Assignment5

Prashant Kubsad

6/14/2020

Question 11.1

The process I followed in answering this question is to determine significant factors for each of the following methods.

- Stepwise:
  - Backward Elimination
  - Forward Selection
  - Stepwise (Both directions)
  - Select the model that is best amongst above for stepwise
- Lasso
- Elastic Net
  - Run loop for alpha = 0.1 to 0.99 and create multiple models.
  - Select the best model using lowest mean cross-validated error
- Run a PCA analysis on the given data.
- Stepwise on Principal Components
- Lasso on Principal Components
- Elastic net on principal Components.
- Run Leave one out cross validation on all the above 6 models
- Calculate R2 for each of the models using above step.
- Conclusion of best method using R2 from above step.

Stepwise regression

Before doing stepwise regression, I would like to do a **backward elimination** and **forward selection** variable selection mehtods and compare it with **stepwise** variable selection method. As mentioned in the piazza post : https://piazza.com/class/ka28g4qhcw67bo?cid=668 , there is no need of scaling the data for stepwise regression.

Backward elimination

I am using step function to perform the backward elimination first.

```
#cleaning environment and starting fresh.
rm(list = ls())
#setting seed for consistent results
set.seed(777)
library(caret)
```

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

```
## Loading required package: ggplot2
```

```
library(olsrr)
```

```
##
```

```
## Attaching package: 'olsrr'

## The following object is masked from 'package:datasets':
##
##     rivers
```

```r
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```r
library(ggplot2)
library(MASS)
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:olsrr':
##
##     cement
```

```r
library(glmnet)
```

```
## Loading required package: Matrix

## Loaded glmnet 4.0
```

```r
#Read the text file into data.
data<-read.table("uscrime.txt", header=TRUE)
cat("****** backward step regression ")
```

```
## ****** backward step regression
```

```r
backward<-step(lm(Crime~.,data=data),direction = "backward")
```

```
## Start:  AIC=514.65
## Crime ~ M + So + Ed + Po1 + Po2 + LF + M.F + Pop + NW + U1 +
##     U2 + Wealth + Ineq + Prob + Time
##
##          Df Sum of Sq     RSS    AIC
## - So      1        29 1354974 512.65
## - LF      1      8917 1363862 512.96
## - Time    1     10304 1365250 513.00
## - Pop     1     14122 1369068 513.14
## - NW      1     18395 1373341 513.28
## - M.F     1     31967 1386913 513.74
## - Wealth  1     37613 1392558 513.94
## - Po2     1     37919 1392865 513.95
## <none>                1354946 514.65
## - U1      1     83722 1438668 515.47
## - Po1     1    144306 1499252 517.41
## - U2      1    181536 1536482 518.56
## - M       1    193770 1548716 518.93
## - Prob    1    199538 1554484 519.11
## - Ed      1    402117 1757063 524.86
## - Ineq    1    423031 1777977 525.42
##
## Step:  AIC=512.65
```

```
## Crime ~ M + Ed + Po1 + Po2 + LF + M.F + Pop + NW + U1 + U2 +
##     Wealth + Ineq + Prob + Time
##
##          Df Sum of Sq     RSS    AIC
## - Time    1     10341 1365315 511.01
## - LF      1     10878 1365852 511.03
## - Pop     1     14127 1369101 511.14
## - NW      1     21626 1376600 511.39
## - M.F     1     32449 1387423 511.76
## - Po2     1     37954 1392929 511.95
## - Wealth  1     39223 1394197 511.99
## <none>                1354974 512.65
## - U1      1     96420 1451395 513.88
## - Po1     1    144302 1499277 515.41
## - U2      1    189859 1544834 516.81
## - M       1    195084 1550059 516.97
## - Prob    1    204463 1559437 517.26
## - Ed      1    403140 1758114 522.89
## - Ineq    1    488834 1843808 525.13
##
## Step:  AIC=511.01
## Crime ~ M + Ed + Po1 + Po2 + LF + M.F + Pop + NW + U1 + U2 +
##     Wealth + Ineq + Prob
##
##          Df Sum of Sq     RSS    AIC
## - LF      1     10533 1375848 509.37
## - NW      1     15482 1380797 509.54
## - Pop     1     21846 1387161 509.75
## - Po2     1     28932 1394247 509.99
## - Wealth  1     36070 1401385 510.23
## - M.F     1     41784 1407099 510.42
## <none>                1365315 511.01
## - U1      1     91420 1456735 512.05
## - Po1     1    134137 1499452 513.41
## - U2      1    184143 1549458 514.95
## - M       1    186110 1551425 515.01
## - Prob    1    237493 1602808 516.54
## - Ed      1    409448 1774763 521.33
## - Ineq    1    502909 1868224 523.75
##
## Step:  AIC=509.37
## Crime ~ M + Ed + Po1 + Po2 + M.F + Pop + NW + U1 + U2 + Wealth +
##     Ineq + Prob
##
##          Df Sum of Sq     RSS    AIC
## - NW      1     11675 1387523 507.77
## - Po2     1     21418 1397266 508.09
## - Pop     1     27803 1403651 508.31
## - M.F     1     31252 1407100 508.42
## - Wealth  1     35035 1410883 508.55
## <none>                1375848 509.37
## - U1      1     80954 1456802 510.06
## - Po1     1    123896 1499744 511.42
## - U2      1    190746 1566594 513.47
```

```
## - M       1    217716 1593564 514.27
## - Prob    1    226971 1602819 514.54
## - Ed      1    413254 1789103 519.71
## - Ineq    1    500944 1876792 521.96
##
## Step:  AIC=507.77
## Crime ~ M + Ed + Po1 + Po2 + M.F + Pop + U1 + U2 + Wealth + Ineq +
##     Prob
##
##          Df Sum of Sq     RSS    AIC
## - Po2     1     16706 1404229 506.33
## - Pop     1     25793 1413315 506.63
## - M.F     1     26785 1414308 506.66
## - Wealth  1     31551 1419073 506.82
## <none>                1387523 507.77
## - U1      1     83881 1471404 508.52
## - Po1     1    118348 1505871 509.61
## - U2      1    201453 1588976 512.14
## - Prob    1    216760 1604282 512.59
## - M       1    309214 1696737 515.22
## - Ed      1    402754 1790276 517.74
## - Ineq    1    589736 1977259 522.41
##
## Step:  AIC=506.33
## Crime ~ M + Ed + Po1 + M.F + Pop + U1 + U2 + Wealth + Ineq +
##     Prob
##
##          Df Sum of Sq     RSS    AIC
## - Pop     1     22345 1426575 505.07
## - Wealth  1     32142 1436371 505.39
## - M.F     1     36808 1441037 505.54
## <none>                1404229 506.33
## - U1      1     86373 1490602 507.13
## - U2      1    205814 1610043 510.76
## - Prob    1    218607 1622836 511.13
## - M       1    307001 1711230 513.62
## - Ed      1    389502 1793731 515.83
## - Ineq    1    608627 2012856 521.25
## - Po1     1   1050202 2454432 530.57
##
## Step:  AIC=505.07
## Crime ~ M + Ed + Po1 + M.F + U1 + U2 + Wealth + Ineq + Prob
##
##          Df Sum of Sq     RSS    AIC
## - Wealth  1     26493 1453068 503.93
## <none>                1426575 505.07
## - M.F     1     84491 1511065 505.77
## - U1      1     99463 1526037 506.24
## - Prob    1    198571 1625145 509.20
## - U2      1    208880 1635455 509.49
## - M       1    320926 1747501 512.61
## - Ed      1    386773 1813348 514.35
## - Ineq    1    594779 2021354 519.45
## - Po1     1   1127277 2553852 530.44
```

```
##
## Step:  AIC=503.93
## Crime ~ M + Ed + Po1 + M.F + U1 + U2 + Ineq + Prob
##
##        Df Sum of Sq     RSS    AIC
## <none>              1453068 503.93
## - M.F   1    103159 1556227 505.16
## - U1    1    127044 1580112 505.87
## - Prob  1    247978 1701046 509.34
## - U2    1    255443 1708511 509.55
## - M     1    296790 1749858 510.67
## - Ed    1    445788 1898855 514.51
## - Ineq  1    738244 2191312 521.24
## - Po1   1   1672038 3125105 537.93
```

```
backward
```

```
##
## Call:
## lm(formula = Crime ~ M + Ed + Po1 + M.F + U1 + U2 + Ineq + Prob,
##     data = data)
##
## Coefficients:
## (Intercept)            M           Ed          Po1          M.F           U1
##    -6426.10        93.32       180.12       102.65        22.34     -6086.63
##          U2         Ineq         Prob
##      187.35        61.33     -3796.03
```

```
backwardModel<-lm(Crime~M + Ed + Po1 + M.F + U1 + U2 + Ineq + Prob,data=data)
```

We can see from the output above that, we started with all the factors in the data and got the **AIC = 514.65**. As we learnt in lecture week 3, models with lower AIC is preferred. The output at the start shows, if we remove the field So, the resulting model's AIC will be 512.65. Similarly, if we remove the field LF, the resulting model's AIC will be 512.96. Going with lower AIC, field 'So' was removed from the model. In the second step, 'Time' column was ordered as the field, removing which will result in lowest AIC. These steps are repeated until we reach row for 'none' - which means, the model is at the lowest AIC number and no more changes are needed. The steps are documented below and you can see the **model at step7 has AIC > step 6's AIC**, which means the predictors at step 6 are the best set of predictors.

Step	Field Removed	Resulting AIC	Remaining Predictors
0	-	514.65	ALL
1	So	512.65	M + Ed + Po1 + Po2 + LF + M.F + Pop + NW + U1 + U2 + Wealth + Ineq + Prob + Time
2	Time	511.01	M + Ed + Po1 + Po2 + LF + M.F + Pop + NW + U1 + U2 + Wealth + Ineq + Prob
3	LF	509.37	M + Ed + Po1 + Po2 + M.F + Pop + NW + U1 + U2 + Wealth + Ineq + Prob
4	Po2	506.33	M + Ed + Po1 + M.F + Pop + U1 + U2 + Wealth + Ineq + Prob
5	Pop	505.07	M + Ed + Po1 + M.F + U1 + U2 + Wealth + Ineq + Prob
6	Wealth	503.93	M + Ed + Po1 + M.F + U1 + U2 + Ineq + Prob
7	M.F	505.16	M + Ed + Po1 + U1 + U2 + Ineq + Prob

Variable selection using backward elimination outcome: M + Ed + Po1 + M.F + U1 + U2 + Ineq + Prob

Forward selection

Lets run the forward step selection method on the step data. We start with just the intercept and then keep adding factors one by one. The starting AIC for the model with just intercept is **AIC=561.02**. The output shows the first field to be added is **Po1** and by adding that field, the overall AIC reduces to **AIC=532.94**.

```
cat("***** forward step regression ")
```

```
## ***** forward step regression
```

```
forward<-step(lm(Crime~1,data=data),direction = "forward",scope=~ Crime ~ M + So + Ed + Po1 + Po2 + LF +
```

```
## Start:  AIC=561.02
## Crime ~ 1
##
##          Df Sum of Sq     RSS    AIC
## + Po1     1   3253302 3627626 532.94
## + Po2     1   3058626 3822302 535.39
## + Wealth  1   1340152 5540775 552.84
## + Prob    1   1257075 5623853 553.54
## + Pop     1    783660 6097267 557.34
## + Ed      1    717146 6163781 557.85
## + M.F     1    314867 6566061 560.82
## <none>                6880928 561.02
## + LF      1    245446 6635482 561.32
## + Ineq    1    220530 6660397 561.49
## + U2      1    216354 6664573 561.52
## + Time    1    154545 6726383 561.96
## + So      1     56527 6824400 562.64
## + M       1     55084 6825844 562.65
## + U1      1     17533 6863395 562.90
## + NW      1      7312 6873615 562.97
##
## Step:  AIC=532.94
## Crime ~ Po1
##
##          Df Sum of Sq     RSS    AIC
## + Ineq    1    739819 2887807 524.22
## + M       1    616741 3010885 526.18
## + M.F     1    250522 3377104 531.57
## + NW      1    232434 3395192 531.82
## + So      1    219098 3408528 532.01
## + Wealth  1    180872 3446754 532.53
## <none>                3627626 532.94
## + Po2     1    146167 3481459 533.00
## + Prob    1     92278 3535348 533.72
## + LF      1     77479 3550147 533.92
## + Time    1     43185 3584441 534.37
## + U2      1     17848 3609778 534.70
## + Pop     1      5666 3621959 534.86
```

```
## + U1      1      2878 3624748 534.90
## + Ed      1       767 3626859 534.93
##
## Step:  AIC=524.22
## Crime ~ Po1 + Ineq
##
##          Df Sum of Sq     RSS    AIC
## + Ed      1    587050 2300757 515.53
## + M.F     1    454545 2433262 518.17
## + Prob    1    280690 2607117 521.41
## + LF      1    260571 2627236 521.77
## + Wealth  1    213937 2673871 522.60
## + M       1    181236 2706571 523.17
## + Pop     1    130377 2757430 524.04
## <none>                2887807 524.22
## + NW      1     36439 2851369 525.62
## + So      1     33738 2854069 525.66
## + Po2     1     30673 2857134 525.71
## + U1      1      2309 2885498 526.18
## + Time    1       497 2887310 526.21
## + U2      1       253 2887554 526.21
##
## Step:  AIC=515.53
## Crime ~ Po1 + Ineq + Ed
##
##          Df Sum of Sq     RSS    AIC
## + M       1    239405 2061353 512.37
## + Prob    1    234981 2065776 512.47
## + M.F     1    117026 2183731 515.08
## <none>                2300757 515.53
## + Wealth  1     79540 2221218 515.88
## + U2      1     62112 2238646 516.25
## + Time    1     61770 2238987 516.26
## + Po2     1     42584 2258174 516.66
## + Pop     1     39319 2261438 516.72
## + U1      1      7365 2293392 517.38
## + LF      1      7254 2293503 517.39
## + NW      1      4210 2296547 517.45
## + So      1      4135 2296622 517.45
##
## Step:  AIC=512.37
## Crime ~ Po1 + Ineq + Ed + M
##
##          Df Sum of Sq     RSS    AIC
## + Prob    1    258063 1803290 508.08
## + U2      1    200988 1860365 509.55
## + Wealth  1    163378 1897975 510.49
## <none>                2061353 512.37
## + M.F     1     74398 1986955 512.64
## + U1      1     50835 2010518 513.20
## + Po2     1     45392 2015961 513.32
## + Time    1     42746 2018607 513.39
## + NW      1     16488 2044865 513.99
## + Pop     1      8101 2053251 514.19
```

```
## + So      1      3189 2058164 514.30
## + LF      1      2988 2058365 514.30
##
## Step:  AIC=508.08
## Crime ~ Po1 + Ineq + Ed + M + Prob
##
##          Df Sum of Sq     RSS    AIC
## + U2      1    192233 1611057 504.79
## + Wealth  1     86490 1716801 507.77
## + M.F     1     84509 1718781 507.83
## <none>                1803290 508.08
## + U1      1     52313 1750977 508.70
## + Pop     1     47719 1755571 508.82
## + Po2     1     37967 1765323 509.08
## + So      1     21971 1781320 509.51
## + Time    1     10194 1793096 509.82
## + LF      1       990 1802301 510.06
## + NW      1       797 1802493 510.06
##
## Step:  AIC=504.79
## Crime ~ Po1 + Ineq + Ed + M + Prob + U2
##
##          Df Sum of Sq     RSS    AIC
## <none>                1611057 504.79
## + Wealth  1     59910 1551147 505.00
## + U1      1     54830 1556227 505.16
## + Pop     1     51320 1559737 505.26
## + M.F     1     30945 1580112 505.87
## + Po2     1     25017 1586040 506.05
## + So      1     17958 1593098 506.26
## + LF      1     13179 1597878 506.40
## + Time    1      7159 1603898 506.58
## + NW      1       359 1610698 506.78
```

```
forwardModel<-lm(Crime~Po1 + Ineq + Ed + M + Prob + U2,data=data)
```

I have created the similar table for forward selection as well. You can see the AIC criterion increases at step 7 , so the predictors at step 6 are the best set of predictors using this approach.

Step	Field Added	Resulting AIC	Predictors in the model
0	-	561.02	-
1	Po1	532.94	Po1
2	Ineq	524.22	Po1 + Ineq
3	Ed	515.53	Po1 + Ineq + Ed
4	M	512.37	Po1 + Ineq + Ed + M
5	Prob	508.08	Po1 + Ineq + Ed + M + Prob
6	U2	504.79	Po1 + Ineq + Ed + M + Prob + U2
7	Wealth	505.00	Po1 + Ineq + Ed + M + Prob + U2 + Wealth

Variable selection using backward elimination outcome: Po1 + Ineq + Ed + M + Prob + U2

Stepwise regression:

Now lets run the stepwise regression with **direction=both**. This starts similar to forward selection with only intercept and starting AIC criterion as **AIC=561.02**. In the next step, adding field Po1 to the model, brings down the AIC to **AIC=532.94**. In the second step, we see that adding Ineq to the model results in **AIC=524.22** but removing Po1 field results in **AIC=561.02**. So there is nothing to be removed here and we can continue to next step.

```
cat("**** step regression-both ")
```

```
## **** step regression-both
```

```
both<-step(lm(Crime~1,data=data),direction = "both",scope=~ Crime ~ M + So + Ed + Po1 + Po2 + LF + M.F
```

```
## Start:  AIC=561.02
## Crime ~ 1
##
##          Df Sum of Sq     RSS    AIC
## + Po1     1   3253302 3627626 532.94
## + Po2     1   3058626 3822302 535.39
## + Wealth  1   1340152 5540775 552.84
## + Prob    1   1257075 5623853 553.54
## + Pop     1    783660 6097267 557.34
## + Ed      1    717146 6163781 557.85
## + M.F     1    314867 6566061 560.82
## <none>                6880928 561.02
## + LF      1    245446 6635482 561.32
## + Ineq    1    220530 6660397 561.49
## + U2      1    216354 6664573 561.52
## + Time    1    154545 6726383 561.96
## + So      1     56527 6824400 562.64
## + M       1     55084 6825844 562.65
## + U1      1     17533 6863395 562.90
## + NW      1      7312 6873615 562.97
##
## Step:  AIC=532.94
## Crime ~ Po1
##
##          Df Sum of Sq     RSS    AIC
## + Ineq    1    739819 2887807 524.22
## + M       1    616741 3010885 526.18
## + M.F     1    250522 3377104 531.57
## + NW      1    232434 3395192 531.82
## + So      1    219098 3408528 532.01
## + Wealth  1    180872 3446754 532.53
## <none>                3627626 532.94
## + Po2     1    146167 3481459 533.00
## + Prob    1     92278 3535348 533.72
## + LF      1     77479 3550147 533.92
## + Time    1     43185 3584441 534.37
## + U2      1     17848 3609778 534.70
## + Pop     1      5666 3621959 534.86
## + U1      1      2878 3624748 534.90
## + Ed      1       767 3626859 534.93
## - Po1     1   3253302 6880928 561.02
##
```

```
## Step:  AIC=524.22
## Crime ~ Po1 + Ineq
##
##          Df Sum of Sq     RSS    AIC
## + Ed      1    587050 2300757 515.53
## + M.F     1    454545 2433262 518.17
## + Prob    1    280690 2607117 521.41
## + LF      1    260571 2627236 521.77
## + Wealth  1    213937 2673871 522.60
## + M       1    181236 2706571 523.17
## + Pop     1    130377 2757430 524.04
## <none>                2887807 524.22
## + NW      1     36439 2851369 525.62
## + So      1     33738 2854069 525.66
## + Po2     1     30673 2857134 525.71
## + U1      1      2309 2885498 526.18
## + Time    1       497 2887310 526.21
## + U2      1       253 2887554 526.21
## - Ineq    1    739819 3627626 532.94
## - Po1     1   3772590 6660397 561.49
##
## Step:  AIC=515.53
## Crime ~ Po1 + Ineq + Ed
##
##          Df Sum of Sq     RSS    AIC
## + M       1    239405 2061353 512.37
## + Prob    1    234981 2065776 512.47
## + M.F     1    117026 2183731 515.08
## <none>                2300757 515.53
## + Wealth  1     79540 2221218 515.88
## + U2      1     62112 2238646 516.25
## + Time    1     61770 2238987 516.26
## + Po2     1     42584 2258174 516.66
## + Pop     1     39319 2261438 516.72
## + U1      1      7365 2293392 517.38
## + LF      1      7254 2293503 517.39
## + NW      1      4210 2296547 517.45
## + So      1      4135 2296622 517.45
## - Ed      1    587050 2887807 524.22
## - Ineq    1   1326101 3626859 534.93
## - Po1     1   3782666 6083423 559.23
##
## Step:  AIC=512.37
## Crime ~ Po1 + Ineq + Ed + M
##
##          Df Sum of Sq     RSS    AIC
## + Prob    1    258063 1803290 508.08
## + U2      1    200988 1860365 509.55
## + Wealth  1    163378 1897975 510.49
## <none>                2061353 512.37
## + M.F     1     74398 1986955 512.64
## + U1      1     50835 2010518 513.20
## + Po2     1     45392 2015961 513.32
## + Time    1     42746 2018607 513.39
```

```
## + NW      1     16488 2044865 513.99
## + Pop     1      8101 2053251 514.19
## + So      1      3189 2058164 514.30
## + LF      1      2988 2058365 514.30
## - M       1    239405 2300757 515.53
## - Ed      1    645219 2706571 523.17
## - Ineq    1    864671 2926024 526.83
## - Po1     1   4000849 6062202 561.07
##
## Step:  AIC=508.08
## Crime ~ Po1 + Ineq + Ed + M + Prob
##
##          Df Sum of Sq     RSS    AIC
## + U2      1    192233 1611057 504.79
## + Wealth  1     86490 1716801 507.77
## + M.F     1     84509 1718781 507.83
## <none>                1803290 508.08
## + U1      1     52313 1750977 508.70
## + Pop     1     47719 1755571 508.82
## + Po2     1     37967 1765323 509.08
## + So      1     21971 1781320 509.51
## + Time    1     10194 1793096 509.82
## + LF      1       990 1802301 510.06
## + NW      1       797 1802493 510.06
## - Prob    1    258063 2061353 512.37
## - M       1    262486 2065776 512.47
## - Ed      1    598315 2401605 519.55
## - Ineq    1    968199 2771489 526.28
## - Po1     1   3268577 5071868 554.69
##
## Step:  AIC=504.79
## Crime ~ Po1 + Ineq + Ed + M + Prob + U2
##
##          Df Sum of Sq     RSS    AIC
## <none>                1611057 504.79
## + Wealth  1     59910 1551147 505.00
## + U1      1     54830 1556227 505.16
## + Pop     1     51320 1559737 505.26
## + M.F     1     30945 1580112 505.87
## + Po2     1     25017 1586040 506.05
## + So      1     17958 1593098 506.26
## + LF      1     13179 1597878 506.40
## + Time    1      7159 1603898 506.58
## + NW      1       359 1610698 506.78
## - U2      1    192233 1803290 508.08
## - Prob    1    249308 1860365 509.55
## - M       1    400611 2011667 513.22
## - Ed      1    776207 2387264 521.27
## - Ineq    1    949221 2560278 524.56
## - Po1     1   2817067 4428124 550.31
```

```
both
```

```
##
## Call:
```

```
## lm(formula = Crime ~ Po1 + Ineq + Ed + M + Prob + U2, data = data)
##
## Coefficients:
## (Intercept)          Po1         Ineq           Ed            M         Prob
##    -5040.50       115.02        67.65       196.47       105.02     -3801.84
##          U2
##       89.37
```

I have gathered the information for stepwise regression below. The path followed is exactly same as forward selection for our data.

Step	Field Added	Field Removed	Resulting AIC	Predictors in the model
0	-	-	561.02	-
1	Po1	-	532.94	Po1
2	Ineq	-	524.22	Po1 + Ineq
3	Ed	-	515.53	Po1 + Ineq + Ed
4	M	-	512.37	Po1 + Ineq + Ed + M
5	Prob	-	508.08	Po1 + Ineq + Ed + M + Prob
5	U2	-	504.79	Po1 + Ineq + Ed + M + Prob + U2
6	Wealth	-	505.00	Po1 + Ineq + Ed + M + Prob + U2 + Wealth

Variable selection using stepwise method: Po1 + Ineq + Ed + M + Prob + U2

We can run a compare of the linear regression models generated by using predictors listed by backward and forward selection methods. We can see the adjusetd R-Squared for both the models are almost same. I would go with factors suggested by forward selection or stepwise regression(both gave same results) as the number of predictors is less. This makes model simpler compared to the predictors suggested by backward elimination.

```r
smr<-summary(backwardModel)
r2BackwardModel<-summary(backwardModel)$r.squared
adjustedR2BackwardModel<-summary(backwardModel)$adj.r.squared
cat("R-Squared for model using predictors from backward elimination method:",r2BackwardModel)
```

```
## R-Squared for model using predictors from backward elimination method: 0.7888268
```

```r
cat("Adjusted R-Squared for model using predictors from backward elimination method:",adjustedR2Backwar
```

```
## Adjusted R-Squared for model using predictors from backward elimination method: 0.7443692
```

```r
r2ForwardModel<-summary(forwardModel)$r.squared
adjustedR2ForwardModel<-summary(forwardModel)$adj.r.squared
cat("R-Squared for model using predictors from step wise regression method:",r2ForwardModel)
```

```
## R-Squared for model using predictors from step wise regression method: 0.7658663
```

```r
cat("Adjusted R-Squared for model using predictors from step wise regression method:",adjustedR2Forward
```

```
## Adjusted R-Squared for model using predictors from step wise regression method: 0.7307463
```

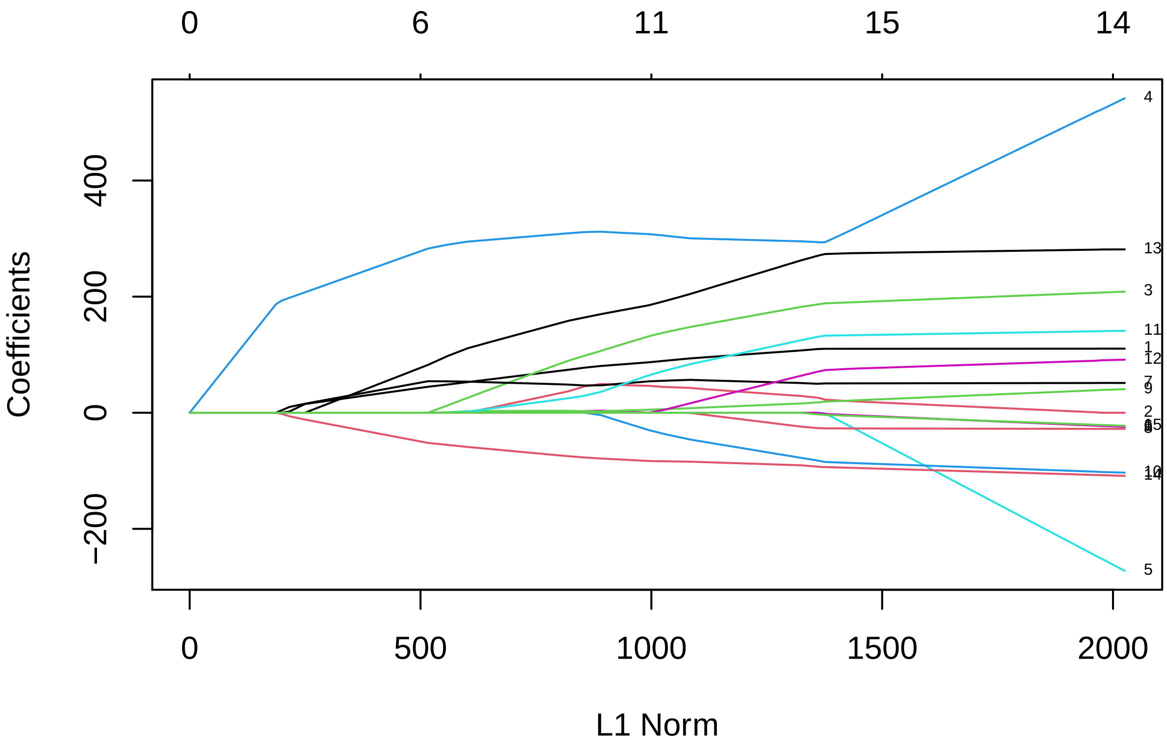
Lasso

As mentioned in the hw notes, I am using glmnet function for Lasso method to identify significant number of parameters. As per the documentation of glmnet function, and relating it to the lecture video, glmnet solves for the equation: $\alpha \sum_{i=1}^{j} |ai| + (1-\alpha) \sum_{i=1}^{j} a_i^2$. If $\alpha = 1$, this results in the equation $\sum_{i=1}^{j} |ai|$ which is a lasso equation. If $\alpha = 0$, this results in the equation $\sum_{i=1}^{j} a_i^2$ which is the equation for ridge regression. If $0 < \alpha < 1$, it will be a combination of both penalties which is the equation for Elastic net. First we have to normalize the data in the file except 'So' column which is already a factor on 0 and 1. Then I have run glm() function with $\alpha = 1$.

```
set.seed(777)

normalize <- function(dat, columns) {
  # clone copy of the input
  result <- dat
  # for each column that needs normaliztion
  for (col in columns) {
    # calculate mean for each column
    mu <- mean(dat[,col])
    # calculate std dev for each column
    sigma <- sd(dat[,col])
    result[,col] <- sapply(result[,col], function(x) (x - mu) / sigma)
  }
  return(result)
}


#columns that needs normalization, everything except column -'So'
colNames <- colnames(data[,-2])[1:14]
#normalizing the data
normalizedData <- normalize(data, colNames)
#running glmnet function
m<-glmnet(x=as.matrix(normalizedData[,-16]),
          y=as.matrix(normalizedData$Crime),
          alpha=1,
          standardize = TRUE)
#plot to show significance of factors in the model.
plot(m, label = TRUE)
```



When we plot this model, we will see how the predictors are influencing the outcome. On the x-axis the different values of lambda are shown. Each line in the graph represent one of the predictors and its role in the model. In the plots we can see when each variable entered in the model and to which extent they influenced the response variable. Analysing the plot we can say the col 4(Po1, blur line going upwards) influences the model most. It enters the model first and steadily positvely effects the response variable. Similarly col 14(Prob, pink line going downwards) enters next in the model and affects negatively on the response variable.

To select the right lambda value we can run cv.glmnet(), which runs cross validation to determine different models and different lambda values. I have calculated the R-Squared of the model with lambda = 1se, which gives the most regularized model such that error is within one standard error of the minimum. In the MSE plot below, the 2 dotted lines are for lambda= min and lambda=1se. For 1se line you can see the spread of the prediction is narrow compared to lambda=min. With this model, the signficant parameters are **M,Po1,M.F,Ineq,Prob.**

```
set.seed(777)
#cv.glmnet to run 10 fold CV
lasso=cv.glmnet(x=as.matrix(normalizedData[,-16]),
                y=as.matrix(normalizedData$Crime),
                alpha=1,
                nfolds = 10,
                type.measure="mse",
                family="gaussian")
lasso$lambda.1se
```

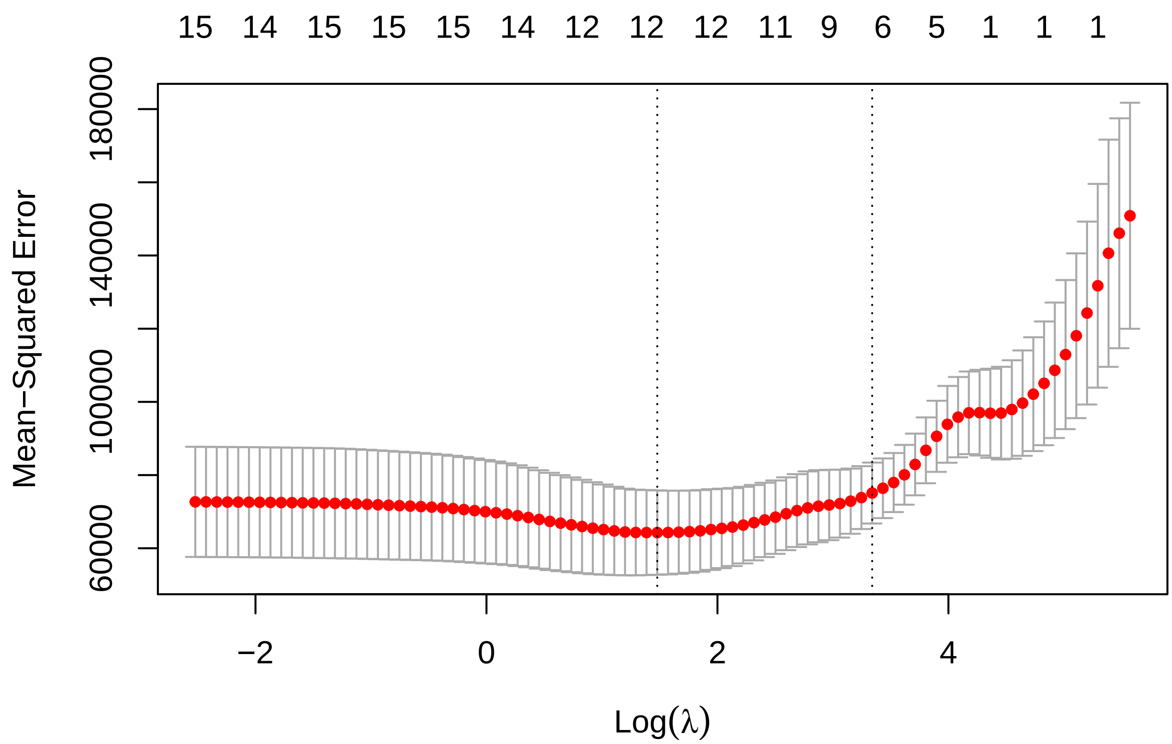
```
## [1] 28.21086
```

```
#co-efficients filtered with 1se
coeffs<-coef(lasso, s=lasso$lambda.1se)
coeffs
```

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                         1
```

```
## (Intercept) 905.0851064
## M             48.6326938
## So             .
## Ed            12.8572539
## Po1          289.6363150
## Po2            .
## LF             .
## M.F           54.0372427
## Pop            .
## NW             0.8228341
## U1             .
## U2             .
## Wealth         .
## Ineq          97.9391908
## Prob         -55.4515431
## Time           .
```

```
plot(lasso)
```



Elastic Net

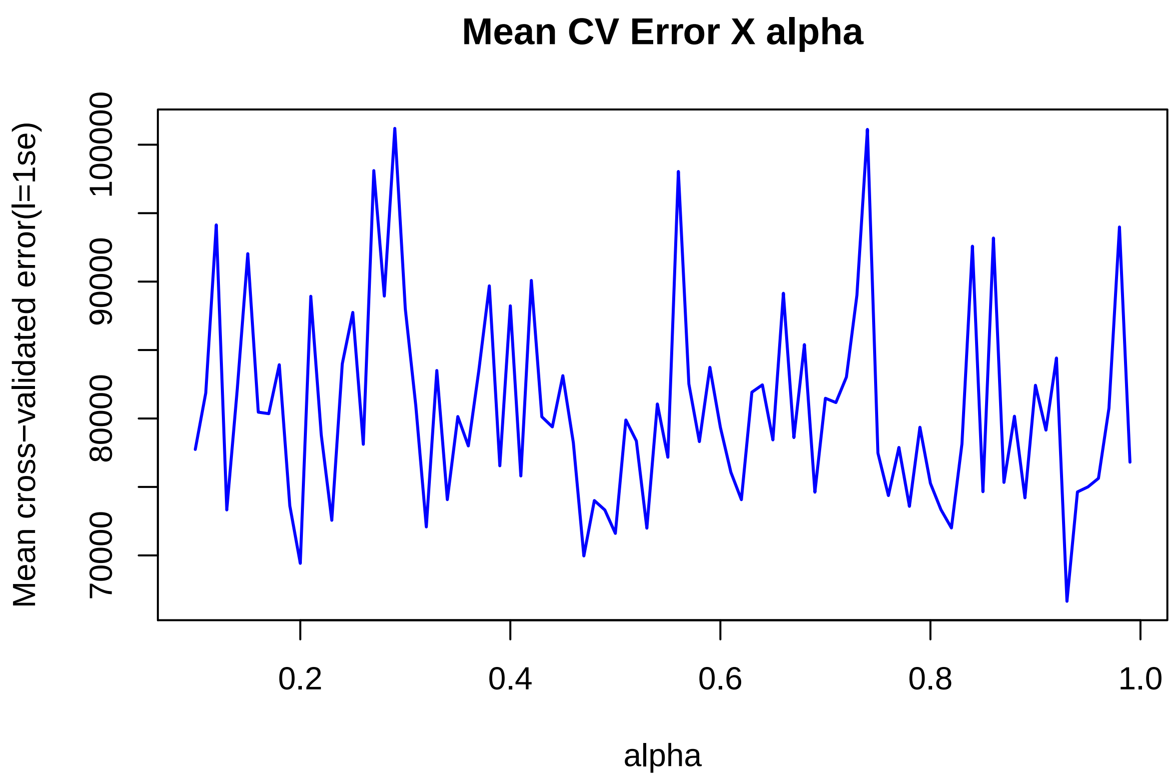
As explained above, changing the alpha value between 0,1 we can run different versions on elastic net models.I have run a loop for **0.1<alpha<0.99** and using cross validated error for each value of alpha as criteria to select the best alpha value. Lower the CVM value, better the model is. I employed this approach because the data points in the file were only 47 and there is not enough data to split it into training and testing data. Based on this approach we see that the best alpha = 0.93, the lowest point on the Mean CV Error X alpha chart.

```
set.seed(777)
enModels <-data.frame(alpha=numeric(), cvm=numeric())
```

```r
for(i in seq(0.1,.99,0.01)){
  enModel<-cv.glmnet(x=as.matrix(normalizedData[,-16]),
                  y=as.matrix(normalizedData$Crime),
                  alpha=i,
                  nfolds = 10,
                  type.measure="mse",
                  family="gaussian")
  #cross validation error for model whose lambda is equal to 1se
  cvm<-enModel$cvm[which(enModel$lambda==enModel$lambda.1se)]
  enModels[nrow(enModels) + 1,] = c(i,cvm)
}
#row with lowest CVM error
temp1<-enModels[which.min(enModels$cvm),]
#alpha value of the above record
temp1$alpha
```

```
## [1] 0.93
```

```r
plot(enModels$alpha, enModels$cvm, type="l",
     lwd=1.5, xlab="alpha", ylab="Mean cross-validated error(l=1se)",
     main="Mean CV Error X alpha",
     col=ifelse(enModels$alpha==temp1$alpha, "red","blue"))
```




```r
#cv.glmnet to run 10 fold CV and alpha = lowest CVM found above
enBestModel=cv.glmnet(x=as.matrix(normalizedData[,-16]),
                  y=as.matrix(normalizedData$Crime),
                  alpha=temp1$alpha,
                  nfolds = 10,
                  type.measure="mse",
                  family="gaussian")
```

```r
#coeffecients for lambda value of 1se when alpha = lowest CVM found above
coeffs1<-coef(enBestModel, s=enBestModel$lambda.1se)
coeffs1
```

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept) 905.085106
## M            40.632662
## So            .
## Ed            .
## Po1         270.536083
## Po2           .
## LF            .
## M.F          49.797829
## Pop           .
## NW            1.178023
## U1            .
## U2            .
## Wealth        .
## Ineq         70.414774
## Prob        -48.022408
## Time          .
```

```r
#significant paramters based on the coeffs
formula1=Crime~M+Po1+M.F+NW+Ineq+Prob



#function to calculate RSquared by performing loocv method.
loocv<-function(formula,data){
  SStot <- sum((data$Crime - mean(data$Crime))^2)
  totsse <- 0
  for(i in 1:nrow(data)) {
    model = lm(formula, data = data[-i,])
    pred_i <- predict(model,newdata=data[i,])
    totsse <- totsse + ((pred_i - data[i,16])^2)
  }
  R2_mod <- 1 - totsse/SStot
  return(R2_mod)
}


result1<-loocv(formula = formula1, data)
#R2 for model using predictors suggested  by elastic net when alpha = lowest CVM found above
result1
```

```
##         1
## 0.5878755
```

Stepwise, Lasso and Elastic Net on PCA models:

Lets take this analysis to next step by running the above analysis on Principal Components instead of the actual data. Now we have factors selected by all the 3 methods on data and on principal components. We can run loocv with all the models and see which has the best R2 value.

```r
#***** PCA ****
set.seed(777)
#running pca on the data with scale =true
pca <- prcomp(data[,1:15], scale. = TRUE)
#creating data set wiht pcs along with response variable
pcData <- as.data.frame(cbind(pca$x, data[,16]))
#adding the column name to the result column.
colnames(pcData)[16] <- "Crime"



bothPCA<-step(lm(Crime~1,data=pcData),direction = "both",scope=~ Crime ~ PC1 + PC2 + PC3  + PC4 + PC5 +
             PC7 + PC8 + PC9 + PC10 + PC11 + PC12 + PC13 + PC14+PC15 , trace = FALSE)
bothPCA
```

```
##
## Call:
## lm(formula = Crime ~ PC5 + PC1 + PC2 + PC12 + PC4 + PC7 + PC14 +
##     PC6 + PC15, data = pcData)
##
## Coefficients:
## (Intercept)          PC5          PC1          PC2         PC12          PC4
##      905.09      -229.04        65.22       -70.08       289.61        69.45
##         PC7         PC14          PC6         PC15
##      117.26       219.19       -60.21      -622.21
```

```r
#***** lasso****
lassoPCA=cv.glmnet(x=as.matrix(pcData[,-16]),
                y=as.matrix(pcData$Crime),
                alpha=1,
                nfolds = 10,
                type.measure="mse",
                family="gaussian")

#co-efficients filtered with 1se
coeffs<-coef(lassoPCA, s=lassoPCA$lambda.1se)
coeffs
```

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                        1
## (Intercept)  905.08511
## PC1           46.42112
## PC2          -42.53599
## PC3            .
## PC4           26.67432
## PC5         -181.93988
## PC6            .
## PC7           35.97418
## PC8            .
## PC9            .
## PC10           .
## PC11           .
## PC12         160.82656
## PC13           .
## PC14          28.49137
```

```
## PC15               .
```

```r
#**** Elastic Net with alpha value determined above****
#cv.glmnet to run 10 fold CV and alpha = lowest CVM found above
enPCA=cv.glmnet(x=as.matrix(pcData[,-16]),
                 y=as.matrix(pcData$Crime),
                 alpha=temp1$alpha,
                 nfolds = 10,
                 type.measure="mse",
                 family="gaussian")
#coeffecients for lambda value of 1se when alpha = lowest CVM found above
coeffs1<-coef(enPCA, s=enPCA$lambda.1se)
coeffs1
```

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                        1
## (Intercept)  905.085106
## PC1           51.940134
## PC2          -50.781858
## PC3            2.731584
## PC4           39.719477
## PC5         -195.368205
## PC6          -17.374761
## PC7           60.855161
## PC8            .
## PC9            .
## PC10           .
## PC11           .
## PC12         199.610975
## PC13           .
## PC14          87.208433
## PC15        -153.381962
```

Using all the data from above analysis, we can list the factors selected by each method and run loocv and determine the R2 value. As mentioned earlier, since the data points are only 47 of them, its not a good fit to split into training and validation set. Instead, am running loocv to validate how good the model is performing. Based on this analysis, for the given data, predictors suggested by stepwise regression method results in highest RSquared value.

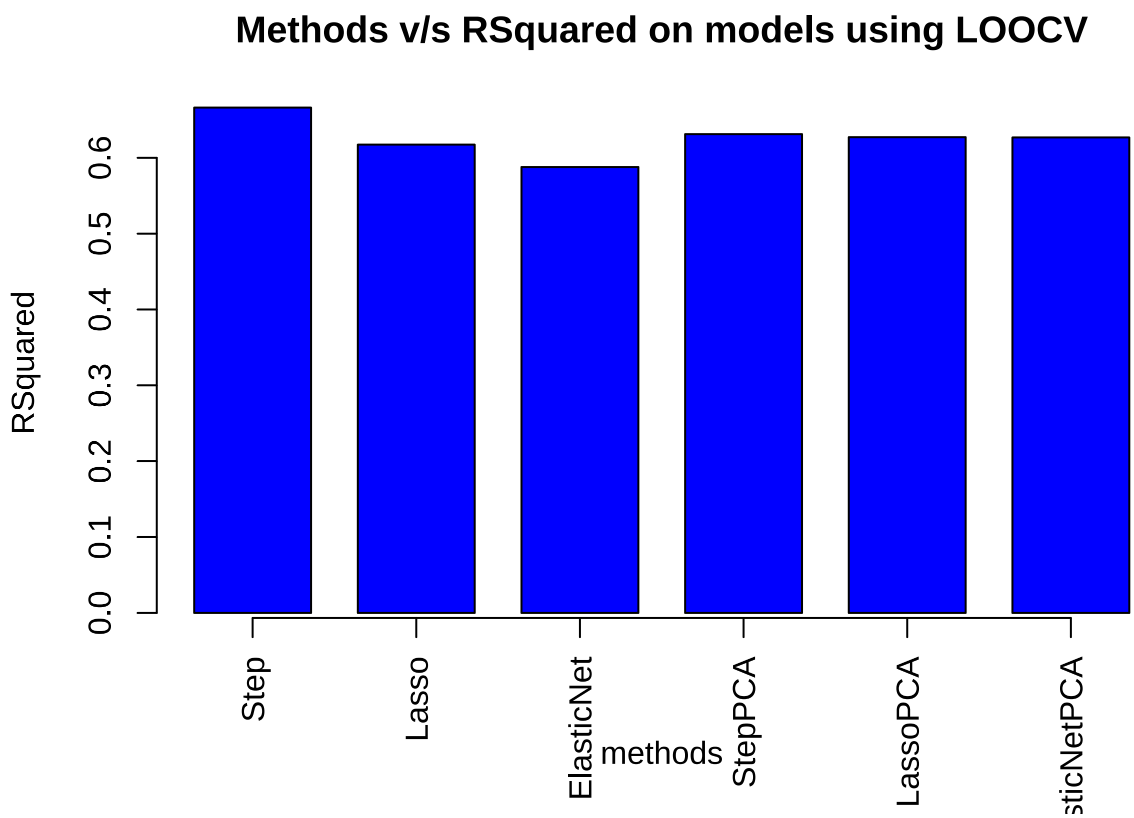
```r
#Creating formula objects using predictors suggested by all the methods above

formulaStepLR <-Crime~Po1 + Ineq + Ed + M + Prob + U2
formulaLassoLR <-Crime~M +Ed+ Po1 + M.F + NW+ Ineq + Prob
formulaENLR<-Crime~M+Po1+M.F+NW+Ineq+Prob
formulaStepPCA<-Crime~PC5+PC1+PC2+PC12+PC4+PC7+PC14+PC6+PC15
formulaLassoPCA<-Crime~PC1+PC2+PC4+PC5+PC7+PC12+PC14
formulaENPCA<-Crime~PC1+PC2+PC3+PC4+PC5+PC6+PC7+PC12+PC14+PC15

#Calculating RSquared value for each of the above formulae
#Using the loocv function created above
r2StepLR=loocv(formulaStepLR,data)
r2LassoLR=loocv(formulaLassoLR,data)
r2EnLR=loocv(formulaENLR,data)
r2StepPCA=loocv(formulaStepPCA,pcData)
```

```r
r2LassoPCA=loocv(formulaLassoPCA,pcData)
r2EnPCA=loocv(formulaENPCA,pcData)

#creating a table with mehtods against their r2
methods<-c("Step","Lasso","ElasticNet","StepPCA","LassoPCA","ElasticNetPCA")
r2s<-c(r2StepLR,r2LassoLR,r2EnLR,r2StepPCA,r2LassoPCA,r2EnPCA)
result<-data.frame(r2s,methods)
barPlot <- barplot(height = result$r2s, names=result$methods, space=0.4, xaxt='n', xlab="methods", ylab=
                   main="Methods v/s RSquared on models using LOOCV", col = "blue")
axis(1,las=2, at=barPlot, labels=methods)
```




```r
result
```

```
##         r2s       methods
## 1 0.6661638          Step
## 2 0.6173366         Lasso
## 3 0.5878755    ElasticNet
## 4 0.6311924       StepPCA
## 5 0.6271988      LassoPCA
## 6 0.6267523 ElasticNetPCA
```

Question 12.1

Describe a situation or problem from your job, everyday life, current events, etc., for which a design of experiments approach would be appropriate.

Answer:

I am working as a developer at one of the mortgage insurance companies. My team are responsible for all the customer facing apps like our company websites, mobile apps, Alexa skills , google assistants and numerous chat bots. One of the constant challenges in UI/UX world is to determine how best to present something to a customer that will keep customer more engaged in all the channels. Example: when there is a new loan product to be launched, designing UI/UX for collecting data. Customers should input all the required data for loan processing with least hassle. Few of the questions we want to address are:

- Do we have to put a input box for a user to enter loan amount or provide pre populated tabs(pills) with commonly used loan amounts?
- Should we provide check box to select gender or provide a sliding tab?
- Should all the data gathering be in one page or split into multiple pages?
- On alexa skill, should we provide prompt guide for user or trust user to provide the right input?
- On chatbot, a functionality that needs lot of inputs is good fit for chatbot application?

A common practice in UI/UX world is to perform interviews with subset of target user group using a mocked prototype of the application. We select the group of people for interview with good mix representation for most demographics like age, role of current job, current work location, nature of work (desk job vs on the road),etc. As mentioned in the lecture, we also have comparison and control checks in place during the whole process. When we are comparing options for 2 approaches, we have to make sure both the options are relatable with similar number of inputs, similar number of screens.

There is a interesting branch in UI/UX study about interviews and A/B testing. All the modern application hosting providers like (AWS, Kubernetes, GCP) all provide AB testing capabilities out of the box. We host 2 versions of our website at the same time. The incoming traffic is controlled to split it between the 2 versions(usually 50-50 but can be controlled).We let it run for a month or so and collect the usage data. Based on the data, if we prefer 'A' version of website better than 'B' version, we kill the 'B' pods and divert all the traffic to "A" pods.

If you are interested you can refer to this quick read about AB Testing in UI/UX: https://usabilitygeek.com/a-b-testing-optimizing-the-ux/

Question 12.2

As mentioned in the HW, I am using the FrF2 function in R to run fractional factorial 2 level designs. I have given 10 factors as F1,F2,F3…F10. We can give proper names like wooden flooring, yard, patio and so on, but for the sake of hw I have selected F1-F10. If we have 10 factors, to get all combinations of 10 factors, we have to run 2^10 runs. Instead if we have 16 runs available, this function will gives the maximum number of combinations we can cover in 16 runs. Looking at the result, we get the mix of houses that needs to be shown to customers. Like house number 1 should have features F1,F2,F3,F8 and F10, but should not have rest. Similarlly House 2 should have F3,F4,F6,F7,F8., and so on.

```r
#cleaning environment and starting fresh.
rm(list = ls())
#setting seed for consistent results
set.seed(1)
library(FrF2)
```

```
## Loading required package: DoE.base

## Loading required package: grid

## Loading required package: conf.design

## Registered S3 method overwritten by 'partitions':
##   method            from
##   print.equivalence lava
```

```
## Registered S3 method overwritten by 'DoE.base':
##   method           from
##   factorize.factor conf.design

##
## Attaching package: 'DoE.base'

## The following objects are masked from 'package:stats':
##
##     aov, lm

## The following object is masked from 'package:graphics':
##
##     plot.design

## The following object is masked from 'package:base':
##
##     lengths
```

```r
result<-FrF2(16,10,factor.names=c('F1','F2','F3','F4','F5','F6','F7','F8','F9','F10'),
     default.levels=c('Yes','No'))
result
```

```
##     F1  F2  F3  F4  F5  F6  F7  F8  F9 F10
## 1  Yes Yes Yes  No  No  No  No Yes  No Yes
## 2   No  No Yes Yes  No Yes Yes Yes  No  No
## 3  Yes  No  No Yes Yes Yes  No  No Yes  No
## 4  Yes Yes Yes Yes  No  No  No  No Yes  No
## 5   No Yes Yes Yes Yes Yes  No Yes Yes Yes
## 6   No Yes  No  No Yes  No Yes  No Yes Yes
## 7   No  No Yes  No  No Yes Yes  No Yes Yes
## 8  Yes  No Yes Yes Yes  No Yes  No  No Yes
## 9  Yes Yes  No  No  No Yes Yes Yes Yes  No
## 10 Yes Yes  No Yes  No Yes Yes  No  No Yes
## 11 Yes  No Yes  No Yes  No Yes Yes Yes  No
## 12  No Yes Yes  No Yes Yes  No  No  No  No
## 13  No Yes  No Yes Yes  No Yes Yes  No  No
## 14 Yes  No  No  No Yes Yes  No Yes  No Yes
## 15  No  No  No  No  No  No  No  No  No  No
## 16  No  No  No Yes  No  No  No Yes Yes Yes
## class=design, type= FrF2
```

Question 13.1

a. Binomial A good example of Binomial distribution is rolling a 7 in a game of craps. In a game of craps at a casino each player rolls 2 dices looking for a favourable output of 7. With 2 dices, there are 6 different ways of getting 7 (4-3,3-4,5-2,2-5,1-6,6-1). Probability of getting 7 with dices is $6/16 = 16.6\%$. If we plot distribution of the probablity of rolling 7 with 2 dices for 50 throws follows binomial distribution.

b. Geometric Extending the previous example, If we want to determine the number of times we have to roll dice to get before we hit 7, that will follow geometric distribution. As discussed above, the probability of getting 7 is 0.16, if we want to find the probability of rolling 7 in first 10 tries can be calculated with the mass function discussed in the lecture :

- $P(X = x) = (1 - p)^x p$
- $P(X) = (1 - 0.16)^{10-1} * 0.16$
- $P(X) = 3.3\%$

c. Poisson In a FIFA World cup, avg goals scored per game is 2.5 goals.Modelling this to determine the probablility of 'k' number of goals scored in a game follows poisson distribution. In this example, $\lambda = 2.5$, applying the mass function as discussed in the lecture $f_x(X) = \lambda^x e^{-\lambda}/x!$ probability of no goals scored in the game:

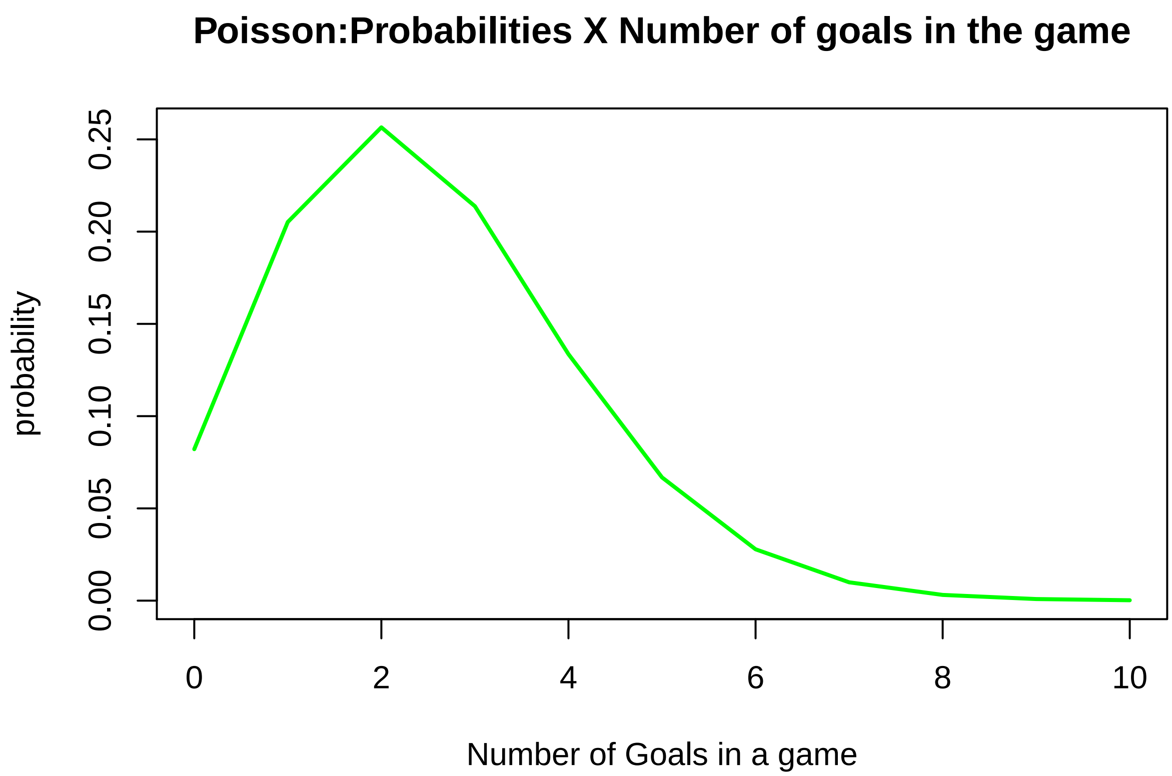
- $f(X = 0) = 2.5^0 e^{-2.5}/0! = 8.2\%$
- $f(X = 1) = 2.5^1 e^{-2.5}/1! = 20.5\%$
- $f(X = 2) = 2.5^2 e^{-2.5}/2! = 25.6\%$
- $f(X = 3) = 2.5^3 e^{-2.5}/3! = 21.3\%$
- $f(X = 4) = 2.5^4 e^{-2.5}/4! = 13.36\%$

Simple R-prog to plot the above data:

```
lambda=2.5
probabilitiesOfGoals <-data.frame(p=numeric(), numberOfGoalsInAGame=numeric())
for(g in seq(0,10)){
  p<-(lambda^g*exp(1)^-lambda)/ factorial(g)
  probabilitiesOfGoals[nrow(probabilitiesOfGoals) + 1,] = c(p,g)
}
probabilitiesOfGoals
```

```
##                p numberOfGoalsInAGame
## 1  0.0820849986                    0
## 2  0.2052124966                    1
## 3  0.2565156207                    2
## 4  0.2137630172                    3
## 5  0.1336018858                    4
## 6  0.0668009429                    5
## 7  0.0278337262                    6
## 8  0.0099406165                    7
## 9  0.0031064427                    8
## 10 0.0008629007                    9
## 11 0.0002157252                   10
```

```
plot(probabilitiesOfGoals$numberOfGoalsInAGame, probabilitiesOfGoals$p, type="l", col="green", lwd=2, xl
```



d. Exponential

Extending the above example, the time scored between each goal will follow exponential distribution.

e. Weibull Good example for weibull distribution is explain the frequency of cellphone charging. When the phone is new, battery is new it holds charge for longer duration. As phone gets older, the amount of charge retention goes down and have keep charging the phone more frequently. This is a good example of $K>1$ as explained in the lecture video.