Detecção de ataques em fluxos de rede utilizando Naive Bayes

Carolina G. de A. B. dos Santos, Eduardo A. P. V. de Oliveira, Isabela M. de M. Nascimento,
Karina L. de Oliveira, Kauanny K. D'A. A. e Barros
Centro de Informática (CIn), Universidade Federal de Pernambuco (UFPE)
Recife, Brasil
{cgabs, eapvo, immn, klo, kkdab}@cin.ufpe.br

***Abstract*—Essa pesquisa utiliza os modelos de aprendizado de máquina para identificar padrões em fluxos de rede, classificando atividades como benignas ou como ataques cibernéticos populares por meio do modelo Naive Bayes. A análise exploratória avalia inconsistências e correlações entre variáveis, enquanto métricas como precisão, recall e acurácia medem o desempenho do modelo. Implementado em Python no Google Colaboratory, o projeto explora o potencial da Inteligência Artificial na cibersegurança.**

***Index Terms*—Cibersegurança, Naive Bayes, classificação, detecção de intrusões.**

I. Introdução

A crescente digitalização dos processos e a interconexão global através da internet têm trazido inúmeros benefícios, mas também expõem organizações e indivíduos a crescentes ameaças cibernéticas, podendo causar prejuízos financeiros e operacionais significativos. Nesse contexto, a detecção de ataques cibernéticos é essencial para proteger redes e sistemas. Esste projeto busca desenvolver um modelo baseado no algoritmo Naive Bayes para detecção de ataques em fluxos de rede, utilizando aprendizado de máquina para identificar padrões anômalos e diferenciar comportamentos legítimos de maliciosos, contribuindo para o fortalecimento da cibersegurança.

II. Objetivo(s)

O objetivo deste projeto é realizar a análise do dataset *CIC-IDS2017* [1], que contém informações sobre fluxos de rede, representando tanto dados benignos como ataques populares. Essa análise busca compreender as características dos dados, identificando padrões relevantes que auxiliem na detecção de intrusões. Além disso, será conduzida uma análise exploratória dos dados para:

- Identificar e tratar possíveis inconsistências ou valores ausentes no dataset;
- Analisar a distribuição das classes e o equilíbrio entre elas;
- Investigar a correlação entre as variáveis e sua relevância para tarefa de classificação.

Posteriormente, um modelo de aprendizado de máquina foi treinado com o objetivo de realizar a classificação dos dados de teste. A análise de desempenho foi realizada utilizando métricas como precisão, recall, F1-score e acurácia, permitindo uma avaliação detalhada da eficácia do modelo no contexto do problema.

III. Justificativa

A área de cibersegurança está ganhando cada vez mais espaço no mundo atual, impulsionada pelo aumento exponencial de dispositivos conectados e pela crescente sotisficação de ataques cibernéticos. Com organizações enfrentando desafios significativos para proteger suas infraestruturas, dados e operações, torna-se essencial explorar abordagens inovadoras para detecção e mitigação de ameaçsas.

Nesse contexto, a escolha da temática deste projeto justifica-se pela relevância de aplicar técnicas de *Inteligência Artificial* (IA) à cibersegurança, buscando soluções eficientes, rápidas e escaláveis. Modelos de aprendizado de máquina, como o *Naive Bayes* [2], tem se mostrado particularmente úteis devido à sua simplicidade, rapidez de treinamento e eficiência em tarefas de classificação, mesmo em cenários com grandes volumes de dados.

Ao analisar o comportamento do modelo *Naive Bayes* por meio de métricas como precisão, recall, F1-score e acurácia, este projeto visa demonstrar como a IA pode ser aplicada para identificar padrões em fluxos de rede e classificar atividades como benignas ou maliciosas. Essa abordagem possibilita não apenas um entendimento mais profundo do potencial desse modelo específico, mas também oferece uma avaliação prática de como soluções baseadas em IA podem ser aplicadas na cibersegurança.

IV. Base de dados

O *CIC-IDS2017* é um conjunto de dados amplamente utilizado em sistemas de detecção de intrusão (IDS) e sistemas de prevenção de intrusão (IPS), ele foi desenvolvido com o objetivo de fornecer dados realistas para a avaliação de abordagens de detecção de intrusão baseadas em anomalias. Este dataset possui o tráfego de rede beningno e ataques comuns, representando dados do mundo real, capturados por meio de pacotes de captura de rede (PCAPs), dentre as informações detalhadas de tráfegos encontram-se IPs de origem e destino, portas de origem e destino, protocolos e tipos de ataque. Além disso, os fluxos de rede são etiquetados com base em métricas como timestamps, o que facilita a análise e a classificação de eventos.

O dataset foi capturado entre os dias 3 e 7 de julho de 2017, com um foco particular em um dia típico de tráfego (segunda-feira) e a execução de diversos ataques nos dias subsequentes.

Durante esse período, ataques como Brute Force FTP, Brute Force SSH, DoS, Heartbleed, Web Attack, Infiltração, Botnet e DDoS foram realizados, permitindo a análise de uma variedade de cenários de segurança. Esses ataques foram realizados em diferentes horários, tanto pela manhã quanto à tarde, durante dias específicos, garantindo uma cobertura abrangente e representativa dos diversos tipos de ameaça cibernética.

Além disso, o *CIC-IDS2017* inclui uma topologia de rede completa, com diversos dispositivos e sistemas operacionais, abrangendo tanto as redes dos atacantes quanto das vítimas. A comunicação entre diferentes máquinas, incluindo servidores e clientes, foi capturada, representando uma rede variada com sistemas operacionais como Windows, Ubuntu e Mac OS. Ele possui dados de tráfego de protocolos comuns em redes modernas, como HTTP, HTTPS, FTP, SSH e e-mail. Isso oferece uma visão detalhada dos principais tipos de comunicação que ocorrem em ambientes de rede.

Completamente etiquetado, o dataset oferece metadados detalhados, como hora de ocorrência, tipo de ataque, fluxos e rótulos, permitindo que pesquisadores e engenheiros de segurança realizem análises aprofundadas. A estrutura do dataset é composta por arquivos no formato CSV, contendo mais de 80 features extraídas dos fluxos de rede, o que torna o conjunto de dados ideal para análise utilizando técnicas de aprendizado de máquina e aprendizado profundo.

A. *Processamento do Dataset*

O processamento do dataset é uma etapa essencial para garantir a qualidade dos dados utilizados no treinamento do modelo. Inicialmente, o dataset, que estava no formato CSV, foi carregado para o ambiente de trabalho. Após o carregamento, foi realizado um ajuste nos nomes das colunas, removendo espaços em branco. Esse procedimento padroniza a nomenclatura, facilitando as operações subsequentes e reduzindo possíveis erros durante o processamento.

Em seguida, foi realizada a limpeza dos dados para garantir que apenas informações relevantes e consistentes sejam utilizadas. Essa etapa inclui a remoção de registros duplicados, que poderiam introduzir redundância e vieses no modelo. Também serão tratados os valores ausentes (NaN, Null ou NA) e não finitos (como infinito ou -infinito), que podem comprometer a integridade das análises. Esses registros problemáticos foram excluídos para evitar impactos negativos no desempenho do modelo.

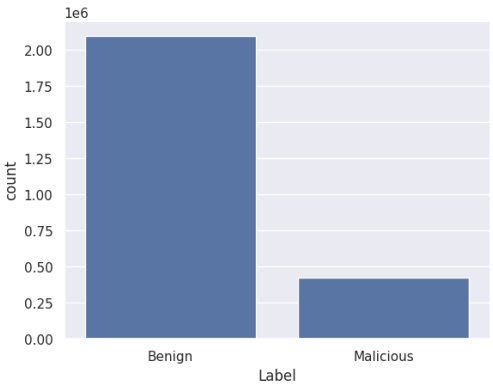
Após a limpeza, foi conduzida uma análise exploratória dos dados, com o objetivo de compreender a distribuição das variáveis, identificar padrões importantes e detectar possíveis outliers. Essa análise incluiu a criação de visualizações gráficas que facilitaram a interpretação dos dados. Posteriormente, o dataset foi dividido em dois subconjuntos distintos: dados de treino e de teste. Além disso, foi realizada uma análise de correlação entre as features para identificar variáveis que apresentam alta correlação entre si. Tais variáveis redundantes foram descartadas, uma vez que podem dificultar o aprendizado do modelo ao introduzir informações repetitivas.

Outra etapa crucial foi a normalização dos dados, ajustando todas as variáveis para uma escala comparável. Esse procedimento é especialmente importante para algoritmos como o Naive Bayes, que podem ser sensíveis à escala das features. A normalização garantirá que nenhuma variável tenha peso excessivo no treinamento do modelo. Por fim, o modelo de Naive Bayes foi treinado utilizando o conjunto de dados pré-processado. Essa etapa final integrou todo o trabalho realizado previamente, permitindo a construção de um modelo eficiente e preciso. O objetivo principal é obter o melhor desempenho possível, garantindo a confiabilidade das previsões realizadas pelo modelo.

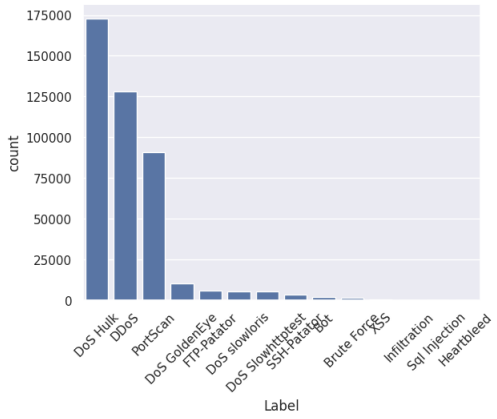
V. Análise Exploratória dos Dados

A análise exploratória de dados é uma etapa fundamental para compreender as características do conjunto de dados que está sendo analisado. Esse processo é essencial para garantir a quantidade e a confiabilidade das informações, além de possibilitar a descoberta de padrões e tendências entre as variáveis. O principal objetivo dessa análise é examinar detalhadamente a distribuição dos dados, frequentemente utilizando ferramentas como representações gráficas para facilitar a interpretação. Essa etapa envolve o estudo da quantidade de instâncias benignas e maliciosas, a distribuição dos ataques e a identificação de padrões nos dados.

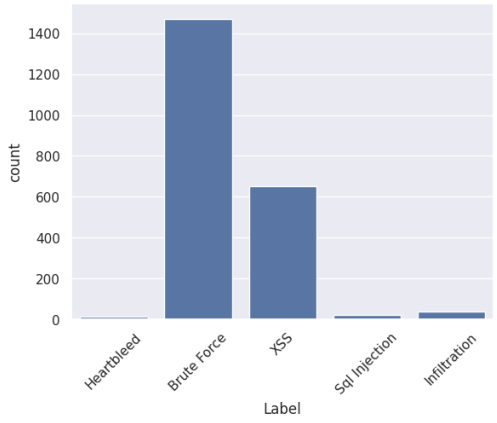
Ao fazer a verificação da quantidade de instâncias benignas x maliciosas, constatou-se que há um grande desbalanceamento entre as classes, com uma quantidade significativamente maior de tráfego benigno em comparação com os tráfegos maliciosos, conforme apresentado no gráfico abaixo. Esse desbalanceamento pode ser explicado pela maior prevalência de tráfego benigno na rede, já que a maioria das interações é legítima. Além disso, ataques maliciosos são menos frequentes, resultando em menos amostras disponíveis para análise.



Ao analisar a distribuição das instâncias por tipo de ataque, observa-se que os ataques *DoS Hulk* e *DDoS* se destacam, representando a maior parte do tráfego malicioso. Essa predominância ocorre devido à natureza desses ataques, que geram um alto volume de requisições em um curto intervalo de tempo, sobrecarregando os recursos do sistema alvo e dificultando sua defesa e mitigação.



Ao analisar os ataques menos representados no conjunto de dados, destacam-se *Heartbleed, SQL Injection* e *Infiltration*, que apresentam uma ocorrência significativamente menor em comparação com os demais. Essa distribuição sugere que certos tipos de ataques são menos frequentes no dataset analisado, o que pode impactar o desempenho do modelo de classificação adotado.



Além disso, ao executar o comando *dataset.describe()*, foi gerada uma análise estatística descritiva das colunas numéricas do conjunto de dados. O resultado apresenta diversas métricas relevantes, incluindo: a contagem de valores (count), que indica o número total de observações disponíveis para cada variável; a média (mean), que representa a tendência central dos dados; e o desvio padrão (std), que mede a dispersão dos valores em relação à média. Além disso, são fornecidos o valor mínimo (min), os quartis 25% (25%), 50% (50%, também correspondente à mediana) e 75% (75%), que permitem analisar a distribuição dos dados, bem como o valor máximo (max). Essas informações são essenciais para compreender a variabilidade e a estrutura dos dados, identificar possíveis outliers e auxiliar na tomada de decisões para etapas posteriores da análise.

	Destination Port	Flow Duration	Total Fwd Packets	Total Backward Packets	Total Length of Fwd Packets	Total Length of Bwd Packets	Fwd Packet Length Max	Fwd Packet Length Min	Fwd Packet Length Mean	Fwd Packet Length Std	...
count	2.522009e+06	2.522009e+06	2.522009e+06	2.522009e+06	2.522009e+06	2.522009e+06	2.522009e+06	2.522009e+06	2.522009e+06	2.522009e+06	...
mean	8.701432e+03	1.658364e+07	1.027750e+01	1.156751e+01	6.116607e+02	1.813569e+04	2.311241e+02	1.919733e+01	6.347899e+01	7.728840e+01	...
std	1.902225e+04	3.522618e+07	7.942294e+02	1.056668e+03	1.058573e+04	2.397602e+06	7.562104e+02	6.079830e+01	1.955137e+02	2.968147e+02	...
min	0.000000e+00	-1.300000e+01	1.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	...
25%	5.300000e+01	2.080000e+02	2.000000e+00	1.000000e+00	1.200000e+01	6.000000e+00	6.000000e+00	0.000000e+00	6.000000e+00	0.000000e+00	...
50%	8.000000e+01	5.058700e+04	2.000000e+00	2.000000e+00	6.600000e+01	1.560000e+02	4.000000e+01	2.000000e+00	3.613084e+01	0.000000e+00	...
75%	4.430000e+02	5.330376e+06	6.000000e+00	5.000000e+00	3.320000e+02	9.910000e+02	2.020000e+02	3.700000e+01	5.200000e+01	7.417179e+01	...
max	6.553500e+04	1.200000e+08	2.197590e+05	2.919220e+05	1.290000e+07	6.554530e+08	2.482000e+04	2.325000e+03	5.940857e+03	7.125597e+03	...

8 rows × 78 columns

Em conclusão, a análise exploratória dos dados forneceu uma visão detalhada sobre as características do conjunto analisado, destacando a predominância de ataques *DoS Hulk* e *DDoS*, que geram um volume significativo de tráfego malicioso. Observou-se também um desbalanceamento entre as instâncias benignas e maliciosas, com uma maior quantidade de tráfego benigno, o que reflete uma distribuição natural dos dados. Além disso, foram notados ataques como *Heartbleed, SQL Injection* e *Infiltration*, que, embora menos frequentes, adicionam diversidade ao conjunto de ameaças. A análise estatística descritiva das variáveis numéricas revelou informações importantes sobre a distribuição dos dados, como a tendência central e a dispersão, fornecendo uma base sólida para as etapas subsequentes.

VI. Classificador Ingênuo de Bayes

Nesta seção, detalhamos o funcionamento do Classificador Naive Bayes, destacando suas premissas fundamentais e suas principais aplicações. Além disso, apresentamos as duas variações do modelo utilizadas neste estudo: o *Gaussian Naive Bayes*, adequado para dados contínuos que seguem uma distribuição normal, e o *Bernoulli Naive Bayes*, projetado para variáveis binárias. Essas abordagens permitem flexibilidade no uso do Naive Bayes em diferentes cenários de aprendizado de máquina, tornando-o uma escolha versátil para problemas de classificação.

A. Teorema de Bayes

O Teorema de Bayes, um princípio fundamental na teoria das probabilidades e na estatística, é utilizado para calcular a probabilidade de um evento com base em informações prévias relacionadas a ele. Ele descreve como ajustar uma estimativa inicial ao considerar novas evidências ou dados observados, produzindo uma probabilidade revisada mais precisa.

Esse teorema leva o nome do matemático e pastor inglês Thomas Bayes (1701-1761), que desenvolveu ideias relacionadas à probabilidade condicional e sua aplicação em distribuições binomiais. Embora seus trabalhos só tenham ganhado destaque após sua morte, a contribuição de Bayes fundamenta muitos dos avanços na estatística moderna, especialmente no campo da inferência bayesiana. Uma das fórmulas mais conhecidas do Teorema de Bayes é:

$$P(A|B) = \frac{P(B|A) \times P(A)}{P(B)}$$

Onde $P(A \mid B)$ é a probabilidade de A ocorrer dado que B ocorreu, $P(B \mid A)$ é a probabilidade de B dado A, $P(A)$ é a probabilidade inicial de A, e $P(B)$ é a probabilidade de B.

O Teorema de Bayes tem aplicações amplas e práticas. Ele é essencial em áreas como aprendizado de máquina, diagnóstico médico, análise de risco e até mesmo em sistemas de recomendação. Um exemplo clássico é o uso em classificadores probabilísticos, como o Classificador Naive Bayes, que calcula a probabilidade de uma instância pertencer a uma categoria específica com base em seus atributos.

Além disso, o Teorema de Bayes é a base da abordagem bayesiana na estatística, que busca incorporar conhecimento prévio em análises, tornando-a uma ferramenta poderosa para problemas em que há incerteza ou dados limitados. Essa metodologia permite atualizações iterativas e dinâmicas das probabilidades à medida que novas informações surgem, ampliando sua utilidade em diversas disciplinas.

B. Classificador do Naive Bayes

O Classificador Naive Bayes é um dos algoritmos mais conhecidos e amplamente utilizados no aprendizado de máquina, tanto no meio acadêmico quanto no mercado. Ele apresenta uma abordagem direta e eficiente para resolver problemas de classificação, com base em princípios estatísticos sólidos.

O algoritmo é inspirado no trabalho do matemático Thomas Bayes e aplica o Teorema de Bayes para realizar previsões. A palavra ”naive” (ingênuo) refere-se à suposição simplificadora de que as características (ou atributos) dos dados são completamente independentes entre si. Embora essa premissa possa não ser verdadeira em muitos casos do mundo real, ela permite que o algoritmo seja computacionalmente eficiente e fácil de implementar.

Outro pressuposto importante do Naive Bayes é que todas as variáveis (features) têm igual relevância no resultado final. Quando essa condição não é atendida, o desempenho do algoritmo pode ser prejudicado, tornando-o menos adequado para situações onde as dependências entre atributos são significativas.

Apesar de suas limitações, o Naive Bayes é amplamente reconhecido por sua simplicidade, velocidade e eficácia, especialmente em problemas que envolvem grandes volumes de dados ou características de alta dimensionalidade. Ele é comumente utilizado em aplicações como filtragem de spam, análise de sentimentos, diagnóstico médico e classificação de textos. Para muitos iniciantes no campo do aprendizado de máquina, o Naive Bayes representa uma introdução acessível e poderosa. Sua formulação matemática clara e intuitiva oferece um ponto de partida ideal para entender conceitos como probabilidade condicional e inferência estatística.

Embora tenha sido projetado com base em hipóteses simplificadoras, o Naive Bayes se destaca por sua robustez em diversos cenários, sendo frequentemente uma escolha inicial para resolver problemas de classificação antes de explorar métodos mais complexos.

C. Variações Experimentadas

Para treinar o modelo Naive Bayes, utilizamos duas variações amplamente conhecidas e disponíveis na biblioteca *Scikit-learn*: o *Gaussian Naive Bayes* e o *Bernoulli Naive Bayes*. O *Gaussian Naive Bayes* é particularmente eficaz quando os dados seguem uma distribuição normal, sendo amplamente empregado em problemas que envolvem variáveis contínuas. Já o *Bernoulli Naive Bayes* é mais adequado para dados binários, onde as variáveis assumem apenas dois valores, como 0 e 1, sendo frequentemente utilizado em tarefas de classificação de texto e detecção de spam. Essas variações do Naive Bayes são fundamentais para diferentes cenários de aprendizado de máquina, permitindo abordagens versáteis e eficientes no processamento e na análise de dados. Além das variações mencionadas, o *Scikit-learn* disponibiliza outras implementações do Naive Bayes, como o *Multinomial Naive Bayes* e o *Complement Naive Bayes*. No entanto, optamos por não utilizá-las neste estudo, pois essas abordagens não aceitam dados negativos, o que inviabilizaria sua aplicação direta ao nosso conjunto de dados. Apesar disso, seria possível adaptar o dataset por meio de técnicas de pré-processamento, de modo a permitir o uso dessas variações.

D. Gaussian Naive Bayes

O modelo *Gaussian Naive Bayes (GNB)* é uma abordagem estatística amplamente utilizada para classificação de dados numéricos, assumindo que os atributos seguem uma distribuição normal dentro de cada classe. Baseado no teorema de Bayes, o *GNB* calcula a probabilidade condicional de uma classe dado um conjunto de atributos, considerando que cada atributo contribui de forma independente para a probabilidade final. O *GNB* apresenta vantagens significativas, como a eficiência computacional, a simplicidade na implementação e a capacidade de lidar com grandes volumes de dados. Além disso, mesmo quando a suposição de independência dos atributos não é completamente válida, o modelo pode fornecer resultados satisfatórios. No entanto, sua principal limitação reside na exigência de que os atributos sigam uma distribuição normal, o que nem sempre ocorre na prática, e na impossibilidade de capturar correlações entre os atributos.

E. Bernoulli Naive Bayes

O modelo *Bernoulli Naive Bayes (BNB)* é uma abordagem estatística baseada no teorema de Bayes, utilizada para classificação de dados binários. Diferente do *Gaussian Naive Bayes*, que assume uma distribuição normal para os atributos, o *BNB* considera que cada atributo segue uma distribuição de Bernoulli, onde os valores são representados por presença (1) ou ausência (0) de uma determinada característica. Durante o treinamento, o modelo estima a probabilidade de ocorrência de cada atributo para cada classe e, posteriormente, utiliza essas probabilidades para prever a classe mais provável de novas amostras. A suposição de independência entre os atributos facilita o cálculo das probabilidades e torna o modelo eficiente para grandes volumes de dados. Uma das principais vantagens do *BNB* é sua robustez em relação a dados esparsos, além disso, o modelo é rápido, simples de implementar e não exige grande volume de dados para treinamento. No entanto, uma das limitações do *BNB* é sua dependência da suposição de

independência dos atributos, o que pode reduzir sua precisão em cenários onde as variáveis são altamente correlacionadas.

VII. Experimentos

Inicialmente, foi realizado a análise exploratória dos dados do dataset *CIC-IDS2017* para compreensão de como as variáveis se comportam e vizualisar suas distribuições. Seguido dessa etapa, foram utilizados os modelos classificadores *Gaussian Naive Bayes* e *Bernoulli Naive Bayes* para classificar os dados como benignos ou como algum ataque de rede popular. Posteriormente, foi realizada a análise das métricas dos modelos para visualizar como eles se performam neste cenário.

Para isso, todo projeto foi desenvolvido na linguagem *Python*, em um ambiente de desenvolvimento do *Google Colaboratory*. As principais bibliotecas utilizadas foram: *Numpy, Pandas, Os, Re, Google, Matplotlib, Seaborn* e *Scikit learn*.

A. Ferramentas Utilizadas

A implementação dos classificadores *Gaussian Naive Bayes* e *Bernoulli Naive Bayes* foi conduzida utilizando a linguagem Python no ambiente Google Colaboratory, com o suporte de bibliotecas amplamente reconhecidas como Scikit-learn, Pandas e NumPy. A Scikit-learn facilitou a aplicação dos algoritmos Naive Bayes, enquanto Pandas e NumPy foram responsáveis pela manipulação e análise eficiente dos dados. Essa combinação de ferramentas permite a exploração completa dos classificadores no contexto de cibersegurança.

Os dados do projeto passaram por uma etapa essencial de análise e tratamento para garantir qualidade e relevância. Em seguida, foram divididos em dois subconjuntos: treinamento e teste. O conjunto de treinamento foi utilizado para ajustar os modelos e identificar padrões, enquanto o conjunto de teste serviu para avaliar suas eficiências na classificação de novos dados, verificando suas capacidades de manter um bom desempenho fora do conjunto de treinamento. Essa abordagem permitiu medir métricas de desempenho, como precisão, recall e acurácia, essenciais para validar a eficácia dos classificadores em detectar comportamentos maliciosos no tráfego de rede. Dessa forma, o projeto ofereceu uma base sólida para a identificação de ataques cibernéticos e para a compreensão das características mais relevantes na classificação, contribuindo significativamente para o fortalecimento da cibersegurança.

B. Métricas Analisadas

Para avaliar o desempenho dos modelos, foram coletadas diversas métricas relevantes para análise. As métricas utilizadas foram:

- Acurácia: Mede a proporção de previsões corretas em relação ao total de previsões.
- True Positive Rate (TPR): Representa a taxa de verdadeiros positivos em relação ao total de positivos reais.
- TPR por ataque: Avaliação do desempenho do modelo em identificar corretamente cada tipo de ataque presente no dataset.
- False Positive Rate (FPR): Mede a taxa de falsos positivos em relação ao total de negativos reais.
- Precisão: Mede a proporção de verdadeiros positivos entre todas as previsões positivas feitas pelo modelo.
- Recall: Mede a capacidade do modelo de identificar corretamente todas as ocorrências de uma determinada classe.
- F1-score: Média harmônica entre precisão e recall, útil para avaliar a qualidade do modelo em conjuntos de dados desbalanceados.
- Matriz de Confusão: Representação visual da performance do modelo, mostrando os verdadeiros e falsos positivos e negativos.
- Curva ROC e AUC-ROC: A curva ROC avalia a relação entre TPR e FPR, enquanto a AUC-ROC mede a área sob essa curva, indicando a capacidade discriminativa do modelo.

VIII. Análise dos resultados

Neste tópico, comparamos o desempenho de dois algoritmos de classificação aplicados ao dataset estudado: Gaussian Naive Bayes (GaussianNB) e Bernoulli Naive Bayes (BernoulliNB). Ambos são variantes do modelo Naive Bayes, que se destaca por ser simples, porém eficiente em problemas de classificação. No entanto, as diferenças formas de tratar os dados impactaram diretamente seus desempenhos ao realizar a detecção de anomalias em tráfego de rede.

A. Desempenho do GaussianNB

O GaussianNB é um algoritmo mais utilizado com variáveis contínuas que seguem uma distribuição gaussiana. Os seguintes resultados foram observados:

```
Classification Report:
              precision    recall  f1-score   support

           0       0.99      0.35      0.52    838454
           1       0.24      0.98      0.38    170350

    accuracy                           0.46   1008804
   macro avg       0.61      0.67      0.45   1008804
weighted avg       0.86      0.46      0.50   1008804
```

Ao identificar 98% das instâncias de ataque, isso prova que o GaussianNB demonstrou alta sensibilidade (recall) na detecção de ataques. Porém, teve baixa precisão (24%) em relação a essa classe, uma vez que gerou uma grande quantidade de falsos negativos, ou seja, classificou erroneamente fluxos benignos como maliciosos. Como o recall obtido foi 35% na classe de tráfego benigno, isso indica que o modelo possui dificuldades em reconhecer comunicações legítimas, o que impacta na confiabilidade do sistema.

B. Desempenho do BernoulliNB

O algoritmo BernoulliNB é mais adequado para variáveis binárias. Portanto, possui uma boa performance em cenários onde as características dos dados são representadas com valores booleanos.

```
Classification Report:
              precision    recall  f1-score   support

           0       0.92      0.85      0.89    838454
           1       0.47      0.64      0.54    170350

    accuracy                           0.82   1008804
   macro avg       0.70      0.75      0.71   1008804
weighted avg       0.84      0.82      0.83   1008804
```

A partir dos dados apresentados, nota-se que o BernoulliNB conseguiu um equilíbrio superior entre precisão e recall nas duas classes. Na classe benigno, foi eficaz em reconhecer tráfego normal, já que possui 85% de recall e 92% de precisão. Na classe ataque, obteve praticamente o dobro de precisão (47%) do GaussianNB e 64% de recall. A acurácia global de 82% indica que o BernoulliNB é significativamente mais eficaz do que o GaussianNB para o dataset estudado.

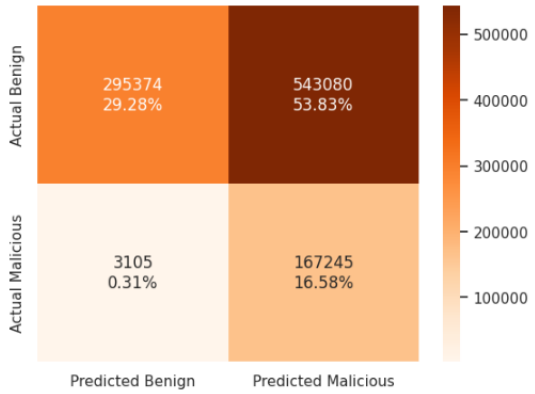
C. Comparação de performances

O BernoulliNB funciona melhor com dados binários ou booleanos, o que é coerente com os atributos do dataset CICIDS2017. Embora seja um modelo simples e eficiente, o BernoulliNB superou o GaussianNB ao não depender de suposições de distribuição contínua e normal, que não são válidas na maior parte dos dados de tráfego de rede. Além disso, o BernoulliNB apresentou menos falsos positivos em comparação com o GaussianNB, o que mostra que ele possui uma maior precisão na classe de ataque e garante uma maior confiabilidade para um sistema de detecção de intrusão. Portanto, o BernoulliNB é o modelo mais adequado para a detecção de anomalias no dataset CICIDS2017.

D. Matriz de confusão

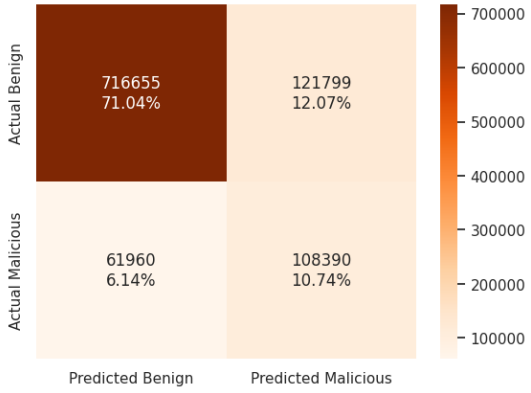
A matriz de confusão é uma tabela que indica os erros e acertos do seu modelo, comparando com o resultado esperado. Possui duas linhas e duas colunas que informam o número de verdadeiros positivos, falsos negativos, falsos positivos e verdadeiros negativos.

Para o GaussianNB, a distribuição de resultados se deu dessa maneira:



A matriz mostra que o GaussianNB gerou uma grande quantidade de falsos negativos (543.080 fluxos benignos que ele identificou como ataques).

Para o BernoulliNB, a distribuição de resultados ficou assim:



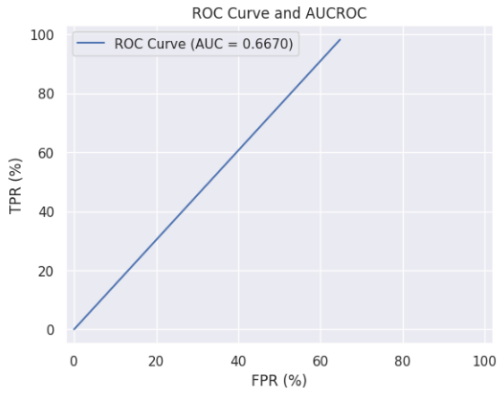
Nota-se que o BernoulliNB reduziu drasticamente os falsos negativos (121.799). Porém, como tem um recall menor, deixou passar mais ataques do que o GaussianNB (61.091 falsos positivos, ou seja, fluxos maliciosos que ele interpretou como benignos).

E. ROC Curve

A ROC Curve (Receiver Operating Characteristic) mostra o desempenho de um modelo classificatório, variando o limiar de decisão. Ele compara duas taxas importantes: a Taxa de Verdadeiros Positivos (mede quantos ataques reais o modelo detectou) e a Taxa de Falsos Positivos (mede quantos benignos ele classificou como ataque).

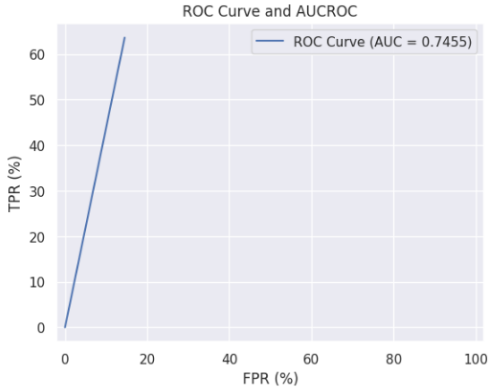
A AUC (Área sob a Curva) mede o quão boa a curva é, onde AUC = 1 indica um modelo perfeito e AUC = 0.5 indica o desempenho de um classificador aleatório.

Para o GaussianNB, a ROC Curve ficou assim:



Como ele possui uma curva que sobe rápido no eixo Y (Taxa de Verdadeiros Positivos), mas não consegue manter a Taxa de Falsos Positivos baixa, isso mostra que o modelo possui alta sensibilidade (detecta quase todos os ataques), mas gera muitos falsos positivos. Como AUC = 0.6670, ele se distancia de um modelo perfeito, e mostra que não consegue diferenciar tão bem um ataque e um benigno.

Para o BernoulliNB, a ROC Curve ficou assim:



Isso mostra que ele possui uma curva melhor balanceada, isto é, possui uma boa Taxa de Verdadeiros Positivos sem deixar a Taxa de Falsos Positivos crescer demais. Conclui-se, então, que o modelo consegue diferenciar melhor o que é ataque e o que é benigno. Nota-se que AUC = 0.7455, que é maior que a do GaussianNB. Portanto, possui uma melhor capacidade discriminativa.

IX. Conclusões e discussões

Nesse projeto, foi realizada uma análise de anomalias em fluxos de redes utilizando o dataset CICIDS2017, que contém registros apresentando tanto tráfegos benignos quanto diversos tipos de ataques cibernéticos. Inicialmente, o dataset foi tratado com técnicas de pré-processamento, visando melhorar o desempenho dos algoritmos.

O estudo consistiu em aplicar e comparar os modelos probabilísticos de classificação GaussianNB e BernoulliNB. Eles foram avaliados com base em métricas de desempenho, como matriz de confusão, curva ROC (Receiver Operating Characteristic) e AUC (Área Sob a Curva).

Através da análise exploratória, concluiu-se que o GaussianNB apresenta alta sensibilidade, detectando a maior parte dos ataques, mas com um alto índice de falsos negativos, o que reflete uma baixa precisão. Já o BernoulliNB demonstrou um melhor equilíbrio entre precisão e recall, além de possuir uma taxa de falsos negativos significativamente menor.

Apesar dos bons resultados obtidos, há possibilidades de melhorias futuras para esse estudo, como um tratamento mais aprofundado dos dados, bem como testar outros algoritmos de machine learning, para que seja possível extrair melhores resultados.

References

[1] Dataset CIC-IDS2017, disponível em https://www.unb.ca/cic/datasets/ids-2017.html

[2] Classificador Naive Bayes, disponível em https://scikit-learn.org/stable/modules/naive_bayes.html