Leveraging Machine Learning for Credit Card Fraud Defense

PROJECT GUIDE

Mr.N.Kannan B.E.,M.E.,(PhD)

Assistant Professor, AI

PROJECT MEMBERS

- ➔ Saravana P (23ALR092)
- ➔ Sanjay N (23ALR087)
- ➔ Saruha S M (23ALR093)

Transform Yourself

KONGU ENGINEERING COLLEGE PERUNDURAI

LOVE * KNOWLEDGE * SERVICE

Estd : 1984

KONGU ENGINEERING COLLEGE

PROBLEM STATEMENT

- Create a model to classify credit card transactions as fraudulent or legitimate using a dataset. The model will analyze features like transaction amount, type, time, and other relevant details to predict whether a transaction is fraud or not.

KONGU ENGINEERING COLLEGE

LITERATURE REVIEW

Authors	Title of the Paper	Journal Name	Yearr of Publication
▪ Abdul RehmanKhalid ▪ Nsikak Owoh ▪ OmairUthmani ▪ Moses Ashawa ▪ John Adejoh	Enhancing Credit Card Fraud Detection: An Ensemble Machine Learning Approach	MDPI	2024
▪ K. A. Bakar ▪ N. F. M. Noor ▪ N. M. M. Yusof	Credit Card Fraud Detector Based on Machine Learning Techniques	JCSTS	2023
▪ Seyedeh Khadijeh ▪ Hashemi Seyedeh Leili Mirtaheri ▪ Sergio Greco	Fraud Detection in Banking Data by ML Techniques	IEEE Access	2022
▪ Haritha Nair ▪ V. S. R. Anjaneyulu ▪ K. R. Venugopal	An Intelligent Approach to CCFD Using an Optimized Light Gradient Boosting Machine	IEEE Access	2022
▪ Zhi-Hua Zhou ▪ Zheng Zhang ▪ Jian-Kang Liu ▪ Xiao-Hua Wu	Performance Evaluation of ML Methods for CCFD Using SMOTE and AdaBoost	IEEE Access	2021

OBJECTIVE

- The objective of our paper titled "Leveraging Machine Learning for Credit Card Fraud Defense" is to develop and evaluate a framework for detecting credit card fraud using machine learning techniques.
- Our study specifically aims to address the challenge of class imbalance in fraud detection datasets by employing the Synthetic Minority Over-sampling Technique (SMOTE).
- Additionally, We explores the enhancement of various machine learning models, including Support Vector Machine (SVM), Random Forest (RF), and Extreme Gradient Boosting (XGBoost), among others, through the use of the Adaptive Boosting (AdaBoost) method to improve classification performance.

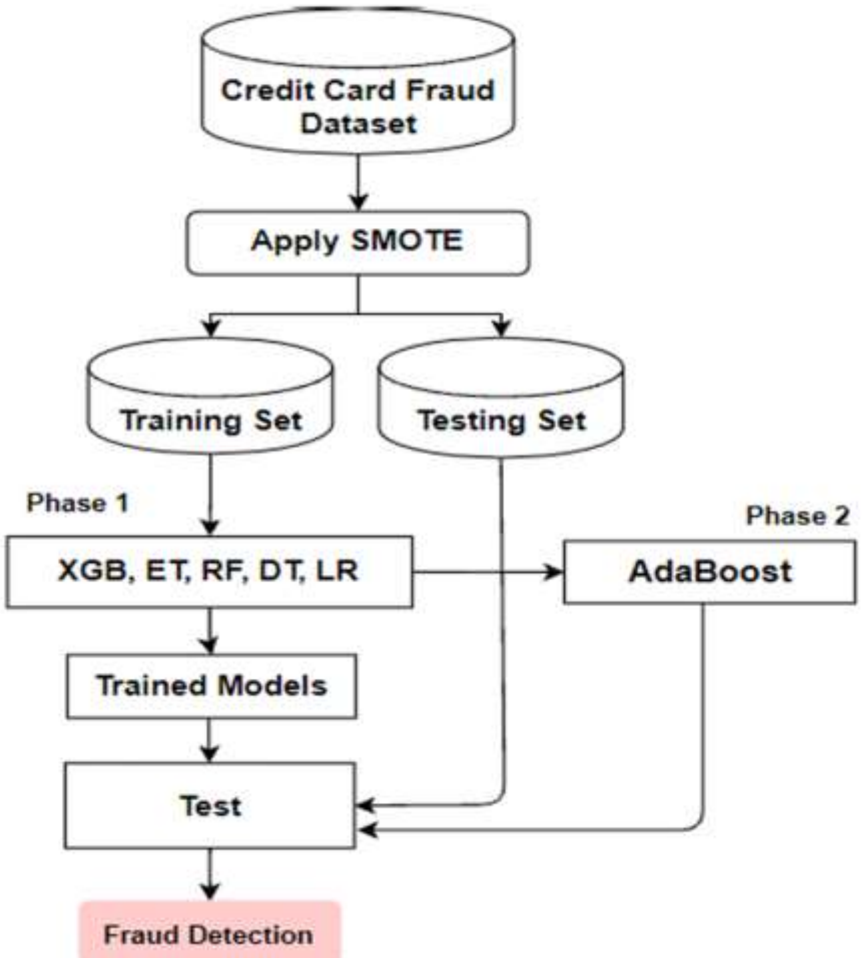
DATASET USED

Credit Card Fraud Detection- The Dataset made by the credit card holders in September 2013.

	Unnamed: 0	Time	V1	V2	V3	V4	V5	V6	V7	V8	...	V21	V22	V23	V24	V25	V26	V27	V28	Amount	Class
0	NaN	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	...	-0.018307	0.277838	-0.110474	0.066928	0.128539	-0.189115	0.133558	-0.021053	149.62	0.0
1	2.0	0.0	1.191857	0.266151	0.166480	0.448154	NaN	-0.082361	-0.078803	0.085102	...	-0.225775	-0.638672	0.101288	-0.339846	0.167170	0.125895	-0.008983	0.014724	2.69	0.0
2	3.0	NaN	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	...	0.247998	0.771679	0.909412	-0.689281	-0.327642	-0.139097	-0.055353	-0.059752	378.66	0.0
3	4.0	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	...	-0.108300	0.005274	-0.190321	-1.175575	0.647376	-0.221929	0.062723	0.061458	123.50	NaN
4	5.0	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	NaN	...	-0.009431	0.798278	-0.137458	0.141267	NaN	0.502292	0.219422	0.215153	69.99	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
284802	284803.0	172786.0	NaN	10.071785	-9.834783	-2.066656	-5.364473	-2.606837	-4.918215	7.305334	...	0.213454	0.111864	NaN	-0.509348	1.436807	0.250034	0.943651	0.823731	0.77	0.0
284803	284804.0	172787.0	-0.732789	-0.055080	2.035030	-0.738589	0.868229	1.058415	0.024330	0.294869	...	0.214205	0.924384	0.012463	-1.016226	-0.606624	-0.395255	0.068472	-0.053527	24.79	0.0
284804	284805.0	172788.0	NaN	-0.301254	-3.249640	-0.557828	2.630515	3.031260	-0.296827	0.708417	...	0.232045	0.578229	-0.037501	0.640134	0.265745	-0.087371	0.004455	-0.026561	67.88	0.0
284805	284806.0	172788.0	-0.240440	0.530483	0.702510	0.689799	-0.377961	0.623708	NaN	0.679145	...	0.265245	0.800049	-0.163298	0.123205	-0.569159	0.546668	0.108821	0.104533	10.00	0.0
284806	284807.0	172792.0	-0.533413	-0.189733	0.703337	-0.506271	-0.012546	-0.649617	1.577006	-0.414650	...	0.261057	0.643078	NaN	0.008797	-0.473649	-0.818267	-0.002415	0.013649	217.00	0.0

https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

PROPOSED WORK (WORK FLOW)



KONGU ENGINEERING COLLEGE

Advantages of the Proposed System

Higher Fraud Detection Accuracy - Using multiple models and boosting techniques like AdaBoost helps catch more fraud cases and reduces mistakes

Better Handling of Imbalanced Data - Using SMOTE helps the model deal with data imbalance, so it doesn't just focus on non-fraud cases.

Adaptable Models - Including models like XGBoost, Random Forest, and Logistic Regression helps the system identify various fraud patterns.

Easy to Understand - Decision Trees make it simpler to see why a transaction was marked as fraud or not.

Scalable - The system can handle a lot of transaction data, making it suitable for real-world use with high data volumes.

MODULES OF THE PROJECT
Implementation

LOGISTIC REGRESSION

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, mean_absolute_error, r2_score
model =LogisticRegression()
model.fit(X_train,y_train)
y_pred=model.predict(X_test)
accuracy=accuracy_score(y_test,y_pred)
print(f'Accuracy: {accuracy}')
# Calculate Mean Absolute Error (MAE)
mae = mean_absolute_error(y_test, y_pred)
print(f"Mean Absolute Error (MAE): {mae}")

# Calculate R-squared (R²) score
r2 = r2_score(y_test, y_pred)
print(f"R-squared (R²) score: {r2}")
```

```
Accuracy: 0.9389561975768872
Mean Absolute Error (MAE): 0.06104380242311277
R-squared (R²) score: 0.755822062722677
```

KONGU ENGINEERING COLLEGE

XG BOOST

```
import xgboost as xgb

# Initialize the XGBoost model without the 'use_label_encoder' parameter
model = xgb.XGBClassifier(eval_metric='mlogloss')

# Train the model
model.fit(X_train, y_train)

# Make predictions
y_pred = model.predict(X_test)

# Evaluate accuracy
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy}')
# Calculate Mean Absolute Error (MAE)
mae = mean_absolute_error(y_test, y_pred)
print(f"Mean Absolute Error (MAE): {mae}")

# Calculate R-squared (R²) score
r2 = r2_score(y_test, y_pred)
print(f"R-squared (R²) score: {r2}")
```

```
Accuracy: 0.999621938138529
Mean Absolute Error (MAE): 0.00037806186147110025
R-squared (R²) score: 0.9984877365504403
```

EXTRA TREE MODEL

```
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.metrics import accuracy_score

# Initialize the Extra Trees model
et_model = ExtraTreesClassifier(random_state=42)

# Train the model
et_model.fit(X_train, y_train)

# Make predictions
y_pred_et = et_model.predict(X_test)

# Evaluate accuracy
accuracy_et = accuracy_score(y_test, y_pred_et)
print(f'Extra Trees Accuracy: {accuracy_et}')
# Calculate Mean Absolute Error (MAE)
mae = mean_absolute_error(y_test, y_pred_et)
print(f"Mean Absolute Error (MAE): {mae}")

# Calculate R-squared (R²) score
r2 = r2_score(y_test, y_pred_et)
print(f"R-squared (R²) score: {r2}")
```

```
Extra Trees Accuracy: 0.999841741546361
Mean Absolute Error (MAE): 0.0001582584536390652
R-squared (R²) score: 0.9993669594862309
```

KONGU ENGINEERING COLLEGE

RANDOM FOREST

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score

# Initialize the Random Forest model
rf_model = RandomForestClassifier(random_state=42)

# Train the model
rf_model.fit(X_train, y_train)

# Make predictions
y_pred_rf = rf_model.predict(X_test)

# Evaluate accuracy
accuracy_rf = accuracy_score(y_test, y_pred_rf)
print(f'Random Forest Accuracy: {accuracy_rf}')
# Calculate Mean Absolute Error (MAE)
mae = mean_absolute_error(y_test, y_pred_rf)
print(f"Mean Absolute Error (MAE): {mae}")

# Calculate R-squared (R²) score
r2 = r2_score(y_test, y_pred_rf)
print(f"R-squared (R²) score: {r2}")
```

```
Random Forest Accuracy: 0.9998593258189875
Mean Absolute Error (MAE): 0.0001406741810125024
R-squared (R²) score: 0.9994372973210941
```

DECISION TREE

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score

# Initialize the Decision Tree model
dt_model = DecisionTreeClassifier(random_state=42)

# Train the model
dt_model.fit(X_train, y_train)

# Make predictions
y_pred_dt = dt_model.predict(X_test)

# Evaluate accuracy
accuracy_dt = accuracy_score(y_test, y_pred_dt)
print(f'Decision Tree Accuracy: {accuracy_dt}')
# Calculate Mean Absolute Error (MAE)
mae = mean_absolute_error(y_test, y_pred_dt)
print(f"Mean Absolute Error (MAE): {mae}")

# Calculate R-squared (R²) score
r2 = r2_score(y_test, y_pred_dt)
print(f"R-squared (R²) score: {r2}")
```

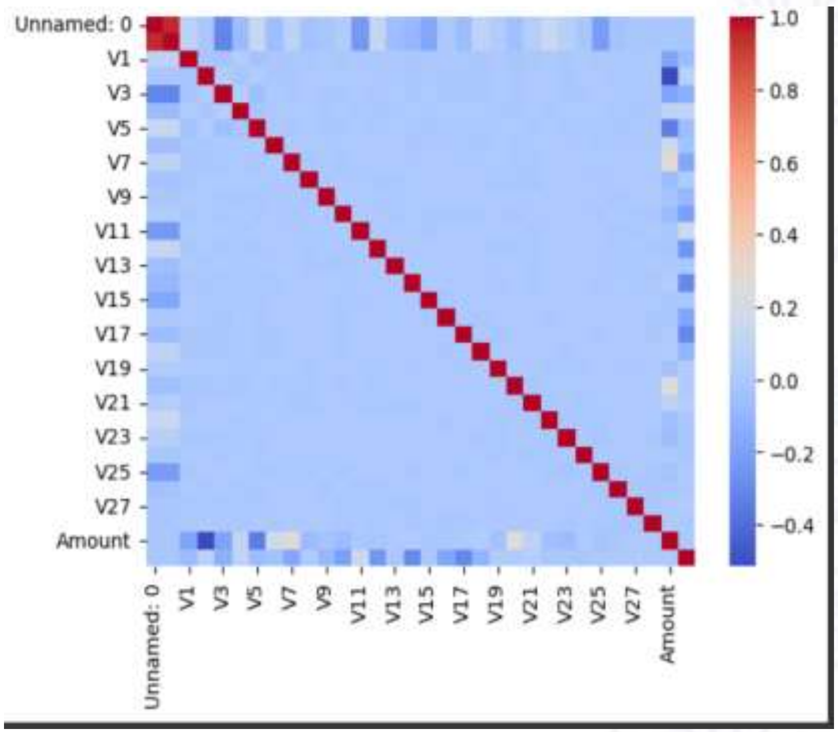
```
Decision Tree Accuracy: 0.9980569378747648
Mean Absolute Error (MAE): 0.0019430621252351896
R-squared (R²) score: 0.9922276692476122
```

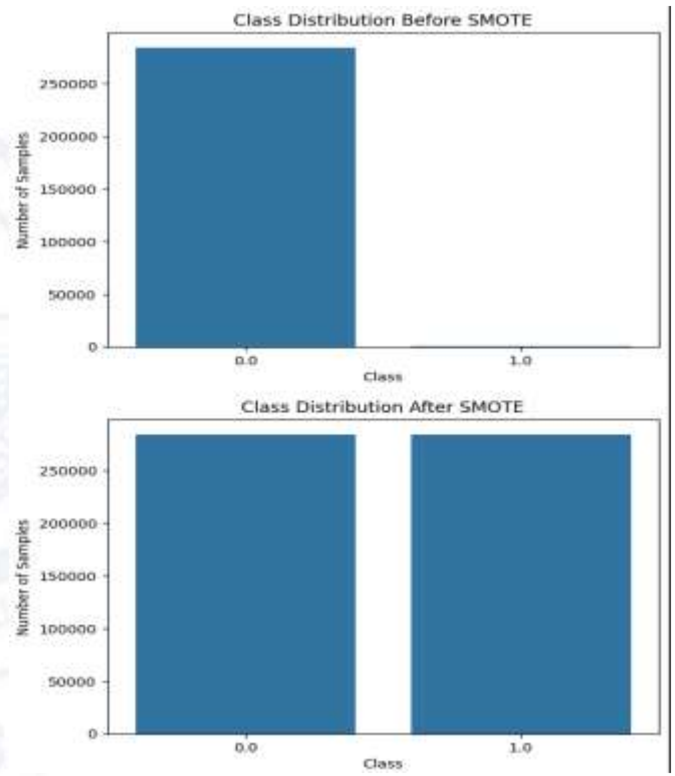
KONGU ENGINEERING COLLEGE

Comparative Analysis of Accuracy: Existing Model vs. Proposed System

Algorithmused	Accuracy of existing models	Accuracy of proposed work
Random forest	94.9999	99.9868
Logistic Regression	92.1645	93.9035
K-Nearest Neighbors (KNN)	93.2223	---
SVM	99.9595	---
XGBoost (XGB)	---	99.9745
Extra Trees Classifier	---	99.9832
Decision Tree Classifier	---	99.8030

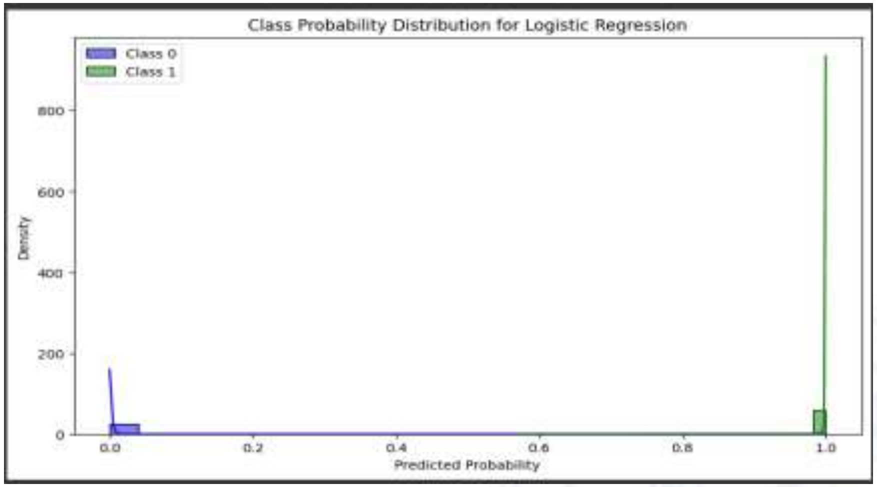
PERFORMANCE METRICS



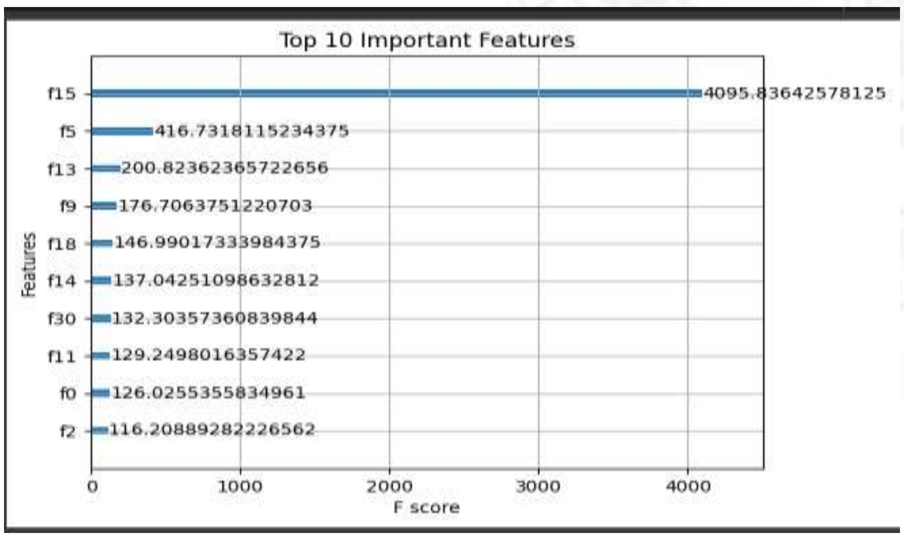


CORRELATION MATRIX

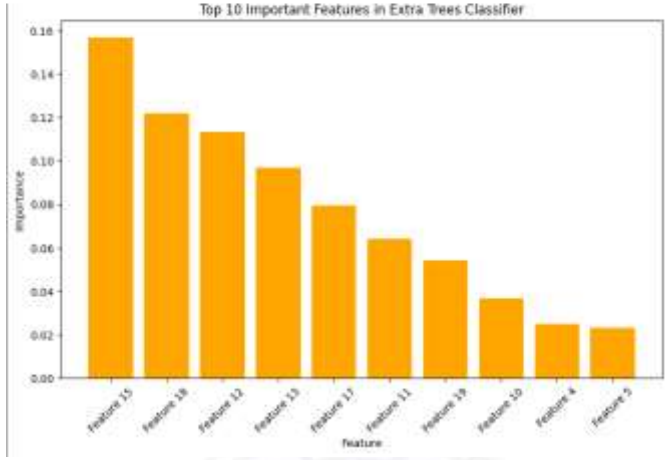
SMOTE



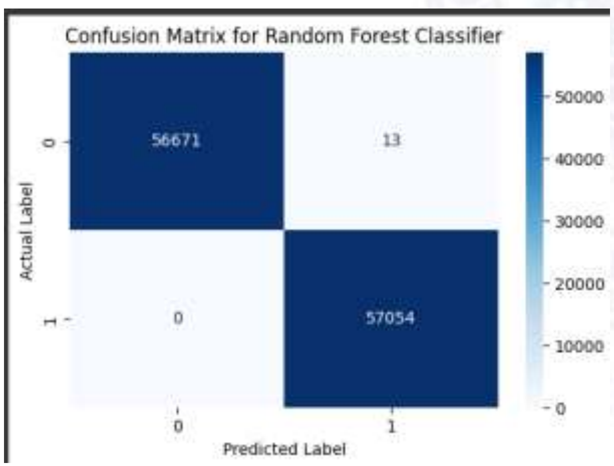
LOGISTIC REGRESSION

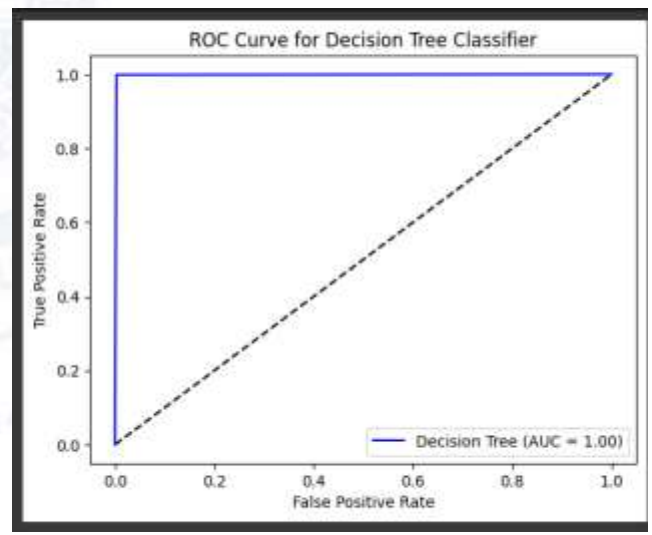


XG BOOST



EXTRA TREES CLASSIFIER





RANDOM FOREST CLASSIFIER

DECISION TREE CLASSIFIER

KONGU ENGINEERING COLLEGE

REFERENES

Zhi-Hua Zhou, Zheng Zhang, Jian-Kang Liu, Xiao-Hua Wu (2021). *Performance Evaluation of ML Methods for CCFD Using SMOTE and AdaBoost*. IEEE Access.

Seyedeh Khadijeh Hashemi, Seyedeh Leili Mirtaheri, Sergio Greco (2022). *Fraud Detection in Banking Data by ML Techniques*. IEEE Access.

Abdul Rehman Khalid, Nsikak Owoh, Omair Uthmani, Moses Ashawa, John Adejoh (2024). *Enhancing Credit Card Fraud Detection: An Ensemble Machine Learning Approach*. MDPI.

Haritha Nair, V. S. R. Anjaneyulu, K. R. Venugopal (2022). *An Intelligent Approach to CCFD Using an Optimized Light Gradient Boosting Machine*. IEEE Access.

K. A. Bakar, N. F. M. Noor, N. M. M. Yusof (2023). *Credit Card Fraud Detector Based on Machine Learning Techniques*. JCSTS.

THANK YOU