Modeling Sparse Engine Test Data Using Genetic Programming

Tina Yu
Chevron Information Technology Company
6001 Bollinger Canyon Road
San Ramon, CA 94583
U. S. A.
001-925-842-2393
tiyu@chevron.com

Jim Rutherford
Chevron Oronite Company LLC
100 Chevron Way
Richmond, CA 94802
U. S. A.
001-510-242-3410
jaru@chevron.com

ABSTRACT

We demonstrate the generation of an engine test model using Genetic Programming. In particular, a two-phase modeling process is proposed to handle the high-dimensionality and sparseness natures of the engine test data. The resulting model gives high accuracy prediction on training data. It is also very good in predicting low range data values. However, at least partly due to limitations of the data set, its accuracy on validation data and high range data values is not satisfactory. Moreover, the subject experts could not interpret its real-world meaning. We hope the results of this study can benefit other engine oil modeling applications.

Keywords

Data Modeling; Genetic Programming; Sparse Data; High Dimensionality; Virtual Testing.

1. INTRODUCTION

Laboratory engine tests are among the tools used to measure engine oil performance. These tests are specified in various engine oil performance categories for licensing and certification [3][4][12]. Lubricant additive companies and engine testing laboratories implement and exercise these tests to produce high-quality engine oil.

One of the engine tests used is Sequence IIIE. Early in the year 2000, capability to run this test had nearly been eliminated due to engine parts becoming unavailable. In response to this change, the American Society for Testing and Materials (ASTM) Sequence II/III Surveillance Panel formed the Virtual Test Task Force (VTTF) in May of 2000. The mission of VTTF was to investigate and develop a process, if appropriate, for the use of mathematical models based on IIIE data as a substitute for the Sequence IIIE test.

A virtual engine test protocol was subsequently devised and reported back to the Panel after four months of investigation. However, the proposed process did not receive enough support to be implemented. We believe that it is neither technical nor practical issues that hinder the implementation. Instead, it is the lack of familiarity and comfort with the proposed procedures that prevents the adoption of virtual testing [23].

In this work, we demonstrate how an engine test model can be created using Genetic Programming (GP) [14]. It is hoped that through understanding the data modeling process, the related organizations will become more comfortable with the concept of virtual engine testing. Moreover, we hope other engine oil modeling applications can benefit from this study.

The paper is organized as follows. Section 2 explains the Sequence IIIE engine test data. Section 3 presents GP algorithm as a data-modeling tool. In Section 4, experimental setup is given and in Section 5, the experimental results are presented. Section 6 gives our analysis and Section 7 discusses the results of the study. Section 8 reviews related work and Section 9 contains the conclusions.

2. SEQUENCE IIIE ENGINE TEST DATA

The test has been running for over 10 years. As a result, we have a relatively large data set. However, many of the data have missing information. For example, many potential predictors such as base oil characteristics were not recorded. We made improvement on 172 data records, which are used in this study to generate an engine test model.

There are nine passing criteria for the Sequence IIIE engine test [4]. The criteria are *percent viscosity increase*, *average piston varnish*, *average camshaft plus lifter wear*, *maximum camshaft plus lifter wear*, *average engine sludge*, *oil ring land deposits*, *oil consumption*, *oil related stuck rings*, and *stuck lifters*. A complete engine test system is a suite of nine models; each model predicts one of the nine passing criteria. In this work, we focus on the viscosity increase model. The methodology can be applied to generate other models.

Besides the test results (for the nine passing criteria), each test record contains information about the ingredients of the tested engine oil. For example, viscosity index improver (VII) and dispersants are common engine oil additives. Due to the diversity

of the additives and complex naming conventions, the number of additive variables is large (109). Moreover, it is common for an additive to be present in very few of the data records due to the experimental nature of oil formulation. As a result, the data set is very sparse.

Figure 1 shows that 28% of the 109 additive variables appear only in one test record within the entire data set. More than 50% of the 109 variables appear in less than 5 test records. The combination of high-dimensionality and sparseness has made the engine test data difficult for most data modeling tools.

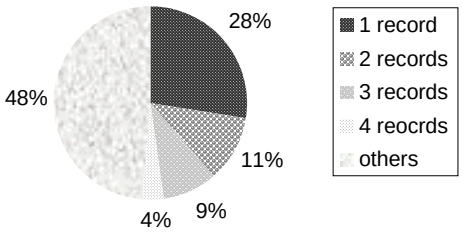

Figure 1: Variables in the data set.

2.1 Aggression and Distribution

Data aggregation and distribution are mechanisms to organize data sets. In this study, we group the additive information into "families" to reduce the size of variables and to increase the density of the data set.

Initially, the expertise of engine oil formulators was used to rearrange and collapse variables in the data. We group the list of 109 additives into 13 families of similar additives. In some cases, additive concentration was simply the sum of concentrations of the additives in the family. In other cases, equivalency relationships based on known or suspected mechanisms were applied. For example, equivalent antioxidancy was derived for the various antioxidants based on chemical functionality. The number of additives in each family varies, ranging from 2 to 25.

After the family grouping is defined, each family is represented with two columns in the data set: one column contains the additive name and the other gives the additive amount used. Table 1 shows the aggregated format for VII additives. If an additive family is not present in a test record, the additive-name is "none" and the additive-amount is 0.

Table 1: Aggregated formats for VII additives.

…	…	VII-name	VII-amount	…
…	…	vii-name-1	0.256	…
…	…	none	0.0	…
…	…	vii-name-2	21.3	…

With this aggregation method, the 109 additives are reduced to 26 variables in the data set. Adding other testing related information, such as end of test date, viscosity grade, and base oil characteristics, the total number of variables is 39. At the end of this aggression process, not only the number of variables is reduced, the density of the data set is also increased.

We used this data set for SGI MineSet [16] to generate a regression tree using its default setup:

- The software performs the splitting of training and testing data in a random manner.
- No cross-validation is performed.
- The software uses a normalized mutual information as the splitting criteria for tree nodes.
- The software uses a confidence-based algorithm to perform tree pruning.

The following model is generated in one run (note that the status window shows the number of training data is 115 while the number of testing data is 57):

```
Viscosity Increase =
       If (saturates <= 98.18) then 118.478
              else if detergent <= 13.473
              then  170.333
                     else 5242.8
```

This result is not satisfactory, as its accuracy (mean absolute error 1007.9) is not good enough to be a useful engine test. We believe the inherent multicollinearity of chemical additives is a challenge to most modeling tools, such as neural networks, support vector machines and linear regression.

As the first attempt to explore the possibility of modeling such a data set using GP, we applied Discipulus software [9] to generate a mathematical expression model (see Section 3 for examples). This approach requires two phases because the model representation in this GP software does not support categorical values (e.g. VII-name).

In the first phase, the 13 additive-name columns (categorical variables) are removed from the data set. The number of the variables is reduced to 26. The purpose of this modeling phase is *features selection*. In the second phase, each of the selected additive-amount variables is expanded with its associated additive name (column distribution). Table 2 shows the distributed format for VII additives (This is the original format before aggregation).

Table 2: Distributed formats for VII additives.

…	…	VII-name-1	VII-name-2	…
…	…	0.256	0.0	…
…	…	0.0	0.0	…
…	…	0.0	21.3	…

In the following sections, we will present the work using Discipulus and the two-phase modeling process to generate an engine test model.

3. GENETIC PROGRAMMING

GP is a machine learning algorithm that is suitable for data modeling [5]. Figure 2 depicts the GP algorithm cycle:

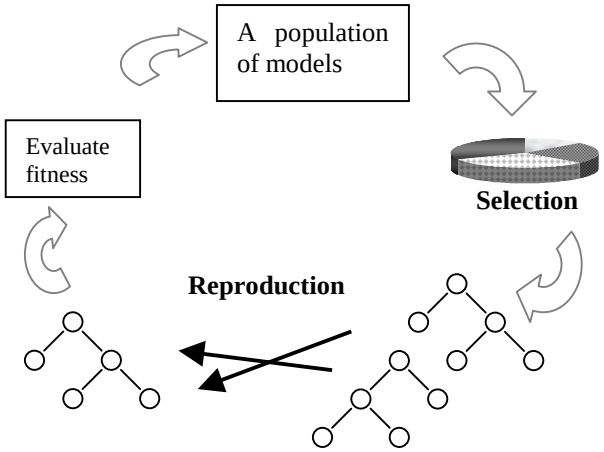

Figure 2: GP algorithm cycle.

Initially, a population of models is randomly created. Based on their fitness, better models are selected for reproduction. Using alternation operations, such as crossover and mutation, new offspring models are generated for fitness evaluation. This process of selection, alternation and fitness evaluation continues until a satisfactory model is generated.

Various representations, selection and alternation schemes have been proposed to suit different applications [25][26]. The Discipulus GP software uses a linear representation to generate mathematical expressions. The following is an example model:

```
Viscosity Increase =

    3.4*detergent+ 3*saturates²

    - aromatics/visindex – 9
```

4. EXPERIMENTS

The 172 data records come from two different engine test laboratories. We used data from one laboratory (104) for training and the other (68) for validation.

In Discipulus, training data is used to evaluate the fitness of the evolved models. This is the fitness that selection for reproduction is based on. In contrast, validation data do not participate in the model generation process. It serves as an unseen data set to give an indication of the robustness of a model. Validation fitness is the selection criterion for the final model, in order to avoid overfitting.

A dynamic training subset selection mechanism [10] is implemented in Discipulus. The subset selection criteria include difficulty, age and randomness. We considered using this feature but decided not to due to the small size of the data set. We believe different results would have been produced if this feature were applied.

Table 3 summarizes the parameters used to conduct the experiments.

Table 3: GP parameters.

Objective	Generate a model that predicts viscosity increase values.
Functions	addition; subtraction; multiplication; division; abs; sqrt; data transformation
Terminals	1st phase: 26 variables 2nd phase: variables selected from 1st phase Constants: 0, 0.5, 1.
Fitness	Linear absolute error
Selection	Tournament (4 candidates/2 winners)
Pop size	100,000
Max Gen	9,000,000
Max Length	256
Genetic Operators	50% crossover (Homologous 95%), 95% mutation (Block mutation rate 30%, Instruction mutation rate 30%, Instruction mutation rate 40%)

5. RESULTS

We made 10 runs and the final model is the one with the best validation fitness:

```
Viscosity Increase = F + 2 * abs (G),
```

where F and G are equations, defining additive usage relating to different oil characteristics. The interpretation of its real-world meaning is not clear (see Section 7).

We used five measurements to evaluate the generated models (other measurements such as uncertainty will be included in the future work). Table 4 summarizes the results of the final model.

Table 4: Experimental results of the final model.

	Training Data	**Validation Data**
Mean Error	37.89	76.89
Median Error	18.14	48.81
Worst Case Error	282.86	452.03
Correlation	0.98	0.71
Coefficient of Variation (R^2)	0.96	0.50

The three accuracy measurements (mean, median and worst case errors) are calculated on data records whose target viscosity-

increase values are less than 1000. This means that three records in the training data and four records in the validation data are excluded from the calculation. This decision is based on the fact that 375 is the maximum allowable viscosity-increase to pass the engine test (see Table 5). Beyond this threshold, as long as the model gives a > 375 prediction, it meets the business needs. Indeed, the GP model gives a high enough value for each of these seven cases to indicate that they fail the test.

The relationship measurements (correlation and R^2) on training data are very good (0.98 and 0.96). However, those on validation data are not as impressive (0.71 and 0.50). Similarly, the accuracy measurements (mean, median and worst case errors) on training data are far superior to those on validation data. Section 7 will provide some possible explanations of such discrepancies.

6. ANALYSIS

Depending on the performance category that the engine oil is tested for, different viscosity increase limits are allowed (see Table 5). For example, the maximum percent viscosity increase value for API CH-4 category is 200. Any value within this threshold is acceptable. The same applies to the other two thresholds (100 and 375).

Table 5: Viscosity increase thresholds vs. test category.

Category	Viscosity Increase (%)
API SG, SH, SJ; ILSAC GF-1 GF-2	375 maximum
API CH-4, ACEA A2-96	200 maximum
ACEA A1-98, A3-98	100 maximum

For the purpose of issuing licenses, what is required of a testing system is its ability to predict whether the performance of the tested engine oil is within the required threshold or not. The actual prediction value is not as important. Based on this merit, the engine test model is performing a classification task; it classifies the tested engine oil to be in one of the following 4 viscosity-increase ranges:

- < 100
- between 100 and 200
- between 200 and 375
- > 375

We analyze the accuracy of the GP model in classifying the engine test data using confusion matrices.

In Table 6 and 7, each row represents the actual values while the column gives the predicted value. As shown, the model is very good at predicting < 100 range. Within the training set, there are 69 such kind of records; the model correctly predicted 66 of them (96% accuracy rate). The accuracy rate on validation data is 91% for this range. Between the range of 100 and 200, the performance drops (24% on training data and 0% on validation data). The model made no correct prediction on 200 to 375 range values. For data value > 375, the accuracy is 100% on training data and 40% on validation data. The overall accuracy is 73% on training data and 50% on validation data.

Table 6: Confusion matrix analysis on training data.

(a)

A \ P	<100	100-200	200-375	>375	Total
<100	66	3	0		69
100-200	20	7	2	0	29
200-375	3	0	0	0	3
>375	0	0	0	3	3
Total	89	10	2	3	104

(b)

A \ P	<100	100-200	200-375	>375	Total
<100	96%	4%	0%	0%	100%
100-200	69%	24%	7%	0%	100%
200-375	100%	0%	0%	0%	100%
>375	0%	0%	0%	100%	100%
Total					73%

Table 7: Confusion matrix analysis on validation data.

(a)

A \ P	<100	100-200	200-375	>375	Total
<100	32	2	1	0	35
100-200	18	0	0	0	18
200-375	10	0	0	0	10
>375	3	0	0	2	5
Total	63	2	1	2	68

(b)

A \ P	<100	100-200	200-375	>375	Total
<100	91%	6%	3%	0%	100%
100-200	100%	0%	0%	0%	100%
200-375	100%	0%	0%	0%	100%
>375	60%	0%	0%	40%	100%
Total					50%

The 0% accuracy rate on data range values between 200 and 375 is the result of small number (3) of training data. As a data-driven modeling method, GP is less likely to generate a good model without enough training data.

7. DISCUSSION

After presenting the model to subject experts, some concerns were raised. First, the accuracy on validation data is much lower than that on training data. We investigated the characteristics of training and validation data and found there are many differences.

For example, eight validation data have large quantities (e.g., 1074 or 1236) of equivalent antioxidancy that produce low

viscosity-increase values (<200). In contrast, this equivalent antioxidancy is of much smaller quantities (e.g., 267, 537, etc.) in the training data. Another example is a frequently used dispersant in training data is hardly used in validation data. Furthermore, validation data used ZNDTP A much more often than ZNDTP B while training data is the other way around. Such discrepancies have made it difficult for GP to generate a common model that works well for both data sets.

These differences, although confounded with the laboratories, do not seem to be caused by laboratory differences, according to a subject expert. They are probably artifacts of the shifting concentration of testing between the laboratories, while there were concurrent changes of industry testing severity and the change of formulating strategies. This information suggests that we should consider the whole data set as one trend of engine test records. Instead of splitting the data based on laboratory association, it might be more appropriate to split them based on other criterion, such as the viscosity increase data range.

Another suggested method to increase the generality of the model is to use a predictor ensemble. With this approach, each sub-model in the ensemble is trained differently, e.g. by using different partition of the data set or different GP parameters etc. As a result, each sub-model would give a different prediction for the same inputs. The final output of the ensemble is the weighted average of the outputs by all sub-models. Numerous researchers have shown empirically that such ensembles generalize well [6][21].

The second concern that subject experts raised is with model fit on training data. The three extreme high value data (1152, 13519 and 19393) are very influential to the calculation of the relationship measurements. In Figure 3, all data except these three extreme high value data are clustered at the lower left corner. The trend line gives high correlation between the actual and the predicated values.

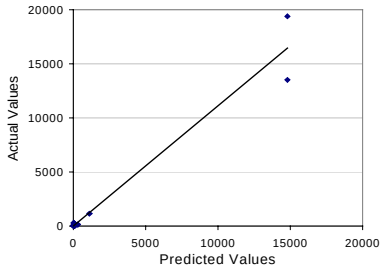

Figure 3: Model fit on training data.

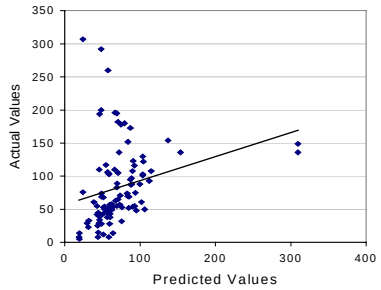

Figure 4: Model fit on training data excluding the three extreme high value data.

However, within the cluster, the correlation between actual and predicted values is not good (see Figure 4). This phenomenon highlights a common dilemma when modeling data with a very wide range of values:

- High value data points are necessary to train a model to be able to predict high range data values;
- However, these high value data points also bias leaning to compromise low range value data.

There are a couple of known methods to work with data set with a wide range of values:

- Convert the data values into logarithm values.
- Customize the fitness function to give proper bias (weight) on both high and low value data. For example, the data with target viscosity increase value greater than 375 can be evaluated with a different standard: when a model gives a prediction greater than 375, the error is 0 on this data point.

Finally, the subject experts also concern with the interpretation of the model. It is hard to attribute real world meaning to terms and operators such as absolute value. Maybe a different representation, one without absolute value operator, is more appropriate.

"The models were not adequate," said one subject expert. "The data should take most of the blame but I also have doubts that GP is an appropriate tool. Performance with the validation set was not good. There were also problems with the model fit to the training data. I don't think this would comfort those people who aren't already comfortable with modeling."

8. RELATED WORK

Using mathematical models for engine testing has been implemented in various applications. For example, Rutherford, Schip and Duteurtre used statistically designed experiments to develop predictions of engine test results from engine oil formulation [22]. Similarly, automotive industry uses mathematical models to predict airflow dynamics instead of wind tunnel testing, or to predict crash performance [19].

U.S. Governments have also adopted the use of mathematical models for testing. The United States Environmental Protection Agency and the California Air Resources Board allow fuel producers to demonstrate clean fuel performance through the use of mathematical models derived from emission test databases [7][8].

In the Machine Learning community, feature selection has long been an active research topic [1]. One approach is using heuristic search algorithms. For example, a rough sets-based algorithm [17] and a Chi2 algorithm [15] were designed to find the relevant features within a larger set of attributes.

Another approach is using decision trees algorithms, such as C4.5 [20]. One result based on the study of Boolean functions indicates that the algorithm is not suitable for filtering irrelevant features [2]. A similar feature selection tool in MineSet is "Column Importance". This algorithm is based on Bayes's theorem; i.e. it assumes the independence of variables. This tool is not

appropriate for data sets where interdependency of variables is abundant, such as the Sequence IIIE engine test data.

Genetic Algorithms (GAs) have also been used to perform feature selection in various applications. For example, Yang and Honavar applied a GA to select features from medical data sets [24]. Another work is by Guerra-Salcedo, Chen, Whitley and Smith, who used hybrid GA-based strategies to filter relevant features in 3 different kinds of data set: a satellite, a DNA and a Cloud data sets [11].

Opitz also proposed a genetic ensemble feature selection algorithm (GEFS) to select a set of feature subsets for ensemble [18]. He demonstrated that this approach produces better ensembles on average than that produced by Bagging and Boosting.

9. CONCLUSIONS

Data modeling for testing is not a new concept. Various statistical approaches and machine learning algorithms have been applied to create models from data to perform testing tasks. We demonstrated the data modeling process using GP with data aggregation and distribution. This approach has generated an engine test model that can predict the viscosity increase of engine oil.

The generated model, however, has not received much support from subject experts due to the following reasons:

- Its accuracy on validation data and high range data values is not satisfactory;
- The model fit on training data is biased;
- The representation is not easy to interpret.

We hope to acquire more quality data to improve the accuracy of the model. Meanwhile, methods to adjust GP learning bias will be developed. We are also considering different model representation to better suit the applications.

10. ACKNOWLEDGEMENTS

We would like to thank Wolfgang Banzhaf and the reviewers for their comments and suggestions. We also thank Ileana Krumme for her support in writing up this work.

11. REFERENCES

[1] Aha, D. W. and Bankert, R. L. A comparative evaluation of sequential feature selection algorithms. In *Proceedings of the Fifth International Workshop on Artificial Intelligence and Statistics*, 1995. Springer-Verlag, NY. Pages 1-7.

[2] Almuallim, H. and Dietterich, T. G. Learning Boolean concepts in the presence of many irrelevant features. *Artificial Intelligence*, 69 (1-2), November 1994. Pages 279-305.

[3] API. *American Petroleum Institute Engine Oil Licensing and Certification System*. 1999.

[4] ASTM. *American Society for Testing and Materials D4485-99b*. Standard Specification for Performance of Engine Oils. 1999.

[5] Banzhaf, W., Nordin, P., Keller, R. and Francone, F. *Genetic Programming: An Introduction*. Morgan Kaufmann Publishers, Inc. San Francisco, CA. 1998.

[6] Breiman, L. Bagging predictors. *Machine Learning* 24 (2). 1996. Pages 123-140.

[7] CARB. *California Procedure for Evaluating Alternative Specifications for Phase 2 Reformulated Gasoline Using the California Predictive Model*. California Air Resources Board, adopted April 20, 1995 and last amended December 11, 1999, Sacramento, California.

[8] CFR 40. *Title 40 of the Code of Federal Regulations*, Part 80, Section 80.45.

[9] Discipulus. Register Machine Learning Technologies, Inc. Littleton, CO. 1998.

[10] Gathercole, C. and Ross, P. Dynamic training subset selection for supervised learning in genetic programming. In *Parallel Problem Solving from Nature III*. 1994. LNCS Vol. 866. Pages 312-321.

[11] Guerra-Salcedo, C., Chen, S., Whitely, D. and Smith, S. Fast and accurate feature selection using hybrid genetic strategies. In *Proceedings of 1999 Congress on Evolutionary Computation*. IEEE. Pages 177-184.

[12] JASO. *Japan-America Society of Oregon Engine Oil Standards*. 2000.

[13] Kira, K. and Rendell, L. A. The feature selection problem: traditional methods and a new algorithm. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, 1992. AAAI/MIT Press, Pages 129-134.

[14] Koza, J. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press. Cambridge, MA. 1992.

[15] Liu, H. and Setiono, R. Chi2: feature selection and discretization of numeric attributes. In *Proceedings of the 7th IEEE International Conference on Tools with Artificial Intelligence*, 1995. Pages 338-391.

[16] MineSet. Silicon Graphics, Inc. Version 3.0. Mountain View, CA. 1999.

[17] Modrzejewski, M. Feature selection using rough sets theory. In *Proceedings of the European Conference on Machine Learning*, 1993, Pages 213-226.

[18] Opitz, D. Feature selection for ensembles. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, 1999, AAAI/MIT Press, Pages 379-384.

[19] Pescovitz, D. Monsters in a box. *WIRED*, December 2000, pages 340-342.

[20] Quinlan, J. R. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA. 1993.

[21] Quinlan, J. R. Bagging, boosting, and c4.5. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, 1996, AAAI/MIT Press, Pages 725-730.

[22] Rutherford, J. A. Van't Schip, C. J., and Duteurtre, Ph. Experience with statistically designed experiments in the VW 1431 test. In *Proceedings of the Third International Symposium on the Performance Evaluation of Automotive Fuels and Lubricants*. 1989.

[23] Rutherford, J. Some statistical, technical, and practical issues in virtual engine testing. *Society of Automotive Engineers Technical Paper Series*, No. 2001-01-1906. 2001.

[24] Yang, J. and Honavar, V. Feature subset selection using a genetic algorithm. *In Genetic Programming 1997: Proceedings of the Second Annual Conference*. MIT Press. Pages 380-385.

[25] Yu, T. and Miller, J. Neutrality and the evolvability of Boolean function landscape. In *Proceedings of the 4th European Conference in Genetic Programming*. 2001. LNCS 2083, Springer-Verlag. Pages 204-217.

[26] Yu, T. Structure abstraction and genetic programming. In *Proceedings of 1999 Congress on Evolutionary Computation*. IEEE. Pages 652-659.