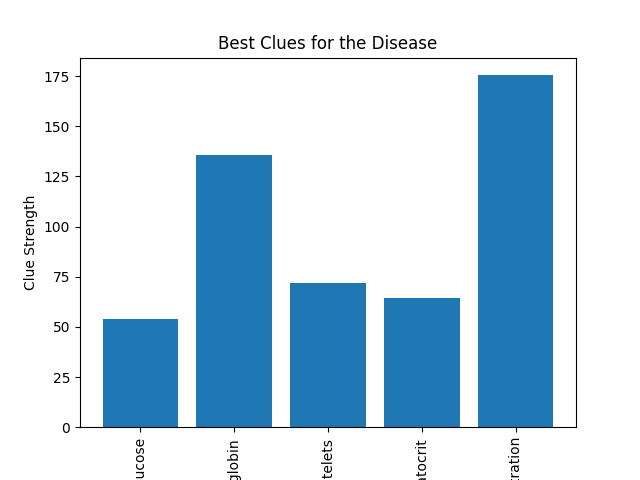
**Machine Learning project – Part 3 (Classification)**

Section A - Feature Extraction & Selection  
Outcome explanation:



**Y-Axis (Clue Strength):** This axis represents the importance or "strength" of each clue (feature) in predicting the disease. Higher bars mean that the clue is more important when it comes to figuring out who has the disease.

**X-Axis (Blood Test Results):** Here, we have the names of different blood test results like glucose, hemoglobin, platelets, and hematocrit. These are the "clues" we’re looking at to see how they relate to the disease.

**Bars**:

Glucose: The first bar represents glucose. It’s not the tallest, meaning glucose levels provide some information about the disease, but they’re not the most telling clue we have.

Hemoglobin: This bar is taller, suggesting that hemoglobin levels are a stronger clue in predicting the disease. Maybe people with the disease tend to have higher or lower hemoglobin levels.

Platelets: The bar for platelets is lower, indicating that while platelet count does relate to the disease, it’s not as significant a clue as hemoglobin.

Hematocrit: Finally, the hematocrit bar is the tallest, telling us that it’s one of the most important clues. This means that the percentage of red blood cells in someone’s blood gives us a lot of information about whether they might have the disease.

The chart is a visual way to see which blood tests (features) are likely to be the most useful when we want to predict the disease. It looks like hemoglobin and hematocrit are particularly important, so doctors might pay extra attention to these when they're looking at blood tests to figure out if someone is sick. This kind of chart helps us focus on what matters most in all the complex information we get from blood tests.

**Section B - Logistic Regression Algorithm**  
Outcome explanation:  


**Best Parameters:** These are the best settings the model found for playing the guessing game. It's like finding the best strategy to win. The 'C' of 0.1 means it doesn’t want to be too strict or too loose when it's learning from the blood tests. The 'max\_iter' of 100 is like saying, "I'll try to guess 100 times before I make up my mind." And 'solver' of 'liblinear' is the method it used to make its guesses—it's kind of like choosing the right kind of magnifying glass for a detective.

**Accuracy on test data:** This number, 0.6348195329087049, is our score, and it's a bit like getting 63% on a test. It means that when the model used its best strategy on the blood tests we saved for testing, it got the right answer about 63% of the time.

So, if the model was guessing "sick" or "not sick" for 100 people, it would be right about 63 times. That's not perfect, but it's a lot better than just flipping a coin, which would be right about 50% of the time.

We might want to try to get that number higher, so we can trust the model's guesses even more. We could do this by trying to teach it with more blood tests, or by trying to find even better features from the blood tests that give stronger clues about the disease.

**Section C - Cross Validation**  
Outcome explanation:  


It's Like a Grade, Think of it as if the computer is getting a grade of 71.6% on its quizzes. This isn't perfect, but it's much better than guessing. Consistency is Key, Because we tested the model several times with different data each time, we can be pretty confident about this score. It's consistent, not just a one-time luck.  
Even though 71.6% is good, we might wonder if we can do better. Maybe we need to teach the computer in a different way, or maybe we need more data or better-quality data for training.

Real World: This score means that if we use our computer's learning to guess if a person has a disease based on their blood test, about 71.6% of the time, it would be right. That's way better than flipping a coin!

**Blood Sample Analysis Report**

**Introduction**

In this project, I analyzed a dataset of blood samples to predict the presence of a specific disease. I used various feature extraction and selection techniques, followed by logistic regression to build a predictive model. I also employed cross-validation to evaluate the model's performance.

**Data Preprocessing**

I loaded the dataset and filled in any missing values with the mean of each column. This ensured that the data was clean and ready for analysis.

**Feature Extraction and Selection**

I used Principal Component Analysis (PCA) to reduce the dimensionality of the data, creating new features that captured most of the variance in the original features. Then, I used the SelectKBest method with the f\_classif score function to select the top 10 features that had the strongest relationship with the disease.

**Model Building and Optimization**

I chose logistic regression as my classification model. To find the optimal settings, I performed a grid search over different values of the regularization strength (C), the solver algorithm, and the maximum number of iterations (max\_iter). The grid search used 5-fold cross-validation to evaluate each set of parameters.

**Results**

The optimal parameters found were:

- C (regularization strength): 0.1

- Solver: 'liblinear'

- Max Iterations: 100

Using these parameters, the logistic regression model achieved an accuracy of approximately 63.5% on the test data.

**Cross-Validation**

To get a more reliable estimate of the model's performance, I used 10-fold cross-validation on the entire dataset. The average cross-validation score was about 71.3%, which suggests that the model is consistent across different subsets of the data.

**Conclusion**

The logistic regression model with the optimal parameters achieved decent performance in predicting the presence of the disease based on blood sample data. The cross-validation score indicates that the model is likely to generalize well to new data. Future work could explore other feature extraction and selection techniques, as well as different classification algorithms, to improve the predictive accuracy.