DOI: 10.48047/IJIEMR/V11/I06/104

Title **ELECTRICITY THEFT DETECTION IN POWER GRIDS WITH DEEP LEARNING AND RANDOM FORESTS**

Volume 11, Issue 06, Pages: 1661-1669

Paper Authors
Mr.D.Rambabu, A.Srihitha, E.Gayathri

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

ELECTRICITY THEFT DETECTION IN POWER GRIDS WITH DEEP LEARNING AND RANDOM FORESTS

[1]Mr.D.Rambabu,[2]A.Srihitha,[3]E.Gayathri
[1]Assistant Professor, Department Of Cse, Sreenidhi Institute Of Science And Technology.
[2,3] UG Scholar, Department Of Cse, Sreenidhi Institute Of Science And Technology.

ABSTRACT
When electricity is stolen from distribution networks, non-technical losses (NTLs) have a negative impact on the quality of power supply and lower operating profits. Convolutional neural networks (CNNs) and random forests (RFs) are used in this research to help utility firms deal with the issues of wasteful electricity inspection and unpredictable power use. Attribute variations at different times of the day and on certain days can be learned using convolutional neural networks (CNN) that condense and down-sample vast amounts of data from smart meters During training, strategies such as back propagation and a dropout layer are used to prevent the network from being overfit. After then, the RF is taught to look for signs that a customer is attempting to steal electricity using the information it has gleaned. A grid search algorithm can be used to optimize the RF parameters of a hybrid model. To illustrate that the proposed detection strategy is more accurate and efficient than current methods, actual energy usage data is used.

1.INTRODUCTION

Purpose:

The reduction of energy loss in electricity transmission and distribution is a serious concern for power suppliers all over the world. One of the most prevalent ways energy is wasted is through technical and non-technical losses (NTLs). Fraudulent acts involving energy may result in lower profits for power firms.

The use of a CNN to extract features from high-resolution smart-meter data piques our interest in detecting electricity theft. This study uses convolutional neural networks (CNNs) to investigate how electricity is stolen from smart grids. As in a traditional single hidden layer feedforward neural network (SLFN), backpropagation is utilized to train the softmax classifier layer in a basic CNN. A decline in the SLFN's ability to generalize is possible when utilizing the backpropagation technique.

Scope:

The Goal of the Initiative Several methods exist for obtaining free electricity, causing the utility provider to suffer a loss of revenue. CNN data from smart meters is critical to the power theft detection model's operation since it automatically collects many aspects of customer usage behaviors. It is used as a replacement for the softmax classifier to improve detection performance. Customers' real-world data from Ireland and London's energy utilities was used to develop and test this model.

II.LITERATURE SURVEY

Technical and nontechnical losses in power system and its economic consequence in Indian economy

AUTHORS: J. P. Navani, N. K. Sharma, and S. Sapra,

ABSTRACT : Electricity shortages are a widespread problem in India, and they are only getting worse. These shortages have had a negative impact on the country's overall economic growth. The overall loss is equal to the sum of the distribution system's technical and non-technical losses. Large losses can be attributed to a lack of transmission and distribution (T&D), excessive transformation processes, inappropriate load distribution, and significant rural electrification. In the simplest terms, metered electricity losses can be described as the time-varying discrepancy between meter readings taken from the distribution network and those taken from other meters in the building. Technical losses in an electrical system include network impedance, current flows, and auxiliary power. Technical losses may be directly linked to network investment or network functioning. Thieves and unbilled accounts are just two of the many non-technical losses that might occur, as well as inaccuracies in metering and estimations of non-metered consumption. Analyzing power system losses through case studies and MATLAB simulations is one of the primary objectives of this work.

2.2 A framework for enhanced metering systems to detect energy theft using several sensors,

S. McLaughlin, B. Holbert, A. Fawaz, R. Berthier, and S. Zonouz are the authors of this book.

The smart grid's advanced metering infrastructure (AMI) replaces analog devices with computerized smart meters, making it a critical component of the system. As a result of the widespread adoption of smart meters, AMI has become a prime target for energy theft via remote attacks and local physical tampering. For the most part, the many sensors and data sources that smart meters use to detect energy theft are ineffective because they produce many false positives. Our AMIDS intrusion detection system leverages sensor and smart meter data fusion to better identify energy theft, and we describe it in this paper. AMIDS models and detects theft-related behavior more accurately by combining meter audit logs and consumption data from physical and cyber events. AMIDS is able to accurately detect energy theft based on our experiments with normal and abnormal load profiles. More fundamental analyses incorrectly labeled as malicious legitimate changes to the load profile that AMIDS accurately assessed as such.

Smart grid security and privacy issues P. McDaniel and S. McLaughlin are the authors.

This is one of the most significant shifts in the history of the electrical grid—the transition to smart grid technologies. With this new infrastructure, consumers and energy providers can better control and create electricity. The smart grid, like many new technologies, raises new questions about security. Ongoing smart grid installations around the world are examined here in terms of the reasons they are being implemented and the potential consequences of failures in security. Future projects may address the security issues that may be faced by future deployments.

Nontechnical loss detection using artificial intelligence is difficult, according to a new survey.

P. Glauner, J. A. Meira, P. Valtchev, R. State, and F. Bettinger are the authors of this article.

ABSTRACT: Researchers in electrical engineering and computer science are becoming increasingly interested in detecting non-technical losses (NTL), such as electricity theft, defective meters, or billing problems. NTLs have a considerable negative impact on the economy, as they account for up to 40% of the total electricity provided in some countries. There is a strong focus on using artificial intelligence to determine if a consumer is responsible for NTL. NTL definitions and economic effects, such as decreased revenue and profit for energy providers as well as decreased grid stability and reliability, are briefly discussed in this study. After that, it takes a look at the most recent and most thorough research on the methods, features, and data sets that have been used. There are still many scientific and engineering obstacles to overcome before NTL detection can be improved. State estimation and variance analysis for non-technical loss detection

III.SYSTEM ANALYSIS

EXISTING SYSTEM:

Existing machine learning methods include classification and clustering models. However, despite the uniqueness and impressiveness of the machine learning detection algorithms discussed above, their performance is still insufficient for practical use. Most of these methods need manual feature extraction due to their limited ability to handle high-dimensional data. It's true that the average consumption data includes the mean, standard deviation, maximum, and minimum. There are no manual methods for extracting the 2D properties of smart meter data because they are time-consuming and tiresome.

The existing system's drawbacks are detailed in this section.

The performances are still not excellent enough to allow for practice.

Time-consuming and can't gather 2D characteristics from smart meter data.

PROPOSED SYSTEM:

Utilities will be provided with a ranked list of its clients, according to the likelihood that they have an anomaly in their electric meters, through this process. Figure 1 depicts the three primary steps of the electricity theft detecting system:

To better understand why a CNN is employed for feature extraction, let's look at the factors that affect how people utilize electricity. Data cleaning (resolving outliers), missing value imputation, and data transformation are all part of data preprocessing (normalization).

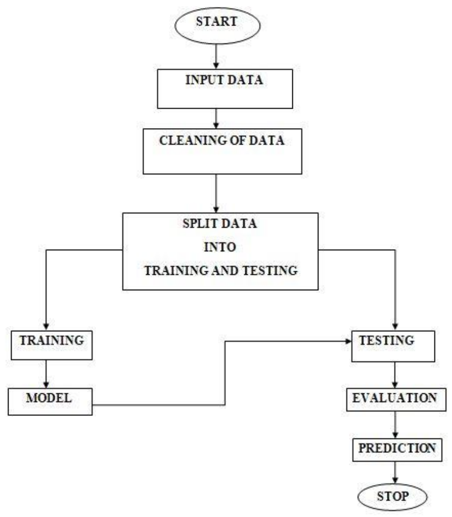
Using cross-validation, we partition the preprocessed dataset into train and test datasets to assess how well the methods in this paper perform.

WHY THE PROPOSED SYSTEM IS ADVANTAGEOUS

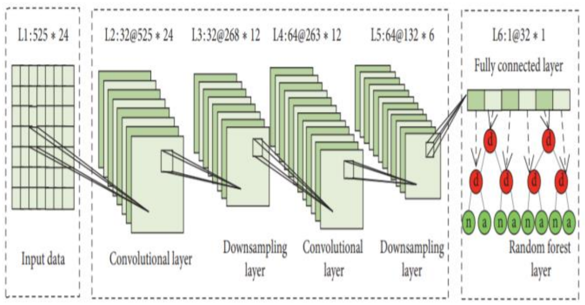
To determine whether or not a customer is stealing electricity.

IV.SYSTEM DESIGN

Block Diagram

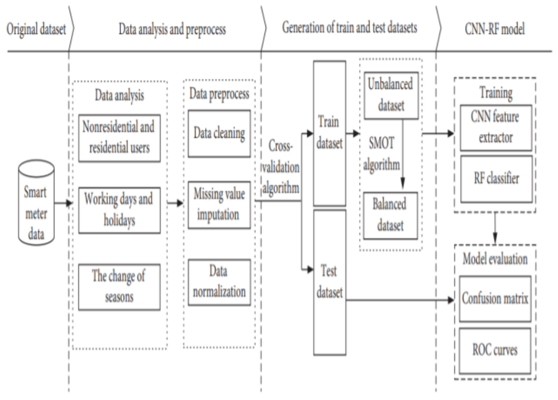


Architecture



Architecture of the hybrid CNN-RF model.

FLOWCHART



Flow of electricity theft detection.

IMPLEMENTATION:

MODULES:

The following steps are being taken by the author of the proposed paper:

Using this module, you may read the power consumption data.

2) Normalize and clean dataset: in this module, we will remove missing dataset and normalize the dataset.

(3) Train the CNN Model: Using this module, we'll develop a theft prediction model by training CNN using a dataset, then extracting trained features from CNN and feeding them into the random forest algorithm. We've implemented a DROPOUT layer to get rid of superfluous functionality.

To calculate precision, recall, FSCORE, and accuracy, we'll use this module to train Random Forest with CNN features.

Use this module to train SVM with CNN features, and then calculate precision, recall, FSCORE, and accuracy using the CNN features as inputs.

On a typical dataset, we trained a random forest without CNN features and then calculated precision, recall, FSCORE, and accuracy. 6) Train Random Forest without CNN:

Predictive modeling with SVM rather than CNN: We performed this analysis by first training the SVM on a typical dataset without incorporating CNN characteristics.

8) Graph of Comparison: we will use this to provide a graph of comparison of all methods.
In this module, we upload test data and CNN-RF makes a prediction about whether or not the records contain ENERGY THEFT.

Algorithms:

A single layer of a neural network with sigmoid activation is equal to a logistic regression model for binary classification. The sigmoid function that emerges from linear regression has a range of 0 to 1 points. Aberrant patterns have values less than or equal to 0, whereas normal patterns have values greater than or equal to 0.50.

Nonlinear separable problems are transformed into linear ones by projecting data into feature space and then finding the ideal separate hyperplane for the support vector machine classifier. The study uses the characteristics of handcrafted goods to predict consumer behavior.

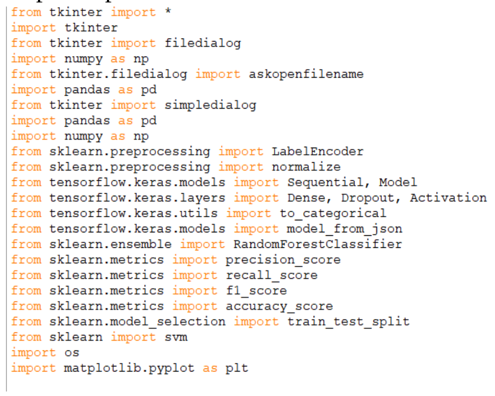
Overfitting can be minimized and performance improved by combining multiple decision trees into a single decision tree in a random forest. High-dimensional data can be handled by the RF classifier while yet preserving a high level of efficiency.

Gradient Boosting Decision Tree is an iterative strategy that consists of several decision trees. A majority vote is used in both random forests and the general Bayesian decision tree (GBDT), however the GBDT adds up each outcome or weight individually.

In contrast to the suggested method, CNN, CNN-GBDT, and CNN-SVM use softmax in the final layer.

V.Implementation and Results

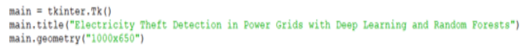
Step 1:import libraries:



```
from tkinter import *
import tkinter
from tkinter import filedialog
import numpy as np
from tkinter.filedialog import askopenfilename
import pandas as pd
from tkinter import simpledialog
import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import normalize
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.layers import Dense, Dropout, Activation
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.models import model_from_json
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import f1_score
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn import svm
import os
import matplotlib.pyplot as plt
```

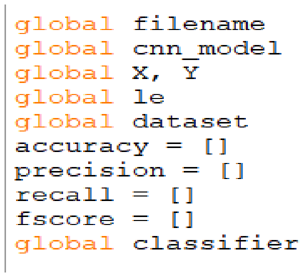
(where tkinter used for GUI(front end) ,pandas - data preprocessing, numpy - mathematical , matplotlib - data visualization , sklearn - machine learning , tensorflow and keras - machine learning and AI back end purpose)

Step 2: Defining the main function and setting the title & size of tkinter



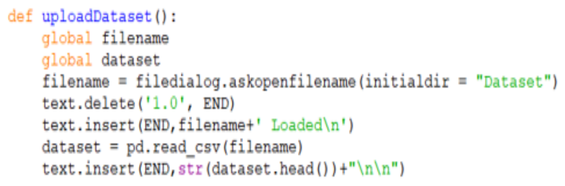
```
main = tkinter.Tk()
main.title("Electricity Theft Detection in Power Grids with Deep Learning and Random Forests")
main.geometry("1000x650")
```

Step 3: Defining the global function



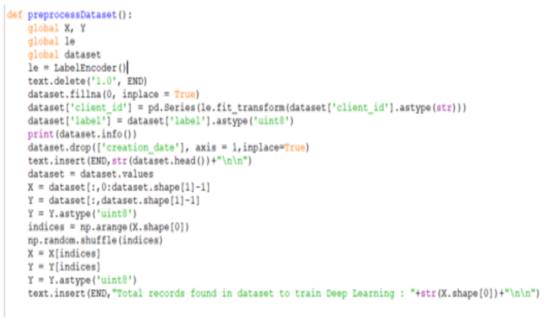
```
global filename
global cnn_model
global X, Y
global le
global dataset
accuracy = []
precision = []
recall = []
fscore = []
global classifier
```

Step 4: Defining the upload function



```
def uploadDataset():
    global filename
    global dataset
    filename = filedialog.askopenfilename(initialdir = "Dataset")
    text.delete('1.0', END)
    text.insert(END,filename+' Loaded\n')
    dataset = pd.read_csv(filename)
    text.insert(END,str(dataset.head())+"\n\n")
```

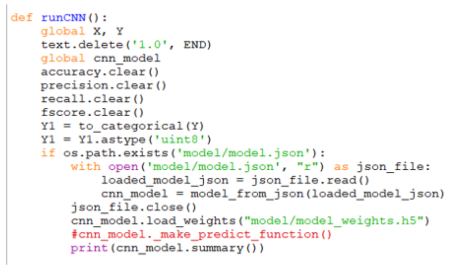
Step 5:Defining the preprocess Data set function



```
def preprocessDataset():
    global X, Y
    global le
    global dataset
    le = LabelEncoder()
    text.delete('1.0', END)
    dataset.fillna(0, inplace = True)
    dataset['client_id'] = pd.Series(le.fit_transform(dataset['client_id'].astype(str)))
    dataset['label'] = dataset['label'].astype('uint8')
    print(dataset.info())
    dataset.drop(['creation_date'], axis = 1,inplace=True)
    text.insert(END,str(dataset.head())+"\n\n")
    dataset = dataset.values
    X = dataset[:,0:dataset.shape[1]-1]
    Y = dataset[:,dataset.shape[1]-1]
    Y = Y.astype('uint8')
    indices = np.arange(X.shape[0])
    np.random.shuffle(indices)
    X = X[indices]
    Y = Y[indices]
    Y = Y.astype('uint8')
    text.insert(END,"Total records found in dataset to train Deep Learning : "+str(X.shape[0])+"\n\n")
```

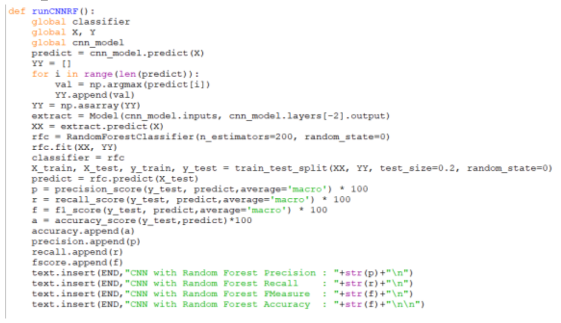
Model building

Step 6: CNN



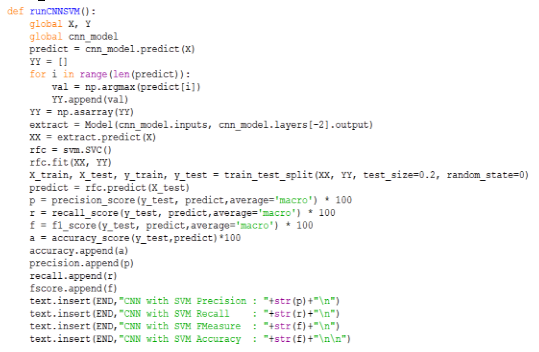
```
def runCNN():
    global X, Y
    text.delete('1.0', END)
    global cnn_model
    accuracy.clear()
    precision.clear()
    recall.clear()
    fscore.clear()
    Y1 = to_categorical(Y)
    Y1 = Y1.astype('uint8')
    if os.path.exists('model/model.json'):
        with open('model/model.json', "r") as json_file:
            loaded_model_json = json_file.read()
            cnn_model = model_from_json(loaded_model_json)
        json_file.close()
        cnn_model.load_weights("model/model_weights.h5")
        #cnn_model._make_predict_function()
        print(cnn_model.summary())
```

Step 7: CNNRF



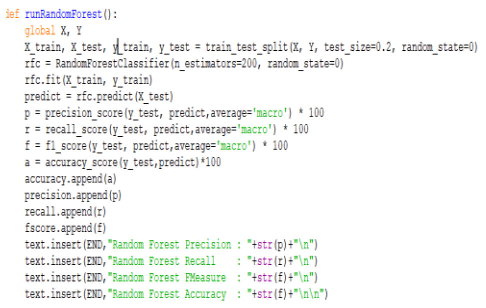
```
def runCNNRF():
    global classifier
    global X, Y
    global cnn_model
    predict = cnn_model.predict(X)
    YY = []
    for i in range(len(predict)):
        val = np.argmax(predict[i])
        YY.append(val)
    YY = np.asarray(YY)
    extract = Model(cnn_model.inputs, cnn_model.layers[-2].output)
    XX = extract.predict(X)
    rfc = RandomForestClassifier(n_estimators=200, random_state=0)
    rfc.fit(XX, YY)
    classifier = rfc
    X_train, X_test, y_train, y_test = train_test_split(XX, YY, test_size=0.2, random_state=0)
    predict = rfc.predict(X_test)
    p = precision_score(y_test, predict,average='macro') * 100
    r = recall_score(y_test, predict,average='macro') * 100
    f = f1_score(y_test, predict,average='macro') * 100
    a = accuracy_score(y_test,predict)*100
    accuracy.append(a)
    precision.append(p)
    recall.append(r)
    fscore.append(f)
    text.insert(END,"CNN with Random Forest Precision : "+str(p)+"\n")
    text.insert(END,"CNN with Random Forest Recall    : "+str(r)+"\n")
    text.insert(END,"CNN with Random Forest FMeasure  : "+str(f)+"\n")
    text.insert(END,"CNN with Random Forest Accuracy  : "+str(f)+"\n\n")
```

Step 8: CNNSVM



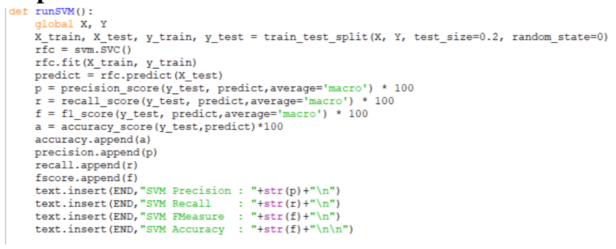
```
def runCNNSVM():
    global X, Y
    global cnn_model
    predict = cnn_model.predict(X)
    YY = []
    for i in range(len(predict)):
        val = np.argmax(predict[i])
        YY.append(val)
    YY = np.asarray(YY)
    extract = Model(cnn_model.inputs, cnn_model.layers[-2].output)
    XX = extract.predict(X)
    rfc = svm.SVC()
    rfc.fit(XX, YY)
    X_train, X_test, y_train, y_test = train_test_split(XX, YY, test_size=0.2, random_state=0)
    predict = rfc.predict(X_test)
    p = precision_score(y_test, predict,average='macro') * 100
    r = recall_score(y_test, predict,average='macro') * 100
    f = f1_score(y_test, predict,average='macro') * 100
    a = accuracy_score(y_test,predict)*100
    accuracy.append(a)
    precision.append(p)
    recall.append(r)
    fscore.append(f)
    text.insert(END,"CNN with SVM Precision : "+str(p)+"\n")
    text.insert(END,"CNN with SVM Recall    : "+str(r)+"\n")
    text.insert(END,"CNN with SVM FMeasure  : "+str(f)+"\n")
    text.insert(END,"CNN with SVM Accuracy  : "+str(f)+"\n\n")
```

Step 9:Random Forest



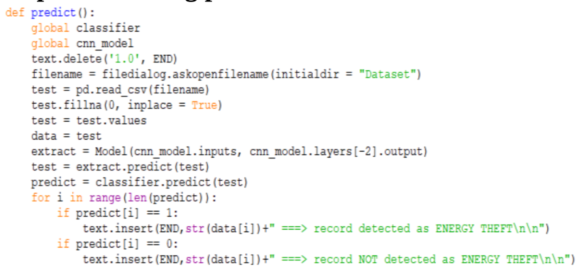
```
def runRandomForest():
    global X, Y
    X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2, random_state=0)
    rfc = RandomForestClassifier(n_estimators=200, random_state=0)
    rfc.fit(X_train, y_train)
    predict = rfc.predict(X_test)
    p = precision_score(y_test, predict,average='macro') * 100
    r = recall_score(y_test, predict,average='macro') * 100
    f = f1_score(y_test, predict,average='macro') * 100
    a = accuracy_score(y_test,predict)*100
    accuracy.append(a)
    precision.append(p)
    recall.append(r)
    fscore.append(f)
    text.insert(END,"Random Forest Precision : "+str(p)+"\n")
    text.insert(END,"Random Forest Recall    : "+str(r)+"\n")
    text.insert(END,"Random Forest FMeasure  : "+str(f)+"\n")
    text.insert(END,"Random Forest Accuracy  : "+str(f)+"\n\n")
```

Step 10: SVM



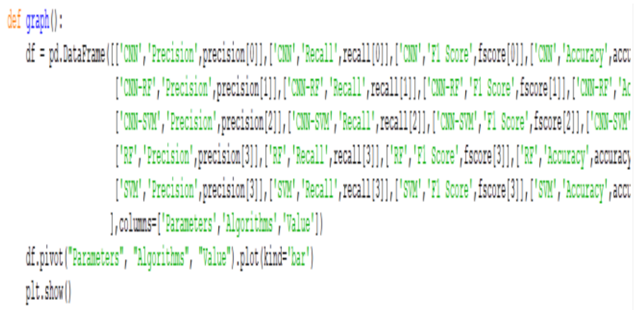
```
def runSVM():
    global X, Y
    X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2, random_state=0)
    rfc = svm.SVC()
    rfc.fit(X_train, y_train)
    predict = rfc.predict(X_test)
    p = precision_score(y_test, predict,average='macro') * 100
    r = recall_score(y_test, predict,average='macro') * 100
    f = f1_score(y_test, predict,average='macro') * 100
    a = accuracy_score(y_test,predict)*100
    accuracy.append(a)
    precision.append(p)
    recall.append(r)
    fscore.append(f)
    text.insert(END,"SVM Precision : "+str(p)+"\n")
    text.insert(END,"SVM Recall    : "+str(r)+"\n")
    text.insert(END,"SVM FMeasure  : "+str(f)+"\n")
    text.insert(END,"SVM Accuracy  : "+str(f)+"\n\n")
```

Step 11:Defining prediction function



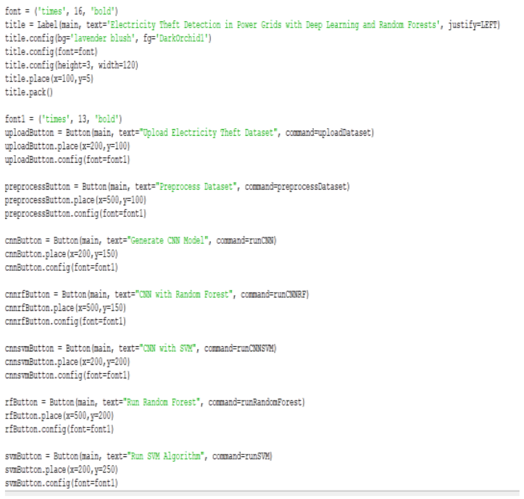
```
def predict():
    global classifier
    global cnn_model
    text.delete('1.0', END)
    filename = filedialog.askopenfilename(initialdir = "Dataset")
    test = pd.read_csv(filename)
    test.fillna(0, inplace = True)
    test = test.values
    data = test
    extract = Model(cnn_model.inputs, cnn_model.layers[-2].output)
    test = extract.predict(test)
    predict = classifier.predict(test)
    for i in range(len(predict)):
        if predict[i] == 1:
            text.insert(END,str(data[i])+" ===> record detected as ENERGY THEFT\n\n")
        if predict[i] == 0:
            text.insert(END,str(data[i])+" ===> record NOT detected as ENERGY THEFT\n\n")
```

Step 12:Accuracy comparison graph



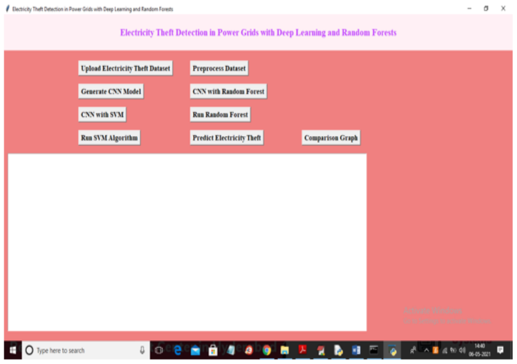
```
def graph():
    df = pd.DataFrame([['CNN','Precision',precision[0]],['CNN','Recall',recall[0]],['CNN','F1 Score',fscore[0]],['CNN','Accuracy',accu
                       ['CNN-RF','Precision',precision[1]],['CNN-RF','Recall',recall[1]],['CNN-RF','F1 Score',fscore[1]],['CNN-RF','Ac
                       ['CNN-SVM','Precision',precision[2]],['CNN-SVM','Recall',recall[2]],['CNN-SVM','F1 Score',fscore[2]],['CNN-SVM'
                       ['RF','Precision',precision[3]],['RF','Recall',recall[3]],['RF','F1 Score',fscore[3]],['RF','Accuracy',accuracy
                       ['SVM','Precision',precision[3]],['SVM','Recall',recall[3]],['SVM','F1 Score',fscore[3]],['SVM','Accuracy',accu
                      ],columns=['Parameters','Algorithms','Value'])
    df.pivot("Parameters", "Algorithms", "Value").plot(kind='bar')
    plt.show()
```

Step 13:Defining the button size and configuration

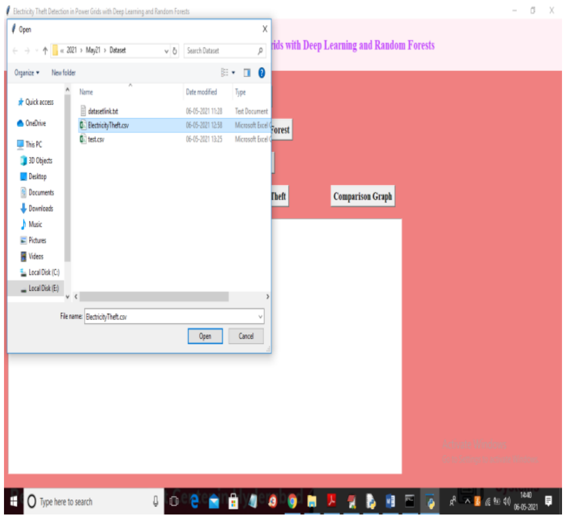


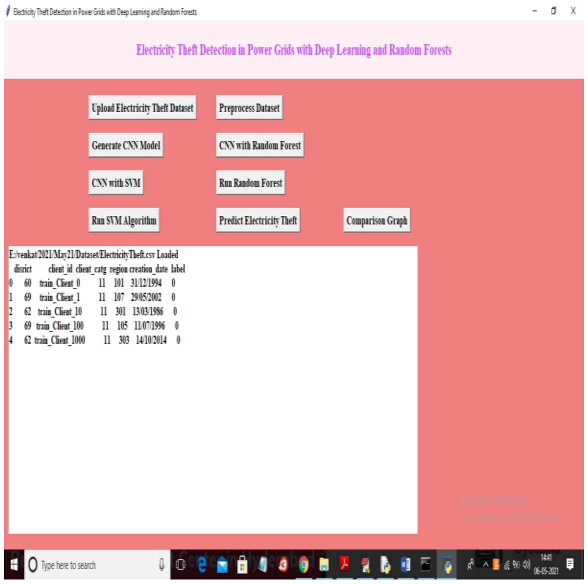
```
font = ('times', 16, 'bold')
title = Label(main, text='Electricity Theft Detection in Power Grids with Deep Learning and Random Forests', justify=LEFT)
title.config(bg='lavender blush', fg='DarkOrchid1')
title.config(font=font)
title.config(height=3, width=120)
title.place(x=100,y=5)
title.pack()

font1 = ('times', 13, 'bold')
uploadButton = Button(main, text="Upload Electricity Theft Dataset", command=uploadDataset)
uploadButton.place(x=200,y=100)
uploadButton.config(font=font1)

preprocessButton = Button(main, text="Preprocess Dataset", command=preprocessDataset)
preprocessButton.place(x=500,y=100)
preprocessButton.config(font=font1)

cnnButton = Button(main, text="Generate CNN Model", command=runCNN)
cnnButton.place(x=200,y=150)
cnnButton.config(font=font1)

cnnrfButton = Button(main, text="CNN with Random Forest", command=runCNNRF)
cnnrfButton.place(x=500,y=150)
cnnrfButton.config(font=font1)

cnnsvmButton = Button(main, text="CNN with SVM", command=runCNNSVM)
cnnsvmButton.place(x=200,y=200)
cnnsvmButton.config(font=font1)

rfButton = Button(main, text="Run Random Forest", command=runRandomForest)
rfButton.place(x=500,y=200)
rfButton.config(font=font1)

svmButton = Button(main, text="Run SVM Algorithm", command=runSVM)
svmButton.place(x=200,y=250)
svmButton.config(font=font1)
```

To run project double click on 'run.bat' file to get below screen

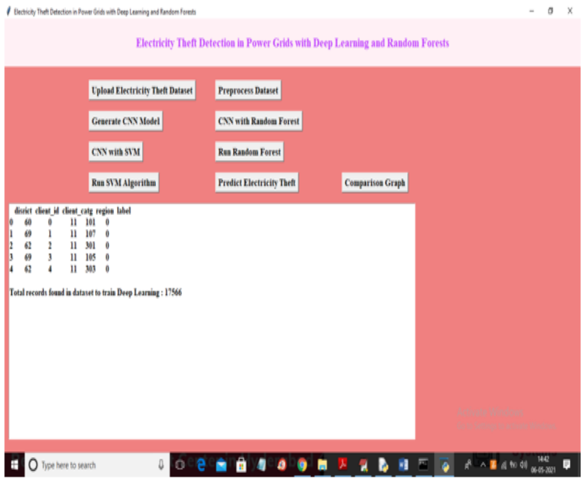

Electricity Theft Detection in Power Grids with Deep Learning and Random Forests
Upload Electricity Theft Dataset
Preprocess Dataset
Generate CNN Model
CNN with Random Forest
CNN with SVM
Run Random Forest
Run SVM Algorithm
Predict Electricity Theft
Comparison Graph

In above screen click on 'Upload Electricity Theft Dataset' button to upload dataset



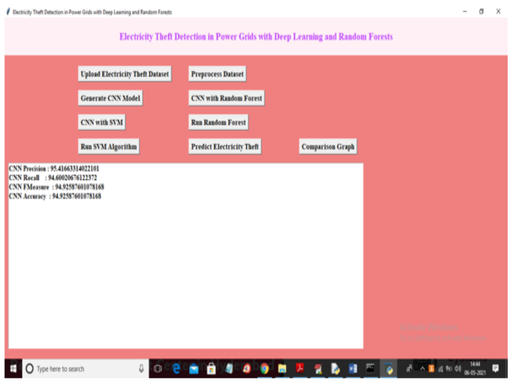

Electricity Theft Detection in Power Grids with Deep Learning and Random Forests
Upload Electricity Theft Dataset
Preprocess Dataset
Generate CNN Model
CNN with Random Forest
CNN with SVM
Run Random Forest
Run SVM Algorithm
Predict Electricity Theft
Comparison Graph

```
E:/venkat/2021/May21/Dataset/ElectricityTheft.csv Loaded
  disrict        client_id client_catg region creation_date label
0     60   train_Client_0       11    101   31/12/1994    0
1     69   train_Client_1       11    107   29/05/2002    0
2     62   train_Client_10      11    301   13/03/1986    0
3     69  train_Client_100      11    105   11/07/1996    0
4     62 train_Client_1000      11    303   14/10/2014    0
```


Electricity Theft Detection in Power Grids with Deep Learning and Random Forests
Upload Electricity Theft Dataset
Preprocess Dataset
Generate CNN Model
CNN with Random Forest
CNN with SVM
Run Random Forest
Run SVM Algorithm
Predict Electricity Theft
Comparison Graph

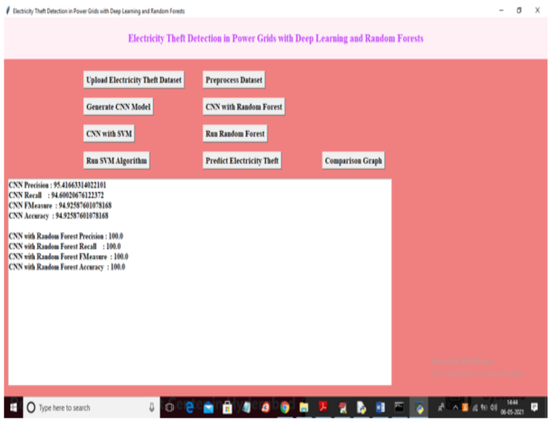
```
  disrict  client_id  client_catg  region  label
0     60        0          11    101     0
1     69        1          11    107     0
2     62        2          11    301     0
3     69        3          11    105     0
4     62        4          11    303     0

Total records found in dataset to train Deep Learning : 17566
```

In above screen dataset converted to numeric format and now click on 'Generate CNN Model' button to train CNN with above dataset


Electricity Theft Detection in Power Grids with Deep Learning and Random Forests
Upload Electricity Theft Dataset
Preprocess Dataset
Generate CNN Model
CNN with Random Forest
CNN with SVM
Run Random Forest
Run SVM Algorithm
Predict Electricity Theft
Comparison Graph

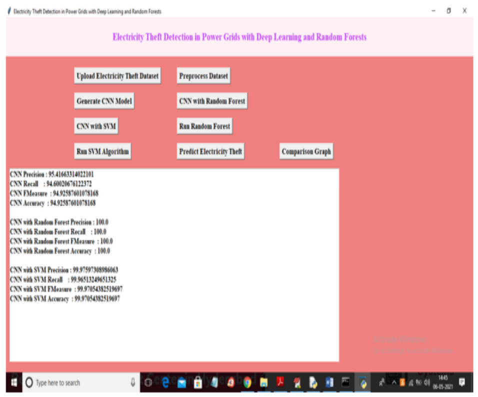
```
CNN Precision : 95.41663314022101
CNN Recall    : 94.60020676122372
CNN FMeasure  : 94.92587601078168
CNN Accuracy  : 94.92587601078168
```

In above screen with normal CNN we got 94% accuracy and now click on 'CNN with Random Forest' button to train CNN with RF


Electricity Theft Detection in Power Grids with Deep Learning and Random Forests
Upload Electricity Theft Dataset
Preprocess Dataset
Generate CNN Model
CNN with Random Forest
CNN with SVM
Run Random Forest
Run SVM Algorithm
Predict Electricity Theft
Comparison Graph

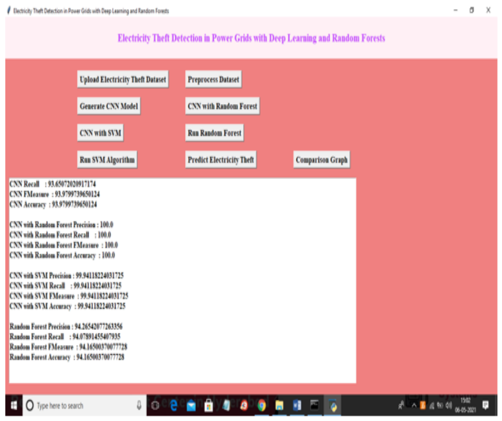
```
CNN Precision : 95.41663314022101
CNN Recall    : 94.60020676122372
CNN FMeasure  : 94.92587601078168
CNN Accuracy  : 94.92587601078168

CNN with Random Forest Precision : 100.0
CNN with Random Forest Recall    : 100.0
CNN with Random Forest FMeasure  : 100.0
CNN with Random Forest Accuracy  : 100.0
```

In above screen with CNN-RF we got 100% accuracy and now click on 'CNN with SVM' button to train dataset with CNN and SVM


Electricity Theft Detection in Power Grids with Deep Learning and Random Forests
Upload Electricity Theft Dataset
Preprocess Dataset
Generate CNN Model
CNN with Random Forest
CNN with SVM
Run Random Forest
Run SVM Algorithm
Predict Electricity Theft
Comparison Graph

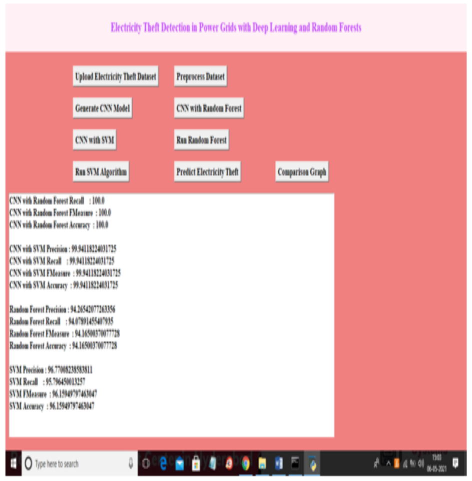
```
CNN Precision : 95.41663314022101
CNN Recall    : 94.60020676122372
CNN FMeasure  : 94.92587601078168
CNN Accuracy  : 94.92587601078168

CNN with Random Forest Precision : 100.0
CNN with Random Forest Recall    : 100.0
CNN with Random Forest FMeasure  : 100.0
CNN with Random Forest Accuracy  : 100.0

CNN with SVM Precision : 99.97597308986063
CNN with SVM Recall    : 99.96513249651325
CNN with SVM FMeasure  : 99.97054382519697
CNN with SVM Accuracy  : 99.97054382519697
```

In above screen with CNN-SVM we got 99% accuracy and now click on 'Run Random Forest' button to train alone RF on dataset


Electricity Theft Detection in Power Grids with Deep Learning and Random Forests
Upload Electricity Theft Dataset
Preprocess Dataset
Generate CNN Model
CNN with Random Forest
CNN with SVM
Run Random Forest
Run SVM Algorithm
Predict Electricity Theft
Comparison Graph

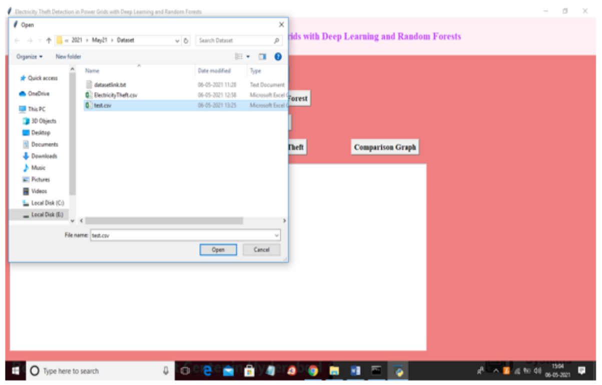
```
CNN Recall    : 93.65072020917174
CNN FMeasure  : 93.9799739650124
CNN Accuracy  : 93.9799739650124

CNN with Random Forest Precision : 100.0
CNN with Random Forest Recall    : 100.0
CNN with Random Forest FMeasure  : 100.0
CNN with Random Forest Accuracy  : 100.0

CNN with SVM Precision : 99.94118224031725
CNN with SVM Recall    : 99.94118224031725
CNN with SVM FMeasure  : 99.94118224031725
CNN with SVM Accuracy  : 99.94118224031725

Random Forest Precision : 94.26542077263356
Random Forest Recall    : 94.07891455407935
Random Forest FMeasure  : 94.16500370077728
Random Forest Accuracy  : 94.16500370077728
```

In above screen with alone Random Forest we got 94% accuracy and now click on 'Run SVM Algorithm' button to train alone SVM with above dataset

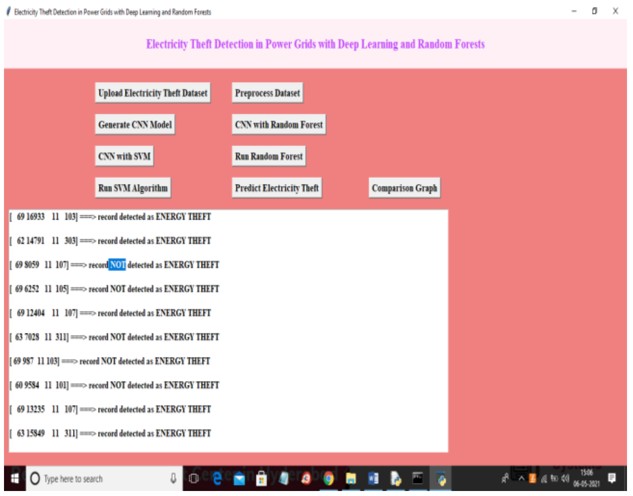

Electricity Theft Detection in Power Grids with Deep Learning and Random Forests
Upload Electricity Theft Dataset
Preprocess Dataset
Generate CNN Model
CNN with Random Forest
CNN with SVM
Run Random Forest
Run SVM Algorithm
Predict Electricity Theft
Comparison Graph

```
CNN with Random Forest Recall    : 100.0
CNN with Random Forest FMeasure  : 100.0
CNN with Random Forest Accuracy  : 100.0

CNN with SVM Precision : 99.94118224031725
CNN with SVM Recall    : 99.94118224031725
CNN with SVM FMeasure  : 99.94118224031725
CNN with SVM Accuracy  : 99.94118224031725

Random Forest Precision : 94.26542077263356
Random Forest Recall    : 94.07891455407935
Random Forest FMeasure  : 94.16500370077728
Random Forest Accuracy  : 94.16500370077728

SVM Precision : 96.77008238583811
SVM Recall    : 95.796450013257
SVM FMeasure  : 96.15949797463047
SVM Accuracy  : 96.15949797463047
```

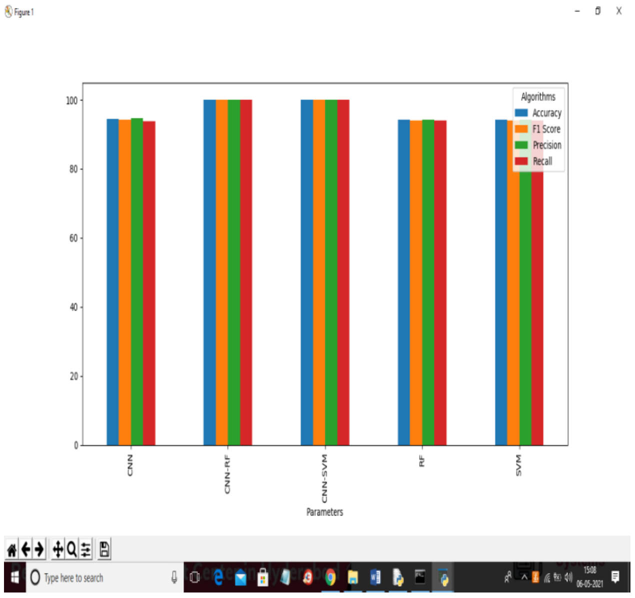
In above screen with alone SVM we got 96% accuracy and now click on 'Predict Electricity Theft' button to upload test data



In above screen selecting and uploading 'test.csv' file and then click on 'Open' button to load test data and to get below prediction result


Electricity Theft Detection in Power Grids with Deep Learning and Random Forests
Upload Electricity Theft Dataset
Preprocess Dataset
Generate CNN Model
CNN with Random Forest
CNN with SVM
Run Random Forest
Run SVM Algorithm
Predict Electricity Theft
Comparison Graph

```
[  69 16933   11   103] ===> record detected as ENERGY THEFT

[  62 14791   11   303] ===> record detected as ENERGY THEFT

[ 69 8059   11  107] ===> record NOT detected as ENERGY THEFT

[ 69 6252   11  105] ===> record NOT detected as ENERGY THEFT

[  69 12404   11   107] ===> record detected as ENERGY THEFT

[ 63 7028   11  311] ===> record NOT detected as ENERGY THEFT

[69 987  11 103] ===> record NOT detected as ENERGY THEFT

[ 60 9584   11  101] ===> record NOT detected as ENERGY THEFT

[  69 13235   11   107] ===> record detected as ENERGY THEFT

[  63 15849   11   311] ===> record detected as ENERGY THEFT
```



CONCLUSION

A new CNN-RF model for detecting electricity theft is provided in this research. In this architecture, the CNN serves as a feature extractor similar to a smart meter data analyzer, while the RF serves as a classifier for the results. In order to avoid overfitting, a fully linked layer with a dropout rate of 0.4 is developed during the training phase of the process. SMOT is also employed to address the issue of data imbalance. SVM, RF, GBDT, and LR are just a few of the deep learning and machine learning approaches that have been tested on the SEAI and LCL datasets. Due to two characteristics, the suggested CNN-RF model appears to be a promising classification method for the detection of electricity theft: Hybrid models can automatically extract features from a dataset, whereas standard classifiers rely heavily on hand-designed features, which is a time and labor-intensive process. Another advantage of the hybrid model is that it incorporates the most widely used and successful classifiers for detecting electricity theft, RF and CNN.

Aiming for the Future:

Detection of electricity theft has an impact on the privacy of consumers. Future research will look into how the granularity and duration of data collected by smart meters affects that confidentiality. Using the hybrid CNN-RF model for additional purposes (such as load forecasting) is something that should be investigated further.

REFERENCES

Journals:

[1] S. S. S. R. Depuru, L. Wang, and V. Devabhaktuni, “Electricity theft: overview, issues, prevention and a smart meter based approach to control theft,” Energy Policy, vol. 39, no. 2, pp. 1007–1015, 2011.

[2] J. P. Navani, N. K. Sharma, and S. Sapra, “Technical and nontechnical losses in power system and its economic consequence in Indian economy,” International Journal of Electronics and Computer Science Engineering, vol. 1, no. 2, pp. 757–761, 2012.

[3] S. McLaughlin, B. Holbert, A. Fawaz, R. Berthier, and S. Zonouz, “A multi-sensor energy theft detection framework for advanced metering infrastructures,” IEEE Journal on Selected Areas in Communications, vol. 31, no. 7, pp. 1319–1330, 2013.

[4] P. McDaniel and S. McLaughlin, “Security and privacy challenges in the smart grid,” IEEE Security & Privacy Magazine, vol. 7, no. 3, pp. 75–77, 2009.

[5] T. B. Smith, “Electricity theft: a comparative analysis,” Energy Policy, vol. 32, no. 1, pp. 2067–2076, 2004.

[6] J. I. Guerrero, C. Leon, I. Monedero, F. Biscarri, and ´ J. Biscarri, “Improving knowledge-based systems with statistical techniques, text mining, and neural networks for nontechnical loss detection,” Knowledge-Based Systems, vol. 71, no. 4, pp. 376–388, 2014.

[7] C. C. O. Ramos, A. N. Souza, G. Chiachia, A. X. Falcão, and J. P. Papa, “A novel algorithm for feature selection using harmony search and its application for non-technical losses detection,” Computers & Electrical Engineering, vol. 37, no. 6, pp. 886–894, 2011.

[8] P. Glauner, J. A. Meira, P. Valtchev, R. State, and F. Bettinger, “-e challenge of non-technical loss detection using artificial intelligence: a surveyficial intelligence: a survey,” International Journal of Computational Intelligence Systems, vol. 10, no. 1, pp. 760–775, 2017.

[9] S.-C. Huang, Y.-L. Lo, and C.-N. Lu, “Non-technical loss detection using state estimation and analysis of variance,” IEEE Transactions on Power Systems, vol. 28, no. 3, pp. 2959–2966, 2013.

[10] O. Rahmati, H. R. Pourghasemi, and A. M. Melesse, “Application of GIS-based data driven random forest and maximum entropy models for groundwater potential mapping: a case study at Mehran region, Iran,” CATENA, vol. 137, pp. 360–372, 2016.

Textbooks:

- Programming Python,Mark Lutz
- Head First Python,Paul Barry
- Core Python Programming,R. Nageswara Rao
- Learning with Python, Allen B. Downey

Websites:

1. https://www.w3schools.com/python/

2. https://www.tutorialspoint.com/python/index.htm
3. https://www.javatpoint.com/python-tutorial
4. https://www.learnpython.org/
5. https://www.pythontutorial.net/
6.