Machine Learning Project

Phase 2

Name	BN	Section
Ahmed Mohmed Ahmed Lotfy	8	1

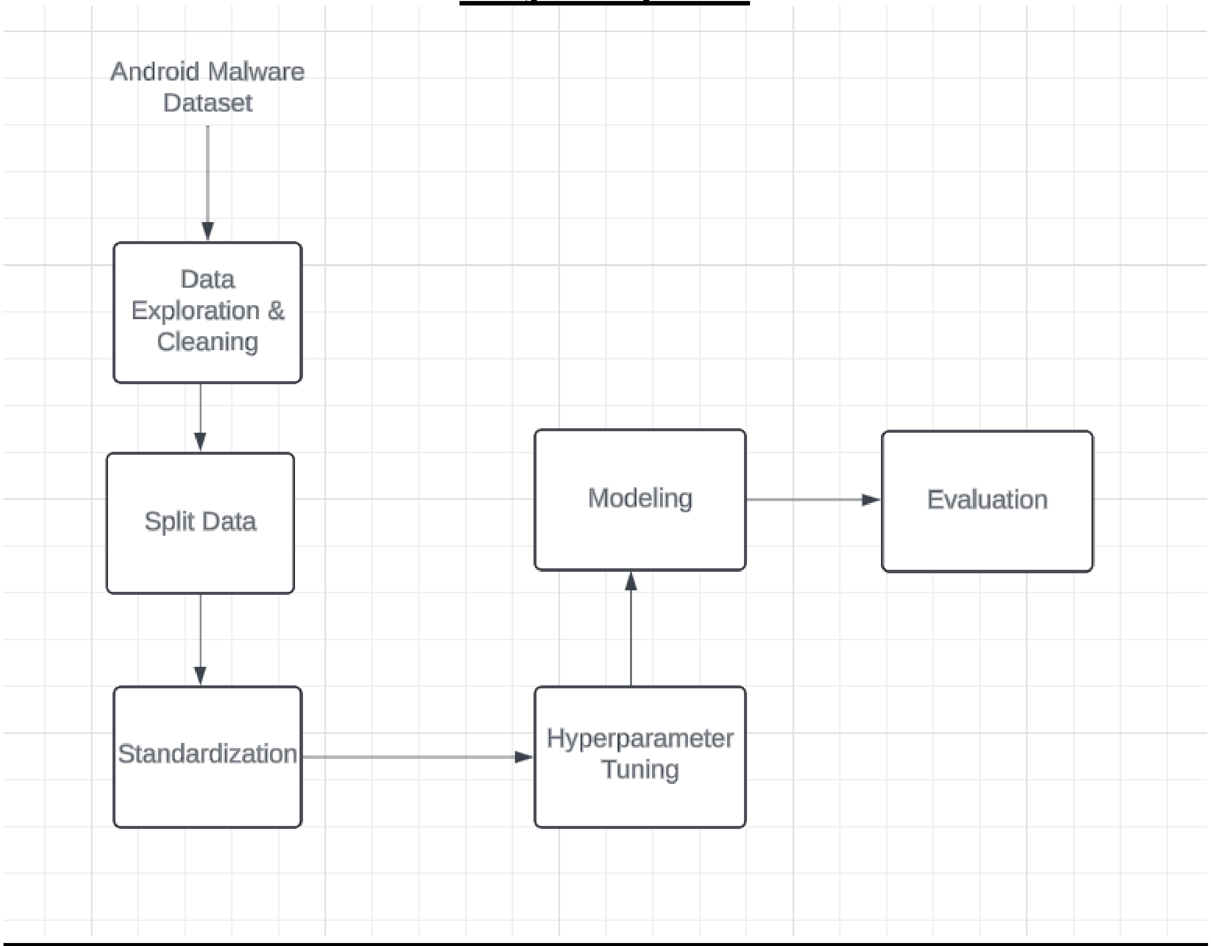
Problem statement

The widespread adoption of smartphones has made them a prime target for malicious software (malware). Android, the dominant mobile operating system, faces a significant threat from malware that can steal user data, disrupt phone functionality, or even cause financial harm. Developing effective methods for detecting and preventing this malware is crucial for protecting users and ensuring a safe mobile environment.

Dataset

Link: Android Malware Dataset

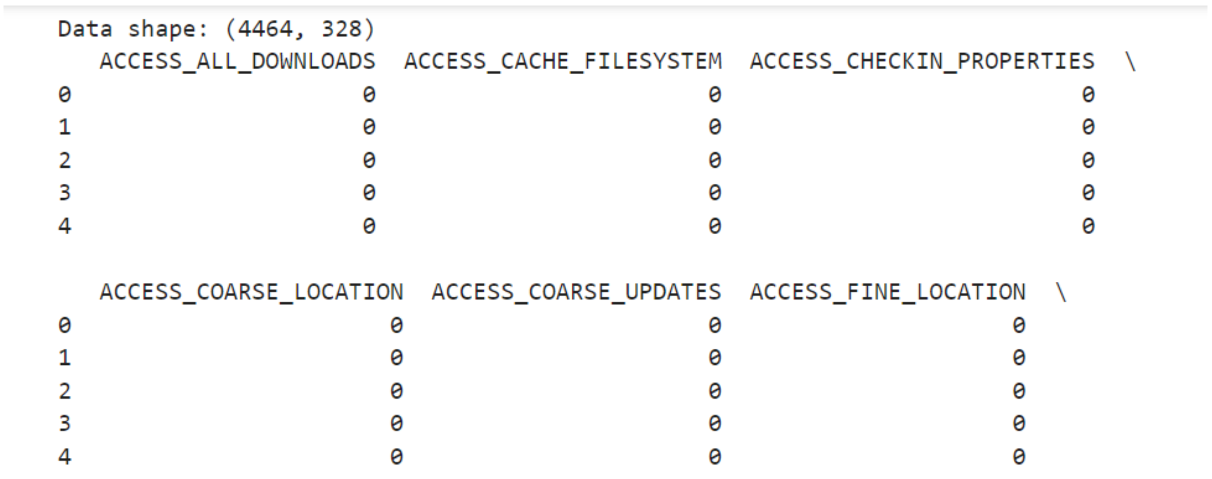
Project Pipeline



Exploration

4. Exploratory Data Analysis (EDA):

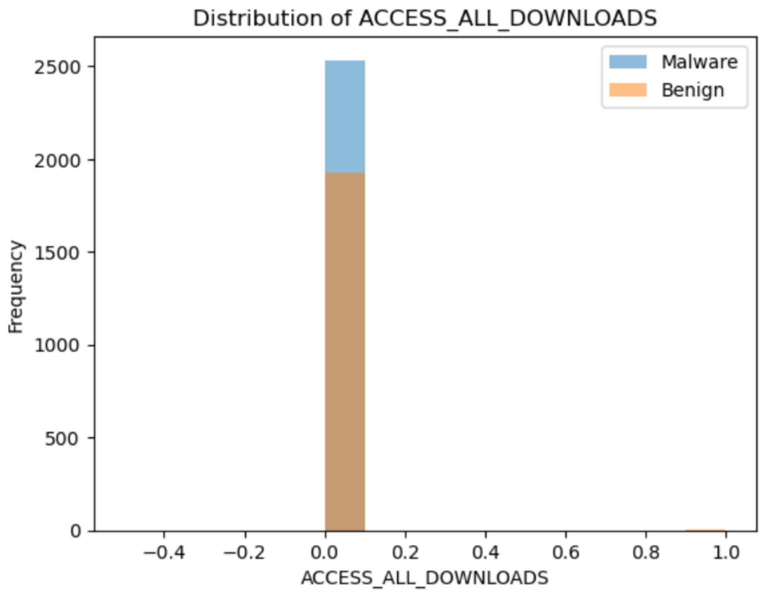
- **Data Shape:** Prints the shape of the dataset (number of rows and columns).
- **Head of the Data:** Displays the first few rows of the dataset.
- **Data Types:** Prints the data types of each column in the dataset.



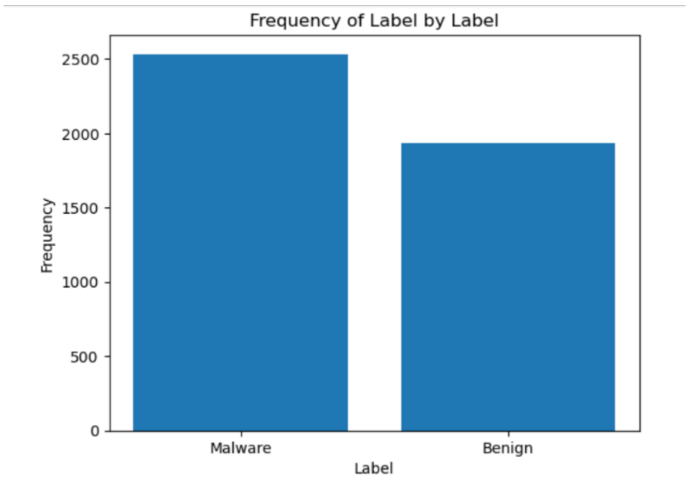
```
Data shape: (4464, 328)
   ACCESS_ALL_DOWNLOADS  ACCESS_CACHE_FILESYSTEM  ACCESS_CHECKIN_PROPERTIES  \
0                     0                        0                          0
1                     0                        0                          0
2                     0                        0                          0
3                     0                        0                          0
4                     0                        0                          0

   ACCESS_COARSE_LOCATION  ACCESS_COARSE_UPDATES  ACCESS_FINE_LOCATION  \
0                       0                      0                     0
1                       0                      0                     0
2                       0                      0                     0
3                       0                      0                     0
4                       0                      0                     0
```

5. Visualizing Data Distributions:

- **Continuous Features:** Histograms are plotted for each continuous feature, showing the distribution of values for both "Malware" and "Benign" labels.
  For example,

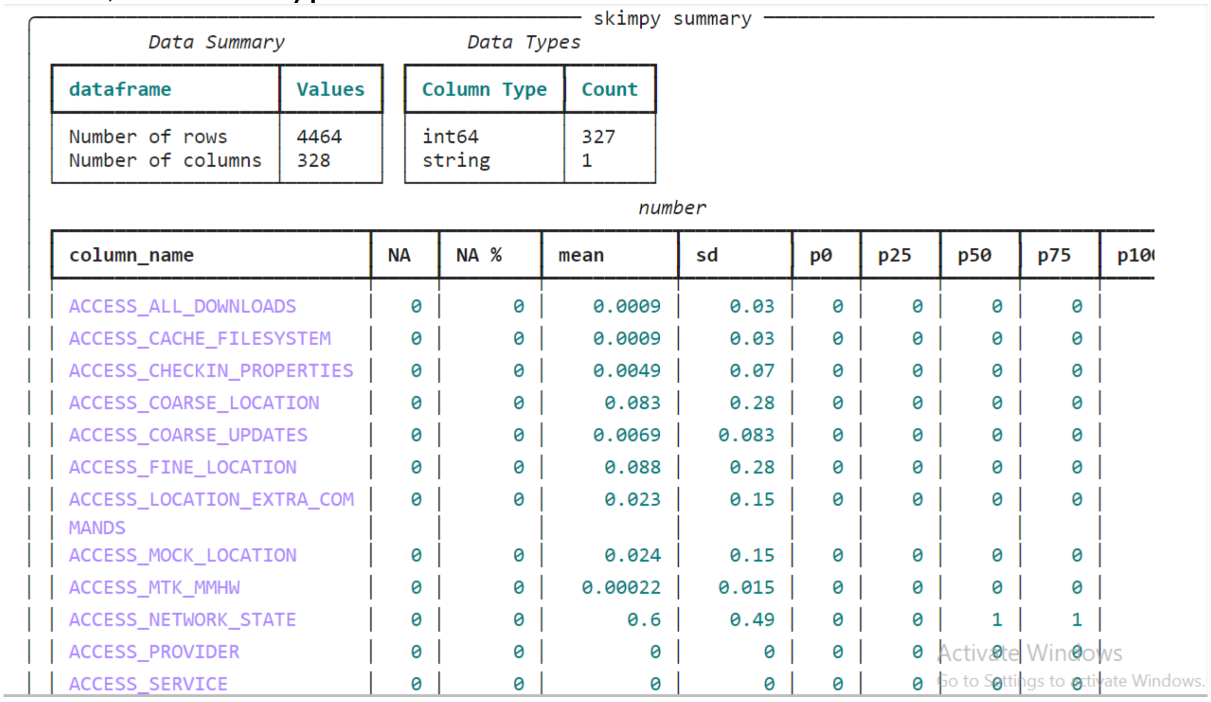


- **Categorical Features:** Bar plots are created to visualize the frequency of each category for categorical features, segmented by the "Label".



Malware values is greater than Bengin values in the dataset.

6. Data Summary: The **skim()** function from **pandas_profiling** generates a comprehensive summary of the dataset, including statistics, missing values, and data types.

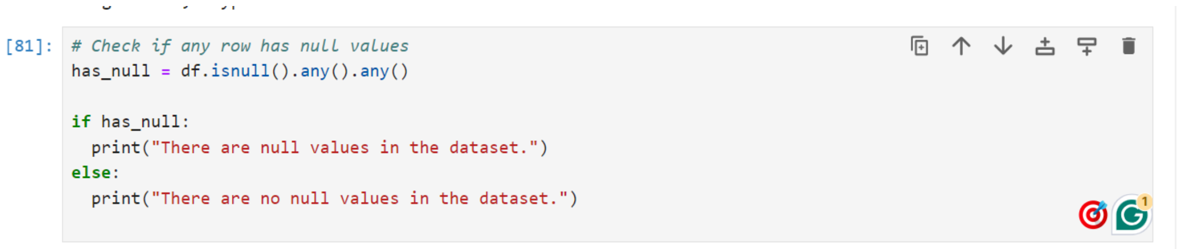


```
──────────────────────────────── skimpy summary ────────────────────────────────
         Data Summary                      Data Types
 ┏━━━━━━━━━━━━━━━━━━━┳━━━━━━━━┓  ┏━━━━━━━━━━━━━┳━━━━━━━┓
 ┃ dataframe         ┃ Values ┃  ┃ Column Type ┃ Count ┃
 ┡━━━━━━━━━━━━━━━━━━━╇━━━━━━━━┩  ┡━━━━━━━━━━━━━╇━━━━━━━┩
 │ Number of rows    │ 4464   │  │ int64       │ 327   │
 │ Number of columns │ 328    │  │ string      │ 1     │
 └───────────────────┴────────┘  └─────────────┴───────┘
                                       number
 ┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━┳━━━━━━┳━━━━━━━━━━┳━━━━━━━━┳━━━━━┳━━━━━┳━━━━━┳━━━━━┳━━━━━
 ┃ column_name                  ┃ NA  ┃ NA % ┃ mean     ┃ sd     ┃ p0  ┃ p25 ┃ p50 ┃ p75 ┃ p10
 ┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━╇━━━━━━╇━━━━━━━━━━╇━━━━━━━━╇━━━━━╇━━━━━╇━━━━━╇━━━━━╇━━━━━
 │ ACCESS_ALL_DOWNLOADS         │   0 │    0 │   0.0009 │   0.03 │   0 │   0 │   0 │   0 │
 │ ACCESS_CACHE_FILESYSTEM      │   0 │    0 │   0.0009 │   0.03 │   0 │   0 │   0 │   0 │
 │ ACCESS_CHECKIN_PROPERTIES    │   0 │    0 │   0.0049 │   0.07 │   0 │   0 │   0 │   0 │
 │ ACCESS_COARSE_LOCATION       │   0 │    0 │    0.083 │   0.28 │   0 │   0 │   0 │   0 │
 │ ACCESS_COARSE_UPDATES        │   0 │    0 │   0.0069 │  0.083 │   0 │   0 │   0 │   0 │
 │ ACCESS_FINE_LOCATION         │   0 │    0 │    0.088 │   0.28 │   0 │   0 │   0 │   0 │
 │ ACCESS_LOCATION_EXTRA_COM    │   0 │    0 │    0.023 │   0.15 │   0 │   0 │   0 │   0 │
 │ MANDS                        │     │      │          │        │     │     │     │     │
 │ ACCESS_MOCK_LOCATION         │   0 │    0 │    0.024 │   0.15 │   0 │   0 │   0 │   0 │
 │ ACCESS_MTK_MMHW              │   0 │    0 │  0.00022 │  0.015 │   0 │   0 │   0 │   0 │
 │ ACCESS_NETWORK_STATE         │   0 │    0 │      0.6 │   0.49 │   0 │   0 │   1 │   1 │
 │ ACCESS_PROVIDER              │   0 │    0 │        0 │      0 │   0 │   0 │   0 │   0 │
 │ ACCESS_SERVICE               │   0 │    0 │        0 │      0 │   0 │   0 │   0 │   0 │
```

From this summary we can see some features have zero standard deviation which will cause a NaN correlation with the label so we will remove them.

7. Handling Missing Values:

- Null values are identified using the **isnull()** function, and the sum of null values for each column is printed.
- Missing values in numeric features are filled with the mean value of the respective column using **fillna()** function.
- Missing values in categorical features are filled with the mode (most frequent value) of the respective column using **fillna()** function.


[81]:
```
# Check if any row has null values
has_null = df.isnull().any().any()

if has_null:
  print("There are null values in the dataset.")
else:
  print("There are no null values in the dataset.")
```

By the way the dataset does not have any nulls.

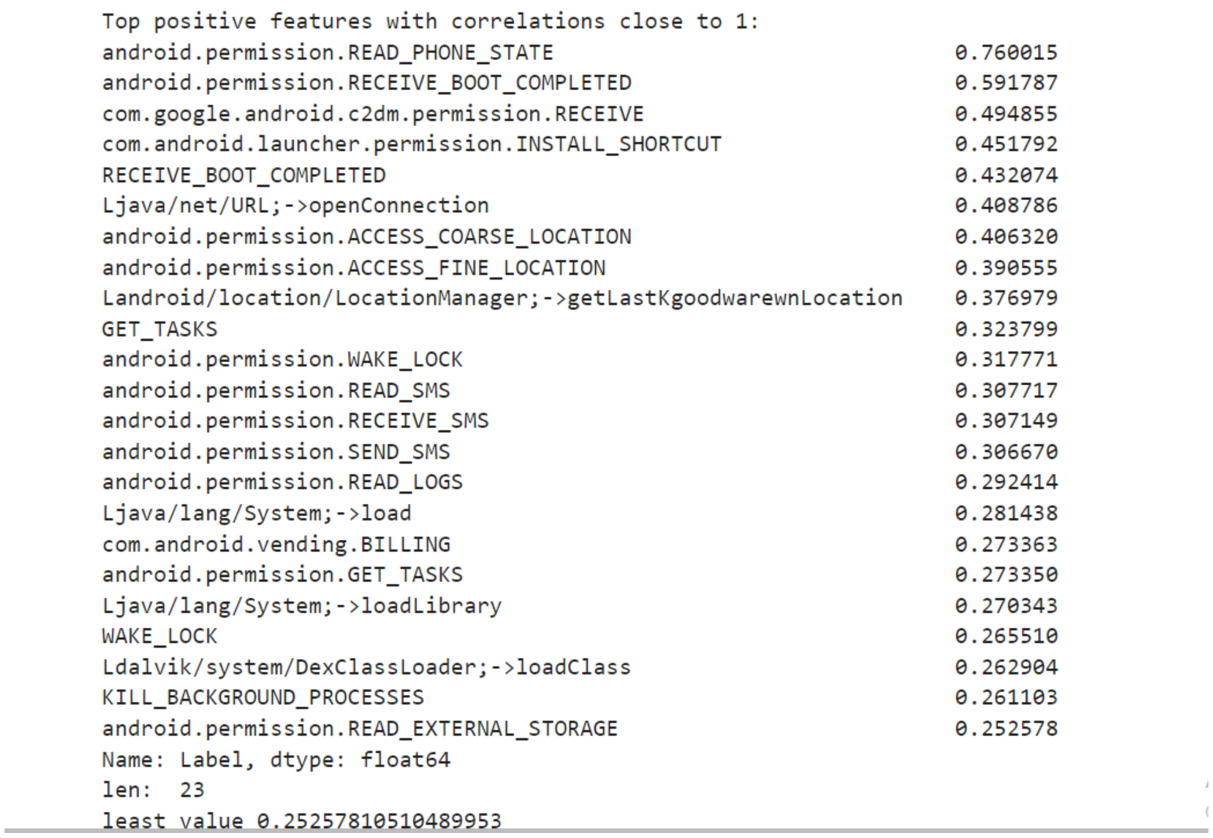
8. Outlier Detection and Removal: The **detect_and_remove_outliers()** function detects outliers using the Interquartile Range (IQR) method for each feature (excluding the target label). Outliers are visualized using boxplots and optionally removed from the dataset.

Feature Selection and Correlation Analysis

1. **Label Encoding: The "Label" column in the dataset is encoded using label encoding. This converts the categorical labels ("Malware" and "Benign") into numerical values (0 and 1). This transformation enables machine learning algorithms to work with categorical data.**

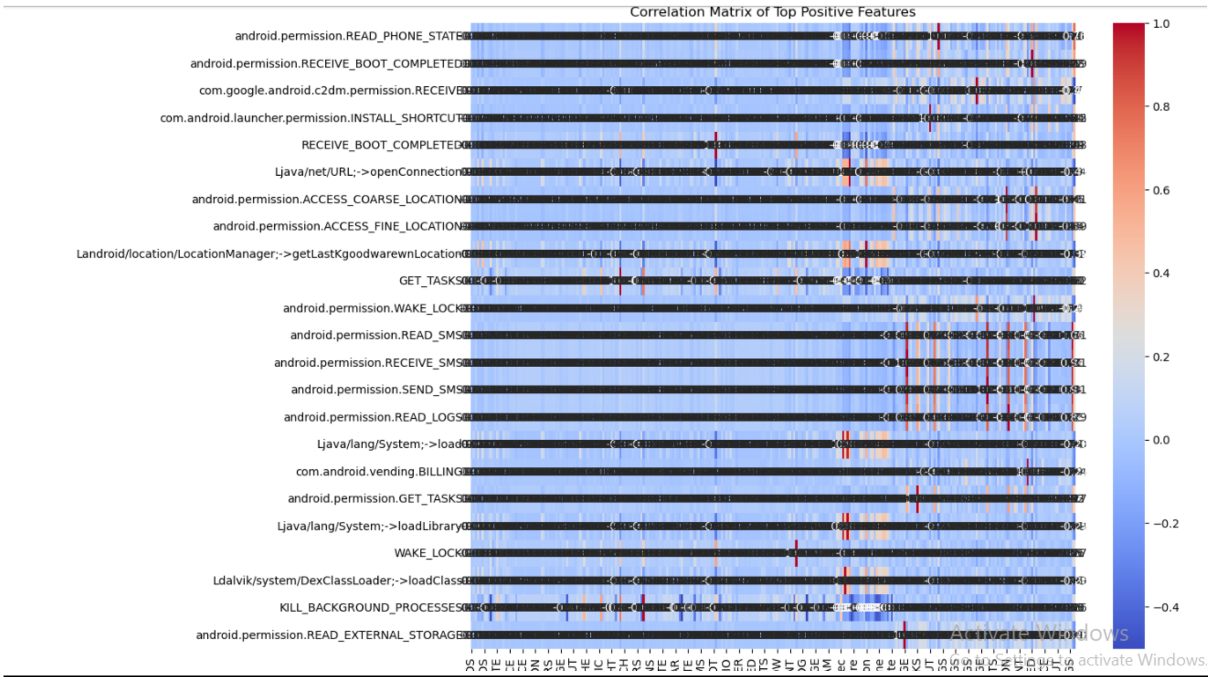
2. **Removing Columns with Zero Standard Deviation:** Columns with zero standard deviation are identified and removed from the dataset. Columns with zero standard deviation indicate that all values in that column are the same, providing no variability. These columns are not informative for predictive modeling and are thus dropped from the dataset.

3. **Correlation Analysis:**
   - **I got the correlation values near +1 and near -1 but there is no correlation values near to -1.**



```
Top positive features with correlations close to 1:
android.permission.READ_PHONE_STATE                                 0.760015
android.permission.RECEIVE_BOOT_COMPLETED                           0.591787
com.google.android.c2dm.permission.RECEIVE                          0.494855
com.android.launcher.permission.INSTALL_SHORTCUT                    0.451792
RECEIVE_BOOT_COMPLETED                                              0.432074
Ljava/net/URL;->openConnection                                      0.408786
android.permission.ACCESS_COARSE_LOCATION                           0.406320
android.permission.ACCESS_FINE_LOCATION                             0.390555
Landroid/location/LocationManager;->getLastKgoodwarewnLocation      0.376979
GET_TASKS                                                           0.323799
android.permission.WAKE_LOCK                                        0.317771
android.permission.READ_SMS                                         0.307717
android.permission.RECEIVE_SMS                                      0.307149
android.permission.SEND_SMS                                         0.306670
android.permission.READ_LOGS                                        0.292414
Ljava/lang/System;->load                                            0.281438
com.android.vending.BILLING                                         0.273363
android.permission.GET_TASKS                                        0.273350
Ljava/lang/System;->loadLibrary                                     0.270343
WAKE_LOCK                                                           0.265510
Ldalvik/system/DexClassLoader;->loadClass                           0.262904
KILL_BACKGROUND_PROCESSES                                           0.261103
android.permission.READ_EXTERNAL_STORAGE                            0.252578
Name: Label, dtype: float64
len:  23
least value 0.25257810510489953
```

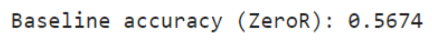
Correlation Matrix:



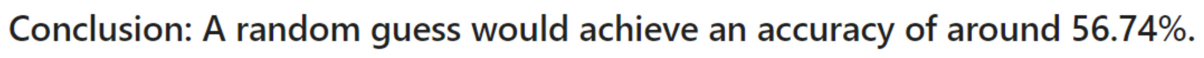
The correlation between features and each other is weak as provided above.

Model Development and Hyperparameter Tuning

Baseline Model (ZeroR): A baseline model using ZeroR algorithm is trained to establish a simple benchmark for comparison. The ZeroR algorithm predicts the most frequent class in the dataset for all instances.



```
Baseline accuracy (ZeroR): 0.5674
```



Conclusion: A random guess would achieve an accuracy of around 56.74%.

Standardization:

- **Features are scaled using standardization to ensure all features have the same scale. This is performed using scikit-learn's StandardScaler.**

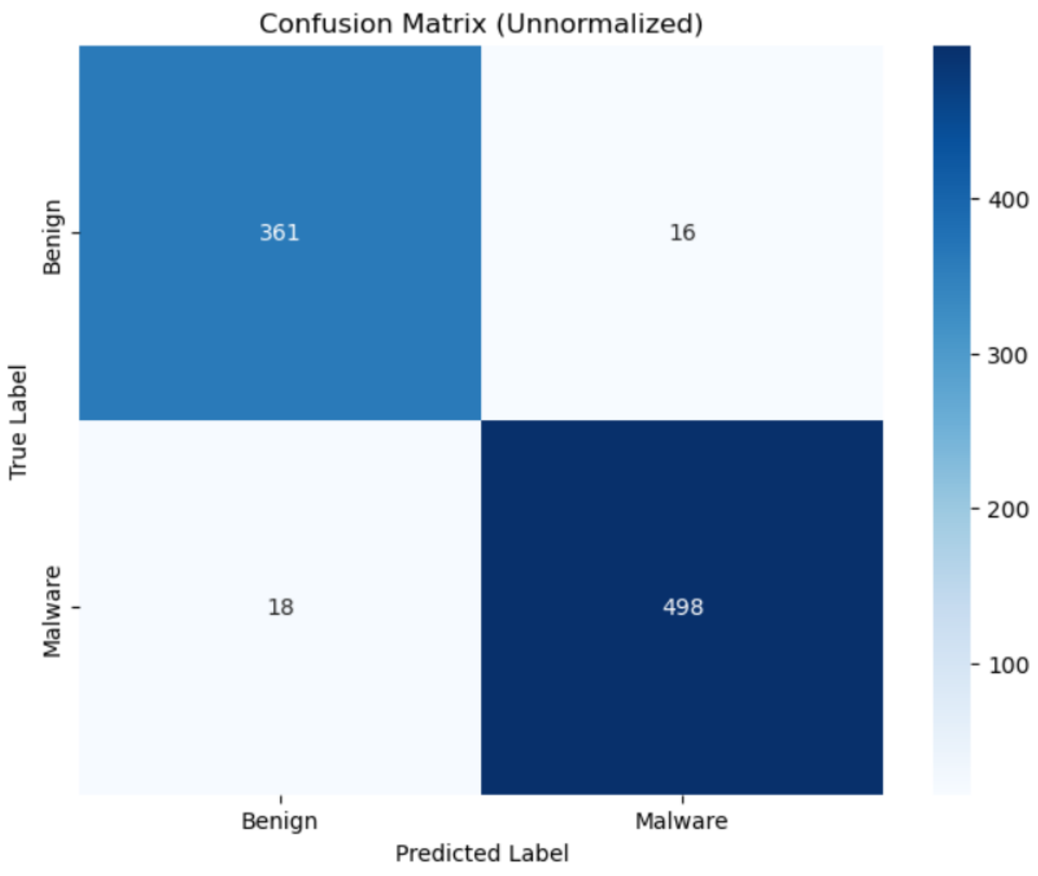
Hyperparameter Tuning

- Support Vector Machine (SVM) models with different kernels (linear, radial basis function (RBF), and polynomial) are trained and tuned using GridSearchCV for hyperparameter optimization.
- For each kernel, a parameter grid is defined containing different values for regularization parameters and kernel-specific parameters (e.g., gamma for RBF kernel, degree for polynomial kernel).
- A 5-fold stratified cross-validation is used to evaluate model performance during hyperparameter tuning.
- The best estimator and corresponding hyperparameters are stored for each kernel.

Evaluation:

- Predict the test set and get the evaluation metrics:

Confusion Matrix:



```
Accuracy: 0.9619
Precision: 0.9689
Recall: 0.9651
F1-Score: 0.9670
```

Thank You