CS 441

HW 3: PDFs and Outliers

Due Date: Oct 13, 2025 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. Spam Detection with Naive Bayes Classifier [50 points]

We want to classify text messages as "spam" (unwanted) or "ham" (genuine). We will use data (**spam.csv**) from the Kaggle SMS spam dataset. We've provided the loading and pre-processing code to generate:

- unique_words: the unique set of words in the dataset
- (`x_train`, `y_train`, `msg_train`): counts of words in each message, spam (y=1) or not spam (y=-1) labels, and the message string for each training sample; `x_train[n][j]` is the count of the *j*th word in the *n*th sample.
- `*_val` and `*_test`, similar to above, for the val and test splits

We will use a Naive Bayes Classifier.

- The spam score is log P(message & spam) - log P(message & ham):

  $$spamScore(x_n) = log\, P(y = 1, x_n) - log\, P(y = -1, x_n)$$

- The log of P(y,x) can be computed using the Naive Bayes assumption, that each word is conditionally independent of the other words given the label. E.g., for spam:

  $$log\, P(y = 1, x_n) = logP(y = 1) + \Sigma_j logP(w_j|spam)x_{n,j}$$

- The probability of a word given spam can be computed by counting how many times the word appears in the spam messages and how many words in total are in the spam

messages. We add α to each word count as a kind of prior or regularizer, which is like pretending that each word occurred an additional α times in the spam messages.

$P(w_j|spam) = (\alpha + \Sigma_n \delta(y_n = 1)\, x_{n,j}) / (\alpha|w| + \Sigma_j \Sigma_n \delta(y_n = 1)\, x_{n,j}))$

- The equations for ham (y=-1) are similar, and the same α value should be used.

1. **Train your Naive Bayes Classifier** (i.e. P(y), P(w|y)) using the **train** set with α=1 and compute the accuracy on the **val** set. You should get P(y=1)=0.142, P(call|spam) = 0.0104, and P(call|ham)=0.0029. Your validation accuracy should be higher than 95%. Do not use sklearn's Naive Bayes models.

2. **Data exploration**. What are the 10 spammiest words (i.e. words with highest $logP(w_j|spam)$- $logP(w_j|ham)$)? What are the 10 hammiest words? Which **val** message is the spammiest ham (message with highest spam score but y=-1)? Which is hammiest spam (message with lowest spam score but y=1)? Spammiest spam? Hammiest ham?

3. **Precision-recall trade-off.** You want to flag spam messages with minimal false positives. Using the **val** set, compute precision/recall and display the PR curve. Programmatically, find the score threshold with highest recall, where precision > 0.99. Report the accuracy, precision, and recall on the **test** set using the same model and the selected threshold.

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 [50 points]

a. For spam detection, try to improve your recall on the test set, still using a threshold that gives at least 99% precision on the val set. You can try adjusting α, using different classifiers, or using different parameters in `CountVectorizer`, which is called in the starter code using a simple setting. Maybe it would help to consider all numbers the same, etc. Points are awarded based on what is tried [5-15 pts] and whether successful [5 points].

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. These stretch goals are difficult for many students.

b. 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. [15 points]
c. 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. [15 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.