Title Page

Assignment 2: Ensemble I Models using SAS Enterprise Miner

Melissa Hunfalvay

Email: Melissa.Hunfalvay@gmail.com

Data 640 9040

Spring 2022

Professor Steve Knode

Date: February 13th, 2022

Introduction

The dataset chosen was "car lemon dataset.csv" (Figure 1). The data included various features of cars (such as color and make) and whether the cars were determined to be good or bad purchases. The dataset included 34 variables and 72,983 rows or observations. The type of variables included numeric (n = 15) and character (n = 19) variables. Of the numeric variables, 11 were interval, the other two were binary.

The purpose of the analysis was to identify the bad vehicle purchases, from good vehicle purchases. Ensemble models are "models of models" based on the concept that large numbers of models are more accurate than any one model (Surowiecki, 2005).

"Is Bad Buy" was the target binary variable ("0" = not a bad buy, "1" = bad buy). The target variable was unevenly proportioned and skewed heavily in favor of not bad purchases (n = 64,007; 87.70%). Bad purchases totaled 8,976 of the 72,983 observations (12.30%). The imbalance percentage is 75.4%.

Data Cleaning and Preparation

Data cleaning and preparation tasks are outlined in Table 1 for Ensemble models and are compared with SVM models. Due to the number of decision trees being built, and the ability for decision trees to mitigate outlier influence, several cleaning steps were not necessary and therefore were eliminated in this assignment (Hastie, Tibshirani, Friedman, 2009; Lindoff & Berry, 2011).

The Condorcet Jury Theorem requires certain conditions be present for the theory to hold true such as diversity of opinions or input parameters (Rokach, 2012. Therefore, a correlation matrix was run to determine if input variables were "unique" (Figure 2). MMR Acquisition Auction "Clean" versus "Average variables were highly correlated (that is above .80, Hastie, Tibshirani, Friedman, 2009). Therefore, all the "clean" variables were set to rejected".

Table 1: Summary of data cleaning changes from SVM to Ensemble 1 models

Cleaning Task	Support Vector Machines	Ensemble 1 models
Accept SAS rejected variables?	Yes	Yes
Reject ID variables?	Yes	Yes
Impute missing values?	Yes, default methods used	No, not needed for decision tree models (Lindoff & Berry, 2011)
Adjust outliers?	Yes, 3 standard deviations from the mean	No, not needed for decision tree models (Hastie, Tibshirani, Friedman, 2009)
Transform skewness?	Yes, Warrenty Cost Log 10 transformation	No, not needed for decision tree models (Lindoff & Berry, 2011)
Correlated variables?	Yes, remove vehicle year	Yes, remove vehicle year
Correlation matrix?	No	Yes, as ensemble methods require independence of input (Rokach, 2012).

Predictive Model Development

Predictive models were developed first by varying parameters within each type of model (e.g. bagging or boosting, see Table 2, 3, 4, 5 & Figures 3, 4, 5, 6). Then, cut off thresholds for imbalanced targets were added to each type of model as a final parameter change. Total cost of models was also considered in this step. A "best in class" model was chosen that represented the best set of parameters for a specific model type based on the analytics goal of identifying rare, bad buys. In other words, the strongest HP Forest model was chosen, the strongest Gradient boosting model etc. (Figure 7). Finally, the four strongest models were compared with one another using the Model Comparison node. This determined the "Champion model". A total of 26 models were developed.

For each model there was no imputation or transformation of data. Skewness was not adjusted. Selection criteria for each model was set to Validation: Misclassification rate. Data was partitioned 70% training, 30% validation for each model.

Table 2: Predictive Models Developed and Parameters Outlined for Bagging (Figure 3)

Model Type	Explanation	Algorithm	Models Developed: Parameters varied		Progressive Reasoning/Decision Making
Bagging	Bagging (*B*ootstrap *Agg*regation) builds decision trees by resampling method which independently sampling replacements from an existing sample data with the same sample size n.	Steps for boostrapping:	Bag 1	Default settings. Index count 10, percentage of data used in each tree 10%. Leaf signficance level 0.2.	Establish a baseline from default settings
		θ = population	Bag 2	Index count 10, percentage of data used in each tree 20%. Leaf signficance level 0.2.	Double data used for each tree to determine if this improves results
		n = Sample size	Bag 3	Index count 100, percentage of data used in each tree 20%. Leaf signficance level 0.2.	Abbott (2014) when target is categorical build more models.
		B = replacement times i.e. bootstrap sample	Bag 4	Index count 5, percentage of data used in each tree 20%. Leaf signficance level 0.2.	Decrease the trees created to 5 to see how this changes results.
		Construct sampling distribution and estimate error	Bag 5	Index count 10, percentage of data used in each tree 10%. Leaf signficance level 0.1	Change the significance level of the tree splits to see if this improved the misclassification rate.
		Yen (2019)	Bag 6	Index count 10, percentage of data used in each tree 10%. Leaf signficance level 0.1. Added cut-off node for unbalanced target variable to 0.13.	Change the cut off criterion to see if the adjustment of the imbalanced target improved the misclassification rate.

Table 3: Predictive Models Developed and Parameters Outlined for Boosting (Figure 4)

		$$p_i = \frac{1 + m_i^4}{\sum(1 + m_i^4)}$$	Boost1	Default settings. Index count 10. This refers to the number of iterations changing weights each time. Percentage of data used in each tree 10%.	Establish a baseline from default settings
			Boost 2	Index count 20, percentage of data used in each tree 10%	Double the iteration to see if misclassification rate is improved.
Boosting	Boosting (Adaboost) builds decision trees by changing the weights of the difficult-to-classify correctly cases, increasing the importance of misclassified data.	For the *i* th case, the srcx4 weights are calculated by the formula above, whereby	Boost 3	Index count 20, percentage of data used in each tree 10%.	Change the splitting rule parameters to see if a more restrictive split helps with classification
		where 0<=m*i* <=k is the number of time that the *i* th case is misclassified in the preceding steps.		Change defaults on the splitting rule to nominal variables = GINI index. Changed significance level for training from default of 0.2 to 0.13	
			Boost 4	Index count 5, percentage of data used in each tree 10%.	Reduce the tree iterations to see how this affects the results. Assuming it reduces
			Boost5	Tried custom cutoff threshold of 0.13 from the default of 0.5	Change the cut off criterion to see if the adjustment of the imbalanced target improved the misclassification rate.

Table 4: Predictive Models Developed and Parameters Outlined for Gradient Boosting (Figure 5)

			Gradient1	Default settings. Number of iterations = 50	Establish a baseline from default settings
		Gradient boosting performs the same as boosting except	Gradient2	Number of iteration = 25	Reduced the number if iterations in the default settings
	Gradient boosting is another form of boosting whereby decision trees are created and hard to classify cases are given more weight, however, with gradient boosting the target variable for the subsequent trees consists of the residual from the previous tree.	it uses gradients in the loss function.	Gradient3	Number of iterations = 100	Increased iterations systematically to see where the number of iterations provided the best result
Gradient Boosting		y=ax+b+e	Gradient4	Number of iterations = 200	Increased iterations
		e is the error term	Gradient5	Number of iterations = 500	Increased iterations
			Gradient6	Number of iterations = 1000	Increased iterations
		Singh (2018)	Gradient7	Number of iterations = 1000	
				Changed splitting rule from maximum depth of 2 (the default) to 5.	Change the splitting rule and depth to see if any further improvements could be made.
			Gradient8	Number of iterations = 1000	
				Added cutoff node. Changed default of 0.5 to 0.13	Change the cut off criterion to see if the adjustment of the imbalanced target improved the misclassification rate.

Table 5: Predictive Models Developed and Parameters Outlined for Random Forest (Figure 6)

			Forest1	Default settings. Number of trees built (i.e. index count) = 100	Establish a baseline from default settings
	Random forest builds decision trees whereby each iteration of the decision trees is developed using only a random subset of the possible inputs.	When performing random forest on classification data the formula is based on the GINI index	Forest2	Number of trees built = 200	Per Knode (2022) recommended building more trees.
Random Forest		$$Gini = 1 - \sum_{i=1}^{C} (p_i)^2$$	Forest3	Number of trees built = 200.	
				Changed the significance level for the training data to 0.10, from the default which is 0.05	Loosened the signifcance restrictions to determine if this improved misclassification rate.
		Th eformula uses the class probability to determine the Gini in each branch	Forest4	Number of trees built = 500.	Knode (2022) recommended building more trees.
		pi = relative frequency		Kept significance level at 0.1	
		c = number of classes	Forest5	Number of trees built = 200.	Changed the count (or number) of observations rather than proportion to determine the training sample
				Changed type of sampling from proportional to count.	(SAS Enterprise Miner Forest Node, n.d.)
		Schott (2019)	Forest6	Number of trees built = 200	To check the impact of overfitting reduced the mininum leaf size.
				Changed type of sample back to proportional	Specifies what percentage of observations is used for each tree when the Type of
				Changed default value on proportion of Obs sample from 0.6 to 0.4	Sample property is set to Proportion (SAS Enterprise Miner Forest Node, n.d.)
			Forest7	Number of trees built = 200	
				Added cutoff node. Changed default of 0.5 to 0.13	Change the cut off criterion to see if the adjustment of the imbalanced target improved the misclassification rate.

These models were also compared to the “Champion model” and “Runner-Up” Champion from Support Vector achines (SVM; see Table 1, Appendix).

Table 6: Winner and Runner-Up Support Vector Machine Models

Kernel	Explanation	Algorithm	Models Developed: Parameters varied	
Radial Based	A spherical (circular) function	K(u,v) = exp[-p (u - v)2]		70/30 Data Partition
Function	where any of the line segments	*exp* is the exponential function	9	Cutoff threshold 0.24 (custom – best fit)
(RBF)	from a central point to the	K = Kernel function.		RBF Degree: 3
	perimeter (Abbey, He &Wang, n.d.).	T is the transpose of vector u.		70/30 Data Partition
		u and v are vectors in the input space.	10	Cutoff threshold 0.47 (custom – best fit)
				RBF Degree: 1

Imbalanced Target

The binary target variable "*Is Bad Buy*" was imbalanced. A cutoff criterion of 0.13, which was reflective of the proportion of rare events in the dataset, was implemented using the Cutoff Node, in one of each type of model (see Tables 8-12). The cost function was also calculated to determine overall cost of each model. Proportions of cost for false negatives was set to $1 and false positives to $6.90. The True results cost was $0 (see Figure 8). In tables 8-12, both total cost for each model and "Cost Normalized" was calculated via percentages to compare models.

Accuracy Measures and Results

The purpose of this assessment was to identify rare (positive or "1") events, which are bad buys. Therefore, models were assessed in a "best in class" or best type of model using the following order of importance to achieve this goal. First, true positives, second, sensitivity, third, precision, fourth, F1 score, fifth, accuracy. All these measures were examined in both training and validation data to determine the ability of the model to generalize (see Tables 7-11).

In phase 2, when comparing the "best in-class" type of models to one another, several other factors were considered (see Table 12, Figure 9). First, Receiver Operating Curves (ROC), were used in the Model Comparison Node to visually compare the tradeoff between sensitivity and specificity. Second, cumulative lift, which indicates the models ability to predict beyond chance (Figure 10). Third, the cost of the model. Fourth, the misclassification rate.

Table 8: Phase 1: Model Comparisons Accuracy Measures for Bagging

Model Name & Number	Number of Trees Built	Cutoff Criterion	Significance Level of Splitting Criteria	Maximum Tree Depth	Unique Model Adjustments	Training/Validation	FN(%)	FN(#)	TN(%)	TN(#)	FP(%)	FP(#)	TP(%)	TP(#)	Sensitivity %	Accuracy %	Precision %	Specificity %	F1 Score	Training & Validation Accuracy	Total Cost	Cost Normalized as a %	Notes	Conclusion
Bag1	10	0.5	0.2	6	None	Training	9.45	4824	87.02	44461	5.46	343	2.85	1458	23.19	89.88	80.96	99.23	0.36	0.20	$33,656	$65	Highest TP rate of bagging models. Sensitivity poor.	Reject
						Validation	9.33	2044	87.11	19074	0.59	129	2.97	650	24.13	90.08	83.44	99.33	0.37		$14,233	$65	Precision high. No overfitting	
Bag2	10	0.5	0.2	6	20% data per tree	Training	9.62	4916	87.14	44516	0.56	288	2.67	1366	21.74	89.81	82.59	99.36	0.34	0.23	$34,208	$66	Increasing data per tree didn't change anything sig-	Reject
						Validation	9.49	2077	87.22	19099	0.47	104	2.82	617	22.90	90.04	85.58	99.46	0.36		$14,435	$66	nificantly in the model. No overfitting.	
Bag3	100	0.5	0.2	6	20% data per tree	Training	9.53	4871	87.11	44500	0.60	304	2.76	1411	22.46	89.87	82.27	99.32	0.35	0.19	$33,914	$66	Adding more trees did not change anything sig-	Reject
						Validation	9.42	2063	87.18	19089	0.52	114	2.88	631	23.42	90.06	84.70	99.41	0.37		$14,349	$66	nificantly in the model. No overfitting.	
Bag4	5	0.5	0.2	6	20% data per tree	Training	9.66	4935	87.15	44521	0.55	283	2.64	1347	21.44	89.79	82.64	99.37	0.34	0.24	$34,335	$66	Highest FN rate, leading to highest cost.	Reject
						Validation	9.51	2082	87.24	19102	0.46	101	2.79	612	22.72	90.03	85.83	99.47	0.36		$14,467	$66	No overfitting	
Bag5	10	0.5	0.1	6	None	Training	9.44	4824	87.03	44461	0.67	343	2.85	1458	23.21	89.89	80.96	99.23	0.36	0.19	$33,629	$65	Highest TP , sensitivity (though still poor), precision,	Accept, Phase 2
						Validation	9.33	2044	87.11	19074	0.59	129	2.97	650	24.13	90.08	83.44	99.33	0.37		$14,233	$65	accuracy. No overfitting.	
Bag 6	10	0.13	0.1	6	None	Training	9.44	4824	87.03	44461	0.67	343	2.85	1458	23.21	89.89	80.96	99.23	0.36	0.19	$33,629	$65	Adding cut off node made no difference to the scores	Reject
						Validation	9.33	2044	87.11	19074	0.59	129	2.97	650	24.13	90.08	83.44	99.33	0.37		$14,233	$65	These models appear the same.	

Table 9: Phase 1: Model Comparisons Accuracy Measures for Boosting

Model Name & Number	Number of Trees Built	Cutoff Criterion	Significance Level of Splitting Criteria	Maximum Tree Depth	Unique Model Adjustments	Training/Validation	FN(%)	FN(#)	TN(%)	TN(#)	FP(%)	FP(#)	TP(%)	TP(#)	Sensitivity %	Accuracy %	Precision %	Specificity %	F1 Score	Delta in Training & Validation Accuracy	Total Cost	Cost Normalized as a %	Notes	Conclusion
Boost1	10	0.5	0.2	6	None	Training	12.30	6282	87.70	44804	0.00	0	0.00	0	0.00	87.70	0.00	100.00	0.00	0.01	$43,346	$85	Model could no pick up ANY rare events. No sensitivity.	Reject
						Validation	12.30	2694	87.70	19203	0.00	0	0.00	0	0.00	87.70	0.00	100.00	0.00		$18,589	$85	No Overfitting	
Boost2	20	0.5	0.2	6	None	Training	7.33	3747	58.93	30104	28.78	14700	4.96	2535	40.35	63.89	14.71	67.19	0.22	0.54	$40,554	$81	TP & sensitivity & accuracy moderate. Precision low.	Reject
						Validation	7.55	1653	58.60	12831	29.10	6372	4.75	1041	38.64	63.35	14.04	66.82	0.21		$17,778	$81	No Overfitting. Costly.	
Boost3	20	0.5	0.13	6	Splitting rule: nominal variables = GINI index	Training	5.11	2611	46.16	23579	41.55	21225	7.19	3671	58.44	53.34	14.75	52.63	0.24	0.69	$39,241	$78	Best TP & sensitivity, though still only moderate-to-adequate.	Accept, Phase 2.
						Validation	5.26	1151	45.61	9987	42.09	9216	7.05	1543	57.28	52.66	14.34	52.01	0.23		$17,158	$78	Precision low. Accuracy modeerate. No Overfitting	
boost 4	5	0.5	0.2	6	None	Training	7.78	3977	33.02	16869	54.68	27935	4.51	2305	36.69	37.53	7.62	37.65	0.13	0.22	$55,376	$109	Very costly due to very high FP.	Reject
						Validation	7.91	1733	33.36	7305	54.34	11898	4.39	961	35.67	37.75	7.47	38.04	0.12		$23,856	$109	No Overfitting	
Boost 5		N/A*				Training	* No cutoff node able to be used because all probabilities are constant.																	
						Validation																		

Table 10: Phase 1: Model Comparisons Accuracy Measures for Gradient Boosting

Model Name & Number	Number of Trees Built	Cutoff Criterion	Significance Level of Splitting Criteria	Maximum Tree Depth	Unique Model Adjustments	Training/Validation	FN(%)	FN(#)	TN(%)	TN(#)	FP(%)	FP(#)	TP(%)	TP(#)	Sensitivity %	Accuracy %	Precision %	Specificity %	F1 Score	Training & Validation Accuracy	Total Cost	Cost Normalized as a %	Notes	Conclusion
Gradient1	50	0.5	N/A	2	None	Training	9.62	4913	87.15	44520	0.56	284	2.68	1369	21.79	89.83	82.82	99.37	0.35	0.24	$34,184	$66	TP low, sensitivity low. Accuracy & precision high.	Reject
						Validation	9.47	2074	87.24	19102	0.46	101	2.83	620	23.01	90.07	85.99	99.47	0.36		$14,412	$66	No Overfitting	
Gradient2	25	0.5	N/A	2	None	Training	9.66	4935	87.15	44521	0.55	283	2.64	1347	21.44	89.79	82.64	99.37	0.34	0.24	$34,335	$66	TP low, sensitivity low. Accuracy & precision high.	Reject
						Validation	9.51	2082	87.24	19102	0.46	101	2.79	612	22.72	90.03	85.83	99.47	0.36		$14,467	$66	Decreasing trees did not change results much. No Overfitting	
Gradient 3	100	0.5	N/A	2	None	Training	9.52	4863	87.12	44505	0.59	299	2.78	1419	22.59	89.90	82.60	99.33	0.35	0.19	$33,854	$65	TP low, sensitivity low. Accuracy & precision high.	Reject
						Validation	9.40	2059	87.19	19092	0.51	111	2.90	635	23.57	90.09	85.12	99.42	0.37		$14,318	$65	Increasing trees did not change results much -$1. No Overfitting	
Gradient 4	200	0.5	N/A	2	None	Training	9.44	4825	87.08	44484	0.63	320	2.85	1457	23.19	89.93	81.99	99.29	0.36	0.18	$33,613	$65	TP low, sensitivity low. Accuracy & precision high. Furth	Reject
						Validation	9.35	2048	87.16	19085	0.54	118	2.95	646	23.98	90.11	84.55	99.39	0.37		$14,249	$65	increasing trees did not change results much. No Overfitting	
Gradient 5	500	0.5	N/A	2	None	Training	9.37	4787	87.07	44483	0.63	321	2.93	1495	23.80	90.00	82.32	99.28	0.37	0.13	$33,351	$65	Very little change when 2.5 x trees built from prior model	Reject
						Validation	9.30	2037	87.14	19080	0.56	123	3.00	657	24.39	90.14	84.23	99.36	0.38		$14,178	$65	No Overfitting	
Gradient 6	1000	0.5	N/A	2	None	Training	9.30	4753	87.07	44479	0.64	325	2.99	1529	24.34	90.06	82.47	99.27	0.38	0.08	$33,121	$65	Very little change when 2 x trees built from prior model	Reject
						Validation	9.27	2029	87.10	19073	0.59	130	3.04	665	24.68	90.14	83.65	99.32	0.38		$14,130	$65	No Overfitting	
Gradient 7	1000	0.5	N/A	5	None	Training	9.19	4697	87.13	44511	0.57	293	3.10	1585	25.23	90.23	84.40	99.35	0.39	0.08	$32,702	$64	Still low TP, sensitivity. High accuracy & precision. -$1 c	Accept, phase 2.
						Validation	9.26	2027	87.10	19073	0.59	130	3.05	667	24.76	90.15	83.69	99.32	0.38		$14,116	$64	No Overfitting	
Gradient 8	1000	0.13	N/A	2	None	Training	9.19	4697	87.13	44511	0.57	293	3.10	1585	25.23	90.23	84.40	99.35	0.39	0.08	$32,702	$64	Change in cutoff threshold made no change to results from	Reject
						Validation	9.26	2027	87.10	19073	0.59	130	3.05	667	24.76	90.15	83.69	99.32	0.38		$14,116	$64	Gradient 7 model to this model. No Overfitting	

Table 11: Phase 1: Model Comparisons Accuracy Measures for Random Forests

Model Name & Number	Number of Trees Built	Cutoff Criterion	Significance Level of Splitting Criteria	Maximum Tree Depth	Unique Model Adjustments	Training/Validation	FN(%)	FN(#)	TN(%)	TN(#)	FP(%)	FP(#)	TP(%)	TP(#)	Sensitivity %	Accuracy %	Precision %	Specificity %	F1 Score	Training & Validation Accuracy	Total Cost	Cost Normalized as a %	Notes	Conclusion
Forest1	100	0.5	0.05	50	None	Training	9.49	4848	87.13	44509	0.58	295	2.81	1434	22.83	89.92	82.94	99.34	0.36	0.07	$33,746	$65	Highest cost of Forest models.	Reject
						Validation	9.40	2058	87.09	19070	0.61	133	2.90	636	23.61	89.99	82.70	99.31	0.37		$14,333	$65	Sensitivity poor. No Overfitting	
Forest2	200	0.5	0.05	50	None	Training	9.44	4823	87.07	44483	0.63	321	2.86	1459	23.23	89.93	81.97	99.28	0.36	0.09	$33,600	$65	Doubling trees built did not improve results markedly.	Reject
						Validation	9.37	2051	87.08	19069	0.61	134	2.94	643	23.87	90.02	82.75	99.30	0.37		$14,286	$65	Sensitivity poor. No Overfitting	
Forest3	200	0.5	0.1	50	None	Training	9.43	4816	87.09	44489	0.62	315	2.87	1466	23.34	89.96	82.31	99.30	0.36	0.09	$33,545	$65	Change in significance level did not change results marked	Reject
						Validation	9.35	2048	87.10	19072	0.60	131	2.95	646	23.98	90.05	83.14	99.32	0.37		$14,262	$65	Sensitivity poor. No Overfitting	
Forest 4	500	0.5	0.1	50	None	Training	9.43	4819	87.09	44489	0.62	315	2.86	1463	23.29	89.95	82.28	99.30	0.36	0.09	$33,566	$65	Most number of trees built didn't improve the identificatio	Reject
						Validation	9.36	2049	87.10	19072	0.60	131	2.95	645	23.94	90.04	83.12	99.32	0.37		$14,269	$65	of rare events. Sensitivity poor. No Overfitting	
Forest 5	200	0.5	0.05	50	Sample from proportional to count	Training	9.43	4816	87.09	44489	0.62	315	2.87	1466	23.34	89.96	82.31	99.30	0.36	0.09	$33,545	$65	Changing sample type to count did not improve the result:	Reject
						Validation	9.35	2048	87.10	19072	0.60	131	2.95	646	23.98	90.05	83.14	99.32	0.37		$14,262	$65	Sensitivity poor. No Overfitting	
Forest 6	200	0.5	0.05	50	Proportion of obs from 0.6 to 0.4	Training	9.41	4809	87.04	44464	0.67	340	2.88	1473	23.45	89.92	81.25	99.24	0.36	0.13	$33,522	$65	Highest TP. Low sensitivity. Precision & Accuracy high.	Accept, Phase 2.
						Validation	9.34	2045	87.09	19070	0.61	133	2.96	649	24.09	90.05	82.99	99.31	0.37		$14,244	$65	Sensitivity poor. No Overfitting	
Forest 7	200	0.13	0.05	50	None	Training	9.44	4823	87.05	44471	0.65	333	2.86	1459	23.23	89.91	81.42	99.26	0.36	0.11	$33,612	$65	Change in cut off threshold to represent target imbalance d	Reject
						Validation	9.36	2049	87.08	19067	0.62	136	2.95	645	23.94	90.02	82.59	99.29	0.37		$14,274	$65	not improve results. Sensitivity poor. No Overfitting	

Table 12: Phase 1: Model Comparisons Accuracy Measures for SVM

Model Name & Number	Number of Trees Built	Cutoff Criterion	Significance Level of Splitting Criteria	Maximum Tree Depth	Unique Model Adjustments	Training Validation	FN(%)	FN(#)	TN(%)	TN(#)	FP(%)	FP(#)	TP(%)	TP(#)	Sensitivity %	Accuracy %	Precision %	Specificity %	F1 Score	Training & Validation Accuracy	Total Cost	Cost Normalized	Notes	Conclusion
RBF (SVM Model 9)	N/A	0.25	N/A	N/A	None	Training	0.17	24	22.88	3155	64.83	8941	12.12	1672	98.58	34.99	98.58	26.08	0.27	1.34	$34,053	$64	Good identification of True positivies. False negatives low. High sensitivity. High .	Champion SVM
						Validation	0.82	113	95.64	3776	1.44	66	13.44	614	84.00	33.65	84.46	98.28	0.87		$14,262	$64	precision. Low-moderate accuracy. Second lowest cost. Better specificity compareed with other models in phase 2 (except #10)	
(SVM Model 10)	N/A	0.47	N/A	N/A	None	Training	7.00	966	87.62	12084	87.62	12	5.29	730	43.04	91.39	43.04	99.90	0.60	1.50	$29,773	$67	Not enough identification of TP leading to lowest sensitivity in phase 2. Good Accuracy & precision.	Runner-Up SVM
						Validation	4.21	580	29.73	3816	83.52	26	3.22	147	20.00	89.89	20.22	99.32	0.33		$14,269	$67	Best F1 score. Lowest overall cost. Highest in overfitting of all the models, but still acceptable range.	

Table 13: Model Comparison Results and Insights

	Training			Validation		
Name in Diagram	ROC Index	Misclassification Rate	Cumulative Lift	ROC Index	Misclassification Rate	Cumulative Lift
RBF	0.92	0.09	6.67	0.69	0.11	3.67
RBF 47	0.96	0.11	7.14	0.69	0.10	3.55
Bag5	0.7	0.10	3.17	0.71	0.10	1.2
Boost3	0.74	0.47	3.82	0.71	0.47	3.62
Gradient7	0.81	0.10	4.4	0.77	0.10	3.81
Forest6	0.77	0.10	3.88	0.76	0.10	3.74

Conclusion and Takeaways

The goal of this analysis was to determine the bad buys (true positives) within the car dataset. None of the Ensemble models beat the SVM model 9 which identified 12.12% of a total of 12.30%, true positives, with very high sensitivity (98.58%) and precision (98.58%). Therefore, thus far the SVM model 9 is still the champion model. However, this model is overfitting as the ROC Index drops 23% from training to validation (see Figure 7). Therefore, a caveat to this analysis is the generalizability of this model.

The "champion" model from the ensemble models was the gradient boosting model (Boost3). This model had the highest true positives (7.19% of the 12.30%), with a moderate sensitivity (58.44%) and accuracy (53.34%), but low precision (14.75%). Boost3 did not overfit as both accuracy and ROC numbers remained constant between training and validation. However, the cost of this model was high ($78), especially when compared with SVM Model 9 ($64) as the Boost3 model had some difficultly identifying the false positives. All "finalist" models produced significantly high lift rate (>2.00) except for Bag5 (Figure 8). Finalists also had low misclassification rates and moderate to high ROC Indexes (Table 13, Figure 10).

The greatest overall difference between the SVM models and the Ensemble models was the low true positive and sensitivity rates in the ensemble models. Other than the gradient boosting models, all other ensemble models had 2-3% true positives and 20-24% sensitivity. These metrics are the *most important* to effectively answer the question being asked of the data which is to identify the bad buys. Hence the reason SVM model 9 is still the overall "Champion" model.

It was also observed that the imbalanced target did not affect results of the ensemble models unlike the SVM models which required tweaking the cut off parameter. Some models, particularly bagging and HP Forest (Figure 9), did not show variation in results as parameters were vastly different. However, boosting models varied significantly and were by far the costliest. This highlights a *limitation* of these models in terms of their complexity. The complexity of the model generation makes it difficult to understand why some models did not change at all with very different inputs and others changed considerably. In summation, I would still recommend the car sales company use SVM model 9 to identify the bad car sales, helping to improve inventory which will lead to happier customers.

References

Abbott, D. (2014). *Applied Predictive Analytics: Principles and Techniques for the Professional Data Analyst.* Indianapolis, IN: Wiley Publishing.

Adjorlolo, S. (2018). Diagnostic accuracy, sensitivity, and specificity of executive function tests in moderate traumatic brain injury in Ghana. *Assessment, 25,* 498–512. DOI: 10.1177/1073191116646445

Hastie, T.,Tibshirani, R., Friedman, J. (2009). *The Elements of Statistical Learning.* Springer, NY: New York.

Knode, S. (2022). Ensemble Model 2: Bagging, Boosting & Random Forests. Retrieved February 7th, 2022, from: https://learn.umgc.edu/d2l/le/content/627222/viewContent/25080884/View

Lindoff, G.S., & Berry, M.J.A. (2011). *Data Mining Techniques.* Wiley. Indianapolis, IN.

Rokach, L. (2012). *Ensemble Learning: The Wisdom of Crowds (of Machines).* Retrieved February 5th, 2022, from: https://www.slideshare.net/liorrokach/ensemble-learning-the-wisdom-of-crowds-of-machines

SAS Enterprise Miner (n.d.). *HP Forest Node.* Cary, NC: SAS Institute Inc. Retrieved February 3rd, 2022, from: https://documentation.sas.com/doc/en/emref/15.1/p1uhmtoprigyvkn147i1tw9e2ax0.htm

SAS Enterprise Miner (n.d.). *Ensemble Models and Portioning Algorithms in SAS Enterprise Miner.* Cary, NC: SAS Institute Inc. Retrieved February 5th, 2022, from: https://www.sas.com/apps/webnet/video-sharing.html?bcid=4363855671001

Schott, M. (2019). Random Forest Algorithms for Machine Leaning. Retrieved February 5th, 2022, from: https://medium.com/capital-one-tech/random-forest-algorithm-for-machine-learning-c4b2c8cc9feb

Singh, H. (2018). Understanding gradient boosting machines. *Towards Data Science.* Retrieved from: https://towardsdatascience.com/understanding-gradient-boosting-machines-9be756fe76ab#:~:text=While%20the%20AdaBoost%20model%20identifies,it%20is%20the%20error%20term

Surowiecki, J. (2005). *The Wisdom of Crowds.* Anchor Books. New York, NY.

Yen, L. (2019). *An Introduction to Bootstrap Method. Toward Data Science.* Retrieved February 5th, 2022, from: https://towardsdatascience.com/an-introduction-to-the-bootstrap-method-58bcb51b4d60

Appendix

Name	Label	Role	Level	Number of Levels	Percent Missing	Minimum	Maximum	Mean	Standard Deviation	Skewness
Transmission	Automatic, manual	Input	Nominal	3	0.00137	.	.	.	.	.
TopThreeAmericanName	Manufacturers	Input	Nominal	5	0	.	.	.	.	.
VehicleAge	Years	Input	Nominal	10	0	.	.	.	.	.
WarrantyCost	Zip code where bought	Input	Interval	.	0	462	7498	1276.581	598.8468	2.070831
VNZIP1	Color	Input	Interval	.	0	2764	99224	58043.06	26151.64	-0.10353
Color		Input	Nominal	17	0	.	.	.	.	.
Size	Size category e.g. SUV	Input	Nominal	13	0	.	.	.	.	.
Nationality	Manufacturer country	Input	Nominal	5	0	.	.	.	.	.
PRIMEUNIT	Demand status	Input	Nominal	3	0	.	.	.	.	.
WheelType	Alloy, covers	Input	Nominal	4	0	.	.	.	.	.
VehBCost	At acquisition	Input	Interval	.	0	1	45469	6730.934	1767.846	0.715931
VehOdo	Auction market price	Input	Interval	.	0	4825	115717	71500	14578.91	-0.45315
MMRAcquisitionAuctionAveragePric	Online purchase	Input	Interval	.	0.024663	0	35722	6128.909	2461.993	0.463641
IsOnlineSale	Retail market price	Input	Binary	2	0	.	.	.	.	.
MMRAcquisitionRetailAveragePrice	Auction provider	Input	Interval	.	0.024663	0	39080	8497.034	3156.285	0.209214
Auction	Guarantee	Input	Nominal	3	0	.	.	.	.	.
AUCGUART	Auction clean price	Input	Nominal	3	0	.	.	.	.	.
MMRAcquisitionAuctionCleanPrice	Retail clean price	Rejected	Interval	.	0.024663	0	36859	7373.636	2722.492	0.466501
MMRAcquisitonRetailCleanPrice	Auction clean price	Rejected	Interval	.	0.024663	0	41482	9850.928	3385.79	0.1763
MMRCurrentAuctionCleanPrice		Rejected	Nominal	21	0	.	.	.	.	.
BYRNO	Unique buyer ID	Rejected	Interval	.	0	835	99761	26345.84	25717.35	2.129225
WheelTypeID	Wheel Type ID	Rejected	Nominal	5	0	.	.	.	.	.
Make	Make of car	Rejected	Nominal	21	0	.	.	.	.	.
VehYear	Model of car	Rejected	Nominal	10	0	.	.	.	.	.
MMRCurrentRetailAveragePrice	Retail average price	Rejected	Nominal	21	0	.	.	.	.	.
MMRCurrentRetailCleanPrice	Retail clean price	Rejected	Nominal	21	0	.	.	.	.	.
MMRCurrentAuctionAveragePrice	Auction average price	Rejected	Nominal	21	0	.	.	.	.	.
Model	Car ID	Rejected	Nominal	21	0	.	.	.	.	.
RefId	Car submodel	Rejected	Interval	.	0	1	73014	36511.43	21077.24	-.000203
SubModel	State car purchased	Rejected	Nominal	21	0	.	.	.	.	.
VNST	Car trim level	Rejected	Nominal	21	0	.	.	.	.	.
Trim	Bay avoidable	Rejected	Nominal	20	4.367863	.	.	.	.	.
IsBadBuy	purchase	Target	Binary	2	0	.	.	.	.	.
PurchDate		Time ID	Interval	.	0	.	.	.	.	.

Figure 1: Variables after cleaning

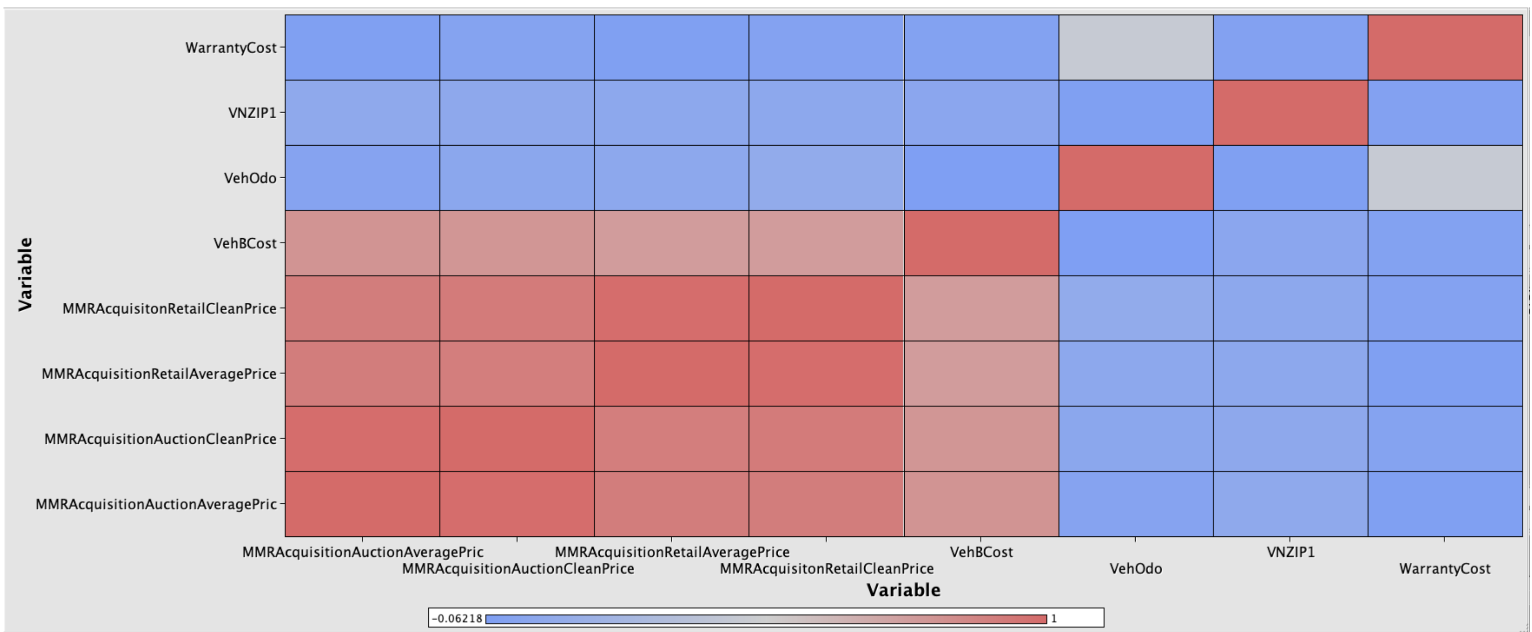

Figure 2: Correlation matrix

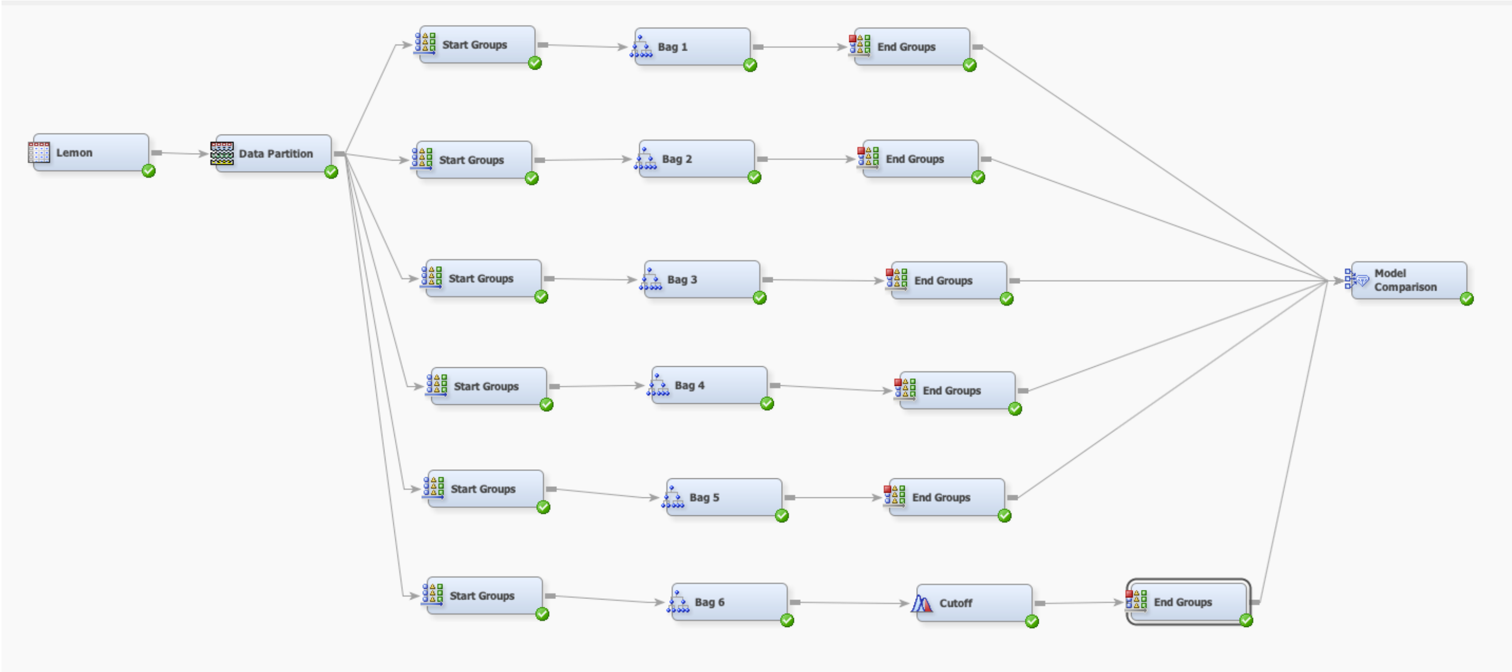

Figure 3: Phase 1 Model comparison for Bagging

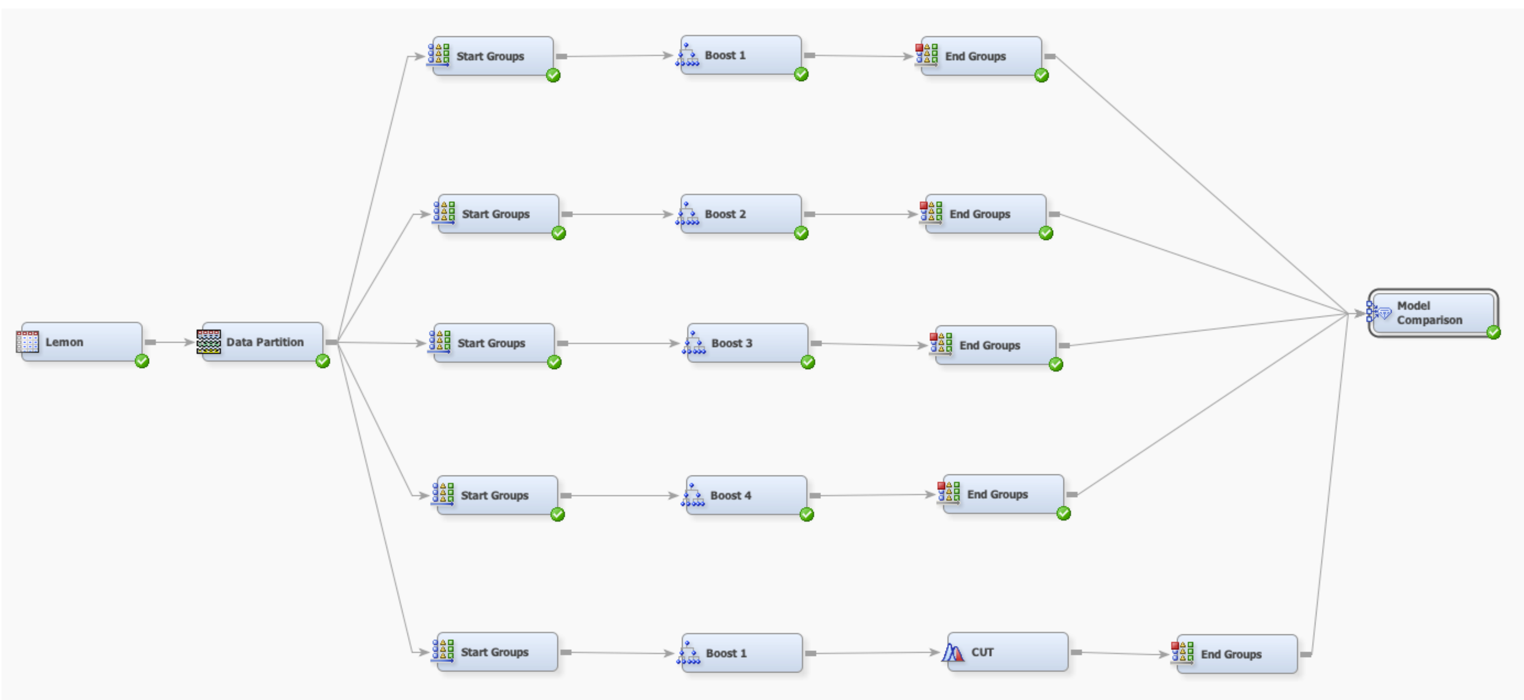

Figure 4: Phase 1 Model comparison for Boosting

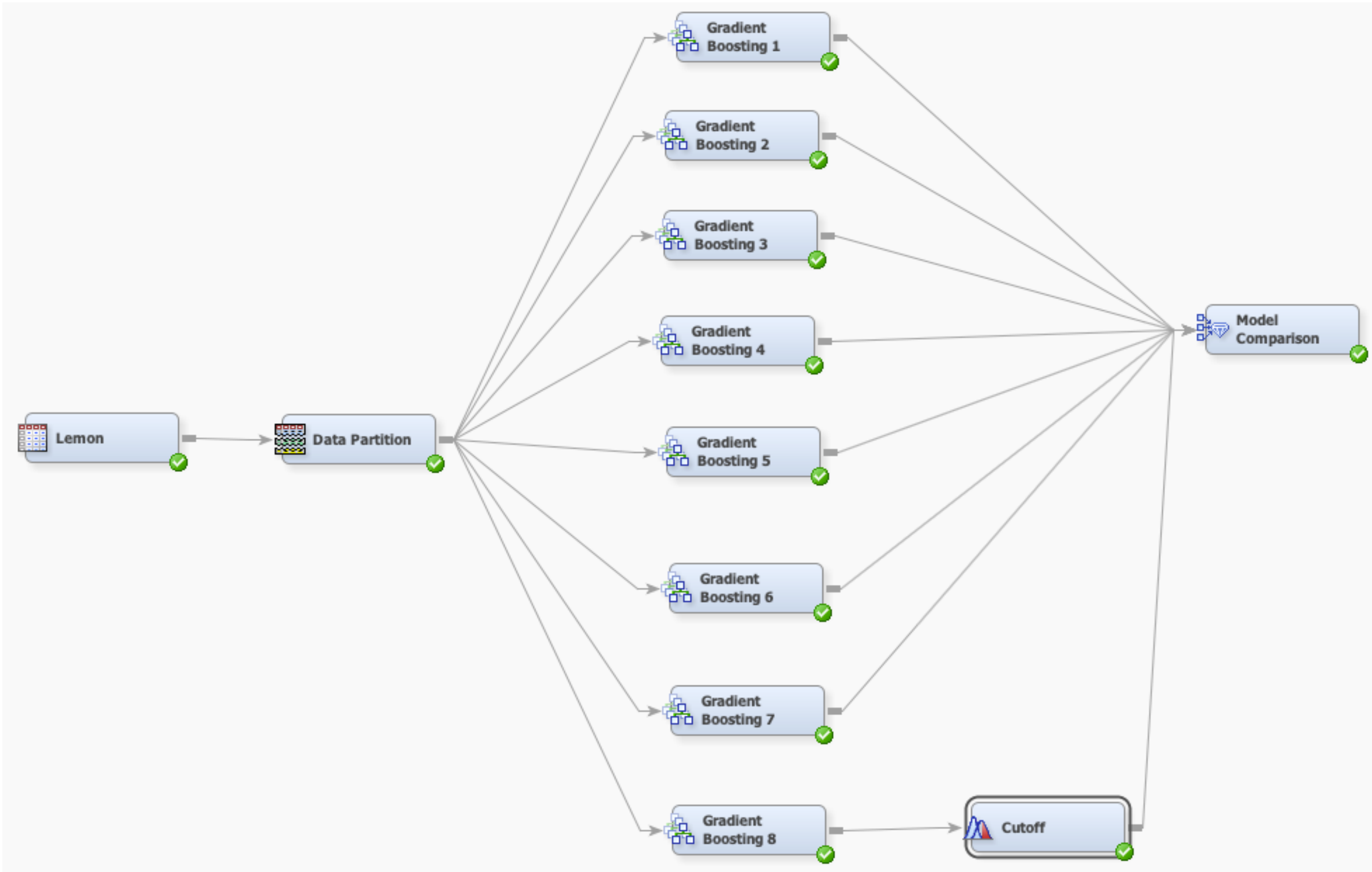

Figure 5: Phase 1 Model comparison for Gradient Boosting

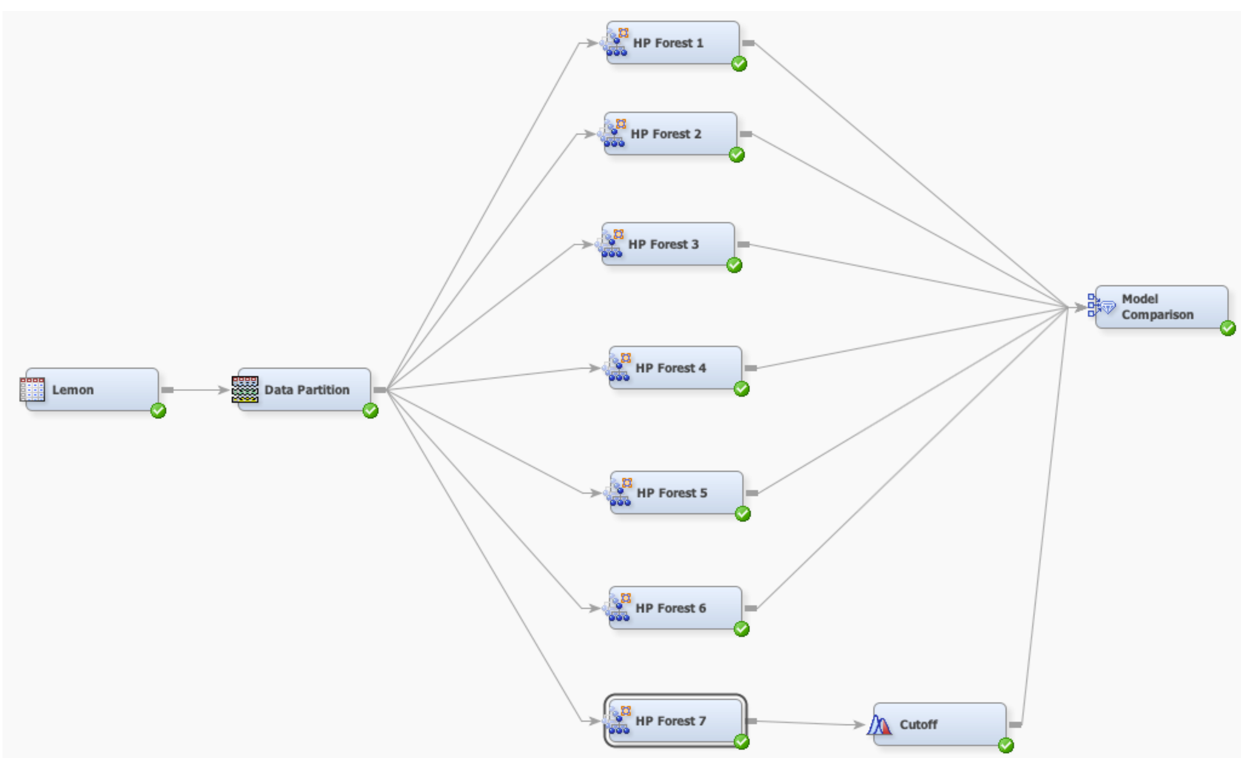

Figure 6: Phase 1 Model comparison for Random Forest

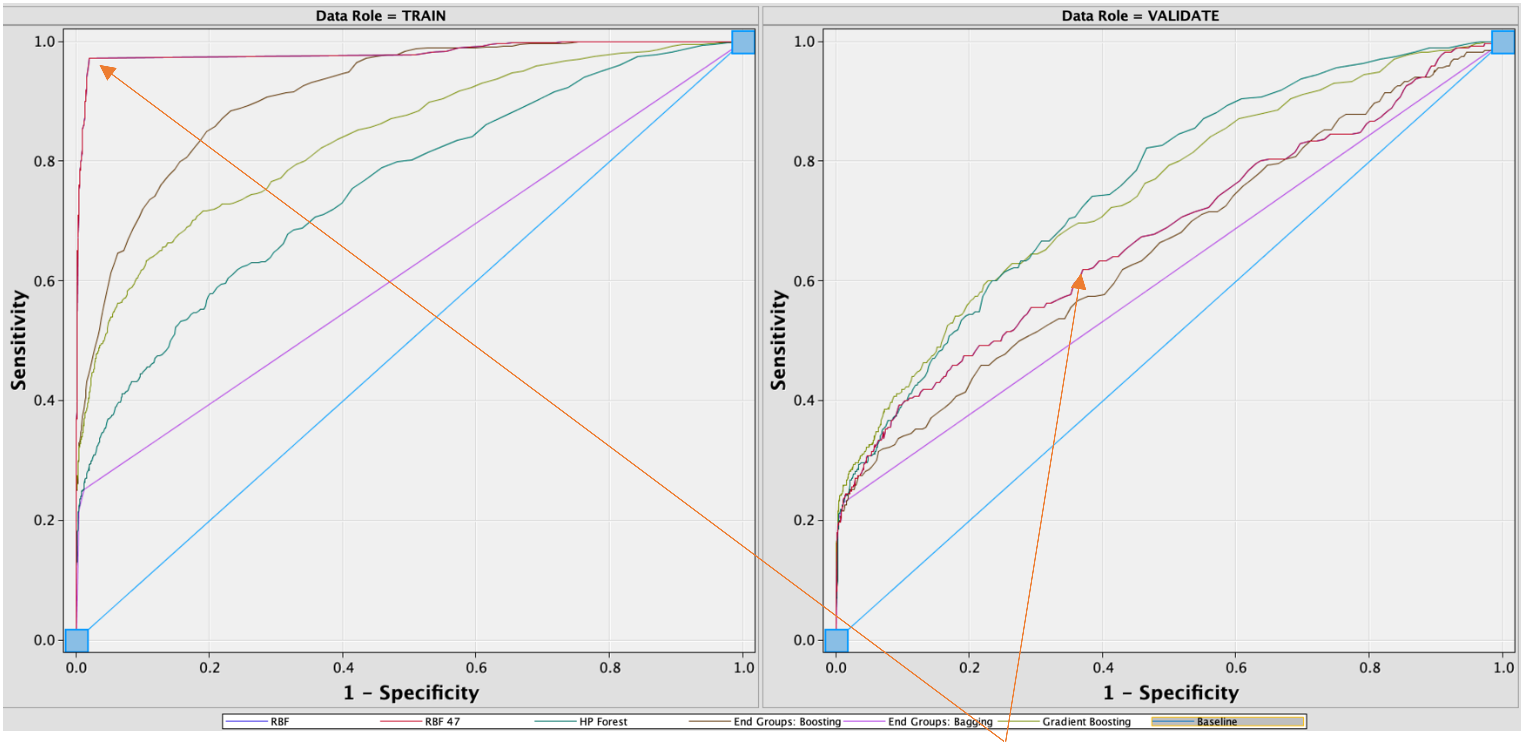

Figure 7: Model Comparison of Finalists ROC curves. Note the overfitting in SVM models.

Enter weight values for the decisions

Level	DECISION1	DECISION2
1 ...	0.0	6.9
0 ...	1.0	0.0

Figure 8: Cost Function Values

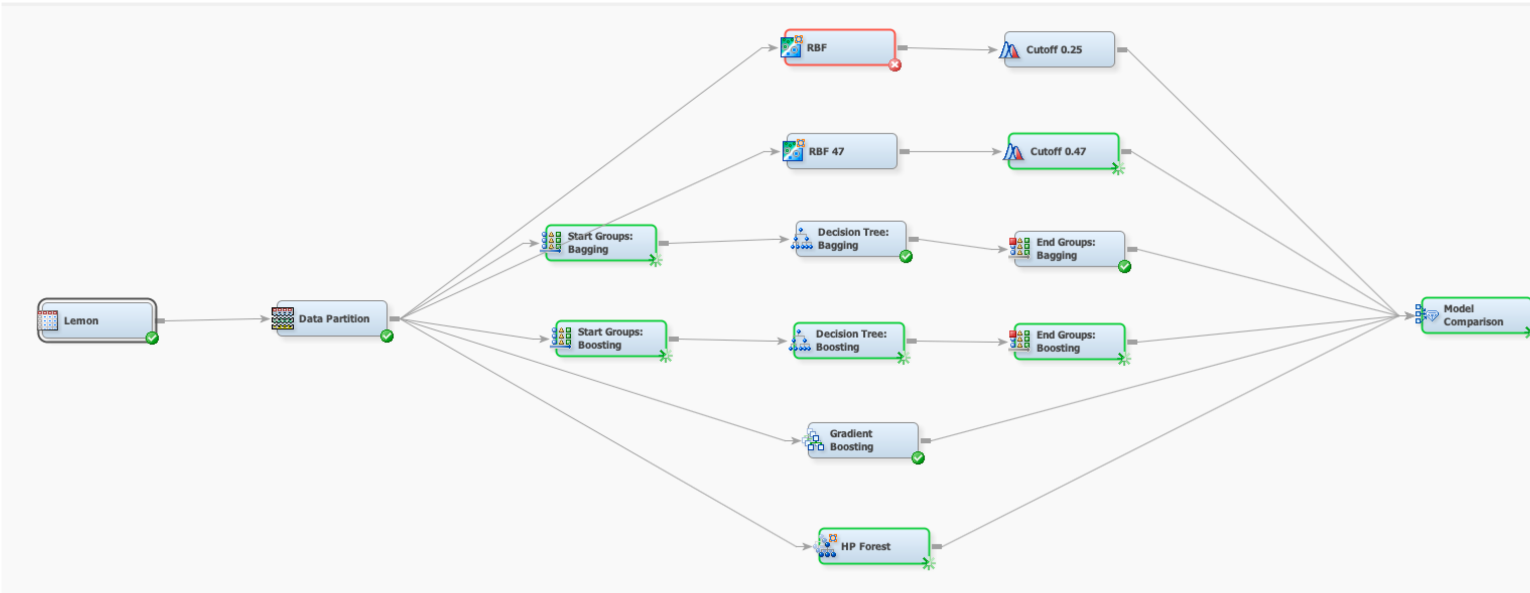

Figure 9: Model Comparison Diagram for all Finalist Models

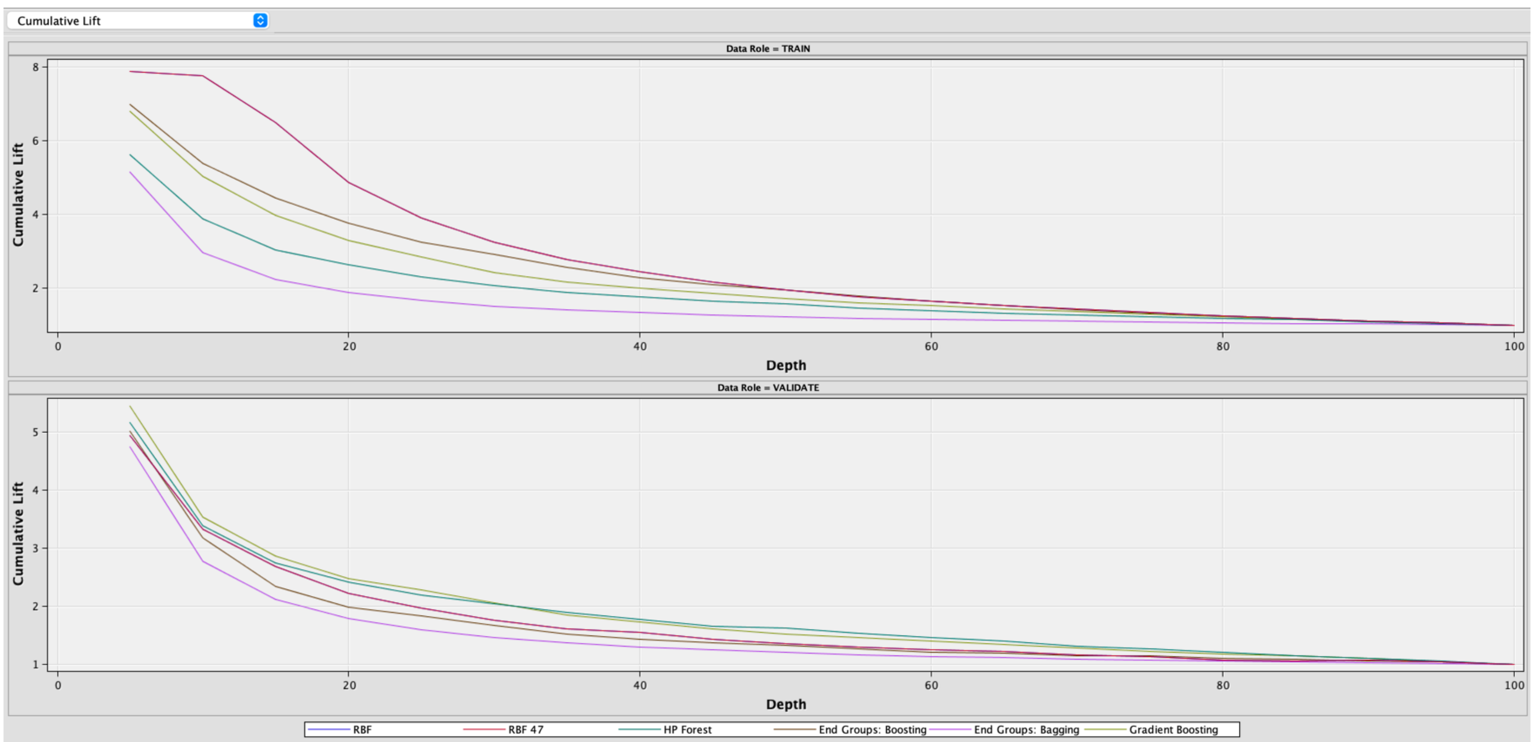

Figure 9: Cumulative lift for all finalist models

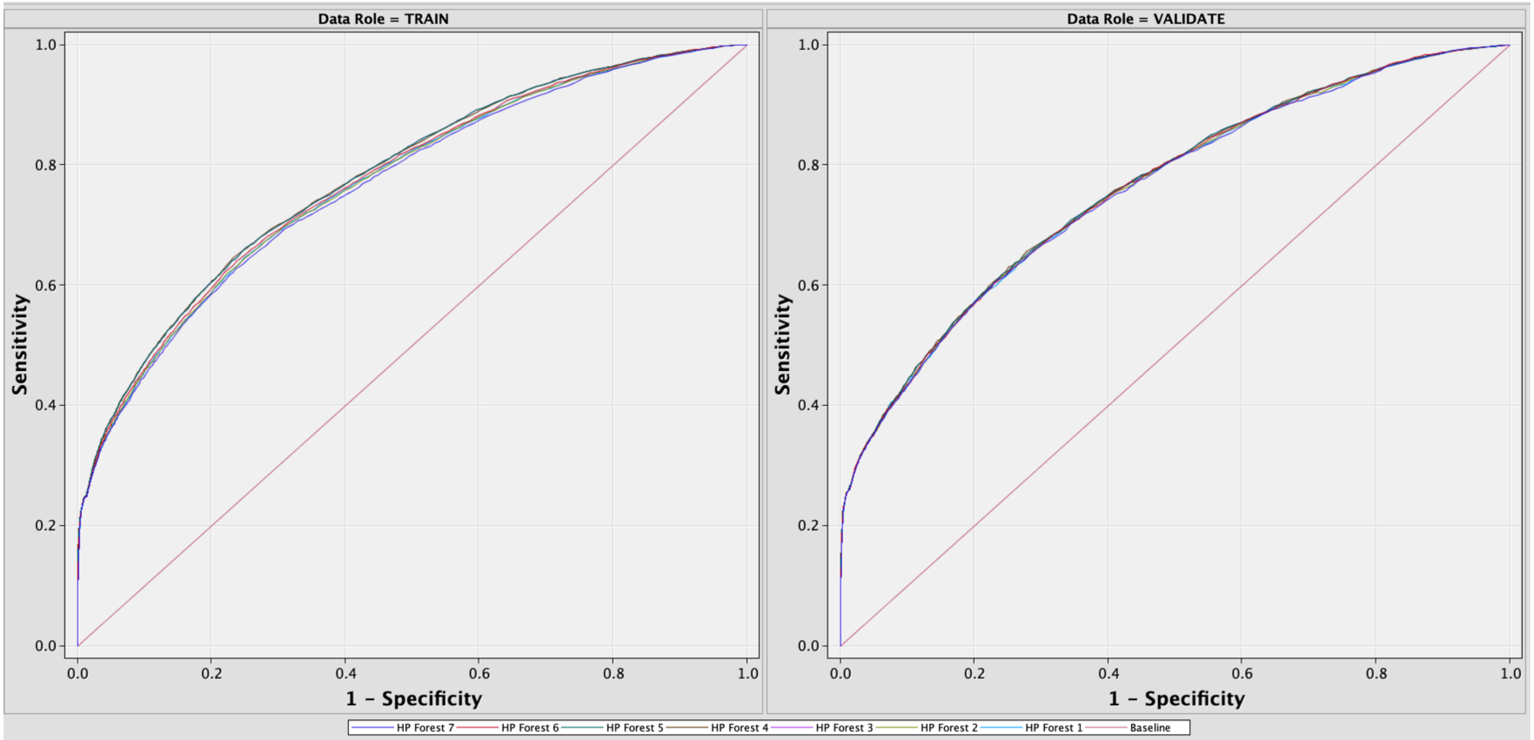

Figure 9: Random Forest models showing little change with varying parameters. No overfitting.

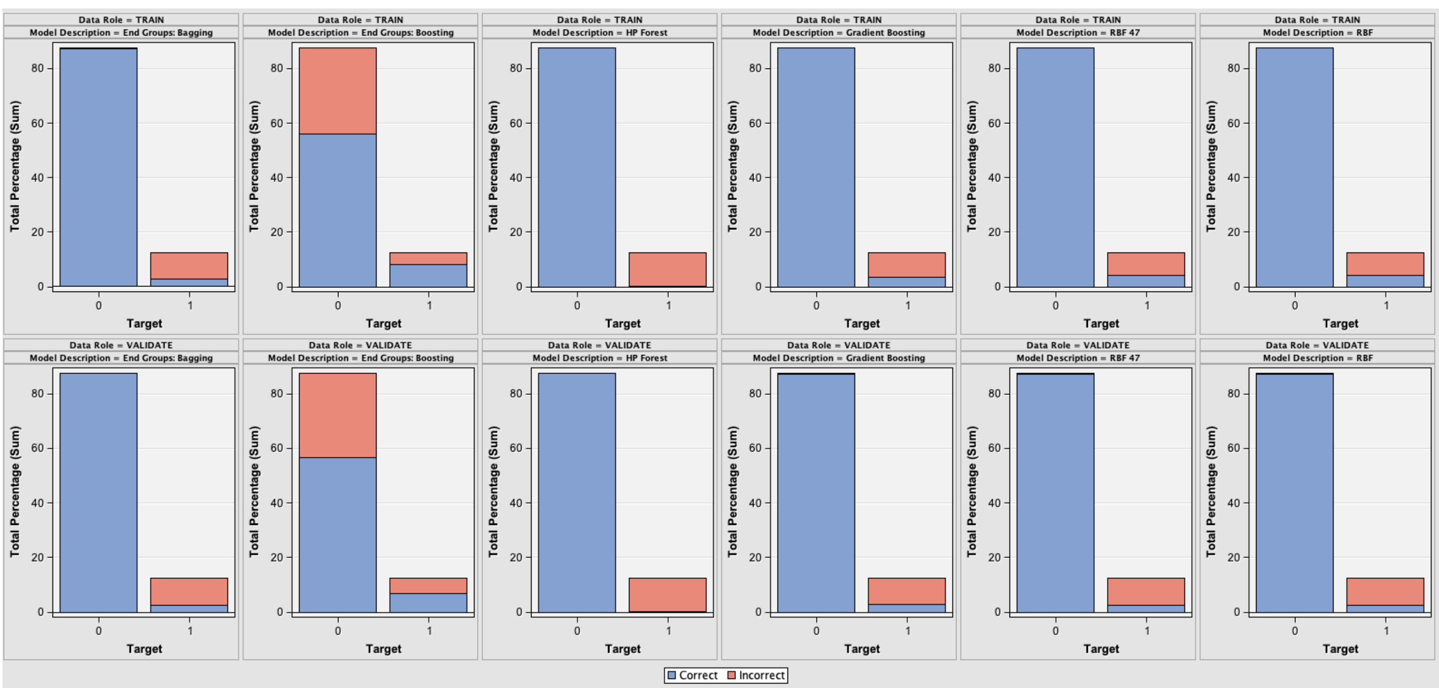

Figure 10: Classification charts for Finalist Models. Training and Validation