CASE - MONITORING ANALYST POSITION

Rafael Kujo Monteiro

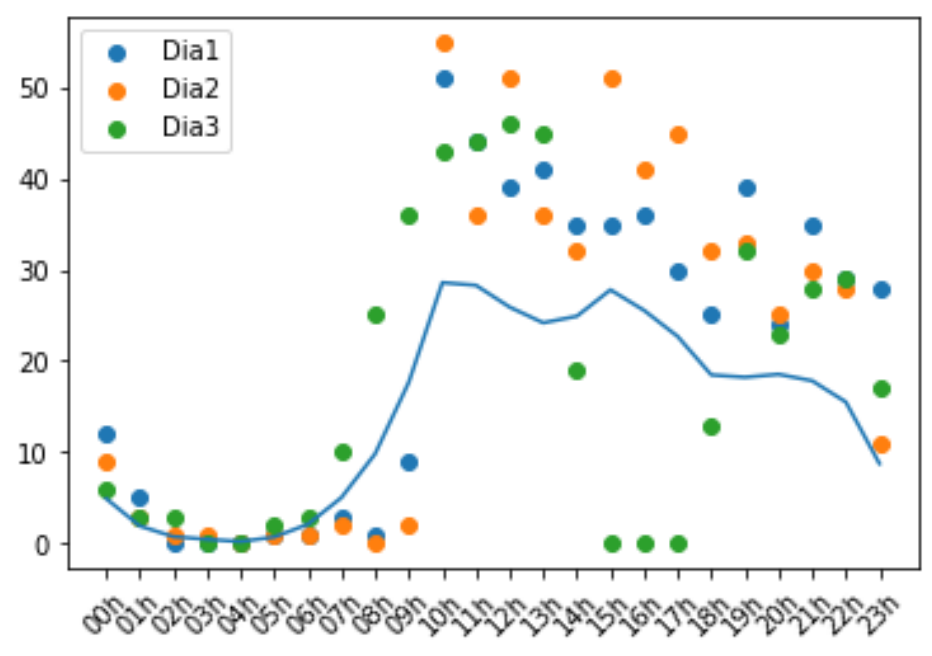
TAREFA 1

Identificação de Anomalias – Checkout Data

Resumo

- 2 arquivos .csv com dados sobre de operações de checkout;
- Informações detalhadas por horário, de hora em hora;
- Objetivo de encontrar anomalias para os 3 dias disponíveis;

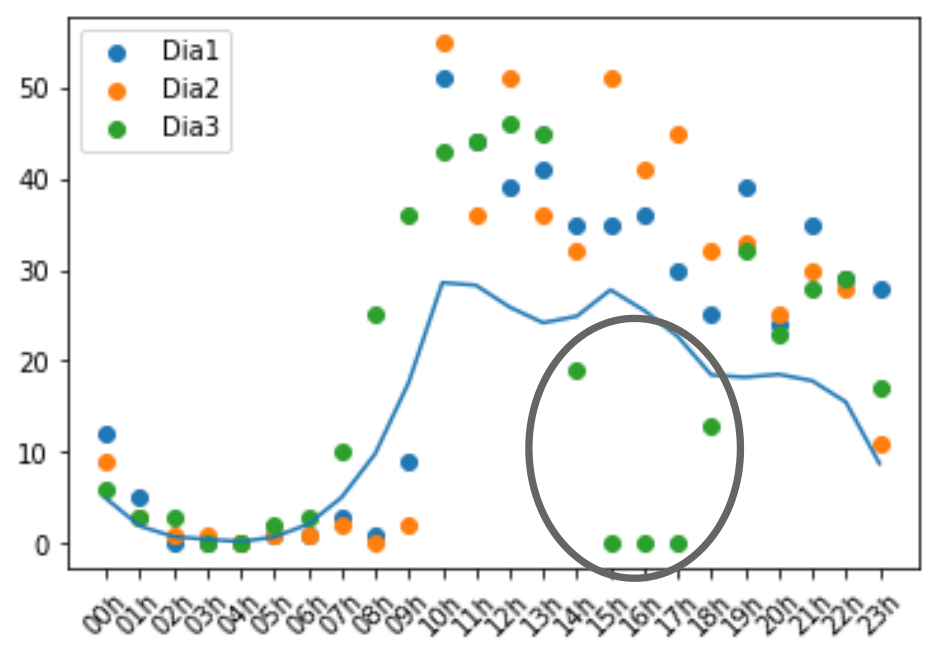
Visualização Inicial



- Anomalia1: Dia 3, das 14h às 18h.
- Anomalia2: Dia1 e Dia2, das 8h às 10h.
- Anomalia3: Dia2, às 11h e 17h.

Necessário confirmar as suspeitas com mais evidências.

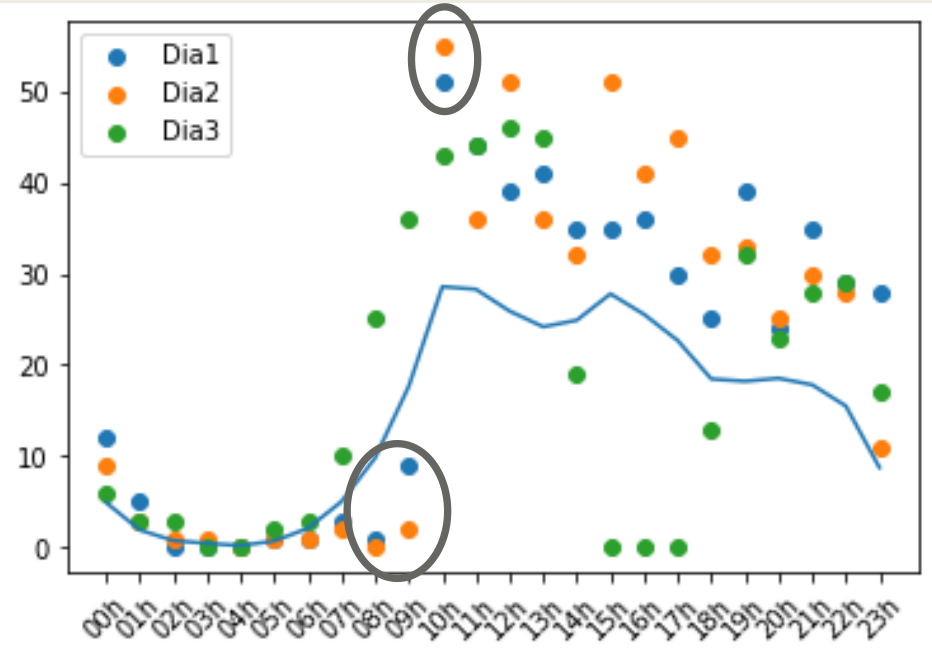
Anomalia 1 (D3 14h-18h)



- Movimento mais claramente fora do padrão na amostra.
- Valores zerados próximos de um horário de pico no mês.

Anomalia clara, com algum bloqueio nas operações começando entre 14h e 15h e depois retomando entre 17h e 18h.

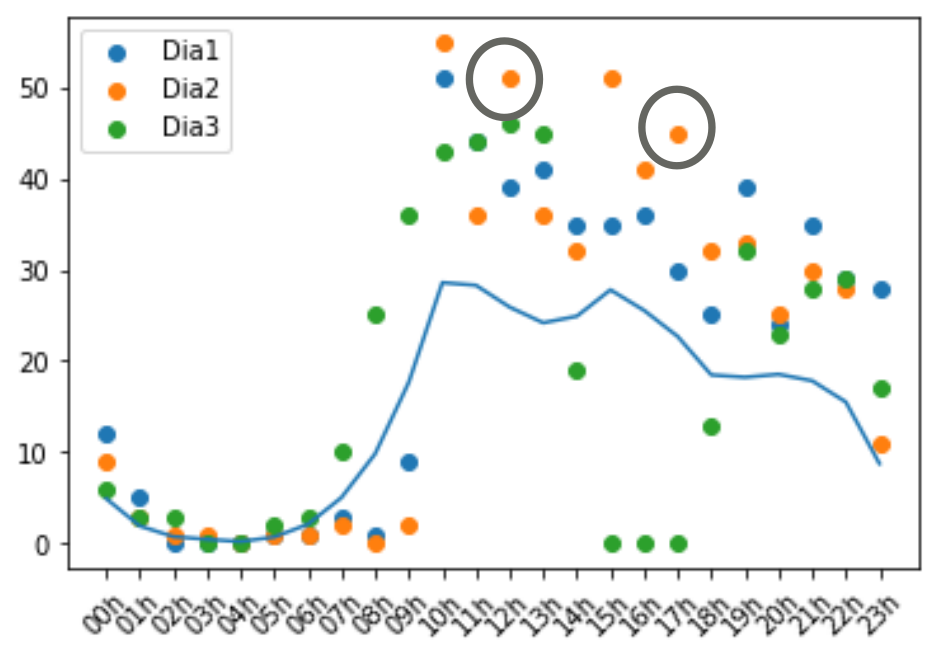
Anomalia 2 (D1&D2 8h-10h)



- Movimento mais sutil, pode não ser anomalia e sim movimento dentro do padrão diário.
- Operações em estabilidade/queda quando esperava subida vertiginosa. Na sequência, explosão para valores mais altos da série.

Indicação de possível represamento entre 8h e 9h, com liberação às 10h.

Anomalia 3 (D2 12h / 17h)



- Movimentos de subida, em momentos em que a média mensal mostrava queda;
- Além disso, muito acima da média histórica nos dois momentos.

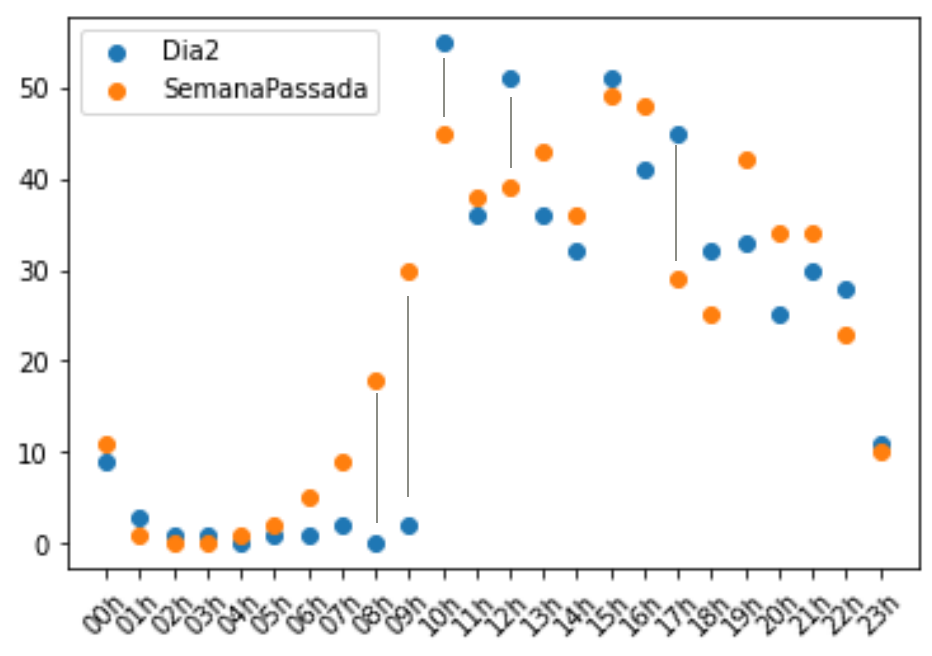
Mais difícil confirmar se é anomalia sem amostra maior, mas é um sinal.

SQL - Query

```
SELECT

dia,

time,

quant_ops,

ABS(quant_ops - CAST(avg_last_month AS
FLOAT)) AS dif_avg_month


FROM dbo.Checkouts_dias1a3


ORDER BY dif_avg_month DESC


OFFSET 0 ROWS

FETCH NEXT 10 ROWS ONLY
```

	dia	time	quant_ops	dif_avg_month
1	3	15h	0	27,78
2	2	10h	55	26,65
3	2	12h	51	25,58
4	3	16h	0	25,53
5	2	15h	51	23,29
6	2	17h	45	22,72
7	3	17h	0	22,67
8	1	10h	51	22,65
9	3	13h	45	20,83
10	1	19h	39	20,33

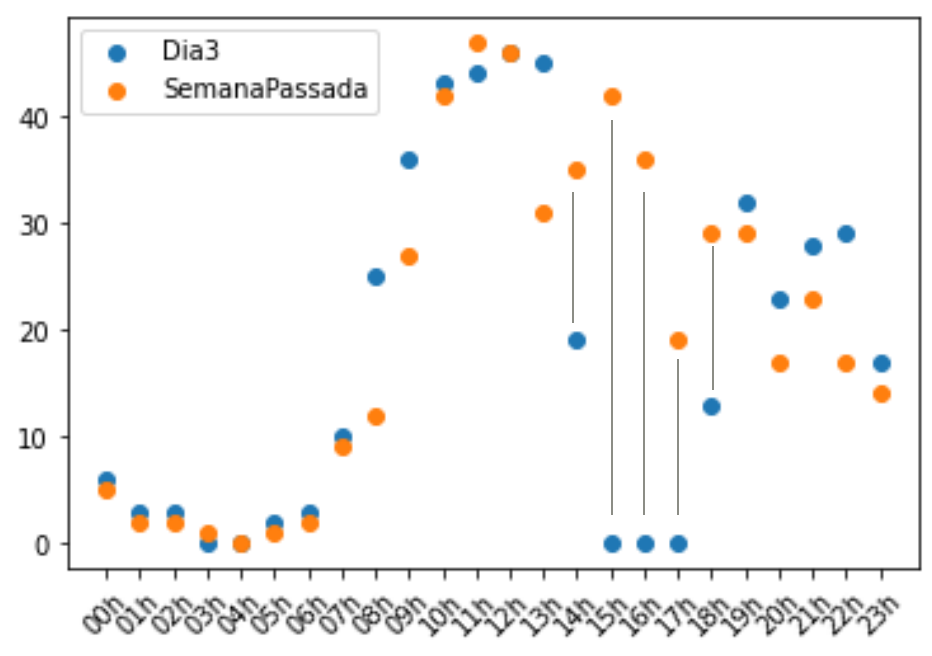
Aprofundando a análise:



- Anomalia2 (08h-10h) reforçada;
- Anomalia3 (12h/17h) reforçada.

Nos horários analisados, os valores estão relativamente distantes do observado na semana anterior, não descartando serem fora do comum.

Aprofundando a análise:



- Anomalia1 (15-17h) praticamente confirmada.

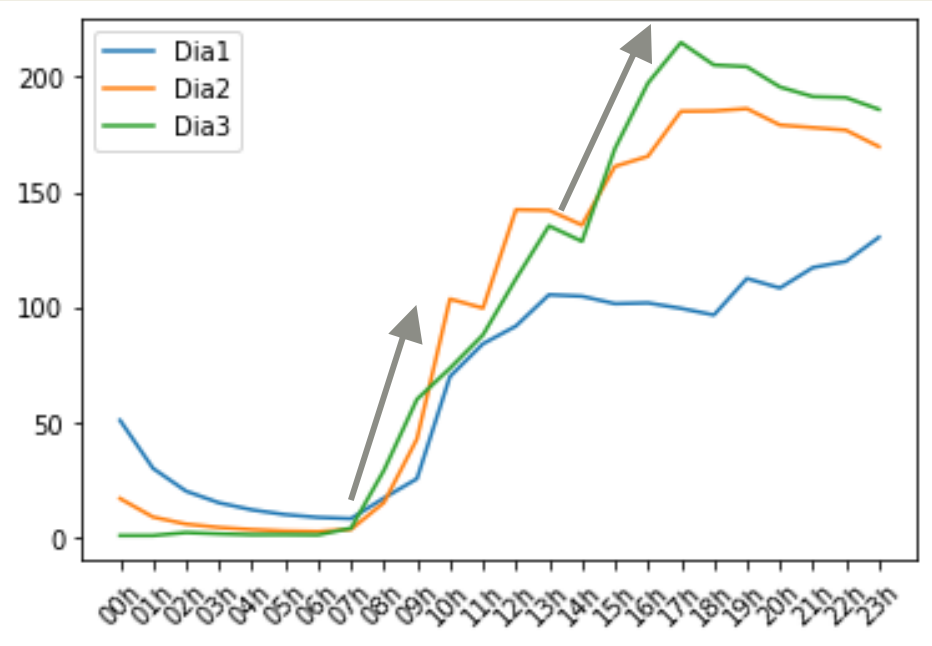
Nos horários analisados, os valores estão bem distantes do observado na semana anterior, reforçando a ideia de serem fora do comum.

SQL - Query

```
SELECT
dia,
time,
quant_ops,
ABS(quant_ops - same_day_last_week) as dif_week

FROM dbo.Checkouts_dias1a3

WHERE dia <> 1

ORDER BY dif_week DESC

OFFSET 0 ROWS
FETCH NEXT 10 ROWS ONLY
```

	dia	time	quant_ops	dif_week
1	3	15h	0	42
2	3	16h	0	36
3	2	09h	2	28
4	3	17h	0	19
5	2	08h	0	18
6	2	17h	45	16
7	3	14h	19	16
8	3	18h	13	16
9	3	13h	45	14
10	3	08h	25	13

Confirmação final – Desvio da Média



- Desvio: Erro ao quadrado em relação à média mensal (EQM);
- Gráfico evidencia que dias 2 e 3 estão mais distantes da média;
- Horários destacados compatíveis com anomalias observadas anteriormente.

SQL - Query

```
SELECT
dia,
time,
CAST (desvio_c/10 AS FLOAT) AS desvio_c

FROM dbo.Checkouts_CW

WHERE time = '23h'

ORDER BY desvio_c DESC
```

	dia	time	desvio_c
1	3	23h	185,9
2	2	23h	169,7
3	1	23h	130,4

SQL - Query

```
SELECT

dia,

time,

quant_ops,

ABS(quant_ops - CAST(avg_last_month AS FLOAT)) *
ABS(quant_ops - CAST(same_day_last_week AS FLOAT))
as mult


FROM dbo.Checkouts_dias1a3


ORDER BY mult DESC


OFFSET 0 ROWS

FETCH NEXT 10 ROWS ONLY
```

	dia	time	quant_ops	mult
1	3	15h	0	1166,76
2	3	16h	0	919,08
3	2	09h	2	477,96
4	3	17h	0	430,73
5	2	17h	45	363,52
6	1	23h	28	346,5
7	2	12h	51	306,96
8	3	13h	45	291,62
9	2	10h	55	266,5
10	1	09h	9	211,47

Conclusões

- Anomalia1 e Anomalia2 confirmadas;
- Anomalia3 suspeita – necessitaria mais informações.

TAREFA 2

Criação Monitoring System – Transaction Data

Resumo

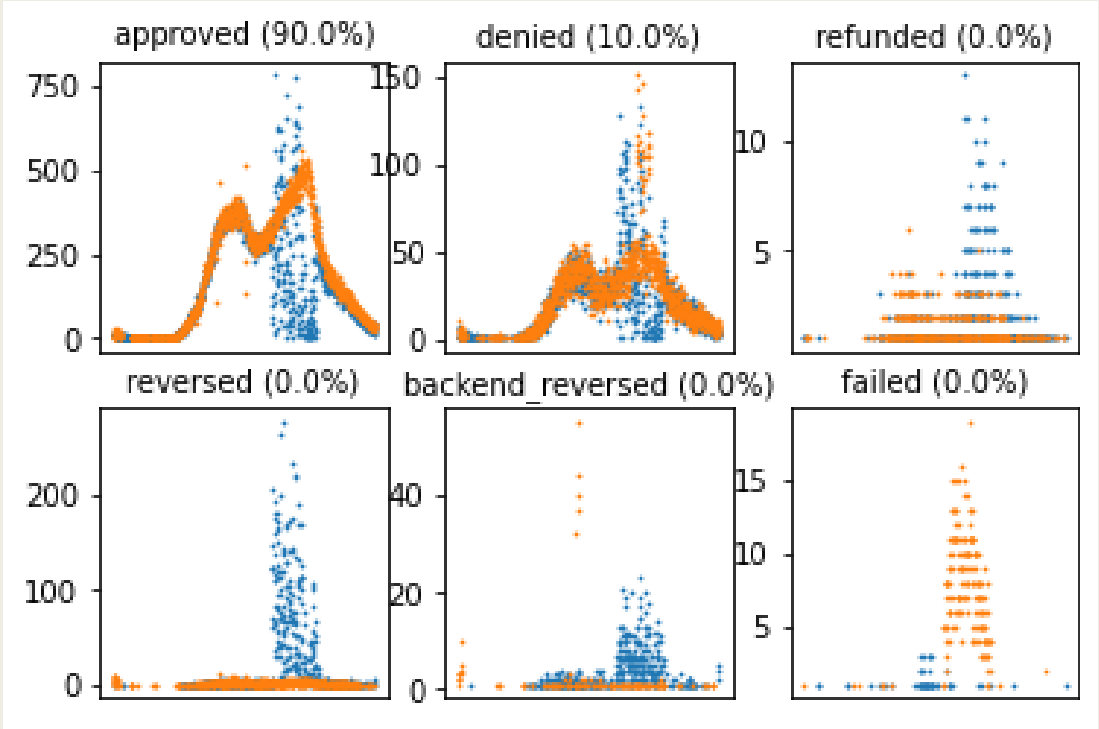
- 2 arquivos .csv com dados de transações;
- Informações detalhadas por horário, de minuto em minuto;
- Dados agregados por status das operações a cada minuto;
- Objetivo: criar um sistema de monitoramento para este tipo de transações.

Requisitos

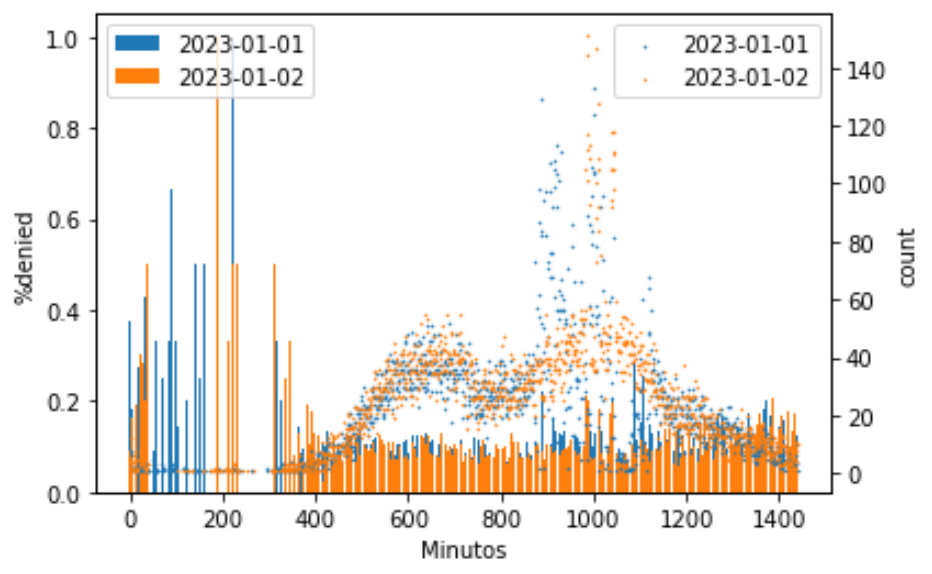
- End-point que recebe os dados e retorna recomendação de alerta ou não;
- Query para organizar os dados existentes;
- Gráfico para visualização em tempo real;
- Modelo que alerte automaticamente as anomalias.

Análise Exploratória

- - Comportamento muito parecido entre ‘approved’ e ‘denied’;
- Diferença grande, para todos os status, de um dia para o outro – dificulta encontrar padrões;
- ‘failed’ e ‘reversed’ com poucas ocorrências em um dia e pico em outro.



Denied Operations



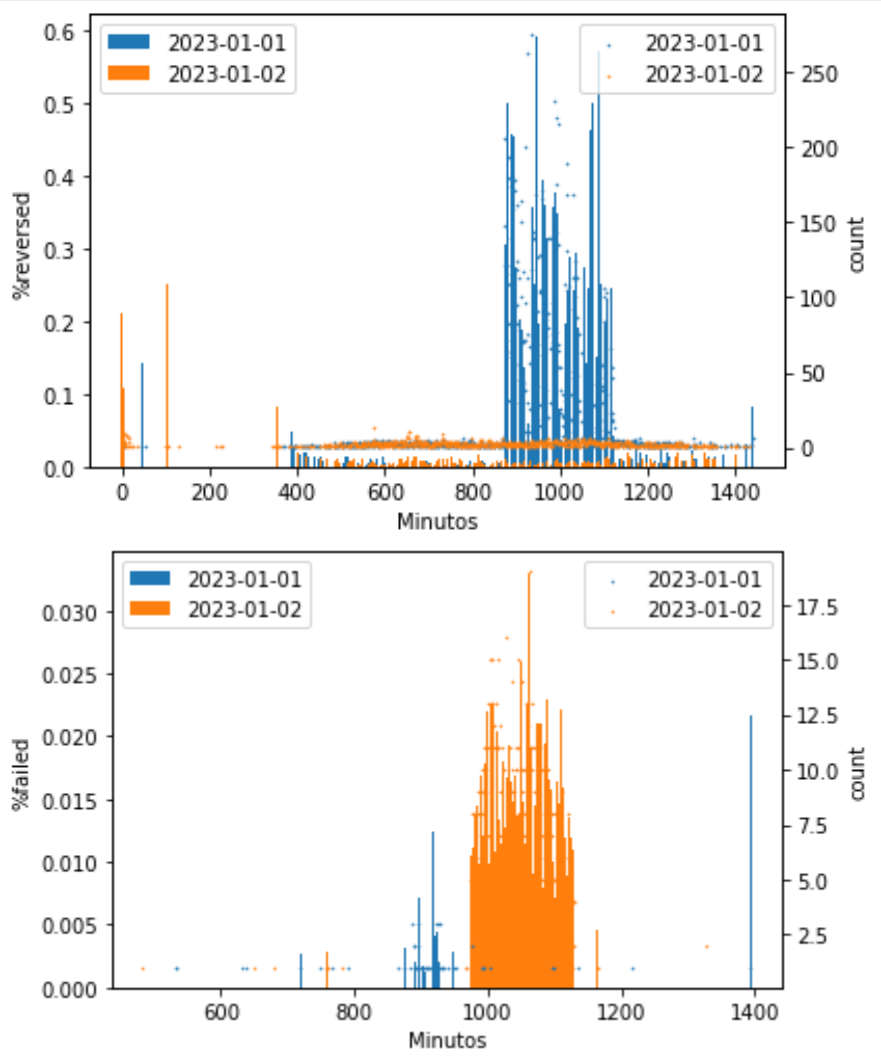
- Valores absolutos e relativos não são diretamente proporcionais entre si – relação com ‘approved’ e total;

```
df_denied['%denied'].corr(df_denied['count'])
-0.17937890457053512
```

- Dependência do horário: pré 06h e pós 06h (~360min);

- Critérios prévios de análise de risco aumenta % de madrugada.

Reversed e Failed Operations



- Valores absolutos e relativos possuem forte proporcionalidade – 1 regra para os dois pode ser suficiente;

```
df_reversed['%reversed'].corr(df_reversed['count'])
0.6793800662701042

df_failed['%failed'].corr(df_failed['count'])
0.9443500641939518
```

- Tendência de serem baixos o dia todo – quando aumentam, é porque aumentam nos dois critérios;

- Por indicar questões técnicas, problemas podem ser mais aleatórios.

SOLUÇÃO

- Rule-based System: *query* SQL busca a base histórica e gera percentis de interesse para cada *status* analisado (‘denied’, ‘failed’ e ‘reversed’);
- Operações novas chegam via API;
- Após verificar que operações são novas, análise é feita para ver se dados estão dentro do esperado (regra de percentil / horário);
- Caso alguma anomalia seja detectada, é realizado envio de email notificando;
- Operações novas são enviadas via *query* para o banco de dados;
- Processo em loop;
- Dashboard em Power BI conectado com o banco de dados para visualização em tempo real.

Funções

```
#criando função final para chamar API, analisar e update via query SQL
def query_data(api,cnxn_str):
```

```
    # função para coleta da base de dados em SQL
    def get_database(cnxn_str):
```

```
    # criando função que analisa os dados e faz envio de email, se precisar
    def analyze_data(base, dic_perc, dic_abs, data):
```

```
        # criando função para envio do email
        def send_alert(time_str, status_str):
```

Regras

- ‘denied’ operations: Divisão por horário (0h – 6h; 6h – 24h)
- ‘reversed’ e ‘failed’ operations: Mesmo critério o dia todo;
- Para as 3, critério é estar acima do 95º percentil da base histórica, ou em valor absoluto ou em questão relativa ao total de operações do horário.

Código final

```
while True:
    query_data(API_ENDPOINT,cnxn_str)

    tm.sleep(10)
```

Exemplo de Funcionamento

```
data = {
    "operations": [
        {
            "dia":{"2023-01-01"},
            "time": {"00h 01"},
            "status": {"denied"},
            "count": {"110"}
        },
        {
            "dia":{"2023-01-01"},
            "time": {"00h 01"},
            "status": {"approved"},
            "count": {"100"}
        }
    ]
}

for operation in data['operations']:
    operation['time'] = list(operation['time'])
    operation['status'] = list(operation['status'])
    operation['count'] = list(operation['count'])
    operation['dia'] = list(operation['dia'])

data_api = json.dumps(data)
```

- Operações recebidas via API fictícia API_ENDPOINT = "https://example.com/api";

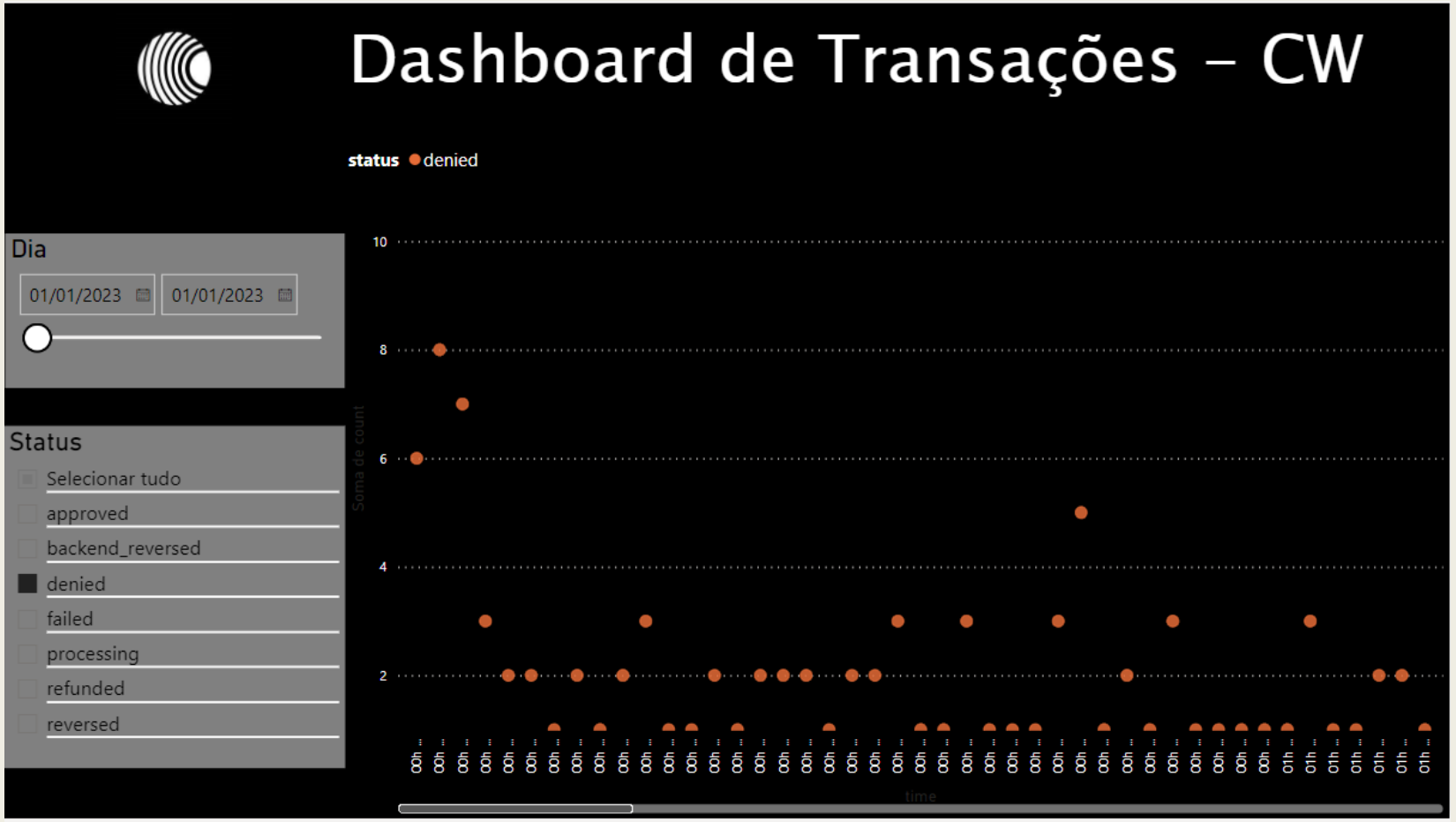
- Deve avisar, visto que a quantidade de operações ‘denied’ é muito alta, tanto em valores absolutos quanto relativos.

Situação Pré-Operações

```
SELECT *

FROM dbo.Transactions_CW

WHERE dia = '2023-01-01' AND time = '00h 01'
```

	dia	time	status	count	minutos	id_op
1	2023-01-01	00h 01	denied	8	1	2023-01-0100h 018denied
2	2023-01-01	00h 01	approved	13	1	2023-01-0100h 0113approved

Situação Pré-Operações – Power BI




```
= Sql.Database("LAPTOP-RE2P03AH", "Teste", [Query="SELECT *#(lf)#(lf)FROM dbo.Transactions_CW#(lf)#
    (lf)WHERE DATEDIFF(day,dia,GETDATE()) between 0 and 365"])
```

Código Teste

```
base, dic_perc, dic_abs = get_database(cnxn_str)

df_hora = analyze_data(base, dic_perc, dic_abs, response.text)

try:
    cnxn = pyodbc.connect(cnxn_str)

    cursor = cnxn.cursor()

    for index, row in df_hora.iterrows():
        cursor.execute("INSERT INTO dbo.Transactions_CW (dia,time,status,count,minutos,id_op) values("
                    +"'"+str(df_hora.dia[index])
                    +"',"
                    +"'"+str(df_hora.time[index])
                    +"',"
                    +"'"+str(df_hora.status[index])
                    +"',"
                    +str(df_hora['count'][index])
                    +","
                    +str(df_hora['minutos'][index])
                    +","
                    +"'"+str(df_hora['id_op'][index])+
                    "')"
                   )

    cnxn.commit()
    cursor.close()

    print('Dados enviados à base de dados')
```

Substituir response.text por data_api para simular com dados fictícios e não uma API.

Resultados

```
Base de dados coletada com sucesso.
Error sending alert: (535, b'5.7.8 Username and Password not
accepted. Learn more at\n5.7.8  https://support.google.com/mail/?
p=BadCredentials n6-20020a4a4006000000b0052a77e38722sm2444615ooa.26
- gsmtp')
Dados enviados à base de dados
```

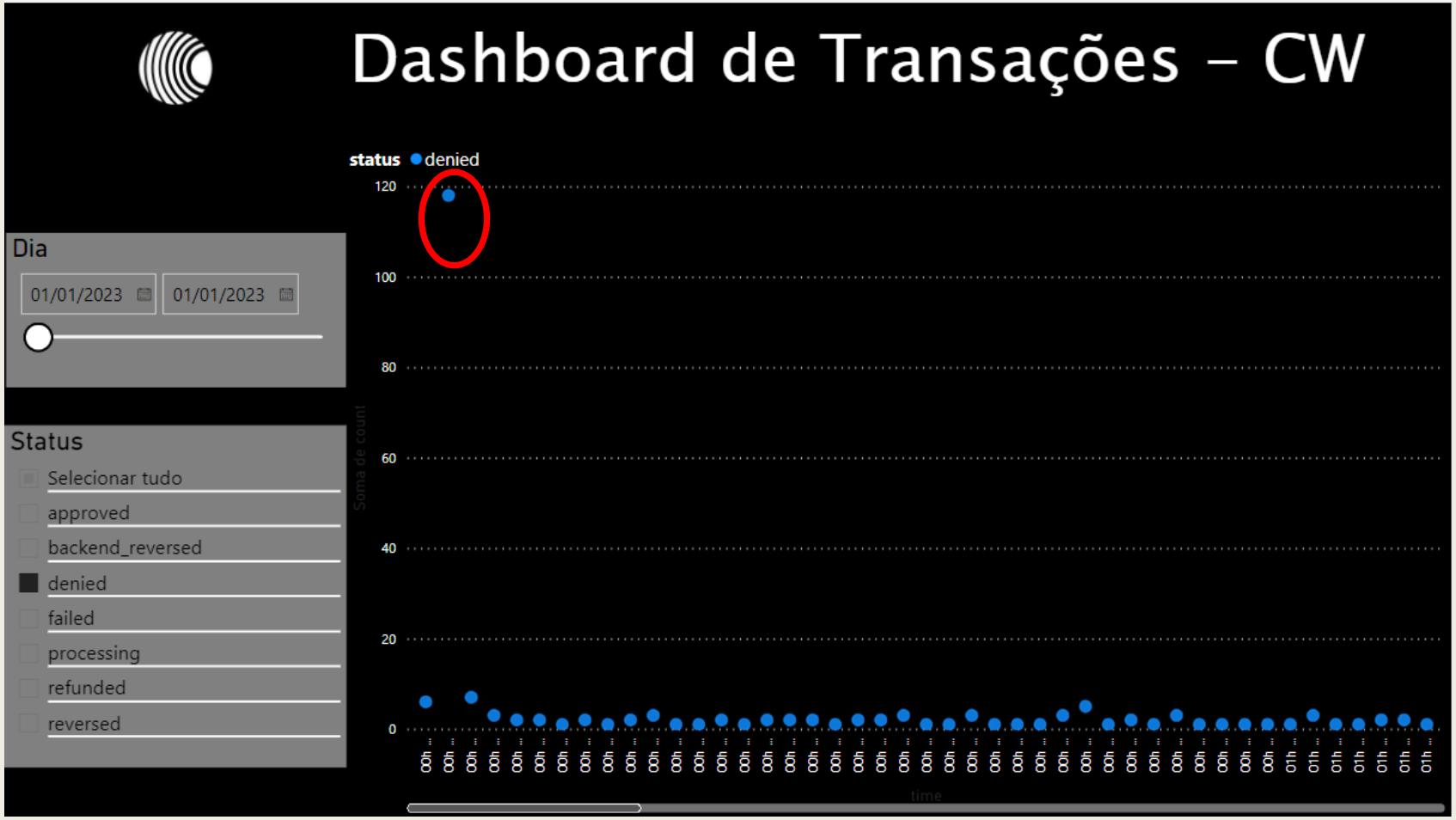
message = "Alerta: Comportamento suspeito no seguinte horário: "+time_str+", quando operações "+status_str+" estavam acima do esperado. Favor investigar."

Situação Pós-Operações

```
SELECT *

FROM dbo.Transactions_CW

WHERE dia = '2023-01-01' AND time = '00h 01'
```

	dia	time	status	count	minutos	id_op
1	2023-01-01	00h 01	denied	8	1	2023-01-0100h 018denied
2	2023-01-01	00h 01	approved	13	1	2023-01-0100h 0113approved
3	2023-01-01	00h 01	denied	110	1	2023-01-0100h 01110denied
4	2023-01-01	00h 01	approved	100	1	2023-01-0100h 01100approved

Situação Pós-Operações




```
= Sql.Database("LAPTOP-RE2P03AH", "Teste", [Query="SELECT *#(lf)#(lf)FROM dbo.Transactions_CW#(lf)#
    (lf)WHERE DATEDIFF(day,dia,GETDATE()) between 0 and 365"])
```

Obrigado!

Todos os códigos, com comentários, disponíveis em
https://github.com/rkujom/case_cw

- Tarefa 1: “Anomalias_Chekouts_Dias1a3.py”
- Tarefa 2: Análise Exploratória “Anomalias_Transactions_CW.py”

  Sistema Monitoramento: “Monit_system_CW.py”

  Dashboard Power BI: “Gráfico_Transactions_CW.pbix”