White Blood Cell Classification Project Report
By Anthony Medina

- **Problem Statement**

Neutrophils, monocytes, lymphocytes and eosinophils are white blood cells with varying jobs. Being able to classify white blood cells has potential help in disease diagnosis, monitoring health status, treatment planning, research in clinical trials, transfusion capacity, hematological disorders, infectious disease identification, research and development of therapies, and many more. What can we learn from a blood test once we apply machine learning to help identify white blood cell types?

- **Data Wrangling**

We initially loaded the data with the csv that was provided with the data, but it was missing values and had an extra column. Instead, I created my own data frame with folders of images that I had for exploration. I created a new data frame that cycled through all of the images of cells that I had and included their names, image paths, shapes, and images themselves. At this point, I knew that I had no missing data.

- **Exploratory Data Analysis**

My data exploration was motivated by a few questions. How many images of each cell type do I have? Are the images distinct enough for them to be recognized? How many different image sizes are there? I looked at an image of each type that I picked randomly. I created a histogram of each cell type to see the distribution of images. There were a little over 3000 images of each type. Each image was 240 x 320.

- **Model Preprocessing with feature Engineering**

I needed to get the data in a different format for each model to consume it. With the number of images, I ran into hardware limitations, so a solution was to reduce the size of the images so that I could train the model effectively. For the Efficient Net, the data was flattened and transformed so that the range of images was normalized. Then the labels were encoded, then the data was split into train and test sets. With the Neural Net, I resized the images to half the size. This decreased the run time of the algorithm to about 15 minutes. Finally, for the Random Forest model, the images were flattened.

- **Algorithms used to build the model evaluation metric**
    - Efficient Net: .247
    - Neural Network: .545
    - Random Forest: .665

High recall is crucial because it's more important to catch all the true cases, while minimizing false negatives, even if it means accepting some false positives. There are other metrics we could have used, but since we're looking at samples of blood cells for medical diagnosis, all of the algorithms used recall as the metric.

- **Winning Model and Scenario Modeling**

The Random Forest ended up being the best model, initially. After hyperparameter tuning, I found that the final model would have no maximum depth, a minimum of 4 samples in each split and 200 different estimators.

The final model had a .685 recall score using the test data.

- **Recommendation**

The model has been saved and is available to use for other images of white blood cells. The attached notebook has instructions for resizing new images and using the model.

- **Conclusion**

I started work on an image recognition problem because I was interested in a classification model and hadn't had much experience. I learned a lot about image processing in this notebook. I learned about the limitations of memory in my computer and how much data I'm able to process with a model. I ran into performance issues when I tried to run my notebook and included all of the data. None of the models would run with all of the full sized images. To get around this, I tried using an AWS EC2 instance, but ran into billing issues. I ended up running the Efficient Net with fewer full sized images and running the Neural Net and Random Forest with the size of the images greatly reduced.

- **Future Scope of Work**

One of the biggest setbacks was my limitation in hardware. When I tried to run models with all of the full sized images, my kernel would stop because I was out of memory. Resizing the images made the models less efficient. Also, when I looked at other people's notebooks that had worked with this data set, they were unable to get beyond .50 accuracy with their models.