Department of CSE – (DS,CS,AIML)

2022-2023

Micro Project Report

On

"EARLY PREDICTION OF DIABETIC RETINOPATHY"

Bachelor of Technology

In

COMPUTER SCIENCE AND ENGINEERING

CSE - DATA SCIENCE

By

G.VARSHITHA REDDY	-20R21A6727
D.SAI BHARATH VARMA	-20R21A6713
G.HARSHVARDHAN	-21R25A6703
G.NIHARIKA	-20R21A6723

UNDER THE GUIDANCE OF

Mr. KIRAN KUMAR REDDY

CONTENTS

CERTIFICATE

This is to certify that the project entitled "**EARLY PREDICTION OF DIABETIC RETINOPATHY**" has been submitted by G. VARSHITHA REDDY- (20R21A6727) D.SAI BHARATH VARMA- (20R21A6713) G.HARSHVARDHAN -(21R25A6703) G.NIHARIKA- (20R21A6723) in the partial fulfillment of the requirements for the award of degree of Bachelor of Technology in Computer Science and Engineering (Data Science) from MLR Institute of Technology, Hyderabad. The results embodied in this project have not been submitted to any other University or Institution for the award of any degree or diploma.

Internal Guide **Head of the Department**

External Examiner

DECLARATION

I hereby declare that the project entitled "**EARLY PREDICTION OF DIABETIC RETINOPATHY**" is the work done during the period from SEPT 2022 to DEC 2022 and is submitted in the partial fulfillment of the requirements for the award of degree of Bachelor of technology in Computer Science and Engineering (Data Science) from MLR Institute of Technology, Hyderabad. The results embodied in this project have not been submitted to any other university or Institution for the award of any degree or diploma.

20R21A6727-G.VARSHITHA REDDY

20R21A6713-D.SAI BHARATH VARMA

21R25A6703-G.HARSHVARDHAN

20R21A6723-G.NIHARIKA

ABSTRACT

The main aim for this project is early prediction of diabetic retinopathy (blindness caused due to high blood pressure) using machine learning from scanB ultra sonography screening images.Diabetic Retinopathy (DR) is an eye disease in humans with diabetes which may harm the retina of the eye and may cause total visual impairment. Therefore it is critical to detect diabetic retinopathy in the early phase to avoid blindness in humans.In at least 90% of new cases, progression to more aggressive forms of sight threatening retinopathy and maculopathy could be reduced with proper treatment and monitoring of the eyes. Our aim is to detect the presence of diabetic retinopathy by applying machine learning classifying algorithms. Hence we try and summarize the various models and techniques used along with methodologies used by them and analyze the accuracy and results.

KEYWORDS: Diabetic Retinopathy, Blindness, Machine Learning, SVM (Support Vector Machine) , Ultra Sonography, Maculopathy.

1) INTRODUCTION

People who have diabetes may develop diabetic retinopathy, an eye disorder that can lead to blindness and vision loss. The retina's blood vessels are impacted (the light-sensitive layer of tissue in the back of your eye). It's crucial to undergo a thorough dilated eye exam at least once a year if you have diabetes. Although diabetic retinopathy may not initially present with any symptoms, detecting it early might help you take precautions to preserve your vision. You can avoid or delay vision loss by controlling your diabetes by staying active, eating well, and taking your medications.

There are typically no symptoms in the early stages of diabetic retinopathy. Some patients experience vision changes, such as difficulty reading or seeing objects in the distance. These adjustments could come and go. Blood vessels in the retina begin to bleed into the vitreous in later stages of the disease (gel-like fluid that fills your eye). If this occurs, you might notice cobweb-like black, floaty areas or streaks. The spots may occasionally go away on their own, but it's still crucial to seek medical attention right soon. In the rear of the eye, scarring may develop if left untreated. Additionally, blood vessels may start to bleed once again or the bleeding may worsen.

Any kind of diabetes, including type 1, type 2, and gestational diabetes, can cause diabetic retinopathy (a type of diabetes that can develop during pregnancy). Your risk increases the longer you have diabetes. Over time, more than half of those with diabetes will develop diabetic retinopathy. The good news is that you can lower your chance of getting diabetic retinopathy by controlling your diabetes. Women with diabetes who become pregnant or acquire gestational diabetes are more likely to have diabetic retinopathy. If you have diabetes and are expecting, get a comprehensive dilated eye exam soon away. If you think you'll need any additional eye exams while you're pregnant, ask your doctor.

Diabetes-related elevated blood sugar results in diabetic retinopathy. The region of your retina that detects light and transmits messages to your brain via a nerve in the back of your eye may sustain long-term damage if there is too much sugar in your blood (optic nerve). All over the body, blood arteries are harmed by diabetes. Sugar damages your eyes by clogging the tiny blood vessels that supply the retina, which can lead to bleeding or fluid leakage. Your eyes then develop new, poorly functioning blood vessels to make up for these blocked blood vessels. These new blood vessels frequently bleed or leak blood.

2) LITERATURE REVIEW

A computer programme is said to learn from experience and from some tasks, and performance on some activities, as evaluated by, improves with experience, according to Tom Mitchell. The majority of machine learning algorithms in use are focused on identifying and/or utilising relationships between datasets. Machine learning is a combination of correlations and relationships. Once Machine Learning Algorithms are able to focus on specific correlations, the model can either generalise the data to highlight intriguing patterns or use these connections to forecast future observations.

Diabetes affects the retina's tiny blood vessels, which results in diabetic retinopathy. It may result in "blind patches," fuzziness, and vision loss. From one day to the next, or even from daylight to night, vision may vary. This "fluctuating vision" can make many, if not all, daily activities difficult. People with diabetic retinopathy can lead fulfilling lives with the aid of early detection, proper and continuous treatment, and the availability of specialised low vision and vision rehabilitation programmes.

There are many different types of algorithms used in machine learning, including regression, linear regression, logistic regression, the Bayes theorem, Naive Bayes classifier, decision tree, entropy, ID3, SVM (support vector machines), K-means algorithm, random forest, and others.

Arthur Samuel first used the term "machine learning" in 1959. Studying and creating algorithms that can learn from data and generate predictions is a topic covered by machine learning. Computational statistics, which similarly focuses on making predictions using computers, is closely connected to (and frequently overlaps with) machine learning. It is closely related to mathematical optimization, which provides the discipline with methods, theory, and application domains. Data mining, a field that focuses more on exploratory data analysis and is also known as unsupervised learning, is sometimes confused with machine learning.

Machine learning is a technique used in the field of data analytics to create intricate models and algorithms that are conducive to prediction; in the context of business, this is known as predictive analytics. Through learning from previous linkages and trends

in the data, these analytical models enable researchers, data scientists, engineers, and analysts to "create dependable, repeatable judgments and results" and find "hidden insights."

Machine learning tasks are typically classified into several broad categories:

1. Supervised learning
2. Semi-supervised learning
3. Active learning
4. Unsupervised learning
5. Reinforcement learning

FEATURES OF MACHINE LEARNING:

- It is nothing but automating the Automation.
- Getting computers to program themselves.
- Machine Learning is a combination of Algorithms, Datasets, and Programs.
- There are Many Algorithms in Machine Learning through which we will provide us the exact solution in predicting the disease of the patients.
- There are 3 types of machine learning supervised, unsupervised, and reinforcement.

EXISTING SYSTEM:

Traditional approaches and models for prediction contain a variety of risk factors and consist of many algorithms' metrics, including datasets, programmes, and much more. The classification of patients as High-risk or Low-Risk is based on the results of the group tests. However, these models are only useful in clinical settings; they are not useful in broad industry sectors. We therefore applied the principles of machine learning and supervised learning methods to construct the predictions system in order to include the illness predictions in many health-related industries.

Disadvantages:

- Difficulty in maintaining patient's symptoms.
- Not accurate.
- Algorithm takes more run time to give output.

PROPOSED SYSTEM:

The proposed system of diabetic retinopathy detection is that we have used many techniques and algorithms and all other various tools to build a system which predicts whether the patient is suffering with diabetic retinopathy or not using the symptoms and by taking those symptoms we are comparing with the system's dataset that is previously available. By taking those datasets and comparing with the patient's disease we will predict the accurate percentage of diabetic retinopathy of the patient. The dataset and symptoms go to the prediction model of the system where the data is pre-processed for the future references and then the future selection is done by the user where he will enter the various symptoms. Then the classification of those data is done with the help of various algorithms and techniques such as SVM etc. Here we have combined the overall structure and unstructured form of data for the overall risk analysis that is required for doing the prediction of the disease. We use SVM to categorize the features of the processed image and categorize them based on their matched features.

SVM:-

- SVM or Support Vector Machine is a linear model for classification and regression problems. It can solve linear and non-linear problems and work well for many practical problems.
- The idea of SVM is simple: The algorithm creates a line or a hyperplane which separates the data into classes.
- At first approximation what SVMs do is to find a separating line (or hyperplane) between data of two classes.
- SVM is an algorithm that takes the data as an input and outputs a line that separates those classes if possible.
- According to the SVM algorithm we find the points closest to the line from both the classes. These points are called support vectors.

- This distance is called the margin. Our goal is to maximize the margin. The hyperplane for which the margin is maximum is the optimal hyperplane.
- This way we maximise the accuracy of the model prediction.

Advantages:

- Faster processing when compared to existing one.
- Accurate Results.
- Early prediction is possible.

3) SOFTWARE REQUIREMENTS

INTRODUCTION

Python is a multi-paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of its features support functional programming and aspect-oriented programming (including by metaprogramming and metaobjects. Many other paradigms are supported via extensions, including design by contract and logic programming. Python uses dynamic typing and a combination of reference counting and a cycle-detecting garbage collector for memory management. It also features dynamic name resolution (late binding), which binds method and variable names during program execution.

Python's developers strive to avoid premature optimization, and reject patches to non-critical parts of Python that would offer marginal increases in speed at the cost of clarity. When speed is important, a Python programmer can move time-critical functions to extension modules written in languages such as C, or use PyPy, a just-in-time compiler. Python is also available, which translates a Python script into C and makes direct C-level API calls into the Python interpreter. An important goal of Python's developers is keeping it fun to use. Python's design offers some support for functional programming in the Lisp tradition. It has filter, map, and reduce functions, list comprehensions, dictionaries, sets, and generator expressions. The standard library has two modules (itertools and functools) that implement functional tools borrowed from Haskell and Standard ML.

BENEFITS OF PYTHON

- Presence of Third-Party Modules
- Extensive Support Libraries
- Open Source and Community Development
- Learning Ease and Support Available
- User-friendly Data Structures

4) HARDWARE REQUIREMENTS

--NO HARDWARE REQUIRMENTS, IT IS A SOFTWARE MACHIINE TRAINING --

5) SYSTEM DESIGN

IMPORTANCE OF DESIGN:

The Design goals consist of various design which we have implemented in our system early prediction of diabetic retinopathy.

We have designed our system in such a way that whenever a user gives a retina image of a patient which was scanned using ultrasonography the model process the given image and predict the diabetic retinopathy. This way by early prediction of the disease we can reduce the severity of the disease.

Disease prediction: The predictive model predicts the disease of a person he might have, based on the ultrasonography scanned image.

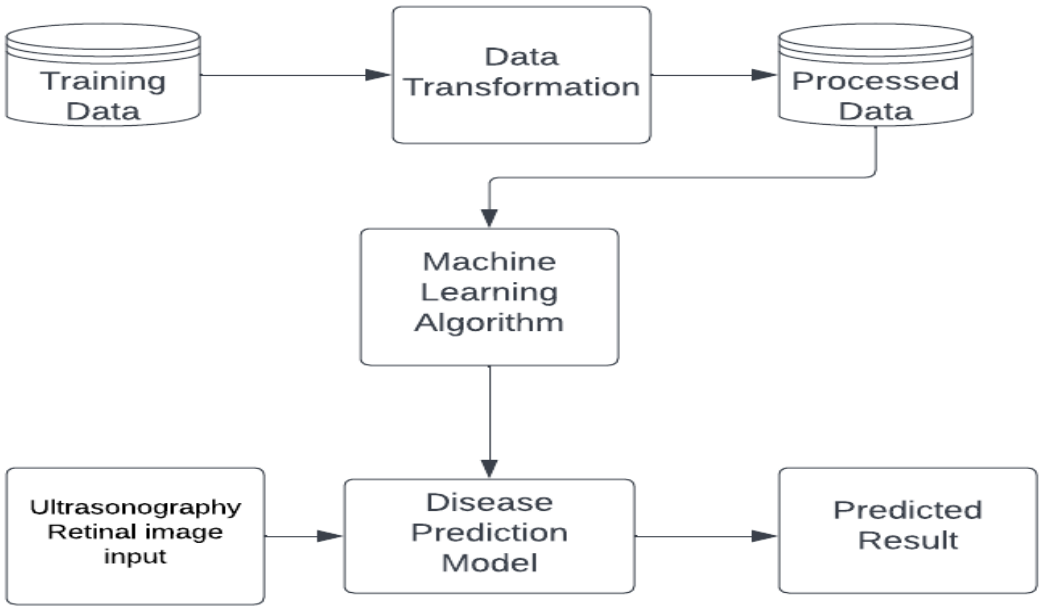


Fig: SYSTEM ARCHITECTURE

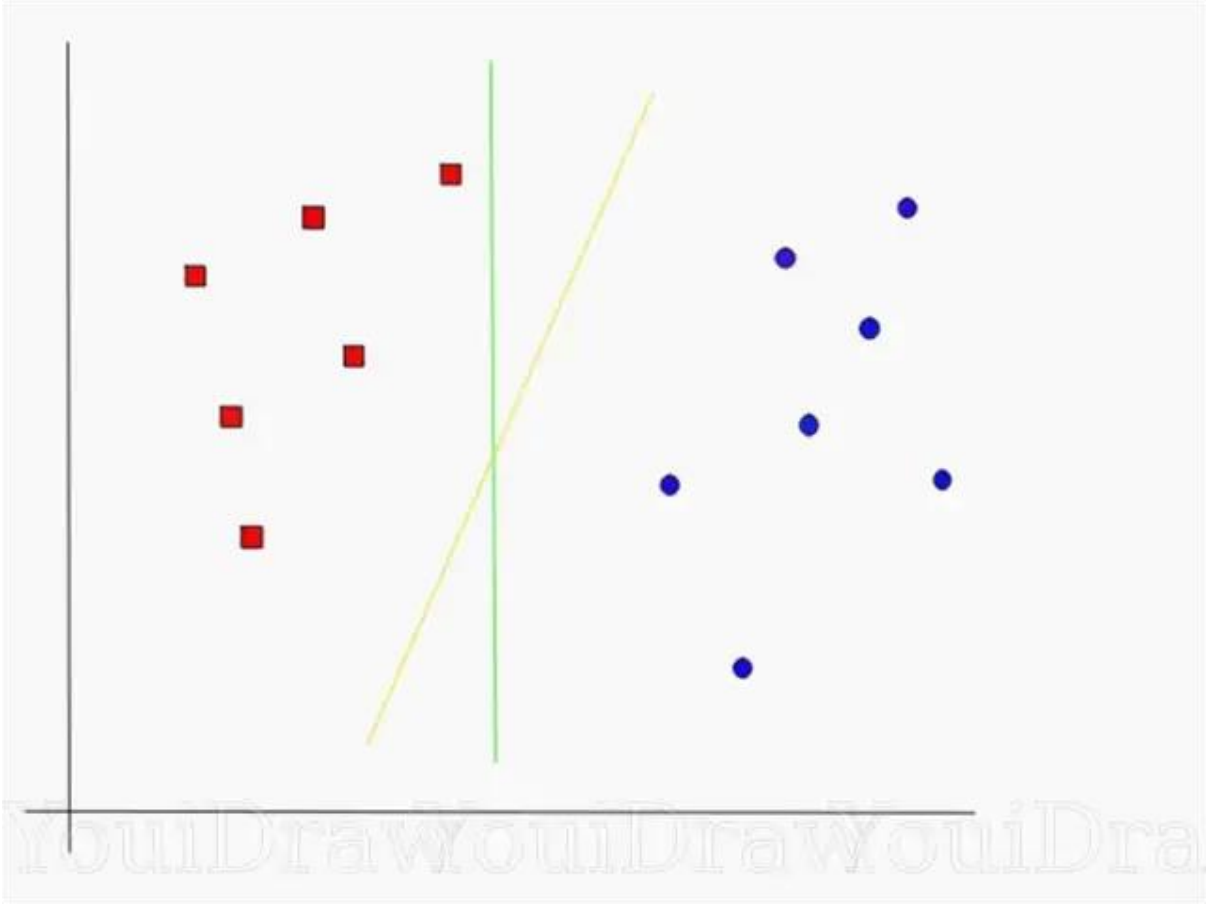
First we have taken a training data and trained the model from a single source and we transform that data and we process the data and pass that data into a machine learning model which uses SVM (SUPPORT VECTOR MACHINE) classifier, now when a user provides some input i.e a retinal image of a ultrasonographic scan the trained model will extract the features of the passed image and finds the similarities to the feature points which are extracted from the training data set.

If any features matched then the algorithm predicts the presence of the disease.

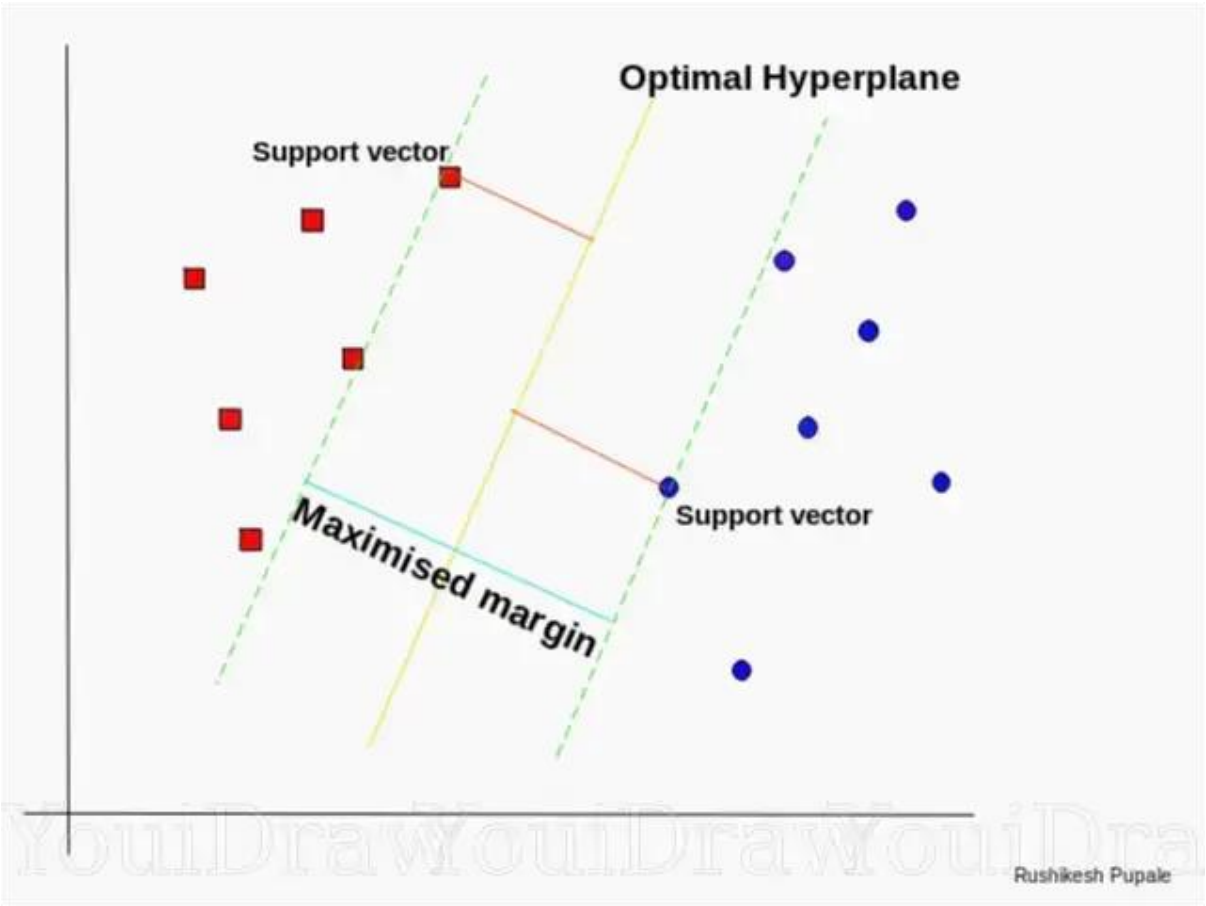
6) IMPLEMENTATION

MODULE DESCRIPTION

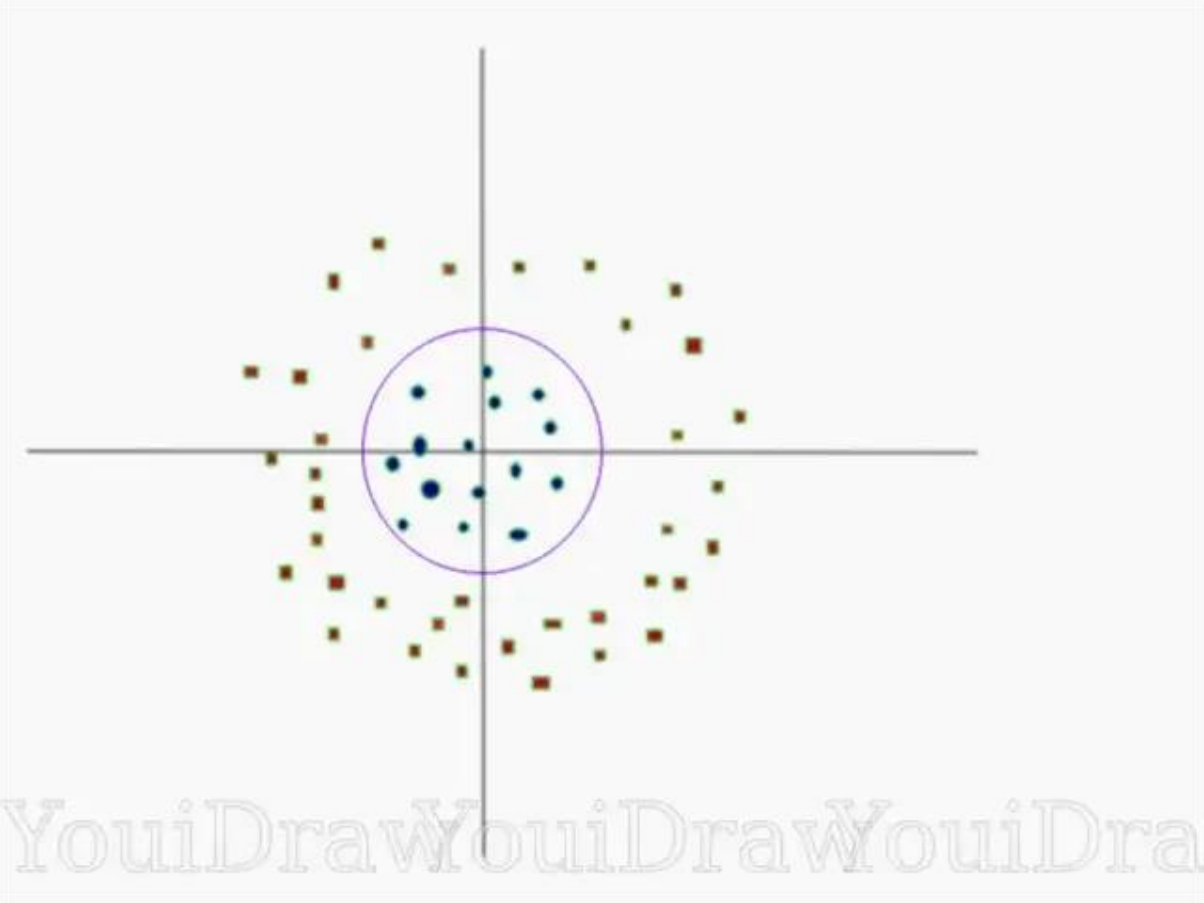
The following model we are building is to predict the diabetic retinopathy in the early stages. Here we use SVM(Support Vector machine) to categorize the features of the processed image and categorize them based on their matched features.



The algorithm creates a line or a hyperplane which separates the data into classes. At first approximation what SVMs do is to find a separating line(or hyperplane) between data of two classes.



SVM is an algorithm that takes the data as an input and outputs a line that separates those classes if possible.



According to the SVM algorithm we find the points closest to the line from both the classes. These points are called support vectors. This distance is called the margin. Our goal is to maximize the margin. The hyperplane for which the margin is maximum is the optimal hyperplane.

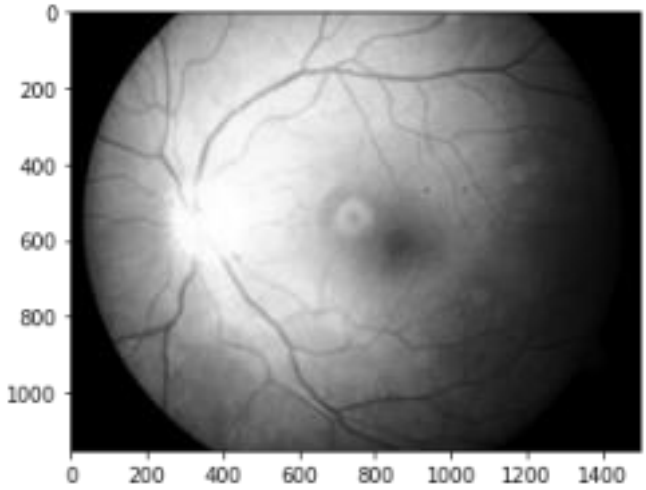
First we take the input from training data set and process the image by converting the following images into grayscale images and normalize the descriptors of the images and get the black and white image of the processed image.

Now the processed images are converted into binary format and stored in 2D matrix. By the application of max pooling the images are scaled down. Now we pass these binary images to the SVM classifier which extracts the features from the passed images.

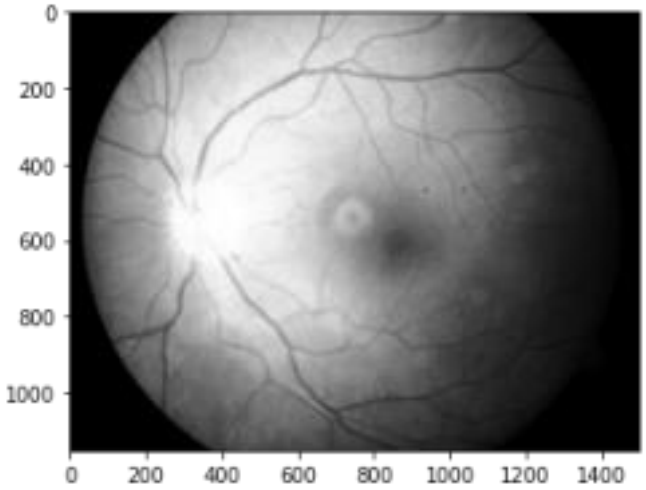
SAMPLE CODE:

```
from scipy import misc
from PIL import Image
from skimage import exposure
from sklearn import svm

import scipy
from math import sqrt,pi
from numpy import exp
from matplotlib import pyplot as plt
import numpy as np
import glob
import matplotlib.pyplot as pltss
import cv2
from matplotlib import cm
import pandas as pd
from math import pi, sqrt
import pywt

#img_rows=img_cols=200
immatrix=[]
im_unpre = []
#image_path =
Image.open('C:\Users\Rohan\Desktop\Diabetic_Retinopathy\diaretdb1_v_1
#image = misc.imread(image_path)

for i in range(1,90):
img_pt = r'C:\Users\G Harsh
Vardhan\Desktop\dataset\resources\images\ddb1_fundu
if i < 10:
       img_pt = img_pt + "00" + str(i) + ".png"
else:
       img_pt = img_pt + "0" + str(i)+ ".png"

       img = cv2.imread(img_pt)
       #im_unpre.append(np.array(img).flatten())
       img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
       equ = cv2.equalizeHist(img_gray)
       immatrix.append(np.array(equ).flatten())
       #res = np.hstack((img_gray,equ))

       np.shape(np.array(equ).flatten())



         np.shape(immatrix)
         np.shape(equ)
```

```
plt.imshow(immatrix[78].reshape((1152,1
500)),cmap='gray') plt.show()
```



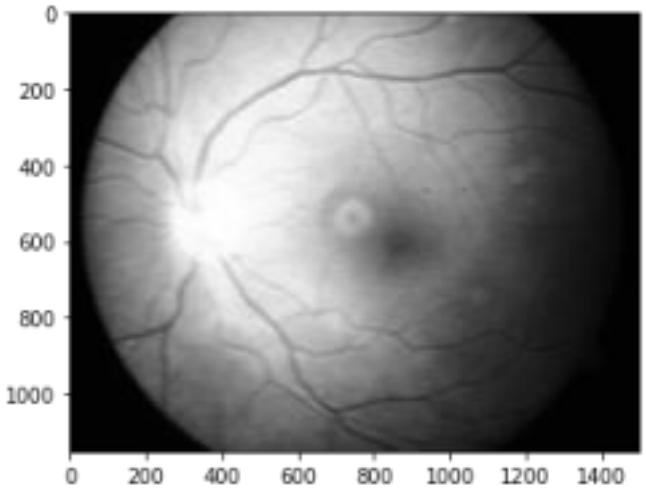

```
imm_dwt = []
for equ in immatrix:
 equ = equ.reshape((1152,1500))
 coeffs = pywt.dwt2(equ, 'haar')
 equ2 = pywt.idwt2(coeffs, 'haar')
 imm_dwt.append(np.array(equ2).flatten())

np.shape(imm_dwt)
np.shape(equ2)
plt.imshow(imm_dwt[78].reshape((1152,1500)),cmap='gray')
plt.show()
```



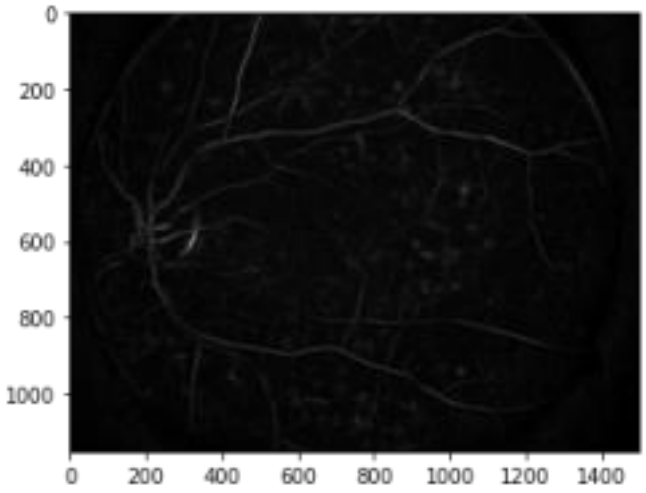

```
def _filter_kernel_mf_fdog(L, sigma, t = 3, mf = True):
 dim_y = int(L)
 dim_x = 2 * int(t * sigma)
 arr = np.zeros((dim_y, dim_x), 'f')

 ctr_x = dim_x / 2
 ctr_y = int(dim_y / 2.)
```

```
 # an un-natural way to set elements of the array
 # to their x coordinate.
 # x's are actually columns, so the first dimension of the
iterator is used  it = np.nditer(arr,
flags=['multi_index'])
 while not it.finished:
     arr[it.multi_index] = it.multi_index[1] - ctr_x
 it.iternext()

 two_sigma_sq = 2 * sigma * sigma


 sqrt_w_pi_sigma = 1. / (sqrt(2 * pi) * sigma)
 if not mf:
     sqrt_w_pi_sigma = sqrt_w_pi_sigma / sigma ** 2

 #@vectorize(['float32(float32)'], target='cpu')
 def k_fun(x):
     return sqrt_w_pi_sigma * exp(-x * x / two_sigma_sq)

 #@vectorize(['float32(float32)'], target='cpu')
 def k_fun_derivative(x):
            return -x * sqrt_w_pi_sigma * exp(-x * x /
two_sigma_sq)

 if mf:
     kernel = k_fun(arr)
     kernel = kernel - kernel.mean()
 else:
     kernel = k_fun_derivative(arr)

 # return the "convolution" kernel for filter2D
 return cv2.flip(kernel, -1)

def show_images(images,titles=None, scale=1.3):
 """Display a list of images"""
 n_ims = len(images)
 if titles is None:
     titles = ['(%d)' % i for i in range(1,n_ims +
     1)]  fig = plt.figure()
     n = 1
 for image,title in zip(images,titles):
     a = fig.add_subplot(1,n_ims,n) # Make subplot
 if image.ndim == 2: # Is image grayscale?
     plt.imshow(image, cmap = cm.Greys_r)
 else:
     plt.imshow(cv2.cvtColor(image, cv2.COLOR_RGB2BGR))
     a.set_title(title)
     plt.axis("off")
     n += 1
     fig.set_size_inches(np.array(fig.get_size_inches(),
dtype=np.float) * n_ims / s  plt.show()
```

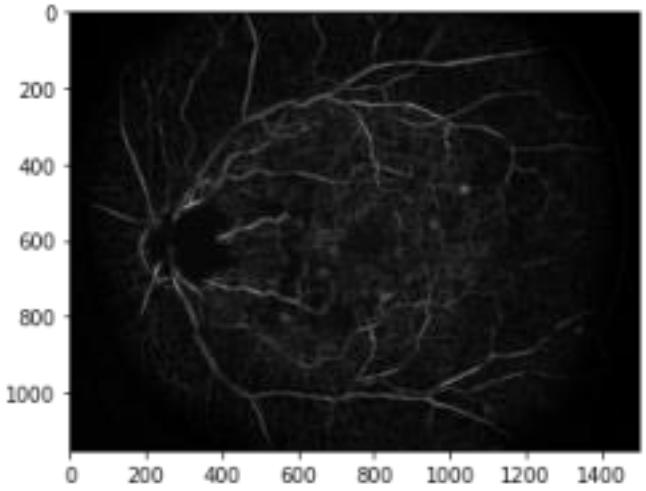
```
def gaussian_matched_filter_kernel(L, sigma, t = 3):
 '''
 K = 1/(sqrt(2 * pi) * sigma ) * exp(-x^2/2sigma^2), |y| <=
L/2, |x| < s * t  '''
 return _filter_kernel_mf_fdog(L, sigma, t, True)

#Creating a matched filter bank using the kernel generated from
the above functions def createMatchedFilterBank(K, n = 12):
 rotate = 180 / n
 center = (K.shape[1] / 2, K.shape[0] / 2)
 cur_rot = 0
 kernels = [K]

 for i in range(1, n):
 cur_rot += rotate
 r_mat = cv2.getRotationMatrix2D(center, cur_rot, 1)
 k = cv2.warpAffine(K, r_mat, (K.shape[1], K.shape[0]))
 kernels.append(k)

 return kernels
#Given a filter bank, apply them and record maximum response

def applyFilters(im, kernels):

 images = np.array([cv2.filter2D(im, -1, k) for k in kernels])
 return np.max(images, 0)


gf = gaussian_matched_filter_kernel(20, 5)
bank_gf = createMatchedFilterBank(gf, 4)

imm_gauss = []
for equ2 in imm_dwt:
 equ2 = equ2.reshape((1152,1500))
 equ3 = applyFilters(equ2,bank_gf)
 imm_gauss.append(np.array(equ3).flatten())

# the array ranges from 0 - 89
np.shape(imm_gauss)
plt.imshow(imm_gauss[78].reshape((1152,1500)),cmap='gray')
plt.show()
```




```
def createMatchedFilterBank():
 filters = []
 ksize = 31
 for theta in np.arange(0, np.pi, np.pi / 16):
 kern = cv2.getGaborKernel((ksize, ksize), 6, theta,12, 0.37, 0,
ktype=cv2.C  kern /= 1.5*kern.sum()
 filters.append(kern)
 return filters

def applyFilters(im, kernels):
 images = np.array([cv2.filter2D(im, -1, k) for k in kernels])
 return np.max(images, 0)

bank_gf = createMatchedFilterBank()
#equx=equ3
#equ3 = applyFilters(equ2,bank_gf)
imm_gauss2 = []
for equ2 in imm_dwt:
 equ2 = equ2.reshape((1152,1500))
 equ3 = applyFilters(equ2,bank_gf)
 imm_gauss2.append(np.array(equ3).flatten())

# the array ranges from 0 - 89
np.shape(imm_gauss2)
plt.imshow(imm_gauss2[20].reshape((1152,1
500)),cmap='gray') plt.show()
```

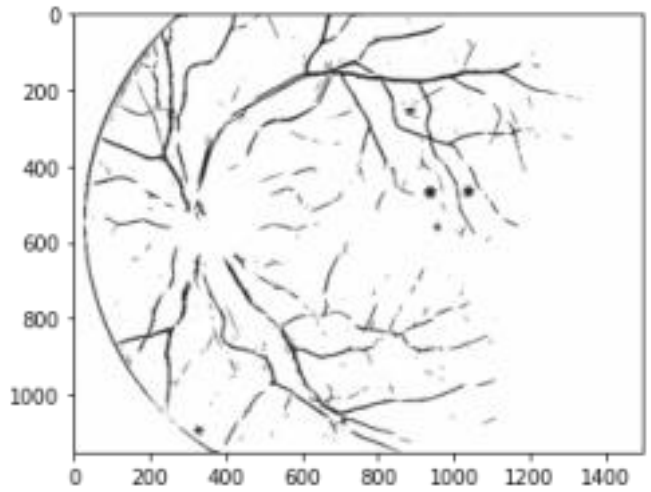



```
# the array ranges from 0 - 89
  np.shape(imm_gauss2)
  plt.imshow(imm_gauss2[1].reshape((1152,1
  500)),cmap='gray') plt.show()
```




```
e_ = equ3
  np.shape(e_)
  e_=e_.reshape((-1,3))
  np.shape(e_)


  img = equ3
  Z = img.reshape((-1,3))

  # convert to np.float32
  Z = np.float32(Z)

  k=cv2.KMEANS_PP_CENTERS


  # define criteria, number of clusters(K) and apply kmeans()
  criteria = (cv2.TERM_CRITERIA_EPS +
  cv2.TERM_CRITERIA_MAX_ITER, 10, 1.0) K = 2
  ret,label,center=cv2.kmeans(Z,K,None,criteria,10,k)
```

```
  # Now convert back into uint8, and make original image
  center = np.uint8(center)
  res = center[label.flatten()]
  res2 = res.reshape((img.shape))

imm_kmean = []
  for equ3 in imm_gauss2:
   img = equ3.reshape((1152,1500))
   Z = img.reshape((-1,3))

   # convert to np.float32
   Z = np.float32(Z)

   k=cv2.KMEANS_PP_CENTERS


   # define criteria, number of clusters(K) and apply kmeans()
   criteria = (cv2.TERM_CRITERIA_EPS +
  cv2.TERM_CRITERIA_MAX_ITER, 10, 1.0)  K = 2
   ret,label,center=cv2.kmeans(Z,K,None,criteria,10,k)

   # Now convert back into uint8, and make original image
   center = np.uint8(center)
   res = center[label.flatten()]
   res2 = res.reshape((img.shape))
   imm_kmean.append(np.array(res2).flatten())

# the array ranges from 0 - 89
  np.shape(imm_kmean)
  plt.imshow(imm_kmean[78].reshape((1152,1500)),cmap="gray")
  plt.show()
```




```
from sklearn.svm import SVC
  clf = SVC()

Y = np.ones(89)

Y[1]=Y[5]=Y[7]=Y[17]=Y[6]=0
```

```
            clf.fit(imm_kmean, Y)

            pred = clf.predict(imm_kmean)

           k=[1,3,4,9,10,11,13,14,20,22,24,25,26,27,28,29,35,36,38,42,53,55,57
      ,64,70,79,84,8


        k = k-np.ones(len(k))

        k
```

```
      array([ 0., 2., 3., 8., 9., 10., 12., 13., 19., 21., 23., 24., 25.,  26.,
27., 28., 34., 35., 37., 41., 52., 54., 56., 63., 69., 78.,  83., 85.])
```

```
      k =[int(x) for x in k]

      k
```

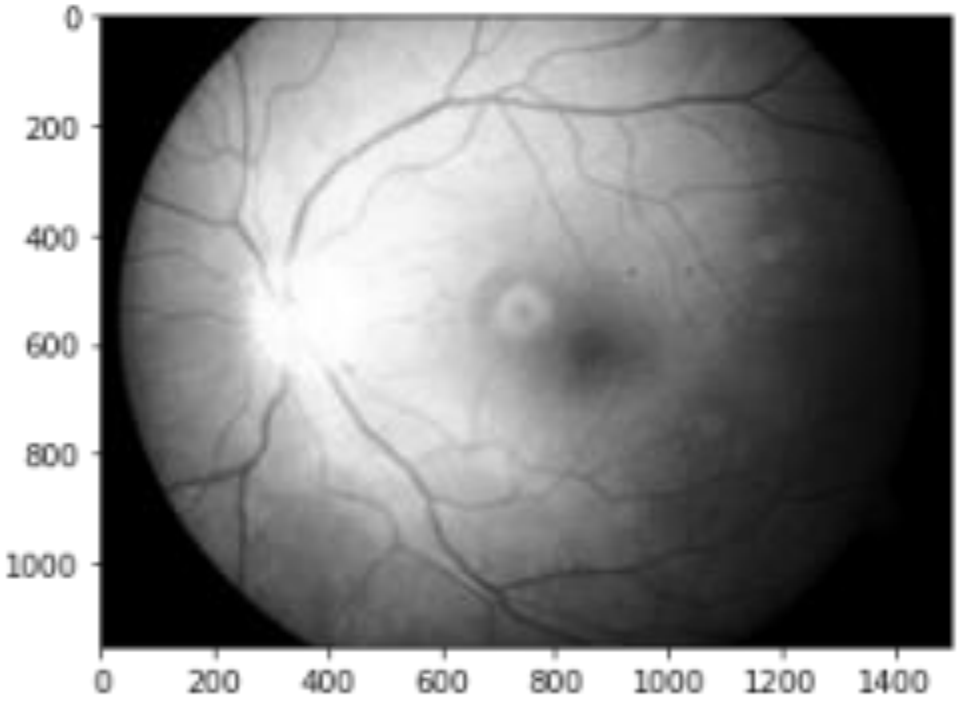
```
imm_train = []

y_train = []
k.append(5)
k.append(7)
for i in k:
 imm_train.append(imm_kmean[i])
  y_train.append(Y[i])


y_train
clf.fit(imm_train, y_train)

y_pred=clf.predict(immkmean)
import sklearn
sklearn.metrics.accuracy_score(Y,y_pred)

0.9438202247191011
```

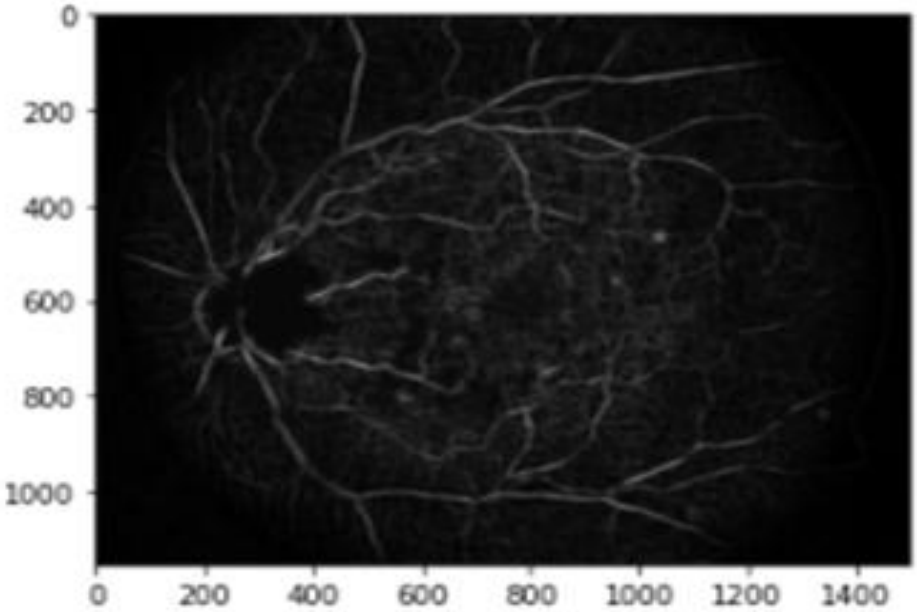
7) TESTING

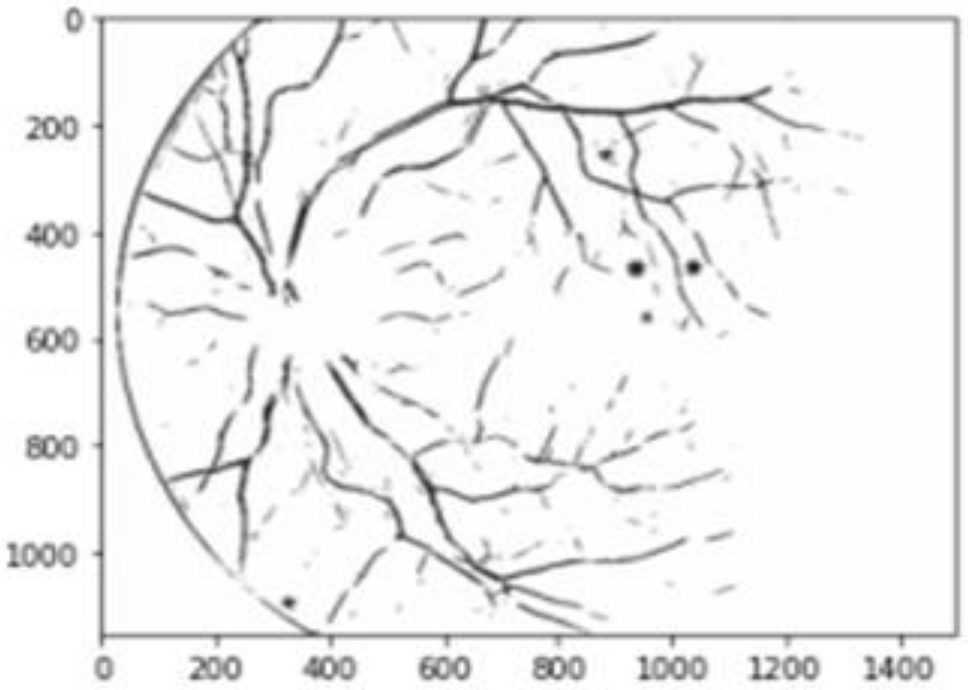
Testing is a procedure that identifies programming errors. It is the primary quality control tool used by software developers. During testing, a series of test cases are used to execute the program , and the output for each test case is assessed to see if the

program is functioning as intended. Software testing is an essential part of software quality assurance since it provides the last analysis of the specification, design, and code. The increasing prominence of software as a system component and the costs associated with a software failure served as our motivation to prepare using testing. Running a program with the intention of finding any faults is how a program is tested. Making tests for software and other manufactured goods can be as challenging as the initial design of the product itself.

8) RESULTS


0
200
400
600
800
1000
0
200
400
600
800
1000
1200
1400


0
200
400
600
800
1000
0
200
400
600
800
1000
1200
1400



9) CONCLUSION

Last but not least, we'll note that this project, Early prediction Diabetic Retinopathy, is very beneficial to everyone's day-to-day life and that it is particularly significant for the healthcare sector because it is they who use these systems daily to forecast all kinds of diseases of the patients based on their general information and symptoms that they have experienced. Currently, the health sector plays a significant role in treating patients' diseases, so this system is also helpful for the sector to inform the user and is beneficial to the user in the event that he or she chooses not to visit a hospital or other clinic. By entering the symptoms and other pertinent information, the user can learn what disease they are currently experiencing, and the sector can also gain from this system by simply asking the user. If

the health industry embraces this idea, then doctors' workloads particularly those of eye doctors may be decreased and they can quickly diagnose patients' illnesses. The goal of the illness prediction is to offer forecasts for the different and frequently occurring eye disorders that, if left untreated or sometimes even neglected, can become lethal, create severe problems, and even render the patient completely blind.

REFERENCES

1) https://towardsdatascience.com/https-medium-com-pupalerushikesh-svm-f4b42800e98910)

2) https://www.researchgate.net/publication/349657050_Performance_Analysis_of_Diabetic_Retinopathy_Prediction_using_Machine_Learning_Models

3) Clinical Manifestations of Supra-Large Range Nonperfusion Area in Diabetic Retinopathy (hindawi.com)

4) A Machine Learning Ensemble Classifier for Early Prediction of Diabetic Retinopathy - PubMed (nih.gov)

5) Early prediction of diabetic retinopathy by mpv and p-lcr (researchgate.net)