Machine Learning Algorithms - Summary

Chris Winsor

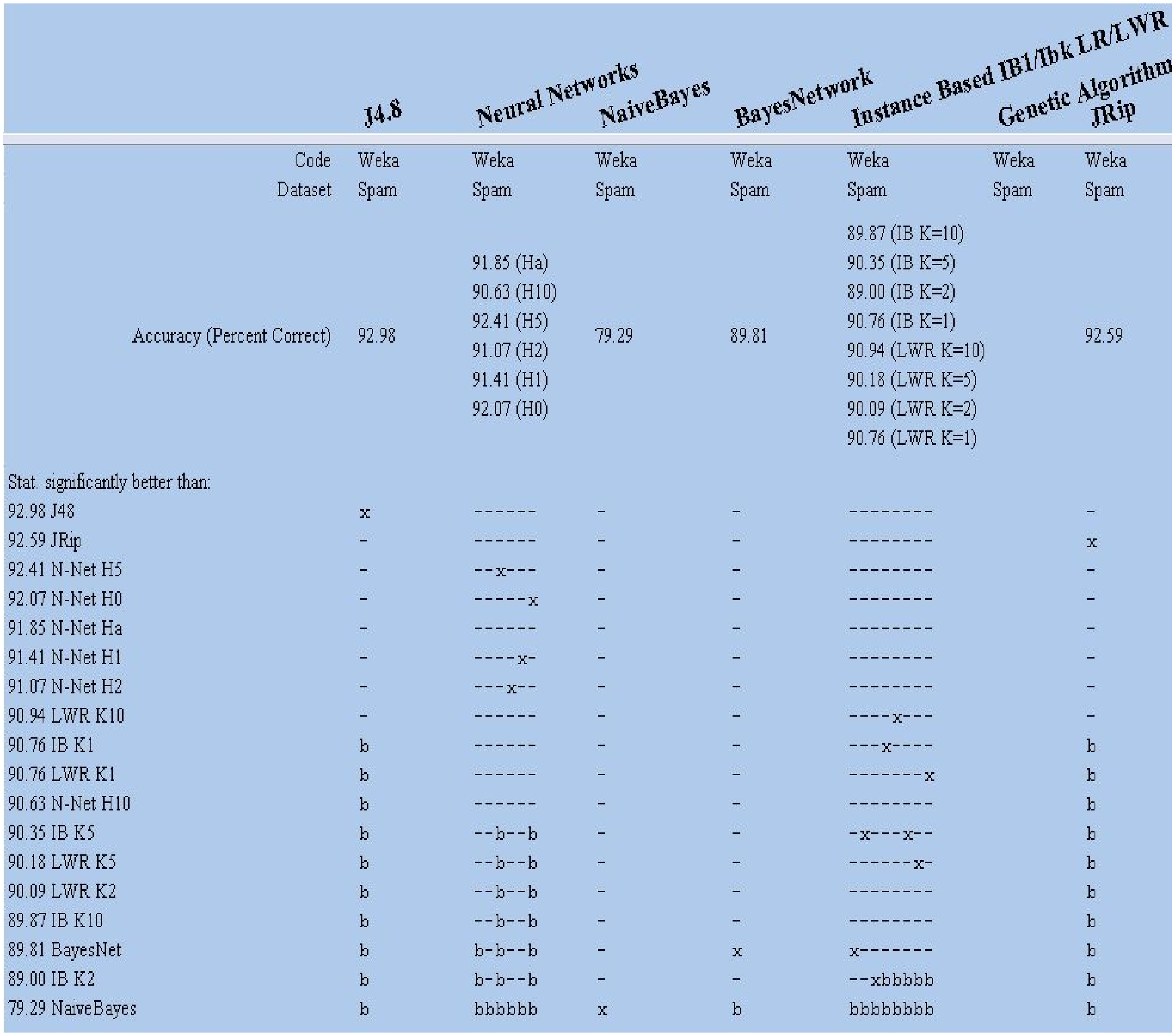
A head-to-head comparison of seven popular Machine Learning algorithms is performed. Detailed analysis of each technique is available in earlier exercises. Using the identical dataset we summarize performance with respect to accuracy, model size, readability, computational requirements and other factors.

Metrics

Experiments were run using a variety of Machine Learning techniques against the spambase dataset. For J4.8 we used 4 Internal folds. Neural Networks used Learning Rate of 0.3 and Momentum 0.2. The search algorithm for Bayes Net was K2. The search algorithm for IBK was LinearNNSearch. JRIP used 4 Internal Folds. Ten-fold cross-validation was used in all cases.

Performance (% Correct)

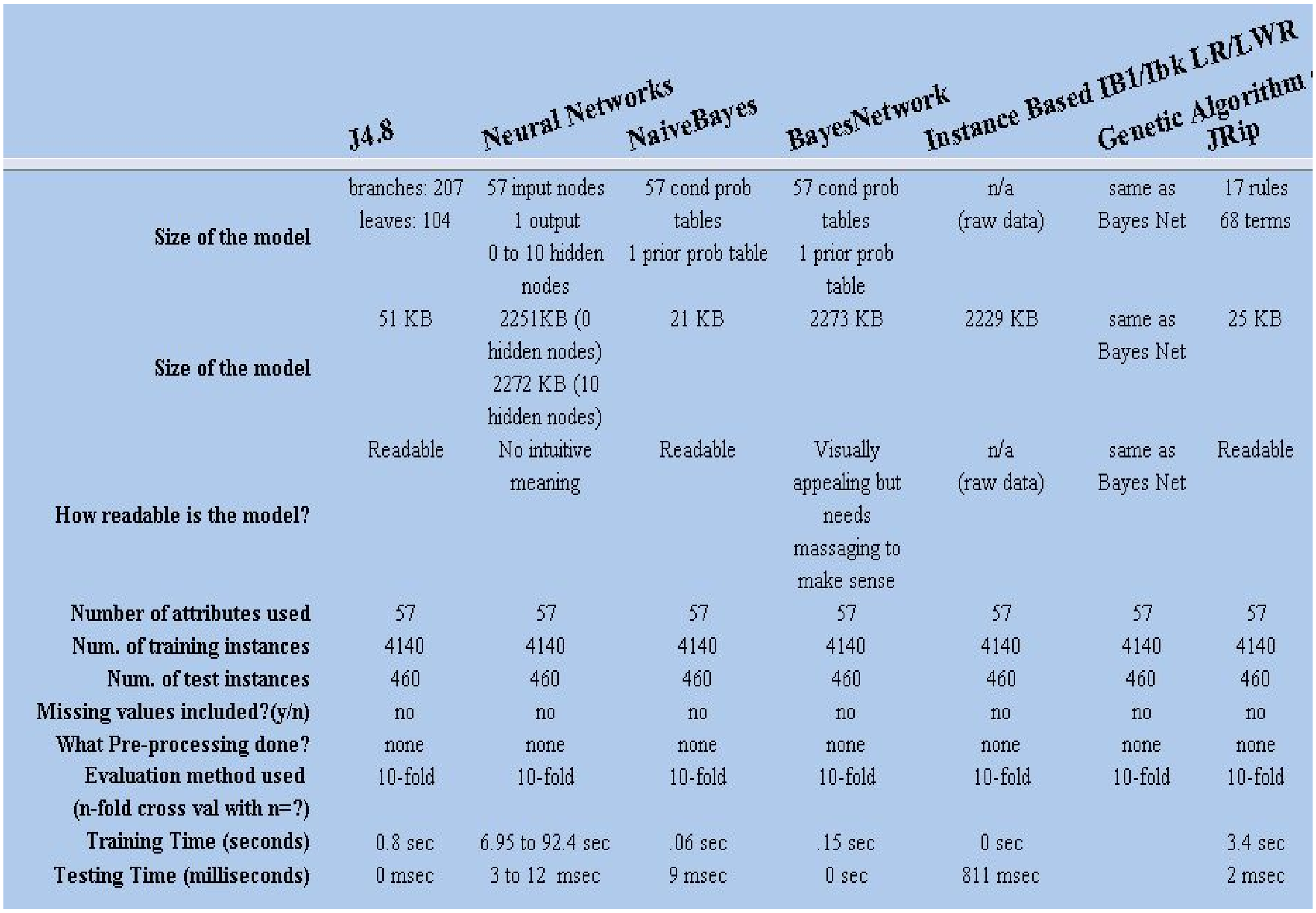
Paired T-test was used to measure performance of the various models. Weka provides automated T-testing for all except FOIL which is covered using manual analysis in a later section.



	J4.8	Neural Networks	NaiveBayes	BayesNetwork	Instance Based IB1/Ibk LR/LWR	Genetic Algorithm	JRip
Code	Weka	Weka	Weka	Weka	Weka	Weka	Weka
Dataset	Spam	Spam	Spam	Spam	Spam	Spam	Spam
Accuracy (Percent Correct)	92.98	91.85 (Ha) 90.63 (H10) 92.41 (H5) 91.07 (H2) 91.41 (H1) 92.07 (H0)	79.29	89.81	89.87 (IB K=10) 90.35 (IB K=5) 89.00 (IB K=2) 90.76 (IB K=1) 90.94 (LWR K=10) 90.18 (LWR K=5) 90.09 (LWR K=2) 90.76 (LWR K=1)		92.59
Stat. significantly better than:							
92.98 J48	x	------	-	-	--------		-
92.59 JRip	-	------	-	-	--------		x
92.41 N-Net H5	-	--x---	-	-	--------		-
92.07 N-Net H0	-	-----x	-	-	--------		-
91.85 N-Net Ha	-	------	-	-	--------		-
91.41 N-Net H1	-	----x-	-	-	--------		-
91.07 N-Net H2	-	---x--	-	-	--------		-
90.94 LWR K10	-	------	-	-	----x---		-
90.76 IB K1	b	------	-	-	---x----		b
90.76 LWR K1	b	------	-	-	-------x		b
90.63 N-Net H10	b	------	-	-	--------		b
90.35 IB K5	b	--b--b	-	-	-x---x--		b
90.18 LWR K5	b	--b--b	-	-	------x-		b
90.09 LWR K2	b	--b--b	-	-	--------		b
89.87 IB K10	b	--b--b	-	-	--------		b
89.81 BayesNet	b	b-b--b	-	x	x-------		b
89.00 IB K2	b	b-b--b	-	-	--xbbbbb		b
79.29 NaiveBayes	b	bbbbbb	x	b	bbbbbbbb		b

Model Size and Testing/Training Time

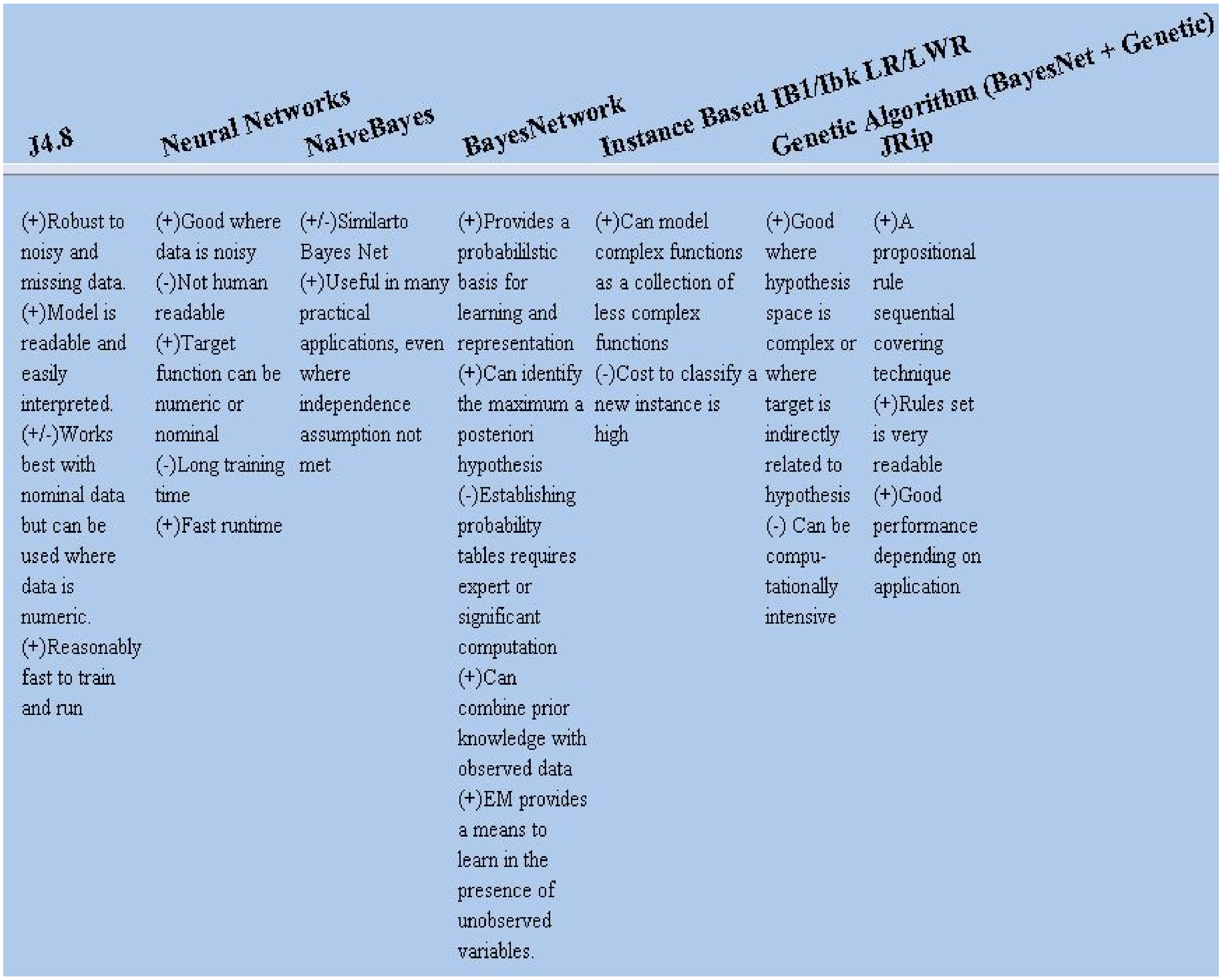
A comparison of model size and test/training time is summarized below.



	J4.8	Neural Networks	NaiveBayes	BayesNetwork	Instance Based IB1/Ibk LR/LWR	Genetic Algorithm	JRip
Size of the model	branches: 207 leaves: 104	57 input nodes 1 output 0 to 10 hidden nodes	57 cond prob tables 1 prior prob table	57 cond prob tables 1 prior prob table	n/a (raw data)	same as Bayes Net	17 rules 68 terms
Size of the model	51 KB	2251KB (0 hidden nodes) 2272 KB (10 hidden nodes)	21 KB	2273 KB	2229 KB	same as Bayes Net	25 KB
How readable is the model?	Readable	No intuitive meaning	Readable	Visually appealing but needs massaging to make sense	n/a (raw data)	same as Bayes Net	Readable
Number of attributes used	57	57	57	57	57	57	57
Num. of training instances	4140	4140	4140	4140	4140	4140	4140
Num. of test instances	460	460	460	460	460	460	460
Missing values included?(y/n)	no	no	no	no	no	no	no
What Pre-processing done?	none	none	none	none	none	none	none
Evaluation method used (n-fold cross val with n=?)	10-fold	10-fold	10-fold	10-fold	10-fold	10-fold	10-fold
Training Time (seconds)	0.8 sec	6.95 to 92.4 sec	.06 sec	.15 sec	0 sec		3.4 sec
Testing Time (milliseconds)	0 msec	3 to 12 msec	9 msec	0 sec	811 msec		2 msec

Strengths / Weaknesses

Strengths and weakness of the techniques are summarized.



J4.8	Neural Networks	NaiveBayes	BayesNetwork	Instance Based IB1/Ibk LR/LWR	Genetic Algorithm (BayesNet + Genetic)	JRip
(+)Robust to noisy and missing data. (+)Model is readable and easily interpreted. (+/-)Works best with nominal data but can be used where data is numeric. (+)Reasonably fast to train and run	(+)Good where data is noisy (-)Not human readable (+)Target function can be numeric or nominal (-)Long training time (+)Fast runtime	(+/-)Similarto Bayes Net (+)Useful in many practical applications, even where independence assumption not met	(+)Provides a probabililstic basis for learning and representation (+)Can identify the maximum a posteriori hypothesis (-)Establishing probability tables requires expert or significant computation (+)Can combine prior knowledge with observed data (+)EM provides a means to learn in the presence of unobserved variables.	(+)Can model complex functions as a collection of less complex functions (-)Cost to classify a new instance is high	(+)Good where hypothesis space is complex or where target is indirectly related to hypothesis (-) Can be computationally intensive	(+)A propositional rule sequential covering technique (+)Rules set is very readable (+)Good performance depending on application

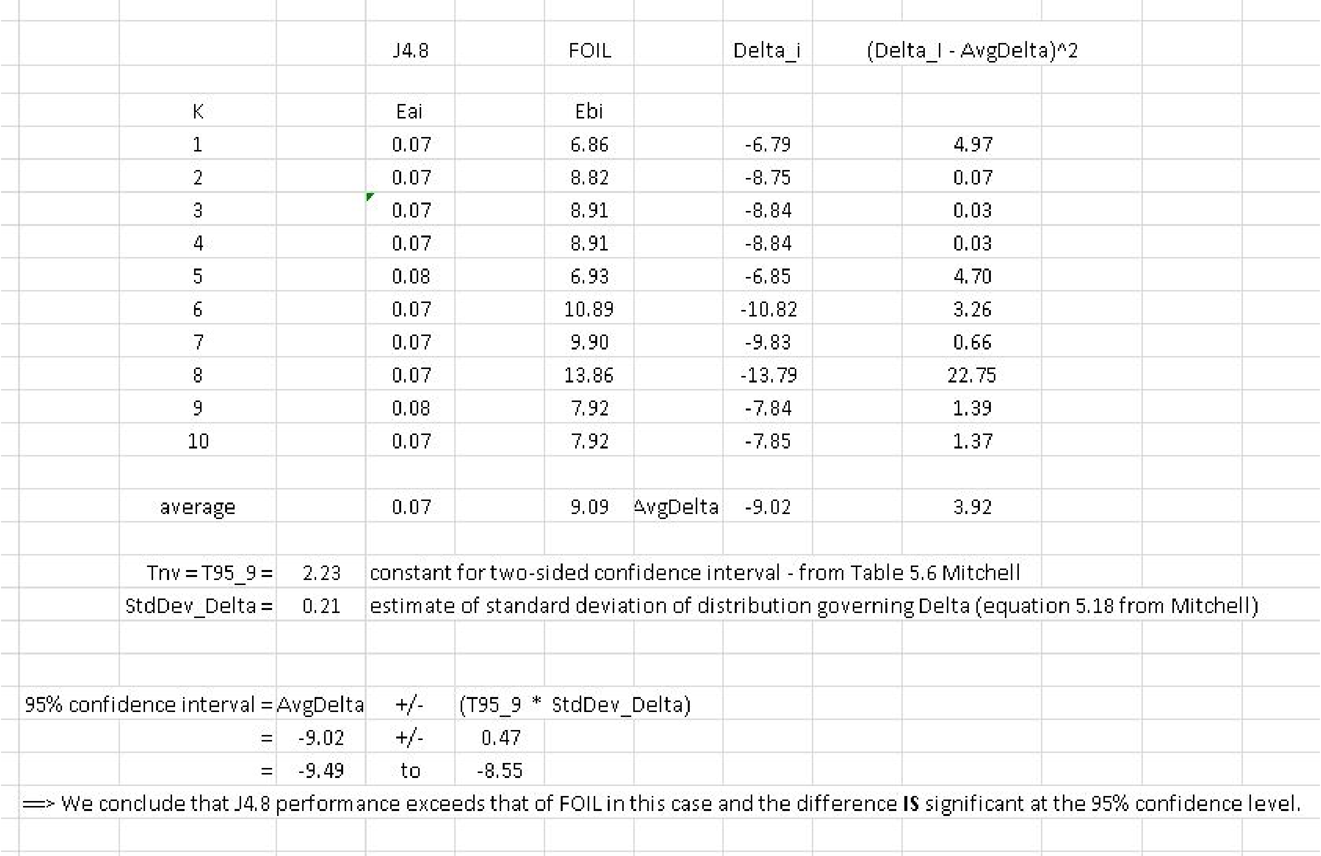
Paired T-Test (FOIL vs J4.8)

Performance comparison between FOIL and J4.8 is performed using Paired T-test analysis. A manual procedure is required because Weka does not support FOIL so there automated T-Test is not available from Weka.

The performance metric was "percent correct predictions". A confidence interval of 95% and K=10 were used. Mitchell (section 5.6) identifies the procedure.

Preprocessing:

- Starting with the original Spambase data, Weka was used for attribute discretization and selection. Selection was made using CfsSubsetEval.
- Ten test/training sets are made, each with a 70%/30% train/test split with instances randomly chosen without replacement.
- J4.8 was then run on each of the ten datasets.
- FOIL was then run on each of the ten datasets. Terms, constants and variables were as in Part I above.
- The results were then compared using the Paired T-test procedure.



	J4.8	FOIL		Delta_i	(Delta_I - AvgDelta)^2
K	Eai	Ebi			
1	0.07	6.86		-6.79	4.97
2	0.07	8.82		-8.75	0.07
3	0.07	8.91		-8.84	0.03
4	0.07	8.91		-8.84	0.03
5	0.08	6.93		-6.85	4.70
6	0.07	10.89		-10.82	3.26
7	0.07	9.90		-9.83	0.66
8	0.07	13.86		-13.79	22.75
9	0.08	7.92		-7.84	1.39
10	0.07	7.92		-7.85	1.37
average	0.07	9.09	AvgDelta	-9.02	3.92

Tnv = T95_9 = 2.23 constant for two-sided confidence interval - from Table 5.6 Mitchell

StdDev_Delta = 0.21 estimate of standard deviation of distribution governing Delta (equation 5.18 from Mitchell)

95% confidence interval = AvgDelta +/- (T95_9 * StdDev_Delta)

= -9.02 +/- 0.47

= -9.49 to -8.55

==> We conclude that J4.8 performance exceeds that of FOIL in this case and the difference **IS** significant at the 95% confidence level.

From this analysis we conclude that J4.8 performance exceeds that of FOIL in this case and the difference is significant at the 95% confidence level.

Summary

Bayes Network and Naive Bayes had lowest performance of the group (demonstrated to be statistically lower performing than J4.8 and JRIPPER. But performance aside, Bayes' probabilistic foundation is compelling when considering a system that needs to regularly deal with data that is ambiguous, or in cases where knowledge is partial. Bayes networks can be established using information from a knowledge expert (in which case the structure makes intuitive sense), or can be derived from the data (in which case it may or may not make intuitive sense).

Neural Networks demonstrate a high level of accuracy and flexibility. The training time for a NNet can be long but once the model is created runtime performance is fast. This makes NNets suited to applications where the model can be created in advance and deployed.

J4.8 Decision Tree - The highest performing solution of the bunch - J4.8 slightly outperformed JRIPPER although this is not demonstrated with a 95% confidence level. The decision tree is very readable and can make intuitive sense. Its training and runtime is very reasonable.

JRIPPER (sequential covering using propositional rules) This appears to be a very good technique with performance on par with J4.8. But training and runtime are fairly high (3.4sec training 2ms runtime) which makes this the slowest of the group except for NNets. The resulting model (a set of propositional rules) is fairly readable but flat and uninspired (my opinion). Of course it does not provide the benefits of a first-order language. So this is a very reasonable technique in a field of many competitors.

IBK and LWLR (Instance-based) - The big advantage of instance-based techniques is their ability to cover a complex domain by focusing on a smaller subset of instances. This is done by localizing the model to just K relevant nodes. A challenge for these models is runtime performance, as evidenced in our data with a "test" time of 811ms - by far the largest in the group. A good implementation would need a lookup table to find K neighbors, and means of managing additions of new instances. But for complex domains this technique is compelling solution.

Genetic Algorithms - We were unable to get through a single pass using Genetic Algorithms as our search strategy for Bayes Network. GA can be computationally intensive as it needs to iterate through generations across many individuals. GA can provide novel solutions through mutation and crossover.

FOIL - The use of first-order language is compelling especially when modeling mathematical functions. In these situations FOIL established crisp and generalized solutions. When faced real-world (noisy,ambiguous) data the models were much less crisp and intuitive. But for certain types of problems a first-order language and FOIL are an exceptional combination.