DATA 621 – Business Analytics and Data Mining

Homework #3 Assignment Requirements

Alain Kuiete Tchoupou

1. INTRODUCTION

The objective of this assignment is to build a logistic regression model on a training data to predict whether a neighborhood is at risk for high crime levels.

Table: 3.1.1: Short description of the variables of interest in the data set.

	Variables	Description
	zn	proportion of residential land zoned for large lots (over 25000 square feet) (predictor variable)
	indus	proportion of non-retail business acres per suburb (predictor variable)
	chas	a dummy var. for whether the suburb borders the Charles River (1) or not (0) (predictor variable)
	nox	nitrogen oxides concentration (parts per 10 million) (predictor variable)
	rm	average number of rooms per dwelling (predictor variable)
	age	proportion of owner-occupied units built prior to 1940 (predictor variable)
	dis	weighted mean of distances to five Boston employment centers (predictor variable)
	rad	index of accessibility to radial highways (predictor variable)
	tax	full-value property-tax rate per $10,000 (predictor variable)
	ptratio	pupil-teacher ratio by town (predictor variable)
	black	1000(B k - 0.63) 2 where B k is the proportion of blacks by town (predictor variable)
	lstat	lower status of the population (percent) (predictor variable
	medv	median value of owner-occupied homes in $1000s (predictor variable
	target	whether the crime rate is above the median crime rate (1) or not (0) (response variable)

2. DATA EXPLORATION

head(crime.train)

We explore the top of the dataset

Table 3.2.1: Head of the train data

	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	lstat	medv	target
1	0	19.58	0	0.605	7.929	96.2	2.0459	5	403	14.7	3.7	50	1
2	0	19.58	1	0.871	5.403	100	1.3216	5	403	14.7	26.82	13.4	1
3	0	18.1	0	0.74	6.485	100	1.9784	24	666	20.2	18.85	15.4	1
4	30	4.93	0	0.428	6.393	7.8	7.0355	6	300	16.6	5.19	23.7	0
5	0	2.46	0	0.488	7.155	92.2	2.7006	3	193	17.8	4.82	37.9	0
6	0	8.56	0	0.52	6.781	71.3	2.8561	5	384	20.9	7.67	26.5	0

All the 12 predictors are numeric.

We look at the structure of variables

str(crime.train)

```
## 'data.frame': 466 obs. of 13 variables:
 ## $ zn : num 0 0 0 30 0 0 0 0 0 80 ...
 ## $ indus : num 19.58 19.58 18.1 4.93 2.46 ...
 ## $ chas : int 0 1 0 0 0 0 0 0 0 0 ...
 ## $ nox : num 0.605 0.871 0.74 0.428 0.488 0.52 0.693 0.693 0.515 0.392 ...
 ## $ rm : num 7.93 5.4 6.49 6.39 7.16 ...
 ## $ age : num 96.2 100 100 7.8 92.2 71.3 100 100 38.1 19.1 ...
 ## $ dis : num 2.05 1.32 1.98 7.04 2.7 ...
 ## $ rad : int 5 5 24 6 3 5 24 24 5 1 ...
 ## $ tax : int 403 403 666 300 193 384 666 666 224 315 ...
 ## $ ptratio: num 14.7 14.7 20.2 16.6 17.8 20.9 20.2 20.2 20.2 16.4 ...
 ## $ lstat : num 3.7 26.82 18.85 5.19 4.82 ...
```

$ medv : num 50 13.4 15.4 23.7 37.9 26.5 5 7 22.2 20.9 ...
$ target : int 1 1 1 0 0 0 1 1 0 0 ...

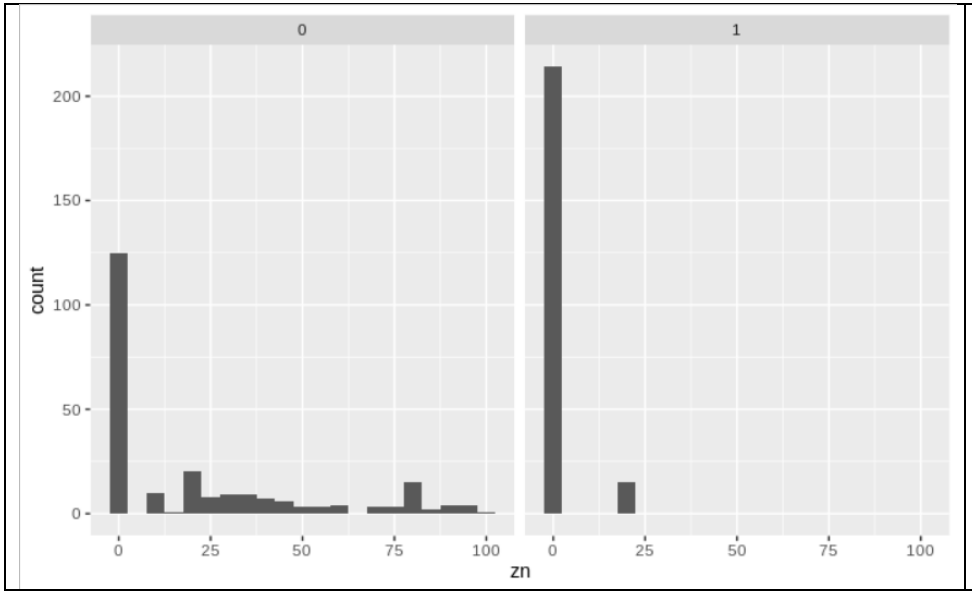
The predictor chas can be classified as factor of two levels

summary(crime.train)

Table 3.2.2: Summary of the train data

	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	lstat	medv	target
Min	0.00	0.460	0.0000	0.3890	3.863	2.90	1.130	1.00	187.0	12.6	1.730	5.00	0.0000
1stQuantile	0.00	5.145	0.0000	0.4480	5.887	43.88	2.101	4.00	281.0	16.9	7.043	17.02	0.0000
Median	0.00	9.690	0.0000	0.5380	6.210	77.15	3.191	5.00	334.5	18.9	11.350	21.20	0.0000
Mean	11.58	11.10	0.0708	0.5543	6.291	68.37	3.796	9.53	409.5	18.4	12.631	22.59	0.4914
3rdQuantile	16.25	18.10	0.0000	0.6240	6.630	94.10	5.215	24.00	666.0	20.2	16.930	25.00	1.0000
Max	100.00	27.74	1.0000	0.8710	8.780	100.0	12.127	24.00	711.0	22.0	37.970	50.00	1.0000

There are no missing values We can look for correlations between variables



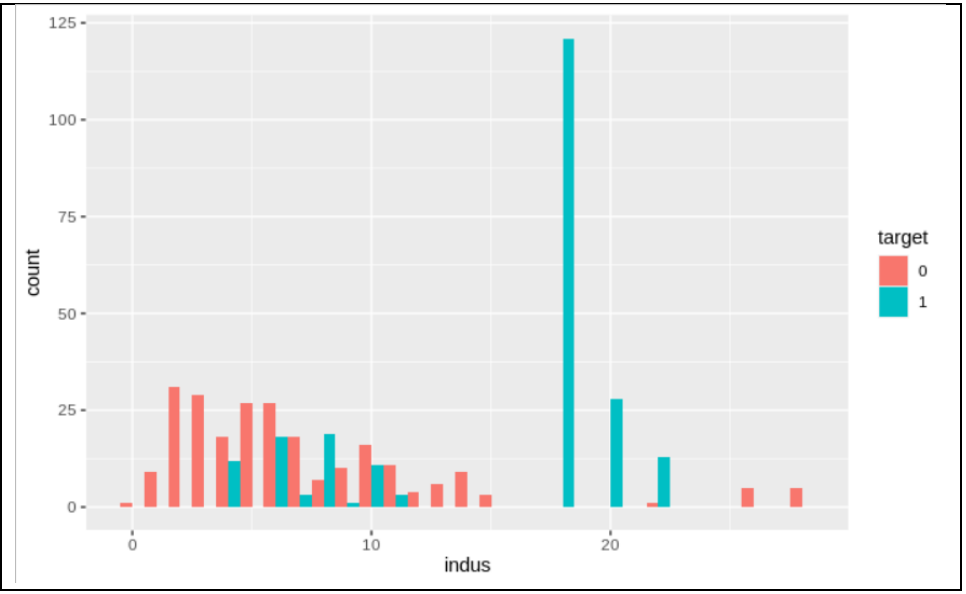


Fig.3.2.1 a : Histogram of the zn predictor by factors of target. B: Histogram of indus by the factors of target

There are a lot of zero value in the zn predictors The distribution of two predictors are strongly skewed.

We transform the predictor chas into factor of two levels 0 and 1 . We get for statistic 433 count for 9 and 13 count for 1, Only 33 neighborhoods over 466 border the Charles Rivers (7.1%)

The factor of target gives 237 for 0 and 229 for 1. That means 49% of neighborhood are at risk with high crime against 51% no at risk.

pairs(crime.train, col=crime.train$target)

cor(crime.train)

Table 3.2.3: Correlation matrix of the train data

	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	lstat	medv	target
zn	1	-0.538	-0.04	-0.517	0.32	-0.573	0.66	-0.315	-0.319	-0.391	-0.433	0.377	-0.432
indus	-0.538	1	0.061	0.76	-0.393	0.64	-0.704	0.601	0.732	0.395	0.607	-0.496	0.605
chas	-0.04	0.061	1	0.097	0.091	0.079	-0.097	-0.016	-0.047	-0.129	-0.051	0.162	0.08
nox	-0.517	0.76	0.097	1	-0.295	0.735	-0.769	0.596	0.654	0.176	0.596	-0.43	0.726
rm	0.32	-0.393	0.091	-0.295	1	-0.233	0.199	-0.208	-0.297	-0.36	-0.632	0.705	-0.153
age	-0.573	0.64	0.079	0.735	-0.233	1	-0.751	0.46	0.512	0.255	0.606	-0.378	0.63
dis	0.66	-0.704	-0.097	-0.769	0.199	-0.751	1	-0.495	-0.534	-0.233	-0.508	0.257	-0.619
rad	-0.315	0.601	-0.016	0.596	-0.208	0.46	-0.495	1	0.906	0.471	0.503	-0.398	0.628
tax	-0.319	0.732	-0.047	0.654	-0.297	0.512	-0.534	0.906	1	0.474	0.564	-0.49	0.611
ptratio	-0.391	0.395	-0.129	0.176	-0.36	0.255	-0.233	0.471	0.474	1	0.377	-0.516	0.251
lstat	-0.433	0.607	-0.051	0.596	-0.632	0.606	-0.508	0.503	0.564	0.377	1	-0.736	0.469
medv	0.377	-0.496	0.162	-0.43	0.705	-0.378	0.257	-0.398	-0.49	-0.516	-0.736	1	-0.271
target	-0.432	0.605	0.08	0.726	-0.153	0.63	-0.619	0.628	0.611	0.251	0.469	-0.271	1

Now we can check the correlation between variables. Stating that the threshold for correlation between two variables is 0.7 or more, we can extract correlated predictors in pairs:

Indus – nox

indus – dis

indus - tax

nox – age

nox – dis
rm – medv
age – dis
tax – rad (high correletion)
lstat – medv

The target variable is linear correlated with some predictors as idus, nox, age, dis, tax.

We can observe those correlation in the matrix plot below.

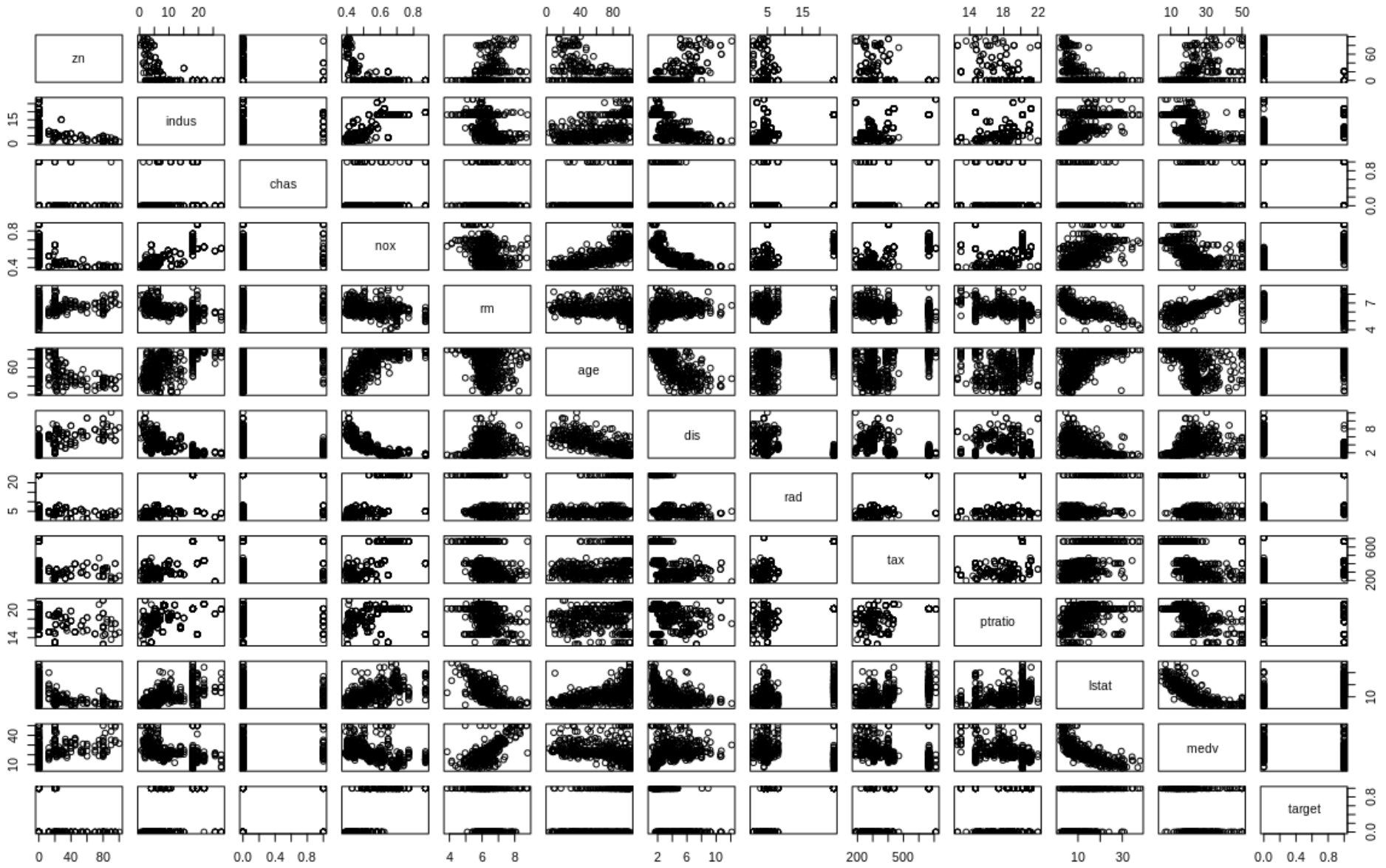
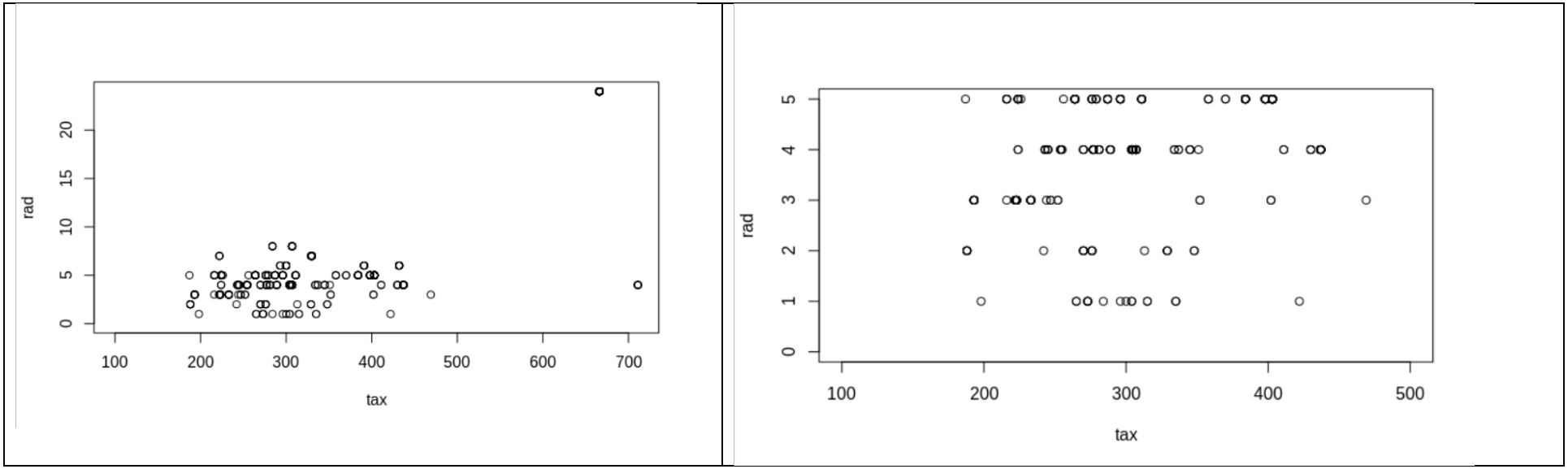


Fig. 1: Matrix plot of the train data



The plot of rad against tax shows that the correlation between tax and rad of 90% is made by the influential points. The two predictors are not really correlated.

3. DATA PREPARATION

Since there are not missing data, we going to split the train data in another train set and a testing set.

Splitting the data into training and testing set

```
X <- cbind(zn, indus, chas, nox, rm, age, dis, rad, tax, ptratio, lstat, medv)
 y <- cbind(target)
```

We put 370 rows into training set and and the rest into the test set.

```
n.train <- sample(seq(dim(crime.train)[1]),370, replace = FALSE)
 X.Train <- X[n.train,]
X.Test <- X[-n.train,]
 y.Train <- y[n.train]
 y.Test <- y[-n.train]
```

```
data.Train <- crime.train[n.train,]
data.Test <- crime.train[-n.train,]
```

4. BUILD MODELS

4.1 Model with logit and probit

4.1.1 Logit model with the entire predictors

```
summary(logit.cr)

##
## Call:
 ## glm(formula = target ~ ., family = binomial(link = "logit"),
## data = data.Train)
 ##
## Deviance Residuals:
## Min 1Q Median 3Q Max
## -1.7741 -0.1637 -0.0032 0.0059 3.4328
##
```

Table 3. 4.1: Coefficients of logit.cr model

Predictors	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	-48.756	8.625	-5.653	0.000
zn	-0.046	0.037	-1.232	0.218
indus	-0.113	0.061	-1.854	0.064
chas	1.265	0.845	1.497	0.134
nox	63.274	11.086	5.707	0.000
rm	-1.266	0.900	-1.407	0.159
age	0.037	0.017	2.147	0.032
dis	0.807	0.260	3.100	0.002
rad	0.762	0.201	3.796	0.000
tax	-0.006	0.003	-1.861	0.063
ptratio	0.602	0.165	3.653	0.000
lstat	0.002	0.066	0.035	0.972
medv	0.227	0.082	2.774	0.006

Null deviance: 512.40 on 369 degrees of freedom
Residual deviance: 154.17 on 357 degrees of freedom
AIC: 180.17
##
Number of Fisher Scoring iterations: 9

The predictors zn. Chas, rm, lsat are not valid parametres. Their p-values are too high.

4.1.2 Probit model with the entire predictors

summary(probit.cr)

##
Call:
glm(formula = target ~ ., family = binomial(link = "probit"),
data = data.Train)
##
Deviance Residuals:
Min 1Q Median 3Q Max
-1.7731 -0.1618 0.0000 0.0000 3.4880
##
Coefficients:
##

Table 4.2. Coefficients of probit.cr model

Predictor	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	-25.528	4.389	-5.817	0.000
zn	-0.016	0.016	-0.968	0.333
indus	-0.058	0.034	-1.723	0.085
chas	0.722	0.453	1.596	0.110
nox	33.507	5.704	5.874	0.000
rm	-0.623	0.477	-1.305	0.192
age	0.016	0.009	1.736	0.083
dis	0.382	0.133	2.872	0.004
rad	0.399	0.105	3.793	0.000
tax	-0.004	0.002	-1.965	0.049

ptratio	0.321	0.088	3.655	0.000
lstat	0.009	0.036	0.248	0.805
medv	0.114	0.042	2.678	0.007

Null deviance: 512.40 on 369 degrees of freedom
Residual deviance: 157.18 on 357 degrees of freedom
AIC: 183.18
##
Number of Fisher Scoring iterations: 10

There area 6 parameters with higher p-values

4.2 Choosing a model with AIC, R.SQUARE, or Mallow's CP metrics

The regsubsets function of the leaps package allow us to choose the better combination of predictors.

4.2.1 Choosing a model with AIC metric

```
best.lin <- regsubsets(target ~., data = data.Train , method = "exhaustive", nvmax = 12)
rs <- summary(best.lin)
rs$which
```

Table 4.2.1 Combination of predictors. True means the predictor is present in the model with n predictors

	(Intercept)	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	lstat	medv
1	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
2	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE
3	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE
4	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE	TRUE	FALSE	TRUE	FALSE	FALSE	FALSE	TRUE
5	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE	TRUE	FALSE	TRUE	FALSE	TRUE	FALSE	TRUE
6	TRUE	FALSE	FALSE	FALSE	TRUE	FALSE	TRUE	FALSE	TRUE	TRUE	TRUE	FALSE	TRUE
7	TRUE	FALSE	TRUE	FALSE	TRUE	FALSE	TRUE	FALSE	TRUE	TRUE	TRUE	FALSE	TRUE
8	TRUE	FALSE	TRUE	TRUE	TRUE	FALSE	TRUE	FALSE	TRUE	TRUE	TRUE	FALSE	TRUE
9	TRUE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	TRUE	TRUE	TRUE	FALSE	TRUE
10	TRUE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	TRUE
11	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	TRUE
12	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE

We compute and plot the AIC

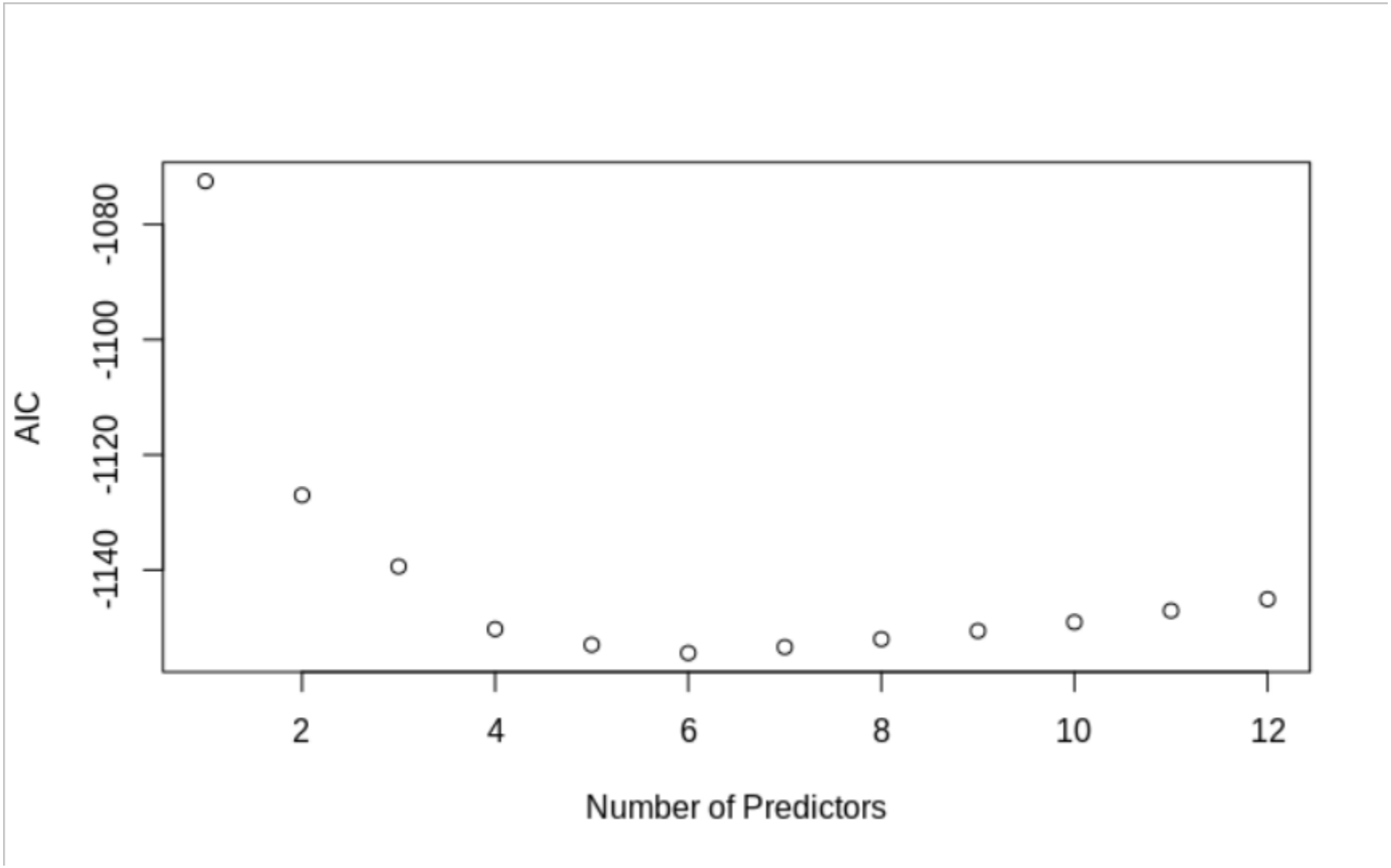


Fig 1 AIC for models with varying numbers of predictors using the crime

```
which.min(AIC)
```

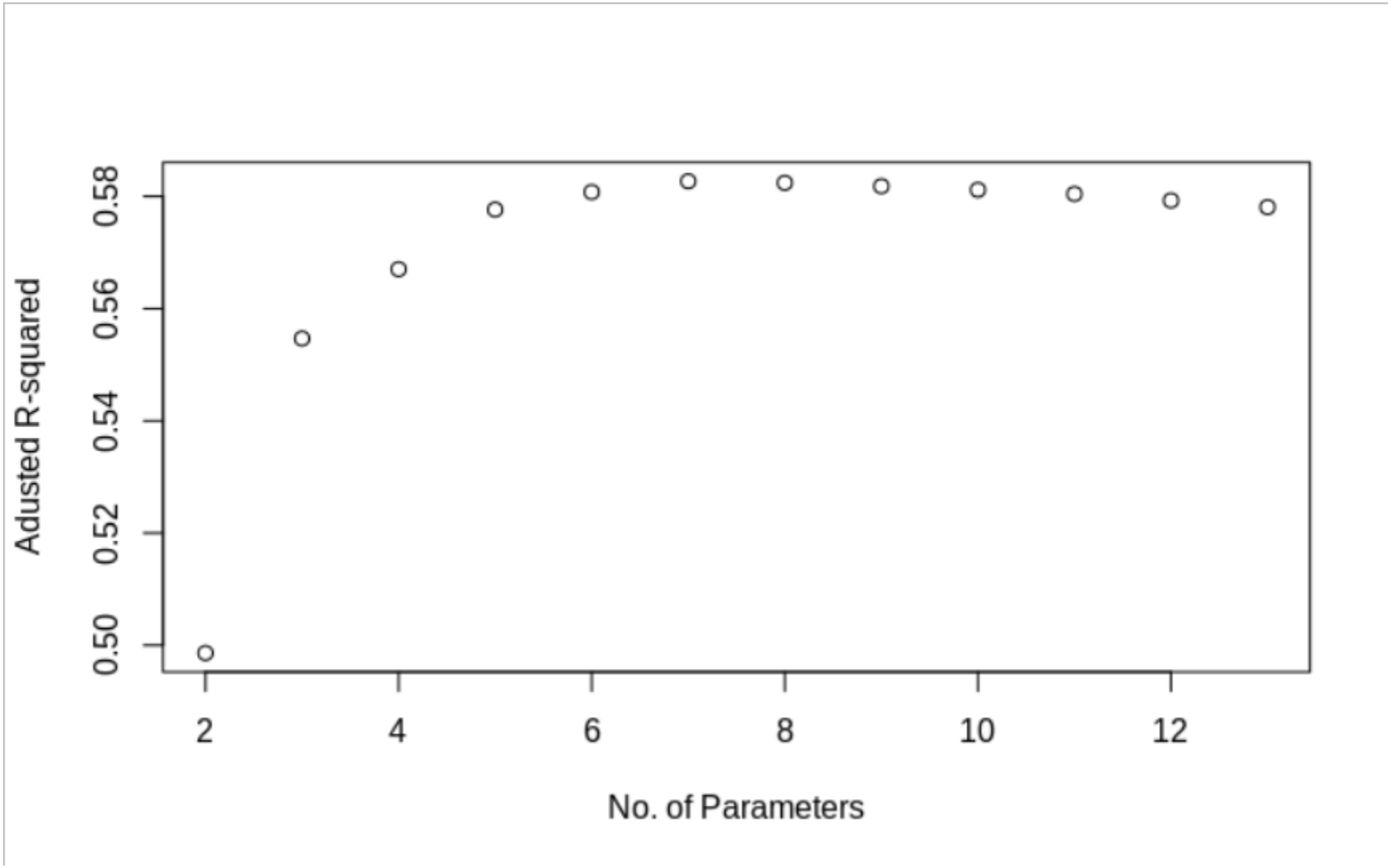
```
## [1] 5
```

With the AIC metric, there are 5 predictors in the best model, those predictors are nox, age, rad, ptratio, and medv

4.2.2 Choosing a model with R.square

Choice of model using adjusted SR^2$

```
plot(2:13, rs$adjr2, xlab = "No. of Parameters", ylab = "Adusted R-squared")
```



```
which.max(rs$adjr2)
```

```
## [1] 7
```

Wiith the ADJUSTED R.SQUARE metric, there are 6 predictors in the best model, those predictors are nox, age, rad, ptratio, tax, and medv.

Compare, to the AIC method, The ADJ R.SQUARE adds one predictor that is tax.

4.2.3 Choosing a model with Mallow's Cp statistic

```
plot(2:13, rs$cp, xlab = "No. of Parameters", ylab = "Cp Statistic")
 abline(0,1)
```

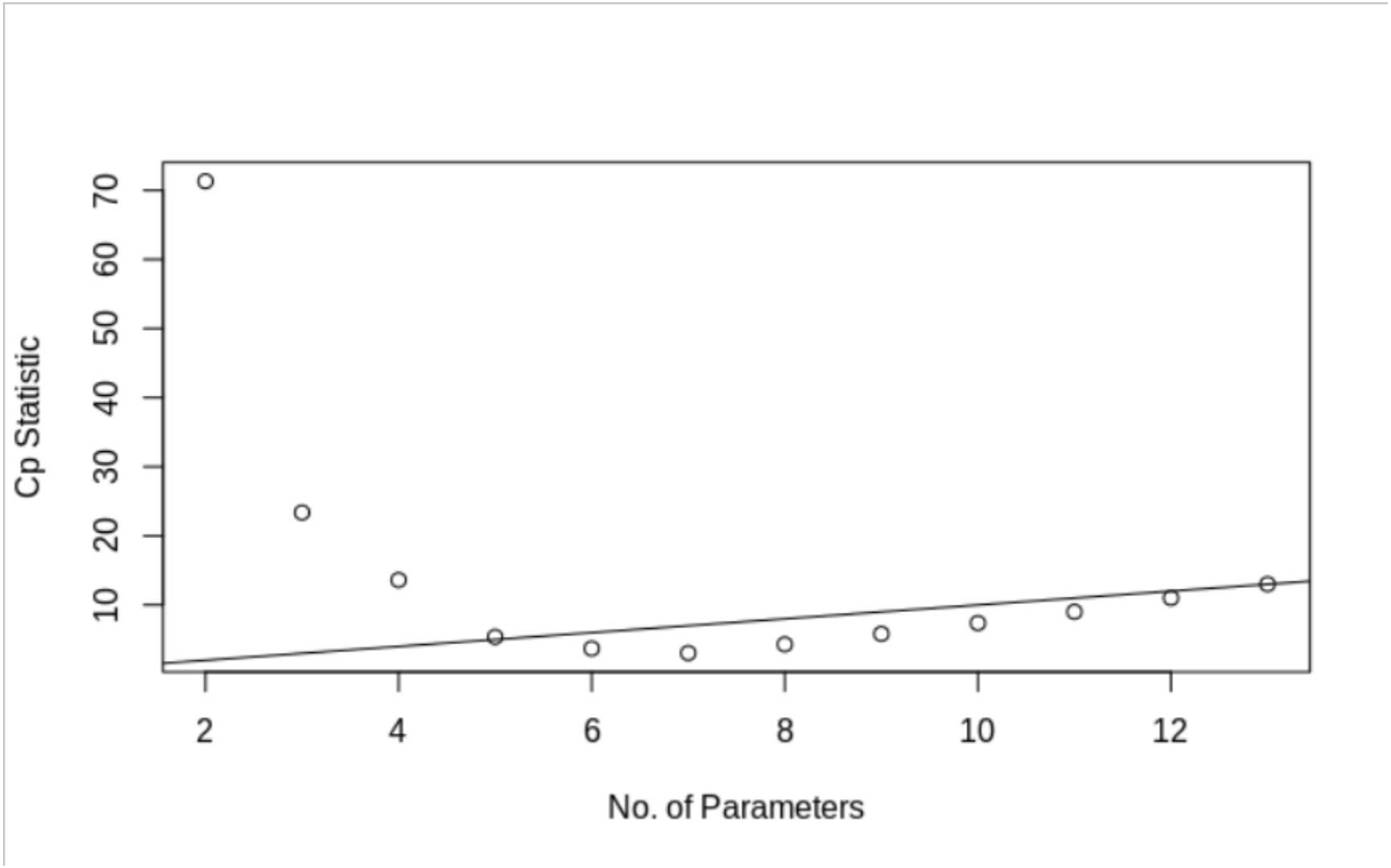


Fig 4.2.3 Mallow's CP Statistic

Observing the graph 4.2.3, the Mallos's cp curvr met the abline at 5 , means there are 4 predictors in it best model. But we will chose 5 predictors to avoid underfitting.

```
coef(best.lin, 5)
```

```
## (Intercept) nox age rad ptratio medv
## -1.472893785 2.026920513 0.003520715 0.016619502 0.013162300 0.008851064
```

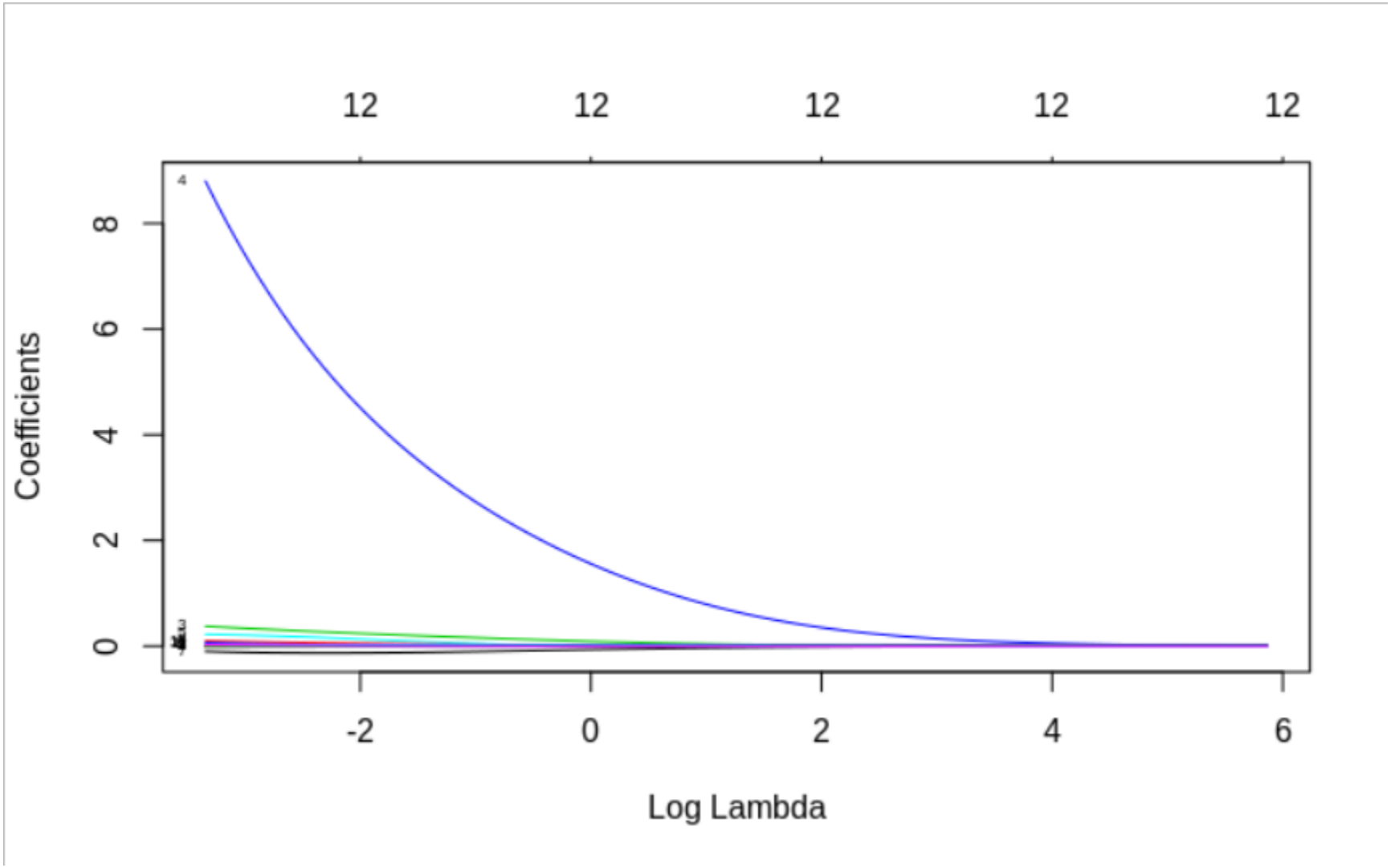
```
leap.mod.logit <- glm(target ~ zn + nox + rm + rad + ptratio + lstat + medv,
 data = data.Train,
family = binomial(link = "logit" ))


summary(leap.mod.logit)
 leap.mod.probit <- glm(target ~ zn + nox + rm + rad + ptratio + lstat + medv,
 data = data.Train,
family = binomial(link = "probit" )


summary(leap.mod.probit)
```

Using step function

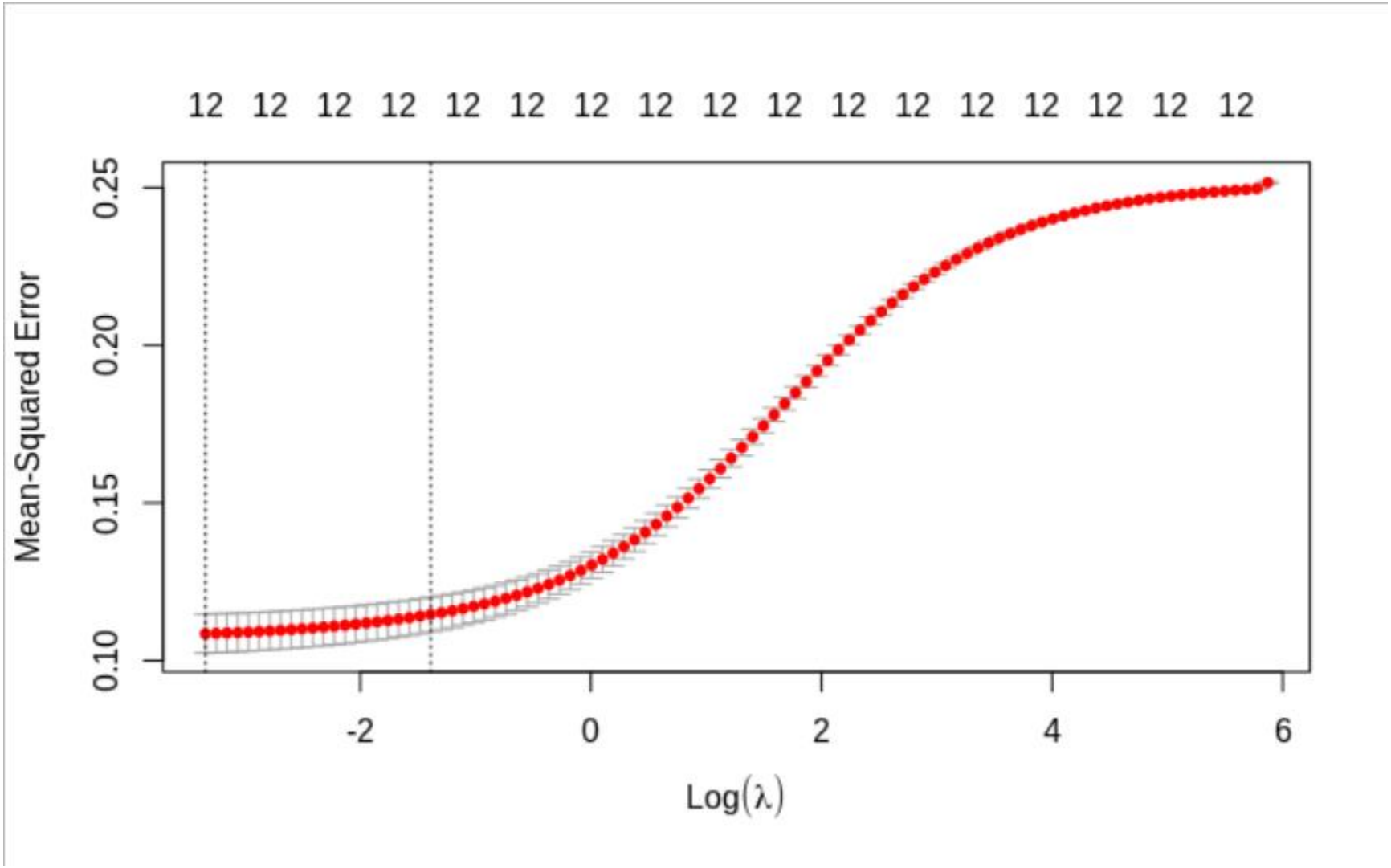
```
step.glmod.logit <- glm(target ~ ., data = data.Train, family = binomial(link = "logit"))
 step(step.glmod.logit, trace = FALSE)

step(step.glmod.probit, trace = FALSE)
```

glmnet

```
fit.ridge <- glmnet(x=X.Train,y=y.Train, alpha=0, family="binomial")
 plot(fit.ridge, xvar= "lambda", label=TRUE)
```

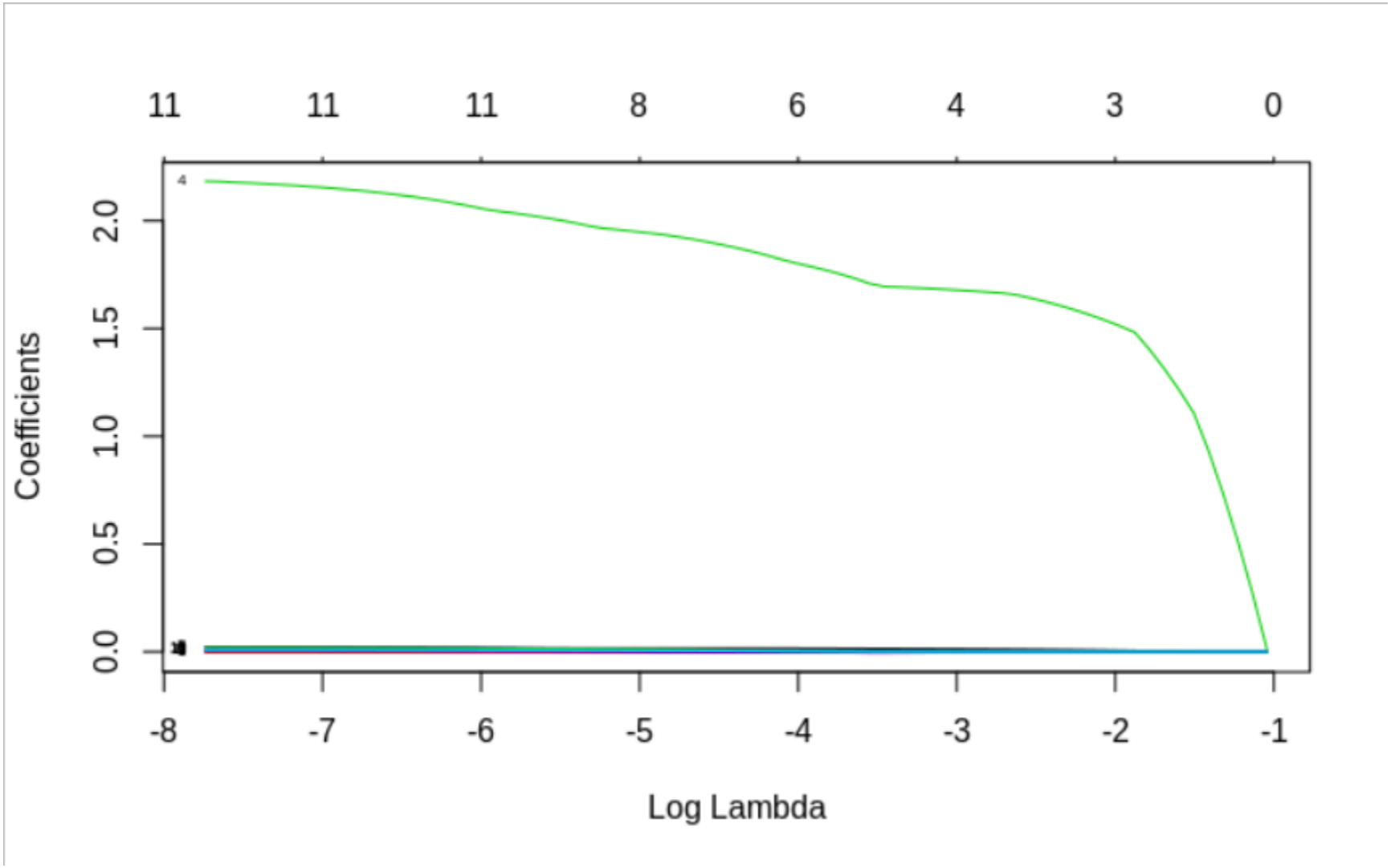



```
cv.ridge <- cv.glmnet(x=X.Train, y=y.Train, alpha=0)
 plot(cv.ridge)
```

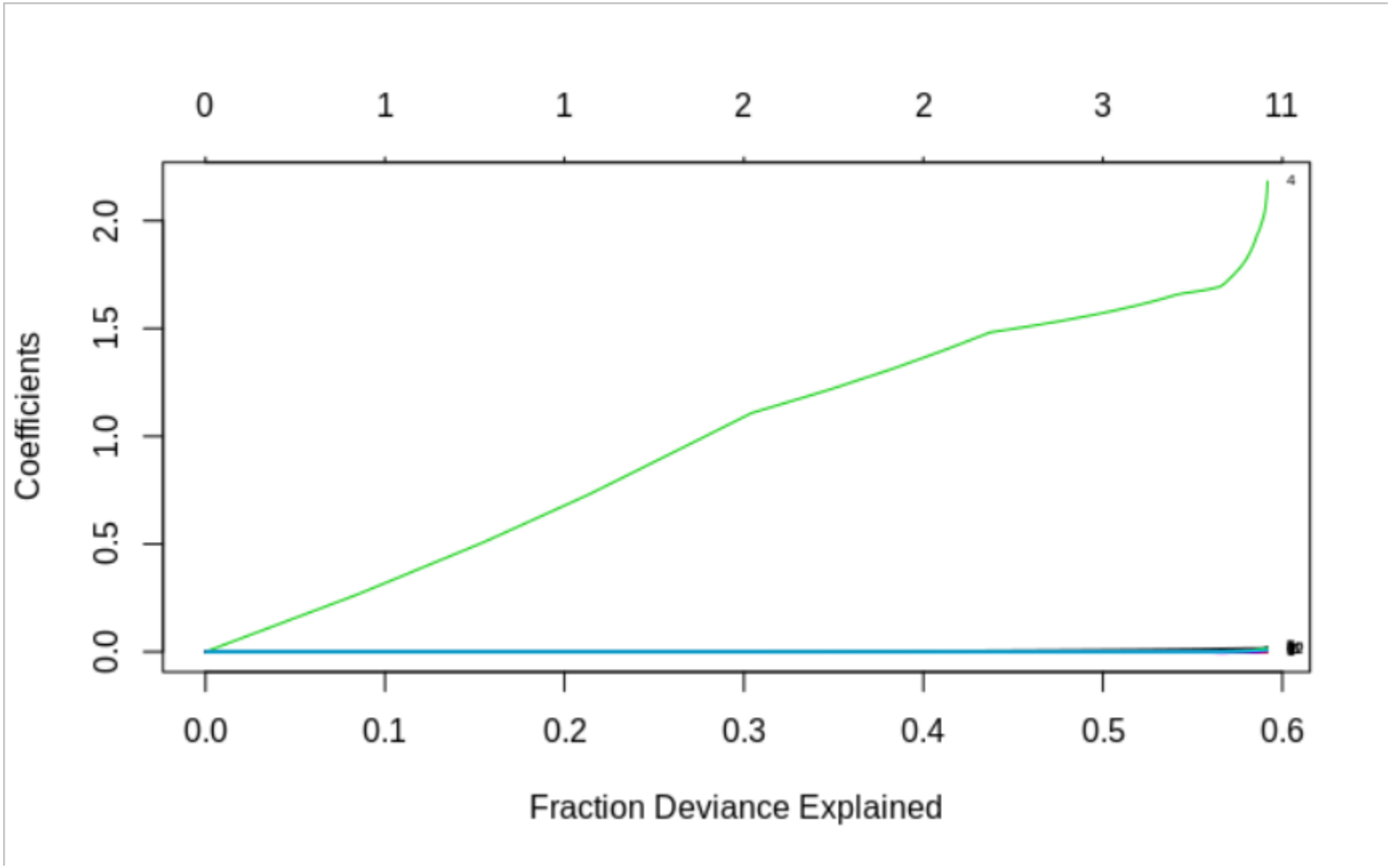


The plot shows that the log of the optimal value of lambda (i.e. the one that minimises the root mean square error) is approximately –1.5. The exact value can be viewed by examining the variable lambda_min in the code below. In general though, the objective of regularisation is to balance accuracy and simplicity. In the present context, this means a model with the smallest number of coefficients that also gives a good accuracy. To this end, the cv.glmnet function finds the value of lambda that gives the simplest model but also lies within one standard error of the optimal value of lambda.

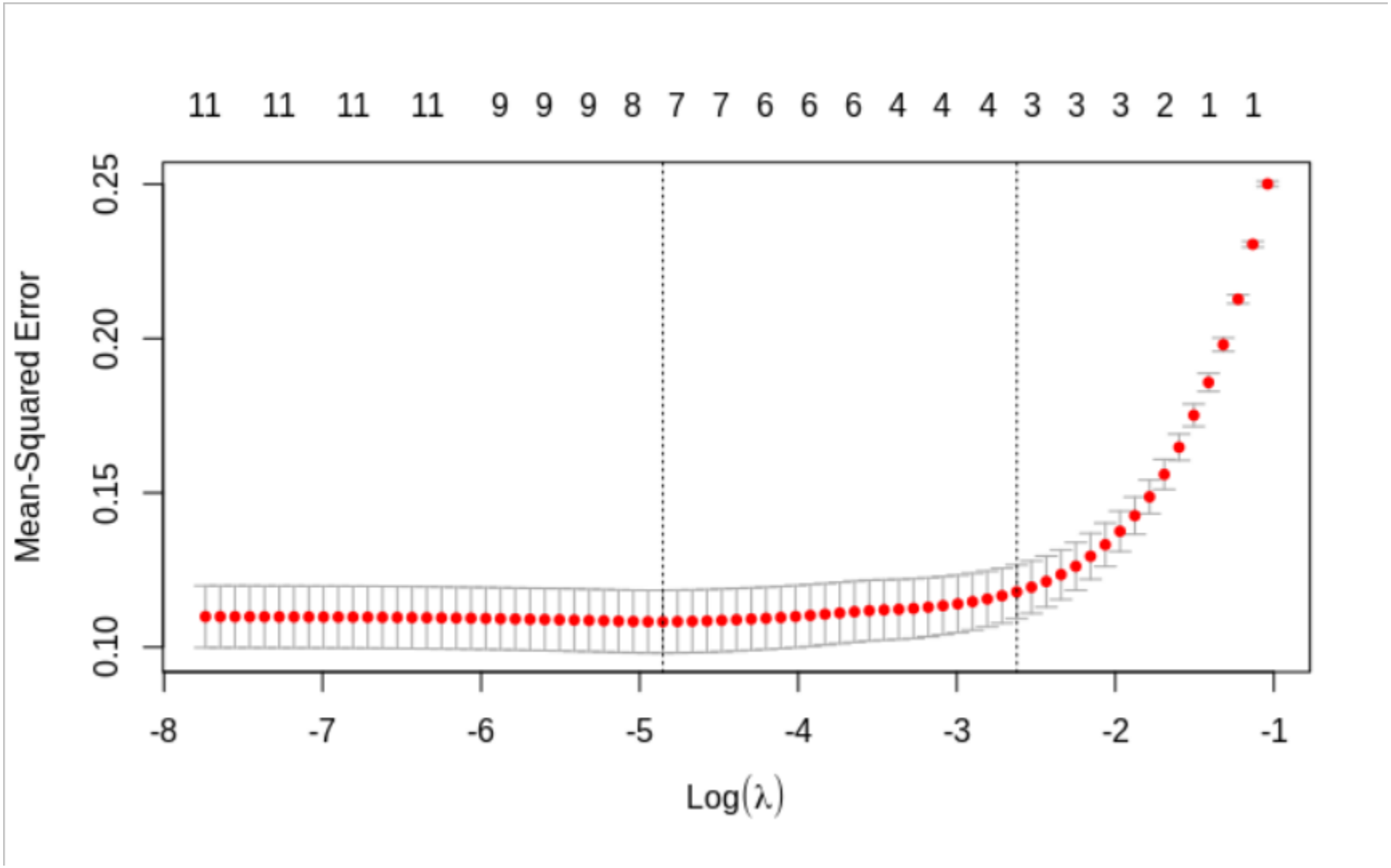
```
fit.lasso <- glmnet(X.Train,y.Train)
 plot(fit.lasso, xvar = "lambda", label = TRUE)
```




```
plot(fit.lasso, xvar = "dev", label = TRUE)
```




```
cv.lasso <- cv.glmnet(X.Train,y.Train)
 plot(cv.lasso)
```



The plot shows that the log of the optimal value of lambda (i.e. the one that minimises the root mean square error) is approximately -5. The exact value can be viewed by examining the variable lambda_min in the code below. In general though, the objective of regularisation is to balance accuracy and simplicity. In the present context, this means a model with the smallest number of coefficients that also gives a good accuracy. To this end, the cv.glmnet function finds the value of lambda that gives the simplest model but also lies within one standard error of the optimal value of lambda. This value of lambda (lambda.1se) is what we'll use in the rest of the computation

5. MODEL SELECTION

Table: calculated metrics with the testing set

	Accuracy	Precision	Sensitivity	Specificity	F1score
logit.cr	0.875	0.892	0.805	0.927	0.846
probit.cr	0.885	0.895	0.829	0.927	0.861
leap.mod.logit	0.802	0.789	0.732	0.855	0.759
leap.mod.probit	0.802	0.789	0.732	0.855	0.759
step.glmod.logit	0.875	0.892	0.805	0.927	0.846
step.glmod.probit	0.885	0.895	0.829	0.927	0.861
fit.ridge	0.802	0.806	0.707	0.873	0.753
fit.lasso	0.812	0.871	0.659	0.927	0.75

summary(t(metrcis.mod))

	Accuracy	Precision	Sensitivity	Specificity	F1score
Mininmum	0.8021	0.7895	0.6585	0.8545	0.7500
1st Quantile	0.8021	0.8015	0.7256	0.8682	0.7579
Median	0.8438	0.8814	0.7683	0.9273	0.8028
Mean	0.8424	0.8536	0.7622	0.9023	0.8045
3rd Quantile	0.8776	0.8926	0.8110	0.9273	0.8498
Maximun	0.8854	0.8947	0.8293	0.9273	0.8608

Table 5.3 Confusion matrix of the model with all predictors

	Predicted Values		
		0	1
Observed Values	0	228	56
	1	9	173

#Table: Extracted Metrics using the test set

	Accuracy	Precision	Sensitivity	Specificity	F1
logit.cr	0.9063	0.8679	0.9583	0.8542	0.9109
probit.cr	0.9063	0.8679	0.9583	0.8542	0.9109
leap.mod.logit	0.8854	0.8491	0.9375	0.8333	0.8911
leap.mod.probit	0.8750	0.8462	0.9167	0.8333	0.8800
step.glmod.logit	0.9063	0.8679	0.9583	0.8542	0.9109
step.glmod.probit	0.9063	0.8679	0.9583	0.8542	0.9109
fit.ridge	0.8646	0.8723	0.8542	0.8750	0.8632
fit.lasso	0.8750	0.8913	0.8542	0.8958	0.8723

Table: Extracted Metrics using the whole train set

	Accuracy	Precision	Sensitivity	Specificity	F1
logit.cr	0.918	0.917	0.917	0.920	0.917
probit.cr	0.916	0.917	0.913	0.920	0.915
leap.mod.logit	0.871	0.876	0.860	0.882	0.868
leap.mod.probit	0.871	0.879	0.856	0.886	0.867
step.glmod.logit	0.918	0.917	0.917	0.920	0.917
step.glmod.probit	0.916	0.917	0.913	0.920	0.915
fit.ridge	0.824	0.862	0.764	0.882	0.810
fit.lasso	0.856	0.940	0.755	0.954	0.838

Using pROC packege.

We can plot the ROC curve and extract the AUC value.

```
ggroc(roc1)
```

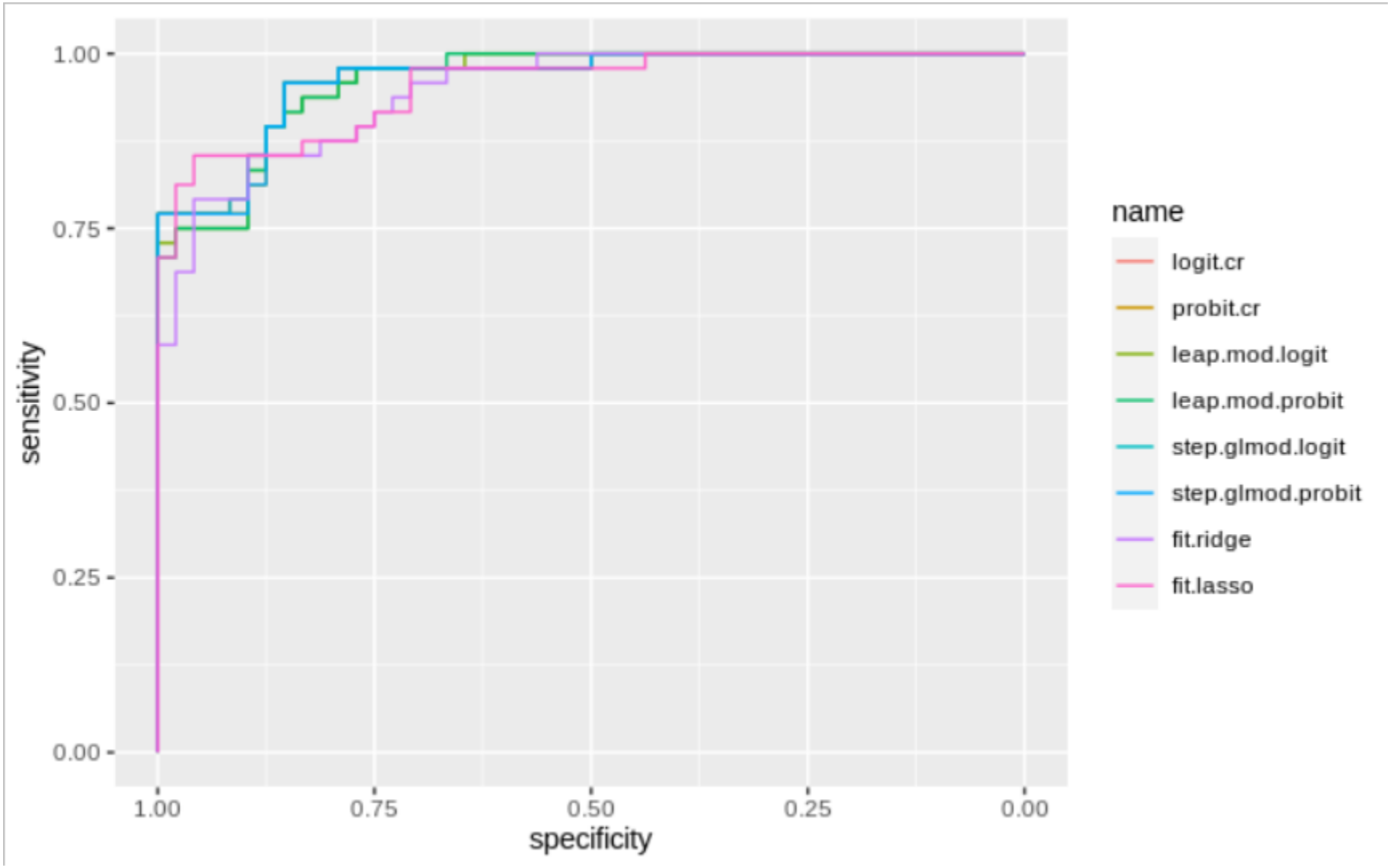


Fig: ROC using the test set

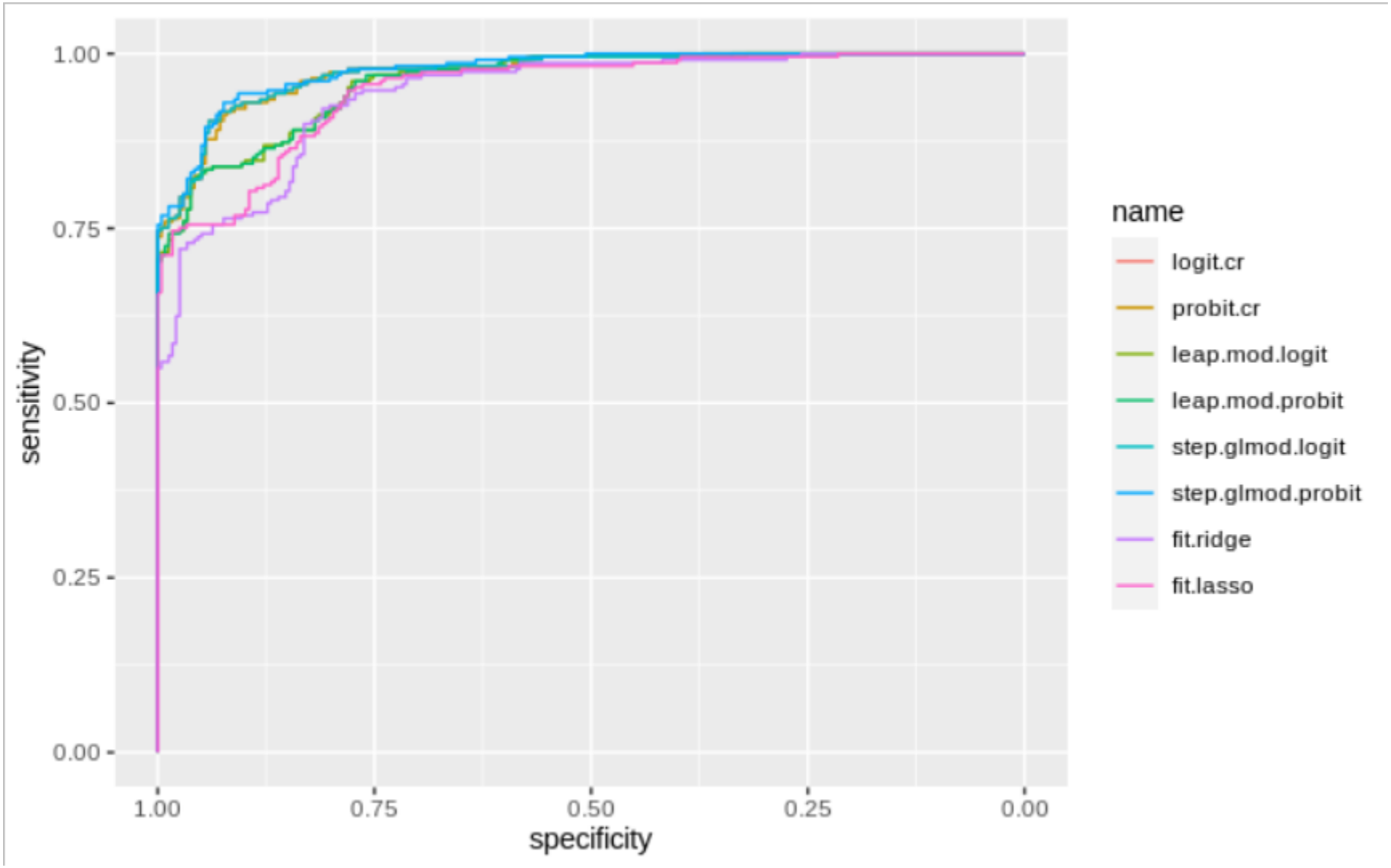


Fig: ROC curve using the entire train set

The step glm model is the best

AUC

Table: AUC using the entire train set

Model	AUC

logit.cr	0.973
probit.cr	0.972
leap.mod.logit	0.962
leap.mod.probit	0.962
step.glmod.logit	0.973
step.glmod.probit	0.972
fit.ridge	0.939
fit.lasso	0.949

The logit regression with the entire predictors and the step model with 8 predictors have the maximal auc. The model less predictors will be better.

Selected Model

Two models have the same maximal accuacy and auc, we choose the model with less predictors We refit the model with the entire train data

```
final.model <- glm(formula = target ~ zn + nox + age + dis + rad + tax + ptratio +
medv, family = binomial(link = "logit"), data = crime.train)
```

We find the predicted values and print the confusion matrix

Table 6.1 Confusion Matrix and Statistics

	Predicted Values		
		0	1
Observed Values	0	218	22
	1	19	207

95% CI : (0.8825, 0.9361)
No Information Rate : 0.5086
P-Value [Acc > NIR] : <2e-16
##
Kappa : 0.8239
##
Mcnemar's Test P-Value : 0.7548
##

Metric	Value
Accuracy	0.912

Sensitivity	0.904
Specificity	0.920
Pos Pred Value	0.916
Neg Pred Value	0.908
Precision	0.916
Recall	0.904
F1	0.910
Prevalence	0.491
Detection Rate	0.444
Detection Prevalence	0.485
Balanced Accuracy	0.912

Appendicle 1: Evaluation data set

	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	lstat	medv	probability	target
1	0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	4.03	34.7	0.052	0
2	0	8.14	0	0.538	6.096	84.5	4.4619	4	307	21	10.26	18.2	0.656	1
3	0	8.14	0	0.538	6.495	94.4	4.4547	4	307	21	12.8	18.4	0.729	1
4	0	8.14	0	0.538	5.95	82	3.99	4	307	21	27.71	13.2	0.426	0
5	0	5.96	0	0.499	5.85	41.5	3.9342	5	279	19.2	8.77	21	0.108	0
6	25	5.13	0	0.453	5.741	66.2	7.2254	8	284	19.7	13.15	18.7	0.313	0
7	25	5.13	0	0.453	5.966	93.4	6.8185	8	284	19.7	14.44	16	0.388	0
8	0	4.49	0	0.449	6.63	56.1	4.4377	3	247	18.5	6.53	26.6	0.014	0
9	0	4.49	0	0.449	6.121	56.8	3.7476	3	247	18.5	8.44	22.2	0.006	0
10	0	2.89	0	0.445	6.163	69.6	3.4952	2	276	18	11.34	21.4	0.002	0
11	0	25.65	0	0.581	5.856	97	1.9444	2	188	19.1	25.41	17.3	0.502	1
12	0	25.65	0	0.581	5.613	95.6	1.7572	2	188	19.1	27.26	15.7	0.417	0
13	0	21.89	0	0.624	5.637	94.7	1.9799	4	437	21.2	18.34	14.3	0.841	1
14	0	19.58	0	0.605	6.101	93	2.2834	5	403	14.7	9.81	25	0.743	1
15	0	19.58	0	0.605	5.88	97.3	2.3887	5	403	14.7	12.03	19.1	0.65	1
16	0	10.59	1	0.489	5.96	92.1	3.8771	4	277	18.6	17.27	21.7	0.149	0
17	0	6.2	0	0.504	6.552	21.4	3.3751	8	307	17.4	3.76	31.5	0.403	0
18	0	6.2	0	0.507	8.247	70.4	3.6519	8	307	17.4	3.95	48.3	0.967	1
19	22	5.86	0	0.431	6.957	6.8	8.9067	7	330	19.1	3.53	29.6	0.079	0
20	90	2.97	0	0.4	7.088	20.8	7.3073	1	285	15.3	7.85	32.2	0	0

21	80	1.76	0	0.385	6.23	31.5	9.0892	1	241	18.2	12.93	20.1	0	0
22	33	2.18	0	0.472	6.616	58.1	3.37	7	222	18.4	8.93	28.4	0.052	0
23	0	9.9	0	0.544	6.122	52.8	2.6403	4	304	18.4	5.98	22.1	0.152	0
24	0	7.38	0	0.493	6.415	40.1	4.7211	5	287	19.6	6.12	25	0.199	0
25	0	7.38	0	0.493	6.312	28.9	5.4159	5	287	19.6	6.15	23	0.178	0
26	0	5.19	0	0.515	5.895	59.6	5.615	5	224	20.2	10.56	18.5	0.677	1
27	80	2.01	0	0.435	6.635	29.7	8.344	4	280	17	5.99	24.5	0	0
28	0	18.1	0	0.718	3.561	87.9	1.6132	24	666	20.2	7.12	27.5	1	1
29	0	18.1	1	0.631	7.016	97.5	1.2024	24	666	20.2	2.96	50	1	1
30	0	18.1	0	0.584	6.348	86.1	2.0527	24	666	20.2	17.64	14.5	1	1
31	0	18.1	0	0.74	5.935	87.9	1.8206	24	666	20.2	34.02	8.4	1	1
32	0	18.1	0	0.74	5.627	93.9	1.8172	24	666	20.2	22.88	12.8	1	1
33	0	18.1	0	0.74	5.818	92.4	1.8662	24	666	20.2	22.11	10.5	1	1
34	0	18.1	0	0.74	6.219	100	2.0048	24	666	20.2	16.59	18.4	1	1
35	0	18.1	0	0.74	5.854	96.6	1.8956	24	666	20.2	23.79	10.8	1	1
36	0	18.1	0	0.713	6.525	86.5	2.4358	24	666	20.2	18.13	14.1	1	1
37	0	18.1	0	0.713	6.376	88.4	2.5671	24	666	20.2	14.65	17.7	1	1
38	0	18.1	0	0.655	6.209	65.4	2.9634	24	666	20.2	13.22	21.4	1	1
39	0	9.69	0	0.585	5.794	70.6	2.8927	6	391	19.2	14.1	18.3	0.802	1
40	0	11.93	0	0.573	6.976	91	2.1675	1	273	21	5.64	23.9	0.395	0

Appendicle 2 R Code