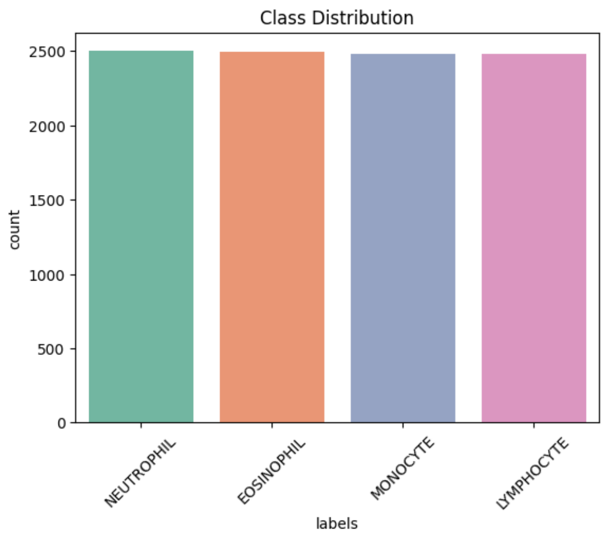
Data Collection and Preprocessing Phase

Date	05 July 2025
Team ID	SWTID1749835721
Project Title	HematoVision – Blood Cell Classification using Transfer Learning
Maximum Marks	6 Marks

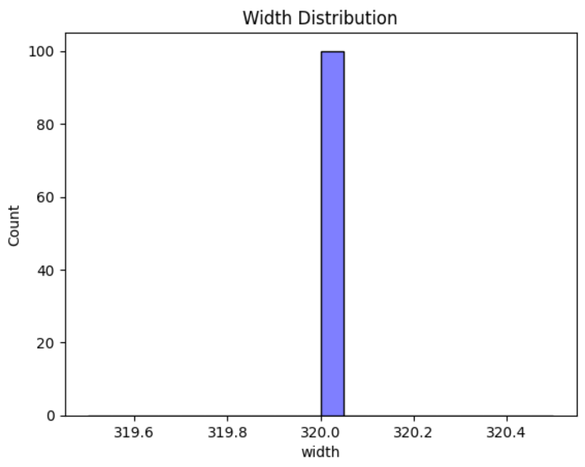
Data Exploration and Preprocessing Report

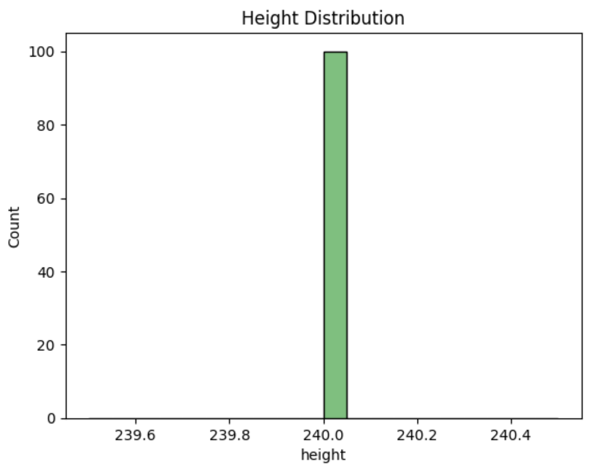
In this phase, image data of blood cells are analyzed and preprocessed for use in a deep learning pipeline. The dataset comprises images from four blood cell classes: **NEUTROPHIL**, **LYMPHOCYTE**, **MONOCYTE**, and **EOSINOPHIL**. Each image is linked to a class label and stored in a structured directory format. The dataset is explored for class balance, image integrity, and shape variation, and preprocessed to support accurate training of image classification models.

Section	**Description**
Data Overview	**Dimension:** • Number of images: XXXX (replace with actual count) • Number of classes: 4 **Class Labels:** • NEUTROPHIL • LYMPHOCYTE • MONOCYTE • EOSINOPHIL **Format:** • Image formats: .jpeg, .jpg, .png • Directory structure used to assign labels **Descriptive Statistics:** • **Class Distribution:** The count of images per class is analyzed to detect any class imbalance.

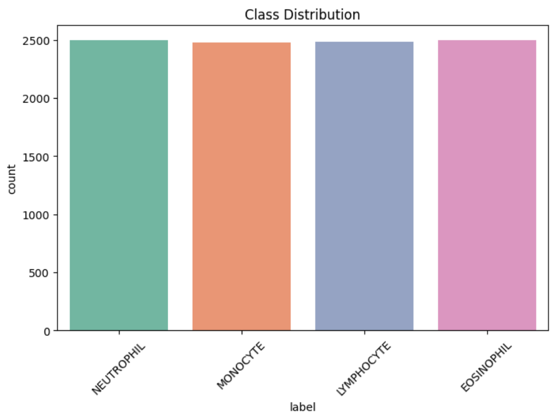
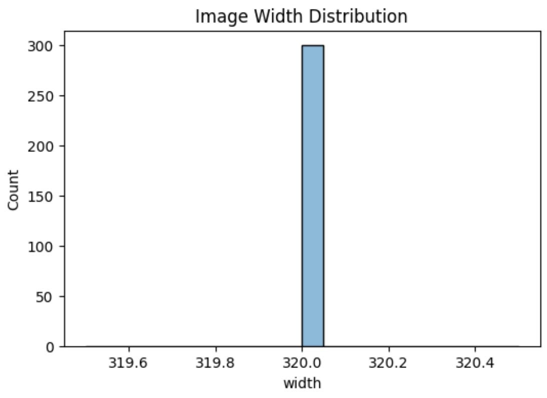
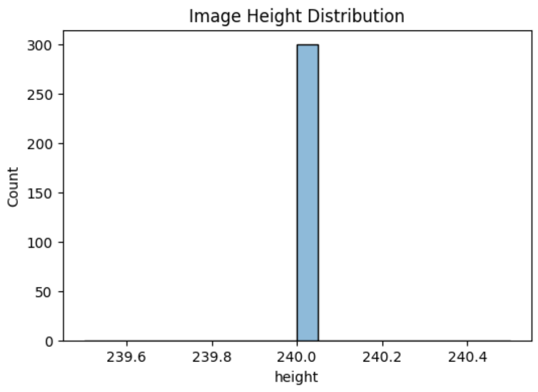


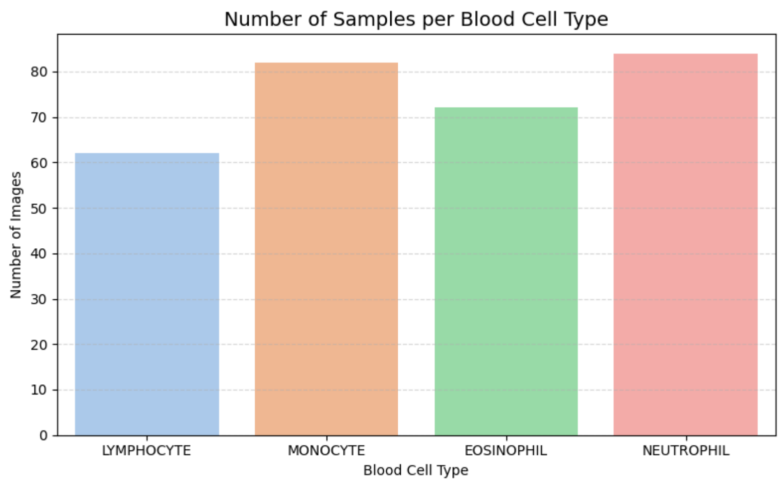
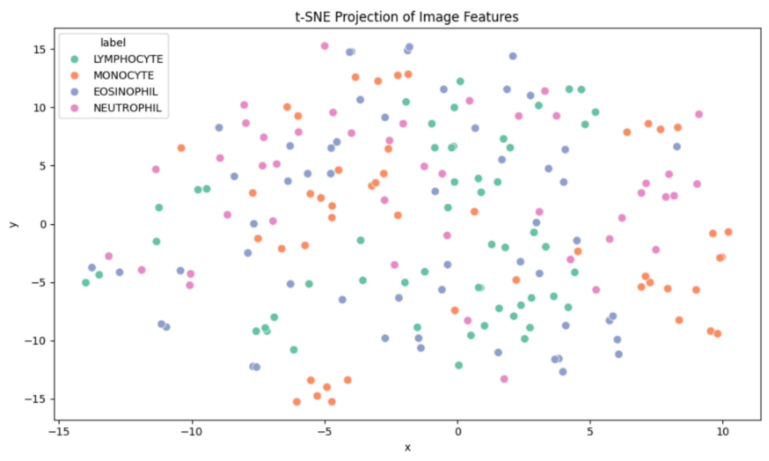
- **Image Dimensions:**
  A sample of images is used to understand the variation in width and height.
  This ensures input shapes are consistent before feeding to a CNN.



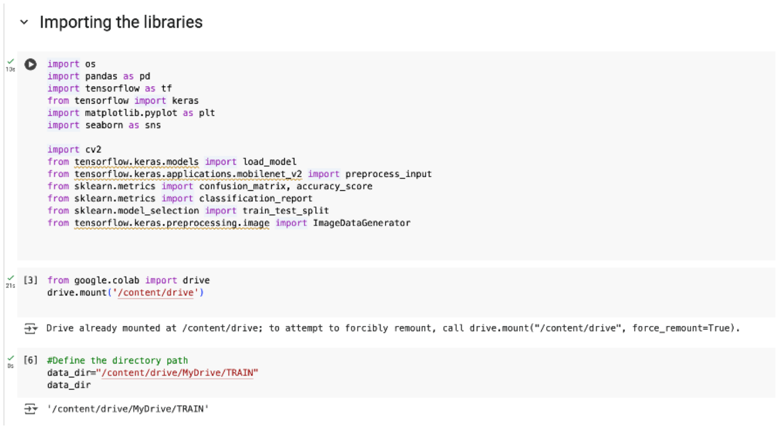


Univariate Analysis	

Bivariate Analysis	
Multivariate Analysis	

Data Preprocessing Code Screenshots

Loading Data	

Importing the libraries

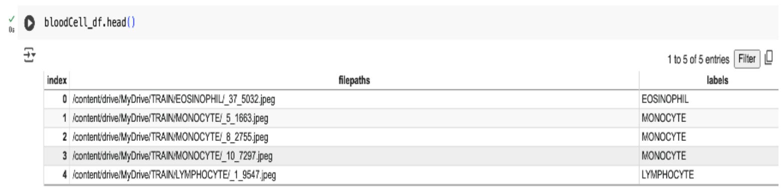
```
import os
import pandas as pd
import tensorflow as tf
from tensorflow import keras
import matplotlib.pyplot as plt
import seaborn as sns

import cv2
from tensorflow.keras.models import load_model
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.metrics import classification_report
from sklearn.model_selection import train_test_split
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

```
[3] from google.colab import drive
    drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
[6] #Define the directory path
    data_dir="/content/drive/MyDrive/TRAIN"
    data_dir
```

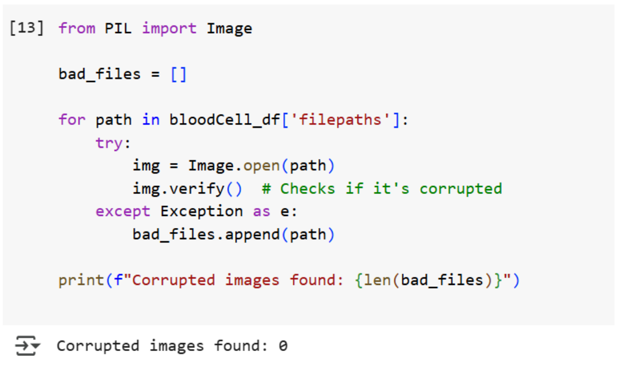
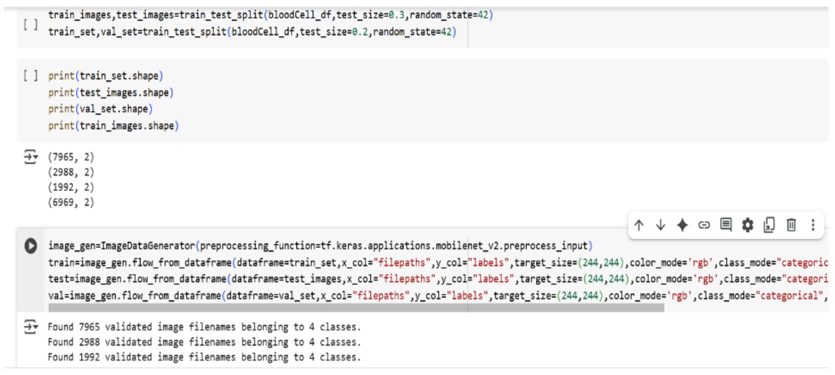
```
'/content/drive/MyDrive/TRAIN'
```

```
bloodCell_df.head()
```

1 to 5 of 5 entries

index	filepaths	labels
0	/content/drive/MyDrive/TRAIN/EOSINOPHIL/_37_5032.jpeg	EOSINOPHIL
1	/content/drive/MyDrive/TRAIN/MONOCYTE/_5_1663.jpeg	MONOCYTE
2	/content/drive/MyDrive/TRAIN/MONOCYTE/_8_2755.jpeg	MONOCYTE
3	/content/drive/MyDrive/TRAIN/MONOCYTE/_10_7297.jpeg	MONOCYTE
4	/content/drive/MyDrive/TRAIN/LYMPHOCYTE/_1_9547.jpeg	LYMPHOCYTE

Handling Missing Data	
Data Transformation	

```
[13] from PIL import Image

     bad_files = []

     for path in bloodCell_df['filepaths']:
         try:
             img = Image.open(path)
             img.verify()  # Checks if it's corrupted
         except Exception as e:
             bad_files.append(path)

     print(f"Corrupted images found: {len(bad_files)}")
```

```
Corrupted images found: 0
```

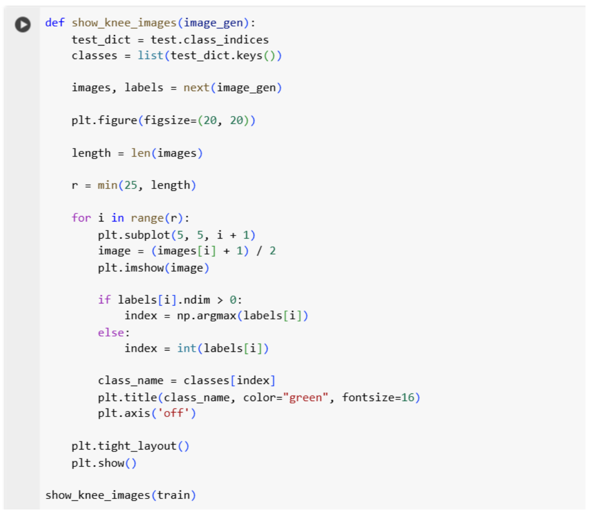
```
[ ] train_images,test_images=train_test_split(bloodCell_df,test_size=0.3,random_state=42)
    train_set,val_set=train_test_split(bloodCell_df,test_size=0.2,random_state=42)
```

```
[ ] print(train_set.shape)
    print(test_images.shape)
    print(val_set.shape)
    print(train_images.shape)
```

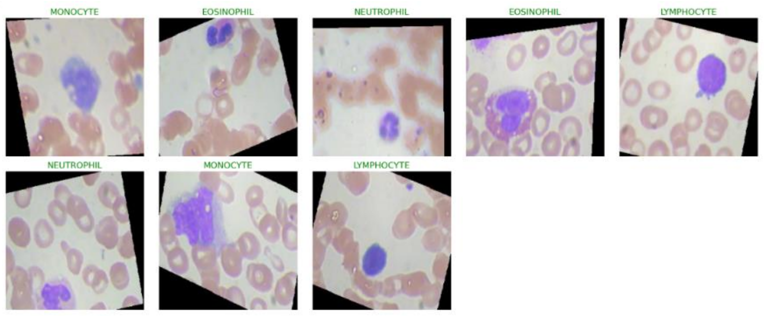
```
(7965, 2)
(2988, 2)
(1992, 2)
(6969, 2)
```

```
image_gen=ImageDataGenerator(preprocessing_function=tf.keras.applications.mobilenet_v2.preprocess_input)
train=image_gen.flow_from_dataframe(dataframe=train_set,x_col="filepaths",y_col="labels",target_size=(244,244),color_mode='rgb',class_mode="categoric
test=image_gen.flow_from_dataframe(dataframe=test_images,x_col="filepaths",y_col="labels",target_size=(244,244),color_mode='rgb',class_mode="categori
val=image_gen.flow_from_dataframe(dataframe=val_set,x_col="filepaths",y_col="labels",target_size=(244,244),color_mode='rgb',class_mode="categorical",
```

```
Found 7965 validated image filenames belonging to 4 classes.
Found 2988 validated image filenames belonging to 4 classes.
Found 1992 validated image filenames belonging to 4 classes.
```



```python
def show_knee_images(image_gen):
    test_dict = test.class_indices
    classes = list(test_dict.keys())

    images, labels = next(image_gen)

    plt.figure(figsize=(20, 20))

    length = len(images)

    r = min(25, length)

    for i in range(r):
        plt.subplot(5, 5, i + 1)
        image = (images[i] + 1) / 2
        plt.imshow(image)

        if labels[i].ndim > 0:
            index = np.argmax(labels[i])
        else:
            index = int(labels[i])

        class_name = classes[index]
        plt.title(class_name, color="green", fontsize=16)
        plt.axis('off')

    plt.tight_layout()
    plt.show()

show_knee_images(train)
```



Feature Engineering	Attached the codes in final submission.
Save Processed Data	-