# Abstract

# Introduction

With the popularity of Cloud computing, more and more user’s data are stored on Cloud servers. This raises concerns about privacy. In order to get practical guidance on our decision making that cloud and [machine learning](https://subscription.packtpub.com/search?released=Available&concept=Machine%20Learning) provide, we need to share our personal information. The traditional encryption schemes do not allow running any computation on encrypted data. So we need to choose between storing our data encrypted in the cloud and downloading it to perform any useful operations or providing the decryption key to service providers which risks our privacy.

Homomorphic encryption can solve this problem. If the user encrypts the data using homomorphic encryption, the Cloud can perform meaningful computation on the encrypted data in order to provide services to users without revealing the user’s privacy.

Microsoft’s Simple Encrypted Arithmetic Library (SEAL) is a [free](https://en.wikipedia.org/wiki/Free_software) and [open-source](https://en.wikipedia.org/wiki/Open-source_software) [cross platform](https://en.wikipedia.org/wiki/Cross_platform) software library developed by [Microsoft Research](https://en.wikipedia.org/wiki/Microsoft_Research) that implements various forms of [homomorphic encryption](https://en.wikipedia.org/wiki/Homomorphic_encryption).

## SVM

## FHE

The idea of fully homomorphic encryption was first proposed by Rivest et al. in the 1970s. Compared with the general encryption algorithm, homomorphic encryption can let you do computation on ciphertexts. An encryption function with homomorphism is an encryption function in which two plaintexts a and b satisfy Dec(Enc(a) LEnc(b)) = a Nb, where Enc stands for encryption, and Dec stands for decryption, L N , corresponds to operations on the plaintext and ciphertext. When L represents addition on the plaintext, the encryption is said to be an additive homomorphism: when L represents multiplication on the plaintext, the encryption is said to be a multiplicative homomorphism. A cryptosystem that supports arbitrary computation on ciphertexts is known as fully homomorphic encryption (FHE). That is, Dec(f(Enc(m1), Enc(m2), ..., Enc(mk))) = f(m1, m2, ..., mk), or written as: f(Enc(m1), Enc(m2), ..., Enc(mk)) = Enc(f(m1, m2, ..., mk)), for arbitrary function f. How to construct a fully homomorphic encryption scheme is an open challenge. Until 2009, Gentry proposed the first fully homomorphic cryptosystem based on the ideal lattice[1], which made a breakthrough in this field. Then many cryptographers have done meaningful work in the research of the homomorphic encryption scheme.

# Implementationn details

Tools:

### Scikit-learn

Scikit-learn is an open source machine learning Python library that supports supervised and unsupervised learning. It also provides various tools for model fitting, data preprocessing, model selection and evaluation, and many other utilities.

Scikit-learn provides dozens of built-in machine learning algorithms and models, called [estimators](https://scikit-learn.org/stable/glossary.html#term-estimators). Each estimator can be fitted to some data using its [fit](https://scikit-learn.org/stable/glossary.html#term-fit) method.

Scikit provides build in methods for:

1. A pre-processing step that transforms or imputes the data.
2. Fitting in the training phase
3. A final predictor that predicts target values
4. Functions for model evaluation loke cross validation
5. Functions for automatically find the best parameter combinations (via cross-validation).

For further details one can refer to: <https://scikit-learn.org/stable/index.html>

### Anaconda and Jupyter notebook

**Anaconda** is a [free and open-source](https://en.wikipedia.org/wiki/Free_and_open-source)[[5]](https://en.wikipedia.org/wiki/Anaconda_(Python_distribution)#cite_note-5) distribution of the [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) and [R](https://en.wikipedia.org/wiki/R_(programming_language)) programming languages for [scientific computing](https://en.wikipedia.org/wiki/Scientific_computing) , that aims to simplify [package management](https://en.wikipedia.org/wiki/Package_management) and deployment. Package versions are managed by the [package management system](https://en.wikipedia.org/wiki/Package_manager) [*conda*](https://en.wikipedia.org/wiki/Conda_(package_manager)). The Anaconda distribution includes data-science packages suitable for Windows, Linux, and MacOS.

Anaconda Navigator is a desktop [graphical user interface (GUI)](https://en.wikipedia.org/wiki/Graphical_user_interface) included in Anaconda distribution that allows users to launch applications and manage conda packages, environments and channels without using [command-line commands](https://en.wikipedia.org/wiki/Command-line_interface). Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository, install them in an environment, run the packages and update them. It is available for [Windows](https://en.wikipedia.org/wiki/Windows), [macOS](https://en.wikipedia.org/wiki/MacOS) and [Linux](https://en.wikipedia.org/wiki/Linux).

Jupyter notebook is a [web-based interactive](https://en.wikipedia.org/wiki/Rich_Internet_application) computational environment for creating Jupyter notebook documents . Notebook documents can contain phyton code that can be executed on the spot through a notebook kernel . A notebook kernel is a “computational engine” that executes the code contained in a [Notebook document](https://jupyter-notebook-beginner-guide.readthedocs.io/en/latest/what_is_jupyter.html#notebook-document). The ipython kernelexecutes python code. Kernels for many other languages exist .

In my work I found more convenient to work with Jupyter notebook , rather with more sophystacted editor like phycharm.

### SEAL

Microsoft SEAL is a homomorphic encryption library that allows additions and multiplications to be performed on encrypted integers or real numbers. Other operations, such as encrypted comparison, sorting, or regular expressions, are in most cases not feasible to evaluate on encrypted data using this technology. Therefore, only specific privacy-critical cloud computation parts of programs should be implemented with Microsoft SEAL.

Microsoft’s SEAL Homomorphic Encryption library implements two homomorphic cryptographic schemes, BFV scheme and CKKS scheme.

### Visual studio

Microsoft visual studio 2019 is used as the code editor and compiler.

### GitHub

I have used GitHb as my source control.

All the code the uploaded to the following repository :

<https://github.com/GALSAV/SecureCloudComputingSEAL>

Work details

### Assignment

The laboratory assignment was to train an

### DataSets

In my work I’ve used 2 datasets

1. Iris dataset : <https://archive.ics.uci.edu/ml/datasets/Iris>
2. Mashroom dataset : <https://archive.ics.uci.edu/ml/datasets/mushroom>

Iris dataset , is a simple datasets which is used in many beginners guide’s for machine learning.

The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant.

Each instance has 4 attributes : sepal length, sepal width,petal length, petal width .

All attributes are measured in cm.

One class is linearly separable from the other 2. The Predicted attribute is a class of iris plant.

In my work I’ve have used only 2 classes.

Mashroom dataset , contains 8124 instances with 22 attributes each.

This data set includes descriptions of hypothetical samples corresponding to 23 species of gilled mushrooms. Each species is identified as definitely edible, definitely poisonous, or of unknown edibility and not recommended.  This latter class was combined with the poisonous one.

The Predicted attribute is one of two classes : edible or poisonous .

### Training Pahse

For training phase I used scikit to train a model with 20% of the data , and get the best separation parametrs .

#### Iris dataset

Iris Clasiffication can be done with linear kernel with 100% accurancy, the scikit phyton training code can be found at :

From the scikit we get the svm model parametrs which is used with c# code described bellow.

For futher details and classification of this dataset can be found in : <https://www.kaggle.com/ash316/ml-from-scratch-with-iris>

<https://www.kaggle.com/biphili/seaborn-matplotlib-plot-to-visualize-iris-data>

#### Mashroom dataset

Mashroom Clasiffication can be done with linear polynimial with 100% accurancy, the scikit phyton training code can be found at :

For futher details and classification of this dataset can be found in :

# Machine Learning

# SVM

“Support Vector Machine” (SVM) is a supervised [machine learning algorithm](https://courses.analyticsvidhya.com/courses/introduction-to-data-science-2?utm_source=blog&utm_medium=understandingsupportvectormachinearticle) which can be used for both classification or regression challenges. However,  it is mostly used in classification problems. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well (look at the below snapshot).

A classification task usually involves separating data into training and testing sets. Each instance in the training set contains one “target value” (i.e. the class labels) and several “attributes” (i.e. the features or observed variables). The goal of SVM is to produce a model (based on the training data) which predicts the target values of the test data given only the test data attributes. Given a training set of instance-label pairs (xi , yi), i = 1, . . . , l where xi ∈ Rn and y ∈ {1, −1} l , the support vector machines (SVM) (Boser et al., 1992; Cortes and Vapnik, 1995) require the solution of the following optimization problem: min w,b,ξ 1 2 w T w + C X l i=1 ξi subject to yi(w T φ(xi) + b) ≥ 1 − ξi , (1) ξi ≥ 0. 1 Table 1: Problem characteristics and performance comparisons.

Here training vectors xi are mapped into a higher (maybe infinite) dimensional space by the function φ. SVM finds a linear separating hyperplane with the maximal margin in this higher dimensional space. C > 0 is the penalty parameter of the error term. Furthermore, K(xi , xj ) ≡ φ(xi) T φ(xj ) is called the kernel function. Though new kernels are being proposed by researchers, beginners may find in SVM books the following four basic kernels: • linear: K(xi , xj ) = x T i xj . • polynomial: K(xi , xj ) = (γxi T xj + r) d , γ > 0. • radial basis function (RBF): K(xi , xj ) = exp(−γkxi − xjk 2 ), γ > 0. • sigmoid: K(xi , xj ) = tanh(γxi T xj + r). Here, γ, r, and d are kernel parameters.

We propose that beginners try the following procedure first: • Transform data to the format of an SVM package • Conduct simple scaling on the data • Consider the RBF kernel K(x, y) = e −γkx−yk 2 • Use cross-validation to find the best parameter C and γ • Use the best parameter C and γ to train the whole training set5 • Test

2 Data Preprocessing 2.1 Categorical Feature SVM requires that each data instance is represented as a vector of real numbers. Hence, if there are categorical attributes, we first have to convert them into numeric data. We recommend using m numbers to represent an m-category attribute. Only one of the m numbers is one, and others are zero. For example, a three-category attribute such as {red, green, blue} can be represented as (0,0,1), (0,1,0), and (1,0,0). Our experience indicates that if the number of values in an attribute is not too large, this coding might be more stable than using a single number. 5The best parameter might be affected by the size of data set but in practice the one obtained from cross-validation is already suitable for the whole training set. 3 2.2 Scaling Scaling before applying SVM is very important. Part 2 of Sarle’s Neural Networks FAQ Sarle (1997) explains the importance of this and most of considerations also apply to SVM. The main advantage of scaling is to avoid attributes in greater numeric ranges dominating those in smaller numeric ranges. Another advantage is to avoid numerical difficulties during the calculation. Because kernel values usually depend on the inner products of feature vectors, e.g. the linear kernel and the polynomial kernel, large attribute values might cause numerical problems. We recommend linearly scaling each attribute to the range [−1, +1] or [0, 1]. Of course we have to use the same method to scale both training and testing data. For example, suppose that we scaled the first attribute of training data from [−10, +10] to [−1, +1]. If the first attribute of testing data lies in the range [−11, +8], we must scale the testing data to [−1.1, +0.8]. See Appendix B for some real examples. 3 Model Selection Though there are only four common kernels mentioned in Section 1, we must decide which one to try first. Then the penalty parameter C and kernel parameters are chosen

Find a linear decision surface (“hyperplane”) that can separate patient classes and has the largest distance (i.e., largest “gap” or “margin”) between border-line patients (i.e., “support vectors”);

• If such linear decision surface does not exist, the data is mapped into a much higher dimensional space (“feature space”) where the separating decision surface is found; • The feature space is constructed via very clever mathematical projection (“kernel trick”).

Binary classification:

<http://www.robots.ox.ac.uk/~az/lectures/ml/lect2.pdf>

Given training data (xi, yi) for i = 1 ...N, with xi ∈ Rd and yi ∈ {−1, 1}, learn a classifier f(x) such that f(xi) ( ≥ 0 yi = +1 < 0 yi = −1 i.e. yif(xi) > 0 for a correct classification.

# FHE

# Homomorphic Encryption refers to a [new type of encryption technology](https://www.microsoft.com/en-us/research/project/homomorphic-encryption) that allows computation to be directly on encrypted data, without requiring any decryption in the process.

# SEAL