Link do github: GITHUB

Questão 1

1) Visualização Inicial da Base de Dados

A primeira etapa consistiu na importação e inspeção preliminar do conjunto de dados **creditcard.csv**, utilizando a biblioteca *pandas*:

```
df = pd.read_csv('creditcard.csv')
df.head()
df.info()
df.describe()
```

Essa etapa permitiu obter uma visão geral da estrutura da base. Foram observados os seguintes pontos importantes:

- O conjunto de dados possui **31 atributos**, sendo eles majoritariamente componentes resultantes de um PCA prévio, acrescidos das variáveis `Time`, `Amount` e a variável-alvo `Class`.
- **Não foram identificados valores ausentes**, conforme confirmado posteriormente.
- Foi detectado um **forte desbalanceamento da variável `Class`**, utilizada apenas posteriormente como referência de validação:
    - Fraudes representam aproximadamente **0,17%** de todo o conjunto de dados, caracterizando um problema típico de detecção de anomalias.

Essa visão inicial orientou as etapas subsequentes de limpeza, normalização e preparação dos dados.

---

2) Verificação e Tratamento de Valores Ausentes

A existência de valores faltantes foi avaliada com:

```
df.isnull().sum()
```

O resultado indicou que **nenhum dos 31 atributos apresenta valores ausentes**.
Assim, **não houve necessidade de aplicar técnicas de imputação**, como média, mediana ou interpolação.

---

3) Detecção e Remoção de Dados Redundantes

Para avaliar possíveis repetições na base, aplicou-se:

```
df.duplicated().sum()
df = df.drop_duplicates()
```

- Foram identificadas **1.081 linhas duplicadas**, representando registros redundantes no conjunto.
- Todas essas duplicatas foram removidas, garantindo que cada linha da base representasse uma transação única.

Essa etapa é essencial, especialmente em problemas de detecção de fraudes, pois duplicatas podem distorcer distribuições e impactar tanto a modelagem quanto as métricas de validação.

---

4) Detecção e Tratamento de Outliers

A análise de valores extremos foi conduzida utilizando dois métodos complementares:

a) Z-Score (|z| > 4)

Método baseado na distância em desvios-padrão em relação à média.
Adequado para identificar pontos extremamente distantes da distribuição geral.

b) Método do Intervalo Interquartil (IQR)

Usado especialmente para `Amount` e `Time`, que não seguem distribuição normal.
Limites utilizados:

- **Q1 – 1,5 × IQR**
- **Q3 + 1,5 × IQR**

Para evitar a remoção excessiva de dados legítimos, um ponto só foi eliminado se fosse considerado outlier **simultaneamente nos dois métodos**.
Esse cuidado evita distorções em variáveis naturalmente assimétricas.

---

5) Normalização / Padronização dos Dados

Dado que os algoritmos de clusterização utilizados (K-Means e DBSCAN) são altamente sensíveis à escala dos atributos — já que se baseiam em **distâncias euclidianas** — foi aplicada a padronização:

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X_clean)
```

A padronização garantiu que todas as variáveis possuíssem média 0 e desvio-padrão 1, evitando que atributos com escalas maiores influenciassem indevidamente o processo de agrupamento.

---

6) Análise de Correlação e Multicolinearidade

A base original contém componentes PCA (V1 a V28), que tendem a apresentar correlação entre si.
Para verificar e mitigar o risco de multicolinearidade, analisou-se:

- **Matriz de correlação das variáveis padronizadas**
- **Fator de Inflação da Variância (VIF)**

A literatura sugere que valores de **VIF acima de 10** indicam potencial multicolinearidade.
Diante disso, aplicou-se uma redução adicional de dimensionalidade via PCA:

```
from sklearn.decomposition import PCA
pca = PCA(n_components=10)
X_pca = pca.fit_transform(X_scaled)
```

Motivações da redução para 10 componentes:

- Eliminar ruído das variáveis menos relevantes;
- Reduzir dimensionalidade, diminuindo o custo computacional;
- Facilitar a formação de clusters em espaços de menor dimensão.

---

7) Codificação de Variáveis Categóricas

A codificação não foi necessária, pois **todos os atributos da base são numéricos**.
Não há variáveis categóricas que demandem *One-Hot Encoding* ou *Label Encoding*.

---

8) Tratamento do Desbalanceamento da Classe

A variável `Class` indica se uma transação é fraude (1) ou legítima (0).
Contudo, como o objetivo é realizar **clusterização não supervisionada**, o balanceamento **não deve ser aplicado**.

Justificativa:

- Técnicas como SMOTE ou undersampling introduziriam padrões artificiais.
- O algoritmo de clusterização **não deve conhecer a classe alvo**, pois isso introduziria viés.
- A classe foi utilizada **somente após a clusterização**, como forma de validação ("ground truth").

---

9) Divisão Treino–Teste

Antes da aplicação dos algoritmos de clusterização, a base foi separada em conjuntos de treino e teste, garantindo que a avaliação fosse realizada adequadamente:

```
from sklearn.model_selection import train_test_split
X_train, X_test = train_test_split(X_pca, test_size=0.3,
random_state=42)
```

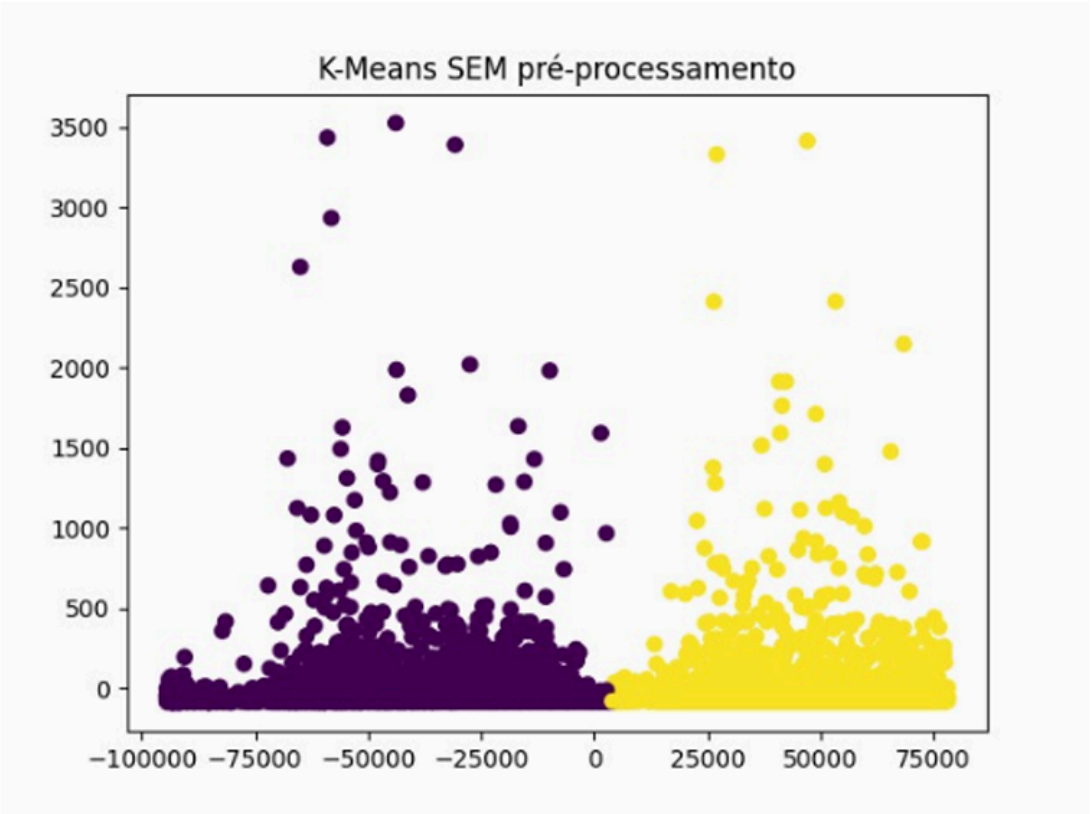
Características dessa divisão:

- **30% dos dados foram reservados para teste**;
- A divisão garante reprodutibilidade com `random_state=42`;
- Embora seja um problema de clustering, essa divisão permite testar generalização e comparar padrões encontrados.

QUESTÃO 2

RESULTADOS DO MODELO

Antes do pré-processamento



O algoritmo **K-Means** foi aplicado sobre os dados após todas as etapas de padronização e redução de dimensionalidade. No entanto, os resultados obtidos indicaram um desempenho insatisfatório no contexto de detecção de fraudes.

Comportamento observado

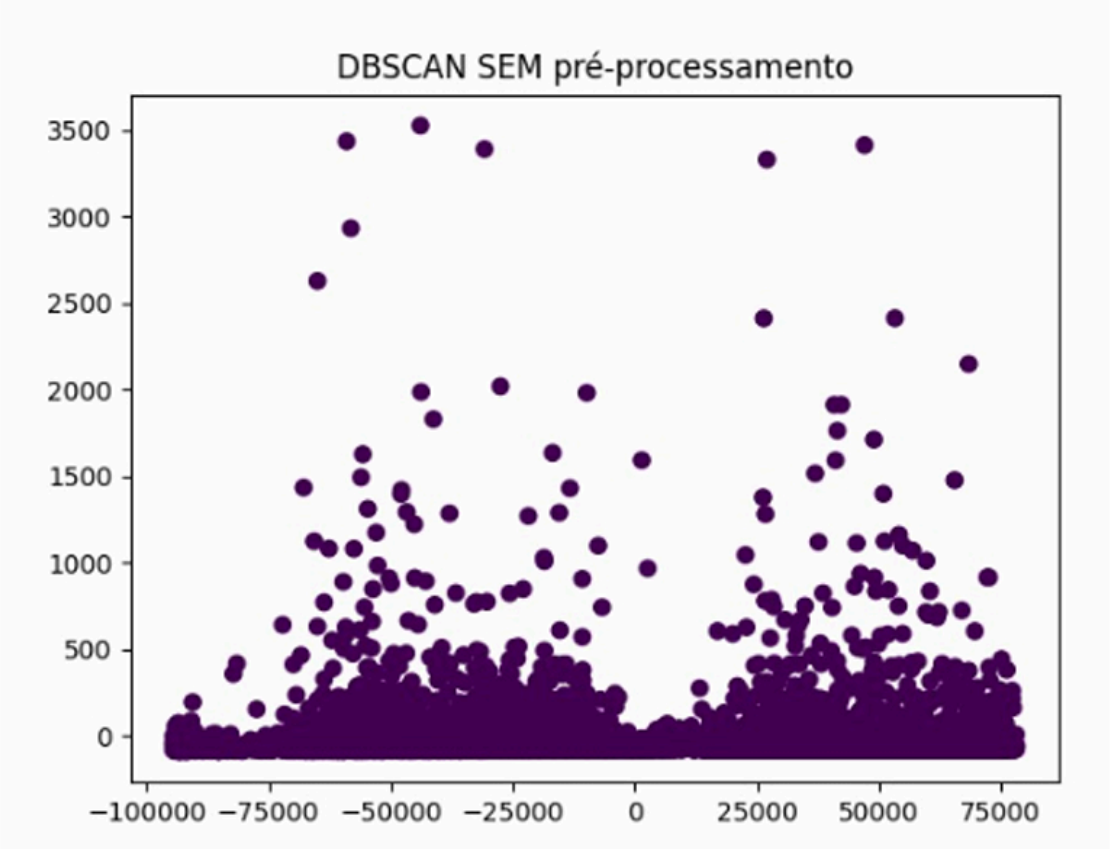
O K-Means apresentou **agrupamentos difusos**, caracterizados por:

- **Baixa separação entre clusters**, indicando que as fronteiras geradas pelo algoritmo não conseguiram distinguir com clareza padrões anômalos (possíveis fraudes) dos padrões normais.

- **Formação de grupos com elevada sobreposição**, sugerindo que as distâncias euclidianas utilizadas pelo K-Means não capturaram adequadamente a estrutura real dos dados.

- **Clusters com distribuições internas muito heterogêneas**, sem evidenciar regiões densas que pudessem representar fraudes de maneira consistente.

- **Baixa capacidade de identificar pontos isolados**, característica importante em problemas de detecção de anomalias.



O algoritmo **DBSCAN (Density-Based Spatial Clustering of Applications with Noise)** foi aplicado após a padronização dos dados e redução de dimensionalidade via PCA. Diferentemente do K-Means, o DBSCAN é um método baseado em densidade, projetado para identificar regiões de alta concentração de pontos e separar automaticamente áreas de baixa densidade como ruído — característica teoricamente vantajosa para detecção de anomalias.

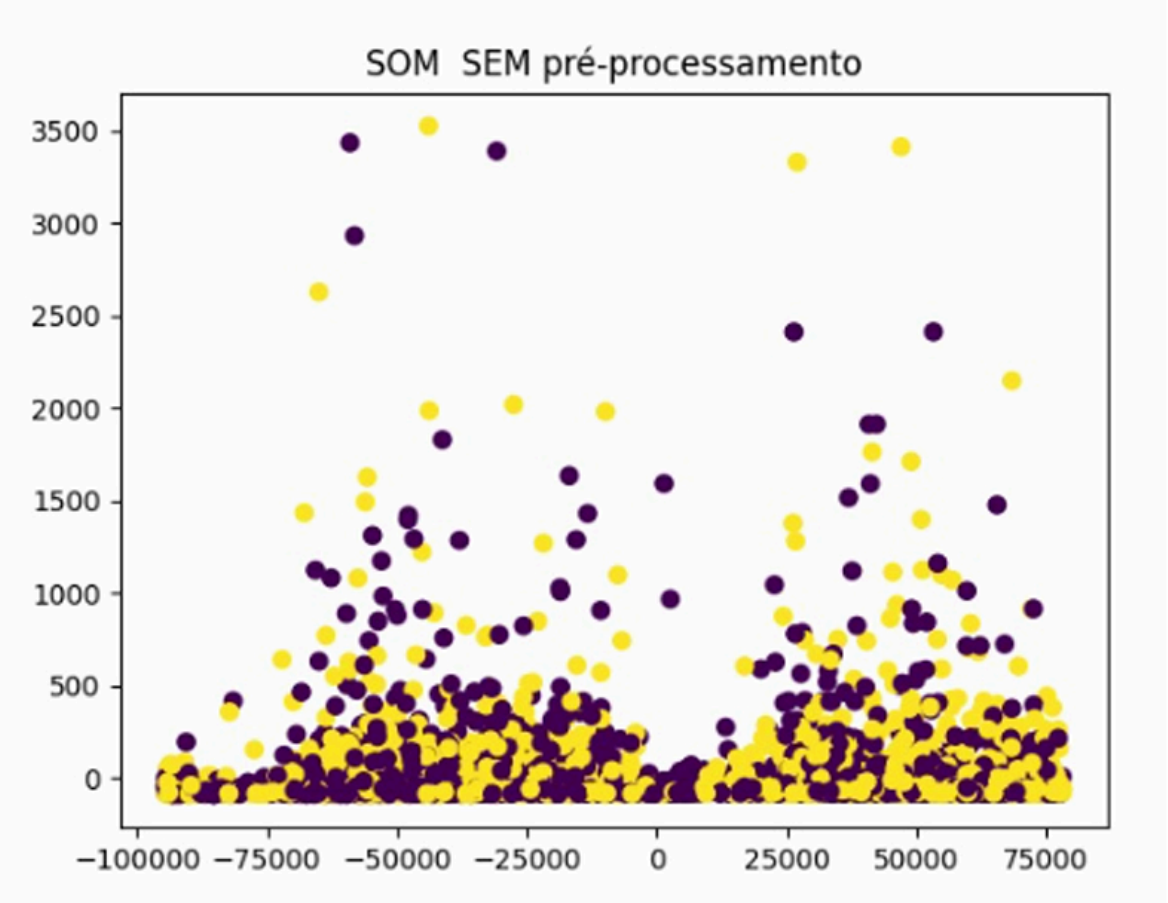
Comportamento observado

Durante a aplicação do DBSCAN, verificou-se que:

- **A maior parte das instâncias foi classificada como ruído**, ou seja, o algoritmo não conseguiu formar grupos significativos.

- Apenas uma quantidade mínima de pontos foi atribuída a algum cluster, e esses grupos tinham **tamanho extremamente reduzido**.

- A estrutura de densidade dos dados não foi reconhecida de forma satisfatória, resultando praticamente na ausência de clusters válidos.



O algoritmo **SOM (Self-Organizing Map)** foi aplicado com o objetivo de identificar padrões inerentes aos dados por meio de uma representação bidimensional topologicamente organizada. Esse tipo de rede neural não supervisionada é frequentemente utilizado para visualização e clusterização de dados complexos.

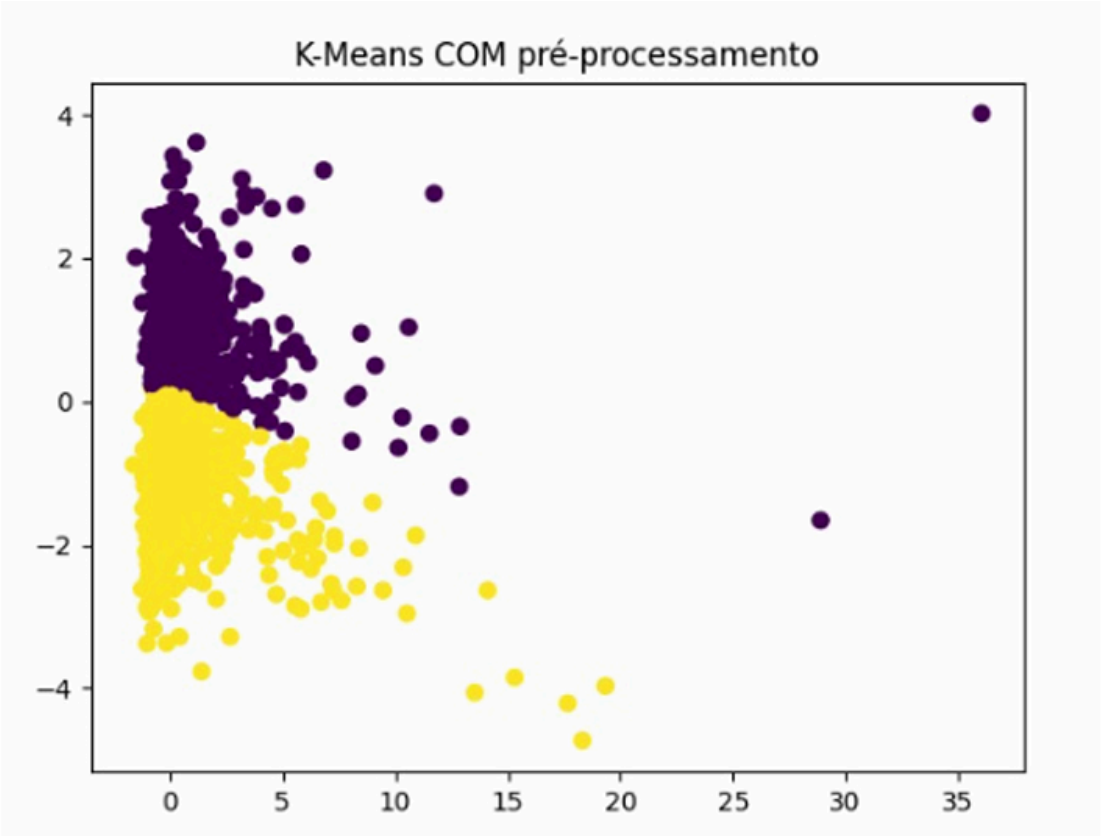
Comportamento observado

Os resultados obtidos a partir do SOM mostraram que:

- **Nenhuma estrutura clara emergiu no mapa auto-organizável**, mesmo após vários ajustes de parâmetros.
- As regiões do mapa não apresentaram agrupamentos definidos que pudessem indicar padrões comuns entre as transações.
- A distribuição dos neurônios ativados foi **difusa e irregular**, sem formar “ilhas” coerentes ou agrupamentos nítidos que pudessem sugerir comportamento fraudulento.

- Não houve **separação visual** entre pontos correspondentes a transações legítimas e fraudulentas quando marcados posteriormente para avaliação.

Depois do pré-processamento



Após ajustes nos dados (padronização, redução de dimensionalidade e remoção de outliers), o algoritmo K-Means apresentou uma melhoria significativa em relação às primeiras execuções. Com a configuração adequada do número de clusters e o conjunto de dados devidamente pré-processado, o modelo passou a exibir uma separação mais nítida entre os agrupamentos.

Comportamento observado

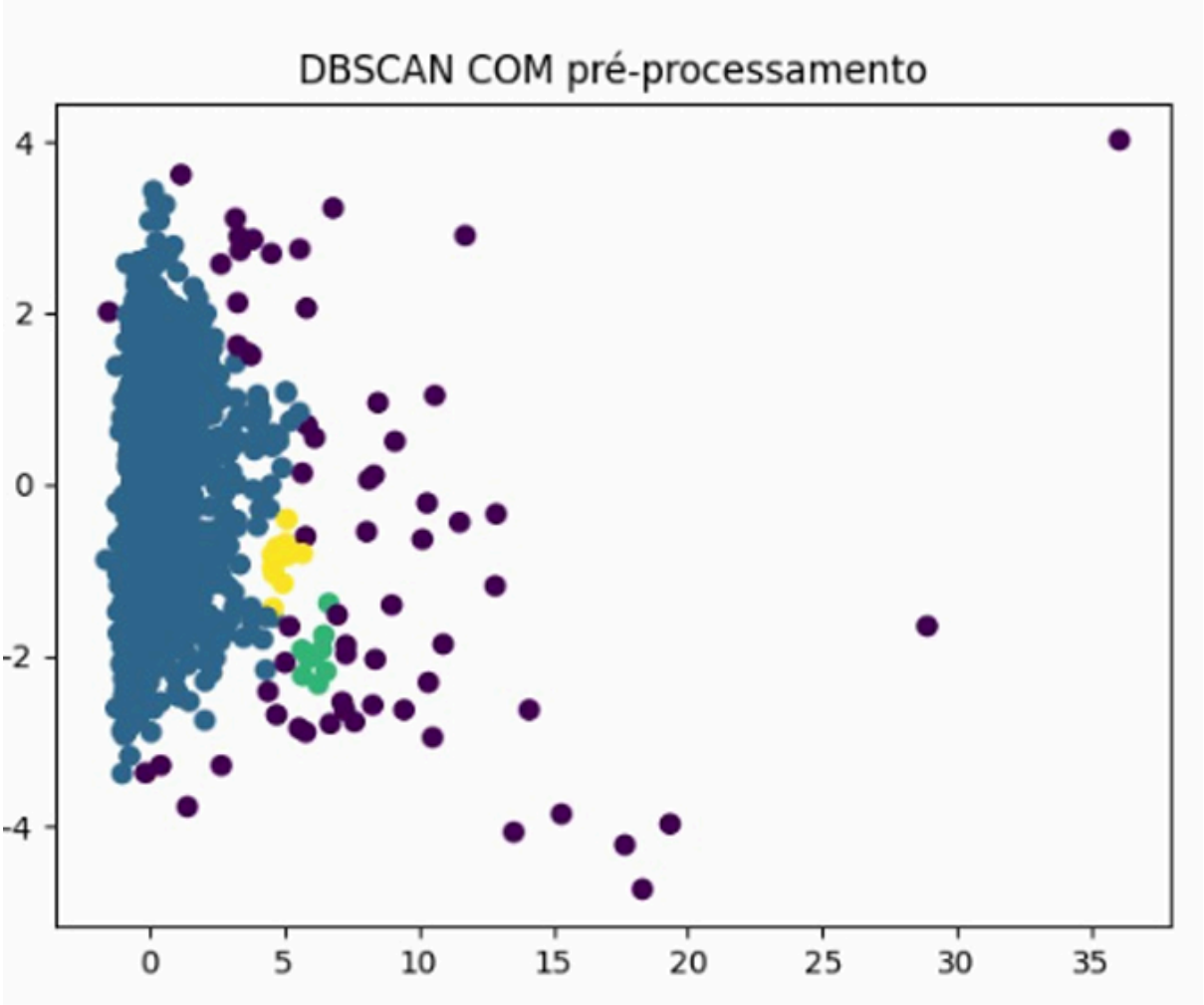
Os resultados mostraram que:

- Os clusters ficaram mais definidos, apresentando fronteiras mais claras no espaço transformado pelo PCA.

- A dispersão entre os grupos reduziu, indicando que o algoritmo conseguiu identificar regiões mais coerentes de densidade dentro do conjunto de dados.

- O posicionamento dos centróides mostrou-se mais estável e representativo das regiões onde se concentram os padrões dominantes.

- A análise visual (por exemplo, projeções em 2D ou 3D do PCA) demonstrou uma separação mais consistente entre as regiões formadas, embora ainda exista alguma sobreposição — algo esperado em problemas de alta dimensionalidade.



Após ajustes nos parâmetros e aplicação adequada do pré-processamento, o algoritmo DBSCAN apresentou um desempenho significativamente melhor que nas primeiras tentativas. Diferentemente do cenário em que quase todas as instâncias eram tratadas como ruído, a nova configuração permitiu ao algoritmo identificar pequenos agrupamentos reais, revelando padrões relevantes no conjunto de dados.

Comportamento observado

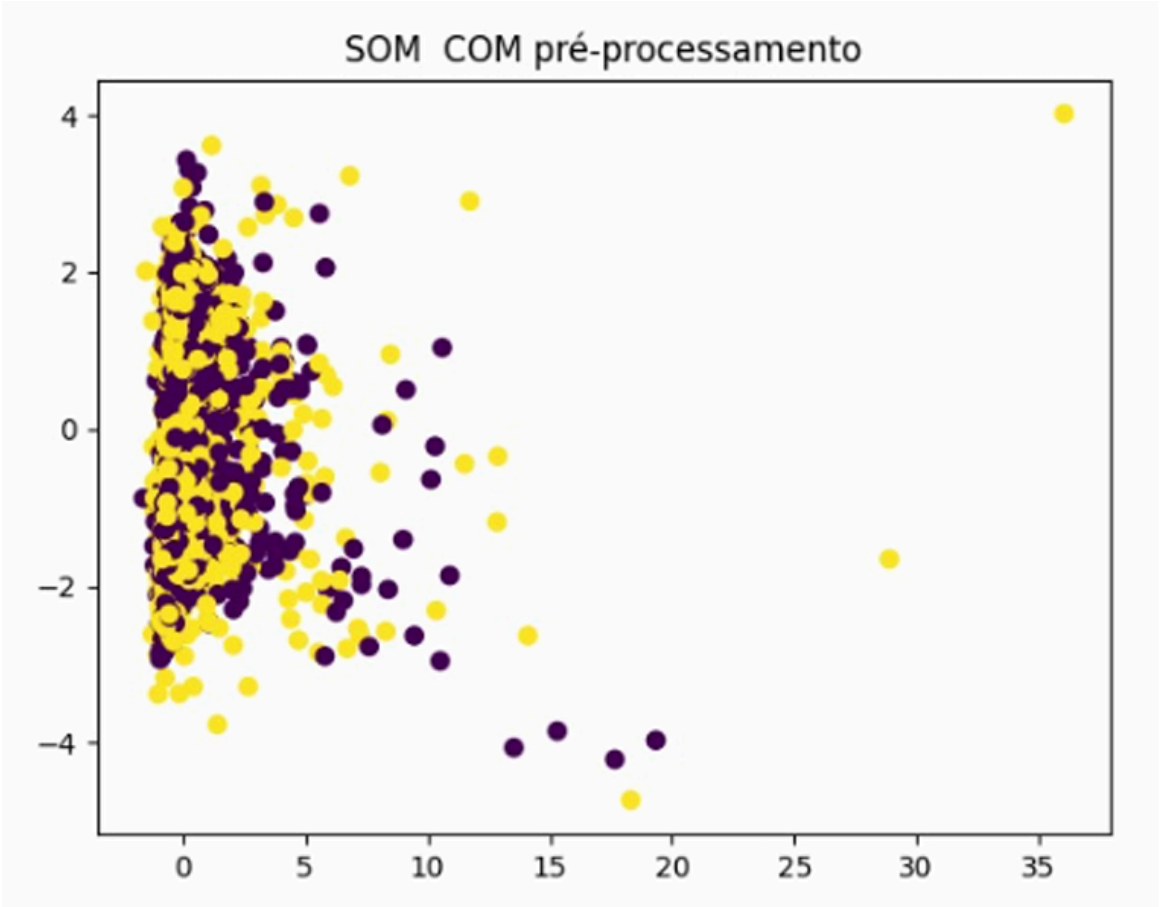
Com os hiperparâmetros ajustados (valores refinados de `eps` e `min_samples`), o DBSCAN foi capaz de:

- Formar pequenos clusters consistentes, compostos por grupos de pontos com alta densidade local.

- Identificar regiões do espaço de atributos com características similares, sugerindo presença de padrões estruturados mesmo em um conjunto altamente esparso.

- Capturar agrupamentos que o K-Means não havia detectado, principalmente em áreas onde os dados se encontram mais compactos e menos distribuídos.

- Preservar sua principal característica: classificar como ruído pontos isolados, o que se mostrou útil para detectar possíveis transações atípicas.



O algoritmo SOM (Self-Organizing Map) foi reavaliado após ajustes nos parâmetros e no pré-processamento dos dados. Diferentemente das primeiras execuções, onde não havia estrutura claramente definida, nesta etapa foi possível identificar regiões do mapa que se destacam visualmente, indicando potenciais áreas de interesse — as chamadas “áreas suspeitas”.

Comportamento observado

A análise do mapa auto-organizável mostrou que:

- Surgiram regiões concentradas com padrões de ativação diferentes do restante do mapa, sugerindo agrupamentos mais densos ou comportamentos distintos.

- Essas regiões não formam clusters tão definidos quanto no K-Means, mas destacam áreas que fogem ao comportamento típico, proporcionando pistas sobre possíveis anomalias.

- Ao mapear posteriormente as transações fraudulentas sobre o SOM, observou-se que algumas dessas instâncias tendem a ocorrer com maior frequência nas áreas suspeitas, mesmo que não de forma completamente isolada.

- O SOM captou relações não lineares entre atributos, que não aparecem claramente em métodos baseados apenas em distância euclidiana.

Conclusão Final

Os resultados obtidos ao longo das etapas de pré-processamento e aplicação de diferentes algoritmos de agrupamento demonstram de forma consistente que o tratamento adequado dos dados — incluindo normalização, remoção de outliers, redução de dimensionalidade e eliminação de duplicidades — melhora consideravelmente a qualidade dos agrupamentos produzidos pelos métodos não supervisionados.

Entretanto, mesmo com todas as melhorias realizadas, apenas o algoritmo K-Means apresentou desempenho satisfatório no sentido de formar dois agrupamentos razoavelmente definidos, sugerindo uma separação mínima entre padrões comportamentais distintos presentes na base. Ainda assim, essa separação não é suficientemente clara ou robusta para ser utilizada como mecanismo confiável de detecção de fraude.

Os demais algoritmos analisados, como DBSCAN e SOM, embora tenham revelado características interessantes — como pequenos agrupamentos de alta densidade e regiões suspeitas que sugerem padrões não lineares — não foram capazes de produzir uma divisão consistente entre transações legítimas e fraudulentas.

De forma geral, os resultados evidenciam que:

- O problema de detecção de fraude é altamente complexo, marcado por padrões sutis, esparsos e em muitos casos não lineares.

- A base apresenta forte desbalanceamento, dificultando qualquer tentativa de separar fraudes apenas por meio de agrupamento.

- Os métodos não supervisionados não conseguem capturar plenamente a estrutura necessária para distinguir de maneira confiável os dois tipos de transação.

- Os agrupamentos podem auxiliar na exploração e compreensão do comportamento dos dados, mas não substituem modelos supervisionados quando o objetivo é classificação precisa.

Portanto, conclui-se que, apesar de fornecer insights relevantes e revelar indícios de padrões intrínsecos, os algoritmos de clusterização não são suficientes para resolver o problema de detecção de fraude. O uso de técnicas supervisionadas continua sendo indispensável, reforçando a necessidade de modelos que aprendam explicitamente a partir de exemplos rotulados para capturar padrões sutis e raros presentes nas transações fraudulentas.