a gene is the number of connections that it has to other genes in the PPI network. The **ring** represents the number of hops between the gene and the closest seed gene. HeatDiff and InfoDiff are two network diffusion-based features, and NetRank and NetShort are two biology-informed topological metrics.

HeatDiff and **InfoDiff** are obtained by a heat diffusion process over networks, that is among the most used process for disease gene prioritisation and prediction. These processes are based on the heat diffusion equation in normal environments, where the heat tends to spread from the warmer area to the colder one, and in the same way, they want to represent how the information spreads in the network, from important (warmer) nodes, to less important (colder) nodes.

HeatDiff is computed starting with a distribution of weights, with positive values only on the seed genes, their evolution is determined by using the diffusion equation on graphs:

$$s'(t) + \mathcal{L}s(t) = 0 \tag{1}$$

Where $\mathcal{L} = D - A$ is the Graph Laplacian Matrix. D is the diagonal matrix of the degrees of the nodes, and A is the adjacency matrix of the PPI network. The weights at a time t are given by

$$s(t) = exp(-\mathcal{L}t)s(0) \tag{2}$$

where exp is the exponential of the matrix, and $s(0)$ is the initial weight distribution: $s(0) = s_i$ for seed genes, where s_i is the GDA score of the gene.

By using $\mathcal{L}_b = I - D^{-1}A$, another version of the Graph Laplacian matrix, it is possible to compute the InfoDiff feature. Unlike $\mathcal{L}$, that diffuses the same amount of score for each link, $\mathcal{L}_b$ diffuses the information for each node. This implies a different short time behaviour of the diffusion process on the graph.

In the first case, the diffusion process proceeds faster from well-connected nodes with high weight. Whereas in the second case, the same amount of weight diffuses from each node and therefore there is no great difference between well-connected and poorly connected nodes with regard to the outflow of the weight. Similarly to HeatDiff, it is computed as follows:

$$s(t) = exp(-\mathcal{L}_b t)s(0) \tag{3}$$

The **NetRank** measure is based on the concept of ring structure, generalized to a set of seed nodes. Starting from seed nodes, the following partition of the graph in subgraphs, or rings, is introduced

$$R(l) = \{j \in V \mid min_{i \in \Sigma}(l_{ij}) = l\} \tag{4}$$

Where l_{ij} is the length of the shortest path between i and j. $R(l)$ contains all the non-seed genes with a minimal distance l from at least one seed node.

Starting from an initial rank $\hat{r}_i$, equal to the GDA score, the rank of a seed gene i, with k_i neighbors, is then computed by combining the initial rank with the average rank of the neighbours:

$$r_i(0) = \alpha\hat{r}_i + (1-\alpha)\frac{1}{k_i}\sum_{j|A_{ij}\neq 0}\hat{r}_j \tag{5}$$

Allowing seed nodes that have many other seed nodes as neighbors to have the highest ranks.

Then, the rank of non-seed genes in the ring l is defined by

$$r_i(l) = l + \frac{1}{k_i}(\sum_{j \in R_i(l-1)}(r_j(l-1) - (l-1)) + \sum_{j \in O_i}\hat{r}_j) \tag{6}$$

With equation 6, the rank of non-seed gene i is computed by summing to the ring level l the average rank of the neighbors, by dividing them into two groups: the ones in the previous ring (i.e., $R_i(l-1)$), and the ones in the same ring and in the next one, grouped in the set O_i. This measure rewards non-seed nodes that are close to high-ranking nodes in the previous ring and that are linked with a few nodes of the same or higher ring.

The **NetShort** measure is based on the idea that a generic node is topologically important for a disease if a large number of seed nodes must be traversed to reach it. Therefore, the measure is implemented computing the weighted average shortest paths reaching the node under consideration, where each link is weighted favouring links connecting seed nodes and penalising links connecting non-seed nodes. The edges' weights $w_{i,j}$ are computed by the following equation:

$$w_{i,j} = A_{i,j}\frac{2}{\tilde{s_i} + \tilde{s_j}}, \text{ where } \tilde{s_i} = \begin{cases} \frac{s_i}{max\ s} \text{ if } i \in \Sigma \\ \alpha\frac{min\ s}{max\ s} \text{ if } i \notin \Sigma \end{cases} \tag{7}$$

Where $min/max\, s$ are respectively the minimum and maximum GDA scores of seed genes, α is the penalisation parameter given to non-seed genes, and Σ is the set of seed genes.

Finally, the NetShot value of a non-seed node i is defined as

$$NS_i = \sum_{j \neq i} \frac{1}{d_{ij}} \tag{8}$$

Where d_{ij} is the length of the weighted shortest path from i to j.

Table 1 shows the range and the density of the values for each feature.

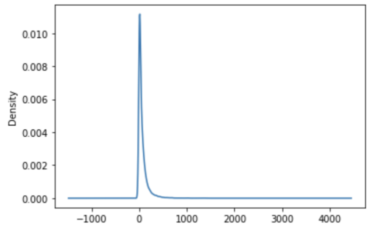
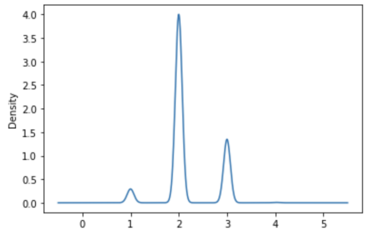
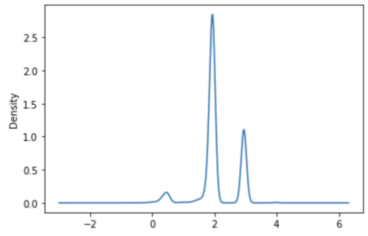
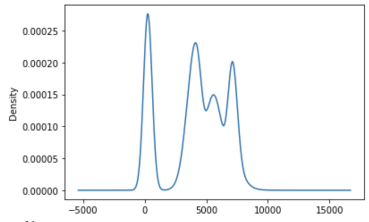
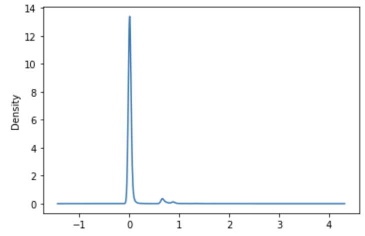
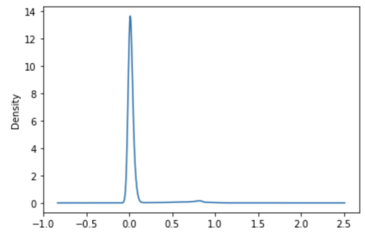
Feature name	Min. value	Max. value	Density plot
Degree	1	2968	
Ring	1	4	
NetRank	−0.67	3.99	
NetShort	134.24	11176.67	
HeatDiff	3.71e-07	2.88	
InfoDiff	1.61e-07	1.67	

Table 1: Range and density of values for each feature

3 Models

Three different classifiers have been trained on the **NeDBIT** features:

- Support Vector Machine (SVM) [3]
- Random Forest Classifier (RFC) [4]
- Multi-layer Perceptron (MLP) [5]

to analyze how each model performs and to select the best one.

SVM is a supervised learning model for solving classification and regression problems. The main idea behind SVM is to create a hyperplane which separates the dataset into classes.

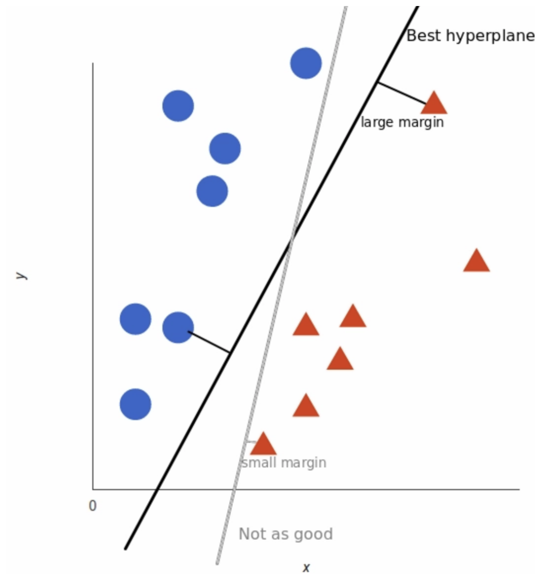


Figure 1: Visual representation of an hyperplane learned by a SVM model

As SVM, **Random Forest Classifier** is a supervised learning algorithm that can be used for classification and regression problems. Random Forests can be regarded as an **ensemble method**; indeed, they are comprised of decision trees, that are the building blocks. The actual prediction of a random forest is chosen via majority voting between the predictions of the decision trees.

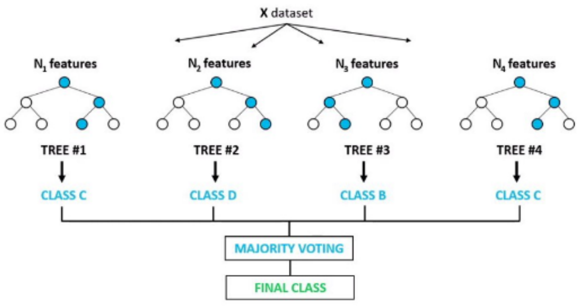


Figure 2: Ensemble of decision trees forming a random forest

Unlike SVM and Random Forest, **Multi-Layer Perceptron** is a Deep-Learning algorithm, meaning that it is built using different layers: the input/output layers plus the hidden layers.

The input layer receives the input signal to be processed, the output layer performs the required task, such as regression or classification, and an arbitrary number of hidden layers are placed in between the input and output layers, forming the true computational engine of the MLP.

The layers are composed by neurons, whose operation is regulated by an activation function. If the weighted sum of the inputs coming into a neuron exceed a certain threshold, then the neuron is activated. The values of the weights associated to each feature are trained by a back propagation learning algorithm.

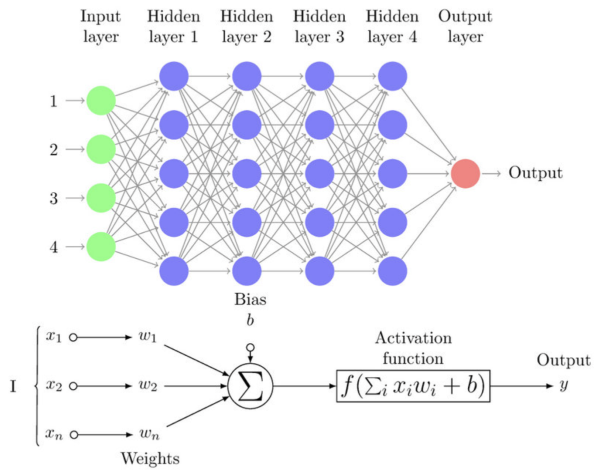


Figure 3: MLP with 4 hidden layers

4 Explainability Methods

On the models described in 3, two different explainability methods have been used: Local Interpretable Model-agnostic Explanations (**Lime**) [6] and SHapley Additive exPlanation (**Shap**) [7].

4.1 LIME

The overall goal of LIME is to identify an interpretable model over the interpretable representation that is locally faithful to the classifier. The explanation produced by LIME is obtained by solving the optimization problem described by the following equation:

$$\xi(x) = argmin_{g \in G} \; \mathcal{L}(f, g, \pi_x) + \Omega(x) \tag{9}$$

Where $x \in \mathbb{R}^d$ is the original representation of an instance being explained, and $x' \in \{0,1\}^{d'}$ is its interpretable representation. G is the class of interpretable models, and the domain of g is $\{0,1\}^{d'}$, hence, g works over the presence/absence of interpretable components. $f : \mathbb{R}^d \to \mathbb{R}$ is the model to be explained and π_x is a proximity measure between an instance z to x, to define a locality around x. $\mathcal{L}(f, g, \pi_x)$ is a loss functions that measures how unfaithful g is in approximating f in the locality defined by π_x. Since not every model is interpretable in the same way, it is used $\Omega(g)$ as a metric for the complexity of the explanations of g.

Since a desired property of an explainer is to be **model-agnostic**, the loss function $\mathcal{L}$ is minimized without making any assumption on f.

In order to understand the local behavior of f, nonzero elements of x' are draw uniformly at random, generating **perturbed samples**. Given a perturbed sample $z' \in \{0,1\}^{d'}$, it is recovered its original representation $z \in \mathbb{R}^d$ and obtained $f(z)$, which is used as label for the explanation model. Given the dataset $\mathcal{Z}$ of perturbed samples, Eq. 9 is optimized to compute the explanation $\xi(x)$.

4.2 SHAP

SHAP is a game theoretic approach to explain the output of any Machine Learning model [8]. In game theory are implied a 'game' and some 'players'; when applying Shapley Values to ML, the 'game' is reproducing the outcome of the model, while the 'players' are the features learned by the model [9]. To generate the explanations, SHAP computes the average marginal contribution that each feature bring to the prediction made by the model.

The base idea behind Shapley values is that the outcome of each possible combination of features $f \in F$, where F is the set of all features, should be considered in order to understand the importance of each feature.

The number of combinations is $2^{|F|}$, and for each combination SHAP must train a distinct model. Each model is equivalent to the other, differing only on the set of features they are trained on. Since the exact computation of SHAP values is challenging, one solution is to compute the contributions for only a few samples of all the possible coalitions.

5 Results

In this chapter are shown the results obtained by the three different models (SVM, RFC, MLP) on the diseases 'malignant neoplasm of breast' (disease ID C0006142) and 'schizophrenia' (C0036341), with the binary classification task, where the models are asked to classify if a gene is positive (P) or unlabelled (U); then, the two explainability methods (Lime, Shap) are applied to the trained models to analyze the differences in the learning processes on the same dataset. Finally, the explanations are used to find features that are not relevant for the classification process, hence that can be removed, and the models are tested on the smaller dataset.

5.1 Classification results

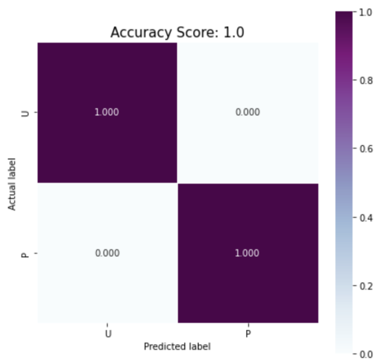
- Malignant neoplasm of breast

SVM

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      5621
           1       1.00      1.00      1.00       308

    accuracy                           1.00      5929
   macro avg       1.00      1.00      1.00      5929
weighted avg       1.00      1.00      1.00      5929
```

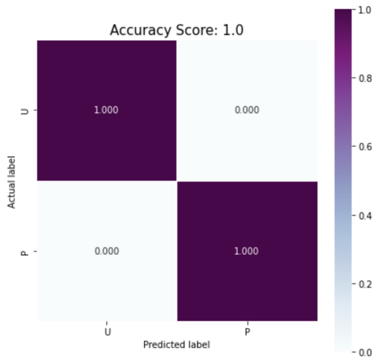


RFC

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      5621
           1       1.00      1.00      1.00       308

    accuracy                           1.00      5929
   macro avg       1.00      1.00      1.00      5929
weighted avg       1.00      1.00      1.00      5929
```

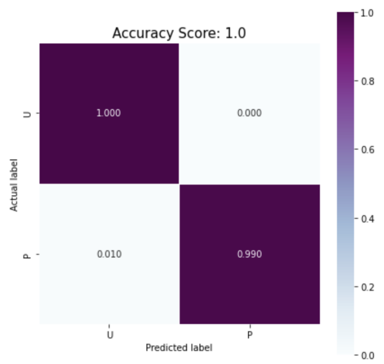


MLP

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      5621
           1       1.00      0.99      1.00       308

    accuracy                           1.00      5929
   macro avg       1.00      1.00      1.00      5929
weighted avg       1.00      1.00      1.00      5929
```



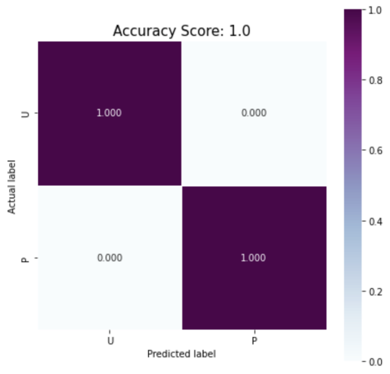
- Schizophrenia

SVM

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      5679
           1       1.00      1.00      1.00       250

    accuracy                           1.00      5929
   macro avg       1.00      1.00      1.00      5929
weighted avg       1.00      1.00      1.00      5929
```

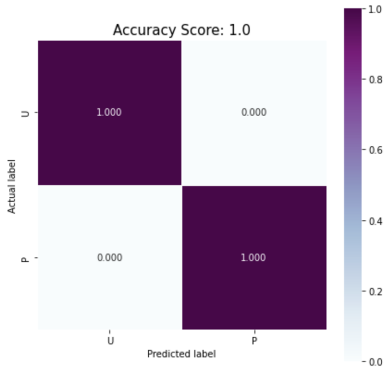


RFC

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      5679
           1       1.00      1.00      1.00       250

    accuracy                           1.00      5929
   macro avg       1.00      1.00      1.00      5929
weighted avg       1.00      1.00      1.00      5929
```

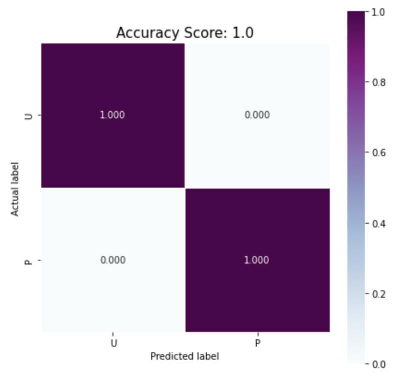


MLP

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      5679
           1       1.00      1.00      1.00       250

    accuracy                           1.00      5929
   macro avg       1.00      1.00      1.00      5929
weighted avg       1.00      1.00      1.00      5929
```



5.2 Explanations

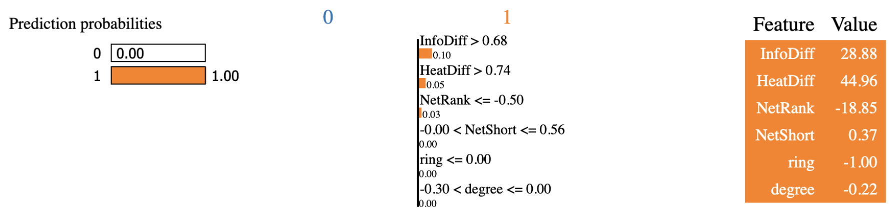
As we can see from the results, the models learned with very high accuracies the classification task. In this case it makes sense to investigate the results of the explanations to compare the learning processes.

5.2.1 Lime

The local explanations conducted by Lime refer to the same sample for each disease. The following plots allow to understand how the different models learned on the same sample.

SVM

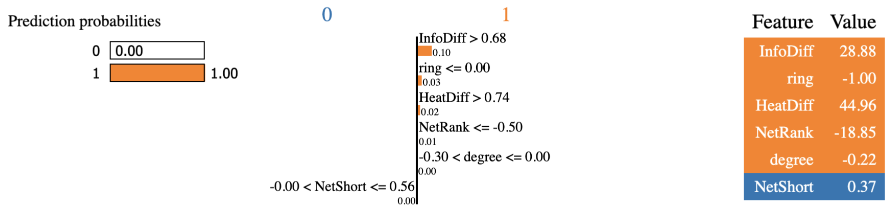
```
Intercept -0.010348769057692483
Prediction_local [0.16703822]
Right: 0.9999933132048965
```



Feature	Value
InfoDiff	28.88
HeatDiff	44.96
NetRank	-18.85
NetShort	0.37
ring	-1.00
degree	-0.22

RFC

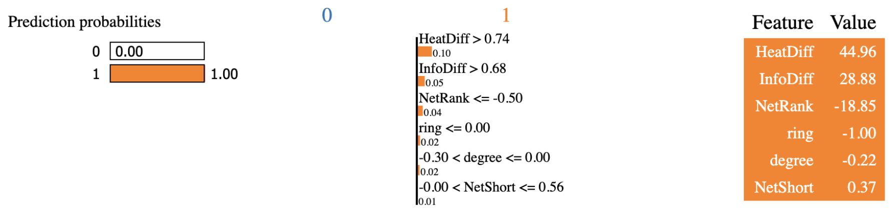
```
Intercept -0.0023593424889832987
Prediction_local [0.1630883]
Right: 1.0
```



Feature	Value
InfoDiff	28.88
ring	-1.00
HeatDiff	44.96
NetRank	-18.85
degree	-0.22
NetShort	0.37

MLP

```
Intercept -0.04960652973619277
Prediction_local [0.19130348]
Right: 0.9999124439682101
```



Feature	Value
HeatDiff	44.96
InfoDiff	28.88
NetRank	-18.85
ring	-1.00
degree	-0.22
NetShort	0.37

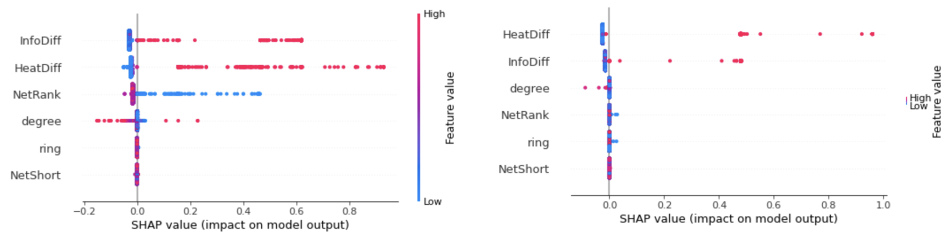
By analyzing the local explanations on the same sample, it can be noted how the models learned the features with the same importance.

5.2.2 Shap

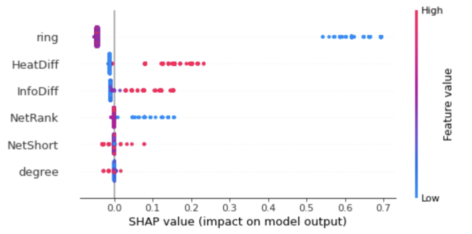
Shap allows to gain an insight on the global behavior of the model. By looking at the **beeswarm** plot [10], it is possible to understand the relation between a feature's value and its impact on the model prediction.

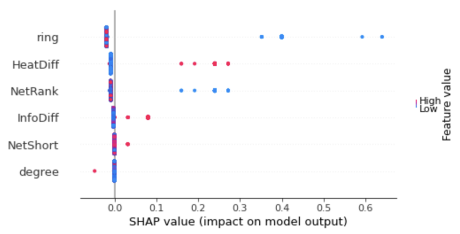
For each model type, the plot on the left refers to the Malignant neoplasm of breast while the plots on the right to the Schizophrenia. The following plots allow to understand, for each model, the features' relevance for different disease, mirroring how genes associated to different diseases may behave differently.

- SVM

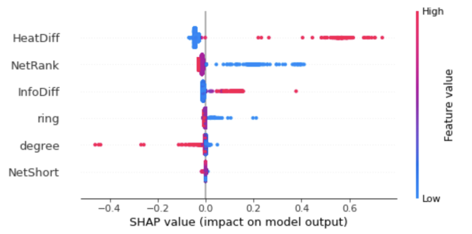


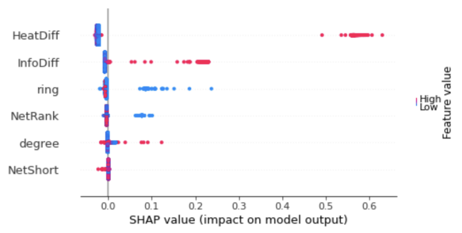
- RFC





- MLP





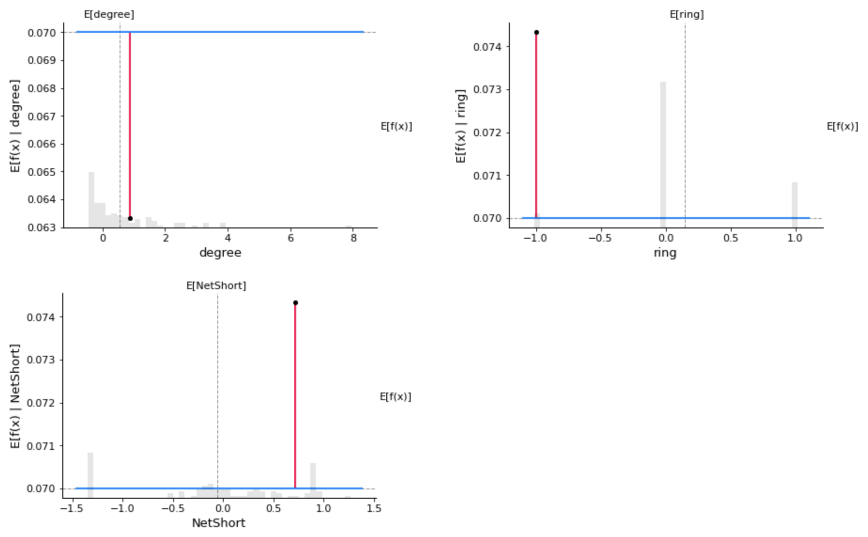
5.3 Dimensionality Reduction

In this chapter, it is investigated a possible solution of the **Curse of Dimensionality** through the use of explainability.

To understand a feature's importance in a model it is necessary to understand both how changing that feature impacts the model's output, and also the distribution of that feature's values. To visualize this, the **partial dependence plots** from Shap [11] can be used. The gray horizontal line in the plots represents the expected value of the model, the vertical gray line represents the average value of a feature, while the blue line represents the partial dependence plot (which is the average value of the model output when a feature has a fixed value).

The basic idea is that if a change in values of a feature does not affect the expected output of the model, then the feature is not influential in the learning process, hence, can be removed from the dataset.

For example, by analyzing the partial dependence plots of the SVM model trained on the Malignant neoplasm of breast disease, can be noted three features whose trend does not affect the model output:



From the plots above, it is observable how the expected value of the prediction function does not change when the features' values vary in all the possible range. Indeed, by removing 'degree', 'ring' and 'NetShort' from the dataset the performance of the model does not get worse, neither of a small drop.

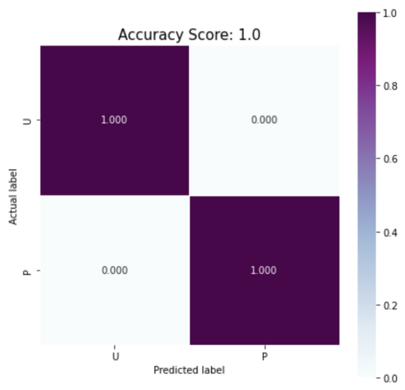


Figure 7: Confusion matrix of SVM after training on the reduced dataset

Moreover, by removing the three features from the dataset, the memory usage of the dataset is dropped by 50% (926.4 KB original dataset, 463.3 KB reduced dataset). On a small dataset, like the one the models have been trained on, the gain is not considerable, but achieving such dimensionality reduction on bigger dataset will also imply faster training times, improving efficiency not only in memory usage but also in power consumption. [1]

[1] Project's source code: https://github.com/GiDeCarlo/excellencepath2022

6 Conclusions

With this work it has been conducted an insight on the NeDBIT features, designed to satisfy the separability and smoothness assumptions, that allow three simple models to achieve very high performances on a hard task, such as GDA. Then, two of the most relevant explainability methods have been analyzed and then applied to the trained models, uncovering how they have learned from the same dataset in a similar way, even if they were using three different learning approaches.

At the end, the explanations have been used to achieve another goal, very different from what they were designed for. Indeed, by analyzing the importance of several features in the classification task by using the Shapley values, non-relevant features have been unveiled and removed from the dataset, reducing its memory usage by 50%, without impacting the models' performances.

References

[1] Hao Yuan et al. “On Explainability of Graph Neural Networks via Subgraph Explorations”. In: (2021). URL: http://arxiv.org/abs/2102.05152.

[2] Paola Stolfi et al. “Adaptive Positive-Unlabelled Learning via Markov Diffusion”. In: *CoRR* abs/2108.06158 (2021). arXiv: 2108.06158. URL: https://arxiv.org/abs/2108.06158.

[3] *Scikit Learn SVM documentation.* https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html.

[4] *Scikit Learn RFC documentation.* https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html.

[5] *Scikit Learn MLP documentation.* https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html.

[6] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. “”Why Should I Trust You?”: Explaining the Predictions of Any Classifier”. In: *CoRR* abs/1602.04938 (2016). arXiv: 1602.04938. URL: http://arxiv.org/abs/1602.04938.

[7] Scott M. Lundberg and Su-In Lee. “A unified approach to interpreting model predictions”. In: *CoRR* abs/1705.07874 (2017). arXiv: 1705.07874. URL: http://arxiv.org/abs/1705.07874.

[8] *SHAP documentation.* https://shap.readthedocs.io/en/latest/index.html.

[9] *SHAP Values Explained Exactly How You Wished Someone Explained to You.* https://towardsdatascience.com/shap-explained-the-way-i-wish-someone-explained-it-to-me-ab81cc69ef30.

[10] *SHAP Beeswarm plot documentation.* https://shap.readthedocs.io/en/latest/example_notebooks/api_examples/plots/beeswarm.html.

[11] *SHAP Beeswarm plot documentation.* https://shap.readthedocs.io/en/latest/example_notebooks/overviews/An%20introduction%20to%20explainable%20AI%20with%20Shapley%20values.html#A-more-complete-picture-using-partial-dependence-plots.