Predicting Binary Outcomes for Fragile Families

Mack Lee
Princeton University
mackl@princeton.edu

Abstract

The Fragile Families Challenge aims to use data collected from questions answered by members of "fragile families" (unmarried parents and their children) to help predict six different outcomes. In this assignment, we build a model to try and accurately predict the three binary outcomes (eviction, layoff, and job training). We explore six different classifiers, the effects of imputation methods and feature selection, and evaluate their performance using stratified 10-fold cross-validation on the given data set before submitting to the FFC platform which tests predictions on a held-out test set. We find random forests to be the most accurate and robust classifier for Layoff and Job Training and logistic regression to be the best for Eviction based off of accuracy and confidence intervals obtained by bootstrapping, precision/recall, and AUC scores whereas Naive Bayes performed significantly worse for all three outcomes. We found that the most important features related to hometown, social relationships, and academic performance.

1 Introduction

The Fragile Families and Child Wellbeing Study has collected data of nearly 5,000 American children over the course of 15 years, most of whom were born to unmarried parents. The Fragile Families Challenge seeks to improve the lives of disadvantaged children in these "fragile families" using predictive modeling, causal inference, and in-depth interviews [1]. The objective of the challenge is to use the data collected on these families from birth to year 9 to predict six outcomes (grit, GPA, material hardship, eviction, layoff, and job training) in the year 15 test data. This assignment aims to build models specifically for the three binary outcomes: layoff, eviction, and job training. More specifically, we will answer the following questions:

1. What imputation methods yield the best results?
2. Which classifiers and hyper-parameters perform the best? More complex classifiers?
3. How confident can we be with our predictions? (Ex. Bootstrap, Cross-Validation)
4. How can we improve the performance and computational time of classifiers and imputation methods? (Ex. Feature Selection, Variance Threshold)
5. Which features have the most predictive value?
6. How well do our models predict the 3 binary outcomes? (Submitting to FFC)

2 Related Work

Classification, particularly binary classification, is a common problem in machine learning that applies to many fields. Researchers have compared the performance of several different classifiers including Naive Bayes, Logistic Regression, K-nearest-neighbors, Random Forests, and Decision Trees [5]. More complex models like Gaussian processes have also been studied and found to be accurate estimators[14] [17]. When considering large data sets, missing data is very common. There are two different approaches to handling this: deleting the data or replacing/imputing the missing value[12]. The two most common techniques to deleting data is "listwise" and "pairwise" when considering missing values that are MCAR (missing completely at random)- listwise removes all

data that has one or more missing values whereas pairwise uses correlation between variables [16]. Michy first uses a threshold to remove data with more than a certain percentage of data missing before imputing values [11]. Several methods have been developed to impute missing data including mean imputation, regression, and matching [10]. Shrive found that using multiple imputation methods was the most accurate on their target data set compared to single imputation and regression methods [19]. A study done by Ambler and Omar conducted a comparison of multiple imputation methods using MICE and several single imputation methods, concluding that mean imputation was the superior single imputation method and MICE performed the best among multiple imputation methods [2]. Feature selection is also a common technique to decrease the complexity and dimensionality of classifiers [9]. Vaghela found varied accuracy when feature selection was applied to different classifiers and datasets but all improved computational speed [20]. For measuring prediction error, Borra and Di Ciaccio found that a 10-fold cross-validation and Parametric Bootstrap estimator to have the highest performance in their simulations[3].

2.1 Data processing

For the background data set consisting of the answers of 4242 children to 12944 survey questions, we start by removing the questions in which more than 10% of the data were missing (all negative values were treated as missing) as suggested by Michy [11]. We decided to remove features based off of a threshold because they contain too little data to help accurately impute missing values based on results. Reducing the number of features allowed us to attempt imputation methods that required more computational power and avoid the curse of dimensionality when training our model. We perform this preprocessing step under the assumption that data are missing at random (MAR) - missing values have no predictive power on whether a value is missing. We make this assumption to keep imputation methods within the scope of our computational resources. We believe that a missing value is independent of other missing data. The threshold of 10% was selected based off relative performance of our models using different imputation methods. Since classification methods in Pythons SciKitLearns libraries require exclusively numerical data, we used SciKitLearns LabelEncoder to convert categorical features to numerical data[15]. Features with a mix of numerical and categorical data were treated as numerical data.

We compared the performance of different single and multiple imputation methods. We used Python's SciKitLearn's preprocessing libraries to implement mode, mean, and median imputation. We used R's MICE package to implement Regression imputation (Bayesian Linear Regression) and predictive mean matching [21]. We used MATLAB to implement a k-nearest-neighbor matching method. We compared various feature reduction techniques including feature selection and using a variance threshold.

2.2 Classification methods

We use eight different classification methods from the SciKitLearn Python libraries [15]. All parameterizations are the default unless specified. Several of the parameters were tuned to perform better for the given dataset.

1. *Naive Bayes classifier* (NB): using multinomial implementation
2. *Logistic regression with ℓ_2 penalty* (LR): using gradient descent and regularization
3. *K-nearest neighbors* (KNN): using five nearest neighbors and the "KDTree" algorithm
4. *Random forest* (RF): using Gini impurity scores, 100 trees, and max depth of 10
5. *Decision tree* (DT): using Gini impurity scores
6. *Gaussian process* (GP): using Laplace approximation

We want to use a mix of linear methods(NB, LR), ensemble learning methods (RF, DT), non-parametric methods (KNN), and more complex methods (GP). NB and LR provide a discriminative and generative classifier which could have different results on our dataset. We decided to use KNN because it is the one of the simplest classifiers (Occam's razor).

2.3 Evaluation

Since we lack a split of training data and test data, we used stratified 10-fold cross validation to determine the optimal imputation methods and classifiers and to tune hyper-parameters. The same

folds were used across all classifiers to maintain fairness and shuffled folds to randomize the distribution of the binary classes across folds. We use the mean accuracy of the folds to measure the performance of the imputation methods and classifiers for initial model selection. For now, we use accuracy to determine generally how well a classifier performs on a data set. Once we have determined which classifiers perform the best for each outcome in general and remove the inferior ones, we conduct further evaluation based on precision, recall, F1-score, and wall clock time to build our model for prediction [18]. We use other measures to gain insight into what types of misclassification models tend to make. Precision, recall, and F_1 score are defined as follows:

$$\text{precision} = \frac{TP}{TP + FP}, \text{recall} = \frac{TP}{TP + FN}, F_1 = 2\frac{precision \times recall}{precision + recall}$$

where TP = # true positives, FP = # false positives, FN = # false negatives, TN = # true negatives[7].

We also plot the ROC curves to provide another metric of the performance using the AUC score. Once we have built our model, we predict on our test set and use a simple Leave-One-Out bootstrap method to calculate a confidence interval for the accuracies of the classifiers. This allowed us to create an estimate of the true prediction error before submitting to the FFC platform.

3 Methods

3.1 Spotlight classifier: Logistic Regression

The Logistic Regression classifier is the "go-to" linear classifier for binary classification problems [4]. It is one of the most common probabilistic classifiers (meant to choose y that maximizes the posterior $Pr(y|x)$)[8]. It is often compared to the Naive Bayes method in that Logistic Regression is a discriminative where as Naive Bayes is generative (LR is considered to be the discriminative analog of NB) [13]. Naive Bayes is a generative classifier because it indirectly estimates y using the likelihood $Pr(x|y)$ and the prior $Pr(y)$ whereas Logistic Regression computes a likelihood for the posterior $Pr(y|x)$ (and selects class with the the highest probability) directly as follows [8]:

$$Pr(y_i|x_i) \sim Bernoulli(\mu(B^T x_i))$$

where the covarites enter the probability of the response through a linear combination with the coefficients. This linear combination is passed through μ to be appropriate as a parameter for the distribution of the response [6]. To compute the intercept β_0, we find where $\beta_0 = -\beta^T x_i$, or where:

$$Pr(y_i = 1|x_i) = \frac{1}{1 + e^{-(\beta_0 + \beta^T x_i)}} = \frac{1}{2}$$

or where

$$(\beta_0 + \beta^T x_i) = 0$$

The coefficients can be represented by the following:

$$\log \frac{Pr(y_i = 1|x_i)}{1 - Pr(y_i = 1|x_i)} = \beta^T x_i$$

which is an odds ratio (probability of success over the probability of failure) [6]. Logistic regression can be extended to the multivariate case where:

$$Pr(y_i = 1|x_i) = \frac{1}{1 + e^{-\sum_{j=1}^{p} \beta_j^T x_{i,j}}}$$

We fit the logistic regression models by maximizing the conditional likelihood:

$$\hat{\beta} = \arg\max_{\beta} \sum_{i=1}^{n} \log Pr(y_i|x_i, \beta)$$

by applying calculus to the objective function L where L_i is the log likelihood of the ith data point:

$$L = \sum_{i=1}^{n} y_i \log \mu(\beta^T x_i) + (1 - y_i) \log(1 - \mu(\beta^T x_i))$$

We can also regularize logistic regression to produce an efficient way to find a sparse solution. Since Logistic Regression is a linear classifier, data must be linearly separable to do well [13].

4 Results

4.1 Evaluation results: imputation

Removing features with a percentage of missing data below a certain threshold improved classifier performance. We saw an overall increase in accuracy as we lowered the threshold as seen in Table 1. Decreasing the number of features allowed us to compare more complex imputation methods such as pmm and regression.

Threshold	# features	imp time	LR	NB	KNN	RF	DT	GP
<50% missing	3529	0.174	0.74	0.46	0.74	0.79	0.64	0.79
<25% missing	1014	0.042	0.77	0.47	0.75	0.79	0.66	0.79
<10% missing	188	0.007	0.79	0.44	0.76	0.79	0.68	0.79
<5% missing	178	0.007	0.79	0.44	0.76	0.79	0.65	0.79

Table 1: **Mean accuracy for six classifiers on Layoff with varying missing data thresholds.** For each threshold, we report the mean accuracy using 10-fold stratified cross-validation for layoffs using mode imputation.

We used accuracy to determine the best threshold since we are primarily interested in the error rate, not the type of errors. From Table 1, The thresholds of 10% and 5% were the most effective in improving accuracy for all three outcomes. It is clear that NB and DT methods perform considerably worse than the other classifiers so we decided not to consider them in further analyses. A possible explanation for why NB performs badly on the data set is because of high correlation between features, making the independence assumption inaccurate. The problems in the questionnaire are related and are even separated into categories (questions relating to the child's mother vs. the child's teacher). We continue to explore different imputation methods using a threshold of $<25\%$ missing data to allow for better comparison.

Method	Time	LR	NB	KNN	RF	DT	GP
mode	0.372	0.758	0.475	0.786	0.791	0.647	0.791
mean	0.350	0.789	0.480	0.789	0.791	0.666	0.791
median	0.423	0.761	0.479	0.788	0.791	0.676	0.791
categorical	0.814	0.640	0.359	0.773	0.789	0.617	0.789
pmm (5)	232.1	0.791	0.44	0.789	0.791	0.650	0.791
regression (5)	234.6	0.791	0.44	0.782	0.674	0.653	0.791
matching	1.92	0.791	0.436	0.789	0.791	0.67	0.791

Table 2: **Mean accuracy for six classifiers using different imputation methods.** We report the mean accuracy using 10-fold stratified cross-validation for layoffs. (5) indicates 5 iterations for multiple imputation methods.

In general, different imputation methods did not improve the performance of the top classifiers RF and GP. Similar results were observed for the other two outcomes. The categorical method (encoding different missing values as separate categorical values) had the worst performance. Of the single imputation methods (mode, mean, and median), mean imputation boasted the best performance as it attains higher accuracy for LR, NB, and KNN without sacrificing accuracy for top classifiers RF and GP. Using multiple imputation methods improved the performance of LR but did not seem to significantly affect other classifiers. We have decided to continue further analysis using mean imputation because of its computational simplicity (faster) and good relative performance.

4.2 Evaluation results: classifier performance and feature selection

From our results of precision, recall, and F_1 score, all of the classifiers in Table 3 had similar performance. KNN seemed to have very slightly worse performance overall. This may be because KNN does not scale well with large datasets. All of the top classifiers were nearly flawless in recall (we consider a False classified as False to be considered a "True Positive"). Judging from the ROC curves and Table 3, RF performs the best for Layoff and Job Training whereas LR performs the best for Eviction at least according to our cross-validation results. The confidence intervals were relatively small so we were reasonably confident that these classifiers were accurate. The intervals, however, overlap so we cannot use this metric to distinguish the performance of the classifiers. We continue to investigate whether features selection will help further distinguish the top classifiers before we try and predict.

	Layoff				Eviction				Job Training			
Classifier	Prec	Recall	F_1	Time (s)	Prec	Recall	F_1	Time (s)	Prec	Recall	F_1	Time (s)
LR	0.790	0.998	0.882	0.10	0.940	0.998	0.968	0.40	0.765	0.995	0.865	0.14
KNN	0.790	0.989	0.878	0.01	0.940	0.999	0.969	0.01	0.764	0.979	0.858	0.02
RF	0.791	0.999	0.883	0.47	0.940	0.999	0.969	0.45	0.765	0.998	0.866	0.43
GP	0.791	0.999	0.883	0.82	0.940	0.999	0.969	1.06	0.765	1.000	0.867	0.765

Table 3: **Results of six classifiers on the three binary outcomes.** For each classifier and binary outcome, we report the precision, recall, F_1-scores, and wall clock time in seconds from 10-fold cross-validation.

Classifier	Layoff	Eviction	Job Training
LR	[0.783, 0.793]	[0.938, 0.942]	[0.762, 0.767]
KNN	[0.769, 0.797]	[0.938, 0.942]	[0.753, 0.758]
RF	[0.789, 0.793]	[0.938, 0.942]	[0.762, 0.767]
GP	[0.789, 0.793]	[0.938, 0.942]	[0.764, 0.766]

Table 4: **95% confidence intervals for accuracies of the 3 outcomes.** For each outcome, we used the "case resampling" bootstrap method to compute a 95% confidence interval for the accuracies for 10 samples.

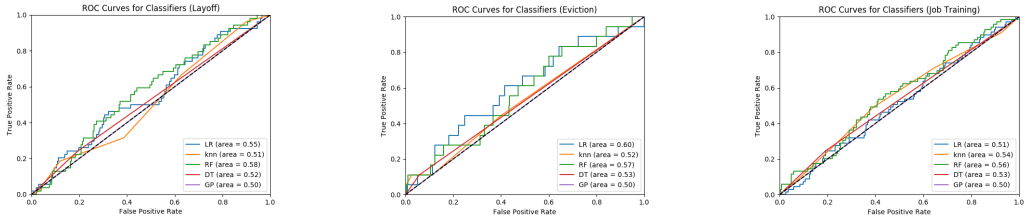

Figure 1: ROC curves for classifiers of the three outcomes

Feature selection slightly improved the accuracy of KNN, LR, and RF. It did not seem to have an effect on GP. We saw the best improvement using mutual information feature selection and chi2 feature selection. These methods were both suggested by SciKit Learn for classification problems and appear in other studies [15] [20]. Feature selection also improved the computational time of all classifiers. This is expected since feature selection reduces the dimensionality of the data set. Since Vaghela found that performance enhancement from features selection is dependent on the data set, we were not surprised by the little improvement[20]. For the other two outcomes (results not reported here), we found similar results which confirmed that mutual info feature selection and throwing out the bottom 30% of features had the best, albeit small, improvement.

	No feature selection		FS (MI top 70%)		FS (LSVM)		FS (Chi2 top 70%)		FS (GradientBoost)	
Classifier	Time	Accuracy	Time	Accuracy	Time	Accuracy	Time	Accuracy	Time	Accuracy
LR	0.10	0.787	0.11	0.790	0.02	0.790	0.11	0.789	0.03	0.789
KNN	0.01	0.783	0.02	0.791	0.01	0.785	0.02	0.788	0.01	0.785
RF	0.47	0.791	0.40	0.791	0.38	0.792	0.42	0.792	0.39	0.791
GP	0.82	0.791	0.60	0.791	0.43	0.790	0.59	0.791	0.46	0.791

Table 5: **Results of feature selection on six classifiers for Layoff.** For each classifier, we report the accuracy and wall clock time in seconds with and without feature selection.

We explored the top features selected in the feature selection process and found that questions asked of the mothers were important. All three outcomes shared the following top features: m1citywt, m1a15, cm1age, m1f1a, m1f7, m1f7, m1b1a, m1b1b, hv5_ppvtae (this is only a subset). Specifically, questions relating to the neighborhood the family resides in, the child's birth, relationships to significant others, and the child's performance in school exhibited predictive content in all three outcomes.

4.3 Computational speed

There was not high variance when it comes to computational speed. KNN was the fastest followed by LR, RF, and GP respectively. KNN was fast on this data set because we had a small number of features. GP was the slowest because it uses the whole samples and features information to perform prediction. It is generally slow when the number of features exceeds a few dozen [15].

5 Discussion and Conclusion

In this assignment we compared 6 classifiers with a background set of 4242 samples and 12495 features with lots of missing data. We compared different imputation methods and found multiple imputation methods to be superior to single imputation methods, matching the results found by Shrive et al. [19]. We also found that using a threshold of missing data to remove features improved accuracy as suggested by Michy [11]. The random forest, gaussian process, and logistic regression classifiers exhibited the best performance for the three outcomes with random forest being the most robust in our evaluation metrics. Based off metrics from cross-validation and small confidence intervals, we were confident that the random forest classifier was the best for layoff and job training whereas logistic regression was the best for eviction. Feature selection improved accuracy slightly but more importantly, decreased the computational time of all classifiers. Our submission on the FFC website achieved the following scores (under netid `mackl`): GPA: 0.39273, Grit:0.21997, Material_hardship:0.0288, Eviction:0.0566, Layoff:0.22453, Job_training:0.27925. Further studies could explore imputation without MNAR instead of MCAR as well as isolating different feature categories such as including only questions asked of the mother or father.

Acknowledgments

I would like to acknowledge Professor Xiaoyan Li and the TA's for assisting me with my questions on Piazza. I would also like to acknowledge Professor Engelhardt for providing the background necessary for me to complete the assignment.

References

[1] Fragile families challenge. http://www.fragilefamilieschallenge.org/. Accessed: 2017-03-31.

[2] G. Ambler and O. Rumana. A comparison of imputation techniques for handling missing predictor values in a risk model with a binary outcome. *Statistical Methods in Medical Research*, 16, 2007.

[3] S. Borra and A. Di Ciaccio. Measuring the prediction error. a comparison of cross-validation, bootstrap and covariance penalty methods. *Computational Statistics Data Analysis*, 54:2976–2989, 2010.

[4] Jason Brownlee. Logistic regression for machine learning.

[5] Rich Caruana and Alexandru Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 161–168, New York, NY, USA, 2006. ACM.

[6] Barbara Engelhardt. Cos424 lecture 9. Accessed: 2017-03-31.

[7] Barbara Engelhardt. Fast classification of newsgroup posts. 2017.

[8] D. Jurafsky and J. H. Martin. *Speech and Language Processing*. 2016.

[9] O. Kummer, Savoy J., and R. E. Argand. Feature selection in sentiment analysis. 2012.

[10] R. Little and D. Rubin. Statistical analysis with missing data. 1987.

[11] Alice Michy. Imputing missing data with r.

[12] C. Musil, C. Warner, P. Yobas, and S. Jones. A comparison of imputation techniques for handling missing data. *Western Journal of Nursing Research*, 24, 2002.

[13] A. Y. Ng and M. I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. *Advances in neural information processing systems*.

[14] M. Opper. Gaussian processes for classification: Mean-field algorithms. *Neural Computation*.

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[16] J. L. Peugh and C. K. Enders. Missing data in educational research: A review of reporting practices and suggestions for improvement. *Review of Educational Research*.

[17] M Ebden S Reece N Gibson S Roberts, M Osborne and S Aigrain. Gaussian processes for time-series modelling. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*.

[18] Dipanjan Sarkar. *Text Analytics with Python: A Practical Real-World Approach to Gaining Actionable Insights from your Data*. "Apress", 2016.

[19] Fiona M. Shrive, Heather Stuart, Hude Quan, and William A. Ghali. Dealing with missing data in a multi-question depression scale: a comparison of imputation methods. *BMC Medical Research Methodology*.

[20] V. B. Vaghela and B. M. Jadav. Analysis of various sentiment classification techniques. *Analysis*, 140, 2016.

[21] S. van Buuren. Multivariate imputation by chained equations, 2017.