Submitted by
Piyush Soni
2020EE10522

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_{new0}, w_{new1} till w_{newn}, w_{newn+1}. What is the likely relationship between w_{new0}, w_{new1}, w_{newn}, and w_{newn+1}?*

    - w_{new0}: The weight associated with the intercept term is unlikely to change significantly unless the duplicated feature has a significant impact on the model's bias.

    - w_{new1} to w_{newn}: The weights corresponding to the original features (excluding the duplicated feature) are expected to be similar to the weights learned in the initial model, assuming the duplicated feature does not introduce substantial collinearity or information redundancy.

    - w_{newn+1}: The weight associated with the duplicated feature (n+1) is likely to be similar to the weight w_{newn} assigned to the original feature n. This assumes that the duplicated feature captures similar information and patterns as the original feature.

    In summary, the weights w_{new0}, w_{new1} to w_{newn} are expected to be consistent with the initial model, and w_{newn+1} is likely to be similar to w_{newn}, reflecting the duplicated feature's influence on the model.

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*

    Correct answer is option 3: "Both D and E are better than A with 95% confidence. Both B and C are worse than A with over 95% confidence."

    "We have too little data to conclude that A is better or worse than any other template with 95% confidence." - This statement is not correct because we can compare the CTRs of templates B, C, D, and E with A.

"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." - This statement is not correct. We can compare the CTRs directly, and based on the given information, E has a higher CTR than A with over 95% confidence. However, we can also conclude that B and C have lower CTRs than A with over 95% confidence without needing to run the test longer.

"Both D and E are better than A with 95% confidence. Both B and C are worse than A with over 95% confidence." - This statement is correct based on the information provided. E has a higher CTR than A with over 95% confidence, and D also has a higher CTR than A with over 95% confidence. Additionally, B and C have lower CTRs than A with over 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?*

   The approximate computational cost of each gradient descent iteration for logistic regression in the described scenario is O(m * k), where:

   - m is the number of training examples.
   - k is the average number of non-zero entries in each training example.

   The sparsity of the feature vectors allows for computational efficiency, and the cost per iteration is proportional to the number of non-zero entries in the feature vectors. Modern well-written packages often exploit the sparsity of data, resulting in a more efficient implementation.

4. *We are interested in building a high quality text classifier that categorizes news stories into 2 categories - information and entertainment. We want the classifier to stick with predicting the better among these two categories (this classifier won't try to predict a percent score for these two categories). You have already trained V1 of a classifier with 10,000 news stories from the New York Times, which is one of 1000 new sources we would like the next version of our classifier (let's call it V2) to correctly categorize stories for. You would like to train a new classifier with the original 10,000 New York Times news stories and an additional 10,000 different news stories and no more. Below are approaches to generating the additional 10,000 pieces of train data for training V2.*
   1. *Run our V1 classifier on 1 Million random stories from the 1000 news sources. Get the 10k stories where the V1 classifier's output is closest to the decision boundary and get these examples labeled.*
   2. *Get 10k random labeled stories from the 1000 news sources we care about.*

3. *Pick a random sample of 1 million stories from 1000 news sources and have them labeled. Pick the subset of 10k stories where the V1 classifier's output is both wrong and farthest away from the decision boundary.*

Ignore the difference in costs and effort in obtaining train data using the different methods described above. In terms of pure accuracy of classifier V2 when classifying a bag of new articles from 1000 news sources, what is likely to be the value of these different methods? How do you think the models will rank based on their accuracy?

The models are likely to rank based on accuracy as follows:

I. **Method 2: Get 10k random labeled stories from the 1000 news sources we care about.**

This method directly samples labeled stories from the desired sources. It provides diverse and representative examples for training the classifier on the target distribution. This approach is likely to contribute positively to the accuracy of the V2 classifier.

II. **Method 3: Pick a random sample of 1 million stories from 1000 news sources and have them labeled. Pick the subset of 10k stories where the V1 classifier's output is both wrong and farthest away from the decision boundary.**

While this method aims to focus on challenging examples by selecting stories where V1 is wrong and far from the decision boundary, it introduces uncertainty by relying on V1's performance. The labeled data might include ambiguous cases, potentially making it less effective than directly sampling from the target distribution.

III. **Method 1: Run our V1 classifier on 1 Million random stories from the 1000 news sources. Get the 10k stories where the V1 classifier's output is closest to the decision boundary and get these examples labeled.**

This method relies on uncertainty around the decision boundary of the V1 classifier. It may select examples that are difficult to classify accurately, introducing noise into the training set. As a result, it might not contribute as effectively to improving the accuracy of the V2 classifier compared to direct sampling (Method 2).

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?

a. **Maximum Likelihood Estimate (MLE):**

   MLE estimate for p is k/n, where k is the number of times the coin comes up heads, and n is the total number of coin tosses.

b. **Bayesian Estimate:**

   Bayesian estimate involves updating the prior with the observed data. In this case, assuming a continuous uniform prior, the Bayesian estimate (posterior mean) is (k + 1) / (n + 2), which is the expected value of the posterior distribution.

c. **Maximum a Posteriori (MAP) Estimate:**

   The MAP estimate corresponds to the mode of the posterior distribution. With a continuous uniform prior, the MAP estimate is (k + 1) / (n + 2), similar to the Bayesian estimate.