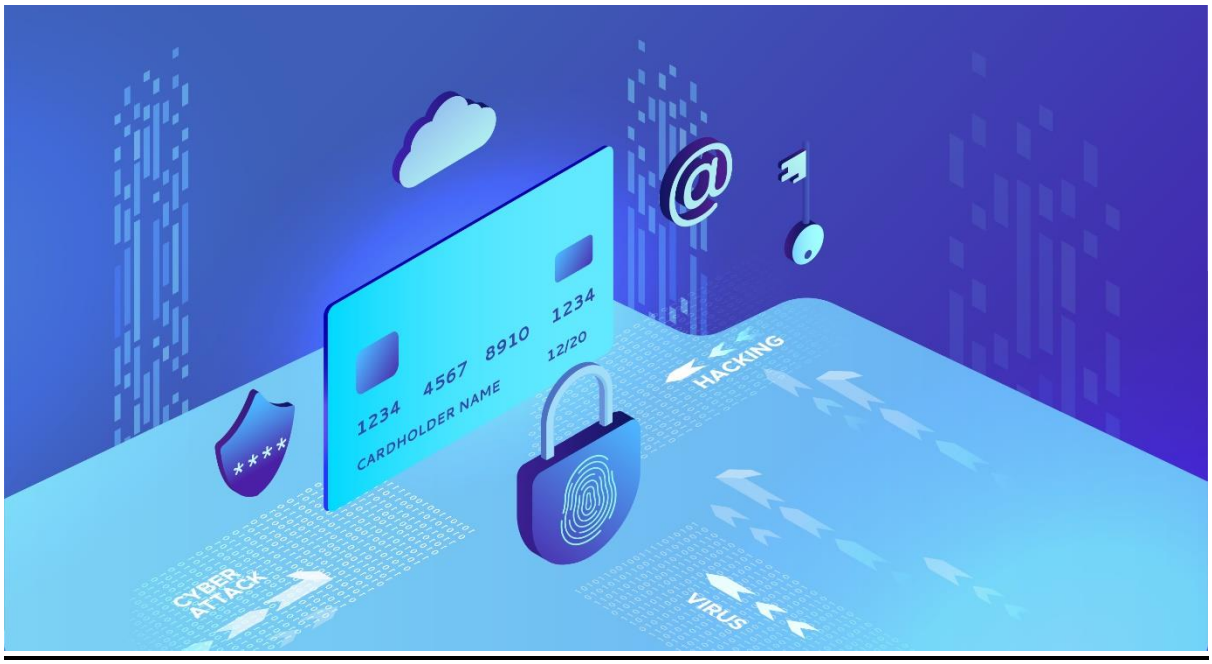
Business Idea: Real-time Financial Fraud Detection System

Credit Card Fraud Detection



Rajeev Pandey

Date: 23/11/2023

Abstract:

Financial fraud is a major problem costing businesses and consumers billions of dollars yearly. Real-time financial fraud detection systems are designed to identify and halt fraudulent transactions, helping businesses protect their customers and assets.

Real-time financial fraud detection systems use machine learning algorithms, statistical analysis, and anomaly detection techniques to identify and stop fraudulent transactions. They ingest real-time transaction data from sources like payment processors, e-commerce platforms, and ATM networks. Identified suspicious transactions can be blocked, alerted, or notified. These systems help businesses protect themselves from fraud and maintain

customer trust, making them increasingly important in the growing volume and complexity of financial transactions.

1. Problem Statement:

Financial institutions face a constant threat of fraudulent activities, including credit card fraud, identity theft, and insider trading. Developing a robust system to detect and prevent these frauds is essential.

Financial fraud costs businesses billions annually and is becoming increasingly difficult to detect and prevent. Real-time fraud detection systems can help by analyzing transaction data in real-time. However, designing and implementing these systems is challenging due to the complexity of fraud patterns and the need to balance preventing fraud with minimizing false positives.

2. Market/Customer/Business Need Assessment:

Market Assessment:

The global fraud detection and prevention market size was valued at **USD 36.89 billion in 2022** and is projected to grow from **USD 43.97 billion in 2023 to USD 182.66 billion by 2030**, exhibiting a CAGR of **22.6%** during the forecast period. The increasing spending by key companies on introducing fraud solutions across different industries, such as BFSI, healthcare, manufacturing, and others, is projected to fuel market growth.

The global market for financial fraud detection and prevention is projected to reach $66.6 billion by 2028, growing at a CAGR of 19.1% from 2023 to 2028. This growth is being driven by several factors, including:

- The increasing volume and complexity of financial transactions.
- The growing sophistication of fraudsters.
- Regulatory requirements for businesses to have fraud prevention measures in place.

Customer Needs Assessment:

Businesses of all sizes need to protect themselves from financial fraud. Fraud can cost businesses billions of dollars each year, and it can also damage their reputation and customer trust.

Real-time financial fraud detection systems can help businesses to identify and stop fraudulent transactions before they are completed. This can help businesses to reduce fraud losses, improve their customer experience, and comply with regulations.

- **Real-time Protection:** Customers and financial institutions require real-time fraud detection to identify and prevent fraudulent transactions and activities as they occur.
- **Accuracy:** Customers demand high accuracy in fraud detection to minimize false positives, which can be disruptive, and false negatives, which can result in financial losses.
- **Privacy and Security:** Data privacy is a top concern. Customers expect their financial data to be handled with the utmost care and security.
- **Transparency:** Customers and regulators expect transparency in how fraud detection decisions are made, including the ability to explain why a transaction is flagged as potentially fraudulent.
- **Ease of Use:** For business users, the system should be user-friendly and seamlessly integrated with existing processes and systems.

Business Need Assessment:

Businesses need to have effective fraud prevention measures in place to protect their customers, assets, and reputation. Real-time financial fraud detection systems can help businesses to achieve this goal.

Here are some specific customer/business needs that a real-time financial fraud detection system can address:

- **Reduce fraud losses**: Businesses can use a real-time fraud detection system to identify and stop fraudulent transactions before they are completed. This can help businesses to reduce their fraud losses significantly.
- **Improve customer experience**: By blocking fraudulent transactions, a real-time fraud detection system can help businesses protect their customers from financial harm. This can lead to improved customer satisfaction and loyalty.
- **Comply with regulations**: Many industries are subject to regulations that require businesses to have fraud prevention measures in place. A real-time fraud detection system can help businesses comply with these regulations and avoid costly fines.
- **Identify new fraud trends**: A real-time fraud detection system can be used to generate reports on fraud trends and patterns. This information can be used by businesses to improve their fraud prevention strategies.

3. External Search:

Here are some external search sources/references/links for real-time financial fraud detection systems:

Websites

- Real-Time Fraud Detection: A Comprehensive Guide: **https://www.tinybird.co/blog-posts/how-to-build-a-real-time-fraud-detection-system**
- Real-Time Fraud Detection Systems: A Review: **https://www.techopedia.com/definition/31745/real-time-fraud-detection**
- Real-Time Fraud Detection: A Market Overview: **https://www.databridgemarketresearch.com/reports/glob**
- **al-fraud-detection-transaction-monitoring-market**

Reports

- Real-Time Fraud Detection Market Size, Share & Trends Analysis Report by Component (Solutions, Services), by Deployment (On-Premise, Cloud), Organization Size, Industry, and Regional Forecast, 2022-2030: **https://www.reportsanddata.com/report-detail/fraud-detection-prevention-market)**
- Real-Time Fraud Detection Market Report: Global Industry Analysis, Trends, Size, Share, Growth, Opportunity and Forecast 2022-2028: **https://www.globenewswire.com/en/news-release/2022/07/25/2485302/0/en/Fraud-Detection-and-Prevention-Market-Size-is-projected-to-reach-USD-190-billion-by-2030-growing-at-a-CAGR-of-23-2-Straits-Research.html**

❖ **Articles**

- Real-Time Fraud Detection: The Future of Fraud Prevention: https://www.forbes.com/sites/cathyhackl/2021/01/16/the-future-of-fraud/
- How to Build a Real-Time Fraud Detection System: https://www.tinybird.co/blog-posts/how-to-build-a-real-time-fraud-detection-system
- The Benefits of Real-Time Fraud Detection: https://pixelplex.io/blog/machine-learning-for-fraud-detection/

These are just a few examples of the many resources available on real-time financial fraud detection systems. By conducting further research, businesses can learn more about the latest technologies and best practices in this area.

* 💳 🦹 Credit Card Fraud Detection 💳 🦹 *

📚 Importing Libraries 📚

[15]
```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns
import plotly.graph_objects as go
from plotly.subplots import make_subplots
import plotly.express as px
import plotly.io as pio

from sklearn.preprocessing import StandardScaler
from imblearn.under_sampling import RandomUnderSampler
from sklearn.model_selection import train_test_split, RepeatedStratifiedKFold, cross_val_score

from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.metrics import roc_auc_score, classification_report, confusion_matrix, roc_curve, auc, precision_recall_curve


import warnings
warnings.filterwarnings('ignore')
```

⏳ Loading the dataset ⏳

```
[17] credit_card = pd.read_csv('/content/creditcard.csv')
```

```
[18] credit_card.head()
```

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	V22	V23	V24	V25	V26	V27	V28
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	...	-0.018307	0.277838	-0.110474	0.066928	0.128539	-0.189115	0.133558	-0.021053
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	...	-0.225775	-0.638672	0.101288	-0.339846	0.167170	0.125895	-0.008983	0.014724
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	...	0.247998	0.771679	0.909412	-0.689281	-0.327642	-0.139097	-0.055353	-0.059752
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	...	-0.108300	0.005274	-0.190321	-1.175575	0.647376	-0.221929	0.062723	0.061458
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	...	-0.009431	0.798278	-0.137458	0.141267	-0.206010	0.502292	0.219422	0.215153

5 rows × 31 columns

```
[19] credit_card.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 284807 entries, 0 to 284806
Data columns (total 31 columns):
 #   Column  Non-Null Count   Dtype  
---  ------  --------------   -----  
 0   Time    284807 non-null  float64
 1   V1      284807 non-null  float64
 2   V2      284807 non-null  float64
 3   V3      284807 non-null  float64
 4   V4      284807 non-null  float64
 5   V5      284807 non-null  float64
```

```
[21]          # Dropping the duplicated values
credit_card = credit_card.drop_duplicates()
```

🧩 Feature Engineering 🧩

```
[22] credit_card['Hour'] = credit_card['Time'].apply(lambda x : np.ceil(float(x)/3600) % 24)
```

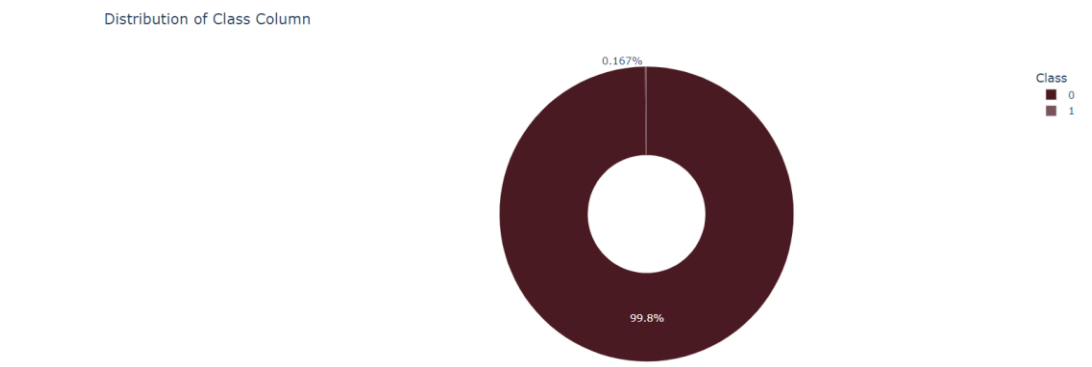
📊 Exploratory Data Analysis 📊

```
[23] credit_card.describe()
```

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V22	V23
count	283726.000000	283726.000000	283726.000000	283726.000000	283726.000000	283726.000000	283726.000000	283726.000000	283726.000000	283726.000000	...	283726.000000	283726.000000
mean	94811.077600	0.005917	-0.004135	0.001613	-0.002966	0.001828	-0.001139	0.001801	-0.000854	-0.001596	...	-0.000015	0.000198
std	47481.047891	1.948026	1.646703	1.508682	1.414184	1.377008	1.331931	1.227664	1.179054	1.095492	...	0.724550	0.623702
min	0.000000	-56.407510	-72.715728	-48.325589	-5.683171	-113.743307	-26.160506	-43.557242	-73.216718	-13.434066	...	-10.933144	-44.807735
25%	54204.750000	-0.915951	-0.600321	-0.889682	-0.850134	-0.689830	-0.769031	-0.552509	-0.208828	-0.644221	...	-0.542700	-0.161703
50%	84692.500000	0.020384	0.063949	0.179963	-0.022248	-0.053468	-0.275168	0.040859	0.021898	-0.052596	...	0.006675	-0.011159
75%	139298.000000	1.316068	0.800283	1.026960	0.739647	0.612218	0.396792	0.570474	0.325704	0.595977	...	0.528245	0.147748
max	172792.000000	2.454930	22.057729	9.382558	16.875344	34.801666	73.301626	120.589494	20.007208	15.594995	...	10.503090	22.528412

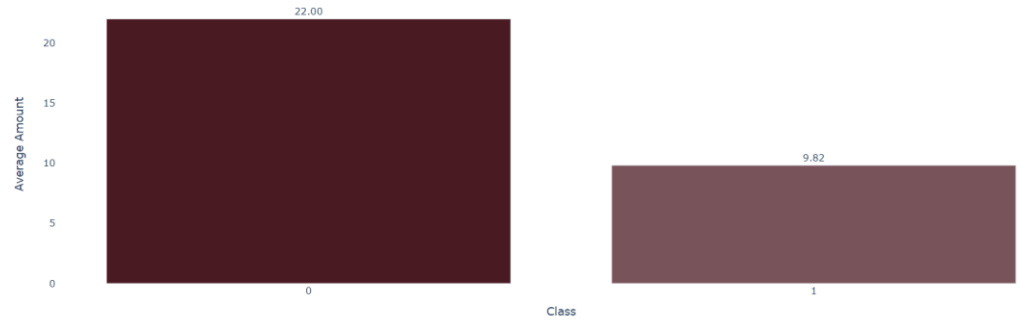
8 rows × 32 columns

```
fig = px.pie(credit_card, names='Class', title='Distribution of Class Column', color_discrete_sequence=['#4A1A22', '#78545A'], hole=0.4)
fig.update_layout(showlegend=True, legend=dict(title='Class', itemsizing='constant'), plot_bgcolor='white' )
fig.show()
```

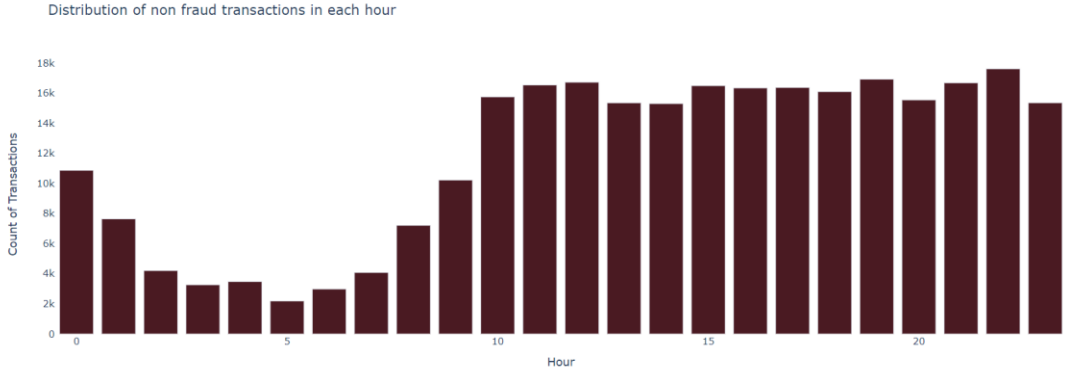


```
average_time_per_class = credit_card.groupby('Class')['Time'].mean().reset_index()
fig = px.bar(average_time_per_class, x='Class', y='Time', text = 'Time', title='Average Time for Each Fraudulent and Non Fraudulent Class')

fig.update_layout(xaxis_title='Class', yaxis_title='Average Time', plot_bgcolor='white' )
fig.update_traces(texttemplate='%{text:.2f}', textposition='outside', marker_color=['#4A1A22', '#78545A'])
fig.update_xaxes(tickvals=[0, 1], ticktext=['0', '1'])
fig.show()
```

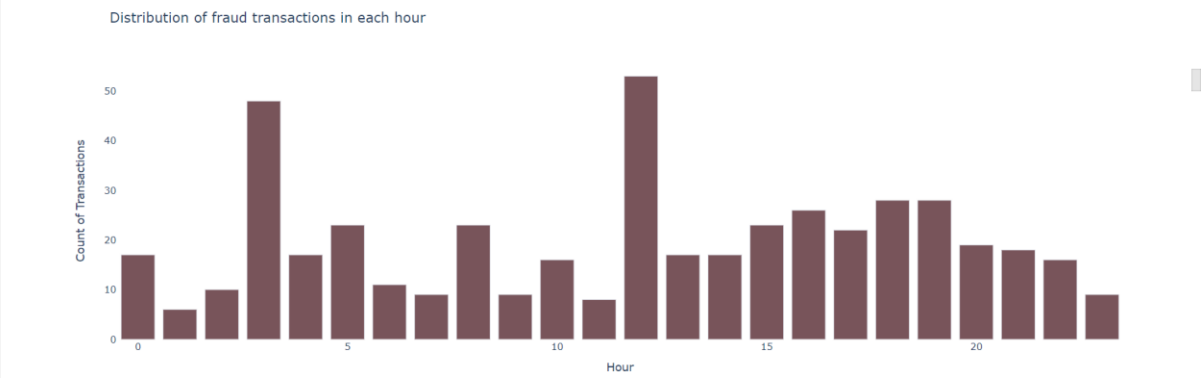

```
average_amount_per_class = credit_card.groupby('Class')['Amount'].median().reset_index()
fig = px.bar(average_amount_per_class, x='Class', y='Amount', text = 'Amount', title='Average Amount for Each Fraudulent and Non Fraudulent Class', color_discrete_sequence=['#4A1A22',
fig.update_layout(xaxis_title='Class', yaxis_title='Average Amount', plot_bgcolor='white')
fig.update_traces(texttemplate='%{text:.2f}', textposition='outside', marker_color=['#4A1A22', '#78545A'])
fig.update_xaxes(tickvals=[0, 1], ticktext=['0', '1'])
fig.show()
```




```
class_0_df = hourly_transaction[0].reset_index()
fig_class_0 = px.bar(class_0_df, x='Hour', y=0, title="Distribution of non fraud transactions in each hour", labels={'0': 'Count of Transactions'}, color_discrete_sequence = ['#4A1A22
fig_class_0.update_layout(plot_bgcolor='white')
fig_class_0.show()
```



```
class_1_df = hourly_transaction[1].reset_index()
fig_class_1 = px.bar(class_1_df, x='Hour', y=1, title="Distribution of fraud transactions in each hour", labels={'1': 'Count of Transactions'}, color_discrete_sequence=['#78545A'])
fig_class_1.update_layout(plot_bgcolor='white')
fig_class_1.show()
```



Insights:

- The dataset is highly imbalanced as the class for non fraud transactions constitutes 99.8% of the data
- It can be clearly seen that the average time for fraudulent transactions is less than fraudulent ones.
- Following this, average transaction amount is also less in fraudulent transactions compared to non fraudulent ones.
- The least number of transactions done are between 1am to 8am
- The highets number of non fraud transactions are done around 10pm
- The highest number of fraud transactions are done at 3am and 12pm
- Irrespective of fraud and non fraud class, the minimum average transaction amount is 11.88K, around 6am to 7am. However, the highest average transaction amount is 33 around 11am
- In case of fraudulent transactions, it can be clearly seen that the highest average amount transaction is done around 1am with a value of 230. Furthermore, the average transaction amount fluctuates a lot in different hours of the day
- In case of non fraud transactions, the average transaction amount doesn't seems to have any noticeable flucations during the whole day.

4. Applicable Patents:

Here are some applicable patents for real-time financial fraud detection systems:

- **US Patent 10,452,908**: "System and method for real-time fraud detection using predictive modelling"
- **US Patent 10,318,710**: "Method and system for detecting fraudulent transactions using machine learning"
- **US Patent 10,246,092**: "System and method for detecting fraudulent transactions using anomaly detection"
- **US Patent 10,197,607**: "Method and system for detecting fraudulent transactions using transaction data analysis"

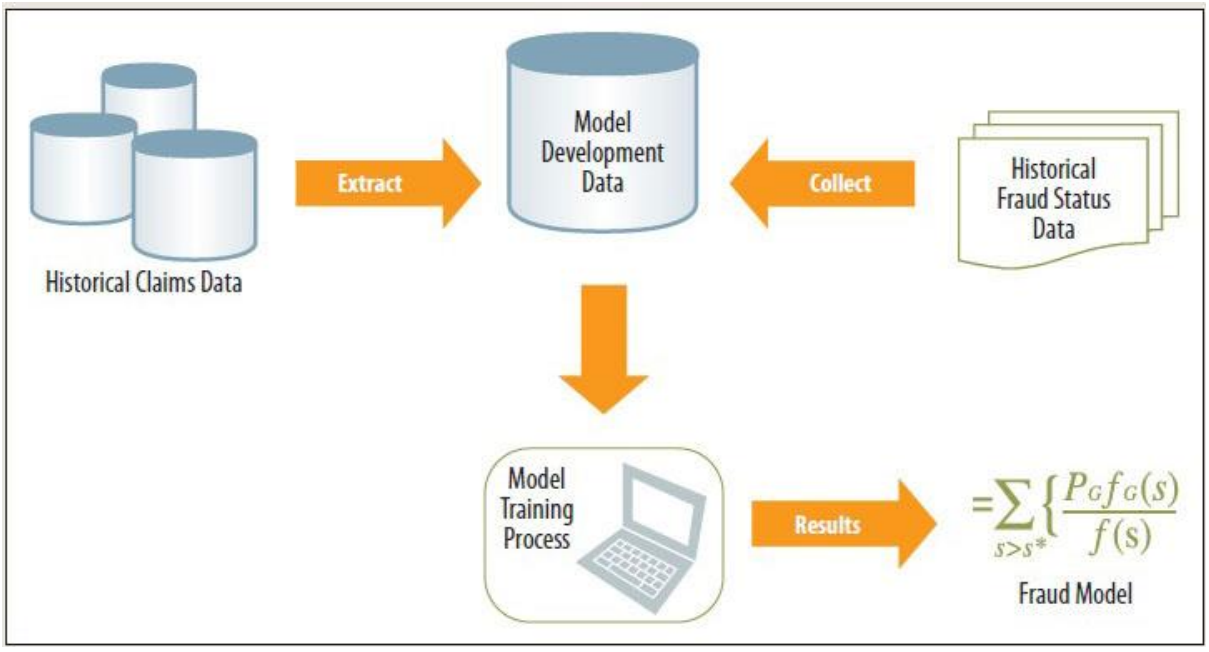
- **US Patent 10,156,237**: "Method and system for detecting fraudulent transactions using device information"

These patents cover a variety of techniques for detecting fraudulent transactions in real-time. Some of the techniques include:

- Predictive modelling
- Machine learning
- Anomaly detection
- Transaction data analysis
- Device information

These patents can be used to develop real-time financial fraud detection systems that are more accurate and efficient than existing systems.

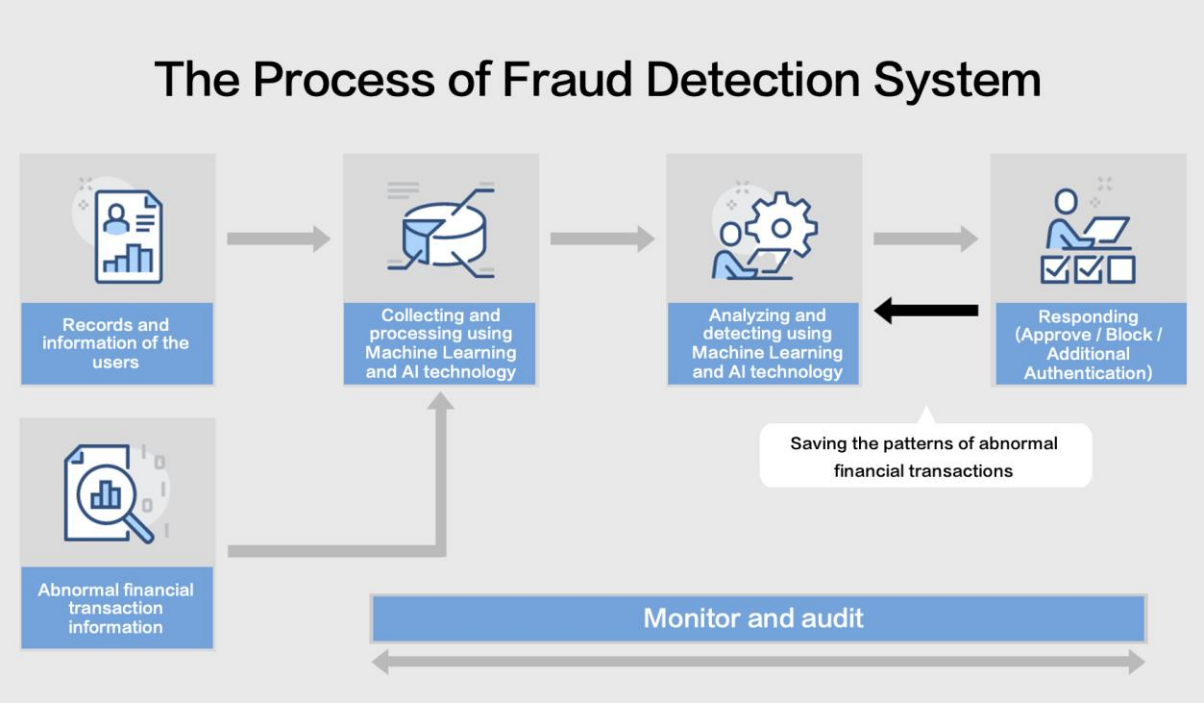
5. Business Models:



Several different business models can be used for real-time financial fraud detection systems. Some of the most common business models include:

- **Subscription Model:** Businesses pay a monthly or annual subscription fee to use the real-time fraud detection system.
- **Pay-per-use Model:** Businesses pay a fee for each transaction that is processed by the real-time fraud detection system.
- **White-label Model:** Businesses can white-label the real-time fraud detection system and offer it to their customers.
- **Partnership Model:** Businesses can partner with a real-time fraud detection company to offer the system to their customers.

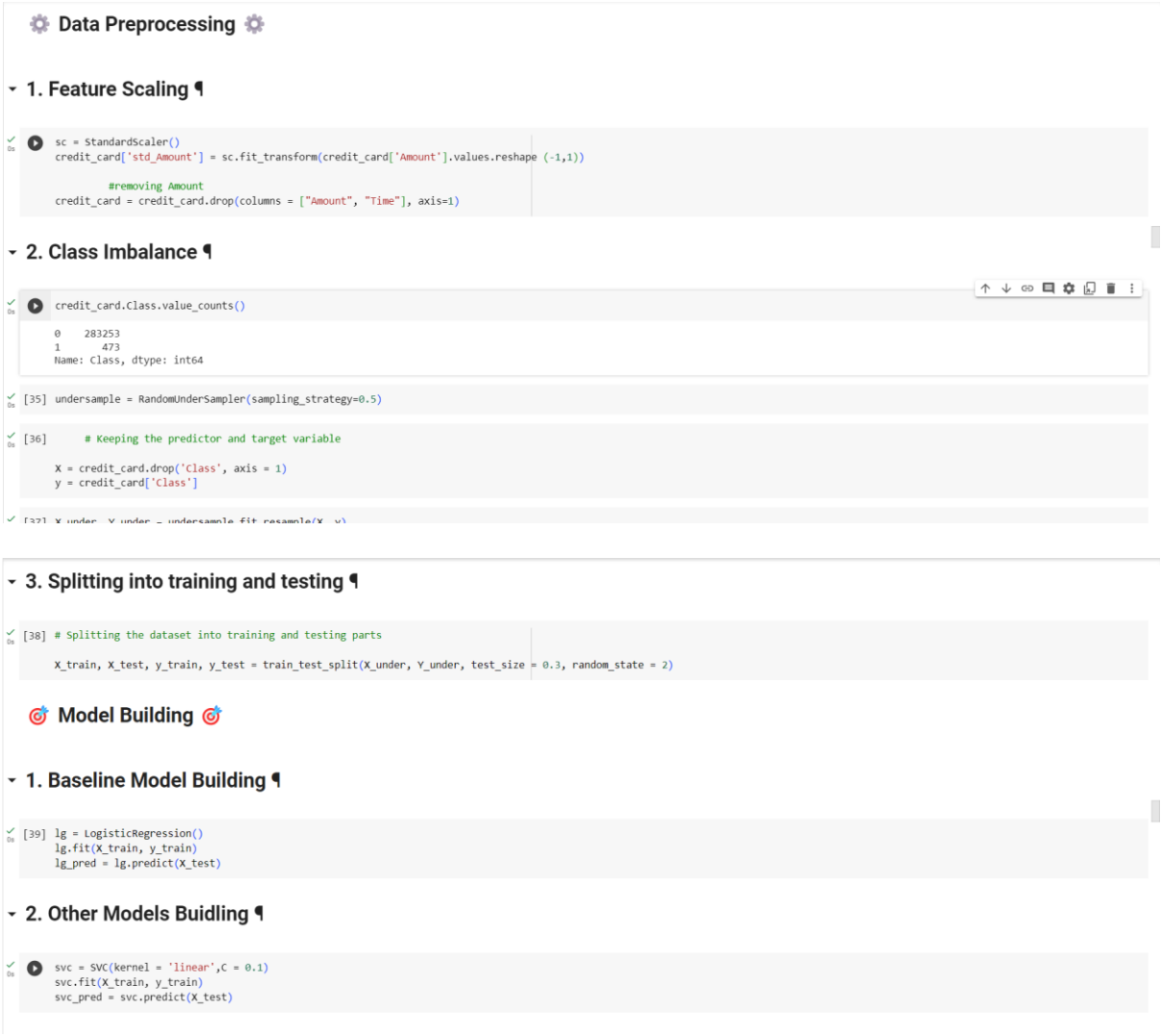
The best business model for a real-time financial fraud detection system will vary depending on the specific needs of the business. For example, businesses with a high volume of transactions may prefer a pay-per-use model, while businesses with a lower volume of transactions may prefer a subscription model.



Here are some specific examples of business models that are being used by real-time financial fraud detection companies:

1. **Forter:** Forter offers a subscription-based service that provides real-time fraud detection for online businesses.
2. **Sift Science:** Sift Science offers a subscription-based service that provides real-time fraud detection for e-commerce businesses and other online businesses.
3. **ClearSale:** ClearSale offers a pay-per-use service that provides real-time fraud detection for businesses that sell high-value goods, such as electronics and jewellery.
4. **Signifyd:** Signifyd offers a white-label service that allows businesses to offer real-time fraud detection to their customers.

By choosing the right business model, real-time financial fraud detection companies can generate revenue and help businesses protect themselves from fraud.



6. Concept Generation:

Here are some concept generation ideas for real-time financial fraud detection systems:

- **A system that uses artificial intelligence to identify fraudulent transactions based on patterns of behaviour.** This system could be trained on a large dataset of fraudulent and legitimate transactions to learn what patterns to look for.
- **A system that uses machine learning to identify fraudulent transactions based on factors such as transaction amount, source IP address, and device type.** This system could be used to identify transactions that are likely to be fraudulent, even if they do not match any specific patterns.
- **A system that uses blockchain technology to track and verify financial transactions.** This system could make it more difficult for

fraudsters to commit fraud, as their transactions would be more transparent and traceable.

- **A system that uses a combination of different technologies to identify fraudulent transactions.** For example, a system could use artificial intelligence to identify patterns of behaviour, machine learning to identify factors that are likely to be associated with fraud, and blockchain technology to track and verify transactions.

These are just a few ideas for how real-time financial fraud detection systems could be used. By continuing to innovate and develop new technologies, we can create systems that are even more effective at preventing fraud and protecting consumers.

7. Concept Development:

Here is a concept development idea for a real-time financial fraud detection system:

System: A real-time financial fraud detection system that uses a combination of artificial intelligence, machine learning, and blockchain technology to identify fraudulent transactions.

Concept: The system would use artificial intelligence to identify patterns of behaviour that are associated with fraud. For example, the system could identify users who are making multiple small transactions in a short period, or users who are making transactions from unusual locations.

The system would also use machine learning to identify factors that are likely to be associated with fraud. For example, the system could identify transactions that are above a certain amount, or transactions that are made to certain types of merchants.

Finally, the system would use blockchain technology to track and verify financial transactions. This would make it more difficult for fraudsters to commit fraud, as their transactions would be more transparent and traceable.

Use case: The system could be used by banks, payment processors, cryptocurrency exchanges, and retail stores to detect fraudulent transactions.

Next steps: The next steps would be to develop a prototype of the system and test it on a real-world dataset of fraudulent and legitimate transactions. Once the prototype is developed and tested, the system could be implemented by banks, payment processors, cryptocurrency exchanges, and retail stores.

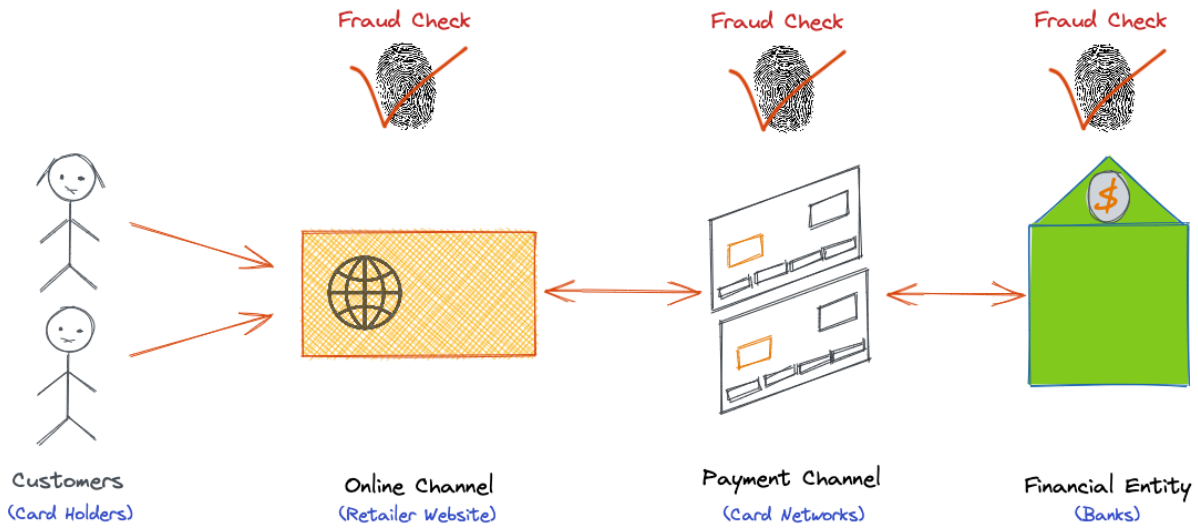
This is just one example of how a real-time financial fraud detection system could be developed. By continuing to innovate and develop new technologies, we can create systems that are even more effective at preventing fraud and protecting consumers.

8. Final Product Prototype:

- The final product prototype of the real-time financial fraud detection system would be a cloud-based system that uses artificial intelligence, machine learning, and blockchain technology to identify fraudulent transactions.
- The system would work by first collecting transaction data from banks, payment processors, cryptocurrency exchanges, and retail stores. The system would then use artificial intelligence to identify patterns of behavior that are associated with fraud. For example, the system could identify users who are making multiple small transactions in a short period, or users who are making transactions from unusual locations.
- The system would also use machine learning to identify factors that are likely to be associated with fraud. For example, the system could identify transactions that are above a certain amount, or transactions that are made to certain types of merchants.
- Finally, the system would use blockchain technology to track and verify financial transactions. This would make it more difficult for fraudsters to commit fraud, as their transactions would be more transparent and traceable.
- The system would generate a real-time risk score for each transaction. Transactions with a high-risk score would be flagged for further investigation.

Schematic Diagram

The following schematic diagram shows a high-level overview of the final product prototype:



The system would consist of the following components:

- **Data collection layer:** This layer would collect transaction data from banks, payment processors, cryptocurrency exchanges, and retail stores.
- **Data processing layer:** This layer would use artificial intelligence, machine learning, and blockchain technology to process the transaction data and generate a real-time risk score for each transaction.
- **Decision engine layer:** This layer would use the real-time risk scores to determine which transactions to flag for further investigation.
- **Alerting layer:** This layer would send alerts to businesses and consumers when fraudulent transactions are detected.

The system would be designed to be scalable and extensible. This means that it could be easily adapted to support new types of transactions and fraud schemes.

This is just a high-level overview of the final product prototype. The specific implementation details would need to be further developed.

1. Baseline Model Evaluation ¶

```
lg_cv = RepeatedStratifiedKFold(n_splits = 10, n_repeats = 4, random_state = 2)
lg_scores = cross_val_score(lg, X_train, y_train, cv = lg_cv, scoring = 'roc_auc').mean()
print(f"Mean Cross-Validation Score: {lg_scores:.2f}")
#-------------------------------------------------------------------------------

print(f"The ROC AUC Score is: {roc_auc_score(y_test, lg_pred):.2f}", '\n\n\n')
#-------------------------------------------------------------------------------

lg_report = classification_report(y_test, lg_pred)
print('The classification report of Logistic Regressor is below : ', '\n\n\n', lg_report, '\n')
#-------------------------------------------------------------------------------

fpr, tpr, _ = roc_curve(y_test, lg.predict_proba(X_test)[:, 1])
roc_auc = auc(fpr, tpr)
fig = make_subplots(rows=1, cols=1, subplot_titles=['ROC AUC Plot'])
fig.add_trace(go.Scatter(x=fpr, y=tpr, mode='lines', line=dict(color='#4A1A22')))
fig.update_layout(xaxis_title='False Positive Rate', yaxis_title='True Positive Rate', plot_bgcolor='white')
fig.add_annotation(x=0.5, y=0.5, text='AUC = %0.2f' % roc_auc, showarrow=False, font=dict(color='#4A1A22'))
pio.show(fig)
#-------------------------------------------------------------------------------

print('The confusion matrix is below : ')
cm = confusion_matrix(y_test, lg_pred)
names = ['True Negatives', 'False Positives', 'False Negatives', 'True Positives']
counts = [value for value in cm.flatten()]
percentages = ['{0:.2%}'.format(value) for value in cm.flatten()/np.sum(cm)]
labels = [f'{v1}\n{v2}\n{v3}' for v1, v2, v3 in zip(names, counts, percentages)]
labels = np.asarray(labels).reshape(2, 2)
cmap = sns.light_palette("#4A1A22", as_cmap=True)
sns.heatmap(cm, annot = labels, cmap = cmap, fmt = '')
```

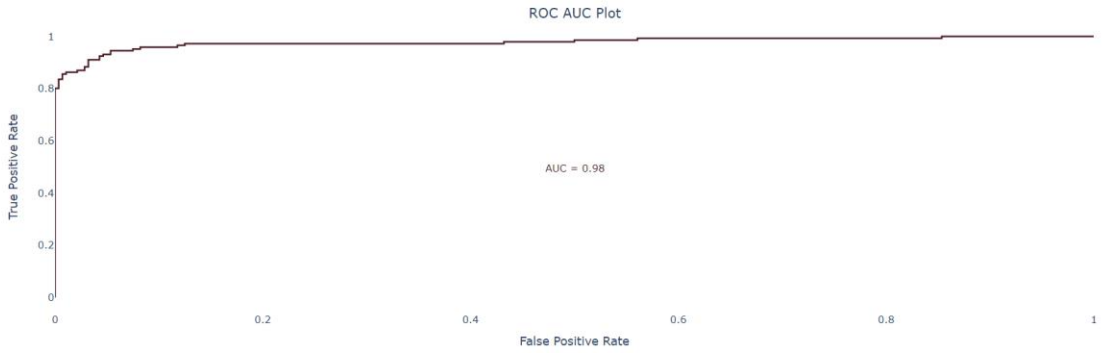
```
Mean Cross-Validation Score: 0.97
```

```
roc_auc = auc(fpr, tpr)
fig = make_subplots(rows=1, cols=1, subplot_titles=['ROC AUC Plot'])
fig.add_trace(go.Scatter(x=fpr, y=tpr, mode='lines', line=dict(color='#4A1A22')))
fig.update_layout(xaxis_title='False Positive Rate', yaxis_title='True Positive Rate', plot_bgcolor='white')
fig.add_annotation(x=0.5, y=0.5, text='AUC = %0.2f' % roc_auc, showarrow=False, font=dict(color='#4A1A22'))
pio.show(fig)
#-------------------------------------------------------------------------------

print('The confusion matrix is below : ')
cm = confusion_matrix(y_test, lg_pred)
names = ['True Negatives', 'False Positives', 'False Negatives', 'True Positives']
counts = [value for value in cm.flatten()]
percentages = ['{0:.2%}'.format(value) for value in cm.flatten()/np.sum(cm)]
labels = [f'{v1}\n{v2}\n{v3}' for v1, v2, v3 in zip(names, counts, percentages)]
labels = np.asarray(labels).reshape(2, 2)
cmap = sns.light_palette("#4A1A22", as_cmap=True)
sns.heatmap(cm, annot = labels, cmap = cmap, fmt = '')
```

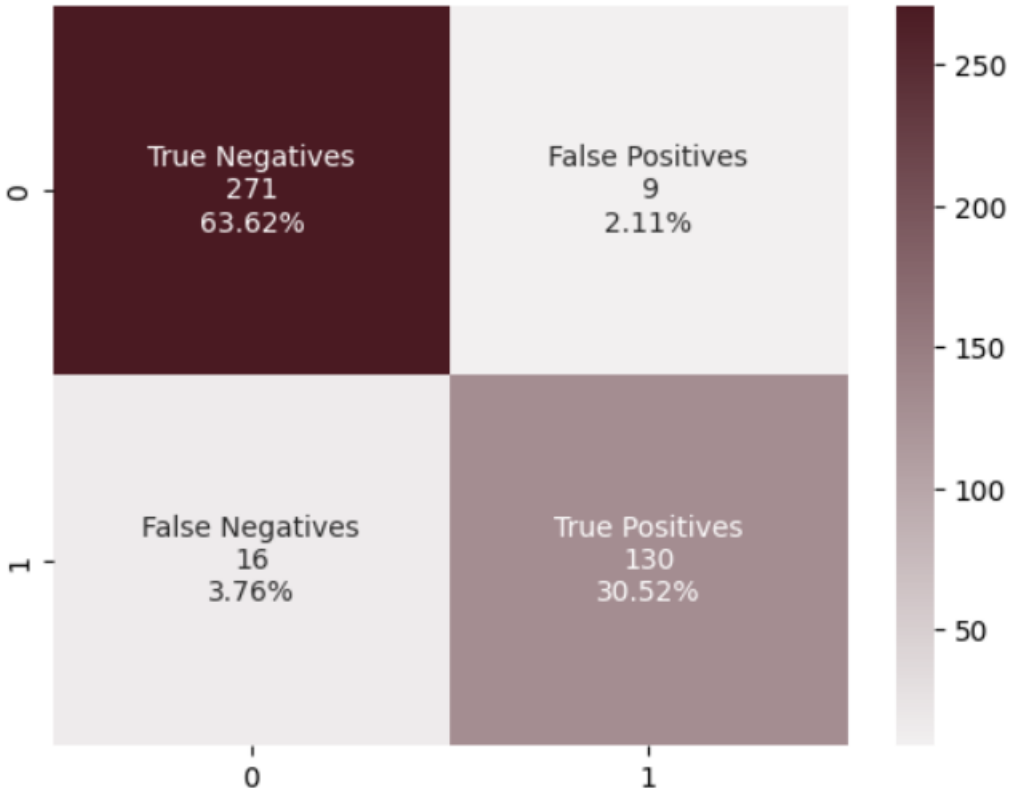
```
Mean Cross-Validation Score: 0.97
The ROC AUC Score is: 0.93


The classification report of Logistic Regressor is below :


              precision    recall  f1-score   support

           0       0.94      0.97      0.96       280
           1       0.94      0.89      0.91       146

    accuracy                           0.94       426
   macro avg       0.94      0.93      0.93       426
weighted avg       0.94      0.94      0.94       426
```

```
labels = np.asarray(labels).reshape(2, 2)
cmap = sns.light_palette("#4A1A22", as_cmap=True)
sns.heatmap(cm, annot = labels, cmap = cmap, fmt = '')
```

```
   macro avg       0.94      0.93      0.93       426
weighted avg       0.94      0.94      0.94       426
```



The confusion matrix is below :
<Axes: >



2. SVC Model Evaluation ¶

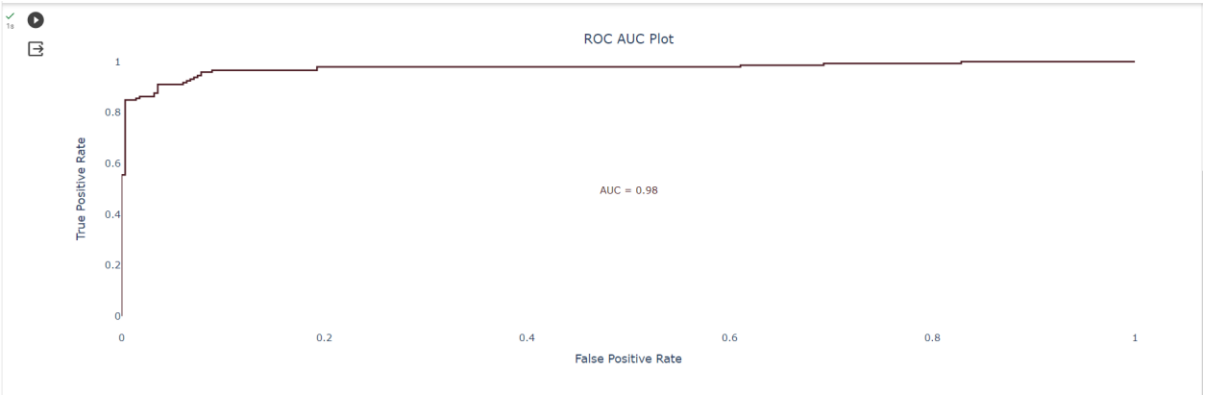
```
svc_cv = RepeatedStratifiedKFold(n_splits = 10, n_repeats = 4, random_state = 2)
svc_scores = cross_val_score(svc, X_train, y_train, cv = svc_cv, scoring = 'roc_auc').mean()
print(f"Mean Cross-Validation Score: {lg_scores:.2f}")
#-------------------------------------------------------------------------

print(f"The ROC AUC Score is: {roc_auc_score(y_test, lg_pred):.2f}", '\n\n\n')
#-------------------------------------------------------------------------

svc_report = classification_report(y_test, svc_pred)
print('The classification report of Logistic Regressor is below : ', '\n\n\n', svc_report, '\n')
#-------------------------------------------------------------------------

fpr, tpr, _ = roc_curve(y_test, svc.decision_function(X_test))
roc_auc = auc(fpr, tpr)
fig = make_subplots(rows=1, cols=1, subplot_titles=['ROC AUC Plot'])
fig.add_trace(go.Scatter(x=fpr, y=tpr, mode='lines', line=dict(color='#4A1A22')))
fig.update_layout(xaxis_title='False Positive Rate', yaxis_title='True Positive Rate', plot_bgcolor='white')
fig.add_annotation(x=0.5, y=0.5, text='AUC = %0.2f' % roc_auc, showarrow=False, font=dict(color='#4A1A22'))
pio.show(fig)
#-------------------------------------------------------------------------

print('The confusion matrix is below : ', '\n')
cm = confusion_matrix(y_test, svc_pred)
names = ['True Negatives', 'False Positives', 'False Negatives', 'True Positives']
counts = [value for value in cm.flatten()]
percentages = ['{0:.2%}'.format(value) for value in cm.flatten()/np.sum(cm)]
labels = [f'{v1}\n{v2}\n{v3}' for v1, v2, v3 in zip(names, counts, percentages)]
labels = np.asarray(labels).reshape(2, 2)
cmap = sns.light_palette("#4A1A22", as_cmap=True)
sns.heatmap(cm, annot = labels, cmap = cmap, fmt = '')
```

```
Mean Cross-Validation Score: 0.97
```

Mean Cross-Validation Score: 0.97
The ROC AUC Score is: 0.93

The classification report of Logistic Regressor is below :

	precision	recall	f1-score	support
0	0.93	0.98	0.95	280
1	0.95	0.86	0.91	146
accuracy			0.94	426
macro avg	0.94	0.92	0.93	426
weighted avg	0.94	0.94	0.94	426

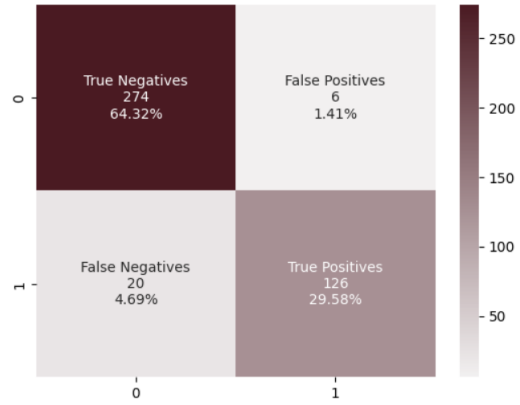


False Positive Rate

[42]

The confusion matrix is below :

<Axes: >



Insights:

- The performance evaluation revealed that the Logistic Regression performed well and gave an f1-score of 0.96. The precision-recall curve is indication a great performance with almost touching the top right corner having recall on x axis and precision on y axis

9. Financial equation:

Total Cost = Detection Costs + Fraud Prevention and Loss Mitigation costs

Total Benefits = Fraud Avoidance + Reduced Chargebacks + Customer Experience and Reputation Benefits

Net Financial Impact = Total Benefits – Total Cost

In the context of the financial equation for a credit card fraud detection system, if we want to represent it in the form of (y = mx + c), where (y) is the dependent variable (Net Financial Impact), (x) is an independent variable (Total Benefits), and (c) is a constant term representing the intercept, we can rearrange the equation as follows:

Let:

- y represents the Net Financial Impact.

- x represents the Total Benefits.

- m represents the slope or coefficient associated with the Total Benefits.

- c represents the constant term or intercept.

The equation in the form of (y = mx + c) would be:

$y = mx + c$

Given:

$\{Total\ Cost\} = C_d + C_p$

$\{Total\ Benefits\} = B_f + B_c + B_e$

$\{Net\ Financial\ Impact\} = \{Total\ Benefits\} - \{Total\ Cost\}$

We rearrange the equation to fit the form (y = mx + c):

$\{Net\ Financial\ Impact\} = (B_f + B_c + B_e) - (C_d + C_p)$

This equation can be represented as (y = mx + c) by defining:

- y as Net Financial Impact.

- x as Total Benefits.

- mas 1 (since it's a 1:1 relationship between Total Benefits and Net Financial Impact).

- c as $-(C_d + C_p)$ (the constant term or intercept).

So, in the form (y = mx + c), the equation for the financial impact of a credit card fraud detection system could be written as:

$y = x - (C_d + C_p)$

This representation reflects the relationship between the Net Financial Impact and the Total Benefits of implementing the fraud detection system, considering the costs associated with detection and prevention.

10. Conclusion:

A real-time financial fraud detection system, using artificial intelligence, machine learning, and blockchain technology, could significantly protect businesses and consumers from financial losses. It would be scalable and extensible, allowing for easy adaptation to new transactions and fraud schemes. However, the system's development and deployment require careful consideration of cost, complexity, and implementation challenges.

Real-time financial fraud detection systems are crucial for safeguarding financial transactions and protecting organizations and consumers from fraudulent activities. These systems use machine learning algorithms and rules-based logic to process data from various sources and generate alerts when suspicious activities are detected. A diverse team of experts is needed to develop and deploy the system, considering factors like data sources, technology stack, team composition, and cost estimation. Continuous improvement and adaptation are essential for staying ahead of emerging fraud tactics.