Ocean Engineering 188 (2019) 106282

Contents lists available at ScienceDirect

Ocean Engineering

journal homepage: www.elsevier.com/locate/oceaneng

Machine learning models for predicting ship main engine Fuel Oil Consumption: A comparative study

Christos Gkerekos [a,*], Iraklis Lazakis [a], Gerasimos Theotokatos [b]

[a] Department of Naval Architecture, Ocean and Marine Engineering, University of Strathclyde, Glasgow, UK
[b] Maritime Safety Research Centre, Department of Naval Architecture, Ocean and Marine Engineering, University of Strathclyde, Glasgow, UK

ARTICLE INFO

Keywords:
FOC prediction
Ship energy efficiency
Multiple regression
Support Vector Machines
Neural Networks
Ensemble methods
Machine learning

ABSTRACT

As Fuel Oil Consumption (FOC) constitutes over 25% of a vessel's overall operating cost, its accurate forecasting, and the reliable prediction of the relevant ship operating expenditures can majorly impact the ship operation sustainability and profitability. This study presents a comparison of data-driven, multiple regression algorithms for predicting ship main engine FOC considering two different shipboard data acquisition strategies, noon-reports and Automated Data Logging & Monitoring (ADLM) systems. For this, various multiple regression algorithms including Support Vector Machines (SVMs), Random Forest Regressors (RFRs), Extra Trees Regressors (ETRs), Artificial Neural Networks (ANNs), and ensemble methods are employed. The effectiveness of the tested algorithms is investigated based on a number of key performance indicators, such as the mean and median average error and the coefficient of determination (R^2). ETR and RFR models were found to perform best in both cases, whilst the existence of an ADLM system increased accuracy by 7% and reduced the required period for data collection by up to 90%. The derived models can accurately predict the FOC of vessels sailing under different load conditions, weather conditions, speed, sailing distance, and drafts.

1. Introduction

Efficient operation of vessels can lead to a reduction of operating costs and subsequent increase in profitability. Therefore, it constitutes a direction pursued by a number of maritime industry stakeholders such as ship operators, maritime regulators, and policy makers. This can furthermore be justified by financial reasons, such as reduced Fuel Oil Consumption (FOC) and decreased maintenance costs. Ronen (2011) notes that when bunker fuel price is at around 500 USD per tonne, fuel costs correspond to approximately 75% of the total operating cost of a large containership. Accordingly, Stopford (2009) notes that the FOC constitutes approximately two-thirds of a vessel's voyage costs, and over one-quarter of a vessel's overall running costs. For this reason, shipping companies have been focusing on implementing fuel efficiency measures. In order to monitor and increase the fuel efficiency, eventually offering a formalised optimisation approach, a suitable modelling framework that can take into account relevant variables (measurements) and their interdependencies is required. Deriving a model that can accurately predict vessel performance under varying ship operational profiles and environmental conditions can assist in the identification of optimal operating profiles. Additionally, the existence of such a model could help recognise deviating performance patterns that could imply the vessel's systems and/or subsystems degradation (Cipollini et al., 2018; Raptodimos and Lazakis, 2018; Lazakis et al., 2018, 2019).

In this respect, several modelling attempts have been reported in the pertinent literature, ranging from first-principles approaches focusing on semi-empirical formulae from model-tests to data-driven (Machine Learning (ML)) models coupled with installed Automated Data Logging & Monitoring (ADLM) systems that mines information from a vast amount of data points. ML models, although more computationally expensive, offer the benefit of providing results tailored to specific hull forms, hull and Main Engine (M/E) conditions, and operational profiles.

ML can be defined as "the use of formal structures (machines) to do inference (learning)" (Clarke et al., 2009). An alternative definition by Alpaydin (2014), identifies ML as the process of programming computers in order to optimise a certain performance criterion based on example (i.e. training) data or past experience. Machine learning problems can be usually classified into supervised and unsupervised problems. Unsupervised learning refers to the machine learning problem where training examples only comprise input values and the algorithm goal is to provide some insight on that input (Bishop, 2006b). In the case of supervised learning, the algorithm's aim is to utilise training data that comprise examples of inputs and the relevant target

[*] Corresponding author.
E-mail address: christos.gkerekos@strath.ac.uk (C. Gkerekos).

https://doi.org/10.1016/j.oceaneng.2019.106282
Received 14 January 2019; Received in revised form 27 June 2019; Accepted 5 August 2019
Available online 26 August 2019

(output) to learn a mapping that returns a relevant target value for new observations. In cases where a finite number of discrete model output categories is present, this problem is called classification, whereas in the case of continuous target variables the task is called regression. Therefore, in reference to this taxonomy, the problem of estimating the FOC (continuous target value) of a vessel, dependent on the parameters (inputs) affecting the overall resistance (e.g. weather conditions, vessel load, and speed), is a regression problem.

The purpose of this study is to examine the efficacy of several multiple regression algorithms on the task of ship FOC forecasting under different data sampling frequencies. For this reason, a diverse set of machine learning regression algorithms, namely Linear Regression (LR) (both with and without regularisation), Decision Tree Regressors (DTRs), Random Forest Regressors (RFRs), Extra Trees Regressors (ETRs), Support Vector Reggressors (SVRs), K-Nearest Neighbours (KNN), Artificial Neural Networks (ANNs), and ensemble method algorithms are employed.

It is expected that the methodology developed in this study will be used to train optimal models pertaining to ship FOC prediction. This will help track vessel performance degradation, optimise shipping operations, accurately reflect ship emissions and eventually be used as a basis for route optimisation purposes.

The remaining of this article is structured as follows. Section 2 refers to the research background, including an overview of previous attempts at modelling FOC. Section 3 elaborates on the proposed methodology, focusing on aspects of data pre-processing, model training and verification. Section 4 details the setup of the case studies for the verification of the proposed work, presenting and discussing the results. Finally, in Section 6, overall conclusions are provided.

2. Literature review

This section provides an overview of scientific literature, pertinent to this study. First, methodologies relevant to fuel efficiency and FOC modelling are analysed. A synopsis of data-driven techniques relevant to the modelling requirements of this paper are then presented.

Bialystocki and Konovessis (2016) performed a statistical analysis of noon-reports of a Roll-on/Roll-off vessel (Ro-Ro) in order to identify the influence of factors such as ship's draft, displacement, weather velocity and direction, and hull and propeller roughness. Once several corrections suggested are applied to the obtained data along with relevant filtering, curves for each frequently-observed sea state are fitted. This provides a simple algorithm that approximates FOC. Lu et al. (2015) developed a semi-empirical method for the prediction of operational performance of ships. This method is based on modelling still water and added resistance components. Through that, the ship's operational performance is modelled, taking into consideration the weather and relevant sea state. This model is then utilised to optimise the ship's route.

Beşikçi et al. (2016) suggested the use of ANNs for the prediction of ship FOC at various operational conditions. Additionally, a Decision Support System (DSS) is elaborated for real-time, energy efficient operations. The suggested methodology is compared against Multiple Regression (MR) analysis, displaying superior results. Petersen et al. (2012) evaluated ferry main engine FOC modelling approaches, also based on ANNs. The output of the derived models were used for trim optimisation purposes. Meng et al. (2016) suggest a data pre-processing methodology based on outlier-score-based data. Following that, two regression models are developed in order to link available data with the vessel's FOC. The first model connects the ship's FOC with its speed and displacement. The second model builds on the first, utilising the information provided by the first while also including weather conditions. They validated the work performed utilising noon-report data from 13000-TEU containerships. Simonsen et al. (2018) proposed a method of utilising Automatic Identification System (AIS) data to estimate the FOC of cruise ships sailing Norwegian waters. The authors note that the outcome of this method can be used to also estimate Green House Gas (GHG) emissions. Lundh et al. (2016) proposed a method to estimate the FOC of vessels equipped with diesel electric propulsion systems. This is used to optimise the use of individual generators in a multi-generator set-up, offering fuel savings of up to 6% when applied to a large cruise ship. Moreno-Gutiérrez et al. (2015) provide a comparative analysis of first-principle approaches to estimating the energy consumption of vessels. Mao et al. (2016) compared linear regression, first-order autoregressive, and a mixed effect models for the speed prediction of a container ship. Accordingly, Yao et al. (2012) investigated the correlation between FOC and the ship speed of containerships of different sizes.

Cichowicz et al. (2015) provided a methodology for first-principles, time-domain modelling of main and auxiliary engines for assessment of life-cycle ship performance and energy efficiency. Speed and draft are taken into consideration, along with hull fouling and deterioration of engine performance. Sea state is included implicitly by considering an additional M/E load (sea margin). The methodology was demonstrated using data from 3700-TEU containership. Coraddu et al. (2017) performed a comparison of white, grey, and black box models for the estimation of FOC of a Handymax chemical/product tanker, concluding that grey-box models can effectively forecast FOC when only limited historical data are available.

Trodden et al. (2015) focused on data pre-processing and suggest a methodology, ancillary to the ones elaborated above, for splitting available ship data into steady-state chunks that can then be used for fuel efficiency monitoring. Perera and Mo (2018) suggested another ancillary methodology for the compression of ship performance and navigation data. This is implemented through an autoencoder system, compressing data before transmission and then expanding them upon receipt. Such an implementation is extremely beneficial as the amount of data that can be transferred given any bandwidth and cost constraints is increased, potentially leading to more accurate models. Tsitsilonis and Theotokatos (2018) developed a systematic methodology for energy management of ship prime movers. A statistical analysis is combined with energy and exergy analyses to identify key areas where energy savings can be obtained. This methodology was applied in both ADLM and noon-report data. Wang et al. (2018) proposed a Least Absolute Shrinkage and Selection Operator (LASSO) regression model for the estimation of a vessel's FOC. This model was shown to have optimal performance when compared to ANN, SVR, and Gaussian Processes (GPs) models in a case study utilising low-frequency data obtained from a fleet of containerships.

From the above, it can be deduced that modelling of vessels' FOC is an active research field with multiple different approaches being realised concurrently. However, up to the present, most studies utilise different datasets, with different acquisition and modelling particularities and other inherent assumptions such as data filtered for adverse weather. Due to these inconsistencies in the current literature, it is impossible to identify the modelling approaches that yield optimal results for the common problem of modelling the FOC of a vessel. This work aims to alleviate this gap by proposing an end-to-end pre-processing and model training pipeline in order to efficiently quantify the differences that are due to the data acquisition and model selection strategy, independently from other factors. Through this pipeline, a ranking of most conventional data-driven modelling approaches can be obtained, along with a quantification of the benefits obtained by implementing an ADLM system instead of noon reports.

3. Methodology

The methodology elaborated in this section consists of the following steps: a) the description of a suitable pre-processing technique for the acquired dataset; b) the development and implementation of multiple models following different modelling methodologies; c) the optimisation of the hyperparameters of these models; and d) the comparison of

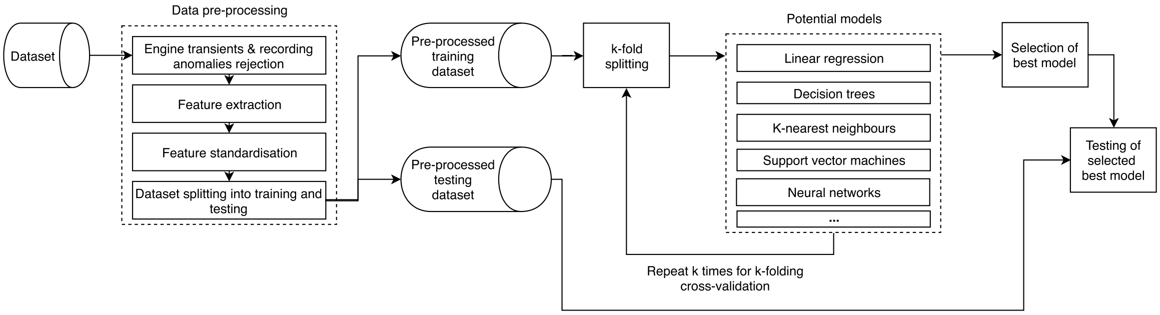
Dataset
Data pre-processing
Engine transients & recording anomalies rejection
Feature extraction
Feature standardisation
Dataset splitting into training and testing
Pre-processed training dataset
Pre-processed testing dataset
k-fold splitting
Potential models
Linear regression
Decision trees
K-nearest neighbours
Support vector machines
Neural networks
...
Repeat k times for k-folding cross-validation
Selection of best model
Testing of selected best model

Fig. 1. Visual representation of the suggested methodology.

these models to identify the modelling techniques that offer the best performance.

A visual representation of the developed methodology is presented in Fig. 1, illustrating all suggested modules and their relevant interconnections.

3.1. Dataset acquisition

The Data Acquisition (DAQ) required for model training can either obtained through an ADLM system or through the processing of noon-reports depending on relevant availability. Compare to parsing noon-reports, ADLM systems provide higher-frequency data of increased accuracy, albeit at an elevated cost. Due to the potential existence of measuring anomalies and undesirable data points (e.g. engine transients), pre-processing follows the data acquisition stage.

3.2. Data pre-processing

In this section, the steps employed for the pre-processing of an acquired dataset are presented. Namely, engine transients and recording anomalies are identified and rejected, additional useful features are extracted from the raw data and finally the remaining data is prepared for ML use through standardisation. This procedure is performed in order to deliver the dataset in the state required for modelling.

3.2.1. Engine transients & recording anomalies rejection

In order to detect operating periods that do not correspond to the employed operating profile considered for the model, manoeuvring and engine transients are identified and rejected. In this respect, M/E Original Equipment Manufacturers (OEMs) provide a minimum engine speed for continuous operation (MAN B&W Diesel A/S, 2004). Usually, this limit is at 15–20% of the engine's nominal maximum continuous (L_1) speed for electronically-controlled engines and at 20–25% for camshaft-controlled engines. Any observations corresponding to measured speed below that threshold is then rejected as an engine transient or manoeuvring. Additionally, observations where the engine power varies by more than 5% hourly (Tsitsilonis and Theotokatos, 2018) are also discarded as transients in order to only retain data points reflecting steady-state operation.

Either data acquisition process may potentially include inconsistent and/or faulty data entries (e.g. due to recording inconsistencies, human error, or sensor faults) that need to be discarded before model training. Therefore, while loading the dataset, these elements are detected and eventually rejected. For this, feature vectors are scanned for elements with values beyond $\mu \pm 3\sigma$ and corresponding observations are dropped from all feature vectors. Variable μ corresponds to the mean value of each vector and σ to its standard deviation. Assuming a normal distribution, 99.7% of normal data should be within $\mu \pm 3\sigma$. Therefore, this formulation filters out most abnormal data points from the training dataset, without affecting the vast majority of normal points.

3.2.2. Feature engineering

Given domain knowledge of the available parameters for FOC modelling, transformations can be performed to engineer new features that better capture the information contained in the raw dataset.

For example, forward and aft draft observations can be transformed into draft amidships and trim features, as these features can be, potentially, more accurate predictors for the FOC of the vessel. Accordingly, in cases where flow meters are installed in both the inlet and return lines, the difference of the two measurements can be computed to obtain a single target variable for the model.

3.2.3. Feature standardisation

All numerical attributes in the dataset are standardised by removing the mean and scaling to unit variance. Therefore, for a numerical attribute x, a standardised attribute x' is produced by

$$x' = \frac{x - \mu}{\sigma} \tag{1}$$

where μ is the mean value of all values belonging to that attribute and σ its standard deviation. All attributes are standardised so that all attributes can contribute equally to the objective function that is used for model training.

3.3. Modelling methodologies

All modelling methodologies presented below are methodologies related to regression analysis. Regression models may be derived with a varying level of complexity and consequently accuracy of results. Therefore, possible methods span a wide range of options, from closed-form linear models to deep (i.e. multi-layered) neural networks (Bishop, 2006b; Russell and Norvig, 2010).

3.3.1. Parametric versus non-parametric modelling

Modelling approaches can be split into two major categories: parametric and non-parametric. Parametric models assume some finite set of parameters θ that are obtained from the training set during the learning phase (Bishop, 2006b). Following that phase, the training set is discarded and any future predictions x are independent of the observed dataset D so that:

$$P(x \mid \theta, D) = P(x \mid \theta) \tag{2}$$

In other words, θ is assumed to capture all variance contained in the dataset D (Clarke et al., 2009). Therefore, even if the complexity of a dataset is unbounded (potentially infinite), the complexity of the model is bounded (Russell and Norvig, 2010). Models such as linear regression, Artificial Neural Networks (ANNs), and Support Vector Reggressors (SVRs) with a linear kernel are parametric models.

In contrast to that, non-parametric models assume that the dataset distribution cannot be defined using any finite number of parameters. For this reason, in non-parametric models, training data, or at least a subset of them, are kept and utilised during the prediction phase (Bishop, 2006b). Therefore, the amount of information that θ

can capture grows with the number of training data points in dataset D. Decision tree regressors, random forest regressors and SVRs with a Radial Basis Function (RBF) kernel are considered non-parametric as the number of parameters grows with the size of D.

Following the above, non-parametric modelling approach can potentially provide higher-performance models due to a reduced number of parameter assumptions. However more training data are required and the computation cost is increased.

Finding the optimal model-derivation methodology is non-trivial as this is affected, among others, by the quantity and quality of available data, and the nature (and also complexity) of the problem at hand.

3.3.2. Multiple linear regression

LR, a parametric model, constitutes the simplest regression algorithm, involving a linear combination of the input variables $\mathbf{x} = (x_1, \dots, x_D)$ (Bishop, 2006b):

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1\,x_1 + \dots + w_D\,x_D = w_0 + \sum_{j=1}^{D} w_j\,x_j \tag{3}$$

Parameters w_j, $j \in (0, \dots, D)$ of Eq. (3) can then be estimated using a Least Squares (LS) approach as:

$$\hat{w} = \underset{w}{\operatorname{argmin}} \left\{ \sum_{i=1}^{N} \left(y_i - w_0 - \sum_{j=1}^{D} \left(w_j\,x_{ij} \right) \right)^2 \right\} \tag{4}$$

Multiple Linear Regression (MLR) constitutes an extension of LR, in cases where $D > 1$ (Hastie et al., 2009). LR and MLR models are often used as a baseline, against which the performance of other models is evaluated.

3.3.3. Ridge & LASSO regression

Ridge Regression (RR) follows the concept of MLR but instead of using the parameters w_i derived through LS, in RR these parameters are shrunk by imposing a penalty on the square of each parameter (Hastie et al., 2009). In this case, Eq. (4) obtains an additional regularisation parameter and becomes:

$$\hat{w}_{\mathrm{RR}} = \underset{w}{\operatorname{argmin}} \left\{ \sum_{i=1}^{N} \left(y_i - w_0 - \sum_{j=1}^{D} \left(w_j\,x_{ij} \right) \right)^2 + \lambda \sum_{j=1}^{D} w_j^2 \right\} \tag{5}$$

where hyperparameter $\lambda > 0$ is a user-selectable parameter that controls the amount of shrinkage. This shrinkage helps avoid overfitting the training dataset.

Least Absolute Shrinkage and Selection Operator (LASSO) is another shrinkage method, similar to RR, with the main difference being that the penalty is imposed on the absolute value of each parameter instead of their squares. Therefore, parameters w_i can now be predicted as

$$\hat{w}_{\mathrm{LASSO}} = \underset{w}{\operatorname{argmin}} \left\{ \frac{1}{2} \sum_{i=1}^{N} \left(y_i - w_0 - \sum_{j=1}^{D} \left(w_j\,x_{ij} \right) \right)^2 + \lambda \sum_{j=1}^{D} |w_j| \right\} \tag{6}$$

An extended version of LASSO and RR are Elastic Nets, where both absolute-value and squared regularisations are implemented concurrently, with the regularisation term of Eqs. (5), (6) which become

$$\lambda_1 \sum_{j=1}^{D} |w_j| + \lambda_2 \sum_{j=1}^{D} w_j^2 \tag{7}$$

In the Scikit-learn implementation (Pedregosa et al., 2011), hyperparameters α and $\lambda 1_{ratio}$ are used instead. The following equations transform λ_1 and λ_2 to α and $\lambda 1_{ratio}$:

$$\alpha = \lambda_1 + \lambda_2 \tag{8a}$$

$$\lambda 1_{ratio} = \frac{\lambda_1}{\lambda_1 + \lambda_2} \tag{8b}$$

3.3.4. Decision tree regressors

DTRs are a non-parametric, regression method. DTR models partition the feature space into rectangles and learn a simple (e.g. constant) model in each of those (Hastie et al., 2009).

DTRs do not produce a continuous output in the traditional sense. Instead, these models are trained on a training set whose outputs lie on a continuous range. Their output ends up being the mean value of the training set observations that reside in the same node.

One of the most common methods for tree-based regression is Classification And Regression Trees (CART) (Breiman et al., 1984). In this case, the original feature space is split into two regions, selecting the split point and dependent variable (feature) to obtain the best model fit (Hastie et al., 2009). This is performed recursively, until the activation of a stopping rule.

Assuming that the feature space has been partitioned into M regions, namely $R_1, \dots, R_M$, and that the model's prediction at each region is c_m, the DTR model will have the following formulation:

$$y(\mathbf{x}) = \sum_{m=1}^{M} c_m\,\mathbf{1}\left\{ x \in R_m \right\} \tag{9}$$

where $\mathbf{1}$ is the indicator function, returning 1 where the condition in brackets is true, and 0 in any other case. Following the same optimisation problem as with MLR, the best $\hat{c}_m$ can be obtained through the minimisation of the fit's LS, $\sum \left(y_i - f(x_i) \right)^2$, obtaining as value the average of the observations lying in that region (Hastie et al., 2009):

$$\hat{c}_m = \operatorname{ave}\left(y_i | x_i \in R_m \right) \tag{10}$$

Whilst the optimal c_m values can be easily computed, the same is not true for the region splitting. For this reason, a greedy algorithm is used recursively to find an optimal splitting, until the stopping rule is activated. This relates to the size of the tree and is a user-selectable parameter that relates to the data available and the complexity of the underlying problem. More specifically, user-selectable parameters are the maximum depth of the tree (`max_depth`, reflecting the number of permitted splits), the minimum amount of samples required to split an internal node (`min_samples_split`), the minimum amount of samples required to exist at each leaf (`min_samples_leaf`), and the maximum amount of features `max_leaf` considered when the splitting optimisation is performed.

3.3.5. K-nearest neighbours

Nearest Neighbours is one of the simplest non-parametric models. There, given a point x_q, the algorithm identifies the k nearest neighbours distance-wise (Russell and Norvig, 2010), with the parameter k being user-selectable (`n_neighbours`).

Different algorithms exist for the computation of the nearest neighbours but Scikit-learn selects the most appropriate automatically, based on the input values. If a non-brute-force approach is used, an algorithm hyperparameter is leaf size (`leaf_size`) that affects the speed and memory usage of the algorithm and depends on the underlying problem's nature.

In order to calculate the distance between x_q and any other point x_j, usually Minkowski distance L^p is used

$$L^p(x_j, x_q) = \left(\sum_i |x_{j,i} - x_{q,i}|^p \right)^{1/p} \tag{11}$$

with $p = 1$ this corresponds to the Manhattan distance and with $p = 2$ to the Euclidean distance (Hu et al., 2016).

Additionally, the weighting function is user-selectable (`weights`), as all k points can contribute equally ("uniform" weights) or the weight of each contributing point can be equal to the inverse of its distance from point x_q.

3.3.6. Support vector machines

Support Vector Machines (SVMs) in their simplest form constitute a two-class classifier in cases where the two classes are linearly separable. SVMs work by deriving the optimal hyperplane, i.e. the hyperplane that offers the widest possible margin between instances of the two classes. Their functionality can be extended by the introduction of a non-linear kernel, allowing them to learn non-linear mappings, i.e., classify between non-linearly separable classes (Theodoridis and Koutroumbas, 2008). Depending on the properties of the selected kernel, SVMs can either be parametric or non-parametric models.

SVMs can also be built as regressors (Smola and Schölkopf, 2004). Support Vector Reggressors (SVRs) work in a similar way, this time trying to fit a hyperplane that accurately predicts the target values of training samples within a margin of tolerance ϵ. In the simpler case where a linear kernel is used, a SVR model will be of the form

$$f(\mathbf{x}) = \mathbf{x}^T\,\mathbf{w} + w_0 \tag{12}$$

Model parameters w are obtained through the minimisation of the function

$$H(\mathbf{w}, w_0) = \sum_{i=1}^{N} V\left(y_i - f(x_i)\right) + \frac{\lambda}{2}\|\mathbf{w}\|^2 \tag{13}$$

where function V is defined as

$$V_\epsilon(r) = \begin{cases} 0 & \text{if } |r| < \epsilon \\ |r| - \epsilon & \text{otherwise} \end{cases} \tag{14}$$

and λ represents a regularisation term, similarly to, e.g., LASSO models. This formulation of V allows errors of less than ϵ to be ignored (Hastie et al., 2009).

In the case of non-linear kernels, where the regression function is approximated in terms of a set of basis functions $\left\{h_m(x)\right\}$ where $m = 1, \ldots, M$, Eq. (12) becomes of the form

$$f(\mathbf{x}) = \sum_{m=1}^{M} \mathbf{w}_m\, h_m(x) + w_0 \tag{15}$$

and accordingly, Eq. (13) becomes

$$H(\mathbf{w}, w_0) = \sum_{i=1}^{N} V\left(y_i - f(x_i)\right) + \frac{\lambda}{2}\|\mathbf{w}_m\|^2 \tag{16}$$

An often-used non-linear kernel is the RBF kernel, formulated as

$$K\left(\mathbf{x}, \mathbf{x}'\right) = \exp\left(-\gamma\|\mathbf{x} - \mathbf{x}'\|^2\right) \tag{17}$$

In the ν-SVM implementation of the LIBSVM library (Chang and Lin, 2011) used in Scikit-learn, the penalty parameter of the error term is expressed by a parameter C, and the upper bound of the fraction of training errors is expressed by a parameter ν.

3.3.7. Shallow & deep neural networks

ANNs are computing systems, inspired by the way biological nervous systems work. Various ANN architectures exist, offering superior performance at many machine learning tasks, including classification and regression. ANNs are extremely versatile as they can accurately model complex non-linear behaviours.

ANNs are based on an interconnected group of connected units (neurons) where each connection between these units transmits a signal from one to another, when the linear combination of its inputs exceeds some threshold (Russell and Norvig, 2010). The receiving unit can process that signal and then pass it on to the next unit. Due to the described architecture, ANNs are considered parametric models.

Two important parameters of ANNs are the number of hidden (between input and output) layers and the number of units per layer (`hidden_layer_sizes`). Excluding the input and output layers that always exist, different architectures call for different number of hidden layers and units. Accordingly, different activation functions can be implemented, altering the complexity learnable by the model.

Consequently, depending on the number of layers implemented, ANNs can be classified as shallow and deep. Whilst no formal rule exists to separate shallow and deep neural networks (Schmidhuber, 2015), usually networks that have more than 1 hidden layer are considered deep. As the number of layers increases, the model can "learn" more non-linear behaviours. At the same time, training becomes more computationally expensive and the risk of overfitting the dataset also increases.

Given an ANN regressor with an input layer $\mathbf{x}$, a hidden layer with M nodes Z_m, and an output layer consisting of a single node, each node is of the form (Hastie et al., 2009)

$$\begin{aligned} Z_m &= \sigma\left(\alpha_{0m} + a_m^T\mathbf{x}\right) \\ Y &= f(\mathbf{x}) = g(w_0 + w^T Z) \end{aligned} \tag{18}$$

where $Z = \left(Z_1, Z_2, \ldots, Z_M\right)$, $\sigma(\cdot)$ is a user-selectable `activation` function (nowadays tanh or ReLu (Rectified Linear Units) are the preferred choice) and $g(\cdot)$ is the user-selectable output function. In the case of regression, the identity function is used as output function (Hastie et al., 2009). The formulation presented above can easily be extended for the case of multiple hidden layers by using the output of each layer as input for the next, and so forth. Accordingly, in order to derive optimal weight parameters α_i and w_i for an ANN, a least-squares approach can be implemented. Similarly to other models, regularisation at its most basic form is applied through a parameter `a` that controls the L2 regularisation term.

3.3.8. Ensemble methods

The base idea behind ensemble learning is the derivation of a prediction model by combining a number of simpler models. Two of the most prominent ensemble methods are boosting and bagging (bootstrap aggregating). Bagging uses bootstrap (i.e. with replacement) samples of the original dataset to train models of reduced variance (Bishop, 2006a; Hastie et al., 2009). In contrast, each boosting model instance utilises the whole dataset, assigning increased weights to observations where previous models underperformed (Bishop, 2006a).

Random forest regressors. Random forests are based on the bagging meta-algorithm, where a number (`n_estimators`) of de-correlated decision tree regressors are produced based on the available training set. Then, the output of the random forest regressor is calculated by averaging the results of individual decision trees. In Scikit-learn's implementation, the minimum number of samples required to split an internal node is controlled similarly to DTRs, through a `min_samples_split` parameter.

Extra trees. Extra (extremely randomised) trees consist a variation of RFRs where the whole dataset is used at each instance (Breiman, 1998), and where the tree-splits are chosen completely at random. In Scikit-learn's implementation, the same hyperparameters as in the case of DTRs are in place, with the addition of `n_estimators` to specify the number of trees used.

Adaboost. AdaBoost (adaptive Boosting) is a boosting meta-algorithm where a number of weak learners are combined into a weighted sum that represents the final output of the model.

3.4. Model hyperparameter optimisation

A number of model hyperparameters can be altered to affect the model performance. As the optimal hyperparameter values cannot be known *a priori*, an optimisation routine is employed to identify the best hyperparameter values for each model. A naïve method to do so would imply building a grid containing all possible combinations of selected hyperparameters and exhaustively evaluating each to select the best combination. However, this carries a significant cost due to the sheer number of combinations that are evaluated (especially in the case of multiple tuneable hyperparameters per model). Another

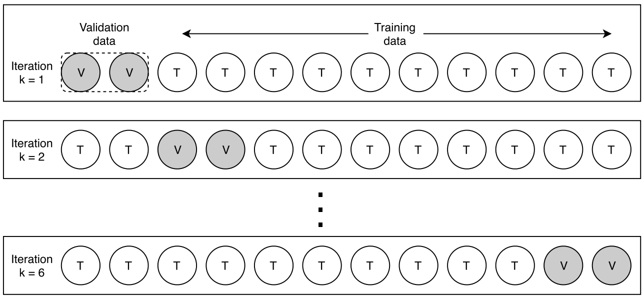

Fig. 2. Visual representation of k-folding for 12 data points and k=6. In this example, the dataset is split in six subsets, and for every value of k, a different subset is selected for validation whilst the rest are used for model training. This helps prove the robustness of the model and its hyperparameters.

Table 1
V1 and V2 dataset measurements used for model training.

#	Name	V1 Units	V2 Units
1	Vessel speed	knots	knots
2	Engine speed	rev/min	rev/min
3	Sea current[a]	knots	knots
4	Wind speed	Beauford scale	m/s
5	Wind direction[a]	12 direction bins	degrees
6	Daily ME FOC	t/day	t/day
7	Daily distance run	nm	nm
8	Sea state	Douglas sea scale	m[b]
9	Sea direction[a]	12 direction bins	degrees
10	Slip	%	%
11	Draft fore	m	m
12	Draft aft	m	m

[a] Relative to vessel.

[b] Wave height.

approach is to employ a random search implementation; there, all hyperparameter ranges are randomly sampled — usually producing more accurate results given a predefined number of draws (Bergstra and Bengio, 2012).

Considering the benefits provided by random search, a random search optimisation loop was set-up for all models. 1000 iterations were used for the hyperparameter optimisation of all models.

3.5. Selection of optimal model

In order to reasonably ensure that selected hyperparameter values are actually close to optimal and not merely overfitting the model, cross-validation is implemented in the form of k-folding. According to this, the training dataset is split into k subsets and an iterative process runs k times, using k-1 subsets for training and the remaining one for testing. A visual example of this procedure is presented in Fig. 2. Therefore, for each hyperparameter combination, several results are obtained and averaged.

Using the same technique for all models, allows us to identify the model that performs best while at the same time ensuring good generalisation capabilities.

3.6. Validation of models' generalisation capabilities

To test the model performance against the testing dataset, a number of metrics can be employed, each emphasising different model performance aspects. These will be analysed in the following subsections.

3.6.1. Explained variance

EV expresses the amount of variance that a model can capture from a given dataset. Having the true target output y, the estimated target output may be obtained as $\hat{y} = f(\mathbf{x})$, where $f(\cdot)$ refers to any derived model. Then, explained variance EV can be calculated as

$$EV(y, \hat{y}) = 1 - \frac{\sigma^2_{(y-\hat{y})}}{\sigma^2_y} \tag{19}$$

where σ_x refers to the standard deviation of parameter x. The best EV score is 1.0, obtained when $\sigma^2_{(y-\hat{y})} \to 0$, with lower values being worse.

3.6.2. Mean Absolute Error

Mean Absolute Error (MAE) corresponds to the expected value of the absolute (L^1 norm) error and can be calculated as

$$MAE(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^{n} \left| y_i - \hat{y}_i \right| \tag{20}$$

where n refers to the number of samples in y, and y_i to the Ith sample of y.

A variant of MAE is Mean Absolute Percentage Error (MAPE), expressed in a percent form, as

$$MAPE(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \cdot 100\% \tag{21}$$

At first glance, MAPE seems to combine the benefits of MAE with an easier interpretation; in practice, a major drawback is that it becomes numerically unstable when there exists an i such that $y_i = 0$. However, there exists a ceiling of 100% error for under-estimated outputs, whereas no ceiling exists for overestimation. Due to this, underestimated forecasts are wrongly promoted, when comparing between models. For the above reasons, MAPE is not a considered model comparison metric in this study.

3.6.3. Mean Squared Error

Following the same formulation as above, the Mean Squared Error (MSE) can be calculated as

$$MSE(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \hat{y}_i \right)^2 \tag{22}$$

MSE corresponds to the expected value of the quadratic error. Omitting the $\frac{1}{n}$ term, MSE becomes the L^2 loss function. Used as a cost function for optimisation purposes, both yield similar results.

Comparing to MAE, MSE puts a larger weight on major deviations between true and estimated targets. For the same reason, however, MAE remains more robust against outliers.

3.6.4. Mean Squared Logarithmic Error (MSLE)

The Mean Squared Logarithmic Error (MSLE) can be calculated as

$$MSLE(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^{n} \left(\ln\left(1 + y_i\right) - \ln\left(1 + \hat{y}_i\right) \right)^2 \tag{23}$$

MSLE tends to penalise more under-predictions rather than over-predictions. Furthermore, this loss function tends to under-penalise actual-estimated differences when both take large values; this can be of benefit when some observations momentarily take larger-than-usual values (e.g. full speed ahead at design draft).

3.6.5. Median Absolute Error

The Median Absolute Error (MedAE) can be calculated as

$$M_{ed}AE(y, \hat{y}) = \text{median}\left(\left| y_1 - \hat{y}_1 \right|, \dots, \left| y_i - \hat{y}_i \right| \right) \tag{24}$$

MedAE is especially robust to outliers due to only considering median performance.

Table 2
Descriptive statistics for V1 dataset (noon-reports).

	Distance run (nm)	Draft aft (m)	Draft forward (m)	M/E Speed (RPM)	M/E FOC (tn/day)	Vessel Speed (kn)	Propeller Slip (–)	Sea Current (kn)	Sea Direction (0–12)	Sea Force (0–12)	Wind Direction (0–12)	Beaufort wind force (0–12)
Count	745	745	745	745	745	745	745	745	745	745	745	745
Mean	327.32	7.67	6.38	110.93	21.68	14.56	0.14	0.26	5.06	3.62	4.98	4.53
σ	66.11	0.58	1.03	11.71	6.92	1.82	0.06	0.68	3.07	1.19	3.09	1.21
Min	24.00	6.20	3.50	47.40	0.70	1.00	0.00	−4.00	0.00	1.00	0.00	2.00
25%	301.00	7.30	5.85	104.70	15.80	13.87	0.11	0.00	3.00	3.00	3.00	4.00
50%	349.00	7.65	6.48	115.10	23.10	14.88	0.14	0.00	4.00	3.00	4.00	4.00
75%	374.60	8.17	7.10	120.00	27.50	15.75	0.17	0.60	8.00	4.00	7.00	5.00
Max	433.00	8.94	8.50	126.00	33.60	18.00	0.71	4.00	12.00	7.00	12.00	12.00

Table 3
Descriptive statistics for V2 dataset (ADLM).

	Distance run (nm)	Draft aft (m)	Draft forward (m)	M/E Speed (RPM)	M/E FOC (tn/day)	Vessel Speed (kn)	Propeller Slip (–)	Sea Current (kn)	Sea Direction (degrees)	Sea Force (m)	Wind Direction (degrees)	Wind Speed (m/s)
Count	768	768	768	768	768	768	768	768	768	768	768	768
Mean	283.23	14.24	13.67	62.76	38.22	11.80	0.09	−0.46	189.92	0.55	136.38	20.94
σ	43.20	4.46	5.10	6.70	7.86	1.80	0.08	0.77	123.42	0.65	135.03	9.39
Min	26.75	7.22	6.73	12.90	5.02	1.11	0.00	−4.85	0.05	0.00	2.24	0.41
25%	268.80	9.00	7.60	58.73	30.74	11.20	0.06	−0.92	64.91	0.00	20.34	13.64
50%	293.64	17.40	17.45	66.99	43.02	12.23	0.08	−0.44	171.68	0.30	49.99	20.76
75%	308.34	18.25	18.20	67.57	44.58	12.85	0.10	0.02	311.73	1.02	298.83	27.58
Max	386.05	18.30	18.30	68.58	54.17	16.09	0.70	2.47	359.82	3.05	350.07	52.53

Table 4
Model training time and considered hyperparameters along with their range.

Model	Training time (min.)	Hyperparameters tuned
LR	<1	None
LASSO	<1	None
	1	$\lambda \in \left[10^{-7}, 10^{5}\right]$
RR	<1	None
	2	$\lambda \in \left[10^{-7}, 10^{5}\right]$
Elastic net	<1	None
	1	$\alpha \in \left[10^{-7}, 10^{5}\right]$, $\lambda 1_{ratio} \in [0, 1]$
DTR	<1	None
	8	`max_tree_depth` $\in \left[10^{-7}, 10^{5}\right]$, `min_samples_split` $\in [2, 20]$, `min_samples_leaf` $\in [3, 20]$, `max_features` $\in [3, 10]$
RFR	<1	None
	34	`n_estimators` $\in [1, 200]$, `min_samples_split` $\in [2, 20]$
KNN	<1	None
	2	`n_neighbours` $\in [1, 50]$, `weights` $\in \left[\text{uniform}, \text{distance}\right]$, `leaf_size` $\in [2, 100]$
SVR	<1	None
	2	$\gamma \in 2^{[-15,3]}$, $C \in 2^{[-5,5]}$, $\nu \in \left[10^{-4}, 1\right]$
ETR	<1	None
	42	`n_estimators` $\in [1, 100]$, `max_features` $\in \left[10^{-5}, 1\right]$, `min_samples_split` $\in \left[10^{-5}, 1\right]$, `max_samples_leaf` $\in \left[10^{-5}, 0.5\right]$
ANN	<1	None
	38	`activation` $\in \left[\text{relu}, \text{tanh}\right]$, `alpha` $\in \left[10^{-8}, 10^{-1}\right]$, `hidden_layer_sizes` $\in [1, 50]$ or $[1, 50; 1, 50]$

3.6.6. *Coefficient of determination (R^2)*

The coefficient of determination (R^2) can be computed as

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2}{\sum_{i=1}^{n}\left(y_i - \bar{y}\right)^2} \tag{25}$$

where $\bar{y}$ is the mean value of y, i.e. $\bar{y} = \frac{1}{n}\sum_{1}^{n} y_i$.

Table 5
Correlation of FOC to other measured attributes for V1 dataset.

Attribute	Correlation coefficient (–)
M/E speed (RPM)	0.842532
Speed (kn)	0.596552
Sea state	0.325735
Wind speed (m/s)	0.314318
Propeller slip	0.140361
Draft aft (m)	0.095032
Draft forward (m)	−0.030739
Sea current (kn)	−0.087620
Wind direction	−0.193913
Sea direction	−0.198821

R^2 provides a representation of the quality of future model output (predictions). The best R^2 score is 1, with lower values being worse. Furthermore, taking EV equation (Eq. (19)), $\sigma^2_{y-\hat{y}}$ can be re-written into $\frac{1}{n}\sum_{i=1}^{n}\epsilon^2 - \bar{\epsilon}$, where $\epsilon = y - \hat{y}$. From that, we observe that when $\bar{\epsilon} \to 0$, EV's equation is transformed into Eq. (25).

4. Methodology application

Having discussed the modelling foundations in Section 3, this section presents a case study comparing data acquired from a reefer ship (V1) through its noon-reports and from a Newcastlemax Bulk carrier (V2) equipped with an ADLM system. In specific, the aim of this case study is to compare the performance of data-driven regression models in the estimation of FOC. Similar input parameters were used for both vessels (Table 1), albeit at different sampling rates.

To increase the transparency of the included case studies and building atop Fig. 1, the exact roadmap followed to obtain the results is as follows

1. Dataset is loaded.
2. Unneeded features are discarded.
3. Engine transients are identified and discarded.
4. Observations containing Not a Number (NaN) elements are identified and discarded.
5. Extract additional features (e.g. FOC, current, distance run, slip).
6. Discard points where slip ≤ 0 caused by round-off errors.
7. Split dataset into training and test set.
8. Scale training set and utilise same scaling parameters for test set.
9. Populate list of potential models
10. For every model in list of models implement k-folding cross-validation and

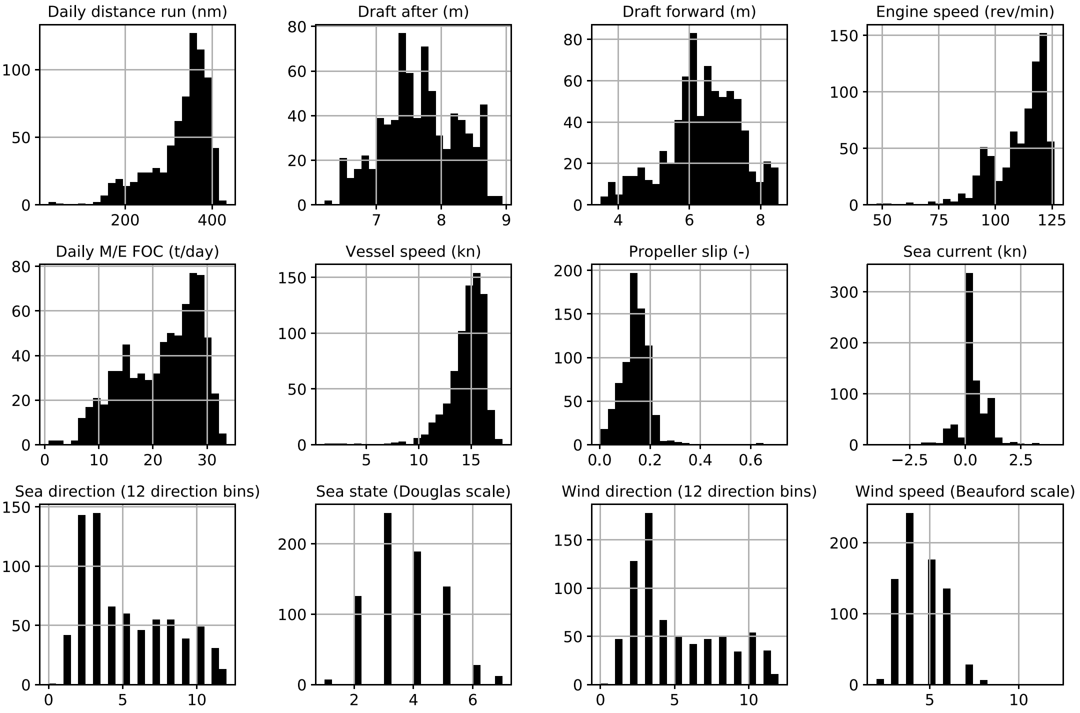

Fig. 3. Histogram plot of attributes used for model input after pre-processing for V1 dataset. The horizontal axis of each plot denotes the number of points corresponding to each histogram bin.

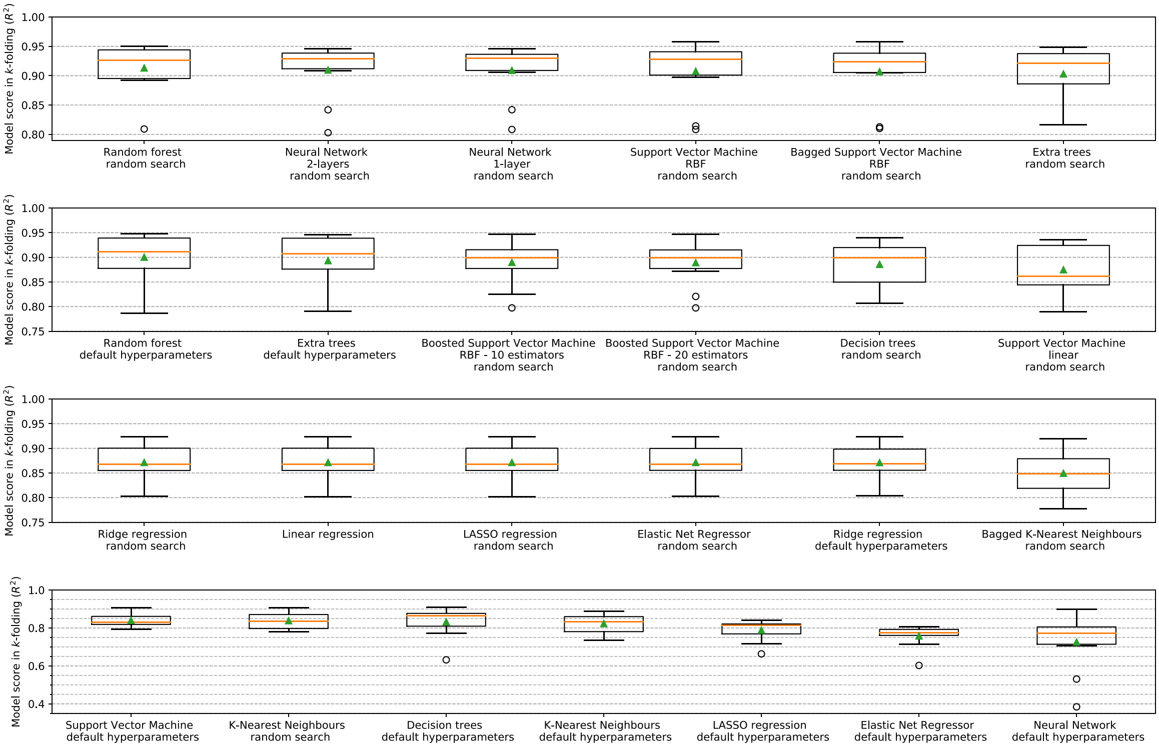

Fig. 4. Box plot of the R^2 obtained from different models and hyperparameters in K-folding for V1 dataset.

(a) train using default Scikit-learn hyperparameters.

(b) train using hyperparameter optimisation

i. identify search space for each hyperparameter.

ii. run random search over the search space and evaluate results.

(c) Using optimal hyperparameters, train a model using the whole training set (no validation).

(d) Evaluate model results on test set and compute performance metrics.

11. Evaluate all models based on metrics and reach overall conclusions.

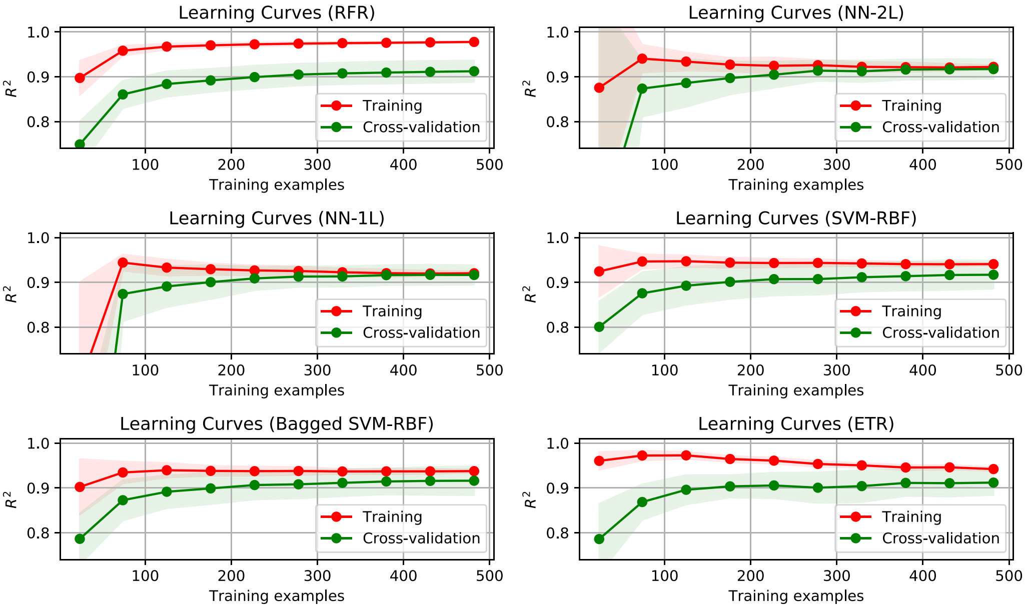

Fig. 5. Training curves for the models that achieved best R^2 at training for dataset V1. The lightly tinted areas denote the 95% confidence interval.

5. Results and discussion

Both vessel datasets were filtered to only include observations with the M/E speed being above the OEM lowest continuous running limit. This filtering was applied in order to only take into account the data points that correspond to relatively steady state conditions, without significant transient instances, e.g. manoeuvring.

Following this, 745 points were available for V1 and 768 for V2, corresponding to overall sailing periods of 2.5 years and 3 months respectively. The descriptive statistics results of these two datasets are shown in Tables 2 and 3.

The same amount of training points were selected from both datasets, in order to keep them similar, thus avoiding a biased comparison. Aiming for an approximately 80%–20% split in training and testing data, for both cases 603 training points were retained, with the remaining 20% being used for testing.

Datasets were then normalised following the steps elaborated in Section 3 in order to be used as training input for all relevant models. Each model was then trained using the default Scikit-learn (Pedregosa et al., 2011) hyperparameters. Additionally, random search over hyperparameters pertinent to each model was performed to identify optimal values. The hyperparameters that were considered for each model and their range of values are presented in Table 4 along with the time required for the models to train. Training was repeated 10 times for each dataset in order to get consistent results, which were then averaged for the two datasets in order to get one common, consistent result.

Both datasets exhibited similar training times, which was expected given that both contained the same number of training points.

In order to identify optimal models and hyperparameters, the coefficient of determination (R^2) (Glantz and Slinker, 2000) was evaluated for each model produced at each fold. For the evaluation of models post-training, a number of metrics are calculated, as discussed in Section 3.6.

5.1. Noon-report (V1) data

In this subsection, the methodology was applied for the case study of a reefer vessel dataset, populated through noon-reports. 834 data points were available, corresponding to approximately 2.5 years of noon-report data. Histograms of the measured parameters used for model training are shown in Fig. 3.

Additionally, as a brief investigation of the available training dataset, a correlation matrix was obtained, focusing on the daily FOC correlation to other measured parameters, shown in Table 5. Correlation matrices express the relationship between available quantities. As in this case, FOC is the independent variable, the relationship between all other quantities and FOC is examined. It is important to note that a correlation matrix, and correlation in general, only expresses the amount of linear relationship between two variables; any non-linear connection will not be captured by this.

An overview of the obtained results is presented in the box plots of Fig. 4. The line inside each box corresponds to the median (second quartile) score of this model in k-folding, the top and bottom of the box respectively correspond to the first and third quartiles. The whiskers represent the lowest point of data within 1.5 Interquartile Range (IQR) of the lowest quartile and the highest point of data within 1.5 IQR of the upper quartile. Accordingly, the mean of the dataset is noted by a triangle. Data points beyond the whisker range as shown individually in the form of hollow circles.

Fig. 4 shows how the default hyperparameters included in Scikit-learn are reasonably effective, as in many cases only a minuscule gain in R^2 was obtained after the hyperparameter optimisation loop. Additionally, most modelling attempts delivered overall good results, with a mean/median R^2 of over 85% in most cases. In this case, RFRs yielded the best results, closely followed by ANNs of 1- and 2-layers and the SVR with the RBF kernel.

Therefore, in this case, the evaluation of different models should be prioritised against a thorough hyperparameter optimisation. The only case where this was not true was in the case of regularised LR (i.e. LASSO, Ridge, and Elastic Net regressors), where the default regularisation term yielded suboptimal results. At the same time, due to the relatively small dataset size and lack of outliers, regularisation did not yield any improvement, with the unregularised LR providing some of the best results for this category of modelling techniques.

Regarding ensemble techniques, in the case of SVRs, bagging provided better results than boosting, but a single regressor still provided a better mean R^2, albeit with a slightly increased variance.

Another way of evaluating results is through their achieved R^2 when the number of training points is altered. This is visualised in Fig. 5, showing both training and cross-validation R^2 along with their respective 95% confidence intervals for the top-six models derived.

All six models achieve an R^2 of 85 to 90% when approximately 80 points are used, increasing to over 90% for more points. At the

Table 6
Performance indices for different modelling approaches — V1/testing dataset.

		Expl. variance	MeanAE	MSLE	MSE	MedianAE	R^2
		(%)	(tn/day)	(log(tn/day)^2)	((tn/day)^2)	(tn/day)	(%)
Linear Regression		87.25	1.674	0.050	6.108	1.217	86.79
LASSO	Default	74.94	2.742	0.075	12.544	2.463	72.86
	Randomised	87.25	1.674	0.050	6.108	1.217	86.79
Ridge	Default	87.22	1.677	0.050	6.125	1.208	86.75
	Randomised	87.25	1.674	0.050	6.110	1.216	86.78
Elastic Net	Default	71.80	2.888	0.080	13.815	2.676	70.11
	Randomised	87.24	1.676	0.050	6.115	1.218	86.77
Decision trees	Default	78.35	2.099	0.056	10.174	1.400	77.99
	Randomised	84.44	1.836	0.052	7.557	1.271	83.65
Random Forests	Default	87.85	1.506	0.049	5.753	0.995	87.55
	Randomised	88.75	1.454	0.047	5.297	1.009	88.54
KNN	Default	81.42	2.227	0.066	9.834	1.590	78.73
	Randomised	77.60	2.419	0.073	12.272	1.862	73.45
SVM	Default	88.08	1.521	0.047	5.576	1.025	87.94
	Randomised (RBF)	**91.52**	**1.226**	**0.042**	**3.950**	**0.817**	**91.46**
	Randomised (linear)	88.44	1.566	0.049	5.506	1.066	88.09
Extra trees	Default	89.65	1.405	0.045	4.944	0.840	89.31
	Randomised	88.89	1.434	0.047	5.192	1.011	88.77
Boosting	SVR ×10	90.77	1.315	0.043	4.311	0.911	90.67
	SVR ×20	90.94	1.299	0.042	4.242	0.910	90.82
Bagging	KNN	81.62	2.198	0.065	9.996	1.560	78.37
	SVR	91.02	1.292	0.043	4.218	0.886	90.87
NN	Default	85.02	1.892	0.054	7.068	1.411	84.71
	Randomised (1-layer)	89.52	1.414	0.043	4.869	0.945	89.47
	Randomised (2-layer)	88.99	1.432	0.044	5.121	0.926	88.92

same time, the RFR and ETR models exhibit significantly reduced uncertainty compared to other models, especially when a small number of points is used. After approximately 300 points, all models seem to plateau, without any tangible increase in model performance when additional model points are included. Nevertheless, the RFR model exhibited high variance that can be seen in the large gap that is present between the training and cross-validation curves. This signifies that while RFR presented the best results, there also exist a minor tendency to model the random noise in the data additionally to the actual features (overfitting).

Having identified that RFR overall exhibited the best R^2 in K-folding, the same parameters are now tested in the dataset held aside for validation. There, an $R^2 = 88.5\%$ was obtained, along with a mean error of 1.45 t/day and a median error of 1.0 t/day. While normally this would be the only model that would be evaluated against the testing dataset, in the case of this investigation, the performance of all models is included in Table 6. This is due to the fact that the aim of this study is to investigate the performance of different models whilst obtaining useful insights and not necessarily to derive a single model to model the FOC of this specific vessel.

Through Table 6, it is observed that in the testing dataset ANNs performed slightly better than RFRs, obtaining an R^2 increased by approximately 0.75%. Furthermore, the RBF-based SVR obtained an even increased R^2 at approximately 91.50%. This discrepancy can be justified by the increased variance that this model exhibited at the validation stage.

5.2. Automatic sensor-based DAQ (V2) data

In this second case study, 768 hourly-collected data points acquired from a bulk carrier of approximately 200,000 DWT equipped with an ADLM system are analysed. In this case, while ADLM systems usually provide a wider range of measured parameters, the same parameters as in the case of the noon-report data are considered (Table 3). This decision was made so that the study focuses on the quality of the data provided by the different data acquisition strategies. However, given that in this case, data is provided at an hourly sampling rate, in order to retain the same amount of training and testing points so that an unbiased comparison of the two data acquisition strategies is made, a three-month time window is selected. Filtering conditions are similar to the ones presented in Section 5.1, without the need to filter for daily steam hours. Following this filtering process, 603 points were retained and used for training. A histogram of these remaining data points is shown in Fig. 6.

Table 7
Correlation of FOC to other measured attributes for V2 case study.

Attribute	Correlation coefficient (–)
M/E speed (RPM)	0.897682
Draft aft (m)	0.848487
Draft forward (m)	0.846745
Speed (kn)	0.571559
Sea direction	−0.053529
Propeller Slip	0.000047
Wind direction	−0.054408
Sea state	−0.059535
Sea current (kn)	−0.066582
Wind speed (m/s)	−0.168847

A correlation matrix was also derived for this case, providing insight on how the daily FOC correlates with the other measured parameters, shown in Table 7. Evaluating against Table 5, it is inferred that in both cases the M/E speed (and, therefore, also vessel speed) is identified as significantly correlated with the FOC of the vessel. However, whilst in this case, both draft forward and aft have a high increased correlation with FOC, which is not the case for dataset V1. This is mostly due to the fact that, as can be seen in Fig. 6, vessel V2 essentially only takes two draft values: one for ballast condition and one for laden conditions, whereas vessel V1 operates in a wide range of draft values. This can be explained by the fact that vessel V1 is a reefer vessel, going from port to port and loading/unloading containers; rarely being neither at full load nor at ballast condition. At the same time, vessel V2 is a large bulk carrier, always leaving at full load from the departure port, heading to destination to unload all cargo, and then moving at ballast condition

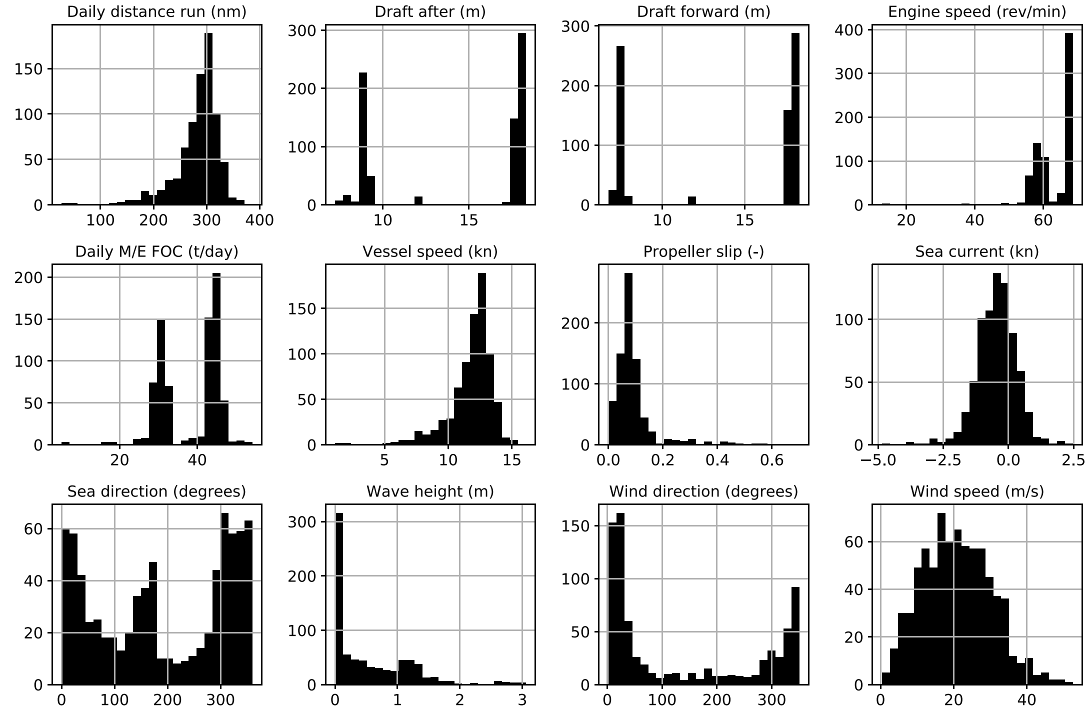

Fig. 6. Histogram plots of attributes used for model input after pre-processing for V2 dataset. The horizontal axis of each plot denotes the number of points corresponding to each histogram bin.

from that port to another, where she will load cargo again; essentially alternating between these two load conditions.

Following the model training, the derived R^2 coefficient of each model is visualised in Fig. 7. From the results presented, it can be deduced that the best performing model was the ETR, with hyperparameter optimisation through random search, achieving an average coefficient of determination (R^2) of 97.7% and a median of over 98%. Random forests also yielded a comparable R^2, followed by 1-, and 2-layer Neural Networks.

The learning curves of this dataset are presented in Fig. 8. ETR, RFR, and 1-layer ANN models performed similarly, with an R^2 of approximately 90% when 100 training points are used. As the number of points increases, the cross-validation R^2 is asymptotically approaching 100%. Nevertheless, in the case of the 2-layer ANN a high confidence interval is obtained, coupled with a lacking performance when a small amount of points is present. At the same time, when a larger number of points is used for training, a promising slope is present in the cross-validation R^2 curve, signifying that the R^2 coefficient may be increased for cases where an even larger number of training points is used.

Following the selection of ETR as optimal model due to providing the highest mean R^2 value, this model was evaluated on the testing dataset to ensure its generalisation capabilities in previously-unseen input.

In this case, ETR obtained an R^2 of 97.3%, with a mean error of 0.5 t/day and a median error of 0.2 t/day.

As discussed in the previous subsection, the performance of all models is included in Table 8. Here, is inferred that ETR yielded the best results across all metrics. At the same time, RFR models also yielded comparable performance. However, the employed ANN models that performed exceptionally well at validation, did not perform equally well at the testing stage yielding a mean average error of over 0.9 t/day.

5.3. Key findings

In both cases, the best results were obtained from ETRs, ANNs, and RFRs, followed by SVRs. For both datasets, highly accurate results were obtained against the testing dataset, with an R^2 of approximately 89% in V1 dataset and 97% in the case of V2 dataset.

Similarly, in both cases the selection of an optimal model architecture had a higher impact on the results, compared to hyperparameter optimisation. Albeit at a significant computational cost, simultaneously performing optimisation evaluating different architectures and hyperparameter values is the only way to ensure an optimal model selection. In addition, naïve models such as LR provided comparable results, yielding a deviation of only 4% from the optimal in the case of V2 and of 5% in the case of V1.

On the other hand, is was observed that for dataset V2 where data were acquired through an ADLM system, data from a 3-month period suffice to create a model with an R^2 than exceeds that of the noon-report model (V1), after a data collection period of 2.5 years.

Furthermore, when, following the process outlined in Section 3.2.2, trim and draft amidships were extracted from the draft forward & aft measurements provided but in both V1 and V2 cases, the R^2 difference was estimated at ±0.1%.

6. Conclusions

This study presents a data-driven methodology for the estimation of M/E FOC of sailing vessels, evaluating the results obtained when the data source is noon-reports compared to an ADLM system. The main findings of the research conducted are as follows:

- The derived models can accurately predict the FOC of vessels sailing under different load conditions, weather conditions, speed, sailing distance, and drafts.
- Using noon-report data, an R^2 of approximately 90% was obtained through the best performing modelling approaches.
- ADLM systems can increase modelling R^2 by 5–7% compared to noon-reports, whilst reducing the required data acquisition period by up to 90%.
- Optimising hyperparameters may increase model's R^2 coefficient, but evaluating several modelling architectures should be the first step.

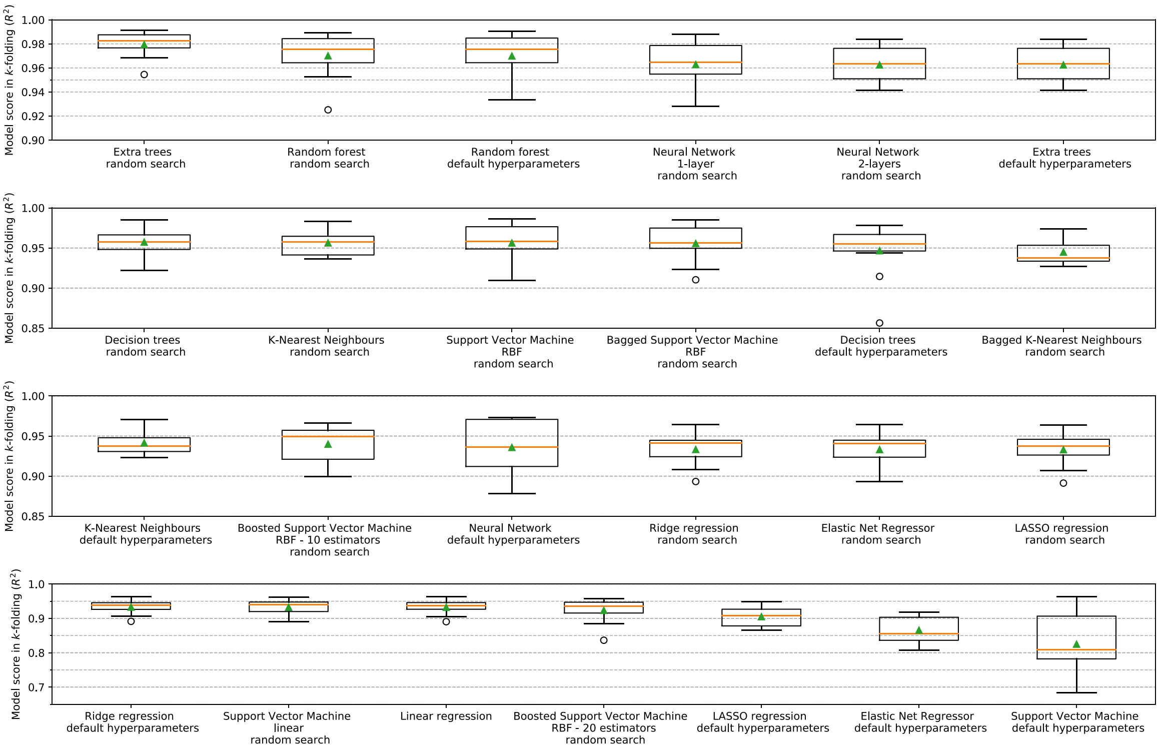

Fig. 7. Box plot of the R^2 obtained from different models and hyperparameters in K-folding for V2 dataset.

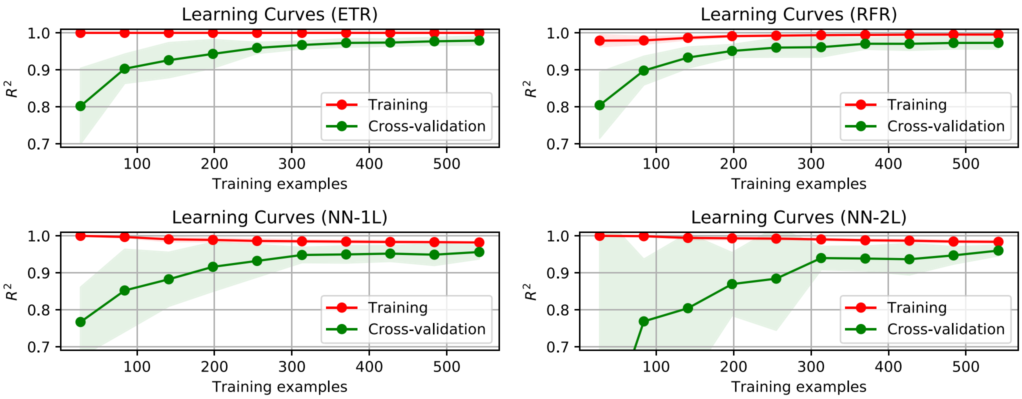

Fig. 8. Training curves for the models that achieved the best R^2 at training for dataset V2. The lightly tinted areas denote the 95% confidence interval.

- ETRs, RFRs, SVRs, and ANNs yielded the best performance results for both datasets, but LR, a significantly simpler model, attained comparable results.
- Due to the inherent limitations of DTR-based models (e.g. ETRs, RFRs), these models should only be preferred in cases where no extrapolation is required.
- The quality of the model output correlates with the quality of its training input; e.g. different vessel operating profiles affect how the effects of vessel draft are perceived to affect FOC.
- Feature extraction did not help attain any perceivable increase in model performance.

The proposed methodology was elaborated and showcased through case studies referring to a reefer vessel where data acquisition was done manually, through noon-reports and a Newcastlemax bulk carrier, equipped with an ADLM system. This being a black-box approach, no additional domain knowledge is required for such models to be derived. Accordingly, this methodology can be applied to essentially any vessel. Especially in the case where an ADLM system is installed, data acquisition period can be significantly reduced. It is expected that this methodology will be used to create models that will help track vessel performance degradation, optimise shipping operations, accurately reflect ship emissions and eventually be used as a basis for route optimisation purposes.

Acknowledgements

The work presented in this paper is partially funded by Integrated Ship Energy & Maintenance Management System (ISEMMS) project. ISEMMS project has received research funding from Innovate UK under Project No. 102435. This publication reflects only the authors' views and Innovate UK is not liable for any use that may be made of the information contained within. The authors would like to also acknowledge the assistance of the shipping company and consultancy firm that provided the data utilised in this study.

Table 8
Performance indices for different modelling approaches — V2/testing dataset.

		Expl. variance (%)	MeanAE (tn/day)	MSLE ($\log(\text{tn/day})^2$)	MSE ($(\text{tn/day})^2$)	MedianAE (tn/day)	R^2 (%)
Linear Regression		94.49	1.409	0.006	3.670	1.106	94.48
LASSO	Default	93.77	1.331	0.005	4.151	0.715	93.76
	Randomised	94.51	1.410	0.006	3.658	1.134	94.50
Ridge	Default	94.48	1.413	0.006	3.671	1.144	94.48
	Randomised	94.64	1.404	0.005	3.570	1.203	94.63
Elastic Net	Default	89.95	1.732	0.014	6.696	1.091	89.93
	Randomised	94.69	1.398	0.005	3.532	1.201	94.69
Decision trees	Default	95.13	0.679	0.003	3.255	0.269	95.11
	Randomised	94.64	0.844	0.005	3.569	0.318	94.63
Random Forests	Default	96.27	0.570	0.003	2.487	0.192	96.26
	Randomised	96.38	0.564	0.003	2.405	0.234	96.38
KNN	Default	95.83	0.943	0.004	2.802	0.484	95.79
	Randomised	95.90	0.675	0.002	2.739	0.242	95.88
SVM	Default	73.11	1.843	0.039	18.161	0.809	72.69
	Randomised (RBF)	95.98	0.895	0.003	2.713	0.465	95.92
	Randomised (linear)	94.49	1.445	0.005	3.672	1.325	94.48
Extra trees	Default	96.22	0.756	0.008	2.522	0.311	96.21
	Randomised	**97.31**	**0.534**	**0.002**	**1.804**	**0.178**	**97.29**
AdaBoost	SVR ×10	95.89	1.142	0.004	2.737	0.740	95.89
	SVR ×20	95.23	1.240	0.004	3.171	0.911	95.23
Bagging	KNN	95.26	0.966	0.006	3.210	0.454	95.17
	SVR	95.94	0.925	0.003	2.751	0.467	95.86
NN	Default	96.96	0.903	0.007	2.035	0.501	96.94
	Randomised (1-layer)	90.87	1.314	0.016	6.081	0.699	90.86
	Randomised (2-layer)	95.58	0.939	0.006	2.984	0.468	95.51

References

Alpaydin, E., 2014. Introduction to Machine Learning. MIT press.

Bergstra, J., Bengio, Y., 2012. Random search for hyper-parameter optimization. J. Mach. Learn. Res. 13 (Feb), 281–305.

Beşikçi, E.B., Arslan, O., Turan, O., Ölçer, A., 2016. An artificial neural network based decision support system for energy efficient ship operations. Comput. Oper. Res. 66, 393–401. http://dx.doi.org/10.1016/j.cor.2015.04.004, URL http://www.sciencedirect.com/science/article/pii/S0305054815000842.

Bialystocki, N., Konovessis, D., 2016. On the estimation of ship's fuel consumption and speed curve: A statistical approach. J. Ocean Eng. Sci. 1 (2), 157–166. http://dx.doi.org/10.1016/j.joes.2016.02.001, URL http://www.sciencedirect.com/science/article/pii/S2468013315300127.

Bishop, C.M., 2006a. Pattern Recognition and Machine Learning. In: Information Science and Statistics, Springer-Verlag, New York, p. 738, URL http://electronicimaging.spiedigitallibrary.org/article.aspx?doi=10.1117/1.2819119.

Bishop, C.M., 2006b. Pattern Recognition and Machine Learning (Information Science and Statistics). Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Breiman, L., 1998. Arcing classifiers. Ann. Statist. 26 (3), 801–849. http://dx.doi.org/10.1214/aos/1024691079, URL http://projecteuclid.org/euclid.aos/1024691079.

Breiman, L., Friedman, J.H., Olshe, R.A., Stone, C.J., 1984. Classification and Regression Trees. Taylor & Francis.

Chang, C.-C., Lin, C.-J., 2011. LIBSVM: A library for support vector machines. ACM Trans. Intell. Syst. Technol. (TIST) 2 (3), 27.

Cichowicz, J., Theotokatos, G., Vassalos, D., 2015. Dynamic energy modelling for ship life-cycle performance assessment. Ocean Eng. 110, 49–61. http://dx.doi.org/10.1016/j.oceaneng.2015.05.041, URL http://www.sciencedirect.com/science/article/pii/S0029801815002413.

Cipollini, F., Oneto, L., Coraddu, A., Murphy, A.J., Anguita, D., 2018. Condition-based maintenance of naval propulsion systems with supervised data analysis. Ocean Eng. 149, 268–278. http://dx.doi.org/10.1016/J.OCEANENG.2017.12.002, URL https://www.sciencedirect.com/science/article/pii/S0029801817307242.

Clarke, B., Fokoue, E., Zhang, H.H., 2009. Principles and Theory for Data Mining and Machine Learning. Springer Science & Business Media.

Coraddu, A., Oneto, L., Baldi, F., Anguita, D., 2017. Vessels fuel consumption forecast and trim optimisation: A data analytics perspective. Ocean Eng. 130, 351–370. http://dx.doi.org/10.1016/j.oceaneng.2016.11.058, URL http://www.sciencedirect.com/science/article/pii/S0029801816305571.

Glantz, S., Slinker, B., 2000. Primer of Applied Regression & Analysis of Variance. McGraw-Hill Education.

Hastie, T., Tibshirani, R., Friedman, J., 2009. The Elements of Statistical Learning. In: Springer Series in Statistics, Springer New York, New York, NY, http://dx.doi.org/10.1007/978-0-387-84858-7, URL https://link.springer.com/content/pdf/10.1007%2F978-0-387-84858-7. http://link.springer.com/10.1007/978-0-387-84858-7.

Hu, L.-Y., Huang, M.-W., Ke, S.-W., Tsai, C.-F., 2016. The distance function effect on k-nearest neighbor classification for medical datasets. SpringerPlus 5 (1), 1304. http://dx.doi.org/10.1186/s40064-016-2941-7, URL http://www.ncbi.nlm.nih.gov/pubmed/27547678. http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4978658.

Lazakis, I., Gkerekos, C., Theotokatos, G., 2019. Investigating an SVM-driven, one-class approach to estimating ship systems condition. Ships Offshore Struct. 14 (5), 432–441. http://dx.doi.org/10.1080/17445302.2018.1500189.

Lazakis, I., Raptodimos, Y., Varelas, T., 2018. Predicting ship machinery system condition through analytical reliability tools and artificial neural networks. Ocean Eng. 152, 404–415. http://dx.doi.org/10.1016/J.OCEANENG.2017.11.017, URL https://www.sciencedirect.com/science/article/pii/S0029801817306844?via.

Lu, R., Turan, O., Boulougouris, E., Banks, C., Incecik, A., 2015. A semi-empirical ship operational performance prediction model for voyage optimization towards energy efficient shipping. Ocean Eng. 110, Part B, 18–28. http://dx.doi.org/10.1016/j.oceaneng.2015.07.042, URL http://www.sciencedirect.com/science/article/pii/S0029801815003558.

Lundh, M., Garcia-Gabin, W., Tervo, K., Lindkvist, R., 2016. Estimation and optimization of vessel fuel consumption. IFAC-PapersOnLine 49 (23), 394–399. http://dx.doi.org/10.1016/j.ifacol.2016.10.436, https://www.sciencedirect.com/science/article/pii/S2405896316320249.

MAN B&W Diesel A/S, 2004. Instruction Book 'Operation' for 50-108MC/MC-C Engines, second ed. Copenhagen, DK.

Mao, W., Rychlik, I., Wallin, J., Storhaug, G., 2016. Statistical models for the speed prediction of a container ship. Ocean Eng. 126, 152–162. http://dx.doi.org/10.1016/j.oceaneng.2016.08.033, URL https://www.sciencedirect.com/science/article/pii/S0029801816303699.

Meng, Q., Du, Y., Wang, Y., 2016. Shipping log data based container ship fuel efficiency modeling. Transp. Res. B 83, 207–229. http://dx.doi.org/10.1016/j.trb.2015.11.007, URL http://www.sciencedirect.com/science/article/pii/S0191261515002386.

Moreno-Gutiérrez, J., Calderay, F., Saborido, N., Boile, M., Rodríguez Valero, R., Durán-Grados, V., 2015. Methodologies for estimating shipping emissions and energy consumption: A comparative analysis of current methods. Energy 86, 603–616. http://dx.doi.org/10.1016/j.energy.2015.04.083, URL https://www.sciencedirect.com/science/article/pii/S0360544215005502.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al., 2011. Scikit-learn: Machine learning in Python. J. Mach. Learn. Res. 12 (Oct), 2825–2830.

Perera, L.P., Mo, B., 2018. Ship performance and navigation data compression and communication under autoencoder system architecture. J. Ocean Eng. Sci. 3 (2), 133–143. http://dx.doi.org/10.1016/j.joes.2018.04.002, URL https://linkinghub.elsevier.com/retrieve/pii/S2468013317301109.

Petersen, J.P., Winther, O., Jacobsen, D.J., 2012. A machine-learning approach to predict main energy consumption under realistic operational conditions. Ship Technol. Res. 59 (1), 64–72. http://dx.doi.org/10.1179/str.2012.59.1.007, URL http://www.tandfonline.com/doi/full/10.1179/str.2012.59.1.007.

Raptodimos, Y., Lazakis, I., 2018. Using artificial neural network-self-organising map for data clustering of marine engine condition monitoring applications. Ships Offshore Struct. 13 (6), 649–656. http://dx.doi.org/10.1080/17445302.2018.1443694, URL https://www.tandfonline.com/doi/full/10.1080/17445302.2018.1443694.

Ronen, D., 2011. The effect of oil price on containership speed and fleet size. J. Oper. Res. Soc. 62 (1), 211–216. http://dx.doi.org/10.1057/jors.2009.169.

Russell, S., Norvig, P., 2010. Artificial Intelligence: A Modern Approach. Pearson, URL http://linkinghub.elsevier.com/retrieve/pii/S0004370211000142.

Schmidhuber, J., 2015. Deep learning in neural networks: An overview. Neural Netw. 61, 85–117.

Simonsen, M., Walnum, H., Gössling, S., 2018. Model for estimation of fuel consumption of cruise ships. Energies 11 (5), 1059. http://dx.doi.org/10.3390/en11051059, URL http://www.mdpi.com/1996-1073/11/5/1059.

Smola, A.J., Schölkopf, B., 2004. A tutorial on support vector regression. Stat. Comput. 14 (3), 199–222.

Stopford, M., 2009. Maritime Economics, third ed. Routledge.

Theodoridis, S., Koutroumbas, K., 2008. Pattern Recognition, fourth ed. Academic Press.

Trodden, D., Murphy, A., Pazouki, K., Sargeant, J., 2015. Fuel usage data analysis for efficient shipping operations. Ocean Eng. 110, Part B, 75–84. http://dx.doi.org/10.1016/j.oceaneng.2015.09.028, URL http://www.sciencedirect.com/science/article/pii/S0029801815005004.

Tsitsilonis, K.M., Theotokatos, G., 2018. A novel systematic methodology for ship propulsion engines energy management. J. Cleaner Prod. 204, 212–236. http://dx.doi.org/10.1016/j.jclepro.2018.08.154.

Wang, S., Ji, B., Zhao, J., Liu, W., Xu, T., 2018. Predicting ship fuel consumption based on LASSO regression. Transp. Res. D 65, 817–824. http://dx.doi.org/10.1016/J.TRD.2017.09.014, URL https://www.sciencedirect.com/science/article/pii/S1361920917302109.

Yao, Z., Ng, S.H., Lee, L.H., 2012. A study on bunker fuel management for the shipping liner services. Comput. Oper. Res. 39 (5), 1160–1172. http://dx.doi.org/10.1016/J.COR.2011.07.012, URL https://www.sciencedirect.com/science/article/pii/S030505481100205X.