Pset2_JuliaCode

March 7, 2017

```
In [1]: using JuMP, Gurobi, PyPlot
```

```
INFO: Precompiling module JuMP.
```

```
In [27]: X = readcsv("Framingham.csv");
         X_train = X[2:1500,1:15];
         y_train = X[2:1500,16]
         X_validate = X[1501:3001,1:15];
         y_validate = X[1501:3001,16]
         X_test = X[3001:3659,1:15];
         y_test = X[3001:3659,16];
```

```
In [22]: n = 3658
         d = 15
         cost_vector = zeros(n,1)
         for i = 1:n
             if X[i+1,d+1] == -1
                 cost_vector[i] = 1
             else
                 cost_vector[i] = 5
             end
         end
```

```
In [85]: function nominal_SVM(data,label)
             SVMn = Model(solver=GurobiSolver(OutputFlag=0))
             n = length(label)
             d = size(data,2)
             @variable(SVMn, z[i=1:n]>=0)
             @variable(SVMn, w[j=1:d])
             @variable(SVMn, b)
             for i = 1:n
                 @constraint(SVMn, z[i] >= 1 - label[i] * (sum(data[i,j]*w[j] for j=1:d) - b) )
             end
             @objective(SVMn, Min, sum(cost_vector[i]*z[i] for i=1:n) )
             solve(SVMn)
             return [getvalue(w);getvalue(b)]
         end
```

```
WARNING: Method definition nominal_SVM(Any, Any) in module Main at In[63]:2
```

```
Out[85]: nominal_SVM (generic function with 1 method)
```

```
overwritten at In[85]:2.
```

```
In [95]: function RobustSVM_feature(data,label,Γ)
             RSVMf = Model(solver=GurobiSolver(OutputFlag=0))
             n = length(label)
             d = size(data,2)
             @variable(RSVMf, z[i=1:n]>=0)
             @variable(RSVMf, w[j=1:d])
             @variable(RSVMf, b)
             @variable(RSVMf, wmax)
             for j = 1:d
                 @constraint(RSVMf, wmax>=w[j])
             end
             for i = 1:n
                 @constraint(RSVMf, z[i] >= 1 - label[i] * (sum(data[i,j]*w[j] for j=1:d) - b) + Γ*wmax
             end
             @objective(RSVMf, Min, sum(cost_vector[i]*z[i] for i=1:n) )
             solve(RSVMf)
             return [getvalue(w);getvalue(b)]
         end
```

```
Out[95]: RobustSVM_feature (generic function with 1 method)
```

```
In [148]: function RobustSVM_label(data,label,Γ)
              RSVMl = Model(solver=GurobiSolver(OutputFlag=0))
              n = length(label)
              d = size(data,2)
              M = 1e4
              @variable(RSVMl, w[j=1:d])
              @variable(RSVMl, b)
              @variable(RSVMl, q>=0)
              @variable(RSVMl, r[i=1:n]>=0)
              @variable(RSVMl, ϕ[i=1:n]>=0)
              @variable(RSVMl, ξ[i=1:n]>=0)
              @variable(RSVMl, s[1:n],Bin)
              @variable(RSVMl, t[1:n],Bin)


              for i = 1:n
                  @constraint(RSVMl, q + r[i] >= cost_vector[i]*(ϕ[i] - ξ[i]) )
                  @constraint(RSVMl, ξ[i] >= 1 - label[i] * (sum(data[i,j]*w[j] for j=1:d) - b) )
                  @constraint(RSVMl, ξ[i] <= 1 - label[i] * (sum(data[i,j]*w[j] for j=1:d) - b) + M*(1-s
                  @constraint(RSVMl, ξ[i] <= M*s[i])
                  @constraint(RSVMl, ϕ[i] >= 1 + label[i] * (sum(data[i,j]*w[j] for j=1:d) - b) )
                  @constraint(RSVMl, ϕ[i] <= 1 + label[i] * (sum(data[i,j]*w[j] for j=1:d) - b) + M*(1-
                  @constraint(RSVMl, ϕ[i] <= M*t[i])
              end
                  @objective(RSVMl, Min, Γ*q+ sum(cost_vector[i]*ξ[i]+r[i] for i=1:n) )
              solve(RSVMl)
              return [getvalue(w);getvalue(b)]
          end
```

```
WARNING: Method definition RobustSVM_label(Any, Any,
```

```
Out[148]: RobustSVM_label (generic function with 1 method)
```

```
Any) in module Main at In[138]:2 overwritten at In[148]:2.
```

```
In [149]: function predict(w,data,label)
              n = length(label)
              d = size(data,2)
              predict_label = sign(data*vec(w[1:end-1]) - w[end])
              return sum(abs(predict_label-label))/2/n
          end
```

```
WARNING: Method definition predict(Any, Any, Any)
```

```
Out[149]: predict (generic function with 2 methods)
```

```
in module Main at In[139]:2 overwritten at In[149]:2.
```

```
In [176]: en = predict(nominal_SVM(X_train, y_train),X_test,y_test);
          ef = []
          el = []
          Γlist = [1,5,10,15,20,25,50,100,120]
          for Γ in Γlist
              push!(ef, predict(RobustSVM_feature(X_train, y_train, Γ),X_test,y_test) );
              push!(el, predict(RobustSVM_label(X_train,y_train,Γ),X_test,y_test) );
          end
```

```
In [ ]: plot(Γlist,[en;en;en;en;en;en;en;en;en],label="nominal")
        plot(Γlist,ef,label="feature")
        plot(Γlist,el,label="label")
        legend(loc="upper left")
        xlabel("Gamma")
        ylabel("Error Rate")
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```