Leveraging the Wisdom of Crowds with Modern Regression, Machine Learning, and Ensembles with Application to Army Software Sustainment

Christian B. Smart, Ph.D., CCEA®
Kimberly Roye, CCEA®
Cheryl Jones
James Doswell
Paul Janusz
Brad Clark

Abstract: Cost estimating often relies on the log-transformed ordinary least squares method for the development of cost estimating relationships. This method has weaknesses; the most significant of which is that it provides estimates that are biased low. These deficiencies can be corrected, and predictive accuracy can be improved, using modern regression methods and applying machine learning techniques. Statisticians have found that predictive accuracy can be even further improved through the combination of multiple models in an ensemble, or crowd approach.

The article discusses these methods in detail and applies them to an extensive dataset of 192 Army systems. Data analysis reveals several types of cost estimating relationships based on release type, release rhythm, and categories of data. This article discusses significance testing and goodness-of-fit metrics for all models developed.

Introduction

Cost estimating is defined as the process of collecting and analyzing historical data and applying quantitative models, techniques, tools, and databases to predict the future cost of an item, product, program or task. Commonly, Log-Transformed Ordinary Least Squares (LOLS) method is used in the development of cost estimating relationships (CERs). There are some disadvantages with using LOLS: estimates are usually biased low and, in some cases, the estimates are not optimal.

LOLS is biased in the sense that it is estimating the median, rather than the mean, of a lognormally distributed estimate. For a lognormal, the median is always less than the mean. This is an issue in cost estimating as estimates are rarely conducted in isolation, but rather as part of a larger WBS or a portfolio of systems. When these values are summed, the resulting total is less than the median of the total estimate. However, the means always add, so it is important to develop estimates at the mean value. (Smart 2017)

To correct for the bias, analysts can consider applying nonlinear regression methods and machine learning methods to develop models and predict future costs. In this paper, we will provide introductions to these less commonly used methods and discuss how the results can be combined to increase the predictive accuracy of cost estimates.

Using a dataset of Army software programs, the analysis will compare results of a log-transformed OLS model to the results using the application of cross-validation using a crowd approach to determine how predictive power is influenced by this method.

Understanding the Crowd Approach

Many people have encountered a situation in which they had to solve a math problem and used more than one way to solve it. As long as both methods yielded the same result, it is reasonable to think this would provide more confidence in the answer. In studying machine learning, research has shown that using multiple methods can improve predictive accuracy. For example, John Elder has been a pioneer in the use of ensemble models and has repeatedly found this to be the case (Elder 2003). Typically, when a model is too complex, it does not predict well in practice. However, the more models one uses, the more complex the overall prediction engine, but the predictions do better in practice than single models used by themselves. This was a revelation to the authors.

An early example of ensemble predictions is a competition to guess the weight of an ox at an English county fair in 1906. Eight hundred people entered the contest. The statistician Francis Galton (he coined the term regression) was interested in the results, and thought that the average, or mean, result, would be far from the actual weight. He was surprised to discover that the actual weight of 1,197 pounds was only one pound from the mean of the 800 submissions. (Surowiecki 2004)

More recently, the data science company Kaggle started hosting competitions to solve problems requiring prediction. The best known of these is the $1 million Netflix prize, where people submitted models to improve the company's recommendation system. The catch with Kaggle submissions is that you get a score on how you do, but you cannot see the actuals to see the error of each estimate. Ensemble techniques have consistently proven to be crucial to winning submissions to these competitions, including the Netflix prize. In that particular competition, two teams tied for first. The tie breaker went to the team who submitted first. The first-place finisher submitted twenty minutes before the second-place finisher and won the entire $1 million prize. (Siegel 2016)

Ensembles are prominent in weather forecasting, especially in storm prediction. Whenever a tropical storm or a hurricane is discussed in a weather forecast, there will be a predicted plot for several different models. Most model predictions will cluster near a common path, but occasionally one or two predictions may be very different. The true path will likely be much closer to the path most of the model predictions rather than the outliers. Using multiple models helps to avoid being influenced by such predictions.

The idea is counter-intuitive. In most applications, you should expect that the best is better than the average. You would not expect two mediocre athletes to perform better on average than a superstar. However, that is what happens with models. The average of several models, some of which may be good, and others that may be mediocre, will on average perform better than the best model. Studies have shown the improvement ranges between five and 30 percent. Even better, using ensembles has been shown to improve out-of-sample prediction, meaning that models will predict well when used in practice. (Siegel 2016)

In *The Wisdom of Crowds* by James Suroweicki (2004), the concept of the crowd approach is discussed. When you put together a large enough and diverse enough group of people and ask them to make decisions affecting matters of general interest, that group's decisions will be intellectually superior to the isolated individual. Applying this concept to cost estimating, we can determine that in the right circumstances, an average forecast is better than a single forecast.

The use of multiple techniques for prediction is called the ensemble approach. An ensemble is a group of items viewed as a whole rather than individually. Suppose we have multiple models from which we would like to choose the "best". The models could be constructed using different datasets, methods, variables or equation forms.

There are (at least) two ways to combine estimates:

- Using a simple average
- Using a more general method that considers correlation between the estimates

We will examine the benefits to be obtained by each of these methods.

Simple Averaging
Simple averaging combines the estimates by computing the means of the estimates.

$$(x_1 + x_2 + \cdots + x_n)/n$$

Let the residuals of a cost estimating relationship (CER) equation be defined by ε. The residuals could be:

- Absolute, as with a linear equation: $\varepsilon = y - f(x)$
- Percentage, as with a nonlinear equation: $\varepsilon = \frac{y-f(x)}{f(X)}$

Regardless of the residual form, the variance of an individual is defined as $\hat{\sigma}^2_{Individual} = E[\varepsilon^2]$. If there are multiple equations, the variance of the average of the CERs is $\hat{\sigma}^2_{Average} = E\left[\left(\frac{1}{N}\sum_{i=1}^{n}\varepsilon\right)^2\right]$. Assuming independence,

$$\hat{\sigma}^2_{Average} = E\left[\left(\frac{1}{N}\sum_{i=1}^{n}\varepsilon\right)^2\right] = \frac{1}{N^2}E\left[\sum_{i=1}^{n}\varepsilon^2\right] + \frac{1}{N^2}E\left[\sum_{i\neq j}\varepsilon_i\varepsilon_j\right]$$

$= \frac{1}{N^2}E[\sum_{i=1}^{n}\varepsilon^2] = \frac{1}{N}\left[\frac{\sum_{i=1}^{n}\varepsilon^2}{N}\right]$.

The quantity $\left[\frac{\sum_{i=1}^{n}\varepsilon^2}{N}\right]$ is the mean of the variances of the individual models. Thus, the SPE of the average of the models is the average of the variances divided by N.

For example, suppose for two nonlinear models, we have standard deviations equal to 30 percent and 50 percent, respectively. The first model has variance $=0.3^2=0.09$ and the second model has variance $=0.5^2=0.25$.

The variance of the simple average is $\frac{1}{2}\left(\frac{0.09+0.25}{2}\right) = 0.085$, which is lower than the better of the two models. By reducing the variance, we have also decreased the uncertainty in the estimate.

Weighted Average
With the simple average, we observe improvement over a single model. What happens when our estimates are correlated, meaning, they use the same data sources or we are comparing similar methods and model forms? When this occurs, the need to use the weighted average approach to incorporate correlation arises.

To calculate the weighted average among models, a correlation matrix, ***nxn***, between the estimates should be created.

The i,jth element of the correlation matrix is $\boldsymbol{E[\varepsilon_i\varepsilon_j]}$. Let $\boldsymbol{\alpha}$ denoted the ***nx1*** vector of weights for the estimates.

The SPE of the weighted average is $\boldsymbol{\alpha^T C\alpha}$**.** The weights should be constrained so that their sum is equal to 1, and the weights should be chosen to minimize the SPE.

With these constraints, we use the Lagrangian multipliers method and minimize

$$\boldsymbol{L = \alpha^T C\alpha - 2\lambda(\alpha^T\vec{1} - 1)}$$

where $\vec{\mathbf{1}}$ denotes the ***nx1*** vector of all ***1s.***

To minimize, we take the first derivative and set it equal to 0.

$$\frac{\partial \boldsymbol{L}}{\partial \boldsymbol{\alpha}} = \boldsymbol{2C\alpha - 2\lambda\vec{1} = 0}$$

Rewriting yields

$$\boldsymbol{2C\alpha = 2\lambda\vec{1}}$$

Dividing both sides by 2 and multiplying both sides by $\boldsymbol{C^{-1}}$ results in

$$\boldsymbol{\alpha = \lambda C^{-1}\vec{1}}$$

Multiplying both sides by $\boldsymbol{\vec{1}^T}$ yields

$$\boldsymbol{\vec{1}^T\alpha = \lambda\vec{1}^T C^{-1}\vec{1}}$$

The left side of the equation is the sum of the weights, which we constrained to be 1, thus

$$\boldsymbol{\lambda = \frac{1}{\vec{1}^T C^{-1}\vec{1}}}$$

Since $\boldsymbol{\vec{1}^T C^{-1}\vec{1}} = \boldsymbol{\sum_{i=1}^{n}\sum_{j=1}^{n} C_{ij}^{-1}}$, if we plug the expression $\boldsymbol{\lambda}$ back into $\boldsymbol{\alpha = \lambda C^{-1}\vec{1}}$ the result is

$$\boldsymbol{\alpha = \frac{C^{-1}\vec{1}}{\sum_{i=1}^{n}\sum_{j=1}^{n} C_{ij}^{-1}}}$$

The SPE of the weighted average method is equal to

$$\boldsymbol{MSE} = \frac{(C^{-1}\vec{1})^T C(C^{-1}\vec{1})}{\left(\sum_{i=1}^{n}\sum_{j=1}^{n} C_{ij}^{-1}\right)^2} = \frac{\vec{1}^T C^{-1}\vec{1}}{\left(\vec{1}^T C^{-1}\vec{1}\right)^2} = \frac{1}{\vec{1}^T C^{-1}\vec{1}} = \frac{1}{\sum_{i=1}^{n}\sum_{j=1}^{n} C_{ij}^{-1}}$$

In the uncorrelated case,

$$\boldsymbol{\alpha_i = \frac{\frac{1}{\hat{\sigma}_i^2}}{\sum_{j=1}^{n}\frac{1}{\hat{\sigma}_j^2}}\ MSE = \frac{1}{\sum_{i=1}^{n}\frac{1}{\hat{\sigma}_i^2}}}$$

To compare the weighted average SPE with others (in the uncorrelated case), we need the following:

Lemma – If **a** and **b** are positive numbers, then $\boldsymbol{\frac{a}{b} + \frac{b}{a} \geq 2}$ with equality when a =b

Proof:

$$\boldsymbol{(a - b)^2 \geq 0}$$

if and only if

$$\mathbf{a^2 - 2ab + b^2 \geq 0}$$

if and only if

$$\mathbf{a^2 + b^2 \geq 2ab}$$

if and only if

$$\frac{a}{b}+\frac{b}{a} \geq 2$$

Comparing Simple Average to Weighted Average
The Variance of the weighted average is less than or equal to the SPE of the simple average with equality only when all the individual SPEs are the same. Note that:

$$Variance_{Simple\ Average} = \frac{1}{N^2}\sum_{i=1}^{N}\hat{\sigma}_i^2$$

$$\text{Variance}_{\text{Weighted Average}} = \frac{1}{\sum_{i=1}^{n}\frac{1}{\hat{\sigma}_i^2}}$$

The weighted average variance is smaller than the simple average variance. To see this, consider

$$\frac{1}{N^2}\sum_{i=1}^{N}\hat{\sigma}_i^2 \geq \frac{1}{\sum_{j=1}^{N}\frac{1}{\hat{\sigma}_j^2}}$$

This is true if and only if

$$\sum_{i=1}^{N}\hat{\sigma}_i^2\sum_{j=1}^{N}\frac{1}{\hat{\sigma}_j^2} \geq N^2$$

The left side of this inequality is a sum of ratios

$$\hat{\sigma}_i^2\frac{1}{\hat{\sigma}_j^2}$$

This expression is equal to 1 when $i = j$. When $i \neq j$, there is always a pair $\frac{\hat{\sigma}_i^2}{\hat{\sigma}_j^2}+\frac{\hat{\sigma}_j^2}{\hat{\sigma}_i^2}$. There are N values for $i = j$ and for other pairs there are $\binom{N}{2} = \frac{N(N-1)}{2}$ values. Therefore, the expression on the left, using the Lemma, is at least $\mathbf{N + 2 * \frac{N(N-1)}{2} = N + N^2 - N = N^2}$ and equality only occurs if all the variances are equal. Therefore, the weighted average variance is smaller than the simple average variance and is strictly smaller when the variances are not all the same value.

We can also show that the weighted average has a variance smaller than the best single model in a crowd. Recall that $MSE_{Weighted\ Average} = \frac{1}{\sum_{i=1}^{n}\frac{1}{\hat{\sigma}_i^2}}$. To see that the variance of the weighted average is less than or equal to the minimum of the individual variances, note that

$\hat{\sigma}_{Min}^2 \geq \frac{1}{\sum_{i=1}^{n}\frac{1}{\hat{\sigma}_i^2}}$ if and only if

$$\hat{\sigma}_{Min}^2\sum_{i=1}^{n}\frac{1}{\hat{\sigma}_i^2} \geq 1$$

Without loss of generality, assume $\hat{\sigma}_1^2 = \hat{\sigma}_{Min}^2$. Then, $\hat{\sigma}_{Min}^2\left(\frac{1}{\hat{\sigma}_1^2}+\cdots+\frac{1}{\hat{\sigma}_N^2}\right) = 1 + \hat{\sigma}_{Min}^2\left(\frac{1}{\hat{\sigma}_2^2}+\cdots+\frac{1}{\hat{\sigma}_N^2}\right) \geq 1.$

In summary, the weighted average approach should be used when estimates or datasets are correlated. Though, in analyses with few data points, correlation estimates may not be as accurate as the simple average.

Practical Example Using Army Data

Army Data Description

Through the support of Office of the Deputy Assistant Secretary of the Army – Cost and Economics (DASA-CE) leadership, the software sustainment initiative has succeeded over the past five years of moving the U.S. Army from a position of making educated guesses on what was being spent on software sustainment and its utility, to being able to provide deep insights from an Army-wide perspective into how software sustainment is being performed, how much it costs, and what software is being delivered to the warfighter. The initiative created an Army Software Sustainment Data Questionnaire which is used to collect system context-information, annual cost and effort data, software release data, and data on software licenses.

The information in the database includes software release level data as well as management and process data on over 192 Army systems in sustainment. The information in the database supports the detailed analysis of software sustainment cost, schedule and risk drivers, and provides insight into the state of software sustainment management and processes practices.

The results establish a robust foundation for software sustainment fact-based decisions, including:

- Allocations of Costs by Work Breakdown Structure (WBS) Elements
- Cost & Schedule Estimating Relationships
- Cost Benchmarks

The amount of data collected resulted in over 411,000 repository data fields based on 192 Systems, 1,040 Releases and 3,434 software licenses, Figure 1.

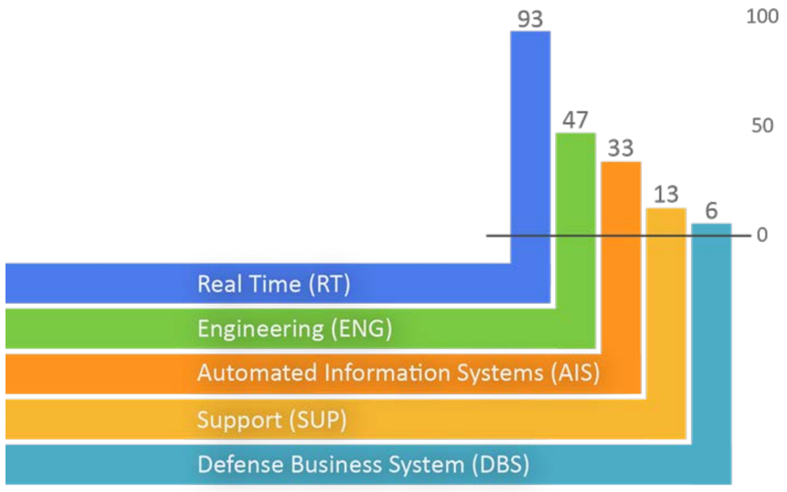

Figure 1. Data Demographics

The Army dataset used for this analysis includes the following variables:

- Total Release Hours (Dependent)
- Super Domain (Independent)
- Total Software Changes (Independent)
- Acquisition Category (ACAT) (Independent)

These variables were selected based on the causal analysis on the data (Jones, et al. 2020). Total Release Hours is defined as the effort (in hours) required to maintain software in the WBS Element 1.0, Software Change Product. This effort changes the software to improve its capability or repair a problem. When systems were divided into application super domains, there were 93 Real-Time Systems (RT), 47 Engineering Systems (ENG), 33 Automation Information Systems (AIS), 13 Support Systems (SUP), and 6 Defense Business Systems

(DBS). In the dataset used for this analysis, there are no DBS datapoints. These three independent variables were identified as being influential in past studies conducted by the army. See "Using Army Software Sustainment Cost Estimating Results (DASA-CE)" from September 2018 for more details.

Systems were asked to report the size measures that were used within their program. Software Changes (SC) was the most common size measure with data provided for 571 releases. SCs are enhancements or maintenance changes to the software.

US DoD's ACAT levels are also analyzed. There are three levels and an additional category for non-Program of Record (non-POR). The difference between each level depends on the location of a program in the acquisition process, funding amount for Research, Development, Test and Evaluation, total procurement cost, Milestone Decision Authority special interest and decision authority. ACAT I programs are major defense acquisition programs.

The upper and lower 10% of the data was trimmed from the dataset to subset extreme cases for separate analysis. Trimming was based on unit cost (total release hours / #software changes). While the data had been scrubbed for hours and cost outliers, some of the unit costs were extremely low and some were extremely high.

Using the trimmed dataset, we used two nonlinear methods and four machine learning methods to predict Total Hours given Super Domain, Total Software Changes, and ACAT level.

Nonlinear Regression Methods
We will introduce two nonlinear methods: Maximum Likelihood Estimation Regression for Lognormal Error (MRLN) and Zero-Percent Bias Minimum Percent Error (ZMPE). Dr. Christian Smart developed the MRLN method, which uses Maximum Likelihood Estimation (MLE) to directly estimate the mean lognormal without the use of transformations (Smart 2017). This method does not require lognormal transformations of either the dependent or independent variables.

The likelihood function, which represents the likelihood of obtaining the sample data, is:

$L(\theta) = \prod_{i=1}^{n} \Pr\,(X_i = A_i|\theta).$

The vector, θ, maximizes the likelihood function in the MLE. Using this technique provides a major advantage: the likelihood function is almost always available. LOLS is an MLE of the median when the residuals are lognormally distributed.

Applying the MLE directly to the residuals yields an estimate of the lognormal mean. The goal for MRLN is to maximize the function:

$$l(\beta_0, \beta_1, \dots, \beta_p, \theta) = -\frac{n}{2}\ln(\theta) - \frac{1}{2\,\theta}\sum_{i=1}^{n}\left(ln(y_i) - \ln(\beta_0) - \sum_{j=1}^{p}\beta_j \ln(X_{ij}) + \frac{\theta}{2}\right)^2$$

Excel Solver can be used to perform this task. When Solver converges on a solution, Excel calculates the optimal values for ***a*** and ***b*** to form the power equation.

Dr. Steve Book developed the ZMPE method, which focuses on minimizing the sum of squared errors subject to the constraint that the sample bias is zero.

The goal of this method is to find a function, $y = f(x)(1+\varepsilon)$, with a multiplicative error, $\varepsilon = \frac{y-f(x)}{f(x)}$, that fits a data set so that the following are satisfied:

- Let ***y*** denote the actual and ***f(x, a, b) = a X^b*** denote the estimate. ZMPE minimizes

$$\sum_{i=1}^{n}\left(\frac{y_i - f(x_i, a, b)}{f(x_i, a, b)}\right)^2$$

subject to the constraint that the sample bias is zero,

i.e.,

$$\sum^{n}\left(\frac{y_i - f(x_i, a, b)}{f(x_i, a, b)}\right) = 0$$

The result is an optimal solution, ***a*** and ***b***, and can be calculated in Excel Solver.

Machine Learning Methods – A Refresher

In "Beyond Regression: Applying Machine Learning to Parametrics" (Roye, Smart 2019), multiple machine learning techniques were discussed in detail. Four supervised learning methods are used for this analysis and are briefly described in the following sections:

- Regression Trees
- Random Forests
- Support Vector Machines
- K-Nearest Neighbors

Supervised learning techniques in machine learning is the process of an algorithm learning from a subset of a given dataset, referred to as the training dataset. In supervised learning, the input variables and output variables are named. The algorithm learns from the mapping function from the input and output. With this method, the goal is to approximate the mapping function so that new outputs can be predicted using new input data.

Regression Trees

A decision tree is a decision support tool useful in classifying data. Tree-based methods are options for analysis, because the data are split into homogenous groups, and the graphs present these splits with the use of branches (called decision nodes) and leaves (terminal nodes). The goal of tree-based methods is to partition data into smaller regions where interactions are manageable. They are useful when there is a non-linear and complex relationship between dependent and independent variables. There are two types of trees: classification and regression trees.

Regression trees are used when the dependent variable of interest is continuous. Figure 2 presents the components of a regression tree.

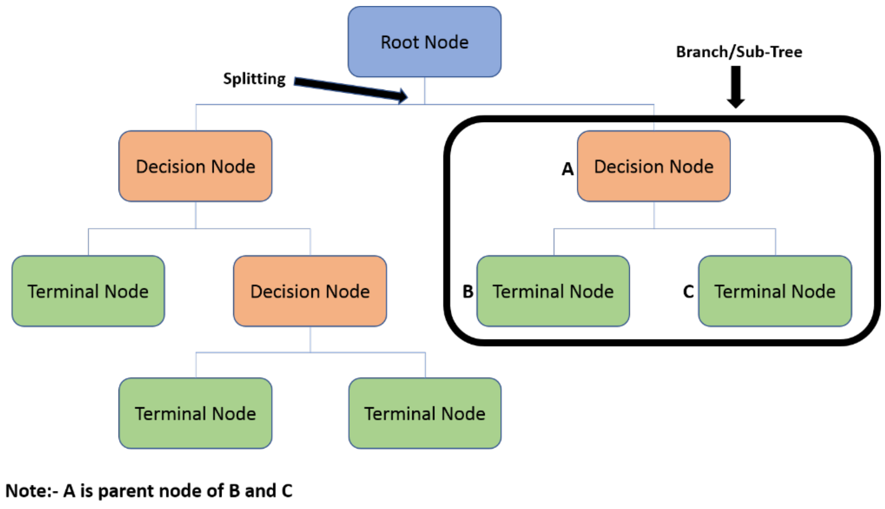

Figure 2: Regression Tree Layout

The root node represents the entire population, or most commonly, the sample dataset that is being explored. Decision trees recursively split a dataset into partitions based on a criterion. Starting with the root node of the tree, the method asks a sequence of yes and no questions to determine the decision nodes. The root node splits into two or more decision nodes. The decision nodes represent the first set of homogenous groups discovered within the dataset. When the algorithm determines which cut-off point minimizes the variance of y for a regression task, the branch ends in a leaf, or terminal node. Leaves represent a cell of partition and have a simple model for that cell; the model is the sample mean of the dependent variable.

Figure 3 provides the regression tree using the example data. At the first decision node, if a release has less than 61 software changes, the left side of the tree when followed to the next decision tree node which splits again on less than 13 software changes. This first decision node represents the first set of homogeneous software releases within the dataset. Based on each smaller group, the tree splits again on either the number of software changes, ACAT level I or II programs, or Real Time super domain releases. The tree ends at the terminal nodes and provides the average Total Release Hours (lognormally transformed) of the data points included in each node. In Figure 3, the numbers in the oblong circles above the nodes (root, decision and terminal nodes) are the average Total Release Hours and the percent of the sample. At the root node, the average Total Release Hours is 8.42 and since this is before any splits occur, there is 100% of the sample included.

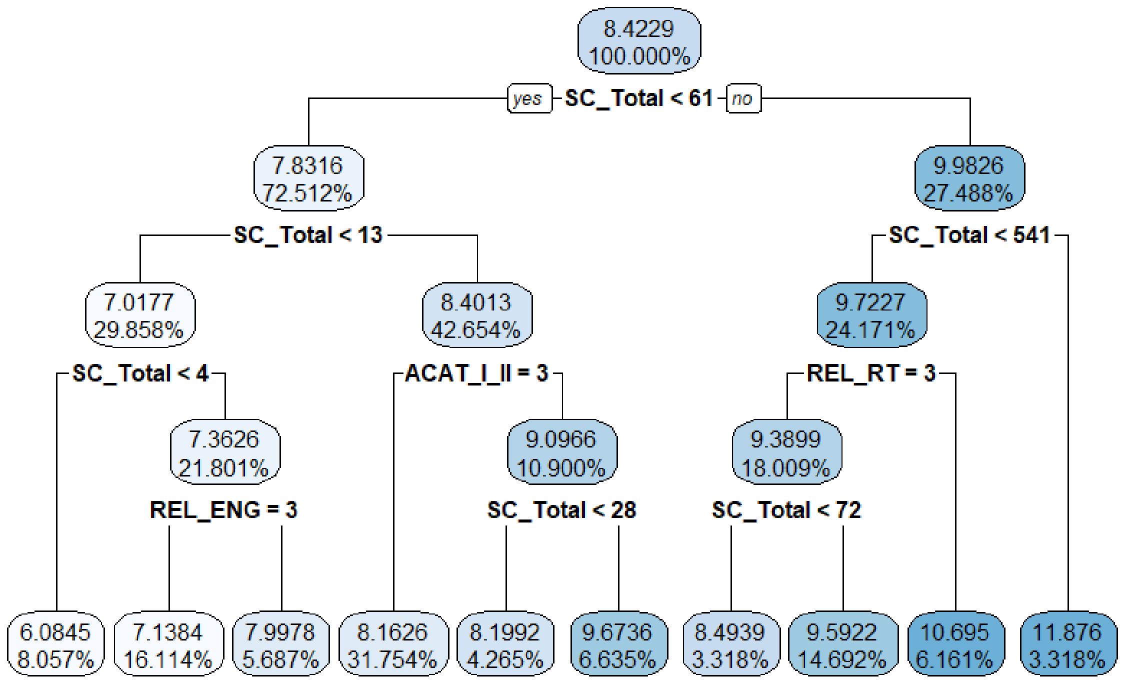

Figure 3: SW Sustainment Regression Tree

Random Forests

The Random Forest algorithm can be thought of as an ensemble approach using regression trees. This approach combines the estimates of multiple regression trees to produce an average. Random forests have been proven to provide better prediction ("wisdom of the crowd" effect). They are also more stable (robust to small amounts of noise). However, since the predictions are rather complex, there is no single equation or CER.

Random forest adds additional randomness to a model. The algorithm searches for the best feature among a random subset of features, which results in more diversity that usually results in better prediction.

Support Vector Machines

The application of Support Vector Machines (SVM) in the 1990s to optical character recognition was very successful. (Boser, et al., 1992) The basic idea for classification with this method is to maximize the margin between classes, which yields maximally robust classification. To apply to continuous output, the analogous idea is to find an equation that is:

- As "flat" as possible, i.e., the coefficients are as small as possible
- Emphasis on sparseness, parsimony
- Makes model less sensitive to errors in inputs
- Minimizes the residuals that are outside a specified range of the estimate (ε-insensitive), e.g., 15%

For a linear equation ***Y = a+bX***, with ***n*** data points the problem becomes

$$\boldsymbol{Minimize\colon \frac{1}{2}(a^2+b^2)+C*\sum_{i=1}^{n}\delta_i}$$

$$\boldsymbol{Subject\ to\ |y_i-a-bx_i|\le z+\delta_i\ for\ all\ i=1,\ldots n}$$

where the delta values are non-negative, and the loss function is insensitive to residuals less than z (user specified), and a weight equal to C is given to the errors (controls for degree of parsimony). For an example of insensitive losses, for a \$10 million project, you may not care about the residual as long as it is no larger than \$1 million.

Given a nonlinear equation ***Y = aX^b^***, take log transforms of the data and apply the linear support vector set up. The insensitivity is now in log-space – the log of the differences between the actual and the estimate.

As an alternative to logarithmic transformation, you can apply the same notion to the absolute value of percentage difference between the actuals and the estimates, i.e.

$$Minimize\colon \frac{1}{2}(a^2+b^2)+C*\textstyle\sum_{i=1}^{n}\delta_i$$

$$Where\ \delta_i=\begin{Bmatrix}\left|\frac{y_i-ax_i^b}{ax_i^b}\right|-0.15\ if\left|\frac{y_i-ax_i^b}{ax_i^b}\right|\ge 15\% \\ 0\ otherwise\end{Bmatrix}\ for\ i=1,\ldots n$$

For solving this optimization problem, Excel's Solver capability can be used.

Results
First, we will present the CER developed using LOLS.

$$\begin{aligned}Y\ (Total\ Release\ Hours)\\ &= 6.42-0.55*ENG\ SD-0.055*RT\ SD-0.95*AIS\ SD+0.30*ACAT+0.72\\ &*Total\ Software\ Changes\end{aligned}$$

Super Domain and ACAT were defined as categorical variables (1 or 0). If ACAT is 1, then the datapoint was obtained from an ACAT I or II program. As a reminder for the Super Domain designations, Real-Time Systems (RT), Engineering Systems (ENG), and Automation Information Systems (AIS). Super Domain and ACAT variables are categorical variables, while Total Software Changes and Total Release hours are continuous variables.

The data used for the nonlinear and machine learning analysis was log-transformed to facilitate the comparison of the results to the LOLS model.

CERs were also developed using ZMPE and MRLN.

Next, we will discuss how to compute the simple and weighted averages using the estimates ($\hat{Y}$) for each of the methods.

For the simple average, the average of $\hat{Y}$ for MRLN, ZMPE, Regression Trees, Random Forest, SVM, and KNN was calculated. This average $\hat{Y}$ was used to calculate the goodness-of-fit statistics.

For example, consider the estimates for the first 10 datapoints for the six methods presented in Table 1.

Observation	MRLN	ZMPE	R-Tree	R-Forest	SVM	KNN
1	7204.21	9768.36	5448.91	8091.82	4074.67	5222.58
2	6885.73	9308.97	5448.91	8108.96	4017.75	4513.26
3	7166.47	4888.93	5448.91	4597.52	5310.05	5316.64
4	574.66	660.47	682.06	4990.52	1758.86	2467.58
5	9015.69	12405.18	5448.91	7944.49	4695.12	3620.33
6	2596.23	1657.38	5448.91	3695.92	3080.11	2635.91
7	8660.85	10453.97	24690.77	13522.31	16743.01	23840.75
8	6885.73	9308.97	5448.91	8108.96	4017.75	4513.26
9	31608.07	47208.50	22762.34	14850.36	38877.70	32809.06
10	2035.47	1278.89	1956.78	3248.21	2335.26	2843.57

Table 1: Estimates for Nonlinear Prediction Methods

We will then take the simple average of the estimate for the methods, such that:

$$Simple\ Average = \frac{\hat{Y}_{MRLN} + \hat{Y}_{ZMPE} + \hat{Y}_{RTree} + \hat{Y}_{RForest} + \hat{Y}_{SVM} + \hat{Y}_{KNN}}{6}$$

Table 2 presents the averages of the six explored methods for the first 10 data points. This new average estimate is then used to calculate the goodness-of-fit statistics.

Observation	MRLN	ZMPE	R-Tree	R-Forest	SVM	KNN	Average
1	7204.21	9768.36	5448.91	8091.82	4074.67	5222.58	6635.09
2	6885.73	9308.97	5448.91	8108.96	4017.75	4513.26	6380.60
3	7166.47	4888.93	5448.91	4597.52	5310.05	5316.64	5454.75
4	574.66	660.47	682.06	4990.52	1758.86	2467.58	1855.69
5	9015.69	12405.18	5448.91	7944.49	4695.12	3620.33	7188.29
6	2596.23	1657.38	5448.91	3695.92	3080.11	2635.91	3185.74
7	8660.85	10453.97	24690.77	13522.31	16743.01	23840.75	16318.61
8	6885.73	9308.97	5448.91	8108.96	4017.75	4513.26	6380.60
9	31608.07	47208.50	22762.34	14850.36	38877.70	32809.06	31352.67
10	2035.47	1278.89	1956.78	3248.21	2335.26	2843.57	2283.03

Table 2: Estimates and Averages for Nonlinear Prediction Methods.

For the weighted average, the calculation is a little more involved. First, all six models were included in the weighted average. The algorithm returned negative weights for two models – MRLN and KNN. Because of the ambiguity of negative weights, we decided to remove those models, and apply the weighted average over four models instead. This resulted in weights for ZMPE, Regression Trees, Random Forests, and SVM equal to 27.0%, 26.0%, 38.4%, and 8.6%, respectively.

MRLN	ZMPE	R-Tree	R-Forest	SVM	Weighted Average
7204.21	9768.36	5448.91	8091.82	4074.67	7511.86
6885.73	9308.97	5448.91	8108.96	4017.75	7389.51
7166.47	4888.93	5448.91	4597.52	5310.05	4958.84
574.66	660.47	682.06	4990.52	1758.86	2423.28
9015.69	12405.18	5448.91	7944.49	4695.12	8220.58
2596.23	1657.38	5448.91	3695.92	3080.11	3548.33
8660.85	10453.97	24690.77	13522.31	16743.01	15874.64
6885.73	9308.97	5448.91	8108.96	4017.75	7389.51
31608.07	47208.50	22762.34	14850.36	38877.70	27710.52
2035.47	1278.89	1956.78	3248.21	2335.26	2302.21

Table 3: Estimates and Weighted Averages for Nonlinear Prediction Methods.

Nonlinear models need different measures of goodness-of-fit than are used for linear ones. The goodness-of-fit measures we use are commonly used in nonlinear modeling and are the nonlinear analogues to traditional linear regression. These are *Pearson's R²*, the *Standard Percent Error*, and *Sample Percent Bias*.

Goodness-of-fit metrics were calculated for each method and used to compare to the LOLS model.

- Pearson's R^2- The square of the correlation coefficient between the actual and estimated effort
- Standard Percent Error (SPE) – The standard deviation of the difference between the actual and estimated effort as a percentage of the estimated effort
- Sample Percent Bias – Average percentage error

Pearson's R^2, which we will refer to as R^2, is defined as:

$$\frac{\left\{n\sum_{i=1}^{n} x_i y_i - \left(\sum_{i=1}^{n} x_i\right)\left(\sum_{i=1}^{n} y_i\right)\right\}^2}{\left\{n\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2\right\}\left\{n\sum_{i=1}^{n} y_i^2 - \left(\sum_{i=1}^{n} y_i\right)^2\right\}}$$

Standard Percent Error is defined as:

The standard percent error is a nonlinear analog to the regression standard error, and is defined as

$$SPE = \sqrt{\frac{1}{n-k} * \sum_{i=1}^{n} \left[\frac{y_i - f(x_i)}{f(x_i)}\right]^2} * 100\%$$

where n is the sample size, and k is the number of fitted coefficients. In this case, lower values are desired.

Sample Percent Bias, which we refer to as Bias, and is defined as:

$$\sum_{i=1}^{n}\left(\frac{y_i - f(x_i)}{f(x_i)}\right)/n$$

The goodness-of-fit statistics were calculated for the in-sample and out-of-sample datapoints. For machine learning techniques, a training and a test sample from the dataset are randomly sampled. For this analysis, an 80% training sample (211 data points) was taken from the dataset, with the remaining 20% (52 data points) being used for testing. The training sample is used to fit each machine learning model, while the test sample provides an unbiased evaluation of the final model fit on the training sample. The in-sample results are calculated from the training sample; the out-of-sample results are calculated from the test sample.

The machine learning methods were all biased low initially because of the log transformation. A sample bias correction adjustment was made, in line with the adjustment MRLN makes, to correct for this sample bias.

The in-sample results are shown in Table 4 and the out-of-sample results are displayed in Table 5.

Method (In-Sample)	**R^2**	**SPE**	**Bias**
LOLS	51.10%	178.74%	-58.17%
MRLN	51.10%	104.34%	2.20%
ZMPE	47.87%	97.93%	0.00%
Regression Trees	62.01%	124.40%	0.00%
Random Forest	47.08%	133.74%	0.00%
SVM	63.90%	135.72%	0.00%
KNN	57.50%	123.43%	0.00%
Simple Average	62.69%	97.05%	13.56%
Weighted Average*	61.45%	78.91%	37.19%

Table 4: Goodness-of-Fit Statistics for the In-Sample Data

Method (Out-of-Sample)	**R^2**	**SPE**	**Bias**
LOLS	92.52%	263.15%	-98.57%
MRLN	92.52%	151.20%	-22.78%
ZMPE	90.71%	143.79%	-17.99%
Regression Trees	84.84%	241.76%	-67.21%
Random Forest	45.59%	307.37%	-48.03%
SVM	45.18%	262.01%	-44.26%
KNN	65.55%	289.17%	-54.06%
Simple Average	91.88%	216.40%	-51.82%
Weighted Average*	90.10%	153.78%	-10.10%

Table 5. Goodness-of-Fit Statistics for the Out-of-Sample Data

For the in-sample data, the single best SPE belongs to ZMPE, followed by MRLN. Among single models, SVM has the highest R^2, followed by MRLN. The simple average has an R^2 comparable to the best of any single model, along with an SPE comparable to the best of any single model. The weighted average has a similar R^2 but a much smaller SPE – however it has a significantly positive bias.

The out-of-sample data is a better indicator of how the models will perform when applied in practice, as the coefficients were not influenced by any of these data. For the out-of-sample data, ZMPE and MRLN are the two best single models – they both have R^2s in excess of 90%, and SPEs significantly better than the other methods. Out-of-sample R^2s can sometimes outperform in-sample R^2s if the training data contains more complicated relationships between the dependent and independent variables than the test dataset. The simple average has a -50% bias and a much higher SPE than MRLN or ZMPE. The weighted average, however, has an R^2 and an SPE, that is comparable to both MRLN and ZMPE, as well as a lower bias. The weighted average is better overall than any single model.

Conclusion

Whether the scenario involves guessing the weight of an animal at a county fair or trying to determine how much a new variant of a combat vehicle will cost, an ensemble approach can often produce a better estimate than a single model. This is a little counterintuitive. To obtain the cost estimate for a program, one might think it would be optimal to find the best cost model to give you an estimate. But a better result may be obtained by averaging estimates from two (or more) mediocre cost models instead. Ensembles seem to consistently produce more accurate estimates.

The authors applied the ensemble concept to estimating Army software sustainment costs. For these data, we have shown that weighted averages generalize better than most models and perform as well or better out of sample than the single best model. Though it is often most common to produce a single regression estimate, introducing the ensemble approach can increase the accuracy of prediction. The authors also applied machine learning methods that can compete with traditional regression methods.

In Appendix A, an additional concept of cross validation is discussed. Though this method was not implemented in this paper, it is an important concept to consider.

References

Boser, Bernhard E.; Guyon, Isabelle M.; Vapnik, Vladimir N. (1992). "A training algorithm for optimal margin classifiers". Proceedings of the fifth annual workshop on Computational learning theory – COLT '92. p. 144.

Dyson, F., "A Meeting with Enrico Fermi," Nature 427 (22 January 2004), page 297.

Elder IV, John F , Ph.D, (2003). The Generalization Paradox of Ensembles. Journal of Computational and Graphical Statistics, 12 (4), 853–864.

Harrell. F.E., *Regression Modeling Strategies*, 2010, Springer-Verlag, New York.

Jones, C., J. Doswell, B. Clark, R. Charette, J. Judy, and P. Janusz, "New Army Software Sustainment Cost Estimating Results," presented at the ICEAA annual workshop, Tampa, May 2019.

Jones, C., J. Doswell, P. Janusz, B. Clark, C. Smart, and K. Roye, "Improving Software Estimating Relationships for Army Software Sustainment Data," ICEAA annual workshop, Virtual, May 2020.

Mitchell, T., *Machine Learning*, 1997, McGraw-Hill, Boston.

Petty, C., C. B. Smart, and J. Lawlor, "Seven Degrees of Separation," presented at ICEAA Professional Development & Training Workshop, June 2015, San Diego, CA.

Roye, K., and C.B. Smart, "Beyond Regression: Applying Machine Learning to Parametrics," presented at the ICEAA Professional Development & Training Workshop, May 2019, Tampa, FL.

Siegel, E., *Predictive Analytics*, 2016, John Wiley & Sons, Hoboken, page 202.

Silver, N., *The Signal and the Noise: Why So Many Predictions Fail – But Some Don't*, 2012, Penguin Books, New York.

Smart, C.B., "The Signal and the Noise in Cost Estimating", presented at the ICEAA International Training Symposium, Bristol, October 2016.

Smart, C.B., "Maximum Likelihood Estimation for Regression of Log Normal Error," presented the ICEAA Professional Development and Training Workshop, Portland, June 2017.

Suroweicki, J., *The Wisdom of Crowds: Why the Many Are Smarter Than the Few and How Collective Wisdom Shapes Business, Economies, Societies and Nations*, 2004, Doubleday, New York.

Dr. ***Christian B. Smart*** *is the Chief Scientist for Galorath Federal. He has a Ph.D. in Applied Mathematics, with more than 20 years of experience with the application of predictive analytics, machine learning, and risk management to defense and aerospace programs. In 2020, Dr. Smart published* Solving for Project Risk Management: Understanding the Critical Role of Uncertainty in Project Management *with McGraw-Hill. He is the 2021 recipient of the Frank Freiman Lifetime Achievement Award from ICEAA.*

Ms. ***Kimberly Roye*** *is a Senior Data Scientist with Bank of America. Starting her career as a Mathematical Statistician for the US Census Bureau, Kimberly transitioned to a career in Cost Analysis over 10 years ago. She has supported several Department of Defense hardware, software and vehicle programs, as well as NASA and the Department of Homeland Security (DHS). She is currently a lead developer of Machine Learning training for the Army and DHS. Kimberly earned a MS in Applied Statistics from Rochester Institute of Technology and a dual BS in Mathematics/Statistics from the University of Georgia.*

Ms. ***Cheryl Jones*** *is a senior systems engineer at the US Army Futures Command and the technical lead for the Army Software Sustainment Cost Estimation initiative. The objective of this project is to provide estimation approaches to accurately estimate and justify software resources. Ms. Jones is the project manager of Practical Software and Systems Measurement and the DoD representative to ISO SC7, System and Software Engineering. She is co-editor of ISO/IEC/IEEE 15288, Systems Life Cycle Processes.*

Dr. ***Brad Clark*** *is Vice-President of Software Metrics Inc. His area of expertise is software cost and schedule data collection, analysis and parametric modeling. He co-authored a book with Barry Boehm titled Software Cost Estimation with COCOMO II and another with Ray Madachy titled Software Cost Estimation Metrics Manual for Defense Systems. Dr. Clark received a Ph.D. in Computer Science in from the University of Southern California.*

Mr. ***Paul Janusz*** *is a senior software quality engineer at the US Army Futures Command at Picatinny Arsenal. He is responsible for implementing software measurement and independent verification and validation for a wide variety of armament software intensive systems. Currently, Mr. Janusz is supporting the Army Software Sustainment Cost Estimation initiative, assessing how software sustainment is being performed on Army programs, analyzing their estimated and actual measurement data, and evaluating the resulting impacts upon the software that is delivered to the warfighter. He is a member of the Practical Software and Systems Measurement (PSM) project, adapting the measurement framework to account for the issues faced by sustainment projects and the unique characteristics of continuous iterative development projects.*

Mr. ***James Doswell*** *is a senior software cost analysis at DASA-CE, responsible for independent government cost estimates for Army systems. He serves as the division's Software Team Lead, responsible for overseeing software analysis, developing predictive models, data collection, and organization wide software cost estimates.*