Project Name - Anemia Sense: Leveraging Machine Learning For Precise Anemia Recognitions

Team ID: SWTID1720080033

Final Project Report

1.Introduction

1.1 Project Overview

Millions of people throughout the world suffer from anaemia, this is characterized as the lack of red blood cells or haemoglobin. If left untreated, anaemia can have major negative effects on health. In order to help healthcare providers identify anaemia and enable earlier intervention and better patient outcomes, the project intends to create a dependable, effective, and easily available diagnostic tool by utilising machine learning.

1.2 Objectives

The principal aim of this project is to create a machine learning model that can precisely identify anaemia in patients by utilising medical data.

Scope of the Project:

- Data Collection from Kaggle:

Make use of a dataset that contains information on gender, haemoglobin levels, MCHC, MCV, and test results.

- Data preprocessing and cleaning:

To deal with missing values, remove outliers, and guarantee consistency, clean up the data.If needed, normalise or standardise the data to get it ready for analysis.

- Model Building:

Investigate and put into practice different machine learning methods, including Support Vector Machines, Random forests, Logistic Regression, Decision Trees, and Naïve Bayes.

- Validation and Training of Models:

Utilise a dataset to train the chosen model. To maximise performance, cross-validate the models and adjust the hyper-parameters, if needed.

- Web Development and Deployment:

Create an intuitive user interface using Flask frameworks.Make sure the interface enables efficient patient data entry and diagnostic result delivery for healthcare practitioners.

Anticipated Advantages:

- Early Detection
- Enhanced Accessibility
- Efficiency
- Scalability

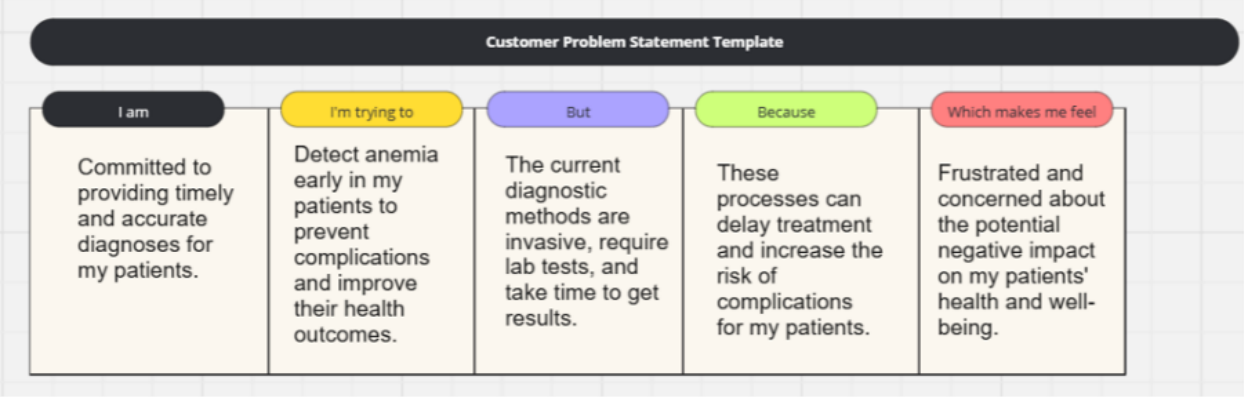
2. Project Initialization and Planning Phase

2.1. Define Problem Statement

Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
I am a patient who frequently experience fatigue and weakness .	Worried if these symptoms might be due to anemia.	Get an accurate diagnosis to understand what's going on in my body and start treatments if required.	Getting an appointment with the doctor, getting the tests done and waiting for results would take more time.	I have limited access to healthcare facilities since a I stay far away from such locations.	Frustrated and worried about the health issues I might get if not checked and diagnosed well in advance
I am a product manager at a healthcare technology company.	Focused on developing medicine-related innovations that could improve diagnosis and accuracy.	Create an efficient and reliable tool that can sense anemia and can be easily adopted in healthcare providing centres.	Present solutions are either costly or inefficient	There is a need for an accessible and accurate tool that can diagnose and be used in various health care settings.	Challenged by the technical and market demands, but working with the aim to develop a solution that could essentially solve the problems

Problem Statement:

I am a primary care physician in a community health clinic.



2.2. Project Proposal (Proposed Solution)

Project Overview	
Objective	This project's main goal is to create a machine learning model that, given certain parameters and medical data, can correctly diagnose anaemia in patients. Themodel aims to assist healthcare professionalsin early diagnosis and management of anemia by providing a reliable and efficient tool for analysing patients.
Scope	● Collect blood testdata from reliable sources such as Kaggle. ● Pre-Processingand Data cleaning ● Identify and selectthe most relevant features that contribute toanemia detection. ● Explore and select suitable machine learning algorithms for classification ● Test the modelwith a separate validation dataset to ensure it generalizes well to unseen data. ● Develop a user-friendly web application for healthcare professionals to input patient data and receive anemia diagnosis results. ● Maintain comprehensive documentation throughout the project, including data sources, preprocessing steps, model development, evaluation metrics, and deployment procedures. **Boundaries:** ● The project will focus solely on anemia detection and will notextend to diagnosing other blood disorders. ● The model will use a predefined set of blood parameters andwill not incorporate additional tests or patient history. ● The deployment will be limited to a software tool without hardware integration.

Problem Statement	
Description	Anemia is a common blood disorder characterized by a deficiency of red blood cellsor hemoglobin, leading to reduced oxygentransport inthe body. It affects millions of people worldwide, causing fatigue, weakness, and other health complications. Early detection and diagnosis of anemia are crucial for effective management and treatment. Traditional diagnostic methods often require multiple laboratory tests and can be time-consuming. With machine learning, there is an opportunity to develop an automated, efficient, and accurate tool to assist healthcare professionals in diagnosing anemia usingreadily available blood parameters.
Impact	● A reliable andaccurate machine learning model for anemiadetection. ● Improve patient diagnosis and management of anemia. ● Streamline the process of detecting anemia by reducing dependency on traditional medical practices. ● An easy-to-use interface for healthcare professionals. ● Lower overall healthcare costs associated withmanaginganemia-related complications.
Proposed Solution	
Approach	● Obtain data fromKaggle and other reliable sources ● Handle imbalanced data to createa robust model which is notbiased. ● Visualize the data in order to find correlation among thefeatures. ● Evaluate various algorithms suitable for the task (Linear Regression, decision trees, random forests, Naïve Bayes, support vector machines, Gradient Boosting). ● Assess and choose the best model in-order to provide the most accurate results. ● Create a webapplication using flaskand integrate themodel.
Key Features	● Utilizes machine learning algorithms to accurately detect anemia basedon comprehensive analysis of blood parameters.

	● Identifies and leverages crucial blood parameters (e.g., hemoglobin levels) specifically relevant to anemia detection. ● Provides detailed reports and visualizations to users, communicating findings, methodology, and insightseffectively. ● Web Application allowing users to themselves to enter theirdata and check if they are anemic or not.

2.2. Project Proposal (Proposed Solution)

Resource Requirements

Resource Type	Description	Specification/Allocation
Hardware		
Computing Resources	CPU specifications, numberof cores	Intel(R) Core(TM) i5-10210UCPU @ 1.60GHz, 2112 Mhz, 4 Core(s), 8
Memory	RAM specifications	8 GB
Storage	Disk spacefor data, models,and logs	500 GB SSD
Software		
Frameworks	Python frameworks	Flask
Libraries	Additional libraries	sklearn, pandas, numpy, pickle,
Development Environment	IDE, version control	Jupyter Notebook, Git
Data		
Data	Kaggle, 34KB,CSV	Kaggle dataset – 1421dataentries

2.3. Initial Project Planning

Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members	Sprint Start Date	Sprint End Date (Planned)
Data Collection and Preprocessing	USN-1	Collect and clean datasets	1,2	Medium	Manas Gunti Ashish David John Rajashri S Vishnu Adithya C	8th July,2024	8th July,2024
Model Development and valuation	USN-2	Develop and train ML models	1,2	High	Manas Gunti Ashish David John Rajashri S Vishnu Adithya C	8th July,2024	8th July,2024
Model selection and tuning	USN-3	Evaluate model performance to validate and tune the best model	1,2	High	Manas Gunti Ashish David John Rajashri S Vishnu Adithya C	8th July,2024	8th July,2024
Frontend development	USN-4	Integrate model predictions into frontend	1,2	Medium	Manas Gunti Ashish David John Rajashri S Vishnu Adithya C	9th July,2024	9th July,2024
Deployment using Flask	USN-5	Deploy model using Flask	1,2	Medium	Manas Gunti Ashish David John Rajashri S Vishnu Adithya C	9th July,2024	9th July,2024
Final Report	USN-6	Compile the final report	-	Medium	Manas Gunti Ashish David John Rajashri S Vishnu Adithya C	10th July,2024	10th July,2024
Video Demonstration	USN-7	A video demonstrating the project from the beginning till the end.	-	Medium	Manas Gunti Ashish David John Rajashri S Vishnu Adithya C	11th July,2024	11th July,2024

3. Data Collection and Preprocessing Phase

3.1 Data Collection Plan and Raw Data Sources Identified

Data Collection Plan:

Section	Description
Project Overview	The Anemia Sense project is a model designed to leverage machine learning algorithms to accurately predict cases of anemia, a condition characterized by the deficiency of RBCs or hemoglobin inthe blood, when given a set of relevant data. By predicting such a condition, diagnosis, treatment plans andoverall monitoring canbecome more effective and efficient.
Data Collection Plan	The Datawill be collected fromkaggle.com.
Raw Data Sources Identified	Anemia.csv – kaggle dataset that holds fields such as Gender, Hemoglobin, MCH, MCHC, MCV and dependent field Result

Raw Data Sources:

Souce Name	Description	Location/URL	Format	Size	Access Permissions
Dataset 1 (Anemia.csv)	The kaggle dataset on anemia patients containing relevant data points and correct classifications.	https://drive.google.com/file/d/1KMJFNFGwoaQoAouIPabMEHcT1bvqEXau/view?usp=sharing	CSV	34 KB	Public

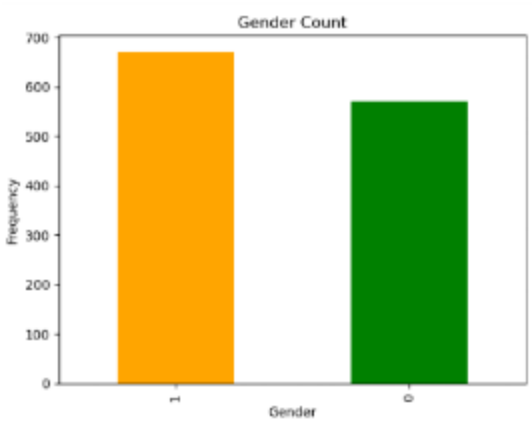
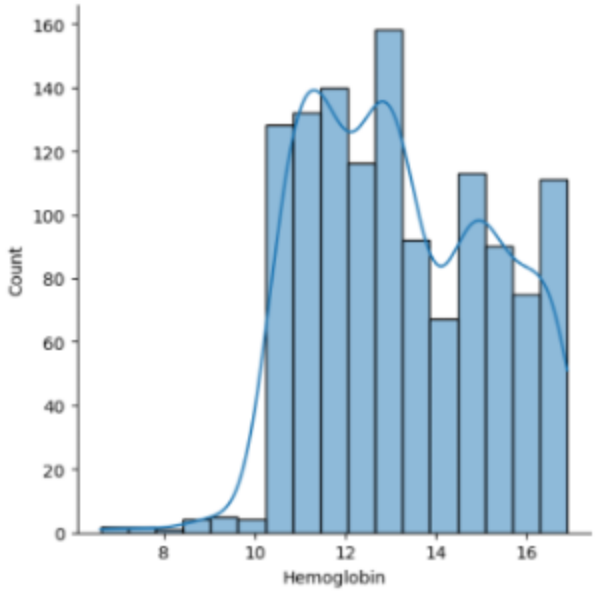
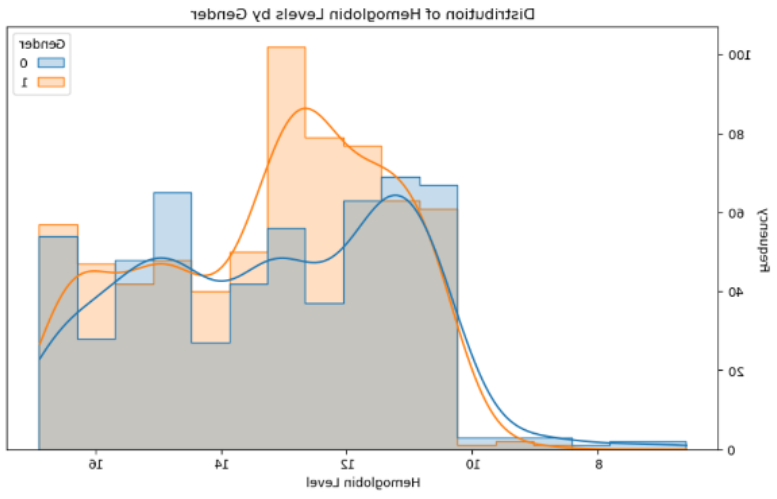
3.2 Data Quality Report

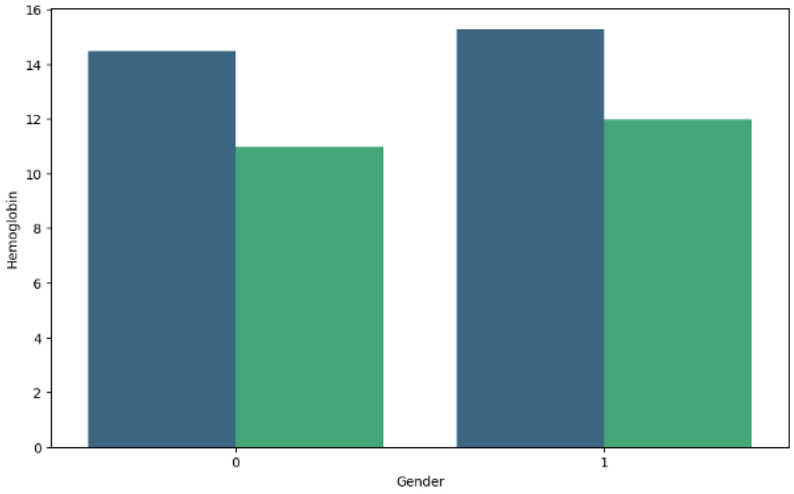
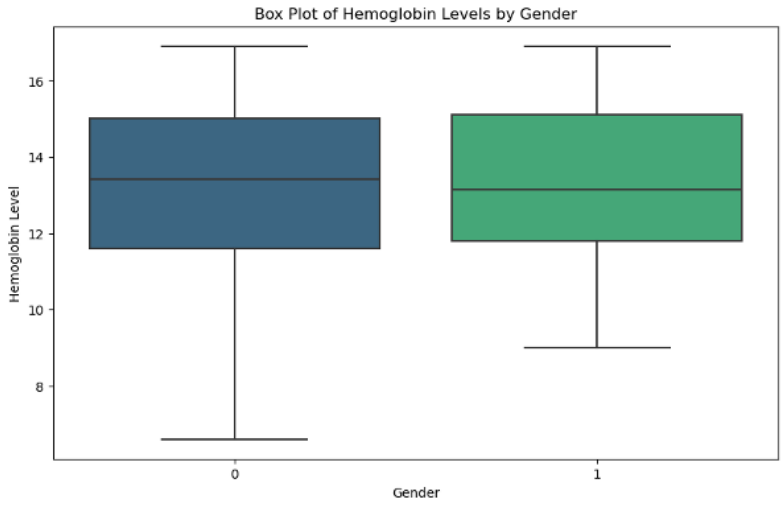
Data Souce	Data Quality Issue	Severity	Resolution Plan
Dataset provided in skill wallet	There were no nullvalues or anyoutliers in the dataset. However,the dataset had unequal gender data entries. This difference in the data of the male and female gender may cause biased results.	Moderate	Handling the imbalanced valuesby undersampling the majority class to be equal to the minorityclass.

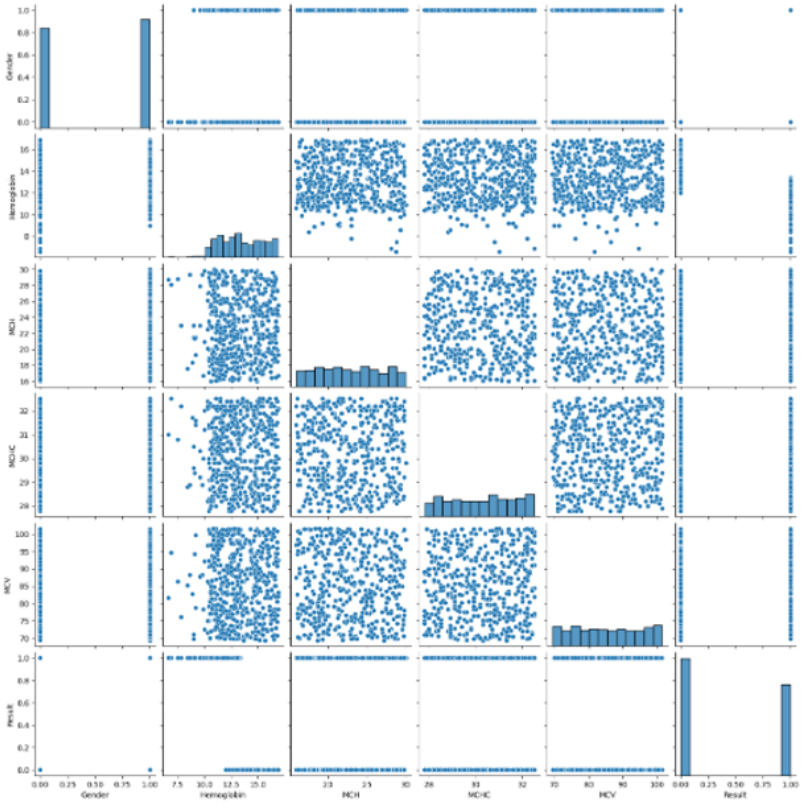
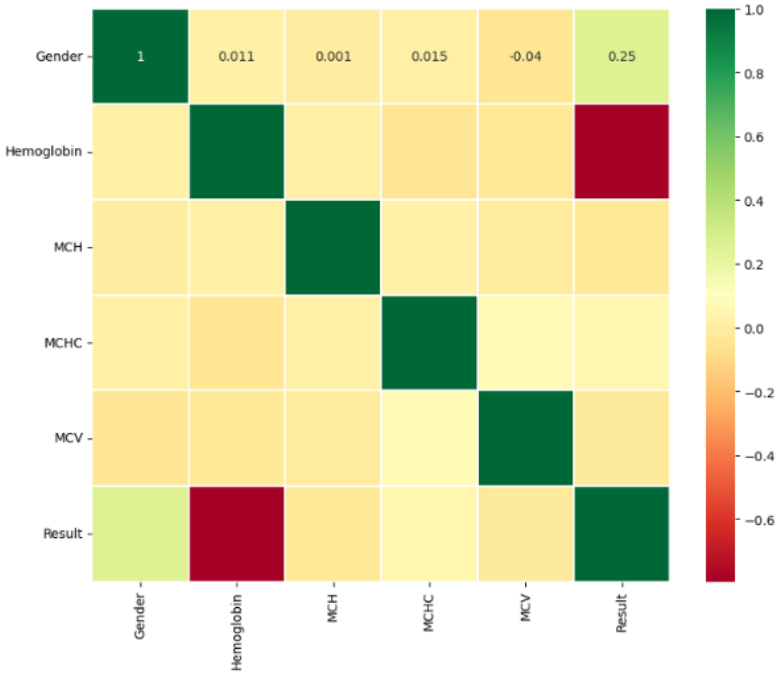
3.3 Data Exploration and Preprocessing

Section	Description
Data Overview	

	Gender	Hemoglobin	MCH	MCHC	MCV	Result
count	1240.000000	1240.000000	1240.000000	1240.000000	1240.000000	1240.000000
mean	0.540323	13.218145	22.903952	30.277984	85.620968	0.500000
std	0.498573	1.976190	3.993624	1.394515	9.673794	0.500202
min	0.000000	6.600000	16.000000	27.800000	69.400000	0.000000
25%	0.000000	11.500000	19.400000	29.100000	77.300000	0.000000
50%	1.000000	13.000000	22.700000	30.400000	85.300000	0.500000
75%	1.000000	14.900000	26.200000	31.500000	94.225000	1.000000
max	1.000000	16.900000	30.000000	32.500000	101.600000	1.000000

Univariate Analysis	  

Bivariate Analysis	 

Multivariate Analysis	 

Outliers and Anomalies	NIL

Data Preprocessing Code Screenshots	
Loading Data	df=pd.read_csv('anemia.csv') `[2]: df=pd.read_csv('anemia.csv')`

Handling Missing Data	df.isnull().sum()
	`[8]: df.isnull().sum()`
	`[8]: Gender 0`
	`Hemoglobin 0`
	`MCH 0`
	`MCHC 0`
	`MCV 0`
	`Result 0`
	`dtype: int64`

Data Transformation	NIL
Feature Engineering	NIL

Save Processed Data	NIL

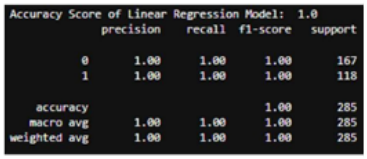
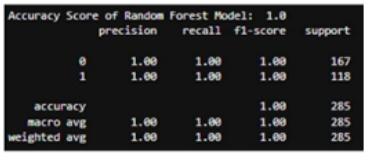
4.Model Development Phase

4.1 Feature Selection Report

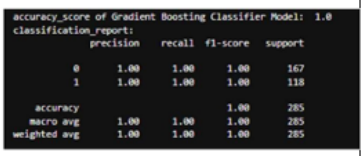
Feature	Description	Selected (Yes/No)	Reasoning
Hemoglobin	Hemoglobin is a protein in red blood cells that carries oxygen	Yes	Anemia is diagnosed whena blood test shows a hemoglobin valueof less than 13.5 gm/dlin a man or lessthan 12.0 gm/dl in a woman
MCH (Mean corpuscular hemoglobi n)	mean corpuscular hemoglobin, which is a measurement of the average amount of hemoglobin in red blood cells	Yes	High or low numbers may indicate a vitamin deficiency or certain typesof anemia.
MCHC (Mean corpuscular	A blood test that measures the average amount of hemoglobin in	Yes	The MCHC value is used to evaluate the severity and cause of anemia.
hemoglobi n concentrati on)	red blood cells (RBCs) in relation to the cell's volume, (Concentration)	Yes	Hemoglobin is a important indicator of anemia

MCV (Mean corpuscular volume)	blood test that measures the average size of red blood cells (RBCs) in a blood sample	Yes **Yes**	MCV blood test can help your healthcare provider determine if you have anemia, liver disease or other conditions.
Gender	Sex of the Patient	Yes	Measurement standards to diagnose anemia are gender-dependent.

4.2 Model Selection Report

Model	Description	Hyperparameters	Performance Metric (e.g., Accuracy, F1 Score)
Logistic Regression Model	Logistic regression is used for predicting the categorical dependent variable using a given set of independent variables. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.	None, as the accuracy of the model was 100% without hyperparameters	
Random Forest Model	Random Forest Model's algorithm is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset. It improves the accuracy and controls overfitting by combining multiple trees.	None, as the accuracy of the model was 100% without hyperparameters	

Decision Tree Model	Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.	None, as the accuracy of the model was 100% without hyperparameters	Accuracy Score of Decision Tree Model: 1.0 precision recall f1-score support 0 1.00 1.00 1.00 167 1 1.00 1.00 1.00 118 accuracy 1.00 285 macro avg 1.00 1.00 1.00 285 weighted avg 1.00 1.00 1.00 285
Naive Bayes Model	Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of conditional independence between every pair of features. It is used for classification tasks.	`param_grid = { 'var_smoothing': [1e-9, 1e-8, 1e-7, 1e-6, 1e-5] }` Accuracy Remained the same even after hyperparametric tuning	Accuracy Score of Naive Bayes Model: 0.9403508771929825 precision recall f1-score support 0 0.93 0.97 0.95 160 1 0.96 0.90 0.93 125 accuracy 0.94 285 macro avg 0.94 0.94 0.94 285 weighted avg 0.94 0.94 0.94 285
SVM Model	SVM is a powerful classification algorithm that works by finding the hyperplane that best separates the data into classes. It uses support vectors, the data points closest to the hyperplane, and aims to maximize the margin between the classes.	`param_grid = { 'C': [0.1, 1, 10, 100], 'kernel': ['linear', 'poly', 'rbf', 'sigmoid'], 'degree': [2, 3, 4], 'gamma': ['scale', 'auto'], 'coef0': [0, 0.5, 1] }`	Before Hyperparametric tuning accuracy_score of SVM Model: 0.9964912280701754 classification_report: precision recall f1-score support 0 1.00 0.99 1.00 167 1 0.99 1.00 1.00 118 accuracy 1.00 285 macro avg 1.00 1.00 1.00 285 weighted avg 1.00 1.00 1.00 285 After Hyperparametric tuning The Best hyperparameters for SVM: {'C': 100, 'coef0': 0, 'degree': 2, 'gamma': 'scale', 'kernel': 'linear'} accuracy_score of SVM: 1.0

Gradient Boosting Classifier Model	An ensemble technique where multiple weak models (typically decision trees) are built sequentially, with each new model attempting to correct errors made by its predecessors. It's known for its effectiveness in predictive modeling and achieving high accuracy.	None, as the accuracy of the model was 100% without hyperparameters	

4.3 Initial Model Training Code,Model Validation and Evaluation Report

Initial Model Training Code:

Splitting Data Into Train and Test

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(X,Y,test_size=0.2,random_state=20)
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

Logistic Regression

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
logistic_regression=LogisticRegression()
logistic_regression.fit(x_train,y_train)
pred=logistic_regression.predict(x_test)
pred
```

```
acc_lr=accuracy_score(y_test,pred)
confusion_matrix(y_test,pred)
acc_lr=accuracy_score(y_test,pred)
c_lr=classification_report(y_test,pred)
print('Accuracy Score: ',acc_lr)
print(c_lr)
```

Random Forest Model

```
from sklearn.ensemble import RandomForestClassifier
rf=RandomForestClassifier(n_estimators=10,criterion='entropy',random_state=0)
rf.fit(x_train,y_train)
pred=rf.predict(x_test)
pred
y_test
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
acc_rf=accuracy_score(y_test,pred)
conmat=confusion_matrix(y_test,pred)
print(acc_rf)
print(conmat)
c_rf=classification_report(y_test,pred)
c_rf
```

Decision tree

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
decision_tree_model = DecisionTreeClassifier()
decision_tree_model.fit(x_train, y_train)
y_pred = decision_tree_model.predict(x_test)
```

```
y_pred
y_test
acc_dt = accuracy_score(y_test, y_pred)
acc_dt
conmat = confusion_matrix(y_test, y_pred)
conmat
c_dt = classification_report(y_test, y_pred)
print(c_dt)
```

Naive Bayes Model

```
from sklearn.naive_bayes import GaussianNB
nb = GaussianNB()
nb.fit(x_train,y_train)
pred = nb.predict(x_test)
acc_nb=accuracy_score(pred,y_test)
c_nb=classification_report(pred,y_test)
print("Accuracy Score: ",acc_nb)
print(c_nb)
```

Support Vector Machine

```
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn import metrics
st_x= StandardScaler()
x_train= st_x.fit_transform(x_train)
x_test= st_x.transform(x_test)
support_vector=SVC()
```

```
support_vector.fit(x_train,y_train)
y_pred=support_vector.predict(x_test)
acc_svc=metrics.accuracy_score(y_test,y_pred)
c_svc=metrics.classification_report(y_test,y_pred)
print("accuracy_score: ",acc_svc)
print("classification_report: ")
print(c_svc)
```

Gradient Boosting Classifier

```
from sklearn.ensemble import GradientBoostingClassifier
GBC= GradientBoostingClassifier()
GBC.fit(x_train,y_train)
y_pred2=GBC.predict(x_test)
acc_gbc=metrics.accuracy_score(y_test,y_pred2)
c_gbc=metrics.classification_report(y_test,y_pred2)
print("accuracy_score: ",acc_gbc)
print("classification_report: ")
print(c_gbc)
```

Model Validation Report:

Model	Classification Report	Accuracy	Confusion Matrix
Logistic Regression	Accuracy Score: 1.0 precision recall f1-score support 0 1.00 1.00 1.00 167 1 1.00 1.00 1.00 118 accuracy 1.00 285 macro avg 1.00 1.00 1.00 285 weighted avg 1.00 1.00 1.00 285	1.0	array([[167, 0], [0, 118]], dtype=int64)
Random Forest Tree	c_lr=classification_report(y_test,pred) print(c_lr) precision recall f1-score support 0 0.97 0.93 0.95 167 1 0.90 0.96 0.93 118 accuracy 0.94 285 macro avg 0.94 0.94 0.94 285 weighted avg 0.94 0.94 0.94 285	1.0	array([[155, 12], [5, 113]], dtype=int64)
Decision Tree	precision recall f1-score support 0 1.00 1.00 1.00 167 1 1.00 1.00 1.00 118 accuracy 1.00 285 macro avg 1.00 1.00 1.00 285 weighted avg 1.00 1.00 1.00 285	1.0	array([[167, 0], [0, 118]], dtype=int64)
Naïve Bayes Model	precision recall f1-score support 0 0.93 0.97 0.95 160 1 0.96 0.90 0.93 125 accuracy 0.94 285 macro avg 0.94 0.94 0.94 285 weighted avg 0.94 0.94 0.94 285	0.940350877 1929825	array([[155, 12], [5, 113]], dtype=int64)

Support Vector Machine	classification_report: precision recall f1-score support 0 1.00 0.99 1.00 167 1 0.99 1.00 1.00 118 accuracy 1.00 285 macro avg 1.00 1.00 1.00 285 weighted avg 1.00 1.00 1.00 285	0.9964912280701754	conmat = confusion_matrix(y_test, pred) conmat array([[155, 12], [5, 113]], dtype=int64)
Gradient Boosting Classifier	classification_report: precision recall f1-score support 0 1.00 1.00 1.00 167 1 1.00 1.00 1.00 118 accuracy 1.00 285 macro avg 1.00 1.00 1.00 285 weighted avg 1.00 1.00 1.00 285	1.0	conmat = confusion_matrix(y_test, y_pred2) conmat array([[167, 0], [0, 118]], dtype=int64)

5.Model Optimization and Tuning Phase:

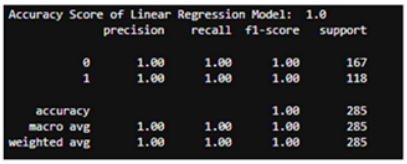
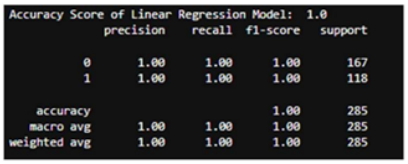
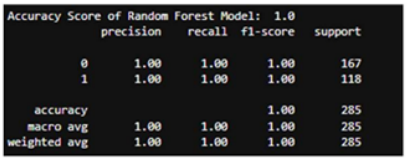
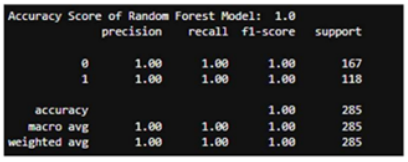
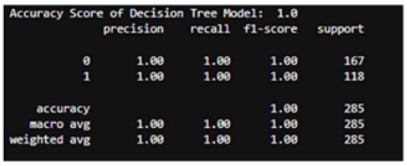
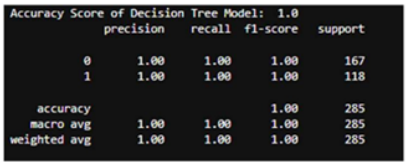
5.1 Hyperparameter Tuning Phase

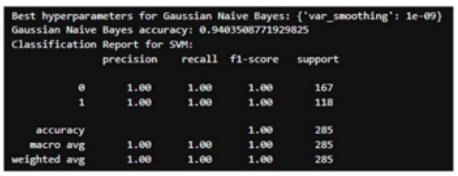
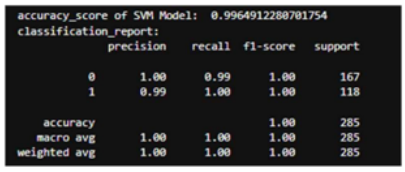
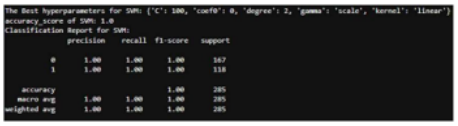
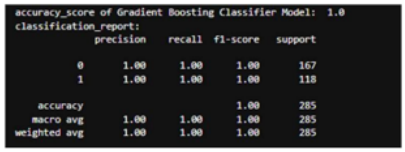
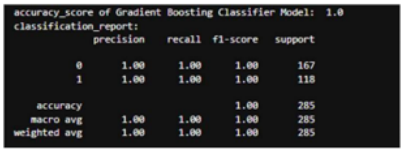
Model	Tuned Hyperparameters	Optimal Values
Logistic Regression Model	None, as the accuracy of the model was 100% without hyperparameter tuning.	Default Values
Random Forest Model	None, as the accuracy of the model was 100% without hyperparameter tuning.	Default Values
Decision Tree Model	None, as the accuracy of the model was 100% without hyperparameter tuning.	Default Values

Naive Bayes Model	param_grid = {
	'var_smoothing': [1e-9, 1e-8, 1e-7, 1e-6, 1e-5]
	}
		Best hyperparameters for Gaussian Naive Bayes: {'var_smoothing': 1e-09}

SVM Model		The Best hyperparameters for SVM: {'C': 100, 'coef0': 0, 'degree': 2, 'gamma': 'scale', 'kernel': 'linear'}
Gradient Boosting Classifier Model	None, as the accuracy of the model was 100% without hyperparameter tuning.	Default Values

5.2 Performance Metrics and Comparison Report

Model	Baseline Metric	Optimized Metric
Logistic Regression Model		
Random Forest Model		
Decision Tree Model		

Naive Bayes		
SVM		
Gradient Boosting Classifier		

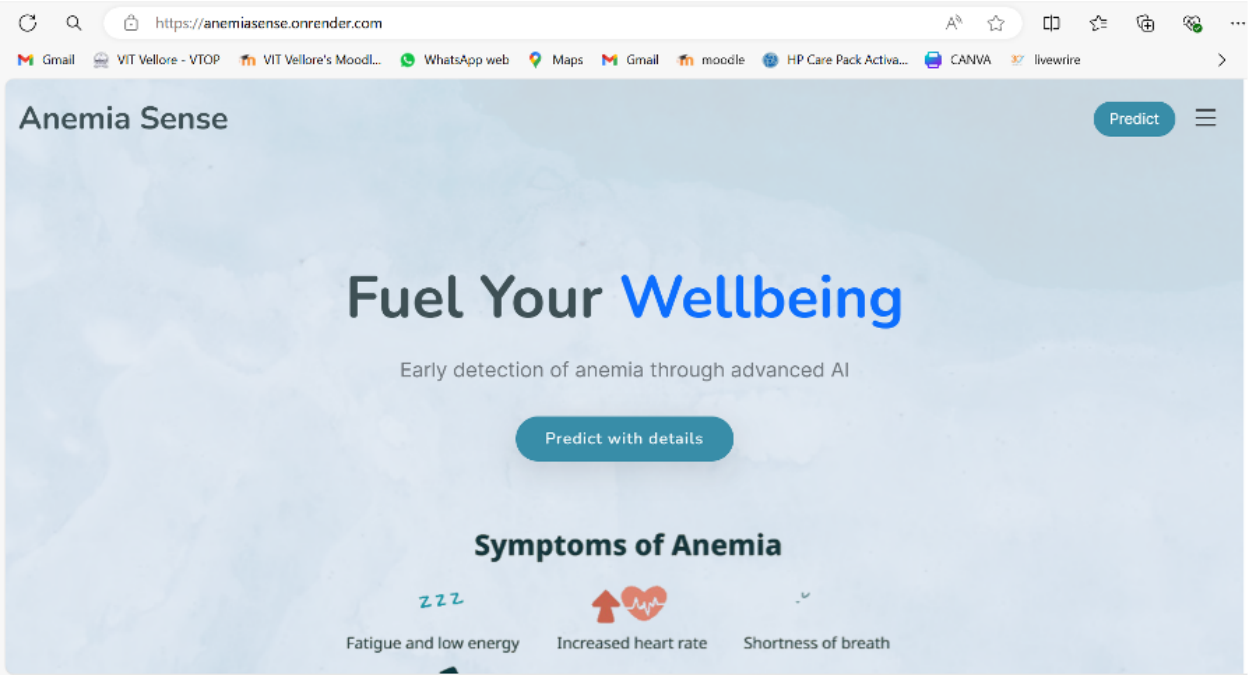
5.3 Final Model Selection Justification

Final Model	Reasoning
Random Forest Model	The Random Forest Model algorithm contains many decision trees on various subsets of the given dataset and takes the mean of it to improve the accuracy of that dataset. It improves the accuracy and reduces overfitting by combining multiple trees. Random Forest Model also analyses feature importance by looking at how much each feature decreases the impurity (entropy) on average across all trees in the forest. Features that lead to larger reductions in impurity are considered more important.

	In the Anemia Detection Process, a lot of blood parameters are considered, finding the most important features amongst them is vital in order to create a model that will understand the underlying pattern. Detecting Anemia is a safety critical process which will directly result in the overall well being of the patient. Therefore, a model which will analyze multiple patterns and give a good accuracy along with identifying the important features in a must.

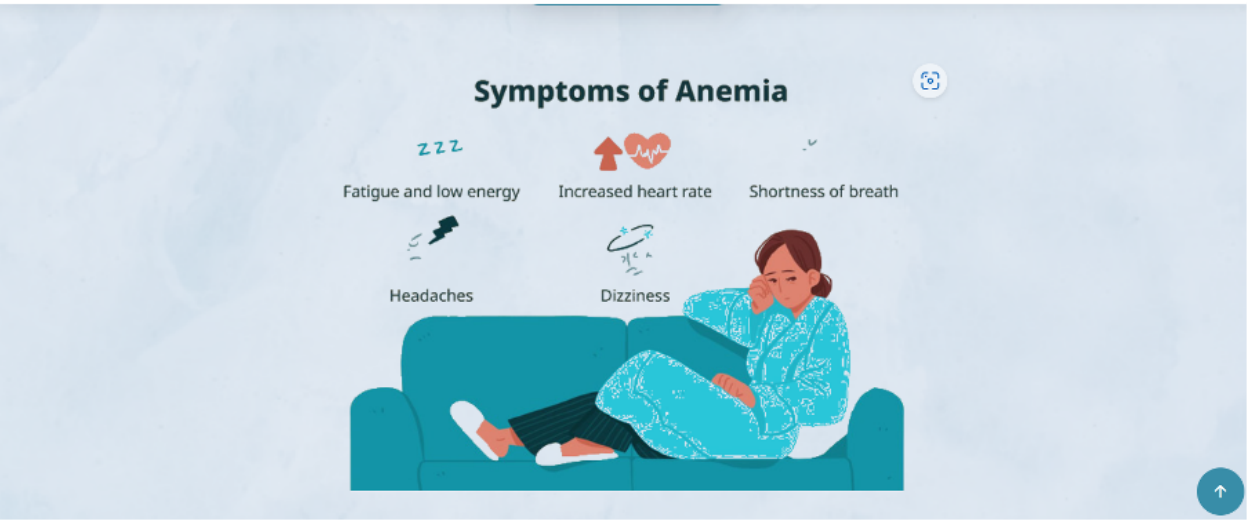
6.Results

6.1 Output screenshots



Anemia Sense

Predict



Anemia Sense

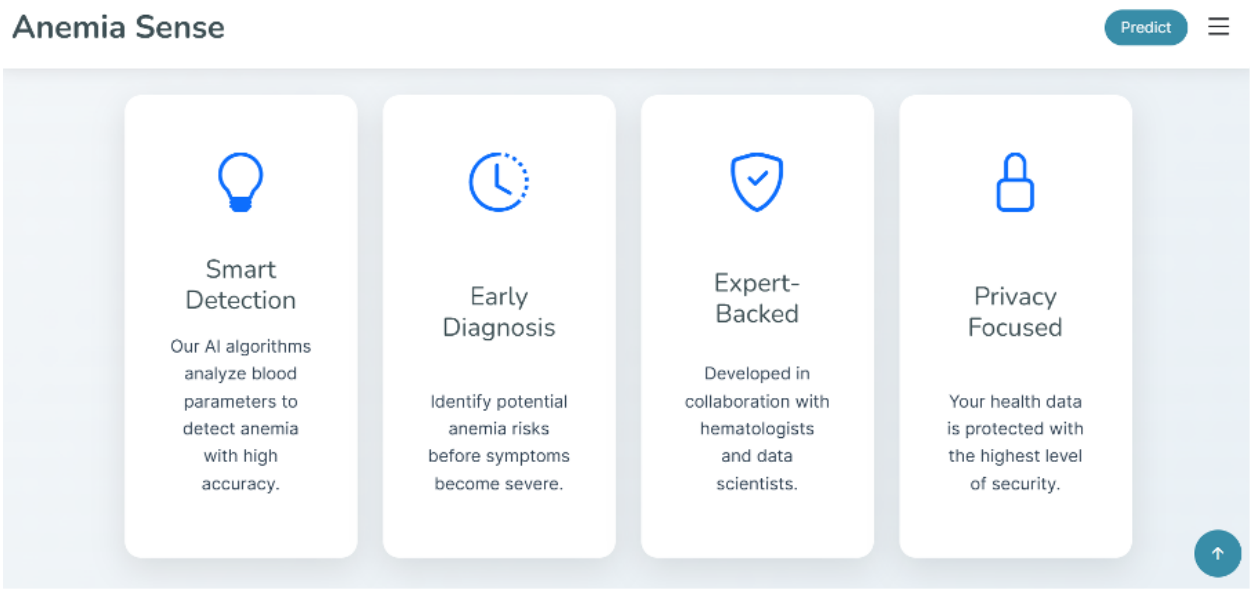
Predict

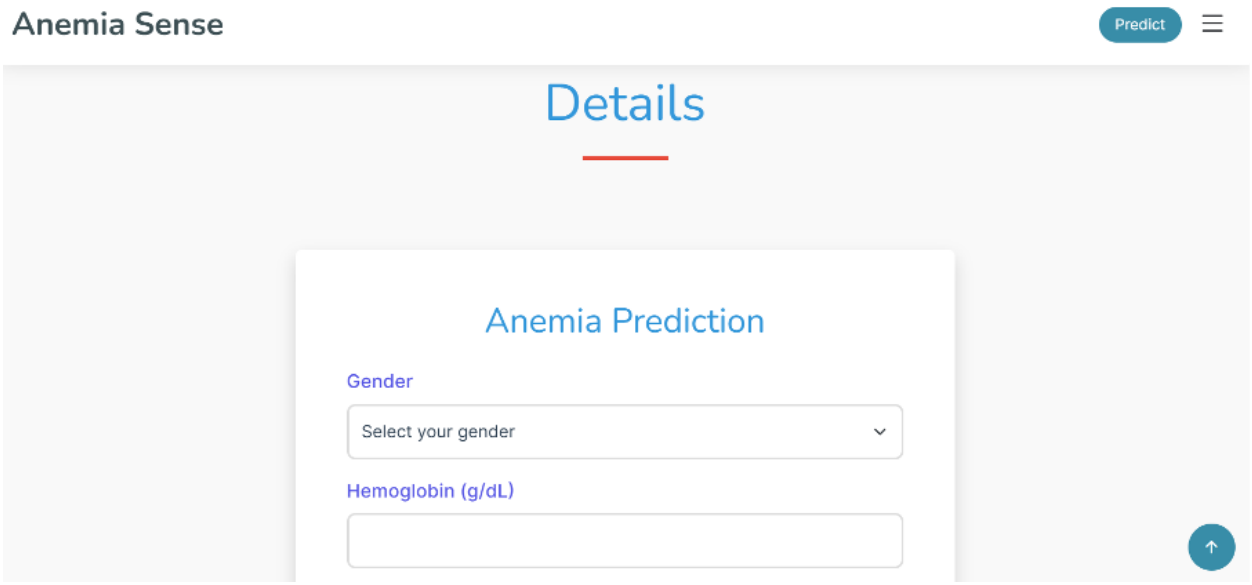
About Anemia Sense

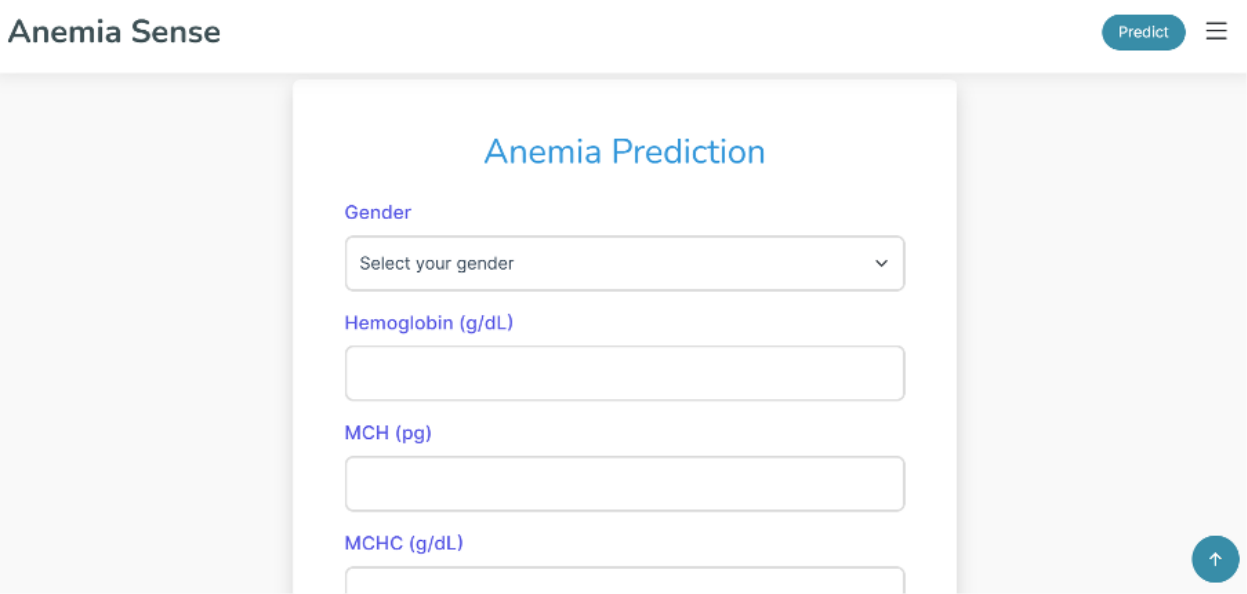
Revolutionizing Anemia Detection

Anemia Sense combines cutting-edge AI technology with medical expertise to provide accurate and early detection of anemia. Our mission is to make anemia screening accessible, quick, and reliable for everyone.

Try it now


Anemia Sense
Predict
Smart Detection
Our AI algorithms analyze blood parameters to detect anemia with high accuracy.
Early Diagnosis
Identify potential anemia risks before symptoms become severe.
Expert-Backed
Developed in collaboration with hematologists and data scientists.
Privacy Focused
Your health data is protected with the highest level of security.


Anemia Sense
Predict
Details
Anemia Prediction
Gender
Select your gender
Hemoglobin (g/dL)



MCH (pg)

MCHC (g/dL)

MCV (fL)

Get Prediction

Anemia Sense

Predict

Contact

Give Us Your Valuable Feedback

Address
VIT Vellore

Call Us
+91 9732972383

Email Us
anemia_sense_pjt@gmail.com

Anemia Sense

Predict

Your Name

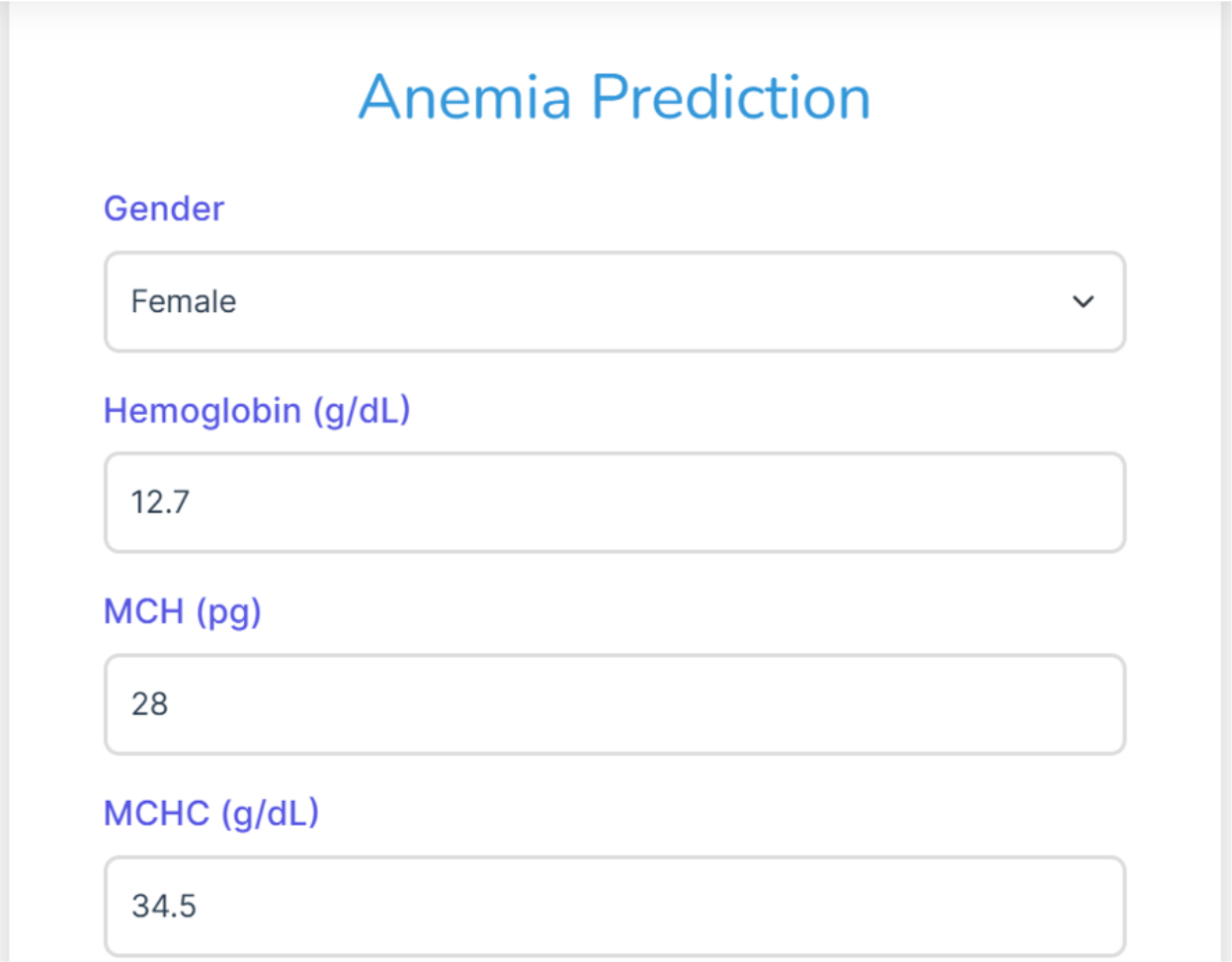
Your Email

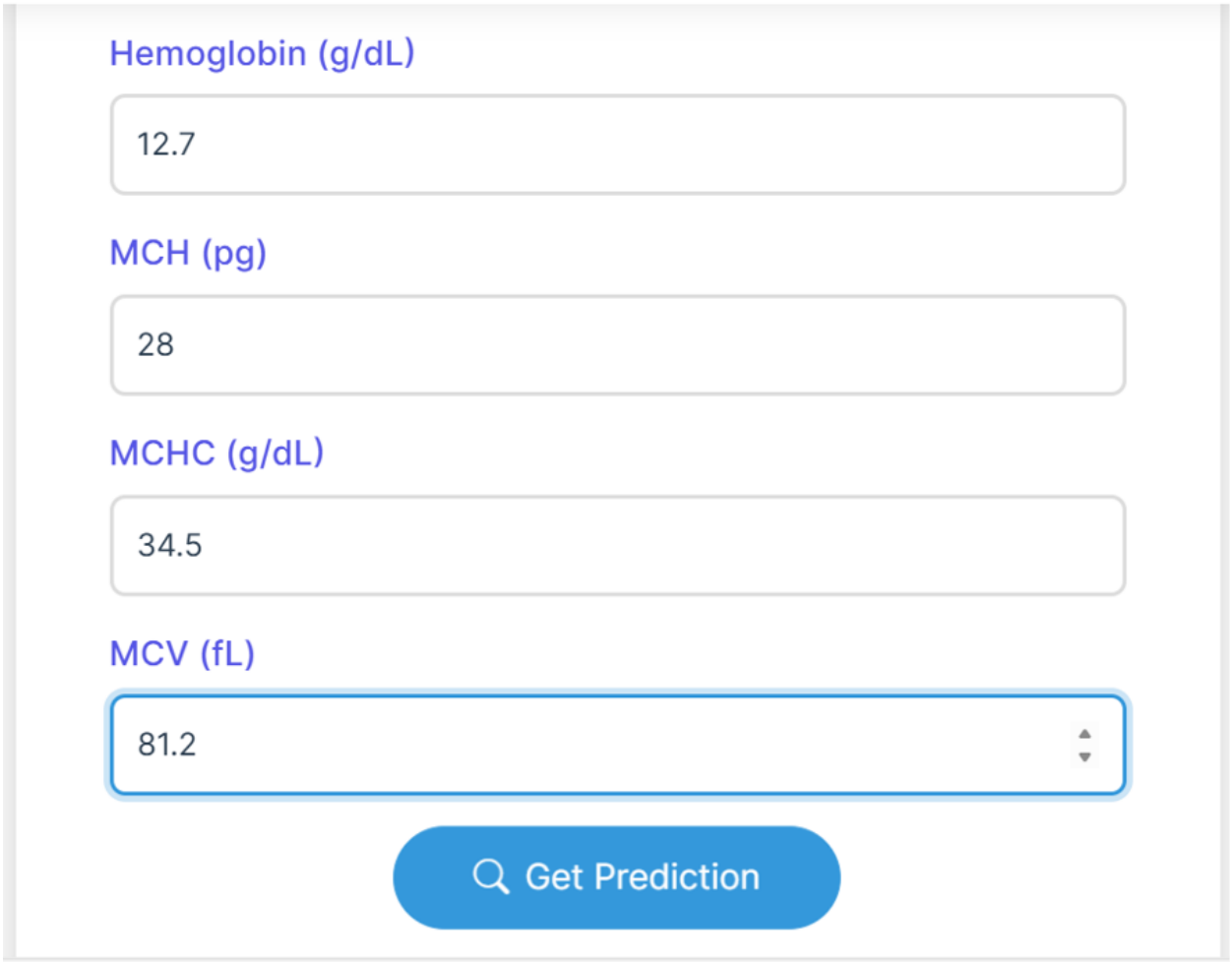
Subject

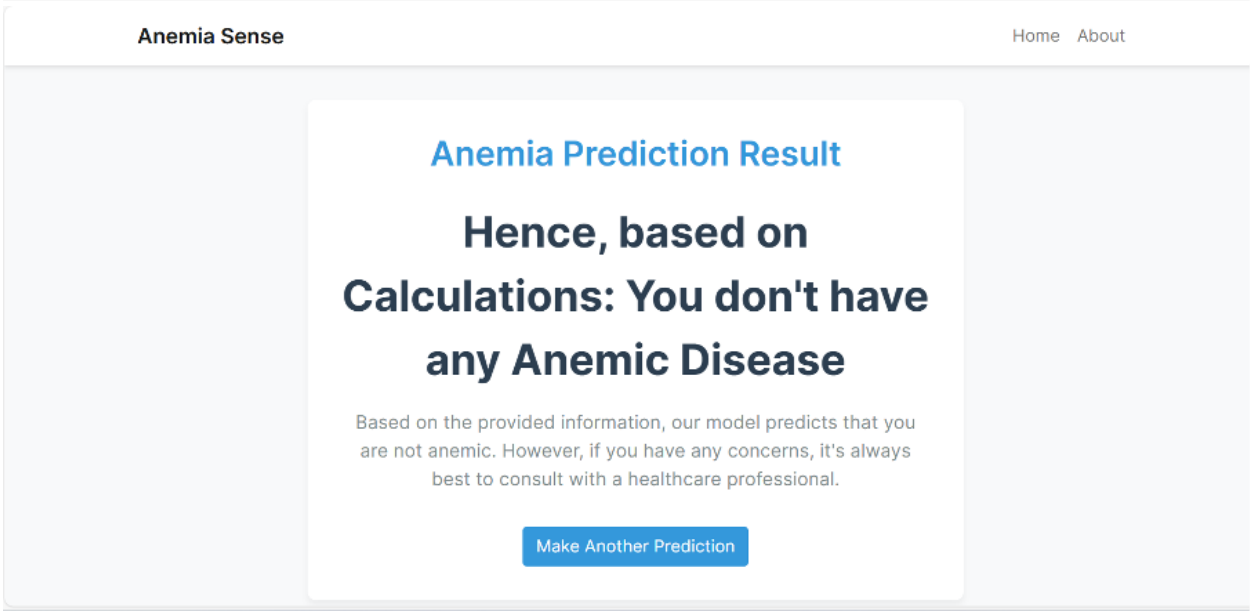
Message

Send Message

Example:


Anemia Prediction
Gender
Female
Hemoglobin (g/dL)
12.7
MCH (pg)
28
MCHC (g/dL)
34.5


Hemoglobin (g/dL)
12.7
MCH (pg)
28
MCHC (g/dL)
34.5
MCV (fL)
81.2
Get Prediction


Anemia Sense
Home
About
Anemia Prediction Result
Hence, based on Calculations: You don't have any Anemic Disease
Based on the provided information, our model predicts that you are not anemic. However, if you have any concerns, it's always best to consult with a healthcare professional.
Make Another Prediction

7.Advantages and Disadvantages

Advantages:

- Early Detection:
  Helps with early anaemia diagnosis, enabling prompt intervention and therapy that can avoid complications and enhance patient outcomes.

- Efficiency:
  Reduces the time and effort needed for healthcare providers to make manual diagnoses by automating the diagnostic procedure. This may result in the more effective use of medical staff and resources.

- Accessibility:
  Provide access to a diagnostic tool that can be utilised in a variety of situations, such as isolated or resource-constrained locations with limited access to specialised medical staff and equipment.

- Consistency and accuracy:
  Machine learning models have the potential to reduce the unpredictability and potential inaccuracies associated with human interpretation by offering consistent and objective diagnoses.

- Scalability:

  The solution is suitable for usage in various clinical settings and with greater patient populations because it can be scaled and incorporated into current healthcare systems.

- Constant Enhancement:
  In order to maintain accuracy and relevance when new data becomes available, the model can be updated and retrained on a regular basis.

Disadvantages:

- Data Quality and Availability:
  Data quality and comprehensiveness have a major impact on the machine learning model's accuracy. Predictions that are not accurate can be caused by partial, biassed, or noisy data.

- Interpretability:

  Certain machine learning models might be challenging to understand, particularly those that are sophisticated like neural networks. This lack of transparency may be problematic in medical situations when it's critical to comprehend the decision-making process.

- Initial Setup and Expenses:
  Time, knowledge, and resources must be committed in large amounts at the first stages of developing and implementing a machine learning model. This covers gathering data, training models, and setting up infrastructure.

- Maintenance & Updates:
  To make sure the model stays accurate and dependable over time, ongoing upkeep, upgrading, and maintenance are required. This continuous work necessitates committed resources.
- Ethical and Legal Issues: Using patient data raises ethical and legal issues, such as data security and patient privacy. It is imperative to guarantee adherence to rules like the Health Insurance Portability and Accountability Act.

- Over-reliance on technology is a potential consequence of using a machine learning model for diagnosis. It's critical to strike a balance and make sure medical professionals keep applying their clinical judgement in addition to the model's suggestions.

8.Conclusion

The development of a machine learning model for diagnosing anemia presents a significant opportunity to enhance healthcare delivery by leveraging advanced data analysis techniques. This project aims to create a reliable and efficient diagnostic tool that can accurately identify anemia based on key health indicators such as gender, haemoglobin levels, MCHC, and MCV. By automating the diagnostic process, the model promises to facilitate early detection and timely treatment of anemia, ultimately improving patient outcomes and reducing the burden on healthcare systems.

The proposed solution offers several notable advantages, including increased diagnostic efficiency, improved accessibility to medical diagnosis in resource-limited areas, consistent and accurate results, and scalability. However, it also presents challenges, such as the need for high-quality data, interpretability of complex models, initial setup costs, ongoing maintenance, and ethical considerations regarding patient data.

In conclusion, the successful implementation of this project could revolutionize the way anemia is diagnosed, making the process faster, more accessible, and more accurate. By continuously refining the model and addressing potential drawbacks, this project has the potential to make a meaningful impact on public health, particularly in underserved communities. As healthcare continues to evolve with technological advancements, integrating machine learning into diagnostic processes represents a critical step towards more efficient, equitable, and effective healthcare solutions.

9.Future Scope

The machine learning project for detecting anaemia has a wide range of potential applications in the future, as well as many avenues for improvement and development. The following are some important areas for project expansion and improvement:

- Integration with Electronic Health Records (EHR)
- Expansion to Other Blood Disorders
- Personalized Treatment Recommendations
- Incorporation of Advanced Machine Learning Techniques
- Mobile Application Development
- Real-Time Monitoring and Alerts
- Global Health Initiatives
- Longitudinal Data Analysis
- User Feedback and Iterative Improvement
- Collaborative Research and Development

10.Appendix

10.1 Source Code

#Data collection and Preprocessing

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
df=pd.read_csv('anemia.csv')
```

```
df.head()
df.tail()
df.columns()
df.isna()
df.info()
df.isnull().sum()
```

#Handling Imbalanced and missing values

```
results = df['Result'].value_counts()
results.plot(kind='bar',color=['blue','green'])
plt.xlabel('Result')
plt.ylabel('Frequency')
plt.title('Count of Result')
df['Gender'].value_counts()
from sklearn.utils import resample
majorclass = df[df['Gender']==1]
minorclass = df[df['Gender']==0]
major_downsample = resample(majorclass, replace=False,
n_samples=len(minorclass),random_state=42)
df=pd.concat([major_downsample,minorclass])
print(df['Gender'].value_counts())
results = df['Gender'].value_counts()
results.plot(kind='bar',color=['blue','green'])
plt.xlabel('Gender')
plt.ylabel('Frequency')
plt.title('Count of Gender')
```

#Descriptive Analysis

```
df.describe()
```

```
plt.figure(figsize=(10, 6))
sns.histplot(df['Hemoglobin'], kde=True)
plt.title('Distribution of Hemoglobin Levels')
plt.xlabel('Hemoglobin Level')
plt.ylabel('Frequency')
plt.show()
```

```
plt.figure(figsize=(10, 6))
sns.histplot(df, x='Hemoglobin', hue='Gender', element='step', kde=True)
plt.title('Distribution of Hemoglobin Levels by Gender')
plt.xlabel('Hemoglobin Level')
plt.ylabel('Frequency')
plt.show()
```

#Univariate Analysis

```
output=df['Gender'].value_counts()
output.plot(kind='bar',color=['orange','green'])
plt.xlabel('Gender')
plt.ylabel('Frequency')
plt.title('Gender Count')
plt.show()
sns.displot(df['Hemoglobin'],kde=True)
```

#Bivariate Analysis

```
df = pd.read_csv('anemia.csv')

mean_hemoglobin = df.groupby(['Gender', 'Result'])['Hemoglobin'].mean().reset_index()

plt.figure(figsize=(10, 6))

sns.barplot(x='Gender', y='Hemoglobin', hue='Result', data=mean_hemoglobin, palette='viridis')

plt.title('Mean Hemoglobin Levels by Gender and Result')

plt.xlabel('Gender')

plt.ylabel('Mean Hemoglobin Level')

plt.legend(title='Result')

plt.show()

#box plot

plt.figure(figsize=(10, 6))

sns.boxplot(x='Gender', y='Hemoglobin', data=df, palette='viridis')

plt.title('Box Plot of Hemoglobin Levels by Gender')

plt.xlabel('Gender')

plt.ylabel('Hemoglobin Level')

plt.show()
```

#Multivariate Analysis

```
sns.pairplot(df)

sns.heatmap(df.corr(),annot=True,cmap='RdYlGn',linewidths=0.2)

fig=plt.gcf()

fig.set_size_inches(10,8)

plt.show()
```

#Splitting Data Into Train and Test

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(X,Y,test_size=0.2,random_state=20)
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

#Logistic Regression

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
logistic_regression=LogisticRegression()
logistic_regression.fit(x_train,y_train)
pred=logistic_regression.predict(x_test)
pred
acc_lr=accuracy_score(y_test,pred)
confusion_matrix(y_test,pred)
acc_lr=accuracy_score(y_test,pred)
c_lr=classification_report(y_test,pred)
print('Accuracy Score: ',acc_lr)
print(c_lr)
```

#Random Forest Model

```
from sklearn.ensemble import RandomForestClassifier
rf=RandomForestClassifier(n_estimators=10,criterion='entropy',random_state=0)
rf.fit(x_train,y_train)
pred=rf.predict(x_test)
```

```
pred
y_test
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
acc_rf=accuracy_score(y_test,pred)
conmat=confusion_matrix(y_test,pred)
print(acc_rf)
print(conmat)
c_rf=classification_report(y_test,pred)
c_rf
```

#Decision tree

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
decision_tree_model = DecisionTreeClassifier()
decision_tree_model.fit(x_train, y_train)
y_pred = decision_tree_model.predict(x_test)
y_pred
y_test
acc_dt = accuracy_score(y_test, y_pred)
acc_dt
conmat = confusion_matrix(y_test, y_pred)
conmat
c_dt = classification_report(y_test, y_pred)
print(c_dt)
```

#Naive Bayes Model

```
from sklearn.naive_bayes import GaussianNB
```

```
nb = GaussianNB()
nb.fit(x_train,y_train)
pred = nb.predict(x_test)
acc_nb=accuracy_score(pred,y_test)
c_nb=classification_report(pred,y_test)
print("Accuracy Score: ",acc_nb)
print(c_nb)
```

#Support Vector Machine

```
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn import metrics
st_x= StandardScaler()
x_train= st_x.fit_transform(x_train)
x_test= st_x.transform(x_test)
support_vector=SVC()
support_vector.fit(x_train,y_train)
y_pred=support_vector.predict(x_test)
acc_svc=metrics.accuracy_score(y_test,y_pred)
c_svc=metrics.classification_report(y_test,y_pred)
print("accuracy_score: ",acc_svc)
print("classification_report: ")
print(c_svc)
```

#Gradient Boosting Classifier

```
from sklearn.ensemble import GradientBoostingClassifier
GBC= GradientBoostingClassifier()
```

```
GBC.fit(x_train,y_train)
y_pred2=GBC.predict(x_test)
acc_gbc=metrics.accuracy_score(y_test,y_pred2)
c_gbc=metrics.classification_report(y_test,y_pred2)
print("accuracy_score: ",acc_gbc)
print("classification_report: ")
print(c_gbc)
```

#Testing the Models

```
#Logistic Regression Model
pred_lr=logistic_regression.predict([[0,11.6,0,30.9,74.5]])
print("Logistic Regression:",pred_lr)
if(pred_lr==0):
    print("You don't have Anemic Disease")
elif(pred_lr==1):
    print("You have Anemic Disease")

#Random forest model
pred_rf=rf.predict([[0,11.6,0,30.9,74.5]])
print("Random forest:",pred_rf)
if(pred_rf==0):
    print("You don't have Anemic Disease")
elif(pred_rf==1):
    print("You have Anemic Disease")

#Decision Tree Model
pred_dt=decision_tree_model.predict([[0,11.6,0,30.9,74.5]])
```

```
print("Random forest:",pred_dt)
if(pred_dt==0):
    print("You don't have Anemic Disease")
elif(pred_dt==1):
    print("You have Anemic Disease")

#Gaussian Navies Bayes
pred_nb=nb.predict([[0,11.6,0,30.9,74.5]])
print("Random forest:",pred_nb)
if(pred_nb==0):
    print("You don't have Anemic Disease")
elif(pred_nb==1):
    print("You have Anemic Disease")

#Support Vector Machine
pred_svm=support_vector.predict([[0,11.6,0,30.9,74.5]])
print("Random forest:",pred_svm)
if(pred_svm==0):
    print("You don't have Anemic Disease")
elif(pred_svm==1):
    print("You have Anemic Disease")

#Gradient Boosting Classifier
pred_gbc=GBC.predict([[0,11.6,0,30.9,74.5]])
print("Random forest:",pred_gbc)
if(pred_gbc==0):
    print("You don't have Anemic Disease")
```

```
elif(pred_gbc==1):
    print("You have Anemic Disease")
```

#Performance Testing

```
model = pd.DataFrame({'Model':['Linear Regression','Decision Tree Classifier','Random Forest Classifier','Gaussian Naive Bayes','Support Vector Machine',
                               'Gradient Boosting Classifier'],
                      'Score':[acc_lr,acc_dt,acc_rf,acc_nb,acc_svc,acc_gbc],
                     })
model
```

10.2 GitHub and Project Link

Github link:

https://github.com/Vishnu-Adi/Anemia_Sense.git

Presentation link:

https://drive.google.com/file/d/1pLpP6MtpWn1UUn1KYzlJkBZNDtMZWkAD/view?usp=sharing

Website link:

https://anemiasense.onrender.com/