CS 412 Data Mining (22Sp): Assignment 4

Xu Ke (UIN: 675776713 NetID: kex5)

April. 24, 2022

1 Problem 1. Decision Tree

1.1 Table 1 consists of training data from an employee database. department, age, salary are attributes of the employee. For example, '36...45' for age represents the age range of 36 to 45, '36K...45K' for salary represents the salary range of 36,000 to 45,000, 'sales' represents the employee who belongs to sales department. Let status indicate the categorical labels of these 10 employees. Please calculate the information gain of each attribute. Based on your calculation, identify which attribute should be our first split.

department	*status*	*age*	*salary*
sales	senior	36...45	36K...45K
sales	junior	26...35	26K....35K
sales	junior	36...45	26K...35K
systems	senior	46...55	46K...55K
systems	senior	36...45	46K...55K
systems	junior	36...45	36K...45K
systems	junior	26...45	26K...35K
marketing	senior	36...45	36K...45K
marketing	junior	36...45	26K...35K
marketing	junior	26...35	26K...35K

Table 1: Attributes of 10 employees.

According to the definition of information gain (Lecture 6 Page 15), we know the formula that:

$$Info(D) = -\sum_{i=1}^{m} p_i log_2(p_i) \tag{1}$$

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j) \tag{2}$$

$$Gain(A) = Info(D) - Info_A(D) \tag{3}$$

Thus, in this given case since D represents status, then we have:

$$Info(D) = I(4,6) = -\frac{4}{10} log_2 \frac{4}{10} - \frac{6}{10} log_2 \frac{6}{10} = 0.97095 \tag{4}$$

Thus, the information gain for three attributes, department, age and salary will be respectively:

$$Gain(department) = Info(D) - Info_{department}(D) = I(4,6) - \frac{3}{10} I(1,2) - \frac{4}{10} I(2,2) - \frac{3}{10} I(1,2) = \boxed{0.0200} \tag{5}$$

$$Gain(age) = Info(D) - Info_{age} = I(4,6) - \frac{6}{10}I(3,3) - \frac{3}{10}I(3,0) - \frac{1}{10}I(1,0) = \boxed{0.3710} \tag{6}$$

$$Gain(salary) = Info(D) - Info_{salary} = I(4,6) - \frac{3}{10}I(2,1) - \frac{5}{10}I(5,0) - \frac{2}{10}I(2,0) = \boxed{0.6955} \tag{7}$$

Since while constructing the decision tree, we need to select the attribute with the highest information gain (Lecture 6 Page 15), in this case, we need to take $\boxed{salary}$ as our first split.

1.2 Table 2 provides the information of 10 employees. Let age be a continuous-valued attribute and status indicate the categorical labels of each employee. To determine the best split point of age, we use the following method: (1) Sort the age values in the increasing order; (2) Calculate the average of each pair of adjacent values as a possible split point; (3) Select the possible split point with maximum information gain as the split point. What are the possible split points we get from step (2)? What is the split point we get from step (3)?

age	37	29	41	54	38	45	26	39	42	27
status	senior	junior	junior	senior	senior	junior	junior	senior	junior	junior

Table 2: Age information of 10 employees.

Let's follow the instruction step by step. First, we sort the age values in the increasing order, we will get:

age	26	27	29	37	38	39	41	42	45	54
status	junior	junior	junior	senior	senior	senior	junior	junior	junior	senior

Then, let's calculate the average of each pair of adjacent values as a possible split point, which are:

$$\boxed{26.5, 28, 33, 37.5, 38.5, 40, 41.5, 43.5, 49.5} \tag{8}$$

which are the possible split points we get from step (2). Then we can calculate the information gain respectively:

$$Gain(26.5) = I(4,6) - \frac{1}{10}I(0,1) - \frac{9}{10}I(4,5) = 0.0790 \tag{9}$$

$$Gain(28) = I(4,6) - \frac{2}{10}I(0,2) - \frac{8}{10}I(4,4) = 0.1710 \tag{10}$$

$$Gain(33) = I(4,6) - \frac{3}{10}I(0,3) - \frac{7}{10}I(4,3) = 0.2813 \tag{11}$$

$$Gain(37.5) = I(4,6) - \frac{4}{10}I(1,3) - \frac{6}{10}I(3,3) = 0.0464 \tag{12}$$

$$Gain(38.5) = I(4,6) - \frac{5}{10}I(2,3) - \frac{5}{10}I(2,3) = 0 \tag{13}$$

$$Gain(40) = I(4,6) - \frac{6}{10}I(3,3) - \frac{4}{10}I(1,3) = 0.0464 \tag{14}$$

$$Gain(41.5) = I(4,6) - \frac{7}{10}I(3,4) - \frac{3}{10}I(1,2) = 0.0058 \tag{15}$$

$$Gain(43.5) = I(4,6) - \frac{8}{10}I(3,5) - \frac{2}{10}I(1,1) = 0.0074 \tag{16}$$

$$Gain(49.5) = I(4,6) - \frac{9}{10}I(3,6) - \frac{1}{10}I(1,0) = 0.1445 \tag{17}$$

Thus, the split point we get from step(3) should be $\boxed{33}$, which has maximum information gain.

2 Problem 2. Model Evaluation

Table 3 shows the confusion matrix of a cancer classification model. The rows refer to ground truth, and the columns refer to the predicted class labels.

[*Note: Please write your answer in percentage form and round the number to 2 decimal places.*]

Actual Class \Predicted Class	cancer=yes	cancer=no	Total
cancer=yes	160	270	430
cancer=no	220	12400	12620
Total	380	12670	13050

Table 3: Confusion Matrix of a Cancer Classification Model.

2.1 What are the sensitivity, specificity, accuracy metrics of the classifier?

According to the definition (Lecture 6 Page 47), sensitivity refers to the true positive recognition rate, specificity refers to the true negative recognition rate and accuracy refers to the percentage of test set tuples that are correctly classified. Thus in this given case, we will have:

$$sensitivity = \frac{TP}{P} = \frac{160}{430} = \boxed{0.3721} \tag{18}$$

$$specificity = \frac{TN}{N} = \frac{12400}{12620} = \boxed{0.9826} \tag{19}$$

$$accuracy = \frac{TP + TN}{All} = \frac{160 + 12400}{13050} = \boxed{0.9625} \tag{20}$$

2.2 What are the precision, recall, F1 metrics of the classifier?

According to the definition (Lecture 6 Page 48), precision refers to the exactness, that is what percentage of tuples that the classifier labeled as positive are actually positive, recall refers to the completeness, that is what percentage of positive tuples did the classifier label as positive. Thus in this given case, we will have:

$$precision = \frac{TP}{TP + FP} = \frac{160}{380} = \boxed{0.4211} \tag{21}$$

$$recall = \frac{TP}{TP + FN} = sensitivity = \frac{160}{430} = \boxed{0.3721} \tag{22}$$

Then, according to the definition (Lecture 6 Page 49), F1 score refers to the harmonic mean of precision and recall, and in this given case we have:

$$F_1 = \frac{2precision \times recall}{precision + recall} = \boxed{0.3951} \tag{23}$$

2.3 For a class imbalance problem such as cancer classification, accuracy is not a good metric to evaluate the model. In the cancer classification problem, a classifier, which always labels the samples as negative, could achieve high accuracy. Can you list one measure to handle this issue? Please justify your answer.

Based on my knowledge, I think $\boxed{F - score}$ could be one measure to handle this issue, since it represents the inverse relationship between precision and recall. To be more specific, as shown above, even though it has high accuracy value, but it scores much lower in F1-measure, which indicates our classifier is still not so good.

3 Problem 3. Support Vector Machines (SVM)

Suppose we have one positive example $x_1 = (1, -1)$, $y_1 = 1$, and one negative example $x_2 = (-1, 1)$, $y_2 = -1$. We aim to build a linear hard-margin SVM to classify these two examples.

3.1 For the SVM you build, what is the maximum margin? What is the corresponding weight vector w of the separating hyperplane? What is the the bias scalar b of the separating hyperplane?

According to the definition of linear SVM (Lecture 7 Page 12), since it is 2-D, we can get two hyperplanes that define the sides of the margin, which are:

$$H_1 : b + w_1x_1 + w_2x_2 \geq 1 for y_i = 1 \tag{24}$$

$$H_2 : b + w_1x_1 + w_2x_2 \leq -1 for y_i = -1 \tag{25}$$

Given two examples, $x_1 = (1, -1)$ with $y_1 = 1$ and $x_2 = (-1, 1)$ with $y_2 = -1$, the maximum margin we can achieve is half of the Euclidean distance, which is:

$$maxmargin = \sqrt{(-1-1)^2 + (1-(-1))^2} = \boxed{2\sqrt{2}} \tag{26}$$

To achieve maximum margin, we just need to minimize $\frac{1}{2}||w||^2$, where we have $||w|| = \sqrt{w_1^2 + w_2^2}$:

$$\begin{cases} -b - w_1 + w_2 + 1 \leq 0 \\ b - w_1 + w_2 + 1 \leq 0 \end{cases} \tag{27}$$

Then, we can easily get the loss function shown as following:

$$Loss(w) = \frac{1}{2}(w_1^2 + w_2^2) + \beta_1(1 - w_1 + w_2 - b) + \beta_2(1 - w_1 + w_2 + b) \tag{28}$$

Then let's take the derivative w.r.t w_1,w_2 and b respectively and assign all of $\beta_1(1-w_1+w_2-b), \beta_2(1-w_1+w_2+b)$ to be zero since they are all smaller or equal to 0 from the proof shown above, then we will have the bias scalar $\boxed{b = 0}$ and the corresponding weight vector to be $\boxed{w = \frac{1}{2}(1, -1)}$.

3.2 In practice, it is often the case that data points cannot be linearly separated. Can you provide a solution to classify linearly inseparable data using SVM? Please briefly explain why it can help address the problem.

Instead, I would suggest we can use $\boxed{soft - marginSVM}$ where we allow but penalize data points to be on the "wrong side" of the margin boundary according to the distance to the margin boundary. Alternatively, we can also use $\boxed{kernelfunctions}$ to map data to higher dimensional space and search a linear separating hyperplane in the new space.

4 Problem 4. Feature Selection

Feature selection aims to select a subset of features that will be used in training. In general, there are three major types of feature selection strategy: filter method, wrapper method and embedded method.

4.1 Which type of feature selection strategy does Fisher Score belong to? Please justify your answer using 1-2 sentences.

It belongs to $\boxed{FilterMethods}$ since fisher score selects features based on goodness and it is independent of specific classification model.

4.2 Suppose we have 6 training examples shown below. Please calculate Fisher Scores for θ_0 and θ_1 and find out which one is more discriminative.

According to the definition (Lecture 7 Page 5), the general fomula for fisher score is:

$$s = \frac{\sum_{j=1}^{c} n_j(\mu_j - \mu)^2}{\sum_{j=1}^{c} n_j\sigma_j^2} \tag{29}$$

example	θ_0	θ_1	label
1	75	27	+
2	56	19	+
3	98	33	-
4	67	26	+
5	88	15	-
6	90	42	-

For θ_0 we will have:

$$\mu_{\theta_0} = \frac{75+56+98+67+88+90}{6} = 79 \tag{30}$$

$$\mu_{\theta_0,+} = \frac{75+56+67}{3} = 66 \tag{31}$$

$$\mu_{\theta_0,-} = \frac{98+88+90}{3} = 92 \tag{32}$$

$$\sigma^2_{\theta_0,+} = \frac{(75-66)^2+(56-66)^2+(67-66)^2}{3} = \frac{182}{3} \tag{33}$$

$$\sigma^2_{\theta_0,-} = \frac{(98-92)^2+(88-92)^2+(90-92)^2}{3} = \frac{56}{3} \tag{34}$$

Similarly, for θ_1 we will have:

$$\mu_{\theta_1} = \frac{27+19+33+26+15+42}{6} = 27 \tag{35}$$

$$\mu_{\theta_1,+} = \frac{27+19+26}{3} = 24 \tag{36}$$

$$\mu_{\theta_1,-} = \frac{33+15+42}{3} = 30 \tag{37}$$

$$\sigma^2_{\theta_1,+} = \frac{(27-24)^2+(19-24)^2+(26-24)^2}{3} = \frac{38}{3} \tag{38}$$

$$\sigma^2_{\theta_1,-} = \frac{(33-30)^2+(15-30)^2+(42-30)^2}{3} = 126 \tag{39}$$

Thus, in this given case, we will have:

$$s_{\theta_0} = \frac{3\times(66-79)^2+3\times(92-79)^2}{3\times\frac{182}{3}+3\times\frac{56}{3}} = \boxed{4.2605} \tag{40}$$

$$s_{\theta_1} = \frac{3\times(24-27)^2+3\times(30-27)^2}{3\times\frac{38}{3}+3\times 126} = \boxed{0.1298} \tag{41}$$

Thus, we can conclude that $\boxed{\theta_0}$ is more discriminative.

4.3 Ridge regression aims to find the optimal weight vector by minimizing the objective function $L(w) = \sum_{i=1}^{n}(y_i - \boldsymbol{w}^T\boldsymbol{x}_i)^2 + \lambda||w||^2$. The closed-form solution of ridge regression is $\boldsymbol{w} = (\boldsymbol{X}\boldsymbol{X}^T + \lambda\boldsymbol{I})^{-1}\boldsymbol{X}\boldsymbol{y}$, where X is a matrix whose i-th column is x_i and y is a column vector whose i-th element is y_i. It is worth pointing out that the closed-form solution of ridge regression is often dense, meaning that w often does not contain zero(s). Based on the given information, what is the difference between LASSO and ridge regression? Can ridge regression be used for feature selection? Why?

It is clear that *LASSO* is a modification of linear regression, where the model is penalized for the sum of absolute values of the weights (the L1 term). Thus, the absolute values of weight will be generally reduced, and many will tend to be zeros. While *Ridge Regression* takes a step further and penalizes the model for the sum of squared value of the weights (the L2 term). Thus, the weights not only tend to have smaller absolute values, but also really tend to penalize the extremes of the weights, resulting in a group of weights that are more evenly distributed.

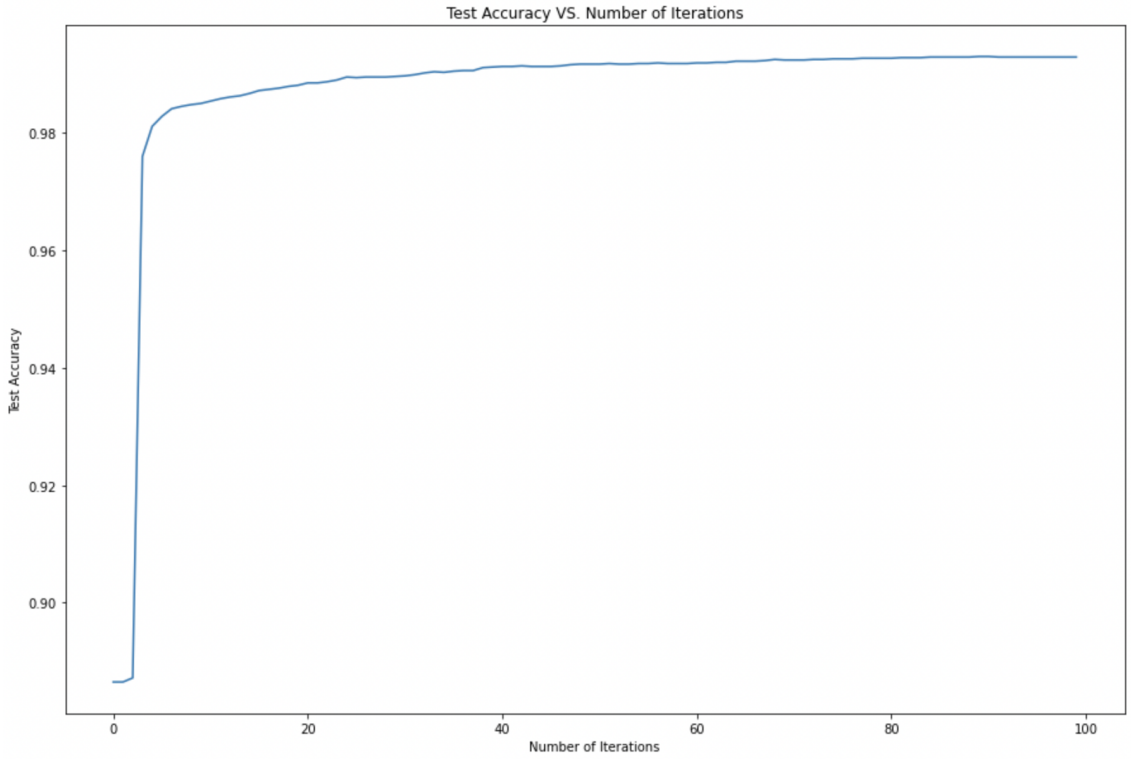
In other words, *Ridge Regression* leads to both low variance (as some coefficient leads to negligible effect on prediction) and low bias (minimization of coefficient reduce the dependency of prediction on a particular variable). The difference between *Ridge* and *LASSO* regression is that *LASSO* tends to make coefficients to absolute zero as compared to *Ridge* which never sets the value of coefficient to absolute zero. By the way, *Ridge Regression* is computationally less intensive than *LASSO*.

$\boxed{No}$, we cannot use ridge regression for feature selection since generally unlike LASSO, ridge does not set weight coefficients to zero, and thus applying ridge penalty won't have this effect.

5 Problem 5. Logistic Regression

5.1 The test accuracy vs. the number of iterations

5.1.1 Plot the test accuracy vs. the number of iterations

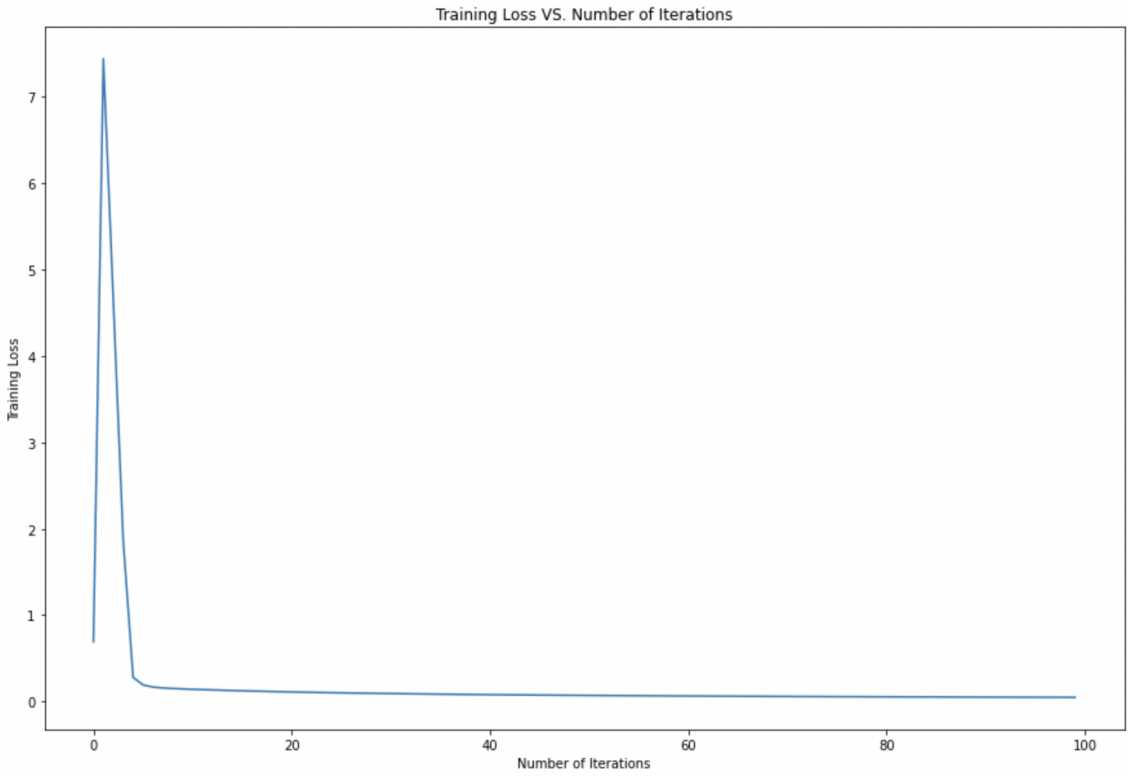


5.1.2 Observations and explanations of the result

Based printout during the training and test process, I know the exact values after each iteration. From the plot, it is clear that the accuracy is relatively low (0.8865) at beginning and it climbs very quickly in the first 5 iteration (from 0.8865 to 0.9811). Then it increases more slower and at about 85th iteration, it achieves steady value (0.9929). This result makes sense since we use gradient descent algorithm to backpropgate and we have relatively large gradient at beginning, and during the training process, the gradient will become smaller and finally reach very small value (i.e. zero), thus the speed of convergence becomes slower and slower and finally achieve convergence.

5.2 The training loss vs. the number of iterations

5.2.1 Plot the training loss vs. the number of iterations



5.2.2 Observations and explanations of the result

Based printout during the training and test process, I know the exact values after each iteration. From the plot, it is clear that the training loss is relatively low (0.69) at the beginning, however, it jumps suddenly to large value (7.43) at the second iteration and then decreases again. At about 5th iteration, it achieves much lower (0.28) than the beginning, and then it stays at low value (0.05) until the end of 100 iterations. This result also makes sense since at this period, the gradient is very large and parameters update to make large difference, thus because the step is too long, making it far away to convergence and have large loss. And then in the process of jumping back and forth, it finally achieve the convergence, which leads to low loss again.