

# Project Report

## Prediction of Anemia Severity Using Machine Learning

- **Project Title:** Prediction of Anemia Severity Using Routine Blood Parameters Using Machine Learning
- **Student Name:** BHUMIKA SHARMA
- **Registration Number:** 25BOE10070
- **Course:** Fundamentals of AI & ML
- **Programme:** B.Tech Bioengineering
- **Institution:** VIT BHOPAL UNIVERSITY
- **Faculty:** Prof. Vivek Parashar

## 2. Introduction

Anemia is a very common blood-related condition, especially in developing countries, and it often gets ignored until it becomes severe. I personally found this topic interesting because, in bioengineering, we study a lot about human physiology and lab measurements, and I realized that even basic blood parameters like Hemoglobin, Hematocrit, RBC count, etc., can tell a lot about someone's health. Since ML is becoming important in the healthcare field, I thought it would be a good idea to build a small model that predicts anemia severity from simple CBC values.

In real hospitals, doctors usually rely on lab reports, but classification sometimes takes time or depends on experience. So the idea of this project is to create an ML model that can help with a quick prediction. It is not meant to replace a doctor or anything like that, but just to support decision-making.

## 3. Problem Statement

The main problem addressed in this project is: "*Can we predict the severity level of anemia (Normal, Mild, Moderate, or Severe) using routine blood test parameters and basic demographic details?*"

This will help in early and quick assessment, especially in setups where advanced tests may not be immediately available.

## 4. Objectives

1. To collect or generate a clean dataset with routine CBC parameters.
2. To preprocess the data and create meaningful features.
3. To build multiple ML models and compare their performance.
4. To classify anemia into 4 severity levels.
5. To evaluate the model using proper metrics and validations.
6. To document the whole pipeline clearly.

## 5. Functional Requirements

1. A functional data-loading module that reads CSV input.
2. Preprocessing module to clean, imputes, and encode data.
3. Feature engineering (MCV categories, RDW flag).
4. Model training module with at least 1–2 ML algorithms.
5. Prediction module that takes input values and outputs severity.
6. Evaluation module with confusion matrix and classification report.

## 6. Non-Functional Requirements

1. **Usability:** Should be simple for a student or beginner to run.
2. **Reliability:** Should handle missing or wrong inputs gracefully.
3. **Performance:** Prediction should take less than a second.
4. **Maintainability:** Code must be modular and readable.
5. **Reproducibility:** Using fixed random seeds and saved models.
6. **Scalability:** Can be extended to use more features in the future.

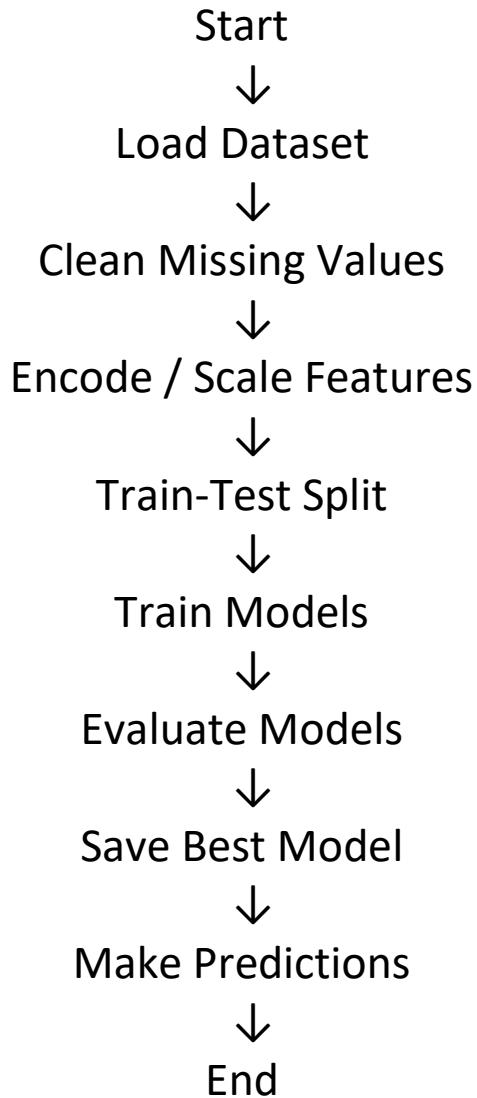
## 7. System Architecture

Below is the architecture structure in a simple flow style: Raw CBC Data → Preprocessing → Feature Engineering → Model Training → Evaluation → Saved Model → Prediction Script

Components:

- **Data Layer:** Stores raw.csv and processed.csv
- **Processing Layer:** Cleaning, imputation, encoding, feature derivation
- **ML Layer:** Model training, selection, evaluation
- **Application Layer:** Prediction module with explainability

## 8. Workflow Diagram (Process Flow)



## 9. UML Use Case Diagram (text description)

**Actors:** Student/Developer, Evaluator, Clinician (optional)

**Use Cases:** - Upload dataset - Train model - Evaluate accuracy - Predict anemia severity - View feature importance

## 10. Sequence Diagram (Prediction Flow)

User → Predictor → Load Model → Apply Preprocessing → Predict Severity → Display Output

## 11. Class Diagram (Descriptive)

- **DataLoader:** load(), clean(), impute()
- **FeatureEngineer:** engineer\_features()
- **ModelTrainer:** train(), evaluate(), tune()
- **Predictor:** predict(), explain()

## 12. Database / Storage Design

(No real database used, only CSV storage)

- /data/raw.csv — original input data
- /data/processed.csv — cleaned data
- /models/ — trained model files (model.pkl, scaler.pkl)

## 13. Design Decisions & Rationale

- **Why ML?** Because anemia severity can be predicted from multiple lab parameters which show patterns.
- **Why Random Forest?** It handles non-linear relations in blood values better.
- **Why SHAP?** To make predictions more interpretable for medical usage.
- **Why CSV?** Simple to maintain and submit.

## 14. Implementation Details

- Python 3 with scikit-learn
- separate modules folder /src/  
Notebooks for EDA and modeling
- Feature scaling wherever required

- Saving model with joblib
- Confusion matrix and classification report generated

## 15. Screenshots / Results

The screenshot shows a code editor interface with the following details:

- Project Structure:** The left sidebar shows a project named "ANEMIAPROJECT" with the following directory structure:
  - DATA: processed.csv, raw.csv
  - DIAGRAMS
  - MODELS
  - PROJECT REPORT
  - SCREENSHOTS
  - src: \_\_pycache\_\_, \_\_init\_\_.py, data\_prep.py, predict.py, train.py, requirements.txt
- Code Editor:** The main area displays a file named "data\_prep.py" with the following code:
 

```
16
17 # Save processed file
18 df.to_csv("DATA/processed.csv", index=False)
19
20 print("Processed dataset saved with Anemia_Label column!")
21
```
- Terminal:** Below the code editor is a terminal window showing the execution of the script and its output:
 

```
Model saved successfully!
PS C:\Users\dell\Desktop\ANEMIAPROJECT> & C:\Users\dell\AppData\Local\Programs\Python\Python311\python.exe c:/Users/dell/Desktop/ANEMI
APROJECT/src/data_prep.py
Processed dataset saved with Anemia_Label column!
PS C:\Users\dell\Desktop\ANEMIAPROJECT>
```
- Bottom Status Bar:** The status bar at the bottom indicates the current file is "data\_prep.py", the Python version is "Python 3.11 (64-bit)", and the file size is "0 0 0".

Age	Sex	Hemoglobin	Hematocrit	RBC_count	MCV	MCH	MCHC	RDW	WBC	Platelets	AnemiaSeverity
56.0	1.0	15.73	47.19	3.95	85.75	34.61	33.0	12.8	5.97	273.08	0.0
69.0	0.0	10.5	31.5	3.58	86.07	23.1	35.43	14.55	7.06	295.17	1.0
46.0	0.0	10.85	32.55	4.29	72.01	23.87	33.93	13.97	5.96	229.28	1.0
32.0	0.0	12.55	37.65	4.38	86.57	27.61	33.09	15.07	7.48	219.57	0.0
60.0	0.0	17.87	53.61	4.47	102.01	39.31	33.13	13.15	6.85	241.57	0.0

## DATA SET PREVIEW, PREDICT, TRAINING AND DATA PREP OUTPUT SCREENSHOTS

## 16. Testing Approach

- Manual testing on a few known samples
  - Cross-validation (StratifiedKFold)
  - Testing for missing/invalid values
  - Simple runtime test for prediction speed

# 17. Challenges Faced

- Honestly, one challenge was understanding the clinical thresholds for anemia severity because the cutoffs differ slightly for males and females.

- Another issue was that moderate and severe cases were fewer, causing slight class imbalance, so I tried using class\_weight or oversampling.
- Getting SHAP to run the first time was also slightly confusing. As previously I have worked on the similar kind of project but not exactly using some things that used this time.

## 18. Learnings & Key Takeaways

- Understood preprocessing of biomedical numerical data.
- Learned how ML models like Random Forest perform with clinical datasets.
- SHAP explainability taught me how features affect predictions.
- Gained confidence in handling a full ML pipeline.

## 19. Future Enhancements

- Include iron studies like Ferritin, TIBC for better predictions.
- Deploy model as a simple medical web tool.
- Validate with larger real-world datasets.

## 20. References

- WHO Anemia Guidelines
- scikit-learn documentation
- SHAP documentation
- Class notes and online ML resources