

CHAPTER – 1

INTRODUCTION

1.1. Chronic Kidney Disease:

Chronic diseases are illnesses that characteristically have a slow, progressive onset and a long duration. Chronic diseases impact every aspect of the individual's and family's life and usually result from repeated or prolonged exposure to an environment or substance that does not support the normal structure and functioning of the body. Chronic diseases are those illnesses that are part of a person's life, with little or no chance for full recovery. In acute disease, treatments focus on returning the individual to full health. With chronic disease, the medical focus is to limit the progression of the disease or to delay any secondary complication that might arise because of the disease. The body's normal structure and function work like a well-coordinated machine, with each part vital to the whole. The structure and function of the human body of a person with a chronic disease, on both the cellular and systemic levels, is permanently altered. It is due to this permanent, and often progressive, cellular change that the person with the chronic disease has an altered ability to function in activities of daily living. Centers for Disease Control and Prevention (CDC) statistics reveal that 1 out of 10 Americans have severe limitations in their daily activities because they have a chronic disease. **Estimated glomerular filtration rate (eGFR)**—a measure of how much blood your kidneys filter each minute. This tells you how well they work. If your eGFR is low, your kidneys are not working as well as they should. As kidney disease progresses, your eGFR goes down. A urine test to check for **albumin**. Albumin is a protein that can pass into the urine when the kidneys are damaged. Too much albumin in your urine is an early sign of kidney damage.

1.2. why kidney disease prediction as web application?

Now a days a quarter of a state population suffers from this disease. Many of the people not aware of this disease. Kidney disease leads to other health problems such as heart disease, stroke ...etc. Even though some people might know their health condition is not good, they are not ready to take a test. Because of the cost for testing and not having the patience to wait for a long time to know the result. That is why I designed this web application. By submitting some information about their body, I predict that do they affected by the disease or not using Deep Learning model.

1.3. Why hands on Cloud Service?

Cloud computing is one of the trends which is going in IT industry these days. The traditional way of building IT environment is now shifting towards the cloud computing. Using cloud service, we able to store large amount of data into virtual database for high availability and scalability. Now I am in India, suppose an end user is in USA. If my application is not in cloud, he may face network latency. To avoid this latency problem, we deploy our project in cloud service. They provide edge locations on all over the world. So, our application will be universally available.

1.4. Why AWS? Why not other services?

Due to the demand of cloud platforms many cloud service providers are rising day by day. IT environment is now a days depend on cloud platforms. There are many cloud providers, famous providers are Amazon Web Service (AWS), Microsoft Azure Cloud, Google Cloud, Oracle Cloud, IBM cloud and many. Among these AWS helping students to learn cloud services by offering free trier for a year.

CHAPTER - 2

PROBLEM STATEMENT

2.1. Project Goal:

- To build Chronic Kidney Disease Prediction web application
- To transform our application which is running in the local server into cloud server for high availability.

2.2. Project Objective:

To bring down the issues while running a web application in physical server. And to show that we can locate our data science prediction model into an amazon ec2 instance with the help of amazon cloud formation service.

CHAPTER – 3

SYSTEM ANALYSIS

3.1. Existing System:

The traditional way of predicting chronic kidney disease is to take many tests and with the result of the tests in the hospital lab they predict for chronic disease with many chemicals tests. So, patient should wait for a long period of time. Meanwhile the patient will be in stress, depression. If the patient is affected by CKD, then it is worth to wait. But the result is negative then it is vain of waiting for long time. And before system is built in physical server if any shutdown are any tragedies arises or something went wrong our website will not be accessible.

3.1.1. Disadvantages:

- Have to wait for a long time
- Work done manually
- Due to this, mistake may occur
- Result may map to wrong person
- Have to be in stress and anxiety
- Network latency
- Sudden shutdown of server
- Not edge optimized

3.2. Proposed System:

The proposed system trained with labeled data and predict CKD using many attributes like Sugar, Albumin, Blood Pressure, etc. By inputting the required query strings our model can able to predict whether one is affected or not. This system required thirteen attributes. Because we need to predict in high accuracy.

3.2.1. Advantages:

- Saves time
- Implemented in Amazon EC2 instance
- No network latency
- Don't need to worry about infrastructure and server maintenance.

- Edge optimized
- Any time available
- Don't need to worry about nature disasters
- Model built using artificial neural network with high accuracy
- Preprocessed labeled dataset used to train the model.

CHAPTER – 4

SYSTEM REQUIREMENTS

4.1. Software Requirements:

Tools: Jupyter Notebook

VSCode

Putty Keygen

Putty

WinSCP

Language: Python

Html

CSS

Cloud: Amazon web service

Services: EC2 instance

IAM

VPC

S3 Bucket

Lambda

Cloud Formation

Operating System: Any

CHAPTER – 5

SYSTEM SPECIFICATIONS

5.1. Python3:

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built-in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

5.1.1. Advantages:

- Presence of third-party modules
- Extensive support libraries (NumPy for numerical calculations, Pandas for data analytics etc.)
- Open source and community development
- Versatile, Easy to read, learn and write

- User-friendly data structures
- High-level language
- Dynamically typed language (No need to mention data type based on the value assigned, it takes data type)
- Object-oriented language
- Portable and Interactive
- Ideal for prototypes – provide more functionality with less coding
- Highly Efficient (Python's clean object-oriented design provides enhanced process control, and the language is equipped with excellent text processing and integration capabilities, as well as its own unit testing framework, which makes it more efficient.)
- (IoT)Internet of Things Opportunities
- Interpreted Language
- Portable across Operating systems

5.2. Deep learning:

Deep learning is a branch of machine learning (ML) that mimics the functioning of the human brain to find correlations and patterns by processing data with a specified logical structure. Also referred to as deep neural networks, deep learning uses multiple hidden layers in the neural network as opposed to traditional neural networks that only contain a handful of hidden layers. Deep learning algorithms map inputs to already *learned* data to deliver an accurate output. The concept underpinning this technology is very similar to how our brains function (biological neural networks). We compare new information with known information to arrive at a conclusion.

5.2.1. Advantages:

- Feature Generation Automation
- Works well with unstructured data
- Better self-learning capabilities
- Support parallel and distributed algorithms
- Advanced analytics
- Fast Computation

5.3. Amazon web service:

AWS needs no formal introduction, given its immense popularity. The leading cloud provider in the marketplace is Amazon Web Services. It provides over 170 AWS services to the developers so they can access them from anywhere at the time of need. AWS has customers in over 190 countries worldwide, including 5000 ed-tech institutions and 2000 government organizations. Many companies like ESPN, Adobe, Twitter, Netflix, Facebook, BBC, etc., use AWS services. Anyhow, we shall learn about AWS and its services used in this project.

Amazon web service is an online platform that provides scalable and cost-effective cloud computing solutions. AWS is a broadly adopted cloud platform that offers several on-demand operations like compute power, database storage, content delivery, etc., to help corporates scale and grow.

5.3.1. AWS History

- In the year 2002 - AWS services were launched
- In the year 2006- AWS cloud products were launched
- In the year 2012 - AWS had its first customer event
- In the year 2015- AWS achieved \$4.6 billion
- In the year 2016- Surpassed the \$10 billion revenue target
- In the year 2016- AWS snowball and AWS snowmobile were launched
- In the year 2019- Released approximately 100 cloud services

5.3.2. AWS Services

AWS has significantly more services, and more features within those services, than any other cloud provider-from infrastructure technologies like compute, storage and databases to emerging technologies, such as machine learning and artificial intelligence, data lakes and analytics and internet of things. This makes it faster, easier and more cost effective to move your existing applications to the cloud and build nearly anything you can imagine. AWS also has the deepest functionality within those services. For example, AWS offers the widest variety of databases that are purpose built for different

types of applications so you can choose the right tool for the job to get the best cost and performance. As well as AWS providing learners by free tier for a year.

CHAPTER – 6

SYSTEM IMPLEMENTATION

6.1. System Architecture

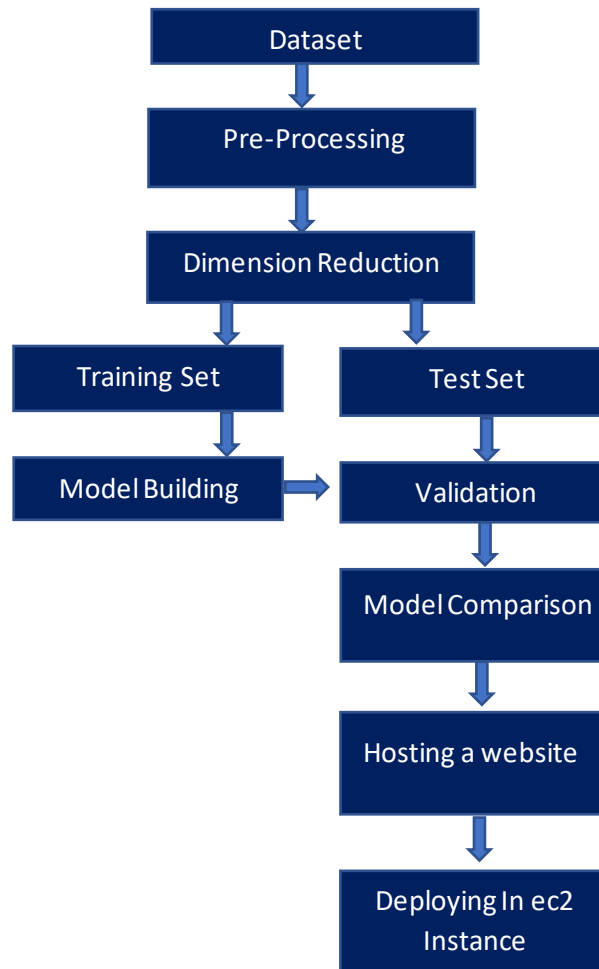


Fig 6.1. (Work flow)

6.2. Machine Learning Algorithms:

Machine Learning algorithms are the programs that can learn the hidden patterns from the data, predict the output, and improve the performance from experiences on their own. There are many different algorithms in machine learning. They are classified into two parts.

- Supervised algorithms

- Unsupervised algorithms
- Reinforcement algorithms

6.2.1. Supervised Algorithms:

Supervised learning is a type of Machine learning in which the machine needs external supervision to learn. The supervised learning models are trained using the labeled dataset. Once the training and processing are done, the model is tested by the test data to check whether it predicts the correct output. We can divide supervised algorithm further into two.

- Classification
- Regression

6.2.1.1. Classification Algorithms:

Classification is the process of recognizing, understanding, and grouping ideas and objects into preset categories or “sub-populations.” Using pre-categorized training datasets, machine learning programs use a variety of algorithms to classify future datasets into categories. We shall discuss some of the classification algorithms deeply which are used in this project.

- Logistic Regression
- Decision Tree
- Support Vector Machine
- K- Nearest Neighbors

Logistic regression:

Logistic regression uses sigmoid function or logistic function to map predicted values to binary values between 0 and 1, refer FIG 6.2. We cannot use linear regression for all the time. Linear regression model does not perform good for classification problems while there are more outliers present in the data. So, we use Logistic Regression. Linear Regression uses least square estimation to find the fit line. In logistic regression we use maximum likelihood. Here the best fit is not a linear line like linear regression. Here it is a ‘S’ curve. It smoothly maps the predicted values.

- Logistic regression assumes linearity of independent variables and log odds which is $\log(p/(1-p))$ where p is probability of success. Logistic regression requires

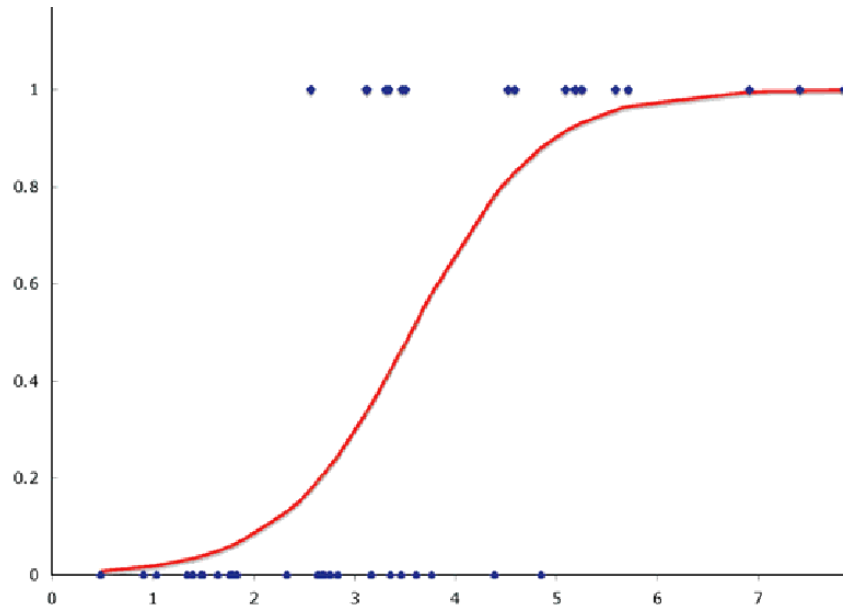


Fig 6.2. (Logistic regression)

there to be little or no multicollinearity among the independent variables. This means that the independent variables should not be too highly correlated with each other.

Math's behind Logistic regression:

We could start by assuming $p(x)$ be the linear function. However, the problem is that p is the probability that should vary from 0 to 1 whereas $p(x)$ is an unbounded linear equation. To address this problem, let us assume, $\log p(x)$ be a linear function of x and further, to bound it between a range of (0,1), we will use logit transformation. Therefore, we will consider $\log p(x)/(1-p(x))$. Next, we will make this function to be linear:

$$\log \frac{P(x)}{1 - P(x)} = \alpha_0 + \alpha \cdot x$$

After solving for $p(X)$:

$$p(x) = \frac{e^{\alpha_0 + \alpha x}}{e^{\alpha_0 + \alpha x} + 1}$$

To make the logistic regression a linear classifier, we could choose a certain threshold, e.g. 0.5. Now, the misclassification rate can be minimized if we predict $y=1$ when $p \geq 0.5$ and $y=0$ when $p < 0.5$. Here, 1 and 0 are the classes. Since Logistic regression predicts probabilities, we can fit it using likelihood. Therefore, for each training data point x , the

predicted class is y . Probability of y is either p if $y=1$ or $1-p$ if $y=0$. Now, the likelihood can be written as:

$$L(\alpha_0, \alpha) = \prod_{i=1}^n p(x_i)^{y_i} (1 - p(x_i))^{1-y_i}$$

The multiplication can be transformed into a sum by taking the log:

$$\begin{aligned} l(\alpha_0, \alpha) &= \sum_{i=0}^n y_i \log p(x_i) + (1 - y_i) \log 1 - p(x_i) \\ &= \sum_{i=0}^n \log 1 - p(x_i) + \sum_{i=0}^n y_i \log \frac{p(x_i)}{1 - p(x_i)} \end{aligned}$$

Further, after putting the value of $p(x)$:

$$l(\alpha_0, \alpha) = \sum_{i=0}^n -\log 1 + e^{\alpha_0 + \alpha} + \sum_{i=0}^n y_i (\alpha_0 + \alpha \cdot x_i)$$

The next step is to take a maximum of the above likelihood function because in the case of logistic regression gradient ascent is implemented (opposite of gradient descent). A method of estimating the parameters of probability distribution by maximizing a likelihood function, in order to increase the probability of occurring the observed data. We can find MLE by differentiating the above equation with respect to different parameters and setting it to be zero. For example, the derivative with respect to one of the components of parameter alpha i.e. α_j is given by:

$$\frac{\partial l}{\partial \alpha_j} = \sum_{i=0}^n (y_i - p(x_i; \alpha_0, \alpha)) x_{ij}$$

Decision Tree:

Decision Tree also a supervised algorithm family. The goal of using a Decision Tree is to create a training model that can use to predict the class or value of the

target variable by **learning simple decision rules** inferred from prior data (training data). In Decision Trees, for predicting a class label for a record we start from the **root** of the tree. We compare the values of the root attribute with the record's attribute. On the basis of comparison, we follow the branch corresponding to that value and jump to the next node (refer FIG 6.3). Decision tree is used for both classification and regression problems.

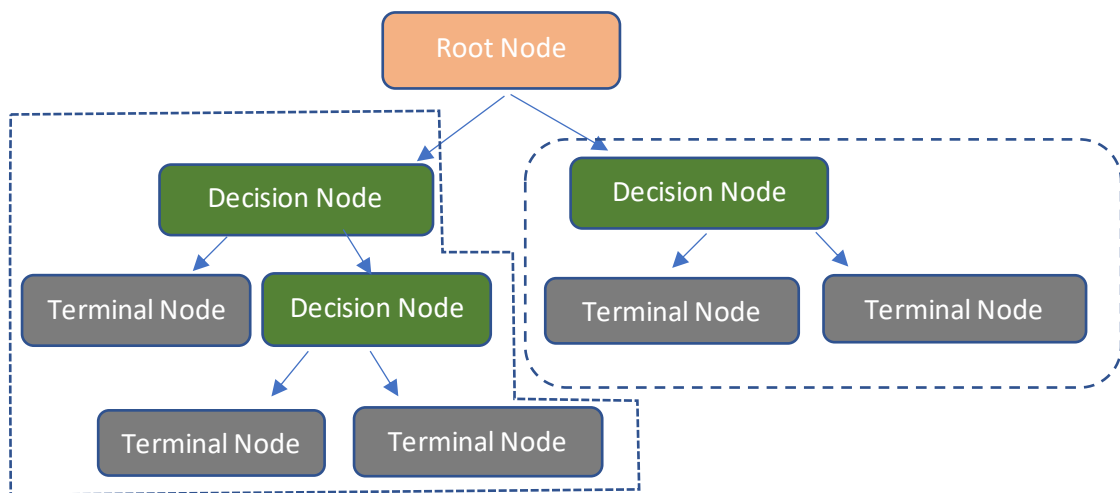


Fig 6.3. (Decision Tree Structure)

Decision trees classify the examples by sorting them down the tree from the root to some leaf/terminal node, with the leaf/terminal node providing the classification of the example. Each node in the tree acts as a test case for some attribute, and each edge descending from the node corresponds to the possible answers to the test case. This process is recursive in nature and is repeated for every subtree rooted at the new node.

Assumptions while creating Decision Tree

Below are some of the assumptions we make while using Decision tree:

- In the beginning, the whole training set is considered as the **root**.
- Feature values are preferred to be categorical. If the values are continuous then they are discretized prior to building the model.
- Records are **distributed recursively** on the basis of attribute values.
- Order to placing attributes as root or internal node of the tree is done by using some statistical approach.

Working Process:

- It begins with the original set S as the root node.

- On each iteration of the algorithm, it iterates through the very unused attribute of the set S and calculates **Entropy (H)** and **Information gain (IG)** of this attribute.
- It then selects the attribute which has the smallest Entropy or Largest Information gain.
- The set S is then split by the selected attribute to produce a subset of the data.
- The algorithm continues to recur on each subset, considering only attributes never selected before.

Attribute Selection Measures

If the dataset consists of **N** attributes, then deciding which attribute to place at the root or at different levels of the tree as internal nodes is a complicated step. By just randomly selecting any node to be the root can't solve the issue. If we follow a random approach, it may give us bad results with low accuracy.

For solving this attribute selection problem, researchers worked and devised some solutions. They suggested using some *criteria* like:

- a) Entropy,
- b) Information gain,
- c) Gini index,
- d) Chi-square

a) Entropy

Entropy is a measure of the randomness in the information being processed. The higher the entropy, the harder it is to draw any conclusions from that information. Flipping a coin is an example of an action that provides information that is random. Entropy $H(X)$ is zero when the probability is either 0 or 1. The Entropy is maximum when the probability is 0.5 because it projects perfect randomness in the data and there is no chance if perfectly determining the outcome.

$$E(S) = \sum_{i=1}^c -p_i \log_2 p_i$$

b) Information Gain:

Information gain or **IG** is a statistical property that measures how well a given attribute separates the training examples according to their target classification. Constructing a decision tree is all about finding an attribute that returns the highest information gain and the smallest entropy.

Information gain is a decrease in entropy. It computes the difference between entropy before split and average entropy after split of the dataset based on given attribute values.

Mathematically, IG is represented as:

$$\text{Information Gain}(T,X) = \text{Entropy}(T) - \text{Entropy}(T, X)$$

T – entropy before split, X – entropy after split

c) Gini Index

You can understand the Gini index as a cost function used to evaluate splits in the dataset. It is calculated by subtracting the sum of the squared probabilities of each class from one. It favors larger partitions and easy to implement whereas information gain favors smaller partitions with distinct values.

$$Gini = 1 - \sum_{i=1}^C (p_i)^2$$

Gini Index works with the categorical target variable “Success” or “Failure”. It performs only Binary splits. Higher value of Gini index implies higher inequality, higher heterogeneity (diverse in character or content).

d) Chi-Square

The acronym CHAID stands for *Chi*-squared Automatic Interaction Detector. It is one of the oldest tree classification methods. It finds out the statistical significance between the differences between sub-nodes and parent node. We measure it by the sum of squares of standardized differences between observed and expected frequencies of the target variable. It works with the categorical target variable “Success” or “Failure”. It can perform two or more splits. Higher the value of Chi-Square higher the statistical

significance of differences between sub-node and Parent node. It generates a tree called CHAID (Chi-square Automatic Interaction Detector).

Mathematically, Chi-squared is represented as:

$$\chi^2 = \sum \frac{(O - E)^2}{E}$$

χ^2 – Chi Squared Obtained, Σ – The sum of, O – Observed Score, E – Expected Score

Support Vector Machine:

“Support Vector Machine” (SVM) is a supervised machine learning algorithm that 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 a 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.

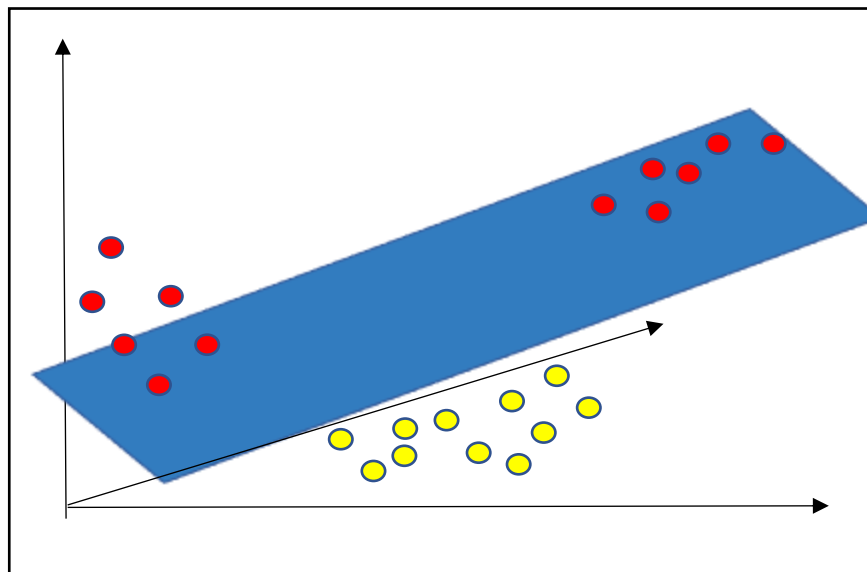


Fig 6.4. (Support Vector Machine)

Here we fit a plane in 3 dimensions. Support vector machine is very useful in fit a plane in 3d space.

K - Nearest Neighbors

K Nearest Neighbor algorithm falls under the Supervised Learning category and is used for classification (most commonly) and regression. It is a versatile algorithm

also used for imputing missing values and resampling datasets. As the name (K Nearest Neighbor) suggests it considers K Nearest Neighbors (Data points) to predict the class or continuous value for the new Datapoint.

The algorithm's learning is:

- Instance-based learning: Here we do not learn weights from training data to predict output (as in model-based algorithms) but use entire training instances to predict output for unseen data.
- Lazy Learning: Model is not learned using training data prior and the learning process is postponed to a time when prediction is requested on the new instance.
- Non -Parametric: In KNN, there is no predefined form of the mapping function.

KNN works on a principle assuming every data point falling in near to each other is falling in the same class. In other words, it classifies a new data point based on similarity. Let us understand the concept by taking our Chronic Kidney Disease Data set as an example. Take a look at FIG 6.5.. In our data set there are two classes “affected by CKD” and “not affected by CKD”. Consider ● as CKD affected and ● as not affected.

Here for given K value the model chooses k nearest value for that new point. If most of the k points are red then the new point classified to red group or most

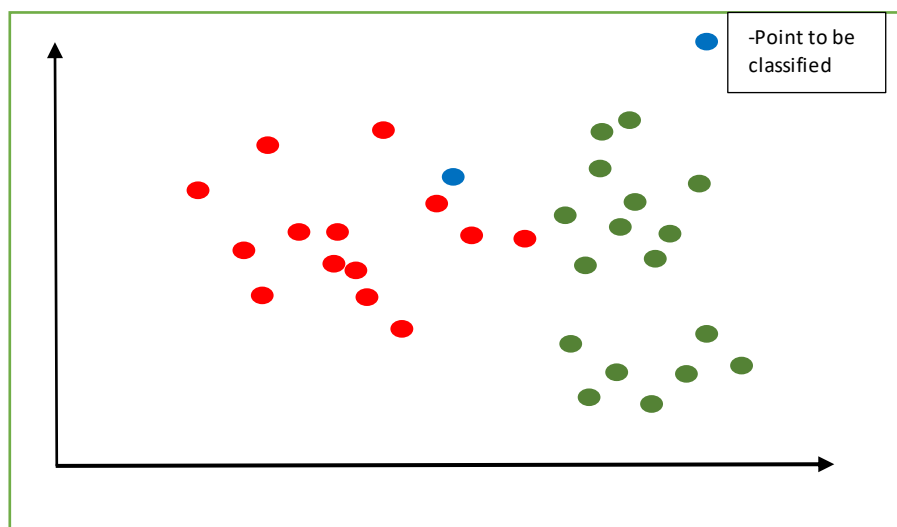


Fig 6.5. (KNN Classifier)

of the points are blue group then the new point classified to blue group. Here KNN calculates distance between the new point and all the points in the dataset. The main thing is to choose the k value. Most of the times we chose k as random value. In this project also we didn't used any mechanism to find the k value. But this is not encouraged. Because if the k value is small noise over the data will increase. Data scientists use cross validation to choose the k- value.

6.2.1.2. Regression Algorithms:

We use regression when our dependent variable is continuous rather than categorical. Often, we use regression algorithms to predict classification dataset. But if there are outliers present in our dataset it will not work perfectly for classification data .

Here are some Regression algorithms.

- Linear Regression
- Ridge Regression
- KNN
- Decision Tree
- Random Forest
- Support Vector Machine
- Polynomial Regression
- Lasso Regression

We are not going to discuss these algorithms. Because in this project we didn't used those algorithms. And we already discussed about KNN, Decision Tree, Support Vector Machine in section 6.2.1.1. We know that the above mentioned three algorithms are used to do both regression and classification predictions.

6.2.2. Unsupervised algorithms:

Unsupervised learning is opposite to supervised learning. It does not need any external supervision to learn. The unsupervised learning models are trained using the unlabeled dataset. It needs neither categorized nor classified data set. Unsupervised can also be divide into two.

- Clustering
- Association

6.2.3 Reinforcement learning:

Reinforcement Learning is a feedback-based Machine learning technique in which an agent learns to behave in an environment by performing the actions and seeing the results of actions. For each good action, the agent gets positive feedback, and for each bad action, the agent gets negative feedback or penalty.

6.3. Neural Networks

Neural networks reflect the behavior of the human brain, allowing computer programs to recognize patterns and solve common problems in the fields of AI, machine learning, and deep learning. Neural networks, also known as artificial neural networks (ANNs) or simulated neural networks (SNNs), are a subset of machine learning and are at the heart of deep learning algorithms. Their name and structure are inspired by the human brain, mimicking the way that biological neurons signal to one another. Artificial neural networks (ANNs) are comprised of a node layer, containing an input layer, one or more hidden layers, and an output layer. Each node, or artificial neuron, connects to another and has an associated weight and threshold. If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network.

6.3.1. Architecture of neural networks:

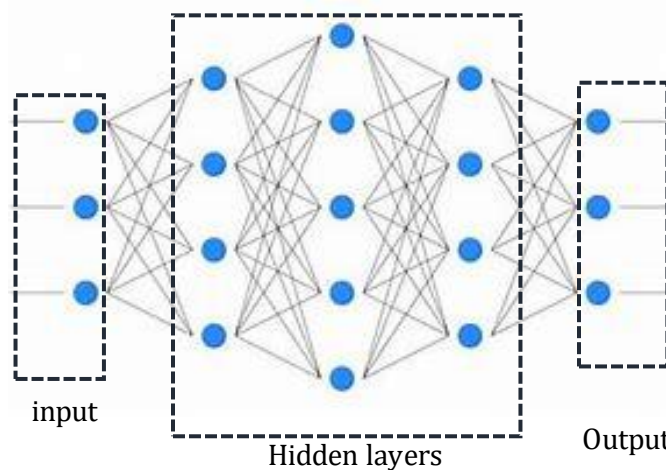


Fig 6.6. (Neural network architecture)

6.3.2. Working Process

The neural network is a weighted graph where nodes are the neurons, and edges with weights represent the connections. It takes input from the outside world. The inputs are multiplied with weights and we find the input of next layer. And the next layer input is multiplied with corresponding weights. This process is called **activation**. Each input is multiplied by its respective weights, and then they are added. A bias is added if the weighted sum equates to zero, where bias has input as 1 with weight b . Then this weighted sum is passed to the activation function. The activation function limits the amplitude of the output of the neuron. There are various activation functions

To optimize the output, we use optimizers until our output is optimized to the threshold. This process is called **Back Propagation**.

6.3.3. Activation Function:

Activation function decides, whether a neuron should be activated or not by calculating weighted sum and further adding bias with it. The purpose of the activation function is to introduce non-linearity into the output of a neuron. There are various activation functions.

- Rectified Linear Unit (ReLU)
- Leaky ReLU
- Sigmoid
- Hyperbolic Tangent (Tanh)
- SoftMax

Rectified linear Unit (ReLU)

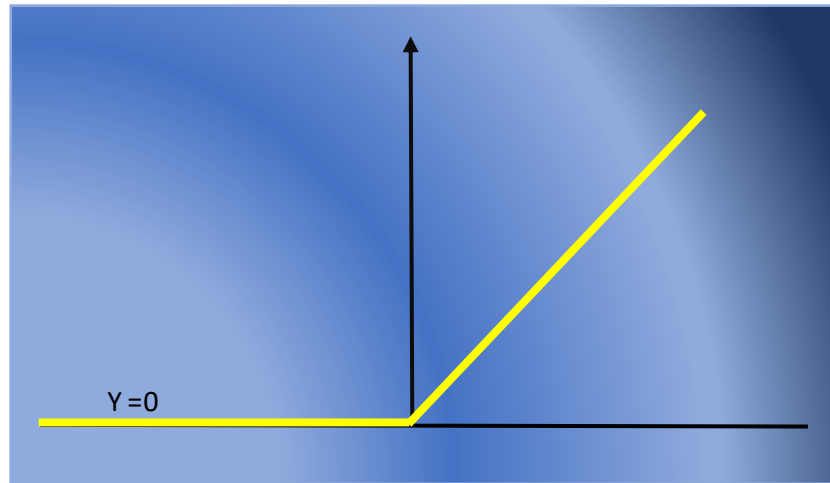


Fig 6.7. (Relu graph)

The rectified linear activation function or ReLU is a **non-linear** function or **piecewise linear** function that will output the input directly if it is positive, otherwise, it will output zero. While seeing FIG 6.6 we can clearly understand it.

$$f(x) = \max(x, 0)$$

Leaky ReLU

We needed the **Leaky ReLU activation** function to solve the '**Dying ReLU**' problem, we can observe that all the negative input values turn into zero very quickly and

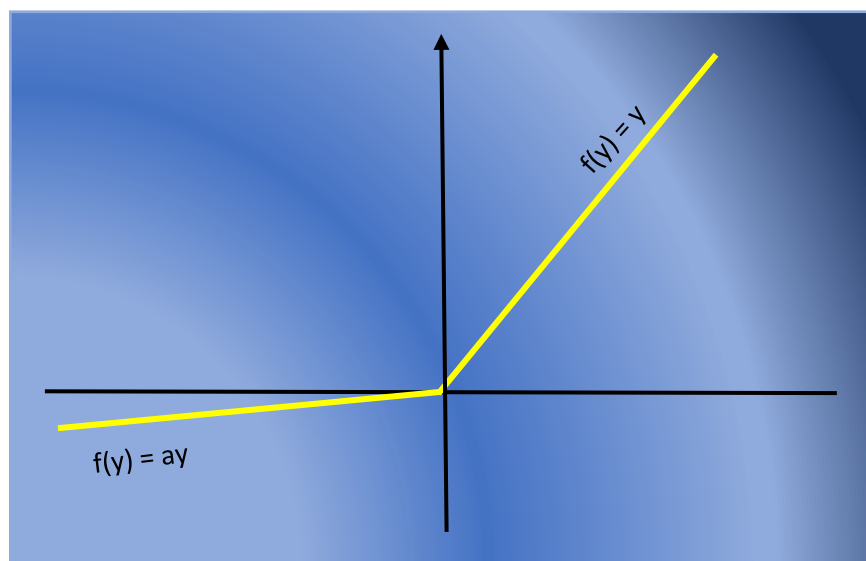


Fig 6.8. (Leaky Relu graph)

in the case of Leaky ReLU we do not make all negative inputs to zero but to a value near to zero which solves the major issue of ReLU activation function. **Leaky Rectified Linear Unit**, or **Leaky ReLU**, is a type of activation function based on a ReLU, but it has a small

slope for negative values instead of a flat slope. The slope coefficient is determined before training, i.e., it is not learnt during training. This type of activation function is popular in tasks where we may suffer from sparse gradients.

Sigmoid:

The sigmoid activation function is used mostly as it does its task with great efficiency, it basically is a probabilistic approach towards decision making and ranges in between **0 to 1**, so when we have to make a decision or to predict an output we use this activation function because of the range is the minimum, therefore, prediction would be more accurate.

$$f(x) = \frac{1}{1 + e^{-x}}$$

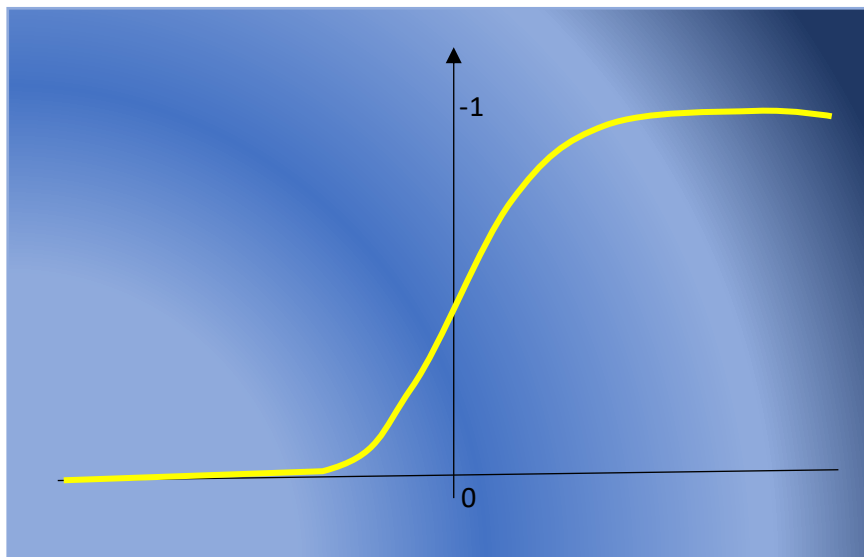


Fig 6.9. (Sigmoid graph)

Tanh

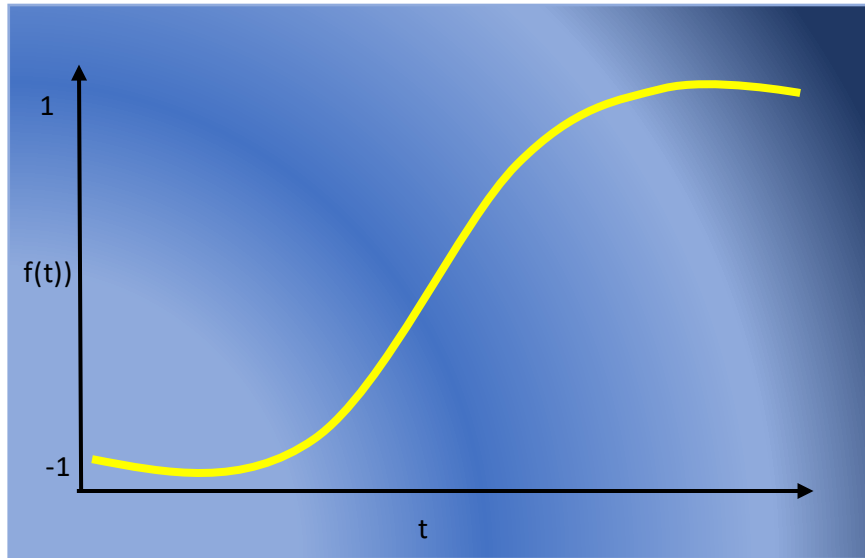


Fig 6.10. (tanh graph)

This activation function is slightly better than the sigmoid function, like the **sigmoid function** it is also used to predict or to differentiate between two classes but it maps the negative input into negative quantity only and ranges in between **-1 to 1**.

SoftMax

Softmax is used mainly at the last layer i.e., output layer for decision making the same as sigmoid activation works, the softmax basically gives value to the input variable according to their weight and the sum of these weights is eventually one. For **Binary classification**, both **sigmoid**, as well as **softmax**, are equally approachable but in case of multi-class classification problem we generally use softmax and cross-entropy along with it. The input values can be positive, negative, zero, or greater than one, but the softmax transforms them into values between 0 and 1, so that they can be interpreted as probabilities. If one of the inputs is small or negative, the softmax turns it into a small probability, and if an input is large, then it turns it into a large probability, but it will always remain between 0 and 1.

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}}$$

where all the z_i values are the elements of the input vector and can take any real value. The term on the bottom of the formula is the normalization term which ensures that all the output values of the function will sum to 1, thus constituting a valid probability distribution.

6.3.4 Optimizers

Optimizers are algorithms or methods used to minimize an error function (*loss function*) or to maximize the efficiency of production. Optimizers are mathematical functions which are dependent on model's learnable parameters i.e., Weights & Biases. Optimizers help to know how to change weights and learning rate of neural network to reduce the losses.

Types of optimizers,

- Gradient Descent
- Stochastic Gradient Descent
- Mini Batch Gradient Descent
- Adaptive Gradient Descent (AdaGrad)
- Root Mean Square Propagation (RMS-Prop)
- Adaptive Delta (AdaDelta)
- Adaptive Moment Estimation (Adam)

Let's see about the optimizer which are used in this project,

Adaptive Moment Estimation (Adam)

Adam optimizer is one of the most popular and famous gradient descent optimization algorithms. It is a method that computes adaptive learning rates for each parameter. It stores both the decaying average of the past gradients, similar to momentum and also the decaying average of the past squared gradients, similar to RMS-Prop and AdaDelta.

$$w_t = w_{t-1} - \frac{\eta}{\sqrt{S_{dw_t} - \epsilon}} * V_{dw_t}$$

W_t – new weight, W_{t-1} – Old Weight, η – learning rate

6.3.5. Loss Functions:

The loss function in a neural network quantifies the difference between the expected outcome and the outcome produced by the machine learning model. From the loss function, we can derive the gradients which are used to update the weights. The average over all losses constitutes the cost.

Types of loss functions,

- Mean Absolute Error
- Mean Squared Error
- Huber Loss
- Cross-Entropy
- Relative Entropy
- Squared Hinge

Loss function used in this project,

Mean Square Error:

Mean Squared Error (MSE) also called **L2 Loss** is also a loss function used for regression. It represents the difference between the original and predicted values extracted by squared the average difference over the data set.

$$MSE = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y})^2$$

MSE is *sensitive towards outliers* and given several examples with the same input feature values, the optimal prediction will be their mean target value. This should be compared with Mean Absolute Error, where the optimal prediction is the median. MSE is thus good to use if you believe that your target data, conditioned on the input, is normally distributed around a mean value, and when it's important to penalize outliers extra much.

6.4. Python Packages:

Various packages are used in this project let us discuss all of the packages.

- Sklearn
- Matplotlib
- Seaborn
- TensorFlow
- Pandas
- Numpy
- pickle
- Flask

Flask:

Flask is the package which is used to turn our model into web application. Flask is a web framework that provides libraries to build lightweight web applications in python. It is developed by **Armin Ronacher** who leads an international group of python enthusiasts (POCCO). It is based on WSGI toolkit and jinja2 template engine. Flask is considered as a micro framework.

6.5. AMAZON WEB SERVICE

List of AWS services which are used in this project:

- VPC
- EC2
- Lambda
- Cloud Formation
- S3 Bucket
- IAM

6.5.1. VPC

Virtual Private Cloud (VPC) is a private network in the public cloud. A VPC is a public cloud offering that lets an enterprise establish its own private cloud-like computing environment on

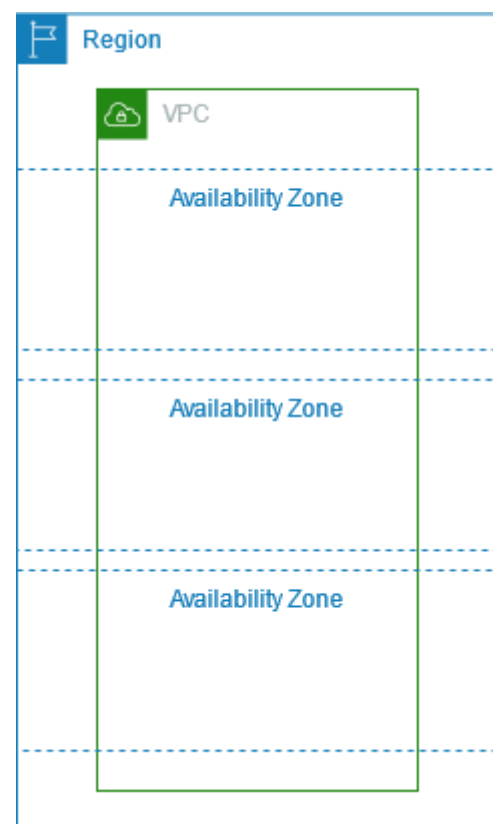


Fig 6.11. (VPC overview)

shared public cloud infrastructure. A VPC gives an enterprise the ability to define and control a virtual network that is logically isolated from all other public cloud tenants, creating a private, secure place on the public cloud.

Imagine that a cloud provider's infrastructure is a residential apartment building with multiple families living inside. Being a public cloud tenant is akin to sharing an apartment with a few roommates. In contrast, having a VPC is like having your own private condominium—no one else has the key, and no one can enter the space without your permission.

Amazon Virtual Private Cloud (Amazon VPC) enables you to launch AWS resources into a virtual network that you've defined. This virtual network closely resembles a traditional network that you'd operate in your own data center, with the benefits of using the scalable infrastructure of AWS. A *virtual private cloud* (VPC) is a virtual network dedicated to your AWS account. It is logically isolated from other virtual networks in the AWS Cloud. You can launch your AWS resources, such as Amazon EC2 instances, into your VPC. You can specify an IP address range for the VPC, add subnets, associate security groups, and configure route tables.

Creation of VPC

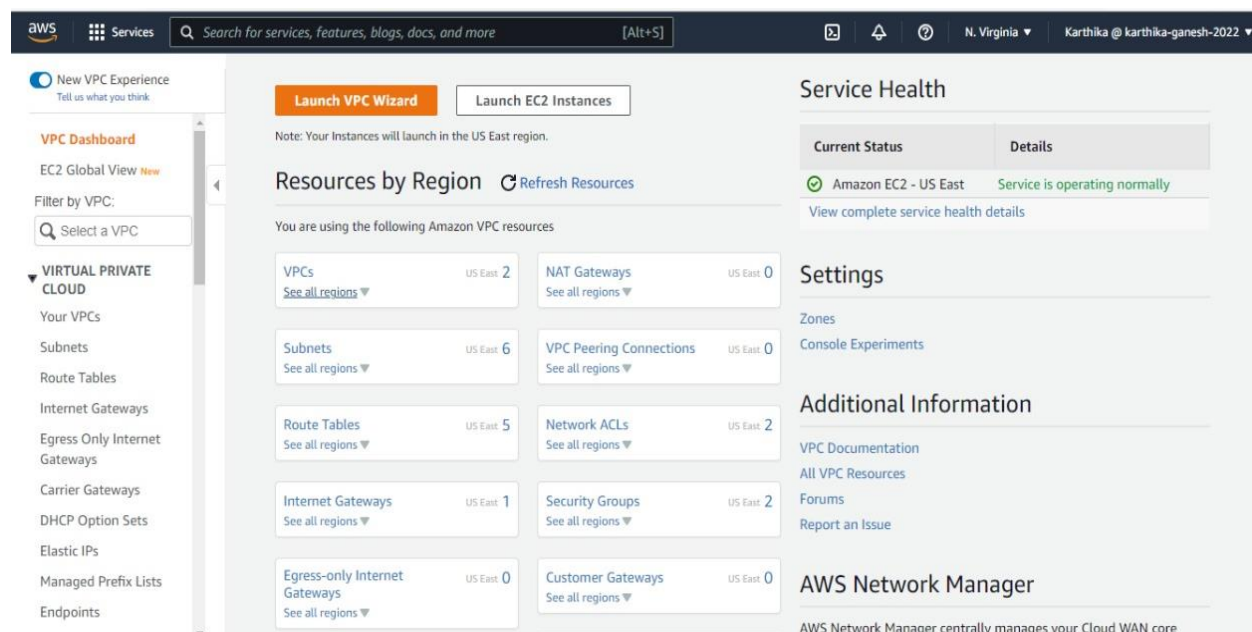


Fig 6.12. (VPC dashboard)

Take a look at FIG 6.13 This is how VPC dashboard looks. Here we can see what are all the resources are currently active and how many resources and in how many regions we created our resource. This kind of information we can able to watch here. And we can

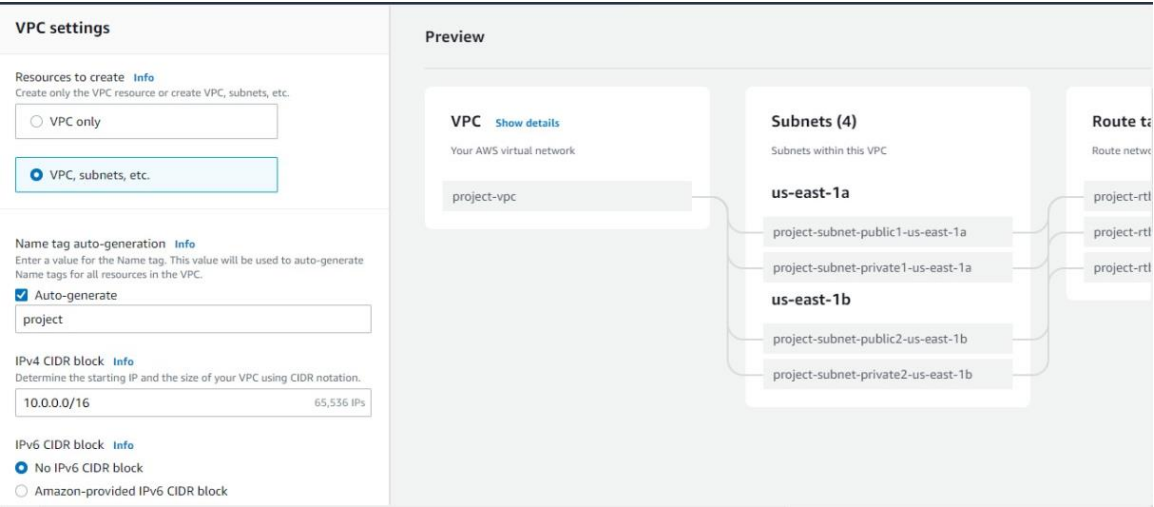


Fig 6.13. (VPC creation page)

easily navigate to any resources inside VPC. In the left side we can see a portion holding all the elements inside virtual private cloud. FIG 6.13 showing the VPC creation window. IPV4 CIDR block is very important. This is the one decides our region IP address. We van create a subnet while creating VPC or else we can create it late. The one benefit while creating subnet while creating VPC is we don't need to worry for calculating the IP address range. But if we create it later, we have to calculate the correct IP range. And the right side showing the pictorial of what we are creating in the left-hand side.

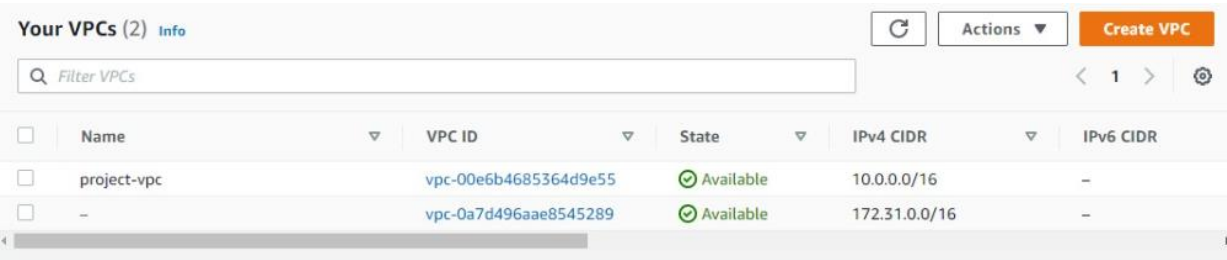


Fig 6.14. (VPC window)

After we enter the create VPC button our VPC created like FIG 6.14.

6.5.1.1. Configure VPC

This tutorial uses one of the VPC creation wizards to create the following:

- The VPC
- One of the subnets
- An Internet gateway

To create your VPC using the VPC wizard

1. On the **VPC Dashboard**, choose **Launch VPC Wizard**.
2. Under **Step 1: Select a VPC Configuration**, on **VPC with a Single Public Subnet**, choose **Select**.
3. Enter the following information into the wizard and choose **Create VPC**.

IP CIDR block

-10.0.0.0/16

VPC name

-ADS VPC

Public subnet

-10.0.0.0/24

Availability Zone

No Preference

Subnet name

-ADS Subnet 1

Enable DNS hostnames

-Leave default selection

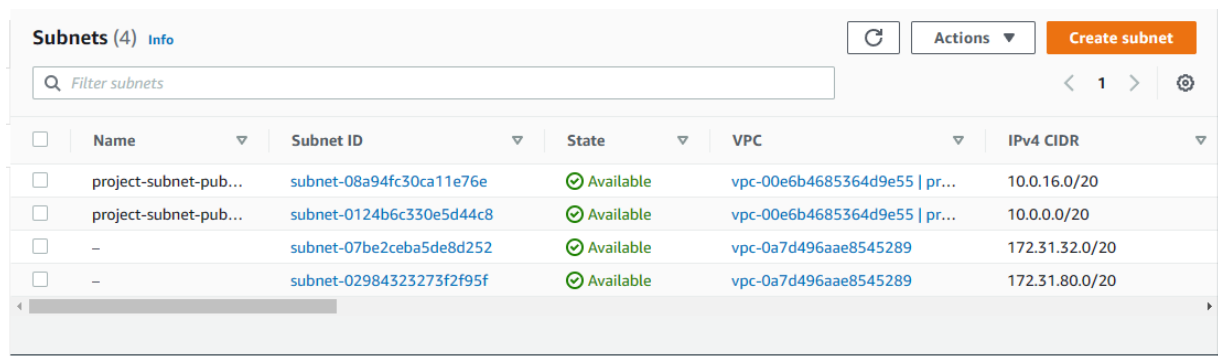
Hardware tenancy

Default

4. It takes several minutes for the VPC to be created. After the VPC is created, proceed to the following section to add a second subnet.

6.5.2. Subnet

A *subnet* is a range of IP addresses in your VPC. You can launch AWS resources into a specified subnet. Use a public subnet for resources that must be connected to the internet, and a private subnet for resources that won't be connected to the internet. Inside a subnet we create availability zones. You can optionally add subnets in a Local



The screenshot shows the AWS Subnets console with a table of subnets. The table has columns for Name, Subnet ID, State, VPC, and IPv4 CIDR. There are four subnets listed, all in an 'Available' state. The first two are public subnets, and the last two are private subnets.

Name	Subnet ID	State	VPC	IPv4 CIDR
project-subnet-pub...	subnet-08a94fc30ca11e76e	Available	vpc-00e6b4685364d9e55 pr...	10.0.16.0/20
project-subnet-pub...	subnet-0124b6c330e5d44c8	Available	vpc-00e6b4685364d9e55 pr...	10.0.0.0/20
-	subnet-07be2ceba5de8d252	Available	vpc-0a7d496aae8545289	172.31.32.0/20
-	subnet-02984323273f2f95f	Available	vpc-0a7d496aae8545289	172.31.80.0/20

Fig 6.15. (Subnet window)

Zone, which is an AWS infrastructure deployment that places compute, storage, database, and other select services closer to your end users. A Local Zone enables your end users to run applications that require single-digit millisecond latencies.

6.5.2.1. Subnet Types

- **Public subnet:** The subnet's IPv4 or IPv6 traffic is routed to an internet gateway or an egress-only internet gateway and can reach the public internet

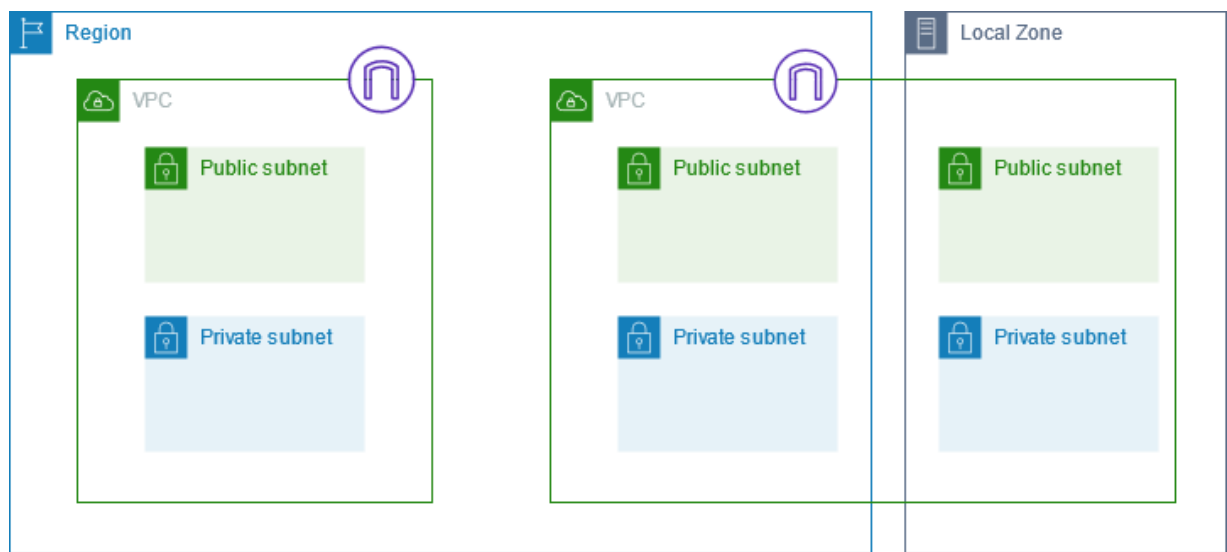


Fig 6.16. (Subnet types)

- **Private subnet:** The subnet's IPv4 or IPv6 traffic is not routed to an internet gateway or egress-only internet gateway and cannot reach the public internet.
- **VPN-only subnet:** The subnet doesn't have a route to the internet gateway, but it has its traffic routed to a virtual private gateway for a Site-to-Site VPN connection.

Regardless of the type of subnet, the internal IPv4 address range of the subnet is always private—we do not announce the address block to the internet.

6.5.3. Route Tables

A *route table* contains a set of rules, called routes, that are used to determine where network traffic from your VPC is directed. You can explicitly associate a subnet

Name	Route table ID	Explicit subnet associations	Edge associations	Main	VPC
-	rtb-03df1b033661273db	-	-	Yes	vpc-0a7d496aae8545...
project-rtb-private...	rtb-0b20fc8b073403849	-	-	No	vpc-00e6b4685364d9
-	rtb-06ac8bb8ca460931d	-	-	Yes	vpc-00e6b4685364d9
project-rtb-private...	rtb-05723a102ecdfe3df	-	-	No	vpc-00e6b4685364d9
project-rtb-public	rtb-0d746ac49225f2ff9	2 subnets	-	No	vpc-00e6b4685364d9

Fig 6.17. (route tables)

with a particular route table. Otherwise, the subnet is implicitly associated with the main

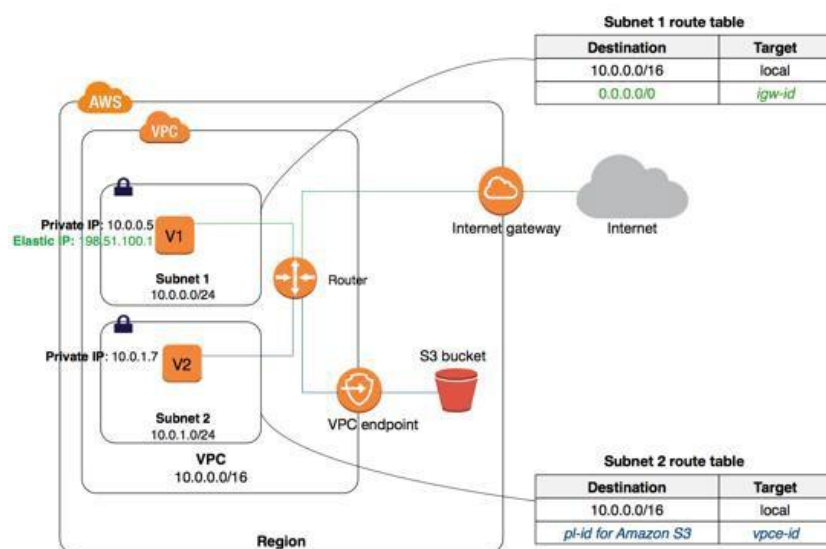


Fig 6.18. (VPC, subnet, Endpoint, Gateways)

route table. Each route in a route table specifies the range of IP addresses where you want the traffic to go (the destination) and the gateway, network interface, or connection through which to send the traffic (the target).

6.5.4. Gate Way

This the one gives permission for our EC2 instance to connect with public network. There are two types of Gateways in VPC.

- Internet Gateway - An internet gateway is a horizontally scaled, redundant, and highly available VPC component that allows communication between your VPC and the internet. An internet gateway enables resources (like EC2 instances) in your public subnets to connect to the internet if the resource has a public IPv4 address or an IPv6 address.
- NAT Gateway - NAT Gateway, also known as Network Address

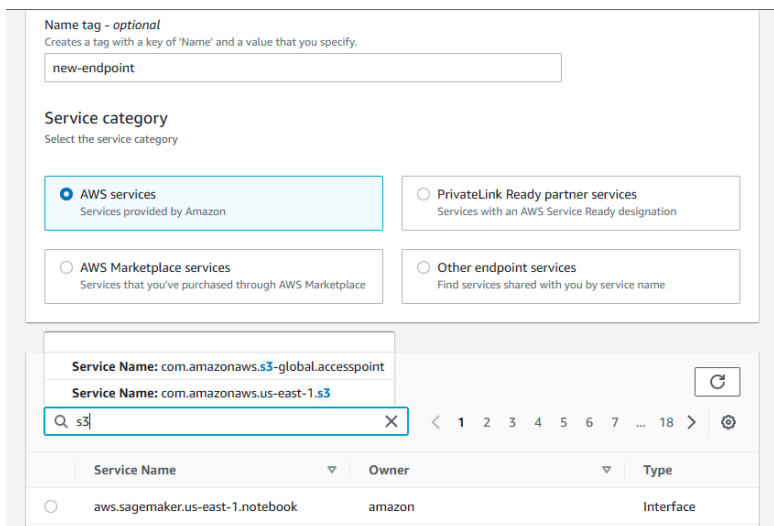


Fig 6.19. (Endpoint)

- Translation Gateway, is used to enable instances present in a private subnet to help connect to the internet or AWS services.

6.5.5. Endpoints

VPC endpoint enables creation of a private connection between VPC to supported AWS services and VPC endpoint services powered by Private Link using its private IP address. Traffic between VPC and AWS service does not leave the Amazon network.

6.5.6. EC2 Instance

Amazon EC2 instance is a virtual server. Amazon Elastic Compute Cloud (Amazon EC2) provides scalable computing capacity in the Amazon Web Services (AWS) Cloud. Using Amazon EC2 eliminates your need to invest in hardware up front, so you can develop and deploy applications faster. You can use Amazon EC2 to launch as many or as few virtual servers as you need, configure security and networking, and manage storage. Amazon EC2 enables you to scale up or down to handle changes in requirements or spikes in popularity, reducing your need to forecast traffic.

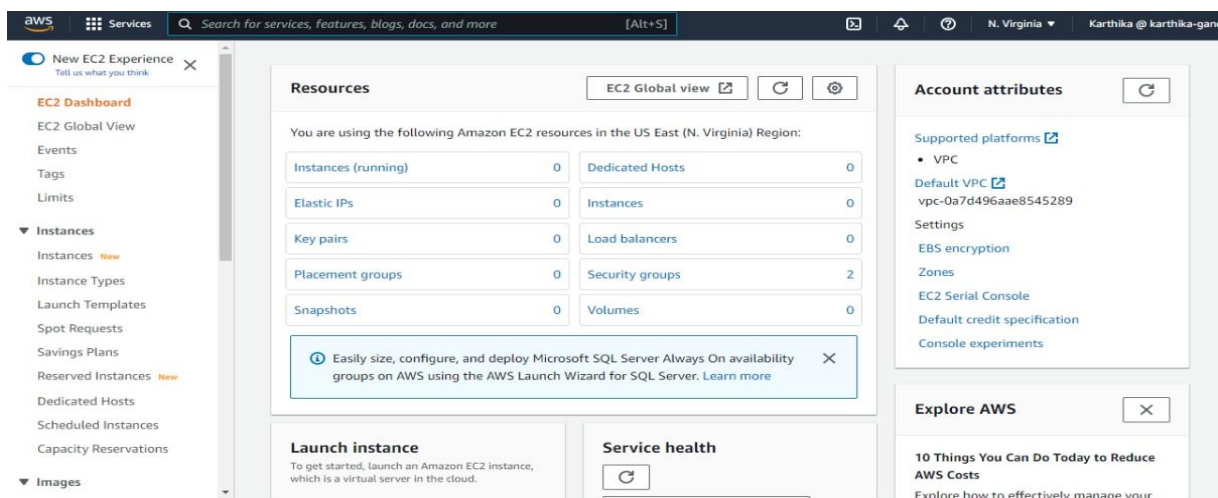


Fig 6.20. (EC2 Dashboard)

This is how ec2 instance dashboard looks. Here we can monitor which are all resources we created inside our instance. If ec2 is public then we add elastic IP address. If an ec2 instance is inside public subnet then our ec2 instance is also public. Public is not like any one access our instance. No one access our resources unless we give permission

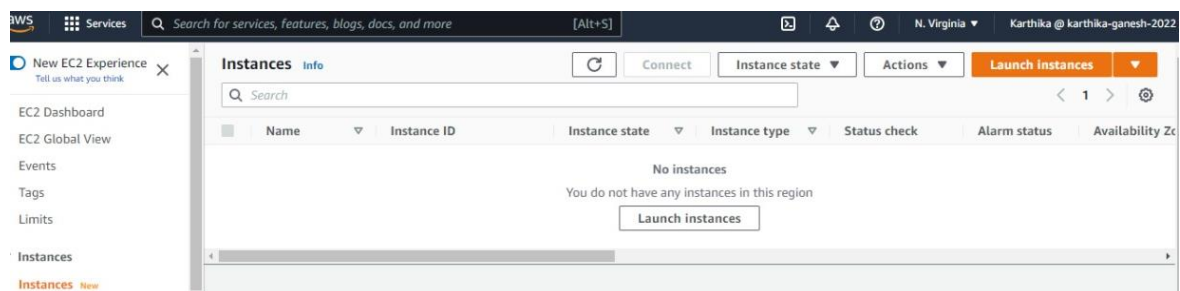


Fig 6.21. (No instances)

in the security group. Dedicated hosts are nothing but we can ask AWS to allocate server

only for us without sharing the server with anyone. It will cost high.

See Fig 6.21. currently there is no instance. By clicking the create instance we can able to create our instance. If we want to create our instance, we need to first create an AMI (amazon machine image). For details about AMI refer section 6.5.9. The AMI is shown in the figure

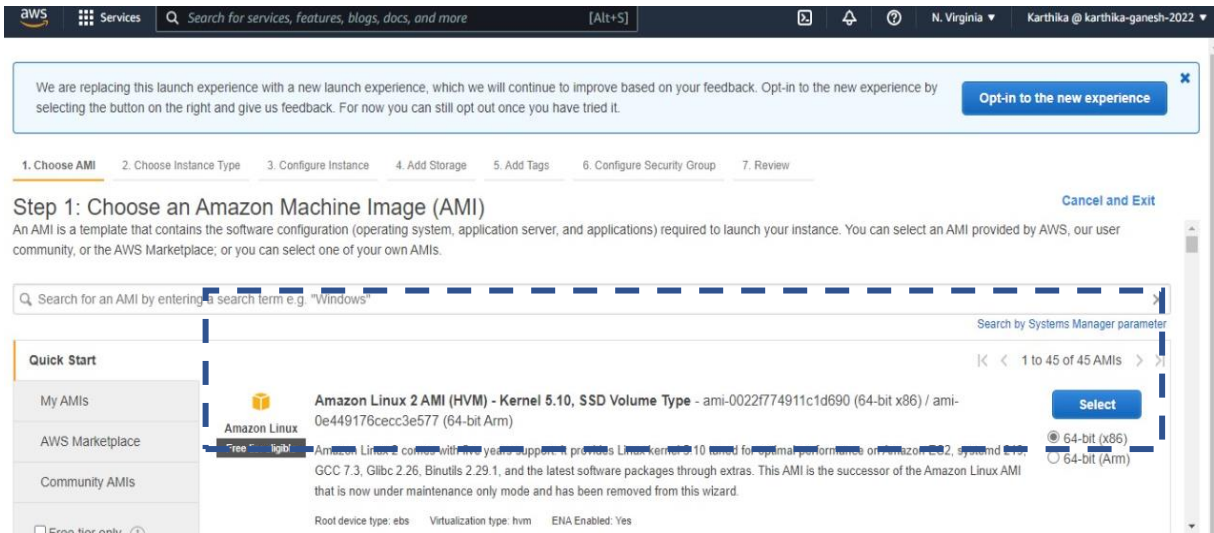


Fig 6.22. (AMI)

This console lists various Operating system. Now we going to use only Linux. The next step is to choose instance type. Now there is no choice for use. Because now we are working on

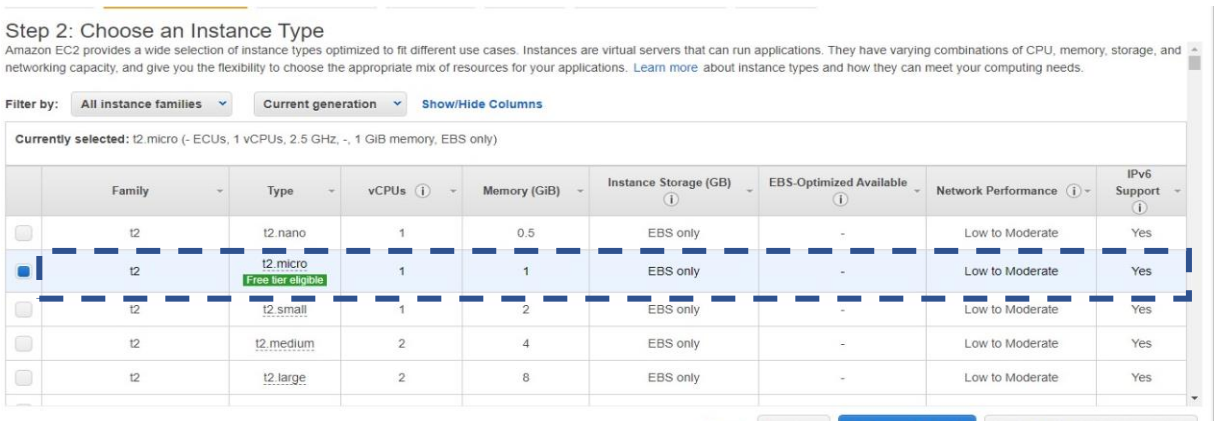


Fig 6.23. Instance type

free tier. So, we need to choose t2. micro instance. More details about instance type take a look on section 6.5.6.2. The instance type page has shown in fig 6.23

The next step is to configure the instance. The configure page looks like the one in the fig 6.24

Fig 6.24. Instance configuration

You can see we are in the third step of creating ec2 instance in the fig.. In user data we can add some html code or something to see whether our instance is working fine or not by checking it. And in the *auto assign public IP* the default aoption is disable. Just change it to yes. Because we want public address to access our instance from outside the VPC. For that we need that IP. Don't worry our instance is private. For a private instance we can have both private and public IP address.

The next step is to setup the storage. By default we allocated with 8 gb storage. If we want to increase our storage we add in the box shown in fig 6.25

Fig 6.25. Instance storage

The next step is to create tags. The page will looke like fig 6.26

Tags are Key Value pair. They are used to identify our instance easily. It is not mandatory, we can skip it.

Step 5: Add Tags
A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver.
A copy of a tag can be applied to volumes, instances or both.
Tags will be applied to all instances and volumes. [Learn more](#) about tagging your Amazon EC2 resources.

Key (128 characters maximum)

Value (256 characters maximum)

Instances ⓘ

Volumes ⓘ

Network Interfaces ⓘ

job

my-kidney-disease-prediction-project

☑

☑

☑

✕

Add another tag

(Up to 50 tags maximum)

Fig 6.26. Instance tags

The next step is to create security groups. Here we can able to add some rules. They are called inbound and outbound rules. The window looks like fig 6.27.

Step 6: Configure Security Group
A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more](#) about Amazon EC2 security groups.

Assign a security group: ☒ Create a new security group
☐ Select an existing security group

Security group name:

Description:

Type ⓘ	Protocol ⓘ	Port Range ⓘ	Source ⓘ	Description ⓘ
SSH ▾	TCP	22	Custom ▾ 0.0.0.0/0	e.g. SSH for Admin Desktop ✕

Fig 6.27. Security groups

Next review window will open. We can check all the configurations. You can see it is 7th step .

After click the create instance submit button key pair window gets open. This is very

Step 7: Review Instance Launch

▼ Instance Details

Edit instance details

Number of instances 1

Purchasing option On demand

Network vpc-00e6b4685364d9e55

Subnet subnet-072c53e2a95fca390

EBS-optimized No

Monitoring No

Termination protection No

Shutdown behavior Stop

Stop - Hibernate behavior Disabled

Capacity Reservation open

IAM role None

Domain join directory None

Tenancy default

Credit specification Standard

Host ID

Host resource group name

Affinity Off

Kernel ID Use default

RAM disk ID Use default

Enclave false

Metadata accessible Enabled

Fig 6.28. Instance review

important. A key pair consists of public that AWS stores, key and private key that we store. Together, they allow us to connect to our instance securely. For windows AMIs, the private

Fig 6.29. Key pair

key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into our instance. Amazon EC2 supports ED25519 and RSA key pair types.

Once the instance is created the window looks like fig 6.30.

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
<input type="checkbox"/>	-	i-0c5c5713b5a7893e2	Running	t2.micro	-	No alarms	us-east-1a

Fig 6.30. Instance page

We can see the public private keys by check the box near the instance. The next step is to generate key using puttygen. When click the load button browse

window will open from there we can load the key we have saved while creating ec2 instance

putty

Putty is a free and open-source terminal emulator, serial console and network file transfer application. It supports several network protocols,

including SCP, SSH, Telnet, rlogin, and raw socket connection. It can also connect to a serial port. The name "PuTTY" has no official meaning.

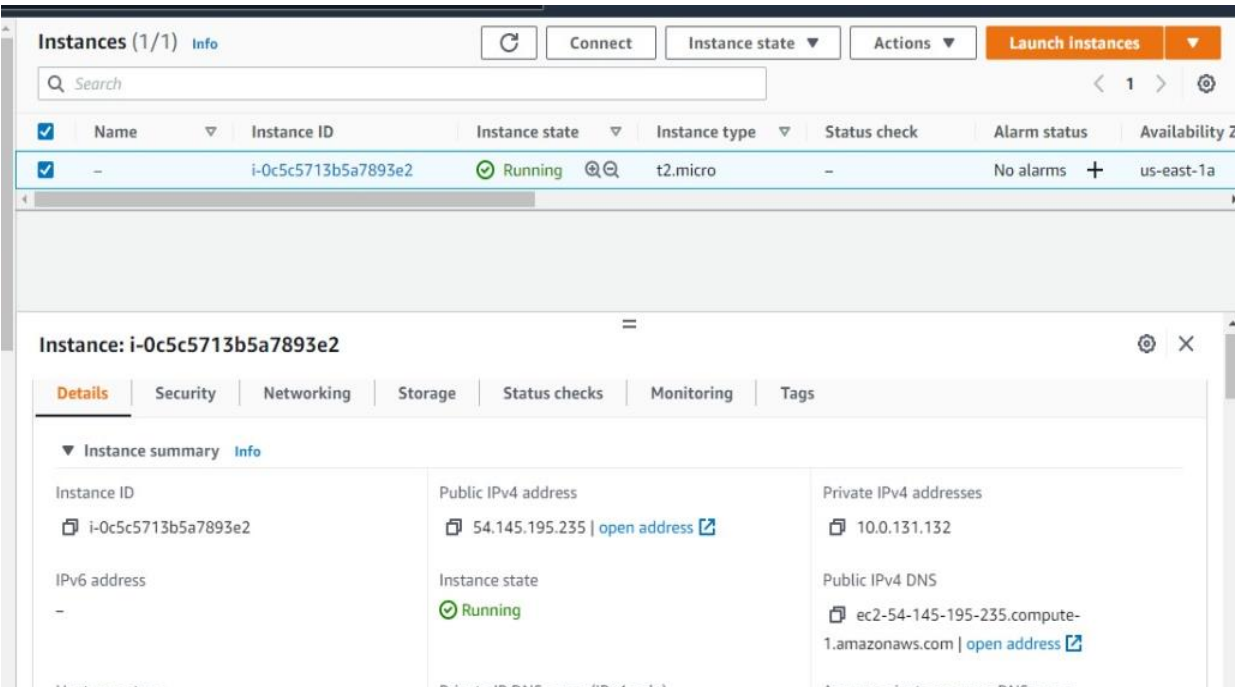


Fig 6.31. Instance properties

PuTTY was originally written for Microsoft Windows, but it has been ported to various other operating systems. Official ports are available for some Unix-like platforms, with work-in-progress ports to Classic Mac OS and macOS, and unofficial ports have been contributed to platforms such as Symbian, Windows Mobile and Windows Phone.

PuTTY was written and is maintained primarily by Simon Tatham, a British programmer. PuTTY supports many variations on the secure remote terminal, and provides user control over the SSH encryption key and protocol version, alternate ciphers such as AES, 3DES, RC4, Blowfish, DES, and Public-key authentication. PuTTY uses its own format of key files – PPK (protected by *Message Authentication Code*).

PuTTY supports SSO through GSSAPI, including user provided GSSAPI DLLs. It also can emulate control sequences from xterm, VT220, VT102 or ECMA-48 terminal emulation, and allows local, remote, or dynamic port forwarding with SSH (including X11 forwarding). PuTTY comes bundled with command line SCP and SFTP clients, called "pscp" and "psftp" respectively, and plink, a command-line connection tool, used for non-interactive sessions.

PuTTY does not support session tabs directly,^[10] but many wrappers are available that do.

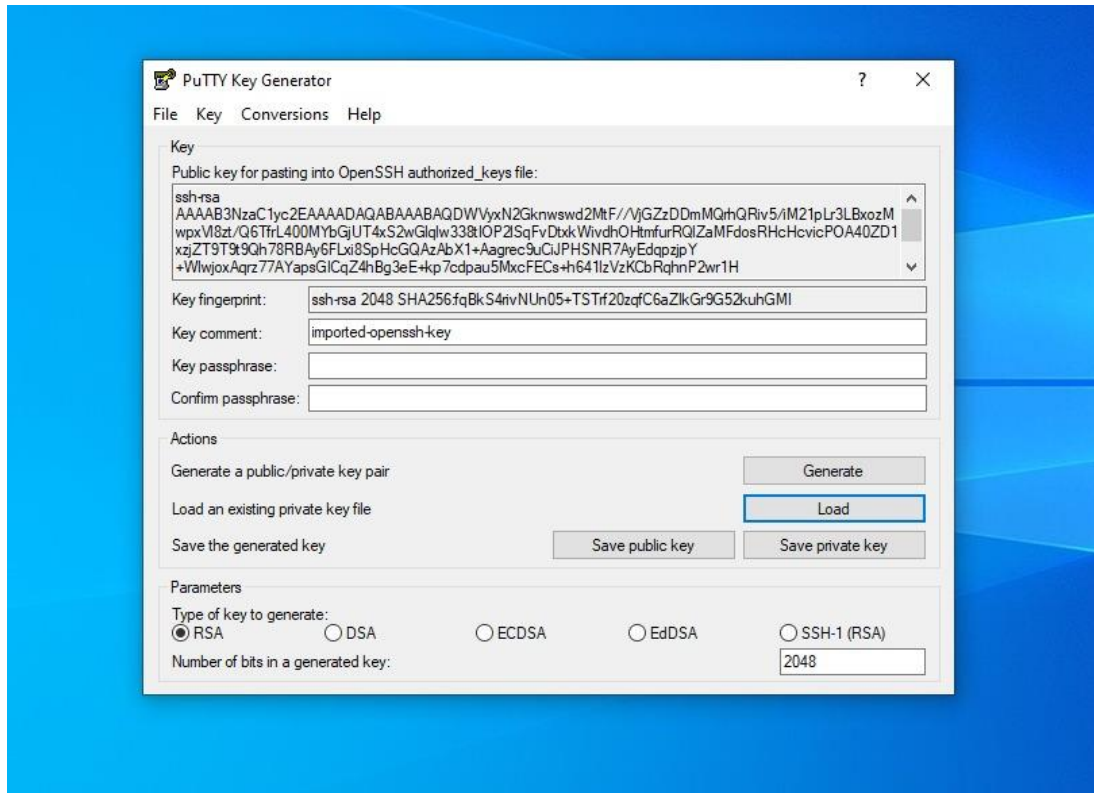


Fig 6.32. Keygen

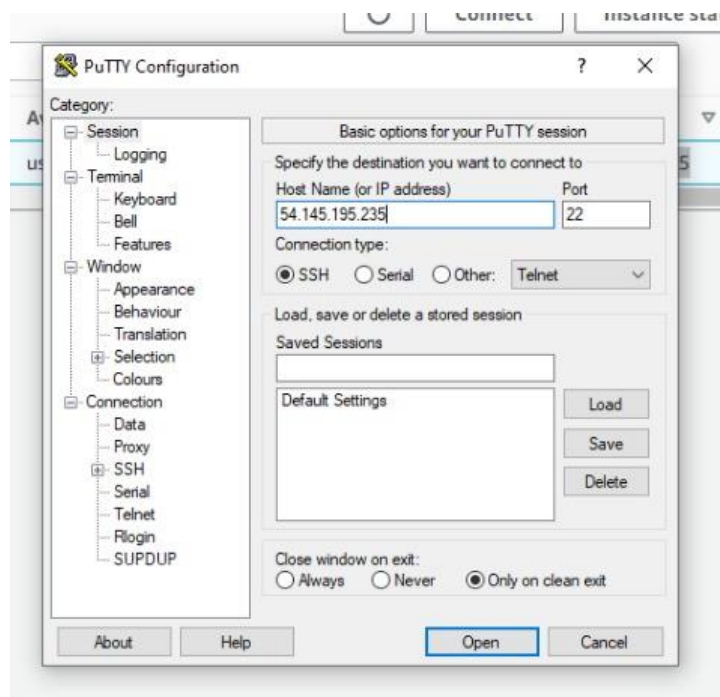


Fig 6.33. Key configuration

In configuration we need to give host name. Host name is nothing but we can get from ec2 instance.

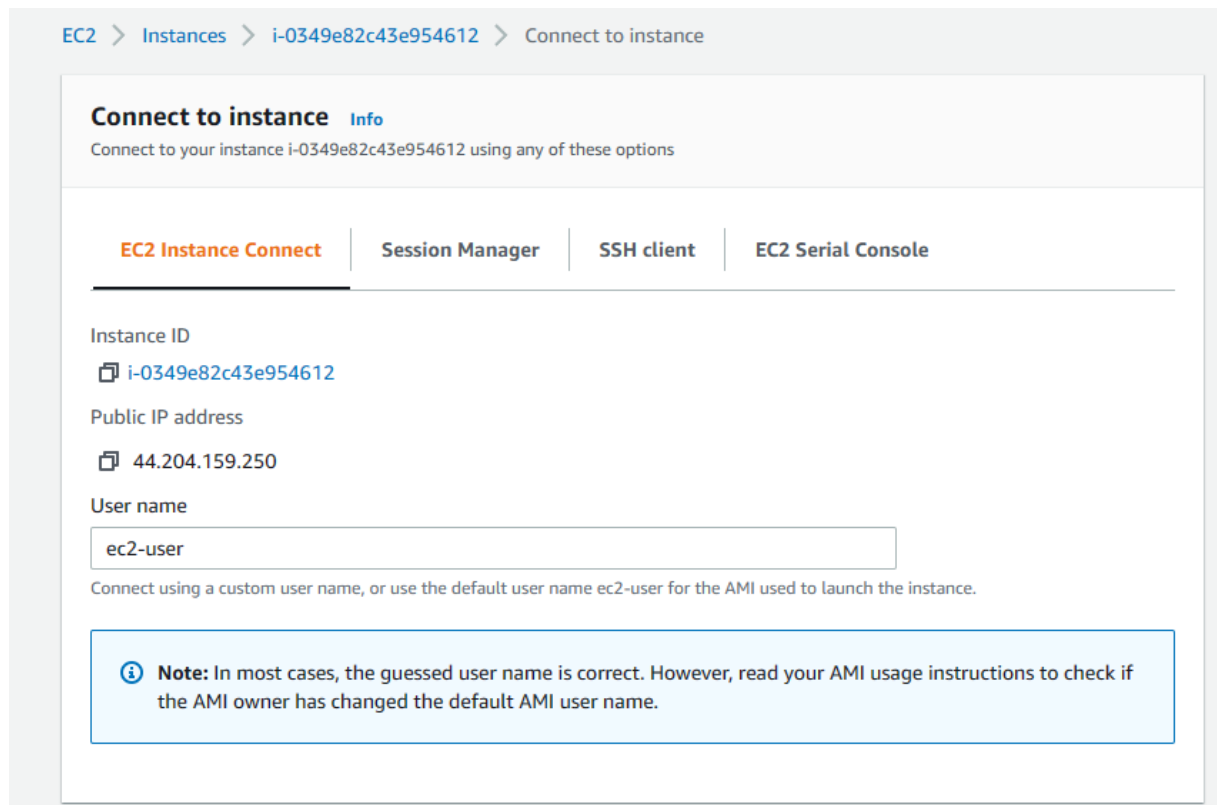


Fig 6.34. Instance connect page

Here we can see the public ip address. This is one which helps us to run our instance to public network. And below that we can see user name. This is our host name.

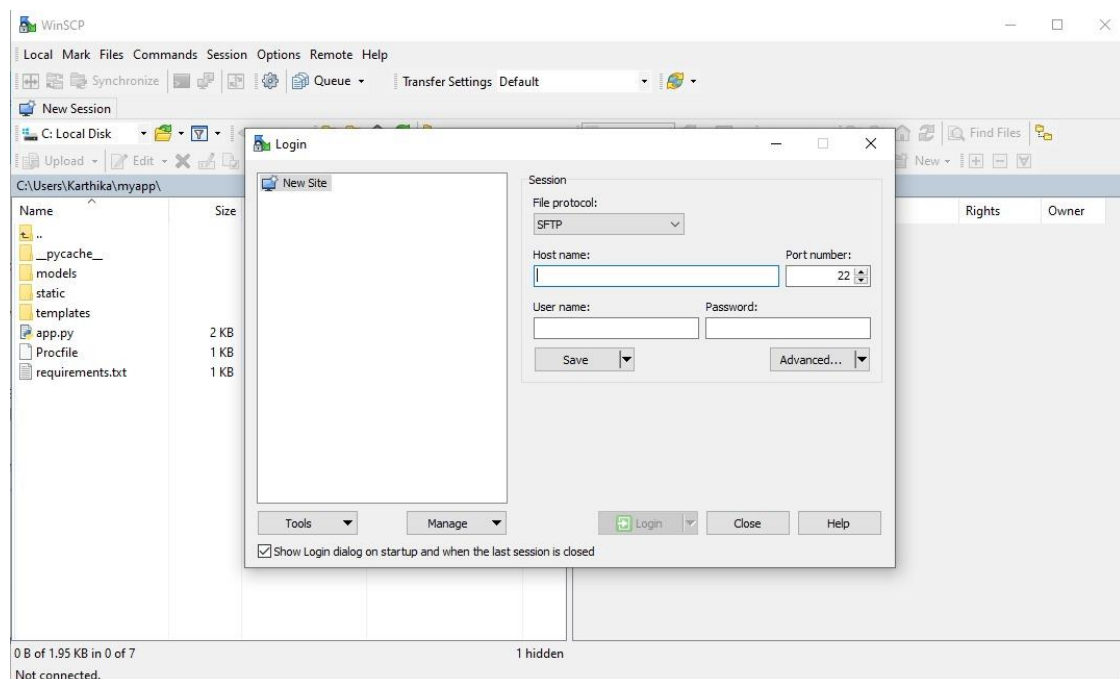


Fig 6.35. WinSCP

Next step is to configure winscp. Here also we need to configure with host name.

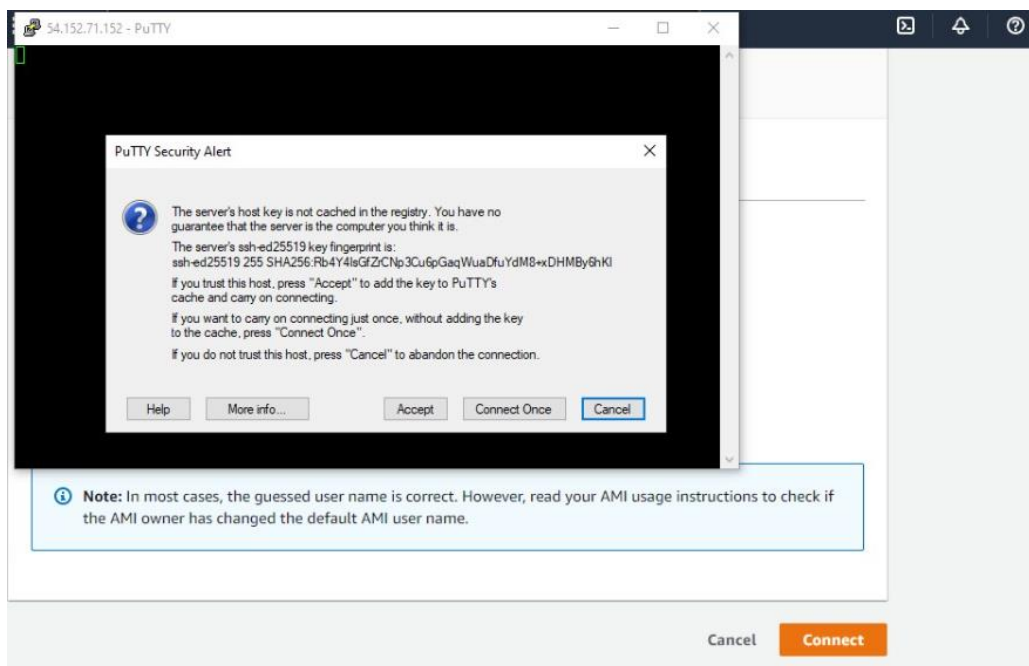


Fig 6.36. Connecting to instance

after all the configuration putty will ask permission to connect to ec2. Then we will connect to our instance. Finally we have to run our code in linux console. Now our

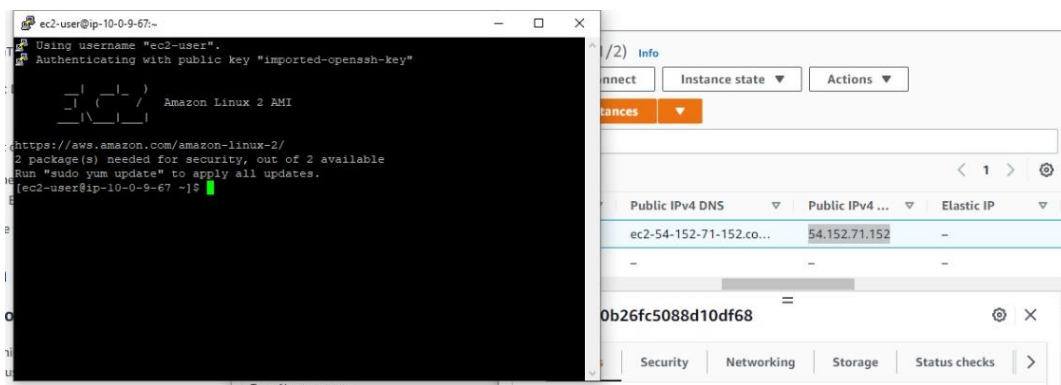


Fig 6.37. Connected to instance

application is running in ec2 instance.

6.5.6.1. Features of Amazon EC2sAmazon EC2 provides the following features:

- Virtual computing environments, known as *instances*
- Preconfigured templates for your instances, known as *Amazon Machine Images (AMIs)*, that package the bits you need for your server (including the operating system and additional software)
- Various configurations of CPU, memory, storage, and networking capacity for your instances, known as *instance types*

- Secure login information for your instances using *key pairs* (AWS stores the public key, and you store the private key in a secure place)
- Storage volumes for temporary data that's deleted when you stop, hibernate, or terminate your instance, known as *instance store volumes*
- Persistent storage volumes for your data using Amazon Elastic Block Store (Amazon EBS), known as *Amazon EBS volumes*
- Multiple physical locations for your resources, such as instances and Amazon EBS volumes, known as *Regions* and *Availability Zones*
- A firewall that enables you to specify the protocols, ports, and source IP ranges that can reach your instances using *security groups*
- Static IPv4 addresses for dynamic cloud computing, known as *Elastic IP addresses*
- Metadata, known as *tags*, that you can create and assign to your Amazon EC2 resources
- Virtual networks you can create that are logically isolated from the rest of the AWS Cloud, and that you can optionally connect to your own network, known as *virtual private clouds* (VPCs)

6.5.6.2. Types of EC2 instance

When you launch an instance, the *instance type* that you specify determines the hardware of the host computer used for your instance. Each instance type offers different compute, memory, and storage capabilities, and is grouped in an instance family based on these capabilities. Select an instance type based on the requirements of the application or software that you plan to run on your instance.

Amazon EC2 provides each instance with a consistent and predictable amount of CPU capacity, regardless of its underlying hardware. Amazon EC2 dedicates some resources of the host computer, such as CPU, memory, and instance storage, to a particular instance. Amazon EC2 shares other resources of the host computer, such as the network and the disk subsystem, among instances. If each instance on a host computer tries to use as much of one of these shared resources as possible, each receives an equal share of that resource.

However, when a resource is underused, an instance can consume a higher

Instance	vCPU*	CPU Credits / hour	Mem (GiB)	Storage	Network Performance
t2.nano	1	3	0.5	EBS-Only	Low
t2.micro	1	6	1	EBS-Only	Low to Moderate
t2.small	1	12	2	EBS-Only	Low to Moderate
t2.medium	2	24	4	EBS-Only	Low to Moderate
t2.large	2	36	8	EBS-Only	Low to Moderate
t2.xlarge	4	54	16	EBS-Only	Moderate
t2.2xlarge	8	81	32	EBS-Only	Moderate

Fig 6.38. T2 family

share of that resource while it's available. In this project we going to use **t2.micro** instance type. It is free tier eligible. Here t is instance class, 2 – generation, micro is size within the instance.EBS is nothing but root volume of our instance. It is call Elastic Block Storage.

6.5.7. Security Groups

- Security Group is the main and fundamental configuration to our EC2 instance. A security group is a virtual firewall which is controlling the traffic to your EC2 instances.
- When you first launch an EC2 instance, you can associate it with one or more security groups.
- A Security group is the first defense against hackers.

6.5.8. Load balancers

Amazon Load Balancer distributes network traffic across its components like Amazon EC2 instances, AWS Lambda. If there are many users for our application. We can scale our ec2 instance into many. Then we can set load balancers for instance. It uses some algorithms and divide the network traffic to our instances.

6.5.9. AMI

AMI abbreviation is Amazon Machine Image. AMI is simply the operating system. Here we choose whatever OS we want and after that we can virtually work on it.

All types of operating system are available in amazon web service.

Example: Linux, Windows, Mac, etc.

6.6. Linux

Linux is the safest operating system. We are going to create a Linux AMI in AWS and going to work on Linux environment.

SSH stands for secure shell. It is a protocol used to securely connect to a remote server/system. SSH is secure in the sense that it transfers the data in encrypted form between the host and the client. It transfers inputs from the client to the host and relays back the output. SSH runs at TCP/IP port 22.

6.7. S3 Bucket

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. Customers of all sizes and industries can use Amazon S3 to store and protect any amount of data for a range of use cases, such as data lakes, websites, mobile applications, backup and restore, archive, enterprise applications, IoT devices, and big data analytics. Amazon S3 provides management features so that you can optimize, organize, and configure access to your data to meet your specific business, organizational, and compliance requirements.

6.7.1. Features of Amazon S3

6.7.1.1. Storage classes

Amazon S3 offers a range of storage classes designed for different use cases. For example, you can store mission-critical production data in S3 Standard for frequent access, save costs by storing infrequently accessed data in S3 Standard-IA or S3 One Zone-IA, and archive data at the lowest costs in S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval, and S3 Glacier Deep Archive. You can store data with changing or unknown access patterns in S3 Intelligent-Tiering, which optimizes storage costs by automatically moving your data between four access tiers when your access patterns change. These four

access tiers include two low-latency access tiers optimized for frequent and infrequent access, and two opt-in archive access tiers designed for asynchronous access for rarely accessed data.

6.7.1.2. Storage management

Amazon S3 has storage management features that you can use to manage costs, meet regulatory requirements, reduce latency, and save multiple distinct copies of your data for compliance requirements.

- *S3 Lifecycle* – Configure a lifecycle policy to manage your objects and store them cost effectively throughout their lifecycle. You can transition objects to other S3 storage classes or expire objects that reach the end of their lifetimes.
- *S3 Object Lock* – Prevent Amazon S3 objects from being deleted or overwritten for a fixed amount of time or indefinitely. You can use Object Lock to help meet regulatory requirements that require *write-once-read-many (WORM)* storages or to simply add another layer of protection against object changes and deletions.
- *S3 Replication* – Replicate objects and their respective metadata and object tags to one or more destination buckets in the same or different AWS Regions for reduced latency, compliance, security, and other use cases.
- *S3 Batch Operations* – Manage billions of objects at scale with a single S3 API request or a few clicks in the Amazon S3 console. You can use Batch Operations to perform operations such as **Copy**, **Invoke AWS Lambda function**, and **restore** on millions or billions of objects.

6.7.1.3. Access management

Amazon S3 provides features for auditing and managing access to your buckets and objects. By default, S3 buckets and the objects in them are private. You have access only to the S3 resources that you create. To grant granular resource permissions that support your specific use case or to audit the permissions of your Amazon S3 resources, you can use the following features.

- *S3 Block Public Access* – Block public access to S3 buckets and objects. By default, Block Public Access settings are turned on at the account and bucket level.
- *AWS Identity and Access Management (IAM)* – Create IAM users for your AWS account to manage access to your Amazon S3 resources. For example, you can use

IAM with Amazon S3 to control the type of access a user or group of users has to an S3 bucket that your AWS account owns.

- Bucket policies – Use IAM-based policy language to configure resource-based permissions for your S3 buckets and the objects in them.
- Amazon S3 access points – Configure named network endpoints with dedicated access policies to manage data access at scale for shared datasets in Amazon S3.
- Access control lists (ACLs) – Grant read and write permissions for individual buckets and objects to authorized users. As a general rule, we recommend using S3 resource-based policies (bucket policies and access point policies) or IAM policies for access control instead of ACLs. ACLs are an access control mechanism that predates resource-based policies and IAM.
- S3 Object Ownership – Disable ACLs and take ownership of every object in your bucket, simplifying access management for data stored in Amazon S3. You, as the bucket owner, automatically own and have full control over every object in your bucket, and access control for your data is based on policies.
- Access Analyzer for S3 – Evaluate and monitor your S3 bucket access policies, ensuring that the policies provide only the intended access to your S3 resources.

6.7.1.4. Data processing

To transform data and trigger workflows to automate a variety of other processing activities at scale, you can use the following features.

- S3 Object Lambda – Add your own code to S3 GET requests to modify and process data as it is returned to an application. Filter rows, dynamically resize images, redact confidential data, and much more.
- Event notifications – Trigger workflows that use Amazon Simple Notification Service (Amazon SNS), Amazon Simple Queue Service (Amazon SQS), and AWS Lambda when a change is made to your S3 resources.
- Amazon CloudWatch metrics for Amazon S3 – Track the operational health of your S3 resources and configure billing alerts when estimated charges reach a user-defined threshold.
- AWS CloudTrail – Record actions taken by a user, a role, or an AWS service in Amazon S3. CloudTrail logs provide you with detailed API tracking for S3 bucket-level and object-level operations.

6.7.2. Creating S3 bucket to store our code files.

Bucket name should be unique over the global. EC2 region and bucket

Create bucket [Info](#)

Buckets are containers for data stored in S3. [Learn more](#)

General configuration

Bucket name

Bucket name must be unique and must not contain spaces or uppercase letters. [See rules for bucket naming](#)

AWS Region

Copy settings from existing bucket - *optional*
Only the bucket settings in the following configuration are copied.

Block Public Access settings for this bucket

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

☒ **Block all public access**
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

- ☒ **Block public access to buckets and objects granted through *new* access control lists (ACLs)**
S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.
- ☒ **Block public access to buckets and objects granted through *any* access control lists (ACLs)**
S3 will ignore all ACLs that grant public access to buckets and objects.
- ☒ **Block public access to buckets and objects granted through *new* public bucket or access point policies**
S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.
- ☒ **Block public and cross-account access to buckets and objects through *any* public bucket or access point policies**
S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

Fig 6.39. Bucket creation

region should be same. Then we may get error. By default, all the public access are denied. If we want to give some access is our wish. After created the bucket the console looks like fig 6.40.

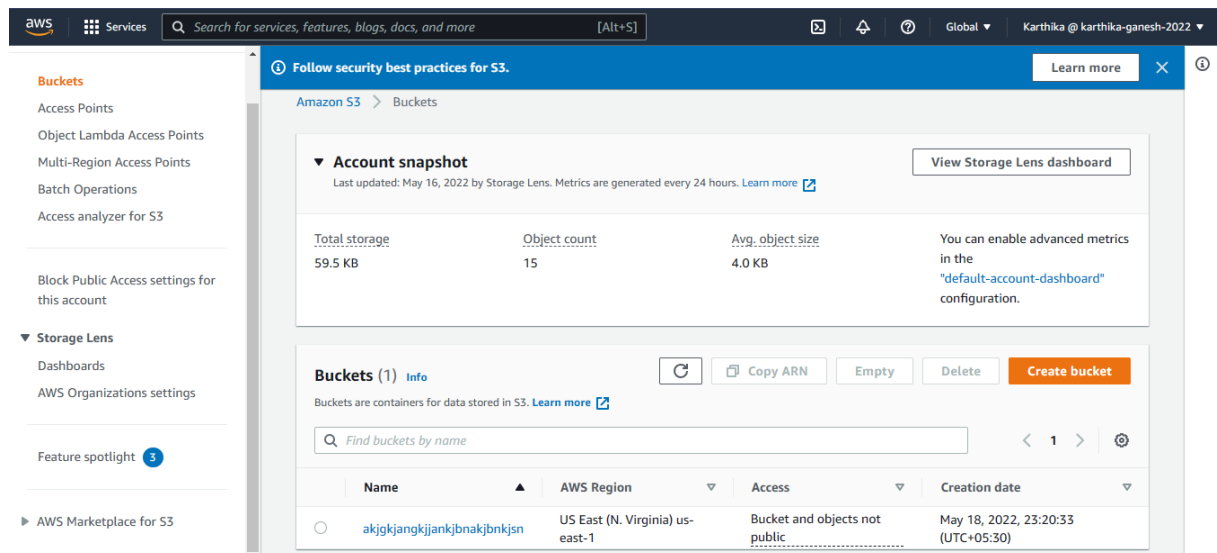


Fig 6.40. Bucket

Now we going to upload all our code files, dependencies into s3 bucket. Simply press the bucket and upload files.

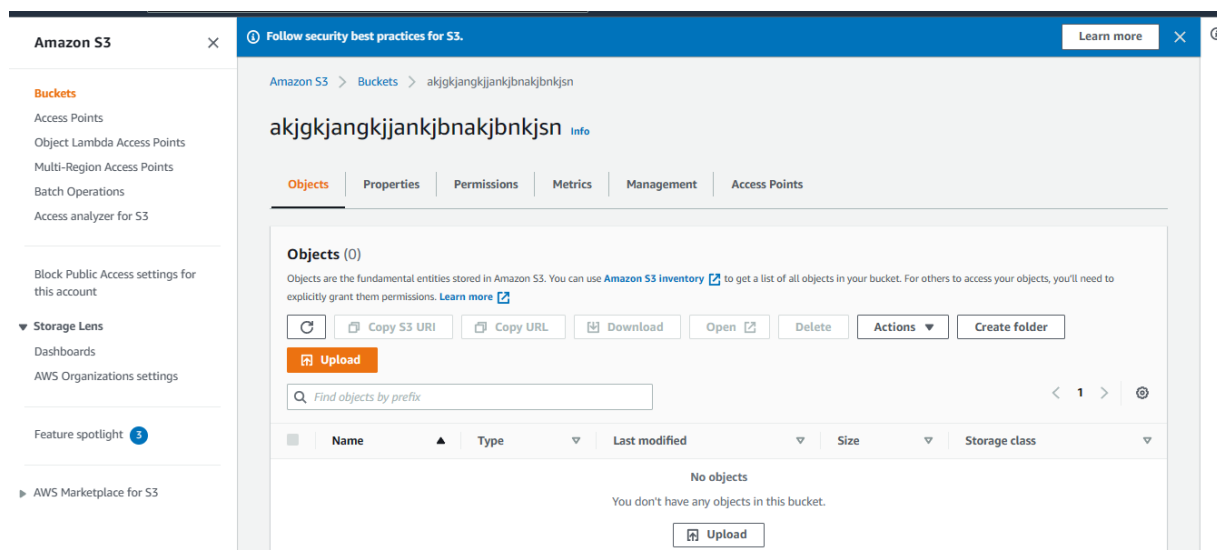


Fig 6.41. Inside bucket

It will browse into our local system. We can upload whatever files we need.

6.8. Cloud Formation

AWS CloudFormation is a service that helps you model and set up your AWS resources so that you can spend less time managing those resources and more time

focusing on your applications that run in AWS. You create a template that describes all the AWS resources that you want (like Amazon EC2 instances or Amazon RDS DB instances), and CloudFormation takes care of provisioning and configuring those resources for you. You don't need to individually create and configure AWS resources and figure out what's dependent on what; CloudFormation handles that. The following scenarios demonstrate how CloudFormation can help.

Now we going to create s3 bucket without the use of amazon console. For that we are going to create a .yaml file.

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  S3Bucket:
    Type: 'AWS::S3::Bucket'
    Description: Creating S3 bucket with lifecycle rule
    Properties:
      BucketName: karthika-cloud-formation-0001
      LifecycleConfiguration:
        Rules:
          - Id: GlacierRule
            Prefix: glacier
            Status: Enabled
            ExpirationInDays: 365
            Transitions:
              - TransitionInDays: 1
                StorageClass: GLACIER
```

Fig 6.42. Yaml code to create bucket

CHAPTER 7

SYSTEM TESTING

7.1. Unit testing:

Unit testing is a level of software testing where individual units' components of software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of software. It usually has one or a few inputs and usually a single output. In procedural programming a unit may be an individual program, function, procedure, etc. In object-oriented programming, the smallest unit is a method, which may belong to a base super class, abstract class or derived child class.

Unit testing is a component of test-driven development (TDD), a pragmatic methodology that takes a meticulous approach to building a product by means of continual testing and revision. Test-driven development requires that developers first write failing unit tests. Then they write code and refactor the application until the test passes. TDD typically results in an explicit and predictable code base. Unit testing involves only those characteristics that are vital to the performance of the unit under test. This encourages developers to modify the source code without immediate concerns about how such changes might affect the functioning of other units or the program as a whole. Once all of

the units in a program have been found to be working in the most efficient and error-free manner possible, larger components of the program can be evaluated by means of integration testing.

7.2. User Acceptance testing:

User acceptance is a type of testing performed by the Client to certify the system with respect to the requirements that was agreed upon. This testing happens in the final phase of testing before moving the software application to Market or Production environment.

The rule number one when trying to understand a new concept is that: the name is always going to be relevant and mostly a literal meaning (in the technical context). Finding out what that is, will give an initial understanding of it and gets started. Let's put this concept to test. UAT stands for User Acceptance Testing. The testing is, acceptance

means approval or agreement. User in the context of a software product is either the consumer of the software or the person who requested it to be built for him/her (client).

7.3. System Testing:

System Testing is a level of the software testing where complete and integrated software is tested. The purpose of this test is to evaluate the system's compliance with the specified requirements. System testing is the testing of a complete and fully integrated software product. Usually, software is only one element of a larger computer-based system. Ultimately, software is interfaced with other software/hardware systems. System testing is actually a series of different tests whose sole purpose is to exercise the full computer-based system.

CHAPTER 8

RESULT

The project starts with preprocessing the dataset and step by step model was built. We have compared many models and finally built the application using deep neural network. Comparing to other machine learning algorithms neural network performed well.

Algorithm	Accuracy (%)	Precision (%)	F1 Score (%)	Recall Score (%)
Logistic regression	0.98	0.94	0.97	1.0
Support Vector Machine	0.98	0.96	0.98	0.99
K-Nearest Neighbors	0.95	0.89	0.94	1.0
Decision Tree	0.97	0.97	0.97	0.97
Neural network	0.99	1.0	1.0	1.0

Table 8.1

After build the web application it is giving accurate results.

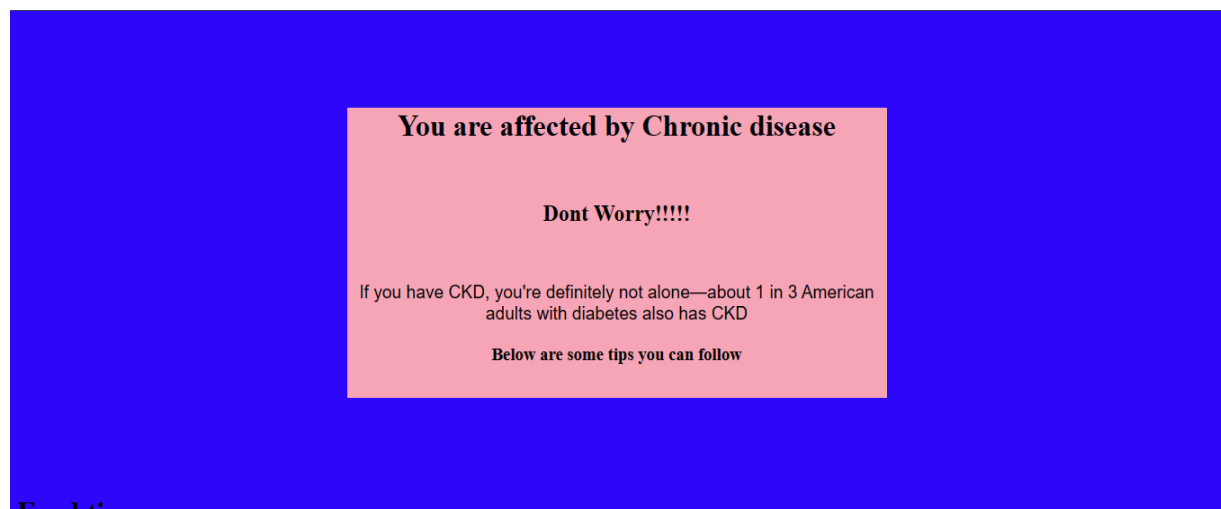


Fig 8.1 (web page)

CHAPTER 9

CONCLUSION

This project's final result satisfied the testing. And web application running on ec2 instance smoothly. Our project goal was also attained. We successfully compared many machine learning algorithms. Based on comparison we decided deep learning is our model. Our application giving accurate answer. In the backend deep learning processing very well with speed and accuracy. We removed many attributes in the dataset. Because they are not necessary. With the final thirteen attributes our model performed good. We saved our model using pickle package. So, we don't need to do the whole process while our program is invoked. This also decrease the latency. Now we worked under Amazon free tier. After successfully ran our program, we can terminate the instance or else we can stop the instance. Because only seven hundred and fifty hours is free per month. If we want to start our instance when ever our program is called, we can do that. At that time AWS lambda will help. It is serverless service in AWS. Here we can write python code and made the issue possible.

Web application link: <https://kidney-disease-predictionn.herokuapp.com/>

CHAPTER 10

FUTURE ENHANCEMENT

Now we created a chronic kidney disease prediction application and deployed it in ec2 instance and stored its data in s3 bucket. But our application will work if and only if the user provided all the needed information like sugar level, blood pressure level, albumin level, urea level, etc. To collect these data users, need to wait for some time to take the test and to get the report. Now a days there are many smart technologies which can tell a person's sugar level, bp level, etc. But many people can't afford more costs. So, in future a data science project should be developed for predicting each and every attribute needed for chronic kidney disease prediction. Persons sugar level can be predict using his daily life cycle details. Or else he can track his health daily in data science website. This may make a person stay healthy 50 percent.

APPENDIX I

Data Set

a) Raw:

age	bp	sg	al	su	rbc	pc	pcc	ba	bgr	bu	sc	sod	pot	hemo	pcv	wbcc	rbcc	htn	dm	cad	appet	pe	ane	class	
48	80	1.02	1	0	?	normal	notprese	notprese	?	121	36	1.2	?	?	15.4	44	7800	5.2	yes	yes	no	good	no	no	ckd
7	50	1.02	4	0	?	normal	notprese	notprese	?	?	18	0.8	?	?	11.3	38	6000	?	no	no	no	good	no	no	ckd
62	80	1.01	2	3	normal	normal	notprese	notprese	?	423	53	1.8	?	?	3.6	31	7500	?	no	yes	no	poor	no	yes	ckd
48	70	1.005	4	0	normal	abnorma	present	notprese	?	117	56	3.8	111	2.5	11.2	32	6700	3.9	yes	no	no	poor	yes	yes	ckd
51	80	1.01	2	0	normal	normal	notprese	notprese	?	106	26	1.4	?	?	11.6	35	7300	4.6	no	no	no	good	no	no	ckd
60	30	1.015	3	0	?	?	notprese	notprese	?	74	25	1.1	142	3.2	12.2	39	7800	4.4	yes	yes	no	good	yes	no	ckd
68	70	1.01	0	0	?	normal	notprese	notprese	?	100	54	24	104	4	12.4	36	?	?	no	no	no	good	no	no	ckd
24	?	1.015	2	4	normal	abnorma	notprese	notprese	?	410	31	1.1	?	?	12.4	44	6900	5	no	yes	no	good	yes	no	ckd
52	100	1.015	3	0	normal	abnorma	present	notprese	?	138	60	1.3	?	?	10.8	33	9600	4	yes	yes	no	good	no	yes	ckd
53	30	1.02	2	0	abnorma	abnorma	present	notprese	?	70	107	7.2	114	3.7	3.5	29	12100	3.7	yes	yes	no	poor	no	yes	ckd
50	60	1.01	2	4	?	abnorma	present	notprese	?	490	55	4	?	?	3.4	28	?	?	yes	yes	no	good	no	yes	ckd
63	70	1.01	3	0	abnorma	abnorma	present	notprese	?	380	60	2.7	131	4.2	10.8	32	4500	3.8	yes	yes	no	poor	yes	no	ckd
68	70	1.015	3	1	?	normal	present	notprese	?	208	72	2.1	138	5.8	3.7	28	12200	3.4	yes	yes	yes	poor	yes	no	ckd
68	70	?	?	?	?	?	notprese	notprese	?	38	86	4.6	135	3.4	3.8	?	?	?	yes	yes	yes	poor	yes	no	ckd
68	80	1.01	3	2	normal	abnorma	present	present	?	157	30	4.1	130	6.4	5.6	16	11000	2.6	yes	yes	yes	poor	yes	no	ckd
40	80	1.015	3	0	?	normal	notprese	notprese	?	76	162	3.6	141	4.3	7.6	24	3800	2.8	yes	no	no	good	no	yes	ckd
47	70	1.015	2	0	?	normal	notprese	notprese	?	39	46	2.2	138	4.1	12.6	?	?	?	no	no	no	good	no	no	ckd
47	80	?	?	?	?	?	notprese	notprese	?	114	87	5.2	133	3.7	12.1	?	?	?	yes	no	no	poor	no	no	ckd
60	100	1.025	0	3	?	normal	notprese	notprese	?	263	27	1.3	135	4.3	12.7	37	11400	4.3	yes	yes	yes	good	no	no	ckd
62	60	1.015	1	0	?	abnorma	present	notprese	?	100	31	1.6	?	?	10.3	30	5300	3.7	yes	no	yes	good	no	no	ckd
61	80	1.015	2	0	abnorma	abnorma	notprese	notprese	?	173	148	3.3	135	5.2	7.7	24	9200	3.2	yes	yes	yes	poor	yes	yes	ckd
60	30	?	?	?	?	?	notprese	notprese	?	?	180	76	4.5	?	10.3	32	6200	3.6	yes	yes	yes	good	no	no	ckd
48	80	1.025	4	0	normal	abnorma	notprese	notprese	?	35	163	7.7	136	3.8	3.8	32	6300	3.4	yes	no	no	good	no	yes	ckd
21	70	1.01	0	0	?	normal	notprese	notprese	?	?	?	?	?	?	?	?	?	?	no	no	no	poor	no	yes	ckd
42	100	1.015	4	0	normal	abnorma	notprese	present	?	?	50	1.4	123	4	11.1	33	8300	4.6	yes	no	no	poor	no	no	ckd
61	60	1.025	0	0	?	normal	notprese	notprese	?	108	75	1.3	141	5.2	3.3	29	8400	3.7	yes	yes	no	good	no	yes	ckd
75	80	1.015	0	0	?	normal	notprese	notprese	?	156	45	2.4	140	3.4	11.6	35	10300	4	yes	yes	no	poor	no	no	ckd
69	70	1.01	3	4	normal	abnorma	notprese	notprese	?	264	87	2.7	130	4	12.5	37	9600	4.1	yes	yes	yes	good	yes	no	ckd
75	70	?	1	3	?	?	notprese	notprese	?	123	31	1.4	?	?	?	?	?	?	no	yes	no	good	no	no	ckd
68	70	1.005	1	0	abnorma	abnorma	present	notprese	?	?	28	1.4	?	?	12.3	38	?	?	no	no	yes	good	no	no	ckd
?	70	?	?	?	?	?	notprese	notprese	?	33	155	7.3	132	4.3	?	?	?	?	yes	yes	no	good	no	no	ckd
73	30	1.015	3	0	?	abnorma	present	notprese	?	107	33	1.5	141	4.6	10.1	30	7800	4	no	no	no	poor	no	no	ckd
61	30	1.01	1	1	?	normal	notprese	notprese	?	153	39	1.5	133	4.3	11.3	34	9600	4	yes	yes	no	poor	no	no	ckd
60	100	1.02	2	0	abnorma	abnorma	notprese	notprese	?	140	55	2.5	?	?	10.1	29	?	?	yes	no	no	poor	no	no	ckd
70	70	1.01	1	0	normal	?	present	present	?	171	153	5.2	?	?	?	?	?	?	no	yes	no	poor	no	no	ckd
65	30	1.02	2	1	abnorma	normal	notprese	notprese	?	270	39	2	?	?	12	36	9800	4.3	yes	yes	no	poor	no	yes	ckd
76	70	1.015	1	0	normal	normal	notprese	notprese	?	32	29	1.8	133	3.3	10.3	32	?	?	yes	no	no	good	no	no	ckd
72	80	?	?	?	?	?	notprese	notprese	?	137	65	3.4	141	4.7	9.7	28	6900	2.5	yes	yes	no	poor	no	yes	ckd

Fig A.1.1. Dataset

b) Data information:

Title: Early stage of Indians Chronic Kidney Disease (CKD)

Relevant Information:

Acronyms:

Age	Age
Bp	Blood pressure
Sg	Specific gravity
Al	Albumin
Su	Sugar
Rbc	Red blood cells
Pc	Pus cell
Pcc	Pus cell dumps

Ba	Bacteria
Bgr	Blood glucose random
Bu	Blood urea
Sc	Serum creatine
Sc	Serum creatine
Sod	Sodium
Pot	Potassium
Hemo	Hemoglobin
Pcv	Packed cell volume
Wc	White blood cells
Rc	Red blood cells
Htn	Hypertension
Dm	Diabetes mellitus
Cat	Coronary artery disease
Pe	Pedal edema
Ane	anemia
Class	class
Number of Instances:	400 (250 CKD, 150 notCKD)
Number of Attributes:	24 + class = 25 (11 numeric, 14 nominal)
Class Distribution:	
CKD	250
notCKD	150

c) Preprocessed:

age	bp	sg	al	su	rbc	bgr	bu	sc	sod	pot	hemo	pcv	wbcc	rbcc	appet	ane	class
62.0	80.0	1.010	2.0	3.0	1	423.0	53.0	1.8	138.0	4.4	9.60	31.0	7500.0	4.8	1	1	0
48.0	70.0	1.005	4.0	0.0	1	117.0	56.0	3.8	111.0	2.5	11.20	32.0	6700.0	3.9	1	1	0
51.0	80.0	1.010	2.0	0.0	1	106.0	26.0	1.4	138.0	4.4	11.60	35.0	7300.0	4.6	0	0	0
24.0	80.0	1.015	2.0	4.0	1	410.0	31.0	1.1	138.0	4.4	12.40	44.0	6900.0	5.0	0	0	0
52.0	100.0	1.015	3.0	0.0	1	138.0	60.0	1.9	138.0	4.4	10.80	33.0	9600.0	4.0	0	1	0
53.0	90.0	1.020	2.0	0.0	0	70.0	107.0	7.2	114.0	3.7	9.50	29.0	12100.0	3.7	1	1	0
63.0	70.0	1.010	3.0	0.0	0	380.0	60.0	2.7	131.0	4.2	10.80	32.0	4500.0	3.8	1	0	0
68.0	80.0	1.010	3.0	2.0	1	157.0	90.0	4.1	130.0	6.4	5.60	16.0	11000.0	2.6	1	0	0
61.0	80.0	1.015	2.0	0.0	0	173.0	148.0	3.9	135.0	5.2	7.70	24.0	9200.0	3.2	1	1	0
48.0	80.0	1.025	4.0	0.0	1	95.0	163.0	7.7	136.0	3.8	9.80	32.0	6900.0	3.4	0	1	0
42.0	100.0	1.015	4.0	0.0	1	121.0	50.0	1.4	129.0	4.0	11.10	39.0	8300.0	4.6	1	0	0
69.0	70.0	1.010	3.0	4.0	1	264.0	87.0	2.7	130.0	4.0	12.50	37.0	9600.0	4.1	0	0	0
68.0	70.0	1.005	1.0	0.0	0	121.0	28.0	1.4	138.0	4.4	12.90	38.0	8000.0	4.8	0	0	0
60.0	100.0	1.020	2.0	0.0	0	140.0	55.0	2.5	138.0	4.4	10.10	29.0	8000.0	4.8	1	0	0
70.0	70.0	1.010	1.0	0.0	1	171.0	153.0	5.2	138.0	4.4	12.65	40.0	8000.0	4.8	1	0	0
65.0	90.0	1.020	2.0	1.0	0	270.0	39.0	2.0	138.0	4.4	12.00	36.0	9800.0	4.9	1	1	0
76.0	70.0	1.015	1.0	0.0	1	92.0	29.0	1.8	133.0	3.9	10.30	32.0	8000.0	4.8	0	0	0
69.0	80.0	1.020	3.0	0.0	0	121.0	103.0	4.1	132.0	5.9	12.50	40.0	8000.0	4.8	0	0	0
82.0	80.0	1.010	2.0	2.0	1	140.0	70.0	3.4	136.0	4.2	13.00	40.0	9800.0	4.2	0	0	0

Fig A.1.2 pre-processed dataset

APPENDIX II

Code

a) Source Code:

Libraries:

```
1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4 import seaborn as sns
5 import pickle
6 from sklearn.preprocessing import LabelEncoder
7 from sklearn.preprocessing import StandardScaler
8 from sklearn.decomposition import PCA
9 from sklearn.linear_model import LogisticRegression
10 from sklearn.linear_model import LinearRegression
11 from sklearn.neighbors import KNeighborsClassifier
12 from sklearn.tree import DecisionTreeClassifier
13 from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score, classification_report
14 from sklearn.model_selection import cross_val_score
15 from sklearn.model_selection import train_test_split
16 from sklearn.svm import SVC
17 from keras import models, layers
18 import warnings
19 warnings.filterwarnings('ignore')
20
21
22
```

Fig A.2.1.

Preprocessing:

```
def preproc():
    df=pd.read_csv("chronic_kidney_disease_full.csv",na_values= "?")
    df=df.drop(["pc","pcc","ba","htn","dm","cad","pe"],axis=1)
    num_col = df.select_dtypes(include='float64')
    for col in num_col:
        df[col].fillna(np.round(df[col].median(),2),inplace=True)
    df['appet']=df['appet'].fillna(df['appet'].mode()[0])
    df['ane']=df['ane'].fillna(df['ane'].mode()[0])
    encoder = LabelEncoder()
    df['class']= encoder.fit_transform(df['class'])

    df['appet']= encoder.fit_transform(df['appet'])

    df['ane']= encoder.fit_transform(df['ane'])
    train_df = df.dropna() ## data frame without NaN values
    test_df = df[df['rbc'].isna()]
    encoder = LabelEncoder()
    train_df['rbc']= encoder.fit_transform(train_df['rbc'])

    rbc=train_df['rbc']
    rbc_train_df=train_df.drop("rbc",axis=1)
    rbc_train_df['rbc']=rbc

    rbc=test_df['rbc']
    rbc_test_df=test_df.drop("rbc",axis=1)
    rbc_test_df['rbc']=rbc

    rbc_x_train = rbc_train_df.iloc[:, :-1]
    rbc_y_train =rbc_train_df[["rbc"]]
    rbc_x_test = rbc_test_df.iloc[:, :-1]
    rbc_y_test = rbc_test_df[["rbc"]]
```

Fig A.2.2.

```

model = LogisticRegression()
model.fit(rbc_x_train,rbc_y_train)
pred = model.predict(rbc_x_test)
test_df['rbc']=pred

new_df=pd.concat([train_df,test_df],ignore_index=True) #concatinating test and train data previously splited

#saving as csv
new_df.to_csv("ChronicKidneyDiseaseProcessed.csv")

```

Fig A.2.3.

Back END:

```

df = pd.read_csv("ChronicKidneyDiseaseProcessed.csv")
X=df.iloc[:, :-1]
Y=df[["class"]]
#we are going to scale our data

sc_X = StandardScaler()
X= sc_X.fit_transform(X)

#Dimensionality reduction

from sklearn.decomposition import PCA
pca = PCA(n_components = 17)
X = pca.fit_transform(X)
explained_ratio= pca.explained_variance_ratio_
np.cumsum(explained_ratio)

## K-Nearest Neighbors

knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(x_train,y_train)
knn_pr=knn.predict(x_test)

filename = 'C:\Users\Karthika\flask_demo\SavedModels\knn_model.pkl'
pickle.dump(knn, open(filename, 'wb'))

## Decison Tree

dsn_tree = DecisionTreeClassifier(criterion="gini",splitter="random",random_state=10)
dsn_tree.fit(x_train,y_train)
dsn_pr = dsn_tree.predict(x_test)

filename = 'C:\Users\Karthika\flask_demo\SavedModels\decision_tree.pkl'
pickle.dump(dsn_tree, open(filename, 'wb'))

## Neural Network
model = models.Sequential()
model.add(layers.Dense(512,activation='relu',input_dim=12))
model.add(layers.Dense(216,activation='relu'))
model.add(layers.Dense(128,activation='relu'))
model.add(layers.Dense(84,activation='relu'))
model.add(layers.Dense(1,activation='sigmoid'))
model.compile(optimizer = 'adam', loss = 'binary_crossentropy', metrics = 'accuracy')
hist = model.fit(x_train,y_train,epochs = 200,verbose=0)
neu_pr = np.round(model.predict(x_test))
model.save('C:/Users/Karthika/flask_demo/SavedModels/DeepLearningModel')

```


Flask Web app development:

```
from flask import Flask, render_template, request
import numpy as np
import pickle
model = pickle.load(open('C:/Users/Karthika/flask_demo/SavedModels/logistic.pkl', 'rb'))
app = Flask(__name__)
@app.route("/")
def home():
    title = 'Kidney Disease Identify'
    return render_template("index.html", title = title)
@app.route("/diseaseinfo")
def disease():
    return render_template("diseaseinfo.html")
@app.route("/details")
def info():
    return render_template("details.html")
@app.route("/disease_prediction", methods = ['POST'])
def disease_prediction():
    title = 'Chronic Kidney Disease Prediction'
    info1 = request.form['age']
    info2 = request.form['blood_preasure']
    info3 = request.form['Specific_Gravity']
    info4 = request.form['Albumin']
    info5 = request.form['Sugar']
    info6 = request.form['Red_Blood_cells']
    if info6=="normal":
        info6 = 0
    else:
        info6 = 1
    info7 = request.form['Blood_Glucose']
    info8 = request.form['Blood_Urea']
    info9 = request.form['Serum_Creatinine']
    info10 = request.form['Sodium']
    info11 = request.form['Potassium']
    info12 = request.form['Hemoglobin']

    x_value = np.array([[info1, info2, info3, info4, info5, info6, info7, info8, info9, info10, info11, info12]])
    predicted = model.predict(x_value)

    return render_template("disease_prediction.html", data = predicted, title = title)

if __name__ == "__main__":
    app.run(debug=False)
```

Fig A.2.4.

FrontEND:

Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="/static/css/style.css">
  <title>Home</title>
</head>
<body>

  <a href="/" style="color: ■white">Home</a>&nbsp;<a href="/" style="color: ■white">about</a>

  <div class="head_box">
    <h1>CHRONIC PREDICTIFY</h1>
  </div>

  <div class="inner_box">
    <b><h3> I will predict do you have disease or not</34><br><br></b>
    <p> > Using Deep Learning Model I going to predict whether you are affected by chronic kidney disease or not. </p>
    <p>Click any one of the link to go that page</p><br><a href="/diseaseinfo" align="right">Chronic Kidney Disease</a><br>
    <a href="/details">Prediction</a>
  </div>

</body>
</html>
```

Fig A.2.5.

Filling Sectio.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Information</title>
</head>
<body bgcolor="#fad707">
  <div class="background-img">
    <br><br>
    <p align="right">
      Enter the below informations.. to predict
    </p>
    <div class="box">
      <form action="{{url_for('disease_prediction')}}" method="post" align="right">
        Age : <input type="text" name="age" placeholder="ex : 45"><br><br>
        Blood Preasure: <input type="text" name="blood_preasure" placeholder="ex : 120"><br><br>
        Specific Gravity: <input type="text" name="Specific_Gravity" placeholder="ex : 1.01"><br><br>
        Albumin: <input type="text" name="Albumin" placeholder="ex : 3"><br><br>
        Sugar: <input type="text" name="Sugar" placeholder="ex : 3"><br><br>
        Red_Blood_cells: <input type="text" name="Red_Blood_cells" placeholder="normal or abnormal"><br><br>
        Blood_Glucose: <input type="text" name="Blood_Glucose" placeholder="ex : 100"><br><br>
        Blood_Urea: <input type="text" name="Blood_Urea" placeholder="ex : 50"><br><br>
        Serum_Creatinine: <input type="text" name="Serum_Creatinine" placeholder="ex : 2"><br><br>
        Sodium: <input type="text" name="Sodium" placeholder="ex : 120"><br><br>
        Potassium: <input type="text" name="Potassium" placeholder="ex : 4.5"><br><br>
        Hemoglobin: <input type="text" name="Hemoglobin" placeholder="ex : 11"><br><br><br><br>
      </b>
      <input type="submit" value="Predict">
    </div>
  </div>
```

Fig A.2.6.

Final Prediction Page.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Prediction</title>
</head>
<body bgcolor="#2d07fa">
  <center>

    <div class="container">

      {%if data == 0%}
      <h1> You are affected by Chronic disease</h1><br>
      <h2>Dont Worry!!!!</h2><br>
      <p>
        If you have CKD, you're definitely not alone—about 1 in 3 American adults with diabetes also has CKD
      </p>

      <h3> Below are some tips you can follow</h3>

      {%else%}
      <h1> You are healthy </h1>

      {%endif%}
      <br><br>
    </div>

  </center>
  <h1>
Food tips:
</h1>
<br>
<br>
<p><b>
  <font face="Comic Sans">
    Eat less salt/sodium. That's a good move for diabetes and really important for CKD. Over time, your kidneys lose
    the ability to control your sodium-water balance. Less sodium in your diet will help lower blood pressure and
    decrease fluid buildup in your body, which is common in kidney disease. Focus on fresh, homemade food and eat
    only small amounts of restaurant food and packaged food, which usually have lots of sodium. Look for low sodium
    (5% or less) on food labels. In a week or two, you'll get used to less salt in your food, especially if you dial
    up the flavor with herbs, spices, mustard, and flavored vinegars. But don't use salt substitutes unless your
    doctor or dietitian says you can. Many are very high in potassium, which you may need to limit.
  </font>
</p>
</b>
<h1>
Exercise tips:
</h1>
<br>
<br>
<p><b>
  <font face="Comic Sans">
    Choose continuous activity such as walking, swimming, bicycling (indoors or out), skiing, aerobic dancing or
    any other activities in which you need to move large muscle groups continuously. Low-level strengthening
    exercises may also be beneficial as part of your program. Design your program to use low weights and high
    repetitions, and avoid heavy lifting.
  </font>
</p>
</b>
```

Fig A.2.7.

Style.css

```

*{
  margin: 0;
  padding: 0;
}

body{
  height: 100vh;
  background-image: url(../images/kidney.jpg);
  background-size: cover;
  background-repeat: no-repeat;
}

.small_box{
  position: relative;
  margin-top: 0%;
  padding: 0;
  margin: 0;
}

.head_box{
  background-image: url(../images/black.jpg);
  position: relative;
  padding: 30px;
  width: 400px;
  position: relative;
  left: 904px;
}

.head_box h1{
  font-size: xx-large;
  font-style: normal;
  font-family: Georgia, 'Times New Roman', Times, serif;
}

.inner_box{
  background-color: rgba(17, 15, 15, 0.815);
  transform: translate(-50%, -50%);
  box-shadow: 0 15px 20px rgba(0,0,0,.2);
  max-width: 650px;
  position: relative;
  background-position: left;
  display: block;
  top: 40%;
  left: 1200px;
  padding: 30px;
  width: 200px;
  text-align: justify;
}

.inner_box h3{
  font-weight: 200;
  color: aquamarine;
}

.inner_box p{
  font-size: larger;
  font-family: 'Gill Sans', 'Gill Sans MT', Calibri, 'Trebuchet MS', sans-serif;
  color: rgb(236, 241, 235);
}

h1{
  color: rgb(235, 224, 221);
  font-size: xx-large;
  font-style: normal;
  font-family: Georgia, 'Times New Roman', Times, serif;
}
```

Fig A.2.8.

REFERENCES

- [1] Logistic Regression Models Used in Medical Research Are Poorly Presented-Statistician Department of Metabolic Diseases and Nutrition, Heinrich-Heine-University Dusseldorf, Po Box 10 10 07, D-40001 Dusseldorf, Germany – Mar 21, 2017 published in IEEE
- [2] Logistic Regression Model Optimization and Case Analysis - Xiaonan Zou; Yong Hu; Zhewen Tian; Kaiyuan Shen-2019 IEEE 7th International Conference on Computer Science and Network Technology (ICCSNT)
- [3] Mathematics Behind Logistic Regression-Vinithavn Oct 21, 2020 published in analytics vidhya
- [4] Data classification using support vector machine- Durgesh K.Srinicastava, Lekha Bhambhu – 2021 published in medium.
- [5] Deep learning beyond cats and dogs: recent advances in diagnosing breast cancer with deep neural networks - Jeremy R Burt, Neslisah Torosdagli, Naji Khosravan, Harish RaviPrakash, Aliasghar Mortazi, Fiona Tissavirasingham, Sarfaraz Hussein and Ulas Bagci - 2019 IEEE 7th International Conference on Computer Science and Network Technology (ICCSNT)
- [6] Understanding of neural network -Saad Albawi; Tareq Abed Mohammed; Saad Al-Zawi 2017 International Conference on Engineering and Technology (ICET)
- [7] A Deep Learning Based Artificial Neural Network Approach for intrusion detection- Saniiban Sekar Roy, Abhinaya Mallik, Rishab Gulati Mohammad S. Obaidat & P. V. Krishna, Jun 14, 2020 published in Analytics Visdhya
- [8] Classification Using Deep Learning Neural Networks for Brain Tumors – Heba Mohsen EL Sayed A. El Dashan El Sayed El Horbaty Abdelbadeeh M Salem,
- [9] Chronic Kidney Disease: Global Dimension and Perspective-Prof Vivekandha Dm Proof Kunitosilseki Zuoli Prof Saraladevi Naicker Brett Plattner Prof Rajiyasaran Prof Angela Yee- Moon Wang Prof Chihei Yang
- [10] Chronic Kidney Disease: Global Challenge- Prof A Maguidel Nahas Aminu K Bello