Crimes in North Carolina data analysis

Aleksandra Mazur, Mateusz Buta

Setup

```
library(class)
library(boot)
library(leaps)
library(splines)
library(tree)
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

Dataset

We based on dataset of crimes in North Carolina from 1981 to 1987. A dataframe contains:

- **county** - county identifier
- **year** - year from 1981 to 1987
- **crmrte** - crimes committed per person
- **prbarr** - 'probability' of arrest
- **prbconv** - 'probability' of conviction
- **prbpris** - 'probability' of prison sentence
- **avgsen** - average sentence, days
- **polpc** - police per capita
- **density** - hundreds of people per square mile
- **taxpc** - tax revenue per capita
- **region** - one of 'other', 'west' or 'central'
- **smsa** - 'yes' or 'no' if in SMSA
- **pctmin** - percentage minority in 1980
- **wcon** - weekly wage in construction
- **wtuc** - weekly wage in trns, util, commun
- **wtrd** - weekly wage in whole sales and retail trade
- **wfir** - weekly wage in finance, insurance and real estate
- **wser** - weekly wage in service industry
- **wmfg** - weekly wage in manufacturing
- **wfed** - weekly wage of federal employees
- **wsta** - weekly wage of state employees
- **wloc** - weekly wage of local governments employees
- **mix** - offence mix: face-to-face/other
- **pctymle** - percentage of young males

Dataset source

```r
crime <- read.csv("/tmp/Crime.csv")
crime <- subset(crime, select=-c(1))
summary(crime)
```

```
##      county           year         crmrte              prbarr
##  Min.   :  1.0   Min.   :81   Min.   :0.001812   Min.   :0.05882
##  1st Qu.: 51.0   1st Qu.:82   1st Qu.:0.018352   1st Qu.:0.21790
##  Median :103.0   Median :84   Median :0.028441   Median :0.27824
##  Mean   :100.6   Mean   :84   Mean   :0.031588   Mean   :0.30737
##  3rd Qu.:151.0   3rd Qu.:86   3rd Qu.:0.038406   3rd Qu.:0.35252
##  Max.   :197.0   Max.   :87   Max.   :0.163835   Max.   :2.75000
##     prbconv            prbpris           avgsen            polpc
##  Min.   : 0.06838   Min.   :0.1489   Min.   : 4.220   Min.   :0.0004585
##  1st Qu.: 0.34769   1st Qu.:0.3744   1st Qu.: 7.160   1st Qu.:0.0011913
##  Median : 0.47437   Median :0.4286   Median : 8.495   Median :0.0014506
##  Mean   : 0.68862   Mean   :0.4255   Mean   : 8.955   Mean   :0.0019168
##  3rd Qu.: 0.63560   3rd Qu.:0.4832   3rd Qu.:10.197   3rd Qu.:0.0018033
##  Max.   :37.00000   Max.   :0.6786   Max.   :25.830   Max.   :0.0355781
##     density            taxpc            region     smsa          pctmin
##  Min.   :0.1977   Min.   : 14.30   central:238   no :574   Min.   : 1.284
##  1st Qu.:0.5329   1st Qu.: 23.43   other  :245   yes: 56   1st Qu.:10.005
##  Median :0.9526   Median : 27.79   west   :147             Median :24.852
##  Mean   :1.3861   Mean   : 30.24                           Mean   :25.713
##  3rd Qu.:1.5078   3rd Qu.: 33.27                           3rd Qu.:38.223
##  Max.   :8.8277   Max.   :119.76                           Max.   :64.348
##       wcon              wtuc              wtrd              wfir
##  Min.   :  65.62   Min.   :  28.86   Min.   :  16.87   Min.   :  3.516
##  1st Qu.: 201.66   1st Qu.: 317.60   1st Qu.: 168.05   1st Qu.:235.705
##  Median : 236.46   Median : 358.20   Median : 185.48   Median :264.423
##  Mean   : 245.67   Mean   : 406.10   Mean   : 192.82   Mean   :272.059
##  3rd Qu.: 269.69   3rd Qu.: 411.02   3rd Qu.: 204.82   3rd Qu.:302.440
##  Max.   :2324.60   Max.   :3041.96   Max.   :2242.75   Max.   :509.466
##       wser              wmfg             wfed             wsta
##  Min.   :   1.844   Min.   :101.8   Min.   :255.4   Min.   :173.0
##  1st Qu.: 191.319   1st Qu.:234.0   1st Qu.:361.5   1st Qu.:258.2
##  Median : 216.475   Median :271.6   Median :404.0   Median :289.4
##  Mean   : 224.671   Mean   :285.2   Mean   :403.9   Mean   :296.9
##  3rd Qu.: 247.155   3rd Qu.:320.0   3rd Qu.:444.6   3rd Qu.:331.5
##  Max.   :2177.068   Max.   :646.9   Max.   :598.0   Max.   :548.0
##       wloc            mix              pctymle
##  Min.   :163.6   Min.   :0.002457   Min.   :0.06216
##  1st Qu.:226.8   1st Qu.:0.075324   1st Qu.:0.07859
##  Median :253.1   Median :0.102089   Median :0.08316
##  Mean   :258.0   Mean   :0.139396   Mean   :0.08897
##  3rd Qu.:289.3   3rd Qu.:0.149009   3rd Qu.:0.08919
##  Max.   :388.1   Max.   :4.000000   Max.   :0.27436
```

```r
attach(crime)
```

Firstly, we checked the corelation between predictors.

```r
cor(crime[-c(11, 12, 25)])
```

```
##               county         year       crmrte      prbarr      prbconv
## county   1.000000000  0.000000000  0.041678634 -0.01856917  0.118875273
```

```
## year     0.000000000  1.000000000  0.002392457 -0.02806708  0.006527245
## crmrte   0.041678634  0.002392457  1.000000000 -0.35855277 -0.113032692
## prbarr  -0.018569168 -0.028067078 -0.358552773  1.00000000  0.035568903
## prbconv  0.118875273  0.006527245 -0.113032692  0.03556890  1.000000000
## prbpris -0.024861396 -0.096085345  0.135537013 -0.07489329 -0.037340175
## avgsen   0.050533674 -0.075257177  0.032416405  0.03403139  0.015304708
## polpc    0.123775461  0.025364518  0.184826442  0.29058128  0.449635003
## density -0.025805624  0.023470549  0.694071923 -0.27122844 -0.115555290
## taxpc   -0.061762755  0.426398384  0.230685484 -0.04121696  0.006732670
## pctmin   0.062318408  0.000000000  0.169020952  0.10005025  0.105076935
## wcon    -0.018974450  0.237150557  0.070500478 -0.08811011 -0.045081738
## wtuc    -0.150137203  0.031648351 -0.002451330 -0.02222389 -0.023258280
## wtrd    -0.028380554  0.164034550  0.232683908 -0.07453934 -0.049324273
## wfir    -0.012928075  0.562787915  0.256156826 -0.10023184 -0.064390923
## wser     0.008726151  0.248629228  0.092340455 -0.09800658 -0.017109807
## wmfg    -0.014159046  0.399441718  0.279718466 -0.07734287 -0.070292025
## wfed    -0.022037661  0.443546890  0.437156980 -0.18547668 -0.072331900
## wsta     0.082825727  0.688788287  0.163966917 -0.11361736 -0.006009803
## wloc     0.020202407  0.793877711  0.234601054 -0.13414775 -0.030400444
## mix      0.056880585 -0.018844849 -0.119574049  0.34753028  0.506170865
## pctymle  0.098272578 -0.127049948  0.224794702 -0.15627328 -0.069297793
##              prbpris       avgsen        polpc     density        taxpc
## county  -0.024861396  0.050533674  0.123775461 -0.02580562 -0.061762755
## year    -0.096085345 -0.075257177  0.025364518  0.02347055  0.426398384
## crmrte   0.135537013  0.032416405  0.184826442  0.69407192  0.230685484
## prbarr  -0.074893294  0.034031393  0.290581283 -0.27122844 -0.041216964
## prbconv -0.037340175  0.015304708  0.449635003 -0.11555529  0.006732670
## prbpris  1.000000000 -0.004299394 -0.057452385  0.16466324 -0.112063099
## avgsen  -0.004299394  1.000000000  0.017129699  0.07807510  0.028189393
## polpc   -0.057452385  0.017129699  1.000000000 -0.03969574  0.108286636
## density  0.164663238  0.078075105 -0.039695742  1.00000000  0.199763395
## taxpc   -0.112063099  0.028189393  0.108286636  0.19976339  1.000000000
## pctmin   0.130723415 -0.032974956  0.031681643 -0.07479444  0.062186027
## wcon    -0.029217895 -0.008107451 -0.071319279  0.13585765  0.120255075
## wtuc    -0.019299775  0.034867217 -0.002208328  0.03658823  0.027688866
## wtrd     0.056636118  0.056503998 -0.002296542  0.29912642  0.094040034
## wfir     0.007831683  0.054784423 -0.037441204  0.38568746  0.266496721
## wser    -0.005040781 -0.071868201 -0.040558787  0.15233695  0.135299691
## wmfg     0.001517853  0.078510947  0.030373270  0.38454381  0.384469148
## wfed     0.134575101  0.004822377 -0.030482922  0.52916123  0.203636594
## wsta    -0.071582345  0.003612127  0.044056577  0.19317887  0.318116076
## wloc    -0.002176090 -0.005479715 -0.016712268  0.30059073  0.384687182
## mix      0.029360316 -0.047411804  0.283141211 -0.09337040  0.004043404
## pctymle -0.133243049  0.058095904 -0.035652621  0.11165552 -0.127112571
##               pctmin         wcon         wtuc         wtrd         wfir
## county   0.062318408 -0.018974450 -0.150137203 -0.028380554 -0.012928075
## year     0.000000000  0.237150557  0.031648351  0.164034550  0.562787915
## crmrte   0.169020952  0.070500478 -0.002451330  0.232683908  0.256156826
## prbarr   0.100050252 -0.088110113 -0.022223893 -0.074539337 -0.100231841
## prbconv  0.105076935 -0.045081738 -0.023258280 -0.049324273 -0.064390923
## prbpris  0.130723415 -0.029217895 -0.019299775  0.056636118  0.007831683
## avgsen  -0.032974956 -0.008107451  0.034867217  0.056503998  0.054784423
## polpc    0.031681643 -0.071319279 -0.002208328 -0.002296542 -0.037441204
## density -0.074794437  0.135857648  0.036588230  0.299126418  0.385687463
```

```
## taxpc    0.062186027  0.120255075  0.027688866  0.094040034  0.266496721
## pctmin   1.000000000 -0.098112962 -0.083962238  0.024095387 -0.006989696
## wcon    -0.098112962  1.000000000  0.009307305  0.101668074  0.245918202
## wtuc    -0.083962238  0.009307305  1.000000000  0.024852933  0.073026894
## wtrd     0.024095387  0.101668074  0.024852933  1.000000000  0.293645601
## wfir    -0.006989696  0.245918202  0.073026894  0.293645601  1.000000000
## wser     0.054403499  0.104093841  0.001567868  0.113298604  0.247331214
## wmfg    -0.091886639  0.195844630  0.066682784  0.280217346  0.582281295
## wfed     0.039949329  0.243416628  0.031339347  0.320832663  0.675544370
## wsta     0.034246855  0.129249414 -0.038416057  0.151724917  0.506902999
## wloc    -0.010602000  0.286261297  0.031161061  0.268343011  0.698374658
## mix      0.236272398 -0.070689347 -0.052751214 -0.048935885 -0.095038607
## pctymle  0.023565002 -0.039238510 -0.068495451 -0.029180796 -0.063311005
##                 wser         wmfg         wfed         wsta         wloc
## county   0.008726151 -0.014159046 -0.022037661  0.082825727  0.020202407
## year     0.248629228  0.399441718  0.443546890  0.688788287  0.793877711
## crmrte   0.092340455  0.279718466  0.437156980  0.163966917  0.234601054
## prbarr  -0.098006580 -0.077342870 -0.185476681 -0.113617356 -0.134147752
## prbconv -0.017109807 -0.070292025 -0.072331900 -0.006009803 -0.030400444
## prbpris -0.005040781  0.001517853  0.134575101 -0.071582345 -0.002176090
## avgsen  -0.071868201  0.078510947  0.004822377  0.003612127 -0.005479715
## polpc   -0.040558787  0.030373270 -0.030482922  0.044056577 -0.016712268
## density  0.152336949  0.384543809  0.529161234  0.193178873  0.300590733
## taxpc    0.135299691  0.384469148  0.203636594  0.318116076  0.384687182
## pctmin   0.054403499 -0.091886639  0.039949329  0.034246855 -0.010602000
## wcon     0.104093841  0.195844630  0.243416628  0.129249414  0.286261297
## wtuc     0.001567868  0.066682784  0.031339347 -0.038416057  0.031161061
## wtrd     0.113298604  0.280217346  0.320832663  0.151724917  0.268343011
## wfir     0.247331214  0.582281295  0.675544370  0.506902999  0.698374658
## wser     1.000000000  0.203243823  0.249167771  0.210822368  0.285944761
## wmfg     0.203243823  1.000000000  0.576507076  0.333929599  0.518249775
## wfed     0.249167771  0.576507076  1.000000000  0.439420768  0.645312669
## wsta     0.210822368  0.333929599  0.439420768  1.000000000  0.640820870
## wloc     0.285944761  0.518249775  0.645312669  0.640820870  1.000000000
## mix     -0.074800654 -0.129246112 -0.129652147 -0.046547858 -0.077462525
## pctymle -0.012805865 -0.032753463 -0.110150589  0.052946978 -0.106198451
##                 mix     pctymle
## county   0.056880585  0.09827258
## year    -0.018844849 -0.12704995
## crmrte  -0.119574049  0.22479470
## prbarr   0.347530281 -0.15627328
## prbconv  0.506170865 -0.06929779
## prbpris  0.029360316 -0.13324305
## avgsen  -0.047411804  0.05809590
## polpc    0.283141211 -0.03565262
## density -0.093370398  0.11165552
## taxpc    0.004043404 -0.12711257
## pctmin   0.236272398  0.02356500
## wcon    -0.070689347 -0.03923851
## wtuc    -0.052751214 -0.06849545
## wtrd    -0.048935885 -0.02918080
## wfir    -0.095038607 -0.06331100
## wser    -0.074800654 -0.01280587
## wmfg    -0.129246112 -0.03275346
```

```
## wfed    -0.129652147 -0.11015059
## wsta    -0.046547858  0.05294698
## wloc    -0.077462525 -0.10619845
## mix      1.000000000 -0.04259313
## pctymle -0.042593133  1.00000000
```

Some initial observations:

- predictors tend to be correlated
- salaries are highly correlated with each other and depend on the year
- the population density is strongly correlated with the crime rate (0.69)
- the probability of arrest is correlated to the crime rate (-0.36)
- the probability of conviction is correlated with the number of policemen per person (0.45)

The probability of being arrested vs minority percentage

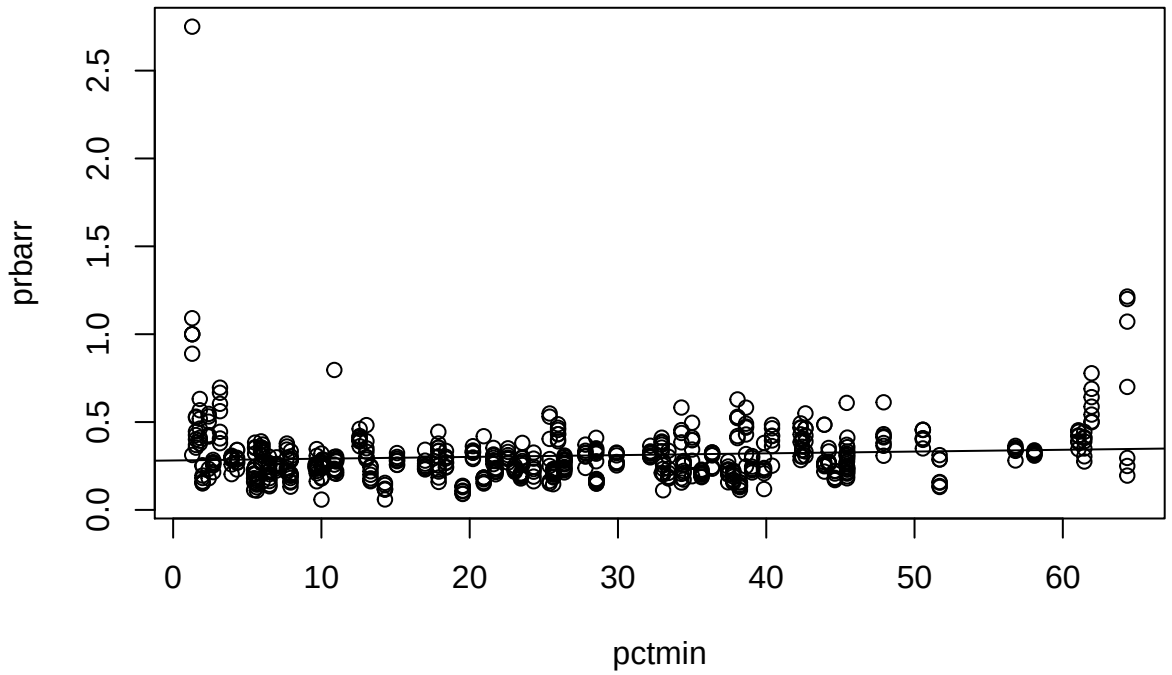
As the first part of the project, we wanted to check if the probability of being arrested depends on minority percentage. So we checked some regression models.

Simple linear regression.

```
lm.fit.simple <- lm(prbarr ~ pctmin, data=crime)
summary(lm.fit.simple)
```

```
##
## Call:
## lm(formula = prbarr ~ pctmin, data = crime)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.23626 -0.08758 -0.02535  0.04659  2.46739
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.2813123  0.0123714   22.74   <2e-16 ***
## pctmin      0.0010133  0.0004021    2.52    0.012 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1705 on 628 degrees of freedom
## Multiple R-squared:  0.01001,    Adjusted R-squared:  0.008434
## F-statistic:  6.35 on 1 and 628 DF,  p-value: 0.01199
```

```
plot(pctmin, prbarr)
abline(lm.fit.simple)
```



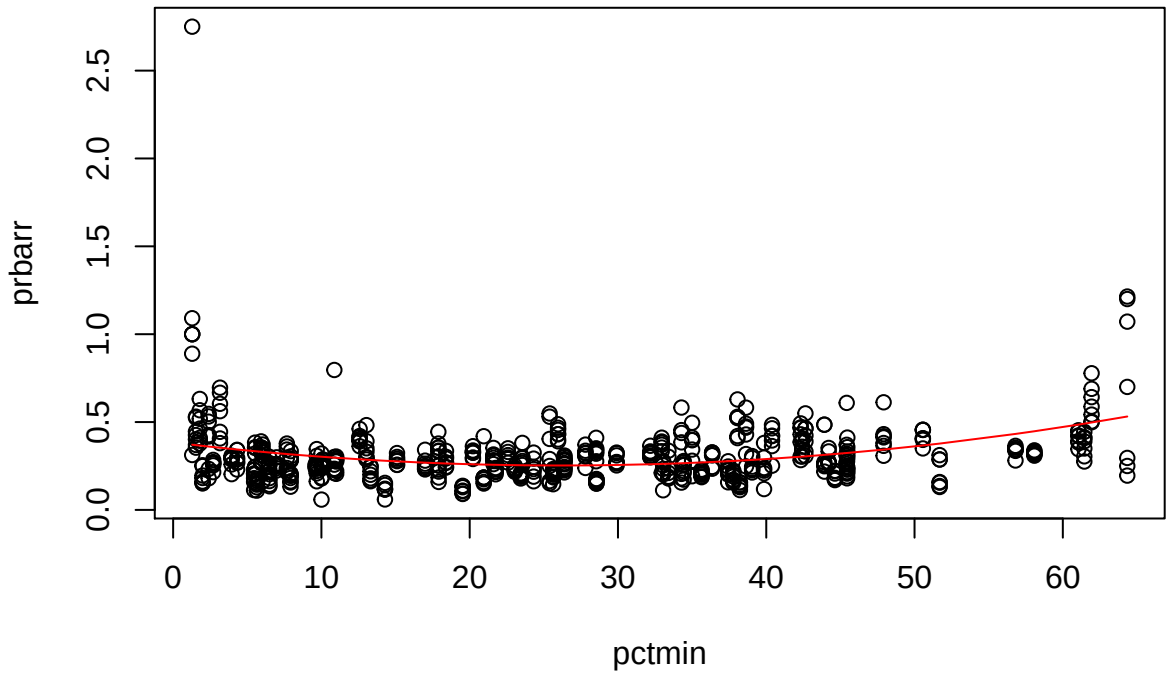
We can observe curves on the left and right side of the plot, so it is worth to try polynomial regression with even degrees, or something more flexible - for example a spline.

Simple poly regression - 2 degree.

```
lm.fit.poly2 <- lm(prbarr ~ poly(pctmin, 2), data=crime)
summary(lm.fit.poly2)
```

```
##
## Call:
## lm(formula = prbarr ~ poly(pctmin, 2), data = crime)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.33623 -0.08251 -0.02367  0.05239  2.37910
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      0.307368   0.006412  47.933   <2e-16 ***
## poly(pctmin, 2)1 0.429595   0.160951   2.669   0.0078 **
## poly(pctmin, 2)2 1.417556   0.160951   8.807   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.161 on 627 degrees of freedom
## Multiple R-squared:  0.119,  Adjusted R-squared:  0.1162
## F-statistic: 42.35 on 2 and 627 DF,  p-value: < 2.2e-16
```

```
plot(pctmin, prbarr)
lines(sort(pctmin), fitted(lm.fit.poly2)[order(pctmin)], col='red')
```




```
anova(lm.fit.simple, lm.fit.poly2)
```

```
## Analysis of Variance Table
##
## Model 1: prbarr ~ pctmin
## Model 2: prbarr ~ poly(pctmin, 2)
##   Res.Df    RSS Df Sum of Sq     F    Pr(>F)
## 1    628 18.252
## 2    627 16.243  1    2.0095 77.57 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

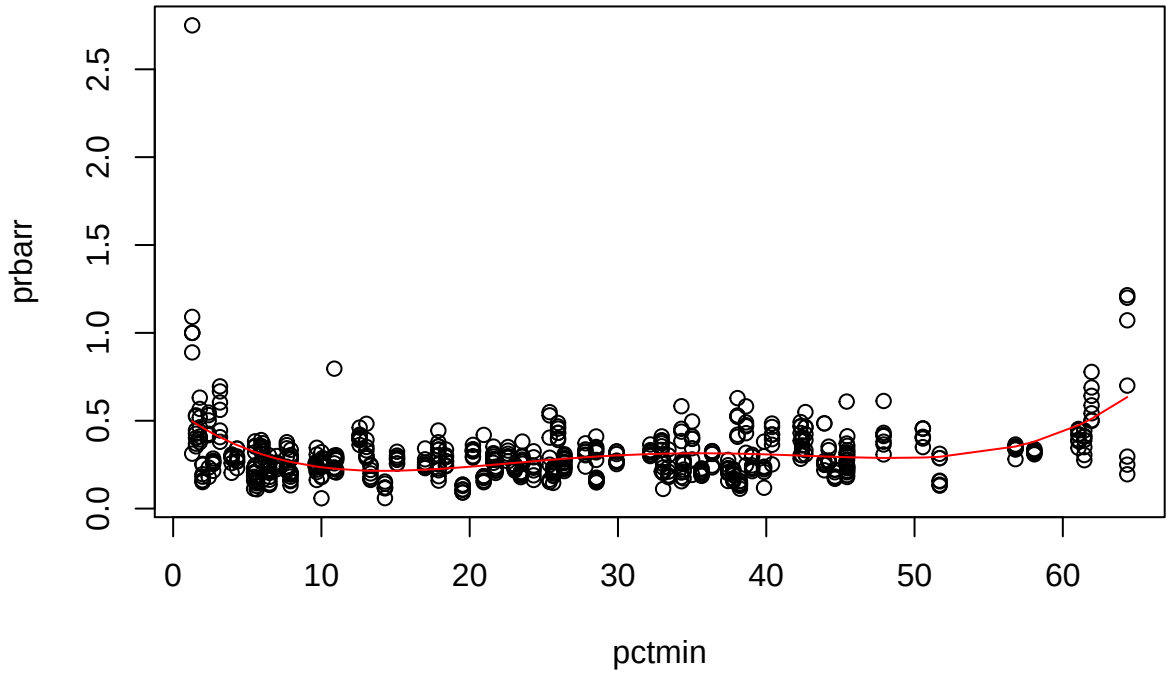
Simple poly regression - 4 degree.

```
lm.fit.poly4 <- lm(prbarr ~ poly(pctmin, 4), data=crime)
summary(lm.fit.poly4)
```

```
##
## Call:
## lm(formula = prbarr ~ poly(pctmin, 4), data = crime)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.43950 -0.07837 -0.01307  0.05788  2.25314
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       0.307368   0.006117  50.245  < 2e-16 ***
## poly(pctmin, 4)1  0.429595   0.153544   2.798  0.00530 **
## poly(pctmin, 4)2  1.417556   0.153544   9.232  < 2e-16 ***
## poly(pctmin, 4)3 -0.399190   0.153544  -2.600  0.00955 **
## poly(pctmin, 4)4  1.161222   0.153544   7.563 1.41e-13 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1535 on 625 degrees of freedom
## Multiple R-squared:  0.2008, Adjusted R-squared:  0.1957
## F-statistic: 39.25 on 4 and 625 DF,  p-value: < 2.2e-16
```

```
plot(pctmin, prbarr)
lines(sort(pctmin), fitted(lm.fit.poly4)[order(pctmin)], col='red')
```




```
anova(lm.fit.poly2, lm.fit.poly4)
```

```
## Analysis of Variance Table
##
## Model 1: prbarr ~ poly(pctmin, 2)
## Model 2: prbarr ~ poly(pctmin, 4)
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1    627 16.243
## 2    625 14.735  2    1.5078 31.978 5.996e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

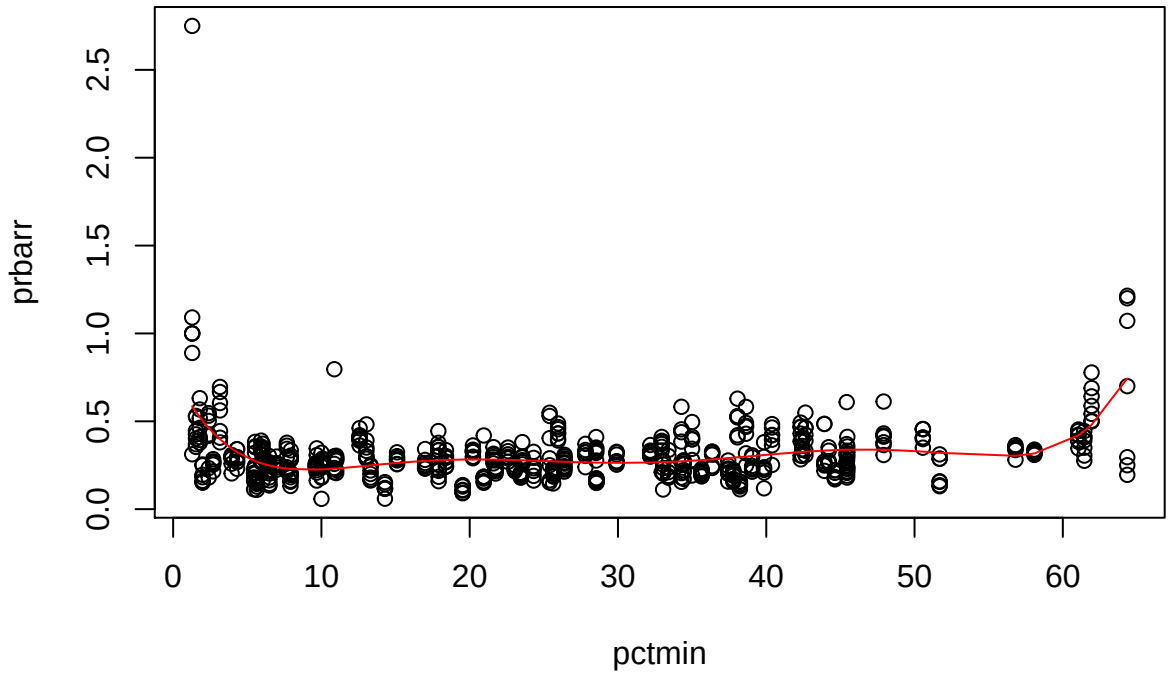
Simple poly regression - 6 degree.

```
lm.fit.poly6 <- lm(prbarr ~ poly(pctmin, 6), data=crime)
summary(lm.fit.poly6)
```

```
##
## Call:
## lm(formula = prbarr ~ poly(pctmin, 6), data = crime)
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -0.54825 -0.07156 -0.00676  0.05885  2.16470
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       0.307368   0.005929  51.843  < 2e-16 ***
## poly(pctmin, 6)1  0.429595   0.148813   2.887  0.00403 **
## poly(pctmin, 6)2  1.417556   0.148813   9.526  < 2e-16 ***
## poly(pctmin, 6)3 -0.399190   0.148813  -2.682  0.00750 **
## poly(pctmin, 6)4  1.161222   0.148813   7.803 2.56e-14 ***
## poly(pctmin, 6)5 -0.221576   0.148813  -1.489  0.13701
## poly(pctmin, 6)6  0.942945   0.148813   6.336 4.51e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1488 on 623 degrees of freedom
## Multiple R-squared:  0.2517, Adjusted R-squared:  0.2445
## F-statistic: 34.92 on 6 and 623 DF,  p-value: < 2.2e-16
```

```
plot(pctmin, prbarr)
lines(sort(pctmin), fitted(lm.fit.poly6)[order(pctmin)], col='red')
```




```
anova(lm.fit.poly4, lm.fit.poly6)
```

```
## Analysis of Variance Table
##
## Model 1: prbarr ~ poly(pctmin, 4)
## Model 2: prbarr ~ poly(pctmin, 6)
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1    625 14.735
## 2    623 13.797  2   0.93824 21.184 1.257e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

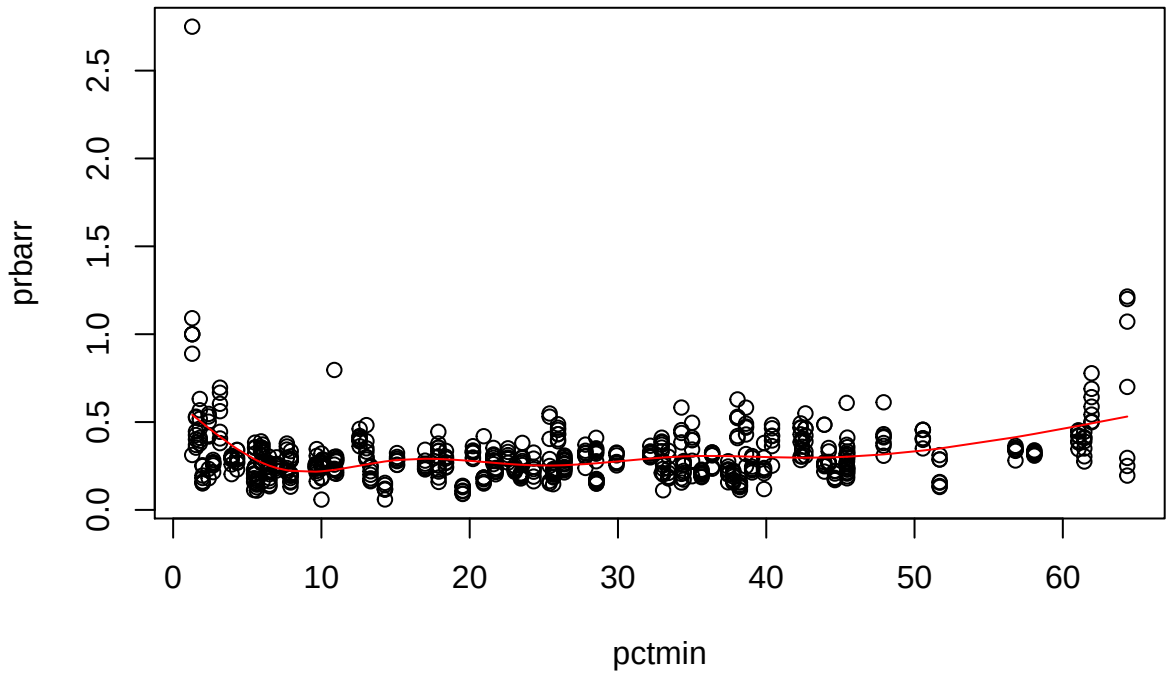
After analysing summaries and anova output we can see that increasing polynomal degree can improve the quality of regression fit.

Natural splines.

```
lm.fit.ns <- lm(prbarr ~ ns(pctmin, df = 6), data = crime)
summary(lm.fit.ns)
```

```
##
## Call:
## lm(formula = prbarr ~ ns(pctmin, df = 6), data = crime)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.34304 -0.08014 -0.01094  0.06014  2.20843
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)           0.54157    0.02550  21.236  < 2e-16 ***
## ns(pctmin, df = 6)1 -0.18102    0.04060  -4.458 9.79e-06 ***
## ns(pctmin, df = 6)2 -0.33907    0.04318  -7.852 1.79e-14 ***
## ns(pctmin, df = 6)3 -0.20853    0.03642  -5.725 1.61e-08 ***
## ns(pctmin, df = 6)4 -0.13816    0.03684  -3.751 0.000193 ***
## ns(pctmin, df = 6)5 -0.48771    0.06613  -7.375 5.27e-13 ***
## ns(pctmin, df = 6)6  0.20024    0.03471   5.769 1.26e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1531 on 623 degrees of freedom
## Multiple R-squared:  0.2075, Adjusted R-squared:  0.1999
## F-statistic: 27.19 on 6 and 623 DF,  p-value: < 2.2e-16
```

```
plot(pctmin, prbarr)
lines(sort(pctmin), fitted(lm.fit.ns)[order(pctmin)], col='red')
```




```
anova(lm.fit.poly6, lm.fit.ns)
```

```
## Analysis of Variance Table
##
## Model 1: prbarr ~ poly(pctmin, 6)
## Model 2: prbarr ~ ns(pctmin, df = 6)
##   Res.Df    RSS Df Sum of Sq F Pr(>F)
## 1    623 13.797
## 2    623 14.611  0  -0.81413
```

Natural splines can also be well-fitted.

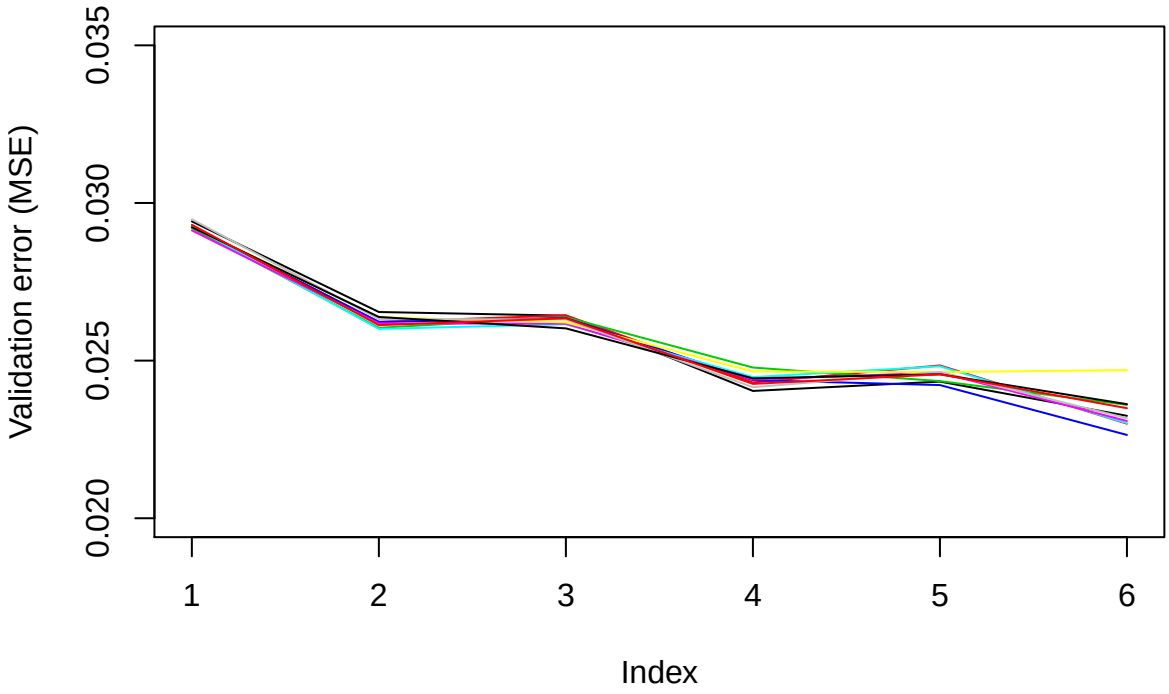
The curves at the ends of the chart can lead to interesting conclusions. Where there is a very large or very small minority percentage, the likelihood of being arrested is higher.

Regression comparison.

Validation error of linear and polynomial regressions to see how well regression fits to actual data.

```
mse.cv <- function(degree, k) {
  fit.glm <- glm(prbarr ~ poly(pctmin, degree), data = crime)
  cv.glm(crime, fit.glm, K = k)$delta[1]
}
mse <- replicate(10, sapply(1:6, mse.cv, k = 10))

plot(x = NULL, pch = 20, type = "l", ylab = "Validation error (MSE)", xlim = c(1, 6), ylim = c(0.02, 0.
for (i in 1:10) {
  points(mse[, i], pch = 20, type = "l", col = i)
}
```



We can see, that polynomial regression with even degree gives better results. The lowest mse was achieved for sixth degree polynomial, but at the same time the mse values for that degree varied the most

Probality of being arrested vs all predictors

The next step was to see which predictor affects the most the arrest probality, so we used the regression with all predictors.

```
lm.fit.all <- lm(prbarr ~ ., data=crime)
summary(lm.fit.all)
```

```
##
## Call:
## lm(formula = prbarr ~ ., data = crime)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.46979 -0.06056 -0.00661  0.04046  1.96283
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.302e-01  4.048e-01   1.557   0.1200
## county      -8.984e-05  9.353e-05  -0.960   0.3372
## year        -2.944e-03  5.510e-03  -0.534   0.5934
## crmrte      -5.431e+00  5.041e-01 -10.773  < 2e-16 ***
## prbconv     -4.238e-02  3.905e-03 -10.853  < 2e-16 ***
## prbpris     -1.134e-01  6.328e-02  -1.792   0.0737 .
## avgsen       3.635e-03  2.012e-03   1.807   0.0712 .
## polpc        3.060e+01  2.377e+00  12.872  < 2e-16 ***
## density      1.938e-02  9.107e-03   2.129   0.0337 *
## taxpc       -4.446e-05  5.830e-04  -0.076   0.9392
```

```
## regionother  8.845e-03  1.532e-02   0.577   0.5640
## regionwest   2.185e-02  1.840e-02   1.187   0.2355
## smsayes     -1.517e-02  3.658e-02  -0.415   0.6784
## pctmin       1.988e-03  5.049e-04   3.937  9.2e-05 ***
## wcon        -1.999e-05  4.510e-05  -0.443   0.6578
## wtuc        -1.631e-05  1.985e-05  -0.822   0.4115
## wtrd         3.506e-06  6.368e-05   0.055   0.9561
## wfir         1.078e-04  1.507e-04   0.715   0.4746
## wser        -5.597e-05  5.269e-05  -1.062   0.2885
## wmfg         1.181e-04  8.883e-05   1.330   0.1842
## wfed         6.884e-05  1.440e-04   0.478   0.6328
## wsta        -2.409e-04  1.489e-04  -1.618   0.1062
## wloc         2.446e-05  2.680e-04   0.091   0.9273
## mix          2.597e-01  2.938e-02   8.839  < 2e-16 ***
## pctymle     -3.476e-01  2.426e-01  -1.433   0.1524
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1295 on 605 degrees of freedom
## Multiple R-squared:  0.4501, Adjusted R-squared:  0.4283
## F-statistic: 20.63 on 24 and 605 DF,  p-value: < 2.2e-16
```

```
glm.fit.all <- glm(prbarr ~ ., data = crime)
cv.glm(crime, glm.fit.all, K = 10)$delta[1]
```

```
## [1] 0.02504085
```

We can see that crmrte, prbconv, polpc, pctmin and mix coefitients are the least likely to be zero. So they are the most important in regressions.

New factor variable

Next we decided to analyse arrest probability in more simple way and introduced a new variable `prbarr_high` indicating if probability of being arrested is high or not.

```
high <- factor(ifelse(prbarr <= 0.3, "No", "Yes"))
crime.h <- data.frame(crime[-4], prbarr_high=high)
detach(crime)
attach(crime.h)
summary(crime.h)
```

```
##      county           year         crmrte            prbconv
##  Min.   :  1.0   Min.   :81   Min.   :0.001812   Min.   : 0.06838
##  1st Qu.: 51.0   1st Qu.:82   1st Qu.:0.018352   1st Qu.: 0.34769
##  Median :103.0   Median :84   Median :0.028441   Median : 0.47437
##  Mean   :100.6   Mean   :84   Mean   :0.031588   Mean   : 0.68862
##  3rd Qu.:151.0   3rd Qu.:86   3rd Qu.:0.038406   3rd Qu.: 0.63560
##  Max.   :197.0   Max.   :87   Max.   :0.163835   Max.   :37.00000
##     prbpris           avgsen           polpc              density
##  Min.   :0.1489   Min.   : 4.220   Min.   :0.0004585   Min.   :0.1977
##  1st Qu.:0.3744   1st Qu.: 7.160   1st Qu.:0.0011913   1st Qu.:0.5329
##  Median :0.4286   Median : 8.495   Median :0.0014506   Median :0.9526
##  Mean   :0.4255   Mean   : 8.955   Mean   :0.0019168   Mean   :1.3861
##  3rd Qu.:0.4832   3rd Qu.:10.197   3rd Qu.:0.0018033   3rd Qu.:1.5078
##  Max.   :0.6786   Max.   :25.830   Max.   :0.0355781   Max.   :8.8277
```

```
##      taxpc            region     smsa         pctmin            wcon
##  Min.   : 14.30   central:238   no :574   Min.   : 1.284   Min.   :  65.62
##  1st Qu.: 23.43   other  :245   yes: 56   1st Qu.:10.005   1st Qu.: 201.66
##  Median : 27.79   west   :147             Median :24.852   Median : 236.46
##  Mean   : 30.24                           Mean   :25.713   Mean   : 245.67
##  3rd Qu.: 33.27                           3rd Qu.:38.223   3rd Qu.: 269.69
##  Max.   :119.76                           Max.   :64.348   Max.   :2324.60
##       wtuc              wtrd              wfir              wser
##  Min.   :  28.86   Min.   :  16.87   Min.   :  3.516   Min.   :   1.844
##  1st Qu.: 317.60   1st Qu.: 168.05   1st Qu.:235.705   1st Qu.: 191.319
##  Median : 358.20   Median : 185.48   Median :264.423   Median : 216.475
##  Mean   : 406.10   Mean   : 192.82   Mean   :272.059   Mean   : 224.671
##  3rd Qu.: 411.02   3rd Qu.: 204.82   3rd Qu.:302.440   3rd Qu.: 247.155
##  Max.   :3041.96   Max.   :2242.75   Max.   :509.466   Max.   :2177.068
##       wmfg             wfed             wsta             wloc
##  Min.   :101.8   Min.   :255.4   Min.   :173.0   Min.   :163.6
##  1st Qu.:234.0   1st Qu.:361.5   1st Qu.:258.2   1st Qu.:226.8
##  Median :271.6   Median :404.0   Median :289.4   Median :253.1
##  Mean   :285.2   Mean   :403.9   Mean   :296.9   Mean   :258.0
##  3rd Qu.:320.0   3rd Qu.:444.6   3rd Qu.:331.5   3rd Qu.:289.3
##  Max.   :646.9   Max.   :598.0   Max.   :548.0   Max.   :388.1
##       mix              pctymle        prbarr_high
##  Min.   :0.002457   Min.   :0.06216   No :367
##  1st Qu.:0.075324   1st Qu.:0.07859   Yes:263
##  Median :0.102089   Median :0.08316
##  Mean   :0.139396   Mean   :0.08897
##  3rd Qu.:0.149009   3rd Qu.:0.08919
##  Max.   :4.000000   Max.   :0.27436
```

```
names(crime.h)
```

```
##  [1] "county"      "year"        "crmrte"      "prbconv"     "prbpris"
##  [6] "avgsen"      "polpc"       "density"     "taxpc"       "region"
## [11] "smsa"        "pctmin"      "wcon"        "wtuc"        "wtrd"
## [16] "wfir"        "wser"        "wmfg"        "wfed"        "wsta"
## [21] "wloc"        "mix"         "pctymle"     "prbarr_high"
```

Generalized logistic regresion

We started with generalized logistic regresion based on all predictors.

```
set.seed(1)
n <- nrow(crime.h)
train <- sample(1:n, n / 2)
test <- -train

fit.logistic <- glm(prbarr_high ~ ., family = binomial, data = crime.h, subset = train)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(fit.logistic)
```

```
##
## Call:
## glm(formula = prbarr_high ~ ., family = binomial, data = crime.h,
##     subset = train)
```

```
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.70803  -0.63356  -0.03274   0.59978   2.10856
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  4.750e+01  1.550e+01   3.064  0.00219 **
## county       2.903e-03  2.896e-03   1.002  0.31629
## year        -6.039e-01  2.108e-01  -2.864  0.00418 **
## crmrte      -1.663e+02  2.783e+01  -5.975 2.30e-09 ***
## prbconv     -4.453e+00  6.950e-01  -6.407 1.48e-10 ***
## prbpris     -5.347e+00  2.256e+00  -2.370  0.01778 *
## avgsen      -6.960e-03  6.767e-02  -0.103  0.91808
## polpc        1.184e+03  2.208e+02   5.363 8.20e-08 ***
## density      1.077e-01  4.520e-01   0.238  0.81170
## taxpc        5.333e-02  2.538e-02   2.101  0.03561 *
## regionother -1.928e-01  5.869e-01  -0.329  0.74246
## regionwest   1.558e+00  6.353e-01   2.452  0.01419 *
## smsayes     -3.155e+00  1.756e+00  -1.797  0.07240 .
## pctmin       1.080e-01  2.525e-02   4.277 1.89e-05 ***
## wcon         1.030e-02  6.470e-03   1.591  0.11155
## wtuc         5.001e-04  6.580e-04   0.760  0.44727
## wtrd         1.630e-02  9.734e-03   1.674  0.09408 .
## wfir         1.963e-03  5.724e-03   0.343  0.73164
## wser         6.888e-03  5.894e-03   1.169  0.24259
## wmfg        -3.695e-03  2.765e-03  -1.336  0.18148
## wfed         3.741e-03  4.883e-03   0.766  0.44365
## wsta         7.747e-03  5.247e-03   1.476  0.13984
## wloc        -6.366e-03  8.332e-03  -0.764  0.44483
## mix          7.867e+00  2.766e+00   2.844  0.00446 **
## pctymle     -4.349e+01  1.689e+01  -2.574  0.01005 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 435.76  on 314  degrees of freedom
## Residual deviance: 249.22  on 290  degrees of freedom
## AIC: 299.22
##
## Number of Fisher Scoring iterations: 7
```

```
pred.logistic <- predict(fit.logistic, crime.h[test,], type = "response")
pred.logistic <- ifelse(pred.logistic > 0.5, "Yes", "No")

conf.logistic <- table(pred.logistic, prbarr_high[test])
conf.logistic
```

```
##
## pred.logistic  No Yes
##           No  157  27
##           Yes  44  87
```

```r
mean(pred.logistic != prbarr_high[test])
```

```
## [1] 0.2253968
```

We can see that most of the coeficients are likely to be 0 and there is not much sense in including them in our model. We used the `regsubsets` function to choose the best subset of predictors.

Choose the best subset of predictors

```r
fit.sub <- regsubsets(prbarr_high ~ ., data = crime.h, nvmax = 24)
fit.sub.summary <- summary(fit.sub)
fit.sub.summary
```

```
## Subset selection object
## Call: regsubsets.formula(prbarr_high ~ ., data = crime.h, nvmax = 24)
## 24 Variables  (and intercept)
##             Forced in Forced out
## county          FALSE      FALSE
## year            FALSE      FALSE
## crmrte          FALSE      FALSE
## prbconv         FALSE      FALSE
## prbpris         FALSE      FALSE
## avgsen          FALSE      FALSE
## polpc           FALSE      FALSE
## density         FALSE      FALSE
## taxpc           FALSE      FALSE
## regionother     FALSE      FALSE
## regionwest      FALSE      FALSE
## smsayes         FALSE      FALSE
## pctmin          FALSE      FALSE
## wcon            FALSE      FALSE
## wtuc            FALSE      FALSE
## wtrd            FALSE      FALSE
## wfir            FALSE      FALSE
## wser            FALSE      FALSE
## wmfg            FALSE      FALSE
## wfed            FALSE      FALSE
## wsta            FALSE      FALSE
## wloc            FALSE      FALSE
## mix             FALSE      FALSE
## pctymle         FALSE      FALSE
## 1 subsets of each size up to 24
## Selection Algorithm: exhaustive
##          county year crmrte prbconv prbpris avgsen polpc density taxpc
## 1  ( 1 ) " "    " "  "*"    " "     " "     " "    " "   " "     " "
## 2  ( 1 ) " "    " "  "*"    " "     " "     " "    " "   " "     " "
## 3  ( 1 ) " "    " "  "*"    "*"     " "     " "    " "   " "     " "
## 4  ( 1 ) " "    " "  "*"    "*"     " "     " "    "*"   " "     " "
## 5  ( 1 ) " "    " "  "*"    "*"     " "     " "    "*"   " "     " "
## 6  ( 1 ) " "    " "  "*"    "*"     " "     " "    "*"   " "     " "
## 7  ( 1 ) " "    " "  "*"    "*"     " "     " "    "*"   " "     " "
## 8  ( 1 ) "*"    " "  "*"    "*"     " "     " "    "*"   " "     " "
## 9  ( 1 ) "*"    " "  "*"    "*"     " "     " "    "*"   "*"     " "
```

```
## 10  ( 1 ) "*"    " "  "*"    "*"     " "     " "    "*"   "*"     " "
## 11  ( 1 ) "*"    " "  "*"    "*"     " "     " "    "*"   "*"     " "
## 12  ( 1 ) "*"    " "  "*"    "*"     " "     " "    "*"   "*"     " "
## 13  ( 1 ) "*"    " "  "*"    "*"     "*"     " "    "*"   "*"     " "
## 14  ( 1 ) "*"    " "  "*"    "*"     "*"     "*"    "*"   "*"     " "
## 15  ( 1 ) "*"    " "  "*"    "*"     "*"     "*"    "*"   "*"     " "
## 16  ( 1 ) "*"    " "  "*"    "*"     "*"     "*"    "*"   "*"     " "
## 17  ( 1 ) "*"    " "  "*"    "*"     "*"     "*"    "*"   "*"     " "
## 18  ( 1 ) "*"    " "  "*"    "*"     "*"     "*"    "*"   "*"     " "
## 19  ( 1 ) "*"    " "  "*"    "*"     "*"     "*"    "*"   "*"     " "
## 20  ( 1 ) "*"    " "  "*"    "*"     "*"     "*"    "*"   "*"     " "
## 21  ( 1 ) "*"    " "  "*"    "*"     "*"     "*"    "*"   "*"     "*"
## 22  ( 1 ) "*"    " "  "*"    "*"     "*"     "*"    "*"   "*"     "*"
## 23  ( 1 ) "*"    " "  "*"    "*"     "*"     "*"    "*"   "*"     "*"
## 24  ( 1 ) "*"    "*"  "*"    "*"     "*"     "*"    "*"   "*"     "*"
##           regionother regionwest smsayes pctmin wcon wtuc wtrd wfir wser wmfg
## 1  ( 1 )  " "         " "        " "     " "    " "  " "  " "  " "  " "  " "
## 2  ( 1 )  " "         " "        " "     "*"    " "  " "  " "  " "  " "  " "
## 3  ( 1 )  " "         " "        " "     "*"    " "  " "  " "  " "  " "  " "
## 4  ( 1 )  " "         " "        " "     "*"    " "  " "  " "  " "  " "  " "
## 5  ( 1 )  " "         " "        " "     "*"    " "  " "  " "  " "  " "  " "
## 6  ( 1 )  " "         " "        " "     "*"    " "  " "  " "  " "  " "  " "
## 7  ( 1 )  " "         "*"        " "     "*"    " "  " "  " "  " "  " "  " "
## 8  ( 1 )  " "         "*"        " "     "*"    " "  " "  " "  " "  " "  " "
## 9  ( 1 )  " "         "*"        " "     "*"    " "  " "  " "  " "  " "  " "
## 10  ( 1 ) " "         "*"        " "     "*"    " "  " "  " "  " "  "*"  " "  " "
## 11  ( 1 ) " "         "*"        " "     "*"    " "  " "  " "  " "  "*"  "*"  " "
## 12  ( 1 ) " "         "*"        " "     "*"    " "  " "  " "  " "  "*"  "*"  " "
## 13  ( 1 ) " "         "*"        " "     "*"    " "  " "  " "  " "  "*"  "*"  " "
## 14  ( 1 ) " "         "*"        " "     "*"    " "  " "  " "  " "  "*"  "*"  " "
## 15  ( 1 ) "*"         "*"        " "     "*"    " "  " "  " "  " "  "*"  "*"  " "
## 16  ( 1 ) "*"         "*"        " "     "*"    "*"  " "  " "  " "  "*"  "*"  " "
## 17  ( 1 ) "*"         "*"        " "     "*"    "*"  " "  "*"  "*"  "*"  " "  " "
## 18  ( 1 ) "*"         "*"        " "     "*"    "*"  " "  "*"  "*"  "*"  " "  " "
## 19  ( 1 ) "*"         "*"        "*"     "*"    "*"  " "  "*"  "*"  "*"  " "  " "
## 20  ( 1 ) "*"         "*"        "*"     "*"    "*"  "*"  "*"  "*"  "*"  " "  " "
## 21  ( 1 ) "*"         "*"        "*"     "*"    "*"  "*"  "*"  "*"  "*"  " "  " "
## 22  ( 1 ) "*"         "*"        "*"     "*"    "*"  "*"  "*"  "*"  "*"  "*"  "*"
## 23  ( 1 ) "*"         "*"        "*"     "*"    "*"  "*"  "*"  "*"  "*"  "*"  "*"
## 24  ( 1 ) "*"         "*"        "*"     "*"    "*"  "*"  "*"  "*"  "*"  "*"  "*"
##           wfed wsta wloc mix pctymle
## 1  ( 1 )  " "  " "  " "  " " " " " "
## 2  ( 1 )  " "  " "  " "  " " " " " "
## 3  ( 1 )  " "  " "  " "  " " " " " "
## 4  ( 1 )  " "  " "  " "  " " " " " "
## 5  ( 1 )  " "  " "  " "  "*" " " " "
## 6  ( 1 )  " "  " "  " "  "*" "*" " "
## 7  ( 1 )  " "  " "  " "  "*" "*" " "
## 8  ( 1 )  " "  " "  " "  "*" "*" " "
## 9  ( 1 )  " "  " "  " "  "*" "*" " "
## 10  ( 1 ) " "  " "  " "  "*" "*" " "
## 11  ( 1 ) " "  " "  " "  "*" "*" " "
## 12  ( 1 ) " "  " "  "*"  "*" "*" " "
## 13  ( 1 ) " "  " "  "*"  "*" "*" " "
```

```
## 14  ( 1 ) " "  " "  "*"  "*" "*"
## 15  ( 1 ) " "  " "  "*"  "*" "*"
## 16  ( 1 ) " "  " "  "*"  "*" "*"
## 17  ( 1 ) " "  " "  "*"  "*" "*"
## 18  ( 1 ) "*"  " "  "*"  "*" "*"
## 19  ( 1 ) "*"  " "  "*"  "*" "*"
## 20  ( 1 ) "*"  " "  "*"  "*" "*"
## 21  ( 1 ) "*"  " "  "*"  "*" "*"
## 22  ( 1 ) "*"  " "  "*"  "*" "*"
## 23  ( 1 ) "*"  "*"  "*"  "*" "*"
## 24  ( 1 ) "*"  "*"  "*"  "*" "*"
```

```
min.sub <- which.min(fit.sub.summary$bic)
min.sub
```

```
## [1] 6
```

```
mask <- fit.sub.summary$which[min.sub, -1]
predictors <- names(which(mask == TRUE))
predictors
```

```
## [1] "crmrte"  "prbconv" "polpc"   "pctmin"  "mix"     "pctymle"
```

We included the most promising predictors: crime rate, conviction probability, police per capita, minority percentage, mix and percentage of young males.

Generalized logistic regresion again

```
fit.logistic <- glm(prbarr_high ~ crmrte + prbconv + polpc + pctmin + mix + pctymle, family = binomial,
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(fit.logistic)
```

```
##
## Call:
## glm(formula = prbarr_high ~ crmrte + prbconv + polpc + pctmin +
##     mix + pctymle, family = binomial, data = crime.h, subset = train)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.70487  -0.72592  -0.09222   0.74494   2.12831
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)    4.05089    1.15616   3.504 0.000459 ***
## crmrte      -132.78485   18.43025  -7.205 5.82e-13 ***
## prbconv       -3.52467    0.58878  -5.986 2.15e-09 ***
## polpc        933.02244  167.16218   5.582 2.38e-08 ***
## pctmin         0.05734    0.01073   5.342 9.19e-08 ***
## mix            7.36109    2.58827   2.844 0.004455 **
## pctymle      -24.09139   12.01982  -2.004 0.045037 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
```

```
##
##     Null deviance: 435.76  on 314  degrees of freedom
## Residual deviance: 274.14  on 308  degrees of freedom
## AIC: 288.14
##
## Number of Fisher Scoring iterations: 7
```

```
probs.logistic <- predict(fit.logistic, crime.h[test,], type = "response")
head(probs.logistic)
```

```
##         4         5         6         7         8         9
## 0.2289272 0.2338282 0.3419389 0.2774085 0.1342704 0.1387424
```

```
pred.logistic <- ifelse(probs.logistic > 0.5, "Yes", "No")

conf.logistic <- table(pred.logistic, prbarr_high[test])
conf.logistic
```

```
##
## pred.logistic  No Yes
##           No  167  26
##           Yes  34  88
```

```
mean(pred.logistic != prbarr_high[test])
```

```
## [1] 0.1904762
```

As we can see the error rate has decreased so we have not dicreased the regression quality by removing some predictors.

In the first part we analysed the arrest probability vs. minority precentage. We decided to do the same with new `prbarr_high` variable.

```
fit.logistic.single <- glm(prbarr_high ~ pctmin, family = binomial, data = crime.h, subset = train)
summary(fit.logistic.single)
```

```
##
## Call:
## glm(formula = prbarr_high ~ pctmin, family = binomial, data = crime.h,
##     subset = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.4018  -1.1017  -0.9515   1.1664   1.4341
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.611228   0.203067  -3.010  0.00261 **
## pctmin       0.019799   0.006585   3.007  0.00264 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 435.76  on 314  degrees of freedom
## Residual deviance: 426.47  on 313  degrees of freedom
## AIC: 430.47
##
```

```
## Number of Fisher Scoring iterations: 4
```

```
probs.logistic.single <- predict(fit.logistic.single, crime.h[test,], type = "response")
pred.logistic.single <- ifelse(probs.logistic.single > 0.5, "Yes", "No")

conf.logistic.single <- table(pred.logistic.single, prbarr_high[test])
conf.logistic.single
```

```
##
## pred.logistic.single  No Yes
##                   No  144  52
##                   Yes  57  62
```

```
mean(pred.logistic.single != prbarr_high[test])
```

```
## [1] 0.3460317
```

```
# poly 4

fit.logistic.single <- glm(prbarr_high ~ poly(pctmin, 4), family = binomial, data = crime.h, subset = t
summary(fit.logistic.single)
```

```
##
## Call:
## glm(formula = prbarr_high ~ poly(pctmin, 4), family = binomial,
##     data = crime.h, subset = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2009  -1.0474  -0.8265   1.2405   1.5879
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)        0.03489    0.16127   0.216  0.82872
## poly(pctmin, 4)1 19.34046    6.42485   3.010  0.00261 **
## poly(pctmin, 4)2 24.96599    7.69581   3.244  0.00118 **
## poly(pctmin, 4)3  7.50436    6.02641   1.245  0.21304
## poly(pctmin, 4)4 11.81987    4.30293   2.747  0.00602 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 435.76  on 314  degrees of freedom
## Residual deviance: 392.98  on 310  degrees of freedom
## AIC: 402.98
##
## Number of Fisher Scoring iterations: 6
```

```
probs.logistic.single <- predict(fit.logistic.single, crime.h[test,], type = "response")
pred.logistic.single <- ifelse(probs.logistic.single > 0.5, "Yes", "No")

conf.logistic.single <- table(pred.logistic.single, prbarr_high[test])
conf.logistic.single
```

```
##
## pred.logistic.single  No Yes
```

```
##                  No  185  80
##                  Yes  16  34
```

```
mean(pred.logistic.single != prbarr_high[test])
```

```
## [1] 0.3047619
```

```
# poly 6

fit.logistic.single <- glm(prbarr_high ~ poly(pctmin, 6), family = binomial, data = crime.h, subset = t
summary(fit.logistic.single)
```

```
##
## Call:
## glm(formula = prbarr_high ~ poly(pctmin, 6), family = binomial,
##     data = crime.h, subset = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0490  -1.0072  -0.8314   1.1915   1.5729
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)       0.08604    0.23186   0.371   0.7106
## poly(pctmin, 6)1 22.26575   11.36971   1.958   0.0502 .
## poly(pctmin, 6)2 29.13378   14.79372   1.969   0.0489 *
## poly(pctmin, 6)3 11.40708   11.64035   0.980   0.3271
## poly(pctmin, 6)4 15.08465    8.73633   1.727   0.0842 .
## poly(pctmin, 6)5  1.69304    7.69223   0.220   0.8258
## poly(pctmin, 6)6  6.47784    5.40577   1.198   0.2308
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 435.76  on 314  degrees of freedom
## Residual deviance: 390.32  on 308  degrees of freedom
## AIC: 404.32
##
## Number of Fisher Scoring iterations: 7
```

```
probs.logistic.single <- predict(fit.logistic.single, crime.h[test,], type = "response")
pred.logistic.single <- ifelse(probs.logistic.single > 0.5, "Yes", "No")

conf.logistic.single <- table(pred.logistic.single, prbarr_high[test])
conf.logistic.single
```

```
##
## pred.logistic.single  No Yes
##                  No  174  73
##                  Yes  27  41
```

```
mean(pred.logistic.single != prbarr_high[test])
```

```
## [1] 0.3174603
```

This time the regression result were very diffent and we could observe that for higher polynomial degrees

many coefficients are likely to be zero. Even for linear regression the significant code was lower.

Classification of high probality of being arrested

Finally we wanted to build clasificator, that could tell us if the probality of being arrested is high or low. Our first idea was to use knn classificator.

Knn classification

```
set.seed(1)
n <- nrow(crime.h)
train <- sample(1:n, n / 2)
test <- -train
train.set <- crime.h[train, c("crmrte", "prbconv", "polpc", "pctmin", "mix", "pctymle")]
test.set <- crime.h[-train, c("crmrte", "prbconv", "polpc", "pctmin", "mix", "pctymle")]

knn.f <- function(k) {
  pred.knn <- knn(train.set, test.set, prbarr_high[train], k = k)
}

knn.preds <- sapply(1:5, knn.f)

mean.f <- function(i) {
  mean.knn <- mean(knn.preds[,i] != prbarr_high[test])
}

knn.means <- sapply(1:5, mean.f)
knn.means
```

```
## [1] 0.1968254 0.2000000 0.2031746 0.2857143 0.2603175
```

```
table(knn.preds[,1], prbarr_high[test])
```

```
##
##        No Yes
##   No  164  25
##   Yes  37  89
```

```
table(knn.preds[,3], prbarr_high[test])
```

```
##
##        No Yes
##   No  167  30
##   Yes  34  84
```

```
table(knn.preds[,5], prbarr_high[test])
```

```
##
##        No Yes
##   No  166  47
##   Yes  35  67
```

We've tested several k values and k=1 turned out to be the best. Next we tried bagging algorithm.

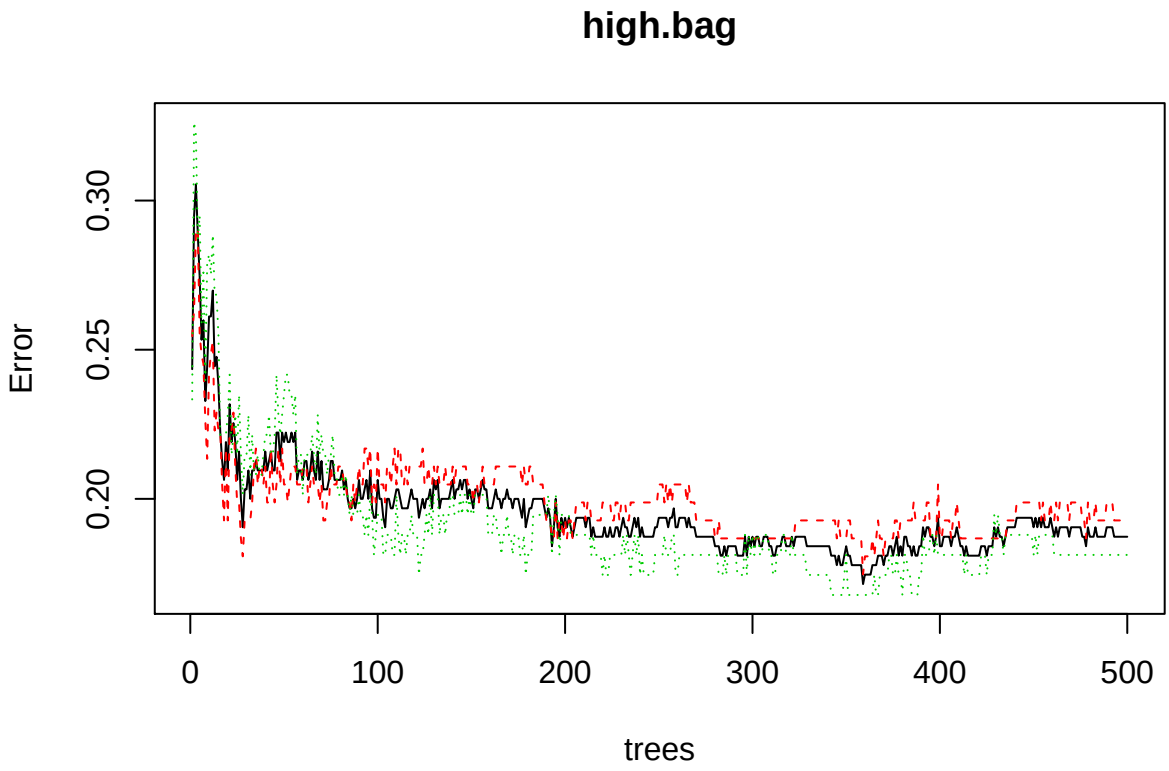
Bagging classification

```
set.seed(1)
n <- nrow(crime.h)
train <- sample(1:n, n / 2)
test <- -train

high.bag <- randomForest(prbarr_high ~ ., data = crime.h, subset = train, mtry = 23, importance = TRUE)
high.bag.pred.train <- predict(high.bag, newdata = crime.h[train,])
mean(high.bag.pred.train  != prbarr_high[train])
```

```
## [1] 0
```

```
plot(high.bag, type = "l")
```

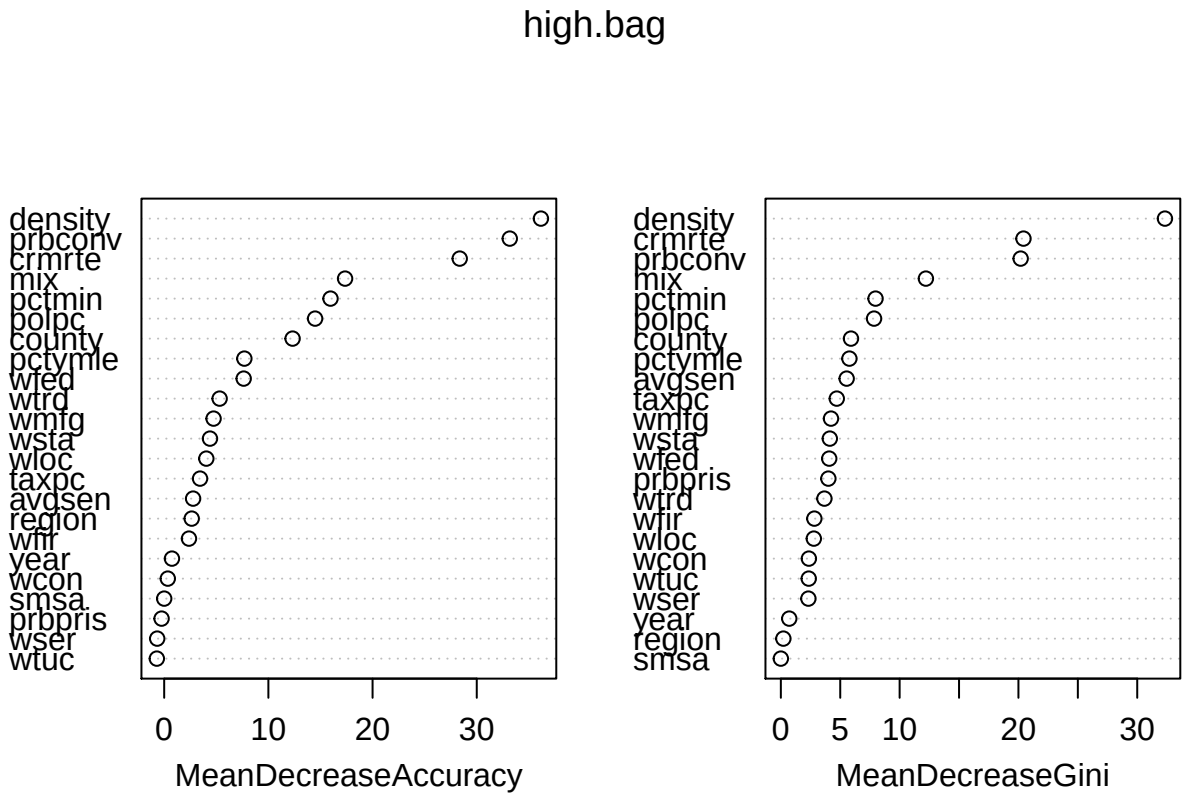



```
importance(high.bag)
```

```
##                 No        Yes MeanDecreaseAccuracy MeanDecreaseGini
## county   4.7290571 12.9769482           12.3252906        5.9041831
## year     1.7257303 -0.6051311            0.7445597        0.7090820
## crmrte  19.8255903 19.7821870           28.3644735       20.4238620
## prbconv 33.4133960 13.0808045           33.1706068       20.1868197
## prbpris  1.1362091 -1.1708974           -0.2498029        4.0001472
## avgsen   2.9364640  0.8623922            2.7752226        5.5398488
## polpc    3.9748969 15.3367431           14.4852178        7.8461596
## density 30.9706433 20.1051324           36.1591770       32.3310289
## taxpc    1.7541186  2.7205787            3.4417192        4.7080945
## region   0.3930627  2.7689991            2.6412219        0.2124385
## smsa     0.0000000  0.0000000            0.0000000        0.0000000
## pctmin  13.1403793  9.6991596           15.9656442        7.9722720
```

```
## wcon    -0.7731892  1.2053536          0.3467319        2.3604827
## wtuc    -0.3222998 -0.5687795         -0.7073388        2.3493978
## wtrd     4.5220134  3.0031565          5.3223320        3.6657084
## wfir    -1.0909950  3.6614726          2.3789531        2.8203738
## wser    -1.8156271  0.7146301         -0.6714202        2.3187490
## wmfg     4.9590244  1.7261484          4.7396087        4.2220418
## wfed     1.4622924  8.5171023          7.6294248        4.0721700
## wsta     0.7308183  4.8830648          4.3962455        4.1233755
## wloc     5.1844445  0.1685732          4.0439315        2.7755639
## mix     17.2944349  8.3443516         17.3613058       12.2090146
## pctymle  4.1792507  5.9445283          7.6950193        5.7721066
```

```
varImpPlot(high.bag)
```




```
high.bag.pred <- predict(high.bag, newdata = crime.h[test,], n.trees = 5000)
mean(high.bag.pred != prbarr_high[test])
```

```
## [1] 0.1650794
```

Bagging turned out to be better classificator than knn. On the plot we can see that density, conviction probality and crime rate have the most important to increasing accuracy.

Finally we wanted to see the example of regression tree.

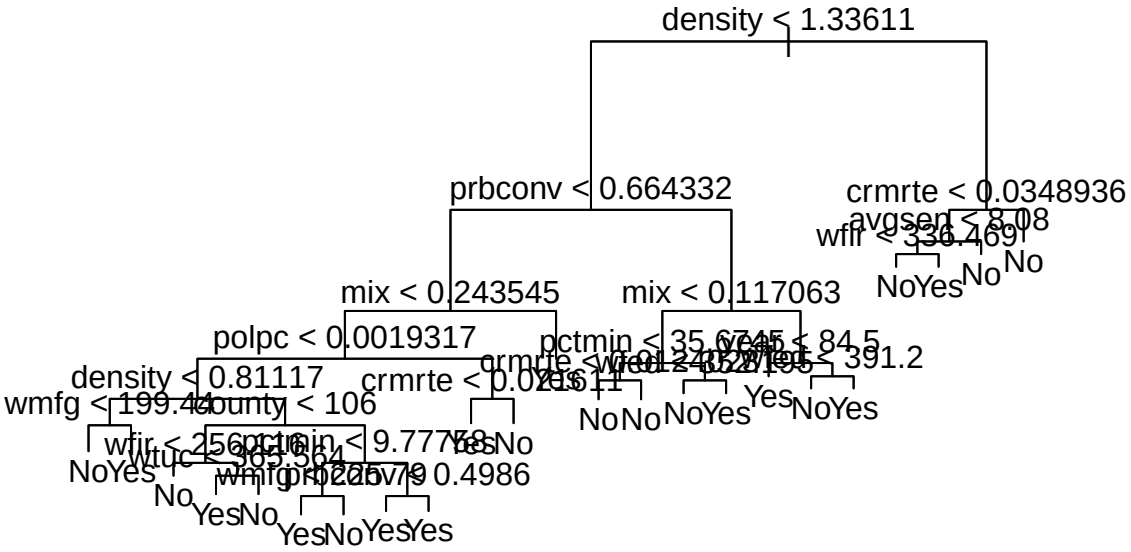
Regression classification

```
prbarr_high.tree <- tree(prbarr_high ~ ., data = crime.h)
summary(prbarr_high.tree)
```

```
##
```

```
## Classification tree:
## tree(formula = prbarr_high ~ ., data = crime.h)
## Variables actually used in tree construction:
##  [1] "density" "prbconv" "mix"     "polpc"   "wmfg"    "county"  "wfir"
##  [8] "wtuc"    "pctmin"  "crmrte"  "wfed"    "year"    "avgsen"
## Number of terminal nodes:  23
## Residual mean deviance:  0.5288 = 321 / 607
## Misclassification error rate: 0.1032 = 65 / 630
```

```
plot(prbarr_high.tree)
text(prbarr_high.tree, pretty = 0)
```



Again we can see the same predictors with the biggest impact on classification results.