CS 441

HW 3: PDFs and Outliers

Due Date: Mar 4, 2024 11:59pm

In this assignment we will explore methods to estimate probability functions and to robustly estimate statistics in the presence of corrupted or missing data.

The aims of this homework are:

1. Be able to estimate probability functions using several methods: per-feature 1D histograms, clustering for joint histograms, and mixture of Gaussian models.
2. Be able to estimate statistics, such as mean, standard deviation, min, and max while being robust to data values that are incorrect or missing.

Read this document and the report template and tips and tricks before beginning to code.

- Report template
- Tips and Tricks
- Starter code

The assignments will not change, but we may update this document to improve clarity in response to student questions.

1. Estimating PDFs [50 points]

A basic problem in computer vision and graphics is being able to segment out an object based on a rough indication from a user. In this case, given a bounding box around an object, we want to identify which pixels correspond to the intended object.

To do this, we compute two probabilities: (1) $P(rgb_i|image)$, the probability of the ith pixel's color (RGB values) given that it comes from the image; and (2) $P(rgb_i|box)$, the probability of the ith pixel's color (RGB values) given that it comes from inside the bounding box. We can then assign a score for whether a pixel is part of the object:

$$score = log[P(rgb_i|box)/P(rgb_i|image)]$$

This gives us a good pixelwise score for segmenting out the object inside the box. To keep it simple, we will compute the box color density $P(rgb_i|box)$ using a cropped image (which prevents having to record and use the box coordinates). We've included one example from Flickr.

We will experiment with three methods of estimating probability density functions (PDFs). Estimate the $P(rgb_i|box)$ from the **crop image** and $P(rgb_i|image)$ from the **full image** using three methods:

1. Estimate the probability of each color channel separately using discrete pdfs (by counting) and then model the joint probability as the product of per-channel probabilities. Try anywhere from 2 to 256 bins per channel.
2. Estimate the joint probability by k-means clustering the pixel colors in the whole image and then estimating the probability of each cluster. Try varying the number of clusters , e.g. between 16 and 512.
3. Estimate the joint probability using a Gaussian mixture model (GMM). Try varying the number of Gaussian components (e.g. between 1 and 10) and whether using full covariance or diagonal covariance. Use `sklearn.mixture.GaussianMixture`.

For each method, display the image, the score map, thresholded score map, and a thresholded image with some threshold of your choice (typically in the range of -2 to 2). Helper code is provided for these displays. Also report your parameters (e.g. number of bins, K, number of Gaussian components, covariance matrix type). Each method can provide good but imperfect segmentations.

2. Robust Estimation [50 points]

The **corrupted salary dataset** has three variables: salary, years, school. Salary is the reported salary of each person. Years is the number of years of experience in the job. School is the university where the person last had a degree. For the core assignment, we'll only use salary, and the stretch goals will use the other two variables. Some of the reported salary information is wrong (some incorrect value is provided), so we want to learn things from the data in a way that is robust to the wrong data. We refer to correctly entered data as "valid".

Estimate the true mean, standard deviation, min, and max of the salaries using three different methods:

1. **Assume no noise**. Compute the statistics for the data as a whole.
2. **Use percentiles**. Assume valid data will fall between the 5th and 95th percentile. Adjust estimates of the min and max by assuming that the valid data has a uniform distribution (see lecture on robust fitting).
3. **Use EM**. Assume valid data follows a Gaussian distribution, while the wrong data has a uniform distribution between the minimum and maximum value of salary. For mean and std, report the estimated mean and std of the valid salary distribution. For min and max, report the min and max salaries that have greater than 50% chance of being valid. Also report the estimated probability that a random sample is valid, and the first five indices of salaries that are not likely to be valid.

3. Stretch Goals

For the salary problem, we will assume that each school has a different mean base salary, salaries from all schools have the same standard deviation, and that each year of experience has an expected increase in salary.

a. Unfortunately, some of the school information is missing. Use EM to estimate the probability of the school for each missing value, and report the estimated mean salary for each school. [20 points]
b. Presumably more years of experience increases the salary. Estimate the expected increase in salary per year of experience in a way that is robust to noise and accounts for the school. [20 points]

Mutual information between two variables x and y can be computed based on the marginal and joint probabilities of those variables:

$$I(x,y) = \frac{1}{N}\sum_{n} \log \frac{P(x,y)}{P(x)P(y)}$$

Our estimate of I(x,y) depends on how we estimate P(x), P(y), and P(x,y). For this problem, we'll estimate mutual information of some pairs of variables in the Diabetes dataset:

c. Estimate the mutual information of sex with age by treating both age and sex as discrete variables. [10 points]
d. Estimate the mutual information of sex and age using a mixture of three Gaussian components for P(age | sex=1) and P(age | sex=2). [10 points]

Submission Instructions

Append two files: (1) completed report template; (2) a pdf version of your Jupyter notebook. Be sure to include your name and acknowledgments. **The report is your primary deliverable**. The notebook pdf is included for verification/clarification by the grader. Submit the combined file to Gradescope.

To create PDF of notebook: Use "jupyter nbconvert" -- see starter code.

To combine PDFs: use https://combinepdf.com/ or another free merge tool.