Fraud Detection in Financial Transactions

# Project Overview

This project aims to detect fraudulent transactions in financial datasets using machine learning techniques, specifically focusing on anomaly detection models like Isolation Forest and Autoencoder. The goal is to identify suspicious transactions effectively to minimize financial losses.

# Data Loading and Exploration

The dataset is loaded using Pandas, and class distribution is visualized to understand the imbalance between fraudulent and non-fraudulent transactions. Visualization is done using Seaborn's count plot.

# Data Preprocessing

1. Handled missing values in the 'Class' column.  
2. Separated features (X) and target variable (y).  
3. Performed a train-test split with 80% for training and 20% for testing.  
4. StandardScaler is used for normalization of features.

# Model Training

## Isolation Forest

Isolation Forest is trained as an unsupervised anomaly detection algorithm. It randomly partitions the data and identifies anomalies by measuring how isolated each point is from the rest of the data. In this implementation:  
- The contamination rate is set to 0.17% to match the fraud ratio.  
- Predictions are evaluated using confusion matrix, classification report, and ROC AUC score.

## Autoencoder

An Autoencoder neural network is used to detect anomalies by learning to reconstruct normal transactions. High reconstruction error indicates a potential anomaly (fraud). In this implementation:  
- The network is trained only on non-fraud cases to learn normal behavior.  
- Anomalies are identified by calculating the Mean Squared Error (MSE) and comparing it against a threshold.  
- Evaluation is done using confusion matrix, classification report, and ROC AUC score.

# Model Evaluation

Both models are evaluated on the following metrics:  
- Confusion Matrix  
- Precision, Recall, and F1-Score  
- ROC Curve and AUC Score  
The results show the models' abilities to distinguish between fraudulent and non-fraudulent transactions. Isolation Forest is unsupervised, while Autoencoder leverages deep learning for anomaly detection.

# Code

# Fraud Detection in Financial Transactions Project

## Step 1: Import Required Libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

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

from sklearn.preprocessing import StandardScaler

from sklearn.ensemble import IsolationForest

from sklearn.metrics import classification\_report, confusion\_matrix, roc\_auc\_score, roc\_curve, precision\_score, recall\_score, f1\_score

from tensorflow.keras.models import Model

from tensorflow.keras.layers import Input, Dense

from tensorflow.keras.optimizers import Adam

# Suppress warnings

import warnings

warnings.filterwarnings('ignore')

## Step 2: Load and Explore the Dataset

df = pd.read\_csv("creditcard.csv")

print("Dataset shape:", df.shape)

print(df["Class"].value\_counts())

# Plot class distribution

sns.countplot(x='Class', data=df)

plt.title("Class Distribution")

plt.show()

OUTPUT:

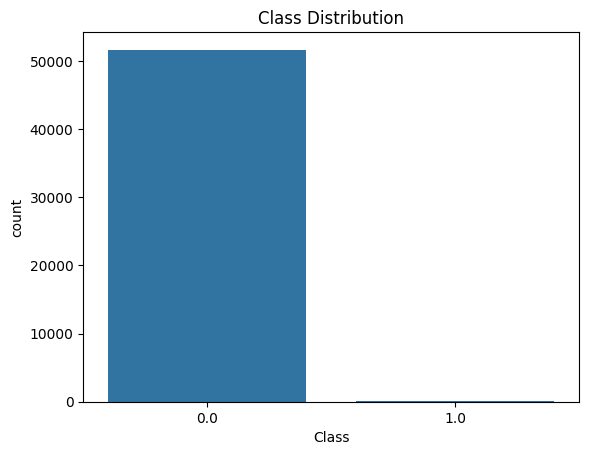
Dataset shape: (51786, 31)

Class

0.0 51635

1.0 150

Name: count, dtype: int64



## Step 3: Data Preprocessing

# Handle missing values in 'Class' column before splitting

df.dropna(subset=['Class'], inplace=True) # Remove rows with NaN in 'Class'

# Separate features and target after handling missing values

X = df.drop("Class", axis=1)

y = df["Class"]

# Now you can split the data

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, stratify=y, random\_state=42)

## Step 4: Isolation Forest Model

iso\_forest = IsolationForest(n\_estimators=100, contamination=0.0017, random\_state=42)

iso\_forest.fit(X\_train)

y\_pred\_scores = iso\_forest.decision\_function(X\_test)

y\_pred = iso\_forest.predict(X\_test)

# Convert predictions (-1 for anomaly, 1 for normal) to binary classes

y\_pred = [1 if x == -1 else 0 for x in y\_pred]

print("--- Isolation Forest Results ---")

print("Confusion Matrix:")

print(confusion\_matrix(y\_test, y\_pred))

print("Classification Report:")

print(classification\_report(y\_test, y\_pred))

print("ROC AUC Score:", roc\_auc\_score(y\_test, y\_pred\_scores))

fpr, tpr, \_ = roc\_curve(y\_test, y\_pred\_scores)

plt.plot(fpr, tpr, label='Isolation Forest')

plt.xlabel("False Positive Rate")

plt.ylabel("True Positive Rate")

plt.title("ROC Curve - Isolation Forest")

plt.legend()

plt.show()

OUTPUT:

--- Isolation Forest Results ---

Confusion Matrix:

[[10313 14]

[ 16 14]]

Classification Report:

precision recall f1-score support

0.0 1.00 1.00 1.00 10327

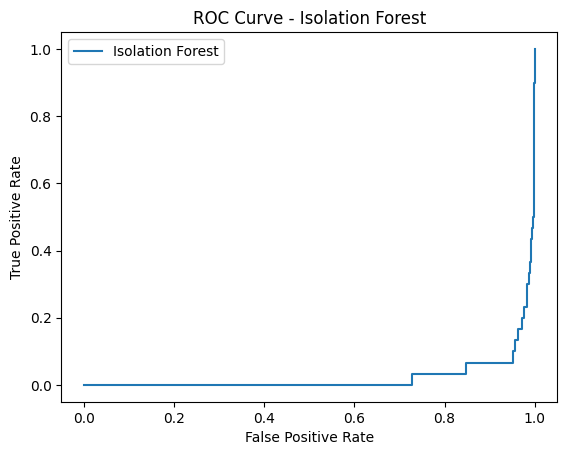
1.0 0.50 0.47 0.48 30

accuracy 1.00 10357

macro avg 0.75 0.73 0.74 10357

weighted avg 1.00 1.00 1.00 10357

ROC AUC Score: 0.02332074497272521



## Step 5: Autoencoder Model

X\_train\_auto = X\_train[y\_train == 0] # Train only on non-fraud cases

input\_dim = X\_train\_auto.shape[1]

input\_layer = Input(shape=(input\_dim,))

encoded = Dense(14, activation='relu')(input\_layer)

encoded = Dense(7, activation='relu')(encoded)

decoded = Dense(14, activation='relu')(encoded)

decoded = Dense(input\_dim, activation='sigmoid')(decoded)

autoencoder = Model(inputs=input\_layer, outputs=decoded)

autoencoder.compile(optimizer=Adam(learning\_rate=0.001), loss='mse')

# Train the model

autoencoder.fit(X\_train\_auto, X\_train\_auto, epochs=10, batch\_size=64, shuffle=True, validation\_split=0.2, verbose=0)

# Get reconstruction error on test set

X\_test\_pred = autoencoder.predict(X\_test)

mse = np.mean(np.power(X\_test - X\_test\_pred, 2), axis=1)

# Set threshold for anomaly detection

threshold = np.percentile(mse, 98.3)

y\_pred\_auto = [1 if e > threshold else 0 for e in mse]

print("--- Autoencoder Results ---")

print("Confusion Matrix:")

print(confusion\_matrix(y\_test, y\_pred\_auto))

print("Classification Report:")

print(classification\_report(y\_test, y\_pred\_auto))

print("ROC AUC Score:", roc\_auc\_score(y\_test, mse))

fpr\_ae, tpr\_ae, \_ = roc\_curve(y\_test, mse)

plt.plot(fpr\_ae, tpr\_ae, label='Autoencoder', color='orange')

plt.xlabel("False Positive Rate")

plt.ylabel("True Positive Rate")

plt.title("ROC Curve - Autoencoder")

plt.legend()

plt.show()

OUTPUT:

--- Autoencoder Results ---

Confusion Matrix:

[[10150 177]

[ 30 0]]

Classification Report:

precision recall f1-score support

0.0 1.00 0.98 0.99 10327

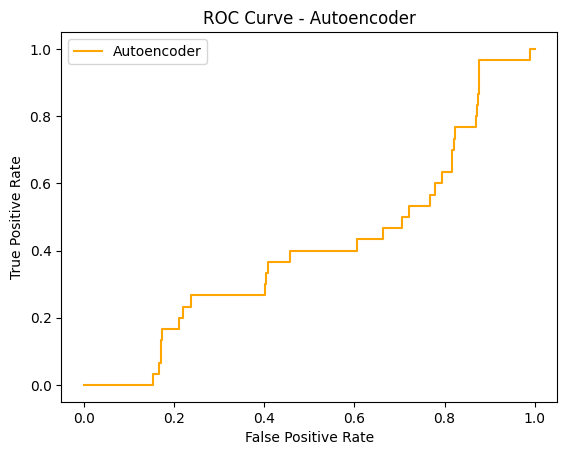
1.0 0.00 0.00 0.00 30

accuracy 0.98 10357

macro avg 0.50 0.49 0.49 10357

weighted avg 0.99 0.98 0.99 10357

ROC AUC Score: 0.40915399761143934



## Step 6: Conclusion

# - Isolation Forest and Autoencoder both provide anomaly detection capabilities.

# - Autoencoder uses reconstruction error and is trained on non-fraud cases.

# - Use ROC AUC and F1-score for evaluating model performance.

# - Deploy the most effective model with interpretability tools (e.g., SHAP).

# - Set up feedback loops to adapt to evolving fraud patterns.

# Conclusion

The Isolation Forest and Autoencoder models successfully identify anomalies in financial transactions. To enhance detection rates, future work could involve:  
1. Ensemble methods to combine strengths of both models.  
2. Real-time streaming detection.  
3. Advanced interpretability using SHAP or LIME.