Random Forest: A Reliable Tool For Patient Response Prediction

David Dittman*, Taghi M. Khoshgoftaar*, Randall Wald*, and Amri Napolitano*
*Department of Computer and Electrical Engineering and Computer Science
Florida Atlantic University, Boca Raton, Florida 33431
Email: ddittman@fau.edu; khoshgof@fau.edu; rdwald@gmail.com; amrifau@gmail.com

***Abstract*—The goal of classification is to reliably identify instances that are members of the class of interest. This is especially important for predicting patient response to drugs. However, with high dimensional datasets, classification is both complicated and enhanced by the feature selection process. When designing a classification experiment there are a number of decisions which need to be made in order to maximize performance. These decisions are especially difficult for researchers in fields where data mining is not the focus, such as patient response prediction. It would be easier for such researchers to make these decisions if either their outcomes were chosen or their scope reduced, by using a learner which minimizes the impact of these decisions. We propose that Random Forest, a popular ensemble learner, can serve this role. We performed an experiment involving nineteen different feature selection rankers (eleven of which were proposed and implemented by our research team) to thoroughly test both the Random Forest learner and five other learners. Our research shows that, as long as a large enough number of features are used, the results of using Random Forest are favorable regardless of the choice of feature selection strategy, showing that Random Forest is a suitable choice for patient response prediction researchers who want to do not wish to choose from amongst a myriad of feature selection approaches.**

***Keywords*-patient response, Microarray, Random Forest**

I. Introduction

The primary goal of cancer treatment is to destroy the cancer while minimizing the harm done to the patient by the treatment. Many treatments can have harmful and life-threatening side effects, even if they successfully combat the disease. Therefore, it is very important to choose the correct treatment for a patient in order to reduce the amount of harm done. Knowing in advance how a patient will respond to a treatment would greatly reduce the number of incorrect treatments prescribed. This paper is a study utilizing machine learning to perform patient response prediction, specifically prediction of multiple myleoma response to the drug bortezomib. As a preliminary work, we focus on a single cancer and drug to refine these techniques prior to application to other drugs and cancers. In addition, very few patient response datasets exist for multiple myleoma.

In 2007, Mulligan et al. [1] performed a study to try to predict patient response to a proteasome inhibitor called bortezomib. The goal was to use gene expression profiles (created with a DNA microarray and testing samples for over 22,000 gene sets) from a number of patients who underwent this treatment and had their responses recorded, and use the results to predict the patient response. This study is particularly interesting due to its focus (the response of a single cancer to a single drug), its scale (169 patients and >22,000 genes), and the room for improvement left by its results; hence its selection as the basis for our work.

One of the main problems with DNA microarray datasets is high dimensionality (a large number of features in each instance). A method of combating this problem is feature selection, the process of selecting a subset of features and only using those features for building models. Despite the elimination of data, feature selection leads to the creation of classifiers that are more efficient and possibly more accurate [2]. However, feature selection can be a daunting task. When choosing a feature selection method there are a number of choices to be made, including which learner to use, whether one should choose to use a wrapper or a filter based method, which method should be picked within the chosen type, how many features need to be selected to achieve accurate classification, etc. Making these decisions is especially difficult for researchers whose main focus is not data mining. Making these choices would be much easier to handle if some of the decisions were less important or were made ahead of time.

We propose that this problem can be solved by using the Random Forest learner, a commonly used ensemble learner implemented in the Weka [3] data mining tool. Ensemble learners use multiple models to achieve better results than any of the models alone. In particular, Random Forest uses an ensemble of unpruned decision trees. Our results show that due to the nature of the learner, the choices for designing the feature selection method either may be preselected or have little impact on the outcome. Thus, practitioners may simply use the parameters discussed in this paper, confident that they will achieve good results even without an extensive background in data mining.

This article is organized as follows. The second section of this paper discusses previous works involving Random Forest, as well as background on our dataset. The following section contains information on the principles of feature selection and specific explanation of the techniques we used. The fourth section contains the methods used when conducting our experiment, while the fifth section shows our results. Lastly, the sixth section presents our conclusions.

II. Related Works

The Random Forest algorithm was introduced by Leo Breiman in 2001. His chapter "Random Forest" in the Springer book "Machine Learning" [4] gave a description of the technique and its mechanics, as well as comparing it to other learners. Random Forest had a number of advantages over the other learners, such as robustness to outliers and noise, speed when compared with similar techniques like bagging and boosting, simplicity of implementation, and ability to provide internal information including error, strength, and correlation. In the end, when compared to other learners, Random Forest performed well and was an "effective tool in prediction."

While patient response is an important possible use for Random Forest, it has previously been used for analyzing DNA microarray data. A 2006 experiment by Diaz-Uriarte and Alvarez de Andres used Random Forest as a classifier on ten different DNA microarray datasets from diseases that affect different areas of the human body [5]. Diaz-Uriarte et al. found that when compared to other methods, Random Forest is "competitive with alternative methods, without requiring any fine-tuning of parameters or pre-selection of variables." Due to its ease of use and its effectiveness in classification, they recommended that Random Forest be considered a standard for microarray data.

As we are using the same dataset as Mulligan et al., some background knowledge of their experiment is warranted. The goal of the study was "to assess the feasibility of prospective pharmacogenomics research in multicenter international clinical trials of bortezomib in multiple myeloma and to develop predictive classifiers of response and survival with bortezomib" [1]. The study ran a number of experiments, including using gene expression profiles to evaluate overall survival and running a Gene Set Enrichment Analysis (GSEA) which looks to see if any of the genes used in the learners belong to a known or annotated set of genes. The part of their paper that is relevant to the present study is the work on predicting patient response through the use of gene expression profiles. In their study, the top 100 differentially-expressed genes were used to build predictors using a classifier based on naïve Bayes.

III. Feature Rankers

There are two main approaches to feature selection, filter and wrapper. Filter feature selection techniques analyze the features without any regard to a classifier. The filter approach uses only the raw dataset to decide which features are to be used to create the best classifier. Since no classifier is used, filter methods must rely on statistical measures. By using various statistical tests, the filter method tries to make decisions on features based on their levels of relevancy (how much the feature has to do with the chosen class) and redundancy (whether the feature gives information already found in other features). Wrappers, unlike filter approaches, use classifiers when making a decision, and often the classifier used to calculate the score of a particular feature subset is the same one that will be used in the post selection analysis. However, building a classifier can be very involved even for only one model, and is compounded when multiple models are being built [6]. Therefore, for the scope of our experiment we only use filter based feature rankers.

Table I
List of 19 Filter-Based Feature Selection Techniques

Abbreviation	Name
AUC	Area Under the Receiver Operating Characteristic (ROC) Curve
Dev	Deviance
F	F-measure
GM	Geometric Mean
GI	Gini Index
GR	Gain Ratio
IG	Information Gain
KS	Kolmogorov-Smirnov statistic
MI	Mutual Information
OR	Odds Ratio
Pow	Power
PR	Probabiltiy Ratio
PRC	Area Under the Precision-Recall Curve
RF	ReliefF
RFW	ReliefF — Weight by Distance
S2N	Signal-to-Noise
SVM	SVM Recursive Feature Elimination
SU	Symmetric Uncertainty
χ^2	χ^2 Statistic

The feature rankers chosen can be placed into three categories: commonly used filter based feature selection techniques, a new filter technique called Signal to Noise, and threshold-based feature selection techniques (TBFS) that were developed by our research team. Table I contains all of the feature selection techniques used and their abbreviations.

A. Non-TBFS Feature Selection Techniques

Eight non-TBFS filter-based feature ranking techniques were used in this work: chi-squared [3], information gain [7], gain ratio [3], two versions of ReliefF [8], symmetric uncertainty [6], SVM [3], [9], and signal-to-noise [10]. All of these feature selection methods, with the exception of signal-to-noise, are available within the Weka machine learning tool [3]. Outside of SVM, Weka's default parameter values were used unless otherwise noted. With SVM we do not perform the recursive feature elimination but we used the weighted list of features created prior to the first elimination as our rankings. Since most of these methods are widely known and for space considerations, the interested reader can consult with the included references for further details.

S2N is less often used in the context of feature selection and therefore some background information is needed. The signal-to-noise ratio, or S2N, as it relates to classification or feature selection, represents how well a feature separates two classes. The equation for signal to noise is:

$$S2N = (\mu_P - \mu_N)/(\sigma_P + \sigma_N)$$

where μ_P and μ_N are the mean values of that particular attribute in all of the instances which belong to a specific class, either P or N (the positive and negative classes). σ_P and σ_N are the standard deviations of that particular attribute as it relates to the class. The larger the S2N ratio, the more relevant a feature is to the dataset [10].

B. Threshold-Based Feature Selection Techniques

This section describes the TBFS method for feature ranking. In TBFS each attribute is evaluated against the class, independent of all other features in the dataset. After normalizing each attribute to have a range between 0 and 1, simple classifiers are built for each threshold value $t \in [0, 1]$ according to two different classification rules. The normalized values are treated as posterior probabilities. However, no real classifiers are being built and therefore the TBFS techniques are still considered filter methods and not wrapper methods. For classification rule 1, examples with a normalized value greater than t are classified P while examples with a normalized value less than t are classified as N (assuming each instance x is assigned to one of two classes $c(x) \in \{P, N\}$). For classification rule 2, examples with a normalized value greater than t are classified N while examples with a normalized value less than t are classified as P. Two different classification rules must be considered to account for the fact that for some attributes, large values of the attribute may have a greater correlation with the positive class, while for other attributes, large values of the attribute may have a greater correlation with the negative class. Metric ω is calculated using the formulas related to each metric either at each threshold t or across all thresholds for both classification rules. Finally, the metric resulting from the classification rule which provides the best value is used as the relevancy measure for that attribute relative to class or class attribute ω. For a more in depth look at the overall algorithm of the TBFS techniques refer to the paper by Dittman et al. [11].

Many of the metrics ω (e.g., Area Under the ROC Curve (AUC), Area Under the Pecision-Recall Curve (PRC), Geometric Mean (GM), F-Measure (F), and the Kolmogorov-Smirnov statistic (KS)) are primarily used to measure the performance of classification models, using the posterior probabilities computed by such models to classify examples as either negative or positive depending on the classification threshold. The normalized attribute values can be thought of as posterior probabilities, e.g., $p(P \mid x) = \hat{X}^j(x)$ for classification rule 1, and the metrics ω are computed against this "posterior." Intuitively, attributes where positive and negative examples are evenly distributed along the distribution of X produce weak measures ω and poor relevancy scores in a similar manner that poor predictive models have positive and negative examples evenly distributed along the distribution of the posterior probability produced by the model. Note further that TBFS can easily be extended to include additional metrics.

For additional information on the AUC, PRC, KS, GM, and F metrics, see [3], [12], [13]. The F metric uses the tunable parameter β to weight precision (PRE) and recall or true positive rate (TPR). In this work, $\beta = 1$ is used, and hence F is the harmonic mean of recall and precision. The Gini index (GI) was introduced by Breiman et al. [14]. Power (Pow), Odds Ratio (OR), and Probability Ratio (PR) were used by Forman [15] in the context of feature selection for text categorization. Additional information on Deviance (Dev) can be found in Khoshgoftaar et al. [16] and Mutual Information (MI) is utilized in Battiti [17].

IV. Methods

As patient response DNA microarray datasets are somewhat rare, especially those which focus on a single cancer and drug, but which have many instances, we chose to focus on the data from Mulligan et al. To highlight the distinctions between the present work and the earlier Mulligan et al. study, we use the same class definitions found in that paper. The classes of interest, or positive classes, consist of Complete Response, Partial Response, and Minimal Response. The other classes, or the negative classes, include No Change and Progressive Disease. Mulligan et al. performed two different experiments which combined these five classes into a "positive" and "negative" class. In both experiments, the classes of interest did not change but were combined into one class called "R." However the negative classes did change between the experiments. In one experiment all the negative classes were combined to form a single class called "NR." The other experiment ignored the No Change class and only concentrated on the Progressive Disease class; therefore, the negative class for that experiment was referred to as "PD" [1].

The first step in the analysis is to rank the entire set of features from the patients. As some of the techniques including the TBFS and S2N techniques require that there only be two classes, we can only use two classes in our classification. Conveniently, the dataset was already split into two classes, the class of interest ("R") and the other class ("NR" or "PD"). A mixed set of nineteen feature rankers, consisting of seven commonly used filter based feature selection techniques in addition to eleven threshold based filter techniques proposed by our research group, and a rarely used feature ranker called signal to noise, was chosen for the analysis of the features in the gene expression profiles. After the rankings were performed the data was classified with the chosen learners.

Before the learners can be applied, the number of features to use in building the learner must be determined. We decided to use fourteen different feature set sizes based on the number of features in the total dataset. The feature set sizes used were: 5, 10, 15, 20, 25, 50, 75, 100, 150, 200, 250, 350, 500, and 1000. The reason why we choose such small numbers of features (compared with the >22,000 found in the complete dataset) is that while every cell in the body contains the entire genetic code of the organism, the genes that are actually activated and used in the cell are a very small subset [18]. The larger numbers (350, 500, and 1000) were chosen to ensure the thoroughness of the study; numbers beyond this range would have begun to subvert the intention of feature selection (i.e., if a large proportion of the features are chosen, the feature selection won't be able to exclude irrelevant features or to aid computational efficiency).

Random Forest (RFT) is an ensemble learner implemented by the data mining tool Weka. The learner builds a set of unpruned decision trees which use majority voting to perform prediction. In order to produce a robust learner, ensemble learners employ randomness. For Random Forest, this randomness is implemented by selecting a bootstrap (sampling with replacement) of instances to build the decision trees. The bootstrapping process is repeated for each tree created by the learner. The trees themselves are built by taking the bootstrap samples and using an algorithm similar to C4.5 (J48 in Weka). There are two main differences between the C4.5 algorithm and the algorithm employed by Random Forest. The first is that at each node the number of possible features to place at that node is restricted to a randomly selected subset of the features, the size of which is determined by the *numFeatures* attribute. The other is that, unlike C4.5, the trees in Random Forest are always unpruned. The number of trees used is determined by the *numTrees* attribute. Previous research [19] shows that the optimum number of trees is 100, so that is the number used in this study. After the trees are created the learner begins testing the instances. Each instance is passed through each tree and the predicted class is chosen. The class that is chosen by the most trees becomes the chosen class of the instance [4].

In this experiment we also used five different classifiers (learners) other than Random Forest in order to perform each classification. The five learners used were the Support Vector Machine (SVM), Naive Bayes (NB), k-Nearest Neighbors (5-NN because we used a value of 5 for k), Logistic Regression (LR), and Multi-Layer Perceptron (MLP). The data was run through each learner for binary classification with respect to patient response. The two possible pairs of classes are R and NR or R and PD.

Due to the fact that they are standard techniques discussed elsewhere, we will not present the details for how each of these learners is built. For additional information on Support Vector Machines see the paper by Liu et al. [20]. Naive Bayes and 5-Nearest Neighbors can be found in the work of Souza et al. [21]. Logistic Regression can be explained by LeCessie et al. [22] and MLP is used by Haykin et al. [23]. Finally Random Forest can be found in the work of Khoshgoftaar et al. [19].

In order to evaluate the learner and its parameters, we used 5 fold cross validation. Cross validation is a commonly used method for evaluating a classifier. This is done by splitting the dataset into the number of folds (n). The first $n-1$ folds are used to train the classifier and the last fold is used to test it. The feature ranking and the building of the classifier described above is performed solely within the training portion. After the model has been built, it is tested using data from the nth fold, thus ensuring that the data used for testing was not used for building the model. The process is repeated until every fold has been the test fold once and each result is recorded [2]. In order to avoid a bad split and to gain a better picture of the performance of the classifier, the cross validation process itself was repeated three more times, for a total of four. The evaluation was also repeated fourteen times with varying numbers of features used in the classification. In total, 5 folds $\times$ 4 runs $\times$ 19 feature rankers $\times$ 14 different numbers of feature subsets $\times$ 2 different class sets $\times$ 6 learners = 63,840 models were built.

V. Results

Tables II through VI contain the results of our experiments using Random Forest on the patient response data. The experiments were repeated twice (R vs NR and R vs PD) and their results are separated into the tables. Our experiments were evaluated using the performance metric Accuracy, as the goal is accurate classification of patient response to the treatment. Tables II and III contain the top accuracy from each learner/filter combination as well as the accuracy across all 19 rankers for each learner. Due to space considerations, we do not present the corresponding feature subset size for each of these accuracies.

As we can see from Table II, in the R vs NR experiment, we find that with the exception of PRC and ReliefF-W, for all rankers the top performing feature subset size for Random Forest outperforms the average of the top performing sizes for the other learners. However in the cases of PRC and ReliefF-W, the differences were very small 0.00473 and 0.00121 respectively. Table III shows that in the R vs. PD experiment, only Symmetric Uncertainty's top-performing subset size of Random Forest did not outperform the average of the best sizes of the other five learners. Even there, in the case of Symmetric Uncertainty, the difference was only 0.01010. It should be noted that the top performers for each ranker using Random Forest in both R vs. NR and R vs. PD were very close to one another in accuracy. In addition to the small amount of variance in accuracy between the filters Random Forest has the largest average accuracy across all filters. As we can see in Tables II and III, in both R vs NR and R vs PD, the average across all nineteen filters for Random Forest was 0.64455 and 0.74327 respectively which is higher than all of the other learners. Due to these attributes the choice of feature ranker is less important when using Random Forest.

Table II
R VS NR TOP RESULTS

Filter	Accuracy						Average
	RFT	5-NN	LR	MLP	NB	SVM	w/o RFT
AUC	0.65258	0.61983	0.62451	0.64666	0.64969	0.63021	0.63418
Dev	0.64229	0.63908	0.62135	0.65258	0.63498	0.62161	0.63392
F	0.63944	0.58124	0.62580	0.64973	0.61867	0.63650	0.62239
GM	0.64528	0.63035	0.61546	0.62571	0.63636	0.61854	0.64528
GI	0.65107	0.60472	0.63347	0.63338	0.63168	0.63926	0.62850
GR	0.64675	0.60660	0.63195	0.62750	0.61881	0.62455	0.62188
IG	0.64523	0.61546	0.60963	0.62304	0.64955	0.61555	0.62265
KS	0.65116	0.62447	0.61110	0.63605	0.63342	0.61662	0.62433
MI	0.64078	0.61974	0.61845	0.64376	0.64822	0.61422	0.62888
OR	0.63645	0.58703	0.63044	0.64082	0.63186	0.62718	0.62347
Pow	0.64501	0.63489	0.64804	0.64795	0.64666	0.64055	0.64362
PR	0.63650	0.61435	0.64358	0.63360	0.63342	0.63610	0.63221
PRC	0.63632	0.63173	0.62032	0.66297	0.65107	0.63917	0.64105
RF	0.65562	0.63614	0.59924	0.63623	0.66907	0.61096	0.63033
RFW	0.63641	0.64091	0.60187	0.64488	0.65553	0.64492	0.63762
S2N	0.64519	0.61854	0.61555	0.64938	0.64661	0.62130	0.63028
SVM	0.65709	0.61065	0.59327	0.62865	0.59447	0.62398	0.61021
SU	0.63957	0.60062	0.62883	0.62438	0.63333	0.63632	0.62470
χ^2	0.64372	0.61979	0.60802	0.63329	0.63641	0.62313	0.62413
AVG	0.64455	0.61769	0.62005	0.63898	0.63789	0.62740	-

Table III
R VS PD TOP RESULTS

Filter	Accuracy						Average
	RFT	5-NN	LR	MLP	NB	SVM	w/o RFT
AUC	0.74623	0.72031	0.71846	0.71062	0.72800	0.70246	0.71597
Dev	0.74031	0.73269	0.72038	0.72638	0.71600	0.72646	0.72438
F	0.73831	0.69862	0.69269	0.71446	0.72215	0.67446	0.70048
GM	0.75038	0.71062	0.70477	0.71262	0.71638	0.68885	0.70665
GI	0.74038	0.72446	0.72854	0.73869	0.72831	0.73454	0.73091
GR	0.74223	0.72269	0.74638	0.73246	0.73246	0.74246	0.73529
IG	0.73638	0.72077	0.71446	0.72469	0.71808	0.72485	0.72057
KS	0.73031	0.71454	0.69692	0.72477	0.71415	0.67900	0.70588
MI	0.73431	0.72462	0.70862	0.72462	0.72431	0.70485	0.71740
OR	0.75023	0.74438	0.71638	0.74646	0.73031	0.74046	0.73560
Pow	0.75408	0.74262	0.75408	0.74246	0.74023	0.75800	0.74748
PR	0.74823	0.72254	0.73054	0.73869	0.73231	0.73862	0.73254
PRC	0.74638	0.74454	0.72454	0.72254	0.72646	0.71662	0.72694
RF	0.75646	0.72262	0.69631	0.72246	0.73846	0.71054	0.71808
RFW	0.75054	0.73854	0.72454	0.72446	0.73062	0.71654	0.72694
S2N	0.74623	0.70431	0.70246	0.72862	0.72254	0.71046	0.71368
SVM	0.74238	0.72846	0.63108	0.72254	0.68115	0.63292	0.67923
SU	0.73038	0.73285	0.74838	0.74238	0.72631	0.75246	0.74048
χ^2	0.73838	0.72092	0.71285	0.72877	0.72792	0.72038	0.72217
AVG	0.74327	0.72479	0.71434	0.72783	0.72401	0.71447	-

Table IV
AVERAGE ACCURACY OF THREE DIFFERENT NUMBERS OF FEATURES

Features	R vs. NR	R vs. PD
5	0.59303	0.69103
200	0.59722	0.68309
1000	0.61298	0.69812

When using the R vs NR and R vs. PD datasets, Random Forest outperforms the average of the other learners when using their top performers. However, in order to further test the performance of feature rankers used in conjunction with the six learners, we choose a range of feature subset sizes to test the rankers when the feature subset size is kept static. We chose a small feature subset size (5), a medium feature subset size (200) and a large feature subset size (1000). Table IV shows the average accuracy across all filters and learners for each of the chosen feature subset sizes. In both R vs. NR and R vs. PD, the top feature subset size is 1000. As we can see from Tables V and VI, when using 1000 features and the Random Forest learner in both the R vs. NR and R vs. PD datasets, we have the largest average across all nineteen rankers when compared to the other learners with 0.63503 and 0.73886 respectively. This shows that 1000 is the ideal feature subset size, as well as showing that 1000 combined with Random Forest produces optimum results.

To further validate our results, we performed statistical analysis on the data. First, a Z-test was performed for each learner, testing the null hypothesis that "choosing the top 1000 features leads to Accuracy values statistically indistinguishable from the best results." We choose 1000 features based on our results from Tables IV through VI which state that 1000 is the optimum feature subset size. The P values derived from the Z-test are presented in Table VII, for both the R vs NR and R vs PD datasets. These are the probabilities of observing the given group means if the null hypothesis were true. For both Random Forest and MLP, the P values are above 5%, a standard value used to create 95% confidence intervals, showing that with either learner, it is safe to simply choose the top 1000 features and trust that these will be equivalent to choosing the best number of features.

In addition, a two-way ANOVA (ANalysis Of VAriance) test was performed to determine the effect of the choice of ranker and learner on the Accuracy levels [24]. The ANOVA analaysis was performed within MATLAB. Here, factor A was the choice of ranker and factor B was the choice of learner. The results are presented in Tables VIII through XI, showing the statistics for the two factors (and the interaction term) for both datasets and both choices of feature size (top 1000 or best). Note that in all four tables, factor B has a Prob>F value less than 0.05, indicating that this factor is significant; this means that two or more of

Table V
RESULTS: R VS NR ACCURACY AT 1000 FEATURES

Filter	RFT	5-NN	LR	MLP	MB	SVM
AUC	0.62879	0.60802	0.58877	0.64666	0.63935	0.63021
Dev	0.63775	0.61114	0.56952	0.65258	0.62460	0.62161
F	0.63053	0.52219	0.56658	0.64073	0.60971	0.61693
GM	0.64528	0.61257	0.59933	0.62571	0.62754	0.60660
GI	0.62888	0.60472	0.54135	0.61395	0.59479	0.58587
GR	0.64675	0.59906	0.56386	0.61725	0.61854	0.59630
IG	0.62313	0.60660	0.55058	0.62304	0.60971	0.59318
KS	0.64091	0.61110	0.57531	0.63605	0.63053	0.61662
MI	0.62451	0.59345	0.55049	0.64376	0.63494	0.61422
OR	0.62322	0.58703	0.56346	0.63320	0.62304	0.62718
Pow	0.64501	0.62883	0.56234	0.63953	0.61863	0.61092
PR	0.62010	0.58596	0.56074	0.63360	0.59336	0.61257
PRC	0.63632	0.63173	0.58271	0.62879	0.65107	0.60775
RF	0.65562	0.62865	0.57237	0.63623	0.63770	0.61096
RFW	0.63641	0.62585	0.60187	0.64488	0.65388	0.64492
S2N	0.63783	0.60655	0.59193	0.64938	0.63053	0.62130
SVM	0.63182	0.61065	0.58436	0.61685	0.59447	0.62398
SU	0.63641	0.59902	0.55655	0.62438	0.61711	0.59911
χ^2	0.63641	0.60209	0.55793	0.63021	0.61716	0.59465
AVG	0.63503	0.60396	0.57053	0.63351	0.62246	0.61236

Table VI
RESULTS: R VS PD ACCURACY AT 1000 FEATURES

Filter	RFT	5-NN	LR	MLP	MB	SVM
AUC	0.74623	0.71462	0.71846	0.71062	0.72800	0.61085
Dev	0.74031	0.73038	0.68085	0.70485	0.71600	0.63485
F	0.73831	0.67862	0.69269	0.71446	0.72215	0.62331
GM	0.75038	0.70054	0.70477	0.71262	0.71638	0.62915
GI	0.73238	0.68469	0.70623	0.72877	0.65623	0.63900
GR	0.73231	0.71438	0.68492	0.72062	0.72415	0.63300
IG	0.73638	0.72038	0.66885	0.72469	0.71808	0.62700
KS	0.72846	0.71454	0.69692	0.71669	0.71415	0.61915
MI	0.73231	0.72238	0.69085	0.72462	0.71015	0.63315
OR	0.75023	0.70838	0.69062	0.74646	0.70046	0.65685
Pow	0.75231	0.72862	0.68669	0.72669	0.67254	0.64685
PR	0.73231	0.68285	0.69238	0.73269	0.65615	0.64485
PRC	0.74623	0.73254	0.68662	0.72085	0.71223	0.65500
RF	0.74246	0.72069	0.64092	0.70477	0.63485	0.61908
RFW	0.74238	0.73854	0.67292	0.71662	0.67300	0.62715
S2N	0.74238	0.69277	0.69254	0.70085	0.72015	0.62277
SVM	0.73646	0.72846	0.61546	0.72254	0.67685	0.63292
SU	0.72423	0.72438	0.68700	0.71285	0.72400	0.63100
χ^2	0.73231	0.71638	0.69477	0.72877	0.72792	0.62885
AVG	0.73886	0.71338	0.68445	0.71953	0.70018	0.63236

Table VII
RESULTS: P VALUES FOR COMPARING BEST AND TOP 1000 FEATURES

Learner	R vs NR	R vs PD
RFT	0.052623	0.151508
5-NN	0.007641	0.010792
LR	0	6.799896×10^{-8}
MLP	0.142611	0.057877
NB	0.003673	0.000006
SVM	0.001981	0

the populations within factor B (in this case, learners) have significantly different scores from one another. In two of the tables (Tables IX and X), factor A has a Prob>F value less than 0.05 which indicates that the factor is significant for these two scenarios. However, in the other two tables the values are greater than 0.05 and therefore insignificant. This suggests that the influence of factor A (the ranker) is marginal at best, even when not specifically examining the Random Forest results. The results from the interaction term (A $\times$ B) further lend credence to this; as none of these are significant, there are no major distinctions between how the rankers fare for one learner versus how they do with a different learner.

We performed Tukey's Honestly Significantly Different (HSD) test to further investigate these results [24]. As with the ANOVA analysis we performed the Tukey HSD within MATLAB. Here, group means are represented as circles, while the 95% confidence intervals around those means are represented with lines. Two values are statistically distinguishable if their lines do not overlap, and are indistinguishable if they do. As can be seen from Figures 1 through 4, for the R vs PD dataset, Random Forest was significantly better than the other learners, while for the R vs NR dataset, Random Forest still showed the best performance, although not in a statistically significant fashion. This further justifies the use of Random Forest as a learner for performing patient response prediction from

Table VIII
RESULTS: ANOVA FOR R VS NR, BEST NUMBER OF FEATURES

Source	Sum Sq.	d.f.	Mean Sq.	F	Prob>F
A	0.08222	18	0.00457	0.75814	0.75136
B	0.22974	5	0.04595	7.62600	4.09E-07
A$\times$B	0.28423	90	0.00316	0.52415	0.99993
Error	13.05080	2166	0.00603		
Total	13.64700	2279			

Table IX
RESULTS: ANOVA FOR R VS NR, TOP 1000 FEATURES

Source	Sum Sq.	d.f.	Mean Sq.	F	Prob>F
A	0.24431	18	0.01357	2.33280	0.00120
B	1.09500	5	0.21900	37.64120	0
A$\times$B	0.32734	90	0.00364	0.62514	0.99767
Error	12.60200	2166	0.00582		
Total	14.26870	2279			

DNA microarray datasets: not only is it safe to use the top 1000 features without careful testing to determine the correct size of feature subset, its results are better than those of other commonly-used learners. In addition, ANOVA results for both factor A alone and for the interaction term (not presented due to space considerations) confirm the lack of statistical significance of choosing one ranker over another, especially when using the Random Forest learner. Although there were two rankers with significantly different means, for the two cases where the ANOVA results for the interaction term showed significance, this distinction disappeared when considering only the Random Forest learner. This further shows that when choosing Random Forest as a learner, the other choices (size of feature subset and choice of ranker) either can be made in advance or do not matter.

VI. CONCLUSIONS

Random Forest has proven to be an effective learner when predicting patient response to the drug bortezomib. In both the R vs NR and R vs PD experiments, the majority of the filters created feature sets which allowed Random Forest

Table X
RESULTS: ANOVA FOR R VS PD, BEST NUMBER OF FEATURES

Source	Sum Sq.	d.f.	Mean Sq.	F	Prob>F
A	0.39252	18	0.02181	4.21910	6.54E-09
B	0.21553	5	0.04311	8.34000	8.04E-08
A$\times$B	0.36639	90	0.00407	0.78765	0.92860
Error	11.19510	2166	0.00517		
Total	12.16950	2279			

Table XI
RESULTS: ANOVA FOR R VS PD, TOP 1000 FEATURES

Source	Sum Sq.	d.f.	Mean Sq.	F	Prob>F
A	0.13686	18	0.00760	1.40510	0.11852
B	2.60950	5	0.52190	96.44600	0
A$\times$B	0.58777	90	0.00653	1.20690	0.09371
Error	11.72080	2166	0.00541		
Total	15.05490	2279			

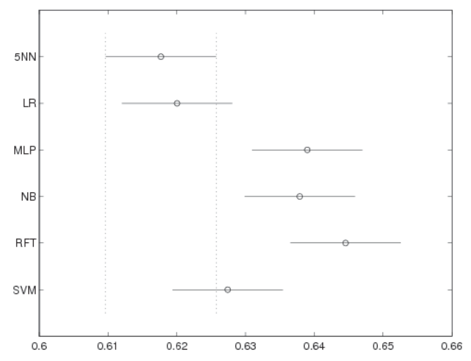

Figure 1. Tukey's HSD: R vs NR Best # of Features

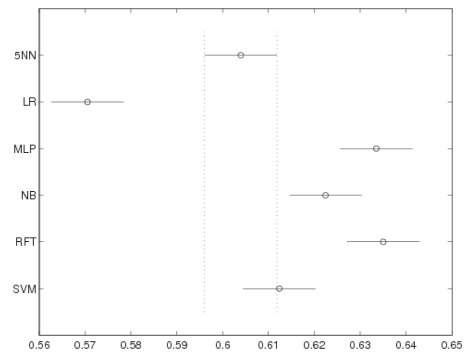

Figure 2. Tukey's HSD: R vs NR 1000 Features

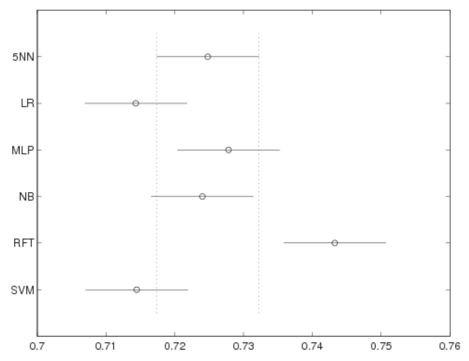

Figure 3. Tukey's HSD: R vs PD Best # of Features

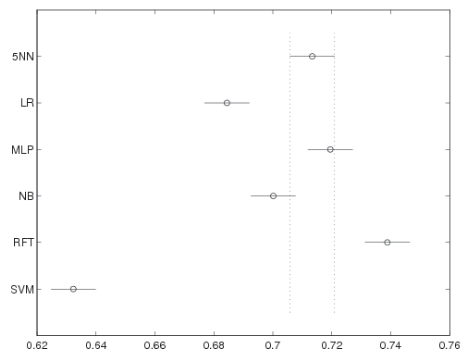

Figure 4. Tukey's HSD: R vs PD1000 Features

to outperform the top performers of the other five learners when using the same filter. This is also borne out in the Tukey results. In R vs PD, Random Forest is significantly better than the other learners. In R vs NR, Random Forest is the top learner and is significantly different from a majority of the other learners, but is not significantly different from all of them.

In addition to the performance, there were two other trends of note. The first trend is that of the nineteen filters, very few are significantly different from the others for any of the learners. The ANOVA results show that the choice of filter (factor A) was only significant in half of the cases (R vs NR using 1000 features and R vs PD using the best-performing feature set size), and even there, the ANOVA results demonstrate that only two filters were significantly different from one another. In addition, as long as the number of features is high (but there is some sort of feature selection), the results from nineteen different filters used with Random Forest will be on average higher than any other learner and feature subset size combination. In both R vs NR and R vs PD, the average across all nineteen filters was the highest, with an average of 0.64455 and 0.74327 respectively. When looking at only using 1000 features, the averages remain the highest with 0.63503 for R vs NR and 0.73886 for R vs PD.

The second trend is that for two of the learners, Random Forest and MLP, the choice of feature size 1000 is equivalent to choosing the best feature size. This was demonstrated with the results of the Z-test, where no statistical difference could be found between using either 1000 features or the best choice of feature subset size. The reasons for this would seem to be that Random Forest benefits from a larger number of features to choose from, so the constituent learners used to create the ensemble will be more diverse. MLP, on the other hand, is known for being good at disregarding irrelevant features; thus, the larger feature subset sizes ensure the learner has all of the important features, and if unimportant features are added they do not impact performance. Although these two learners showed similar results in terms of optimal feature subset size, the greater performance of Random Forest overall makes it a better choice of learner.

Due to our findings, we can state that Random Forest will be a successful classifier if certain conditions are met. The first is that the filter chosen is largely irrelevant, in that the difference between the top results is small when compared to the five other learners. Second is that Random Forest does well with larger numbers of features, around 200 to 1000. This may be contrary to previous research which states that the optimum number of features used in a majority of scenarios should fall between 10 and 100 inclusively [10]. However, we believe that the need for a large amount of diversity when building the ensemble model requires the larger feature set. Still, feature selection is necessary in order to reduce the 22,000 features down to 1000. Our recommendation for using Random Forest is as follows: If one wishes to have accurate classification with little risk and does not need to use the feature set afterwards, then Random Forest combined with a filter based feature selection technique set to choose 1000 features is ideal.

The future for testing the utility of Random Forest in the areas of patient response require one thing: more data. Unfortunately, a large majority of the patient response datasets are not available in the public domain. Nonetheless, future work will examine these sets, both those concerning the same cancer and drug (multiple myeloma and bortezomib, respectively) and additional patient response datasets. Random Forest has the potential to help bring genetic prediction of patient response closer to clinical use.

REFERENCES

[1] G. Mulligan, C. Mitsiades, B. Bryant, F. Zhan, W. J. Chng, S. Roels, E. Koenig, A. Fergus, Y. Huang, P. Richardson, W. L. Trepicchio, A. Broyl, P. Sonneveld, J. Shaughnessy, John D., P. Leif Bergsagel, D. Schenkein, D.-L. Esseltine, A. Boral, and K. C. Anderson, "Gene expression profiling and correlation with outcome in clinical trials of the proteasome inhibitor bortezomib," *Blood*, pp. 3177–3188, 2007.

[2] J. Van Hulse, T. M. Khoshgoftaar, A. Napolitano, and R. Wald, "Feature selection with high dimentional imbalanced data," in *Proceedings of the 9th IEEE International Conference on Data Mining - Workshops (ICDM'09)*. Miami, FL: IEEE Computer Society, December 2009, pp. 507–514.

[3] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2nd edition, 2005.

[4] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[5] R. Diaz-Uriarte and S. Alvarez de Andres, "Gene selection and classification of microarray data using random forest," *BMC Bioinformatics*, vol. 7, pp. 1–13, 2006.

[6] M. A. Hall and L. A. Smith, "Feature selection for machine learning: Comparing a correlation-based filter approach to the wrapper," in *Proceedings of the Twelfth International Florida Artificial Intelligence Research Society Conference*, May 1999, pp. 235–239.

[7] M. A. Hall and G. Holmes, "Benchmarking attribute selection techniques for discrete class data mining," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 6, pp. 392–398, November/December 2003.

[8] I. Kononenko, "Estimating attributes: Analysis and extensions of relief," *Lecture Notes in Computer Science*, pp. 171–182, 1994.

[9] H. Wang, T. M. Khoshgoftaar, and A. Napolitano, "An empirical study of software metrics selection using support vector machine," in *Proceedings of International Conference on Software Engineering and Knowledge Engineering SEKE'11*, July 7-9, 2011, pp. 83–88.

[10] M. Wasikowski and X. wen Chen, "Combating the small sample class imbalance problem using feature selection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, pp. 1388–1400, 2010.

[11] D. Dittman, T. Khoshgoftaar, R. Wald, and J. Van Hulse, "Comparative analysis of dna microarray data through the use of feature selection techniques," in *Proceedings of the Ninth International Conference on Machine Learning and Applications (ICMLA)*. ICMLA, 2010, pp. 147 –152.

[12] N. Seliya, T. M. Khoshgoftaar, and J. Van Hulse, "A study on the relationships of classifier performance metrics," in *Proceedings of the 21^{st} IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2009)*, Newark, NJ, November 2009, pp. 59–66.

[13] W. J. Conover, *Practical Nonparametric Studies*. John Wiley and Sons, 2nd edition, 1971.

[14] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification And Regression Trees*. Chapman and Hall, 1993.

[15] G. Forman, "An extensive empirical study of feature selection metrics for text classification," *J. Mach. Learn. Res.*, pp. 3:1289–1305, 2003.

[16] T. M. Khoshgoftaar, E. B. Allen, and J. Deng, "Using regression trees to classify fault-prone software modules," *IEEE Transaction on Reliability*, vol. 51, no. 4, pp. 455–462, December 2002.

[17] R. Battiti, "Using mutual information for selecting features in supervised neural net learning," *IEEE Transactions On Neural Networks*, vol. 5, no. 4, pp. 537–550, July 1994.

[18] National Center for Biotechnology Information, "Microarrays factsheet," 2007, http://www.ncbi.nlm.nih.gov/About/primer/microarrays.htm. [Online]. Available: http://www.ncbi.nlm.nih.gov/About/primer/microarrays.html

[19] T. M. Khoshgoftaar, M. Golawala, and J. Van Hulse, "An empirical study of learning from imbalanced data using random forest," *Tools with Artificial Intelligence, IEEE International Conference on*, pp. 310–317, 2007.

[20] T.-Y. Liu, "Easyensemble and feature selection for imbalance data sets," *Bioinformatics, Systems Biology and Intelligent Computing, International Joint Conference on*, pp. 517–520, 2009.

[21] J. Souza, N. Japkowicz, and S. Matwin, "Stochfs: A framework for combining feature selection outcomes through a stochastic process," in *Knowledge Discovery in Databases: PKDD 2005*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2005, vol. 3721, pp. 667–674.

[22] S. Le Cessie and J. C. V. Houwelingen, "Ridge estimators in logistic regression," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, pp. 191–201, 1992.

[23] S. Haykin, *Neural Networks: A Comprehensive Foundation 2nd edition*. Prentice Hall, 1998.

[24] M. L. Berenson, M. Goldstein, and D. Levine, *Intermediate Statistical Methods and Applications: A Computer Package Approach 2nd Edition*. Prentice Hall, 1983.