Classification of undesirable events in oil well operation

Evren M. Turan
Department of Chemical Engineering
Norwegian University of Science and Technology (NTNU)
Trondheim, Norway
evren.m.turan@ntnu.no

Johannes Jäschke
Department of Chemical Engineering
Norwegian University of Science and Technology (NTNU)
Trondheim, Norway
johannes.jaschke@ntnu.no

***Abstract*—Various algorithms are compared for the automatic classification of undesirable events during the operation of oil wells. The 3W database compiled by Petrobras and released publicly in 2019 is used to compare classifiers and some aspects of the workflow. Classification is performed in the transient phase of the event, and with the aim to help operators identify which of seven classes of unwanted events is occurring. A decision tree classifier is fitted, and is able to successfully detect the real events, with an F1-score of 85% on test data, with most events classified at 90+% accuracy. Hyperparameters of the workflow were chosen based on the F1-score. Compared to prior work, in which a random forests was fitted, the classifier identified in this work is simpler, while achieving similar performance.**

***Index Terms*—Fault detection and classification, oil well, machine learning**

I. Introduction

Abnormal Event Management is the task of detecting an abnormal event, diagnosing its cause, and returning the process to normal and safe operation [1], [2]. The detection and diagnosis steps can be regarded as a machine learning problem, and there have been various attempts to develop autonomous classification methods to aid operators in this task [3]–[6], [8]. The 3W dataset, compiled by Petrobras, is the first realistic and public dataset of rare undesirable events in oil wells [1]. In this work various classifiers and workflow hyperparameters are compared, in the task of supervised, multiclass classification of undesirable events, during their transient phase. Prior work on the same dataset [8] used a random forest to perform this classification, however only this classifier was fitted, and the work compared one-class, multiple binary, and single multiclass classification. In the current work a (smaller) decision tree is chosen that performs similarly to the optimised random forests developed in [8], and its performance is compared to that of several other classifiers.

A. The 3W dataset

The 3W dataset is made up of over 2000 events, each of which is a time series from a real or simulated offshore well. Hand-drawn profiles of undesirable events are also present in the dataset, however these are not considered in this work as they lack the granularity of the other data. The breakdown of real and simulated data into classes is shown in Table I. Real data was acquired from 21 wells, during actual operation between 2012-2018, while simulated data was obtained from the OLGA Dynamic Multiphase Flow Simulator [7]. Classifier training requires a sufficiently large and representative dataset for each class for good generalisation to future data. Thus, simulated data is required to be used to supplement the real data. The class "Scaling in PCK" has a very low number of events, and it is excluded in this work.

TABLE I
Breakdown of events in the 3W database, for descriptions of each class see [1].

Class Number	Type of event	Real	Simulated	Total
0	Normal	597		597
1	Abrupt Increase of BSW	5	114	119
2	Spurious Closure of DHSV	22	16	38
3	Severe Slugging	32	74	106
4	Flow Instability	344		344
5	Rapid Productivity Loss	12	439	451
6	Quick Restriction in PCK	6	215	221
-	Scaling in PCK	4		4
7	Hydrate in Production Line	3	81	84

Each event corresponding to real wells have measurements from the sensors in Table II, with data recorded every second. The locations of the sensors (excluding the gas lift related sensors) are shown Fig.1. The simulated wells have the same "sensors" apart from P-CKGL and T-CKGL which are not simulated. All the wells in the dataset are naturally flowing wells, i.e. their reservoir pressure is sufficient to allow for hydrocarbon production [1]. They are equipped with a gas lift, to allow for operation in an artificially aided fashion should this be desired [1].

Within a time series, each time entry is labelled as belonging to one of three periods of operation: normal, faulty transient or faulty steady state, with different labels given depending on the fault [1]. Faulty transient periods are those in which dynamics caused by the undesirable event are ongoing, with faulty steady state operation beginning when the dynamics cease. See Fig.2, for an example of an event profile of hydrate formation in the production line. The transient stage was labelled as beginning at 01:01, with the initial dynamics, of P-PDG and P-TPT, not visible in the normalised figure. Between 03:00 - 05:00, it

This research was conducted as part of the AutoPRO project funded by the Norwegian research council.

TABLE II
SENSORS IN THE 3W DATABASE, [1], [8].

Name	Description	Units
P-PDG	Pressure at permanent downhole gauge (PDG)	Pa
P-TPT	Pressure at temperature/pressure transducer (TPT)	Pa
T-TPT	Temperature at TPT	°C
P-MON-CKP	Pressure upstream of production choke (CKP)	Pa
T-JUS-CKP	Temperature downstream of CKP	°C
P-CKGL	Pressure downstream of gas lift choke (CKGL)	Pa
T-CKGL	Temperature downstream of CKGL	°C
QGL	Gas lift flow rate	m^3/s

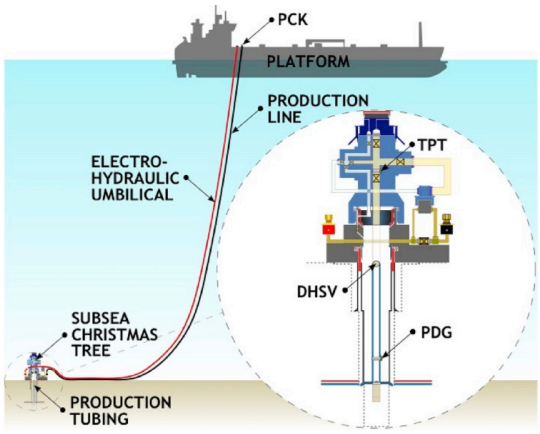

Fig. 1. Simplified schematic of a offshore well, taken from [1].

could be that the hydrate crystals had reached sufficient size to form a plug which severely impacted the flow of hydrocarbons. Note that faults 3 & 4 (severe slugging and flow instability) are characterised by continually changing dynamics and hence are always in the faulty transient period.

In this work various classification methods are compared with one another, with the aim of identifying the occurrence and type of undesirable event during the faulty transient stage with the aim of allowing early classification.

II. WORKFLOW OVERVIEW

A. Data Preprocessing

As previously stated, hand-drawn data is excluded in this work. Simulated data is included as otherwise the frequency of certain classes would be very low. As discussed in Section I-A the simulated events do not include the P-CKGL and T-CKGL sensor data, hence these features are removed from the real well data. This is necessary to prevent the classifiers from distinguishing between simulated and real data sets. To reduce the size and noise of the data, the time series were down-sampled from entries every second to ten seconds, by a local average over this period.

B. Feature engineering

There are various approaches in time series classification (see [9] for a overview), and in this work a feature based approach is considered. The time series is subdivided into windows, and in each window various features of arbitrary complexity are calculated to describe dynamics within the window. This division into time windows and feature calculations reduce the problem to a supervised classification problem. For more details on the features that can be calculated see [10] and the references therein. The choice of time window size is an important hyperparameter as it controls the size of the new dataset and the amount of information in the features. Here, the TSFRESH package [11] is used to extract features, with the following features chosen (for each window):

- The first to fourth moments (mean, variance, skewness, kurtosis) of the time series and the absolute Fourier transform of it
- Miscellaneous features describing the distribution of the data and how it changes: maximum, minimum, median, quantiles, coefficient of variation, mean change, average second derivative
- The coefficients of a linear and third degree polynomial model. The linear model is fitted directly to the data, while the polynomial is fitted as part of a Langevin model as described in [12]

C. Feature selection

The previous step results in the formation numerous features, and it can be desirable to only select a subset of these. There are many options to select a subset of features. The simplest option is to remove features with low or zero variance, i.e. almost unchanging features

Another method is to use some form of statistical test to filter out the most significant features. An example of such a test is implemented in the TSFRESH package [11]. A hypothesis test is used to test the significance of each feature vector for predicting the class. The resulting vector of p-values is evaluated by the Benjamini-Yekutieli procedure [13] to filter the features. For a multiclass problem, one can specify how many classes a feature should be significant for.

One can also perform filtering by fitting a linear classifier, with an L1 norm, and selecting the features with non-zero coefficients to be used by a subsequent classifier. Equivalently this can be performed with a decision tree, with features chosen based on their importance, as reported by the algorithm [14].

A decomposition method can also be used, with Principle Component Analysis (PCA) being a common choice [15]. In PCA a set of orthogonal components is formed which explain decreasing amounts of the total variance, and typically the first N components are chosen and the rest are regarded as noise [15]. An issue with PCA in classification is that low variance components can sometimes be important in the classification task.

Regardless of the method of feature selection, before classification all features should be normalised by subtracting the mean and scaling by the standard deviation of the data. This should be performed only considering the training set, i.e. excluding the currently held out fold in kfolds and the test data. This scaling is necessary, because the pressure data is orders of magnitude larger than the temperature data, and some

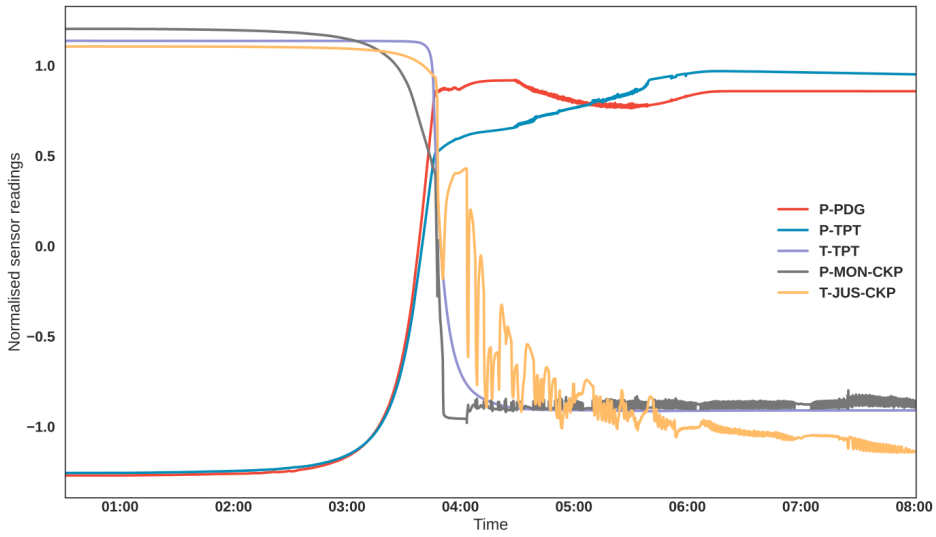

Fig. 2. Example of class 7 event, with normalised sensor data. The transient stage began at 01:01, with approximate steady state reached at 06:21

classification algorithms assume that features have zeros mean and unit variance [14], [15].

D. Classification

There are many classifiers that can be chosen. Linear methods tend to be simple and computationally efficient (especially for large datasets), and if a linear method is able to classify the data well then there is no advantage to using a more complex method. Tree based methods learn decision rules to classify the data, and are among the most flexible and widely used classifiers [9], [16]. The predictions of several decision trees can be combined to improve robustness by using an ensemble method, such as in random forest and AdaBoost [16]. The specific linear and tree based methods compared in this work are:

- **Logistic regression** A classification algorithm, similar to linear regression, where a binary output is modelled by a logistic function.
- **Support Vector Classifier (SVC)** A linear SVC finds the hyperplane that creates the biggest margin between training points of two different classes [16]. Non-linear classification can be performed by the kernel trick.
- **Linear and Quadratic Discriminant Analysis (LDA & QDA)** Both of these methods fit a class conditional, Gaussian distribution to the data. A sample is then allocated to the class that maximises the posterior probability given by Bayes Rule with LDA assuming that the covariance matrix of all classes are equal [14], [16]. These methods do not have hyperparameters.
- **Decision Trees** The feature space is divided into rectangles by a set of if-then-else decisions, with samples within a rectangle allocated to the same class [14], [16]. Over fitting is prevented by pruning (setting a complexity parameter) or setting parameters that control the size of the tree (e.g. maximum depth, minimum samples per split). Minimal cost-complexity pruning is the pruning algorithm used.
- **Random Forest** A random forest is made up of an ensemble of decision trees where each decision tree is grown with a random sample of the training set and with a random subset of features to be used at each splitting point. The prediction of the random forest is made by combining the results of each tree, in sci-kit learn this is done by averaging their probabilistic predictions [14].
- **AdaBoost** A sequence of small decision trees (or some other weak learner) are fitted on modified versions of the data, such that later trees focus on the incorrectly classified instances [14]

The classifier implementations in Scikit-Learn are used, with the Stochastic Gradient Descent optimization technique used for logistic regression and SVC. Methods that are not inherently multiclass (e.g. logistic regression) use a one-vs-rest approach [14]. A similar feature based approach was taken in [8], however a smaller set of features were calculated and PCA was used before classification by a random forest algorithm.

III. Methodology

A. Training, validation and test sets

When comparing hyperparameters and model performance it is important to validate correctly to have an unbiased estimate of the classifiers' performance [15]. To this end 30% of the original events are split into a test set before the work flow. Kfold cross validation was performed on the remaining data to compare hyper parameters, with five folds. In the

validation and test splits, the relative number of classes is kept constant and all time windows belonging to an event are kept in the same set.

B. Metrics

The F1-score is used as the metric to decide the best hyper-parameters and classifier, with the balanced accuracy (or recall) also reported for additional information. The F1-score is the harmonic mean of precision and recall (which take into account Type I and II errors respectively). The unweighted average (macro) F1-score of each class is taken, to correct for the imbalance in the dataset. The balance accuracy is similarly the macro-average of recall scores per class [14].

C. Hyper parameters

A grid search is used to identify the hyperparameters of the workflow, with the sets of values shown in Table III. Additionally, the use of feature selection is investigated by comparing the use of all features, and only selecting features that are relevant for all classes when using the feature selection algorithm of the TSFRESH package [11].

TABLE III
DETAILS OF HYPERPARAMETER GRID SEARCH.

Hyperparameter	Values	Classifier
Time window size (τ, s)	{300, 600, 900}	All
Regularization parameter (RP)*	{1e-7,...,1e-1}	SVC, logistic
Complexity parameter (CP)	{1e-5,...,1e-1}	Decision tree
Number of trees	{50, 100, 150, 175}	Random forest
" "	{100, 250, 400, 550}	AdaBoost
Maximum tree depth	{5, 7, 10, None}	Random forest
" "	{1, 3, 5}	AdaBoost
Number of features at splits	{5, $\sqrt{n_{features}}$, 15}	Random forest
Learning rate	{0.01, 0.1, 1}	AdaBoost

* Also used to calculate the learning rate per [17]

IV. RESULTS

A. Comparison of classifiers

1) Without feature selection: The results of the classifier cross validation are shown in Table IV. The hyperparameters are chosen based on the F1 score, and by this metric the best classifier is the random forest, followed by AdaBoost and the decision tree classier. Interestingly, the ensemble/aggregate tree classifiers have similar F1 score and accuracy to the single decision tree. Ensemble models tend to perform better than their singular counterpart, typically due to instability in the singular model and by avoiding over-fitting. It could be in this data set the number of samples is large and varied enough for the decision tree to be essentially stable. Some of the workflow (e.g. cross validation) and classifiers (e.g. random forest) are partially random. For consistency the random states are fixed, allowing deterministic simulations. The use of different random states give similar results.

Attention should be drawn to the fact that LDA outperforms QDA, which may also appear unusual. It may be that the worse performance is due to QDA being a more flexible classifier than is required, i.e. in comparison the method itself has an increased variance and the fitted model did not have a corresponding reduction in bias [18]. This is also supported due to the good performance of the SVC and logistic regression models which are linear methods like LDA.

TABLE IV
RESULTS OF CLASSIFIER CROSS VALIDATION

Classifier	F1	Accuracy	τ (s)	Hyperparameters
No feature selection				
LDA	0.83	0.86	300	-
QDA	0.69	0.70	300	-
Linear SVC	0.83	0.87	900	RP = 1e-4
Logistic	0.82	0.88	900	RP = 1e-4
Decision Tree	0.89	0.93	600	CP† = 1e-4
Random Forest	0.91	0.94	900	Max depth: 10, max features: 5, Num trees: 150
AdaBoost	0.90	0.92	900	Num trees: 100, max tree depth: 5,learning rate: 0.01
With feature selection				
LDA	0.81	0.83	300	-
QDA	0.75	0.83	900	-
Linear SVC	0.81	0.88	300	RP = 1e-4
Logistic	0.81	0.86	300	RP = 1e-3
Decision Tree	0.90	0.93	900	CP = 1e-4
Random Forest	0.90	0.93	900	Max depth: 10, max features: 5, Num trees: 175
AdaBoost	0.89	0.92	900	Num trees: 100, max tree depth: 5, learning rate: 0.01

† Corresponds to a tree with depth of 16 and 283 nodes.
‡ Corresponds to a tree with depth of 20 and 389 nodes.

2) With feature selection: When selecting feature with the Benjamini-Yekutieli procedure the linear classifiers show slightly worse performance, and QDA is the only classifier to perform notable better with an increase of 0.06 in the F1 score. It should be noted that although the decision tree complexity parameter is the same, a deeper (larger) tree is used to achieve nearly the same score. Also, the chosen time window of the decision tree increased. This is significant as it is preferred to make predictions as early as possible, i.e. to not wait for more time to pass after the abnormal event begins. Although feature selection reduces the fitting time, it has negligible influence on the scoring time, hence once the classifier is trained the speed improvement is not significant. The use of PCA was briefly investigate, and it was found that the features were not strongly correlated, e.g. the dimensionality was mostly retained when using 99% explained variance as a cut-off.

In the previous work [8] a random forest was chosen and accuracy was to identify the hyper parameters (0.97 accuracy with 102 trees, maximum depth of 24). Note that this is a more complex forest than identified in this work, and is likely due to the fact that a smaller set of features were calculated, making the classification more difficult.

Based on the cross validation results the decision tree with no feature selection is chosen as the best classifier. In comparison to the more complex tree methods, there can be minor improvement with the ensemble methods, however this fluctuates depending on the random state set in the fitting procedure, with the decision tree essentially giving the same performance as the ensemble methods throughout. Additionally the use of the smaller time window with the

Fig. 3. Confusion matrix of decision tree.

TABLE V
CLASSIFICATION METRICS FOR DECISION TREE ON TEST DATA

Class	Precision	Recall	F1-score
Normal	0.95	0.85	0.90
Abrupt Increase of BSW	0.83	0.98	0.90
Spurious Closure of DHSV	0.42	0.60	0.49
Severe Slugging	0.97	0.91	0.94
Flow Instability	0.94	0.96	0.95
Rapid Productivity Loss	0.91	0.94	0.92
Quick Restriction in PCK	0.76	0.86	0.81
Hydrate in Production Line	0.84	0.90	0.87
macro avg	0.83	0.88	0.85

decision tree is preferred. Compared to the linear methods the performance is better of the decision tree is better.

B. Test set results

The decision tree with identified hyperparameters is fit to all the training data and results are computed on the test set, shown in the confusion matrix Fig.3 and Table V. The F1-score and accuracy are 0.85 and 0.88 respectively. These are similar to the cross validation results, and suggests that the kfold validation was representative of the test set.

The confusion matrix shows that the model correctly identifies almost all faults, except for class 2, "Spurious closure of DHSV" which only has an accuracy of 60% correct prediction and F1-score of 49%. The poor ability to predict this class may be because this class occurs the least in the dataset (see Table I), and thus there was not sufficient data to train for this class. While this was also an issue in [8] it is not so pronounced, however in that work 15% of hydrate formation was incorrectly predicted as sever slugging. This mismatch is not observed in this data, and is likely due to a feature(s) calculated in the current work that made these classes distinguishable.

V. CONCLUSION

This work describes a method to classify undesirable events in oil well operation during their faulty transient stage. Seven different fault types can be distinguished from normal oil well operations. Features describing the events are calculated in time windows, with feature selection having low impact on classifier training and performance. Classifiers and hyperparameters are compared in a grid search, with 5-fold cross validation, and a decision tree with complexity pruning parameter of 1e-4 is chosen. On the test set the decision tree

achieves an F1-score of 0.85 and balanced accuracy of 0.88. Additionally, the decision tree is able to perform the classification in 10 minute windows, allowing for early identification of the fault. The decision tree is able to classify all faults well, except for "Spurious closure of DHSV" which may be due to its under-representation in the data. Interestingly, ensemble methods such as random forest and AdaBoost did not show an improvement in performance over the decision tree. This could be because the tree is stable enough that there is not a significant reduction in variance when using the ensemble methods. Compared to previous work, the decision tree is smaller than the random forest developed by [8] on the same dataset, likely due to the calculation of additional descriptive features in this work.

ACKNOWLEDGMENT

Thanks to Professor Frank Westad for helpful discussions in the development of this work.

REFERENCES

[1] R. E. V. Vargas, C. J. Munaro, P. M. Ciarelli, A. G. Medeiros, B. G. do Amaral, D. C. Barrionuevo, J. C. D. de Araújo, J. L. Ribeiro, and L. P. Magalhães, "A realistic and public dataset with rare undesirable real events in oil wells," *Journal of Petroleum Science and Engineering*, vol. 181, oct 2019.

[2] V. Venkatasubramanian, R. Rengaswamy, K. Yin, and S. N. Kavuri, "A review of process fault detection and diagnosis," *Computers & Chemical Engineering*, vol. 27, no. 3, pp. 293–311, mar 2003. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0098135402001606

[3] C. G. Bezerra, B. S. J. Costa, L. A. Guedes, and P. P. Angelov, "An evolving approach to unsupervised and Real-Time fault detection in industrial processes," *Expert Systems with Applications*, vol. 63, pp. 134–144, nov 2016. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0957417416303153

[4] G. M. Xavier and J. M. de Seixas, "Fault Detection and Diagnosis in a Chemical Process using Long Short-Term Memory Recurrent Neural Network," in *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, jul 2018, pp. 1–8. [Online]. Available: https://ieeexplore.ieee.org/document/8489385/

[5] R. E. V. Vargas, C. J. Munaro, P. M. Ciarelli, and J. C. D. de Araujo, "Proposal for two classifiers of offshore naturally flowing wells events using k-nearest neighbors, sliding windows and time multiscale," in *2017 6th International Symposium on Advanced Control of Industrial Processes (AdCONIP)*. IEEE, may 2017, pp. 209–214. [Online]. Available: http://ieeexplore.ieee.org/document/7983782/

[6] Y. Liu, K.-T. Yao, S. Liu, C. S. Raghavendra, T. L. Lenz, L. Olabinjo, F. B. Seren, S. Seddighrad, and C. Dinesh Babu, "Failure Prediction for Artificial Lift Systems," in *SPE Western Regional Meeting*. Society of Petroleum Engineers, apr 2010. [Online]. Available: http://www.onepetro.org/doi/10.2118/133545-MS

[7] Schlumberger, "OLGA Dynamic Multiphase Flow Simulator," 2020. [Online]. Available: https://www.software.slb.com/products/olga

[8] M. A. Marins, B. D. Barros, I. H. Santos, D. C. Barrionuevo, R. E. Vargas, T. de M. Prego, A. A. de Lima, M. L. de Campos, E. A. da Silva, and S. L. Netto, "Fault detection and classification in oil wells and production/service lines using random forest," *Journal of Petroleum Science and Engineering*, p. 107879, sep 2020.

[9] A. Bagnall, J. Lines, A. Bostrom, J. Large, and E. Keogh, "The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances," *Data Mining and Knowledge Discovery*, vol. 31, no. 3, pp. 606–660, may 2017.

[10] M. Christ, A. W. Kempa-Liehr, and M. Feindt, "Distributed and parallel time series feature extraction for industrial big data applications," 2016. [Online]. Available: http://arxiv.org/abs/1610.07717

[11] M. Christ, N. Braun, J. Neuffer, and A. W. Kempa-Liehr, "Time Series FeatuRe Extraction on basis of Scalable Hypothesis tests (tsfresh – A Python package)," *Neurocomputing*, vol. 307, pp. 72–77, sep 2018. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0925231218304843

[12] R. Friedrich, S. Siegert, J. Peinke, S. Lück, M. Siefert, M. Lindemann, J. Raethjen, G. Deuschl, and G. Pfister, "Extracting model equations from experimental data," *Physics Letters A*, vol. 271, no. 3, pp. 217–222, jun 2000. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0375960100003340

[13] Y. Benjamini and D. Yekutieli, "The Control of the False Discovery Rate in Multiple Testing under Dependency," *The Annals of Statistics*, vol. 29, no. 4, pp. 1165–1188, 2001.

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

[15] K. Esbensen, B. Swarbrick, and F. Westad, *Multivariate Data Analysis*, 6th ed. Oslo: CAMO Software AS, 2018.

[16] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, ser. Springer Series in Statistics. New York, NY: Springer New York, 2009. [Online]. Available: http://link.springer.com/10.1007/978-0-387-84858-7

[17] L. Bottou, *Neural Networks: Tricks of the Trade*, ser. Lecture Notes in Computer Science, G. Montavon, G. B. Orr, and K.-R. Müller, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, vol. 7700. [Online]. Available: http://link.springer.com/10.1007/978-3-642-35289-8_25 http://link.springer.com/10.1007/978-3-642-35289-8

[18] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning*, ser. Springer Texts in Statistics. New York, NY: Springer New York, 2013, vol. 103. [Online]. Available: http://link.springer.com/10.1007/978-1-4614-7138-7