Forecasting Recidivism in New York County Criminal Data

Jash Ajmera, Connor Concannon, Rob Long, Stella Sun

DSGA-1001 Fall 2017

December 9, 2017

NetID:

Jash: jda448

Connor: cmc1204

Rob: rl2898

Stella: ss8955

1. Introduction

The United States criminal justice system processes millions of defendants each year. According to one estimate, nearly ten million misdemeanor cases are filed annually (Natapoff 2012). There are very limited resources, such as programming and treatment opportunities, compared to the huge number of defendants who are technically eligible for those services. In addition, the prosecution and courts must make high-stakes decisions about which pre-trial defendants can be released into the community and which ones may threaten public safety or order. This is a perfect opportunity to use machine learning to help predict those most at risk of future offending. If a classification algorithm were implemented in production, the state could use this classification algorithm to tailor responses to match the risk level of the individual. Defendants at lower risk levels could be diverted from the criminal justice system altogether.

As part of our project, we will be forecasting recidivism, which is defined as a charge or conviction for any new offence, no matter how minor. We will forecast whether each offender will be arrested within the following 36 months using a statistical learning approach that makes no assumptions about how predictors are related to the outcome. And finally, this will help us in concentrating rehabilitation, treatment and surveillance resources on a small subset of convicted offenders who may be in greatest need, and who pose the greatest risk to society.

2. Data Understanding

The data was acquired thanks to one of the group member's employer, the New York County District Attorney's Office (DANY). DANY has a robust case management system and data infrastructure, and allowed the researcher to extract a sample (n=100,000) of defendants charged between 2010 and 2013, with at least three years of follow-up time between the disposition of the case and the date the data was extracted. The data consists of defendant characteristics, prior criminal history measures, and indicators about the current offense. Crucially, this dataset contains no identifiable information aside from age, race, and gender. To further ensure complete confidentiality, the researcher took an additional step to obfuscate the entire sample.

The synthpop R package was used to generate a synthetic copy of the data which retained similar distributions and properties as the original (Nowok, Rabb, & Dibben 2016). This package was created to balance the need for individual privacy and advancing research. As the authors explain, census data from the United States and United Kingdom contain rich information about individuals, which limits the number of researchers that are authorized to analyze this data. To overcome this burden, the authors developed a package that allows a 'synthesizer' - an individual with access to real data - to create synthetic data for an 'analyst' - someone without access to the actual data. The synthetic data was generated and shared with team members via a secure dropbox.

3. Data Preparation

After the synthetic instances were created, we considered conversion of values, removing or inferring missing values, converting data types, and if any numeric values needed to be normalized. The raw data has 100,000 rows and 83 features.

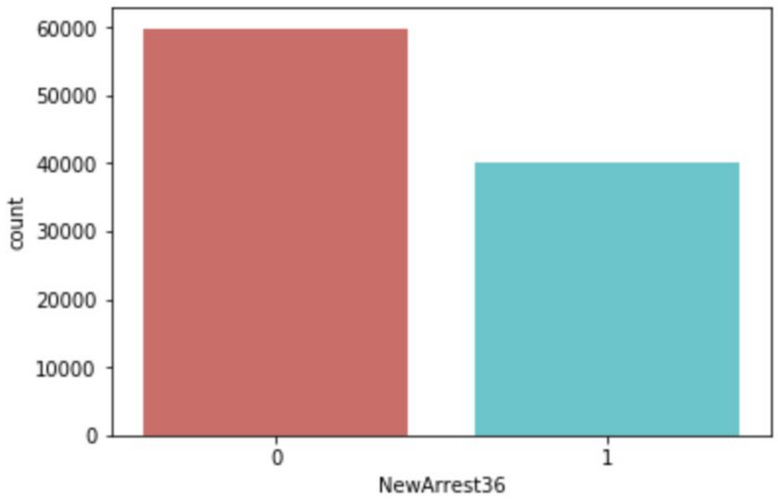
Of those rows, 47660 rows contained at least one missing value. The column AgeAtFirst (age of first conviction) had the most missing values. We soon realized that many of those arrests were first time arrests, and so would not have any value for AgeAtFirst. So we replaced the NA values in AgeAtFirst column with the corresponding value in Age column. For the rest of 25 missing values, we simply dropped them--this would have little effect with our total of around 100,000 data points.

A number of criminal history features measured in slightly different timeframes (previous 10 years, previous 9 years, etc.) were removed due to a high number of features in the dataset. Only criminal history features relating to the previous five years were retained. We also dropped some features--namely CareerLength, MajorGroupDetail and CvtTypes--based on the domain knowledge that was confirmed by the fact that they were highly correlated with other predictors. Next, categorical features representing defendant race and crime category were converted to dummy variables for the use of logistic regression models.

Finally, we examined the target variable. The target variable in the raw data was a multi class variable indicating whether the individual was arrested again and what for what type of crime in New York City within the following three years. For our project, we converted it to a binary class variable, with 0 equal to NoneType, meaning no arrest, and 1 means any type of new arrest. Commonly known as recidivism, this target variable is somewhat analogous to "churn" in commercial modeling contexts. Instead of predicting those that will leave or 'churn' from the criminal justice system, our model predicts those that will return to the criminal justice system.

After data cleaning, our data contained 61 features and 99,852 rows. Within the sample, the base rate of NewArrest36 = 0.40 and No NewArrest36 = 0.60.

```
#check class balance
sns.countplot(x='NewArrest36',data = df,palette='hls')
plt.show()
```



4. Modeling and Evaluation

The goal was to forecast recidivism from the information provided, which will help prosecutors make decisions about supervision and services to be provided to each new defendant. That is, we would like to know the probability of each new instance that will fall into the new arrest class.

The first model we chose was Logistic Regression, since it returned a score instead of simply labelling the prediction. Compare to models such as support vector machines, logistic regression takes less time to run. Given our large amount of training --over 60,000 instances in training data--we decided not to use support vector machines. Support vector machines may be less comprehensible to practitioners as well; we discuss these issues in ‘Deployment’. Our second choice was Random Forest, a less transparent but more powerful model than Logistic Regression. We then compared the two models and assessed their pros and cons.

Model 1: Logistic Regression

1.1 Feature Exploration

We have 60 features. To explore features, we ranked the AUC score and Mutual Information of each individual feature and plotted the AUC scores.

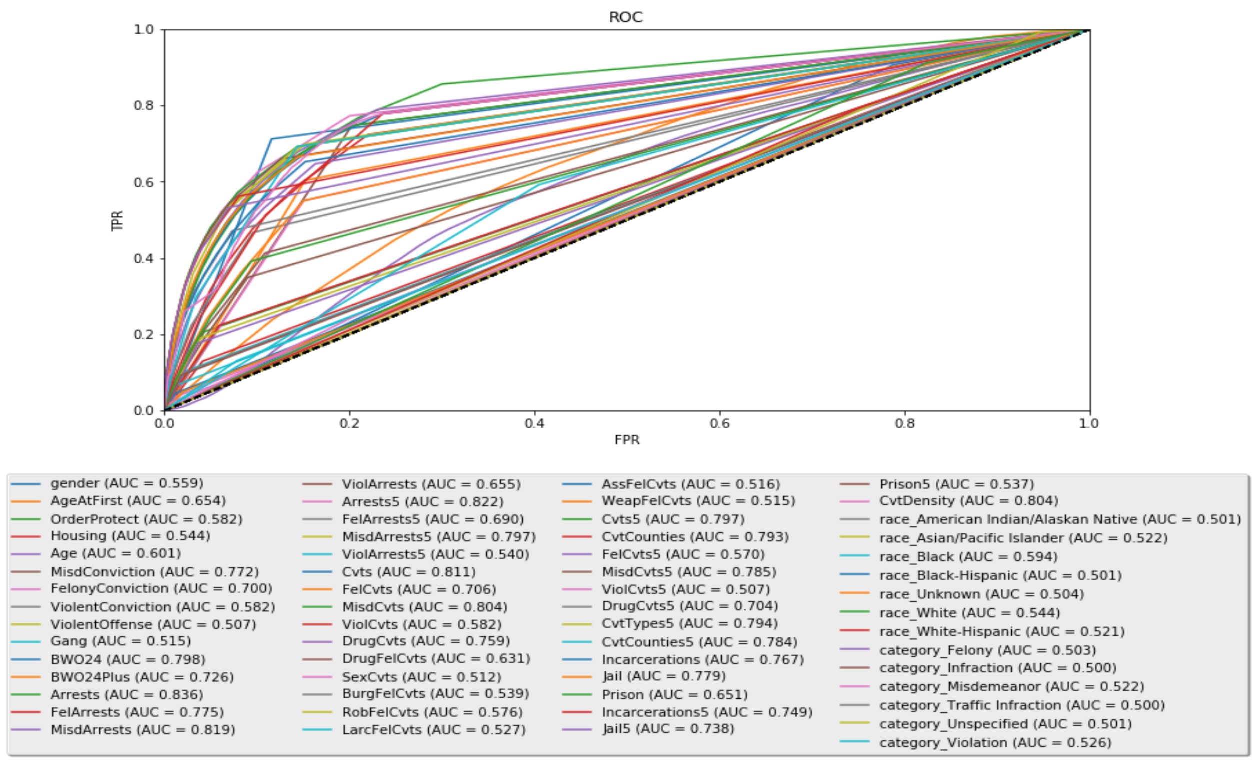


Fig.1. Feature importance

In this case, there was a fairly close linear relationship between AUC ranking and MI ranking (Fig 2). Therefore we could use either of them to build our first LR model.

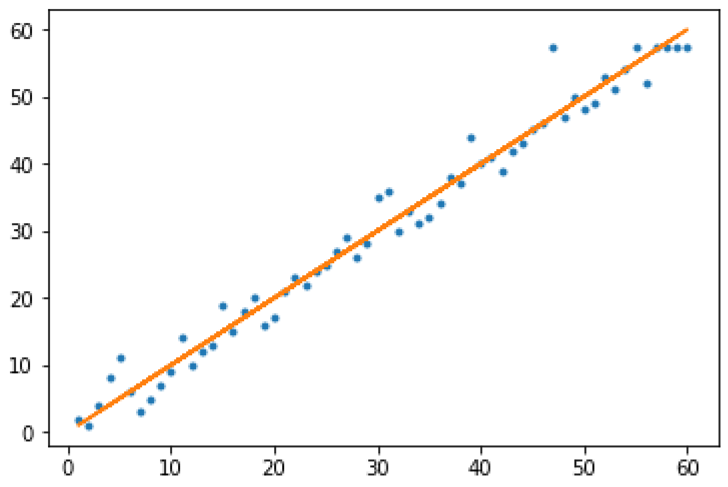


Fig.2. AUC_Rank Vs. MI_Rank

We used the top half of features (i.e. top 30 features) to build the first LR model as our baseline model. Figure 3 shows the performance of this baseline LR model.

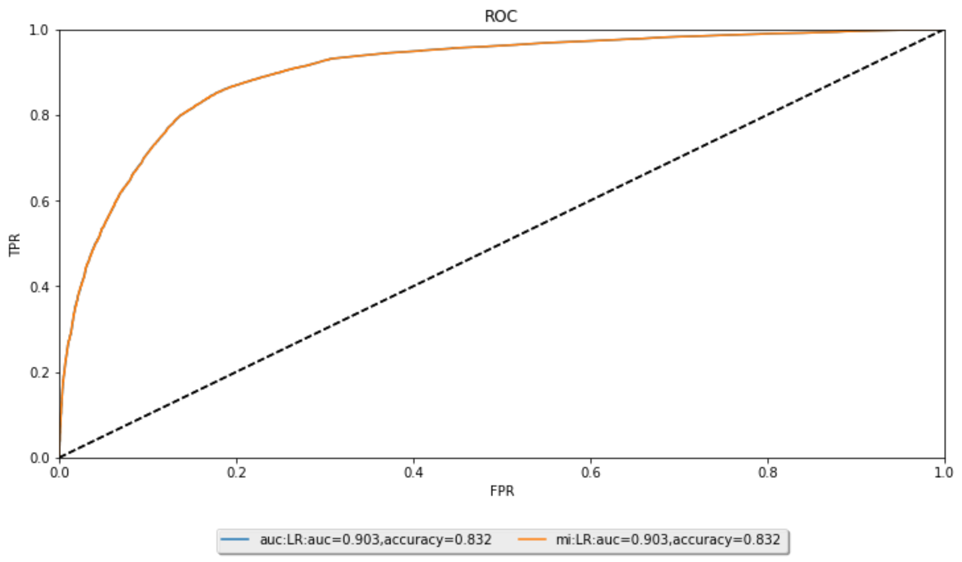


Fig.3. First LR Model with top 30 features

The first baseline LR model gave us an AUC_Score = 0.903 and Accuracy_Score = 0.832. This seems to be a good start. However, it is a relatively complex model with 30 features, and high variance. In order to try to improve on the baseline model, we performed feature selection and hyperparameter evaluation.

1.2 Feature Selection

<u>Method 1: Naïve Feature Selection</u> (choose top k features with best AUC scores).

Since we have no strong antecedent ideas what k to use, we tested how the model improved with top k features in step = 5. We built LR models with top 5, top 10, top 15...top 60 features and plotted how AUC and and accuracy changed for these feature numbers (Fig 4).

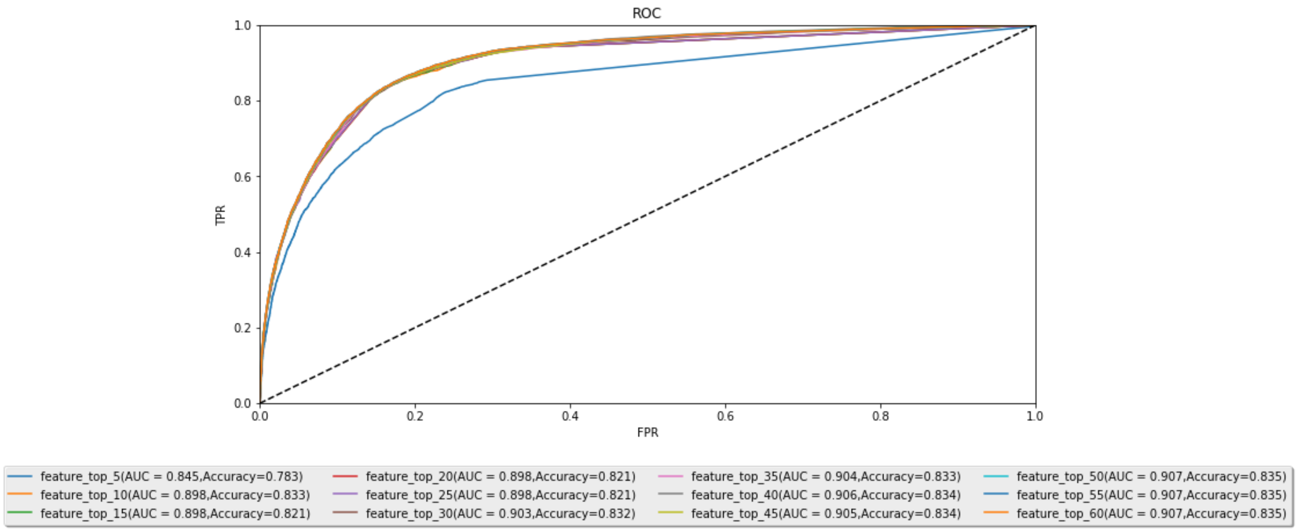


Fig.4. ROC for the use various top K features

Figure 4 shows that as the number of features k increased, our LR model tended to have better AUC. That said, when k >50, both the AUC and accuracy tend to stabilize. The “best” model in this feature selection method is the one with top 50 features (AUC = 0.907, accuracy = 0.835).

This gave us our second LR model: it incorporated top 50 features using the Naïve Feature Selection method, with auc = 0.907 and accuracy_score = 0.835.

Compared to our first baseline model with top 30 features, the second LR model improved on AUC and accuracy, but only marginally. We are given an accuracy boost of merely 0.004. And that came at the cost of the second model being more complex, with higher variance. The complexity had downsides: it was harder to interpret all 50 features, and it took longer time to run, and it increased the risk of overfitting. Therefore, we tried another method called Stepwise Forward Selection method to pick the best subset of features.

Method 2: Stepwise Forward Feature Selection.

We iteratively built the feature set and calculated the cross-validated mean LogLoss (Fig 5). There were several ways to define a stopping criteria: we could choose k = 30, where the error tended to be plateau; or we could choose k = 20, where the error is below the red line, which is 1 standard error. To be conservative, we choose k = 17.

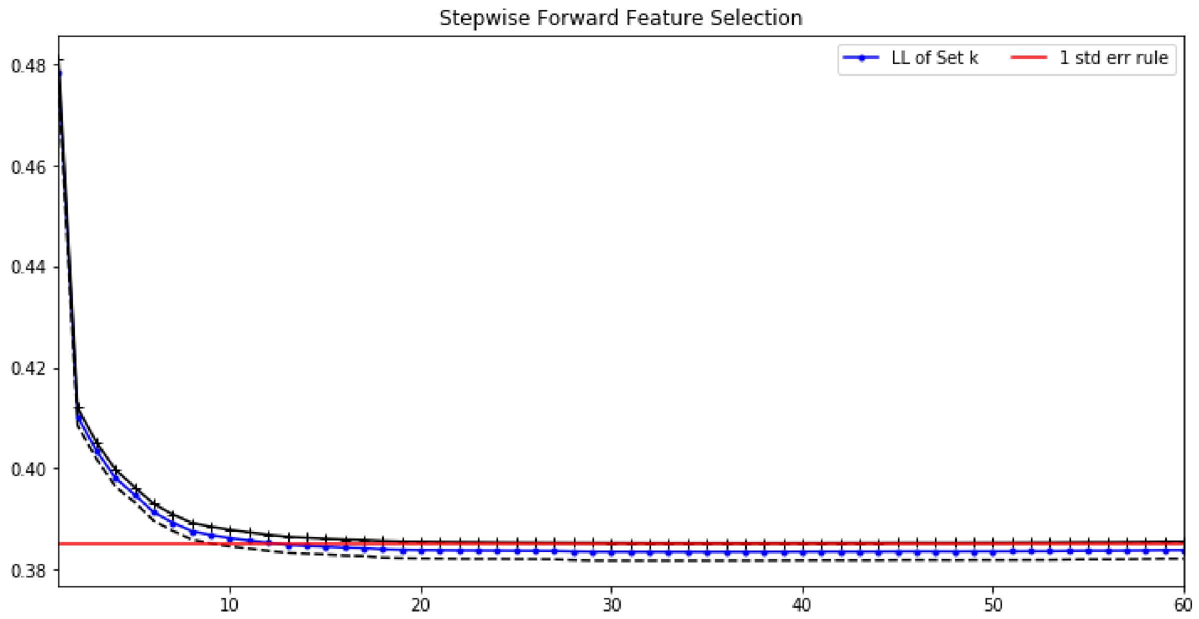


Fig.5. Stepwise Forward Feature Selection

With k = 17, we had our third LR model with 17 features: *CvtDensity, MisdConviction, AgeAtFirst, FelCvts, Age, Arrests5, ViolentOffense, CvtTypes5, race_Asian/Pacificislander, gender, category_Felony, race_White, BWO24,Housing,race_Black, MisdCvts5, Gang*. (See appendix for further explanation of these features).

As a big-picture comparison of all three LR models, consider the AUC and accuracy scores of each (Fig 6).

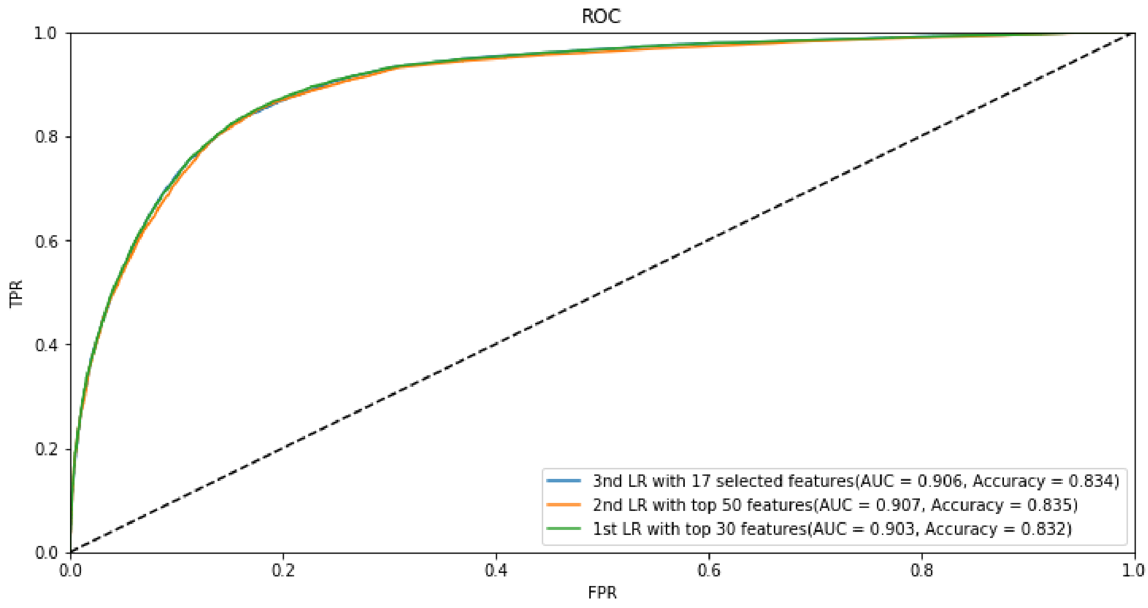


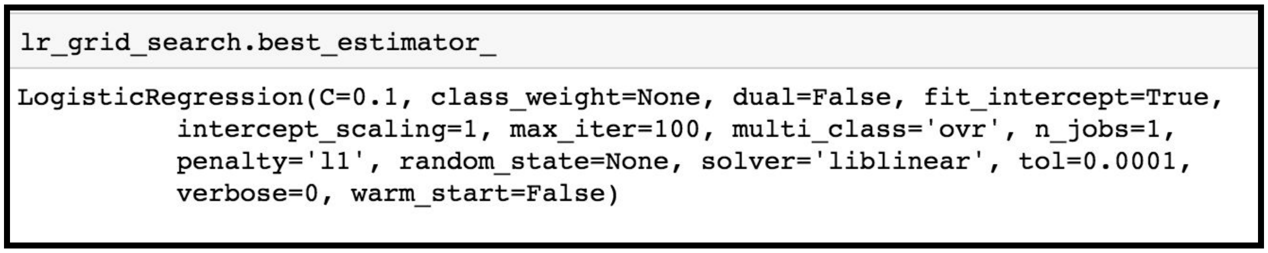
Fig.6. Compare three LR Models

The third LR model with 17 features gave us an AUC of 0.906 and accuracy of 0.834. Compared with model 2, the third model has only slightly lower auc and accuracy. Overall, we think it worthwhile to reduce the features from 50 to 17--decreasing the complexity with little cost in accuracy. Therefore, we consider model 3 to be the best one.

1.3 Hyper-Parameter Evaluation

To further improve our LR model, we tested various hyper-parameters: we used grid search cross-validation to explore various test C parameters and regularizations (L1,L2).

Without preprocessing data, we first tested the C parameter. We tested seven possible Cs, from 10^-3 to 10^3. We got the best auc_score = 0.906 with C = 0.1 and an l1 penalty.

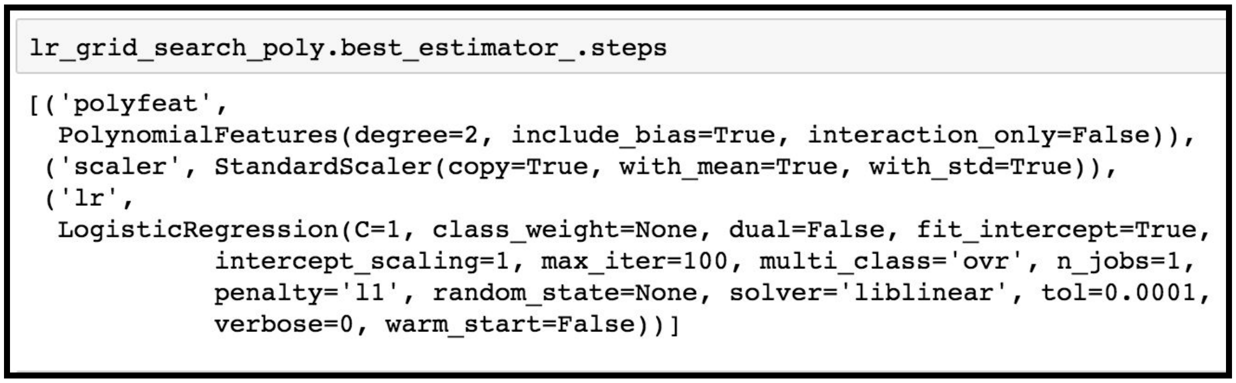


```
lr_grid_search.best_estimator_
```

```
LogisticRegression(C=0.1, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l1', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False)
```

Fig.7. Best estimator without preprocessing data

We also created pipelines to get the best estimator with scaled data (auc_score = 0.9061) and polynomial features. By the end, we ended up with the best estimator as following:



```
lr_grid_search_poly.best_estimator_.steps
```

```
[('polyfeat',
  PolynomialFeatures(degree=2, include_bias=True, interaction_only=False)),
 ('scaler', StandardScaler(copy=True, with_mean=True, with_std=True)),
 ('lr',
  LogisticRegression(C=1, class_weight=None, dual=False, fit_intercept=True,
            intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
            penalty='l1', random_state=None, solver='liblinear', tol=0.0001,
            verbose=0, warm_start=False))]
```

Fig.8. Best estimator with three steps

When comparing these three results, we saw the improvements in AUC was around 0.04 (Fig.9). The best AUC we ended up having was 0.911, and accuracy = 0.84.

To finalize the LR model, we converted the features to polynomial features with degree = 2, and then scaled with standard scalar. In building the 4th LR model, we used C = 1 and L1 penalty. Figure 9 compares: the LR model with default setting, the LR model with scaled features, and the 4rd model with polynomial features, scaled features and selected hyper-parameters.

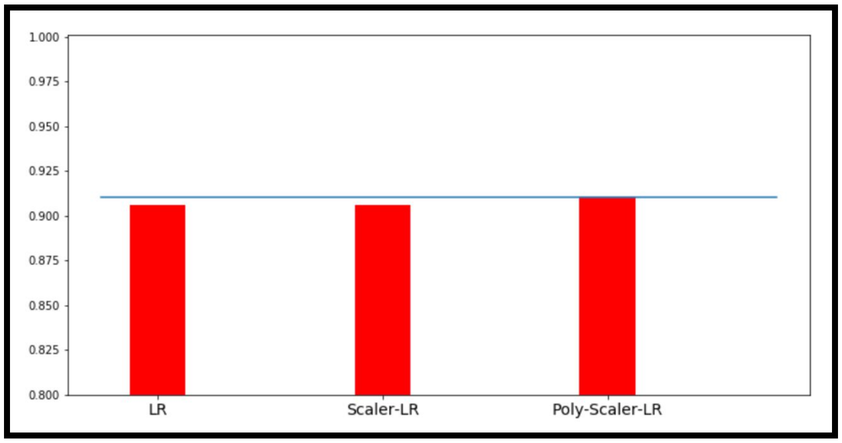


Fig.9. Comparing three models

1.4 Evaluation

To evaluate the LR model, we looked at three aspects: features; roc_auc score and confusion matrix; and the impact of race features on the model.

<u>Features Evaluation</u>

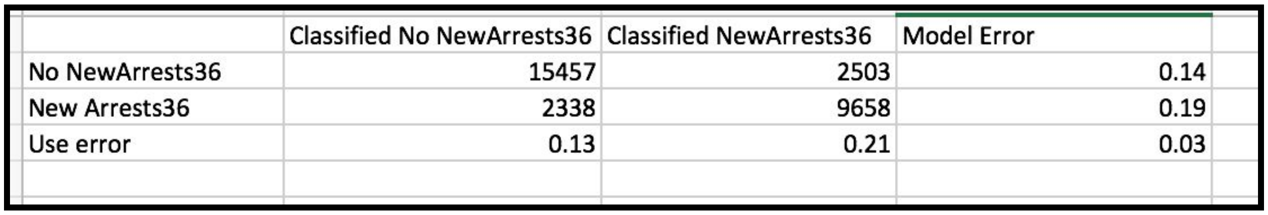
The main goal of the analysis was to develop a useful forecasting model. However, the degree to which our model will be accepted by the adult parole and probation departments (APPDs) depends in part on whether the results broadly make sense. Therefore, we need to evaluate the features we used in the model.

We ended up using 17 features: *CvtDensity, MisdConviction, AgeAtFirst, FelCvts, Age, Arrests5, ViolentOffense, CvtTypes5, gender, category_Felony, BWO24, Housing, MisdCvts5, Gang, race_White, race_Black, race_Asian/Pacificislander.*

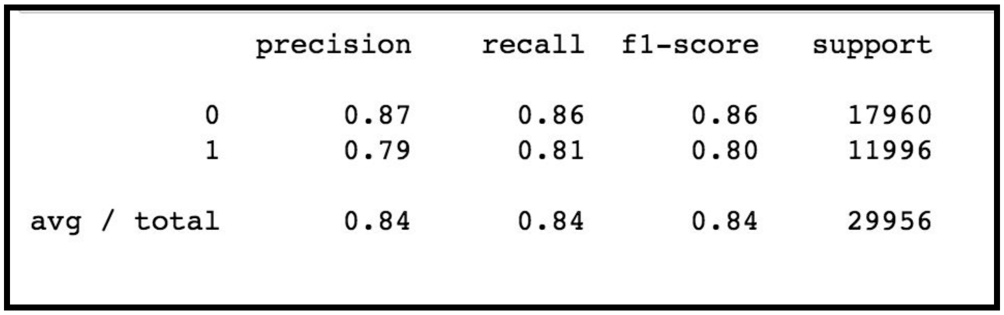
By looking at the AUC and MI score for these features, the highest AUC_Rank feature was *CvtDensity*, and the highest MI_Rank feature was *BWO24* (# of failures to appear for court in previous 24 months). *CvtDensity* is a feature created based on criminology work of Copas (1998). The conviction density predictor was computed as square root((# of convictions)/career length +1). That the predictor ranked highly was not surprising - it was an aggregate measure of criminal history that was composed of a number of features. Similarly, the bench warrant variable measured how many times the individual failed to comply with the orders of the court by returning on their scheduled appearance date. Unsurprisingly, those who disobey the orders of the court were arrested more often. In many cases, it may be that when these individuals were stopped by the police, they were arrested anyway because of the outstanding warrant.

Roc_Auc score and Confusion Matrix

The final model has an AUC of 0.911, which improved by 0.008 from the baseline model. However, to fully evaluate our model's forecasts, we produced a confusion matrix and classification report.(Fig.10)



	Classified No NewArrests36	Classified NewArrests36	Model Error
No NewArrests36	15457	2503	0.14
New Arrests36	2338	9658	0.19
Use error	0.13	0.21	0.03



```
             precision    recall  f1-score   support

          0       0.87      0.86      0.86     17960
          1       0.79      0.81      0.80     11996

avg / total       0.84      0.84      0.84     29956
```

Fig.10. Confusion Matrix and Classification Report

There were 2338 false negatives and 2503 false positives in the prediction. In the criminal justice system, the seeking the “correct” tradeoff between false positives and false negatives is an especially acute issue (Corbett-Davies et al. 2017). In this case, a “false positive” might lead to someone being held without bail, who would not need to be held; a “false negative” might lead to someone not receiving services they need, or committing a crime when they were released. Ultimately, policy makers will need to be very cautious in the *actions* that are taken based on these scores--how the scores are used will ultimately determine whether a morally and politically acceptable balance between false positives and false negatives is being struck. (See the ethics and deployment sections for further discussion).

When we consider overall model error and accuracy--setting aside looking at FP and FN separately--the overall forecasting looks good: we correctly predicted 19 of NewArrests36 out of 100 given cases; the overall model error was 3%, and the average precision and recall were about 84%.

Racial features in our model

We also noticed that there were 3 out of 7 race features in our final model, and to address the topic of racial bias we chose to blind the data from race features and calculate model accuracy. This did impose a cost to accuracy, but not a large one: blinding to race effectively decreased the AUC by 0.02.

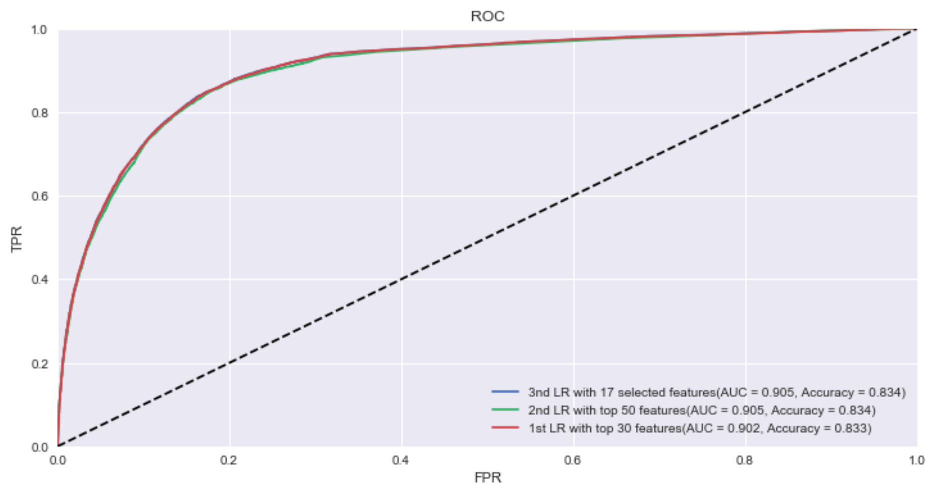


Fig 11. Compare three LR models without race features

The third LR model with 17 features gave us an AUC of 0.905 and accuracy of 0.834. Compared to the best scored model in the above section, our LR model without race features loses only 0.02 on the AUC and is a good approximation of the actual data. We discuss this aspect in more detail in the deployment section of our report.

Model 2: Random Forest

The random forest algorithm was a natural choice for this problem, given its efficiency on large data sets and its accuracy. We began by selecting features, using the built-in function in RandomForestClassifier. We saw that the feature_importance score had a big variance, with the highest score around 0.15 and lowest score very close to 0.

2.1 Feature exploration and First RF Model

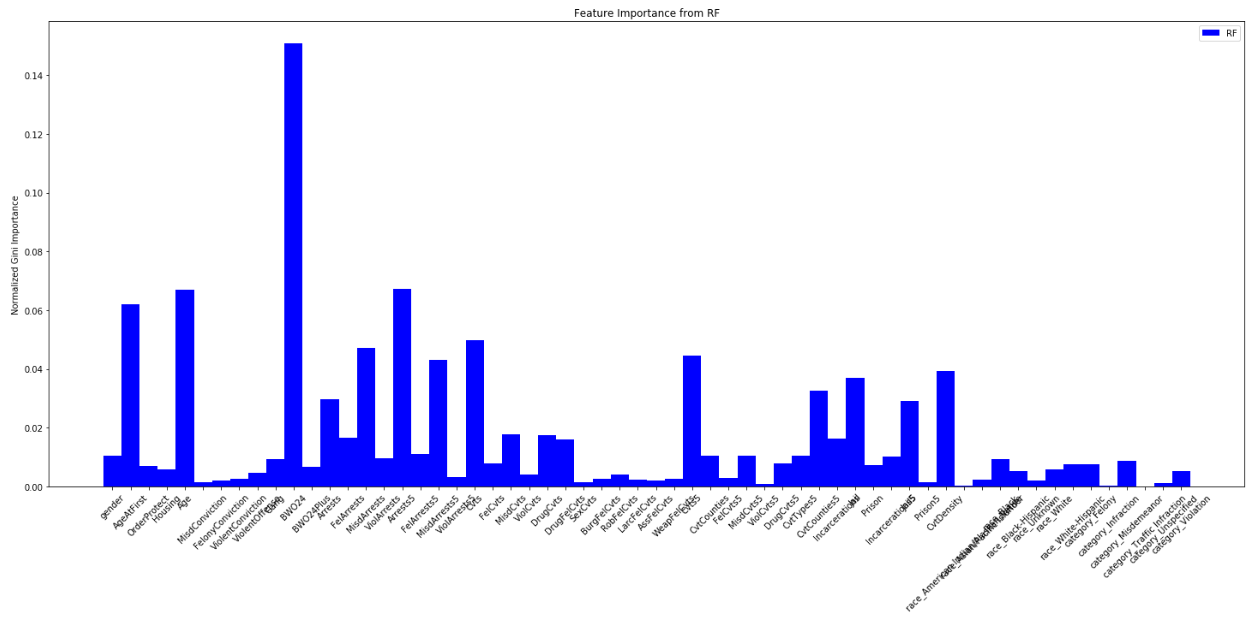


Fig.12. Feature Importance in baseline RF Model

In building our baseline RF model, we got rid of features that were effectively 0 (round to decimal 3). With the remaining 60 features, we built our first baseline RF model with the default setting. As expected, even a default RF achieved reasonably good accuracy of 0.838, with an AUC of 0.903. We noticed that the training score was 0.980, meaning that (as is common) the baseline RF predicted training data very well (accuracy_score = 0.954).

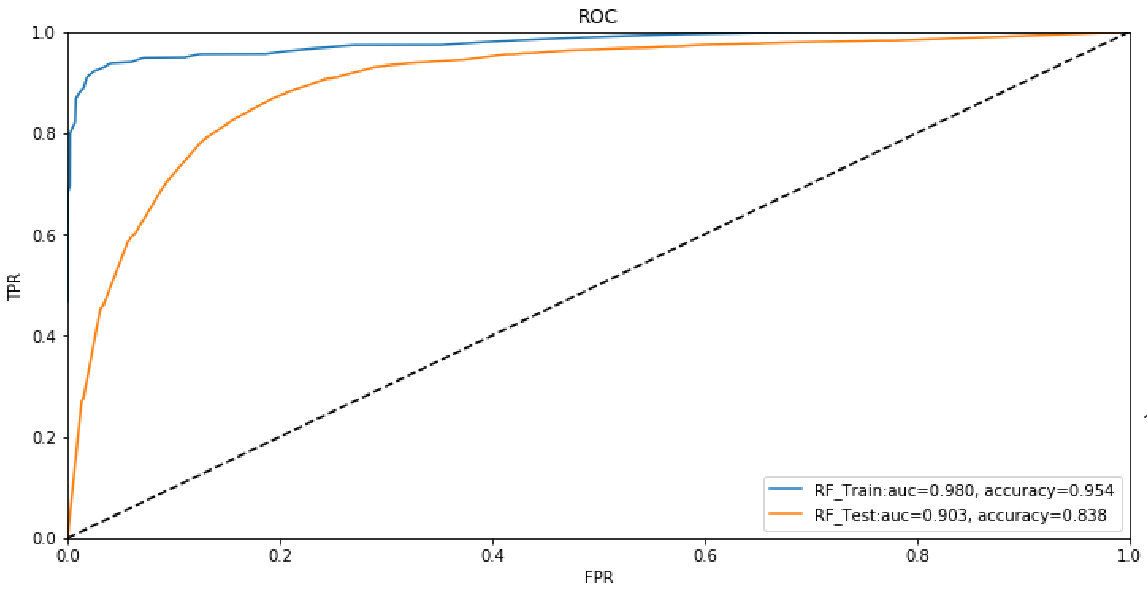


Fig.13. Training AUC and Test AUC in First RF Model

2.2 Hyper-Parameter Evaluation

There are two hyper-parameters that determine the complexity of our RF model: the number of estimators, that is, the number of trees used in our forest; and max_features, that is, the number of features randomly sampled for each tree. (As discussed in class, there is little need to tune the depth of the individual trees).

Instead of using cross-validation to choose the hyper-parameter, we used the much less intensive method of out-of-bag (OOB) error. We explored the following numbers of estimators: 200, 500, 1000, 1500; and the following numbers of features: 10, 20, 30, 40, 50, 60.

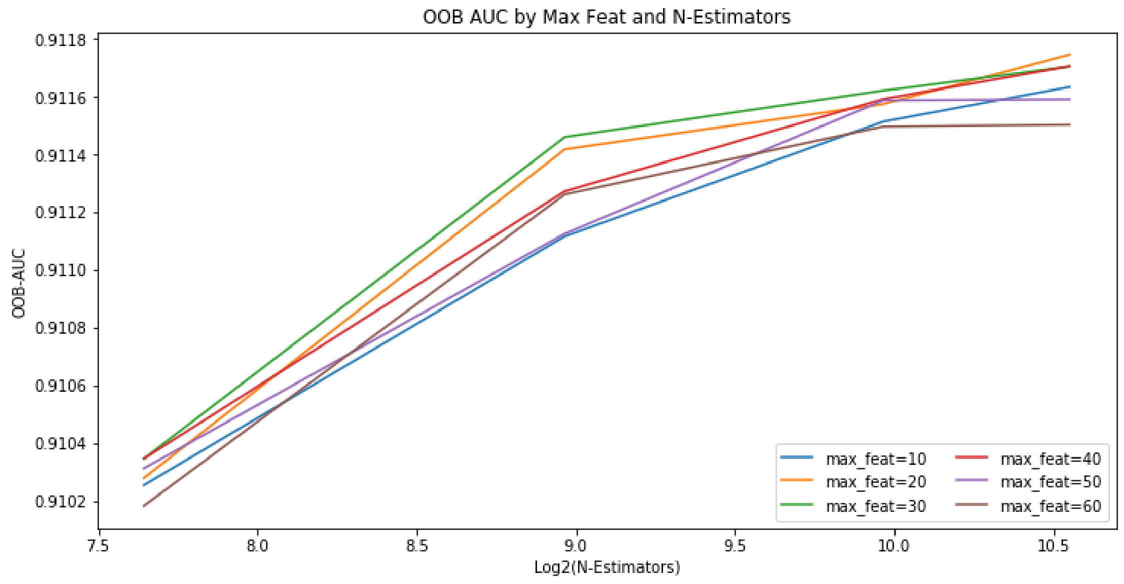


Fig.14. OOB AUC by max_feat and n_estimators

For max_feat between 10 and 30, as the number of trees grows, prediction improved--however, with max_feat > 30, the more number of trees resulted in worse performance. It was hard to decide how many features to use, since the fewer features were used, the greater the bias--while having too much features (e.g. max_feat = 60), creates the potential for high variance as well, since some features are highly correlated. Max_feat of 30 (green line) and max_feat of 20 (orange), seemed to show us the best performance--20 or 30 seemed to be an optimal number features.

To evaluate further, we compared their AUCs on test data. Fig. 15 validated our assumption of not using the max_feat > 30, since we could see that max_feat = 60 had the worst test_auc score, and that max_feat = 40 and 50, while better than than max_feat = 10, were inferior to max_feat = 20 or 30.

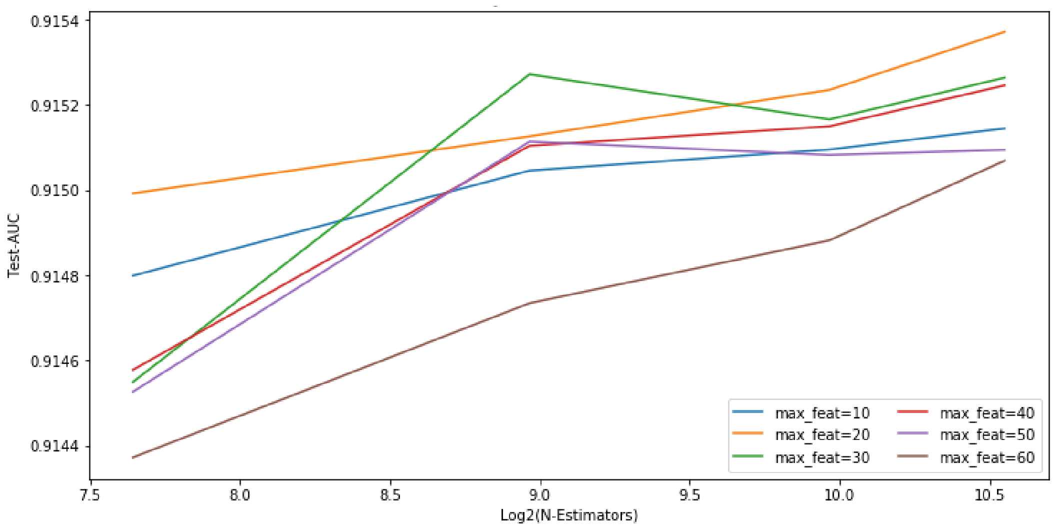


Fig. 15. Test AUC by Max Feat and N-Estimators

Based on these results, we concluded that max_feat = 60 has the worst performance out of all the max_feat; the best were 20 and 30. We then considered how they performed with different n_estimators. Looking at the lines for 30 max_features (green) and for 20 max_features (orange), we saw that 30 performs best with an n_estimator of 900; the line for 20 (orange), on the other hand, was still showing improvement as n_estimator grows. (The orange line had a better score than green when log2(N) was greater than 9). Therefore, to further improve our model, we built two models with these two options of hyper-parameter sets and evaluated them.

2.3 Second and Third RF Models

The second and third RF models with hyper-parameter options (max_feat = 30, n_estimator = 900), (max_feat = 20, n_estimator = 1500) each had the same AUC score up to 3 decimal places (Fig. 16). We also compared the running time for both models. The second model takes 270 seconds and the third model takes around 370 seconds. Given their essentially identical AUC curves, and given that the third model is more complex than the second one, we ended up using the RF model with max_feat =30 and n_estimators = 900.

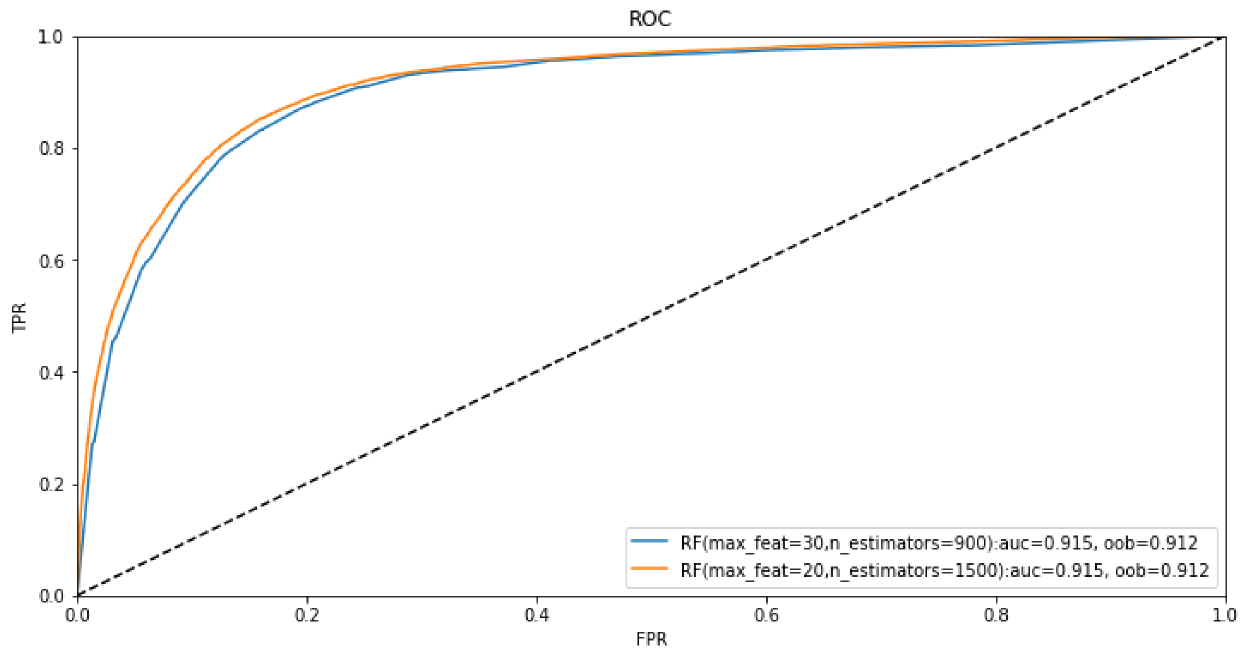
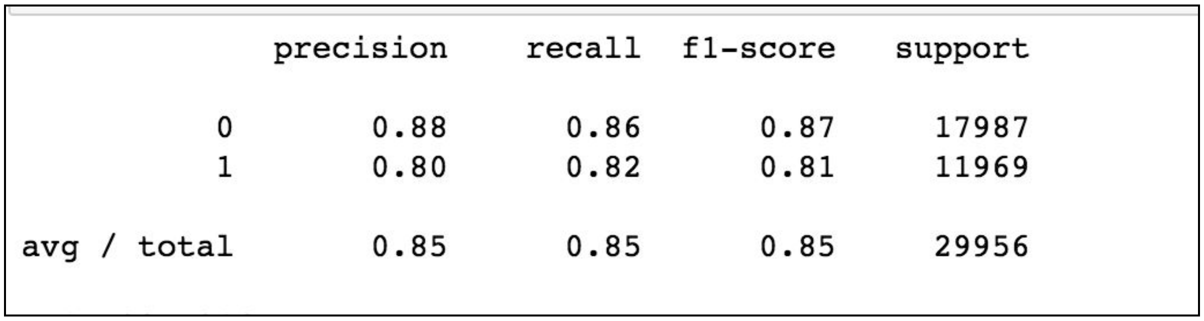


Fig.16. Test AUC for second and third RF models

2.4 Evaluation

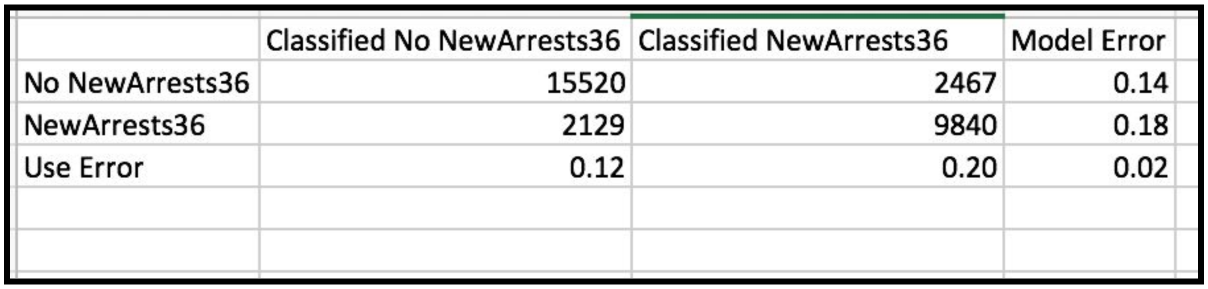
We generated a confusion table for the finalized RF model with max_feat = 30 and estimators = 900.



```
             precision    recall  f1-score   support

          0       0.88      0.86      0.87     17987
          1       0.80      0.82      0.81     11969

avg / total       0.85      0.85      0.85     29956
```



	Classified No NewArrests36	Classified NewArrests36	Model Error
No NewArrests36	15520	2467	0.14
NewArrests36	2129	9840	0.18
Use Error	0.12	0.20	0.02

Fig.17. Confusion Table and Classification Report for RF model using test samples

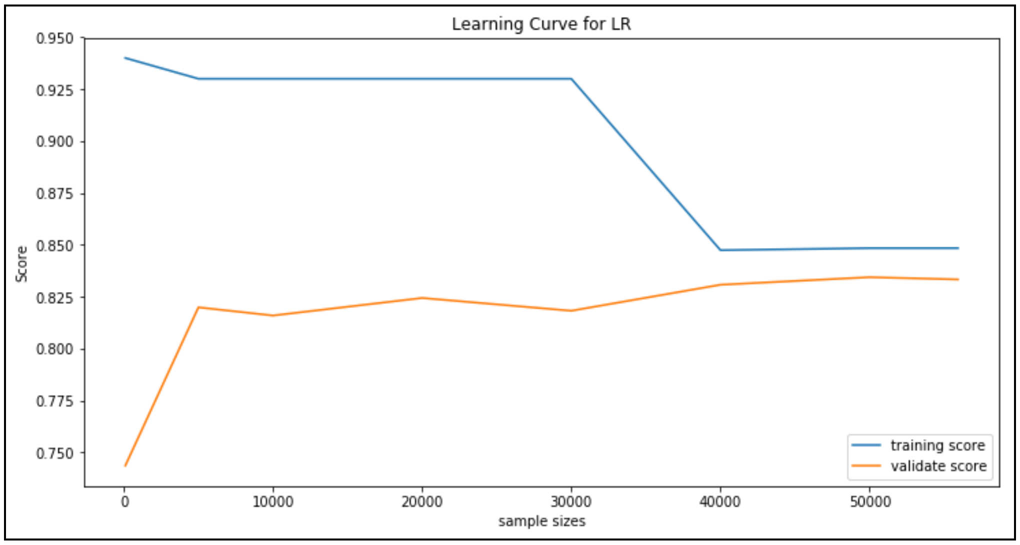
Similar to our LR model, we saw an overall error of 2%, which is a fairly good result. Given the base rate of NewArrest36 = 40%, we correctly forecasted 80% of the total cases; with base rate of No NewArrest36 = 60%, we correctly forecasted 88% of total cases.

In practice, 40 in 100 of the overall population of individuals will be charged with a new crime within 36 months. This gives us a 4 to 1 ratio of true positives to false positives: for every four true positives, we see 1 false positive.

Logistic Regression Vs. Random Forest

When we compared the two finalized LR and RF models, we saw a slight advantage for the RF model: Random Forest achieves a 2% overall error, as compared to a 3% overall error in Logistic Regression. The 1% improvement came from better precision(=TP/P). When using RF model in practice, we forecasted that 40 out of 100 individuals would be charged with crime within 36 months; however, there were (slightly less than) 2 out of 100 *False Negative* cases--people who did recidivate, but which we predicted would not. With the LR model, given that it has only 1% higher overall error, we got the same result in our assessment of 100 individuals. Of course, in deployment the total population will be far larger, and the however, and a 1% difference will be quite significant in practice.

We also took a look at the learning curve for LR model. With smaller sample size, we suffered from bias, and the model overfit the data; as sample size grew, the training score and validation score all converged to 0.84--after the sample size reaches 40,000, we won't necessarily benefit from adding more data points. Indeed, 0.84 was exactly what we ended up getting from the RF model. (As we have learned, RF model has a training score always close to 1, there is no need to plot the learning curve for RF since we always validated the OOB error in the previous section.)



5. Deployment

The technical challenges of deploying the final model will not be overly prohibitive. The final model is a relatively straightforward implementation of the random forest algorithm, is restricted to just 30 features, and does not require too much preprocessing. This is a good feature for this problem, since the consumers of the model predictions will be government workers, who may lack the necessary technical infrastructure to implement a model that is any more complex than a basic implementation of the random forest algorithm.

For these reasons, we believe it would be overkill to implement a Hadoop or other big data based solution. The deployment can be handled using Predictive Modeling Markup Language (PMML). PMML is an XML-based format that allows software applications to interact with the model produced by our team. The format was first introduced in 1998 and supports random forests, which makes it appropriate for our use case. The end user will see a prediction, likely rounded or binned in some way, of the defendant's risk of a new arrest within the next three years.

There are multiple issues that must be addressed before using the predictions derived from this model in practice. First, front-line practitioners must be educated on the target variable, the predictors used, and how the algorithm functions. This is a nontrivial task, and may result in the implementation of a more interpretable algorithm such as a single decision tree. Assuming the education and model predictions are in place, significant policy decisions need to be made about what to do with the predictions. Should all individuals who score under a certain threshold be diverted, and all those above a certain threshold be incarcerated? Should the predictions function as guidance, or as gospel? What should be done in the case of missing or incomplete data? How will overrides to the algorithm's predictions and recommendations be handled? These are not easy questions, especially in such a high-stakes context. But they are also good questions to ask before implementing any prediction instrument.

The increasing use of prediction instruments in criminal justice contexts presents many thorny issues. Pro Publica recently published a thorough analysis of a risk assessment instrument in Broward County, Florida (Angwin et al 2016). They found that Black defendants had a higher false positive rate than whites. The popular media and most readers likely took this to mean that the algorithm was miscalibrated, i.e. that Black defendants were inaccurately rated more risky. Other subsequent analysis by Corbett-Davies, et. al (2017) indicated that it is not quite so simple - the overall recidivism rates vary significantly by race, so that within each risk category, the proportion of defendants who reoffend is approximately the same regardless of race.

Of course, data is a mirror of society, and these patterns and the data almost certainly reflect biases in our criminal justice system. (This potential problem is an example of what d'Allessandro et al (2017)'s "Conscientious Classification" framework calls "Discrimination in Data"). For instance, our model is based heavily on criminal history data. Poor and minority communities have a heavier police presence than affluent or white communities. The presence of more law enforcement may result in more arrests for minor infractions that might otherwise go unnoticed in different communities. Minority or impoverished individuals may subsequently be rated as higher risk simply because they are at increased risk of being arrested, and not that they are at increased risk of committing a crime. Future implementations should tweak the target variable to predict only serious or violent crime, or rely less on criminal history information and more on potentially less biased measures of community attachment, prosocial skills, and the like.

If this prediction model were deployed, it would also need to be continually monitored and evaluated. The data sources that feed the algorithm must remain intact and employ the same definitions that were used to train the model. The target variable - a new arrest within three years - is a long-lead outcome, so while it will be important to continually assess how well the model performs, it should also be recognized that a great deal of the predicted arrests may not occur for one or two more years. Therefore, low accuracy during the initial rollout should not be seen as a failing of the model, but rather a function of the problem formulation. In addition, any deployment outside of New York City should should validated--these patterns likely differ somewhat by geography. Finally, we should note that while the use of algorithmic risk assessments has been upheld as constitutional (State v. Loomis 2016), practitioners must be vigilant to ensure that they employ risk assessments in ways that do not violate relevant legal statutes.

6. Conclusion

The method described above shows it is possible to accurately forecast new arrests using common machine learning methods. With accurate predictions, criminal justice operations and case processing could be improved - primarily by targeting services or incapacitation to those that need it, and by diverting those that pose little to no risk of future crimes. This is not unlike commercial churn modeling scenarios - policy makers can target interventions to those most at risk of some adverse outcome, and ignore or otherwise do away with those that pose no risk of the adverse outcome.

Accurate predictions are, in a sense, the *easy* part in applying machine learning techniques to criminal justice problems. But how to use these predictions about crimes not yet committed in deciding how to treat individuals is fraught with moral and ethical concerns. Only some of those concerns are addressed here. The stakes are high, and generally the system is not amenable to A/B testing like a typical web company. If the research community can find ways to balance the power of predictive algorithms with the right to a fair trial and the concept of innocent until proven guilty, the criminal justice system and society as a whole will be better off.

References

Angwin, J., Larson, J., Mattu, S. Machine bias. ProPublica, 2016

Berk, R., Heidari, H., Jabbari, S., Kearns, M., & Roth, A. (2017). Fairness in Criminal Justice Risk Assessments: The State of the Art. arXiv preprint arXiv:1703.09207.

Chouldechova, A. (2016) “Fair Prediction With Disparate Impact: A Study of Bias in Recidivism Prediction Instruments.” arXiv:1610.075254v1

Copas, J. & Marshall, P. (1998) The Offender Group Reconviction Scale. Applied Statistics, 47, 159-71.

Corbett-Davies, S., Pierson, E., Feller, A., Goel, S., & Huq, A. (2017). Algorithmic decision making and the cost of fairness. arXiv preprint arXiv:1701.08230.

Corbett-Davies, S., Pierson, E., Feller, A., & Goel, S. (2016). A computer program used for bail and sentencing decisions was labeled biased against blacks. its actually not that clear. Washington Post.

d'Alessandro B., O'Neil C., and LaGatta T. (2017). Conscientious Classification: A Data Scientist's Guide to Discrimination-Aware Classification. Big Data., 5(2): 120-134. https://doi.org/10.1089/big.2016.0048

Kleinberg, J., Mullainathan, S., & Raghavan, M. (2016). Inherent trade-offs in the fair determination of risk scores. arXiv preprint arXiv:1609.05807.

Nowok, B., Raab, G. M., & Dibben, C. (2016). Synthpop: Bespoke Creation of Synthetic Data in R. Journal of Statistical Software, 74(11), 1-26. doi:10.18637/jss.v074.i11. URL https://www.jstatsoft. org/article/view/v074i11.

State v. Loomis, 881 N.W.2d 749, 770–71 (Wisconsin Supreme Court Case, 2016).

Zafar, M. B., Valera, I., Rodriguez, M. G., & Gummadi, K. P. (2016). Fairness Beyond Disparate Treatment & Disparate Impact: Learning Classification without Disparate Mistreatment. arXiv preprint arXiv:1610.08452.

Team contributions:

1. **Jash :** data understanding and data preparation, evaluating racial bias in models, report editing
2. **Connor :** data collection and cleaning, initial modeling, report editing
3. **Rob:** drafting and discussing deliverables and replies; double checking LR models; building early RF models (superseded by Stella's); writing and editing of final report; collected research and citations on deployment, ethics, and policy
4. **Stella:** feature exploration, evaluating baseline LR and RF models, model improvement and evaluation, report editing.

Appendix: Features

Field	Type	Description
gender	Binary	Gender
race_American Indian/Alaskan Native	Binary	If the race is American Indian/Alaskan Native
race_Asian/Pacific Islander	Binary	
race_Black	Binary	
race_Black-Hispanic	Binary	
race_Unknown	Binary	
race_White	Binary	
race_White-Hispanic	Binary	
category_Felony	Binary	If the conviction category is Felony
category_Infraction	Binary	
category_Misdemeanor	Binary	
category_Traffic Infraction	Binary	
category_Unspecified	Binary	
category_Violation	Binary	
AgeAtFirst	Numeric	Age at first conviction
OrderProtect	Numeric	# of orders of protection
Housing	Binary	
Age	Numeric	current age

MisdConviction	Binary	
FelonyConviction	Binary	
ViolentConviction	Binary	
ViolentOffense	Binary	Is current offense violent?
Gang	Binary	Individual identified in gang database?
BWO24	Numeric	# of failures to appear for court in previous 24 months
BWO24Plus	Numeric	# of FTA >24months
Arrests	Numeric	# of Arrests
FelArrests	Numeric	# of felony arrests
MisdArrests	Numeric	# of misdemeanor arrests
ViolArrests	Numeric	# of violent arrests
Arrests5	Numeric	# of arrests in previous 5 years
FelArrests5	Numeric	
MisdArrests5	Numeric	
ViolArrests5	Numeric	
Cvts	Numeric	# of criminal convictions
FelCvts	Numeric	# of felony criminal convictions
MisdCvts	Numeric	# of misdemeanor convictions
ViolCvts	Numeric	Violent convictions
DrugCvts	Numeric	drug convictions
DrugFelCvts	Numeric	felony drug convictions
SexCvts	Numeric	sex offense convictions

BurgFelCvts	Numeric	felony burglary convictions
RobFelCvts	Numeric	felony robbery convictions
LarcFelCvts	Numeric	larceny felony convictions
AssFelCvts	Numeric	assault felony convictions
WeapFelCvts	Numeric	weapons felony convictions
Cvts5	Numeric	# of criminal convictions in previous 5 years
CvtCounties	Numeric	# of distinct counties with criminal conviction
FelCvts5	Numeric	
MisdCvts5	Numeric	
ViolCvts5	Numeric	
DrugCvts5	Numeric	
CvtCounties5	Numeric	
Incarcerations	Numeric	# of previous incarcerations > 13 days
Jail	Numeric	# of jail sentences
Prison	Numeric	# of prison sentences
Incarcerations5	Numeric	in previous 5 years
Jail5	Numeric	
Prison5	Numeric	
CvtDensity	Numeric	sqrt(Convictions/Career Length+1) as described in
NewArrest36(Target)	Binary	1 for new arrests within 36 month; 0 for no new arrests in 36 month