21641412_CrimeRates

Jannes Eloff

2023-06-19

Introduction

The aim of the following study is to predict the crime rates for countries based on economic related features. Models with good predictive power can give insight, for policymakers, into factors that causes higher crime rates which in turn leads to more effective solutions.

In order to draw the valid conclusions to find a model with the best predictive ability, several different models was optimally tuned and tested on test data in order to look at predictive accuracy. The following models are implemented with varying degrees of success: OLS rgeression, Lasso regression, ridge regression, decision trees, random forest and gradient boosting.

The code used for this project can be found on github using the following link: https://github.com/Jannes1999/CrimeRates_Project This is where the code chunks and functions are stored.

Data

The dataset used looks at different countries and their respective crime rates. Of course there are a plethora of different factors that leads to the crime rate a country faces and including all the factors in a model is impossible. The dataset provides 8 features primarily focusing on economic factors that possibly effects crime rates. The following features are included in the dataset: Unemployment (%), HDI, Population density (per sq km), weapons per 100 persons, per capita income, Gini coefficient, literacy rate and happiness index. All of these variables are self-explanatory and need not any extra explanation.

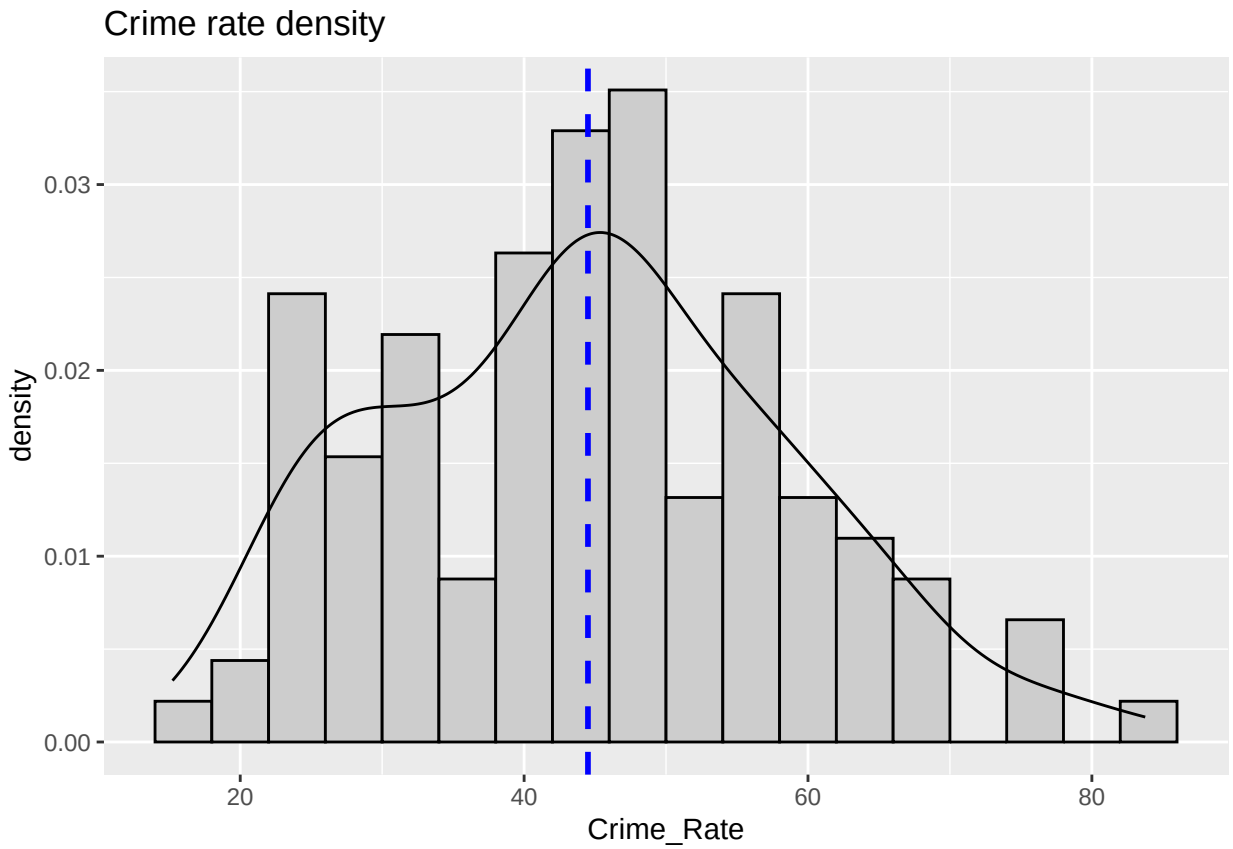
The dataset is available on kaggle: https://www.kaggle.com/datasets/shubhrojyotidey/crime-economics. Not any other formation availble about the dataset (such as the author, how it was collected & when it was collected). Considering this is an analysis on the predictive power of different models the fact tat there is no real extra information available does not hinder the study.

The scope of the dataset suggests the models should be noisy, as there are not many predictors or data points to work with (which is a problem given that we are dealing with a complex target variable).

Initial exploratory data analysis

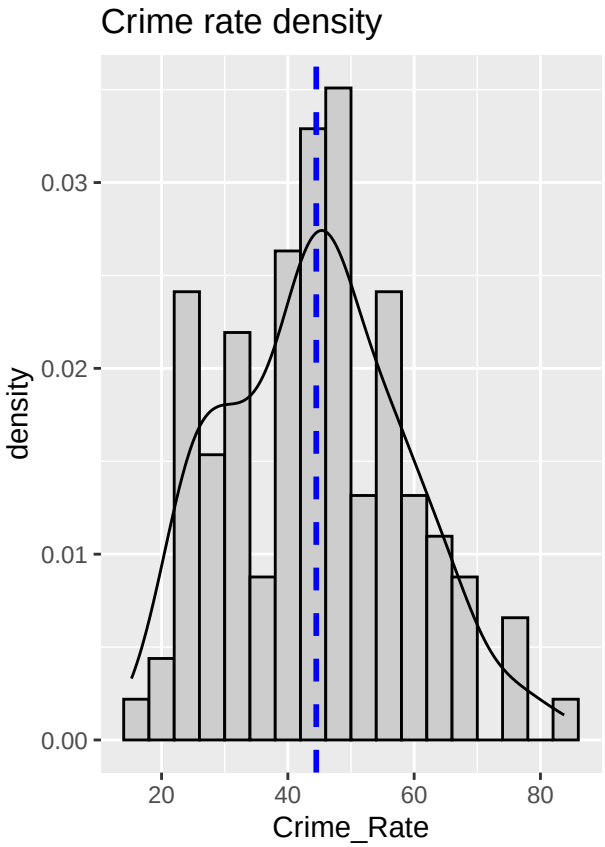
The dataset used is already clean thus no nead to deal with NA's or other irregularities of similar nature. This section mainly looks at the nature of the given target and features. This will aid in the understanding of the data.

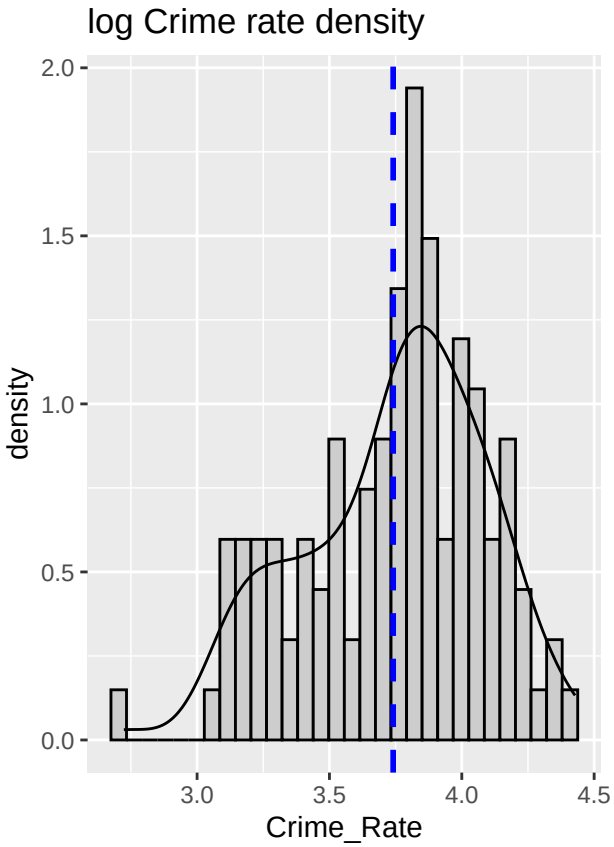
```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine
```




```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

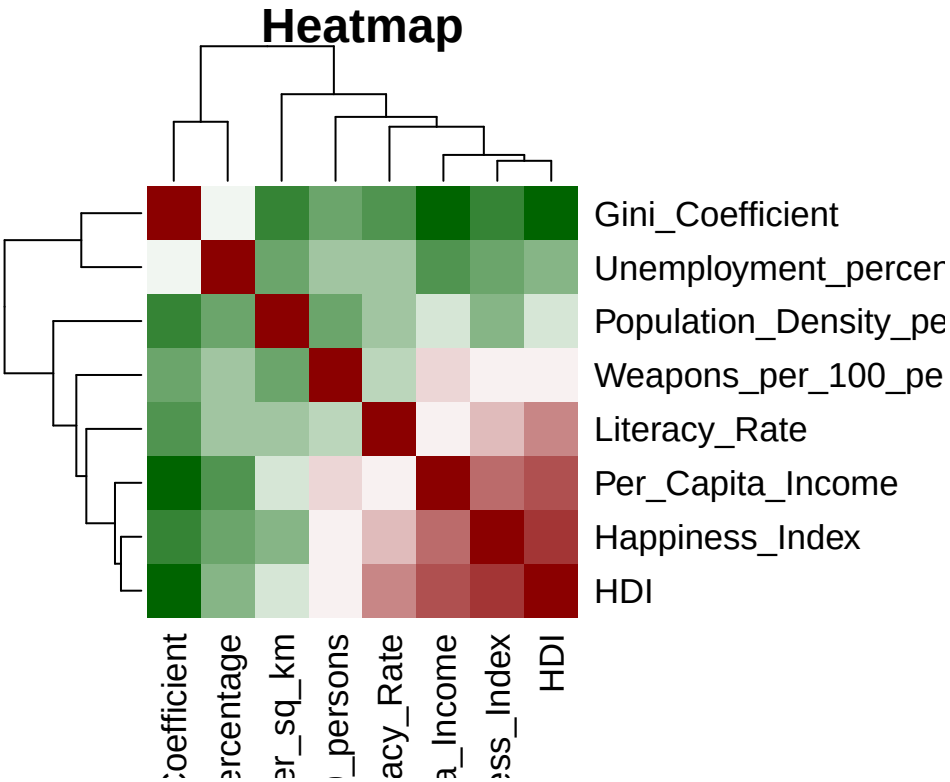
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```





The density plot is approxiamating a somewhat normally distributed look. However for clarity we logged the target variable in o Altering the target with a log transformation will transform most right skewed distributions to be approximately normal (as seen in the figures above).

```
## corrplot 0.92 loaded
```



There is clear positive correlation between HDI and the Happiness index and a strong negative correlation between HDI and the Gini coefficient. These correlations however are not severe enough (> 0.8 or < -0.8) to exclude one of the variables. It is clear as to why these correlations exist. The HDI typically factors in many different variables (which correlates positively with happiness). And similarly some of these factors used in calculation of HDI correlates negatively to the gini coefficient.

Splitting of data

The dataset will be split in a training and test set for the analysis of the different model. A typical 70 / 30 split is performed (70 percent of the data falls into the training set with the remaining 30 percent allocated to the test set).

Multivariate regression

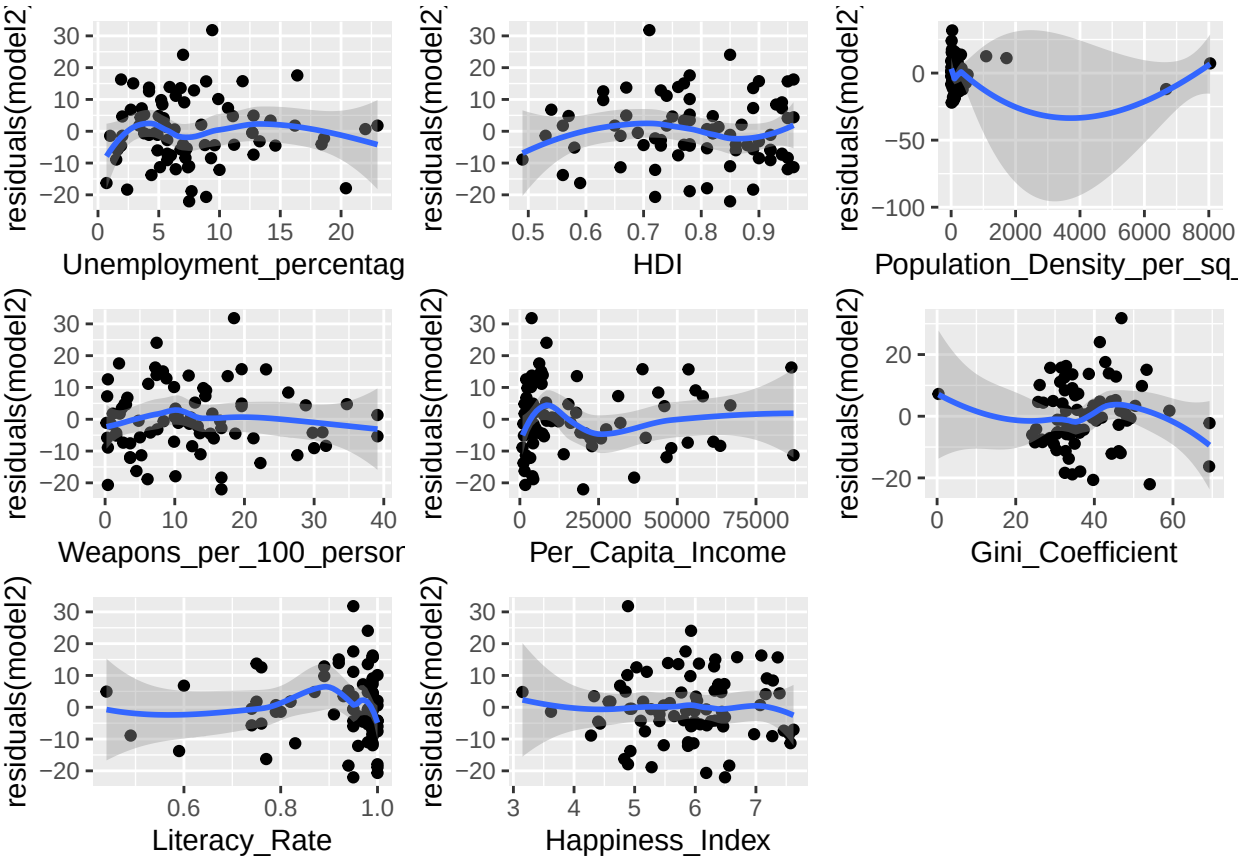
To start of normal OLS regressions can be performed to see how they perform. Typically, they are easy to interpret and clear performance indicators make it easy to understand its validity. They do assume linearity which could potentially oversimplify the relationship between the features and target.

```
##
## ================================================================================================
##                                                Dependent variable:
##                                    ------------------------------------------------------------
##                                                Crime_Rate                      log(Crime_Rate)
##                                          (1)                    (2)                    (3)
## ------------------------------------------------------------------------------------------------
## Unemployment_percentage                0.285                                         0.008
##                                       (0.279)                                       (0.007)
##
## HDI                                 -101.316***            -59.715***             -2.439***
```

	(27.431)	(11.299)	(0.684)
Population_Density_per_sq_km	0.001		0.00001
	(0.001)		(0.00003)
Weapons_per_100_persons	0.026		0.001
	(0.144)		(0.004)
Per_Capita_Income	-0.00000		-0.00000
	(0.0001)		(0.00000)
Gini_Coefficient	0.390**	0.469***	0.008**
	(0.147)	(0.135)	(0.004)
Literacy_Rate	12.907		0.330
	(16.326)		(0.407)
Happiness_Index	4.562*		0.103*
	(2.314)		(0.058)
Constant	67.841***	73.218***	4.377***
	(15.244)	(12.038)	(0.380)
Observations	79	79	79
R2	0.507	0.471	0.489
Adjusted R2	0.451	0.457	0.430
Residual Std. Error	10.997 (df = 70)	10.933 (df = 76)	0.274 (df = 70)
F Statistic	9.011*** (df = 8; 70)	33.873*** (df = 2; 76)	8.357*** (df = 8; 70)
Note:			*p<0.1; **p<0.05; ***p<0.01

In the above table we ran two different regressions. Firstly all the features was used as included in the model, secondly only HDI and the Gini coefficient was added (those that are significant). While logging did not alter the nature of the target in a decisive manner, its worth including the models where Crime Rates (the target variable) was logged. Not one of the models exhibits an adjusted Rsquared of above 0.5 with the normal regressions marginally outperforming the logged ones. Furthermore, the models with only to features contains a lower Std. error than the models containing all the variables.

As mentioned earlier the assumption of linearity could be problematic and should be addressed if needed. To investigate some of the potential nonlinearities present we can plot each feature against the residuals.



Some non-linearities are present above in the following features: Population_Density_per_sq_km, Gini_Coefficient and the Literacy_Rate. Now lets include

```
##
## Crime rate lm models with nonlinearities
## ================================================================
##                                        Dependent variable:
##                                    ----------------------------
##                                     Crime_Rate  log(Crime_Rate)
##                                        (1)            (2)
## ----------------------------------------------------------------
## Unemployment_percentage                0.298          0.008
##                                       (0.284)        (0.007)
##
## HDI                                  -95.942***     -2.361***
##                                       (28.944)       (0.725)
##
## Population_Density_per_sq_km           0.0001        -0.00002
##                                       (0.006)        (0.0002)
##
## Weapons_per_100_persons                0.060          0.002
##                                       (0.153)        (0.004)
##
## Per_Capita_Income                     -0.00000       -0.00000
##                                       (0.0001)       (0.00000)
##
## Gini_Coefficient                       0.657          0.014
```

```
##                                    (0.675)        (0.017)
##
## Literacy_Rate                      139.587         2.417
##                                   (120.534)       (3.020)
##
## Happiness_Index                    4.311*         0.100*
##                                    (2.392)        (0.060)
##
## I(Population_Density_per_sq_km2)   0.00000         0.000
##                                   (0.00000)      (0.00000)
##
## I(Gini_Coefficient2)               -0.004         -0.0001
##                                    (0.008)       (0.0002)
##
## I(Literacy_Rate2)                  -82.772         -1.363
##                                    (78.232)       (1.960)
##
## Constant                           15.206        3.478***
##                                    (48.872)       (1.225)
##
## ------------------------------------------------------------
## Observations                         79             79
## R2                                  0.517          0.493
## Adjusted R2                         0.438          0.410
## Residual Std. Error (df = 67)       11.131         0.279
## F Statistic (df = 11; 67)          6.517***       5.932***
## ============================================================
## Note:                             *p<0.1; **p<0.05; ***p<0.01
```

From the table above we can conclude that introducing nonlinearity into the models makes performance worse all around. From the previous models the Adjusted Rsquared is less and the Residual standard error is higher. We can conclude that the nonlinearities are not captured in the correct manner hence more sophisticated models that accurately capture the nonlinear relationship between features and target should be used for better accuracy.

Regularised Regression

Since we are working with a relatively small dataset regularised regression might be useful to use. While the amount of features are not more than the available data points the lasso and ridge models could still provide valuable insights. Furthermore, for large data sets, the original sample of data may be partitioned into a training set on which to train the model, a validation set on which to validate our models and a test set to evaluate our trained model. However, when we do not have large samples of data, cross-validation is particularly useful. Lasso and Ridge regresions applies penalties to specific features based on their importance. With the lasso penalty some feature coefficients could shrink all the way to 0 whereas in the case of the ridge these coefficients does not shrink all the way to 0.

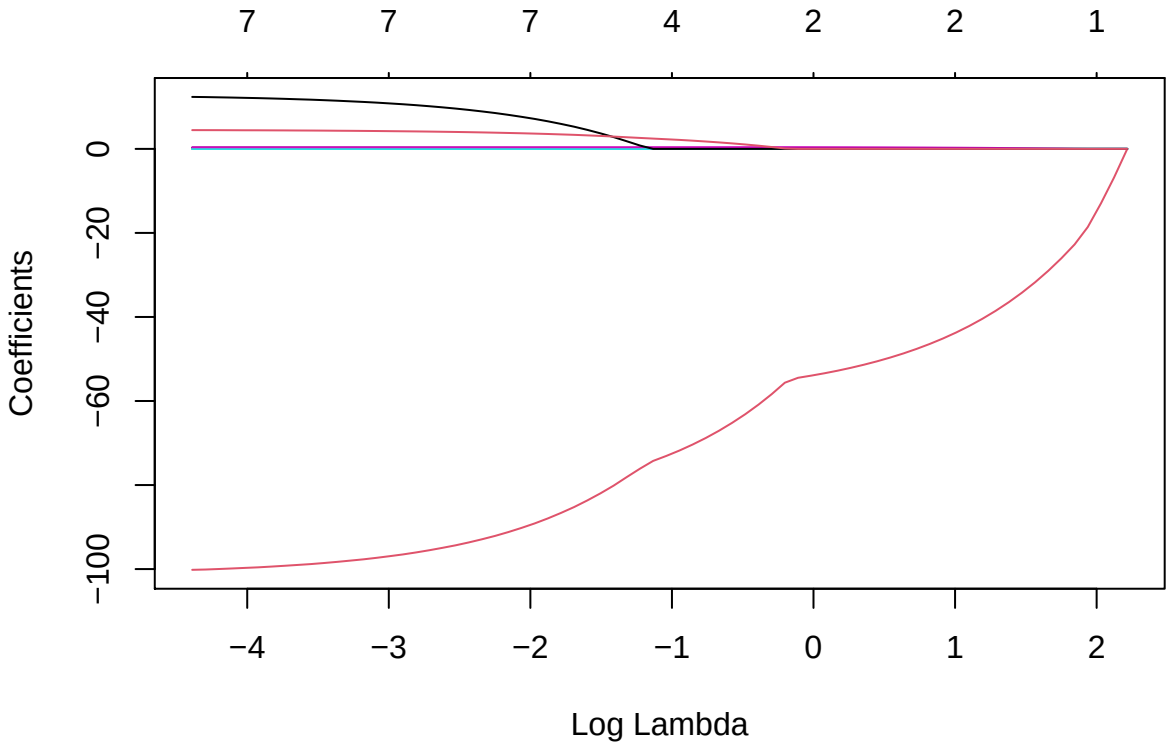
Starting with the Lasso regression we use the glmnet package which allows us to make use of cross validation ans specify alpha which tells glmnet which method to implement: alpha = 0: ridge penalty alpha = 1: lasso penalty (in this case) $0 <$ alpha < 1: elastic net model

Furthermore, glmnet does two things that one needs to be aware of. Standardises features Fits ridge models across wide range of λ values (automatically)

```
## Loading required package: lattice
```

```
##
```

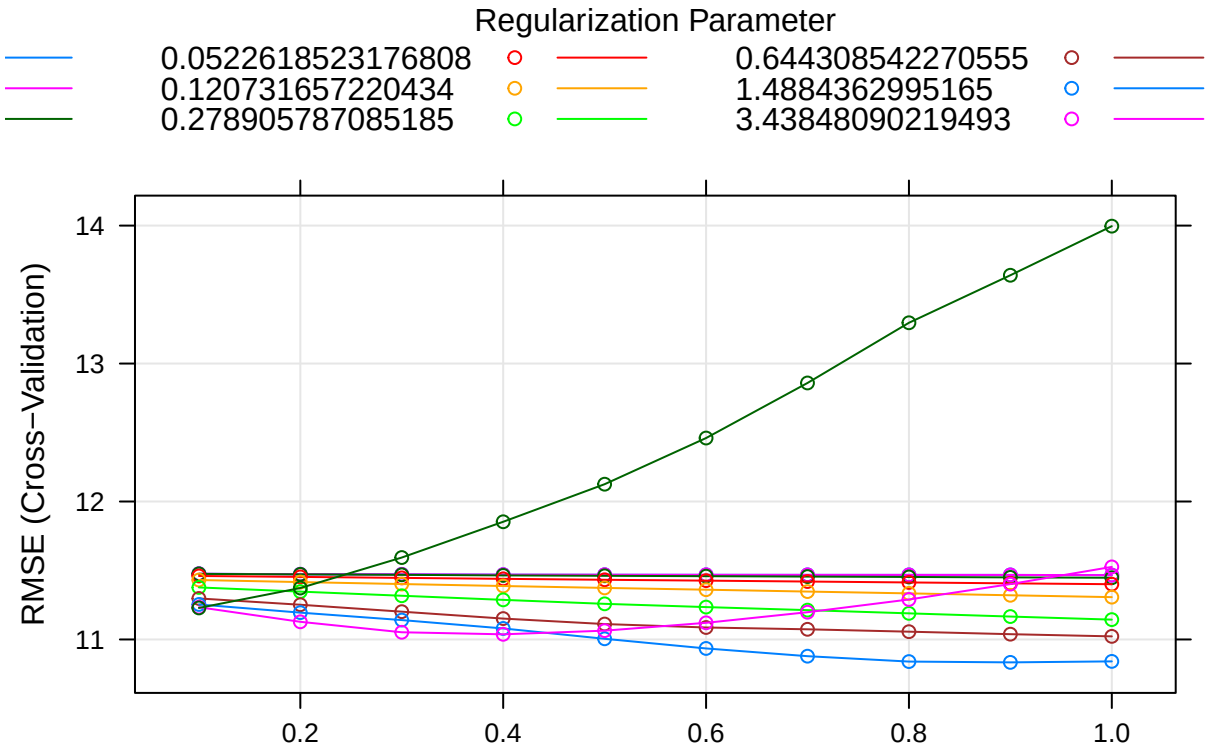
```
## Attaching package: 'caret'
## The following object is masked from 'package:purrr':
##
##     lift
## Loading required package: Matrix
##
## Attaching package: 'Matrix'
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
## Loaded glmnet 4.1-3
```

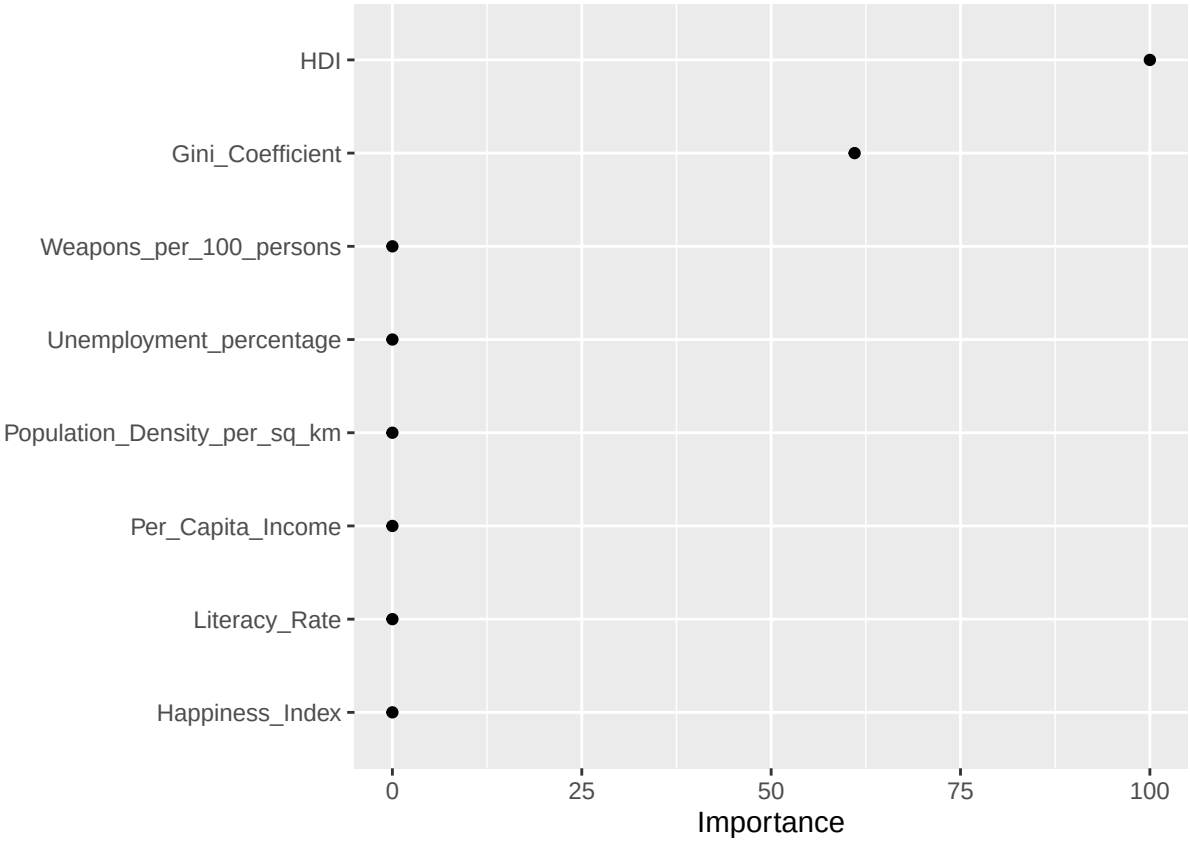


Here is how the coefficienets behave in the lasso (similar in ridge, however the coefficients dissappear completely).

After the model was implemented lambda is one of the hyper parameters that can be tuned in order to produce better results. We can see that, initially, as our penalty (lambda) increases the MSE decreased which suggest that thee normal OLS approach possibly overfits the data. Similar analysis can be done in the ridge regression.

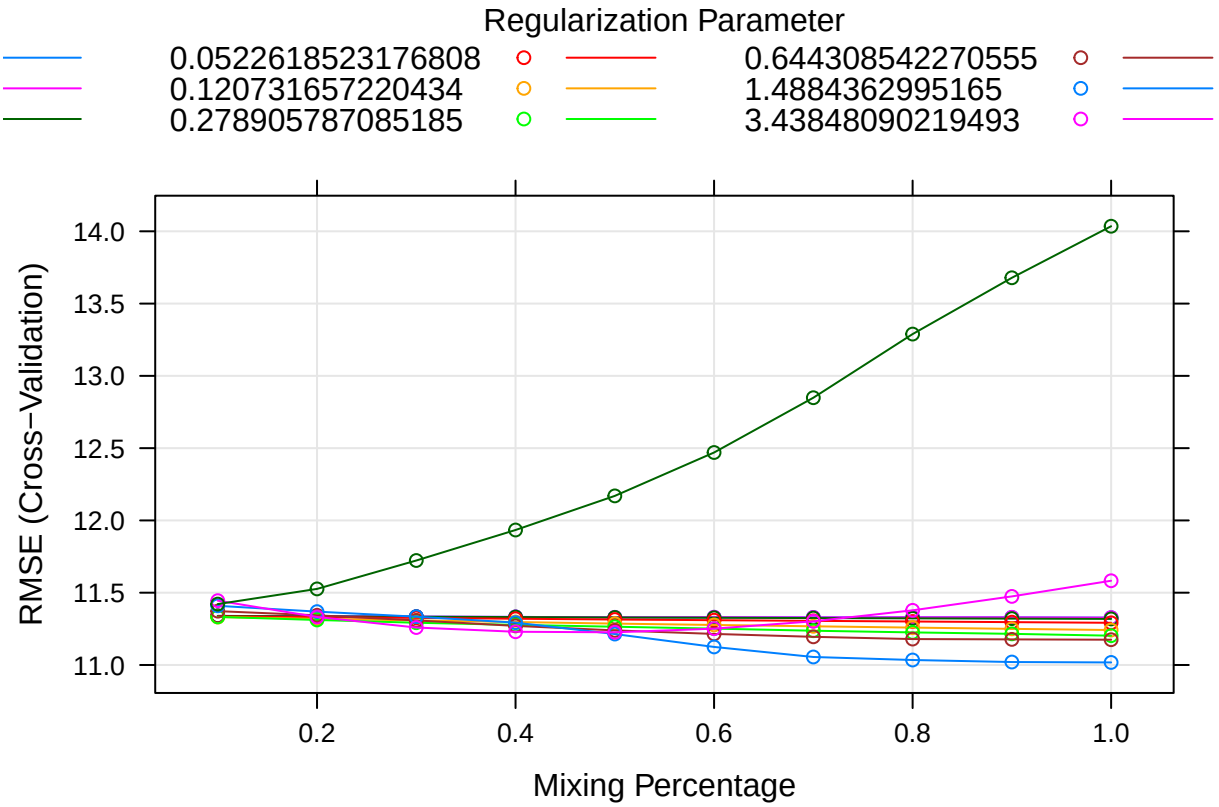
Here are the given feature selection done by both models:





Regularized regression provides many great benefits over traditional GLMs when applied to large data sets with lots of features. It provides a great option for handling the n > p problem, helps minimize the impact of

multicollinearity, and can perform automated feature selection. It also has relatively few hyperparameters which makes them easy to tune, computationally efficient compared to other algorithms discussed in later chapters, and memory efficient.



The figure above optimally select the CV model that produces the lowest RMSE

Now once our hyperparameters are optimally tuned we can look at our results from our training and test sets in order to look at the predictive accuracy:

```
##       RMSE   Rsquare
## 1 10.85627 0.4581301
```

```
##       RMSE    Rsquare
## 1 12.21127 0.03603965
```

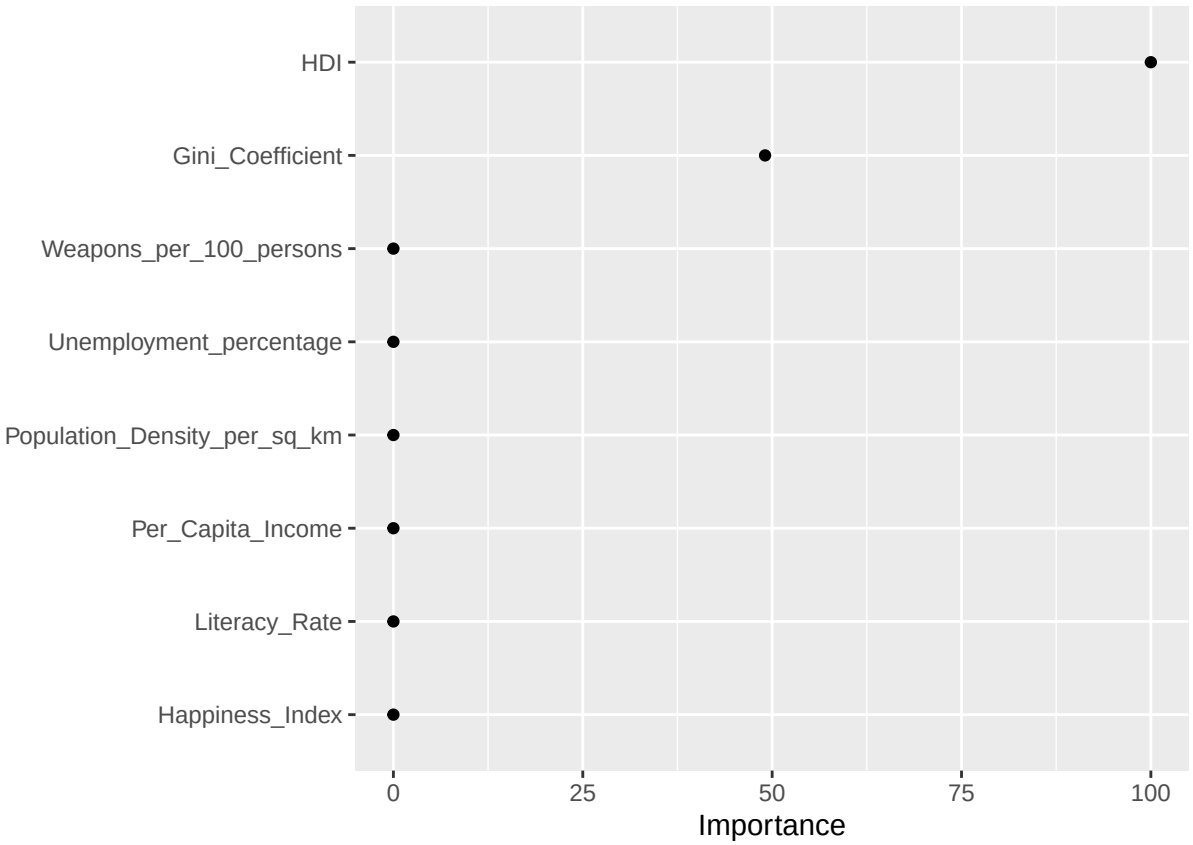
For the lasso model The discrepency between the train and the test is substantial. The train set exhibit an RMSE of 10.85v whereas the test set has a RMSE of 12.21

```
##       RMSE   Rsquare
## 1 10.52564 0.4906328
```

```
##      RMSE    Rsquare
## 1 12.2739 0.02612746
```

For the Ridge regression the RMSE difference (between train and test) is similar to that of the lasso. The RMSE of the train data is 10.52 and the RMSE of the test data is 12.27

These results are comparable to the linear models displayed earlier, hence there are more sophisticated models needed. For further clarity on the variables these regularised models find important the following plot shows variable importance:



This variable importance plot suggest that HDI and Gini coefficient are the only variables that are considered "important".

Tree based methods

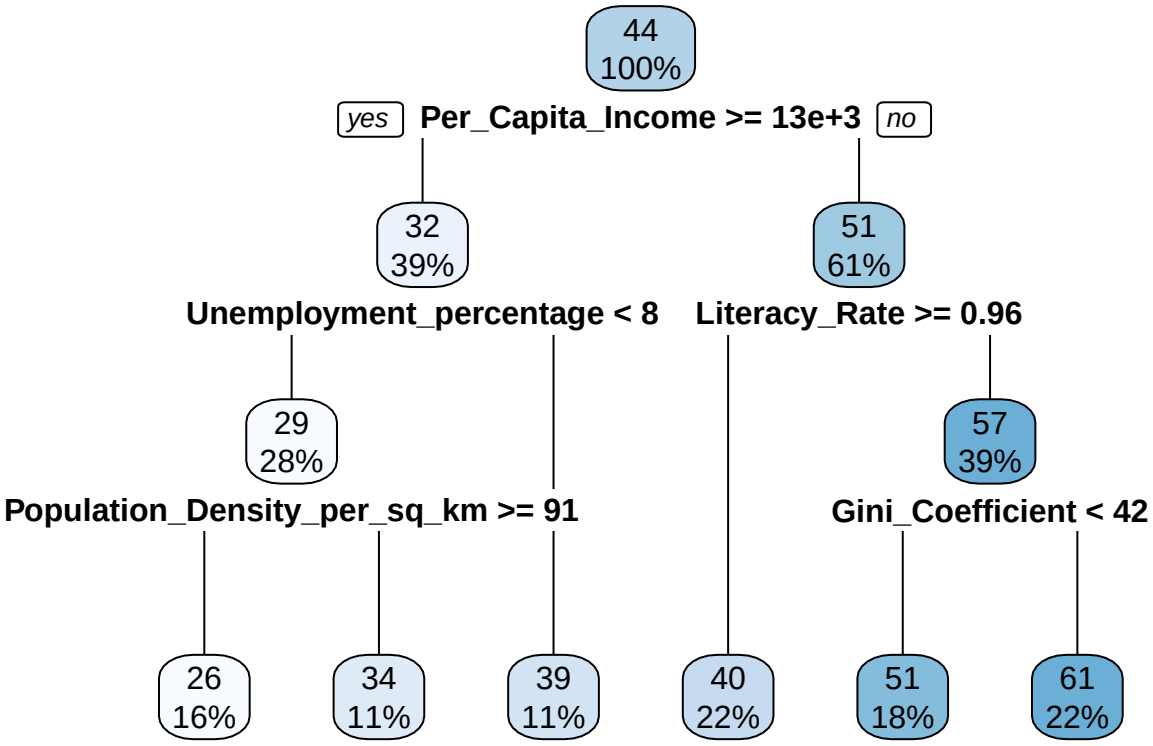
Decision trees

The following section will be looking at tree based models (decision trees, random forrest, gradient boosting).

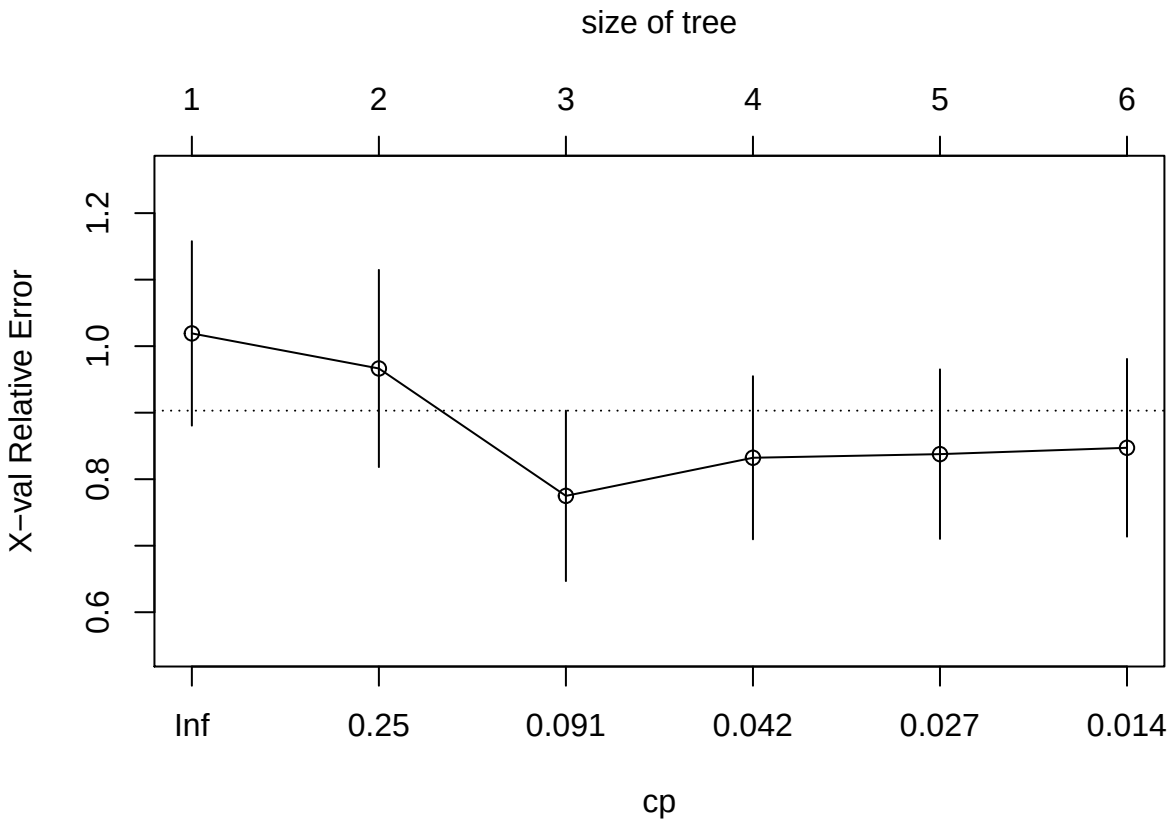
Tree-based models are a class of nonparametric algorithms that work by partitioning the feature space into a number of smaller (non-overlapping) regions with similar response values using a set of splitting rules. When looking at decision tree The objective at each node is to find the "best" feature (xi) to partition the remaining data into one of two regions (R1 and R2) such that the overall error between the actual response (yi) and the predicted constant (ci) is minimized

```
## n= 79
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 79 17182.7800 43.50747
##    2) Per_Capita_Income>=13101.5 31  3002.2500 32.36065
##      4) Unemployment_percentage< 8 22  1540.8630 29.47364
##        8) Population_Density_per_sq_km>=91 13   383.8589 26.16538 *
##        9) Population_Density_per_sq_km< 91 9   809.2114 34.25222 *
##      5) Unemployment_percentage>=8 9   829.7924 39.41778 *
##    3) Per_Capita_Income< 13101.5 48  7841.1050 50.70646
##      6) Literacy_Rate>=0.955 17  1454.3500 39.99588 *
```

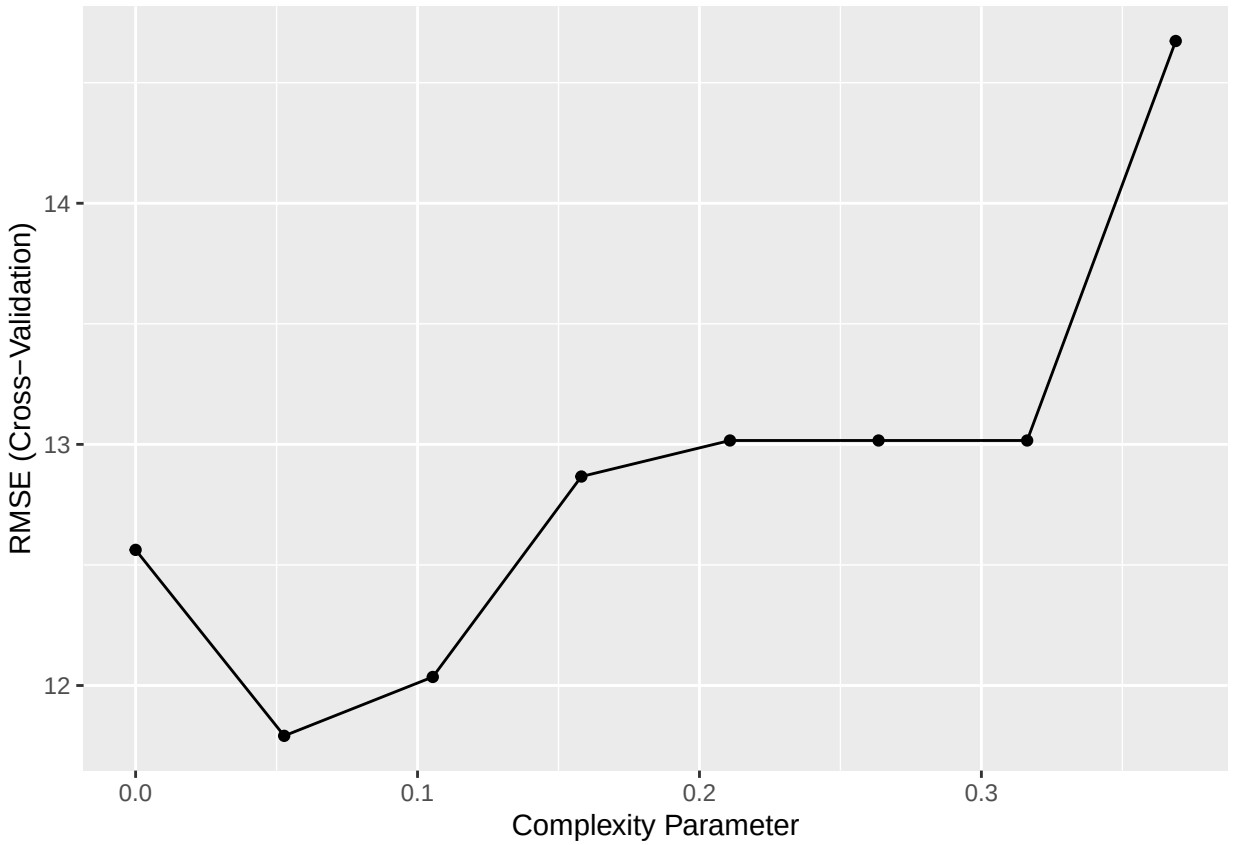
```
##      7) Literacy_Rate< 0.955 31  3367.1220 56.58000
##       14) Gini_Coefficient< 42.35 14  1068.3770 50.95571 *
##       15) Gini_Coefficient>=42.35 17  1491.1840 61.21176 *
```



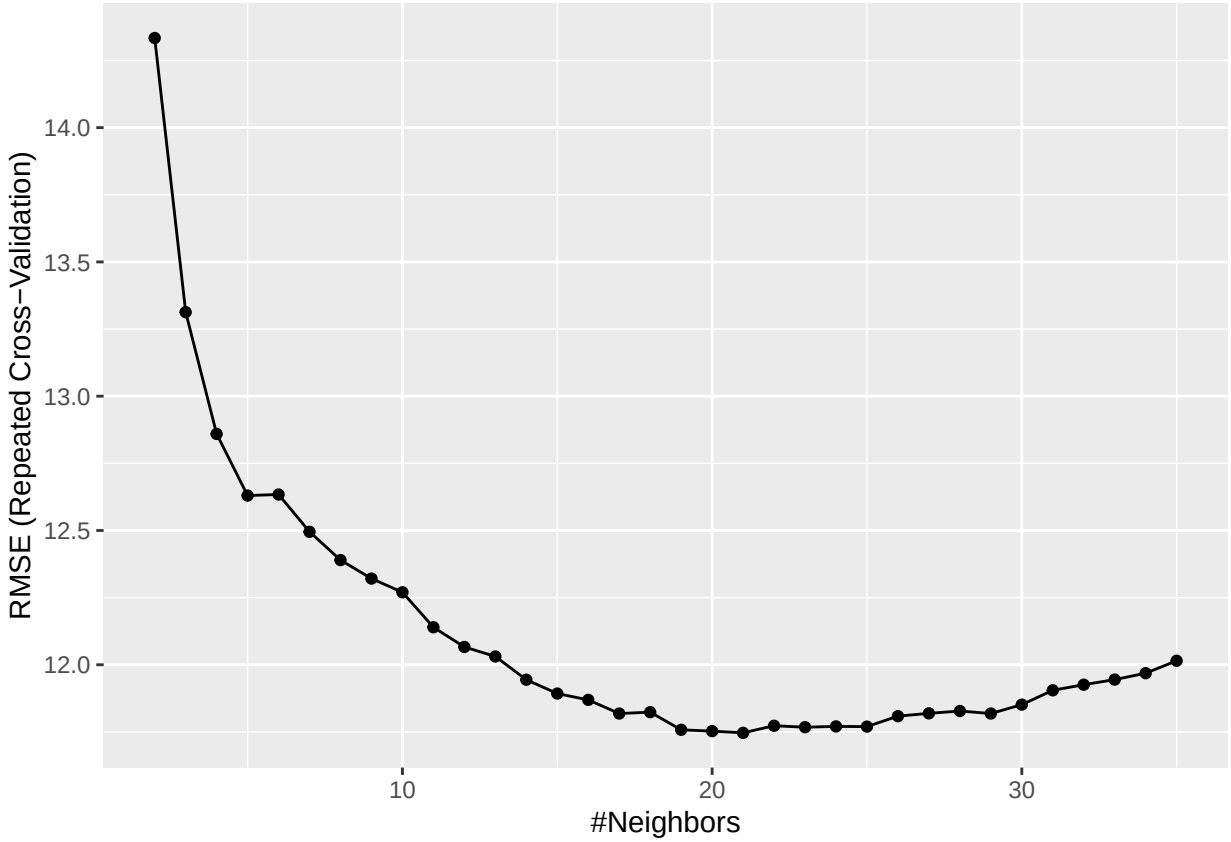
The First figure and table shows exactly how the decision tree was split and where. The first variable split, the variable that gave the largest reduction in SSE, was Per_Capita_Income.



This plot illustrating the relative cross validation error (y-axis) for various cp values (lower x-axis). Smaller cp values lead to larger trees (upper x-axis). Using the 1-SE rule, a tree size of 3 provides optimal cross validation results.



Cross-validated accuracy rate for the 20 different alphaparameter values in our grid search. Lower alpha values (deeper trees) help to minimize errors.




```
## [1] 13.31566
```

```
## [1] 12.84662
```

Above is the RMSE for the normal decision tree and the decision tree using 10-fold cross validation respectively. Given the size of the data set the cross validation helped to lower the RMSE from 13.31 to 12.84

Random forrest

Bagging trees introduces a random component into the tree building process by building many trees on bootstrapped copies of the training data. Bagging then aggregates the predictions across all the trees; this aggregation reduces the variance of the overall procedure and results in improved predictive performance. However, as we saw in Section 10.6, simply bagging trees results in tree correlation that limits the effect of variance reduction.

The model already have default settings regarding the hyperparameters which tend to be optimal, thus the following OOB (out of box) RMSE was reached

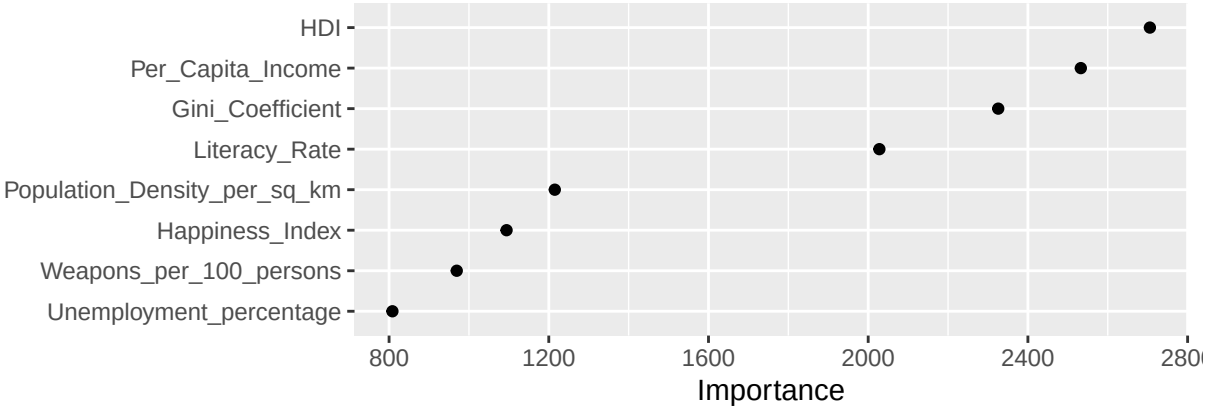
```
## [1] 10.95216
```

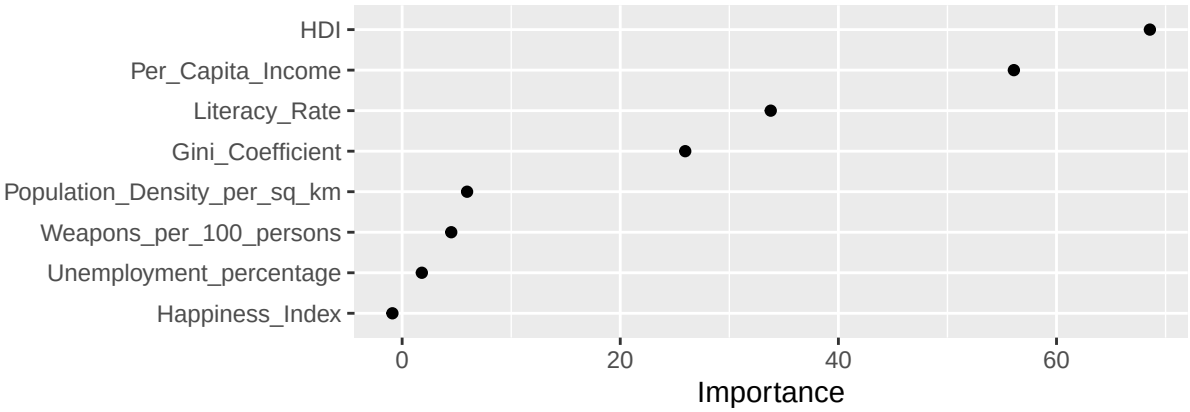
The main hyperparameters to consider include:

The number of trees in the forest The number of features to consider at any given split:
m t r y

The complexity of each tree The sampling scheme The splitting rule to use during tree construction

Random forests provide a very powerful out-of-the-box algorithm that often has great predictive accuracy. They come with all the benefits of decision trees (with the exception of surrogate splits) and bagging but greatly reduce instability and between-tree correlation.





Permutation importance does not reflect to the intrinsic predictive value of a feature by itself but how important this feature is for a particular model.

The resulting VIPs are displayed in Figure 11.5. Typically, you will not see the same variable importance order between the two options; however, you will often see similar variables at the top of the plots (and also the bottom). Consequently, in this example, we can comfortably state that there appears to be enough evidence to suggest that three variables stand out as most influential: (1) HDI (2) Per Capita Income (3) Literacy rate (4) Gini coefficient

These variable importance plots differs drastically from variable importance plot from regularised regression. The variables Per_Capita_Income and Literacy_Rate are not important for the Lasso model however they are important in the random forrest model. This suggest a nonlinnear relationship that is not captured in regularised models.

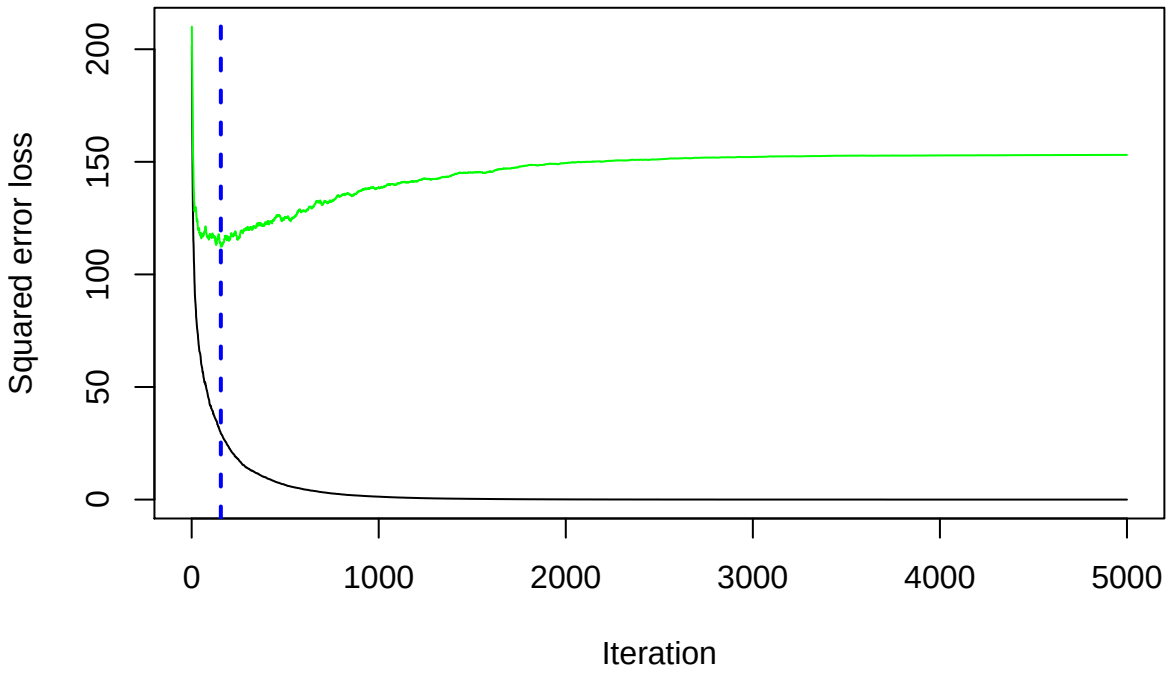
Gradient boosting

Gradient boosting builds an ensemble of shallow trees with each tree learning and building on the previous one (as oppose to random forests, that builds an ensemble of deep independent trees). Boosting can be applied to any model however they are typically and most effectively applied to decision trees (due to being tailored for models exhibiting high bias and low variance).

Here is a snippet of some analysis that can be performed using xgboost, however this is only to showcase one particular ability of gradient bossting.

```
## Warning: package 'xgboost' was built under R version 4.0.5
```

```
## [1] 10.59762
```




```
## [1] 156
```

Now given the package gbm we have already found the optimal number of trees which is 156 and the learning rate of 0.1 (usually the default). Thus, these optimal hyperparameters can now be used to predict crime rates.

Results/Conclusion

The difference in the importance of features in linear models compared to random forest could concievably be because of the linearity assumed by the linear models. The allowance for non linnearities gives deeper insight into the relationship between features and the target hence lower mse's are found in these more complex models.

That being said most models used exhibit RMSE between 10 and 13 without much improvement through processes such as Cross-Validation. This indicates that we are either dealing with a poor set of features (many of the models suggested that some features are obsolete), or other, more advanced, models should be incorporated such as nearul nets, gradient boosting, etc.