DEPENDABLE AND SECURE AI-ML (AI60006)

Assignment 2

Abhinav Bohra **18CS30049**

1. Federated Learning process

Task: Take a screenshot of your outputs and record the timing required to compute the Federated Learning process.

Screenshot of Output

```
Loading data
Error (MSE) that each client gets on test set by training only on own local data:
Hospital 1:     3810.44
Hospital 2:     3982.58
Hospital 3:     3569.32
Hospital 4:     4144.15
Hospital 5:     3848.39
Running distributed gradient aggregation for 50 iterations
Error (MSE) that each client gets after running the protocol:
Hospital 1:     3775.50
Hospital 2:     3775.50
Hospital 3:     3775.50
Hospital 4:     3775.50
Hospital 5:     3775.50


Time Taken to compute federated learning process = 0.0 mins 51.8663330078125 secs
```

Time required to compute = 51.87 secs

2. Privacy-preserving SVM assuming public model private data scenario

Task: Following the similar adequate partial homomorphism in encryption (as discussed in the class and given in the code), implement privacy-preserving SVM assuming public model private data scenario (data in encrypted but model parameters are unencrypted):

Implementation Details

In public model private data scenario, it is not feasible to train SVM classifier on server using encrypted data as it necessitates computing the dot product between two encrypted number: X_i and Y_i as shown in the equation:-

$$weights \mathrel{-}= learning\_rate * (2 * C * weights - dot\_product(X_i, Y_i))$$

Paillier Partially Homomorphic Encryption supports addition and multiplication of an encrypted number by a scalar (constant), but it does not support multiplication of two encrypted numbers directly. Therefore, the above mentioned equation cannot be computed.

If we consider the fact that multiplication can be written as repeated addition, i.e. to compute x*y we can add x to itself y times, using the code below:-

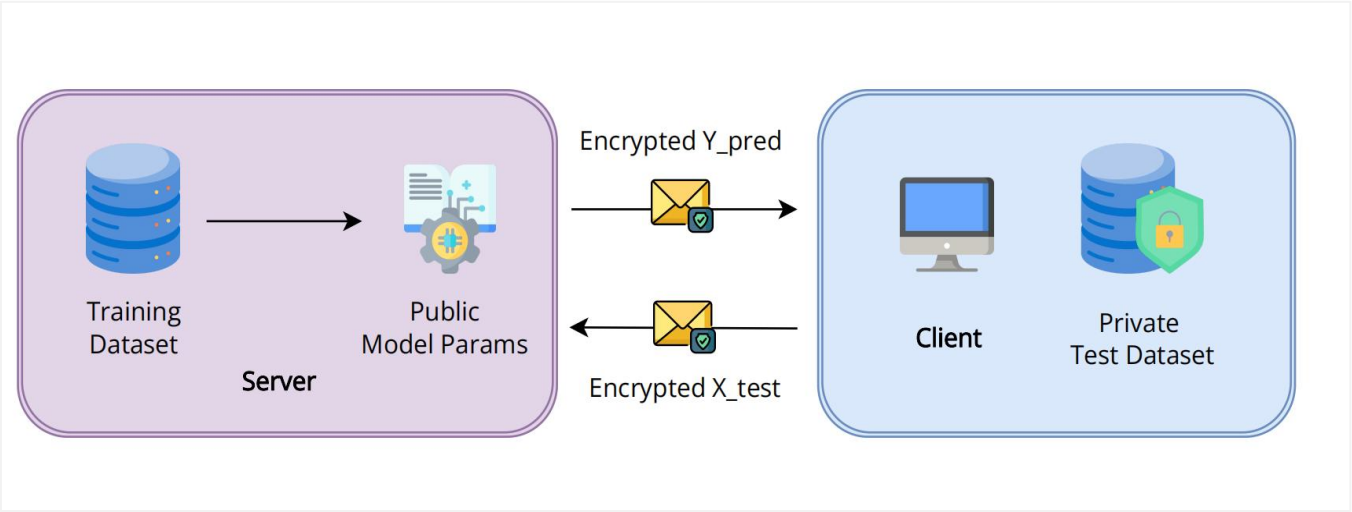
```
ans=0

for i in range(y):
    ans +=x
```

However, in this scenario, we cannot run the loop as we don't know the value of y since it is encrypted.

Thus, I have employed the following strategy to solve this question.

System Design



1. **Train SVM Classifier model on server using public data:** In this step, a support vector machine (SVM) model is trained on the server using public data. The public data is data that is available to everyone and does not contain any sensitive information. This model will be used to make predictions on the encrypted data sent by the client**.**

2. **Encrypt the data on the client:** The client encrypts their private data (X_test) using a homomorphic encryption scheme. Homomorphic encryption allows computations to be performed on encrypted data without decrypting it first, preserving the privacy of the data. The encryption process generates a ciphertext that is sent to the server for prediction.

3. **Send the encrypted X_test to the server:** The encrypted data is sent to the server for prediction. The server can perform computations required for inference on the encrypted data without having access to the original plaintext.

4. **Use unencrypted model parameters for inference:** The server uses the unencrypted model parameters to perform the prediction on the encrypted data. The model parameters are not encrypted and can be used directly for prediction because inference operations required simple addition and multiplication between one encrypted and one unencrypted number

5. **Send model predictions back to the client:** After the prediction is performed, the server sends the encrypted predictions (Y_pred) back to the client.

6. **On the client, decrypt Y_pred and calculate accuracy:** The client decrypts the encrypted predictions (Y_pred) using the private key to obtain the final predictions in plaintext. The accuracy of the predictions can be calculated by comparing the predicted labels with the actual labels of the test data (Y_test)

Partial homomorphism in encryption allows for some limited computations to be performed on encrypted data. In the context of privacy-preserving SVM, it enables the server to perform the prediction on the encrypted data while preserving the privacy of the client's data.

Data-set used

This example involves learning using sensitive medical data from multiple hospitals to predict diabetes progression in patients. The data is a standard dataset from sklearn

- Train-set size: 614 samples
- Test set size: 154 samples
- Number of Features: 8
- Target Outcome: 0 or 1 (Binary Classification)

Screenshot of Output

```
Test accuracy for Normal SVM Model is 0.75

Total Time Taken: = 0.0 mins 3.31 secs
```

```
Test accuracy for Privacy-preserving SVM Model is 0.75

Total Time Taken: = 0.0 mins 33.56 secs
```

Timing details

Time taken by normal SVM model = 3.31 secs

Time taken by privacy preserving SVM model = 33.56 secs

Platform details (for both questions)

- **Google Colab** GPU: 1 x NVIDIA Tesla K80
- VRAM: 12GB
- CPU: 1 x Intel Xeon @ 2.3GHz
- RAM: 12.72 GB
- Disk Space: 68 GB