Prediction of Outcomes for Children of Fragile Families

Gregory McCord
Princeton University '20
gmccord@princeton.edu

Abstract

Every year in the United States, millions of children and their families are evicted each year [1]. Poverty can create a hostile environment for these children to grow up in, and as a result, the cycle of poverty can continue for generations if these children fail to gain access to education, steady employment, or other forms of aid. One such category of children are those that come from fragile families - families which are single-parent and generally low-income. These families have a greater risk of falling into poverty than the traditional nuclear family. In this assignment, we attempt to predict several response variables such as whether the fragile family was evicted or not given survey data accrued over 15 years of research of nearly $5,000$ families. This data comes from the Fragile Families Challenge and is an effort to predict how a child will perform under these circumstances and to discover what indicators in a child's upbringing should trigger red flags and indicate the need for potential third-party aid or intervention. The results of our study (and the results of other participants in the competition) have shown that the response variables were difficult to predict from the supplied data. Despite our eviction classifier (the primary response variable we focused on predicting) placing in the top 5 in the competition, our results were comparable to those of a random classifier. Ultimately, our results show that the Random Forest classifier with 500 trees predicted the outcomes best for the eviction response variable.

1 Introduction

The Fragile Families dataset consists of approximately $13,000$ features, from which about $11,000$ came from questions asked by researchers to the child, the mother, or the father. Furthermore, each feature corresponds to a time period relating to the age of the child - birth, 1, 3, 5, and 9. For our purposes, these features will be used to predict data for when the child is 15.

While the Fragile Families data can be used to predict a number of different outcome variables, we have been tasked with predicting the outcome of just 6: GPA (at 15 years old), grit (measure of psychological determination), material hardship (measure of difficulty of physical situation), eviction (whether the family was evicted or not), layoff (whether or not the guardian lost their job), and job training (whether or not the guardian receives job training). We are attempting to predict critical occurrences in a child's life based on factors seen in their family life during their childhood.

While all of these features are important, we chose to focus on the binary outcomes (eviction, layoff, and job training), and in particular attempted to predict eviction as well as possible. We chose these features because we felt they were likely to be the most predictable of the 6 outcomes given the nature of the collected data.

2 Related Work

This is the premier dataset for analyzing trends in fragile families, and it has been used by hundreds of researchers over the past several years on a spectrum of topics including the effects of parental incarceration on children and the performance of children from low-income households in schools. One particular study analyzed eviction rates among the fragile families and attempted to correlate them with a number of features. They found that lower incomes and separated families tended to have higher rates of eviction [1]. However, while this study focused on the effects of eviction, we aim to identify features that predict eviction.

3 Methods

We took the original data we were given and took subsets of the features to perform feature selection. We downloaded the metadata information for these features from the Fragile Families website and used this file to filter out different categories of features to select features manually. While some of our classifiers perform feature selection internally, additional feature selection helps to reduce the number of features beforehand since there are so many more features than records, which helps prevent the models from overfitting due to noisy features. We trained all of our models on different sized feature sets consisting of all features, only the constructed features, only the question features, and both the constructed and question features. Question features refers to those responses that came from a questionnaire administered to the child/mother/father by a Fragile Families researcher, while constructed refers to features derived from combinations of these question features. In most cases, the models performed best when trained solely on the question features, but there were some exceptions.

3.1 Data Processing

We were given the sensitive data by the Fragile Families & Child Wellbeing Study. It contained approximately $13,000$ features and $4,242$ records. Additionally, we received the training set labels for our 6 outcome variables. The training set comprised $2,121$ records. We first used the Python library numpy to impute missing values by replacing missing values in the data with the mode of all samples for that feature. If the most common value was missing, then we simply set the value to 1. We believed that it is appropriate to impute data here because we assume that no single feature has too much influence over the response prediction. In the next step, we joined the feature data with the set of training labels and removed the rows which were missing a label because we deemed it could damage our model, causing us to misclassify more labels. Finally, there were two instances of categorical data. The first case contained numerical data and some strings labeled ”Other” - we simply replaced ”Other” with an arbitrary value. The second case contained truly categorical data, which we factorized. While categorical data should generally be re-encoded using one-hot encoding for use with linear models, there were very few categorical variables, and the models performed equally well with and without those variables. As a result, we chose to factorize the data in order to keep a smaller number of features for simplicity.

3.2 Classification Methods

We used a number of methods from the SciKitLearn Python library [2] for the purposes of classification. We tuned the hyperparameters of all models using 5-fold cross validation with a few exceptions (some models did not have any hyperparameters to tune). We trained most of the models on the different subsets of data, but some models were simply not suitable for the dataset. The models had the same tuned hyperparameter for each dataset, indicating that feature selection did not change the models too much. The models for classification (prediction of eviction, layoff, and job training) were:

1. *Multinomial Logistic Regression with ℓ_2 penalty* (LR): using a penalty parameter of 0.2 and the SAGA gradient descent solver
2. *Support Vector Machine with ℓ_1 penalty and linear kernel* (SVM): using a penalty parameter of 0.2

3. *Gaussian Naive Bayes classifier* (GNB): there is no hyperparameter
4. *Random Forest* (RF): using 500 trees

3.3 Evaluation of Results

We evaluated our performance in two ways. For our own comparison of methods, we compared our prediction results using 5-fold cross validation using the mean accuracy on the withheld data. While this is not ideal, it allowed us to predict the performance of the models on the test data. For the official comparison of methods, the Fragile Families competition used the Brier loss function, which is an MSE estimate using the equation:

$$BS = \frac{1}{N}\sum_{i=0}^{N}(a_i - p_i)^2$$

where a_i represents the true label (either 0 or 1) of observation i and p_i represents the predicted probability of the i^{th} label being 1.

4 Key Method: Random Forest Classifier

The Random Decision Forest algorithm (Breiman, 2001) is an ensemble method and discriminative classifier that can successfully classify non linear datasets and reduce the noise of unimportant features. The model proved very useful in this experiment for binary classification as a result. A random forest is a collection of decision trees, and so we will begin our discussion with the fundamental unit of the forest, the decision tree.

Given a dataset $\mathcal{D} = (\mathbf{x}_1, z_1), ..., (\mathbf{x}_n, z_n)$, where $z \in \{-1, 1\}$ represents the binary classes, the decision tree learns based on the maximum information gain at any split, which is defined as:

$$IG(Y, i) = H(Y) - H(Y|i)$$

where Y is the label of a and i is the i^{th} variable in the feature set [5]. H(Y) is the information gain entropy function defined by:

$$H(Y) = -P(y = 0)\log P(y = 0) - P(y = 1)\log P(y = 1)$$

The decision tree algorithm splits a node based on the maximum information gain from any feature i and proceeds to recursively split on the subsets of i^* in the same way [4]. Once the decision tree is completely split, some of the leaves are removed in order to reduce the chance of overfitting.

A random forest is simply an ensemble of a multitude of weak learners. In a random forest, each tree is generated randomly by fitting a subset of the samples, and each node is split on a subset of the features. By creating n unique trees, you accomplish a number of things. First, it allows you to more quickly generate the model because you are not creating a n complete decision tree but rather n weak trees. The trees are weak learners because they are not trained on the full set of data or features and thus have weaker prediction rates.

Additionally, by the nature of being decision trees (which tend to have many leaves), each tree has a high amount of variance [4]. However, by virtue of there being many of these weak learners, the model creates an ensemble of these weak trees to create a strong learner (a classifier with a much higher classification rate). By combining these trees with high variance, we can ultimately take the mode of the labels generated by each tree and produce a result with a lower variance (we have averaged out the variance by bagging uncorrelated trees) [3].

Finally, a random forest model can be generated in parallel since all trees are independent of each other. As a result, the run time of the random forest model can be engineered to be much faster than its $O(Knp^2)$ runtime. Assuming perfect parallelization, the training time could be on the order of $O(np^2)$, equivalent to that of a single decision tree.

This model can easily be implemented in an online setting. As records are parsed, they are fed into the decision trees, which each generate a class label for the data. Like in the case of training, the model may be scored and the output labels summed in parallel in order to average the result and obtain the generated response label, reducing the time from complexity from $O(K \log p)$ to $O(\log K \log p)$. This is critical for online scoring where the data must be processed and scored in milliseconds (e.g. Facebook or Google determining which add to display to a user as the webpage loads).

Finally, the Random Forest model identifies influential features despite being a bagged, non-parametric algorithm, allowing us to better understand the data. While uninterpretable algorithms like K Nearest Neighbors tend to classify data well in non-linearly separable datasets, they do not grant us this extra knowledge like Random Forests, which allow us to make further improvements to our model.

Specifically in our experiment, we trained our random forest model on several subsets of the data using a variety of different hyperparameters. We did this for each binary response variable and achieved the results described in the next section. As detailed before, the random forest classifier proved to be our most effective method when it came to predicting eviction.

5 Results

5.1 Model Predictions

We begin our discussion of the results with the data contained in Table 1. The information about the specific model is contained on the left side of the table while the results for the binary response variables are contained on the right. The results were measured by the Fragile Family Competition and represent the Brier Score of the model. Each model was produced using a classification method with the hyperparameter as described in Section 3.2, the only exception being the Random Forest model, for which we included data for both 100 and 500 trees. Additionally, each model was trained on one of the data sets as described in Section 3. Lastly, the model was used to output either prediction labels or probabilities - methods that could not discriminate well between the two response classes (i.e. they probabilities tended to be near 50%) performed better using prediction labels.

Models that used strictly constructed features performed strictly worse on average than those that included the question features based on cross validation comparisons, likely because there were nearly 20 times more question features than constructed, allowing for more information to be encoded. We used these results to estimate the prediction power of our models.

Method	Data	Response Type	Eviction	Layoff	Job Training
LR	All	Binary	0.05849	0.23208	0.27925
LR	All	Probability	0.07754	0.18478	0.2108
LR	Both	Probability	0.07756	-	-
GNB	Question	Binary	0.38283	-	-
GNB	Question	Probability	0.38302	-	-
SVM	All	Probability	0.10755	-	-
RF 500	All	Probability	0.05332	0.18895	0.22446
RF 500	Question	Binary	0.05660	0.22453	0.27736
RF 500	Question	Probability	0.05283	0.19	0.22449
RF 100	Question	Probability	0.05331	0.22642	0.27547

Table 1: **Model results for binary response variables on test cases.** The first three columns represent the model used, how it was created, and what it was predicting while the latter set of columns show the Brier scores as reported by the Fragile Families competition website.

As seen in the table, the Random Forest Classifier with 500 trees performed the best after some feature selection on the eviction response variable while Logistic Regression performed best with all features included for both layoff and job training. Despite fear of overfitting, the random forest with more trees performed significantly better for all response variables than those with less because the dataset is very ”fat” (many more features than training cases). Surprisingly, we expected random forests to perform better than they did on layoff and job training, but these response variables appear to be more linearly separable and therefore can be well predicted by a linear model such as logistic regression. Finally, as we expected, the Naive Bayes classifiers performed poorly, likely because the features are clearly dependent on each other.

However, we must not lose sight of the overall results - our best models are comparable to a random classifier. Unfortunately, it seems that more precise classification at this point may require significantly more feature selection and more powerful classifiers. Another hypothesis for the poor results could be that the task of predicting future eviction, layoff, or job training is an inherently messy problem. That is, the feature set itself does not carry the predictive power necessary to accurately classify the desired response variables.

5.2 Influential Features

Eviction		**Layoff**		**Job Training**	
f2j18e	f4i18e	f5f26a	m4k4	m3c20b	m5a6g03_102
f3k21	f4l5a	f5f27	m4k5	m4citywt_rep8	m5b17d_101
f3natwt_rep21	m4k24f	f5f28d	m5a6g03_102	m4natwtx_rep19	m5g2a_7
f3natwt_rep33	m4k3b	f5f28e	m5f16a_1	m4natwtx_rep21	t4b1e
f4i18d	m5a6g03_102	m1b23b	m5f26b2_9	m4natwtx_rep22	m3b34c

Table 2: **Display the most influential question features in the Random Forest models.** For each binary response variable, the top 10 features are listed in order of their Gini scores.

Only one feature - m5a6g03_102 (the child wanted to live elsewhere) - appeared in all three lists. Very practically, if a child wishes to live elsewhere, the home environment might not be conducive to their growth and indicate potential future troubles (like eviction or layoff). The features best correlated with eviction dealt with if either the mother or father received money from other sources (e.g. Supplemental Security Income) or the family's ability to gain income. The features most indicative of layoff were responses of the father concerning criminal charges (e.g. time spent in jail). Finally, the features indicative of job training were asked of the mother (who tended to be the primary caregiver) concerning if they had received monetary help from the father. These features show promise for being strong predictors for the response variables and make intuitive sense in the context of the problem.

While the table only displays features from the questionnaire dataset (because most models performed best on this dataset), one constructed feature - cm1relf (the parents' relationship at birth) - showed up in the list of top features for data sets that included all features. However, there were no other new features that appeared.

6 Discussion and Conclusion

In this paper, we compared the results of several different classifiers on the three discrete response variables (eviction, layoff, and job training) in the Fragile Families dataset while focusing primarily on the results of the eviction models. The classifier with the lowest Brier Score (an MSE measurement) was the Random Forest with 500 trees, which we trained using a dataset composed solely of features from the questionnaire. We found several key features that correlated well with all three response features including m5a6g03_102 (the child wanted to live elsewhere), cm1relf (the parents' relationship at birth), and others dealing with incarceration records, family income, and supplemental sources of income. While our results do not indicate the success we had hoped for, the features found most influential could help to further refine the feature selection process, allowing for better classification rates on future iterations of the competition.

Because Random Forests proved to be our most successful classifier in this experiment, it might prove worthwhile to use other more powerful ensemble methods for better classification results. One such example is the gradient boosted tree algorithm (specifically the implementation used in the XGBoost library). While there are many more hyperparameters that require accurate tuning (unlike with random forests), since the data is very small (only around $2,000$ records in the test set) with a little bit of feature selection, tuning can easily and quickly converge using either a simple grid search algorithm or a more advanced bayesian optimization approach.

References

[1] Matthew Desmond and Rachel Tolbert Kimbro. Eviction's fallout: Housing, hardship, and health. *Social Forces*, 94(1):295–324, 2015.

[2] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, and et al. Grisel O. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[3] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[4] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. MIT Press, 2013.

[5] J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.