****

**Project**

**Machine Learning Course**

**Student Name: Wasan Maaitah**

**Student No: 20218056**

**Instructor Name: Prof. Mohammad Al-Azzeh**

**Problem 1**

1. SVM and Logistic Regression for Classification

We will use machine learning algorithms such as SVM, Logistic Regression, and Grid Search to classify a dataset. We will optimize the hyperparameters of SVM and Logistic Regression models with GridSearchCV for best performance and use k-fold cross-validation with k=10 to evaluate their performance. By utilizing these techniques, we aim to achieve high accuracy in our classification task.

1. Datasets(1) Description

The data you have consists of 10 datasets, each with class labels of "true" and "false". These labels indicate the presence or absence of a certain characteristic or feature within the data. The task you are trying to accomplish is to classify this data using both Support Vector Machines (SVMs) and Logistic Regression.

| Data | Num Records | Num features | Class label | Balance /imbalance |
| --- | --- | --- | --- | --- |
| 1 | 121 | 30 | True:9 False:112 | imbalance |
| 2 | 63 | 30 | True:8 False:55 | imbalance |
| 3 | 107 | 30 | True:20 False:87 | imbalance |
| 4 | 36 | 30 | True:8 False:28 | imbalance |
| 5 | 101 | 30 | True:15 False:86 | imbalance |
| 6 | 498 | 22 | True:49 False:449 | imbalance |
| 7 | 1036 | 22 | True:77 False:959 | imbalance |
| 8 | 920 | 37 | True:4 False:916 | imbalance |
| 9 | 753 | 38 | True:70 False:683 | imbalance |
| 10 | 780 | 38 | True:87 False:693 | imbalance |

Table.1

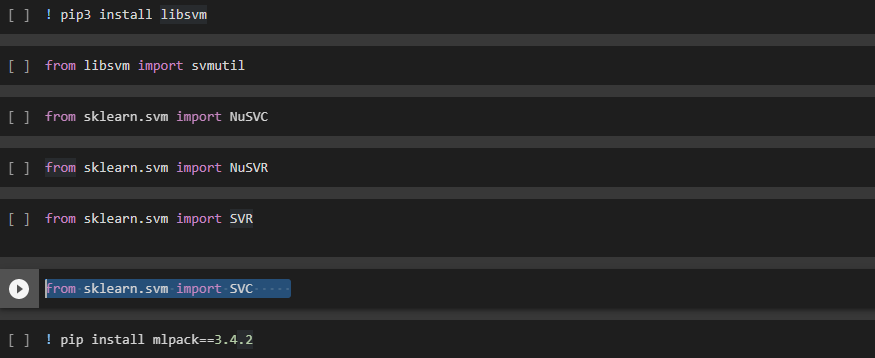
In addition to our classification models, we will use k-fold cross-validation (k=10) on all 10 datasets to evaluate the performance. This technique divides the data into k subsets, trains the model on k-1 and tests it on one, repeating the process k times to get an average performance metric.

Q1) Evaluate the performance of Support Vector Classifier and Logistic Regression, using 10-Folds …

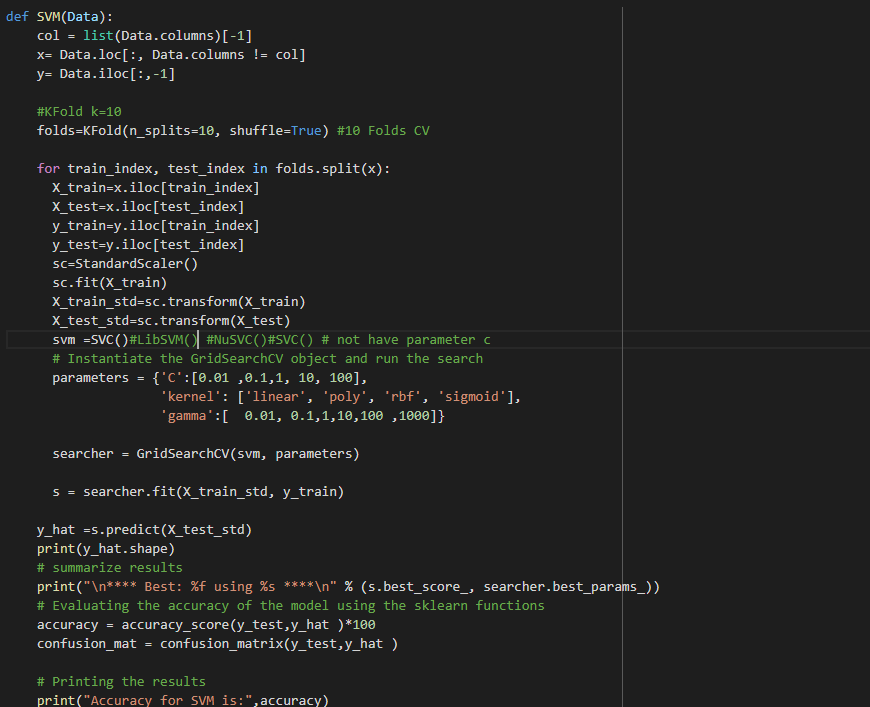
1. Machine learning implementation

3.1. SVM implementation

I found that the radial basis function kernel in scikit-learn's Support VectorClassification algorithm was slow. To improve performance, I tried alternative implementations of SVM with different parameters as (LinearSVC,LinearSVR,NuSVC, NuSVR and SVR) from sklearn and libraries such as libsvm and mlpack,also preprocessing techniques like **standard scaler**, which improved the model's performance.



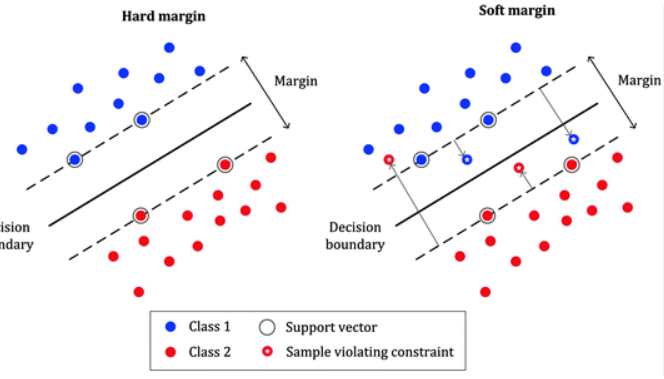
We applied standard scaler to the data before inputting it into the SVM model. This technique normalizes the data by subtracting the mean and dividing by the SD.This step makes the SVC from sklearn faster in addition to improve its performance.



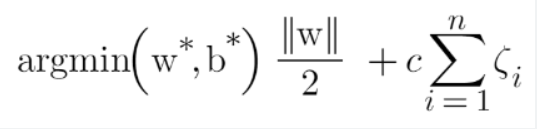
3.1.1 SVM parameters

SVM algorithms has several parameters that can be tuned to improve performance, we will focus on three main parameters: C, kernel, and gamma.

**C** is a regularization parameter in the Support Vector Machines (SVMs) algorithm, which controls the trade-off between maximizing the margin and minimizing the misclassification error. A **lower value of C** corresponds to a stronger regularization, resulting in a wider margin and more misclassifications, while a **higher value of C** corresponds to a weaker regularization, resulting in a narrower margin and fewer misclassifications.

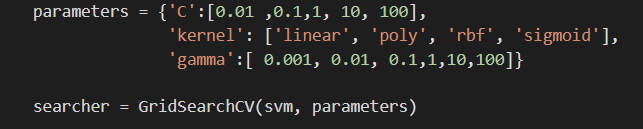


As we can see in the image, the soft margin method allows for misclassification and the control vector 'C' is added to the equation as shown in the next image, it plays a key role in controlling the trade-off between maximizing the margin and minimizing the misclassification.



**The kernel** is a key element in the SVM algorithm, it maps the input data into a higher dimensional space for easy classification by finding the optimal boundary. Different types of kernels are available, each with its own advantages and limitations, and the choice of kernel depends on the data characteristics and the classification task.

**Gamma** is a parameter in the kernel function of SVM that regulates the influence of each training example on the decision boundary. It acts as a coefficient for the RBF, poly, and sigm. **Low gamma** values result in a **softer decision** **boundary**, while **high values** result in a **harder boundary**. Gamma value is inversely proportional to the radius and affects the flexibility of the decision boundary.The appropriate value depends on data and classification task.



I implemented a grid search to find the best parameters for an SVM model. The grid search evaluated different combinations of parameters and identified the set with the highest score.This improved the performance of the SVM model.

3.1.2 Best parameters of SVM on Datasets

The table 2 presented below illustrates the optimal parameters for each dataset,through the implementation of a grid search technique.These parameters were found to yield the highest scores for the respective datasets when utilizing a SVM model.

**Best parameters in SVM:-**

| Data | c | kernel | gamma | Accuracy | F1 | MCC | Describe |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Data 1 | 0..01 | linear | 0.01 | **0.90** | **0.47** | 0.0 | F1 lower than accuracy  because the model could be classifying the majority”F” class correctly as we see in table.1 (F>T) not performing as well on the minority”T” class. |
| Data 2 | **0.01** | poly | 1 | 0.88 | 0.71 | 0.42 | smaller value of C will result in a larger margin |
| Data 3 | 0.1 | **poly** | 0.1 | 0.90 | 0.80 | 0.66 | kernel= poly that can be useful when the data is not linearly separable |
| Data 4 | **1** | sigmoid | 0.01 | 0.78 | 0.4r | 0.6 | larger value of C will result in a smaller margin |
| Data 5 | 1 | **linear** | 0.01 | 0.73 | 0.53 | 0.06 | Kernel useful when the data is not linearly separable |
| Data 6 | 0.1 | **sigmoid** | 0.01 | 0.94 | 0.48 | 0 | non-linear kernels |
| Data 7 | 10 | **rbf** | **10** | 0.96 | 0.66 | 0 | larger gamma value results in a narrower RBF kernel |
| Data 8 | 10 | sigmoid | 0.01 | **1.00** | **1.00** | 0 | Label F =4 < T =916  the majority class is "true" and the model is likely to perform well on it |
| Data 9 | 10 | rbf | 0.1 | 0.91 | **0.59** | 0.18 | the precision and recall are very different, the F1 score will be low |
| Data10 | 10 | rbf | 0.1 | 0.94 | 0.76 | 0.44 | MCC best metric to evaluate pref |

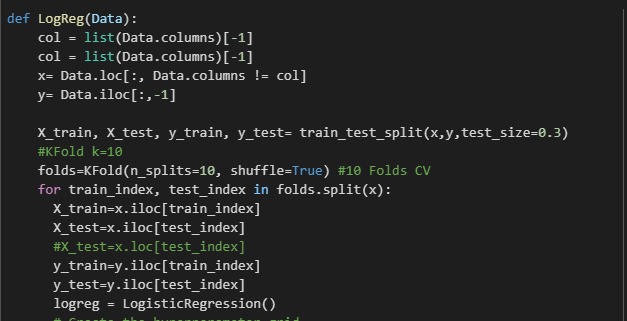
Table 2

3.1.3 Limitations and conclusion

SVM performance varies based on configuration settings. The optimal parameters were different for each dataset, such as a limited range of parameter values were evaluated. In conclusion, appropriate configuration settings are important for SVM and datasets may require different parameters for optimal performance.

3.2. Logistic Regression implementation

I used logistic regression from the sklearn library. It was faster than SVM and ran the model in ten k-folds cross-validation.



3.1.1 Logistic Regression parameters

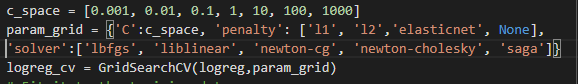
Logistic regression to be tuned to improve performance. There are several parameters that can be adjusted to optimize the model, including C, penalty, and solver.

**C parameter** controls regularization strength by being inversely proportional to lambda (λ), **Lower C** values mean stronger regularization and simpler models, while **higher C** values mean weaker regularization and more complex models, which increases overfitting risk

**The solver** used to optimize the model. Different solvers can have an impact on the model's performance and the choice of solver will depend on the specific problem and dataset. Some of the solvers are 'lbfgs', 'liblinear', 'newton-cg', 'newton-cholesky', 'saga' . Each solver is optimized for a different type of data, 'lbfgs' is good for high-dimensional problems, 'liblinear' is efficient for large number of samples and 'saga' is efficient for large datasets and sparse data. It's important to test different solvers to find the best one.

**The penalty** parameter in logistic regression determines the type of regularization applied.

* L1 regularization (Lasso) adds a penalty term based on the absolute value of the coefficients, resulting in some features being eliminated ,and L1 define by default in LR .
* L2 regularization (Ridge) adds a penalty term based on the square of the coefficients, resulting in small non-zero values for all features. The type of regularization used can greatly impact the performance of the model and should be chosen based on the specific problem and dataset.
* Elastic-net combines L1 and L2 regularization to balance the effects of L1 and L2 regularization. The l1\_ratio controls the balance between L1 and L2 regularization, which helps in feature selection and avoiding overfitting.



3.1.2 Best parameters of Logistic Regression on Datasets

Table 2 shows optimal parameters for each dataset, found by using

grid search with LR model, resulting in best scores for each dataset

**Best parameters in Logistic Regression:-**

| Data | c | penalty | solver | Accuracy | F1 | MCC | Precision |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 0.001 | l1 | saga | 0.75 | 0.43 | 0.0 | 0.38 |
| 2 | 0.001 | l1 | liblinear | 0.83 | 0.78 | 0.68 | 0.90 |
| 3 | 0.01 | l1 | liblinear | 0.80 | 0.44 | -0.08 | 0.40 |
| 4 | 0.1 | l2 | liblinear | 0.67 | 0.67 | 0.18 | 0.75 |
| 5 | 0.001 | l2 | liblinear | 1.00 | 1.00 | 0.27 | 1.00 |
| 6 | 0.001 | l1 | liblinear | 0.86 | 0.46 | -0.018 | 0.43 |
| 7 | 0.001 | l2 | newton-cg | 0.90 | 0.47 | 0.0 | 0.46 |
| 8 | 0.001 | l1 | liblinear | 0.90 | 0.50 | 0.0 | 0.49 |
| 9 | 0.01 | l2 | newton-cg | 0.93 | 0.70 | 0.0 | 0.81 |
| 10 | 0.01 | l2 | newton-cg | 0.91 | 0.77 | 0.38 | 0.95 |

Table 2

3.1.3 Limitations and conclusion

Different datasets need different settings for optimal results, as seen in this study using grid search and finding different parameters for different datasets. It's essential to be aware of its limitations and adjust the model's parameters carefully for each dataset for the best result

Q2) What is the impact of class imbalance on your results, and which is the best evaluati …?

4. Imbalanced data

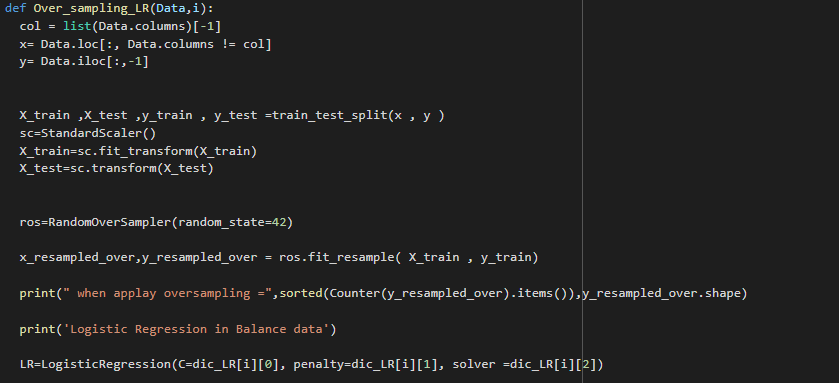
Table.1 shows that the class labels include both 'T' and 'F':majority, but there is a majority ‘F’’ and minority’T’ distribution, causing imbalanced data, the model may become biased towards the majority class, leading to lower accuracy for the minority class. Therefore, instead of relying on accuracy to evaluate the performance of the model, we should use metrics such as F1 ,recall and MCC is considered to be a better F1 in case of imbalanced datasets, and to mitigate the effects of class imbalance, techniques such as **oversampling the minority** class, **undersampling the majority class** as in section 4.1 use resampling (over/under)

Q3) Which class imbalance learning technique could work efficiently with …?

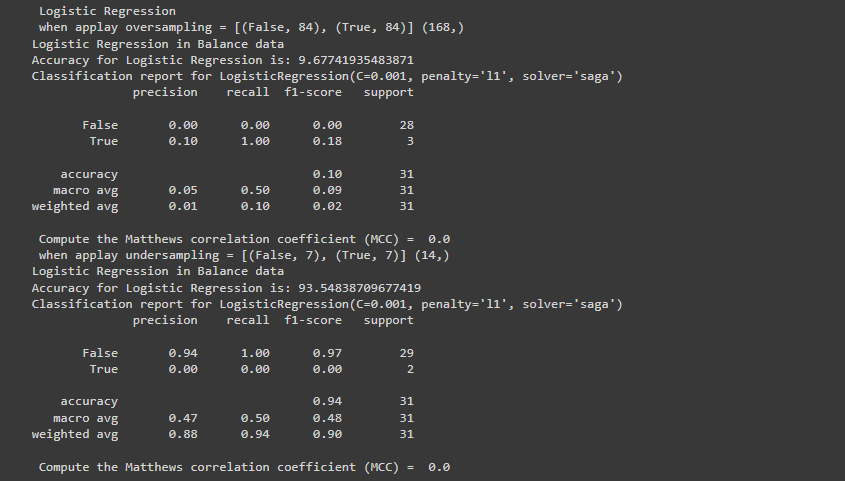
4.1 oversampling

When applying **random over-sampling** in a **logistic regression** model with the best-par

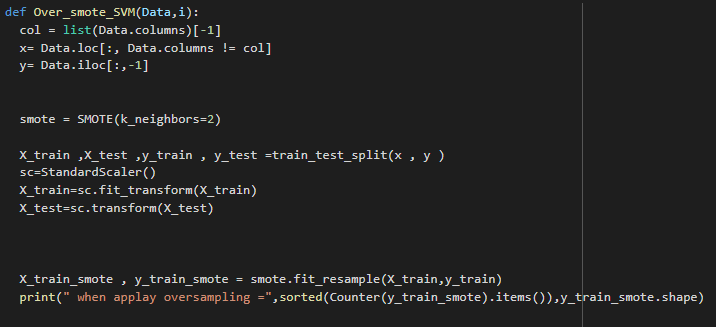
in table 3, and applying over-sampling using the **SVM** model with best-par in table 2



it improved the performance as shown below

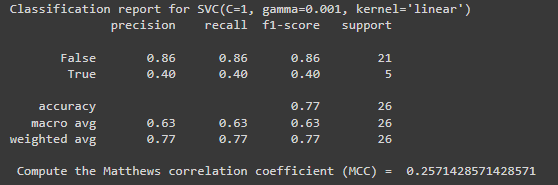


The model's performance when using SMOTE in both SVM and Logistic Regression models is shown below better than random over sampling and under sampling **,so the best performance when use SMOTE in LR and SVM**

****

4.2 undersampling

I used random under-sampling to balance the data by reducing the majority classes in SVM & LR models, but random over-sampling had better performance overall all from all technique



Q4) Is there improvement when apply feature selection? Which features do you think a…?

5. Feature selection

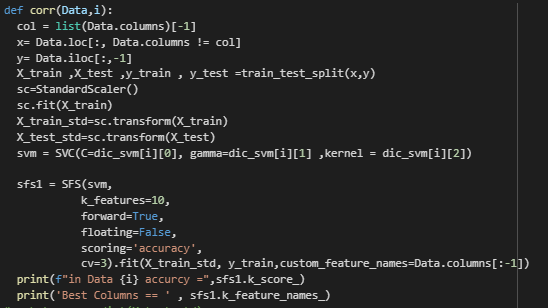
I also utilized feature selection to improve performance, by using the Forward selection method. This method resulted in a significant improvement in performance, particularly in the SVM and Logistic Regression models

5.1 why use Forward selection method

the algorithm starts with an empty set of features and iteratively adds one feature at a time to the set, based on a chosen criterion. **It calculates the accuracy after adding each feature and if the accuracy improves**,it is added to the feature set, otherwise drop

5.2 implementation

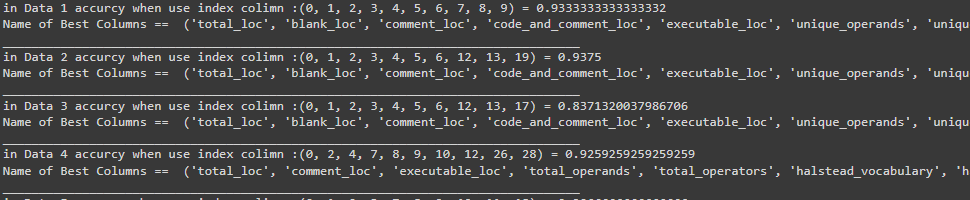
Initially, I implemented the SequentialFeatureSelector from the **mlxtend library**, defining it as **sfs**. This included the model with the best parameters for all data sets and returned the accuracy for each dataset, as well as the selected columns from feature selection.



When comparing the accuracy in Table 5 with the accuracy in Table 2 and 3, it is evident that feature selection has significantly improved the performance, as can be seen tables.

| Data | SVM scoure | Index columns | LR scoure | Index columns |
| --- | --- | --- | --- | --- |
| 1 | 0.95 | 0,1,2,3,4,5,6,7,8,9 | 0.92 | 0,1,2,3,4,5,6,7,8,9 |
| 2 | 0.95 | 0,1,2,3,4,5,12,13,19 | 0.85 | 0,1,2,3,4,5,6,7,8,9 |
| 3 | 0.86 | 0,1,2,3,4,5,6,12,13,17 | 0.82 | 0,1,2,3,4,5,6,7,8,14 |
| 4 | 0.92 | 0,2,4,7,8,9,10,12,26,28 | 0.88 | 2,3,5,12,15,23,24,26,27,28 |
| 5 | 0.85 | 0,1,2,3,7,8,9,10,11,18 | 0.89 | 0,1,2,3,4,5,6,7,8,10 |
| 6 | 0.91 | 0,1,2,3,4,5,6,7,8,9 | 0.88 | 0,1,2,3,4,5,6,7,8,10 |
| 7 | 0.93 | 0,1,2,4,5,6,9,10,11,18,20 | 0.92 | 0,1,2,3,6,7,8,10,13,16 |
| 8 | 0.99 | 0,1,2,3,4,5,6,8,10,12 | 0.99 | 0,1,2,3,4,5,6,7,8,9 |
| 9 | 0.93 | 0,1,3,6,10,13,19,22,23,31 | 0.91 | 2,5,7,11,14,16,22,25,27,35 |
| 10 | 0.92 | 0,1,2,3,9,10,11,13,14,25 | 0.91 | 1,3,4,6,7,9,11,13,14,20,21 |

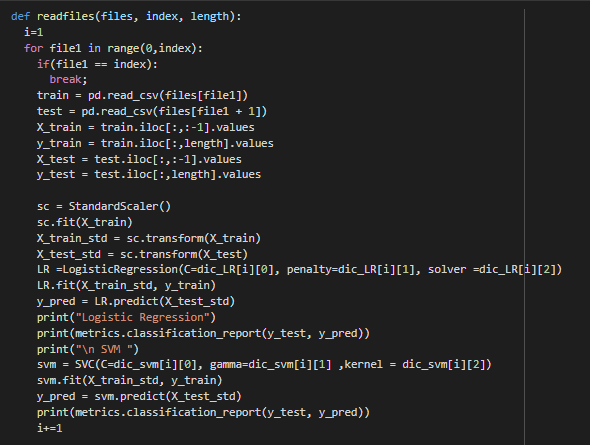
Table.5



Q5) For Datasets with multiple versions, does the accuracy improve when we use pre-release…?

6. Pre-release and pos release

The results improved because we did not split every single dataset into training and test sets. Instead, we combined two datasets, one for training and another data for testing, resulting in a larger dataset for the model to learn from. Having more data leads to better results.



when applying for pre-pos release each data( same version and feature )consider as training data and the next data as a test that returns a better result, I implemented this in two versions of data in the first version (ar : ar1.csv , ar3.csv, ar4.csv, ar5.csv, ar6.csv) :

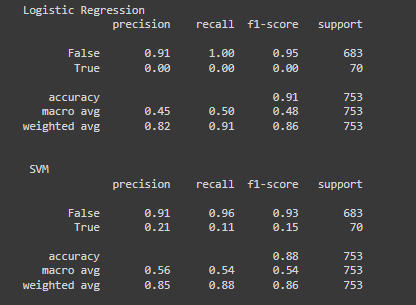
| Train | Test | SVM Accuracy | F1 | MCC | LR Accuracy | F1 | MCC |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Data 1 | Data 2 | 0.78 | 0.66 | 0.39 | 0.87 | 0.47 | 0 |
| Data 2 | Data 3 | 0.83 | 0.54 | 0.28 | 0.81 | 0.45 | 0 |
| Data 3 | Data 4 | **0.89** | **0.84** | 0.68 | 0.78 | 0.44 | 0 |
| Data 4 | Data 5 | 0.84 | 0.46 | -0.04 | 0.88 | 0.63 | 0.04 |

Table.6

| Train | Test | SVM Accuracy | F1 | MCC | LR Accuracy | F1 | MCC |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Data 8 | Data 9 | 0.88 | 0.54 | 0.09 | 0.91 | 0.48 | 0 |
| Data 9 | Data10 | 0.87 | 0.54 | 0.11 | 0.89 | 0.48 | 0.04 |

Table.7

In the second version, I focused on three different data sets, "pc2.csv", "pc3.csv" and "pc4.csv" and found that they had different features in "pc1.csv". Therefore, I selected the **common features between the files (2,3,4)**and presented the results in Table 6. Additionally, in the code, I included the F1-score and precision as evaluation measures.



Q6) what are the necessary steps that you should take care when avoiding overfitting for …?

7. Overfitting

Overfitting in SVM and Logistic Regression can be avoided by using regularization, simpler models, reducing the number of features or using simpler kernels

7.1 avoiding overfitting for logistic regression

**Regularization** is a technique that helps prevent overfitting by limiting the model's complexity. And can be applied by using L1 (LASSO) or L2 (RIDGE), which reduces variance and decrease overfitting on training data by increasing bias. To further decrease overfitting, one can use more data and adequate training data

7.2 avoiding overfitting for Support vector machine

**Soft margins** in SVM allow for some misclassifications to avoid overfitting by introducing slack variables. The margin size can be controlled by adjusting the **C** parameter, which balances margin size and a number of misclassification.

**Problem 2**

8. Datasets(2) Description

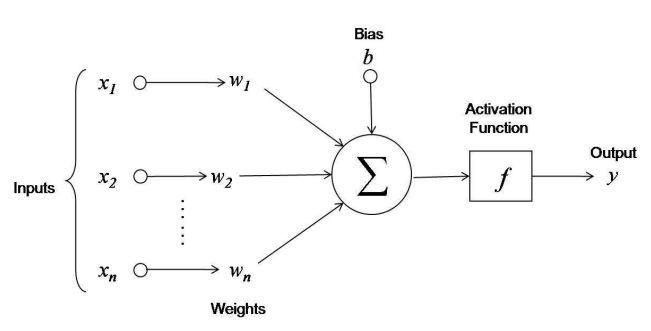
We have a dataset (Diagnosing Dementia )for images that are labeled with 4 different categories: Mild\_Demented, Moderate\_Demented, Non\_Demented, and Very\_Mild\_Demented. Each image in the dataset has a resolution of 128x128 pixels and is represented in grayscale, meaning that each image is a 3-dimensional tensor with shape (128, 128, 1)

9. Artificial Neural Networks (ANN) Convolutional Neural Networks (CNN)

ANN are a type of machine learning algorithm that simulate the structure and function of biological neurons. They can be used for a wide range of tasks such as image classification, natural language processing, and prediction.CNN are a type of ANN that are specifically designed for image and video recognition tasks, they use convolutional layers and pooling layers to learn features and hierarchies.

Q1) Develop appropriate Artificial Neural Network model to predict the status of patient?

9.1 ANN



9.1.1 Forward propagation

The forward propagation process in an ANN for image classification involves

passing the image through the layers of the network to produce the final output.

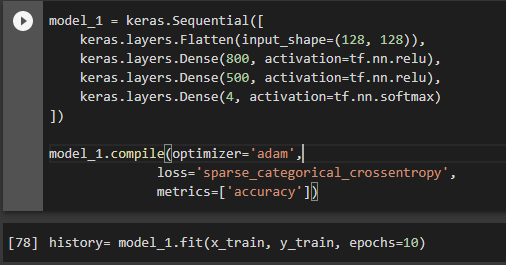
9.1.2 Backword propagation

minimize the error between the predicted and actual output by calculating the

gradients of the error and updating the weights.

9.1.3 ANN implementation

ANN model using the Keras library and after splitting the dataset into features (X) and labels (y) an preprocessing the images, different models were defined, each one differing in the num of layers and optimizer used.The model with the highest accuracy was adopted.



The model is a Sequential model, which means that it is a linear stack of layers as table 4 then Compile model with optimizer, error calculation and accuracy metrics, then train for

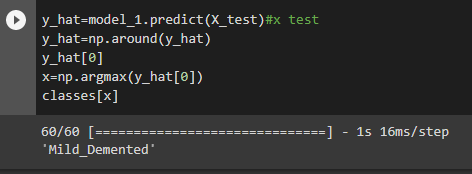
10 epochs

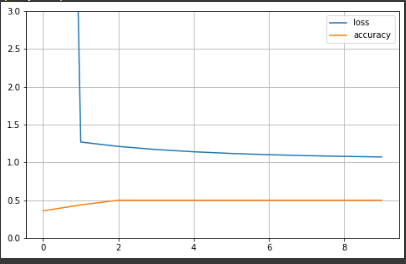
| layers | Name in keras | Activatiofunction | Output | Input to layers | Describe |
| --- | --- | --- | --- | --- | --- |
| Input 1 | Flatten  **128 \*128** |  | a1 |  | Define shape as number of features in input layer so used to reshape the input image of shape (128,128) |
| Hidden layer 1 | **Dens**  **(800)** | Relu | a2 | Wight\*a1 | Define hidden layer by Dens , Receive from input layer |
| Hidden layer 2 | Dens  (500) | **Relu** | a3 | Wight\*a2 | each hidden layer same act-fun so 800 neuron with Relu |
| output | Dens(4) | **softmax** | probability distribution over multiple classes | Wight\*a3 | Last layer i have 4 output so define unit (number of neuron)=4 and the softmax classify them according to the highest probability. |

Table.8

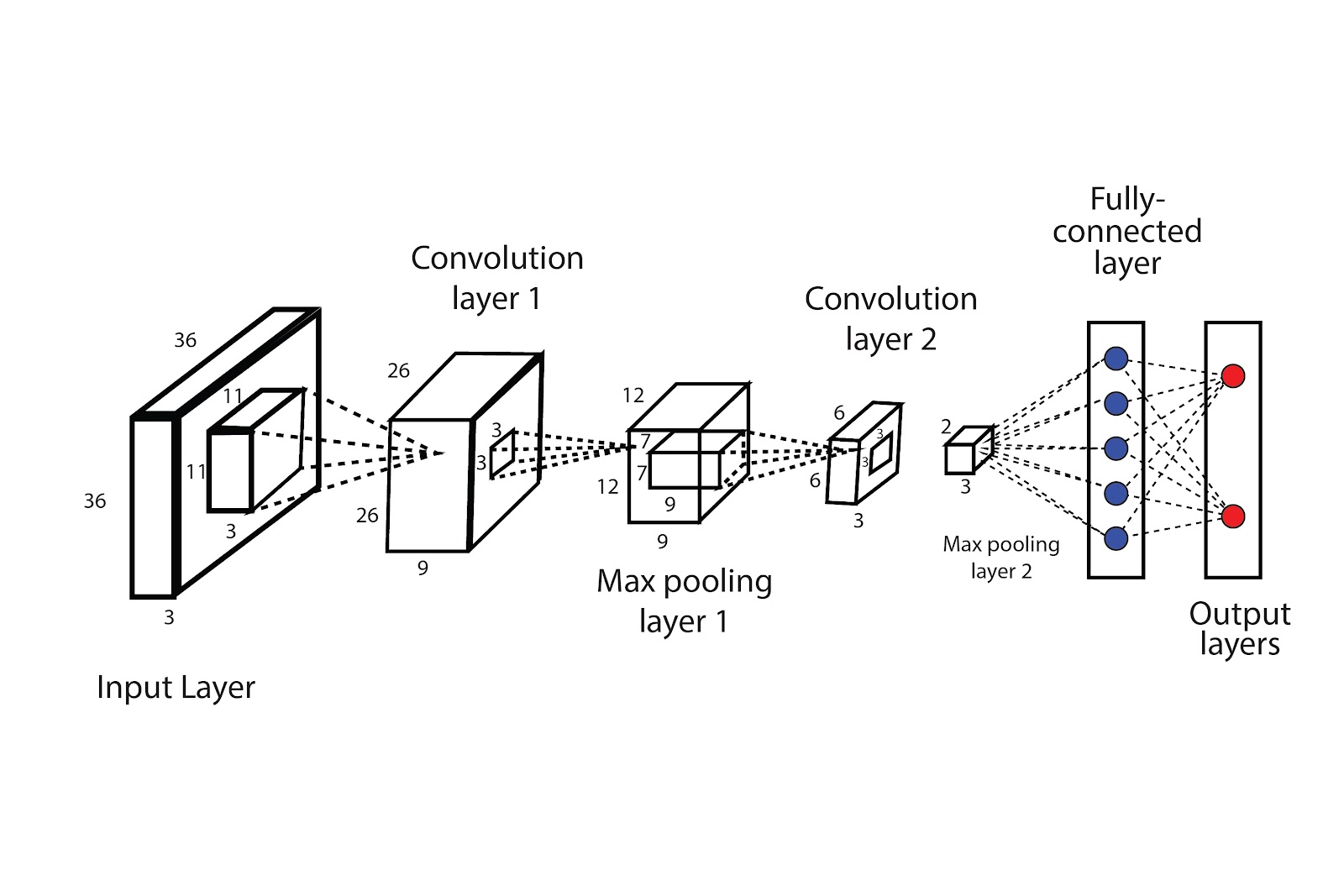
after the training process is complete, you can use the model.evaluate() function to

evaluate the model on a set of test data and see the accuracy and loss of the model



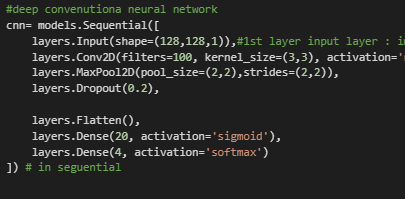


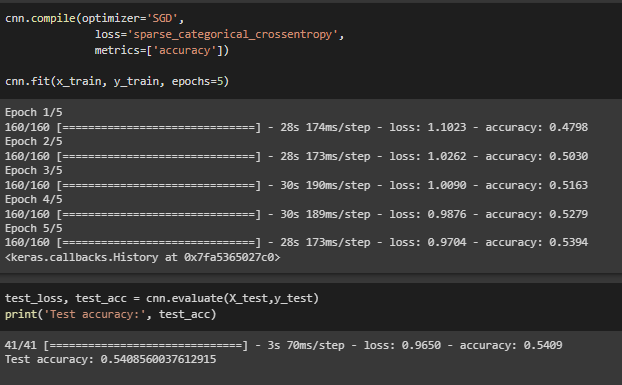
Q2) Develop appropriate Deep Learning model using TensorFlow to predict the status 9.2 Convolutional Neural Networks (CNN)



9.2.1 CNN implementation

Convolutional Neural Networks (CNNs) are preferred for image processing tasks, so when implemented, I imported the Keras library which has a variety of layers such as input, dense, flatten, etc. I used the Sequential, which is an array of objects from these layers, to build the CNN model





Q3) What are the parameters that you used to configure your model; describe how did you tune your module?

9.2.2 CNN parameter

| Parameter | include | describing |
| --- | --- | --- |
| input layer | shape image(128, 128, **1**) | (1) due to the fact that the images are in grayscale |
| Drop out: | 20% (impact of layers) | to avoid effect overfitting |
| convolutional layer | Conv2D( 100 filters , Relu and shape activation fun) | to map filters through the image |
| pooling layer | Padding=add zero values  strides= num shift steps | MaxPooling or AvgPooling |
| Feture Extraction | >>> | “All above” |
| Flattan layer | Start to learning model | switches the data from a 2-dimensional format to a 1-dimensional format. |
| Dens layer | Number of nurons and activation function | Hidden layer to learn model |
| Learning | >>> | “above” |

Table.9

And I applied different types of optimizers and activation fun. I found that using Adam as an optimizer and ReLU as an activation function resulted in the highest accuracy, **(In code in file “cell”Test all trials )**

Q4) Can we apply LSTM and GAN for such problems, explain your answer with evidence?

* LSTM: are particularly effective in handling **sequential data such as time series or text** due to their ability to maintain a memory of past inputs through hidden states
* LSTMs and CNNs have **different architectures** and are designed for different types of problems. While **CNNs are specificallyfor image employ filters to map images and extract features** as data discuss in problem2 , **LSTMs have complex hidden layers that ability to remember past information through the hiddenstates**
* while LSTMs can be used for image processing tasks, they are **not typically used for image classification tasks** with multiple labels such as medical image data with 4 labels, because CNNs are optimized for extracting features from images and handling large-scale image datasets
* GAN: used for **image generation tasks** and although they can be used for image classification, **it are not optimized for this task**. GANs are **designed to learn patterns and features from real images** to generate new images
* So GANs not used for image classification with multiple labels, because GANs are primarily used for image generation tasks, not for image classification. .

