Homework 4

1. Optimization

1. (derivation for gradient)

   As discussed in the lecture, the logistic regression model is:

$$p(y=1|x,\theta) = \frac{1}{1+\exp(-\theta^T x)}$$

$$p(y=0|x,\theta) = 1 - \frac{1}{1+\exp(-\theta^T x)} = \frac{\exp(-\theta^T x)}{1+\exp(-\theta^T x)}$$

   Given that:

$$\sum_{i=1}^{m} \left((y^i - 1)\theta^T x^i - \log(1+\exp(-\theta^T x^i)) \right)$$

   The gradient is:

$$\nabla \ell(\theta) = \frac{\partial \ell(\theta)}{\partial \theta} = \sum_{i=1}^{m} \left(\frac{\partial}{\partial \theta} \left((y^i - 1)\theta^T x^i \right) - \frac{\partial}{\partial \theta} \left(\log(1+\exp(-\theta^T x^i)) \right) \right)$$

   Derivative of the first part:

$$\frac{\partial}{\partial \theta} \left((y^i - 1)\theta^T x^i \right) = (y^i - 1)x^i$$

   Plug in the logistic regression to get the second part:

$$\frac{\partial}{\partial \theta} \left(\log(1+\exp(-\theta^T x^i)) \right) = -\frac{1}{1+\exp(-\theta^T x^i)} \cdot \exp(-\theta^T x^i) \cdot (-x^i) = \frac{\exp(-\theta^T x^i)x^i}{1+\exp(-\theta^T x^i)}$$

   Combining the above we get the gradient as below:

$$\nabla \ell(\theta) = \sum_{i=1}^{m} (y^i - 1)x^i + \frac{\exp(-\theta^T x^i)x^i}{1+\exp(-\theta^T x^i)}$$

2. The gradient design is to initialize θ_0, then repeat until the accuracy reaches convergence.

   The pseudo code is as following:

   Initialize parameter θ_0, interaction t

   While
$$\|\theta^{t+1} - \theta^t\| > \epsilon$$

   Do:
$$\theta^{t+1} = \theta^t + \gamma\nabla\ell(\theta) = \theta^t + \gamma\sum_{i=1}^{m}(y^i - 1)x^i + \frac{\exp(-\theta^{tT}x^i)x^i}{1+\exp(-\theta^{tT}x^i)}$$

3. Similar to gradient descent, the design of stochastic gradient descent is to initialize θ_0, then repeat until the accuracy reaches convergence.

   The pseudo code is as following:

   Initialize parameter θ_0, interaction t

   While
$$\|\theta^{t+1} - \theta^t\| > \epsilon$$

   Select subset, do:
$$\theta^{t+1} = \theta^t + \gamma\nabla\ell(\theta)$$

   The difference is in stochastic gradient descent, the following steps are taken at each iteration:

   Randomly sample a small subset S of data points; Use the gradient estimated using this small subset of data, instead of the full dataset, to update the model parameters; In each iteration, use a different subset S_t to ensure diverse samples across updates; Continue this process, eventually looping through the entire training data. The process may iterate over the data multiple times if necessary. By doing these, stochastic gradient descent approach is able to reduce the processing time as it runs on smaller datasets.

4. The log-likelihood function for logistic regression is given:
$$\ell(\theta) = \sum_{i=1}^{m}\left\{-\log(1+\exp\{-\theta^T x^i\}) + (y^i - 1)\theta^T x^i\right\}$$

   The gradient is:
$$\sum_{i=1}^{m}(y^i - 1)x^i + \frac{\exp(-\theta^T x^i)x^i}{1+\exp(-\theta^T x^i)}$$

Hessian matrix is the second derivative of the equation, where:

$$\frac{\partial \ell(\theta)}{\partial \theta} = \sum_{i=1}^{m} \left(\frac{x^i}{1 + e^{x^i \theta^T}} + x^i (y^i - 1) \right)$$

$$\frac{\partial^2 \ell(\theta)}{\partial \theta \partial \theta^T} = - \sum_{i=1}^{m} \frac{(x^i)^2 e^{\theta^T x^i}}{(e^{\theta^T x^i} + x^i)^2} < 0$$

The Hessian matrix is less than 0. As discussed in the lecture, it is concave. The problem can be solved when gradient reaches 0 (or near) to achieve a single unique global optimum.

2. Bayes Classifier for spam filtering

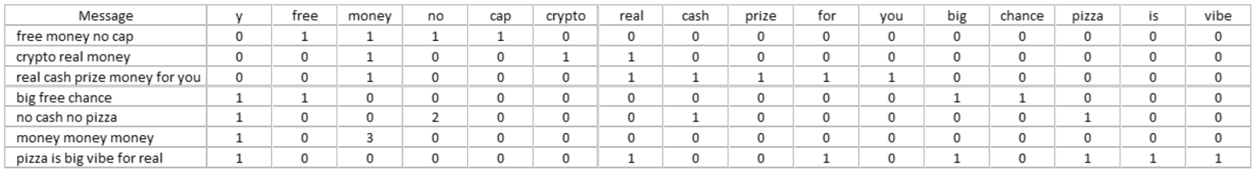
1. Given the 3 spam messages let $y = 0$, and the 4 non-spam messages let $y = 1$, the total is $3 + 4 = 7$

   Therefore the class priors are:

$$P(y = 0) = \frac{3}{7} \approx 0.4286$$

$$P(y = 1) = 1 - P(y = 0) = \frac{4}{7} \approx 0.5714$$

   Feature vectors for spam and non-spam are as shown in Figure 1 below.



Message	y	free	money	no	cap	crypto	real	cash	prize	for	you	big	chance	pizza	is	vibe
free money no cap	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
crypto real money	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0
real cash prize money for you	0	0	1	0	0	0	1	1	1	1	1	0	0	0	0	0
big free chance	1	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0
no cash no pizza	1	0	0	2	0	0	0	1	0	0	0	0	0	1	0	0
money money money	1	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0
pizza is big vibe for real	1	0	0	0	0	0	1	0	0	1	0	1	0	1	1	1

Figure 1: Feature vectors for spam and non-spam

2. To calculate the maximum likelihood estimates is a constrained maximization problem, we can use Lagrange multipliers and use Lagrange function to solve it. First we find the joint probability for the full dataset, then we break it down to two subsets. Then we can add Lagrange multipliers λ_0 and λ_1 for the constraints for classes 0 and 1. Lastly we can take the derivative with respect to 0 to solve θ

   Given the Lagrangian function for joint probability:

$$\mathcal{L} = \sum_{i=1}^{m} \sum_{k=1}^{d} x_k^{(i)} \log \theta_{y^{(i)},k} + \lambda_0 \left(1 - \sum_{k=1}^{d} \theta_{0,k}\right) + \lambda_1 \left(1 - \sum_{k=1}^{d} \theta_{1,k}\right)$$

   Then we break it down with respect to each $\theta_{c,k}$, take the derivatives with respect to the Lagrange multiplier, and solve for θ is 0:

$$\frac{\partial \mathcal{L}}{\partial \theta_{0,k}} = \sum_{i:y^{(i)}=0} \frac{x_k^{(i)}}{\theta_{0,k}} - \lambda_0 = 0 \quad \Rightarrow \quad \theta_{0,k} = \frac{1}{\lambda_0} \sum_{i:y^{(i)}=0} x_k^{(i)}$$

$$\frac{\partial \mathcal{L}}{\partial \lambda_0} = \sum_{k=1}^{d} \theta_{0,k} - 1 = 0$$

$$\lambda_0 = \sum_{k=1}^{d} \sum_{i:y^{(i)}=0} x_k^{(i)}$$

$$\theta_{0,k} = \frac{\sum_{i:y^{(i)}=0} x_k^{(i)}}{\sum_{k=1}^{d} \sum_{i:y^{(i)}=0} x_k^{(i)}}$$

Therefore, $\theta_{0,1}$ free in spam

$$\theta_{0,1} = \frac{x_1^{(1)} + x_1^{(2)} + x_1^{(3)}}{x^{(1)} + x^{(2)} + x^{(3)}} = \frac{1}{13}$$

$\theta_{0,6}$ real in spam

$$\theta_{0,6} = \frac{x_6^{(1)} + x_6^{(2)} + x_6^{(3)}}{x^{(1)} + x^{(2)} + x^{(3)}} = \frac{2}{13}$$

We perform the same for λ_1

$$\theta_{1,k} = \frac{1}{\lambda_1} \sum_{i:y^{(i)}=1} x_k^{(i)} \quad \Rightarrow \quad \sum_{k=1}^{d} \theta_{1,k} - 1 = 0$$

$$\lambda_1 = \sum_{k=1}^{d} \sum_{i:y^{(i)}=1} x_k^{(i)} \quad \Rightarrow \quad \theta_{1,k} = \frac{\sum_{i:y^{(i)}=1} x_k^{(i)}}{\lambda_1}$$

$\theta_{1,2}$ money in non-spam

$$\theta_{1,2} = \frac{0+0+3+0}{16} = \frac{3}{16}$$

$\theta_{1,15}$ vibe in non-spam

$$\theta_{1,15} = \frac{0+0+0+1}{16} = \frac{1}{16}$$

3. For the message money for real, we can calculate the posterior probabilities for spam $y = 0$ and non-spam $y = 1$ using the Naive Bayes classifier.

   From Q2.1 we already get following prior for class 0 and class 1:

$$P(y = 0) = \frac{3}{7}$$

$$P(y = 1) = \frac{4}{7}$$

   Posterior for spam $y = 0$ is prior multiplies class 0 likelihood, as below:

$$P(x|y = 0)P(y = 0) = \left(\theta_{0,2}^{x_2} \cdot \theta_{0,6}^{x_6} \cdot \theta_{0,9}^{x_9}\right) \cdot P(y = 0)$$
$$= \left(\frac{3}{13} \cdot \frac{2}{13} \cdot \frac{1}{13}\right) \cdot \frac{3}{7} = \frac{18}{15379} \approx 0.001170427$$

Posterior for non-spam $y = 1$ is:

$$P(x|y = 1)P(y = 1) = \left(\theta_{1,2}^{x_2} \cdot \theta_{1,6}^{x_6} \cdot \theta_{1,9}^{x_9}\right) \cdot P(y = 1)$$

$$= \left(\frac{3}{16} \cdot \frac{1}{16} \cdot \frac{1}{16}\right) \cdot \frac{4}{7} = \frac{12}{28672} \approx 0.000418527$$

Because the posterior probabilities of spam $y = 0$ are greater than non-spam $y = 1$, we classify "money for real" as spam.

3. Comparing classifiers: Divorce classification/prediction (30 points)

1. We performed accuracy analysis using Python, and the result for testing dataset is as below:

   Naive Bayes testing accuracy is 0.9706

   Logistic Regression testing accuracy is 0.9118

   KNN testing accuracy is 0.9706

   Both Naive Bayes and KNN showed equally higher accuracy, while logistic regression shows lower accuracy result. This may be due to that the marriage data's characteristics tend to be non-linear classification. However, because the testing size is only 20 percent of the dataset, which is relatively small. Also we didn't perform any dimension reduction which can also be a factor that impact the result. Therefore we can't jump to the conclusion yet.

2. The result and plot for three classfiers after using PCA are as shown below and in Figure 2, 3 and 4.

   Naive Bayes training accuracy is 0.9779

   Naive Bayes testing accuracy is 0.9706

   Logistic Regression training accuracy is 0.9853

   Logistic Regression testing accuracy is 1.0000

   KNN training accuracy is 0.9853

   KNN testing accuracy is 0.9706

   Based on the classifiers' output and performances, we observed the following:

   1. The decision boundaries for NB and LR are clear linear, while KNN is an irregular curved line;

   2. NB has the most mis-classified data points, KNN and LR have the same amount of data points that are mis-classified

3. Both NB and LR has much faster run time than KNN (observed while running the code);

4. LR has the highest accuracy score for both training and testing data

Because of the above, Logistic Regression outperformed Naive Bayes and KNN. It suggests a strong fit to the marriage dataset after performing PCA. In contrast, Naive Bayes and KNN models show good fit but did not match Logistic Regression's performance.

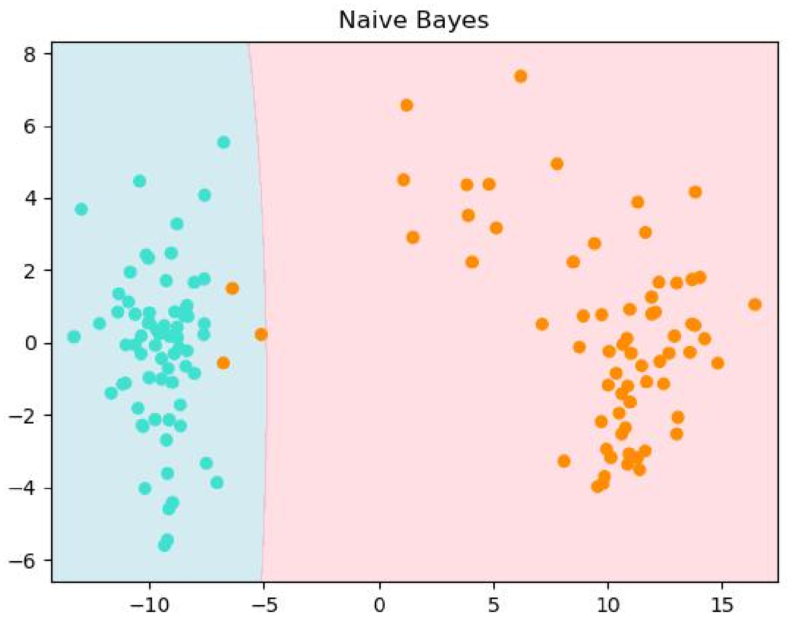


Figure 2: Naive Bayes PCA

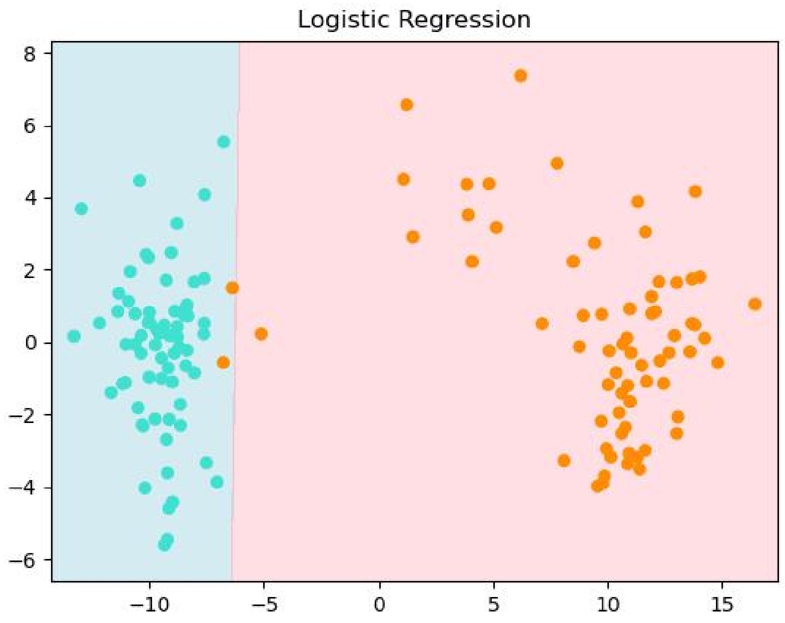


Figure 3: Logistic Regression PCA

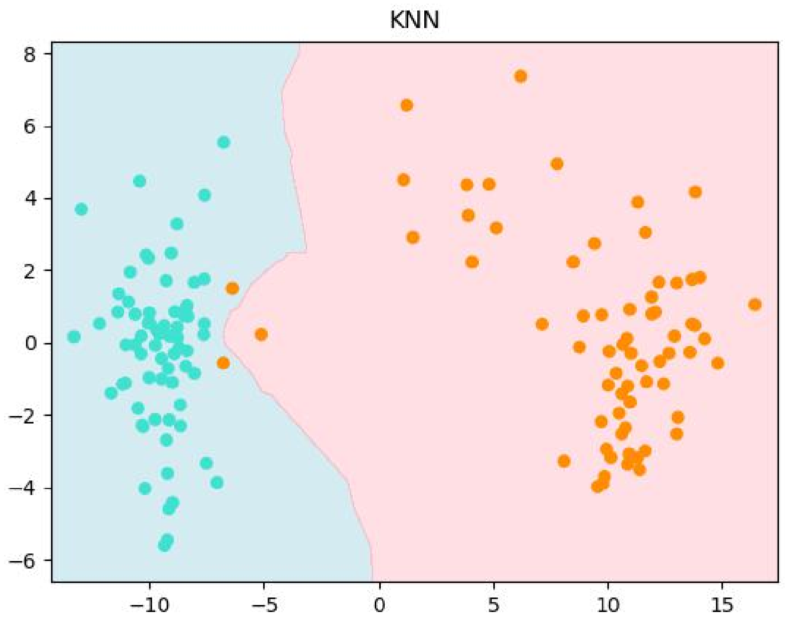


Figure 4: KNN PCA