Imperial College London

Department of Earth Science and Engineering

MSc in Applied Computational Science and Engineering

Independent Research Project
Final Report

Fast Development of Cost Estimates through Machine Learning Algorithms

by

Hisham Taj

Email: hisham.taj21@imperial.ac.uk
GitHub username: acse-hst21
Repository: https://github.com/ese-msc-2021/irp-hst21

Supervisors:

Dr. Francesco Crea
Dr. Lluis Guasch

Project completed in association with Wintershall Dea

September 2022

Contents

1 Introduction **2**

2 Methodology **4**

2.1 Formatting training data 5

2.1.1 Input training data 5

2.1.2 Data cleaning 5

2.1.3 Inflation model 6

2.1.4 Detection of outliers 7

2.2 Training the model 8

3 Code Metadata **9**

4 Results **10**

4.1 Artificial data results 10

4.2 Real data (background) 10

4.3 Real data results 11

5 Evaluation **13**

5.1 Pre-processing 13

5.2 Final predictions 13

6 Conclusion **14**

Abstract

Many databases are available today that contain historical data on well construction and well cost. These databases make use of statistical analysis to produce P10/P30/P50 cost estimates. However, this statistical analysis lacks accuracy due to two key elements:

- The analysis does not account for time dependent factors, such as the variation in the cost of raw materials, equipment and services.
- The analysis considers past cost elements in isolation from their technical descriptors (there is no contextualization for the cost elements).

This lack of accuracy can lead to largely over or under budgeting for an activity.

The aim of this project is to:

(a) Adopt an O&G specific inflation model to account for the time dependent factors that effect the different cost elements.

(b) Replace the statistical analysis with a neural network that can contextualize the cost elements based on a set of given technical parameters.

These two techniques will then be combined to produce higher accuracy cost estimates.

1 Introduction

Engineers in the Wintershall Dea Well Construction Department invest a sizeable amount of time in performing and perfecting cost estimations for potential opportunities and future planned activities. Any over or under estimated cost would represent a poor economic investment, taking more resources from other projects or limiting company growth plans.

Cost estimation can be problematic; in many instances there is little time and/or limited data to produce a viable model. As such, the ability to quickly synthesise algorithms used to produce preliminary cost estimates is highly valuable.

Traditionally, cost estimations were based on proxy, or on the opinion of a select number of experienced individuals. More recently, increased data availability has allowed for more focused quantitative analysis, but these approaches are still time consuming and limited.

Currently, Well Construction staff in Wintershall Dea produce cost estimates out of univariate statistical techniques within Microsoft Excel combined with the expertise of engineers. As such, there is still an overreliance on the experience (and to quote them directly, "gut feeling") of engineers to appropriately tune the statistical results to a meaningful estimation.

A big issue stems from a lack of data. It is not uncommon to face the assessment of areas or operating conditions that have relatively few offset data points.

Given the correlation between reliability of a model and sample size [8], it is necessary to broaden the analysis over the course of the last two decades to ensure the input sample is large enough to produce reliable results. As the time span increases, the effect of time dependant factors rapidly increases (e.g. cost of services, availability of equipment and resources in a given region, etc), and it is necessary to account for these factors to produce a valid cost prediction. Currently in Wintershall Dea, these factors are not quantitatively considered, and it is relied upon the expertise of the engineers, which is ultimately subjective and prone to error, for correction.

This project aims at minimising the involvement of the engineers, in the hopes of producing more precise and replicable cost estimations.

The software is designed to produce six key cost elements to be used as the foundation upon which the final well cost estimate will be produced. The model requires that information regarding the offset wells is inputted (the engineer is still required to select a relevant population of data to be fed to the data model). The code will use the input data to produce the below described cost elements:

- 1) Exploration well cost/day [kUS$/day]
- 2) Appraisal well cost/day [kUS$/day]
- 3) Development well cost/day [kUS$/day]
- 4) Completion cost [kUS$]
- 5) Tangible cost [kUS$]
- 6) Service Development well cost/day [kUS$/day]

The user will first have to select which cost elements they would like to predict (obviously by running the program multiple times, with different data input, the user can produce all 6 cost elements). The user is then given a choice as to whether they would like to use a pretrained model, or if they would like to train a new model. In the case that the user would like to train a new model, the user is required to provide a representative sample as training data.

The dataset is then cleaned to treat any values that have been recorded as *not a number* (NaN), and run through an inflation model to standardise any differences arising from the time period in which the well was constructed. The data is then further cleaned to remove values that are anomalous, before the data is then split into a training and validation set. The training set is used to train the model, with the validation set used to validate the model. Finally, the model produces the requested cost element as a P50 value with an associated confidence interval for the minimum/maximum error associated to the confidence interval.

2 Methodology

The library rapid_cost_estimate has been developed as a standalone piece of code in Python. The user interface can be found in the file *interface.py* in the *scripts* folder. A graphic showing the overall methodology behind the solution can be seen in Fig. 1.

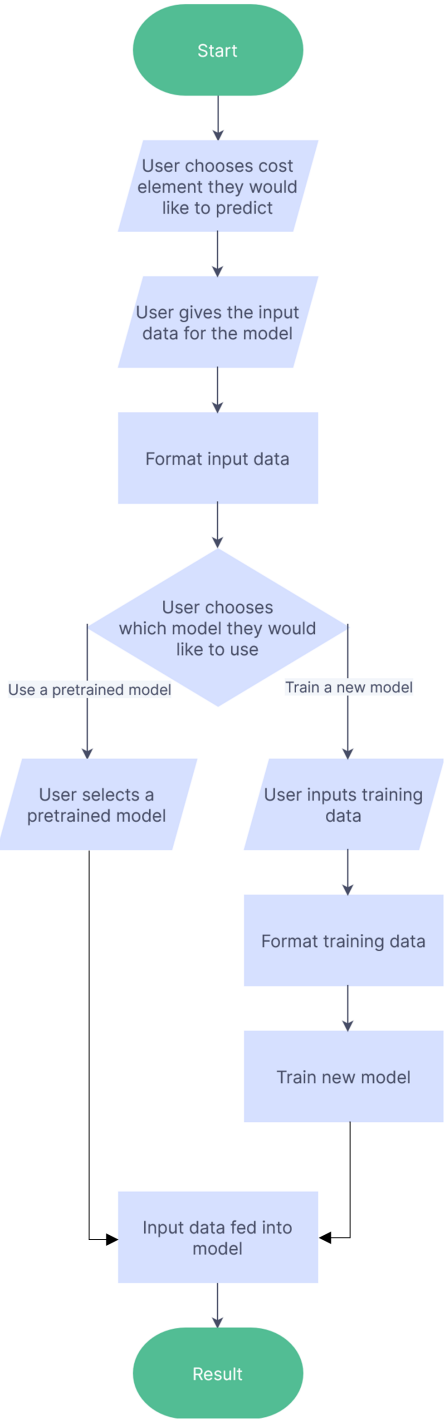

Figure 1: Flowchart depicting the process by which the software operates.

The majority of the workflow as shown in Fig. 1 is self-explanatory and hence only the key elements have been described below.

2.1 Formatting training data

The process of formatting the training data is comprised of multiple stages:

2.1.1 Input training data

There are few commercial databases containing information regarding past well construction activities. Wintershall Dea has access to the IHS Rushmore data repository which allows the user to select offset wells by inserting selection criteria (i.e. region, water depth, coordinates, complexity, type of techniques adopted, etc). Database information is exported as a Microsoft Excel file containing the relevant wells with the associated parameters. The software in this project has been specifically designed to work with data in the IHS Rushmore export file.

The software accepts a Microsoft Excel (.xlsx) file containing the relevant information, with any file format produced by IHS Rushmore considered acceptable. It should be understood, the software is capable of handling any file with data procured from other databases, however, for files not produced from IHS Rushmore, the user will need to ensure this data is formatted accordingly. Specifically, the user will be required to ensure the data conforms to the following requirements:

- The data is passed in as a Microsoft Excel (.xlsx) file
- The parameter names should be found in the 10th row, with the accompanying data found directly below each parameter
- The parameter names will need to match exactly to those found in the IHS Rushmore database (case insensitive)

2.1.2 Data cleaning

Data from IHS Rushmore will likely contain *not a number* (NaN) values, which need to be addressed before the data can be fed into any models. The simplest method to deal with this issue is to remove any well containing missing data within the dataset. Whilst in certain scenarios this is a valid technique, this method is unfeasible in this instance.

As has already been mentioned, due to lack of data, the input sample size is already comparatively small when compared with datasets in comparative scenarios. Thus, removing more data due to NaN values is likely to have a negative impact on the reliability of the model. Thus, a set of criteria were established, such that only in the most extreme scenarios data is removed from the input sample. Data is removed from the model only if one of the following scenarios is met:

- Missing target value - If the target value is missing, all data related to this well is of no use when trying to produce an estimate for this target value.
- Missing spud date - If the spud date for the well is missing it is impossible to feed the data through the inflation model, since accounting for inflation across time is dependent on knowing the time period in question. Without the use of the inflation model, it is impossible to account for time dependent factors effecting the target value.
- Majority of data for a parameter - If the data for a parameter is mainly comprised of missing values there are large uncertainties introduced when trying to model the distribution of data for that parameter.

In instances where NaN values are present but do not meet any of the above criteria, the value is imputed from the data for that parameter. Whilst it is possible to do this using the mean value, a statistical analysis of the relevant parameters suggests that the majority of these parameters have skewed distributions, thus, a more valid technique is to use the median value for data imputation.

2.1.3 Inflation model

In order to account for cost variations with time, all cost elements are fed into an inflation model. This model makes use of oil and gas specific inflation indices from the year 2000 onwards to scale the cost elements accordingly (e.g. cost of materials, cost of services, cost of wells). The software produces an updated list of cost elements (the target value), accurate as of June 2021. All cost elements are estimated and expressed as 'money of today', therefore every estimation is associated with the date of the estimation.

The decision was taken to apply the scaling factor dependent upon the month of the spud date for the well. This was considered the best approach, given that working on the time scale of years would have lacked the necessary resolution to produce accurate cost estimates, but there was not sufficient data to produce accurate scaling on the time scale of days. With regard to the inflation indices upon which this model is based, these are derived from financial reports that are released every financial quarter.

Given that the inflation indices are only updated every three months, but scaling of the cost element is applied every month, there is an issue regarding the continuity of the raw data for the inflation indices, as shown in Fig. 2.

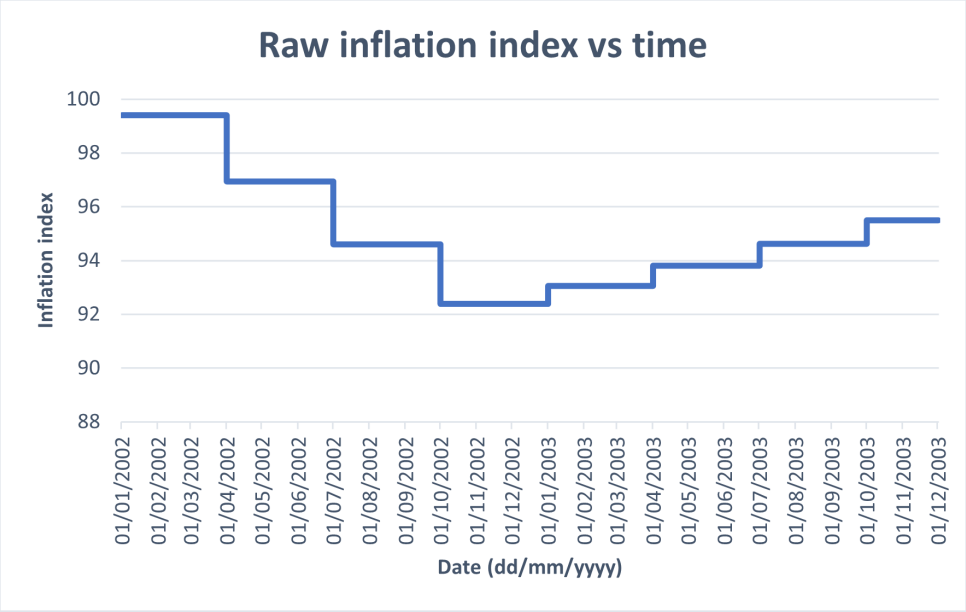

Figure 2: Graph showing how the raw inflation index for well services in Brazil varies through time.

As can be seen in Fig. 2, the lack of data leads to a step change every financial quarter. This is a poor representation of the true variation in value, since inflation indices vary continuously, and thus a curve fitting solution needs to be found to allow for more accurate interpolation in between the given data points. Ultimately, it was decided that linear interpolation was the optimum solution. The results of this are shown in Fig. 3.

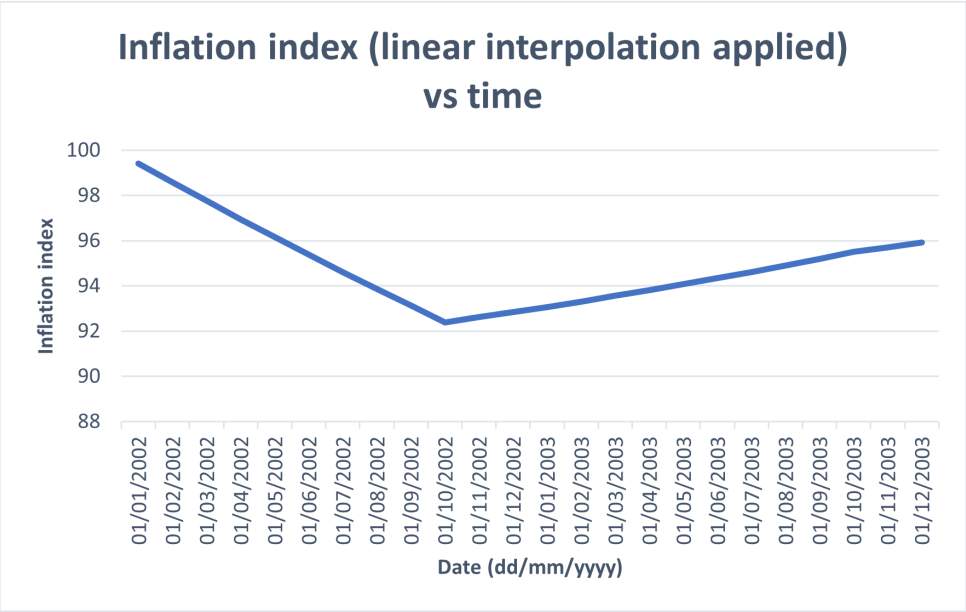

Figure 3: Graph showing how the inflation index for well services in Brazil varies through time when linear interpolation is applied.

It should be understood, given that linear interpolation has been applied between each financial quarter, the graph shown in Fig. 3 is comprised of 7 different linear lines. This however can be difficult to observe directly from the graph, since the gradients of some of these lines are virtually identical. Therefore, the reader is encouraged to view the data directly from the repository to gain a deeper appreciation of the techniques used. The inflation indices can be found at the file path $Data/Inflation\_Indices/Brazil\_Inflation\_Indices.xlsx$.

The final point of consideration is the different cost elements. Obviously, different cost elements will require different inflation indices to scale the values accordingly. As such, the inflation model contains multiple different sets of inflation indices, with the software capable of selecting the appropriate inflation index type based upon the specific cost element requested by the user.

2.1.4 Detection of outliers

As mentioned previously, current modelling techniques are heavily reliant on the expertise of drilling engineers to tune the input data and the statistical analysis. An area in which engineers are heavily involved is anomaly detection. Well construction is by its very nature a very unpredictable endeavour. Extreme weather, equipment failure and many other external factors can affect the cost of drilling and completing a well. Thus, it can be assumed that any dataset fed into the model, even after the data cleaning process, will contain anomalous data. In order for the model to function effectively, these anomalous data points must be removed.

As a result, before the data is fed into the final prediction model, it is first fed into an outlier model. This model is the same type as the final prediction model (a deep, fully connected FFN), but this model is designed to assess the full dataset. The model produces a set of residuals between the true and predicted values, with the largest residuals (the anomalous data points) removed from the dataset. This, in principal, should ensure the final cost estimation model is only exposed to representative data. For specific details regarding the architecture of the network, see 2.2.

2.2 Training the model

As discussed in the Project Plan, the model used to produce the cost estimates themselves will be a deep, fully connected feed forward network (FFN). The model structure itself (not to scale) is shown in Fig. 4.

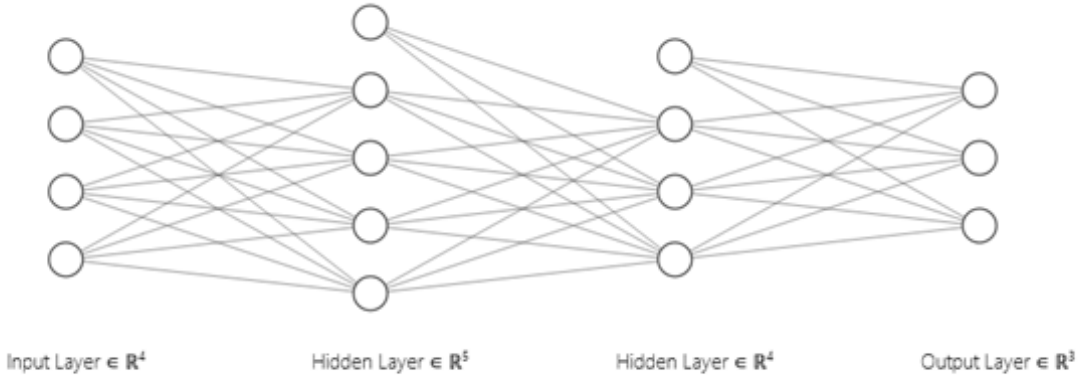

Figure 4: A visual representation of a deep, fully connected feed forward network (bias terms included).

Depending on the target value, the model will be trained on a subsection of the following set of parameters:

- Spud date [date] – the date when the well construction activity started
- Water depth [m] – the distance, expressed in metres, between the sea surface and sea bottom at the well construction location for offshore wells
- Drilled interval [m] – the distance, expressed in metres, of rock drilled
- Complex ratio (dimensionless) – the ratio between the Drilled Interval and the maximum vertical depth reached by the well
- Total number of casings (dimensionless) – an integer number between 1 and 9 expressing the number of tubular strings deployed to isolate formation fluids from the borehole
- Productive days/1000m drilled [days/1000m] – a reflection of the effective number of days employed to drill 1000m of new formation; the effective days exclude all non-productive time or wait on weather time events
- Measured depth (for completion) [m] – the distance, expressed in meters, between the wellhead and the bottom of the borehole
- Completion length [m] – the distance, expressed in meters, between the wellhead and the bottom of the completion string
- Maximum angle through reservoir [degrees] – the maximum inclination along the completed wellbore
- Total completion days/1000m [days/1000m] – the ratio between the total days undertaken for the completion of the well and the measured depth

The activation function used will be the leaky ReLU function, characterised by the equation

$$f(x) = 1(x < 0)(\alpha x) + (x >= 0)(x)$$

since this function is less susceptible to neuron death [16] when the learning rate is set too high, and is immune to issues of saturation, which has been shown to produce "vanishing gradients" [17] in other popular activation functions (i.e. the sigmoid and $Tanh$ functions).

The criterion in use is the L1 loss function, defined as:

$$LossFunction_{L1} = \sum_{i=1}^{n} |y_{true} - y_{model}|$$

This loss function was considered preferable to the L2 loss function given that it is known that the dataset will likely include anomalous data, and the L1 loss function is more robust in the presence of data containing outliers.

The optimiser in use is the adaptive moment estimation (Adam), defined as:

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

where θ is the parameter, η is the learning rate, $\hat{v}_t$ is the bias-corrected first moment (mean) estimate, $\hat{m}_t$ is the bias-corrected second raw moment (uncentred variance) estimate and ϵ is the error [19]. Given that this optimiser makes use of both an adaptive learning rate and a momentum parameter, it has been shown to be an effective choice when considered against other contemporary optimisers [20]. Specifically, in this project when compared against stochastic gradient descent, Adam was by far the more effective optimiser, and thus was the logical choice.

The values for the remaining hyperparameters will be decided via grid search.

3 Code Metadata

This package rapid_cost_estimate (version 1.0.0) was developed in Python 3.10.1 using Jupyter Notebook and Visual Studio Code on a VivoBook_ASUSLaptop X512DA_X512DA with a Windows 10 operating system (version 10.0.19044). Development was primarily done in a PowerShell framework within an Anaconda environment. The project is able to function independently of any platforms outside the generated workflow, with the exception that an input dataset must be provided, whether locally or via the cloud.

In order to write the code to auto-generate the data reports, fpdf was used. Python modules such as *numpy* and *pandas* were used to store and manipulate data, with *matplotlib* being used for data visualisation purposes. PyTorch was the core machine learning library used, both to design the architecture and train the neural networks. Finally, flake8 (to check the code base against the PEP8 coding style), pytest (to run the testing suite) and sphinx (to autogenerate code documentation) were all utilised for continuous integration purposes. For the specific version numbers for these libraries, see the *requirements.txt* file in the GitHub repository.

Owing to security concerns, code relating to the installation of the package has largely been removed from the *setup.py* file and placed in other configuration files. For details on how to install the package, please see the *README.md* file in the base of the GitHub repository.

4 Results

4.1 Artificial data results

Initially, it was necessary to ensure that the prediction algorithm had an effective neural network architecture capable of learning. To prove its learning capabilities the model was exposed to an artificial dataset, constructed from a mixture of real and fictitious data, but significantly less complex. If the model converged to the correct solutions, this would strongly indicate the model was learning effectively; conversely, if the model was incapable of learning on this basic dataset, it was highly unlikely the model had the required complexity to effectively produce cost estimates for real datasets.

The model was tested on the following sets of artificial data:

- Brazil_Data_Constant_TG - Target value constant throughout the dataset
- Brazil_Data_Linear_Depend_TG - Target value a linear function of some of the parameters within the dataset
- Brazil_Data_Log_Depend_TG - Target value a logarithmic function of some of the parameters within the dataset

The results of these tests can be seen in Table 1. Note, the percentage uncertainty for each data point was calculated by taking the residual between the true and model value, and then dividing the residual by the true value.

File name (.xlsx)	**Average percentage uncertainty**
Brazil_Data_Constant_TG	0.132359658%
Brazil_Data_Linear_Depend_TG	0.187063142%
Brazil_Data_Log_Depend_TG	0.144415878%

Table 1: Artificial data - test results

As can be seen from Table 1, the model is capable of converging to the correct solution every time with virtually no error. Thus, it was assumed that the model was capable of learning on the expected type of dataset. It should be noted the residuals could have been further reduced, likely to zero, however, given that this was purely a preliminary test, and this process would have been computationally expensive as well as time intensive, this option was not pursued.

4.2 Real data (background)

As has already been discussed, the model is designed to produce 6 key cost elements:

- 1) Exploration well cost/day [kUS$/day]
- 2) Appraisal well cost/day [kUS$/day]
- 3) Development well cost/day [kUS$/day]
- 4) Completion cost [kUS$]
- 5) Tangible cost [kUS$]
- 6) Service Development well cost/day [kUS$/day]

Since it is not possible to produce all 6 cost elements across a single dataset, a minimum of 2 export files from IHS Rushmore are needed. The variables 1 to 3 are specific to a well type, therefore the

data feed defines the type of variable that is estimated. Thus, different datasets have been used to generate results. For the metadata on the various datasets used, please see Table 2.

				Suitable for producing cost element					
Dataset	**Country**	**Number of wells**	**Number of datapoints**	**1**	**2**	**3**	**4**	**5**	**6**
Dataset A	Norway	123	861	No	No	Yes	No	No	No
Dataset B	Brazil	82	574	Yes	No	No	No	No	No
Dataset C	Brazil	248	1736	No	No	Yes	No	No	No
Dataset D	Brazil	70	490	No	No	No	Yes	Yes	Yes
Dataset E	Malaysia	119	833	No	No	Yes	No	No	No
Dataset F	Malaysia	70	490	No	Yes	No	No	No	No

Table 2: Table showing the metadata for the 6 different datasets used in the testing process

It should be noted that Table 2 only contains information pertaining to the relevant parameters in the datasets (i.e. any redundant parameters have not been considered).

4.3 Real data results

Table 3 shows the results for the performance of the 5 different sets of inflation indices used across the inflation model.

Country	**Type of index**	**Average maximum error**
Norway	Material	1.07%
Norway	Well services	0.78%
Brazil	Material	1.12%
Brazil	Well services	0.79%
Malaysia	Well services	0.78%

Table 3: Table showing the maximum average error for the different inflation indices implemented

Table 4 shows the performance of the most optimised models for the requested datasets

Cost element	**Model name**	**Dataset**	**Total number of wells**	**% data used in final model**	**P50 cost estimate**	**Error**
1	Model_Brazil_1_58.5.tar	B	82	84.1%	639.669 kUS$/day	58.5%
2	Model_Malaysia_2_19.5.tar	F	70	84.3%	398.112 kUS$/day	19.5%
3	Model_Brazil_3_16.9.tar	C	248	84.7%	742.917 kUS$/day	16.9%
3	Model_Norway_3_17.6.tar	A	123	82.9%	838.725 kUS$/day	17.6%
3	Model_Malaysia_3_17.3.tar	E	119	84.9%	329.823 kUS$/day	17.3%
4	Model_Brazil_4_11.5.tar	D	70	84.3%	15.288 kUS$	11.5%
5	Model_Brazil_5_13.6.tar	D	70	84.3%	2.783 kUS$	13.6%
6	Model_Brazil_6_13.8.tar	D	70	84.3%	11.925 kUS$/day	13.8%

Table 4: Table showing the results for the 8 tests run across the 6 different datasets. For access to the models used in the table, as well as extra information, please see the README.md file located in the saved_models folder.

It should be noted that in Table 4, the column "error" is to be understood as the associated error to the measurement provided with a confidence interval of one sigma (standard deviation). For example, in future estimation for cost element 4, the engineer will adopt the P50 cost estimate which is provided with an error of $\pm 11.5\%$ in a confidence interval of one sigma (68%).

With regard to the training process for each model, it should be understood that the target values for each parameter can vary by several orders of magnitude. Since the models trained do not make use of any normalisation techniques in the target values, comparing log loss plots across models is invalid. Furthermore, an analysis of the different training processes shows that the training process for each model was virtually qualitatively identical.

As such, rather than display 8 different log loss plots which would all convey the same information, 1 graph has been shown in Fig. 5 to provide a general summary of the training processes. Once again, it should be emphasised that the target value has NOT been normalised in Fig. 5, and as such the graph is only presented to grant the reader a qualitative understanding of the training process.

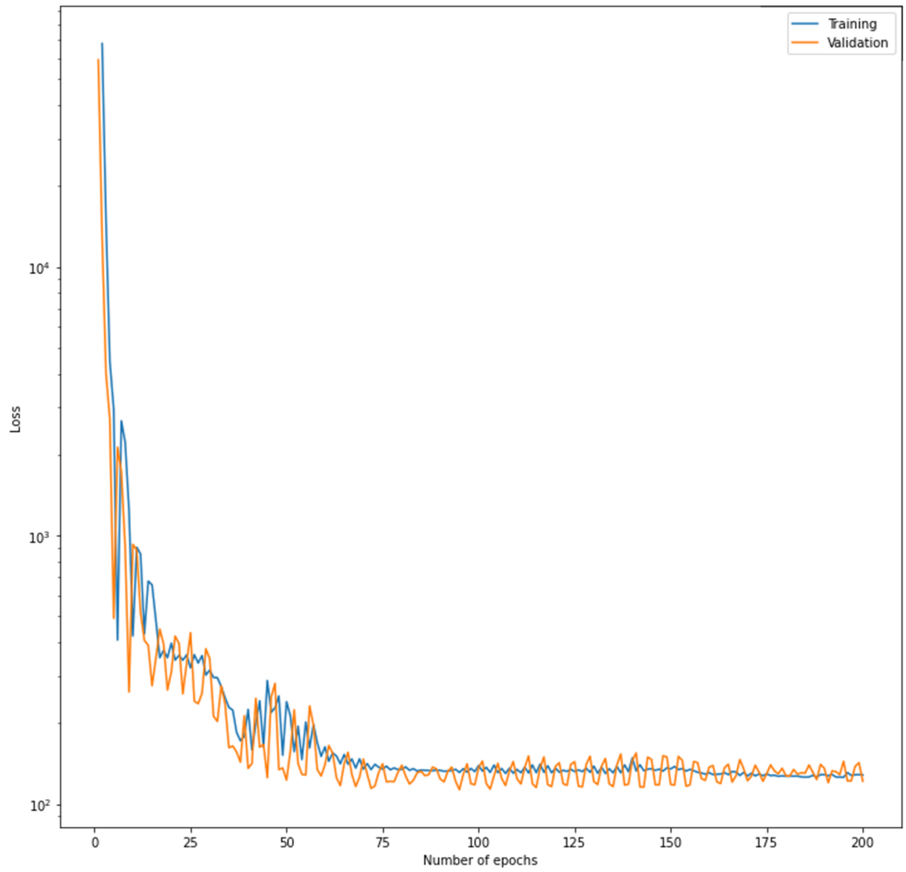

Figure 5: Generic graph showing the training process across the 8 models. Note: this graph is purely to display qualitative trends, and as such, this graph is unsuitable for any quantitative analysis

Finally, it was found that the following were the optimum ranges in which the ideal hyperparameters could be found:

- Hidden size: Between 200 and 600
- Learning rate: Between 0.01 and 0.1
- Number of epochs: Between 50 and 500

The file *grid_search.py* was used to run a grid search within these ranges for these hyperparameters

to produce the optimal models.

5 Evaluation

The evaluation can be split into 2 main sections, pre-processing of data and the final prediction model.

5.1 Pre-processing

The aim of the pre-processing section of the code is to format the data such that it is ready to be fed into the final model. The inflation model functions correctly, able to correctly scale the target value based upon the month of the spud date for the well, as well as select the appropriate inflation index dependent on the provided target variable.

The model is also able to identify wells that are considered "outdated" (any well with a spud date prior to the year 2000) and remove these datapoints from the dataset. All of this is achieved with a maximum average error of 0.91% across the 5 different sets of inflation indices. This is an excellent result, not only because the error is virtually negligible, but also due to the fact this is the maximum error possible, i.e. the worst case scenario. Thus, in reality, the true average error across the set of inflation indices is likely even lower than 0.91%.

It is difficult to draw any significant conclusions from the data cleaning process as this was not part of the "core" scope of work. The data cleaning process was designed specifically to identify NaN values, and treat the values in accordance with the criteria established (see 2.1.2). It should be noted that via separate testing on different datasets, it has already been confirmed that the data cleaning process is functional.

However, it is impossible to ascertain the effect of the data cleaning system on the final cost element estimation for the given datasets, since from the 4984 datapoints provided across the 6 datasets, none of the values are NaN. As such, determining the effectiveness of the data cleaning criteria, specifically with reference to the cost element estimation, is impossible, because none of the criteria have been implemented.

This section of the code functioned well across the 6 given datasets; the anomalous data points were removed effectively, whilst still maintaining a sizeable input sample for the final prediction model (each model managing to maintain between 82-85% of the dataset). This sample size is necessary to ensure the final predictions are reliable.

5.2 Final predictions

Fig. 5 provides a qualitative representation of the training process for the 8 models. As can be seen from this graph, the training and validation set have virtually the same loss values. This is an important observation, as it ensures that the model is not overfitting to either the training or validation samples, proving the model is able to generalise well. One difference between the two samples is the stability of the convergence; whilst the training set converges to a steady value, the validation set displays some oscillatory behaviour. Whilst not ideal, this is to be expected, given the validation set only accounts for 20% of a relatively small input sample.

With regard to the project brief, the main requirements for the model were to produce "fast and accurate estimations ... with enough confidence to issue a non-binding bid or to agree with a partner on a joint venture in a new setting" [24]. Fig. 5 shows the model is able to converge to the final value in approximately 200 epochs. The fact that the model is able to converge in just 200 epochs, a process which takes a matter of minutes (specific times will vary dependent on input sample size,

computational power etc.), demonstrates that the requirement that estimations be "fast" has been met.

With regard to the other requirement, producing an "accurate" estimation, this is significantly harder to evaluate. This is due to the simple fact that accuracy can be summarised as the metric dictating how close the measurement is to the true value [25]. However, all 8 of the cost elements estimated are blind estimates; that is to say the true value is currently unknown. Thus determining the accuracy of the model is impossible. It is however possible to assess the error within the proposed cost estimations.

An analysis of the errors shows 7 of the 8 models were able to produce cost estimates with errors between 10 and 20% (with a one sigma confidence interval), with an average error of 15.4%. There was one exception to this, *Model_Brazil_1_58.5.tar*. Given the error for this variable is approximately triple the next highest error, it is reasonable to assume that the data cleaning process (the removal of the anomalies) is suboptimal and that there are too many heterogenous parameters in the dataset. For the rest of the dataset, an interesting trend can be seen, as shown in Fig. 6.

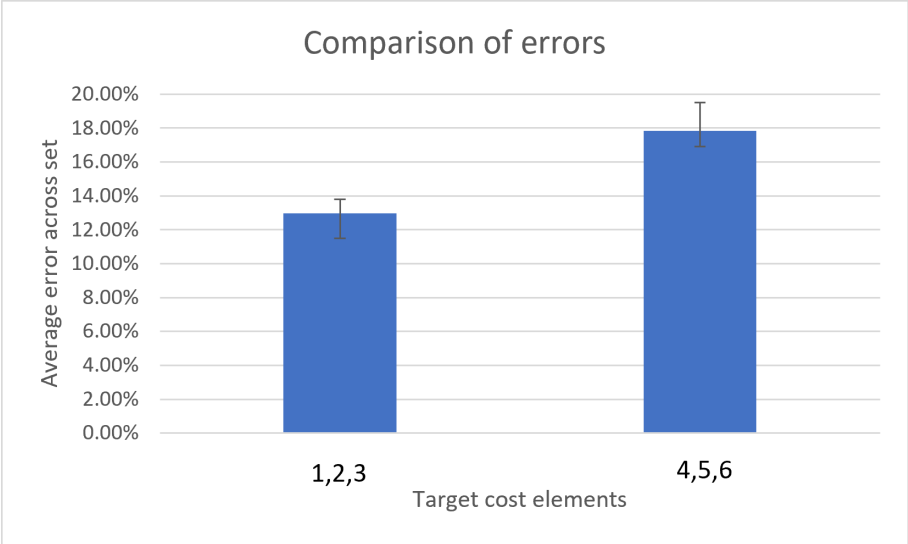

Figure 6: Graph showing the average error for the cost estimates across the sets 1,2,3 and 4,5,6

Fig. 6 shows that the average error across the cost estimates for the set of cost elements 1,2,3 is 37.5% larger than the same errors for the set of cost elements 4,5,6. Not only this, but as can be seen from the error bars, even the lowest error in the set 1,2,3 is still larger than the highest value in the set 4,5,6. Whilst a cursory look at the data could suggest this is purely down to chance, a deeper analysis suggests this is not the case. The likely cause is due to the fact that within these specific sets, all the cost elements have the same input parameters.

From this an obvious conclusion can be drawn. One of the biggest factors influencing the error in the cost element estimation is the data fed into the model, specifically which parameters have been chosen for the input data. This provides an interesting analysis for the user. Ultimately, tuning the model can only improve the final result to a certain degree. Eventually, the type, quality and volume of the data available will become the limiting factor, and as such, this will be the area that requires the most attention.

6 Conclusion

The ability to develop valid cost estimates is vital in order to ensure companies have the necessary information to properly assess business opportunities. Not only do these estimates need to be accurate,

but they should be precise in order to ensure staff can be confident in these estimations. Finally, it is necessary that the ability to synthesise these estimations can be done rapidly, ensuring that the business opportunity can be seized before other competitors.

In this regard, the software works well, specifically when compared with legacy methods. Not only is the error in the cost estimates relatively low, but the training of a new model can be done in a matter of minutes; a significant improvement compared to previous techniques which would typically take a number of weeks/months. The project has shown to the organization the power of a small neural network and how to carry a multi-parameter analysis with a small, yet complex, dataset.

Obviously further improvements can be made in order to further reduce the final error. Whilst the current models reach the limit of what is possible with the current data, it is likely that broader datasets with more parameters would likely improve precision in the final estimation. Thus, engineers at Wintershall should be able to find more relevant input parameters from the IHS Rushmore database, likely further reducing the error.

With regard to improvements to the code itself, the initial project brief has already discussed the possibility of adding automatic dataloaders in further projects [24]. It has also been suggested by those within Wintershall Dea that a GUI could be of use, specifically to employees who are unfamiliar with the computer terminal.

These proposals could be easily implemented. The package has been specifically designed to be easily scaled, with the code intentionally written to allow developers to add extra features. Continuous integration techniques have been used to ensure this scaling process can be run smoothly, without errors occurring, as well as auto-doc technologies to allow for easy reader understanding of the code.

To summarise, the package rapid_cost_estimate version 1.0.0 is a functional library, currently able to produce precise, rapid cost estimates, with the possibility available of improvements in future versions.

References

[1] Kang H. The prevention and handling of the missing data. Korean J Anesthesiol. 2013;64(5):402-406. doi:10.4097/kjae.2013.64.5.402

[2] Donner A. The Relative Effectiveness of Procedures Commonly Used in Multiple Regression Analysis for Dealing with Missing Values. Am Stat. 1982;36(4):378. doi:10.2307/2683092

[3] Graham J. Missing Data Analysis: Making It Work in the Real World. Annu Rev Psychol. 2009;60(1):549-576. doi:10.1146/annurev.psych.58.110405.085530

[4] Fan C, Cui Z, Zhong X. House Prices Prediction with Machine Learning Algorithms. Proceedings of the 2018 10th International Conference on Machine Learning and Computing. 2018. doi:10.1145/3195106.3195133

[5] Rushmore, P. and Rushmore, H., 1993. IHS Rushmore. Aberdeen: Rushmore Reviews.

[6] WEISBERG S. APPLIED LINEAR REGRESSION. 3rd ed. [S.l.]: JOHN WILEY; 2021:21-15.

[7] Dorugade A, Kashid D. Alternative Method for Choosing Ridge Parameter for Regression. Applied Mathematical Sciences,. 2010;4(9):447-456.

[8] Springate S. The effect of sample size and bias on the reliability of estimates of error: a comparative study of Dahlberg's formula. The European Journal of Orthodontics. 2011;34(2):158-163. doi:10.1093/ejo/cjr010

[9] J.F. Lawless and P. Wang, A simulation study of ridge and other regression estimators, Communications in Statistics –Theory and Methods, 14(1976), 1589-1604.

[10] N. Masuo, On the Almost Unbiased Ridge Regression Estimation, Communications in Statistics –Simulation, 17(1988), 729-743.

[11] G. Khalaf and G. Shukur, Choosing ridge parameter for regression Problem, Communications in Statistics. –Theory and Methods, 34(2005), 1177-1182.

[12] Guo Y, Chen Y, Tan M, Jia K, Chen J, Wang J. Content-aware convolutional neural networks. Neural Networks. 2021;143:657-668. doi:10.1016/j.neunet.2021.06.030

[13] Pham T. Recurrent Neural Networks for Structured Data. Deakin University. 2018.

[14] C. Tsai, Y. Chih, W. H. Wong and C. Lee, A Hardware-Efficient Sigmoid Function With Adjustable Precision for a Neural Network System, IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 62, no. 11, pp. 1073-1077, Nov. 2015, doi: 10.1109/TCSII.2015.2456531.

[15] Freedman D, Pisani R, Purves R. Statistics. 4th ed. New York: W. W. Norton & Company; 2018:12-20.

[16] S. C. Douglas and J. Yu, Why RELU Units Sometimes Die: Analysis of Single-Unit Error Backpropagation in Neural Networks, 2018 52nd Asilomar Conference on Signals, Systems, and Computers, 2018, pp. 864-868, doi: 10.1109/ACSSC.2018.8645556.

[17] Roodschild, M., Gotay Sardiñas, J. and Will, A. A new approach for the vanishing gradient problem on sigmoid activation. Prog Artif Intell 9, 351–360 (2020). https://doi.org/10.1007/s13748-020-00218-y

[18] Lee D, In J, Lee S. Standard deviation and standard error of the mean. Korean J Anesthesiol. 2015;68(3):220. doi:10.4097/kjae.2015.68.3.220

[19] Ruder S. An overview of gradient descent optimization algorithms. Cornell University. 2016. doi:https://doi.org/10.48550/arXiv.1609.04747

[20] Z. Zhang. Improved Adam Optimizer for Deep Neural Networks. 2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS). 2018, pp. 1-2, doi: 10.1109/IWQoS.2018.8624183.

[21] Torishniy R. Application of the second-order optimization methods to the stochastic programming problems with probability function. Trudy MAI. 2021;(121). doi:10.34759/trd-2021-121-17

[22] Muller B, Reinhardt J, Strickland M. Neural Networks. 2nd ed. Berlin, Heidelberg: Springer Berlin / Heidelberg; 2013:93-107.

[23] Li H, Xu Z, Taylor G, Studer C, Goldstein T. Visualizing the Loss Landscape of Neural Nets. 2018. doi:https://doi.org/10.48550/arXiv.1712.09913

[24] Crea F. ML Algorithm for Fast Generation of Well Cost Estimate (Updated). Wintershall Dea. 2022.

[25] Johnson K, Hewett S, Holt S, Miller J. Advanced Physics For You. 2nd ed. Glasgow: Oxford University Press; 2015:466-473.

[26] Carpenter W, Tipton R, Byers T. WATER-WELL DRILLING OPERATIONS. Virginia: Department of the Navy; 2008

[27] Jianglin Huang, Yan-Fu Li, Min Xie. An empirical analysis of data preprocessing for machine learning-based software cost estimation. Information and Software Technology. 108-127. 2015; (67). doi.org/10.1016/j.infsof.2015.07.004.