**TRANSFORMER FAILURE CLASSIFICATION USING ENSEMBLE MACHINE LEARNING TECHNIQUE**

**30% code**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.preprocessing import LabelEncoder

from sklearn.model\_selection import train\_test\_split

import joblib

import os

from sklearn.metrics import accuracy\_score

from sklearn.metrics import precision\_score

from sklearn.metrics import recall\_score

from sklearn.metrics import f1\_score

from sklearn.metrics import classification\_report

from sklearn.metrics import confusion\_matrix

data = pd.read\_csv('merged\_dataset.csv')

data

data.info()

data.describe

data.isnull().sum()

data.duplicated().sum()

data.corr()

sns.heatmap(data.corr(), annot = True)

x = data.drop(['Output (S)'],axis=1)

x

y = data['Output (S)']

y

sns.countplot(x=y)

**Existing System**

The traditional system of transformer failure detection relies on manual inspections and the monitoring of electrical parameters such as voltage, current, and oil levels. These inspections are carried out periodically, with oil testing and visual assessments of the transformer’s condition. Offline diagnostic tests, such as dissolved gas analysis (DGA), help identify issues in transformer insulation but are time-consuming and cannot provide real-time feedback. When a fault occurs, the system often depends on threshold-based alarms to notify operators of abnormal conditions, which do not always indicate a failure in the early stages.

**Drawbacks of the Existing System**

* **Manual Dependence** – Inspections and fault detection are heavily reliant on human judgment, which introduces subjectivity and potential errors.
* **Time-Consuming** – Periodic inspections and diagnostic tests take time, leading to delays in detecting and responding to faults.
* **Limited Data Processing** – Traditional methods struggle to process large volumes of real-time data generated by modern transformers.
* **Inability to Detect Early-Stage Faults** – Many faults remain undetected until they reach a critical stage, leading to costly repairs and downtime.
* **Lack of Predictive Insights** – There is no proactive system to predict faults based on historical performance data, making it difficult to prevent failures.
* **Manual Fault Classification** – When faults occur, classification is typically done based on simple tests, which may not capture complex fault patterns accurately.
* **No Real-Time Monitoring** – Transformers are not continuously monitored, so failures can occur without any immediate awareness, leading to disruptions in the grid.
* **Inefficient Use of Resources** – Operators spend significant time on manual checks, diverting resources from other critical grid management tasks.

**PROPOSED METHODOLOGY**

**SVM classifier**

The SVM Classifier is a powerful supervised learning algorithm used for classification tasks. It works by finding an optimal hyperplane that best separates data points into different classes. SVM is particularly effective for complex, high-dimensional datasets.

**How SVM Performs**

1. **Finds the Optimal Hyperplane**
   * SVM looks for the **maximum margin** (widest possible gap) between different classes.
   * It selects **support vectors** (critical data points) to define the boundary.
2. **Kernel Trick for Non-Linear Data**
   * If the data is **not linearly separable**, SVM applies the **kernel trick** to transform it into a higher-dimensional space where a linear boundary can be found.
   * Common kernel functions:
     + **Linear Kernel:** Suitable for linearly separable data.
     + **Polynomial Kernel:** Useful for more complex patterns.
     + **Radial Basis Function (RBF) Kernel:** Best for non-linearly separable data.
     + **Sigmoid Kernel:** Used in neural network-like structures.
3. **Regularization (C Parameter)**
   * Controls trade-off between **margin size** and **misclassification**.
   * A small **C** allows a larger margin with some misclassifications.
   * A large **C** focuses on correctly classifying every point but may lead to **overfitting**.

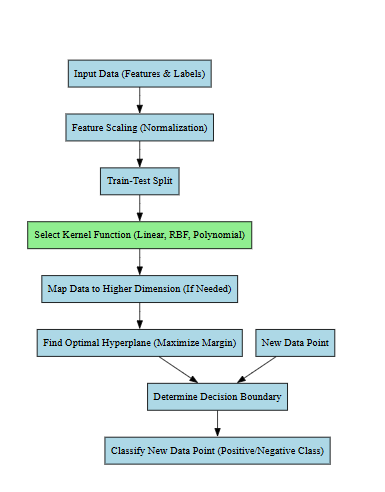


Fig 1 Block diagram for SVM classifier

**Random forest classifier**

The Random Forest Classifier is an ensemble learning method that combines multiple decision trees to improve classification accuracy and robustness. It is widely used due to its high performance, scalability, and ability to handle complex datasets.

**How Random Forest Performs**

1. **Creates Multiple Decision Trees**
   * Instead of relying on a single decision tree, Random Forest builds **multiple decision trees** using different subsets of data and features.
   * Each tree is trained independently.
2. **Bootstrapping (Bagging Technique)**
   * Random Forest selects **random samples (with replacement)** from the dataset to train each tree.
   * This **reduces variance** and prevents overfitting.
3. **Random Feature Selection**
   * At each node, the algorithm picks a **random subset of features** to find the best split.
   * This introduces diversity and reduces correlation among trees.
4. **Majority Voting (For Classification)**
   * When making predictions, each tree votes for a class.
   * The **majority class** is selected as the final output.
5. **Aggregated Predictions (For Regression)**
   * Instead of voting, Random Forest **averages** the predictions of all trees to give the final result in regression tasks.

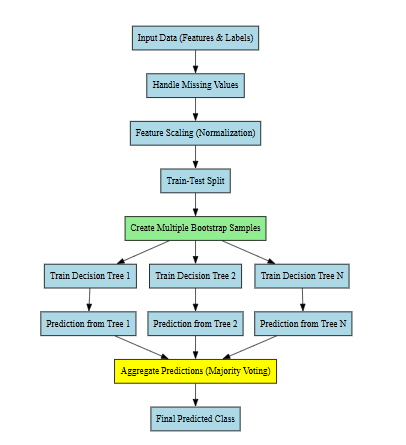


Fig 2 Block diagram for Random forest classifier

**Advantages of Random Forest**

✅ **High Accuracy**

* Combines multiple trees to reduce overfitting and improve generalization.

✅ **Handles Missing Data Well**

* Can maintain accuracy even with some missing values.

✅ **Works with Large Datasets**

* Efficient for big data applications.

✅ **Robust to Outliers and Noise**

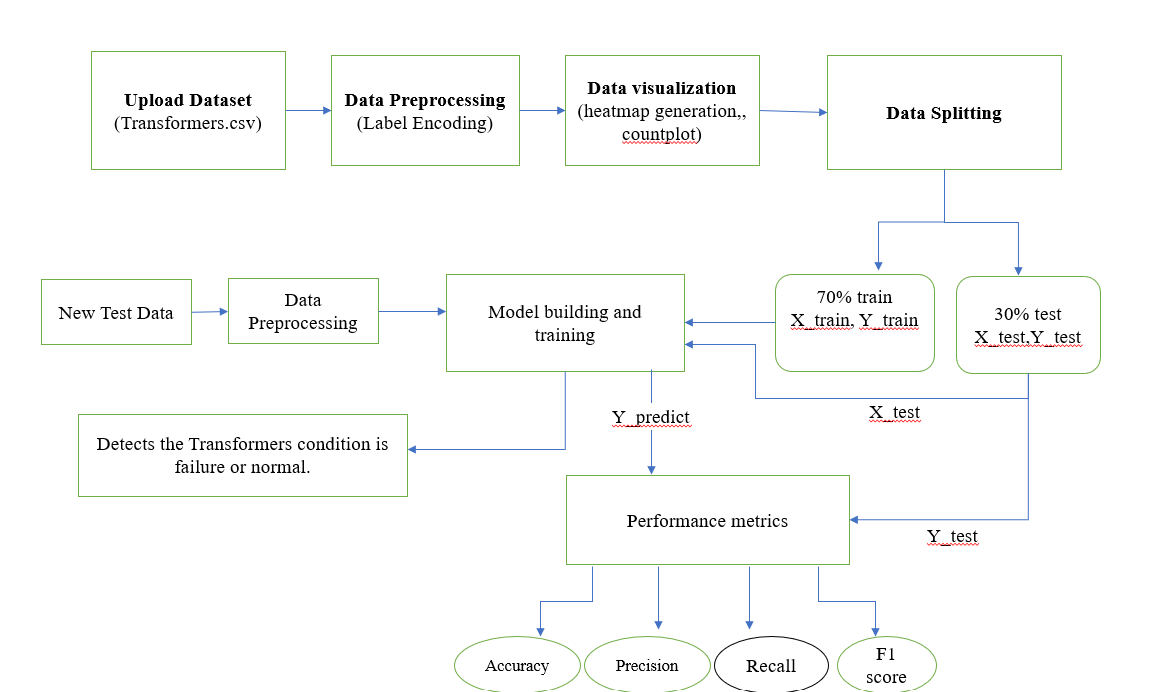
* Individual trees might be affected, but averaging their results makes the model more stable.

✅ **Handles Both Categorical & Numerical Data**

* Can be used for various types of data without major preprocessing.

✅ **Feature Importance Ranking**

* Can identify the most important features that contribute to predictions.

Figure 3 Proposed Block diagram

**Expected output**

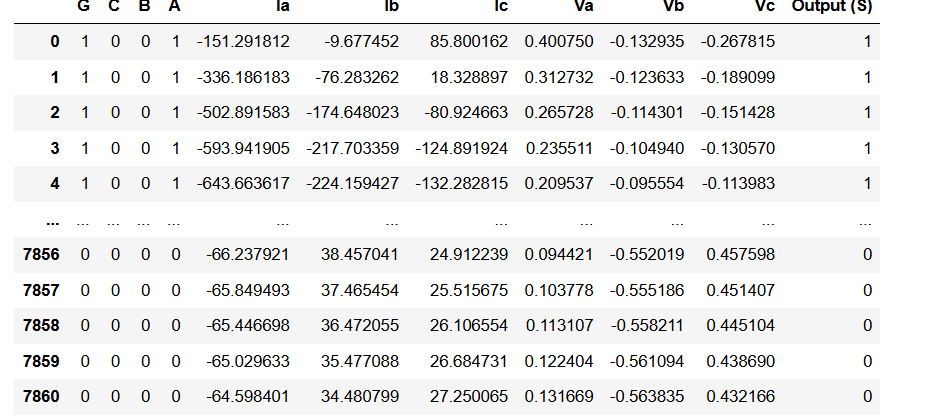


Fig 4 uploading dataset

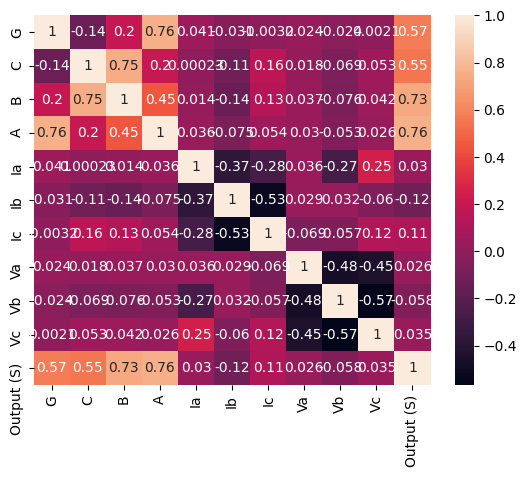


Fig 5 Heatmap

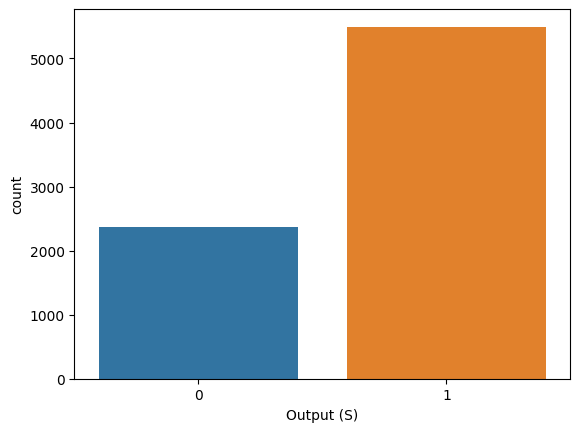


Fig 6 Countplot