SuperAGI Assignment

NAME: VANCHANAGIRI ALEKHYA
ENTRY NUMBER: 2020EE10565
CAMPUS: IIT DELHI

Q/A Assignment:

1. You train Logistic Regression with a certain set of features and learn weights w_0, w_1 till w_n. Feature n gets weight w_n at the end of training. Say you now create a new dataset where you duplicate feature n into feature (n+1) and retrain a new model. Suppose this new model weights are w_{new_0}, w_{new_1} till w_{new_n}, w_{new_{n+1}}. What is the likely relationship between w_{new_0}, w_{new_1} , w_{new_n}, and w_{new_{n+1}}?

Ans. When we duplicate a feature in a dataset and retrain a logistic regression model, the effect on the learned weights can be quite interesting.

1. **Impact on Intercept (w_0 vs w_{new_0}):**
   The intercept term (w_0) is likely to remain relatively unaffected by the duplication of a feature. This term adjusts the decision boundary independently of the feature values.
2. **Impact on Weights of Other Features (w_1 to w_{n-1} vs w_{new_1} to w_{new_{n-1}}):**
   The weights for other features (from w_1 to w_{n-1}) are also likely to remain relatively similar, assuming that these features are not highly correlated with the duplicated feature. If there are correlations, some adjustments might occur due to changes in the overall feature space.
3. **Impact on Weights of Duplicated Features (w_n vs w_{new_n} and w_{new_{n+1}}):**
   The most significant change will be observed in the weights of the duplicated features (w_n, w_{new_n}, and w_{new_{n+1}}). In a typical scenario, logistic regression will distribute the weight of the original feature (w_n) between the two duplicated features (w_{new_n} and w_{new_{n+1}}) in the new model. This distribution might not be even, but the sum of w_{new_n} and w_{new_{n+1}} is likely to be close to the original w_n. This is because the overall contribution of the duplicated feature (now split into two) to the model's decision should remain similar to what it was before duplication, barring any regularization effects or interactions with other features.
4. **Regularization Considerations:**
   If regularization (like L1 or L2) is used in the logistic regression model, the way weights are adjusted might be different. Regularization techniques can influence how weights are distributed among correlated features.

2. You currently have an email marketing template A and you want to replace it with a better template. A is the control_template. You also test email templates B, C, D, E. You send exactly 1000 emails of each template to different random users. You wish to figure out what email gets the highest click through rate. Template A gets 10% click through rate (CTR), B gets 7% CTR, C gets 8.5% CTR, D gets 12% CTR and E gets 14% CTR. You want to run your multivariate test till you get 95% confidence in a conclusion. Which of the following is true?
   a. We have too little data to conclude that A is better or worse than any other template with 95% confidence.
   b. E is better than A with over 95% confidence, B is worse than A with over 95% confidence. You need to run the test for longer to tell where C and D compare to A with 95% confidence.
   c. Both D and E are better than A with 95% confidence. Both B and C are worse than A with over 95% confidence

Ans:
The calculated p-values for the comparisons between Template A and the other templates are as follows:

1. A vs B: ≈ 0.020 $p \approx 0.020$
2. A vs C: ≈ 0.280 $p \approx 0.280$
3. A vs D: ≈ 0.175 $p \approx 0.175$
4. A vs E: ≈ 0.007 $p \approx 0.007$

Given a significance level of 0.05 (95% confidence), we interpret these p-values as follows:

A vs B: The p-value is approximately 0.020, which is less than 0.05, indicating that the difference in CTR between A and B is statistically significant. Template B is worse than A with over 95% confidence.
A vs C: The p-value is approximately 0.280, which is greater than 0.05, suggesting that we cannot conclude with 95% confidence that C is significantly different from A.
A vs D: The p-value is approximately 0.175, which is also greater than 0.05, indicating that the difference in CTR between A and D is not statistically significant at the 95% confidence level.
A vs E: The p-value is approximately 0.007, which is less than 0.05, meaning the difference in CTR between A and E is statistically significant. Template E is better than A with over 95% confidence.

Based on this analysis, the statement that most accurately reflects the results is: Statement 2: "E is better than A with over 95% confidence, B is worse than A with over 95% confidence. You need to run the test for longer to tell where C and D compare to A with 95% confidence."

3. You have m training examples and n features. Your feature vectors are however sparse and average number of non-zero entries in each train example is k and $k << n$. What is the approximate computational cost of each gradient descent iteration of logistic regression in modern well written packages?

Ans: In logistic regression, the computational cost of each gradient descent iteration depends largely on the operations performed for calculating the gradient and updating the weights.

For a dataset with m training examples, n features, and sparsity such that the average number of non-zero entries in each training example is k (with $k \ll n$), the computational cost can be significantly reduced compared to a dense dataset.

1. **Gradient Calculation:** The gradient of the cost function with respect to each weight in logistic regression is computed as the average of the product of the error and the feature value across all training examples. In a sparse dataset, most of the features in each example are zero, and their contribution to the gradient is zero as well. Therefore, the computation needs to consider only the non-zero features.
2. **Feature Sparsity:** Since each training example has, on average, k non-zero entries, and $k \ll n$, the cost of computing the gradient for each feature is proportional to k rather than n.
3. **Total Computational Cost:** For each iteration of gradient descent, we need to calculate the gradient for each of the n features across all m training examples. However, due to sparsity, each example contributes to only about k features on average. Thus, the total computational cost for each iteration is approximately $O(m \times k)$, rather than $O(m \times n)$.
4. **Modern Well-Written Packages:** In practice, well-written machine learning libraries take advantage of sparse representations and efficient computation techniques. They would typically store only the non-zero elements and their indices, avoiding the need to process or store zeros. This can further reduce both memory and computational requirements.

   Therefore, in the case of a sparse dataset with $k \ll n$, the approximate computational cost of each gradient descent iteration in logistic regression within modern, well-optimized software packages would be $O(m \times k)$. This is a significant reduction compared to the $O(m \times n)$ cost that would be incurred if all features were dense (non-sparse).

5. You wish to estimate the probability, p that a coin will come up heads, since it may not be a fair coin. You toss the coin n times and it comes up heads k times. You use the following three methods to estimate p

   a. Maximum Likelihood estimate (MLE)

   b. Bayesian Estimate: Here you assume a continuous distribution uniform prior to p from $[0, 1]$ (i.e. the probability density function for the value of p is uniformly 1 inside this range and 0 outside. Our estimate for p will be the expected value of the posterior distribution of p. The posterior distribution is conditioned on these observations.

   c. Maximum a posteriori (MAP) estimate: Here you assume that the prior is the same as (b). But we are interested in the value of p that corresponds to the mode of the posterior distribution.

   What are the estimates?

Ans: To estimate the probability p of getting heads when flipping a coin, we can use three different approaches as described:

1. Maximum Likelihood Estimate (MLE),
2. Bayesian Estimate, and
3. Maximum a Posteriori (MAP) estimate.

given n coin tosses resulting in k heads,

1. Maximum Likelihood Estimate (MLE): MLE seeks the parameter value that maximizes the likelihood of the observed data. For a binomial distribution, which is appropriate for coin flips, the likelihood of observing k heads in n tosses is given by the binomial probability mass function. The MLE of p is simply the proportion of heads observed:

   $$p\_MLE = k/n$$

2. Bayesian Estimate: With a uniform prior (assuming p is equally likely to be any value between 0 and 1), the posterior distribution after observing k heads in n tosses is a Beta distribution, Beta(k+1,n−k+1).
   The expected value of a Beta distribution Beta(α,β) is $\alpha/\alpha+\beta$. Therefore, the Bayesian estimate for p is:

   $$p\_Bayesian = k+1/n+2$$

3. Maximum a Posteriori (MAP) Estimate: The MAP estimate is the mode of the posterior distribution. For a Beta distribution Beta(α,β), the mode is $\alpha-1/\alpha+\beta-2$, provided both $\alpha>1$ and $\beta>1$. For the Beta distribution Beta(k+1,n−k+1), the MAP estimate of p is:

   p MAP = k/n if k>0 and k<n;

   otherwise, it is not well-defined (for k=0 or k=n, the mode is at the endpoints 0 or 1, respectively)