SUPPLEMENTAL MATERIAL

Supplemental Methods

Handling collinearity, model selection, and using cross-validation

To avoid collinearity problems arising from adding the highly correlated cytokine levels into the model, we transform the cytokine data using the method of independent components analysis [1] (ICA) and calculate the latent, independent factors. We ensure that the variables used in building the logistic regression model will be statistically independent of one another, and the coefficients can be interpreted as logarithms of odds ratios. ICA is a computational method for refactoring multivariate data into additive subcomponents that are mutually statistically independent. The factors calculated by ICA are linear combinations of the original measurements, and can be interpreted as hidden (independent) factors that underlie the correlated original measurements. Unlike principal components analysis, which assumes data is distributed normally, ICA makes no distributional assumptions, and is therefore more effective than PCA for our problem. We then use L2-penalized logistic regression with stepwise variable selection [2] to construct the final model. The predictive power of the model is measured using the C-statistic, or the area under the receiver operating characteristic curve. Since we do not have an independent hold-out set, we estimate out-of-sample performance of a model using *k*-fold cross validation [3,4]. Cross-validation is a robust and widely used technique in statistics for estimating performance of predictive models in contexts where there is no readily available source of independent data for testing models. Its theoretical robustness properties are detailed in Picard and Cook [3]. In *k*-fold cross-validation, the original sample is randomly partitioned into k equal-sized, non-overlapping subsamples. Of the *k* subsamples, a single subsample is retained for testing the model, and the remaining *k*-1 subsamples are used as training data. *k* models are trained, with each of the *k* subsamples used exactly once for validation. Results from the k models are averaged to estimate mean and standard deviation of the model performance measure. Usual choices for *k* range from five to ten, and we use ten-fold cross-validation for all of our analysis. Ten-fold cross-validation was also used to estimate model performance in ADHERE.

Learning ensemble models from data: the AdaBoost algorithm

We use a variation of the classic AdaBoost algorithm [5], called gentle boosting [6], to learn an ensemble model. Boosting associates weights with each training example, which signify its importance in classification. Initially, all the training data are weighted equally, and AdaBoost learns a classifier G_1 for the initial data. For each successive round, the training data are re-weighted. If a data point is incorrectly classified, then it receives more weight in the next round. Correctly classified data points receive less weight in the next round. Thus as rounds proceed, training data points that are difficult to correctly classify receive ever-increasing importance. Each successive classifier is forced to concentrate on those training data that are missed by the previous classifiers in the sequence. Thus, AdaBoost constructs a sequence of simple weighted classifiers, each forced to learn a different aspect of the data, to generate an ensemble classifier.

In our model, we use decision stumps (one-level decision trees) for classification. Gentle boosting produces a weighted sequence of K decision stumps. The final decision produced on each new case is the weighted majority vote among these K decision stump classifiers. As with any method, ensembles have advantages and disadvantages. They can achieve near perfect classification accuracy even on tough problems. However, they are difficult to explain because one typically needs 50 to 100 members in the ensemble to achieve good accuracies. Each model can be represented by a single equation; however the complete model consists of 50 to 100 equations, based on the size of the ensemble. Overfitting can be a serious problem, and so we use a regularization parameter to control it [6], so that out of sample prediction with these classifiers is as good as on the training sample.

Supplemental Tables

Definitions of the latent factors in the ICA analysis

The factors in Table 4 in the main body of the paper are defined as follows.

X2 = 0.013 log TNF - 0.009 log sTNFR1 - 0.014 log sTNFR2 - 0.061 log IL6 + 0.007 log sIL6R - 0.025 wk08 log TNF - 0.086 wk08 log sTNFR1 - 0.109 wk08 log sTNFR2 - 0.089 wk08 log IL6 - 0.029 wk08 log sIL6R + 0.443 wk16 log TNF + **2.006 wk16 log sTNFR1** + **2.212 wk16 log sTNFR2** + 0.354 wk16 log IL6 + 0.96 wk16 log sIL6R + 0.491 wk24 log TNF + **2.198 wk24 log sTNFR1** + **2.411 wk24 log sTNFR2** + 0.447 wk24 log IL6 + 1.037 wk24 log sIL6R

X3 = 0.022 log TNF + 0.055 log sTNFR1 + 0.048 log sTNFR2 + 0.029 log IL6 + 0.018 log sIL6R + 0.478 wk08 log TNF +**2.131 wk08 log sTNFR1** + **2.337 wk08 log sTNFR2** + 0.409 wk08 log IL6 1.106 wk08 log sIL6R + 0.013 wk16 log TNF + 0.099 wk16 log sTNFR1 + 0.086 wk16 log sTNFR2 – 0.209 wk16 log IL6 + 0.037 wk16 log sIL6R + 0.023 wk24 log TNF + 0.017 wk24 log sTNFR1 -0.028 wk24 log sTNFR2 + 0.004 wk24 log IL6 – 0.009 wk24 log sIL6R

X5 = -0.059 log TNF – 0.135 log sTNFR1 -0.098 log sTNFR2 **– 0.595 log IL6** -0.008 log sIL6R - 0.073 wk08 log TNF - 0.149 wk08 log sTNFR1 - 0.138 wk08 log sTNFR2 **- 0.598 wk08 log IL6** - 0.011 wk08 log sIL6R – 0.092 wk16 log TNF - 0.161 wk16 log **sTNFR1 – 0.603 wk16 log sTNFR2** - 0.039 wk16 log IL6 - 0.084 wk16 log sIL6R – 0.084 wk24 log TNF – 0.201 wk24 log sTNFR1 -0.084 wk24 log sTNFR2 **-0.594 wk24 log IL6** – 0.029 wk24 log sIL6R

The same information is presented in tabular form below.

The coefficients of the latent factors X2, X3 and X5 in tabular form

	log TNF	**log sTNFR1**	**log sTNFR2**	**log IL6**	**log sIL6R**
X2	0.013	-0.009	-0.014	-0.061	0.007
X3	0.022	0.055	0.048	0.029	0.018
X5	-0.059	-0.135	-0.098	**-0.595**	-0.008

	wk08	**wk08**	**wk08**	**wk08**	**wk08**
	log TNF	**log sTNFR1**	**log sTNFR2**	**log IL6**	**log sIL6R**
X2	-0.025	-0.086	-0.109	-0.089	-0.029
X3	0.478	**2.131**	**2.337**	0.409	1.106
X5	-0.073	-0.149	-0.138	**-0.598**	-0.011

	wk16 **log TNF**	**wk16** **log sTNFR1**	**wk16** **log sTNFR2**	**wk16** **log IL6**	**wk16** **log sIL6R**
X2	0.443	**2.006**	**2.212**	0.354	0.96
X3	0.013	0.099	0.086	-0.029	0.037
X5	-0.092	-0.161	**-0.603**	-0.039	-0.084

	wk24 **log TNF**	**wk24** **log sTNFR1**	**wk24** **log sTNFR2**	**wk24** **log IL6**	**wk24** **log sIL6R**
X2	0.491	**2.198**	**2.411**	0.447	1.037
X3	0.023	0.017	-0.028	0.004	-0.009
X5	-0.084	-0.201	-0.084	**-0.594**	-0.029

Supplemental References

1. Pierre C. Independent Component Analysis: a new concept? Signal Processing. 1994; 36:287-314.

2. Park MY and Hastie T. Penalized logistic regression for detecting gene interactions. Biostatistics. 2008;9:30-50.

3. Picard R and Cook D. Cross-validation of regression models. Journal of the American Statistical Association. 1984;79:575-583.

4. Kohavi R. A study of cross-validation and bootstrap for accuracy estimation and model selection. Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence. 1995;1:1137-1143.

5. Schapire R. The boosting approach to machine learning. In Denison DD, Hansen MH, Holmes C, Mallick B and Yu B (editors), Non-linear estimation and classification. Springer, Berlin, 2003.

6. Friedman J, Hastie T and Tibshirani R. Additive logistic regression: a statistical view of boosting. The Annals of Statistics. 2000;28:337-407.