FACULTY OF ENGINEERING AND TECHNOLOGY

SCHOOL OF COMPUTING

DEPARTMENT OF COMPUTING TECHNOLOGIES

AY 2023- 24 EVEN

18CSE352T NEURO-FUZZY AND GENETIC PROGRAMMING

Case Study Implementation Report

Paper Title: Fault detection of wind turbines using SCADA data and genetic algorithm-based ensemble learning

SRM INSTITUTE OF SCIENCE & TECHNOLOGY
Deemed to be University u/s 3 of UGC Act, 1956

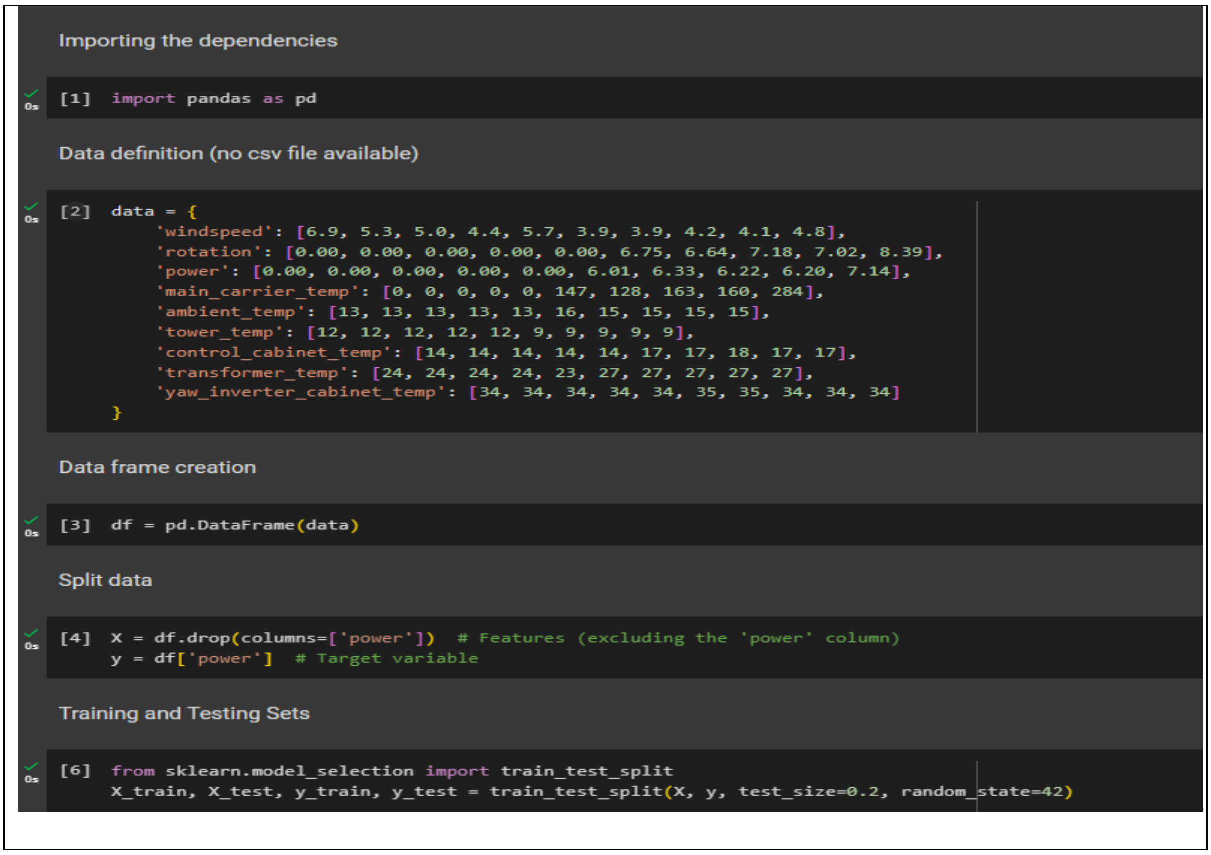
REG NO: RA2111003010653

NAME: Ishit Arhatia

Objective: To display actual vs predicted power for anomaly detection of wind turbines using SCADA data.

Software Used: Google Colaboratory

Screen Shots:



Importing the dependencies

```
[1] import pandas as pd
```

Data definition (no csv file available)

```
[2] data = {
        'windspeed': [6.9, 5.3, 5.0, 4.4, 5.7, 3.9, 3.9, 4.2, 4.1, 4.8],
        'rotation': [0.00, 0.00, 0.00, 0.00, 0.00, 6.75, 6.64, 7.18, 7.02, 8.39],
        'power': [0.00, 0.00, 0.00, 0.00, 0.00, 6.01, 6.33, 6.22, 6.20, 7.14],
        'main_carrier_temp': [0, 0, 0, 0, 0, 147, 128, 163, 160, 284],
        'ambient_temp': [13, 13, 13, 13, 13, 16, 15, 15, 15, 15],
        'tower_temp': [12, 12, 12, 12, 12, 9, 9, 9, 9, 9],
        'control_cabinet_temp': [14, 14, 14, 14, 14, 17, 17, 18, 17, 17],
        'transformer_temp': [24, 24, 24, 24, 23, 27, 27, 27, 27, 27],
        'yaw_inverter_cabinet_temp': [34, 34, 34, 34, 34, 35, 35, 34, 34, 34]
    }
```

Data frame creation

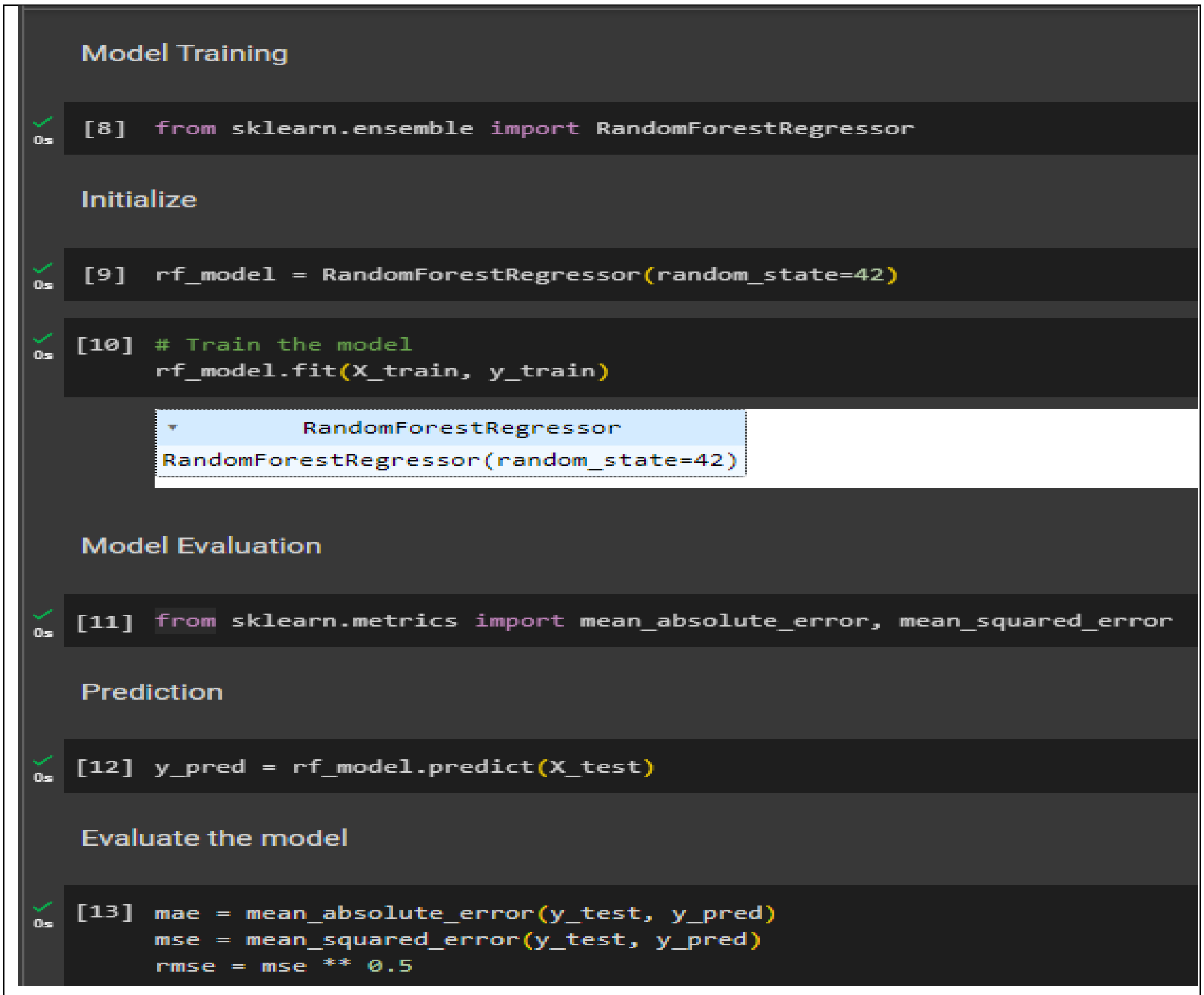
```
[3] df = pd.DataFrame(data)
```

Split data

```
[4] X = df.drop(columns=['power'])  # Features (excluding the 'power' column)
    y = df['power']  # Target variable
```

Training and Testing Sets

```
[6] from sklearn.model_selection import train_test_split
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```



Model Training

```
[8]  from sklearn.ensemble import RandomForestRegressor
```

Initialize

```
[9]  rf_model = RandomForestRegressor(random_state=42)
```

```
[10] # Train the model
     rf_model.fit(X_train, y_train)
```

```
RandomForestRegressor
RandomForestRegressor(random_state=42)
```

Model Evaluation

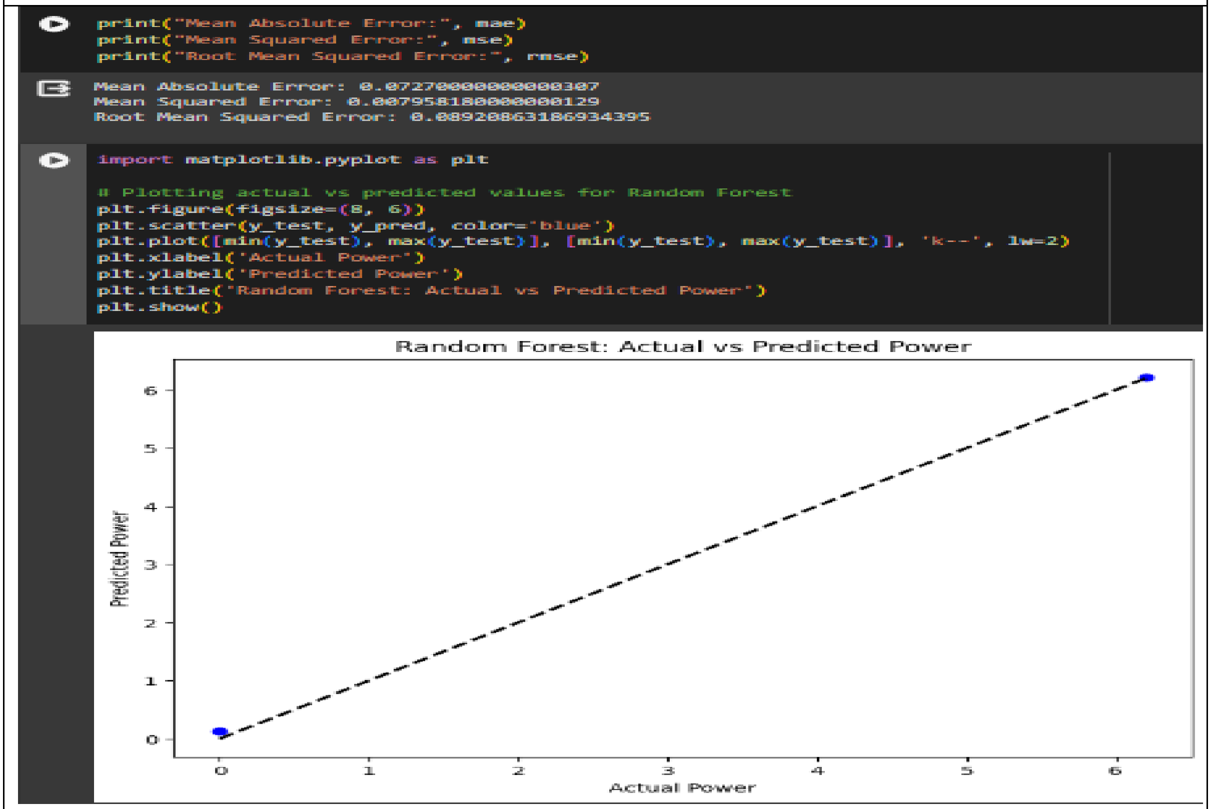
```
[11] from sklearn.metrics import mean_absolute_error, mean_squared_error
```

Prediction

```
[12] y_pred = rf_model.predict(X_test)
```

Evaluate the model

```
[13] mae = mean_absolute_error(y_test, y_pred)
     mse = mean_squared_error(y_test, y_pred)
     rmse = mse ** 0.5
```



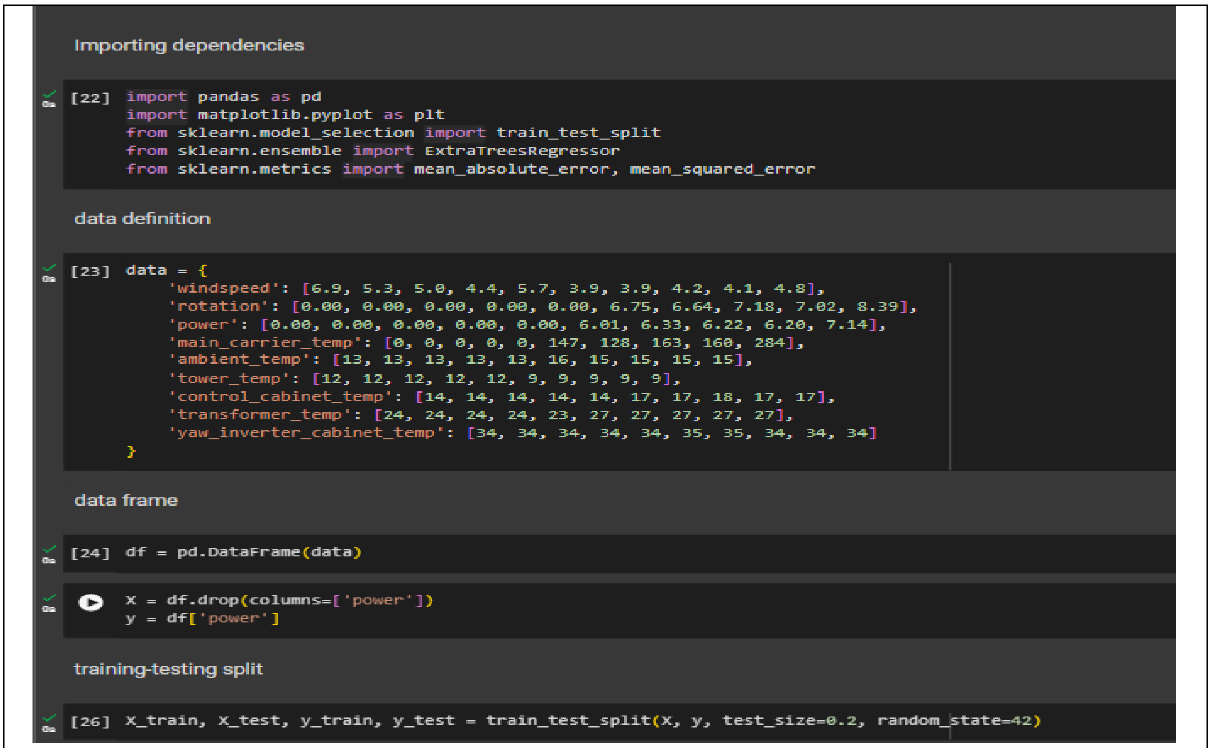
```
print("Mean Absolute Error:", mae)
print("Mean Squared Error:", mse)
print("Root Mean Squared Error:", rmse)
```

```
Mean Absolute Error: 0.07270000000000307
Mean Squared Error: 0.007958180000000129
Root Mean Squared Error: 0.08920863186934395
```

```
import matplotlib.pyplot as plt

# Plotting actual vs predicted values for Random Forest
plt.figure(figsize=(8, 6))
plt.scatter(y_test, y_pred, color='blue')
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], 'k--', lw=2)
plt.xlabel('Actual Power')
plt.ylabel('Predicted Power')
plt.title('Random Forest: Actual vs Predicted Power')
plt.show()
```

Random Forest Screenshots



Importing dependencies

```
[22] import pandas as pd
     import matplotlib.pyplot as plt
     from sklearn.model_selection import train_test_split
     from sklearn.ensemble import ExtraTreesRegressor
     from sklearn.metrics import mean_absolute_error, mean_squared_error
```

data definition

```
[23] data = {
         'windspeed': [6.9, 5.3, 5.0, 4.4, 5.7, 3.9, 3.9, 4.2, 4.1, 4.8],
         'rotation': [0.00, 0.00, 0.00, 0.00, 0.00, 6.75, 6.64, 7.18, 7.02, 8.39],
         'power': [0.00, 0.00, 0.00, 0.00, 0.00, 6.01, 6.33, 6.22, 6.20, 7.14],
         'main_carrier_temp': [0, 0, 0, 0, 0, 147, 128, 163, 160, 284],
         'ambient_temp': [13, 13, 13, 13, 13, 16, 15, 15, 15, 15],
         'tower_temp': [12, 12, 12, 12, 12, 9, 9, 9, 9, 9],
         'control_cabinet_temp': [14, 14, 14, 14, 14, 17, 17, 18, 17, 17],
         'transformer_temp': [24, 24, 24, 24, 23, 27, 27, 27, 27, 27],
         'yaw_inverter_cabinet_temp': [34, 34, 34, 34, 34, 35, 35, 34, 34, 34]
     }
```

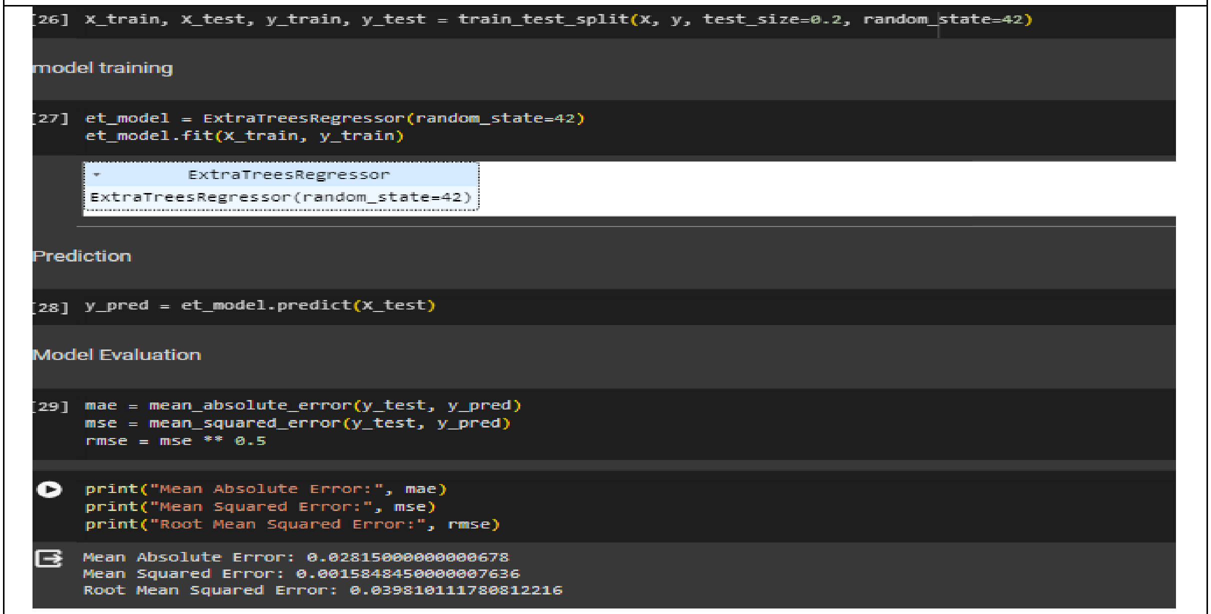
data frame

```
[24] df = pd.DataFrame(data)
```

```
     X = df.drop(columns=['power'])
     y = df['power']
```

training-testing split

```
[26] X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```



```
[26] X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

model training

```
[27] et_model = ExtraTreesRegressor(random_state=42)
     et_model.fit(X_train, y_train)
```

```
        ExtraTreesRegressor
ExtraTreesRegressor(random_state=42)
```

Prediction

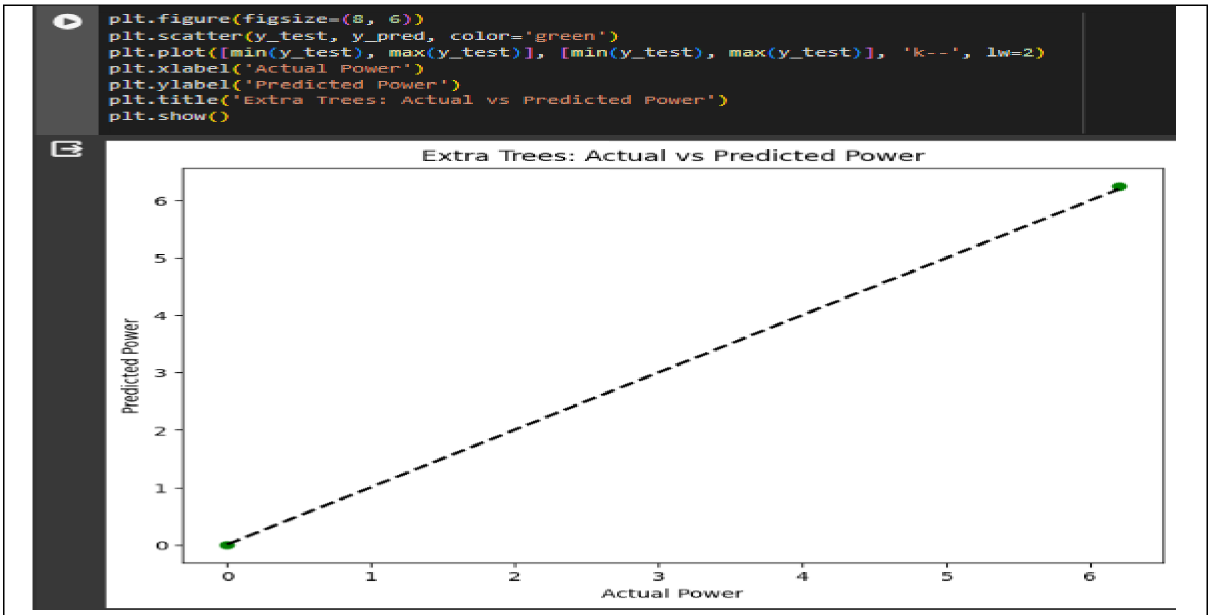
```
[28] y_pred = et_model.predict(X_test)
```

Model Evaluation

```
[29] mae = mean_absolute_error(y_test, y_pred)
     mse = mean_squared_error(y_test, y_pred)
     rmse = mse ** 0.5
```

```
     print("Mean Absolute Error:", mae)
     print("Mean Squared Error:", mse)
     print("Root Mean Squared Error:", rmse)
```

```
Mean Absolute Error: 0.02815000000000678
Mean Squared Error: 0.0015848450000007636
Root Mean Squared Error: 0.039810111780812216
```



```
plt.figure(figsize=(8, 6))
plt.scatter(y_test, y_pred, color='green')
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], 'k--', lw=2)
plt.xlabel('Actual Power')
plt.ylabel('Predicted Power')
plt.title('Extra Trees: Actual vs Predicted Power')
plt.show()
```

Extra Tree Model screenshots

Video Link (Make it Public):

https://drive.google.com/file/d/1RB2r00miFeP_Qld7ewCAxj5vEImXcWGv/view?usp=drive_link

GitHub Link (Make it Public): https://github.com/ia4226/Neuro-Fuzzy-casestudy

Difficulty Faced:

- No proper algorithm was provided.
- No .csv file was available.
- Had to go through just dummy framework codes for Random Forest Algorithm and Extra Tree Algorithm.