Q/A Assignment

1) There are some common patterns:
   a. **Similarity of Wnewn and Wnewn+1 weights:** If the reproduced feature (n+1) has a similar effect to dataset feature n, it is likely that the weights assigned to the two features in the new model will be close. or correlated.
   b. **Effect on other weights:** the implementation of a duplicated feature can affect the proportion of other features in the model, which can cause changes in their weights.
   c. **Normalizing and regulatory effects.** Techniques such as regularization (eg, L1, L2 regularization) or normalization (eg, feature scaling) can affect how weights are distributed among features in the new model. This may affect the relationship between Wnew0 , Wnew1 , Wnewn , and Wnewn+1
   d. **Model complexity and overfitting:** If the model is relatively simple or prone to overfitting, the new weights may not show a clear pattern or relationship between Wnew0, Wnew1, Wnewn and Wnewn+1.

2) To determine which statement is true, analyse the scenario:
   Click **through rates (CTR) are provided:**
   A: 10%
   B: 7%
   C: 8.5%
   D: 12%
   E: 14%
   To compare **models** and **draw** statistically significant conclusions, we can perform a hypothesis test, typically using statistical methods **such as chi-square** tests or z-tests **of** proportions.

   - **Calculating confidence intervals:**
     We want to reach a 95% confidence level for our conclusions.
     From the information provided, it is clear that model A has a click-through rate of 10%.
     - Let's compare other A models: Model B has a lower click-through rate than model A (7%).
     - CTR of model C is slightly lower than A (8.5%).
     - Model D has a higher CTR than Model A (12%).
     - Model E has a significantly higher click-through rate than Model A (14%).
   - **Hypothesis test:**

- Currently based on preliminary results: Model E seems to perform better than Model A with a higher click-through rate.
- Also, Model D seems to have a higher CTR than Model A. The CTR of models B and C seems to be lower than A.
- However, to confirm these findings with a 95% confidence level, we need to perform statistical tests comparing the CTRs of each model with A.
- We have enough data to conclude that E performs better than A with more than 95% confidence based on a significant difference in CTR. We can also conclude that B performs worse than A with **more than** 95% confidence based on **a** significant difference in CTR.
- Regarding **models** C and D: We need more **information** or **additional** statistical analysis to **confirm** whether C and D are significantly different from A at **the** 95% confidence level.

So the correct statement is: 2.

- E is better than A with **more than 95 percent certainty,** B is worse than A with **more than 95 percent certainty.** You need to run the test longer to **know** where C and D compare to A with 95% confidence.

3) In the context of logistic regression with sparse feature vectors, the approximate computational cost of each gradient descent iteration in modern well-written packages is influenced by the sparsity of the feature vectors.
Here's an approximate breakdown of the computational cost for each iteration:

- **Computational Cost for Gradient Calculation:**
  - For each training example, the cost to compute the gradient of the loss function with respect to the weights is roughly O(k) where k is the number of non-zero entries in the sparse feature vector.
- **Total Computational Cost for All Examples:**
  - With m training examples, the total cost for gradient calculation across all examples is approximately O(m·k).
- **Parameter Update:**
  - The parameter update involves adjusting the weights according to the computed gradients. This step typically incurs a cost of O(n), where n is the total number of features.
- **Overall Computational Cost per Iteration:**
  - Considering the costs mentioned above, the overall computational cost per iteration of gradient descent for logistic regression with sparse features can

be approximated as O(m·k+n), where k is the average number of non-zero entries in each training example, m is the number of training examples, and n is the number of features.

Modern well-written packages for machine learning and optimization, like scikit-learn or TensorFlow, often implement optimized algorithms that take advantage of sparse matrix operations and optimization techniques to reduce computational overhead when dealing with sparse data. These implementations optimize the gradient calculation and parameter updates specifically for sparse feature representations, improving the efficiency of the overall algorithm.

4) The potential impact of different approaches on generating additional training data for a V2 classifier that aims to classify articles from 1000 news sources depends on their focus:
   - **Approach 1:** Using V1 on 1 million stories and choosing the story closest to the decision boundary
     - Focus: Select cases where V1' s output is closest to the decision limit.
     - Effect: Can help with V2 hard cases and improve accuracy near the limit. However, there is a lack of variety in the presentation of general information.
   - **Approach 2: Get 10,000 randomly tagged stories from 1,000 news sources**
     - Focus: A random selection of tagged tracks for variety.
     - Impact: Promotes general data diversity, but may not specifically target areas where V1 is inferior or decision-limited.
   - **Approach 3: Selecting misclassified stories farthest from the decision boundaries**
     - Focus: Select cases where the result of V1's is wrong and furthest from the decision boundary. I
     - Impact: V2 can be significantly improved by focusing on the cases where V1 makes the most mistakes, helping V2 correct those mistakes.

Regarding the potential impact on V2 accuracy:

- **Approach 3** (selecting misclassified tracks farthest from the decision boundary) may have the most significant improvement potential by correcting errors V1's
- **Approach 1** (selecting examples closest to the decision boundary) can improve performance in difficult regions.
- **Approach 2** (random selection of flagged stories) may have less impact on improving the performance of the classifier and 1000 news source articles because it does not focus on V1 errors or decision limits.

5) Derive the estimates for the probability p using the three methods given the outcomes of n coin tosses where k of them resulted in heads.
   - **Maximum Likelihood Estimate (MLE):**

     The Maximum Likelihood Estimate for p is simply the proportion of heads observed in the coin tosses.

MLE: $\hat{p}$ MLE=k/n

- **Bayesian Estimate (Expected Value of Posterior Distribution):** With a uniform prior over p from 0 to 1, the posterior distribution of p given k heads in n tosses follows a Beta distribution. The expected value (mean) of the posterior Beta distribution is given by:

  Bayesian Estimate: $\hat{p}$ Bayesian= $k+1/n+2$

- **Maximum a Posteriori (MAP) Estimate (Mode of Posterior Distribution):**
  - The mode of the posterior Beta distribution corresponds to the value of p with the highest probability density. The mode (MAP estimate) of the Beta distribution is given by:

    MAP Estimate: $\hat{p}$ MAP= $k/n+2$

- These estimates provide different perspectives on estimating the probability p of getting heads in a coin toss based on observed data and different assumptions about the prior distribution.
- The Bayesian and MAP estimates incorporate prior information and give slightly different results compared to the MLE, especially when the number of tosses n is relatively small, influencing the impact of the prior information.