```
In [69]: using DataFrames
```

```
In [70]: import Pkg
```

```
In [71]: Pkg.add("CSV")
```

```
  Resolving package versions...
   Updating `C:\Users\Tony Diana\.julia\environments\v1.4\Project.toml`
 [no changes]
   Updating `C:\Users\Tony Diana\.julia\environments\v1.4\Manifest.toml`
 [no changes]
```

```
In [72]: using CSV
         apt= CSV.read("c:\\Users\\Tony Diana\\Documents\\ATL_ORD_2014.csv")
```

Out[72]: 1,815 rows × 15 columns (omitted printing of 9 columns)

	flight	enroute_tm_fln	exc_tm_fln	tm_fln_to_unimp	enroute_m_unimp	enroute_unimp
	Int64	Float64	Float64	Float64	Float64	Float64
1	1	54.66	11.03	1.25	11.04	43.62
2	2	53.21	9.58	1.22	9.59	43.62
3	3	48.42	4.8	1.11	4.8	43.62
4	4	51.34	7.72	1.18	7.72	43.62
5	5	46.99	3.37	1.08	3.37	43.62
6	6	50.64	7.02	1.16	7.02	43.62
7	7	55.62	12.0	1.28	12.0	43.62
8	8	51.74	8.12	1.19	8.12	43.62
9	9	50.56	6.93	1.16	6.94	43.62

```
In [73]: describe(apt)
```

Out[73]: 15 rows × 8 columns

|	variable	mean	min	median	max	nunique	nmissing	eltype
	Symbol	Float64	Real	Float64	Real	Nothing	Nothing	DataType
1	flight	908.0	1	908.0	1815			Int64
2	enroute_tm_fln	46.4302	36.07	46.46	57.04			Float64
3	exc_tm_fln	4.46284	0.0	4.3	14.95			Float64
4	tm_fln_to_unimp	1.10245	0.86	1.1	1.37			Float64
5	enroute_m_unimp	4.30528	-5.66	4.3	14.95			Float64
6	enroute_unimp	42.125	40.94	41.73	43.62			Float64
7	enroute_mls_fln	337.012	326.68	335.97	354.74			Float64
8	exc_mls_fln	10.8198	0.49	9.78	28.55			Float64
9	mls_fln_to_GCR	1.03338	1.0	1.03	1.09			Float64
10	speed	437.908	351.82	434.94	564.13			Float64
11	GCR_mls	326.19	326.19	326.19	326.19			Float64
12	tm_fln	94.5652	71.68	93.92	175.5			Float64
13	mls_fln	587.397	554.5	580.99	848.23			Float64
14	speed_1	373.821	235.42	373.64	478.14			Float64
15	GCR_dist	526.19	526.19	526.19	526.19			Float64

```
In [74]: Pkg.add("Plots")
```

```
  Resolving package versions...
   Updating `C:\Users\Tony Diana\.julia\environments\v1.4\Project.toml`
 [no changes]
   Updating `C:\Users\Tony Diana\.julia\environments\v1.4\Manifest.toml`
 [no changes]
```

```
In [75]: Pkg.add("StatPlots")
```

```
  Resolving package versions...
   Updating `C:\Users\Tony Diana\.julia\environments\v1.4\Project.toml`
 [no changes]
   Updating `C:\Users\Tony Diana\.julia\environments\v1.4\Manifest.toml`
 [no changes]
```

```
In [76]: Pkg.add("PyPlot")
```

```
  Resolving package versions...
   Updating `C:\Users\Tony Diana\.julia\environments\v1.4\Project.toml`
 [no changes]
   Updating `C:\Users\Tony Diana\.julia\environments\v1.4\Manifest.toml`
 [no changes]
```

```
In [77]: using Plots, StatPlots        #import required packages
         pyplot()                       #Set the backend as matplotlib.pyplot
         Plots.histogram((apt[:enroute_tm_fln]),bins=25,xlabel="Unimpeded Enroute Time",labels="Frequency")        #Plot histogram
```

```
┌ Warning: `getindex(df::DataFrame, col_ind::ColumnIndex)` is deprecated, use `df[!, col_ind]` instead.
│   caller = top-level scope at In[77]:3
└ @ Core In[77]:3
```

Out[77]:
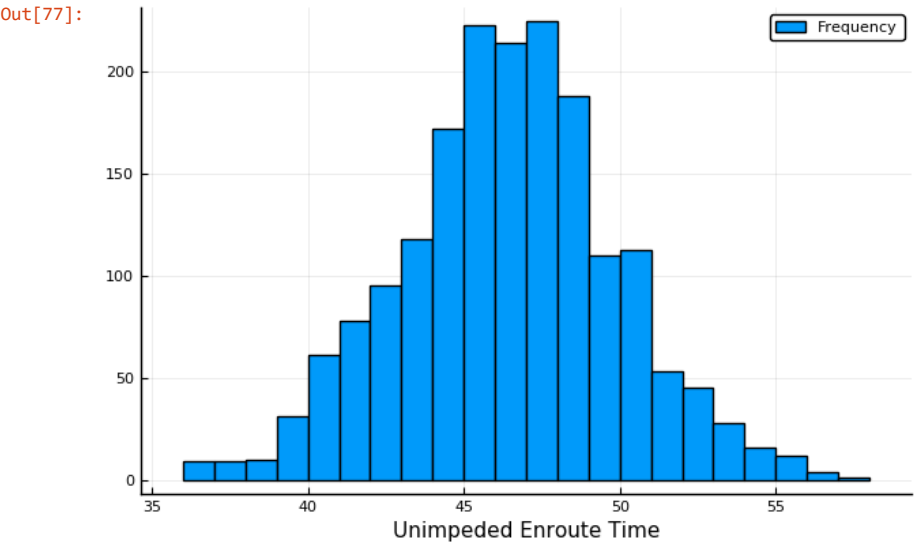
```
In [78]:  Plots.boxplot((apt[:enroute_tm_fln]),bins=25,xlabel="Unimpeded Enroute Time",labels="Frequency")
```

```
Warning: `getindex(df::DataFrame, col_ind::ColumnIndex)` is deprecated, use `df[!, col_ind]` instead.
  caller = top-level scope at In[78]:1
@ Core In[78]:1
Warning: seriestype boxplot has been moved to StatsPlots.  To use: `Pkg.add("StatsPlots")`; using StatsPlots`
@ Plots C:\Users\Tony Diana\.julia\packages\Plots\cc8wh\src\args.jl:1093
Warning: seriestype boxplot has been moved to StatsPlots.  To use: `Pkg.add("StatsPlots")`; using StatsPlots`
@ Plots C:\Users\Tony Diana\.julia\packages\Plots\cc8wh\src\args.jl:1093
```

Out[78]:
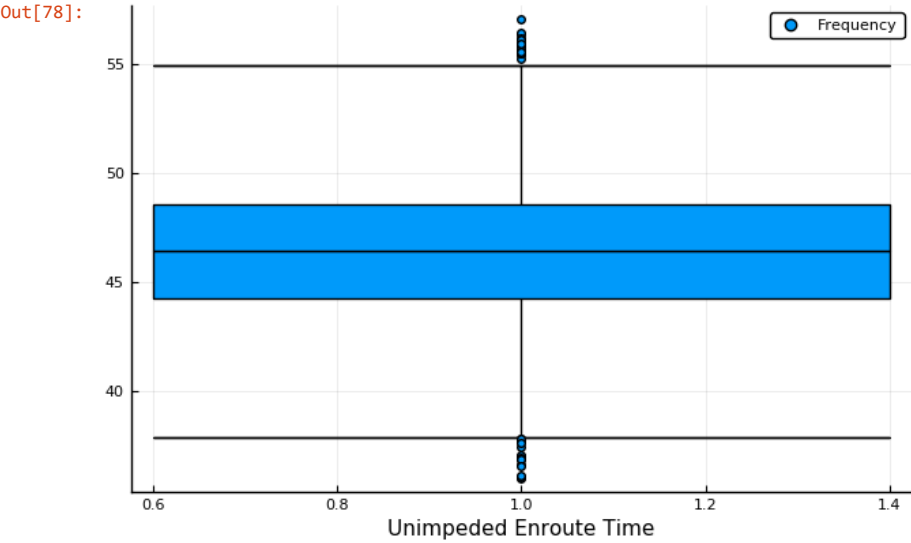

```
In [79]:  import Pkg
          Pkg.add("GLM")
          using GLM
```

```
  Resolving package versions...
   Updating `C:\Users\Tony Diana\.julia\environments\v1.4\Project.toml`
[no changes]
   Updating `C:\Users\Tony Diana\.julia\environments\v1.4\Manifest.toml`
[no changes]
```

```
In [80]:  model = lm(@formula(enroute_tm_fln ~ exc_mls_fln + exc_tm_fln), apt)
```

Out[80]: StatsModels.TableRegressionModel{LinearModel{GLM.LmResp{Array{Float64,1}},GLM.DensePredChol{Float64,LinearAlgebra.Cholesky{Float64,Array{Float64,2}}}},Array{Float64,2}}

enroute_tm_fln ~ 1 + exc_mls_fln + exc_tm_fln

Coefficients:

	Estimate	Std. Error	t value	Pr(>|t|)	Lower 95%	Upper 95%
(Intercept)	41.9733	0.0887986	472.679	<1e-99	41.7991	42.1474
exc_mls_fln	-0.0166281	0.00743014	-2.23792	0.0253	-0.0312006	-0.00205553
exc_tm_fln	1.039	0.00997599	104.15	<1e-99	1.01944	1.05857

```
In [81]:  round.(stderror(model), digits=5)
```

Out[81]: 3-element Array{Float64,1}:
 0.0888
 0.00743
 0.00998

```
In [91]:  println("The coefficient of determination is: ",round.(r2(model)*100, digits=3))
```

The coefficient of determination is: 86.114

```
In [88]:  using PyPlot
          Pkg.add("PyPlot")
          # Simple Scatter(apt[:enroute_tm_fln]),  plot using PyPlot
          Plots.scatter((apt[:exc_mls_fln]), (apt[:enroute_tm_fln]), xaxis="Excess Miles", yaxis="Enroute Time")
```

```
  Resolving package versions...
   Updating `C:\Users\Tony Diana\.julia\environments\v1.4\Project.toml`
[no changes]
   Updating `C:\Users\Tony Diana\.julia\environments\v1.4\Manifest.toml`
[no changes]
Warning: `getindex(df::DataFrame, col_ind::ColumnIndex)` is deprecated, use `df[!, col_ind]` instead.
  caller = top-level scope at In[88]:3
@ Core In[88]:3
Warning: `getindex(df::DataFrame, col_ind::ColumnIndex)` is deprecated, use `df[!, col_ind]` instead.
  caller = top-level scope at In[88]:3
@ Core In[88]:3
```

Out[88]:
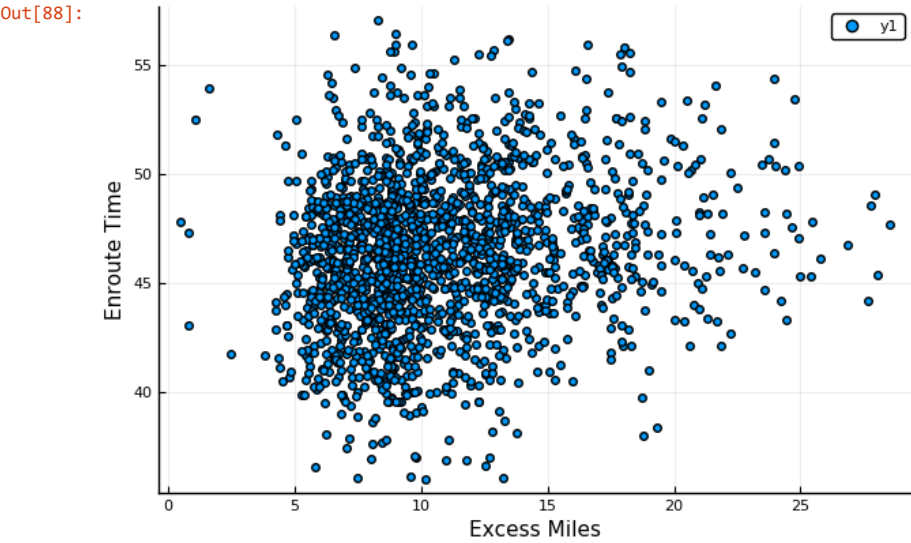
```
In [125]: using CSV, DataFrames, ScikitLearn, Random, PyCall, Statistics
          y=apt.enroute_tm_fln
          X=Matrix(apt[2:14])
```

```
┌ Warning: `getindex(df::DataFrame, col_inds::Union{AbstractVector, Regex, Not})` is deprecated, use `df[:, col_inds]` instead.
│   caller = top-level scope at In[125]:3
└ @ Core In[125]:3
```

```
Out[125]: 1815×13 Array{Float64,2}:
          54.66  11.03  1.25  11.04  43.62  344.44  …  326.19  104.15  596.36  343.56
          53.21   9.58  1.22   9.59  43.62  340.86     326.19  104.75  589.3   337.55
          48.42   4.8   1.11   4.8   43.62  334.38     326.19   96.0   582.44  364.03
          51.34   7.72  1.18   7.72  43.62  338.65     326.19  102.48  587.33  343.87
          46.99   3.37  1.08   3.37  43.62  332.42     326.19   97.32  582.0   358.82
          50.64   7.02  1.16   7.02  43.62  335.85  …  326.19  101.17  583.48  346.04
          55.62  12.0   1.28  12.0   43.62  335.13     326.19  108.0   583.31  324.06
          51.74   8.12  1.19   8.12  43.62  336.06     326.19  109.28  588.27  322.99
          50.56   6.93  1.16   6.94  43.62  335.34     326.19  100.17  582.83  349.1
          53.52   9.9   1.23   9.9   43.62  337.33     326.19  103.92  591.35  341.43
          52.92   9.3   1.21   9.3   43.62  342.71  …  326.19  104.82  585.7   335.26
          49.89   6.27  1.14   6.27  43.62  338.72     326.19   99.55  592.29  356.98
          47.54   3.92  1.09   3.92  43.62  334.26     326.19   98.25  574.82  351.04
            ⋮                                ⋮      ⋱        ⋮
          43.22   1.49  1.04   1.49  41.73  332.59     326.19   84.98  577.53  407.76
          47.31   5.58  1.13   5.58  41.73  338.88     326.19   91.8   586.67  383.44
          46.51   4.78  1.11   4.78  41.73  334.46  …  326.19  112.05  626.46  335.45
          45.17   3.44  1.08   3.44  41.73  334.22     326.19   90.82  602.16  397.82
          45.21   3.48  1.08   3.48  41.73  343.38     326.19   90.92  587.06  387.41
          46.46   4.73  1.11   4.73  41.73  338.87     326.19   92.55  585.46  379.55
          45.77   4.04  1.1    4.04  41.73  335.46     326.19   91.6   592.39  388.03
          46.12   4.39  1.11   4.39  41.73  334.05  …  326.19   88.88  578.94  390.82
          43.91   2.18  1.05   2.18  41.73  334.23     326.19   86.23  576.95  401.45
          43.99   2.26  1.05   2.26  41.73  334.89     326.19   93.53  598.72  384.08
          45.97   4.24  1.1    4.24  41.73  332.89     326.19   92.53  582.78  377.9
          45.24   3.51  1.08   3.51  41.73  333.25     326.19   86.87  571.55  394.76
```

```
In [134]: using Pkg
          Pkg.add("MLDataUtils")
          using MLDataUtils
          # shuffle the data so its not in order when we split it up
          Xs, ys = shuffleobs((transpose(X), y))

          #now split the data into training sets and validation sets
          (X_train1, y_train1), (X_test1, y_test1) = splitobs((Xs, ys); at = 0.67)

           # need to convert the split data back into arrays

             X_train = Array(transpose(X_train1))
             y_train = Array(y_train1)
             X_test = Array(transpose(X_test1))
             y_test = Array(y_test1)
```

```
  Resolving package versions...
  Updating `C:\Users\Tony Diana\.julia\environments\v1.4\Project.toml`
 [no changes]
  Updating `C:\Users\Tony Diana\.julia\environments\v1.4\Manifest.toml`
 [no changes]
```

```
Out[134]: 599-element Array{Float64,1}:
          44.84
          42.87
          46.47
          47.1
          46.41
          47.64
          48.1
          48.57
          45.72
          47.57
          41.51
          48.14
          48.71
           ⋮
          45.66
          49.79
          45.64
          47.01
          50.35
          40.66
          49.79
          40.56
          45.89
          42.66
          50.81
          43.31
```

```
In [2]: import Pkg
        Pkg.add("DataArrays")
```

```
  Updating registry at `C:\Users\Tony Diana\.julia\registries\General`
  Updating git-repo `https://github.com/JuliaRegistries/General.git`
```

```
Fetching: [========================================>]  100.0 %.0 %
```

```
  Resolving package versions...
  Installed DataArrays ─ v0.7.0
  Installed Zlib_jll ──── v1.2.11+12
  Updating `C:\Users\Tony Diana\.julia\environments\v1.4\Project.toml`
 [0fe7c1db] + DataArrays v0.7.0
  Updating `C:\Users\Tony Diana\.julia\environments\v1.4\Manifest.toml`
 [0fe7c1db] + DataArrays v0.7.0
 [83775a58] ↑ Zlib_jll v1.2.11+11 ⇒ v1.2.11+12
```

```
In [4]: d = 3       # sample dimension
        n = 1000    # number of samples

        # prepare data
        w = randn(d+1)     # generate the weight vector
        X = randn(d, n)    # generate input features
        y = sign(X'w[1:d] + w[d+1] + 0.2 * randn(n))  # generate (noisy) response

        # perform estimation
        ret = Regression.solve(
            logisticreg(X, y; bias=1.0),    # construct a logistic regression problem
            reg=SqrL2Reg(1.0e-2),           # apply squared L2 regularization
            options=Options(verbosity=:iter, grtol=1.0e-6 * n))  # set options

        # extract results
        w_e = ret.sol
```

```
MethodError: no method matching +(::Array{Float64,1}, ::Float64)
Closest candidates are:
  +(::Any, ::Any, !Matched::Any, !Matched::Any...) at operators.jl:529
  +(!Matched::Bool, ::T) where T<:AbstractFloat at bool.jl:104
  +(!Matched::Float64, ::Float64) at float.jl:401
  ...

Stacktrace:
 [1] +(::Array{Float64,1}, ::Float64, ::Array{Float64,1}) at .\operators.jl:529
 [2] top-level scope at In[4]:7
```

```
In [ ]:
```