

T.R.
GEBZE TECHNICAL UNIVERSITY
FACULTY OF ENGINEERING
DEPARTMENT OF COMPUTER ENGINEERING

**DETECTION OF ALZHEIMER'S DISEASE USING
MRNA GENE EXPRESSION DATA**

CANBERK ARICI

**SUPERVISOR
ASST. PROF. HABIL KALKAN**

**GEBZE
2022**

T.R.
GEBZE TECHNICAL UNIVERSITY
FACULTY OF ENGINEERING
COMPUTER ENGINEERING DEPARTMENT

DETECTION OF ALZHEIMER'S DISEASE
USING MRNA GENE EXPRESSION DATA

CANBERK ARICI

SUPERVISOR
ASST. PROF. HABIL KALKAN

2022
GEBZE



GRADUATION PROJECT
JURY APPROVAL FORM

This study has been accepted as an Undergraduate Graduation Project in the Department of Computer Engineering on 03/03/2022 by the following jury.

JURY

Member

(Supervisor) : Asst. Prof. Habil Kalkan

Member : Prof. Dr. Yusuf Sinan Akgül

ABSTRACT

This project aims to detect Alzheimer's Disease by using mRNA gene expression data.

Three different datasets are provided, and these datasets are combined in a proper way. Datasets include 3 classes which AD(Alzheimer's Disease), MCI(Mild Cognitive Impairment), CTL(Control). 5 different deep learning models are performed to classify these samples. Different architectures are observed for each models. Appropriate techniques are applied in order to increase performance. AUC Score is used to compare model performances.

In test phase, 22% of the dataset is used for testing. 5 different architectures are aimed to be compared, which are Convolutional Neural Network(CNN), Multilayer Perceptron(MLP), Support Vector Machine(SVM), Random Forest(RF), K-Nearest Neighbor Classifier(KNN).

Keywords: CNN, MLP, SVM, RF, KNN.

ÖZET

Bu proje, mRNA Gen ekspresyon verilerini kullanarak Alzheimer Hastalığı'nı tespit etmeyi amaçlamaktadır.

Üç farklı veri kümesi sağlanmıştır ve bu veri kümeleri uygun bir şekilde birleştirilmiştir. Veri kümeleri, AD(Alzheimer Hastalığı), MCI(Hafif Bilişsel Bozukluk), CTL(Kontrol) olmak üzere 3 sınıf içerir. Bu örnekleri sınıflandırmak için 5 farklı derin öğrenme modeli gerçekleştirilmiştir. Her model için farklı mimariler gözlemlenmiştir. Performansı artırmak için uygun teknikler uygulanır. AUC Puanı, model performanslarını karşılaştırmak için kullanılır.

Test aşamasında, veri setinin %22'si test için kullanılır. Evrimsel Sinir Ağları(CNN), Çok Katmanlı Algılayıcı(MLP), Destek Vektör Makinesi(SVM), Random Forest(RF), K-Nearest Neighbor Sınıflandırıcı(KNN) olmak üzere 5 farklı mimarinin karşılaştırılması amaçlanmıştır.

Anahtar Kelimeler: CNN, MLP, SVM, RF, KNN.

ACKNOWLEDGEMENT

I would like to express my sincere thanks to those who contributed to the preparation of this report, and to Asst. Prof. Habil Kalkan, who set an example for me both with his personality and his academic life and studies.

In addition, I would like to express my respect and love to my family, who supported me in every way during my education life, and to all people who contributed to my development.

Canberk ARICI

LIST OF SYMBOLS AND ABBREVIATIONS

Symbol or

Abbreviation : Explanation

CNN	: Convolutional Neural Network
MLP	: Multilayer Perceptron
SVM	: Support Vector Machine
RF	: Random Forest
KNN	: K-Nearest Neighbor Classifier

CONTENTS

Abstract	iv
Özet	v
Acknowledgement	vi
List of Symbols and Abbreviations	vii
Contents	viii
List of Figures	ix
List of Tables	x
1 BACKGROUND	1
2 METHOD	3
2.1 Dataset	3
2.2 Architectures	5
2.2.1 Convolutional Neural Network(CNN)	5
2.2.2 Multilayer Perceptron(MLP)	8
2.2.3 Support Vector Machine(SVM)	10
2.2.4 Random Forest(RF)	11
2.2.5 K-Nearest Neighbors(KNN)	13
3 EXPERIMENTS AND FINDINGS	15
4 Conclusions	17
Bibliography	18

LIST OF FIGURES

1.1	Simple Neural Network	1
1.2	Artificial Neural Network Architecture	2
2.1	Convolutional Neural Network Architecture	5
2.2	Input of CNN	6
2.3	Output of CNN	6
2.4	Summary of CNN	6
2.5	CNN Architecture in the Project	7
2.6	MLP Architecture	8
2.7	MLP Architecture in the Project	9
2.8	An Example Representation of SVM Classification	10
2.9	Architecture of The Random Forest	12
2.10	An Example Representation of KNN Classification	14

LIST OF TABLES

3.1	Models and F1 Scores	15
3.2	Models and AUC Scores	15

1. BACKGROUND

Artificial intelligence enables people to make life easier and to reach places that people cannot reach. Artificial intelligence is being applied to many different fields today and facilitates the solution of problems. Artificial intelligence has structures called neural networks. Neural networks, also known as artificial neural networks (ANNs) are a subset of machine learning and are at the heart of deep learning algorithms. Neural networks are a set of algorithms that simulate the functions of an animal brain in order to recognize patterns in large volumes of data. As a result, they resemble the neuronal and synaptic connections seen in the brain. Deep learning methods use neural networks with several process layers, which are referred to as "deep" networks.

Although machine learning-based approaches show examples of success, features such as the quality of the data and the complexity of the algorithm affect the level of success. As the complexity and operations of mathematical problems increase, it becomes more difficult for people to solve them. Different approaches have been used in solving health problems from past to present.

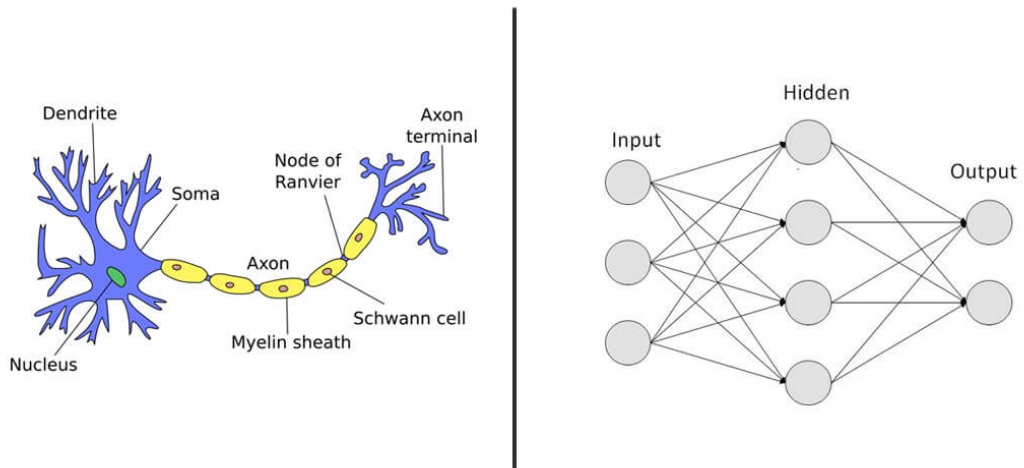


Figure 1.1: Simple Neural Network

[1]

The increase in the amount of data we have due to the development of technology and the fact that the computers that are powered by hardware can perform complex mathematical operations faster, have enabled the realization of neural networks and different architectures which use neural networks, with the emergence and development of graphics cards (GPU) based on parallel processor structures.

When an observation is made on the studies and articles about detection of Alzheimer's Disease, it is seen that ANN and KNN are generally used[2] [3]. It can be attributed that the widespread use of neural networks to the fact that the hardware of the computers at the time of the research was not sufficient or that other algorithms did not find enough space in the field of health research yet. Datasets which are used in experiments include total "Gene Symbol" being 11618; There are 482 AD(Alzheimer Disease), 313 MCI, 467 control samples.

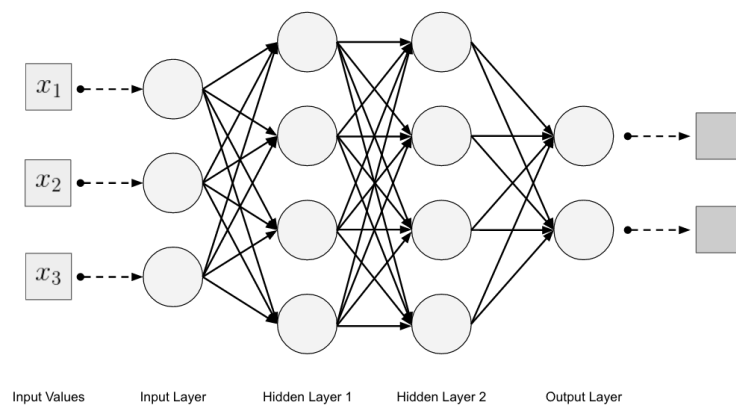


Figure 1.2: Artificial Neural Network Architecture

[4]

2. METHOD

In this section, methods and architectures related to the project will be introduced.

2.1. Dataset

3 different datasets are provided as combined by paying attention to common mRNA genes.

Total "Gene Symbol" being 11618; There are 482 AD(Alzheimer Disease), 313 MCI, 467 control samples. MCI(Mild Cognitive Impairment) is a transitional stage between normal aging and Alzheimer's Disease.

- GSE63060 Normalized Dataset; 29958 Gene Symbol, 329 Samples: 145 AD, 80 MCI, 104 Control[5]

Alzheimer, MCI and control samples from AddneuroMed Cohort (batch 1). Alzheimer case-control samples originate from the EU funded AddNeuroMed Cohort, which is a large cross-European AD biomarker study relying on human blood as the source of RNA. The design is case-control. Cases are either Alzheimer's disease patients, subjects with mild cognitive impairment or age and gender matched controls.

- GSE63061 Normalized Dataset; 24900 Gene Symbol, 388 Samples: 139 AD, 109 MCI, 134 Control [6]

Alzheimer, Mild Cognitive impairment and control samples from AddneuroMed Cohort (batch 2). Alzheimer case-control samples originate from the EU funded AddNeuroMed Cohort, which is a large cross-European AD biomarker study relying on human blood as the source of RNA. The design is case-control. Cases are either Alzheimer's disease patients, subjects with mild cognitive impairment or age and gender matched controls.

- GSE140829 Normalized Dataset; 15987 Gene Symbol, 551 Samples: 198 AD, 124 MCI, 229 Control [7]

Systems-level analysis of peripheral blood gene expression in dementia patients reveals an innate immune response shared across multiple disorders. The role of peripheral inflammation in dementia is an important but complex topic. We present here the largest cohort of peripheral blood gene expression data ever assembled from patients with dementia and matching controls. Importantly, this cohort includes individuals from a diverse set of dementia disorders, including Alzheimer's Disease (AD), mild cognitive impairment (MCI), and multiple disorders within the frontotemporal dementia (FTD) spectrum. We found strong transcriptional evidence of an innate immune inflammatory response, mediated by monocytes and neutrophils, in AD, MCI, and two FTD subtypes, PSP and nfvPPA. This transcriptional inflammatory response is enriched for genetic risk for AD, in part because it is also enriched for microglial genes, which have previously been implicated in AD risk. Finally, we show that this transcriptional response is strongly enriched for binding of the transcription factors PU.1 and RELA, which have previously been linked to AD risk and progression.

Two of the classes which are AD and CTL are used to train models in order to increase model performances.

2.2. Architectures

2.2.1. Convolutional Neural Network(CNN)

When a human perceives an image, the human brain analyzes a massive amount of data. Each neuron has its own receptive field and is coupled to other neurons so that the full visual field is covered. Each neuron in a CNN processes data only in its receptive field, just as each neuron in the biological vision system responds to stimuli only in the confined portion of the visual field termed the receptive field. Simpler patterns are detected first by the layers, which are structured in such a way that they detect them first. One can give computers sight by utilizing a CNN.

Convolutional neural networks are a type of deep learning methods that has grown popular in computer vision and is attracting interest from a wide range of fields. Convolutional neural networks are made up of several layers, such as convolution layers, pooling layers, and fully connected layers, and are used to learn spatial hierarchies of information automatically and adaptively using a backpropagation technique.

Convolutional neural networks are generally used in computer vision but they can also be used in 1D classification. In this project, CNN is used to predict Alzheimer's Disease by using 1D data.

??.

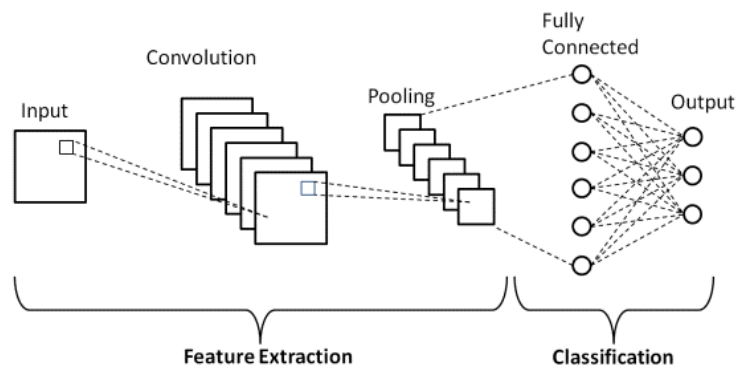


Figure 2.1: Convolutional Neural Network Architecture

[8]

Input & Output in Tensorflow

	Input				
tf.nn.conv1d	batch	in_width	in_channels		
tf.nn.conv2d	batch	in_height	in_width	in_channels	
tf.nn.conv3d	batch	in_depth	in_height	in_width	in_channels

Figure 2.2: Input of CNN

[9]

	Output				
tf.nn.conv1d	batch	filter_width	out_channels		
tf.nn.conv2d	batch	filter_height	filter_width	out_channels	
tf.nn.conv3d	batch	filter_depth	filter_height	filter_width	out_channels

Figure 2.3: Output of CNN

[9]

Summary of CNN




	Conv Direction	Input	Filter	Output
tf.nn.conv1d	1-direction 	3-dim	3-dim	2-dim
tf.nn.conv2d	2-direction 	4-dim	4-dim	3-dim
tf.nn.conv3d	3-direction 	5-dim	5-dim	4-dim

Figure 2.4: Summary of CNN

[9]

Here is the CNN Architecture in the project:

```
def build_model(TIME_PERIODS):
    input_shape=(TIME_PERIODS,)
    model = Sequential()
    model.add(keras.Input(shape=(TIME_PERIODS,1)))
    model.add(Conv1D(512, 2, strides=2, input_shape=(TIME_PERIODS,1)))
    model.add(Activation('relu'))
    model.add(Conv1D(512, 2, strides=2, kernel_initializer='glorot_uniform', padding="same"))
    model.add(Activation('relu'))
    model.add(Conv1D(256, 2, strides=2, kernel_initializer='glorot_uniform', padding="same"))
    model.add(Activation('relu'))
    model.add(Conv1D(256, 2, strides=2, kernel_initializer='glorot_uniform', padding="same"))
    model.add(Activation('relu'))
    model.add(Conv1D(128, 2, strides=2, kernel_initializer='glorot_uniform', padding="same"))
    model.add(Activation('relu'))
    model.add(Conv1D(128, 2, strides=2, kernel_initializer='glorot_uniform', padding="same"))
    model.add(Activation('relu'))
    model.add(Conv1D(64, 1, strides=2, kernel_initializer='glorot_uniform', padding="same"))
    model.add(Activation('relu'))
    model.add(Conv1D(64, 1, strides=2, kernel_initializer='glorot_uniform', padding="same"))
    model.add(Activation('relu'))
    model.add(Conv1D(32, 1, strides=1, kernel_initializer='glorot_uniform', padding="same"))
    model.add(Activation('relu'))
    model.add(Dropout(0.3))
    model.add(Dense(32))
    model.add(Activation('relu'))
    model.add(GlobalAveragePooling1D())
    model.add(Dropout(0.3))
    model.add(Dense(2))
    model.add(Activation('softmax'))
    return(model)
```

Figure 2.5: CNN Architecture in the Project

2.2.2. Multilayer Perceptron(MLP)

MLP is a neural network with a non-linear mapping between inputs and outputs. Input and output layers, as well as one or more hidden layers with many neurons stacked together, make up a Multilayer Perceptron. While neurons in a Perceptron must have an activation function that enforces a threshold, such as ReLU or sigmoid, neurons in a Multilayer Perceptron can have whatever activation function they like.

2.7.

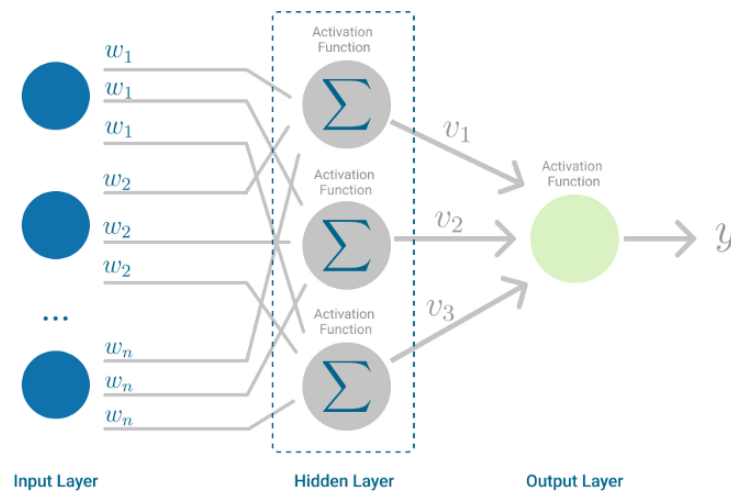


Figure 2.6: MLP Architecture

[10]

In supervised learning problems, multilayer perceptrons are frequently used. They learn to model the correlation (or dependency) between inputs and outputs by training on a collection of input-output pairs. In order to minimize error, the model's parameters, or weights and biases, are adjusted during training. MLPs are suitable for classification prediction problems where inputs are assigned a class or label therefore MLP is one of the architectures in the project.

In the MLP Architecture, 2 hidden layers are used which are Dropout Layer and Dense Layer. There is no need to reset MLP model to its original value in each epoch because information is not accumulated. Categorical Cross Entropy is used for loss calculation. Adam Optimizer is used because Adam Optimizer generally works well with neural networks and also it performed better than other optimizers tried.

Here is the MLP Architecture used in the project:

```
def build_model(optimizer=opt, init_mode='he_uniform'):
    model = Sequential()
    model.add(Dense(64, input_dim=input_dim, kernel_initializer=init_mode, activation='relu'))
    model.add(Dropout(0.1))
    model.add(Dense(64, kernel_initializer=init_mode, activation='relu'))
    model.add(Dense(128, kernel_initializer=init_mode, activation='relu'))
    model.add(Dense(128, kernel_initializer=init_mode, activation='relu'))
    model.add(Dense(256, kernel_initializer=init_mode, activation='relu'))
    model.add(Dense(256, kernel_initializer=init_mode, activation='relu'))
    model.add(Dense(512, kernel_initializer=init_mode, activation='relu'))
    model.add(Dense(512, kernel_initializer=init_mode, activation='relu'))
    model.add(Dropout(0.1))
    model.add(Dense(256, kernel_initializer=init_mode, activation='relu'))
    model.add(Dropout(0.1))
    model.add(Dense(256, kernel_initializer=init_mode, activation='relu'))
    model.add(Dropout(0.1))
    model.add(Dense(128, kernel_initializer=init_mode, activation='relu'))
    model.add(Dropout(0.1))
    model.add(Dense(128, kernel_initializer=init_mode, activation='relu'))
    model.add(Dense(64, kernel_initializer=init_mode, activation='relu'))
    model.add(Dense(64, kernel_initializer=init_mode, activation='relu'))
    model.add(Dense(32, kernel_initializer=init_mode, activation='relu'))
    model.add(Dense(32, kernel_initializer=init_mode, activation='relu'))
    model.add(Dense(2, kernel_initializer=init_mode, activation='softmax'))

    # compile model
    model.compile(loss='categorical_crossentropy',
                  optimizer='adam',
                  metrics=['accuracy'])

    return model
```

Figure 2.7: MLP Architecture in the Project

2.2.3. Support Vector Machine(SVM)

The Support Vector Machine (SVM) is a linear model that can be used to solve classification and regression problems. It can solve both linear and nonlinear problems and is useful in a variety of situations. SVM has a straightforward concept: The method divides the data into classes by drawing a line or a hyperplane.

What is a hyperplane?

A hyperplane is a plane that is parallel to another plane. A hyperplane can be conceived of as a line that linearly separates and classifies a set of data, for example, for a classification problem with only two features.

Intuitively, the further the data points are from the hyperplane, the more certain it is that they have been classified correctly. As a result, the data points should be as far away from the hyperplane as feasible while being on the correct side.

As a result, the class assigned to new testing data is determined by which side of the hyperplane it lands on.

SVM can perform good in classification tasks therefore SVM is selected to be one of the five models to compared.

2.7.

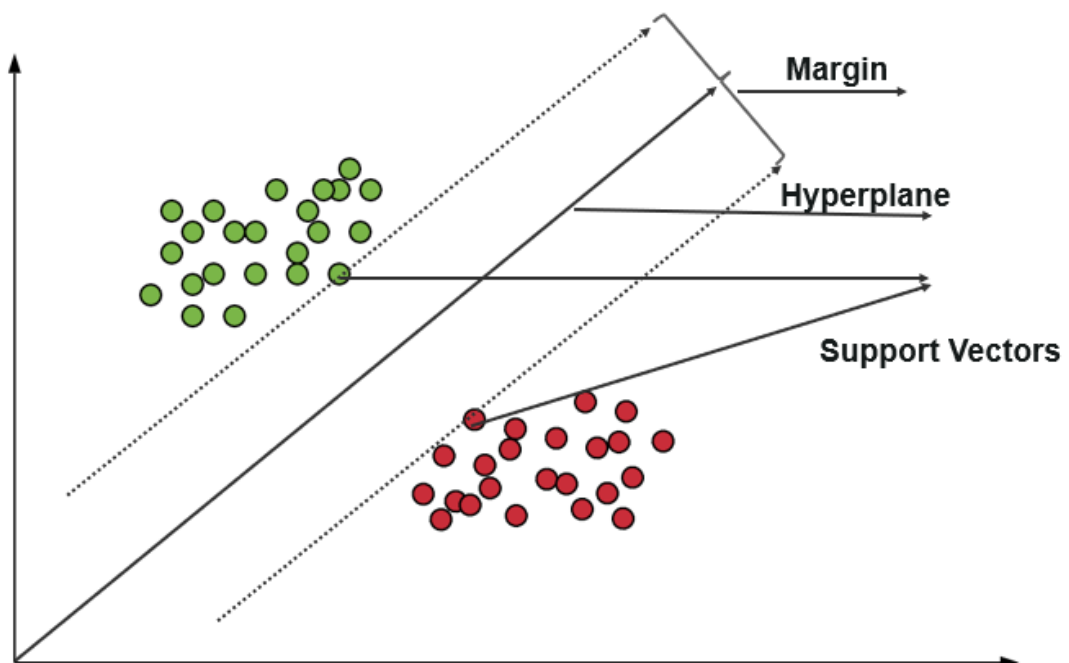


Figure 2.8: An Example Representation of SVM Classification

[11]

2.2.4. Random Forest(RF)

As the name implies, a random forest is made up of a huge number of individual decision trees that work together as an ensemble. Each tree in the random forest produces a class prediction, and the class with the most votes becomes the prediction of our model.

Random forest's core principle is the wisdom of crowds, which is both simple and powerful. Any of the individual constituent models will outperform a large number of reasonably uncorrelated models (trees) working as a committee.

The key is the lack of correlation between models. Uncorrelated models can provide ensemble predictions that are more accurate than any of the individual predictions, similar to how investments with low correlations (such stocks and bonds) combine together to build a portfolio that is larger than the sum of its parts. The explanation for this success is that the trees shield each other from their individual errors (as long as they don't all make the mistake in the same direction).

While some trees will be incorrect, many more will be correct, allowing the trees to progress in the appropriate direction as a group. As a result, the following are the requirements for a successful random forest:

- In order for models built with those features to outperform random guessing, there needs to be some genuine signal in the features.
- Individual trees need to have low correlations between their predictions (and thus mistakes).

Random Forest Classifier can perform good in classification tasks and is used in some health related researches therefore Random Forest Classifier is selected to be one of the five models to compared.

Architecture of The Random Forest 2.7.

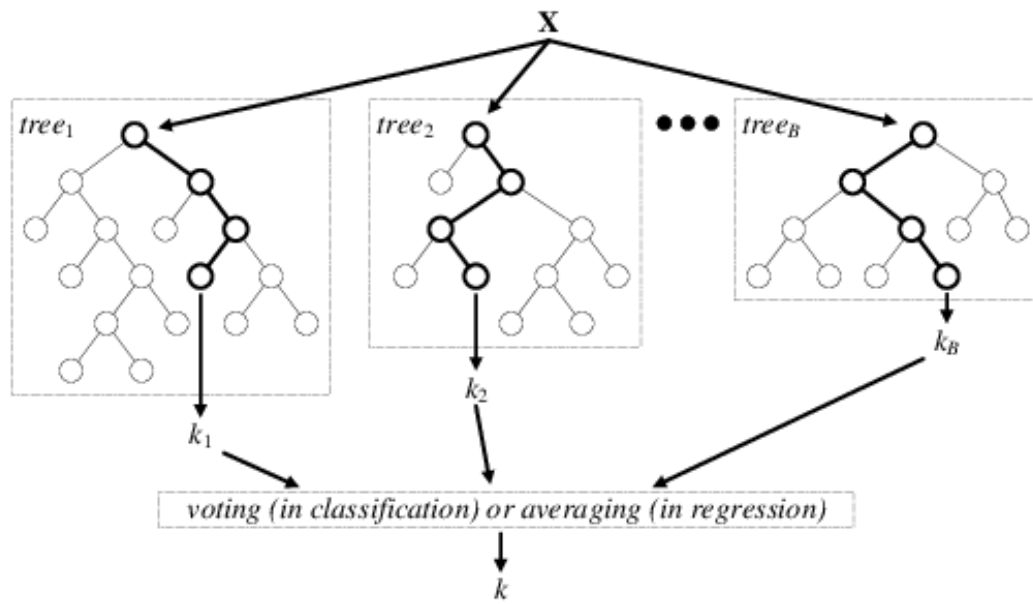


Figure 2.9: Architecture of The Random Forest

[11]

2.2.5. K-Nearest Neighbors(KNN)

The supervised machine learning algorithm K-nearest neighbors (kNN) can be used to handle both classification and regression tasks. kNN is an algorithm like coming from real life. The individuals surrounding them have an impact on people. Friendships form a strong foundation for human behavior. In certain ways, parents influence a child's personality. It is quite likely that a human will love fitness if he or she grows up among people who do. Exceptions are certain to exist. In the same way, kNN works.

The data points that surround a data point define its value. The majority voting principle is used by a kNN classifier to decide the class of a data point. When the value of k is set to 5, the classes of the five closest points are examined. Prediction is done according to the majority class.

What criteria is used to determine how close data points are to one another?

It's calculated how far apart the data points are. Distance can be measured in a variety of ways such as by using one of the most popular distance measurements Euclidean distance (minkowski distance with $p=2$).

KNN Classifier is selected to be one of the five models to compared since KNN Classifier can perform good in classification tasks and is used in some health related researches.

An Example Representation of KNN Classification 2.7.

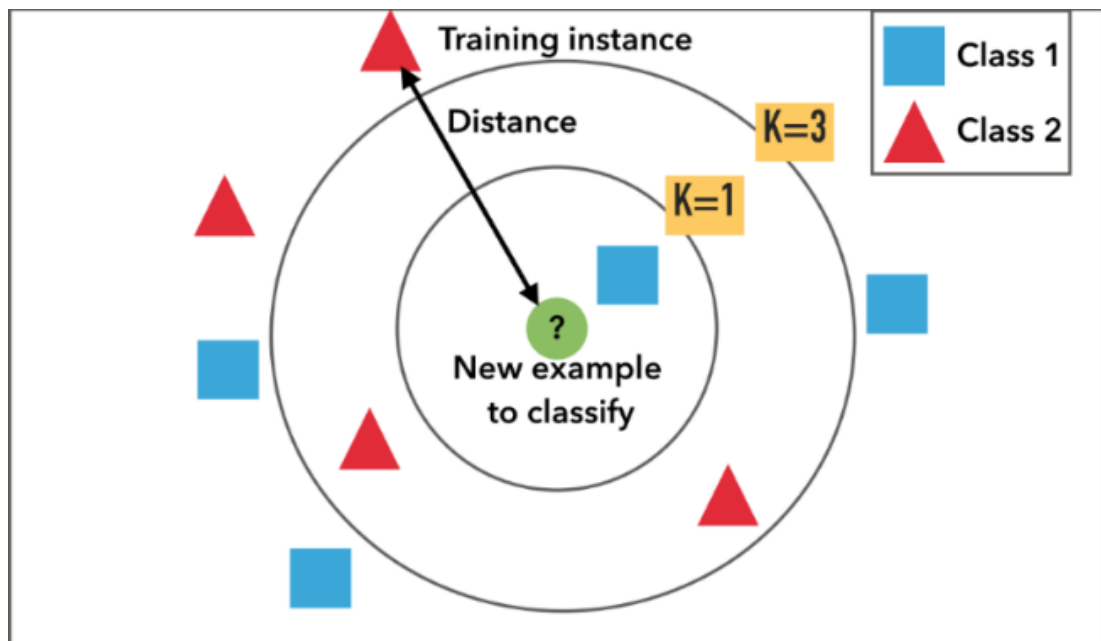


Figure 2.10: An Example Representation of KNN Classification

[12]

3. EXPERIMENTS AND FINDINGS

MODEL	F1 SCORE
CNN	0.73
MLP	0.69
SVM	0.69
KNN	0.70
RF	0.65

Table 3.1: Models and F1 Scores

MODEL	AUC SCORE
CNN	0.82
MLP	0.726
SVM	0.723
KNN	0.685
RF	0.626

Table 3.2: Models and AUC Scores

Before datasets are provided, normalization was applied to the datasets before and after combination. PCA is applied to the combined dataset and number of components is decided by taking what percentage of the information is kept into consideration. Number of components is decided as 225 by keeping 99% of the information. One Hot Encoding and Label Encoding are applied to the class feature for training the models and evaluating the results. Results are also evaluated by plotting Confusion Matrix, ROC Curve and evaluating AUC Score. The training was conducted with 22% testing and 78% training data. GridSearchCV is used to evaluate model performances and find optimal parameters for each model. Different parameters are tried in architectures. Common CNN and MLP architectures are tried such as AlexNet but results were not as good as expected therefore architectures of CNN and MLP models are created from scratch by manual evaluation and with different parameters.

Training time can be decreased by training on a better computer or by using gpu. In order to prevent overfitting in CNN and MLP, EarlyStopping of Keras is used and optimal number of epochs is selected by observing learning curves. The overall auc scores and f1 scores are given in the tables above. The auc scores and f1 scores in the table were obtained by giving the test data, which is 22% of the dataset, to the models.

4. CONCLUSIONS

Input, output sizes, the way the train and test data are given to the models, the layer types and parameters that are considered while creating the architecture of the models are the most important points that affect the result. In general, the project of detection of Alzheimer's Disease by using mRNA gene expression data by using different machine learning models has reached a result that can be considered successful on the test data. The dataset can be enlarged by finding datasets related to detection of Alzheimer's Disease which have common genes with the dataset used in this project. In this way, performance of the models can be increased.

In order to minimize the time of training phase, the dataset can be given to the model in parts or the models can be trained on a computer with high processing power. In order to increase the success rate, the number of nodes of layers in CNN and MLP models can be set to different values and some layers can be removed from the architecture. Although many observations are made, tuning is done by manual and by using GridSearchCV and a lot of effort was spent on the architectures and parameters, it is possible to obtain better results for future studies by doing changes in the models.

In conclusion, among the models in this project, CNN was the model that gave the best results when compared with AUC scores.

BIBLIOGRAPHY

- [1] *Simple neural network*. [Online]. Available: <https://clevertap.com/blog/neural-networks/>.
- [2] “A novel multi-tissue rna diagnostic of healthy ageing relates to cognitive health status,” 2015.
- [3] “Prediction of alzheimer’s disease using blood gene expression data,” 2020.
- [4] . [Online]. Available: <https://www.oreilly.com/library/view/deep-learning/9781491924570/ch04.html>.
- [5] (), [Online]. Available: <https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE63060>.
- [6] (), [Online]. Available: <https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE63061>.
- [7] (), [Online]. Available: <https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE140829>.
- [8] “Convolutional neural network architecture.” (), [Online]. Available: <https://www.upgrad.com/blog/basic-cnn-architecture/>.
- [9] (), [Online]. Available: <https://stackoverflow.com/questions/42883547/intuitive-understanding-of-1d-2d-and-3d-convolutions-in-convolutional-neural-n>.
- [10] “Mlp architecture.” (), [Online]. Available: <https://towardsdatascience.com/multilayer-perceptron-explained-with-a-real-life-example-and-python-code-sentiment-analysis-cb408ee93141>.
- [11] (), [Online]. Available: https://www.researchgate.net/figure/Architecture-of-the-random-forest-model_fig1_301638643.
- [12] (), [Online]. Available: <https://www.edureka.co/blog/support-vector-machine-in-python/c>.