Assignment 5: Ensemble Model Development 2 Using SAS Enterprise Miner

Daanish Ahmed

DATA 640 9041

Fall 2017

dsahmed2334@yahoo.com

Professor Sounak Chakraborty

UMUC

December 3, 2017

Introduction

Heterogeneous models are among the most powerful predictive modeling techniques used today. They involve building several models and combining their outputs to produce a single result. These models tend to be more accurate than individual models, but they are also more difficult to interpret (Knode, 2016a). Heterogeneous models differ from bagging, boosting, and random forests because they involve using different models such as regression or neural networks rather than only using decision trees. They perform well on binary classification problems, and some methods for combining models include the average, maximum, voting average, and voting proportion methods. In this assignment, I will implement all four of these ensemble methods using SAS Enterprise Miner on a dataset containing vehicle accident information. These models will classify accidents based on whether any fatalities occurred. My goal is to accurately identify cases of fatal accidents to prevent deaths in the future. I will implement a variety of distinct model types into each Ensemble node (see Figure 1 in Appendix). My average and maximum models will combine logistic regression, SVM with a linear kernel, neural network, and a decision tree. My voting average model will combine the neural network with a gradient boosting model. Finally, my voting proportion model will use that gradient boosting model and a bagging model. I will compare the ensemble models as well as the individual models to see which is the most accurate and which can identify the highest number of fatal accidents.

My dataset contains 42,183 observations and 24 variables. The target variable is fatalities, which is a binary variable with the values of 0 (no fatalities) and 1 (fatalities). Some of the inputs include speed limit, work zone, interstate highway, pedestrians involved, surface conditions, vehicles involved, and number of injuries (see Figure 3 in Appendix). The dataset contains 3 interval variables, 7 nominal variables, and 14 binary variables—all of which are numeric. The interval variables are number of injuries, speed limit, and vehicles involved. Nominal variables include maximum injury severity, traffic conditions, and surface conditions. Binary variables include weather conditions, pedestrians involved, and work zone. According to the descriptive statistics (see Figure 3 in Appendix), there are no missing values in any of the variables.

Figure 3 also reveals that several inputs are skewed. The interval inputs with the highest skewness are number of injuries (2.74) and vehicles involved (1.87). Examining the dataset reveals that the target variable is skewed as well, since there are only 466 accidents that involved fatalities. Furthermore, some of the inputs have outliers. For instance, number of injuries and vehicles involved have values more than three standard deviations away from the mean. By examining the data (see Figure 2 in Appendix), I also found that "interstate highway" contains errors. This variable should only have the values 0 (no interstate) and 1 (interstate). However, a few observations have a value of 9. These issues will be resolved during data preparation. Finally, I found that the target variable (fatalities) is strongly correlated with maximum injury severity. All cases with a maximum severity of 2 (fatal) also have a fatality value of 1 (fatalities occurred), and vice versa. To solve this issue, I set the maximum injury severity to "rejected."

Data Preparation

I began data preparation by using the StatExplore node to create a bar graph of each input variable's worth (see Figure 4 in Appendix). According to this image, the most significant variables are the number of injuries, injury crash, property damage crash, pedestrians or cyclists involved, and head-on collisions. Likewise, the least important variables are vehicles towed, work zone, traffic way, and whether the accident was on a weekday. The variable distributions reveal that some inputs are highly skewed and may benefit from transformations (see Figure 5 in Appendix). For instance, number of injuries and vehicles involved have clear positive skews—suggesting that most accidents involve few injuries and vehicles. Using the "maximum normal" transform will help to normalize their distributions. I intend to use transformations on some of my individual models, but not all, to see whether they impact the accuracy.

The next step is to handle missing values, outliers, and errors in the data. Some of my individual models (such as regression) will perform poorly without doing these steps, while others (such as decision trees) do not require them. As mentioned earlier, there are no missing values in my dataset. However, variables such as "number of injuries" and "vehicles involved" contain outliers (see Figure 3 in Appendix). To address this, I used the Replacement node to replace all outliers. I set the interval variable default limits method to "standard deviations from the mean" and used the default cutoff value of 3.0 to replace all values outside of this range. I also handled inputs with errors using the Replacement node. I set the class variable unknown levels replacement to "mode," and I used the replacement editor to find variables with incorrect entries. I found that only "interstate highway" contained errors, so I set the error's replacement value to "_UNKNOWN_" to replace these incorrect entries with the mode. Every model will use this node except for the decision tree, as it should not require preprocessing.

Next, I used the Data Partition node on the regression, SVM, neural network, and decision tree models. I designated 50% of data to the training set, 30% to the validation set, and 20% to the test set. Each model uses a different random seed to prevent them from having identical samples. I did not partition the bagging, gradient boosting, or random forest models because these types of ensemble models are typically evaluated on the training data (Knode, 2016a). Next, I implemented two types of transformations on my models: "best" transform and maximum normal.

According to SAS Enterprise Miner Reference Help, these methods involve using multiple transformations on each input. “Best” transform selects the transformation with the strongest impact on the target, while max normal selects the method that maximizes each input’s normality. For each individual model, I selected the transformation that produced the highest accuracy. I used the “best” transform for regression and SVM, and I used “max normal” for the neural network. I did not use transformations for the decision tree, bagging, gradient boosting, or random forest because they perform better without transformations. I will assess these models to determine if using different transformation methods will alter the classification results.

Model Development

I will now describe the creation of my predictive models. My first two Ensemble nodes use the average and maximum methods for combining models. The average method averages all models’ posterior probabilities, while the maximum method selects the maximum probability out of all included models (Knode, 2016b). They will both combine regression, SVM, neural network, and a decision tree. My third Ensemble node uses the voting average method, which averages the results from the most common class (Maldonado, Dean, Czika, & Haller, 2014). It uses gradient boosting and a neural network. The fourth Ensemble node uses the voting proportion method, which computes the output as a ratio of models in the majority group compared to the total number of models (Knode, 2016b). It will use gradient boosting and bagging. I also included a separate HP Forest node to compare with my other models (see Figure 1 in Appendix).

The regression model was created using a Regression node, for which I set the regression type to “logistic regression” and used forward selection as the selection method. Next, I used the

HP SVM node to build my SVM—using the interior point optimization method and setting the kernel to linear. I chose a linear kernel over the other kernels because it produced the highest accuracy while being the simplest to build. I then added Neural Network and Decision Tree nodes into my diagram, keeping each node's default properties. After this, I created my average and maximum ensemble models by using two Ensemble nodes. For the average model, I set both the interval target predicted values and the class target posterior probabilities to "average." Likewise, for the maximum model I set these properties to "maximum." I connected all four individual models to each of the two Ensemble nodes. Finally, I added a Model Comparison node using misclassification rate as the selection statistic. I connected it to all of these models except for the decision tree—which will be analyzed separately since it did not use transformations.

The bagging model was created using three nodes: Start Groups, Decision Tree, and End Groups. For Start Groups, I set the mode to "bagging" and kept the default index count of 10 to prevent long training times. I then attached a Decision Tree node followed by an End Groups node, using the default properties for each. For the gradient boosting and random forest, I used the Gradient Boosting and HP Forest nodes respectively—keeping their default properties. Next, I created my voting average and voting proportion models by adding two more Ensemble nodes. For both nodes, I set the interval target predicted values to "average" and class target posterior probabilities to "voting." I then set the voting posterior probabilities to "average" and "proportion" for the respective models. I linked each Ensemble node with its respective individual models. Finally, I used a Model Comparison node to analyze the output of these two ensemble models as well as the bagging, gradient boosting, and random forest models.

There are several important metrics in this analysis. Firstly, the logistic regression model provides a graph of regression coefficients to find the most important inputs (see Figure 6 in

Appendix). This figure reveals that number of injuries and injury crash have strong negative relationships with the target, implying that fatal accidents often involve fewer people and thus fewer injuries. However, the other models are black box algorithms which makes it difficult to find their important predictors. For comparing models, the ROC curve shows the sensitivity compared to specificity (see Figure 7 in Appendix). We want to select the model with the highest sensitivity per specificity. Furthermore, the fit statistics show the misclassification rate for the training, validation, and test sets (see Figure 9 in Appendix), while the event classification table provides the number of true positives, true negatives, false positives, and false negatives (see Figure 10 in Appendix). The best model will not only have the lowest misclassification rate, but it will also maximize true positives and minimize false negatives. Finally, the classification chart shows a bar graph with the number of correct and incorrect classifications in each class (see Figure 8 in Appendix). It is used to evaluate model performance at identifying 1's and 0's.

Results

I can now assess the results of my analysis to see which model is the best at classifying fatal accidents. I will begin by analyzing my first set of models—namely the regression, SVM, neural network, decision tree, average ensemble, and maximum ensemble models. The ROC curve shows that all models have a near perfect sensitivity per specificity (see Figure 7 in Appendix). This suggests that these models are excellent at identifying 1's in this dataset, but it unfortunately provides little information about which model is the strongest. The classification chart shows that every model contains virtually no incorrect predictions for either the minority or majority classes (see Figure 8 in Appendix). Based on these images, it seems that all of the methods used so far are extremely effective at classification for my dataset. However, this figure once again provides little information about the differences between each model's performance.

To find the differences between models, I examined their fit statistics (see Figure 9 in Appendix). Based on this figure (which excludes the decision tree), all models have validation misclassification rates of 0. The neural network and max ensemble models have extremely small and identical misclassification for their training and test sets. The regression, SVM, and average ensemble model have misclassification rates of 0 for their training and test sets. By looking at the decision tree's fit statistics, we see that it has a misclassification of 0 for the training, validation, and test sets (see Figure 15 in Appendix). These findings show that four out of six models have perfect accuracy. The neural network is the only individual model that contains error. It uses the "max normal" transform instead of "best" transform because I found this method produces stronger results. Thus, its lower accuracy can be attributed not to the transformation type, but rather to the likelihood that neural networks are slightly weaker on this dataset than other models.

I found that the average ensemble method is more accurate than the maximum ensemble method—even though they use the same individual models. To understand why, I examined the event classification table (see Figure 10 in Appendix). We see that no models contain any false negatives, meaning that every model has a perfect sensitivity rate. The only models with any misclassified entries are the neural network and max ensemble model—both of which have one false positive in the training data. This means that both models misclassified a single 0 in the training set, and likely a 0 in the test set as well. Though these results are still extremely accurate, they are not perfect like those of the other models. The max ensemble model used the maximum posterior probability of all models used, which in this case belongs to the neural network. This explains why these models' results are identical and why the max ensemble method has a higher misclassification than the average method. If I were to recommend one ensemble model, I would suggest the average method due to its perfect accuracy. But out of all models, I would suggest using regression since it also has perfect accuracy but is simpler to build than most other models. Furthermore, it is easier to understand and allows us to find the most important inputs.

I will now evaluate the second set of models, which consists of the bagging, gradient boosting, random forest, voting average, and voting proportion models. The ROC curve shows that the gradient boosting and voting proportion models have the lowest sensitivity rates, meaning that they are weaker at identifying true positives (see Figure 11 in Appendix). In this figure, we can ignore the validation and test curves because I did not create a validation or test set for these models. The classification chart appears to indicate a near perfect classification for all models in this set (see Figure 12 in Appendix). However, close examination reveals that gradient boosting has a considerable number of misclassified 1's. This suggests that gradient boosting is the weakest of these models when it comes to identifying true positives.

Based on the fit statistics, the bagging and voting proportion models have misclassification rates of 0—making them the strongest models of this set (see Figure 13 in Appendix). Interestingly, the voting proportion model had the lowest sensitivity in the ROC curve, which is impossible if its misclassification rate is 0. This may be an error caused by SAS Enterprise Miner, but it may also warrant future inquiry. Gradient boosting has by far the highest misclassification rate, followed by voting average and random forest. The event classification table shows a detailed breakdown of each model's misclassified cases (see Figure 14 in Appendix). This figure shows that gradient boosting contains 230 false negatives and only 236 true positives, meaning that it successfully classified only 50.6% of fatal accidents. This is by far the lowest sensitivity of any model, and thus I cannot recommend the gradient boosting model for this dataset.

Figure 14 also shows that the bagging and voting proportion models have no false negatives or false positives, while the random forest contains 1 false negative. As such, bagging is more accurate than random forest in this dataset. Furthermore, we find that the voting average model—which uses gradient boosting and a neural network—is far superior to gradient boosting since it contains 0 false negatives and only 2 false positives. However, it is slightly weaker than the neural network, which only contains 1 false positive. Still, the voting average model has impressive results when considering the gradient boosting model's weak performance at identifying 1's. My recommendation for heterogeneous models would be to use the average ensemble model. Though the average and the voting proportion models both have perfect accuracies, the latter has a much longer training time since it uses bagging and gradient boosting. When considering all models, I would still suggest using regression due to its accuracy and ease of interpretability.

Conclusion

In my analysis, I created 11 models—both individual and ensemble—to classify accidents based on whether fatalities occurred. My goal was to find not only the most accurate model, but also the model that can identify the highest number of fatal accidents. Interestingly, the results show that 6 out of 11 models have a 100% classification accuracy (see Figure 16 in Appendix). The regression, linear SVM, decision tree, average ensemble, bagging, and voting proportion models have successfully classified every observation in this dataset. The gradient boosting model is by far the weakest model, since it failed to classify 230 fatal accidents. But aside from this model, all other 'imperfect' models still have near-perfect accuracy rates. Since there are so many perfect models in this analysis, I will recommend the one that is easiest to implement. With regards to heterogeneous models, I recommend the average method. The voting proportion method also achieved 100% accuracy, but it is considerably slower since it uses bagging and gradient boosting. But when considering all models, I would suggest using logistic regression since it also has perfect accuracy and is simpler than the other models. Furthermore, it is not a black-box algorithm and it provides useful information such as the most important predictors.

I found that transformations do not significantly impact the ensemble models' accuracies. Some individual models perform better when using one transformation type over another. For instance, the neural network achieves a maximum accuracy when using the "max normal" transform. However, heterogeneous models could still achieve perfect accuracy regardless of their individual models' transformation types (or lack thereof). Also, certain methods for combining models proved inferior to others—even when using the same individual models. For instance, the average and maximum methods both use regression, SVM, neural network, and a decision tree. However, the average method achieved a misclassification of 0, while the maximum method had the same error as its neural network. The maximum method was weakened by its most inaccurate model, while the average method was strengthened by having more accurate models.

One "limitation" of this analysis is how easy it is to classify records in this dataset. It may not seem like a problem to have so many models with perfect classification rates. But for analytical purposes, it is more difficult to evaluate the strengths and weaknesses of each model if they are all equally accurate. Furthermore, if many new accidents are added to this dataset, it is likely that the models' performances will be impacted if the new entries are more difficult to classify. Thus, for future analysis I would suggest evaluating these models on a similar dataset containing slightly more "noise." This can help us develop a more realistic understanding of each model while ensuring that the dataset is still relevant to the current business problem. This can also allow us to improve our models, making them better suited to this dataset after adding new records.

References

Knode, S. (2016a, November 1). *Ensemble Models (bagging, boosting, random forest)*. Lecture presented at UMUC. Retrieved November 6, 2017.

Knode, S. (2016b, November 8). *SAS Enterprise Miner Ensemble Models (combining models)*. Lecture presented at UMUC. Retrieved November 11, 2017.

Knode, S. (2016c, November 11). *SAS Enterprise Miner Ensemble Models (combining models) Walkthrough*. Lecture presented at UMUC. Retrieved November 12, 2017.

Maldonado, M., Dean, J., Czika, W., & Haller, S. (2014). *Leveraging ensemble models in SAS Enterprise Miner*. Retrieved from https://support.sas.com/resources/papers/proceedings14/SAS133-2014.pdf

Appendix

Relevant SAS Enterprise Miner Output Images

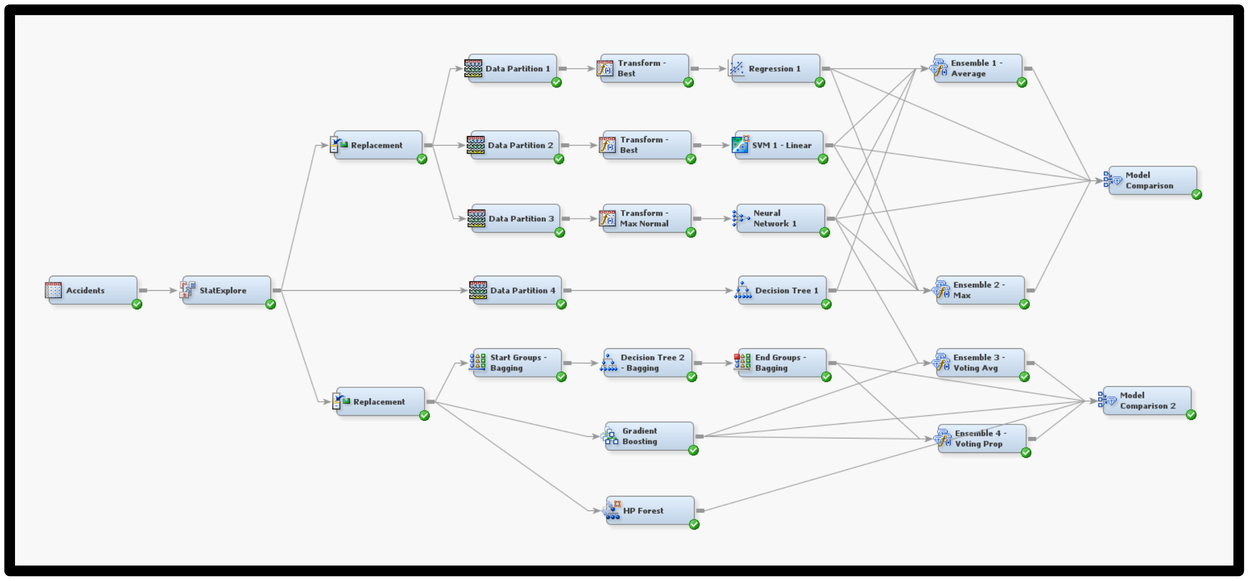
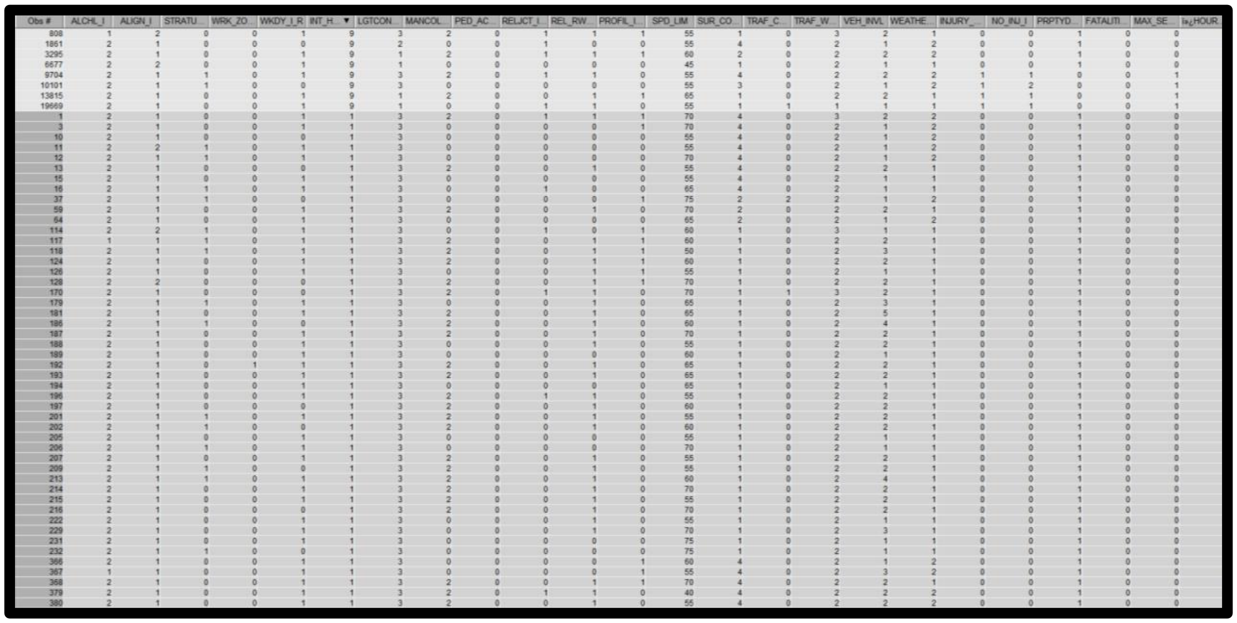
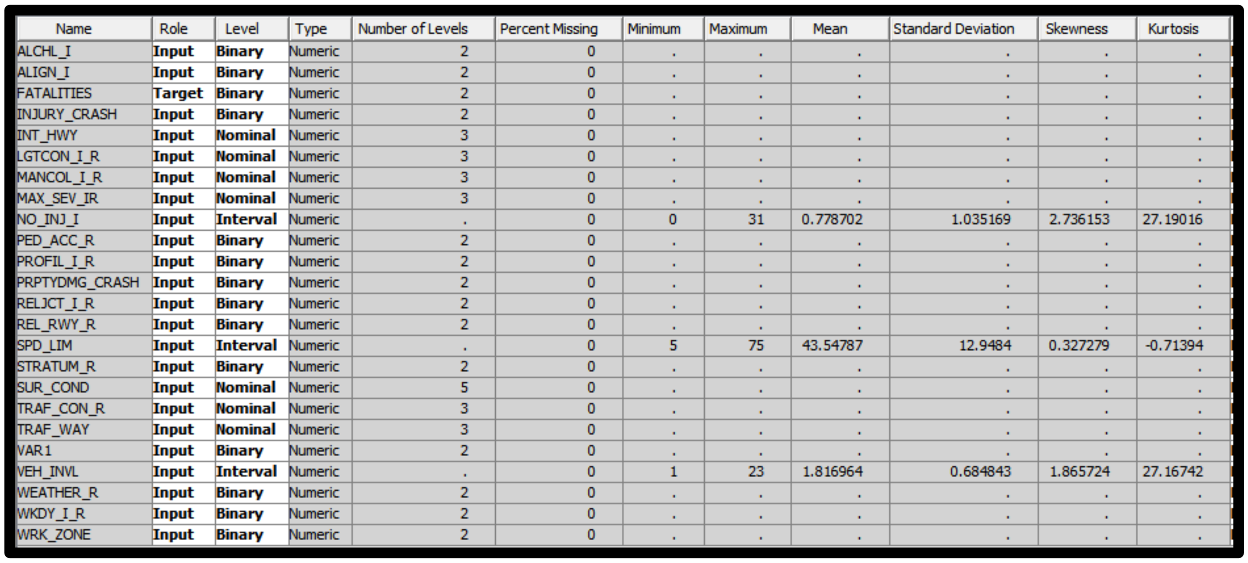


Figure 1. Process Flow Diagram for Accidents Ensemble Models.



Obs #	ALCHL_I	ALIGN_I	STRATU...	WRK_ZO...	WKDY_I_R	INT_H... ▼	LGTCON...	MANCOL...	PED_AC...	RELJCT_I...	REL_RW...	PROFIL_I...	SPD_LIM	SUR_CO...	TRAF_C...	TRAF_W...	VEH_INVL	WEATHE...	INJURY_...	NO_INJ_I	PRPTYD...	FATALITI...	MAX_SE...	ï»¿HOUR...
808	1	2	0	0	1	9	3	2	0	1	1	1	55	1	0	3	2	1	0	0	1	0	0	
1861	2	1	0	0	0	9	2	0	0	1	0	0	55	4	0	2	1	2	0	0	1	0	0	
3295	2	1	0	0	1	9	1	2	0	1	1	1	60	2	0	2	2	2	0	0	1	0	0	
6677	2	2	0	0	1	9	1	0	0	0	0	0	45	1	0	2	1	1	0	0	1	0	0	
9704	2	1	1	0	1	9	3	2	0	1	1	0	55	4	0	2	2	2	1	1	0	0	1	
10101	2	1	1	0	0	9	3	0	0	0	0	0	55	3	0	2	1	2	1	2	0	0	1	
13815	2	1	0	0	1	9	1	2	0	0	1	1	65	1	0	2	2	1	1	1	0	0	1	
19669	2	1	0	0	1	9	1	0	0	1	1	0	55	1	1	1	1	1	1	1	0	0	1	
1	2	1	0	0	1	1	3	2	0	1	1	1	70	4	0	3	2	2	0	0	1	0	0	
3	2	1	0	0	1	1	3	0	0	0	0	1	70	4	0	2	1	2	0	0	1	0	0	
10	2	1	0	0	0	1	3	0	0	0	0	0	55	4	0	2	1	2	0	0	1	0	0	
11	2	2	1	0	1	1	3	0	0	0	0	0	55	4	0	2	1	2	0	0	1	0	0	
12	2	1	1	0	1	1	3	0	0	0	0	0	70	4	0	2	1	2	0	0	1	0	0	
13	2	1	0	0	0	1	3	2	0	0	1	0	55	4	0	2	2	1	0	0	1	0	0	
15	2	1	0	0	1	1	3	0	0	0	0	0	55	4	0	2	1	1	0	0	1	0	0	
16	2	1	1	0	1	1	3	0	0	1	0	0	65	4	0	2	1	1	0	0	1	0	0	
37	2	1	1	0	0	1	3	0	0	0	0	1	75	2	2	2	1	2	0	0	1	0	0	
59	2	1	0	0	1	1	3	2	0	0	1	0	70	2	0	2	2	1	0	0	1	0	0	
64	2	1	0	0	1	1	3	0	0	0	0	0	65	2	0	2	1	2	0	0	1	0	0	
114	2	2	1	0	1	1	3	0	0	1	0	1	60	1	0	3	1	1	0	0	1	0	0	
117	1	1	1	0	1	1	3	2	0	0	1	1	60	1	0	2	2	1	0	0	1	0	0	
118	2	1	1	0	1	1	3	2	0	0	1	1	50	1	0	2	3	1	0	0	1	0	0	
124	2	1	0	0	1	1	3	2	0	0	1	1	60	1	0	2	2	1	0	0	1	0	0	
126	2	1	0	0	1	1	3	0	0	0	1	1	55	1	0	2	1	1	0	0	1	0	0	
128	2	2	0	0	0	1	3	2	0	0	1	1	70	1	0	2	2	1	0	0	1	0	0	
170	2	1	0	0	0	1	3	2	0	1	1	0	70	1	1	3	2	1	0	0	1	0	0	
179	2	1	1	0	1	1	3	0	0	0	1	0	65	1	0	2	3	1	0	0	1	0	0	
181	2	1	0	0	1	1	3	2	0	0	1	0	65	1	0	2	5	1	0	0	1	0	0	
186	2	1	1	0	0	1	3	2	0	0	1	0	60	1	0	2	4	1	0	0	1	0	0	
187	2	1	0	0	1	1	3	2	0	0	1	0	70	1	0	2	2	1	0	0	1	0	0	
188	2	1	0	0	1	1	3	0	0	0	1	0	55	1	0	2	2	1	0	0	1	0	0	
189	2	1	0	0	1	1	3	0	0	0	0	0	60	1	0	2	1	1	0	0	1	0	0	
192	2	1	0	1	1	1	3	2	0	0	1	0	65	1	0	2	2	1	0	0	1	0	0	
193	2	1	0	0	1	1	3	2	0	0	1	0	65	1	0	2	2	1	0	0	1	0	0	
194	2	1	0	0	1	1	3	0	0	0	0	0	65	1	0	2	1	1	0	0	1	0	0	
196	2	1	0	0	1	1	3	2	0	1	1	0	55	1	0	2	2	1	0	0	1	0	0	
197	2	1	0	0	0	1	3	2	0	0	1	0	60	1	0	2	2	1	0	0	1	0	0	
201	2	1	1	0	1	1	3	2	0	0	1	0	55	1	0	2	2	1	0	0	1	0	0	
202	2	1	1	0	0	1	3	2	0	0	1	0	60	1	0	2	2	1	0	0	1	0	0	
205	2	1	0	0	1	1	3	0	0	0	0	0	55	1	0	2	1	1	0	0	1	0	0	
206	2	1	1	0	1	1	3	0	0	0	0	0	70	1	0	2	1	1	0	0	1	0	0	
207	2	1	0	0	1	1	3	2	0	0	1	0	55	1	0	2	2	1	0	0	1	0	0	
209	2	1	1	0	0	1	3	2	0	0	1	0	55	1	0	2	2	1	0	0	1	0	0	
213	2	1	1	0	1	1	3	2	0	0	1	0	60	1	0	2	4	1	0	0	1	0	0	
214	2	1	0	0	1	1	3	2	0	0	1	0	70	1	0	2	2	1	0	0	1	0	0	
215	2	1	0	0	1	1	3	2	0	0	1	0	55	1	0	2	2	1	0	0	1	0	0	
216	2	1	0	0	0	1	3	2	0	0	1	0	70	1	0	2	2	1	0	0	1	0	0	
222	2	1	0	0	1	1	3	0	0	0	1	0	55	1	0	2	1	1	0	0	1	0	0	
229	2	1	0	0	1	1	3	0	0	0	1	0	70	1	0	2	3	1	0	0	1	0	0	
231	2	1	0	0	1	1	3	0	0	0	0	0	75	1	0	2	1	1	0	0	1	0	0	
232	2	1	1	0	0	1	3	0	0	0	0	0	75	1	0	2	1	1	0	0	1	0	0	
366	2	1	0	0	1	1	3	0	0	0	0	1	60	4	0	2	1	2	0	0	1	0	0	
367	1	1	0	0	1	1	3	0	0	0	0	1	55	4	0	2	3	2	0	0	1	0	0	
368	2	1	0	0	1	1	3	2	0	0	1	1	70	4	0	2	2	1	0	0	1	0	0	
379	2	1	0	0	1	1	3	2	0	1	1	0	40	4	0	2	2	2	0	0	1	0	0	
380	2	1	0	0	1	1	3	2	0	0	1	0	55	4	0	2	2	2	0	0	1	0	0	

Figure 2. Accidents Dataset with Errors in "INT_HWY" Variable.



Name	Role	Level	Type	Number of Levels	Percent Missing	Minimum	Maximum	Mean	Standard Deviation	Skewness	Kurtosis
ALCHL_I	**Input**	**Binary**	Numeric	2	0	.	.	.	.	.	.
ALIGN_I	**Input**	**Binary**	Numeric	2	0	.	.	.	.	.	.
FATALITIES	**Target**	**Binary**	Numeric	2	0	.	.	.	.	.	.
INJURY_CRASH	**Input**	**Binary**	Numeric	2	0	.	.	.	.	.	.
INT_HWY	**Input**	**Nominal**	Numeric	3	0	.	.	.	.	.	.
LGTCON_I_R	**Input**	**Nominal**	Numeric	3	0	.	.	.	.	.	.
MANCOL_I_R	**Input**	**Nominal**	Numeric	3	0	.	.	.	.	.	.
MAX_SEV_IR	**Input**	**Nominal**	Numeric	3	0	.	.	.	.	.	.
NO_INJ_I	**Input**	**Interval**	Numeric	.	0	0	31	0.778702	1.035169	2.736153	27.19016
PED_ACC_R	**Input**	**Binary**	Numeric	2	0	.	.	.	.	.	.
PROFIL_I_R	**Input**	**Binary**	Numeric	2	0	.	.	.	.	.	.
PRPTYDMG_CRASH	**Input**	**Binary**	Numeric	2	0	.	.	.	.	.	.
RELJCT_I_R	**Input**	**Binary**	Numeric	2	0	.	.	.	.	.	.
REL_RWY_R	**Input**	**Binary**	Numeric	2	0	.	.	.	.	.	.
SPD_LIM	**Input**	**Interval**	Numeric	.	0	5	75	43.54787	12.9484	0.327279	-0.71394
STRATUM_R	**Input**	**Binary**	Numeric	2	0	.	.	.	.	.	.
SUR_COND	**Input**	**Nominal**	Numeric	5	0	.	.	.	.	.	.
TRAF_CON_R	**Input**	**Nominal**	Numeric	3	0	.	.	.	.	.	.
TRAF_WAY	**Input**	**Nominal**	Numeric	3	0	.	.	.	.	.	.
VAR1	**Input**	**Binary**	Numeric	2	0	.	.	.	.	.	.
VEH_INVL	**Input**	**Interval**	Numeric	.	0	1	23	1.816964	0.684843	1.865724	27.16742
WEATHER_R	**Input**	**Binary**	Numeric	2	0	.	.	.	.	.	.
WKDY_I_R	**Input**	**Binary**	Numeric	2	0	.	.	.	.	.	.
WRK_ZONE	**Input**	**Binary**	Numeric	2	0	.	.	.	.	.	.

Figure 3. Descriptive Statistics of Accidents Dataset.

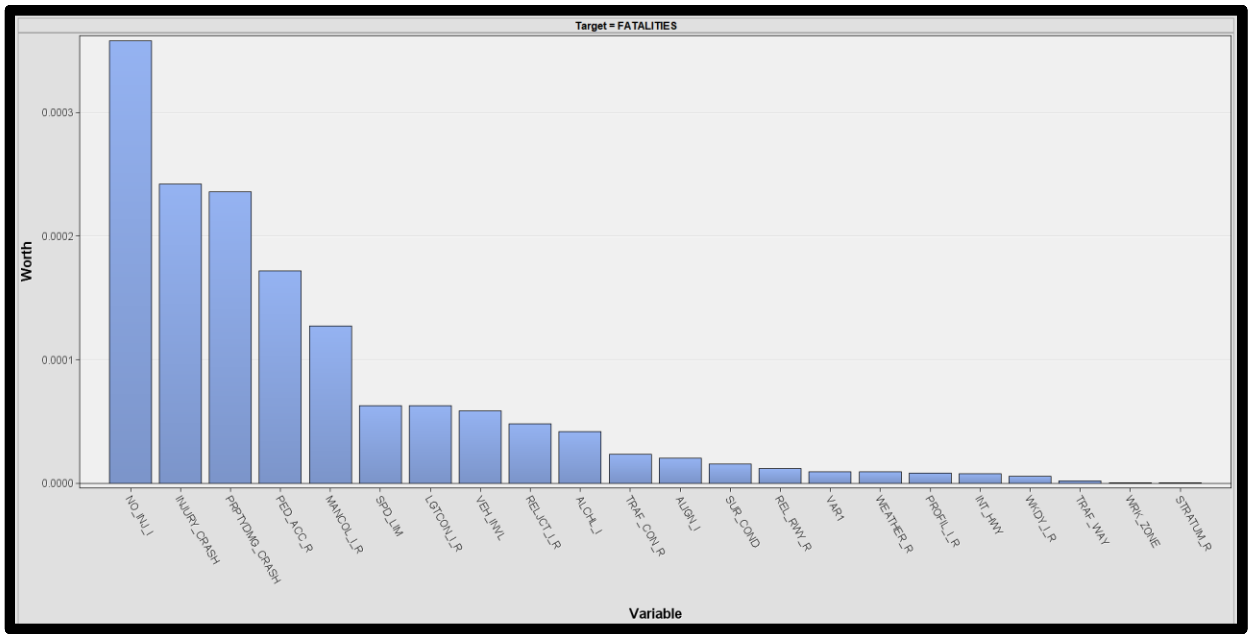
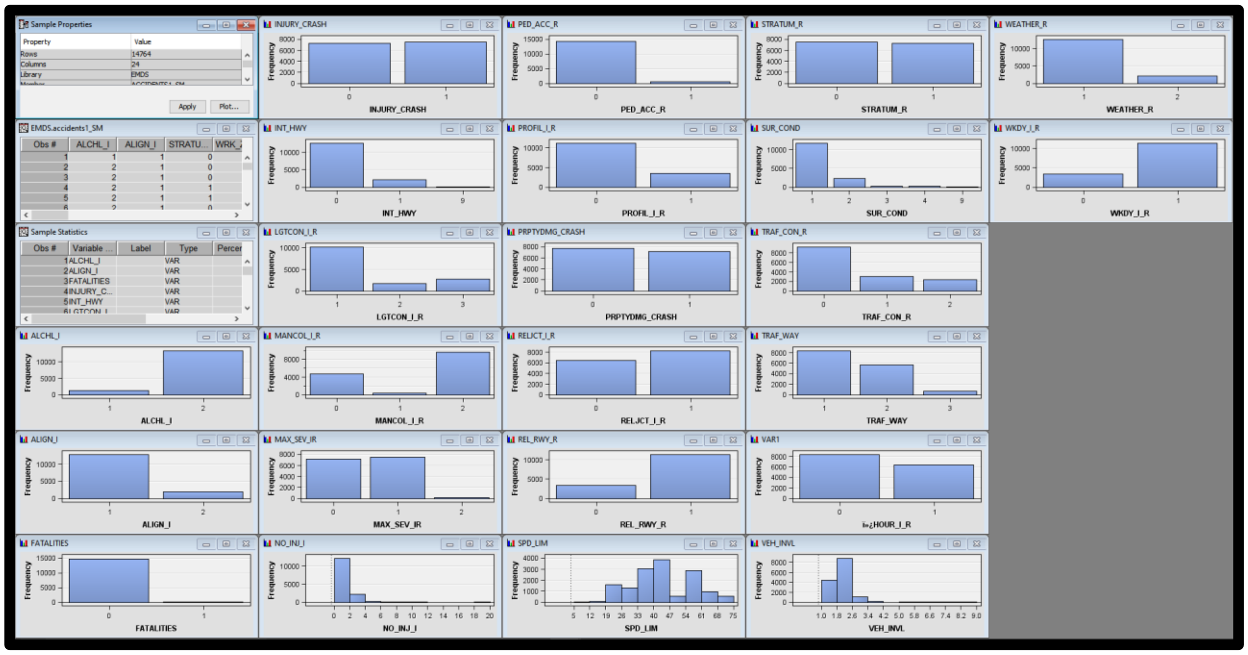


Figure 4. StatExplore Plot of Input Variable Worth.


Sample Properties

Property	Value
Rows	14764
Columns	24
Library	EMDS

EMDS.accidents1_SM

Obs #	ALCHL_I	ALIGN_I	STRATU...
1	1	1	0
2	2	1	0
3	2	1	0
4	2	1	1
5	2	1	1

Sample Statistics

Obs #	Variable ...	Label	Type
1	ALCHL_I		VAR
2	ALIGN_I		VAR
3	FATALITIES		VAR
4	INJURY_C...		VAR
5	INT_HWY		VAR

Figure 5. Initial Distributions of Variables in Accidents Dataset.

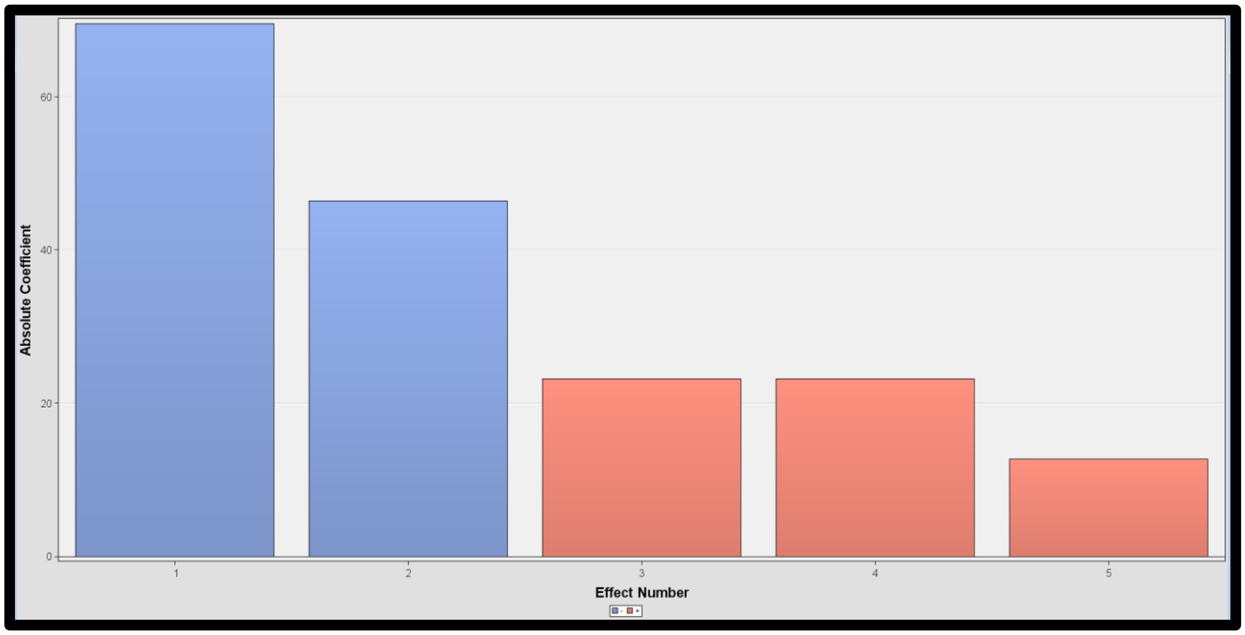


Figure 6. Graph of Regression Coefficients from Logistic Regression Model.

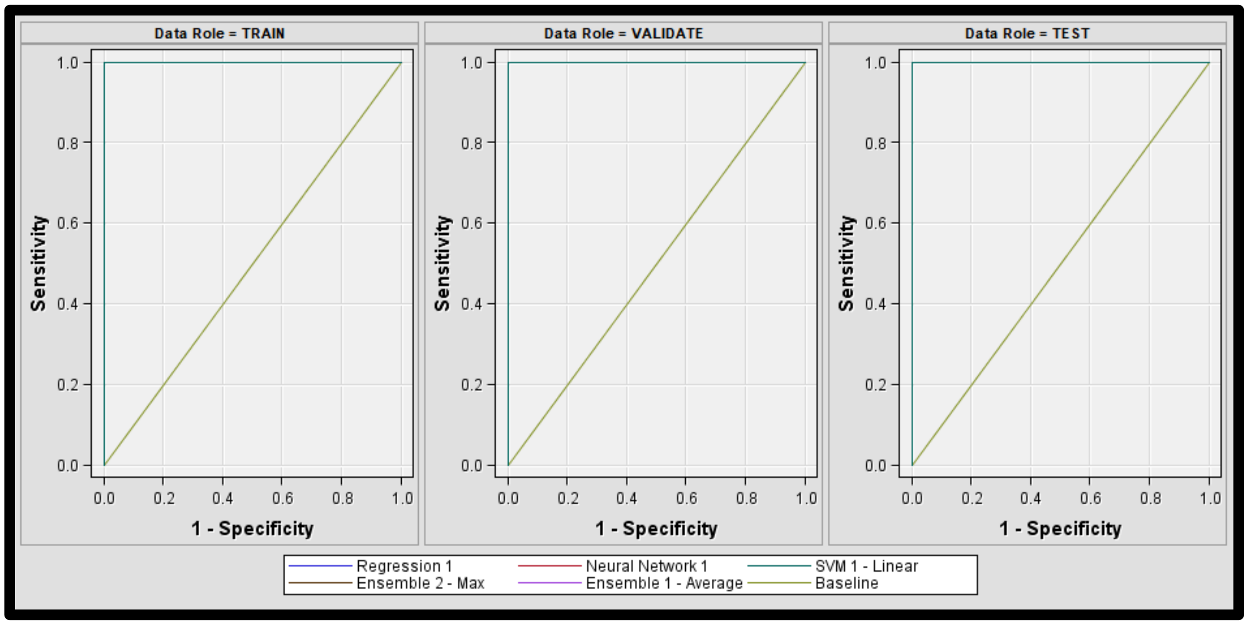


Figure 7. ROC Curve for First Set of Models.

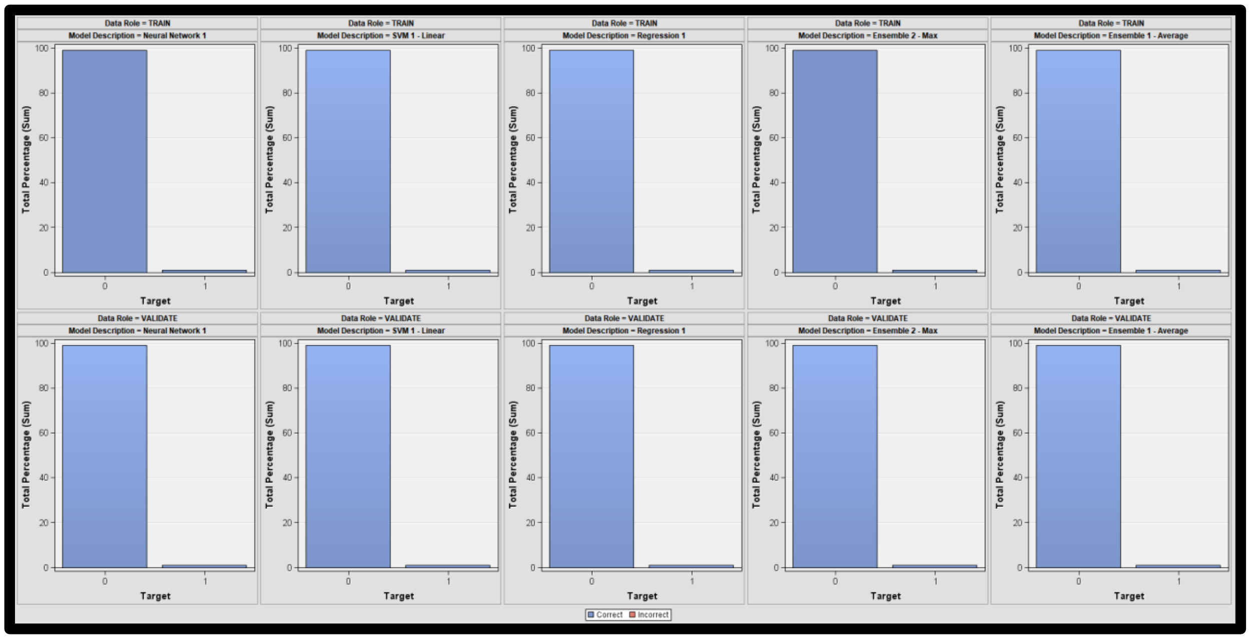
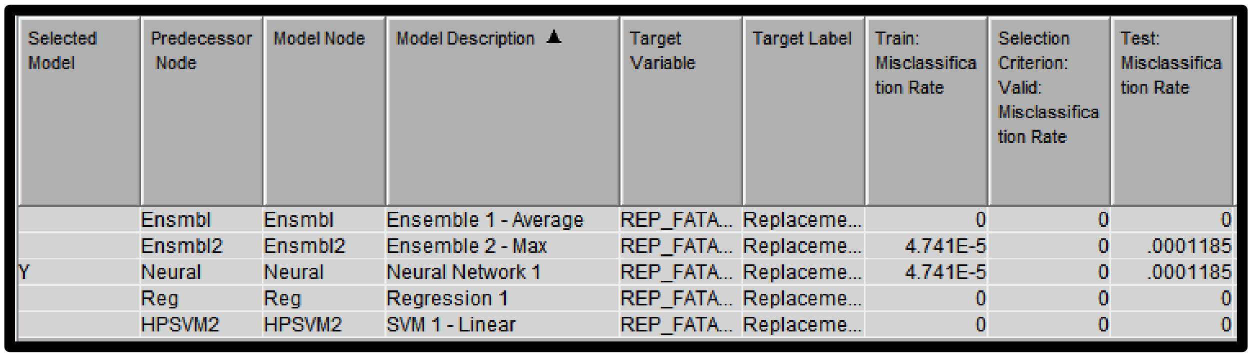
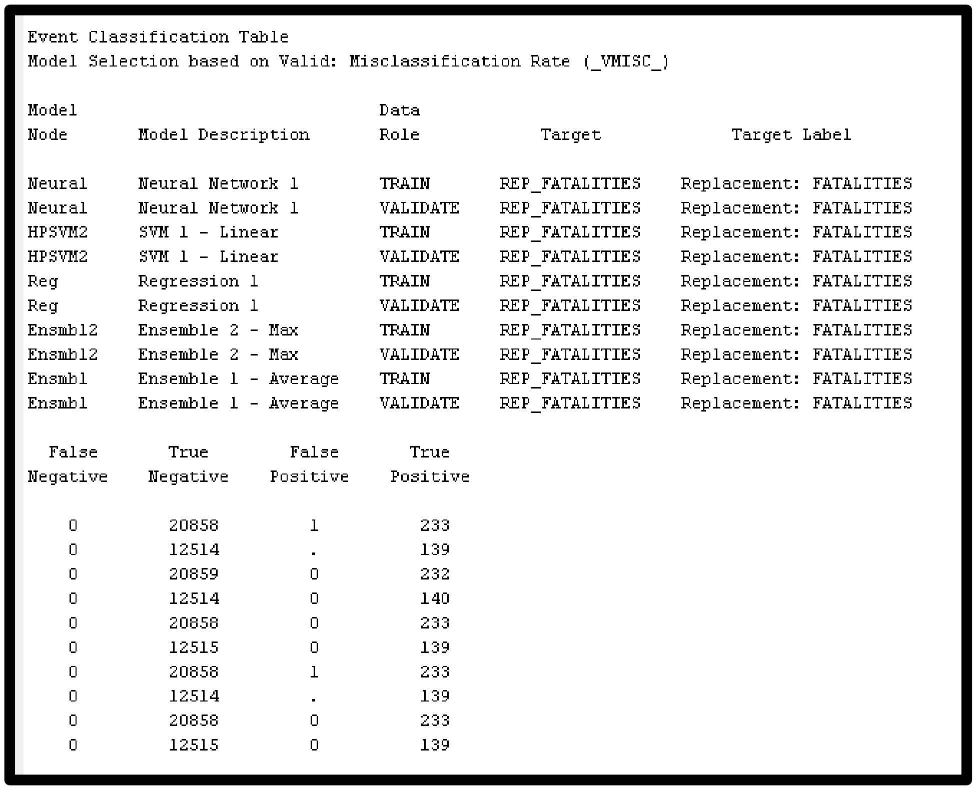


Figure 8. Classification Chart for First Set of Models.



Selected Model	Predecessor Node	Model Node	Model Description ▲	Target Variable	Target Label	Train: Misclassification Rate	Selection Criterion: Valid: Misclassification Rate	Test: Misclassification Rate
	Ensmbl	Ensmbl	Ensemble 1 - Average	REP_FATA...	Replaceme...	0	0	0
	Ensmbl2	Ensmbl2	Ensemble 2 - Max	REP_FATA...	Replaceme...	4.741E-5	0	.0001185
Y	Neural	Neural	Neural Network 1	REP_FATA...	Replaceme...	4.741E-5	0	.0001185
	Reg	Reg	Regression 1	REP_FATA...	Replaceme...	0	0	0
	HPSVM2	HPSVM2	SVM 1 - Linear	REP_FATA...	Replaceme...	0	0	0

Figure 9. Fit Statistics for First Set of Models.



```
Event Classification Table
Model Selection based on Valid: Misclassification Rate (_VMISC_)

Model                               Data
Node       Model Description        Role          Target             Target Label

Neural     Neural Network 1         TRAIN      REP_FATALITIES    Replacement: FATALITIES
Neural     Neural Network 1         VALIDATE   REP_FATALITIES    Replacement: FATALITIES
HPSVM2     SVM 1 - Linear           TRAIN      REP_FATALITIES    Replacement: FATALITIES
HPSVM2     SVM 1 - Linear           VALIDATE   REP_FATALITIES    Replacement: FATALITIES
Reg        Regression 1             TRAIN      REP_FATALITIES    Replacement: FATALITIES
Reg        Regression 1             VALIDATE   REP_FATALITIES    Replacement: FATALITIES
Ensmbl2    Ensemble 2 - Max         TRAIN      REP_FATALITIES    Replacement: FATALITIES
Ensmbl2    Ensemble 2 - Max         VALIDATE   REP_FATALITIES    Replacement: FATALITIES
Ensmbl     Ensemble 1 - Average     TRAIN      REP_FATALITIES    Replacement: FATALITIES
Ensmbl     Ensemble 1 - Average     VALIDATE   REP_FATALITIES    Replacement: FATALITIES

  False        True       False       True
Negative    Negative    Positive    Positive

    0        20858         1         233
    0        12514         .         139
    0        20859         0         232
    0        12514         0         140
    0        20858         0         233
    0        12515         0         139
    0        20858         1         233
    0        12514         .         139
    0        20858         0         233
    0        12515         0         139
```

Figure 10. Event Classification Table for First Set of Models.

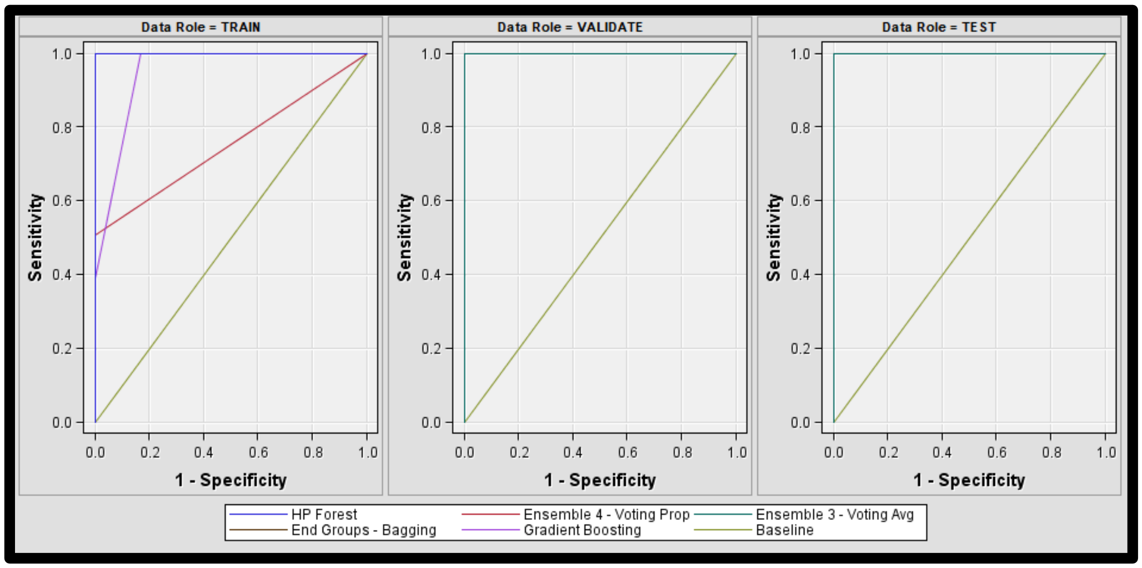


Figure 11. ROC Curve for Second Set of Models.

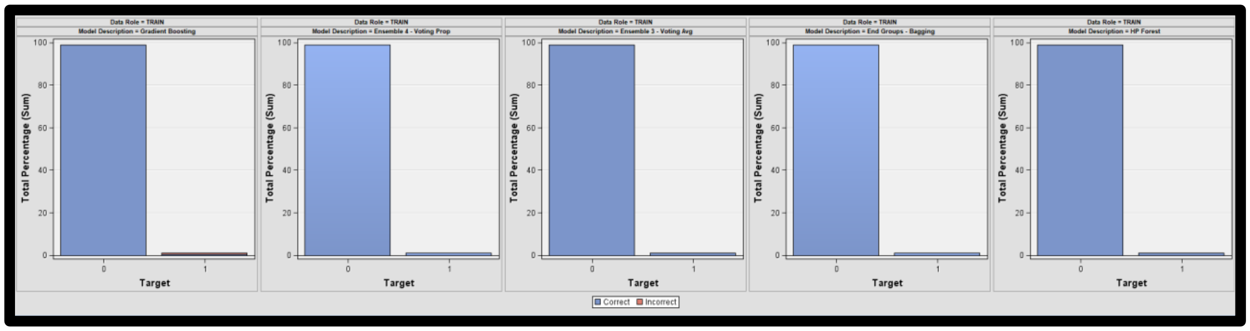
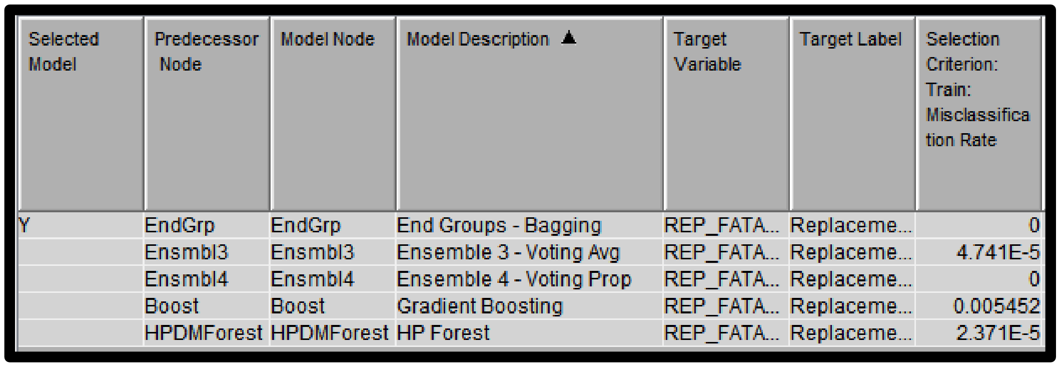
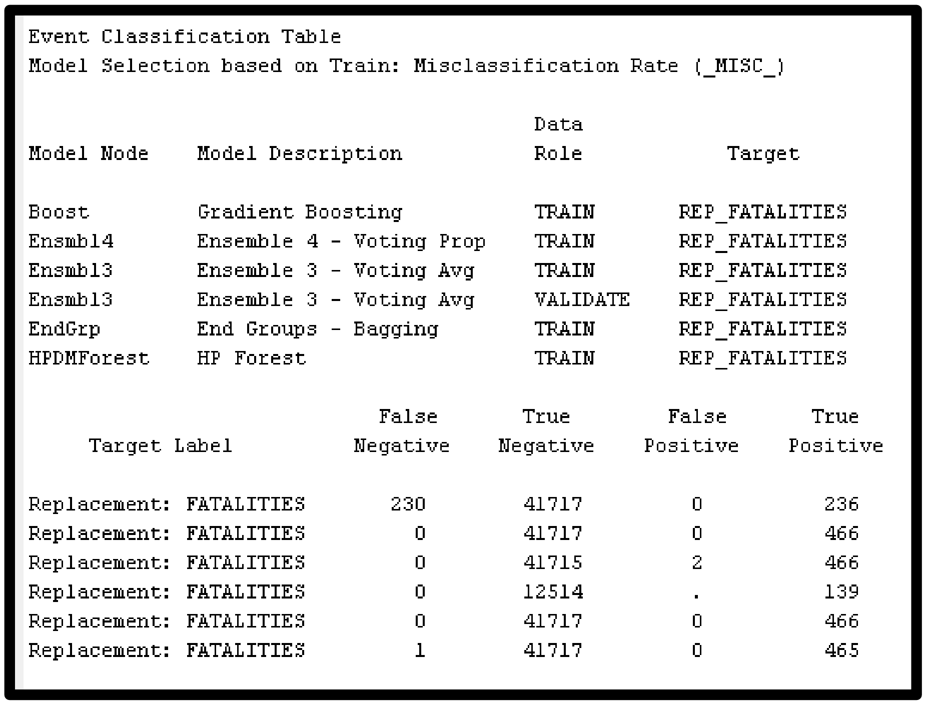


Figure 12. Classification Chart for Second Set of Models.



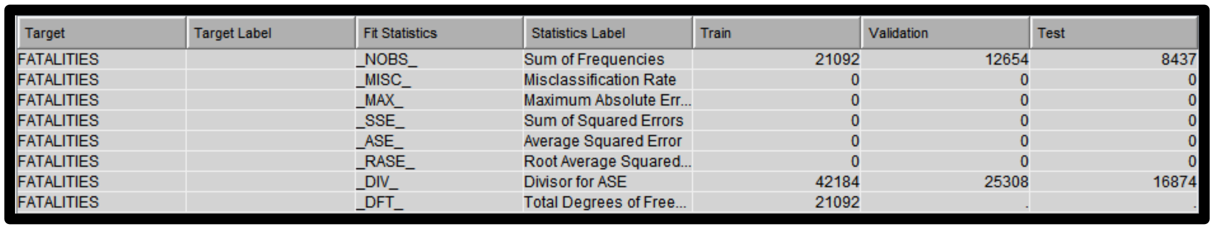
Selected Model	Predecessor Node	Model Node	Model Description ▲	Target Variable	Target Label	Selection Criterion: Train: Misclassification Rate
Y	EndGrp	EndGrp	End Groups - Bagging	REP_FATA...	Replaceme...	0
	Ensmbl3	Ensmbl3	Ensemble 3 - Voting Avg	REP_FATA...	Replaceme...	4.741E-5
	Ensmbl4	Ensmbl4	Ensemble 4 - Voting Prop	REP_FATA...	Replaceme...	0
	Boost	Boost	Gradient Boosting	REP_FATA...	Replaceme...	0.005452
	HPDMForest	HPDMForest	HP Forest	REP_FATA...	Replaceme...	2.371E-5

Figure 13. Fit Statistics for Second Set of Models.



```
Event Classification Table
Model Selection based on Train: Misclassification Rate (_MISC_)

                                              Data
Model Node    Model Description               Role            Target

Boost         Gradient Boosting               TRAIN       REP_FATALITIES
Ensmbl4       Ensemble 4 - Voting Prop        TRAIN       REP_FATALITIES
Ensmbl3       Ensemble 3 - Voting Avg         TRAIN       REP_FATALITIES
Ensmbl3       Ensemble 3 - Voting Avg         VALIDATE    REP_FATALITIES
EndGrp        End Groups - Bagging            TRAIN       REP_FATALITIES
HPDMForest    HP Forest                       TRAIN       REP_FATALITIES

                                 False       True        False       True
     Target Label               Negative    Negative    Positive    Positive

Replacement: FATALITIES            230       41717          0          236
Replacement: FATALITIES              0       41717          0          466
Replacement: FATALITIES              0       41715          2          466
Replacement: FATALITIES              0       12514          .          139
Replacement: FATALITIES              0       41717          0          466
Replacement: FATALITIES              1       41717          0          465
```

Figure 14. Event Classification Table for Second Set of Models.



Target	Target Label	Fit Statistics	Statistics Label	Train	Validation	Test
FATALITIES		_NOBS_	Sum of Frequencies	21092	12654	8437
FATALITIES		_MISC_	Misclassification Rate	0	0	0
FATALITIES		_MAX_	Maximum Absolute Err...	0	0	0
FATALITIES		_SSE_	Sum of Squared Errors	0	0	0
FATALITIES		_ASE_	Average Squared Error	0	0	0
FATALITIES		_RASE_	Root Average Squared...	0	0	0
FATALITIES		_DIV_	Divisor for ASE	42184	25308	16874
FATALITIES		_DFT_	Total Degrees of Free...	21092	.	.

Figure 15. Fit Statistics for Original Decision Tree Model.

Model	**Misclassification**	**False Negatives**	**False Positives**
Logistic Regression	0	0	0
SVM: Linear Kernel	0	0	0
Neural Network	4.741×10^{-5}	0	1
Decision Tree	0	0	0
Ensemble: Average	0	0	0
Ensemble: Maximum	4.741×10^{-5}	0	1
Bagging	0	0	0
Gradient Boosting	0.005452	230	0
Random Forest	2.371×10^{-5}	1	0
Ensemble: Voting Average	4.741×10^{-5}	0	2
Ensemble: Voting Proportion	0	0	0

Figure 16. Model Comparison Table.