GT Introduction to Analytics Modeling - Week 5 HW

Robert Phillips

June 17, 2017

Question 1

Using the crime data set from Homework 3, build a regression model using:

1. Stepwise regression
2. Lasso
3. Elastic net

For Parts 2 and 3, remember to scale the data first - otherwise, the regression coefficients will be on different scales and the constraint won't have the desired effect.

We first load the data and create a matrix of the scaled predictors.

```
crimes = read.csv('uscrime.txt', header=T, sep='\t')
crimes.m = scale(as.matrix(crimes[,1:15]))

str(crimes)
```

```
## 'data.frame':    47 obs. of  16 variables:
##  $ M     : num  15.1 14.3 14.2 13.6 14.1 12.1 12.7 13.1 15.7 14 ...
##  $ So    : int  1 0 1 0 0 0 1 1 1 0 ...
##  $ Ed    : num  9.1 11.3 8.9 12.1 12.1 11 11.1 10.9 9 11.8 ...
##  $ Po1   : num  5.8 10.3 4.5 14.9 10.9 11.8 8.2 11.5 6.5 7.1 ...
##  $ Po2   : num  5.6 9.5 4.4 14.1 10.1 11.5 7.9 10.9 6.2 6.8 ...
##  $ LF    : num  0.51 0.583 0.533 0.577 0.591 0.547 0.519 0.542 0.553 0.632 ...
##  $ M.F   : num  95 101.2 96.9 99.4 98.5 ...
##  $ Pop   : int  33 13 18 157 18 25 4 50 39 7 ...
##  $ NW    : num  30.1 10.2 21.9 8 3 4.4 13.9 17.9 28.6 1.5 ...
##  $ U1    : num  0.108 0.096 0.094 0.102 0.091 0.084 0.097 0.079 0.081 0.1 ...
##  $ U2    : num  4.1 3.6 3.3 3.9 2 2.9 3.8 3.5 2.8 2.4 ...
##  $ Wealth: int  3940 5570 3180 6730 5780 6890 6200 4720 4210 5260 ...
##  $ Ineq  : num  26.1 19.4 25 16.7 17.4 12.6 16.8 20.6 23.9 17.4 ...
##  $ Prob  : num  0.0846 0.0296 0.0834 0.0158 0.0414 ...
##  $ Time  : num  26.2 25.3 24.3 29.9 21.3 ...
##  $ Crime : int  791 1635 578 1969 1234 682 963 1555 856 705 ...
```

Part 1 - Stepwise regression

For stepswise regression, we'll start with the full model and then use the *step* function to select the best predictors.

```
crimes.fit = lm(Crime ~ ., data=crimes)
summary(crimes.fit)
```

```
##
## Call:
## lm(formula = Crime ~ ., data = crimes)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -395.74  -98.09   -6.69  112.99  512.67
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5.984e+03  1.628e+03  -3.675 0.000893 ***
## M            8.783e+01  4.171e+01   2.106 0.043443 *
## So          -3.803e+00  1.488e+02  -0.026 0.979765
## Ed           1.883e+02  6.209e+01   3.033 0.004861 **
## Po1          1.928e+02  1.061e+02   1.817 0.078892 .
## Po2         -1.094e+02  1.175e+02  -0.931 0.358830
## LF          -6.638e+02  1.470e+03  -0.452 0.654654
## M.F          1.741e+01  2.035e+01   0.855 0.398995
## Pop         -7.330e-01  1.290e+00  -0.568 0.573845
## NW           4.204e+00  6.481e+00   0.649 0.521279
## U1          -5.827e+03  4.210e+03  -1.384 0.176238
## U2           1.678e+02  8.234e+01   2.038 0.050161 .
## Wealth       9.617e-02  1.037e-01   0.928 0.360754
## Ineq         7.067e+01  2.272e+01   3.111 0.003983 **
## Prob        -4.855e+03  2.272e+03  -2.137 0.040627 *
## Time        -3.479e+00  7.165e+00  -0.486 0.630708
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 209.1 on 31 degrees of freedom
## Multiple R-squared:  0.8031, Adjusted R-squared:  0.7078
## F-statistic: 8.429 on 15 and 31 DF,  p-value: 3.539e-07
```

Using all predictors indicates that most factors are not signficant. Let's see what a stepwise approach yeilds.

```r
crimes.fit.step = step(crimes.fit)
```

```
## 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
```

The resulting model is:

```r
crimes.fit.step$coefficients
```

```
## (Intercept)           M          Ed         Po1         M.F          U1
## -6426.10102    93.32155   180.12011   102.65316    22.33975 -6086.63315
##          U2        Ineq        Prob
##   187.34512    61.33494 -3796.03183
```

For part 2 and 3, we'll use the *glmnet* package as suggested by the assigment.

```r
require(glmnet)
```

Part 2 - Lasso

According to the *glmnet* documentation, for family="gaussian" this is the lasso sequence if alpha=1, else it is the elasticnet sequence. Therefore we'll set alpha=1 for this part.

```r
crimes.fit.lasso = glmnet(crimes.m, crimes$Crime, family="gaussian", alpha=1)
print(crimes.fit.lasso)
```

```
##
## Call:  glmnet(x = crimes.m, y = crimes$Crime, family = "gaussian", alpha = 1)
##
##        Df    %Dev    Lambda
##   [1,]  0 0.00000 263.10000
##   [2,]  1 0.08027 239.70000
##   [3,]  1 0.14690 218.40000
##   [4,]  1 0.20220 199.00000
##   [5,]  1 0.24820 181.30000
##   [6,]  1 0.28630 165.20000
##   [7,]  1 0.31800 150.60000
##   [8,]  1 0.34430 137.20000
##   [9,]  1 0.36610 125.00000
## [10,]  1 0.38420 113.90000
## [11,]  1 0.39920 103.80000
## [12,]  1 0.41170  94.55000
## [13,]  1 0.42210  86.15000
## [14,]  1 0.43070  78.50000
## [15,]  3 0.44240  71.52000
## [16,]  4 0.45870  65.17000
## [17,]  4 0.48700  59.38000
```

```
## [18,]  5 0.52490  54.11000
## [19,]  5 0.55650  49.30000
## [20,]  5 0.58260  44.92000
## [21,]  5 0.60430  40.93000
## [22,]  5 0.62240  37.29000
## [23,]  5 0.63730  33.98000
## [24,]  6 0.64980  30.96000
## [25,]  7 0.66700  28.21000
## [26,]  9 0.68240  25.70000
## [27,]  9 0.69830  23.42000
## [28,]  9 0.71130  21.34000
## [29,]  9 0.72220  19.44000
## [30,]  9 0.73120  17.72000
## [31,] 10 0.73870  16.14000
## [32,] 10 0.74510  14.71000
## [33,] 11 0.75150  13.40000
## [34,] 11 0.75830  12.21000
## [35,] 11 0.76370  11.13000
## [36,] 11 0.76830  10.14000
## [37,] 11 0.77260   9.23800
## [38,] 11 0.77620   8.41700
## [39,] 12 0.77930   7.66900
## [40,] 12 0.78240   6.98800
## [41,] 12 0.78490   6.36700
## [42,] 12 0.78690   5.80200
## [43,] 12 0.78870   5.28600
## [44,] 12 0.79010   4.81700
## [45,] 12 0.79130   4.38900
## [46,] 12 0.79230   3.99900
## [47,] 12 0.79310   3.64400
## [48,] 12 0.79380   3.32000
## [49,] 12 0.79440   3.02500
## [50,] 12 0.79480   2.75600
## [51,] 12 0.79520   2.51100
## [52,] 12 0.79550   2.28800
## [53,] 13 0.79580   2.08500
## [54,] 13 0.79610   1.90000
## [55,] 13 0.79630   1.73100
## [56,] 13 0.79650   1.57700
## [57,] 14 0.79660   1.43700
## [58,] 14 0.79670   1.30900
## [59,] 15 0.79690   1.19300
## [60,] 15 0.79790   1.08700
## [61,] 15 0.79870   0.99050
## [62,] 15 0.79940   0.90250
## [63,] 15 0.80000   0.82240
## [64,] 15 0.80050   0.74930
## [65,] 15 0.80090   0.68270
## [66,] 15 0.80130   0.62210
## [67,] 15 0.80160   0.56680
## [68,] 15 0.80180   0.51650
## [69,] 15 0.80200   0.47060
## [70,] 15 0.80220   0.42880
## [71,] 15 0.80230   0.39070
```

```
## [72,] 15 0.80240   0.35600
## [73,] 15 0.80250   0.32440
## [74,] 15 0.80260   0.29550
## [75,] 15 0.80270   0.26930
## [76,] 15 0.80270   0.24540
## [77,] 15 0.80280   0.22360
## [78,] 15 0.80280   0.20370
## [79,] 15 0.80290   0.18560
## [80,] 15 0.80290   0.16910
## [81,] 14 0.80290   0.15410
## [82,] 14 0.80290   0.14040
## [83,] 14 0.80300   0.12790
## [84,] 14 0.80300   0.11660
## [85,] 14 0.80300   0.10620
## [86,] 14 0.80300   0.09678
## [87,] 14 0.80300   0.08818
## [88,] 15 0.80300   0.08035
```

We can see from the output that about 75% of the variance is explained when lambda is about 15. This yields the following model. We see that 6 factors are omitted. We can further reduce the number of factors by increasing lambda.

```r
coef(crimes.fit.lasso, s = 15, exact = F)
```

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                        1
## (Intercept) 905.085106
## M            76.898525
## So           20.911080
## Ed           96.439852
## Po1         310.789072
## Po2            .
## LF            2.400085
## M.F          47.266883
## Pop            .
## NW            2.777599
## U1             .
## U2           28.421331
## Wealth         .
## Ineq        162.963648
## Prob        -76.599753
## Time           .
```

Part 3 - Elastic net

```r
crimes.fit.elas = glmnet(crimes.m, crimes$Crime, family="gaussian", alpha=.5)
print(crimes.fit.elas)
```

```
##
## Call:  glmnet(x = crimes.m, y = crimes$Crime, family = "gaussian", alpha = 0.5)
##
##       Df    %Dev    Lambda
##  [1,]  0 0.00000 526.20000
##  [2,]  2 0.05157 479.40000
##  [3,]  2 0.10620 436.90000
```

```
##  [4,]  2 0.15450 398.00000
##  [5,]  2 0.19700 362.70000
##  [6,]  2 0.23420 330.50000
##  [7,]  2 0.26670 301.10000
##  [8,]  2 0.29500 274.40000
##  [9,]  2 0.31950 250.00000
## [10,]  2 0.34070 227.80000
## [11,]  2 0.35900 207.50000
## [12,]  2 0.37470 189.10000
## [13,]  3 0.39030 172.30000
## [14,]  3 0.40400 157.00000
## [15,]  4 0.41950 143.00000
## [16,]  4 0.43400 130.30000
## [17,]  6 0.45330 118.80000
## [18,]  6 0.48160 108.20000
## [19,]  7 0.51220  98.60000
## [20,]  7 0.53990  89.84000
## [21,]  7 0.56390  81.86000
## [22,]  7 0.58470  74.59000
## [23,]  7 0.60270  67.96000
## [24,]  7 0.61820  61.92000
## [25,]  9 0.63480  56.42000
## [26,] 11 0.65300  51.41000
## [27,] 11 0.67000  46.84000
## [28,] 11 0.68470  42.68000
## [29,] 11 0.69740  38.89000
## [30,] 11 0.70840  35.43000
## [31,] 11 0.71800  32.29000
## [32,] 11 0.72620  29.42000
## [33,] 12 0.73470  26.80000
## [34,] 12 0.74260  24.42000
## [35,] 12 0.74920  22.25000
## [36,] 13 0.75520  20.28000
## [37,] 13 0.76080  18.48000
## [38,] 13 0.76560  16.83000
## [39,] 13 0.76980  15.34000
## [40,] 13 0.77340  13.98000
## [41,] 13 0.77660  12.73000
## [42,] 14 0.77940  11.60000
## [43,] 13 0.78220  10.57000
## [44,] 13 0.78460   9.63300
## [45,] 13 0.78660   8.77700
## [46,] 13 0.78840   7.99800
## [47,] 13 0.78990   7.28700
## [48,] 12 0.79120   6.64000
## [49,] 12 0.79220   6.05000
## [50,] 12 0.79300   5.51200
## [51,] 12 0.79360   5.02300
## [52,] 12 0.79420   4.57700
## [53,] 13 0.79470   4.17000
## [54,] 13 0.79510   3.80000
## [55,] 13 0.79550   3.46200
## [56,] 13 0.79580   3.15400
## [57,] 13 0.79600   2.87400
```

```
## [58,] 13 0.79620   2.61900
## [59,] 14 0.79640   2.38600
## [60,] 14 0.79660   2.17400
## [61,] 14 0.79680   1.98100
## [62,] 15 0.79690   1.80500
## [63,] 15 0.79760   1.64500
## [64,] 15 0.79820   1.49900
## [65,] 15 0.79880   1.36500
## [66,] 15 0.79940   1.24400
## [67,] 15 0.79980   1.13400
## [68,] 15 0.80030   1.03300
## [69,] 15 0.80060   0.94120
## [70,] 15 0.80100   0.85760
## [71,] 15 0.80120   0.78140
## [72,] 15 0.80150   0.71200
## [73,] 15 0.80170   0.64870
## [74,] 15 0.80190   0.59110
## [75,] 15 0.80210   0.53860
## [76,] 15 0.80220   0.49070
## [77,] 15 0.80230   0.44710
## [78,] 15 0.80240   0.40740
## [79,] 15 0.80250   0.37120
## [80,] 15 0.80260   0.33820
## [81,] 15 0.80270   0.30820
## [82,] 15 0.80270   0.28080
## [83,] 15 0.80280   0.25590
## [84,] 15 0.80280   0.23310
## [85,] 14 0.80280   0.21240
## [86,] 14 0.80290   0.19360
## [87,] 14 0.80290   0.17640
## [88,] 14 0.80290   0.16070
## [89,] 14 0.80290   0.14640
## [90,] 14 0.80300   0.13340
## [91,] 15 0.80300   0.12160
## [92,] 15 0.80300   0.11080
## [93,] 15 0.80300   0.10090
## [94,] 15 0.80300   0.09195
## [95,] 15 0.80300   0.08378
## [96,] 15 0.80300   0.07634
```

We can see from the output that about 75% of the variance is explained when lambda is about 20. This yields the following model. We see that only 2 factors are omitted. We can further reduce the number of factors by increasing lambda.

```r
coef(crimes.fit.elas, s = 20, exact = F)
```

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                       1
## (Intercept) 905.085106
## M            79.530375
## So           27.963458
## Ed          109.389230
## Po1         219.881379
## Po2          73.137275
## LF            9.113376
```

```
## M.F          56.942241
## Pop                  .
## NW           12.988769
## U1          -26.649296
## U2           60.789074
## Wealth        2.142487
## Ineq        156.632766
## Prob        -83.804380
## Time                 .
```

Question 2

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

Answer: With the intent of maximizing sales, ecommerce sites can benefit from functionality that allows the behavior of the site to vary in an experimental way. The different knobs that can be tuned to change the behavior can be consdered the factors for the experiment. A stakeholder can tune these knobs therefore creating an experiment than can then be tied to results (sales).

Question 3

To determine the value of 10 different yes/no features to the market value of a house (large yard, solar roof, etc.), a real estate agent plans to survey 50 potential buyers, showing a fictitious house with different combinations of features. To reduce the survey size, the agent wants to show just 16 fictitious houses.

Use R's FrF2 function (in the FrF2 package) to find a fractional factorial design for this experiment: what set of features should each of the 16 fictitious houses? Note: the output of FrF2 is 1 (include) or -1 (don't include) for each feature.

```r
require(FrF2)
```

The assignment indicates 10 different factors are being considered, and we want to encapsulate those within 16 different houses. We'll use these parameters to generate designs.

```r
houses = FrF2(nruns=16, nfactors=10)
houses
```

```
##     A  B  C  D  E  F  G  H  J  K
## 1   1  1  1  1  1  1  1  1  1  1
## 2  -1 -1  1 -1  1 -1 -1  1  1 -1
## 3   1  1 -1  1  1 -1 -1  1 -1 -1
## 4   1 -1  1  1 -1  1 -1  1 -1 -1
## 5  -1 -1 -1 -1  1  1  1  1 -1  1
## 6   1 -1 -1  1 -1 -1  1  1  1  1
## 7  -1  1  1  1 -1 -1  1 -1  1 -1
## 8  -1  1 -1 -1 -1  1 -1  1  1 -1
## 9   1  1  1 -1  1  1  1 -1 -1 -1
## 10 -1 -1  1  1  1 -1 -1 -1 -1  1
## 11 -1 -1 -1  1  1  1  1 -1  1 -1
## 12 -1  1  1 -1 -1 -1  1  1 -1  1
## 13  1  1 -1 -1  1 -1 -1 -1  1  1
## 14  1 -1 -1 -1 -1 -1  1 -1 -1 -1
## 15 -1  1 -1  1 -1  1 -1 -1 -1  1
## 16  1 -1  1 -1 -1  1 -1 -1  1  1
```

```
## class=design, type= FrF2
```

Instead of inventing arbitrary feature names, we let the function generate the factors names, which are A - K in this case. As shown in the output, we have 16 different designs represeting the 16 different houses.

Question 4

For each of the following distributions, give an example of data that you would expect to follow this distribution (besides the examples already discussed in class).

a. **Binomial** The Binomial Distribution is prevelant in many games and sports where the goal is to hit a target. Examples are shooting free throws in basketball and hitting the bullseye with an arrow. Application of this distribution in these scenarios assumes that each attempt is independent, and the probability of success is the same. This seems reasonable for both scenarios. In the context of a single competition, the probabability of success would be consistent. And since the type of acitivity is void of external influences (e.g. a defensive opponent in basketball).

b. **Geometric** Using the convention that the Geometric distribution reperesents the number of attempts to achieve the first success, hitting a bullseye with an arrow is one example. As with the Binomial distribution, we assume independence between attempts and a consistent probability of success for each attempt.

c. **Poisson** The Poisson Distribution represents the number of events occuring within a specified interval. We assume that the occurrance of events are independent and the rate of occurance is the same across all intervals. This distribution is commonly applied to model arrivals in retail. This could be arrivals at a brick and mortor storefront, or visitors to an ecommerce web site.

d. **Exponential** The Exponential Distribution models the time between events in Poisson processes. Using our previous example of modeling visitors to an ecommerice web site, this distribution could model the time between each new visitor.

e. **Weibull** The Weibull Distribution is used to model the time to failure. Technologies such as Amazon Web Services and Microsoft Azure utilize large installations of machines. These machines can fail for a number of reasons. For example, this distribution can be used to model the time until hard drive failure.