Detecção de Anomalias, Shill Bidders e Leilões Online

1nd Eduarda Vitoria Albuquerque Sales
Centro de Informática - UFPE
Recife, Brazil
evas@cin.ufpe.br

2nd Julia Zovka de Souza
Centro de Informática - UFPE
Recife, Brazil
jzs@cin.ufpe.br

3rd Leticia Andrade de Oliveira Barros
Centro de Informática - UFPE
Recife, Brazil
laob@cin.ufpe.br

4rd Mariana Beatriz Campelo Ferreira
Centro de Informática - UFPE
Recife, Brazil
mbfc@cin.ufpe.br

***Abstract*—Neste trabalho, utilizamos um conjunto de dados já existente contendo um dados sobre lances de leilões do eBay em cima de Iphones 7. Após realizar uma análise exploratória aprofundada (EDA) e aplicar técnicas de pré-processamento, preparamos o SHillBidding dataset para possibilitar experimentação e análise. Em seguida, treinamos três modelos de aprendizado de máquina — K-Means, Isolation Forest e Autoencoders — com o objetivo de identificar padrões e detectar anomalias nas dinâmicas dos lancistas dos leilões, diferenciando os compradores normais de shill bidders.**

I. Introdução

O crescimento das plataformas de comércio eletrônico e o alto volume de transações realizadas diariamente tornam os ambientes de leilões online, como o eBay, um objeto relevante de estudo. A análise desses dados permite compreender padrões de comportamento, identificar possíveis anomalias e apoiar pesquisas relacionadas ao mercado digital.

Leilões, como os realizados na plataforma eBay, estão particularmente vulneráveis a um tipo de fraude conhecido como Shill Bidding. Nessa prática, o próprio vendedor ou indivíduos associados a ele utilizam contas falsas para realizar lances que não têm a intenção real de adquirir o produto, sendo o objetivo apenas elevar artificialmente o preço final,.

A escolha deste dataset se justifica por três fatores principais: (i) sua relevância prática para estudos de fraude em plataformas digitais; (ii) sua estrutura limpa e sem valores faltantes, facilitando o pré-processamento; e (iii) sua ampla utilização na literatura, o que o torna adequado para comparações e validações de técnicas de aprendizado de máquina.

II. Análise de Dados e Feature Engineering

A. Análise Exploratória dos Dados

O conjunto de dados utilizado neste trabalho é o Shill Bidding Dataset, disponibilizado publicamente no UCI Machine Learning Repository. Esse dataset foi construído a partir da coleta (scraping) de um grande número de leilões do eBay relacionados a um produto popular. Após o pré-processamento realizado pelos autores originais, o SB dataset foi disponibilizado para apoiar a pesquisa acadêmica sobre comportamentos fraudulentos em plataformas de leilões online.

O dataset contém 6321 instâncias e possui 13 atributos, sem valores ausentes. Ele inclui informações relacionadas ao comportamento dos participantes nos leilõe. A base é adequada tanto para tarefas de classificação quanto de agrupamento, conforme indicado na própria documentação, o que motivou o uso de técnicas de aprendizado não supervisionado neste trabalho (K-Means, Isolation Forest e Autoencoders). A fonte oficial do dataset é:

UCI Machine Learning Repository – Shill Bidding Dataset DOI: 10.24432/C5ZG11 Licença: Creative Commons Attribution 4.0 International (CC BY 4.0)

As features representam :

- `Record_ID`: identificador único do registro.
- `Auction_ID`: identificador do leilão.
- `Bidder_ID`: identificador do participante.
- `Bidder_Tendency`: mede se o usuário participa apenas dos leilões de poucos vendedores.
- `Bidding_Ratio`: frequência relativa de participações do usuário nos lances.
- `Successive_ Outbidding`: indica se o usuário dá lances contra ele mesmo.
- `Last_Bidding`: indica se o usuário para de participar no final do leilão.
- `Auction_Bids`: número total de lances no leilão.
- `Auction_Starting_Price`: preço inicial do leilão.
- `Early Bidding`: mede se o usuário dá lances muito cedo no leilão.
- `Winning_Ratio`: porcentagem de leilões vencidos pelo usuário em relação aos que participou.
- `Auction_Duration`: duração total do leilão.
- `Class`: indica se o comportamento é normal (0) ou suspeito de Shill Bidding (1).

1) Análise Exploratória Estrutural: A verificação inicial da estrutura do *dataset* de leilões do eBay foi realizada. O *dataframe* `df` possui **6321 entradas** e **13 colunas**.

A estrutura e os tipos de dados são apresentados a seguir:

```
RangeIndex: 6321 entries, 0 to 6320
Data columns (total 13 columns):
 0   Record_ID              int64
 1   Auction_ID             int64
 2   Bidder_ID              object
 3   Bidder_Tendency        float64
 4   Bidding_Ratio          float64
 5   Successive_Outbidding  float64
 6   Last_Bidding           float64
 7   Auction_Bids           float64
 8   Starting_Price_Average float64
 9   Early_Bidding          float64
 10  Winning_Ratio          float64
 11  Auction_Duration       int64
 12  Class                  int64
```

A base de dados demonstrou alta qualidade estrutural, o que simplificou o pré-processamento:

- **Ausência de Valores Faltantes (Missing Values):** O comando `df.isna().sum()` retornou zero valores ausentes, eliminando a necessidade de imputação.
- **Ausência de Duplicatas e Inconsistências:** Não foram encontradas instâncias duplicadas (`df.duplicated().sum()` $= 0$) nem valores infinitos (`np.isinf()` nulo), garantindo a integridade numérica para a modelagem.

2) Análise de Valores Faltantes e Outliers: Como ja dito, dataset escolhido não possui dados faltantes. Para a identificação de valores atípicos (*outliers*) nas variáveis numéricas, utilizou-se o método da Amplitude Interquartílica (IQR - *Interquartile Range*). Este método define como *outliers* as observações que se encontram abaixo de $Q_1 - 1,5 \times \text{IQR}$ ou acima de $Q_3 + 1,5 \times \text{IQR}$, onde Q_1 e Q_3 representam o primeiro e terceiro quartis, respectivamente.

A implementação foi realizada através de um laço de repetição sobre o conjunto de variáveis numéricas.

Após a execução, os resultados indicaram a existência de outliers nas seguintes variáveis:

Outliers por Variável (Método IQR)

Variável	Contagem	Outliers (%)
BIDDER_TENDENCY	628	9.9%
BIDDING_RATIO	430	6.8%
SUCCESSIVE_OUTBIDDING	843	13.3%
CLASS	675	10.7%

Durante a análise univariada, plotamos também os *boxplots* que sustentam os resultados dessa função:

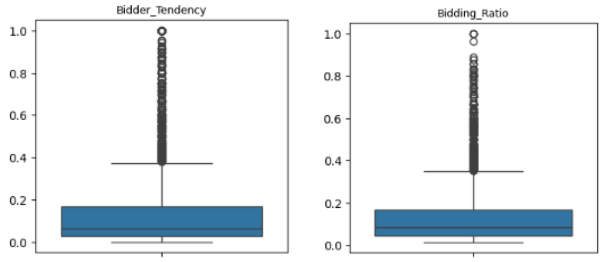


Fig. 1. Resultados dos Boxplots das variáveis: Bidder_tendency, e Bidding_Ratio

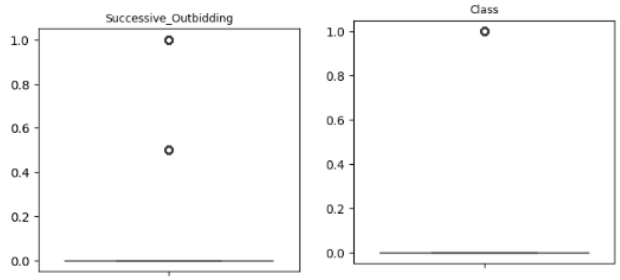


Fig. 2. Resultados dos Boxplots das variáveis: Successive_Outbidding e Class.

Uma vez que a nossa tarefa se concentrava em detecção de anomalias, supomos que esses *outliers* representavam os registros anômalos e não interfeririam no nosso treinamento; portanto, não aplicamos nenhum tratamento a esses dados.

3) Análise Univariada:

a) Tipos de Variáveis e Ranges: As variáveis foram separadas em numéricas (incluindo as features comportamentais, IDs e `Class`) e a única categórica, `Bidder_ID`. A maioria das features comportamentais, como `Bidder_Tendency` e `Bidding_Ratio`, já se encontram em *ranges* normalizados entre 0 e 1, o que favorece algoritmos baseados em distância.

Os intervalos de valores das principais features numéricas são resumidos a seguir:

TABLE I
INTERVALOS DE VALORES DAS FEATURES NUMÉRICAS

Variável	Mínimo	Máximo
Bidder_Tendency	0.0	1.0
Bidding_Ratio	≈ 0.0118	1.0
Successive_Outbidding	0.0	1.0
Winning_Ratio	0.0	1.0
Auction_Duration	1	10

b) Distribuições (Histogramas e Boxplots): A análise das distribuições por histogramas e *boxplots* revelou que a maior densidade está concentrada em valores próximos de 0 para várias features (`Successive_Outbidding`, `Winning_Ratio`, `Auction_Bids`, `Bidding_Ratio`).

Isto sugere um **comportamento conservador** dominante: a maioria dos lancistas realiza poucos lances, raramente vence leilões e não se engaja em alto *outbidding* sucessivo. Esse comportamento é coerente com o perfil da maioria dos participantes legítimos e reforça a ideia de que os desvios (os

outliers comportamentais) são candidatos à investigação de fraude.

c) Análise da Variável Alvo (`Class`): A variável alvo apresenta um **forte desbalanceamento**, crucial para a modelagem de detecção de anomalias:

TABLE II
DISTRIBUIÇÃO DA VARIÁVEL `Class`

Class	Contagem
0 (Normal)	5646
1 (Anomalia)	675

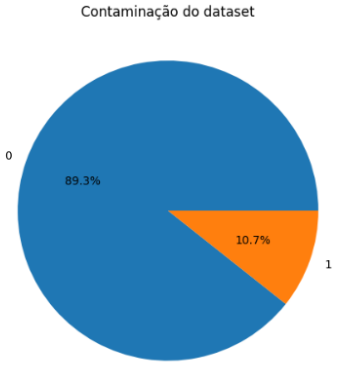


Fig. 3. Distribuição da variável Class.

A taxa de contaminação é de aproximadamente 10.7%.

d) Variável Categórica (`Bidder_ID`): Foram identificados 1.054 lancistas distintos em 6.321 instâncias. A análise de frequência (*counts*) mostrou que alguns poucos lancistas possuem um número de registros muito elevado (como o lancista "a*a"). Dentro do contexto de *shill bidding*, esta concentração de lances em poucos IDs é um forte indicador de comportamento suspeito.

4) Análise Bivariada: Dentro dessa secção fizemos a análise das correlações das variáveis numéricas por meio do cálculo da matriz de correlação e plotagem de um heatmap:

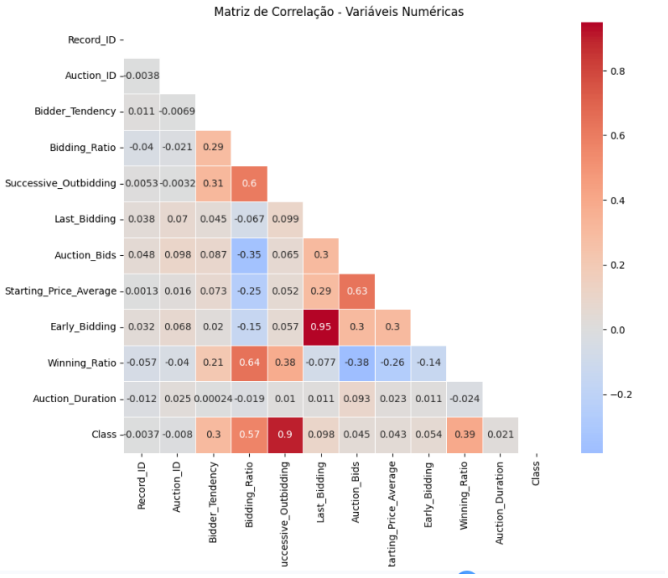


Fig. 4. Heatmap das variáveis numéricas

Insights extraídos:

- Observou-se uma correlação forte entre `Successive_Outbidding` e a variável alvo, `Class`, em torno de 0,90. Isso reforça a ideia de que auto-superar os próprios lances - `Successive_Outbidding`- é um dos sinais mais evidentes de comportamento suspeito (shill bidding). Em termos práticos, lancistas que frequentemente dão lances consecutivos sobre si mesmos tendem a estar associados à classe de lances fraudulentos.
- Correlações moderadas foram identificadas entre alguns pares de variáveis, tais como:
  - `Bidding_Ratio` e `Winning_Ratio`
  - `Auction_Bids` e `Starting_Price_Average`
  - `Bidding_Ratio` `Successive_Outbidding`
- Lancistas com `Bidding_Ratio` mais elevado também tendem a apresentar valores maiores de `Successive_Outbidding`, associando comportamento muito ativo com lances sucessivos.
- Notamos também que `Bidding_Ratio`, `Winning_Ratio` juntamente com `Successive_Outbidding` são bons indicadores para detecção de fraude, uma vez que se relacionam fortemente com `Class`.

5) Análise Multivariada: Para aprofundar a análise conjunta de variáveis e sua relação com a classe, foi construído um pairplot considerando as variáveis, `Successive_Outbidding`, `Bidding_Ratio` e `Class`

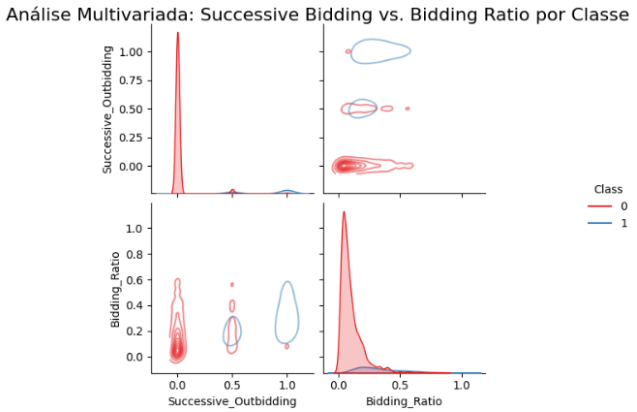


Fig. 5. Pairplot sobre as variáveis Successive_Outbidding, Bidding_Ratio e Class.

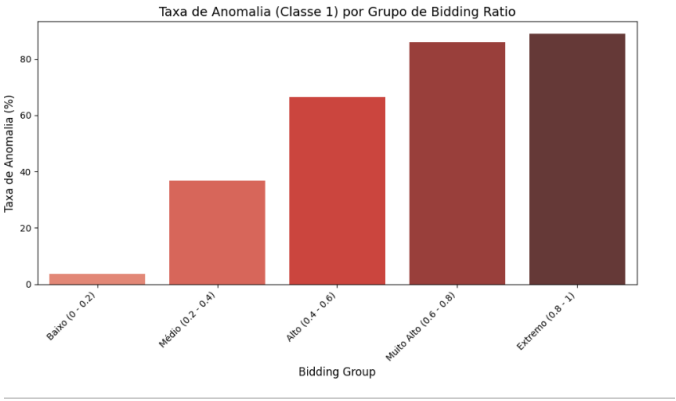


Fig. 6. Pairplot em cima de Successive_Outbidding, Bidding_Ratio e Class

Resultados extraídos:

- Gráfico de densidade conjunta
  - Classe 0 (Vermelho):Esta classe é a majoritária e está concentrada em uma região de baixíssimos valores para ambas as features
  - A Classe 1 (azul) apresenta densidades significativas em regiões de valores mais altos de `Bidding_Ratio` (próximo a 1.5 a 2.0) e `Successive_Outbidding` (entre 5 e 10).
- Análise Marginal Diagonal Superior Esquerda,este gráfico mostra a distribuição da variável `Successive_Outbidding`
  - A variável `Successive_Outbidding` sozinha já é um forte indicador, valores baixos sugerem a Classe 0, e valores mais altos sugerem a Classe 1.
- Análise Marginal Diagonal Inferior Direita Este gráfico mostra a distribuição da variável `Bidding_Ratio` sozinha, separada pelas classes.
  - A variável `Bidding_Ratio` também é bem discriminativa, com faixas de valores claramente diferentes para as duas classes.

Para explorar de forma mais interpretável a relação entre o ritmo de lances `Bidding_Ratio` e a ocorrência de fraudes, a variável foi discretizada em faixas, gerando a variável categórica `Bidding_Group`:

Com isso consegue-se perceber que grupos com `Bidding_Ratio` baixo apresentam menor taxa de fraudes.À medida que se aumenta a faixa de `Bidding_Ratio`, a taxa de anomalias cresce significativamente. Em outras palavras, os registros de lances fraudulentos tendem a se concentrar nas faixas de `Bidding_Ratio` mais altas, reforçando o papel dessa variável como importante indicador de comportamento suspeito.

B. Pré-processamento dos dados

1) Tratamento de Valores Faltantes, Duplicatas e Outliers: O *dataset* é limpo (sem faltantes/duplicatas), e os *outliers* foram mantidos por representarem as amostras anômalas a serem detectadas.

2) Feature Scaling: Utilizou-se o `MinMaxScaler` para transformar todos os atributos para o intervalo $[0, 1]$. Esta técnica é essencial, pois no Autoencoder, estabiliza os gradientes e facilita a reconstrução; já no K-Means, impede que variáveis de maior magnitude dominem o cálculo da distância Euclidiana , promovendo *clusters* mais coerentes.

3) Encoding de Variáveis Categóricas: A única variável categórica, `Bidder_ID` (identificador único), foi **removida**. Não foi aplicado *encoding* porque o ID do participante não contém informação generalizável e sua inclusão levaria o modelo a memorizar participantes específicos, aumentando o risco de *overfitting*.

C. Divisão dos Dados

a) Conjunto de Treino: Para a detecção de anomalias, utilizamos somente amostras benignas (`Class == 0`) no treinamento , permitindo que os algoritmos modelem exclusivamente o comportamento normal dos dados.

```
df_train = df.query('Class == 0').sample(
    frac=0.6, random_state=RANDOM_SEED)
```

b) Conjuntos de Validação e Teste: Esses conjuntos incluíram **amostras benignas e maliciosas**. A inclusão de dados maliciosos no conjunto de validação é essencial para a definição e otimização do *threshold* de detecção de anomalias.

D. Feature Engineering

As variáveis `Early_Bidding` e `Bidder_ID` foram removidas:

- `Early_Bidding` foi removida devido à alta correlação (0.95) com `Last_Bidding`, mantendo-se a última por ser mais informativa em relação à `Class`.
- `Bidder_ID` foi removida por ser um identificador único sem valor preditivo generalizável, prevenindo o risco de *overfitting*.

III. Modelagem

A. K-Means

1) Conceitos Básicos do K-Means: O algoritmo **K-Means** é um método de agrupamento (*clustering*) não supervisionado que divide o conjunto de dados em K grupos distintos. Na detecção de anomalias, a premissa é que os dados normais se agrupam em *clusters* coesos, e as anomalias são pontos que estão **distantes** de todos os centros desses *clusters*.

O processo de agrupamento segue as etapas iterativas até a convergência:

1) **Inicialização:** K centróides são escolhidos aleatoriamente no espaço de dados.
2) **Atribuição:** Cada amostra é atribuída ao centróide mais próximo, minimizando a distância euclidiana.
3) **Atualização:** O centróide de cada *cluster* é recalculado como a média de todas as amostras que lhe foram atribuídas.
4) **Convergência:** As etapas 2 e 3 se repetem até que os centróides se estabilizem ou o número máximo de iterações (max_iter) seja atingido.

a) Aplicação na Detecção de Anomalias: A detecção de anomalias com K-Means é baseada na **distância ao centróide mais próximo**:

- **Escore de Anomalia:** Para uma amostra x_i, o escore de anomalia é definido como a **distância euclidiana mínima** entre x_i e todos os K centróides (μ_j).
- **Classificação:** Pontos com **alta distância** são considerados anomalias, pois estão dispersos dos padrões normais de dados.
- **Threshold:** Um limiar (*threshold*) é definido sobre o escore de anomalia. Distâncias acima desse valor classificam a amostra como anômala (maliciosa).

2) Justificativa para o Uso do K-Means: O K-Means foi selecionado como um método eficaz e interpretável para a detecção de anomalias por:

- **Simplicidade e Eficiência Computacional:** É um algoritmo rápido e fácil de implementar. Sua complexidade de tempo, tipicamente $O(n \cdot K \cdot I \cdot d)$ (onde n é o número de amostras, K é o número de clusters, I é o número de iterações, e d é a dimensionalidade), o torna adequado para **grandes volumes de dados de rede**.
- **Modelagem Direta da Normalidade:** Modela explicitamente as regiões de dados "normais" (centróides). Ataques que desviam dessas densidades são facilmente capturados.
- **Interpretabilidade do Escore:** O escore de anomalia é a distância euclidiana ao centróide mais próximo, oferecendo uma métrica de desvio intuitiva.
- **Adaptabilidade a Múltiplos Padrões:** Permite modelar diversos padrões de tráfego benigno (através de K clusters), aumentando a sensibilidade para anomalias que ocorrem fora de qualquer um desses padrões.

3) Espaço de Busca de Hiperparâmetros: A otimização do modelo K-Means foi realizada no conjunto de validação, focando na maximização do **F1-Score**. O valor de K foi mantido em 6 (baseado na análise anterior), e o *threshold* de anomalia foi ajustado para cada modelo testado usando o **Índice de Youden**.

O espaço de busca cobriu os seguintes hiperparâmetros de execução:

TABLE III
Espaço de Busca Simplificado para o K-Means

Hiperparâmetro	Valores Explorados
n_init	$\{10, 25, 50, 75, 90, 100\}$
max_iter	$\{250, 500, 750, 1000, 1250, 1500\}$
algorithm	{'lloyd', 'elkan'}

Descrição dos Hiperparâmetros de Execução

- **n_init:** Número de vezes que o algoritmo K-Means será executado com diferentes sementes de centróides. O resultado final será o melhor desses runs.
- **max_iter:** Número máximo de iterações permitidas para o K-Means encontrar a convergência em uma única execução.
- **algorithm:** Tipo de algoritmo utilizado para otimizar o cálculo dos centróides.

4) Hiperparâmetros Selecionados: A busca em grade identificou o conjunto de parâmetros que maximizou o **F1-Score** no conjunto de validação. O *threshold* ideal para este modelo foi determinado utilizando o Índice de Youden.

- **F1-Score Vencedor (Validação): 0.8066**
- **Melhor Threshold (Validação): 0.7977**

Configuração do K-Means Vencedor

TABLE IV
Hiperparâmetros Selecionados para o K-Means Final

Hiperparâmetro	Valor Selecionado
K (Número de Clusters)	6 (Fixo)
n_init	90
max_iter	250
algorithm	lloyd

Interpretação da Configuração Selecionada: A seleção de n_init $= \mathbf{90}$ sugere que a inicialização dos centróides é crítica para a convergência ideal do K-Means, garantindo que o algoritmo encontre uma solução robusta. O max_iter $= \mathbf{250}$ foi suficiente, e o algoritmo lloyd foi o vencedor nesta configuração.

B. Autoencoder

1) Conceitos Básicos: O **Autoencoder (AE)** é uma arquitetura fundamental de rede neural artificial não supervisionada, concebida para aprender uma representação comprimida e eficiente (codificação) de um conjunto de dados. O AE é essencialmente um modelo de reconstrução que opera sobre o princípio de minimização da perda de informação.

A arquitetura consiste em duas partes principais e simétricas:

1) **Encoder:** Responsável por mapear os dados de entrada $\mathbf{x} \in R^D$ para uma representação de dimensão inferior $\mathbf{z} \in R^L$, conhecida como **espaço latente** ou *bottleneck*, onde tipicamente $L < D$. Este processo de codificação é dado por:
$$\mathbf{z} = f(\mathbf{x})$$
2) **Decoder:** Encarregado de reconstruir o dado de entrada, $\hat{\mathbf{x}}$, a partir da representação comprimida $\mathbf{z}$. O objetivo é que $\hat{\mathbf{x}}$ seja o mais próximo possível de $\mathbf{x}$. O processo de decodificação é dado por:
$$\hat{\mathbf{x}} = g(\mathbf{z})$$

O treinamento do AE é focado na minimização do **erro de reconstrução** (função de perda), que quantifica a diferença entre a entrada original ($\mathbf{x}$) e a saída reconstruída ($\hat{\mathbf{x}}$). Para as características de tráfego de rede (dados contínuos), o **Erro Quadrático Médio (MSE)** é a função de perda padrão utilizada:

$$\mathcal{L}(\mathbf{x}, \hat{\mathbf{x}}) = \frac{1}{D} \sum_{i=1}^{D} (x_i - \hat{x}_i)^2 \quad (1)$$

Onde D é a dimensão dos dados de entrada.

2) Justificativa para o Uso do Autoencoder (AE): O AE foi selecionado como algoritmo principal para o IDS com base em sua eficácia em modelar a normalidade:

1) **Modelagem da Normalidade:** Treinado *apenas* com tráfego benigno, o AE cria uma fronteira robusta que isola padrões de ataque minoritários ou zero-day.
2) **Métrica de Anomalia Baseada na Reconstrução:** O erro de reconstrução ($\mathcal{L}(\mathbf{x}, \hat{\mathbf{x}})$) serve como Anomaly Score. Ataques (padrões não vistos) resultam em erro significativamente mais alto.
3) **Redução de Dimensionalidade:** O *bottleneck* força o Encoder a aprender a representação mais concisa e relevante, atuando como um extrator de *features* não linear.

3) Espaço de Busca de Hiperparâmetros: A estratégia de otimização de hiperparâmetros consistiu na aplicação de um **Grid Search Exaustivo** sobre os parâmetros críticos que influenciam a capacidade de reconstrução e generalização do Autoencoder. O espaço de busca foi focado na minimização do Erro Quadrático Médio (MSE) de reconstrução no conjunto de validação (composto apenas por amostras benignas).

TABLE V
ESPAÇO DE BUSCA DE HIPERPARÂMETROS PARA O AUTOENCODER (ARQUITETURA 2)

Hiperparâmetro	Valores Testados
Dimensão Latente (Latent_dim)	$\{2, 3, 4\}$
Taxa de Dropout (Dropout_rate)	$\{0.0, 0.02, 0.05\}$
Taxa de Aprendizado (Learning_rate)	{1e-3, 5e-4}
Tamanho do Lote (Batch_size)	$\{64, 128\}$

O espaço de busca definido para o ajuste fino da Arquitetura 2 (a arquitetura vencedora no teste rápido) está detalhado na Tabela V:

A Tabela V lista os hiperparâmetros essenciais que foram testados. A **Dimensão Latente** controla a capacidade de compressão e representação do modelo; a **Taxa de Dropout** é uma técnica de regularização vital para combater o *overfitting*, e a **Taxa de Aprendizado** é fundamental para a convergência eficiente do otimizador Adam.

4) Hiperparâmetros Selecionados: O *Grid Search* identificou a combinação de hiperparâmetros que resultou no menor MSE de reconstrução no conjunto de validação. O treinamento final do **Autoencoder 2** foi realizado com esses parâmetros, utilizando o mecanismo de **Early Stopping** com Paciência e Delta, para garantir que o modelo fosse salvo no ponto de melhor generalização antes do *overfitting*.

Os hiperparâmetros finais selecionados e o regime de treinamento são:

- **Arquitetura Base:** Autoencoder 2 (Profunda com *BatchNorm*)
- **Dimensão Latente (Latent_dim):** 4
- **Taxa de Dropout (Dropout_rate):** 0.0
- **Taxa de Aprendizado (Learning_rate):** 1e-3
- **Tamanho do Lote (Batch_size):** 64
- **Mecanismo de Parada:** Early Stopping (Paciência: 10, Delta: 0.001)

C. Isolation Forest

1) Conceitos Básicos do Isolation Forest (iForest): O **Isolation Forest (iForest)** é um algoritmo de *Machine Learning* não supervisionado eficiente para detecção de anomalias (*outliers*).

A lógica central baseia-se no princípio de que anomalias são facilmente separáveis:

1) **Princípio de Isolamento:** Anomalias são pontos de dados **raros** e **distintos** (baixa densidade), sendo isoladas mais facilmente do que os dados normais.
2) **Construção da Árvore:** Uma coleção de Árvores de Isolamento é construída por divisões aleatórias de *features* e seus valores.
3) **Escore por Caminho:** O escore de anomalia é determinado pelo comprimento do caminho da raiz até a folha:
   - **Caminho Curto → Anomalia:** Anomalias exigem **poucas divisões** para serem isoladas.
   - **Caminho Longo → Normal:** Dados normais exigem **muitas divisões** devido à sua alta densidade.
4) **Nota de Implementação:** O escore de anomalia é a média do comprimento do caminho. Utilizamos, o sinal

é invertido (`-model.decision_function(X)`) para garantir que um escore **mais alto** indique maior probabilidade de anomalia (facilitando a análise da Curva ROC).

2) Justificativa para o Uso Isolation Forest: O Isolation Forest (iForest) foi selecionado como um modelo robusto para a detecção de ataques (anomalias) por razões estratégicas alinhadas à natureza dos dados de segurança e à meta de eficiência:

- **Foco Direto na Anomalia:** Foca em **isolar** os pontos raros (ataques), simplificando a fronteira de decisão, ao invés de modelar a complexa distribuição normal.
- **Eficiência e Escalabilidade:** É notavelmente rápido e ideal para grandes *datasets* de rede, pois sua complexidade de tempo é baixa (linear em amostras, $O(n)$, e logarítmica em *features*, $O(\log(\text{features}))$).
- **Robustez em Alta Dimensionalidade:** Mitiga o problema da "Maldição da Dimensionalidade" selecionando aleatoriamente subconjuntos de *features* em cada divisão, o que garante isolamento eficaz mesmo em dados com muitas colunas.
- **Velocidade de Treinamento/Inferência:** Demonstra desempenho similar a outros métodos de detecção de anomalias, sendo, no entanto, significativamente mais rápido na execução.

3) Espaço de Busca de Hiperparâmetros: Otimizamos o Isolation Forest utilizando um *Grid Search* no conjunto de validação para maximizar o **F1-Score**. O *threshold* ideal para cada combinação foi determinado pelo **Índice de Youden** ($J = TPR - FPR$) da curva ROC.

O espaço de busca cobriu os seguintes hiperparâmetros (totalizando $3 \times 3 \times 3 = 27$ combinações):

TABLE VI
ESPAÇO DE BUSCA SIMPLIFICADO PARA O ISOLATION FOREST

Hiperparâmetro	Valores Explorados
$n\_estimators$	$\{100, 200, 300\}$
$max\_samples$	$\{0.5, 0.75, 1.0\}$
$max\_features$	$\{0.5, 0.75, 1.0\}$
Contaminação	0.107 (Fixo)

Descrição Resumida dos Hiperparâmetros

- $n\_estimators$**:** Número de árvores (estabilidade).
- $max\_samples$**:** Proporção de amostras por árvore (eficácia no isolamento).
- $max\_features$**:** Proporção de *features* consideradas em cada divisão (robustez).
- **Contaminação:** Proporção esperada de anomalias (calibração interna do limiar).

4) Hiperparâmetros Selecionados: A otimização foi conduzida para identificar o modelo que maximizasse o **F1-Score** no conjunto de validação, após o ajuste do *threshold* pelo Índice de Youden. O modelo vencedor apresentou o seguinte conjunto de configurações:

- **Modelo Vencedor (Otimizado por F1-Score):** `0.7915` (F1-Score obtido no conjunto de validação)

Configuração do Isolation Forest Vencedor

TABLE VII
HIPERPARÂMETROS SELECIONADOS PARA O ISOLATION FOREST FINAL

Hiperparâmetro	Valor Selecionado
$n\_estimators$	200
$max\_samples$	0.5
$max\_features$	1.0
$contamination$	0.107 (Fixo)
Melhor Threshold (Validação)	**0.045671**

Interpretação da Configuração Selecionada:

- A seleção de $\mathbf{n\_estimators = 200}$ sugere que um número maior de árvores (200, em vez de 100) foi benéfico para estabilizar o processo de isolamento e reduzir a variância.
- O valor de $\mathbf{max\_samples = 0.5}$ (50% das amostras) indica que o subamostragem foi crucial. Usar apenas metade dos dados em cada árvore ajuda a garantir que as anomalias, que já são raras, sejam isoladas mais facilmente nos subconjuntos.
- O $\mathbf{max\_features = 1.0}$ (utilização de 100% das *features*) sugere que todas as *features* foram relevantes para o isolamento dos ataques, e restringir a dimensionalidade não melhorou o desempenho.

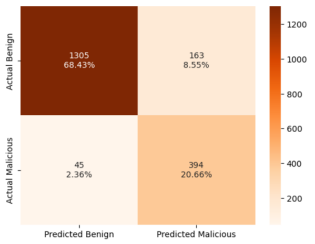
IV. ANÁLISE E COMPARAÇÃO DOS RESULTADOS

Nesta seção são apresentados e analisados os resultados obtidos pelos modelos *K-Means*, *Autoencoder* e *Isolation Forest* aplicados à detecção de anomalias. Inicialmente, cada modelo é analisado individualmente, considerando suas métricas e comportamento, e posteriormente é realizada uma comparação global entre eles.

A. Análise do Modelo K-Means

Seu desempenho depende fortemente da estrutura geométrica dos dados e da escolha adequada do número de clusters.

1) Resultados no Conjunto de Teste: A partir da matriz de confusão obtida, foram calculadas as seguintes métricas:



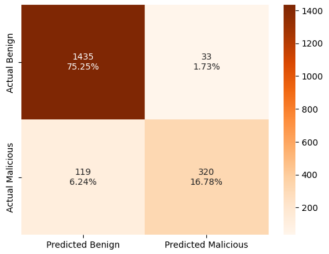
- Accuracy ≈ 0.890
- Recall (TPR) ≈ 0.897
- Precision ≈ 0.708
- F1-Score ≈ 0.791
- AUC ≈ 0.9633

2) Análise: O modelo apresentou alto *recall*, indicando boa capacidade de detectar instâncias maliciosas. No entanto, a *precision* relativamente baixa revela uma quantidade significativa de falsos positivos, o que pode ser problemático em aplicações sensíveis a alarmes incorretos. Apesar disso, a AUC elevada indica boa separabilidade global entre as classes.

B. Análise do Autoencoder

O Autoencoder é um modelo baseado em redes neurais que detecta anomalias a partir do erro de reconstrução. Instâncias anômalas tendem a apresentar maior erro, pois não seguem o padrão aprendido durante o treinamento.

1) Comportamento de Treinamento: As curvas de *loss* de treino e validação apresentaram convergência estável, sem evidências claras de *overfitting*, indicando aprendizado consistente da distribuição dos dados normais.



2) Resultados no Conjunto de Teste: As métricas obtidas foram:

- Accuracy ≈ 0.921
- Recall (TPR) ≈ 0.729
- Precision ≈ 0.907
- F1-Score ≈ 0.809
- AUC ≈ 0.9646

3) Análise: O Autoencoder apresentou alta *precision*, indicando que a maioria das anomalias detectadas é realmente maliciosa. Entretanto, o *recall* inferior demonstra que uma parcela relevante das anomalias não foi detectada. Esse comportamento é típico de modelos conservadores, que priorizam reduzir falsos positivos em detrimento da sensibilidade.

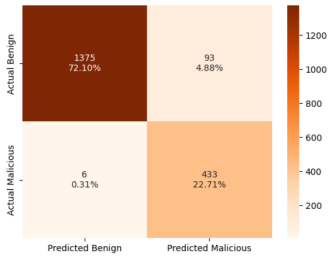
C. Análise do Isolation Forest

O Isolation Forest baseia-se no princípio de que anomalias são mais fáceis de isolar do que observações normais, utilizando partições aleatórias do espaço de atributos.

1) Hiperparâmetros Ótimos:

- `n_estimators`: 100
- `max_samples`: 0.5
- `max_features`: 0.5

2) Resultados no Conjunto de Teste: Após otimização pelo *F1-Score*, o modelo apresentou os seguintes resultados:



- Accuracy ≈ 0.948
- Recall (TPR) ≈ 0.986
- FPR ≈ 0.0633
- Precision ≈ 0.823
- F1-Score ≈ 0.897
- AUC ≈ 0.9905

3) Análise: O Isolation Forest apresentou desempenho superior em praticamente todas as métricas. O *recall* extremamente elevado indica excelente capacidade de detectar anomalias, enquanto a *precision* elevada demonstra bom controle de falsos positivos. A AUC próxima de 1 confirma a robustez do modelo independentemente do limiar de decisão.

D. Comparação Entre os Modelos

A Tabela VIII apresenta a comparação quantitativa entre os modelos avaliados.

TABLE VIII
COMPARAÇÃO DE DESEMPENHO ENTRE OS MODELOS

Modelo	Accuracy	TPR	Precision	F1-Score	AUC
K-Means	0.890	0.897	0.708	0.791	0.9633
Autoencoder	0.921	0.729	0.907	0.809	0.9646
Isolation Forest	0.948	0.986	0.823	0.897	0.9905

1) Discussão Comparativa: O K-Means destaca-se pelo alto *recall*, porém sofre com excesso de falsos positivos. O Autoencoder apresenta comportamento oposto, com alta *precision*, mas menor capacidade de detecção. O Isolation Forest oferece o melhor equilíbrio entre *precision* e *recall*, resultando no maior *F1-Score* e AUC, sendo o modelo mais robusto para o problema proposto.

V. CONCLUSÃO

Com base na análise individual e na comparação entre os modelos, o *Isolation Forest* mostrou-se o método mais adequado para a tarefa de detecção de anomalias, apresentando desempenho superior e consistente em todas as métricas relevantes, especialmente no *F1-Score*, critério principal de otimização adotado neste trabalho. Deverá conter as métricas que foram utilizadas para a análise juntamente com revisão de conceito e justificativas. É fundamental comparar os resultados obtidos entre os diferentes modelos treinados. Além disso, é interessante utilizar ferramentas estatísticas e/ou testes de hipótese quando cabível.

REFERÊNCIAS

[1] A. Alzahrani and S. Sadaoui, *Scraping and Preprocessing Commercial Auction Data for Fraud Classification*, Technical Report CS 2018-05. [Online]. Available: https://doi.org/10.6084/m9.figshare.6272342, 2018.