Input Analysis

Alireza Sheikh-Zadeh, Ph.D.

Using a Fitted Distribution for Monte Carlo Simulation

When performing a simulation study, there is no substitution for actually observing the system and collecting the data required for the modeling effort.

Input modeling begins with data collection, probability, statistics, and analysis.

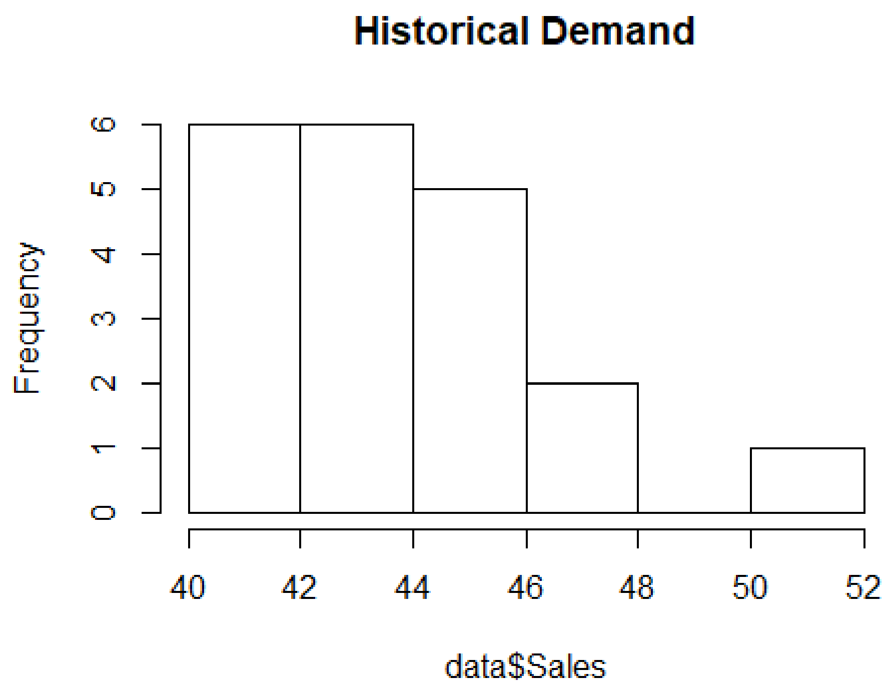
A typical input modeling process includes the following procedures:

1. Determine whether your data is discrete or continuous.
2. Visualize the data: Check the histogram, plot a time series plot, and make an autocorrelation plot of the data. Check if the week, period, or day of week influence the statistical properties of the count data.
3. Hypothesizing distributions. Fortunately, R has a very useful package for fitting a wide variety of discrete and continuous distributions call the fitdistrplus package.
4. Estimating parameters.
5. Checking goodness of fit for hypothesized distributions.

Example In the newsvendor model, suppose we have a historical data for demand. We are interested to know what distribution can we fit for this data.

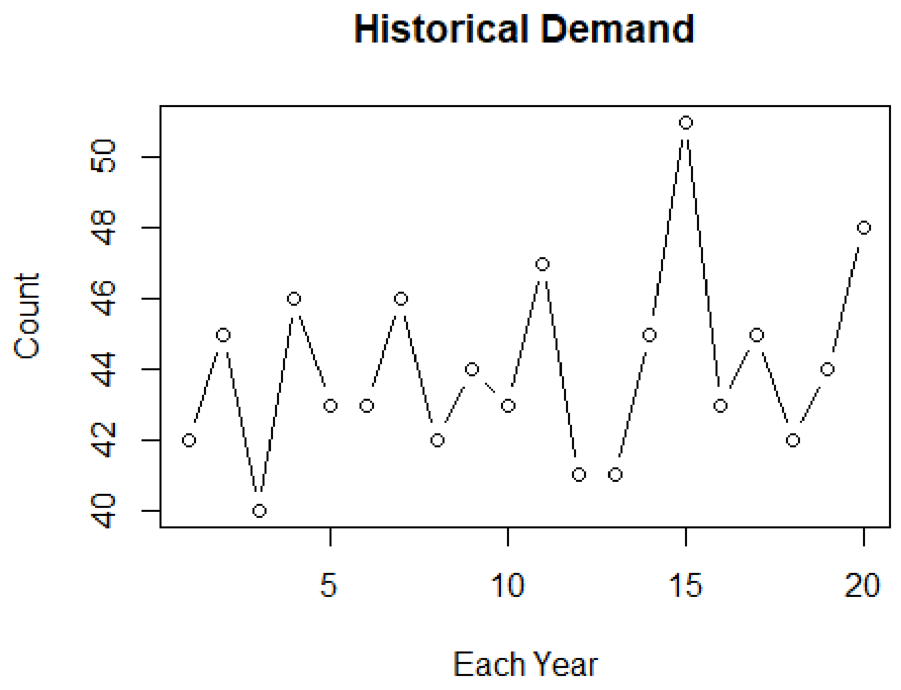
```r
data = read.csv("http://tiny.cc/histDemand")

# 1. Data is discrete.

# 2. visualize data
```

```r
hist(data$Sales, main="Historical Demand")
```



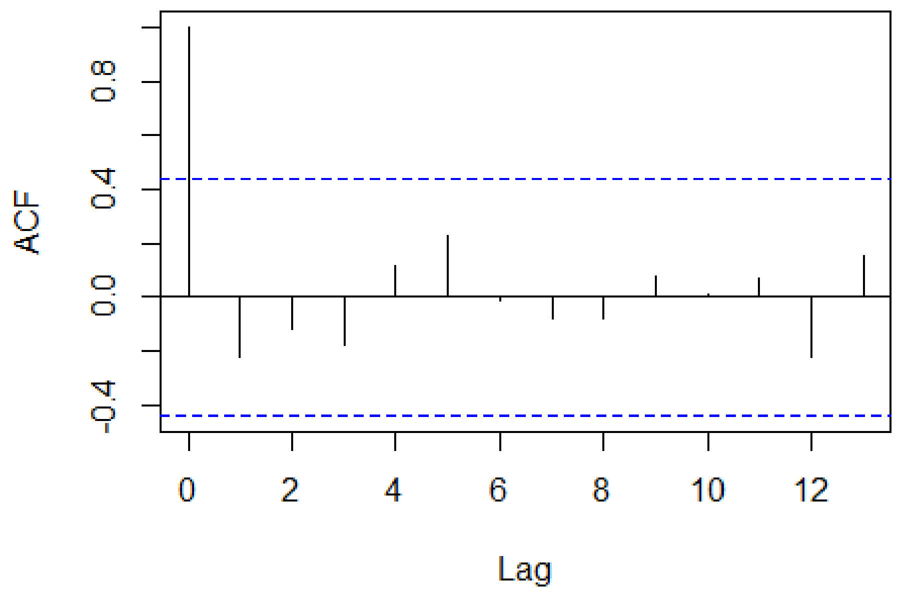
```r
# The distribution of the data is positively (right) skewed and unimodal.
```

```r
plot(data$Sales, type="b", main="Historical Demand", ylab = "Count", xlab =
"Each Year")
```

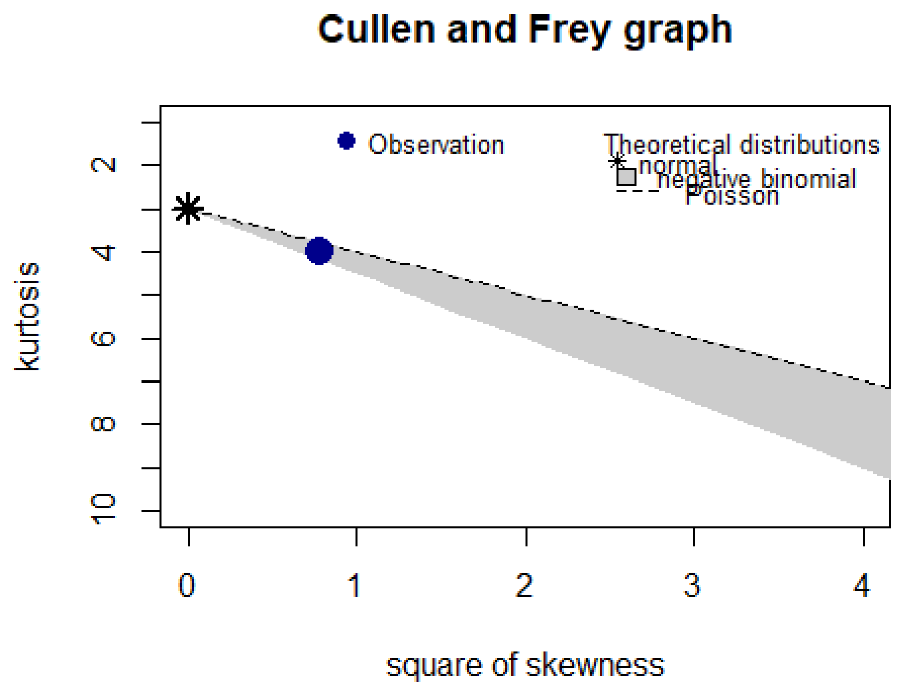


```r
# There is no noticable trend in the time-series plot
```

```r
acf(data$Sales) # Auto-correlation plot
#the autocorrelation plot shows no significant correlation with respect to
observation number (all lags are well within the confidence band). Therefore,
the data appears to be stationary (IID).

# 3. Hypothesizing distributions.
library(fitdistrplus)
```



```
descdist(data$Sales, discrete = TRUE)
```



```
## summary statistics
## ------
## min:  40   max:  51
## median:  43.5
## mean:  44.05
## estimated sd:  2.665076
## estimated skewness:  0.8859026
## estimated kurtosis:  4.019658

# Let's try Poisson
fit.nbinom = fitdist(data$Sales, "nbinom")

## Warning in sqrt(diag(varcovar)): NaNs produced

## Warning in sqrt(1/diag(V)): NaNs produced

## Warning in cov2cor(varcovar): diag(.) had 0 or NA entries; non-finite
## result is doubtful

summary(fit.nbinom)

## Fitting of the distribution ' nbinom ' by maximum likelihood
## Parameters :
##          estimate Std. Error
## size 1.331700e+08        NaN
## mu   4.404782e+01   1.484009
## Loglikelihood:  -57.76313   AIC:  119.5263   BIC:  121.5177
## Correlation matrix:
##      size  mu
## size    1 NaN
## mu    NaN   1
```

```
fit.pois = fitdist(data$Sales, "pois")
summary(fit.pois)
```

```
## Fitting of the distribution ' pois ' by maximum likelihood
## Parameters :
##         estimate Std. Error
## lambda    44.05   1.484082
## Loglikelihood:  -57.76313   AIC:  117.5263   BIC:  118.522
```

```
gofstat(fit.pois)
```

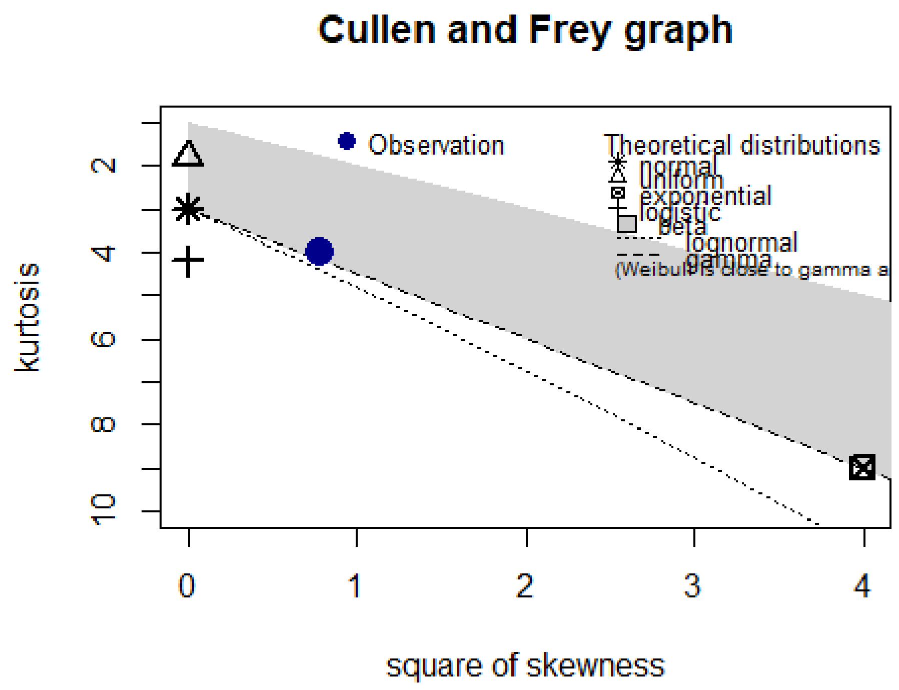
```
## Chi-squared statistic:  15.9012
## Degree of freedom of the Chi-squared distribution:  3
## Chi-squared p-value:  0.001188116
##    the p-value may be wrong with some theoretical counts < 5
## Chi-squared table:
##       obscounts theocounts
## <= 41  3.000000   7.169223
## <= 42  3.000000   1.170599
## <= 43  4.000000   1.199184
## <= 45  5.000000   2.375748
## > 45   5.000000   8.085247
##
## Goodness-of-fit criteria
##                                1-mle-pois
## Akaike's Information Criterion   117.5263
## Bayesian Information Criterion   118.5220
```

```
# H0: Data is distributed by Poisson distribution
# Test is rejected. p-value < 0.05
# Therefore, it's better to try to fit continuous distributions, otherwise,
use resampling (sampling with replacement) for simulation
```

Let’s try continuous distributions:

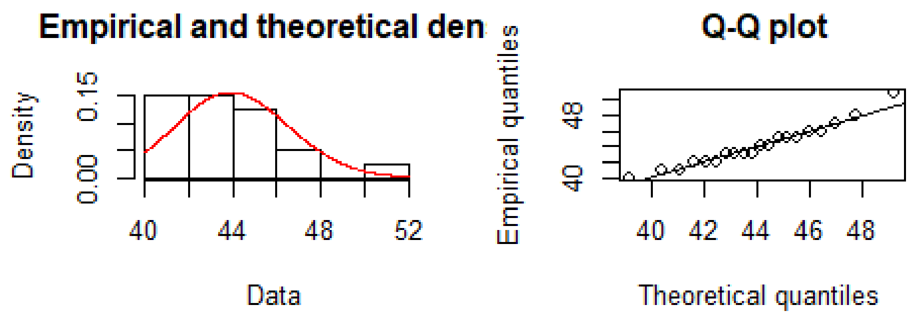
The kurtosis and squared skewness of your sample is plotted as a blue point named “Observation”. It seems that possible distributions include the Gamma Weibull and possibly the Normal distribution.

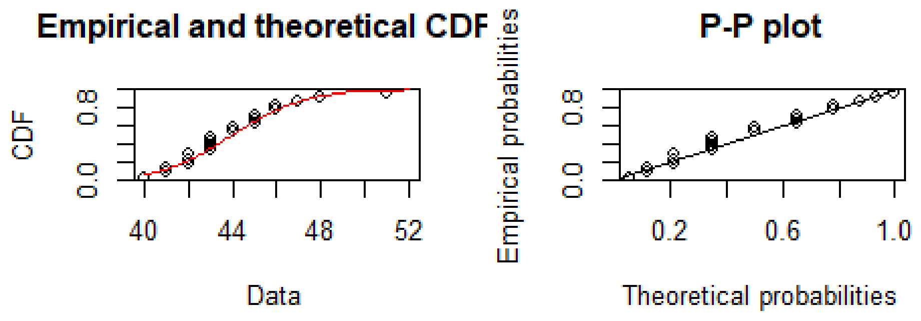
```
# If we assume data is continuous.
library(fitdistrplus)
descdist(data$Sales, discrete = FALSE)
```



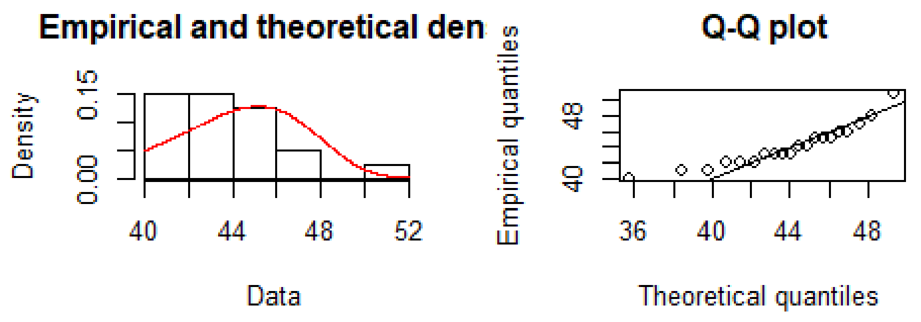
```
## summary statistics
## ------
## min:  40   max:  51
## median:  43.5
## mean:  44.05
## estimated sd:  2.665076
## estimated skewness:  0.8859026
## estimated kurtosis:  4.019658

fit.gamma <- fitdist(data$Sales, "gamma")
summary(fit.gamma)

## Fitting of the distribution ' gamma ' by maximum likelihood
## Parameters :
##         estimate Std. Error
## shape 295.633054  93.434434
## rate    6.711303   2.122893
## Loglikelihood:  -47.17316   AIC:  98.34632   BIC:  100.3378
## Correlation matrix:
##           shape      rate
## shape 1.0000000 0.9991545
## rate  0.9991545 1.0000000

fit.weibull <- fitdist(data$Sales, "weibull")
summary(fit.weibull)

## Fitting of the distribution ' weibull ' by maximum likelihood
## Parameters :
##       estimate Std. Error
## shape 15.47879  2.3942226
## scale 45.34195  0.6971177
## Loglikelihood:  -50.35435   AIC:  104.7087   BIC:  106.7002
## Correlation matrix:
##           shape     scale
## shape 1.0000000 0.3421537
## scale 0.3421537 1.0000000

fit.norm <- fitdist(data$Sales, "norm")
summary(fit.norm)

## Fitting of the distribution ' norm ' by maximum likelihood
## Parameters :
##       estimate Std. Error
## mean 44.050000  0.5808399
## sd    2.597595  0.4107156
## Loglikelihood:  -47.47049   AIC:  98.94098   BIC:  100.9324
## Correlation matrix:
##      mean sd
## mean    1  0
## sd      0  1

# So far Gamma has the best log-likelihood.
```

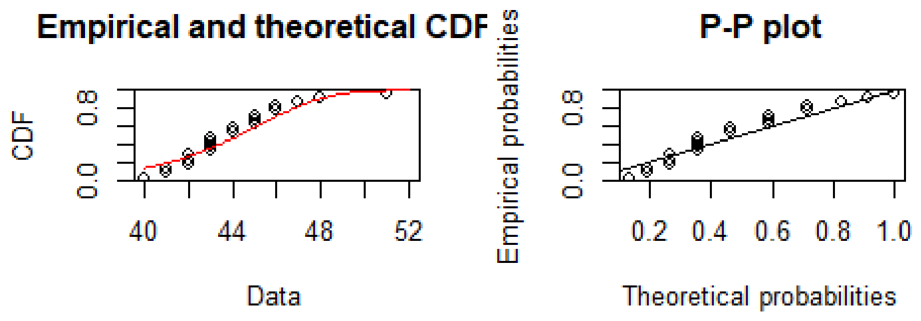
```r
plot(fit.gamma)
```



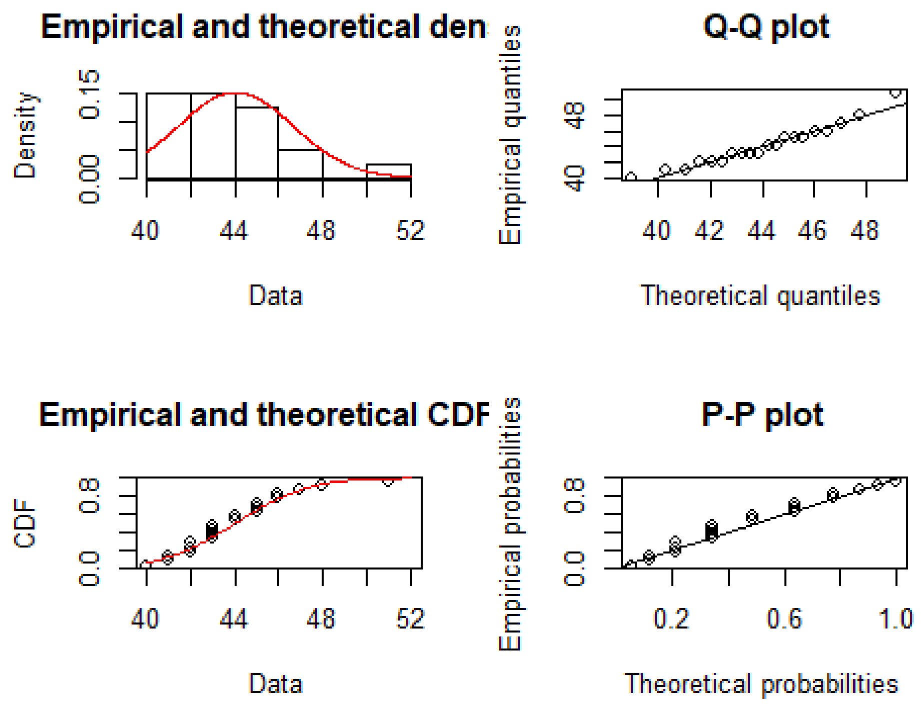


```r
plot(fit.weibull)
```





```
plot(fit.norm)
```



Both Gamma and Normal look good.

```
fit.gamma$loglik

## [1] -47.17316

fit.norm$loglik

## [1] -47.47049
```

In our example, we pick Gamma distribution.

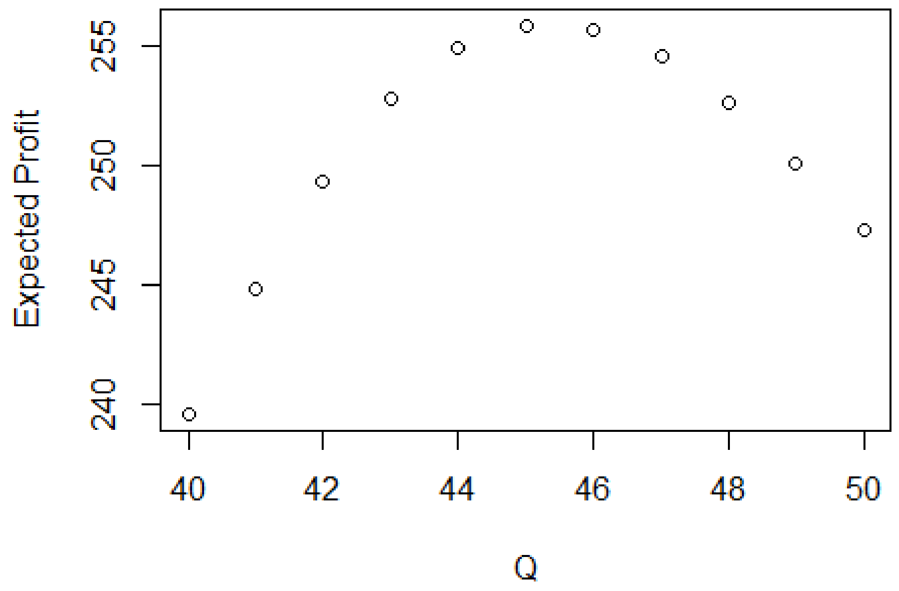
```
summary(fit.gamma)

## Fitting of the distribution ' gamma ' by maximum likelihood
## Parameters :
##         estimate Std. Error
## shape 295.633054  93.434434
## rate    6.711303   2.122893
## Loglikelihood:  -47.17316   AIC:  98.34632   BIC:  100.3378
## Correlation matrix:
##           shape      rate
## shape 1.0000000 0.9991545
## rate  0.9991545 1.0000000
```

Then we produce Gamma random numbers for our model.

```r
# Let's solve the newsVendor model again, this time we assume demand is
distributed by gamma distribution.
# Data
R = 18  # Selling price
C = 12  # Cost
S = 9   # Discount Price
# Model
netProfitFun = function(D, Q, R, S, C){
  R*min(D,Q) + S * max(0, Q-D) - C*Q
}

num_sim = 1000

Qrange = 40:50

sim_D = rgamma(num_sim, shape = 295.63, rate = 6.71)

profitMatrix <- matrix(0, nrow = num_sim, ncol = length(Qrange))

j = 0
for (Q in Qrange) {
  j = j+1
  for (i in 1:num_sim) {
    profitMatrix[i, j] = netProfitFun(sim_D[i], Q, R, S, C)
  }
}

# Expected profit for each Q
colMeans(profitMatrix)
```

```
##  [1] 239.6134 244.8689 249.3444 252.7308 254.9158 255.8122 255.6180
##  [8] 254.5315 252.5635 250.0959 247.3238
```

```r
plot(Qrange, colMeans(profitMatrix), ylab = "Expected Profit", xlab = "Q")
```



```r
# Q* = 45
```

Class Practice

The observations available in the file represent the count of the number of failures on a windmill turbine farm per year. Using the techniques discussed today, recommend an input distribution model for this situation.

```r
numFailure <- read.csv("http://tiny.cc/numFailure")
numFailure <- numFailure$failure
```