EKSPLOATACJA I NIEZAWODNOŚĆ MAINTENANCE AND RELIABILITY

Eksploatacja i Niezawodnosc – Maintenance and Reliability

Volume 23 (2021), Issue 2

journal homepage: http://www.ein.org.pl

Article citation info:
Matuszczak M, Żbikowski M, Teodorczyk A. Predictive modelling of turbofan engine components condition using machine and deep learning methods. Eksploatacja i Niezawodnosc – Maintenance and Reliability 2021; 23 (2): 359–370, http://doi.org/10.17531/ein.2021.2.16.

Predictive modelling of turbofan engine components condition using machine and deep learning methods

Indexed by: Web of Science Group

Michał Matuszczak[a,b,*], Mateusz Żbikowski[a], Andrzej Teodorczyk[a]

[a] Institute of Heat Engineering, Faculty of Power and Aeronautical Engineering, Warsaw University of Technology, Nowowiejska 21/25, 00-665 Warsaw, Poland
[b] General Electric Company Polska sp. z. o. o., Al. Krakowska 110/114, 02-256 Warsaw, Poland

Highlights

- 0-10 condition rank of a turbofan life limiting component is predicted.
- Environmental and engine sensors data preceding the condition observation are used.
- Ensemble meta-model of neural networks shown the best performance.
- Support vector machines and gradient boosted models did not match neural nets.
- Linear model demonstrated the worst performance among considered models.

Abstract

The article proposes an approach based on deep and machine learning models to predict a component failure as an enhancement of condition based maintenance scheme of a turbofan engine and reviews currently used prognostics approaches in the aviation industry. Component degradation scale representing its life consumption is proposed and such collected condition data are combined with engines sensors and environmental data. With use of data manipulation techniques, a framework for models training is created and models' hyperparameters obtained through Bayesian optimization. Models predict the continuous variable representing condition based on the input. Best performed model is identified by detemining its score on the holdout set. Deep learning models achieved 0.71 MSE score (ensemble meta-model of neural networks) and outperformed significantly machine learning models with their best score at 1.75. The deep learning models shown their feasibility to predict the component condition within less than 1 unit of the error in the rank scale.

Keywords

reliability, prognostics, deep learning, machine learning, gas turbine, turbofan engine, neural network, condition-based maintenance.

1. Introduction

A modern aircraft's turbofan engine is a complex mechanical system with numerous components that need to be properly maintained to continue its safe and profitable operation. As the components deteriorate they need to be replaced or repaired which drive the engine off wing for often time consuming overhaul [8] and creates a cost burden requiring proper engine fleet management to continue the aircraft operation [18].

Aircraft engine components condition is assessed on recurring inspections and compared to the limits provided by the engine manufacturer which constitute the Instructions for Continued Airworthiness approved and controlled by the regulatory agency in a form of an engine manual [16]. The engine manual limits proposed by the engine manufacturer are based upon understanding of the physics behind the particular wear out scheme and the condition progression until the part cannot be operated any longer and has to be replaced.

With the complexities of loads that parts are exposed to a variety of competing failure modes occuring at different stages of part's age and progresing at different rates comes with significant impact of environmental factors like volcanic activity [12] and air contaminants presence like dust aerosols as seen in a test [6] and in operation [26].

Additionally, an ease of performing a visual on-wing inspection of the hardware depends on its location in the engine and capability of the inspecting crew and its equipement. Thus with all the factors combined the actual confirmation of the part condition is not always feasible.

It is common that engine components wear occurs at different rates and single components compete in being limiting for the engine useful life. Hence a prediction of the current state of the wear of the components becomes a crucial task in the fleet management. With the development of health monitoring systems and on board diagnostics technologies deployment, a significant amount of data has become available for engineers to analyze which enables enhancement of classical condition based maintenance [29].

In the light of the latest research based in the field of predicting components life this paper proposes a data-driven approach for an aviation turbofan engine.

(*) Corresponding author.

E-mail addresses: M. Matuszczak - michal.matuszczak2.dokt@pw.edu.pl, M. Żbikowski - mateusz.zbikowski@pw.edu.pl, A. Teodorczyk - andrzej.teodorczyk@itc.pw.edu.pl

2. Failure prediction methods overview

2.1 Prognostics approaches

There exist numerous examples of attempts to predict the component failure of a part or the entire system depending on the problem at hand, design phase and data available.

In the concept design phase where numerical models are available Ning Baojun et al. proposed a method to incorporate boundary condition uncertainity into the FEA of a turbofan engine combustor to obtain a stochastic life predition [4]. Another approach is presented by Echarda et al. [10] where a SARFAN's aviation engine blade support is analyzed with a variation of geometry, material properties and load variation to computionally capture the life prediction and its probability. These models can be very accurate and deliver useful information about the type design, however a good understanding of the failure mode is necessary.

With available failure data one can apply different predictive methods. In their article Yang et al. explore potential for matching the failure times of an aeronautical equipment components to probability distributions to the outcome of finding that the normal distribution to best reflects the actual life distribution [38]. Whereas some cases show promise of normal distribution use, the others like the subject studied in the other paper by Yang et al. indicate 3 parameter Weibull to best represent failure probability of airborne equipment [37]. These modelling approach enables the engineer to make predictions of the part failure based on the sample of fielded hardware and employing statistical methods in place of finite element computations with a challenge of collecting sufficient amount of well comprehended data.

The other researches focus on the engine health monitoring and fault diagnostics, where engine sensors are used to look for a signal of a deteriorating engine health or a faulty component. Turbofan engine health degradation and prognostics of the remaining useful life (RUL) was deployed by Zaidan et al. [41] with a use of a Bayesian Network Regression. Xiu et al. present an aviation turbofan engine fault diagnosis scheme based on deep belief network (DBN) [36]. The neural network composed of mulitple layers forming restricted Botlzmann machines (RBM) succesfully modeled engine systems and engine sensory data have been fed into the model and corresponding engine fault state have been predicted.

Another deep learning model is researched in a paper by Sina Tayarani-Bathaie et al. and revealed that dynamic neural networks based on multi-layer perceptron (MLP) networks demonstrated promising performance in prediction of a turbofan engine fault [31]. Also, Heimnes in [14] reports a satisfactory results in RUL prediction with a MLP classifier.

In [19] the researchers are introducing useful classifications of the AI-based methodologies used in the aerospace industry for systems health management; (1) knowledge-based, (2) probabilistic and (3) data-driven with authors pointing out towards the growing interest paid by the scientific community to the deep learning methods. Sikorska et al. [30] report successes in the field of prognostics and prediction of RUL by artificial neural networks (ANN) and making them a separate category of RUL prediction models noting their ability to handle noisy data. Pawełczyk et al [25] have recently reported a succesful use of machine learning methods to predict the condition of high pressure compressor in a stationary gas turbine.

A different take on asset failure prediction is presented in the works of Yoon et al. where deep generative models in semi-supervised learning scheme have been implemented to predict estimated time to failure and show that data-driven approaches are alternatives to the physics-driven modelling [40]. In the presented study for the sparse labelled turbofan data the variational autoencoders have delivered great results over the gated recurrent units (GRU) and long short-term memory (LSTM) network architectures.

Among other network architectures deep convolutional neural networks (CNN) have been demonstrated by Babu et al. [3] to be feasible in predicting a capture a non-linear relationship between RUL and sensor data.

Having in mind the mentioned researches in the field of prognostics, the deep learning methods deliver promising results replacing physics based models provided sufficient understanding of the matter is reached as authors demonstrated in number of publications [23].

2.2. Target variable in researches

An important role in prognostics and health management plays a systems health index (HI) as it reflects the system condition and its potential to perform its function throughout the system useful life. The index is widely used concept across researches based in various industries ranging from electronics equipment, through heavy machinery to the aviation industry.

A paper published by Amir et al. has researched a condition-based health index concept where overall health index was calculated based on the individual indicators [1] and used a 10-grade scale differentiating a system condition from good to bad and enabling to categorize the particular system units. In power transformer application a health index ranging from 0 to 1 have been presented by Lata et al. [2] and incorporated a various input relevant to that particular system to establish the resulting index value.

In the case of turbofan engine, a health index based on engine sensor flight by flight data were used to establish and predict a high-pressure compressor deterioration [33,34].

Another interesting way to develop a health index out of turbofan engine sensor readings have been proposed in [5] where a step by step aggregation of the normalized feature values was proposed. In such arrangement a growing health index would cumulate over time of operation and judgements about RUL can be made.

Based on the solid fundaments established by the research community the subject of this paper uses a condition-based health index with 10 grade scale.

3. Problem description

Turbofan engine components are inspected reccurently at least as often as recommended by the engine manufacturer thus providing a valueable condition data. The considered component operates on the condition based maintenance scheme. The participating engines have been monitored for a period from third quarter of 2014 to first quarter of 2020 to obtain one of the hot gas path component data.

The obvious challenge is in the formulation of the life prediction problem. The intent is to determine, based on available information, at what stage of degradation the given component is. A very efficient technique to determine a moment when a given system would fail is RUL estimation. As the authors of the [11] presented, RUL can be determined by use of a degradation characteristic of an aviation engine as input variable to obtain a survival function that later can be used to predict moment of a probable failure. A degradation characteristic is specific to the system and may depend on the physics of a considered wear out mechanism. For a gas turbine it could be an exhaust gas temperature [14] or a compressor recoup pressure [25], both being related to the system wear out and continuous trend of either could be a signature that can be used to judge incoming expiration of useful life.

However, in the researched system, the component wear out, despite progressing with time, is not picked up by engine sensors and thus a trend as such cannot be the degradation characteristic. Also, there exist no spike in any of the sensor readings when the component reaches the condition at which it is desirable to be removed to avoid further costly engine damage and potential impact to the customers operation schedule. Therefore an anomaly detection methods are not available in this case.

Regardless of its lack of visibility in the engine system sensors, the component life is limiting to the entire system. To adress this problem, the authors propose to use component condition data and the engine operation data preceding the inspection at which the condition rank

was collected. Then, by the means of data science; conducting data cleaning, feature engineering and feature selection train the models to predict the condition. The expectation behind such an approach is that there might be non-obvious or hard to quantify differences between the engines so that the component in one engine fails at different time that the other. The difference could be operational: frequently fully loaded aircraft, high altitute of an airports used, short climb path, environmental: air aerosols and dusts present, high temperatures at the airport or manufacturing related; tolerances stacking up results in different loads that the component is exposed to. It is expected that, since a turbofan engine is a closed system, these differences can be determined by sensors not directly related to the considered component and those that cannot be otherwise used as a degradation characteristic. Such differences accumulated over the operation time could be resonsible for the condition rank progression at different rate and modern models are anticipated to fit to them.

Due to the data amount, complexity and high non-linearity neural networks are main focus of the research, however machine learning models are used for comparison basis. Once models are developed, it would be possible to use them to monitor the remaining fleet and plan maintenance provided the sensor data would be provided as an input to the models.

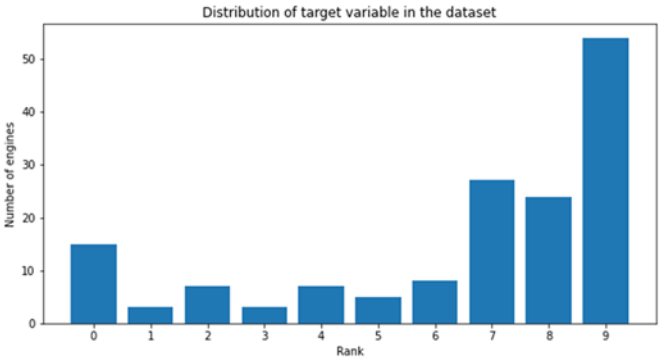

Fig. 1. The number of engines per rank collected during the monitoring program and used as the dataset for this research

Over 150 engines have participated in the monitoring program, running at five different thrust ratings, belonging to 40 different airlines and more importantly operating on different routes across the globe. The engines have been exposed to take-offs and landings in different environmental conditions, altitudes, aircraft loads and runway lenghts, however sharing the same part design. The part condition at the exposure time counted in flight cycles have been recorded. Similarly to authors of [1] a 10-grade scale have been selected to assign meaningful health index, a condtion rank, to the parts based on their actual condition as shown in Table 1. The condition ranks are established based on the inspection limits provided by the engine manufacturer and supported by conclusions from conducting a root cause analysis of this failure mode. In this specific problem, the inspection limits placed in the engine maintenance documentiation have been not sufficient to capture the early progression of the wear and a scale based purely on inspection findings would be highly non-linear. Between the point at which the part exhibits no wear and the point at which first inspection limits for reccuring inspection apply there exist a relatively long period of preceeding damage accumulation that gives away certain symptoms. Upon completed root cause analyses, metallurgical surveys of the components at different damage stages, expert knowledge and numerical simulations the ranks 1-6 have been introduced which improves proportionality of the used scale and makes it more linear. During this procedure limits have been established that enable to assign the rank to inspected hardware. Although, the maintenance documentation enables safe and profitable engine operation, it had to be expanded to be create a proportional scale that can be used in this research to formulate a regression framework. The inspection data have been revisited to assign proper value of the rank per the extended scale as presented in the Table 1. Introduction of new limits that would cause maintenance actions should be carefuly considered as more operation stoppages would be created, driving the aircraft maintenance cost up and are potentially unnecessary. At this stage, authors of this research are trying to study if a model build on such data can deliver results that could be a starting point to reduce the airline maintenance burden by making the findigs at inspection predictable. Nevertheless, as Figure 1 summarizes, the majority of engines labeled are cases requiring replacement and there is a potential class imbalance for a pure classification oriented problem.

As the engine hardware inspection to establish its condition is a recurrent process that needs to be accomodated into the airline maintenance schedule, it puts a time pressure burden with a potential consequence of unplanned delays and it would be beneficial in that regard to obtain a model that could rank the engines prior to obtaining inspection data.

From the perspective of the fleet management such prognostics would enable to plan ahead of time for the replacement hardware delivery and point out to the engines in the fleet needing it first. These are the challenges that authors of this article are trying to adress.

4. Approach

4.1. Dataset creation

Engines are equipped with a number of sensors collecting flight data. Each engine module from front to aft monitors essential operation paramters; pressure, temperature, variable vanes position setting, shafts rotational speeds and fuel flow injected just to name a few. On the top of that, there exist thermodynamics models deployed, validated through testing campaigns, that utilize these readings and

Table 1. 10-grade scale used to assign the health index to the part condition

Rank	Condition	Service limits applicable	Maintenance action
9	Not accetable for further operation	Exceeded	Engine removal & part replacement
8	Conditionally acceptable for a short duration	Allow for operation for short interval	Increased recurrent inspection frequency on wing
7	Conditionally acceptable for a long duration	Allow for operation for long interval	Recurrent inspection on wing
6	Wear progression – subsequent expansion of the affected area on the component	Observed condition is permitted or no specific limits applicable	Monitoring of the progression on scheduled overhauls when part is exposed
5			
4			
3			
2			
1	Visible wear initiation		
0	No wear confirmed visually		No action – no wear

deliver predictions of other useful parameters that are not acquired directly. Additionally, environmental data for arrivals and departure airports are collected with information about ambient temperature, pressure, elevation above sealevel and air aerosols and added to the database. A *Python* programming language with *Keras* [17], *Tensorflow* [34], *Sci-kit learn* [28] and *pandas* [24] libraries are used for data handling and modelling.

Overall the parameters relevant to the engines for which condition-based ranks were established are retrieved from the database and arranged in such a way that every rank at given inspection is preceded by a number of timesteps and the parameters set for each timestep. The strategy to create the dataset is depicted in the Figure 2.

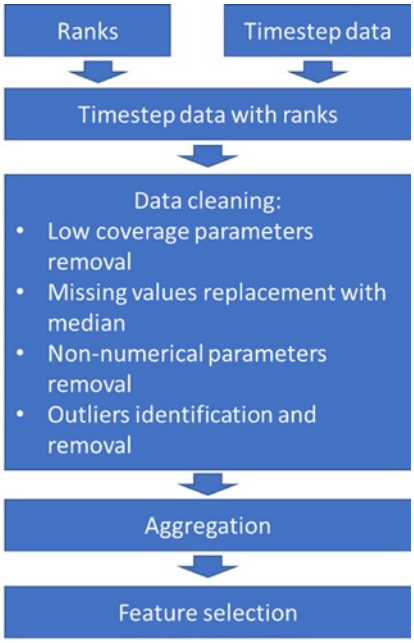

Fig. 2. Dataset creation strategy

In the raw data cleaning process, the parameters having non-numerical values and those not having sufficient coverage over engine operation period are removed. The threshold for lack of coverage is set to be less than 5% of data missing.

Remaining parameters are screened for outlying values, those identified typically come from erroneous sensor readings or faulty data processing and get removed from the set. In an iterative process, all datapoints with standardized score of that parameter, exceeding $\pm 6\sigma$ values are highly suspiscious of being outlying values. Having found such values an investigation has been opened to learn if a sensor malfunctioned, data have been lost or distorted in the migration process or some unexpected event have, in fact, occurred. Upon concluding the investigation, the values were either replaced or removed from the dataset.

The engine's parameters missing values are located and are handled by finding the median value for that particular parameter for the considered engine, then they are filled by that median value. An important consideration is that due to specifics of the aircraft's engine system, each value of parameter should be considered in the missing data management, firstly looking at the data from that engine over time and secondly, if data are too scarce, from the perspective of the sister engine. This minimizes introduction of additional error due to the unknown operational differences.

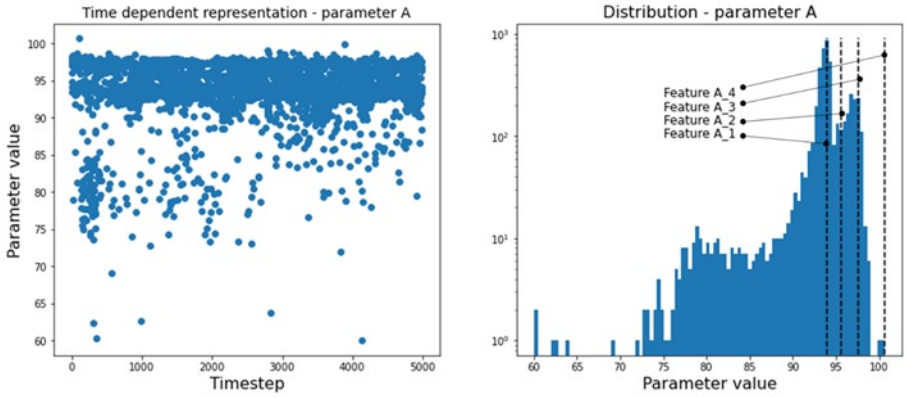

Fig. 3. Feature creation on the example of a single input parameter

4.2. Aggregation and feature selection

To shape the dataset into a problem that can be tackled by machine learning methods the time series data from the sensors are represented by their time independent distributions with the idea depicted in the Figure 3. The values defining the distributions; median, max, 75th percentile value and 95th value are chosen as the new features for the modelling. The selected distribution characteristics come from experimentation with the dataset.

The environmental aerosols data are instead represented by the sum of its departure and arrival values per the flight and accumulated over the total number of flights that engine has completed.

As the engine is a thermodynamic system, a high degree of colinearity is expected between some of its parameters collected during its operation. To adress this issue, a collinerality check is performed within the groups of parameters as shown in the Figure 4. Redundant parameters are identified in this manner that are excluded later from feature creation process.

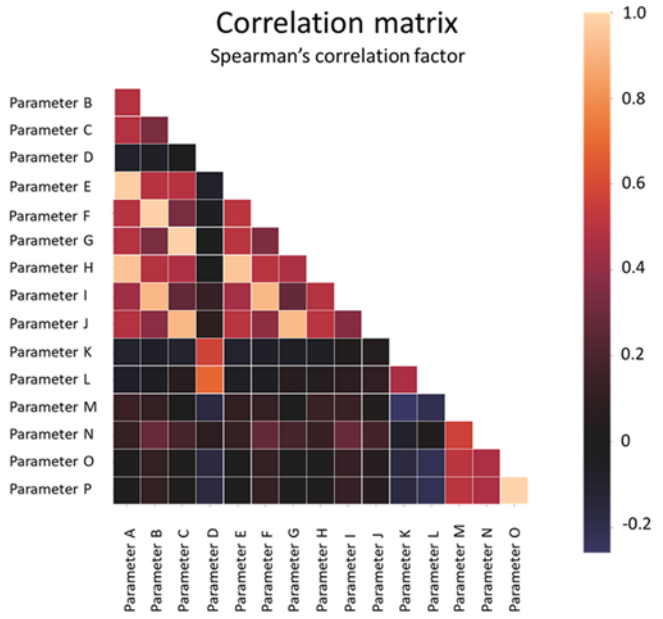

Fig. 4. Correlation matrix

As a final step of feature selection the dataset composed of over 500 features obtained by cleaning and aggregation undergoes a process in which statistically insignificant features are omitted. For that purpose the Boruta algorithm is employed [21]. This procedure limits the number of features to 62 which are later used for developing the best performing model.

4.3. Data transformations

Upon completion of data cleaning and aggregation, the x set is in a form of dataframe of the 62 features by the number of the rows representing the number of the engines and the y are the engine ranks. For the sake of simplicity and having in mind limited number of engines the problem is transformed into a regression problem, where rank is a continuous value from 0 to 9. Additionally, continuous rank is expected to better align with business expectations towards the continuity of the damage progression.

As a next step, the dataset is randomly split into train and validation dataset. The validation dataset is treated as a hold out set and is used eventually to score the models performance against each other. Then, the features are standardized and transformed with *Python scikit-learn* package *StandardScaler* and *PowerTransformer* functions, with the care taken to fitting the functions on the train set, tranforming it and then transforming the validation set, while repeating the procedure feature by feature. The scaling performed by the function follows the equation (1), where x is the value to be scaled:

$$z = (x - \mu)/\sigma \tag{1}$$

μ being a mean value, s is a standard deviation and z is the scaled value.

Additionally, the power transform utilizes the Yeo-Johnson family of equations without the restriction to the values of the variable to be transformed as shown in the equation (2). The input data distribution vary and a transformation to make the distributions more normal is performed. Due to negative values of certain parameters, a simple Box-Cox transformation limited to non-negative values is not feasible. Thus, in the Yeo-Johnson, the λ parameter, representing the transformation parameter, is determined individually for each input feature. In the equation (2), the formulas for λ values at 0 and 2 ensure continuity of the transformation function $\psi(\lambda, y)$ for the entire range y values. The equations for $y \geq 0$ are in fact an equivalent of Box-Cox generalized transformations, whereas the formulas for $y < 0$ enable transformation of negative y values [39]:

$$\psi(\lambda, y) = \begin{cases} \dfrac{\left((y+1)^{\lambda} - 1\right)}{\lambda} & \text{if } \lambda \neq 0,\ y \geq 0 \\ \log(y+1) & \text{if } \lambda = 0,\ y \geq 0 \\ -\dfrac{(-y+1)^{2-\lambda} - 1}{(2-\lambda)} & \text{if } \lambda \neq 2,\ y < 0 \\ -\log(-y+1) & \text{if } \lambda = 2,\ y < 0 \end{cases} \tag{2}$$

4.4. Validation strategy

With the dataset split into train and validation sets, having completed the data cleaning and transformations, a validation strategy for model training, optimization and selection is required.

Hence the train dataset is further used to develop the model, that is to tweak the model and find the best performing hyperparameters on the set. The train set is then often further split into train and test, both complementary subsets of the train set, depending on the need of the specific model. A 7 fold cross-validation (CV) process is used as graphicaly depicted in the Figure 4.1.

As the data become randomly split into k subsets, repeating training over the folds occurs. The model is trained on CV train subset for given set of hyperparameters and scored on CV test subset. In the effect, an average test score from k folds is obtained as shown in the formula (3):

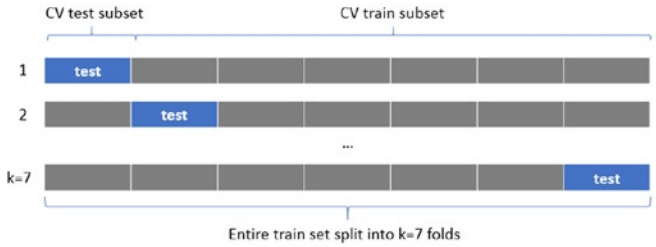

Fig. 4.1. 7 fold cross-validation procedure used in the test

$$CV\ Test\ score = \sum_{i=1}^{k} \frac{test\ score_i}{k} \tag{3}$$

This strategy enables to select the model that performs the best on the train set and has the best average performance while being exposed to the variation present in the train set due to the shuffles made by CV.

The aforementioned validation set is intended to be a hold out set and not used in the model tweaks so that a data leak is avoided and a fair and compenent comparison between the different model possible and to select the one performing best over the specific data. Thus all the comparison scores in this paper are calculated over the validation set via means of multiple further splits into train and test sets with each of the 7 folds of cross-valdation (CV) procedure.

4.4. Hyperparameter optimization strategy

The hyperparameters search is conducted by the means of the Bayesian optimization (BO) [32] where the parameters resulting in the maximum average test score from CV are found. In the Bayesian optimization the objective function $f(x)$ over a dataset is optimized using the benefits of the Bayes' Theorem.

This allows the selection of the most plausible objective function given the prior assumptions regarding the function and hence improve on the performance of the optimization procedure in terms of computational times [7]. In other words, simplifying and applying to the problem at hand, posterior probability of a model M given the evidence (data) E is proportional to the likelihood of E given M multiplied by the prior probability of M (4):

$$P(M \mid E) \propto P(E \mid M) P(M) \tag{4}$$

Instead of *Python scikit-learn* and its *RandomGridSearch* providing the grid search through the hyperparameters,the *bayesopt* package is employed and its implementation of bayesian optimization argument used for every model parameters selection.

4.5. Cost function

As a evaluation score a *mean squared error* (MSE) is calculated as in the equation (5), its used for parameters search in BO and as a mean to compare in between the models. What is more, for the benefit of interpretation ease a R^2 score is calculated however is not used in computations apart from the models comparison:

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{5}$$

In the equations (5) and (6) y_i is the ground truth value, also called a target and $\hat{y}_i$ a model prediction:

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \mu)^2} \tag{6}$$

5. Models overview

This section describes the models that have been considered for this dataset.

5.1. Linear regression

For the sake of establishing a baseline model for the rank prediction capabilities a linear model is used. The *Ridge* model is used from *Python* package as it incorporates a L2-regularization, called Ridge regression, that helps the model to avoid the overfitting. With the considerate number of features compared to the number of datapoints, the ridge regularization introduces a penalty to the minimization objective by adding the magnitude of sum of square of regression coefficients multiplied by α factor as in the formula (7) where objective is the error to be minimized by the objective function optimization:

$$Loss\ function = \sum_{i=1}^{n}\left(y_i - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \alpha \sum_{j=1}^{p} {\beta_j}^2 \tag{7}$$

5.2. Random Forest and Extremely Randomized Trees

A regressor based on the ensemble of tree predictors is selected for evaluation in the presented problem. The tree predictors are grown over randomly selected inputs and their combinations, offer robustness to outliers and data noise while being fast and additionally due to Law of Large Number they are less prone to overfitting. A random subset of candidate features from the set is used to look for discrimative thresholds via splitting into internal nodes and leafs (external nodes). As the subset is random, the tree shape and the thresholds determining the split cause difference between the estimators which predictions are then averaged out. This becomes a strength of the model as some prediction errors can cancel out. The idea is represented in equation (8):

$$\hat{f}_{rf}(\boldsymbol{x}) = \frac{1}{B}\sum_{b=1}^{B} T_b(\boldsymbol{x}) \tag{8}$$

where *B* is the number of predictors, T is a tree.

Each of *n_estimators* trees is grown using *max_features* that is used by the algorithm and with tree depth controlled by *max_depth*. Additionally, minimum samples at internal nodes are controlled by *min_sample_split* and at leafs by *min_samples_leaf.*

The *ExtraTreesRegressor* are a variation of the random forest approach that introduces additional randomness as the thresholds at each node are drawn at random and best of them are then used as a splitting rule. Apart from that similar parameters to random forest are defined.

5.3. Support Vector Machines

A non-linear support vector machines regressor with radial basis function kernel is also considered. The support vector machines can be effective in the case where number of features is large compared to the number of samples with the limitation of being memory consuming. From a high-level standpoint and to describe it, a linear example is used. Let the $g(x)$ be a predictor function. If the data is organized in the manner represented in (9) where **x** are input variables vector and the y is the target:

$$(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_l, y_l) \tag{9}$$

the linear predictor function is shown as in (10).

$$g(x) = \boldsymbol{w}, \boldsymbol{x} + b_0 \tag{10}$$

where $\boldsymbol{w}, \boldsymbol{x}$ is dot product of the model weights and the input variables vectors and b_0 an intercept. Transforming this into an optimization problem it takes a form of:

$$minimize \frac{1}{2}\boldsymbol{w}, \boldsymbol{w} \tag{11.1}$$

$$\text{subject to } \begin{cases} y_j - \boldsymbol{w}, \boldsymbol{x_j} - b_0 \le \varepsilon \\ \boldsymbol{w}, \boldsymbol{x_j} + b_0 \le \varepsilon \end{cases} \tag{11.2}$$

In the equations (11) ε represents an error, meaning the weights vector $\boldsymbol{w}$ that results in the solutions lower than error are found. As stated in [35] it is often desirable to have some errors greater than ε and hence the formula is rewritten with introduction of slack variables δ_j and δ_j^* taking the form of these equations (12):

$$minimize \frac{1}{2}\boldsymbol{w}, \boldsymbol{w} + C\sum_{j=1}^{l}\left(\delta_j + \delta_j^*\right) \tag{12.1}$$

$$\text{subject to } \begin{cases} y_j - \boldsymbol{w}, \boldsymbol{x_j} - b_0 \le \varepsilon + \delta_j \\ \boldsymbol{w}, \boldsymbol{x_j} + b_0 - y_j \le \varepsilon + \delta_j^* \\ \delta_j, \delta_j^* \ge 0 \end{cases} \tag{12.2}$$

Upon optimization the first term of the equation is solved just like in (11.1) ensuring weights take low values whereas the second term $C\sum_{j=1}^{l}\left(\delta_j + \delta_j^*\right)$, where *l* represents the number of observations in the dataset, is known as regularization term and ensures that the optimization problem is feasible. Thus, parameter *C* offers a trade-off between the model complexity and the error values. Both parameters ε and *C* are hyperparameters subject to optimization.

5.4. XGBoost

A tree gradient boosting regression model is also researched for feasibility of use on the dataset at hand. This machine learning model has gained popularity due to its performance, speed and scalability. Authors of [9] deliver a very clear description of the algorithm.

The general idea of the model is represented by formula (13), in a dataset $(\boldsymbol{x}_n, y_n)$ composed of *n* observations and *m* features in a input vector $\boldsymbol{x}_n$ a tree ensemble model uses *K* additive functions to predict the target $\hat{y}_i$:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(\boldsymbol{x}_i) \tag{13}$$

Each tree objective function as shown in (14) contains a loss function term which measures the difference between the prediction y_i and the target $\hat{y}_i$ and added regularization term Ω that penalizes the model complexity:

$$Objective = \sum_i loss(y_i, \hat{y}_i) + \sum_k \Omega(f_k) \tag{14}$$

Table 2. Models' hyperparameters overview

	Python package	Model name	Hyperparameters optimized
Linear model	Sklearn.linear_model	*Ridge*	alpha
Random Forest	Sklearn.ensemble	*Random Forrest Regressor*	• max_features • max_depth • min_sample_split • min_samples_leaf • n_estimators
Extremely Randomized Trees	Sklearn.ensemble	*Extra Trees Regressor*	same as in Random Forest
SupportVector Machines	Sklearn.svm	*SVR*	epsilon, C
XGboost	Xgboost	*XGB Regressor*	• max_depth • learning rate • colsample_bylevel • subsample • n_estimators
ANN MLP	Keras/Tensorflow	*Multilayer Perceptron*	• n_layers • n_units per layer • dropout rate • learning rate • test set size • regularization
MLP ensemble	Keras/Tensorflow	*MLP ensemble*	same as in ANN MLP

Loss if a differentiable function that measures the difference between the prediction and the target.

Parameters selected for hyperparameter optimization are as in Table 2.

5.5. Neural networks – multilayer perceptrons (MLP)

Deep neural network is selected as the last type of the model. A multiple hidden layer network, where the input layer takes inputs from the dataset features and then feeds it forwards to a single output neuron predicting the target is built in *Python tensorflow* using *keras* framework.

Let the number of neurons in the layer be *m, n* the number of samples and *k* represent the index of the layer. On the very basic level, in the fully connected each neuron in the hidden layer obtains signals vector $\boldsymbol{x}_k$ of *m* values that represent the input, it gets adjusted by weights assigned to every connection $\boldsymbol{w}_k$ and a bias b_k and then is summed as in equation (15) to create a single output value of the layer v_k. Then an activation function φ is applied on the v_k to obtain the layer output y_k:

$$v_k = \boldsymbol{w}_k \cdot \boldsymbol{x}_k + b_{\boldsymbol{k}} \tag{15}$$

$$\mathrm{v_k} = \begin{bmatrix} x_1 & x_2 & \cdots & x_m \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{bmatrix} + b_k \tag{16}$$

$$y_k = \varphi\left(v_k\right) \tag{17}$$

Then the output becomes input for the next layer neurons and the process repeats until eventually output of the model for a single sample is obtained $\hat{y}_k$. Eventually, the error in the prediction is calculated via loss function by comparison of y_k to the target. Upon the error calculation the back propagation occurs and the error is back propagated via implemented algorithm to adjust all the network weights based on their contribution to the output error.

In the training process the samples are propagated multiple times until the weights are adjusted so that the loss is minimized. The input data is organized in samples and then into smaller batches, which are passed through the model multiple times. In one *epoch* the model has been exposed to all samples in the training set and during one *iteration* the model has adjusted weights to minimize error one batch. In the approach of this research the batch size is set to 1, meaning a model trains on a single randomly selected sample to adjust the weights.

Dropout layers are employed to help prevent the model overfitting, the dropout value is the percentage of neurons in the layer that are randomly excluded from weight adjustment process and do not partake in the output calculation, it is known to contribute to the model robustness. The dropout undergoes hyperparameter optimization. Moreover, a L2 regularization (Ridge) in the first dense layer is turned on, contributing to the objective function with its α value also determined via the optimization process.

As the problem is presented as a regression an activation function is selected to be Parametric Rectified Linear Unit (PReLU). In one of the landmark papers, Kaiming He and others recognized the downfalls of the typically used ReLUok activation function and proposed the alternative which is improvement over Leaky ReLU and demonstrating improvement in image classification error neural network [13]. Thus, the used activation function is as in formula (18):

$$\varphi\left(v_i\right) = \begin{cases} v_i, & if\ v_i > 0 \\ \alpha_i v_i, & otherwise \end{cases} \tag{18}$$

It is worth noting that the PReLU behaves like ReLU for positive values of input and the return certain parametric linear output for negative values.

As explained in the chapter 4.3 the train set is used for the model training and optimization leaving the validation set acting as a holdout set. The train set is split in advance into the train and test subsets at random using *StratifiedShuffleSplit* function.

The process repeats *k* times as the folds of cross-validation enforce model to train and test on a different batch while test set size is maintained. To achieve the perfect balance for this particular dataset, the train to test split ratio is kept as one of the hyperparameters.

Lastly, learning rate is selected as a hyperparameter, meaning the rate at which the weights are adjusted. Importance of this parameter is undoubted as too low values cause inefficient training and too high may cause the model not to converge at all.

Model training, being in the essence finding such model weights, biases and activations, also called parameters that yield the least error, is possible thanks to a gradient descent algorithm [27]. Let $J(\theta)$ be an objective function to be minimized and $\theta \in R$ be the model parameters, by performing the gradient descent, that is updating the parameters in the opposite direction of the gradient of the objective function $\nabla_\theta J\left(\theta\right)$ thus following the slope of towards a local minimum. A learning rate η, selected as model hyperparameter in this study, determines the size of the step towards the expected minimum. A popular implementation of this idea, shown in (19), is a stochastic gradient descent (SGD), which enables to calculate the objective function on one sample, instead of all in the batch, that significantly expedites the walk towards the minimum:

$$\theta_t = \theta_{t-1} - \eta \cdot \nabla_\theta J(\theta) \tag{19}$$

Too high value can make the optimization process unstable and prevent the model to converge, too low value can make training process ineffective. There exist numerous optimizers that attempt to improve on it, introducing concepts of momentum to pass over local minima and preventing overshoot due to the overpowering momentum (Nesterov Accelerated Gradient). To better deal with data sparsity an adaptive learning rate algorithm was introduced, Adagrad, that preferentially adjusts learning rates for each parameter and to counteract its downfalls manifesting as monotonically decreasing learning rate Adadelta was proposed. Neural networks trained in the research utilized Adaptive Moment Estimation [20], *ADAM*, that computes adaptive learning rates for each parameter like aforementioned adaptive algorithms but proposing features similar to the concept of momentum. Let the $g = \nabla_\theta J(\theta_t)$ be the gradient and ε be a small term preventing division by zero in the formula (20):

$$\theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{\hat{v}_t} + \varepsilon} \hat{m}_t \tag{20}$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \tag{21}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \tag{22}$$

$$m_t = (1 - \beta_1) \cdot g_t + \beta_1 m_{t-1} \tag{23}$$

$$v_t = (1 - \beta_2) \cdot g_t^2 + \beta_2 v_{t-1} \tag{24}$$

where the m_t is a first momentum (23) and the v_t is the second momentum (24) and the β_1, β_2 are decay terms.

5.6. Ensemble

Models collected in an ensemble composed of few best scored neural networks have been explored. In the process of hyperparameter optimization of neural networks, three models with various scores have been obtained. Similar to the concept of the random forest, an ensemble of neural nets can offer an improvement in the overall score as some of the individual model errors can potentially cancel out.

In the study preceding this paper, an ensemble has been created through training meta-model of a similar architecture as single neural network. The meta-model undergoes exactly the same procedure of cross-validated Bayesian hyperparameter optimization with the exception of using the stacked output of the single models as its input and in the prediction is scored with the means of the loss function.

6. Results

Presented results represent the models that have been subjected to hyperparameter optimization described in previous chapters. Both scores R^2 and *MSE* are shown for ease of interpretation, however the *MSE* is selectedfor this regression problem and is used draw conclusions.

The mean squared error score penalizes large errors; as a prediction differs from the true value, the penalty score exhibits quadratic growth. Thus, if used as a loss fucntion in an optimization problem, penalizing large error helps to find model paratmeters that result in minimizing them.

The validation score is calculated over the validation holdout set and the train score represents how model fitted the train set.

As shown in the Figure 5, the best performing model for the specified problem and the data available, has been a neural network meta-model ensemble, achieving *MSE* score of 0.71, that brought 17.4% error decrease from a single best neural network model with a scored at 0.86.

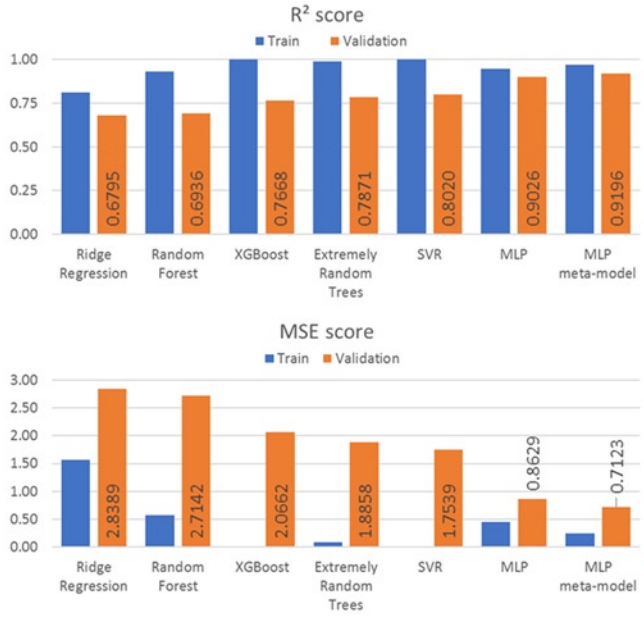

Fig. 5. Results comparison – models' scores. Train and validation series represent model performance on the train and validation sets respectively

The support vector machine regressor model obtained 1.76, that outperformed extremely randomized trees models with a score of 1.88 by a 6.4%. Griadient boosted tree regressor obtained a score of 2.07, random forest model scored 2.71 and ridge regression 2.84.

The difference in error between the score of simple linear model (ridge regression) to the neural net ensemble corresponds to 75% of the linear model score, which justifies the effort invested into deep learning models exploration.

As shown in the Figure 10 even for the best model, there exist outlying residual value in the validation set, which model does not predict well (model underpredicts a 5 distress rank to be little over 3) and increases *MSE* score. Futhermore, a *RMSE* score is also calculated to conclude about the model applicability to the problem at hand.

In addition to the overall models' performance, it has been observed that all researched models have obtained inconsistent score over the ranks as depicted in *RMSE* score plot in Figure 6. Due to scarcity of rank 3 data points, they have not been selected for the validation set via a random selection *train_test_split scikit-learn* function. Hence the error values for rank 3 are not available and models ability to predict rank in this range remains not quantified explicitly.

The highest *RMSE* have been produced by SVR (4.01) and linear model (3.66) for rank 1. The lowest *RMSE* values have been achieved by SVR (0.07) and MLP ensemble model (0.10) while predicting rank 0. As demonstrated in the *RMSE* distribution plotted in Figure 6, the most common value is between 1.0 and 1.5.

All studied models have obtained the lowest error while making predictions for rank 7 with similar values scored as quantified by a standard deviation of 0.19 of *RMSE*. Conversely, the greatest inconsistency have been noted for rank 1; MLP-based models scored low error, yet other models have been producing a high error, which contributed to a standard deviation of 1.10 of *RMSE* for this rank.

The ensemble and single neural network models have a better performance for target variable in range from 0-4 (*RMSE* in range from 0.10 to 1.10) and 7-9 (*RMSE* 0.29 to 0.78), than in predicting ranks 5-6 (*RMSE* 1.47 to 2.92). Errors achieved by the MLP based models

in this range are the greatest among the considered models followed by XGBoost that have obtained 2.14 *RMSE* over rank 5 and SVR with 1.61 *RMSE* over rank 6.

As a general trend and omitting the exceptionally low errors described earlier, the machine learning models have had higher *RMSE* values for ranks 0-4 (1.01 to 4.01), then error decreases for ranks 5-6 (0.07 to 1.61), becomes the low for all for rank 7 (0.56 to 0.94) and then slightly increases for ranks 8-9 (0.90 - 1.88). This error general trend is different than for earlier discussed MLP-based models.

Some exceptions to this trend have occurred; XGBoost demonstrated greater *RMSE* value for ranks 4-5 (2.14 – 2.42) than for ranks 1-2 (0.94 – 1.62), whereas other machine learning models *RMSE* were in a range of 0.07 – 1.61.

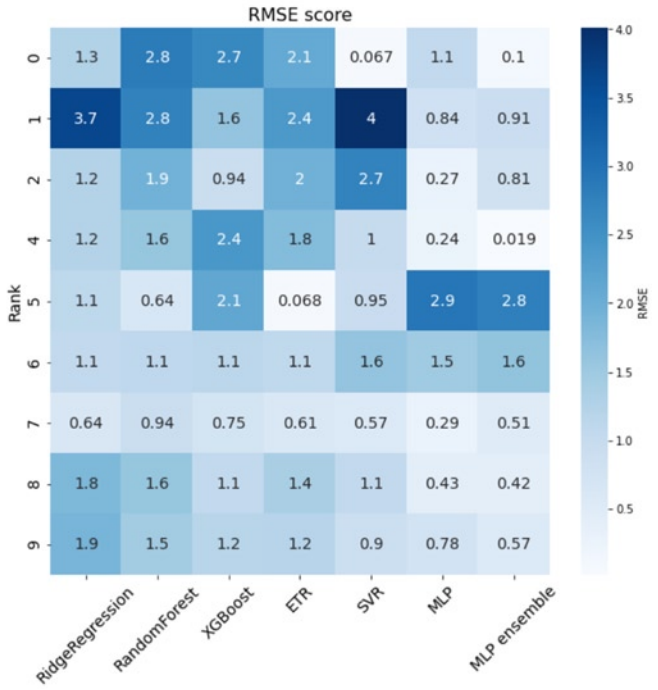

Fig. 6. RMSE score per rank (lower value = less error)

In the tree ensemble based models group; random forest and extremely randomized trees, the latter have, in general, predicted with lower *RMSE* values and offered an improvement in minimum and maximum values. The minimum and maximum values have improved from 0.64 and 2.85 to 0.07 and 2.38, respectively.

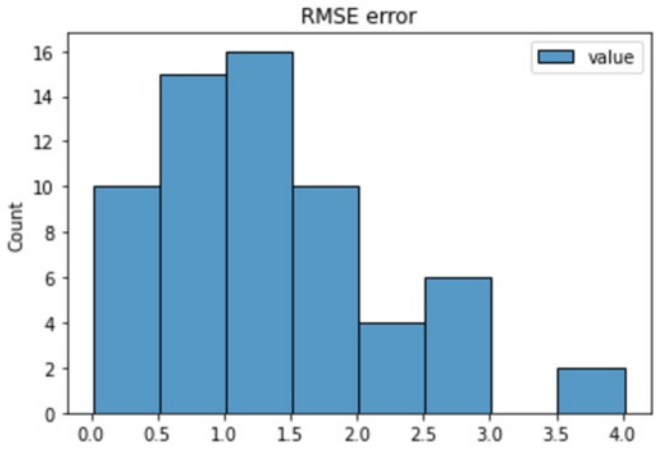

Fig. 7. Distribution of RMSE errors calculated per rank for every model using validation set

Gradient boosted trees model, XGBoost, has surpassed the ETR and random forest models by achieving lower *RMSE* value for ranks 1-2 (XGBoost: 0.94 – 1.62, tree ensemble models: 1.94 – 2.75), however predicted with greater error for ranks 4-5 (XGBoost: 2.14 – 2.42, tree ensemble models: 0.07 – 1.78) and offered some improvement for ranks 8-9 (XGBoost: 1.05 – 1.20, tree ensemble models: 1.21 – 1.59).

SVR *RMSE* values have been low for rank 0 (0.07) and rank 9 (0.9) and comparable to those of MLP ensemble model errors (rank 0: 0.1, rank 9: 0.57). Unfortunately, its prediction error inconsistency through other ranks have been relatively high (*RMSE* 0.6 – 4.01).

Based on the plot in Figure 6 the MLP-based models can make a prediction of low and high ranks with the least error.

The described trends do not correlate with the distribution of the ranks in the training set, training set distribution is similar to that of the entire dataset shown in Figure 7. The data points with rank 9 are most frequent, ranks 8 and 7 occur more rarely and the other ranks data is rather limited. Either of the earlier described trends can be explicitly explained by the distribution of the target variable in the training set.

As the MLP ensemble model predicts with the least error, it is selected as a reference point and the differences in *RMSE* of the others models to the ensemble are calculated and summarized in the plot in Figure 8. The negative difference values, coloured by the shades of red are cases where models have performance debit to the MLP ensemble and conversely, positive values and shades of green show where other models predicted with lower error.

The single MLP model have had a *RMSE* greatest differences for rank 0 (-0.98) and rank 4 (-0.22). MLP ensemble greatly improved error in predicting rank 0. Otherwise, the differences in majority of ranks are between -0.22 and 0.22 values and can be considered similar. An exception to this observation is a rank 2 where single MLP predicted with lower error and the differences was 0.45. Although, there have been ranks where single MLP outperformed the meta-model, the opposite situation has been as frequent and due to the lower overall prediction error, the ensemble model has shown a better performance.

The ensemble meta-model has brought improvement in prediction error it is lower and upper ranges of the target variable. The other models have had, in general, up to -3.10 difference for ranks 0-4 and up to -1.40 difference for ranks 8-9. The models have been within -0.42 to 0.22 in difference to the ensemble for rank 7, with SVR having the least difference (-0.05) and random forest having the greatest

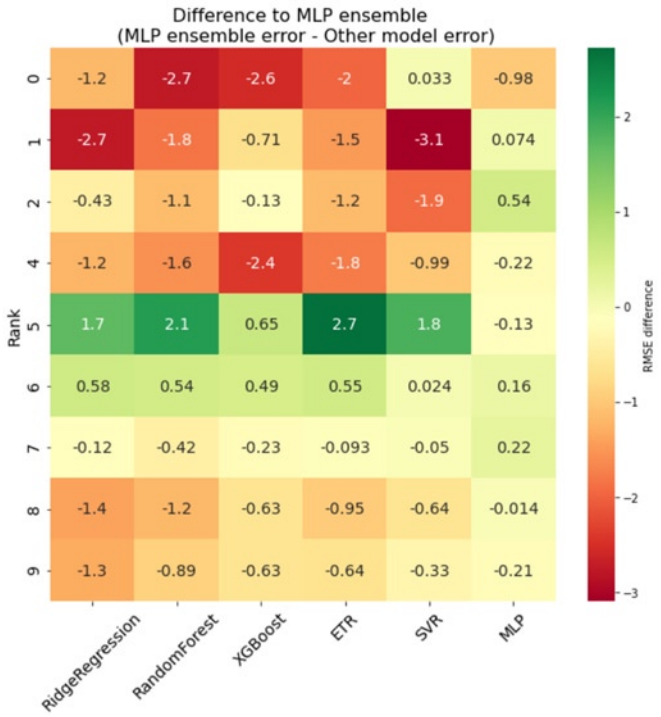

Fig. 8. Difference in error with respect to MLP ensemble model

difference (-0.42). As can be observed, these models outperformed the ensemble in predicting ranks 5-6 with difference up to 2.72 (ETR).

Residual values calculated as a difference between the true and predicted values have been calculated for each model over the train and validation sets and demonstrated for selected models in Figure 9. Non-linear models representing different algorithms families have been chosen: ETR, SVR, XGBoost and MLP.

SVR and XGBoost models have overfitted to the train set, as all prediction values line up closely with their corresponding true values with little residual error, while the validation set residuals are significantly greater. In this particular application, MLP and ETR seem to be less prone to this behaviour and greater train set residuals are visible.

Studied models have also been predicting different outlying values, however due to the noise in the residual values have been hard to interpret. The following observations regarding outlying values have been noted:

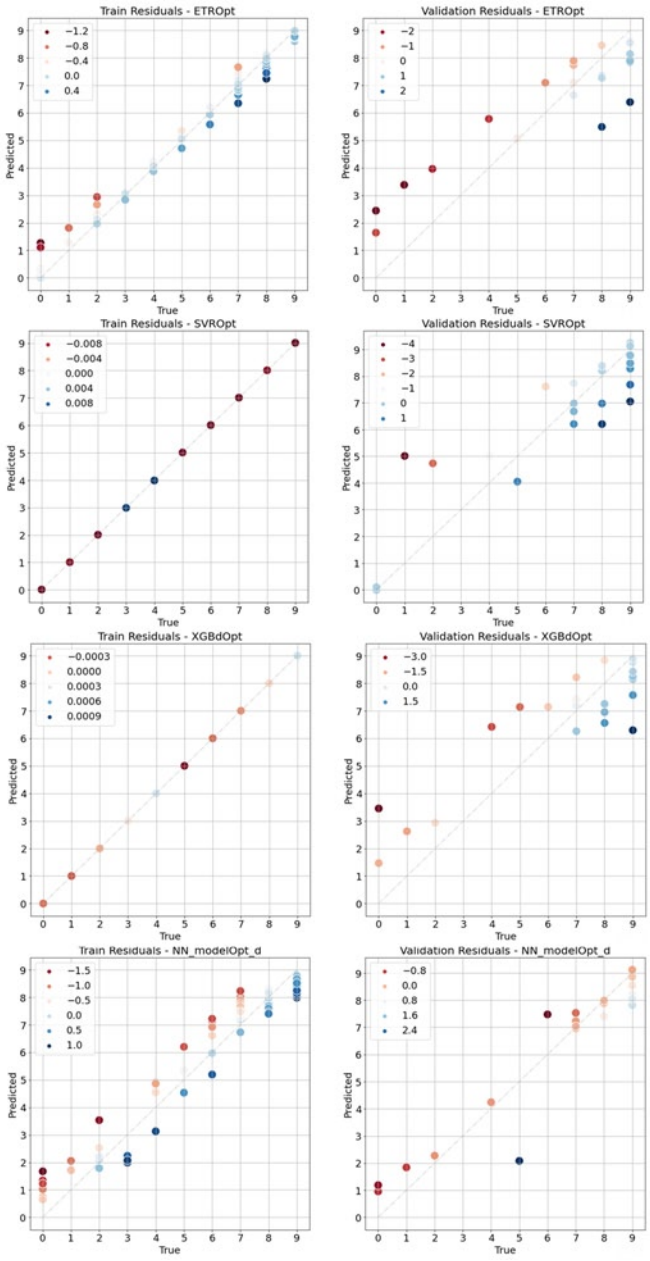

Fig. 9. Residuals plots for selected models

- ETR predictions have the most consistent absolute residual values in the group considered and there are no clear outlying values in the prediction.
- SVR model predicted two outlying values (overpredicted rank 1 and 2).
- XGBoost model residuals are noisy with perhaps one outlying value (overpredicted rank 0).
- MLP predicted one outlying value (underpredicted rank 5).
- SVR, XGBoost and MLP do not predict the same outlying values.

Furthermore, a tendency in over and underprediction have been analysed; XGBoost tends to overpredict the lower ranks and underpredict higher ranks. Similar, however less pronounced, trend is exhibited by ETR. The bull's eye prediction of SVR for rank 0 seems to be an exception and if treated as an outlier, its prediction residual error trend would become similar.

The MLP model is the least noisy in the considered group and does not show a residual error trend exhibited by the other models. What is more, the meta-model ensemble residuals depicted in Figure 10 are similar to the single MLP in lack of the residuals trend and also predict the same outlying value. This explains why ensemble model shares similar performance for rank 5 and demonstrates the ensemble model have not improved the capability to predict this value.

7. Conclusions

Based on results one can observe that certain models have performed better than the others over the given dataset. The promising results presented in the paper align with the recent conclusions of the research community regarding deep learning models applications.

The specifics of the problem have shown that a simple linear model, although useful to certain degree, can be surpassed in performance by more complex architectures. What is more, the superiority of the ensemble model over single neural net model is further confirmed and found in the referenced literatures researchers insights. Additionaly, the neural nets outperformed tree based models and support vector machines. As illustrated in the results, all models have a tendency to overfit to the train set, despite the counter measures taken, however boosted trees, extremely random trees and support vector machines have gravitated towards overfitting more than the others. It might be noted, that the models that have had the lowest difference between train score and validation score are deep learning models. In the effect, their highest validation scores on this dataset could be attributed to their ability to generlize the best and learn without overfitting to the training set.

The best model residuals demonstrate fairly consistent error in continously predicting conditions ranks across the scale and hence it is concluded that it could be satisfactory used for the problem at hand. Translating the *MSE* 0.71 to *RMSE* returns value of 0.84, which, from the forecast perspective, enables to predict ranks with error lower than one condition rank in the scale. Such perspective places the deep learning models considered in this paper as an adequate candidates for the business use, however leaves a room for improvement for future studies for the research community.

The obtained results demonstrate that a neural network model build on the gathered data can predict the rank with average error less than one unit of the rank scale. Although certain models error has not been consistent over the enitre rank scale, a potential business application could benefit by a prediction by few models, keeping in mind their different performance in different rank scale ranges. As a conclusion it may be underlined, that proper data collection and ranking the collected inspection data is a relatively long processes, that is greatly expedited by using established inspection procedures and their findings.

An important challenge has become a selection of a proper rank scale, which should ensure proportionality to formulate a valid regression framework. In the specific example, the existing data based on the engine service limits had to be expanded by introduciton of ranks that represented early wear stages and would normally be omitted per the existing inspection requirements as being acceptable to operate with. Additional ranks required revisiting the collected inspection data and proper re-assignment based on the established scale. The devel-

opment of the scale required a study of the failure mode, conducting destructive tests, application of material knowledge and involvment of industry experts and wihtout this preceding step further research would not be possible.

In the data collection process, a strong bias towards having the majority of data points composed of worn out parts or parts near the end of its useful life have been observed. This is due to the fact, that in the aviation industry, the airlines tend to maximize the time that aircraft is in operation and stopagges due to the inspections and repairs are additional financial burden. Therefore components near its service limits or requiring recurrent inspections of increased frequency are removed earlier. This data is most widely accesible and shared with the engine manufacturer, which explains the bias in the dataset. On the other hand, due to some unexpected events, i.e. foreign object damage to the engine, the component becomes exposed before the wear process is initiated and the dataset has more data points of this stage than few of the subsequent ranks. The least available data are from the early progression stage of the wear from initiation point to the moment of first service limits apply. This is explained by the fact, that such data is considered acceptable per the inspectors and typically not captured in the inspection process as it presents hardware condition that will continue to operate for a significant time between the wear out. This mindset is a challenge for implementation of a data collection process that enables building a high fidelity prediction model, where a model should be trained with a balanced dataset to predict over the entire range of the target variable with an acceptably low error. With such limitation, ranking scale selection process may become a trade off between having sufficiently many grades to capture the physics and number of data points per each rank for the model to be able to fit to it. As a conclusion from this research, implementation of a data collection scheme expanding the scope of the current inspection data would enable further development of such models. However, it should be noted, that a potential data collection processes to keep the models up to date can be done without the modification of the inspection limits and done post inspection by the engine manufacturer. This approach would help to reduce the maintenance cost by providing a way to monitor fleet's health and manage the maintenance without creating additional opeartion stoppages.

Using the model, a prediction for every turbofan engine condition in the fleet can be obtained easily and updating the prediction regularly with the new input data can provide useful information about the progression of the wear and change in the fleet's health. Information about the rank could enable to schedule maintenance and set expectations regarding the condition once engine is visually inspected on-wing. The information available ahead of time can enable a prioritization of engine repairs and ordering replacement hardware. Presented study demonstrates that use of such data can deliver a valuable solution to the industry with relatively low investment of time and resources using the latest developments in deep and machine learning. In the nearest perspective, models might not be feasible to replace the on-wings inspection, but can reduce an inspection burden by making its outcomes more manageable and predictable. Safety has always been a number one factor in the aviation industry and the most likely application of such models is expected in the fleet health monitoring and maintenance management rather than direct replacement of well established inspection processes.

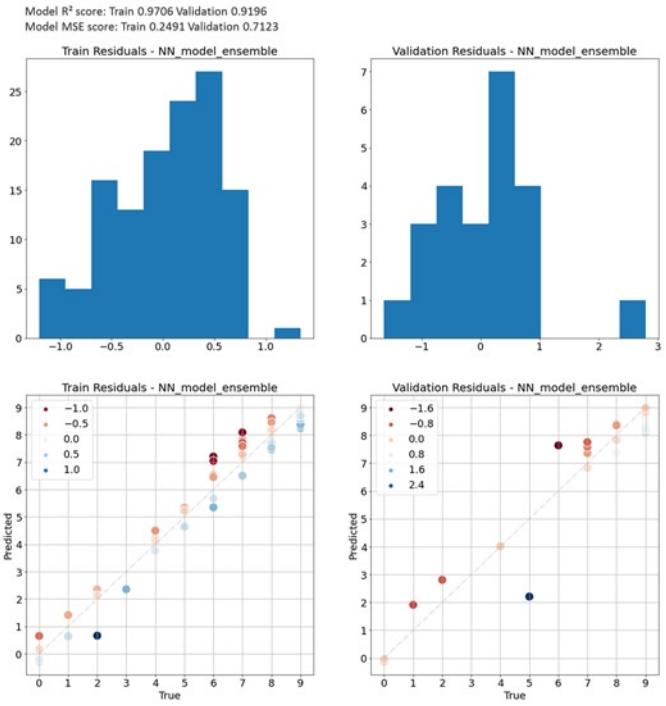

Fig. 10. Ensemble meta-model residuals

8. Next steps

Authors of the article recognize the promising results obtained by the scientific community using recurring neural networks architectures in similarly stated problems, the demonstrated performance of deep Bayesian networks and the advantages of combining the efficiency of semi-supervised learning variational autoencoders with deep Bayesian network models on sparsely labelled data typically encountered in the aviation industry, thus wish to try these methods to further research this particular problem.

References

1. Amir M D M, Muttalib E S A., Health index assessment of aged oil-filled ring main units. IEEE 8th International Power Engineering and Optimization Conference, 24-25 March 2014, https://doi.org/10.1109/PEOCO.2014.6814452.
2. Azipurua J I, Stewart B G, McArthur S D J, Lambert B, Cross J G, Catterson V M., Improved power transformer condition monitoring under uncertainty through soft computing and probabilistic health index., Applied Soft Computing 2019; 85, https://doi.org/10.1016/j.asoc.2019.105530.
3. Babu G S, Zhao P, Li X L., Deep Convolutional Neural Network Based Regression Approach for Estimation of Remaining Useful Life., Database Systems for Advanced Applications. DASFAA 2016. Lecture Notes in Computer Science 9642, https://doi.org/10.1007/978-3-319-32025-0_14.
4. Baojun N, Mei Y, Xingjian S, Peng W. Random FEA and reliability analysis for combustor case., 2017 Prognostics and System Health Management Conference (PHM-Harbin), Harbin 2017: 1-5, https://doi.org/10.1109/PHM.2017.8079268.
5. Bektas O, Jones J A, Sankararaman S, Roychoudhury I, Goebel K., A neural network filtering approach for similarity-based remaining useful life estimation., The International Journal of Advanced Manufacturing Technology 2019; 101: 87-103, https://doi.org/10.1007/s00170-018-2874-0.
6. Bojdo N, Ellis M, Filippone A, Jones M, Pawley A. Particle-Vane Interaction Probability in Gas Turbine Engines., Journal of Turbomachinery 2019, 141(9), https://doi.org/10.1115/1.4043953.
7. Brochu E, Cora V M, de Freitas N. A Tutorial on Bayesian Optimization of Expensive Cost Functions with Application to Active User Modeling and Hierarchical Reinforcement Learning 2010, https://arxiv.org/abs/1012.2599.

8. Cerdeiera J O, Lopes I C, Silva E C., Scheduling the Repairment of Aircrafts' Engines., 2017 International Conference on Control, Artificial Intelligence, Robotics & Optimization (ICCAIRO), Prague, 2017: 259-267, https://doi.org/10.1109/ICCAIRO.2017.56.
9. Chen T, Guestrin C. XGBoost: A Scalable Tree Boosting System. KDD '16: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, August 2016: 785-794, https://doi.org/10.1145/2939672.2939785.
10. Echarda B, Gaytona N, Bignonnet A. A reliability analysis method for fatigue design. International Journal of Fatigue 2014: 59: 292-300, https://doi.org/10.1016/j.ijfatigue.2013.08.004.
11. Gao Z, Li Jiwu, Wang R. Prognostics uncertainty reduction by right-time prediction of remaining useful life based on hidden Markov model and proportional hazard model. Eksploatacja i Niezawodnosc - Maintenance and Reliability 2021; 23(1): 154-164, https://doi.org/10.17531/ein.2021.1.16.
12. Guffanti M, Tupper A. Chapter 4 - Volcanic Ash Hazards and Aviation Risk. Volcanic Hazards Risks and Disasters 2015: 87-108, https://doi.org/10.1016/B978-0-12-396453-3.00004-6.
13. He K, Zhang X, Ren S, Sun J. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. 2015 IEEE International Conference on Computer Vision (ICCV), Santiago 2015: 1026-1034, https://doi.org/10.1109/ICCV.2015.123.
14. Heimes F. Recurrent neural networks for remaining useful life estimation. 2008 International Conference on Prognostics and Health Management 2008: 1-6, https://doi.org/10.1109/PHM.2008.4711422.
15. Illustration, https://res.mdpi.com/information/information-10-00113/article_deploy/html/images/information-10-00113-g003.png (accessed 10 October 2020).
16. Instructions for Continued Airworthiness, Code of Federal Regulations 14 CFR § 33.4 (1980).
17. Keras documentation, https://keras.io/api/ (accessed 10 April 2020).
18. Khan N, Manarvi I A. Identification of delay factors in C-130 aircraft overhaul and finding solutions through data analysis. 2011 Aerospace Conference, Big Sky, MT, 2011: 1-8, https://doi.org/10.1109/AERO.2011.5747591.
19. Khan S, Yairi T. A review on the application of deep learning in system health management. Mechanical Systems and Signal Processing 2018; 107: 241-265, https://doi.org/10.1016/j.ymssp.2017.11.024.
20. Kingma D P, Ba J. ADAM: A Method for Stochastic Optimization. 3rd International Conference for Learning Representations 2015: 1-13, https://arxiv.org/abs/1412.6980.
21. Kursa M B, Rudnicki W R. Feature Selection with the Boruta Package. Journal of Statistical Software 2010; 36(11), http://dx.doi.org/10.18637/jss.v036.i11.
22. Li Y X, Shi J, Gong W, Zhang M. An ensemble model for engineered systems prognostics combining health index synthesis approach and particle filtering. Quality and Reliability Engineering International 2017; 33(8): 2711-2725, https://doi.org/10.1002/qre.2229.
23. Malik K, Zbikowski M, Teodorczyk A. Detonation cell size model based on deep neural network for hydrogen, methane and propane mixtures with air and oxygen. Nuclear Engineering and Technology 2019; 51(2): 424-431, https://doi.org/10.1016/j.net.2018.11.004.
24. Pandas documenation, https://pandas.pydata.org/pandas-docs/stable/ (accessed 20 September 2020).
25. Pawełczyk M, Fulara S, Sepe M, De Luca A, Badora M. Industrial gas turbine operating parameters monitoring and data-driven prediction. Eksploatacja i Niezawodnosc - Maintenance and Reliability 2020; 22(3): 391-399, https://doi.org/10.17531/ein.2020.3.2.
26. Przysowa R, Gawron B, Kulaszka A, Placha-Hetman K. Polish experience from the operation of helicopters under harsh conditions. Journal of Konbin 2018; 48(1): 263-299, https://doi.org/10.2478/jok-2018-0056.
27. Ruder S. An overview of gradient descent optimization algorithms, 2017, https://arxiv.org/abs/1609.04747.
28. Scikit-learn documentation, https://scikit-learn.org/stable/index.html (accessed 20 September 2020).
29. Shin J H, Jun H B. On condition based maintenance policy. Journal of Computational Design and Engineering 2015; 2(2): 119-127, https://doi.org/10.1016/j.jcde.2014.12.006.
30. Sikorska J Z, Hodkiewicz M, Ma L. Prognostic modelling options for remaining useful life estimation by industry. Mechanical Systems and Signal Processing 2011; 25(5): 1803-1836, https://doi.org/10.1016/j.ymssp.2010.11.018.
31. Sina Tayarani-Bathaie S, Sadough Vanini Z N, Khorasani K. Dynamic neural network-based fault diagnosis of gas turbine engines. Neurocomputing 2014; 125: 153-165, https://doi.org/10.1016/j.neucom.2012.06.050.
32. Snoek J, Larochelle H, Adams R P. Practical Bayesian Optimization of Machine Learning Algorithms. Advances in Neural Information Processing Systems 2012; 25, https://arxiv.org/abs/1206.2944.
33. Sun J, Zuo H, Wang W, Pecht M G. Application of a state space modeling technique to system prognostics based on a health index for condition based maintenance. Mechanical Systems and Signal Processing 2012; 28: 585-596, https://doi.org/10.1016/j.ymssp.2011.09.029.
34. Tensorflow documentation, https://www.tensorflow.org/ (accessed 9 May 2020).
35. Wu S, Akbarov A. Support Vector Regression for Warranty Claim Forecasting. European Journal of Operational Research 2011; 213(1): 196-204, https://doi.org/10.1016/j.ejor.2011.03.009.
36. Xu J, Liu X, Wang B, Lin J. Deep Belief Network-Based Gas Path Fault Diagnosis for Turbofan Engines. IEEE Access 2017; 7: 170333-170342, https://doi.org/10.1109/ACCESS.2019.2953048.
37. Yang Y, Ding Y,and Zhao Z. Fault distribution analysis of airborne equipment based on probability plot. 3rd IEEE International Conference on Control Science and Systems Engineering 2017: 239-242, https://doi.org/10.1109/CCSSE.2017.8087931.
38. Yang Y, Guo F. Reliability Analysis of Aero-Equipment Components Life Based on Normal Distribution Model. IEEE 4th Information Technology and Mechatronics Engineering Conference (ITOEC) 2018, Chongqing, China, 2018: 1070-1074, https://doi.org/10.1109/ITOEC.2018.8740448.
39. Yeo I K, Johnson R A. A new family of power transformations to improve normality or symmetry. Biometrika 2000; 87(4): 954-959, https://doi.org/10.1093/biomet/87.4.954.
40. Yoon A S, Lee T, Lim Y, Jung D, Kang P, Kim D, Park K, Choi Y. Semi-supervised Learning with Deep Generative Models for Asset Failure Prediction. KDD17 Workshop on Machine Learning for Prognostics and Health Management 2017, Canada, https://arxiv.org/abs/1709.00845.
41. Zaidan M A, Harrison R F, Mills A R, Fleming P J. Bayesian hierarchical models for aerospace gas turbine engine prognostics. Expert Systems with Applications 2015; 42(1): 539-553, https://doi.org/10.1016/j.eswa.2014.08.007.