

A7_Testing

November 28, 2018

1 Assignment A7: Bayesian Classification

1.0.1 CS 6640

1.0.2 Fall 2018

By: Cade Parkison

You are to develop a vehicle pattern recognition function based on the optimal Bayes classifier. Using the videos found in the CS6640 class data sub-directory A7, select the first 2 (i.e., $< X > 1$ and $< X > 2$) for the three classes, bus, car, and truck, to get training vectors with suitable features, and then build Bayes models and evaluate those models on the remaining videos (i.e., classify the vehicles in each frame). Carefully describe each step of the process; data collection, feature selection and computation, model choice, training results, and evaluation. Implement the requested two functions.

Bayes Classifier:

```
In [146]: %%file CS6640_Bayes.m
function best_class = CS6640_Bayes(x,class_probs,class_models)
% CS6640_Bayes - Bayes classifier
% On input:
%     x (mx1 vector): feature vector
%     class_probs (1xn vector): probabilities of n classes (sums to 1)
%     class_models (1xn vector struct): class models: means and
%     variances
%     (k).mean (mx1 vector): k_th class mean vector
%     (k).var (mxm array): k_th class covariance matrix
% On output:
%     best_class (int): index of best class for x
% Call:
%     c = CS6640_Bayes(x,cp,cm);
% Author:
%     Cade Parkison
%     UU
%     Fall 2018
%
```

```

% number of features
m = length(x);

% number of classes
n = length(class_probs);

% stores the bayes decision function values for each class
bayes = zeros([n,1]);

% loop over each class and find the bayes decision function
for j=1:n
    Pj = class_probs(j);
    Cj = class_models(j).var;
    Mj = class_models(j).mean;

    bayes(j) = log(Pj) - 1/2*log(det(Cj)) - 1/2*((x-Mj)'*inv(Cj)*(x-Mj));
end

 [~,best_class] = max(bayes);

```

Created file 'C:\Users\cadep\School\CS_6640\A7\CS6640_Bayes.m'.

Build Bayes Model:

```

In [22]: %%file CS6640_build_Bayes.m
function model = CS6640_build_Bayes(X)
% CS6640_build_Bayes - build Bayes model for dataset X
% On input:
%     X (nxm array): n samples of m-tuple feature vectors
% On output:
%     model (struct): model: mean and variance
%         .mean (mx1 vector): mean of X
%         .var (mxm array): covariance matrix of X
% Call:
%     m1 = CS6640_build_Bayes([0,0; 0.1,0; -0.1,0]);
% Author:
%     Cade Parkison
%     UU
%     Fall 2018
%

model.mean = mean(X)';
model.var = cov(X);

```

Created file 'C:\Users\cadep\School\CS_6640\A7\CS6640_build_Bayes.m'.

```

In [325]: %%file CS6640_data_collection.m
function X = CS6640_data_collection(object_data)
% CS6640_data_collection - Cleans segmented movie data and extracts features from ea
% On input:
%     object_data (struct) object data of moving objects from M
% On output:
%     X (nxm array): n samples of m-tuple feature vectors
% Call:
%     X = CS6640_data_collection(M)
% Author:
%     Cade Parkison
%     UU
%     Fall 2018
%

M = 480;
N = 640;
frames = size(object_data);
n_frames = frames(2);

% Filter out all but largest objects
for f=1:n_frames
    objs = object_data(f);
    n_objects = objs.num_objects;
    obj_sizes = [];
    for k=1:n_objects
        obj_sizes = [obj_sizes; objs.objects(k).num_pixels];
    end
    [~,largest] = max(obj_sizes);
    % Remove all but the largest object
    %for k=1:n_objects
    %if k~=largest
    %    object_data(f).objects(k) = [];
    %end
    %end
    object_data(f).objects = objs.objects(largest);

end

% Remove objects that are not near tracked line
centroids = []; % (Px2 Array) where P is the number of object centroids in object_d
for f=1:n_frames
    objs = object_data(f);
    %n_objects = objs.num_objects;
    %for k=1:n_objects

```

```

        centroidX = objs.objects.col_mean;
        centroidY = M - objs.objects.row_mean + 1;
        centroids = [centroids; [centroidX, centroidY]];
        %end
    end
    d = centroids(:,1);
    e = centroids(:,2);
    u = [d, ones(length(d),1)]\e; % Data fitting using least squares
    a=u(1);, b=u(2); % Slope and intercept of best-fit line

    % filtered objects
    objs_f = struct(object_data);

    for f=length(object_data):-1:1
        objs = object_data(f);
        %n_objects = objs.num_objects;
        %for k=1:n_objects
            centroidX = objs.objects.col_mean;
            centroidY = M - objs.objects.row_mean + 1;

            Y = a*centroidX + b;
            % Remove object if distance to track line is greater than threshold
            if abs(Y - centroidY) >= 20

                objs_f(f) = [];
            end

        %end
    end

    X = objs_f;

```

Created file 'C:\Users\cadep\School\CS_6640\A7\CS6640_data_collection.m'.

```

In [354]: %%file CS6640_feature_extraction.m
function X = CS6640_feature_extraction(samples)
% CS6640_feature_extraction - Given data samples, generate feature vectors
% On input:
%     samples (n x 1 struct array) object_data samples
% On output:
%     X (n x m array): n samples of m-tuple feature vectors
% Call:
%     X = CS6640_feature_extraction(samples)
% Author:
%     Cade Parkison

```

```

%      UU
%      Fall 2018
%

n = length(samples); % number of samples
m = 1;               % number of features

X = zeros([n,m]);

for i=1:n

    % Size feature
    num_pixels = samples(i).objects.num_pixels;
    X(i,1) = num_pixels;

end

```

Created file 'C:\Users\cadep\School\CS_6640\A7\CS6640_feature_extraction.m'.

```

In [421]: %%file CS6640_bayes_eval.m
function classes = CS6640_bayes_eval(X, cp, cm)
% CS6640_bayes_eval - gets resulting class for each frame in a data set
% On input:
%      X (nxm array): n samples of m-tuple feature vectors
%      class_probs (1xn vector): probabilities of n classes (sums to 1)
%      class_models (1xm vector struct): class models: means and
%      variances
%      (k).mean (mx1 vector): k_th class mean vector
%      (k).var (mxm array): k_th class covariance matrix
% On output:
%      classes (nx1 array): n detected classes from CS6640_Bayes
% Call:
%      classes = CS6640_bayes_eval(X)
% Author:
%      Cade Parkison
%      UU
%      Fall 2018
%

classes = zeros([length(X), 1]);

for i=1:length(X)
    classes(i) = CS6640_Bayes(X(i),cp,cm);
end

```

Created file 'C:\Users\cadep\School\CS_6640\A7\CS6640_bayes_eval.m'.

Testing

```
In [1]: format compact
```

```
In [150]: %plot native
```

```
In [281]: clear all;close all;clc
```

```
In [341]: car1 = VideoReader('car1.avi');
          car2 = VideoReader('car2.avi');
          car1_ = VideoReader('car1.avi');
          car2_ = VideoReader('car2.avi');

          bus1 = VideoReader('bus1.avi');
          bus2 = VideoReader('bus2.avi');
          bus1_ = VideoReader('bus1.avi');
          bus2_ = VideoReader('bus2.avi');

          truck1 = VideoReader('truck1.avi');
          truck2 = VideoReader('truck2.avi');
          truck1_ = VideoReader('truck1.avi');
          truck2_ = VideoReader('truck2.avi');
```

Car:

```
In [283]: M1 = CS6640_MM(car1);
```

```
In [284]: M2 = CS6640_MM(car2);
```

```
In [342]: c1_data = CS6640_object_data(M1,car1_);
          c2_data = CS6640_object_data(M2,car2_);
          size(c1_data),size(c2_data)
```

```
ans =
     1    177
ans =
     1    226
```

Truck:

```
In [338]: M3 = CS6640_MM(truck1);
```

```
In [339]: M4 = CS6640_MM(truck2);
```

```
In [343]: t1_data = CS6640_object_data(M3, truck1_);  
          t2_data = CS6640_object_data(M4, truck2_);  
          size(t1_data), size(t2_data)
```

```
ans =  
     1    175  
ans =  
     1    167
```

Bus:

```
In [340]: M5 = CS6640_MM(bus1);
```

```
In [347]: M6 = CS6640_MM(bus2);
```

```
In [351]: b1_data = CS6640_object_data(M5, bus1_);  
          b2_data = CS6640_object_data(M6, bus2_);  
          size(b1_data), size(b2_data)
```

```
ans =  
     1    243  
ans =  
     1    297
```

Data Collection and filtering:

```
In [348]: c1_objs = CS6640_data_collection(c1_data);  
          c2_objs = CS6640_data_collection(c2_data);  
          size(c1_objs), size(c2_objs)
```

```
ans =  
      1    116  
ans =  
      1    213
```

```
In [349]: t1_objs = CS6640_data_collection(t1_data);  
          t2_objs = CS6640_data_collection(t2_data);  
          size(t1_objs), size(t2_objs)
```

```
ans =  
      1    138  
ans =  
      1    167
```

```
In [352]: b1_objs = CS6640_data_collection(b1_data);  
          b2_objs = CS6640_data_collection(b2_data);  
          size(b1_objs), size(b2_objs)
```

```
ans =  
      1    243  
ans =  
      1    257
```

Generate Feature Vectors:

```
In [355]: car_samples = [c1_objs, c2_objs];  
          Xc = CS6640_feature_extraction(car_samples);
```

```
In [356]: truck_samples = [t1_objs, t2_objs];  
          Xt = CS6640_feature_extraction(truck_samples);
```

```
In [357]: bus_samples = [b1_objs, b2_objs];  
          Xb = CS6640_feature_extraction(bus_samples);
```

```
In [358]: size(Xc),size(Xt),size(Xb)
```



```
ans =  
    329    1  
ans =  
    305    1  
ans =  
    500    1
```

Build Models:

```
In [359]: Mc = CS6640_build_Bayes(Xc);  
          Mt = CS6640_build_Bayes(Xt);  
          Mb = CS6640_build_Bayes(Xb);  
  
          P = [1/3  1/3  1/3];  
          M = [Mc, Mt, Mb];
```

```
In [361]: Mc.mean,Mt.mean,Mb.mean
```

```
ans =  
    931.2401  
ans =  
    2.0942e+03  
ans =  
    3.8603e+03
```

```
In [362]: Mc.var,Mt.var,Mb.var
```

```
ans =  
    2.5207e+05  
ans =  
    9.1540e+05  
ans =  
    7.0009e+06
```

Plot model data:

```
In [376]: plot(x,y1,'-')
```

```
In [444]: clf
```

```
In [472]: clf;
hold on;
y1 = normpdf(x,Mc.mean, Mc.var);

plot(x,y1,'-','DisplayName','Car');
```

```
In [471]: clf;
hold on;

x = linspace(-30000, 40000);

y1 = normpdf(x,Mc.mean, Mc.var);
y2 = normpdf(x,Mt.mean, Mt.var);
y3 = normpdf(x,Mb.mean, Mb.var);

plot(x,y1,'b','DisplayName','Car');
plot(x,y2,'r','DisplayName','Truck');
plot(x,y3,'g','DisplayName','Bus');

scatter(Mc.mean,0,'Marker','x','DisplayName','Car mean');
scatter(Mt.mean,0,'Marker','o','DisplayName','Truck mean');
scatter(Mb.mean,0,'Marker','*','DisplayName','Bus mean');

legend()

xlabel('x')
ylabel('f(x/mu,sigma)')
title('Probability Distributions of each Class')
```

Evaluation:

```
In [404]: c3 = VideoReader('car3.avi');
t3 = VideoReader('truck3.avi');
b3 = VideoReader('bus3.avi');
```

```
c3_ = VideoReader('car3.avi');  
t3_ = VideoReader('truck3.avi');  
b3_ = VideoReader('bus3.avi');
```

```
In [397]: MMc = CS6640_MM(c3);
```

```
In [398]: MMt = CS6640_MM(t3);
```

```
In [405]: MMb = CS6640_MM(b3);
```

```
In [410]: c3_data = CS6640_object_data(MMc,c3_);  
c3_objs = CS6640_data_collection(c3_data);  
Xc3 = CS6640_feature_extraction(c3_objs);
```

```
In [407]: t3_data = CS6640_object_data(MMt,t3_);  
t3_objs = CS6640_data_collection(t3_data);  
Xt3 = CS6640_feature_extraction(t3_objs);
```

```
In [408]: b3_data = CS6640_object_data(MMb,b3_);  
b3_objs = CS6640_data_collection(b3_data);  
Xb3 = CS6640_feature_extraction(b3_objs);
```

```
In [413]: length(Xc3)
```

```
ans =  
112
```

Eval of Training Data:

```
In [414]: X = Xc;
```

```
ans =  
    116      1
```

```
In [416]: cp = [1/3  1/3  1/3];  
          cm = [Mc Mt Mb];
```

```
In [423]: c_classes_t = CS6640_bayes_eval(Xc, cp, cm);  
          t_classes_t = CS6640_bayes_eval(Xt, cp, cm);  
          b_classes_t = CS6640_bayes_eval(Xb, cp, cm);
```

Correct Percentages:

```
In [426]: car = ones([length(Xc), 1]);  
          truck = 2*ones([length(Xt), 1]);  
          bus = 3*ones([length(Xb), 1]);
```

```
In [428]: sum(c_classes_t == car)/length(Xc)
```

```
ans =  
    0.8936
```

```
In [451]: size(c_classes_t)
```

```
ans =  
    329      1
```

```
In [450]: sum(c_classes_t==1)
```

```
ans =  
    294
```

```
In [452]: sum(c_classes_t==2)
```

```
ans =  
35
```

```
In [453]: sum(c_classes_t==3)
```

```
ans =  
0
```

```
In [430]: sum(t_classes_t == truck)/length(Xt)
```

```
ans =  
0.5803
```

```
In [454]: sum(t_classes_t==1)
```

```
ans =  
96
```

```
In [455]: sum(t_classes_t==2)
```

```
ans =  
177
```

```
In [456]: sum(t_classes_t==3)
```

```
ans =  
32
```

```
In [431]: sum(b_classes_t == bus)/length(Xb)
```

```
ans =  
0.4820
```

```
In [457]: sum(b_classes_t==1)
```

```
ans =  
    108
```

```
In [458]: sum(b_classes_t==2)
```

```
ans =  
    151
```

```
In [459]: sum(b_classes_t==3)
```

```
ans =  
    241
```

Evaluation on remaining videos:

```
In [432]: c3_classes = CS6640_bayes_eval(Xc3, cp, cm);  
          t3_classes = CS6640_bayes_eval(Xt3, cp, cm);  
          b3_classes = CS6640_bayes_eval(Xb3, cp, cm);
```

```
In [433]: car3 = ones([length(Xc3), 1]);  
          truck3 = 2*ones([length(Xt3), 1]);  
          bus3 = 3*ones([length(Xb3), 1]);
```

```
In [434]: sum(c3_classes == car3)/length(Xc3)
```

```
ans =  
    0.9286
```

```
In [460]: sum(c3_classes == 1)
```

```
ans =  
    104
```

```
In [461]: sum(c3_classes == 2)
```

```
ans =  
      8
```

```
In [462]: sum(c3_classes == 3)
```

```
ans =  
      0
```

```
In [435]: sum(t3_classes == truck3)/length(Xt3)
```

```
ans =  
      0.2991
```

```
In [463]: sum(t3_classes == 1)
```

```
ans =  
      48
```

```
In [464]: sum(t3_classes == 2)
```

```
ans =  
      67
```

```
In [465]: sum(t3_classes == 3)
```

```
ans =  
      109
```

```
In [436]: sum(b3_classes == bus3)/length(Xb3)
```

```
ans =  
      0.8941
```

```
In [466]: sum(b3_classes == 1)
```

```
ans =  
    0
```

```
In [467]: sum(b3_classes == 2)
```

```
ans =  
    27
```

```
In [468]: sum(b3_classes == 3)
```

```
ans =  
    228
```