ELEN 4720: Machine Learning for Signals, Information and Data

Columbia University, Fall 2022

Homework 2: Due Friday October 28, 2022 by 5:00PM

Problem 1

In this problem you will derive a naive Bayes classifier. For a labeled set of data $(y_1, x_1), \dots, (y_n, x_n)$, where for this problem $y \in \{0, 1\}$ and x is a D-dimensional vector of counts, the Bayes classifier observes a new x_0 and predicts y_0 as

$$y_0 \;=\; \arg\max_y \; p(y_0 = y|\pi) \prod_{d=1}^{D} p(x_{0,d}|\lambda_{y,d}).$$

The distribution $p(y_0 = y|\pi) = \text{Bernoulli}(y|\pi)$. What is "naive" about this classifier is the assumption that all D dimensions of x are independent. Assume that each dimension of x is Poisson distributed with a Gamma prior. The full generative process is

$$\text{Data: } y_i \overset{iid}{\sim} \text{Bern}(\pi), \quad x_{i,d}|y_i \sim \text{Pois}(\lambda_{y_i,d}),\ d = 1, \dots, D \qquad \text{Prior: } \lambda_{y,d} \overset{iid}{\sim} \text{Gamma}(2, 1)$$

Derive the solution for π and each $\lambda_{y,d}$ by maximizing

$$\widehat{\pi}, \widehat{\lambda}_{0,1:D}, \widehat{\lambda}_{1,1:D} \;=\; \arg \max_{\pi, \widehat{\lambda}_{0,1:D}, \widehat{\lambda}_{1,1:D}} \; \sum_{i=1}^{n} \ln p(y_i|\pi) + \sum_{d=1}^{D} \left(\ln p(\lambda_{0,d}) + \ln p(\lambda_{1,d}) + \sum_{i=1}^{n} \ln p(x_{i,d}|\lambda_{y_i,d}) \right).$$

Please separate your derivations as follows:

(a) Derive $\widehat{\pi}$ using the objective above.

(b) Derive $\widehat{\lambda}_{y,d}$ using the objective above, leaving y and d arbitrary in your notation.

Problem 2

In this problem you will implement the naive Bayes classifier derived in Problem 1, as well as the k-NN algorithm. The data consists of examples of spam and non-spam emails, of which there are 4600 labeled examples. The feature vector x is a 54-dimensional vector extracted from the email and $y = 1$ indicates a spam email.[1]

In every experiment below, *randomly* partition the data into 10 groups and run the algorithm 10 different times so that each group is held out as a test set one time. The final result you show should be the cumulative result across these 10 groups.

[1] We have preprocessed the data. The original data is at `https://archive.ics.uci.edu/ml/datasets/Spambase`. More information about the meanings of the 54 dimensions of the data is provided in two accompanying files.

(a) Implement the naive Bayes classifier described above. In a 2×2 table, write the number of times that you predicted a class y data point (ground truth) as a class y' data point (model prediction) in the (y, y')-th cell of the table, where y and y' can be either 0 or 1. There should be four values written in the table in your PDF. Next to your table, write the prediction accuracy—the sum of the diagonal divided by 4600. (The sum of all entries in the table should be 4600.)

(b) In one figure, show a stem plot (`stem()` in Matlab) of the 54 Poisson parameters for each class averaged across the 10 runs. (This average is only used for plotting purposes on this homework. In practice you would relearn these parameters using the entire data set to find their final values.) Use the README file to make an observation about dimensions 16 and 52.

(c) Implement the k-NN classifier for $k = 1, \ldots, 20$. Use the ℓ_1 distance for this problem. Plot the prediction accuracy as a function of k.

Problem 3

In this problem you will implement the Gaussian process model for regression. You will use the same data used for homework 1 to do this, which is again provided in the data zip file for this homework. Recall that the Gaussian process treats a set of N observations $(x_1, y_1), \ldots, (x_N, y_N)$, with $x_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$, as being generated from a multivariate Gaussian distribution as follows,

$$y \sim \text{Normal}(0, \sigma^2 I + K), \quad K_{ij} = K(x_i, x_j) \quad \Big(\text{use: } \exp\Big\{ -\frac{1}{b}\|x_i - x_j\|^2 \Big\}\Big).$$

Here, y is an N-dimensional vector of outputs and K is an $N \times N$ kernel matrix. For this problem use the Gaussian kernel indicated above. In the lecture slides, we discuss making predictions for a new y' given x', which was Gaussian with mean $\mu(x')$ and variance $\Sigma(x')$. The equations are shown in the slides.

There are two parameters that need to be set for this model as given above, σ^2 and b.

a) Write code to implement the Gaussian process and to make predictions on test data.

b) For $b \in \{5, 7, 9, 11, 13, 15\}$ and $\sigma^2 \in \{.1, .2, .3, .4, .5, .6, .7, .8, .9, 1\}$—so 60 total pairs (b, σ^2)—calculate the RMSE on the 42 test points as you did in the first homework. Use the mean of the Gaussian process at the test point as your prediction. Show your results in a table.

c) Which value was the best and how does this compare with the first homework? What might be a drawback of the approach in this homework (as given) compared with homework 1?

d) To better understand what the Gaussian process is doing through visualization, re-run the algorithm by using *only* the 4th dimension of x_i (car weight). Set $b = 5$ and $\sigma^2 = 2$. Show a scatter plot of the data (x[4] versus y for each point). Also, plot as a solid line the predictive mean of the Gaussian process at each point *in the training set*. You can think of this problem as asking you to create a test set by duplicating x_i[4] for each i in the training set and then to predict that test set.

Homework #2   Michelle Campoli (mec2308)

Problem 1

(a) Given x_0

$$p(y_0 = 0) \quad p(x_1 | y = 0)$$

$$p(y_0 = 1) \quad p(x_0 | y = 1)$$

Assume parametric family of models

$$\left.\begin{array}{l} p(x_0 | y = 0, \theta_0) \\ p(x_0 | y = 1, \theta_1) \end{array}\right. \quad \lambda \quad \left.\right\} \text{estimate from training data}$$

$$\begin{bmatrix} x_{01} \\ x_{02} \\ \vdots \\ x_{0D} \end{bmatrix}$$

&

$$p(y_0 = 0) \quad \hat{\pi}_0$$

$$p(y_0 = 1) \quad \hat{\pi}_1$$

estimate from data = "plug-in estimator"

Naive Bayes: assume independence between features
reduces model complexity

$$p(x_0 | y_0 = 0) = \prod_{d=1}^{D} p(x_{0,d} | y = 0)$$

Bayes : $\hat{y}_0 = \underset{y}{\operatorname{argmax}}\ \underset{\text{prior}}{p(y_0 = y)}\ \boxed{p(x_0 | y_0 = y)}$

$\downarrow$ likelihood

Naive Bayes : $\hat{y}_0 = \underset{y}{\operatorname{argmax}}\ \underline{p(y_0 = y)}\ \boxed{\prod_{d=1}^{D} p(x_{0d} | y_0 = y)}$

class probabilities π_0, π_1

Maximum Likelihood

$$\max_{\pi, \lambda}\ p(x_1, y_1 \dots x_n, y_n | \pi, \lambda)$$

assume (x_i, y_i) independent

$$\prod_{i=1}^{n} p(x_i, y_i | \pi, \lambda)$$

$$= \prod_{i=1}^{n} p(y_i | \pi)\, p(x_i | y_i, \lambda)$$

$$\max_{\pi, \lambda} \left(\prod_{i=1}^{n} p(y_i | \pi) \prod_{d=1}^{D} p(x_{id} | y_i, \lambda)) \right)$$

log of products = sum of logs

$$\max_{\pi, \lambda} \sum_{i=1}^{n} \underbrace{\log p(y_i | \pi)}_{\pi} + \underbrace{\sum_{d=1}^{D} \log(x_{id} | y_i, \lambda)}_{\lambda_{y_i, d}}$$

MLE

$$\max_{\pi_0,\pi_1} \log(\text{Data}|\text{parameters})$$

$$\max_{\pi_0,\pi_1} \sum_{i=1}^{n} \log p(y_i|\pi)$$

π_0 $y_i=0$

π_1 $y_i=1$

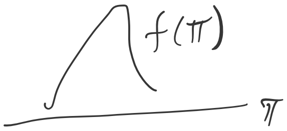
n_0 $y_i=0$ ......

n_1 $y_i=1$ ...—

$$\max_{\pi_0,\pi_1} n_0\log\pi_0 + n_1\log\pi_1$$

$$\max_{\pi_0} n_0\log(\pi_0) + n_1\log(1-\pi)$$

gradient = 0



$$\frac{n_0}{\pi_0} - \frac{n_1}{(1-\pi_0)} = 0$$

$$\frac{n_0}{\hat{\pi}_0} = \frac{n_1}{1-\hat{\pi}_0}$$

$$n_0(1-\hat{\pi}_0) = n_1\hat{\pi}_0$$

$$(n_0+n_1)\hat{\pi}_0 = n_0$$

$$\hat{\pi}_0 = \frac{n_0}{n_0+n_1} \nearrow n$$

$$\hat{\pi}_0 = \frac{n_0}{n} \qquad \hat{\pi}_1 = \frac{n_1}{n}$$

(b) find $\hat{\lambda}_{0,1:D}$ $\hat{\lambda}_{1,1:D}$ for arbitrary y + d

$$\sum_{d=1}^{D}\left(\ln p(\lambda_{0,d}) + \ln p(\lambda_{1,d}) + \sum_{i=1}^{n} \ln p(x_{i,d} \mid \lambda_{y_i,d})\right)$$

$$\text{Gamma}(2,1) = \frac{\cancel{1^2} \cdot \lambda^1 \cdot e^{-\lambda}}{\cancel{\Gamma(2)}}$$

$$\text{Poisson} \quad p(x_{i,d} \mid \lambda_{y_i,d}) = \frac{(\lambda_{y_i,d})^{x_{i,d}}\, e^{-(\lambda_{y_i,d})}}{x_{i,d}!}$$

$$\underset{\lambda_{i,1:D}}{\text{argmax}} \sum_{d=1}^{D}\left(\ln \frac{\lambda_{0,d}\, e^{-\lambda_0}}{\Gamma(2)} + \ln \frac{\lambda_{1,d}\, e^{-\lambda_1}}{\Gamma(2)} + \sum_{i=1}^{n} \ln\left(\frac{\lambda_{y_i,d}^{x_{i,d}}\, e^{-(\lambda_{y_i,d})}}{x_{i,d}!}\right)\right)$$

$$\underset{\lambda_{0,1:D}}{\overset{y_i=0}{\max}} \sum_{d=1}^{D}\left(\ln\left(\frac{\lambda_{0,d}\, e^{-\lambda_0}}{\Gamma(2)}\right) + \ln\left(\frac{\lambda_{1,d}\, e^{-\lambda_1}}{\Gamma(2)}\right) + \sum_{i=1}^{n} \ln\left(\frac{\lambda_{0,d}^{x_{i,d}}\, e^{-(\lambda_{0,d})}}{x_{i,d}!}\right)\right)$$

$$\max_{\lambda_{0,1:D}} \sum_{d=1}^{D}\left(\ln(\lambda_{0,d}) - \lambda_{0,d} + \ln(\lambda_{1,d}) - \lambda_{1,d} + \sum_{i=1}^{n}\left(x_{i,d}\ln(\lambda_{0,d}) - (\lambda_{0,d}) - \ln(x_{i,d}!)\right)\right)$$

$$\nabla_{\lambda_{0,1:D}} = \sum_{d=1}^{D}\left(\frac{1}{\lambda_{0,d}} - 1 + 0 - 0 + \sum_{i=1}^{n}\left(\frac{x_{i,d}}{\lambda_{0,d}} - 1 - 0\right)\right)$$

$$\nabla\lambda_{0,1:D} = \sum_{d=1}^{D}\left(\frac{1}{\lambda_{0,d}} - 1 + \sum_{i=1}^{n}\left(\frac{x_{i,d}}{\lambda_{0,d}} - 1\right)\right) = 0$$

$$\sum_{d=1}^{D}\left(\frac{1}{\lambda_{0,d}} - 1 + \frac{1}{\lambda_{0,d}}\sum_{i=1}^{n} x_{i,d} - \sum_{i=1}^{n} 1\right) = 0$$

$y_i = 1$

$$\nabla_{\lambda_{1,1:D}} = \sum_{d=1}^{D}\left(0 - 0 + \frac{1}{\lambda_{1,d}} - 1 + \sum_{i=1}^{n}\left(\frac{x_{i,d}}{\lambda_{1,d}} - 1 - 0\right)\right)$$

$$\nabla_{\lambda_{1,1:D}} = \sum_{d=1}^{D}\left(\frac{1}{\lambda_{1,d}} - 1 + \sum_{i=1}^{n}\left(\frac{x_{i,d}}{\lambda_{1,d}} - 1\right)\right)$$

$$\nabla_{\lambda_{1,1:D}} = \sum_{d=1}^{D}\left(\frac{1}{\lambda_{1,d}} - 1 + \frac{1}{\lambda_{1,d}}\sum_{i=1}^{n} x_{i,d} - \sum_{i=1}^{n} 1\right) = 0$$

for $d = j$, arbitrary d

$$\frac{1}{\lambda_{1,j}} - 1 + \frac{1}{\lambda_{1,j}}\sum_{i=1}^{n} x_{i,j} - \sum_{i=1}^{n} 1 = 0$$

$$\frac{1}{\lambda_{1,j}}\left(1 + \sum_{i=1}^{n} x_{i,j}\right) - 1 - \sum_{i=1}^{n} 1 = 0$$

$$\frac{1}{\lambda_{1,j}}\left(1 + \sum_{i=1}^{n} x_{ij}\right) = 1 + \sum_{i=1}^{n} 1$$

$$\frac{1 + \sum_{i=1}^{n} x_{ij}}{1 + \cancel{\sum_{i=1}^{n} 1}\; n} = \lambda_{1,j} \Rightarrow \frac{1 + \sum_{i=1}^{n} x_{i,j}}{1 + n} = \lambda_{1,j}$$

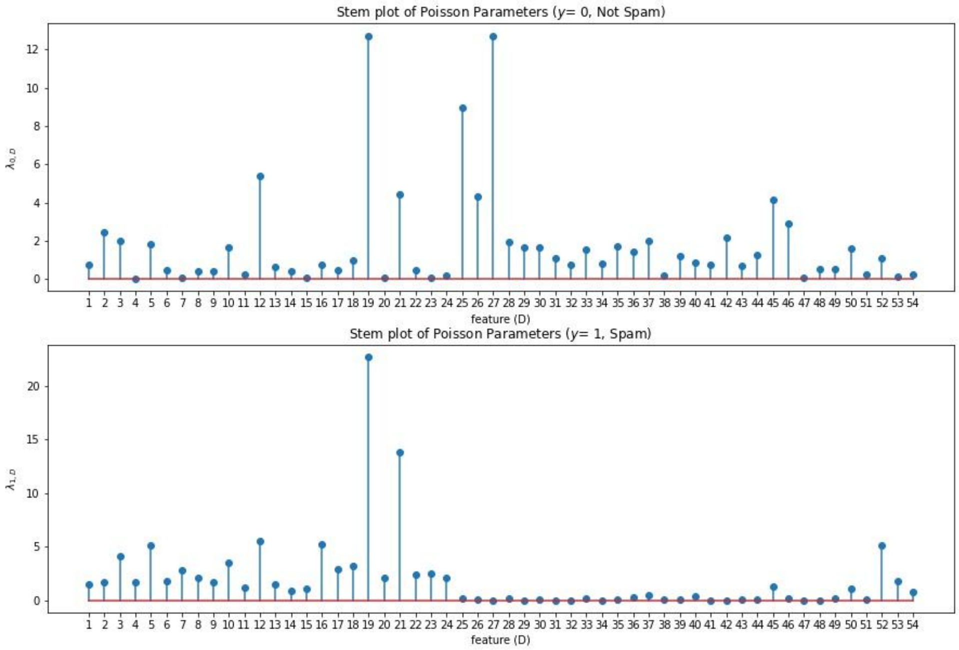
$$\frac{1 + \sum_{i=1}^{n} x_{i,j}}{1 + n} = \lambda_{0,j}$$

Problem 2

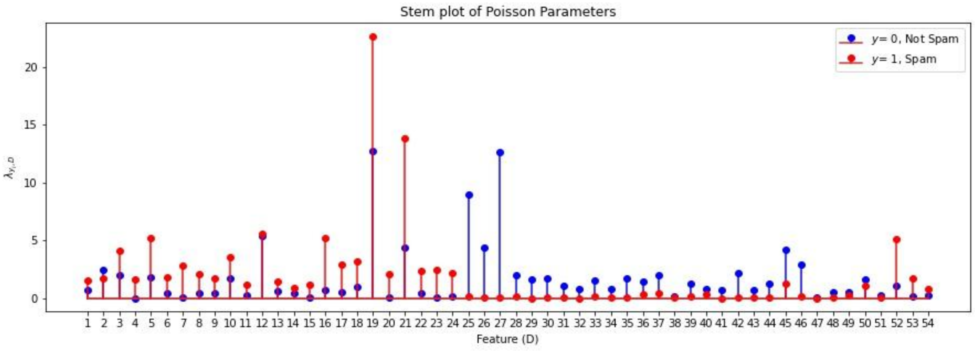
(a) Confusion Matrix:

		Actual Values	
		y=1	y=0
Predicted Values	y=1	1705	491
	y-0	108	2296

$$Prediction\ Accuracy\ = \frac{Sum\ of\ diagonal}{4600} = \frac{1705 + 2296}{4600} = 0.86978$$

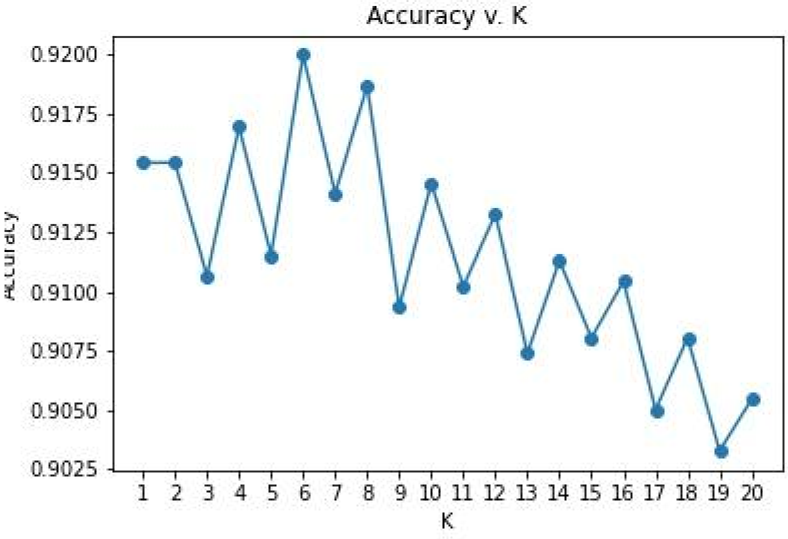


(b)



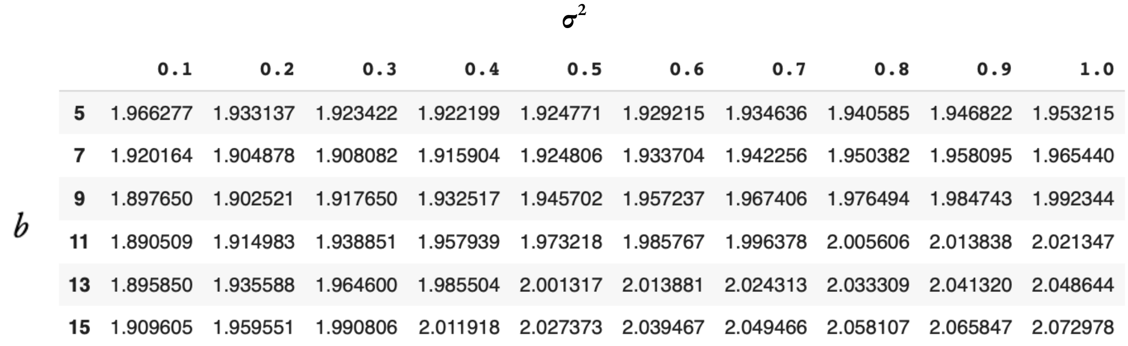
Dimensions 16 - 54 show obvious discrepancies in λ for classifying y=0 and y=1.
Features 25-50 are more likely to classify an email as not spam if they are present, since λ is higher for class y=0. The most prominent are features: 27 = “george”, 26 = “hpl”, 25 = “hp”. Then come features 28-50: 650, lab, labs, telnet, 857, data, 415, 85, technology, 1999, parts, pm, direct, cs, meeting, original, project, re, edu, table, conference, ;, (,].
Features 16 - 24 and 52-54 are more likely to classify an email as spam if they are present, since λ is higher for class y=1. The most prominent features are: 16 = “free”, 19 = “you”, 21 = “ your”, 52 = “!”. The other less prominent features are: money, font, 000, email, business, $, #.
This makes sense logically, as scam emails usually do not refer to a person by name (i.e. “george”) but instead by “you” or “your”. Scam emails also try to impart some urgency by including exclamation points. Non-scam emails also generally refer to technicalities like a lab, a meeting, a conference, or a certain time (i.e. 4:15) rather than money, credit or a monetary value.

(c)



Problem 3

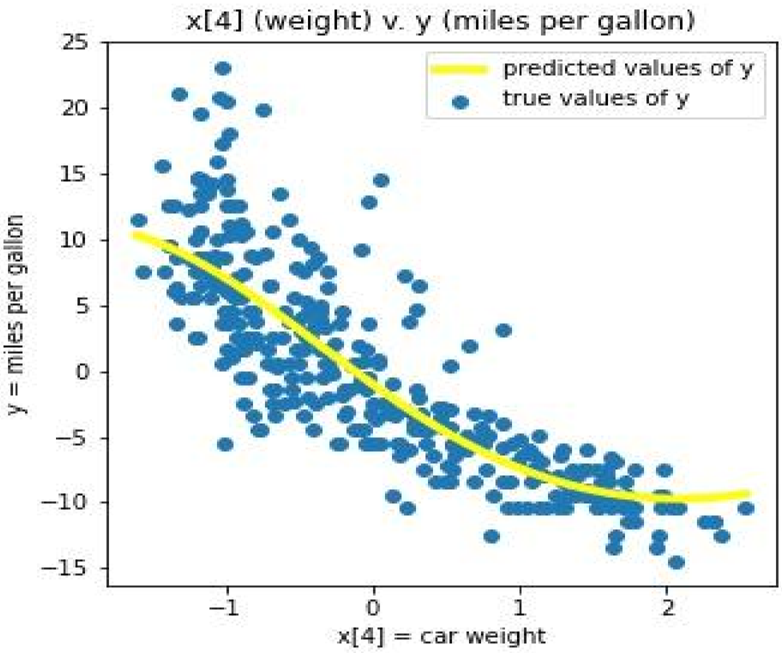
(b) Table: Root Mean Squared Error (RMSE) values



		σ^2									
		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
b	5	1.966277	1.933137	1.923422	1.922199	1.924771	1.929215	1.934636	1.940585	1.946822	1.953215
	7	1.920164	1.904878	1.908082	1.915904	1.924806	1.933704	1.942256	1.950382	1.958095	1.965440
	9	1.897650	1.902521	1.917650	1.932517	1.945702	1.957237	1.967406	1.976494	1.984743	1.992344
	11	1.890509	1.914983	1.938851	1.957939	1.973218	1.985767	1.996378	2.005606	2.013838	2.021347
	13	1.895850	1.935588	1.964600	1.985504	2.001317	2.013881	2.024313	2.033309	2.041320	2.048644
	15	1.909605	1.959551	1.990806	2.011918	2.027373	2.039467	2.049466	2.058107	2.065847	2.072978

(c) Lower values of RMSE indicate a better fit. The best (b, σ^2) pair is: (11,0.1) with RMSE = 1.890509. The lowest RMSE value from homework 1 (giving the best value of λ for ridge regression in that problem) was between 2.1-2.2. A GP regressor is a generative model of the response. This has a few major consequences. While ridge regression can only predict y, a GP can quantify its uncertainty about y. A GP can also generate posterior samples through the generative process. Finally, a GP has a marginal likelihood, and this can be used to find optimal hyperparameters via optimization; in ridge regression, one must search over λ.

(d)



Reference:

G Gunderson. Comparing Kernel Ridge with Gaussian Process Regression. Jan 6 2020.
https://gregorygundersen.com/blog/2020/01/06/kernel-gp-regression/