Trabalho de Conclusão de Curso
2ª Etapa – Projeto do Sistema

ANOMAD IOT
Modelos para Detecção de Anomalias no Tráfego de Rede de Dispositivos IoT: Um Estudo Exploratório

Integrantes do Grupo:

Caio Franco de Souza
Gabriel Moiseis de Lima
José Guides Mequelin

Orientador(a): Prof.º Me. Guilherme Werneck de Oliveira

Co-orientador(a):Prof.º Gabriel Vinicius Canzi Candido

Pinhais
2024

1. Introdução

ADADAWDWADWADWASAWADDDDDDSDSSDSWAWrwCom o decorrer dos anos e com o avanço da tecnologia, principalmente nas áreas de eletrônica, sistemas de sensoriamento e troca de informações expandiram horizontes e expectativas acerca de quais sistemas e dispositivos do mundo físico poderiam ser integrados à internet. Esses sistemas e dispositivos são conhecidos como IoT (sigla em inglês para Internet das Coisas). Estima-se que em 2025, haverá mais de 27 bilhões de dispositivos IoT conectados em todo o mundo (PACETE, 2022).

Ainda que a IoT tenha o papel de facilitar o cotidiano dos usuários, seja como meio de comunicação em empresas, seja para fins domésticos, existem diversos problemas ainda não solucionados satisfatoriamente, como a privacidade e a falta de segurança dos dispositivos em relação aos dados dos usuários. São comuns notícias de tentativas de ataques contra dispositivos IoT no mundo todo. Nos dois primeiros meses de 2023, por exemplo, quase todas as semanas, em média 54% das organizações foram alvo dessas tentativas de ataque, com uma média de quase 60 ataques por organização por semana direcionados a dispositivos de IoT. Uma alta de 41% em comparação com 2022 e mais que o triplo do número de ataques se comparados aos de dois anos atrás (ABRANET, 2023).

Ao decorrer disso, muitas pesquisas em relação a segurança de dispositivos IoT foram realizadas, por empresas, pesquisadores da área de TI e técnicos de cibersegurança. Com isso, foram desenvolvidos milhares de artigos e documentos relacionados à proteção de dados em dispositivos IoT. Trabalhos desenvolvidos com intuito de explicar e ajudar pessoas a compreender e entender, qual e melhor método para proteger seus dados.

Também foram criados protocolos de segurança para redes sem fio de IoT, que foram elaborados com intuito de defender as redes IoT contra ameaças e ataques às informações pessoais dos usuários. Esses são alguns dos protocolos, TLS (*Transport Layer Security*), DTLS (*Datagram Transport Layer Security*), CoAP (*Constrained Application Protocol*) e MQTT (*Message Queuing Telemetry Transport*). Apesar disso ainda se tem inúmeros ataques, pessoas que conseguem explorar as vulnerabilidades desses sistemas.

Assim, este trabalho tem como objetivo desenvolver um estudo de modelos utilizando algoritmos de aprendizagem de máquinas, para detectar anomalias no tráfego de rede do usuário. Essa abordagem será implementada para dispositivos em ambientes domésticos, no intuito de auxiliar a segurança das IoT e a informações e dados pessoais.

2. Lista de requisitos funcionais

REF 01. O método deve detectar anomalias em tempo real.

REF 02. O método deve identificar padrões normais.

REF 03. O método deve notificar quando uma anomalia for detectada.

REF 04. O método deve ter algoritmo de aprendizagem de máquina.

REF 05. O método deve se conectar com os dispositivos domésticos.

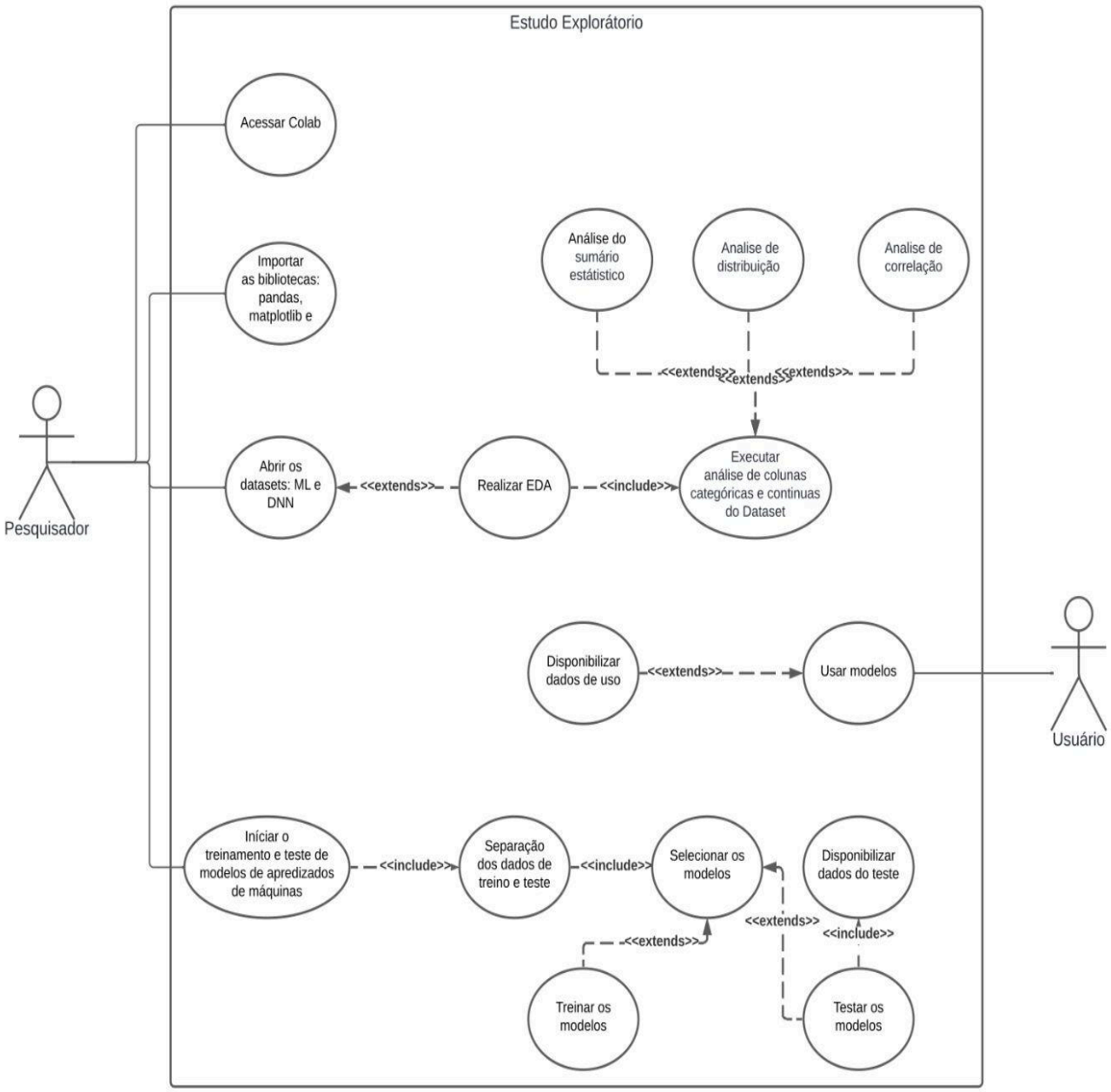
3. Lista de requisitos não funcionais

RNF 01. O método deve ter disponibilidade para identificar e responder rapidamente as falhas ou interrupções no serviço.

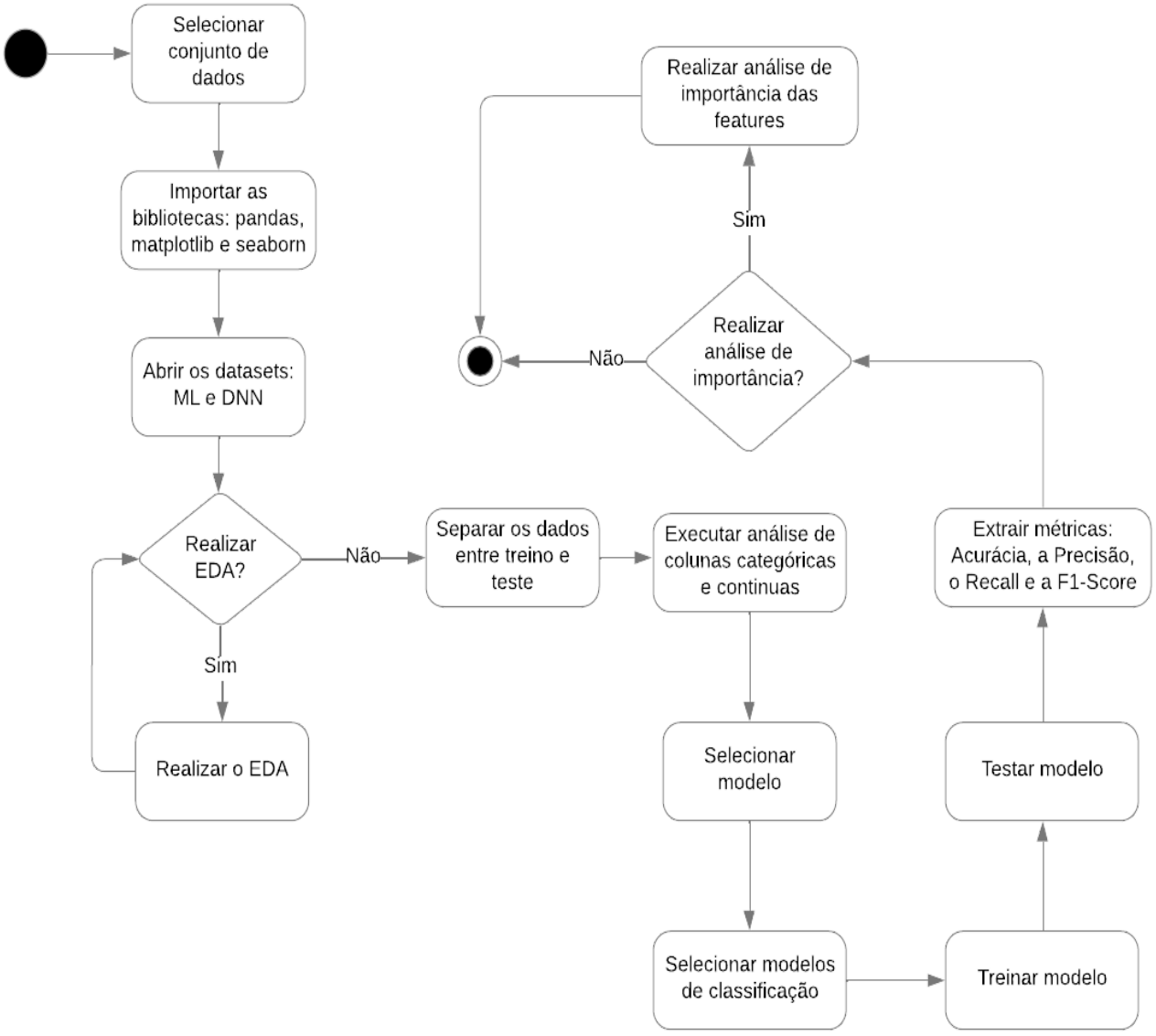
RNF 02. O método deve ser capaz de lidar com um grande número de dispositivos IoT(*internet of things*) e volumes significativos de tráfego de dados sem comprometer o desempenho

RNF. 03 O método deve ser capaz de mandar uma mensagem com instruções auxiliando o administrador em caso de detecção de anomalias.

4. Diagrama de casos de uso



5. Diagrama de atividades



6. Codificação (código-fonte)

importação das bibliotecas

```
[ ]  import pandas as pd
     import numpy as np
     import matplotlib.pyplot as plt
     import seaborn as sns
     from sklearn.model_selection import train_test_split, KFold
     from sklearn import tree
     from sklearn.tree import plot_tree
     from sklearn.inspection import permutation_importance
     from sklearn.ensemble import RandomForestClassifier
     from sklearn.neighbors import KNeighborsClassifier
     from sklearn.naive_bayes import MultinomialNB, GaussianNB, BernoulliNB
     from sklearn.svm import LinearSVC
     from sklearn.svm import SVC
     from sklearn.model_selection import cross_val_score
     from sklearn.metrics import accuracy_score, recall_score, f1_score
     from sklearn.feature_selection import SelectKBest, chi2
     from sklearn.preprocessing import StandardScaler
     from sklearn.metrics import precision_score
     from sklearn.metrics import confusion_matrix
```

conectar e abrir o dataset

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
[ ]  df = pd.read_csv("/content/drive/MyDrive/Edge-IIoTset dataset/Selected dataset for ML and DL/ML-EdgeIIoT-dataset.csv")
     pd.set_option("display.max_columns", None)
     #pd.set_option("display.max_rows", 10)
     df.head()
```

divisão de dados em treino e teste

```
train = df.drop(["frame.time", "ip.src_host", "ip.dst_host", "arp.dst.proto_ipv4", "arp.src.proto_ipv4", "icmp.transmit_timestamp",
                 "http.file_data", "http.request.uri.query", "http.request.method", "http.referer", "http.request.full_uri", "http.request.version",
                 "tcp.options", "tcp.payload", "tcp.srcport", "tcp.seq",
                 "mqtt.protoname", "mqtt.topic", "mqtt.conack.flags", "mqtt.msg",
                 "dns.qry.name.len",
                 "Attack_label", "Attack_type"], axis = 1)
target = df["Attack_label"]

x_train, x_test, y_train, y_test = train_test_split(train, target, test_size = 0.3, random_state = 10)
```

treino e teste com dados normalizados

```
scaler = StandardScaler()
x_train_norm = scaler.fit_transform(x_train)
x_test_norm = scaler.transform(x_test)
```

métricas de avaliação do modelo de aprendizagem

```
dt = tree.DecisionTreeClassifier()
dt.fit(x_train, y_train)

dt_pred = dt.predict(x_test)
dt_pred
```

```
array([1, 0, 1, ..., 1, 0, 1])
```

```
[7] dt_accuracy = accuracy_score(dt_pred, y_test)
    dt_recall = recall_score(dt_pred, y_test)
    dt_f1 = f1_score(dt_pred, y_test)
    dt_precisao = precision_score(dt_pred, y_test)


    print(f'Acurácia DT: {dt_accuracy * 100}')
    print(f'Precisão DT: {dt_precisao * 100}')
    print(f'Recall DT: {dt_recall * 100}')
    print(f'F1 DT: {dt_f1 * 100}')
```

```
Acurácia DT: 99.11491339247993
Precisão DT: 99.49274094804967
Recall DT: 99.46043165467626
F1 DT: 99.47658367790534
```

matriz de confusão para verificar o desempenho do modelo

```
#Matriz de confusão DT
cf_matrix = confusion_matrix(y_test, dt_pred)
print(cf_matrix)
# Criando o heatmap
sns.heatmap(cf_matrix, cmap='coolwarm', annot=True, linewidth=1, fmt='d')
plt.show()
```

importância das features para o modelo obter seu resultado

```
feat_import = pd.DataFrame(dt.feature_importances_, index = train.columns, columns = ["Importância"])
feat_import.sort_values(by = "Importância", ascending = False, inplace = True)
feat_import.plot(kind = "bar", figsize = (8,6))
```

ilustrações das matrizes de confusão

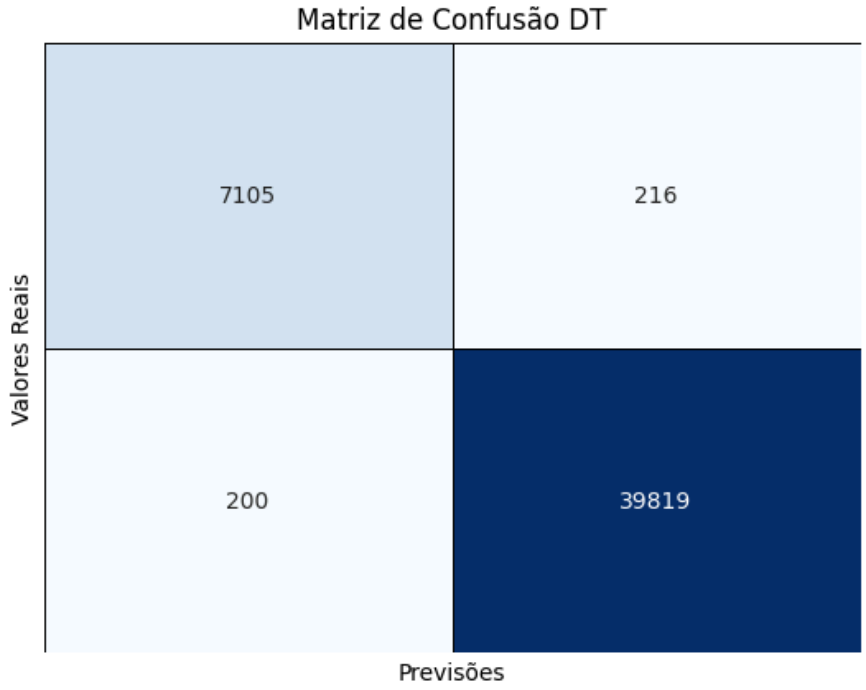
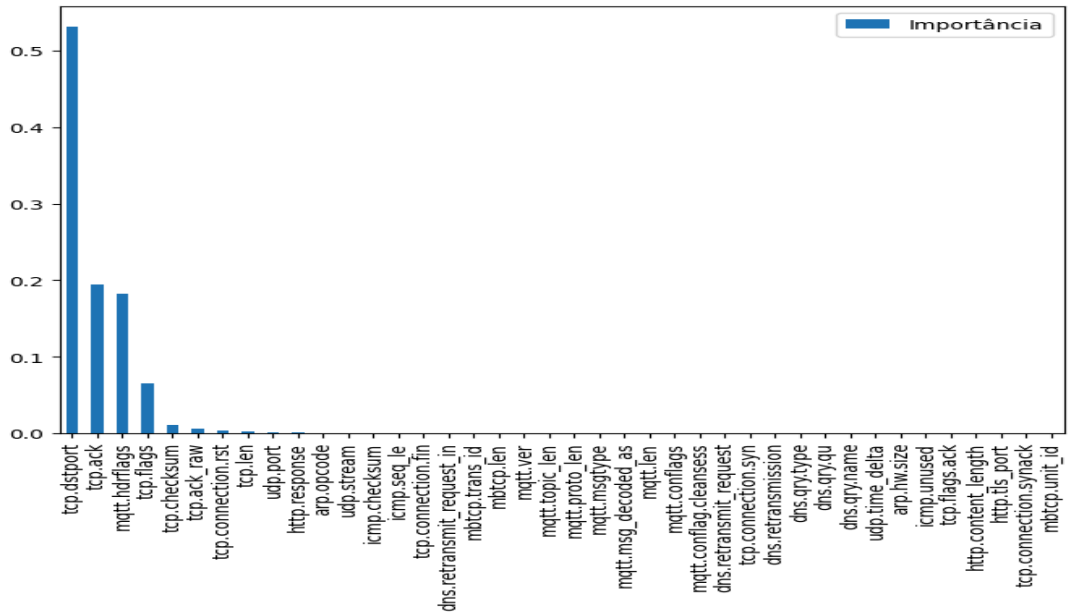


ilustração do gráfico da importância das features

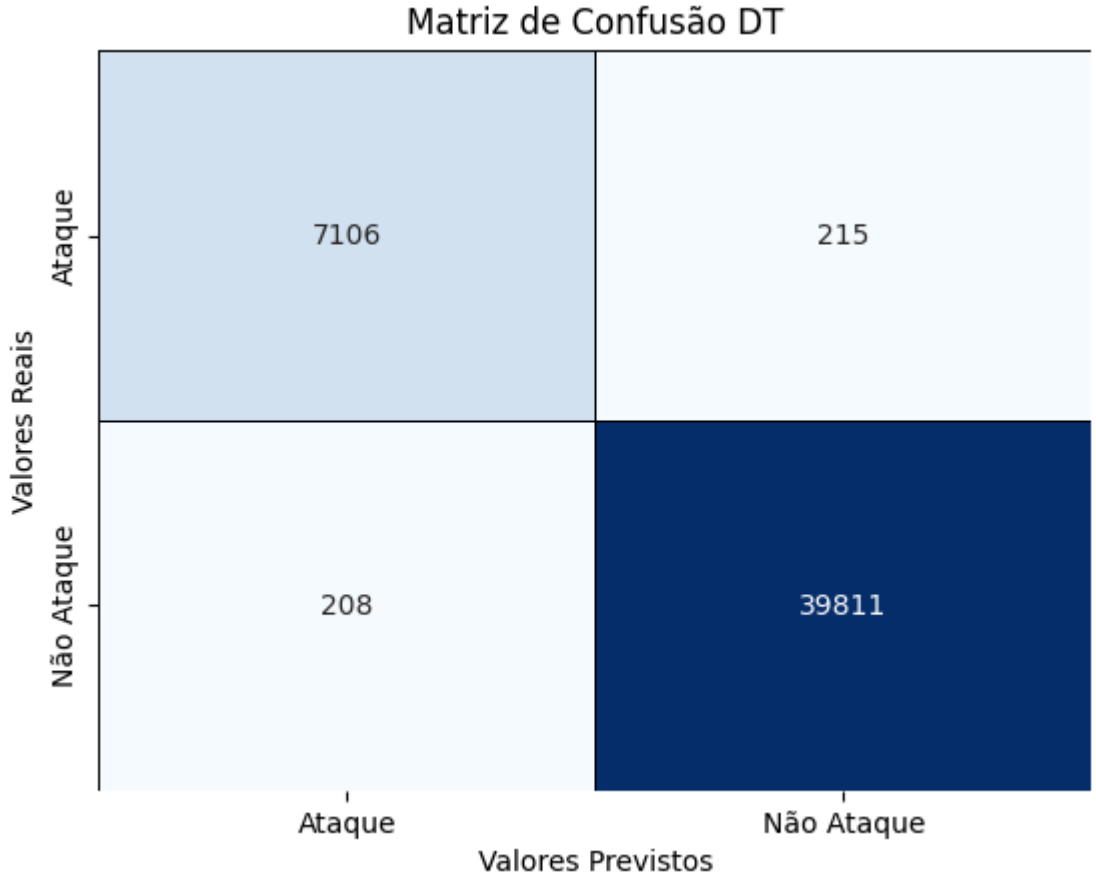


Utilização da programação orientada a objetos como encapsulamento, herança, abstração e polimorfismo.

Incluir o plano de testes de unidade que serão realizados para garantir a qualidade do código e evitar retrabalho. Apresentar o tipo de padrão de projeto adotado (como por exemplo, o *Design Patterns*).

7. Testes

Para aplicação dos testes de aplicação dos modelos, que geram quantidade de acertos e erros contabilizados em sua matriz de confusão



onde o modelo de detecção aplicado (DT, RF, SVM, KNN, NB) verifica o dataframe e a coluna do dataset ML, especificamente a coluna "*Attack_label*", onde armazena dados binários, em que o modelo classifica 1 como ataque e 0 como tráfego normal.

como visto na imagem acima, os quadrados azul claro(VP) e azul forte(VN), são os valores em que o modelo classificou corretamente, que são eles os verdadeiro possitivos(realmente são ataques) e verdadeiro negativos(realmente são tráfegos normais)e os valores em branco são os erros de classificação do modelo. Também foram aplicadas as Métricas de avaliação como: Acurácia, Precisão, *Recall* e *F1-score.*

10. Referências

[Meneghello et al. 2019] Meneghello, F., Calore, M., Zucchetto, D., Polese, M., and Za-nella, A. (2019). Iot: Internet of threats? a survey of practical security vulnerabilities in real iot devices. IEEE Internet of Things Journal, 6(5):8182–8201.

[PACETE 2022] PACETE, L. (2022). Iot: até 2025, mais de 27 bilhões de dispositivos estarão conectados.

[Abranet 2023] Abranet (2023). Ataques a dispositivos da internet das coisas (iot) crescem 41%. https://www.abranet.org.br/Noticias/Ataques-a-dispositivos-da-internet-das-coisas-%28IoT%29-crescem-41%25-4300.html.

[Xenofontos et al. 2021] Xenofontos, C., Zografopoulos, I., Konstantinou, C., Jolfaei, A., Khan, M. K., and Choo, K.-K. R. (2021). Consumer, commercial, and industrial iot (in) security: Attack taxonomy and case studies. IEEE Internet of Things Journal, 9(1):199–221.

[Google-colab 2024] Google-colab (2024). Google colab. Disponível em: https://colab.google/.

[Butun et al. 2019] Butun, I., ¨Osterberg, P., and Song, H. (2019). Security of the internet of things: Vulnerabilities, attacks, and countermeasures. IEEE Communications Surveys & Tutorials, 22(1):616–644.

[IEEE 2024] IEEE (2024). Ieee dataport. Disponível em: https://ieee-dataport.org/.

[Scikit-Learn 2024] Scikit-Learn (2024). Scikit-learn. Dispon´ıvel em: https://scikit-learn.org/.