Assignment 5: Report

Feature Selection

Danny Rogaar (s2393344)
Daan Opheikens (s3038416)
Panagiotis Giagkoulas (s3423883)
Saim Eser Comak (s3432548)
Carlos Huerta (s3743071)
Emile Muller (s3787915)

Group 14

October 7, 2018

1 Introduction

Handling large amounts of raw, and usually user-generated data, requires proper pre-processing of said data, so that they can be efficiently used for data mining, modeling and machine learning. Pre-processing is a broad field of data science and one major aspect of it is feature selection. When faced with data consisting of multiple variables/features, applying feature selection can improve the performance of our models by using the most relevant features to the objective. Moreover, with high dimensionality and computationally complex models, it may even be required to reduce the number of variables and features. Successfully applying feature selection leads to the use of simpler models, which require much less time and resources to train, reduce the possibility of over-fitting and contributes to avoiding the curse of dimensionality.

2 Methods

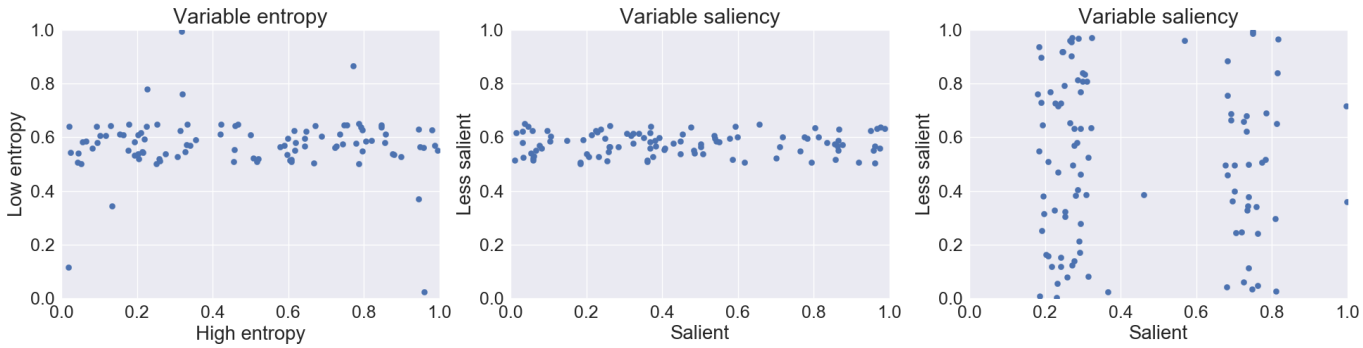
A complementary jupyter notebook with illustrative examples can be accessed via:
https://github.com/RUG-IDS/team-14/tree/master-2/Assignment_5

2.1 Ranking variables

Typical feature selection algorithms rank variables or features before selecting a subset of them. The ranking procedure is performed as it is often a relatively easy step without taking up the bulk of computational resources. Additionally, the ranking has a positive effect on variance as discussed in [1]. The resulting ranked variables can then be filtered according to their relevance which makes up a forward selection phase towards e.g a computationally intensive classifier.

Although variable ranking focuses on individual variables, interactions between those may occur. Simply ranking the variables then, does not account for interactions between variables and instead ranking should be done on the features. The Relief [2] algorithm is used for this problem. We note that the algorithm is only sensitive to interactions between two variables [3], although variables may interact in three in more ways. Additionally, Relief relies on Nearest Neighbour calculations which, as discussed in section 2.5, means that it suffers from high dimensionality in the data, see also [3].

Variable ranking methods can be either supervised or unsupervised. The method of ranking variables by their linear fit with the output variable as described in [4] is a clear example of a supervised method, by relying on the output variable. Unsupervised methods, on the contrary, cannot use the output variable by definition. This can still be useful, however, by selecting variables with positive traits for methods that may be applied to the data. Useful traits include entropy (fig. 1a) that measures how uniform data is w.r.t a variable. A salient variable has high variance (figs. 1b and 1c).



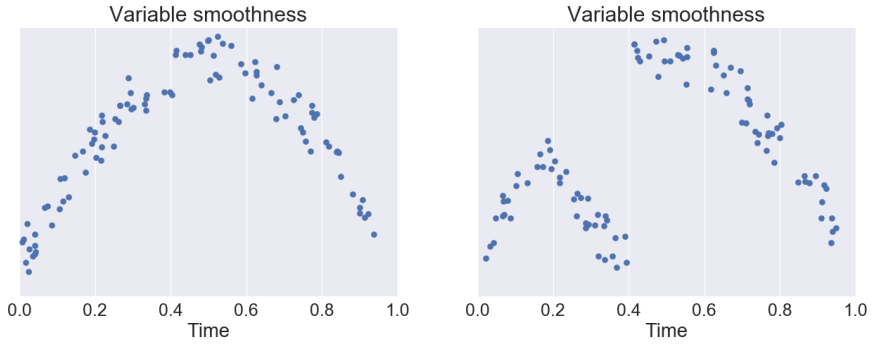
(a) Entropy measures uniform randomness

(b) Saliency measures variance depending on this variable.

(c) Saliency measures variance depending on this variable.

Figure 1: Example data showcasing variable entropy, saliency.

A smooth variable does not change drastically over time, as shown in fig. 2. Illustratively, smoothing techniques are often applied to time series which reduce the impact on model of outliers in the data.



(a) Smooth time-series variable with low local curvatures.

(b) Time-series variable with higher local curvatures making it less smooth.

Figure 2: Example data showcasing variable smoothness.

2.2 Filtering Methods

Filter methods evaluate features using intrinsic properties of the data, they rely on general characteristics of the training data to select relevant features. Filtering methods select candidate features, they calculate the subset score of those features and only after that they evaluate the chosen model based on the performance of the intrinsic properties.

Figure 3: Filtering Methods workflow for feature selection

Filtering methods can be based and ranked according to statistical test such as ANOVA, MANOVA, t-test, χ^2-test for independence, correlation with the target variable, entropy and information-gain based approaches. Filtering methods calculation can also depend on the way we encode our variables for example it is not the same to run a t-test on a categorical encoded variable to 0, 1 and to run a χ^2-test on the expected frequencies of the categories. We must be very careful when we are using filtering methods as we may make strong assumptions on our data.

2.2.1 Advantages

- Filter methods are much faster compared to wrapper methods as they do not involve training our models and they just depend on the velocity of the filtering algorithm that we are using.

- Because of some of the assumptions of linearity or normality of certain filters on our data, filtering methods tend to be less prone to overfitting than wrapper based approaches, they can be used as a prepossessing step to reduce space dimensionallity to then apply a wrapper or embedded approach with more computational efficiency. [4]

2.2.2 Disadvantages

- Because some filtering methods are based on statistical models, sometimes assumptions of the nature of our data must be made. This assumptions can be linear dependence with respect to our independent variable, normality distribution assumption of our features, or homogeneity of variance across different groups or samples.
- Filtering methods can also fail to capture the importance of features that are not relevant by themselves but that they become relevant in combination with another feature of the dataset, **"a variable that is completely useless by itself can provide a significant performance improvement when taken with others."** [4]

2.2.3 Summary of filter methods

Filtering methods can be used as a fast and efficient way to determine the importance of the variables in our data but are much more powerful when used in combination with other techniques. In general some authors recommend the use of wrapper or embedded methods that use a **linear** predictor as a filter, and after that train a **non-linear** model based on the variables selected by the linear filter. [4]

2.3 Wrapper Methods

Wrappers evaluate subsets of variables by using them to train models. Using the performance of the previous model it is decided whether to add or remove features from a subset. This process allows to detect relations between multiple variables. The process consists of two parts, the search algorithm for a subset of features and its evaluation by a machine learning algorithm.

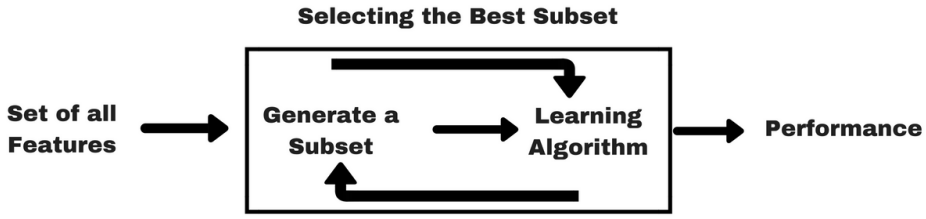


Figure 4: Wrapper Methods workflow for feature selection

Wrapper methods translate the problem of feature selection to a search problem. Any criteria used in the selection of features are included in the search algorithm while the learning algorithm is used simply as an evaluation method, a black box. That makes wrapper methods quite more flexible than other, similar methods, like the embedded approach, since there is no need for adjustments in order for the search and evaluation stages to work properly together. However the choice of the search algorithm is highly important. The wrapper approach uses extensive, greedy search methods so the strategies that are chosen should should aim to minimize the exponential complexity of such searches as best as possible.
The following are three popular strategies/methods:

- **Forward selection**: In this case the initial subset contains no features. In every iteration the most representative feature is selected and added to the subset. This choice is made based on the evaluations from the model. The iterations continue until the termination criteria have been met.
- **Backward elimination**: This is the opposite of the previous method. The initial subset includes all features and in every iteration the least representative feature is eliminated, until the termination criteria is met. Although starting from a full set of features can, theoretically, detect feature interaction more easily, it is too computationally expensive to train classifiers with large number of features. Therefore forward selection is more often used.
- **Recursive feature elimination**: This is a greedy optimization algorithm which aims to find the best performing feature subset. It repeatedly creates models and keeps aside the best or the worst performing feature at each iteration. It constructs the next model with the left features until all the features are exhausted. It then ranks the features based on the order of their elimination.

2.3.1 Advantages

- Promises to deliver the best feature subset. The greedy search approach that is used will detect the best subset, based on the criteria and restrictions enforced on the method.
- No need for adjustments to match the needs of training models, unlike filter methods, since they are used only as evaluation methods.

2.3.2 Disadvantages

- They are very computationally intensive, to the level of being unfeasible to apply when the number of features is too large [5].
- The risk of over-fitting increases when the number of observations is low and therefore there are not enough data to properly train the model [5, 6].

2.3.3 Summary of wrapper methods

Wrapper methods treat the feature selection problem as a search problem. Using different methodologies, feature subsets are created and then evaluated so as to choose the best subset in every iteration. This process continues until the best feature subset has been formed. The evaluation of each subset is done by a machine learning algorithm, which is treated as a black box, making wrappers simple and highly flexible. Compared to other methods, they overcome the filter's inability to detect interactions between multiple features and are far more simple and flexible than embedded methods. Nevertheless, wrappers are very computationally expensive due to the greedy nature of the search they conduct and do risk over-fitting, when used on small data sets.

2.4 Embedded Methods

Embedded methods try combine the advantages of both filtering and wrapper methods. What makes embedded methods different is that these methods perform variable selection as part of the learning procedure. It is a type of learning algorithm that takes advantage of its own variable selection process and performs both feature selection as well as classification simultaneously [7].

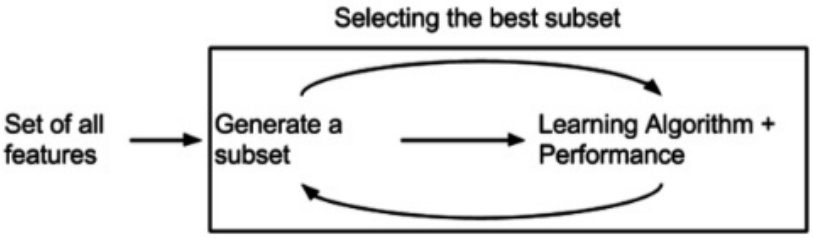


Figure 5: Embedded Methods workflow for feature selection

The embedded method generates a subset of features, and uses a learning algorithm to determine the score of the subset. It takes its own's performance into account and will try to increase it, in contrast to wrapper methods. It takes advantage of its own variable selection process and performs feature selection and classification simultaneously. An example of an embedded method is used in the Microarray application. Duval et al. [8] used an embedded method on a microarray of DNA. They used a Genetic Algorithm to generate subsets of the data, classified it using a Support Vector Machine (SVM) and evaluated this using Classification accuracy (10-fold), to select the best subset to work with.

2.4.1 Advantages

- An embedded approach is an all-in-one method; no further actions are required from the researcher as the program tests its own performance.
- Embedded methods are less computationally expensive.
- By constantly testing its performance, the method is less prone to overfitting.

2.4.2 Disadvantages

- A embedded approach requires multiple algorithms; subset generation, a learning algorithm, as well as an algorithm that measures its performance.
- Because the system constantly creatures new subsets, it is less flexible with backtracking.
- The embedded approach is specific to a learning machine and thus requires the users to use the given learning machine, limiting the freedom of choice.

2.4.3 Summary of embedded methods

Embedded methods combine the advantages of filtering and wrapper methods to create a method that is less prone to overfitting, and uses less computational power to create an all-in-one method for feature selection. By using multiple algorithms, it requires less input from the user and creates subsets with high performance. The downside is the requirement of given learning machines and less backtracking.

2.5 Curse of Dimensionality

With ever increasing computing power and data storage capabilities, very large scale scientific analyses are feasible and necessary. And in the coming century, high-dimensional data analysis will be a very significant activity, and completely new methods of high-dimensional data analysis will be developed. [9] [10]

The curse of dimensionality is a phrase used by several subfields in the mathematical sciences. The curse of dimensionality was coined by Richard Belman in [11], in connection with the difficulty of optimization by exhaustive enumeration on product spaces.

It can also refer to the apparent intractability of systematically searching through a high-dimensional space, simply because the data in the high-dimensional space has become too sparse.

2.5.1 Distance Concentration

Distance concentration is the phenomenon that, as number of dimensions increase, all the pairwise distances (dissimilarities) between data points may converge to the same value. In other words, the distances between data points become more and more similar that using methods like nearest neighbor search become impossible to perform as this method relies on finding the shortest Euclidean distance. Since the shortest and longest distances become very similar, cluster membership becomes more vague. However, laid out in [12], when the number of features that correlate with one another grows at the same speed as the irrelevant features,such as stop-words in text data, distance concentration problem does not occur even with the use of euclidean distances. As the mutual dependence of variables are required among correlated features, the data is not protected against distance concentration, when the dimensions are independent and identically distributed (iid). In real world data, it is often that we find the data dimensions are iid, therefore the aforementioned condition is not met.

2.6 Blessings of Dimensionality

The blessings of dimensionality define the advantages of having high-dimensional spaces.

- One of the blessing include the ability to run models that are much more complex but prone to overfitting.
- Another blessing is the concentration of measure, which is a terminology introduced by V. Milman for a pervasive fact about probabilities on product spaces in high dimensions. With a Lipschitz function f on the d-dimensional sphere, the tails behave at worst like a scalar Gaussian random variable with absolutely controlled mean and variance. In this case the independence of dimensions is needed.

3 Discussion

Feature selection is a powerful tool to our data pre-processing toolbox. It allows us to productively use large amount of data, consisting of numerous features. By properly applying feature selection we can overcome the curse of dimensionality, simplify our models and significantly reduce the resources necessary to train them.
However we should take into account that applying feature selection without proper planning will easily produce the opposite results. Knowing our data and clarifying the objective of our data science project should be the point of reference when choosing which method to apply. Ranking methods are the simplest and can be applied to establish

a base-line or to further examine our data in order to choose an appropriate method. Filter methods are best used when we want to detect relations between two features, while wrapper methods or embedded detect relations among multiple features. Between the later two, wrappers are more flexible but prone to over-fitting with small amounts of data, while embedded methods are more rigid and heavily depend on inside knowledge of the learning algorithm used but are less likely to over-fit.
All in all feature selection should be carried out carefully and with respect to each individual case. In that way we ensure that by removing features from our datasets, we improve our models' performance instead of diminishing it.

References

[1] Trevor Hastie, Rob Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*, volume 1. Springer-Verlag, 01 2001.

[2] Kenji Kira and Larry A. Rendell. A practical approach to feature selection. pages 249–256, 12 1992.

[3] Ryan J. Urbanowicz, Melissa Meeker, William La Cava, Randal S. Olson, and Jason H. Moore. Relief-based feature selection: Introduction and review. *Journal of biomedical informatics*, 85:189–203, 2018.

[4] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182, March 2003.

[5] Ron Kohavi and George H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1):273 – 324, 1997. Relevance.

[6] John Loughrey and Pádraig Cunningham. Overfitting in wrapper-based feature subset selection: The harder you try the worse it gets. In Max Bramer, Frans Coenen, and Tony Allen, editors, *Research and Development in Intelligent Systems XXI*, pages 33–43, London, 2005. Springer London.

[7] Hernandez J. C. H., Duval B, and Hao J.-K. A genetic embedded approach for gene selection and classification of microarray data. In *In Proceedings of the 5th European conference on Evolutionary computation, machine learning and data mining in bioinformatics*, pages 90–101, Heidelberg, Berlin, Germany, 2007. SpringerVerlag.

[8] Duval B, Hao J.-K., and Hernandez J. C. H. A memetic algorithm for gene selection and molecular classification of an cancer. In *In Proceedings of the 11th Annual conference on Genetic and evolutionary computation.*, pages 201–208, New York, USA, 2009. ACM.

[9] Peter Bickel Thomas Bengtsson and Bo Li. Curse-of-dimensionality revisited: Collapse of the particle filter in very large scale systems, 2008.

[10] David L. Donoho. High-dimensional data analysis: The curses and blessings of dimensionality. In *AMS CONFERENCE ON MATH CHALLENGES OF THE 21ST CENTURY*, 2000.

[11] Richard Bellman. *Adaptive Control Processes: A Guided Tour*. Princeton University Press, 1961.

[12] Robert J Durrant and Ata Kabán. When is 'nearest neighbour'meaningful: A converse theorem and implications. *Journal of Complexity*, 25(4):385–397, 2009.