

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
NÚCLEO DE EDUCAÇÃO A DISTÂNCIA
Pós-graduação *Lato Sensu* em Ciência de Dados e Big Data

Adriano da Silva Matias Fonseca

APRENDIZADO DE MAQUINA APLICADO PARA A PREDIÇÃO DOS *LEADS*
MAIS PROPENSOS À COMPRA

Belo Horizonte

2022

Adriano da Silva Matias Fonseca

**APRENDIZADO DE MAQUINA APLICADO PARA A PREDIÇÃO DOS LEADS
MAIS PROPENSOS À COMPRA**

Trabalho de Conclusão de Curso apresentado
ao Curso de Especialização em Ciência de
Dados e Big Data como requisito parcial à
obtenção do título de especialista.

Belo Horizonte

2022

SUMÁRIO

1. Introdução.....	4
1.1. Contextualização	4
1.2. O problema proposto	4
2. Coleta de Dados	5
3. Processamento/Tratamento dos Dados	18
4. Análise e Exploração dos Dados	24
6. Interpretação dos Resultados	81
7. Apresentação dos Resultados	97
PREDICTION TASK	97
DECISIONS.....	97
VALUE PROPOSITION	97
DATA COLLECTION	97
DATA SOURCES.....	97
IMPACT SIMULATION	97
MAKING PREDICTIONS	97
BUILDING MODELS.....	97
FEATURES	97
MONITORING	97
8. Links.....	98
REFERÊNCIAS.....	99

1. Introdução

1.1. Contextualização

Com o objetivo de estreitar a relação entre *online* e *offline*, aprimorando, assim, a experiência dos *Leads* (alguém que forneceu seus dados de contato) e de expandir do *multichannel* para *omnichannel* como estratégia de uso simultâneo e interligado de diferentes canais de comunicação da minha organização, o presente estudo busca identificar e atribuir um score de probabilidade para *Leads* receptivos, através de suas características, utilizando *data science*.

1.2. O problema proposto

“Uma habilidade crucial em *data science* é a capacidade de decompor um problema analítico de dados, de forma que cada parte corresponda a uma tarefa conhecida para a qual ferramentas estão disponíveis. Reconhecer problemas familiares e suas soluções evita desperdício de tempo e de recursos reinventando a roda” (FOSTER; TOM, 2016, p.20).

O presente projeto tem como desafio propor uma abordagem preditiva para identificar a qual classe um *Lead* pertence, com o objetivo de prever se o *Lead* estaria ou não interessado na compra de planos de internet.

(*Why?*) Por que esse problema é importante?

Com a solução, os times de vendas, planejamento e *marketing* esperam conseguir priorizar os *Leads* com maior interesse nos planos de internet.

(*Who?*) De quem são os dados analisados?

Os dados analisados são de uma empresa privada.

(*What?*): Quais os objetivos com essa análise? O que iremos analisar?

O objetivo é analisar o comportamento preditivo dos *Leads* através das suas características de recarga e serviços enriquecidas.

(*Where?*): Trata dos aspectos geográficos e logísticos de sua análise.
O estudo é realizado com *Leads* espalhados em todo território brasileiro.

(*When?*): Qual o período está sendo analisado?
Podemos considerar o terceiro e quarto trimestre do ano 2021.

2. Coleta de Dados

O conjunto de dados utilizado neste projeto, foi obtido via Python e linguagem *Structured Query Language* (SQL), através de conexão no banco de dados *MySQL*, carregando os *Leads* da base de dados do produto *chat* da minha organização.

Um *Lead* é considerado como uma oportunidade de negócio para a organização, é alguém que já demonstrou interesse no produto.

```
def get_contact(contact):
    # Get contact
    query = """
        SELECT DISTINCT
            l.id_contact,
            l.plan_type
        FROM
            lead.leads AS l
        WHERE l.is_chat = 1
            AND l.id_campaign = 25
            AND l.id_contact = {}
            AND l.created_at BETWEEN '2021-08-19 00:00:00' AND '2021-10-31 23:59:59';
    """.format(contact)
    data = pd.read_sql_query(query, db.get_connection('con_mysql'))
    return data
```

Figura 1 – Função seleção dos *Leads* (dados fornecidos pelo *Leads*)

Na continuação da construção do *dataset*, demais características e dados enriquecidos, foram obtidos também via linguagem SQL no banco de dados *MySQL*.

```

def get_recharges(contact):
    # Get recharge frequency and total spent
    query = """
        SELECT DISTINCT
            rec.id_contact,
            SUM(rec.value) AS sum_recharge,
            COUNT(1) AS recharge_frequency
        FROM (
            SELECT DISTINCT
                id_contact, `type`, value, date_time
            FROM
                carrier.costs
            WHERE
                id_contact IN (
                    SELECT id_contact
                    FROM lead.leads
                    WHERE
                        is_chat = 1
                        AND data LIKE '%Chat%'
                        AND id_campaign = 25
                        AND id_contact {}
                        AND created_at BETWEEN '2021-08-19 00:00:00' AND '2021-10-31 23:59:59')) AS rec
        GROUP BY rec.id_contact;
    """.format(contact)
    data = pd.read_sql_query(query, db.get_connection('con_mysql'))
    data.drop(['id_contact'], axis=1, inplace=True)
    return data

```

Figura 2 – Função seleção das características de recargas enriquecidas dos *Leads*

```

def get_recharges_types(contact):
    # Get recharges data
    query = """
        SELECT
        *
        FROM (
            SELECT DISTINCT
                rec.id_contact, rec_online_10,
                rec_online_35_b5, rec_online_15,
                sos_rec_5, rec_online_20_b2,
                chip_pre_rec_10, crip_pre_rec_20,
                rec_online_13, rec_online_50_b8,
                rec_online_30_b4, rec_online_40_b6,
                pct_rec_1190, pct_rec_690,
                rec_online_100_b18, pct_rec_sos_5,
                sos_rec_3, rec_online_8
            FROM (
                SELECT DISTINCT
                    rec.id_contact,
                    SUM(rec.rec_online_10) AS rec_online_10,
                    SUM(rec.rec_online_35_b5) AS rec_online_35_b5,
                    SUM(rec.rec_online_15) AS rec_online_15,
                    SUM(rec.sos_rec_5) AS sos_rec_5,
                    SUM(rec.rec_online_20_b2) AS rec_online_20_b2,
                    SUM(rec.chip_pre_rec_10) AS chip_pre_rec_10,
                    SUM(rec.crip_pre_rec_20) AS crip_pre_rec_20,
                    SUM(rec.rec_online_13) AS rec_online_13,
                    SUM(rec.rec_online_50_b8) AS rec_online_50_b8,
                    SUM(rec.rec_online_30_b4) AS rec_online_30_b4,
                    SUM(rec.rec_online_40_b6) AS rec_online_40_b6,
                    SUM(rec.pct_rec_1190) AS pct_rec_1190,
                    SUM(rec.pct_rec_690) AS pct_rec_690,
                    SUM(rec.rec_online_100_b18) AS rec_online_100_b18,
                    SUM(rec.pct_rec_sos_5) AS pct_rec_sos_5,
                    SUM(rec.sos_rec_3) AS sos_rec_3,
                    SUM(rec.rec_online_8) AS rec_online_8
                FROM (
                    SELECT DISTINCT
                        rec.id_contact, rec.date_time,
                        IF(rec.type LIKE '%RECARGA ONLINE R$10%', 1, 0) AS rec_online_10,
                        IF(rec.type LIKE '%RECARGA ONLINE R$35 + BONUS R$5%', 1, 0) AS rec_online_35_b5,
                        IF(rec.type LIKE '%RECARGA ONLINE R$15%', 1, 0) AS rec_online_15,
                        IF(rec.type LIKE '%SOS RECARGA - R$5%', 1, 0) AS sos_rec_5,
                        IF(rec.type LIKE '%RECARGA ONLINE R$20 + BONUS R$2%', 1, 0) AS rec_online_20_b2,
                        IF(rec.type LIKE '%CHIP PRE + RECARGA R$10%', 1, 0) AS chip_pre_rec_10,
                        IF(rec.type LIKE '%CHIP PRE + RECARGA R$20%', 1, 0) AS crip_pre_rec_20,
                        IF(rec.type LIKE '%RECARGA ONLINE R$13%', 1, 0) AS rec_online_13,
                        IF(rec.type LIKE '%RECARGA ONLINE R$50 + BONUS R$8%', 1, 0) AS rec_online_50_b8,
                        IF(rec.type LIKE '%RECARGA ONLINE R$30 + BONUS R$4%', 1, 0) AS rec_online_30_b4,
                        IF(rec.type LIKE '%RECARGA ONLINE R$40 + BONUS R$6%', 1, 0) AS rec_online_40_b6,
                        IF(rec.type LIKE '%PACOTE RECARGA R$11,90%', 1, 0) AS pct_rec_1190,
                        IF(rec.type LIKE '%PACOTE RECARGA R$6,90%', 1, 0) AS pct_rec_690,
                        IF(rec.type LIKE '%RECARGA ONLINE R$100 + BONUS R$18%', 1, 0) AS rec_online_100_b18,
                        IF(rec.type LIKE '%PACOTE SOS RECARGA R$5,00%', 1, 0) AS pct_rec_sos_5,
                        IF(rec.type LIKE '%SOS RECARGA - R$3%', 1, 0) AS sos_rec_3,
                        IF(rec.type LIKE '%RECARGA ONLINE R$8%', 1, 0) AS rec_online_8
                    FROM (
                        SELECT DISTINCT
                            id_contact, 'type', value, date_time
                        FROM carrier.costs
                        WHERE id_contact IN (
                            SELECT id_contact
                            FROM lead.leads
                            WHERE
                                is_chat = 1
                                AND data LIKE '%Chat%'
                                AND id_campaign = 25
                                AND id_contact {}
                                AND created_at BETWEEN '2021-08-19 00:00:00' AND '2021-10-31 23:59:59')) AS rec
                    GROUP BY rec.id_contact , rec.type , rec.date_time) AS rec
                GROUP BY rec.id_contact) AS rec) AS tab;
        """.format(contact)
    data = pd.read_sql_query(query, db.get_connection('con_mysql'))
    data.drop(['id_contact'], axis=1, inplace=True)
    return data

```

Figura 3 – Função seleção das características de recargas enriquecidas dos Leads

```

def get_services(contact):
    # Get services data
    query = """
        SELECT DISTINCT
            serv.id_contact,
            SUM(serv.value) AS sum_services,
            COUNT(1) AS services_frequency
        FROM (
            SELECT DISTINCT
                id_contact, description, value, date_time
            FROM carrier.services
            WHERE
                is_chat = 1
                AND data LIKE '%Chat%'
                AND id_campaign = 25
                AND id_contact {}
                AND created_at BETWEEN '2021-08-19 00:00:00' AND '2021-10-31 23:59:59')) AS serv
    GROUP BY serv.id_contact;
    """.format(contact)
    data = pd.read_sql_query(query, db.get_connection('con_mysql'))
    data.drop(['id_contact'], axis=1, inplace=True)
    return data

```

Figura 4 – Função seleção das características de serviços enriquecidas dos *Leads*


```

def get_services_types(contact):
    # Get services types
    query = """
        SELECT *
        FROM (
            SELECT DISTINCT
                serv.id_contact, inter_avulsa,
                antivirus, app_educacao,
                app_emprego, app_saude,
                clube, pre_mix_giga,
                entretenimento, games,
                pct_internet_mensal, prezao_diario,
                prezao_mensal, prezao_quinzenal,
                prezao_semanal, recarga_sos,
                servicos_operadora, set1,
                sms_cobrar, sms_internacional,
                transf_entre_regionais, truecaller
            FROM (
                SELECT DISTINCT
                    serv.id_contact,
                    SUM(serv.inter_avulsa) AS inter_avulsa,
                    SUM(serv.antivirus) AS antivirus,
                    SUM(serv.app_educacao) AS app_educacao,
                    SUM(serv.app_emprego) AS app_emprego,
                    SUM(serv.app_saude) AS app_saude,
                    SUM(serv.clube) AS clube,
                    SUM(serv.pre_mix_giga) AS pre_mix_giga,
                    SUM(serv.entretenimento) AS entretenimento,
                    SUM(serv.games) AS games,
                    SUM(serv.pct_internet_mensal) AS pct_internet_mensal,
                    SUM(serv.prezao_diario) AS prezao_diario,
                    SUM(serv.prezao_mensal) AS prezao_mensal,
                    SUM(serv.prezao_quinzenal) AS prezao_quinzenal,
                    SUM(serv.prezao_semanal) AS prezao_semanal,
                    SUM(serv.recarga_sos) AS recarga_sos,
                    SUM(serv.servicos_operadora) AS servicos_operadora,
                    SUM(serv.set1) AS set1,
                    SUM(serv.sms_cobrar) AS sms_cobrar,
                    SUM(serv.sms_internacional) AS sms_internacional,
                    SUM(serv.transf_entre_regionais) AS transf_entre_regionais,
                    SUM(serv.truecaller) AS truecaller
                FROM (
                    SELECT DISTINCT
                        serv.id_contact,
                        serv.date_time,
                        IF(serv.description LIKE '%internet avulsa%'
                            OR serv.description LIKE '%PACOTE DIARIO 100MB + WHATSAPP AD%'
                            OR serv.description LIKE '%PACOTE DIARIO 200MB + WHATSAPP AD%'
                            OR serv.description LIKE '%PCT 500MB MAIS 7 DIAS%'
                            OR serv.description LIKE '%PCT ADICIONAL 1GB 7 DIAS%'
                            OR serv.description LIKE '%PCT DIARIO 100MB + WHATSAPP%'
                            OR serv.description LIKE '%PCT DIARIO 200MB + WHATSAPP%'
                            OR serv.description LIKE '%TUDO POR R$1.49/DIA%', 1, 0) AS inter_avulsa,
                        IF(serv.description LIKE '%NORTON%'
                            OR serv.description LIKE '%SEGURANCA HERO%', 1, 0) AS antivirus,
                        IF(serv.description LIKE '%IDIOMAS BY BUSUU%'
                            OR serv.description LIKE '%FUN ENGLISH WITH DOKI%'
                            OR serv.description LIKE '%INGLES MAGICO APP%', 1, 0) AS app_educacao,

```

```

IF(serv.description LIKE '%MINUTO CARREIRA%', 1, 0) AS app_emprego,
IF(serv.description LIKE '%BTFIT%', 1, 0) AS app_saude,
IF(serv.description LIKE '%CONECTA%'
    OR serv.description LIKE '%FOTO%'
    OR serv.description LIKE '%FOTO - OUTRAS OPERADORAS%'
    OR serv.description LIKE '%FOTO PARA E-MAIL%'
    OR serv.description LIKE '%GAMES%'
    OR serv.description LIKE '%NOTICIAS%'
    OR serv.description LIKE '%CLUBE APPS%', 1, 0) AS clube,
IF(serv.description LIKE '%PRE MIX GIGA%', 1, 0) AS pre_mix_giga,
IF(serv.description LIKE '%BADOO - COMPRA AVULSA%'
    OR serv.description LIKE '%BAND SPORTS%'
    OR serv.description LIKE '%CLUBE KIDS%'
    OR serv.description LIKE '%CLUBE MARVEL%'
    OR serv.description LIKE '%CLUBE STARWARS%'
    OR serv.description LIKE '%CUPIDOO%'
    OR serv.description LIKE '%DESCOMPLICA%'
    OR serv.description LIKE '%DISCOVERY KIDS ON%'
    OR serv.description LIKE '%DISNEY CUBES%'
    OR serv.description LIKE '%EGO%'
    OR serv.description LIKE '%FOX SPORTS GOL%'
    OR serv.description LIKE '%JOGADA MUSICAL%'
    OR serv.description LIKE '%JOGADA DE PREMIO%'
    OR serv.description LIKE '%LIVROH%'
    OR serv.description LIKE '%MIX DE VANTAGENS%'
    OR serv.description LIKE '%PASSATEMPO PREMIADO%'
    OR serv.description LIKE '%PET DICAS%'
    OR serv.description LIKE '%PLAYKIDS%'
    OR serv.description LIKE '%PLAY KIDS%'
    OR serv.description LIKE '%POCOYO HOUSE%'
    OR serv.description LIKE '%PRIME TUBE%'
    OR serv.description LIKE '%QUERO DESCONTOS%'
    OR serv.description LIKE '%RADAR%'
    OR serv.description LIKE '%SOCIAL COMICS%'
    OR serv.description LIKE '%SONHO PREMIADO%'
    OR serv.description LIKE '%SX REVOLUTION%'
    OR serv.description LIKE '%UBOOK%'
    OR serv.description LIKE '%UMBARATO%'
    OR serv.description LIKE '%UOL CURSO DE BOLSO%'
    OR serv.description LIKE '%VIAJE MAIS%'
    OR serv.description LIKE '%VOCE GOURMET%', 1, 0) AS entretenimento,
IF(serv.description LIKE '%APP GAME%'
    OR serv.description LIKE '%FUTCEL%'
    OR serv.description LIKE '%GAMELOFT ITEM DE JOGO%'
    OR serv.description LIKE '%GAMES FUN%'
    OR serv.description LIKE '%JOGOS DE SEMPRE%'

```

```

        OR serv.description LIKE '%NEYMAR JR. EXPERIENCE%'
        OR serv.description LIKE '%PROMO GENIUS GAME%'
        OR serv.description LIKE '%QUIZ SUPER - JOGO%', 1, 0) AS games,
    IF(serv.description LIKE '%PACOTE MENSAL 1GB + WHATSAPP GRATIS%'
        OR serv.description LIKE '%PACOTE MENSAL 2GB + WHATSAPP GRATIS%'
        OR serv.description LIKE '%PACOTE MENSAL 300MB + WHATSAPP GRATIS%'
        OR serv.description LIKE '%PACOTE MENSAL 500MB + WHATSAPP GRATIS%', 1, 0) AS pct_internet_mensal,
    IF(serv.description LIKE '%PREZAO DIARIO 100MB + WHATSAPP + SMS%'
        OR serv.description LIKE '%PREZAO DIARIO 100MB +WHATS+ SMS 1,99/DIA%', 1, 0) AS prezao_diario,
    IF(serv.description LIKE '%PREZAO 34,99 POR MES%'
        OR serv.description LIKE '%PREZAO 14,99 POR MES%'
        OR serv.description LIKE '%PREZAO R$19,99 POR MES%'
        OR serv.description LIKE '%PREZAO FALA MAIS MENSAL%', 1, 0) AS prezao_mensal,
    IF(serv.description LIKE '%PREZAO FALA MAIS 14 DIAS%'
        OR serv.description LIKE '%PREZAO MUITO MAIS 14 DIAS%', 1, 0) AS prezao_quinzenal,
    IF(serv.description LIKE '%PREZAO 9,99 POR SEMANA%'
        OR serv.description LIKE '%PREZAO 14,99 POR SEMANA%', 1, 0) AS prezao_semanal,
    IF(serv.description LIKE '%SOS RECARGA - DEBITO EMPRESTIMO APP%'
        OR serv.description LIKE '%SOS RECARGA - DEBITO R$2,00%'
        OR serv.description LIKE '%SOS RECARGA - DEBITO R$5,00%'
        OR serv.description LIKE '%SOS RECARGA - DEBITO TAXA APP%', 1, 0) AS recarga_sos,
    IF(serv.description LIKE '%ADESAO CHIP NATURA%'
        OR serv.description LIKE '%AJUSTES DE CREDITOS%'
        OR serv.description LIKE '%RECADO%'
        OR serv.description LIKE '%RECADO PREMIUM%'
        OR serv.description LIKE '%ESTORNO AUTOMATICO DE ANATEL 226%'
        OR serv.description LIKE '%ESTORNO TORPEDO%'
        OR serv.description LIKE '%EXPIRACAO DE CREDITO APOS 30 DIAS.%'
        OR serv.description LIKE '%RECADO AVULSO%'
        OR serv.description LIKE '%SOM DE CHAMADA%', 1, 0) AS servicos_operadora,
    IF(serv.description LIKE '%set%', 1, 0) AS set1,
    IF(serv.description LIKE '%TORPEDO A COBRAR%'
        OR serv.description LIKE '%TORPEDO A COBRAR DEGRAU OUTRAS OPER%', 1, 0) AS sms_cobrar,
    IF(serv.description LIKE '%TORPEDO - INTERNACIONAL%', 1, 0) AS sms_internacional,
    IF(serv.description LIKE '%TRANSFERENCIA ENTRE REGIONAIS%', 1, 0) AS transf_entre_regionais,
    IF(serv.description LIKE '%TRUECALLER%', 1, 0) AS truecaller
FROM (
    SELECT DISTINCT
        id_contact, description, value, date_time
    FROM carrier.services
    WHERE id_contact IN (
        SELECT id_contact
        FROM lead.leads
        WHERE
            is_chat = 1
            AND data LIKE '%Chat%'
            AND id_campaign = 25
            AND id_contact {}
            AND created_at BETWEEN '2021-08-19 00:00:00' AND '2021-10-31 23:59:59')) AS serv
GROUP BY serv.id_contact, serv.description, serv.date_time) AS serv
GROUP BY serv.id_contact) AS serv) AS tab;

"""format(contact)
data = pd.read_sql_query(query, db.get_connection('con_mysql'))
data.drop(['id_contact'], axis=1, inplace=True)
return data

```

Figura 5 – Função seleção das características de serviços enriquecidas dos *Leads*

```

def get_abandonments(contact):
    # Get the abandonment volume
    query = """
        SELECT
            id_contact, count(id) as Qnt_abandono
        FROM abandonment.customers
        WHERE
            customers.id_contact {}
            AND customers.id_campaign = 25
            AND customers.params LIKE '%Chat%'
            AND customers.created_at BETWEEN '2021-08-19 00:00:00' AND '2021-10-31 23:59:59'
        ORDER BY customers.id_contact DESC;
    """
    """format(contact)
    data = pd.read_sql_query(query, db.get_connection('con_mysql'))
    data.drop(['id_contact'], axis=1, inplace=True)
    return data

```

Figura 6 – Função seleção de abandono dos *Leads*

```
def get_client_age(contact):
    # Get client age
    query = """
        SELECT
            year(now()) - year(birthday) as idade_cliente
        FROM customer.customers
        WHERE id in (
            SELECT id_customer
            FROM customer.contacts
            WHERE
                id = {}
                AND customers.id_campaign = 25
                AND customers.params LIKE '%Chat%'
                AND customers.created_at BETWEEN '2021-08-19 00:00:00' AND '2021-10-31 23:59:59';
        """.format(contact)
    data = pd.read_sql_query(query, db.get_connection('con_mysql'))
    return data
```

Figura 7 – Função seleção da idade dos *Leads*

```
def get_sale(contact):
    # Get sale
    query = """
        SELECT DISTINCT
            l.id_contact, IF(sl.id IS NULL, 0, 1) AS venda
        FROM
            lead.leads AS l
            LEFT JOIN
            sale.sales AS sl ON l.id_contact = sl.id_contact
        WHERE
            l.is_chat = 1
            AND l.data LIKE '%Chat%'
            AND l.id_campaign = 25
            AND l.id_contact {}
            AND l.created_at BETWEEN '2021-08-19 00:00:00' AND '2021-10-31 23:59:59'
        GROUP BY l.id_contact;
    """.format(contact)
    data = pd.read_sql_query(query, db.get_connection('con_mysql'))
    return data
```

Figura 8 – Função seleção *SALES*

```
def get_main_base(contact):
    # Concat
    data = get_contact(contact)
    data1 = get_recharges(contact)
    data2 = get_recharges_types(contact)
    data3 = get_services(contact)
    data4 = get_services_types(contact)
    data5 = get_abandonments(contact)
    data6 = get_client_age(contact)
    data7 = get_sale(contact)
    res = pd.concat([data, data1, data2, data3, data4, data5, data6, data7], axis=1)
    # Get .csv
    res.to_csv('data_ptd.csv', sep=';', index=False)
    return res
```

Figura 9 – Função *concat* para geração do *dataset*

Nome variáveis/campo	Descrição	Tipo
regional	<i>FEATURE</i> da Região do <i>Leads</i>	<i>int32</i>
idade_cliente	<i>FEATURE</i> da Idade do <i>Leads</i>	<i>int64</i>
plan_type	<i>FEATURE</i> Tipo do plano do <i>Leads</i>	<i>int32</i>
Qnt_abandono	<i>FEATURE</i> de quantas vezes o <i>Leads</i> abandonou <i>chat</i>	<i>float64</i>
sum_recharge	<i>FEATURE</i> com a soma da recarga do <i>Leads</i>	<i>float64</i>
recharge_frequency	<i>FEATURE</i> que traz a frequência de recarga do <i>Leads</i>	<i>float64</i>
rec_online_10	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou a recarga <i>online</i> de 10 reais	<i>float64</i>
rec_online_35_b5	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou a recarga <i>online</i> de 35 reais + bônus	<i>float64</i>
rec_online_15	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou a recarga <i>online</i> de 15	<i>float64</i>
sos_rec_5	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou a recarga SOS 5	<i>float64</i>
rec_online_20_b2	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou a recarga <i>online</i> de 20 reais + bônus	<i>float64</i>
chip_pre_rec_10	<i>FEATURE</i> que representa o	<i>float64</i>

	contador de quantas vezes o <i>Leads</i> utilizou a recarga <i>chip</i> PRÉ 10	
chip_pre_rec_20	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou a recarga <i>chip</i> PRÉ 20	<i>float64</i>
rec_online_13	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou a recarga <i>online</i> 13	<i>float64</i>
rec_online_50_b8	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou a recarga <i>online</i> 50 reais + bônus	<i>float64</i>
rec_online_30_b4	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou a recarga <i>online</i> 30 reais + bônus	<i>float64</i>
rec_online_40_b6	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou a recarga <i>online</i> 40 reais + bônus	
pct_rec_1190	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou a recarga 12	<i>float64</i>
pct_rec_690	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou a recarga de 7	<i>float64</i>
rec_online_100_b18	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou recarga <i>online</i> 100 reais + bônus	<i>float64</i>
pct_rec_sos_5	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou pacote recarga	<i>float64</i>

	SOS 5	
sos_rec_3	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou recarga SOS 3	<i>float64</i>
rec_online_8	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou recarga <i>online</i> 8	<i>float64</i>
sum_services	<i>FEATURE</i> que representa soma dos valores dos serviços	<i>float64</i>
services_frequency	<i>FEATURE</i> que representa número de serviços utilizados pelo <i>Leads</i>	<i>float64</i>
inter_avulsa	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou serviços <i>internet</i> avulsa	<i>float64</i>
antivirus	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou serviços <i>antivírus</i>	<i>float64</i>
app_educacao	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou serviços app educação	<i>float64</i>
app_emprego	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou serviços app emprego	<i>float64</i>
app_saude	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou serviços app saúde	<i>float64</i>
Clube	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou serviços do	<i>float64</i>

	clube	
pre_mix_giga	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou serviços <i>PRÉ mix giga</i>	<i>float64</i>
entretenimento	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou serviços de entretenimento	<i>float64</i>
games	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou serviços de <i>games</i>	<i>float64</i>
pct_internet_mensal	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou serviços pacote <i>internet</i> mensal	<i>float64</i>
prezao_diario	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou serviços <i>prezão</i> diário	<i>float64</i>
prezao_mensal	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou serviços <i>prezão</i> mensal	<i>float64</i>
prezao_quinzenal	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou serviços <i>prezão</i> quinzenal	<i>float64</i>
prezao_semanal	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou serviços <i>prezão</i> semanal	<i>float64</i>
recarga_sos	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou serviços	<i>float64</i>

	recarga SOS	
servicos_operadora	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou serviços da operadora	<i>float64</i>
sms_cobrar	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou serviços sms a cobrar	<i>float64</i>
sms_internacional	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou serviços sms internacional	<i>float64</i>
transf_entre_regionais	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou serviços trans x regionais	<i>float64</i>
truecaller	<i>FEATURE</i> que representa o contador de quantas vezes o <i>Leads</i> utilizou serviços <i>truecaller</i>	<i>float64</i>
jovem	<i>FEATURE</i> segmentação da idade	<i>int64</i>
adulto	<i>FEATURE</i> segmentação da idade	<i>int64</i>
meia_idade	<i>FEATURE</i> segmentação da idade	<i>int64</i>
idoso	<i>FEATURE</i> segmentação da idade	<i>int64</i>
venda	Variáveis que obtém as classes de venda e não venda do <i>dataset</i> , pode ser considerada como objetivo do estudo.	<i>int64</i>

Tabela 1 – Caraterísticas enriquecidas dos *Leads*

3. Processamento/Tratamento dos Dados

Estatísticas do conjunto de dados

Número de variáveis	47
Número de observações	53257
Células ausentes	1474378
Células ausentes (%)	58,9%
Linhas duplicadas	0
Linhas duplicadas (%)	0.0%

Figura 10 – Visão geral do *dataset*

Na Figura 10, podemos observar 47 variáveis carregados no *dataset*, sendo uma delas o **id_contact** que será descartado no estudo inicial. O *dataset* contém 53.257 linhas com 58,9% de células ausentes devido aos pacotes de recargas ou serviços não utilizados pelos Leads. O *dataset* não tem linhas duplicadas, devido ao **id_contact** ser único.

Tipos de variáveis

Numérico	45
Categórico	2

Figura 11 – Visão geral das variáveis

Na Figura 11, vemos que o *dataset* dispõem de 45 variáveis numéricas e 2 variáveis categóricas.

```

In [13]: 1 data.isnull().sum()

Out[13]: id_contact      0
         regional        0
         idade_cliente    0
         plan_type        0
         Qnt_abandono     0
         sum_recharge     36196
         recharge_frequency 36196
         rec_online_10     36196
         rec_online_35_b5  36196
         rec_online_15     36196
         sos_rec_5        36196
         rec_online_20_b2  36196
         chip_pre_rec_10   36196
         chip_pre_rec_20   36196
         rec_online_13     36196
         rec_online_50_b8  36196
         rec_online_30_b4  36196
         rec_online_40_b6  36196
         pct_rec_1190      36196
         pct_rec_690       36196
         rec_online_100_b18 36196
         pct_rec_sos_5     36196
         sos_rec_3        36196
         rec_online_8      36196
         sum_services     35757
         services_frequency 35757
         inter_avulsa     35757
         antivirous       35757
         app_educacao     35757
         app_emprego      35757
         app_saude        35757
         clube            35757
         pre_mix_giga     35757
         entretenimento   35757
         games            35757
         pct_internet_mensal 35757
         prezao_diario    35757
         prezao_mensal    35757
         prezao_quinzenal  35757
         prezao_semanal    35757
         recarga_sos      35757
         servicos_operadora 35757
         sms_cobrar       35757
         sms_internacional 35757
         transf_entre_regionais 35757
         truecaller       35757
         venda            0

```

Figura 12 – Valores ausentes nas variáveis do *dataset*

```
In [10]: 1 msno.bar(data.loc[:, ['sum_services', 'services_frequency', 'inter_avulsa', 'antivirus',
2                               'app_educacao', 'app_emprego', 'app_saude', 'clube', 'pre_mix_giga',
3                               'entretenimento', 'games', 'pct_internet_mensal', 'prezao_diario',
4                               'prezao_mensal', 'prezao_quinzenal', 'prezao_semanal', 'recarga_sos',
5                               'servicos_operadora', 'sms_cobrar', 'sms_internacional',
6                               'transf_entre_regionais', 'truecaller']], figsize=(25,5))
```

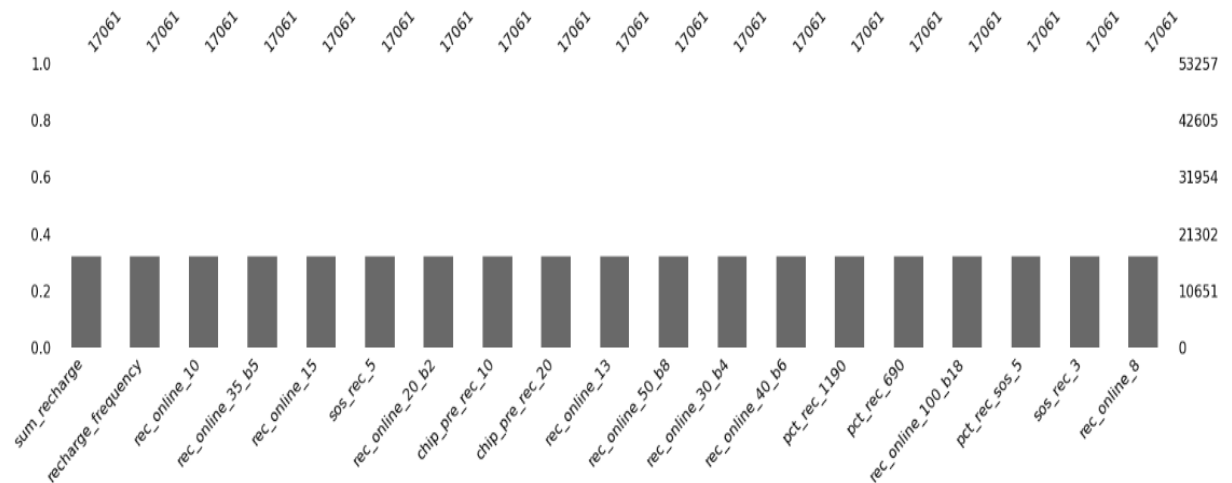


Figura 13 – Valores ausentes do enriquecimento das variáveis de recargas

```
In [13]: 1 msno.bar(data.loc[:, ['sum_services', 'services_frequency', 'inter_avulsa', 'antivirus',
2                               'app_educacao', 'app_emprego', 'app_saude', 'clube', 'pre_mix_giga',
3                               'entretenimento', 'games', 'pct_internet_mensal', 'prezao_diario',
4                               'prezao_mensal', 'prezao_quinzenal', 'prezao_semanal', 'recarga_sos',
5                               'servicos_operadora', 'sms_cobrar', 'sms_internacional',
6                               'transf_entre_regionais', 'truecaller']], figsize=(25,5))
```

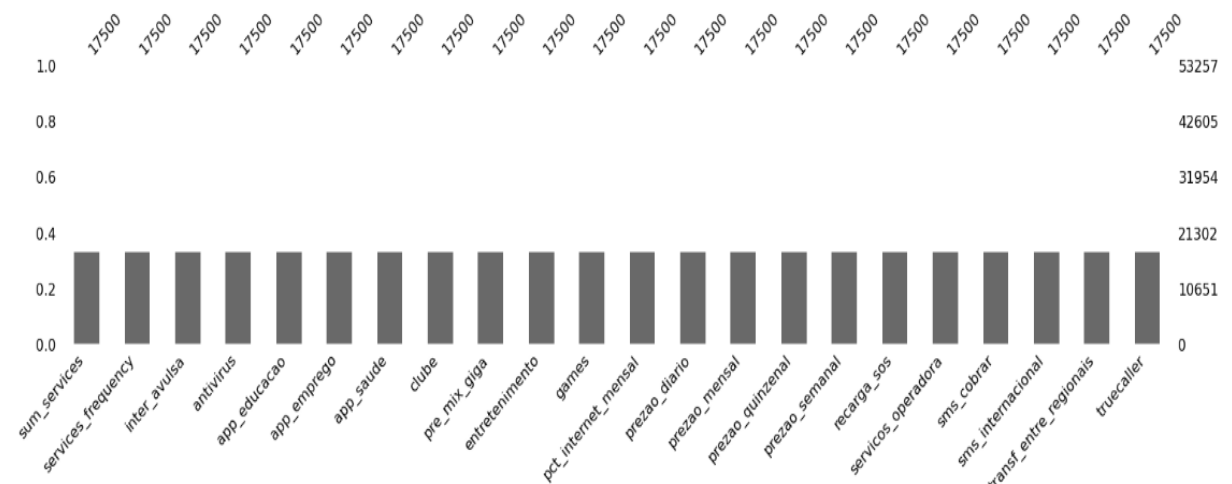


Figura 14 – Valores ausentes do enriquecimento das variáveis de serviços

Observando os valores ausentes na nas Figuras 12, 13 e 14, eles são referentes as variáveis enriquecidas de recargas e serviços e não é descartado inconsistências na imputação de dados, no enriquecimento dos dados, visto que

com êxito, valores zerados para as variáveis de recargas e serviços foram recuperados na base original.

Feito as observações acima, podemos concluir que para o conjunto de dados ausentes, deve-se classificar o mecanismo de ausência destes dados como MCAR – *Missing Completely at Random* (Perdidos completamente de forma Aleatória), pois os dados estão perdidos de forma completamente aleatória, não há justificativa para sua ocorrência. O fato de que um certo valor está faltando não tem nada a ver com seu valor hipotético e com os valores de outras variáveis.

No presente contexto, não justifica eliminarmos as variáveis que foram recuperadas com valores ausentes, sem antes analisarmos a representatividade de cada uma na análise e exploração dos dados. Então, para os valores ausentes nas variáveis de recargas e serviços, vamos eliminar as linhas com valores ausentes, pois 58.9% das células que compõem as variáveis, estão com valores ausentes.

Na sequência, vamos eliminar o *ID Lead* e realizar a eliminação as linhas que contem valores ausentes.

```
# Drop
to_drop=['id_contact']
data.drop(to_drop, axis=1, inplace=True)
```

Figura 15 – Eliminando *ID Lead*

```
# Tratando os valores ausentes
data.dropna(inplace=True)
```

Figura 16 – Tratando valores ausentes

Dando continuidade no tratamento de problemas encontrados no *dataset*, temos a conversão das classes categóricas em numéricas, para realização da conversão, estou utilizando *Label Encoding*.

```
data.select_dtypes(include='object')
```

	regional	plan_type
0	BASE	CONTROLE
13	SP	PRE PAGO
14	RJES	PRE PAGO
17	SP	CONTROLE
19	RJES	PRE PAGO
...
53242	SP	PRE PAGO
53250	NE	PRE PAGO
53253	SP	CONTROLE
53254	SP	PRE PAGO
53256	SP	PRE PAGO

16376 rows × 2 columns

Figura 17 – Variáveis categóricas

```
# Transformar rótulos não numéricos (desde que sejam laváveis e comparáveis) em rótulos numéricos.
var_cat=data.select_dtypes('object')
for col in var_cat:
    data[col] = LabelEncoder().fit_transform(data[col].astype('str'))
```

Figura 18 – Convertendo variáveis categóricas

```

<class 'pandas.core.frame.DataFrame'>
Int64Index: 16376 entries, 0 to 53256
Data columns (total 46 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   regional                              16376 non-null  int32
1   idade_cliente                        16376 non-null  int64
2   plan_type                             16376 non-null  int32
3   Qnt_abandono                         16376 non-null  float64
4   sum_recharge                         16376 non-null  float64
5   recharge_frequency                   16376 non-null  float64
6   rec_online_10                        16376 non-null  float64
7   rec_online_35_b5                     16376 non-null  float64
8   rec_online_15                        16376 non-null  float64
9   sos_rec_5                            16376 non-null  float64
10  rec_online_20_b2                     16376 non-null  float64
11  chip_pre_rec_10                      16376 non-null  float64
12  chip_pre_rec_20                      16376 non-null  float64
13  rec_online_13                        16376 non-null  float64
14  rec_online_50_b8                     16376 non-null  float64
15  rec_online_30_b4                     16376 non-null  float64
16  rec_online_40_b6                     16376 non-null  float64
17  pct_rec_1190                         16376 non-null  float64
18  pct_rec_690                          16376 non-null  float64
19  rec_online_100_b18                   16376 non-null  float64
20  pct_rec_sos_5                        16376 non-null  float64
21  sos_rec_3                            16376 non-null  float64
22  rec_online_8                         16376 non-null  float64
23  sum_services                         16376 non-null  float64
24  services_frequency                   16376 non-null  float64
25  inter_avulsa                         16376 non-null  float64
26  antivirus                            16376 non-null  float64
27  app_educacao                         16376 non-null  float64
28  app_emprego                          16376 non-null  float64
29  app_saude                            16376 non-null  float64
30  clube                                16376 non-null  float64
31  pre_mix_giga                         16376 non-null  float64
32  entretenimento                       16376 non-null  float64
33  games                                16376 non-null  float64
34  pct_internet_mensal                  16376 non-null  float64
35  prezao_diario                        16376 non-null  float64
36  prezao_mensal                        16376 non-null  float64
37  prezao_quinzenal                     16376 non-null  float64
38  prezao_semanal                       16376 non-null  float64
39  recarga_sos                          16376 non-null  float64
40  servicos_operadora                   16376 non-null  float64
41  sms_cobrar                           16376 non-null  float64
42  sms_internacional                    16376 non-null  float64
43  transf_entre_regionais               16376 non-null  float64
44  truecaller                           16376 non-null  float64
45  venda                                16376 non-null  int64
dtypes: float64(42), int32(2), int64(2)
memory usage: 5.7 MB

```

Figura 19 – Dataset tratado

```
# Get .csv
data.to_csv('data_aed.csv', sep=';', index=False)
```

Figura 20 – Gerando .csv para análise e exploração dos dados

4. Análise e Exploração dos Dados

No presente tópico, vamos analisar e explorar os dados, entendendo o relacionamento das variáveis preditoras de recarga e serviços e a variável alvo, retirar bons *insights* entre elas, utilizando a matriz de correlação.

Antes, vamos verificar se existe algum desequilíbrio na variável alvo venda:

```
% of normal transacation      : 0.7048119198827553
Number of normal transaction   : 11542
% of fraud transacation       : 29.518808011724474
Number of fraud transaction    : 4834
```

Figura 21 – Transações da variável alvo venda

Na Figura 21, podemos observar que existe um desequilíbrio entre as classes não venda “0” (classe sem fraude) e venda “1” (classe fraude) da variável alvo venda e que esse desbalanceamento é algo a ser trabalhado no projeto.

O método de correlação utilizado no estudo será o padrão da função `.corr()`, será o coeficiente de correlação Pearson, que é uma medida de correlação linear entre duas variáveis. Seu valor está entre -1 e +1, onde -1 indicando correlação linear negativa total, 0 indicando nenhuma correlação linear e 1 indicando correlação linear positiva total.

O Sinal de cada coeficiente indica a direção da relação, se ela é positiva ou negativa. Se ambas variáveis tendem a aumentar ou diminuir em um conjunto, o coeficiente é positivo, e a linha que representa a correlação inclina para cima. Se uma variável tende a aumentar à medida que outra diminui, o coeficiente é negativo, e a linha que representa a correlação inclina para baixo.

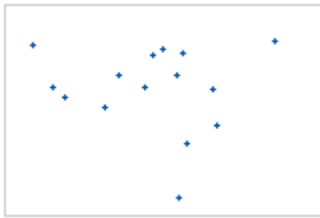


Figura 22 – Nenhum relacionamento: Pearson $r = 0$

- Pontos aleatórios no gráfico, sem correlação linear entre as variáveis na Figura 22.

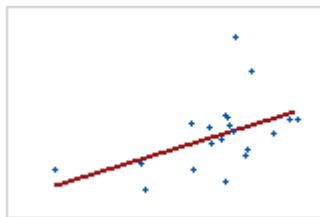


Figura 23 – Relação positiva moderada: Pearson $r = 0,455$

- Alguns pontos próximos da linha, mas outros bem dispersos dela, o que indica apenas uma relação linear moderada entre as variáveis na Figura 23.

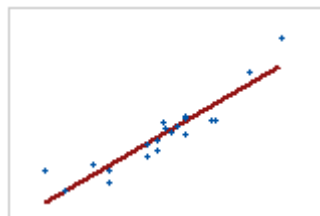


Figura 24 – Relação positiva grande: Pearson $r = 0,927$

- Os pontos estão próximos da linha, o que indica que temos uma forte correlação linear entre as variáveis e é positiva na Figura 24, pois conforme uma variável aumenta, a outra também tem o mesmo movimento.

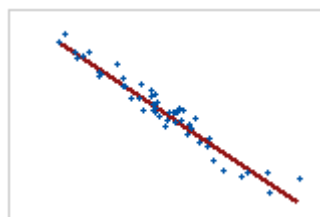


Figura 25 – Relação negativa grande: Pearson $r = 0,972$

- Os pontos estão próximo da linha, o que indica que temos uma forte relação negativa entre as variáveis na Figura 25, pois conforme uma variável aumenta, a outra diminui.

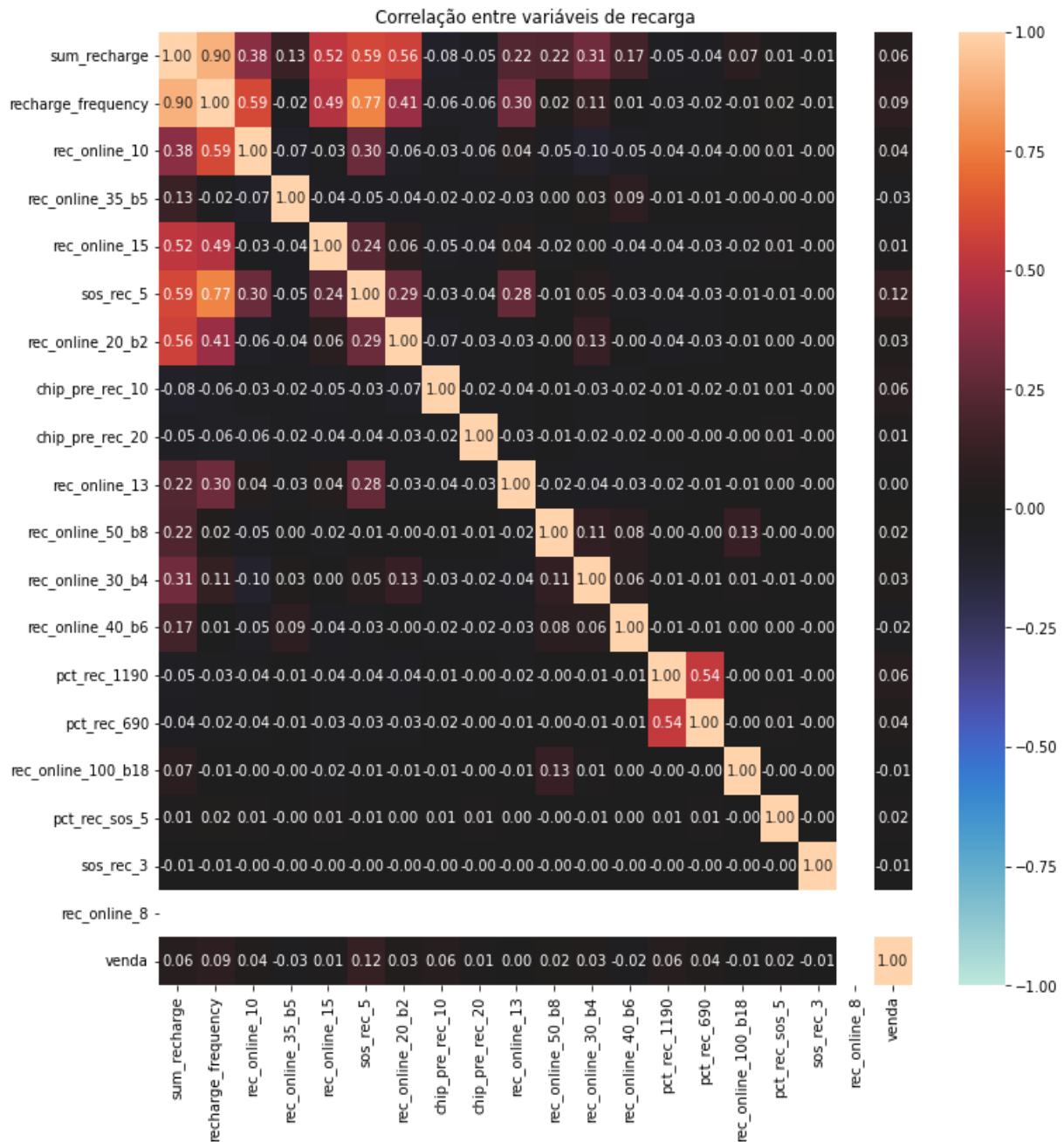


Figura 26 – Matriz de correlação linear das variáveis predictoras de recarga e variável alvo venda

Para cada recurso X do conjunto de dados, a matriz de correlação irá mostrar como esse recurso se relaciona com um recurso Y. A partir dessa visualização, saberemos quais variáveis possuem uma correlação positiva, negativa ou nula.

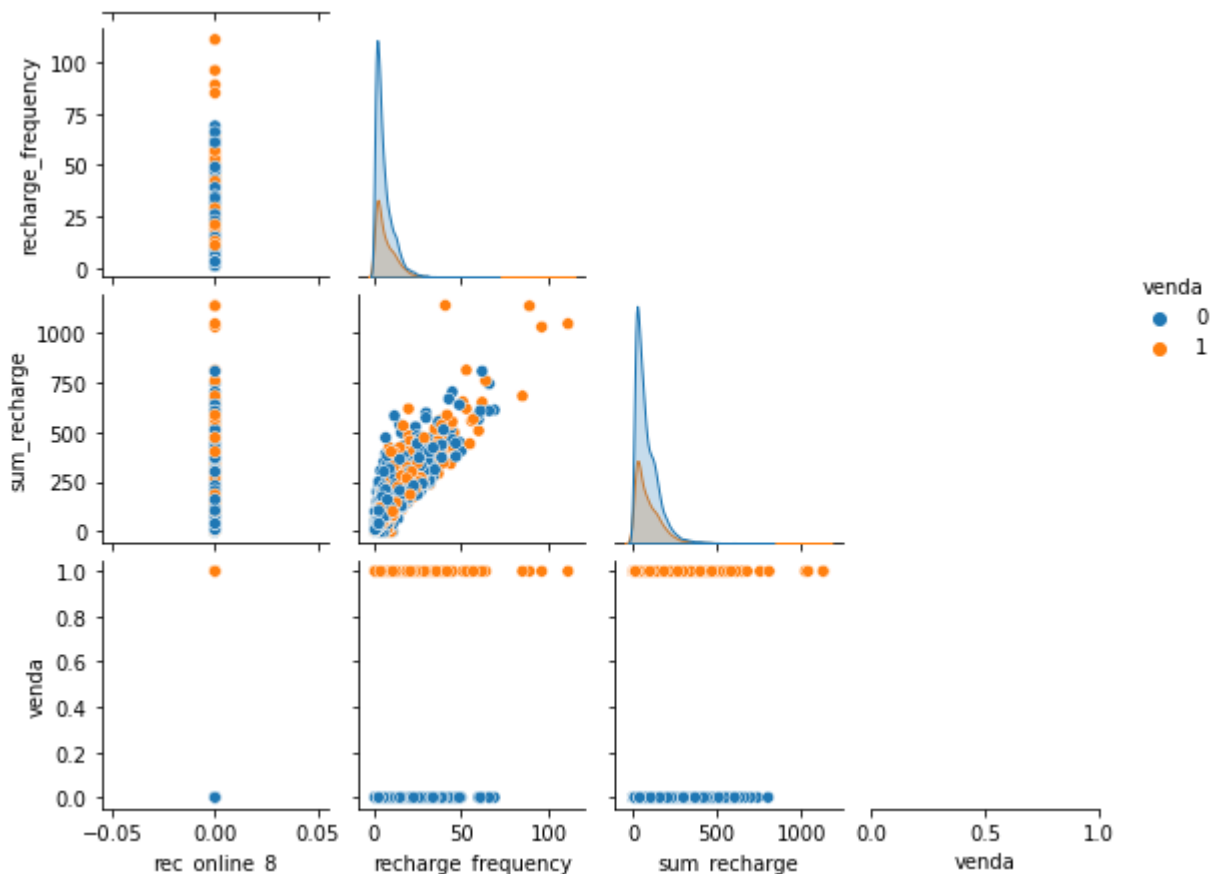


Figura 27 – Estimativas de densidades das variáveis com alguma correlação linear com a variável alvo

Nas Figuras 26 e 27, através da matriz de correlação linear entre as variáveis de recarga e o gráfico de dispersão, podemos observar que a variável **rec_online_8**, tem coeficiente inválido, pois, a variável tem valor constante zerado, 100% da composição da variável é composto do valor zero e a variável será descartada.

Quanto as correlações que podemos observar nas Figuras 26 e 27, vemos que os *Leads* que possuem uma frequência de recarga **recharge_frequency**, tem uma correlação linear forte com a soma dessas recargas **sum_recharge** e as duas variáveis tem alguma correlação linear com a variável alvo venda e as variáveis **recharge_frequency** e **sum_recharge**, ainda na Figura 27, reparando nos gráficos que trazem as densidades das variáveis, podemos observar que as variáveis não separam tão bem as classes da não venda “0”, da venda “1”.

Outra questão que é marcante nas variáveis, é a alta dimensionalidade e dispersão da classe fraudulenta venda “1” e a distribuição das variáveis são com assimetria positiva ou a direita, conforme observados nos gráficos da Figura 27.

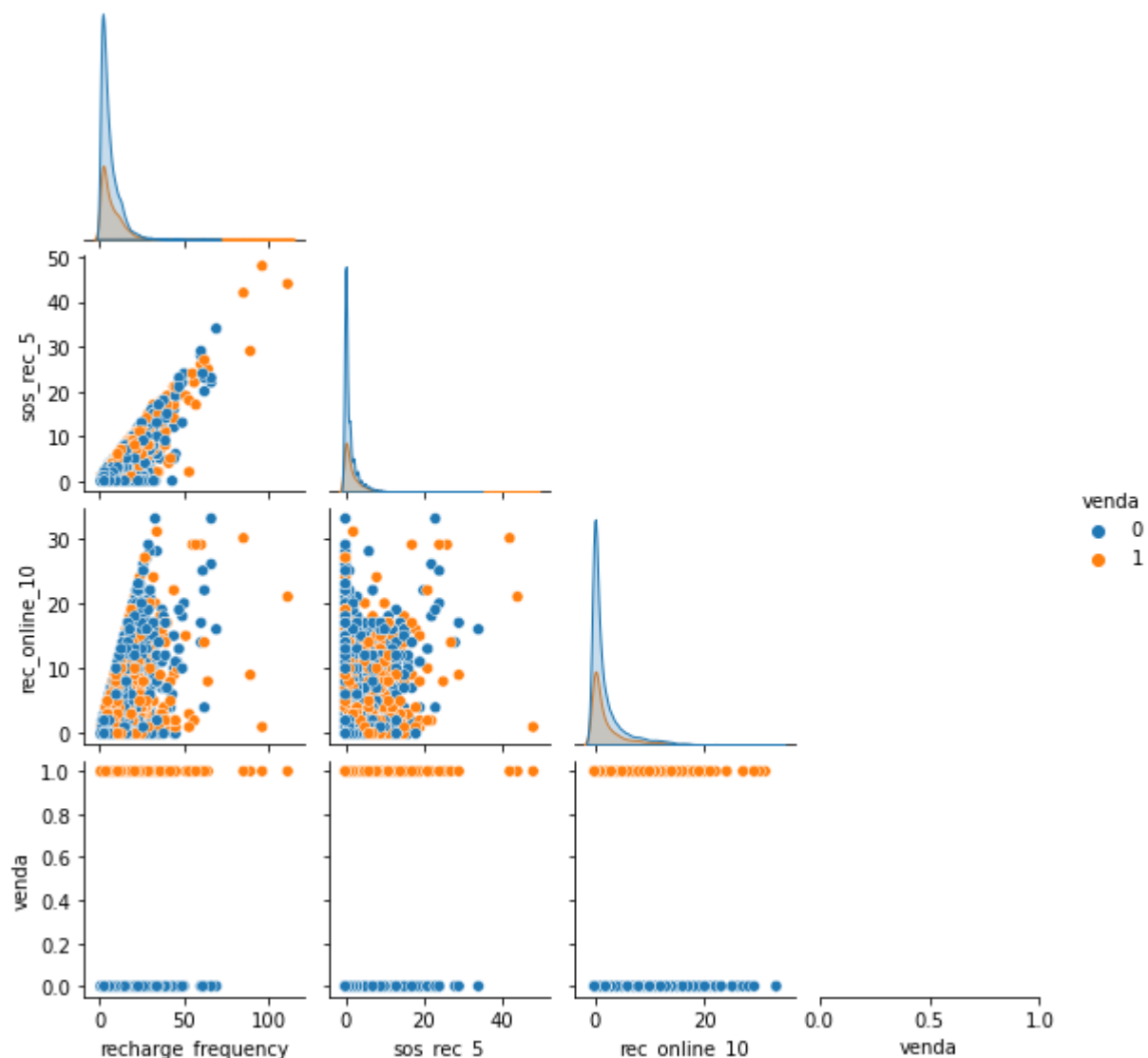


Figura 28 – Estimativas de densidades das variáveis com alguma correlação linear com a variável alvo

Ainda nas Figuras 26 e 28, podemos observar que os *Leads* que sempre realizam recargas de crédito em seu aparelho móvel, estão na maioria das vezes utilizando os pacotes de recargas **sos_rec_5** e **rec_online_10**, tendo assim alguma correlação linear com as variáveis **recharge_frequency** e **sum_recharge**, mas essa correlação linear é apresentada mais forte quando observamos a variável **sos_rec_5** e as variáveis **sos_rec_5** e **rec_online_10**, ainda na Figura 28, reparando nos gráficos que trazem as densidades das variáveis, podemos observar que as variáveis não separam tão bem as classes da não venda “0”, da venda “1”.

Outra questão que é marcante nas variáveis, é a alta dimensionalidade e dispersão da classe fraudulenta venda “1” e a distribuição das variáveis são com assimetria positiva ou a direita, conforme observados nos gráficos da Figura 28.

Os pacotes de recargas com alguma correlação com a soma de recarga **sum_recharge**, frequência de recarga **recharge_frequency** e a variável alvo **venda**, são:

rec_online_15;

Valor	Contar	Frequência (%)
0	8988	54,9%
1	3146	19,2%
2	1547	9,4%
3	931	5,7%
4	524	3,2%
5	333	2,0%
6	213	1,3%
7	174	1,1%
8	116	0,7%
9	90	0,5%
Outros valores (22)	314	1,9%

Figura 29 – Frequência de utilização do pacote rec_online_15

rec_online_20_b2;

Valor	Contar	Frequência (%)
0	9534	58,2%
1	2971	18,1%
2	1503	9,2%
3	852	5,2%
4	536	3,3%
5	312	1,9%
6	250	1,5%
7	153	0,9%
8	81	0,5%
9	52	0,3%
Outros valores (13)	132	0,8%

Figura 30 – Frequência de utilização do pacote rec_online_20_b2**rec_online_13;**

Valor	Contar	Frequência (%)
0	13874	84,7%
1	1523	9,3%
2	427	2,6%
3	232	1,4%
4	136	0,8%
5	60	0,4%
6	43	0,3%
7	27	0,2%
8	17	0,1%
9	10	0,1%
Outros valores (13)	27	0,2%

Figura 31 – Frequência de utilização do pacote rec_online_13

rec_online_30_b4;

Valor	Contar	Frequência (%)
0	14848	90,7%
1	1002	6,1%
2	284	1,7%
3	124	0,8%
4	50	0,3%
5	26	0,2%
6	18	0,1%
7	12	0,1%
8	5	< 0,1%
9	3	< 0,1%
Outros valores (3)	4	< 0,1%

Figura 32 – Frequência de utilização do pacote rec_online_30_b4

As variáveis que existem correlação com a soma de recarga **sum_recharge** devido ao alto valor monetário da recarga e perdem a frequência de recargas **recharge_frequency**, ficando com baixa correlação considerando as demais, são:

rec_online_35_b5;

Valor	Contar	Frequência (%)
0	15660	95,6%
1	395	2,4%
2	169	1,0%
3	107	0,7%
4	20	0,1%
5	15	0,1%
6	5	< 0,1%
7	3	< 0,1%
9	2	< 0,1%

Figura 33 – Frequência de utilização do pacote rec_online_35_b5

rec_online_50_b8;

Valor	Contar	Frequência (%)
0	16052	98,0%
1	196	1,2%
2	78	0,5%
3	25	0,2%
5	9	0,1%
6	5	< 0,1%
4	5	< 0,1%
7	3	< 0,1%
8	1	< 0,1%
11	1	< 0,1%

Figura 34 – Frequência de utilização do pacote rec_online_50_b8

As variáveis de recargas que podem ser descartadas ao final deste capítulo, por não ter nenhuma correlação com a variável alvo **venda**, devido ao alto custo monetário ou a frequência de utilização das recargas, são:

rec_online_40_b6;

Valor	Contar	Frequência (%)
0	15906	97,1%
1	301	1,8%
2	85	0,5%
3	54	0,3%
4	18	0,1%
6	6	< 0,1%
5	5	< 0,1%
10	1	< 0,1%

Figura 35 – Frequência de utilização do pacote rec_online_40_b6

pct_rec_sos_5;

Valor	Contar	Frequência (%)
0	16359	99,9%
1	10	0,1%
2	3	< 0,1%
3	2	< 0,1%
4	1	< 0,1%
7	1	< 0,1%

Figura 36 – Frequência de utilização do pacote pct_rec_sos_5**sos_rec_3;**

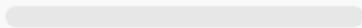
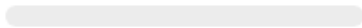
Valor	Contar	Frequência (%)
	16375	> 99,9%
	1	< 0,1%

Figura 37 – Frequência de utilização do pacote sos_rec_3**rec_online_100_b18;**

Valor	Contar	Frequência (%)
	16353	99,9%
	18	0,1%
	3	< 0,1%
	2	< 0,1%

Figura 38 – Frequência de utilização do pacote rec_online_100_b18**pct_rec_1190;**

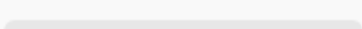

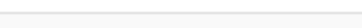
Valor	Contar	Frequência (%)
	16266	99,3%
	103	0,6%
	7	< 0,1%

Figura 39 – Frequência de utilização do pacote pct_rec_1190

chip_pre_rec_20;

Valor	Contar	Frequência (%)
	16128	98,5%
	239	1,5%
	8	< 0,1%
	1	< 0,1%

Figura 40 – Frequência de utilização do pacote chip_pre_rec_20

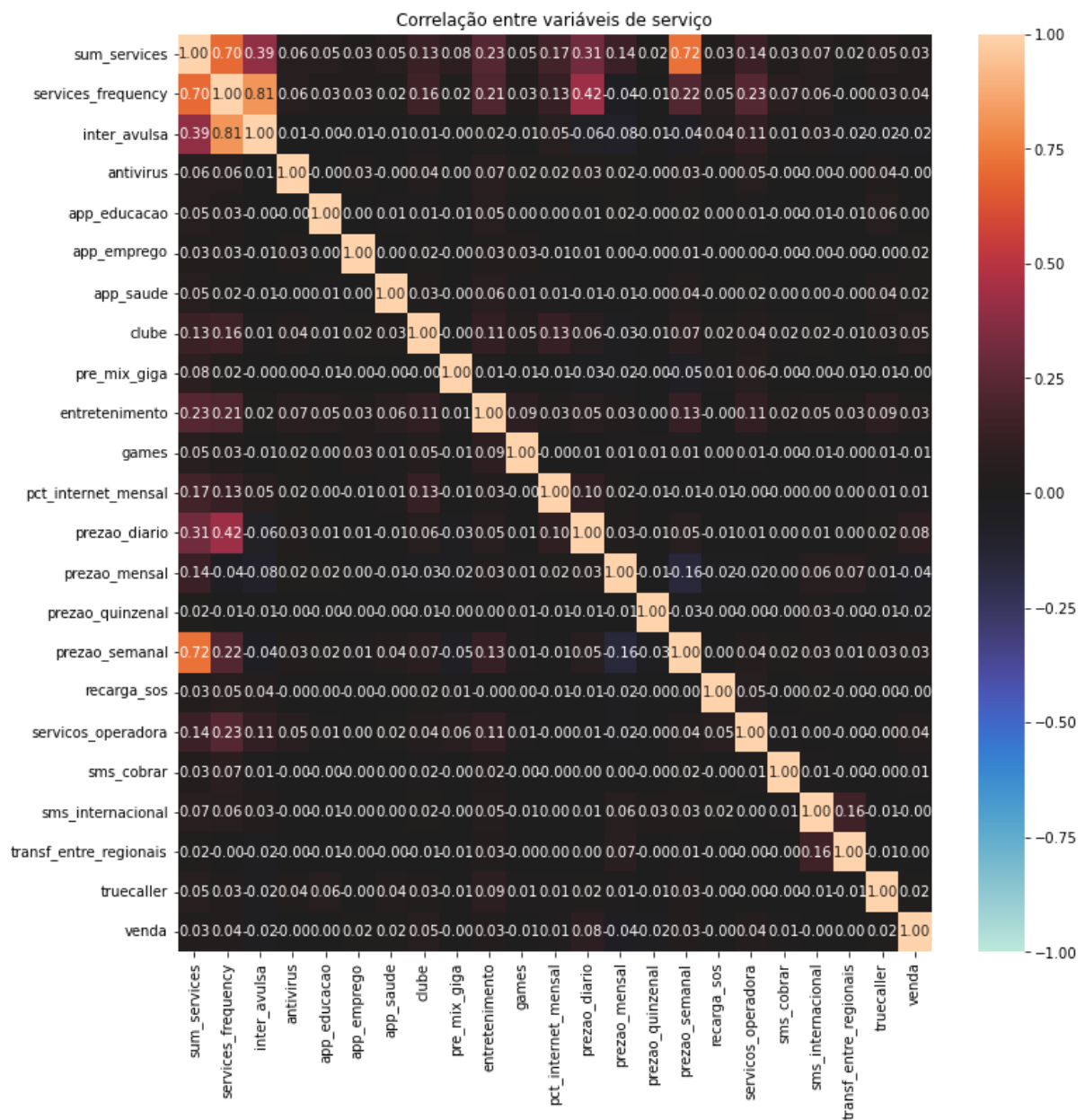


Figura 41 – Matriz de correlação linear das variáveis de serviço e variável alvo venda

Na Figura 41, através da matriz de correlação das variáveis de serviços, podemos observar que diferentemente das variáveis de recargas, em serviços não temos nenhuma variável com o coeficiente inválido.

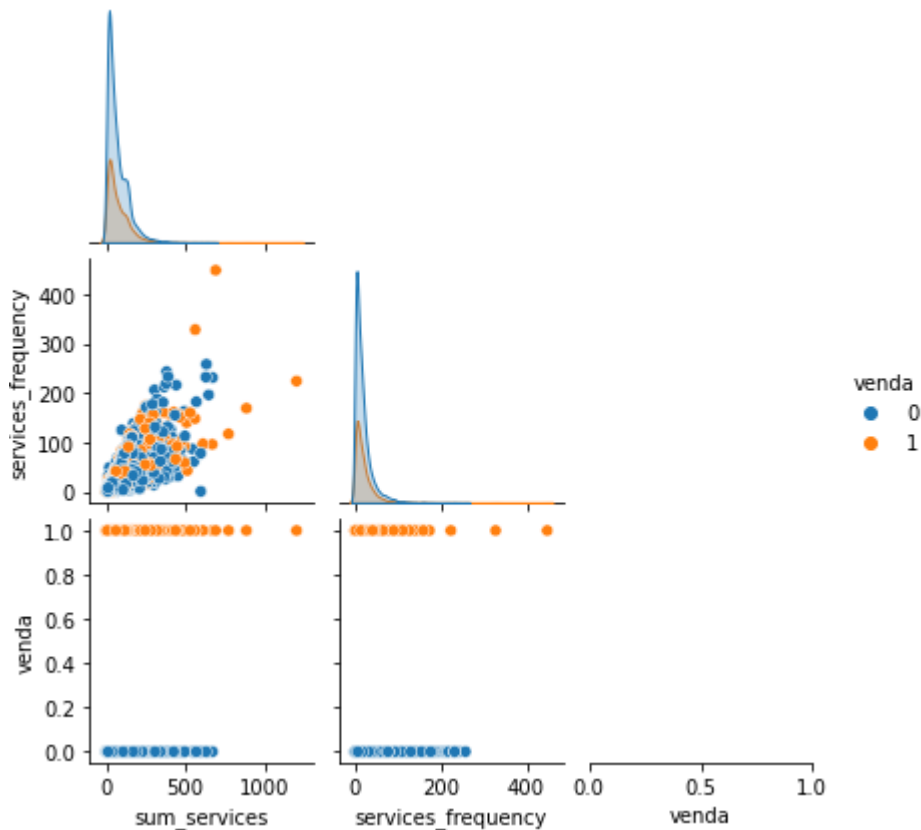


Figura 42 – Estimativas de densidades das variáveis com alguma correlação linear com a variável alvo

Conforme podemos observar, nas Figuras 41 e 42, nas recargas com a soma e frequência, temos em serviços, uma alta correlação entre as variáveis predictoras soma dos valores de serviços **sum_services**, utilizados pelo cliente com a frequência dos serviços **services_frequency**, utilizados pelo cliente e as variáveis **sum_services** e **services_frequency**, ainda na Figura 42, reparando nos gráficos que trazem as densidades das variáveis, podemos observar que as variáveis não separam tão bem as classes da não venda “0”, da venda “1”.

Outra questão que é marcante nas variáveis, é a alta dimensionalidade e dispersão da classe fraudulenta venda “1” e a distribuição das variáveis são com assimetria positiva ou a direita, conforme observados nos gráficos da Figura 42.

Na Figura 41, analisando a matriz de correlação, podemos observar também, alguma correlação linear entre as variáveis predictoras **inter_avulsa**, **prezao_semanal**, **prezao_diario**, com as predictoras **sum_services** e **services_frequency**, indicando que quem utiliza esses serviços, pagam valores mais altos ou utiliza os serviços com certa frequência e vamos analisar no decorrer do estudo, que este fator da utilização de serviços pode influenciar na realização das recargas, tanto no valor, quando na frequência.

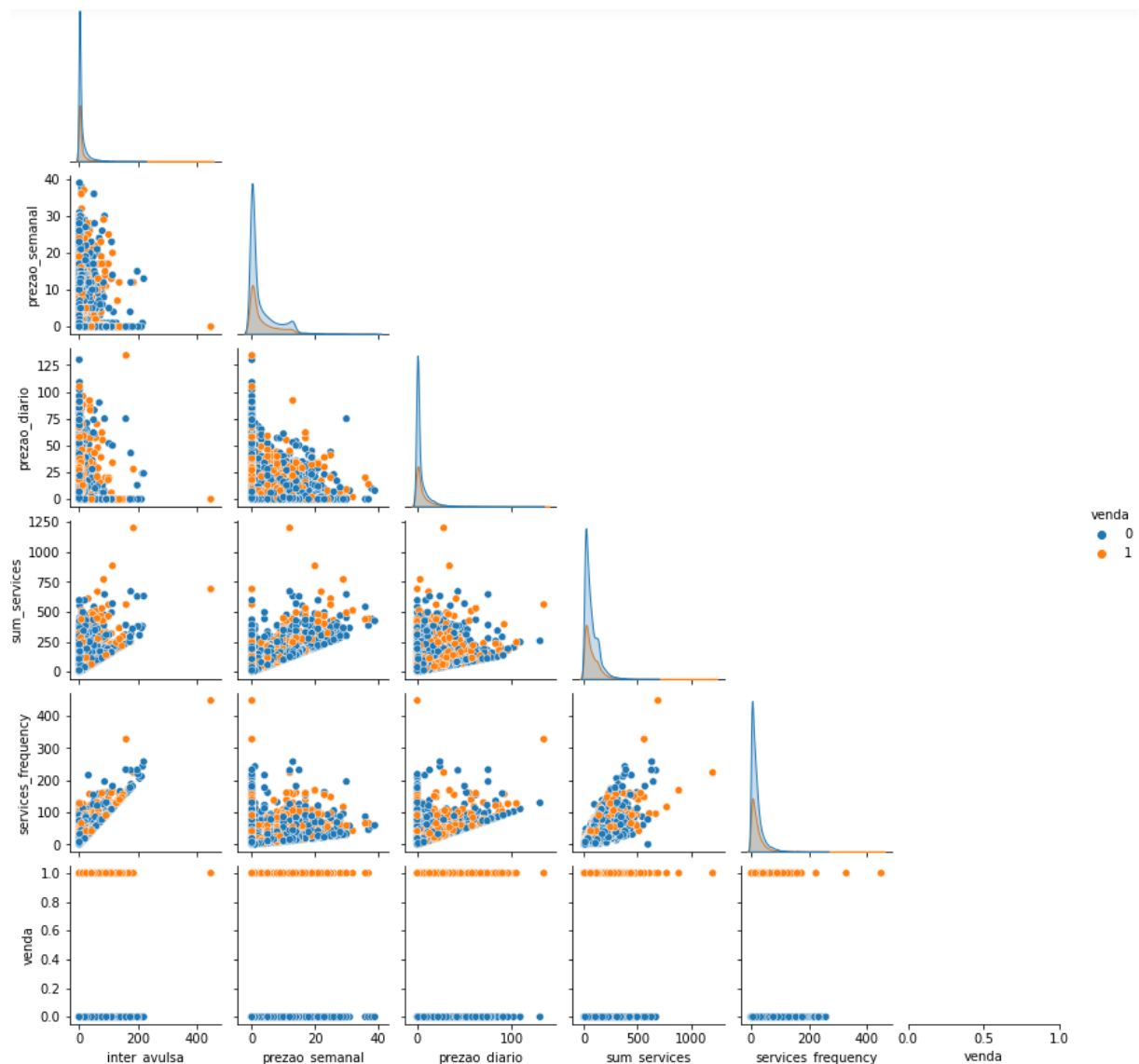


Figura 43 – Estimativas de densidades das variáveis com alguma correlação linear com a variável alvo

Na Figura 43, ao identificarmos as correlações entre as variáveis predictoras de serviços, podemos observar que temos correlações linear entre boa parte das variáveis, ainda na Figura 43, reparando nos gráficos que trazem as densidades das

variáveis, podemos observar que as variáveis não separam tão bem as classes da não venda “0”, da venda “1”.

Outra questão que é marcante nas variáveis, é a alta dimensionalidade e dispersão da classe fraudulenta venda “1” e a distribuição das variáveis são com assimetria positiva ou a direita, conforme observados nos gráficos da Figura 43.

Na Figura 41, a matriz de correlação criada para avaliação da correlação das variáveis preditoras de serviço e a variável alvo, permite observamos que as variáveis preditoras com correlação moderada com a variável alvo **venda**, são:

sum_services;

services_frequency;

clube;

Valor	Contar	Frequência (%)
0	13180	80,5%
1	2023	12,4%
2	621	3,8%
3	242	1,5%
4	95	0,6%
5	66	0,4%
6	55	0,3%
7	31	0,2%
9	16	0,1%
12	9	0,1%
Outros valores (12)	38	0,2%

Figura 44 – Frequência de utilização do serviço clube

entretenimento;

Valor	Contar	Frequência (%)
0	14570	89,0%
1	395	2,4%
2	265	1,6%
3	200	1,2%
4	170	1,0%
5	124	0,8%
6	99	0,6%
7	82	0,5%
13	76	0,5%
8	71	0,4%
Outros valores (22)	324	2,0%

Figura 45 – Frequência de utilização do serviço entretenimento**prezao_diario;**

Valor	Contar	Frequência (%)
0	8736	53,3%
1	1047	6,4%
2	816	5,0%
3	681	4,2%
4	545	3,3%
5	538	3,3%
6	429	2,6%
7	387	2,4%
8	319	1,9%
9	276	1,7%
Outros valores (88)	2602	15,9%

Figura 46 – Frequência de utilização do serviço prezao_diario

prezao_semanal;

Valor	Contar	Frequência (%)
0	7924	48,4%
1	1545	9,4%
2	1121	6,8%
3	854	5,2%
4	684	4,2%
5	589	3,6%
13	503	3,1%
6	483	2,9%
7	415	2,5%
12	402	2,5%
Outros valores (27)	1856	11,3%

Figura 47 – Frequência de utilização do serviço prezao_semanal**servicos_operadora;**

Valor	Contar	Frequência (%)
0	12747	77,8%
1	2298	14,0%
2	419	2,6%
3	174	1,1%
4	111	0,7%
5	72	0,4%
8	66	0,4%
12	63	0,4%
13	61	0,4%
7	61	0,4%
Outros valores (22)	304	1,9%

Figura 48 – Frequência de utilização do serviço servicos_operadora

prezao_mensal;

Valor	Contar	Frequência (%)
0	14370	87,8%
1	813	5,0%
2	555	3,4%
3	471	2,9%
4	90	0,5%
5	41	0,3%
6	20	0,1%
7	7	< 0,1%
8	4	< 0,1%
9	4	< 0,1%

Figura 49 – Frequência de utilização do serviço prezao_mensal

As variáveis de serviços que podem ser descartadas ao final deste capítulo, por não ter nenhuma correlação com a variável alvo **venda**, devido à baixa frequência de utilização dos serviços, são:

antivirus;

Valor	Contar	Frequência (%)
0	16290	99,5%
1	36	0,2%
2	14	0,1%
3	11	0,1%
5	9	0,1%
4	7	< 0,1%
6	3	< 0,1%
12	2	< 0,1%
21	1	< 0,1%
8	1	< 0,1%
Outros valores (2)	2	< 0,1%

Figura 50 – Frequência de utilização do serviço *antivirus*

app_educacao;

Valor	Contar	Frequência (%)
0	16256	99,3%
1	47	0,3%
2	28	0,2%
3	12	0,1%
4	12	0,1%
6	8	< 0,1%
5	8	< 0,1%
7	3	< 0,1%
10	1	< 0,1%
8	1	< 0,1%

Figura 51 – Frequência de utilização do serviço app_educacao**app_emprego;**

Valor	Contar	Frequência (%)
0	16303	99,6%
1	28	0,2%
2	10	0,1%
3	9	0,1%
4	7	< 0,1%
7	4	< 0,1%
5	4	< 0,1%
6	3	< 0,1%
13	2	< 0,1%
10	2	< 0,1%
Outros valores (4)	4	< 0,1%

Figura 52 – Frequência de utilização do serviço app_emprego

app_saude;

Valor	Contar	Frequência (%)
0	16314	99,6%
1	28	0,2%
2	12	0,1%
3	5	< 0,1%
4	5	< 0,1%
8	3	< 0,1%
6	2	< 0,1%
5	2	< 0,1%
9	2	< 0,1%
13	1	< 0,1%
Outros valores (2)	2	< 0,1%

Figura 53 – Frequência de utilização do serviço app_saude**pre_mix_giga;**

Valor	Contar	Frequência (%)
0	16240	99,2%
13	19	0,1%
12	12	0,1%
10	11	0,1%
6	11	0,1%
5	11	0,1%
11	11	0,1%
4	10	0,1%
2	10	0,1%
7	8	< 0,1%
Outros valores (9)	33	0,2%

Figura 54 – Frequência de utilização do serviço pre_mix_giga

games;

Valor	Contar	Frequência (%)
0	16137	98,5%
1	120	0,7%
2	46	0,3%
3	30	0,2%
4	14	0,1%
5	8	< 0,1%
6	5	< 0,1%
8	5	< 0,1%
10	3	< 0,1%
11	3	< 0,1%
Outros valores (3)	5	< 0,1%

Figura 55 – Frequência de utilização do serviço *games***prezao_quinzenal;**

Valor	Contar	Frequência (%)
0	16295	99,5%
3	17	0,1%
6	15	0,1%
1	12	0,1%
5	10	0,1%
7	9	0,1%
4	9	0,1%
2	5	< 0,1%
8	2	< 0,1%
13	1	< 0,1%

Figura 56 – Frequência de utilização do serviço *prezao_quinzenal*

recarga_sos;

Valor	Contar	Frequência (%)
0	16294	99,5%
1	34	0,2%
2	27	0,2%
4	6	< 0,1%
3	6	< 0,1%
6	4	< 0,1%
5	2	< 0,1%
7	2	< 0,1%
8	1	< 0,1%

Figura 57 – Frequência de utilização do serviço recarga_sos

sms_cobrar;

Valor	Contar	Frequência (%)
0	16185	98,8%
1	112	0,7%
2	25	0,2%
3	11	0,1%
4	9	0,1%
6	9	0,1%
5	5	< 0,1%
8	4	< 0,1%
7	3	< 0,1%
79	2	< 0,1%
Outros valores (9)	11	0,1%

Figura 58 – Frequência de utilização do serviço sms_cobrar

sms_internacional;

Valor	Contar	Frequência (%)
0	16109	98,4%
1	180	1,1%
2	55	0,3%
3	13	0,1%
4	9	0,1%
5	4	< 0,1%
6	2	< 0,1%
13	1	< 0,1%
7	1	< 0,1%
8	1	< 0,1%

Figura 59 – Frequência de utilização do serviço sms_internacional

transf_entre_regionais;

Valor	Contar	Frequência (%)
	16278	99,4%
	97	0,6%
	1	< 0,1%

Figura 60 – Frequência de utilização do serviço transf_entre_regionais

truecaller;

Valor	Contar	Frequência (%)
0	16174	98,8%
1	105	0,6%
2	37	0,2%
3	22	0,1%
4	17	0,1%
5	9	0,1%
6	5	< 0,1%
7	4	< 0,1%
9	2	< 0,1%
8	1	< 0,1%

Figura 61 – Frequência de utilização do serviço *truecaller*

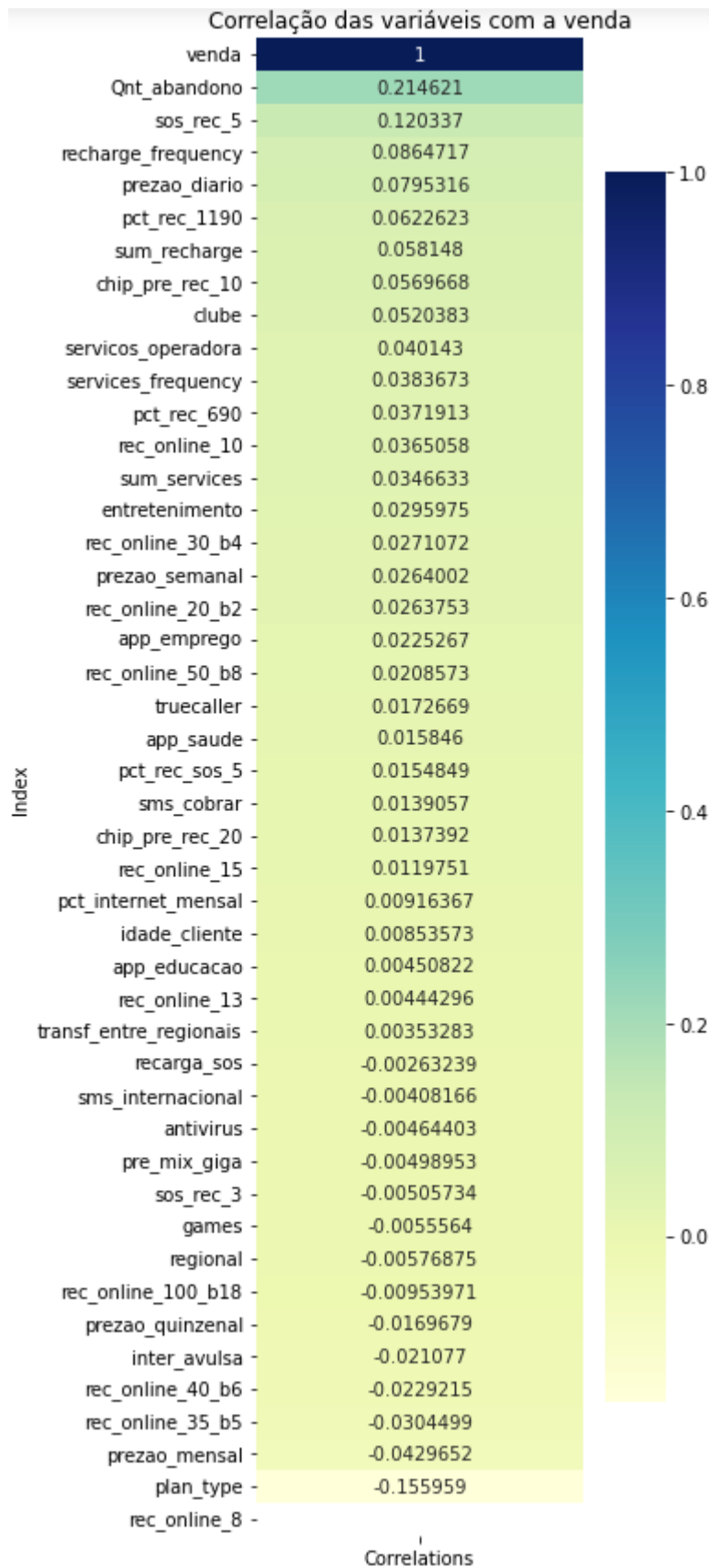


Figura 62 – Correlação de todas as variáveis preditoras com a variável alvo venda

Na detecção de valores discrepantes não variáveis de recargas e serviços, vamos utilizar estatística simples para avaliar e realizar o tratamento de *outliers*.

	count	mean	std	min	25%	50%	75%	max
sum_recharge	16376.0	82.675696	77.137051	0.0	30.0	60.0	118.0	1133.0
recharge_frequency	16376.0	6.333842	6.238281	1.0	2.0	4.0	9.0	111.0
rec_online_10	16376.0	1.913166	3.299767	0.0	0.0	1.0	2.0	33.0
rec_online_35_b5	16376.0	0.078041	0.436381	0.0	0.0	0.0	0.0	9.0
rec_online_15	16376.0	1.295249	2.456976	0.0	0.0	0.0	2.0	36.0
sos_rec_5	16376.0	1.305264	2.427992	0.0	0.0	0.0	2.0	48.0
rec_online_20_b2	16376.0	1.071385	1.933577	0.0	0.0	0.0	1.0	29.0
chip_pre_rec_10	16376.0	0.040730	0.197671	0.0	0.0	0.0	0.0	1.0
chip_pre_rec_20	16376.0	0.015938	0.135993	0.0	0.0	0.0	0.0	6.0
rec_online_13	16376.0	0.305997	1.109285	0.0	0.0	0.0	0.0	42.0
rec_online_50_b8	16376.0	0.034929	0.315505	0.0	0.0	0.0	0.0	11.0
rec_online_30_b4	16376.0	0.157181	0.633103	0.0	0.0	0.0	0.0	12.0
rec_online_40_b6	16376.0	0.047386	0.333314	0.0	0.0	0.0	0.0	10.0
pct_rec_1190	16376.0	0.007145	0.089157	0.0	0.0	0.0	0.0	2.0
pct_rec_690	16376.0	0.012274	0.164393	0.0	0.0	0.0	0.0	6.0
rec_online_100_b18	16376.0	0.001832	0.054110	0.0	0.0	0.0	0.0	3.0
pct_rec_sos_5	16376.0	0.002015	0.080051	0.0	0.0	0.0	0.0	7.0
sos_rec_3	16376.0	0.000061	0.007814	0.0	0.0	0.0	0.0	1.0
rec_online_8	16376.0	0.000000	0.000000	0.0	0.0	0.0	0.0	0.0

Figura 63 – *Describe* variáveis de recargas

Na Figura 63, podemos observar que boa parte das variáveis de recargas estão o desvio padrão muito baixo e isso pode afetar a distribuição e fazer com que o *algoritmo* fique enviesado para variáveis com maior ordem de grandeza.

Vamos trazer o *histograma* das variáveis de recargas para termos a distribuição e analisarmos qual tipo de distribuição temos.

- Histograma para visualização da distribuição das variáveis de recargas

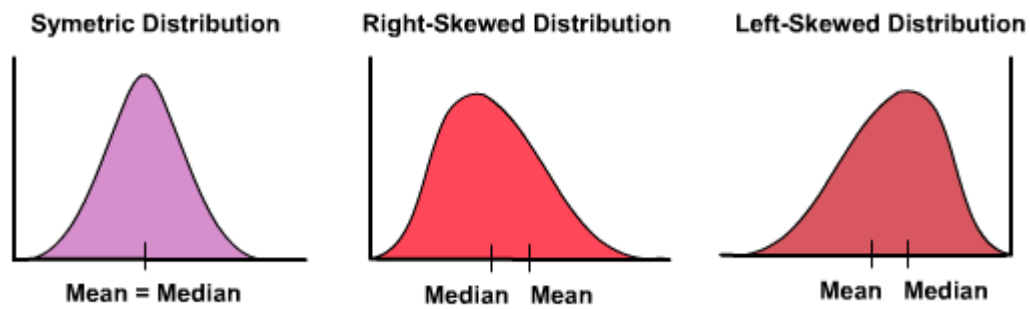


Gráfico de tipos de distribuição

Figura 64 – Tipos de distribuição

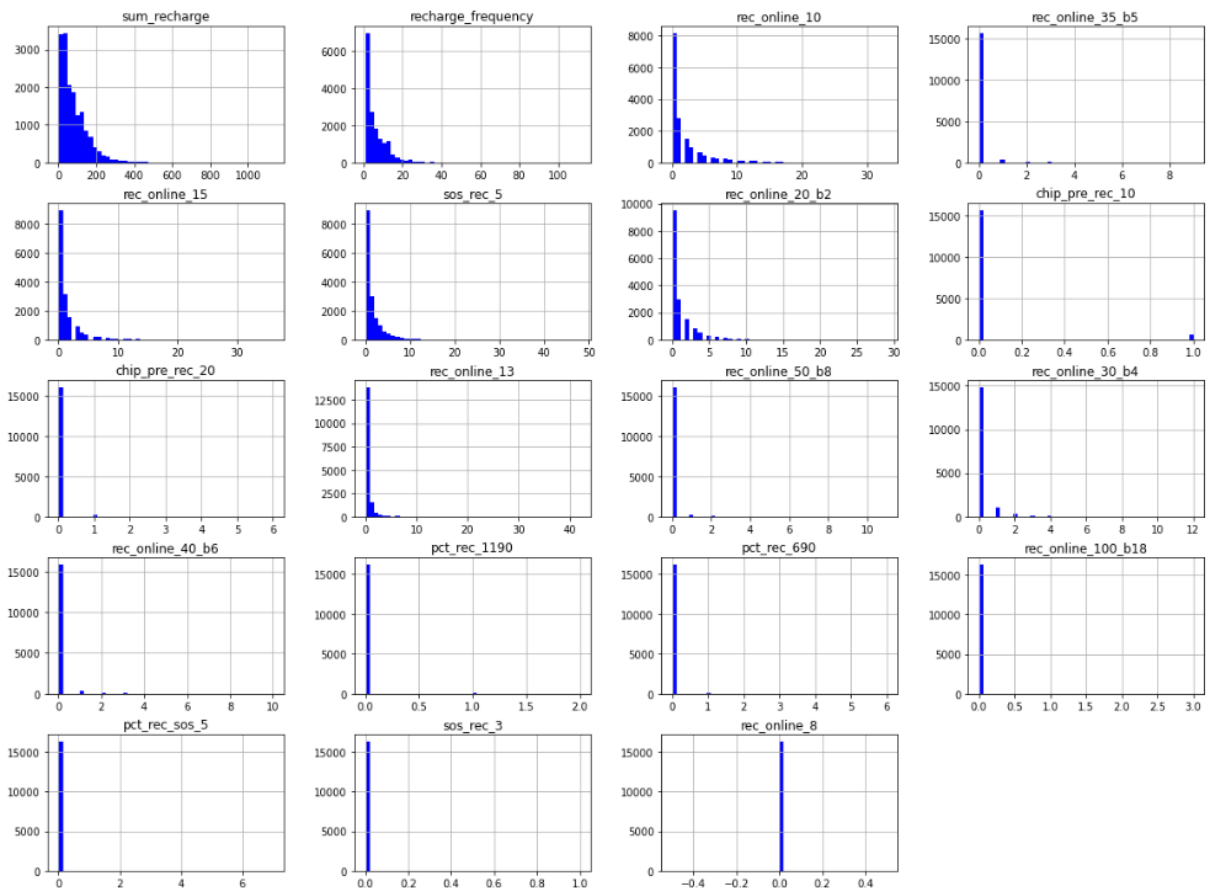


Figura 65 – Distribuição das variáveis de recargas

Nas Figuras 63 e 65, podemos observar que as variáveis **sum_recharge**, **recharge_frequency**, **rec_online_10**, **rec_online_15**, **sos_rec_5**, **rec_online_20_b2** e **rec_online_13**, estão com valores discrepantes, destoando das suas medidas centrais e a distribuição não é normal, tendo assim uma distribuição assimétrica à direita devido ao alto número de valores zerados ou próximo a zero.

As demais variáveis, como vimos anteriormente, estão com alto índice de valores zerados e coeficiente nulo ou quase nulo.

Vamos analisar a presença de outliers nos dados através do *Boxplot*, para termos uma dimensão dos valores discrepantes nas variáveis de recargas.

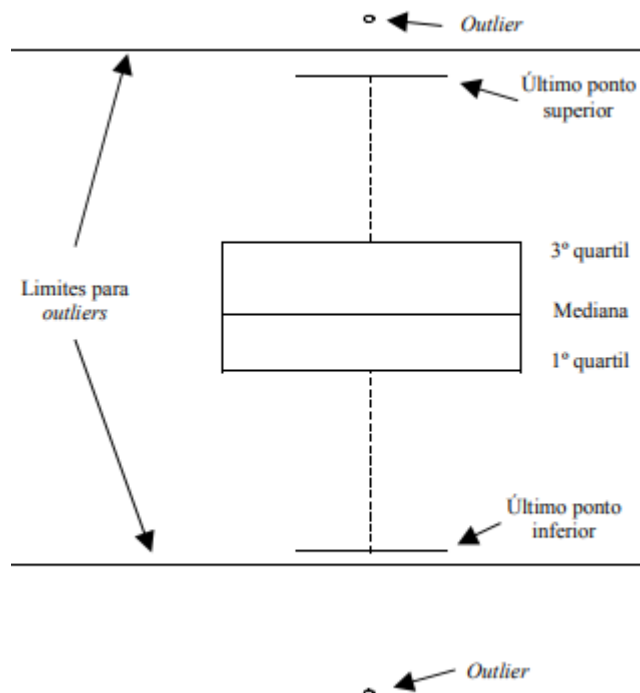


Figura 66 – Legenda *Boxplot*

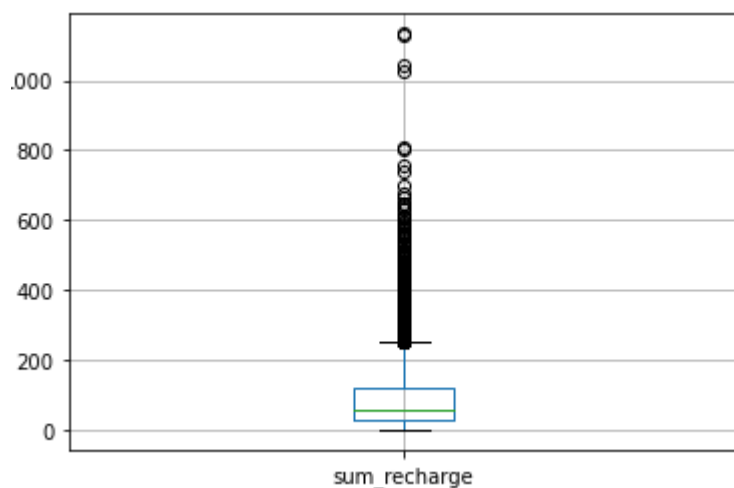


Figura 67 – *Boxplot* *sum_recharge*

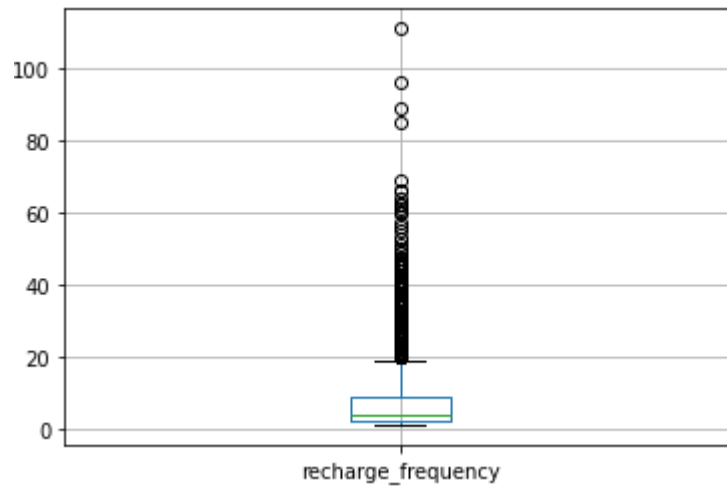


Figura 68 – *Boxplot* recharge_frequency

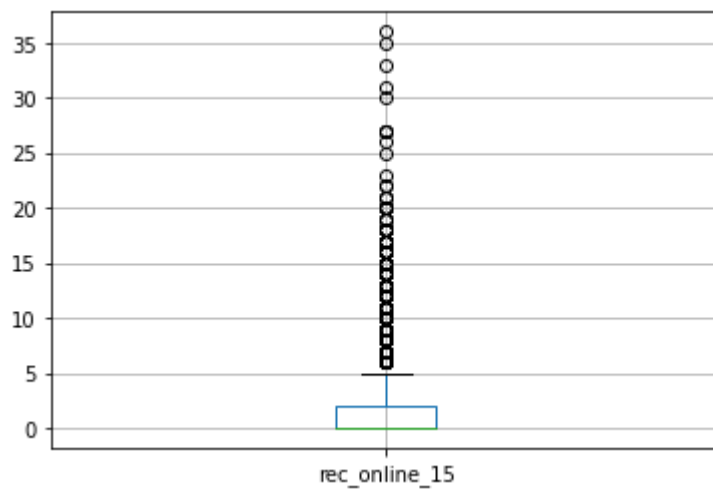


Figura 69 – *Boxplot* rec_online_15

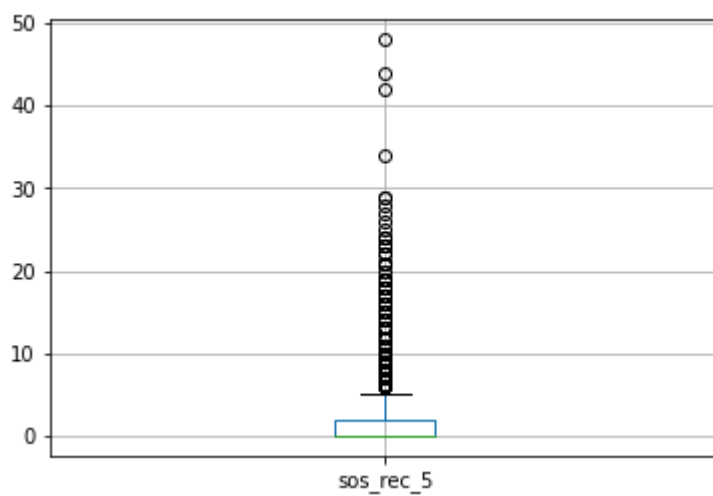


Figura 70 – *Boxplot* sos_rec_5

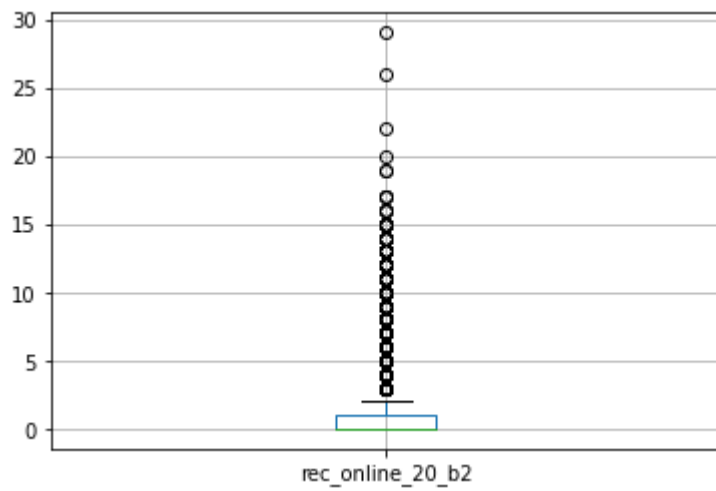


Figura 71 – *Boxplot* `rec_online_20_b2`

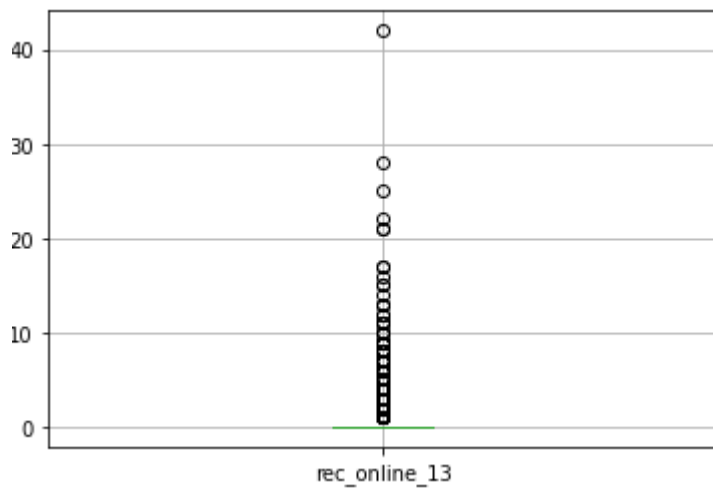


Figura 72 – *Boxplot* `rec_online_13`

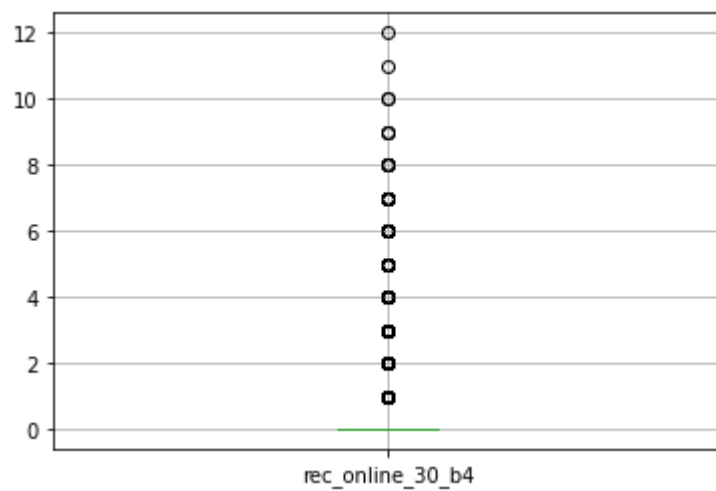


Figura 73 – *Boxplot* `rec_online_30_b4`

Para as variáveis de serviços, vamos realizar o mesmo procedimento realizado nas variáveis de recargas. Veremos a descrição das estatísticas simples das variáveis de serviços e definir tratativas para os *outliers*.

	count	mean	std	min	25%	50%	75%	max
sum_services	16376.0	68.549771	68.741957	0.01	19.98	46.18	100.1125	1202.26
services_frequency	16376.0	18.651380	21.593588	1.00	5.00	12.00	24.0000	448.00
inter_avulsa	16376.0	8.404433	18.052572	0.00	0.00	1.00	8.0000	447.00
antivirus	16376.0	0.016182	0.323597	0.00	0.00	0.00	0.0000	21.00
app_educacao	16376.0	0.019174	0.276508	0.00	0.00	0.00	0.0000	10.00
app_emprego	16376.0	0.015816	0.324181	0.00	0.00	0.00	0.0000	16.00
app_saude	16376.0	0.011419	0.260225	0.00	0.00	0.00	0.0000	13.00
clube	16376.0	0.376099	1.715855	0.00	0.00	0.00	0.0000	155.00
pre_mix_giga	16376.0	0.070103	0.908972	0.00	0.00	0.00	0.0000	43.00
entretenimento	16376.0	0.595078	2.344888	0.00	0.00	0.00	0.0000	40.00
games	16376.0	0.035296	0.400427	0.00	0.00	0.00	0.0000	15.00
pct_internet_mensal	16376.0	0.094529	0.541283	0.00	0.00	0.00	0.0000	15.00
prezao_diario	16376.0	4.655532	9.634188	0.00	0.00	0.00	5.0000	134.00
prezao_mensal	16376.0	0.253786	0.806435	0.00	0.00	0.00	0.0000	18.00
prezao_quinzenal	16376.0	0.021678	0.353418	0.00	0.00	0.00	0.0000	14.00
prezao_semanal	16376.0	3.090620	4.614504	0.00	0.00	1.00	5.0000	39.00
recarga_sos	16376.0	0.011358	0.198604	0.00	0.00	0.00	0.0000	8.00
servicos_operadora	16376.0	0.651136	2.245891	0.00	0.00	0.00	0.0000	36.00
sms_cobrar	16376.0	0.046837	1.122513	0.00	0.00	0.00	0.0000	79.00
sms_internacional	16376.0	0.026808	0.281678	0.00	0.00	0.00	0.0000	14.00
transf_entre_regionais	16376.0	0.006168	0.082842	0.00	0.00	0.00	0.0000	4.00
truecaller	16376.0	0.026991	0.306477	0.00	0.00	0.00	0.0000	9.00

Figura 74 – *Describe* variáveis de serviços

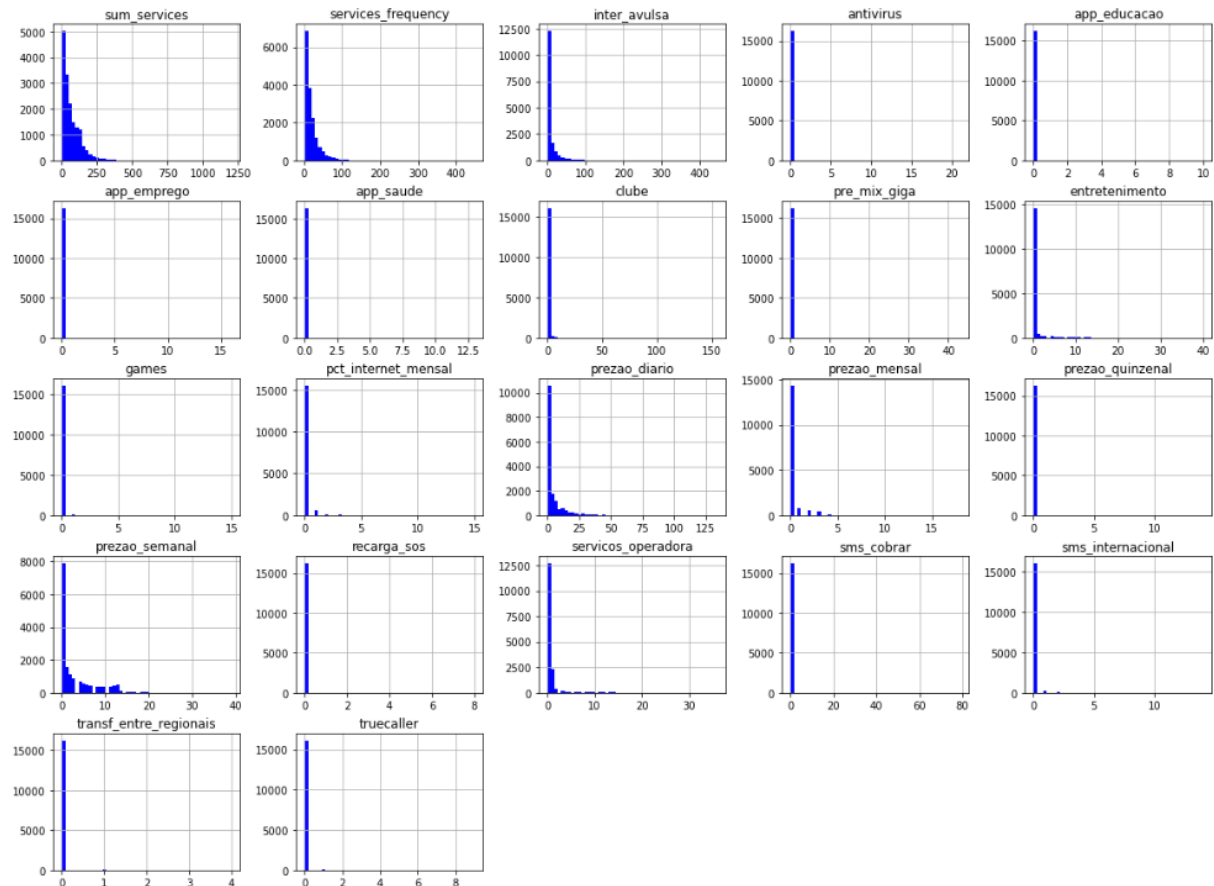


Figura 75 – Distribuição das variáveis de serviços

Nas figuras 74 e 75, podemos observar que as variáveis **sum_services**, **services_frequency**, **inter_avulsa**, **prezao_diario**, **prezao_mensal**, **prezao_semanal**, **serviços_operadora** e **clube**, estão com valores discrepantes, destoando das suas medidas centrais e a distribuição não é normal, tendo assim uma distribuição assimétrica à direita devido ao alto número de valores zerados ou próximo a zero. As demais variáveis, como vimos anteriormente, estão com alto índice de valores zerados e coeficiente nulo ou quase nulo.

Vamos analisar a presença de *outliers* nos dados através do *Boxplot*, para termos uma dimensão dos valores discrepantes nas variáveis de serviços.

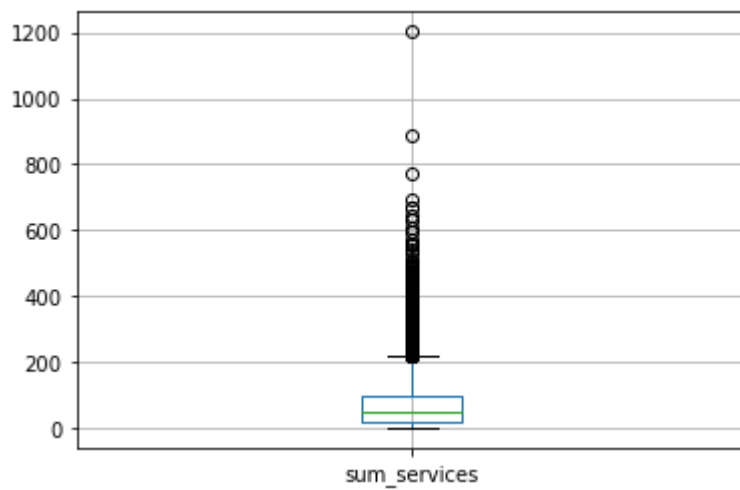


Figura 76 – *Boxplot* sum_services

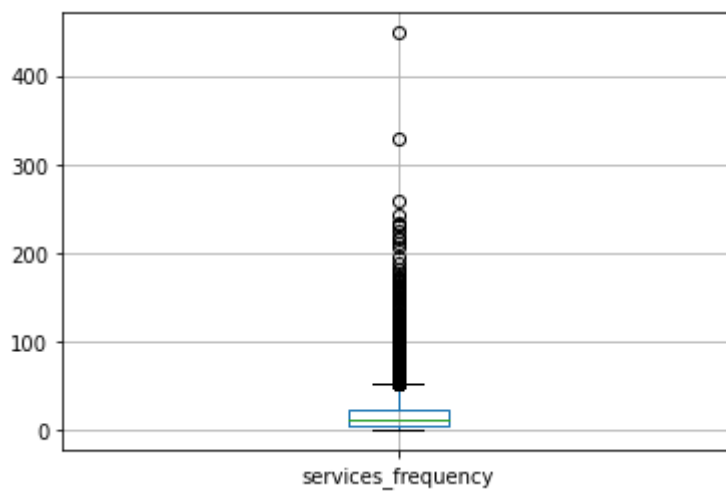


Figura 77 – *Boxplot* services_frequency

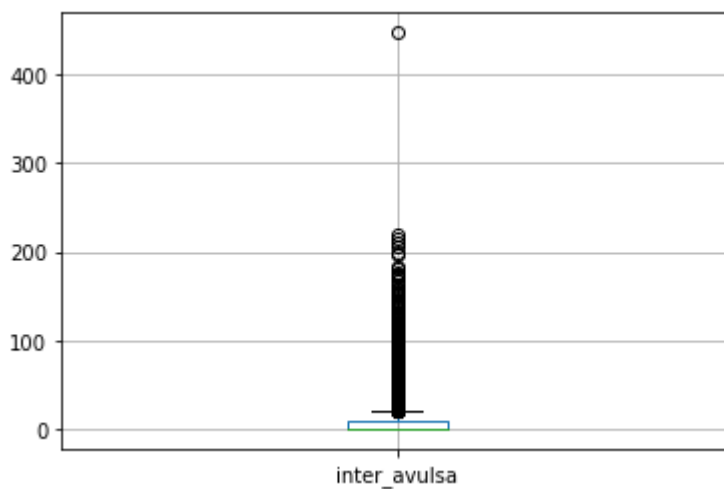


Figura 78 – *Boxplot* inter_avulsa

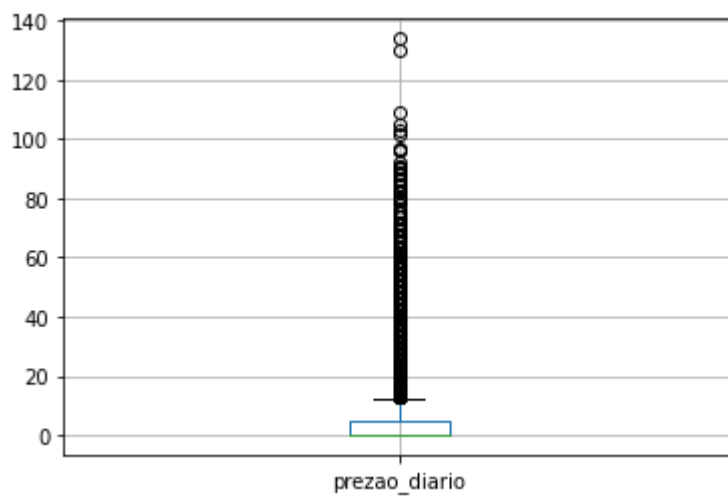


Figura 79 – *Boxplot* prezao_diario

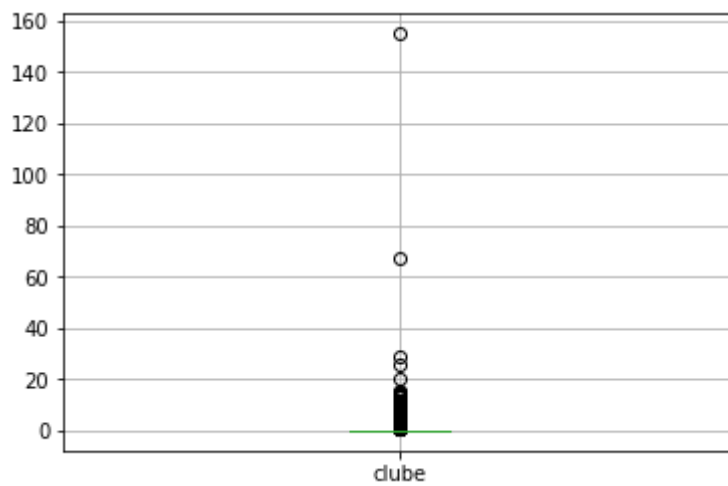


Figura 80 – *Boxplot* clube

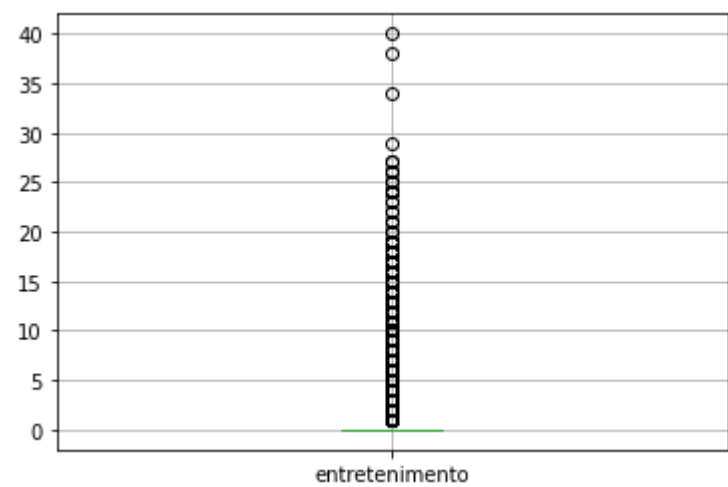


Figura 81 – *Boxplot* entretenimento

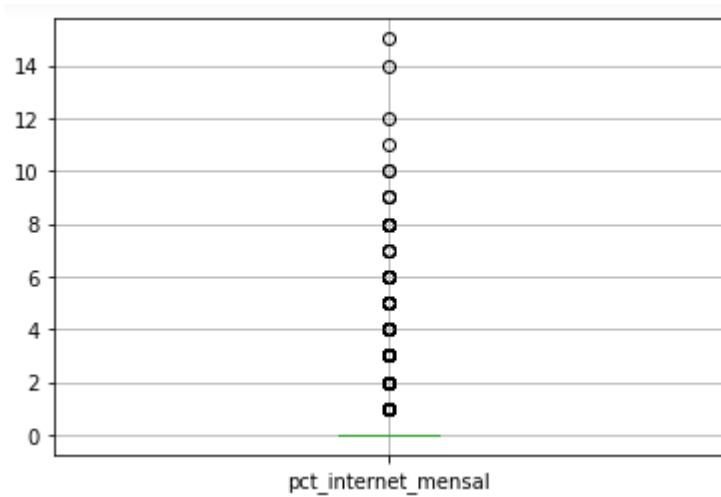


Figura 82 – *Boxplot* pct_internet_mensal

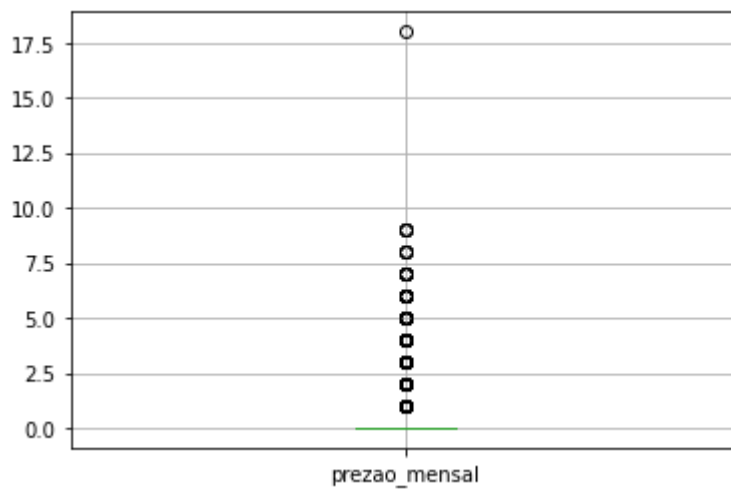


Figura 83 – *Boxplot* prezao_mensal

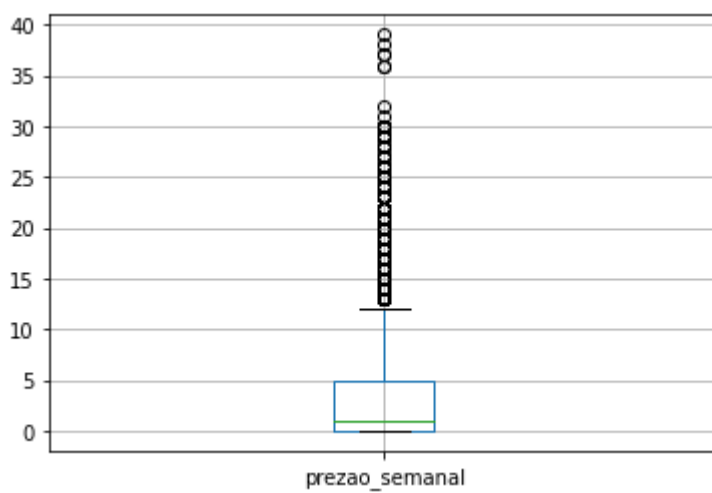


Figura 84 – *Boxplot* prezao_semanal

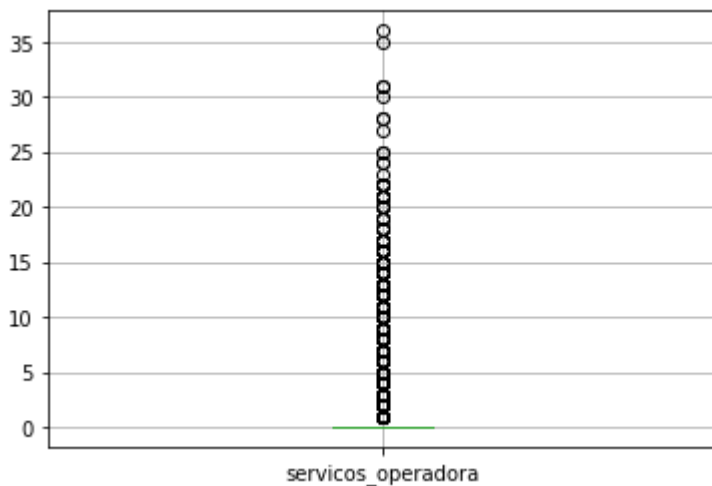


Figura 85 – Boxplot servicos_operadora

Como podemos observar nas Figuras relacionadas ao *describes* e *boxplots*, não temos um *dataframe* de recargas e serviços com dados normalmente distribuídos. Para tratar as faixas de valores diferentes e reduzir a influência dos pesos dos coeficientes, vamos trabalhar melhor os atributos, realizando a normalização *Min-Max* dos dados das variáveis de recargas e serviços.

O estimador *MinMaxScaler*, dimensiona e traduz cada variável individualmente de tal forma que esteja no intervalo dado no conjunto de treinamento, no exemplo que vamos utilizar, utilizaremos o *rechape* entre -1 e 1.

A transformação é dada por:

```
X_std = (X - X.min(axis=0)) / (X.max(axis=0) - X.min(axis=0))
X_scaled = X_std * (max - min) + min
```

Figura 86 – MinMaxScaler

Onde *min*, *max* = *feature_range*

Essa transformação é frequentemente utilizada como alternativa ao escalonamento de média zero e variância unitária, o que vemos muito nas estáticas simples das variáveis de recargas e serviços.

```

# Retirando a alta dimensionalidade
# Normalização Min-Max dos dados
cols = ['regional', 'idade_cliente', 'plan_type', 'Qnt_abandono',
        'sum_recharge', 'recharge_frequency', 'rec_online_10',
        'rec_online_35_b5', 'rec_online_15', 'sos_rec_5', 'rec_online_20_b2',
        'chip_pre_rec_10', 'chip_pre_rec_20', 'rec_online_13',
        'rec_online_50_b8', 'rec_online_30_b4', 'rec_online_40_b6',
        'pct_rec_1190', 'pct_rec_690', 'rec_online_100_b18', 'pct_rec_sos_5',
        'sos_rec_3', 'rec_online_8', 'sum_services', 'services_frequency',
        'inter_avulsa', 'antivirus', 'app_educacao', 'app_emprego', 'app_saude',
        'clube', 'pre_mix_giga', 'entretenimento', 'games',
        'pct_internet_mensal', 'prezao_diario', 'prezao_mensal',
        'prezao_quinzenal', 'prezao_semanal', 'recarga_sos',
        'servicos_operadora', 'sms_cobrar', 'sms_internacional',
        'transf_entre_regionais', 'truecaller']
for col in cols:
    # Ajustar aos dados e transformá-los.
    data[col] = MinMaxScaler().fit_transform(data[col].values.reshape(-1,1))

```

Figura 87 – Dimensionamento *MinMaxScaler*

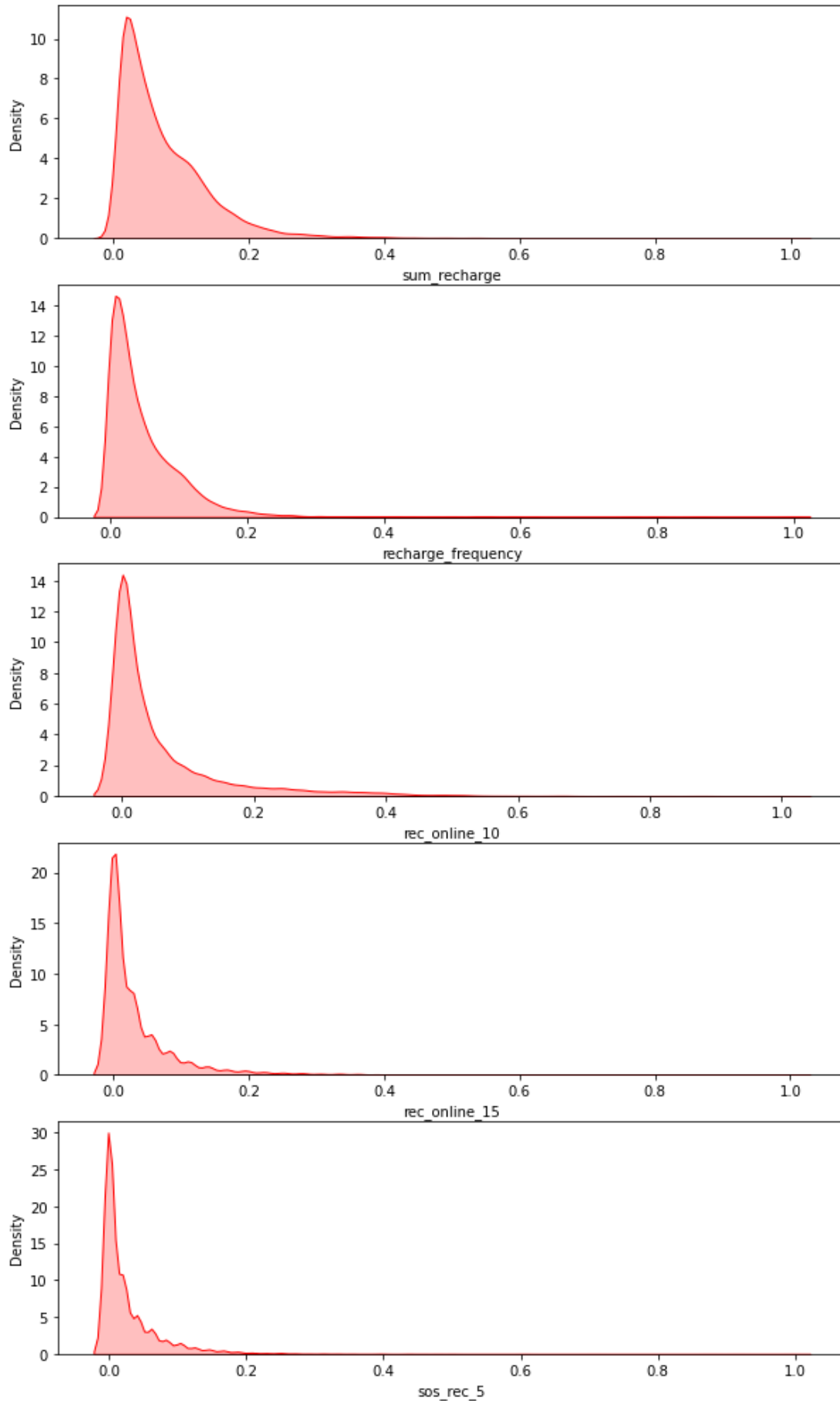


Figura 88 – Distribuição das variáveis de recargas com redução de dimensionalidade *MinMaxScaler*

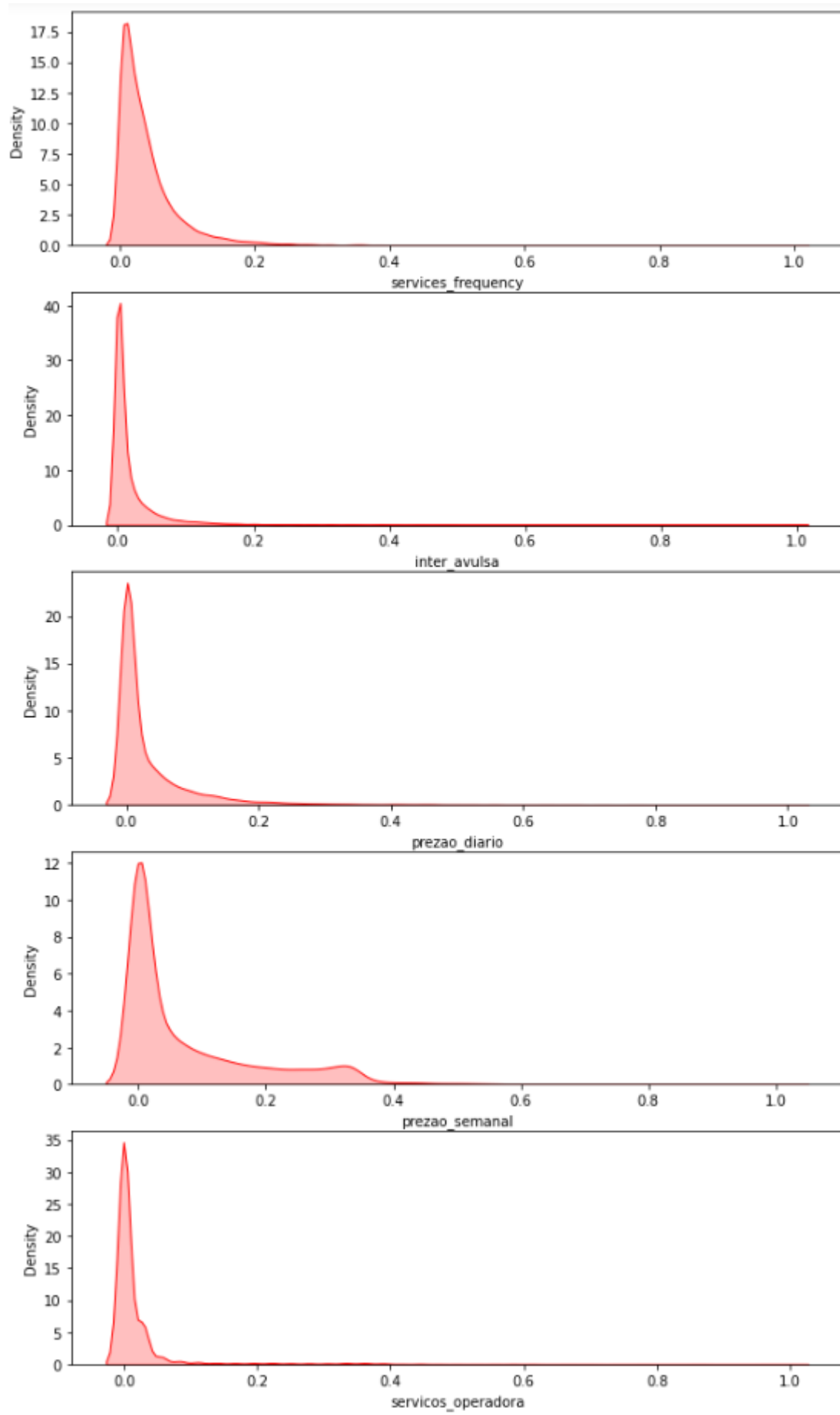


Figura 89 – Distribuição das variáveis de serviços com redução de dimensionalidade *MinMaxScaler*

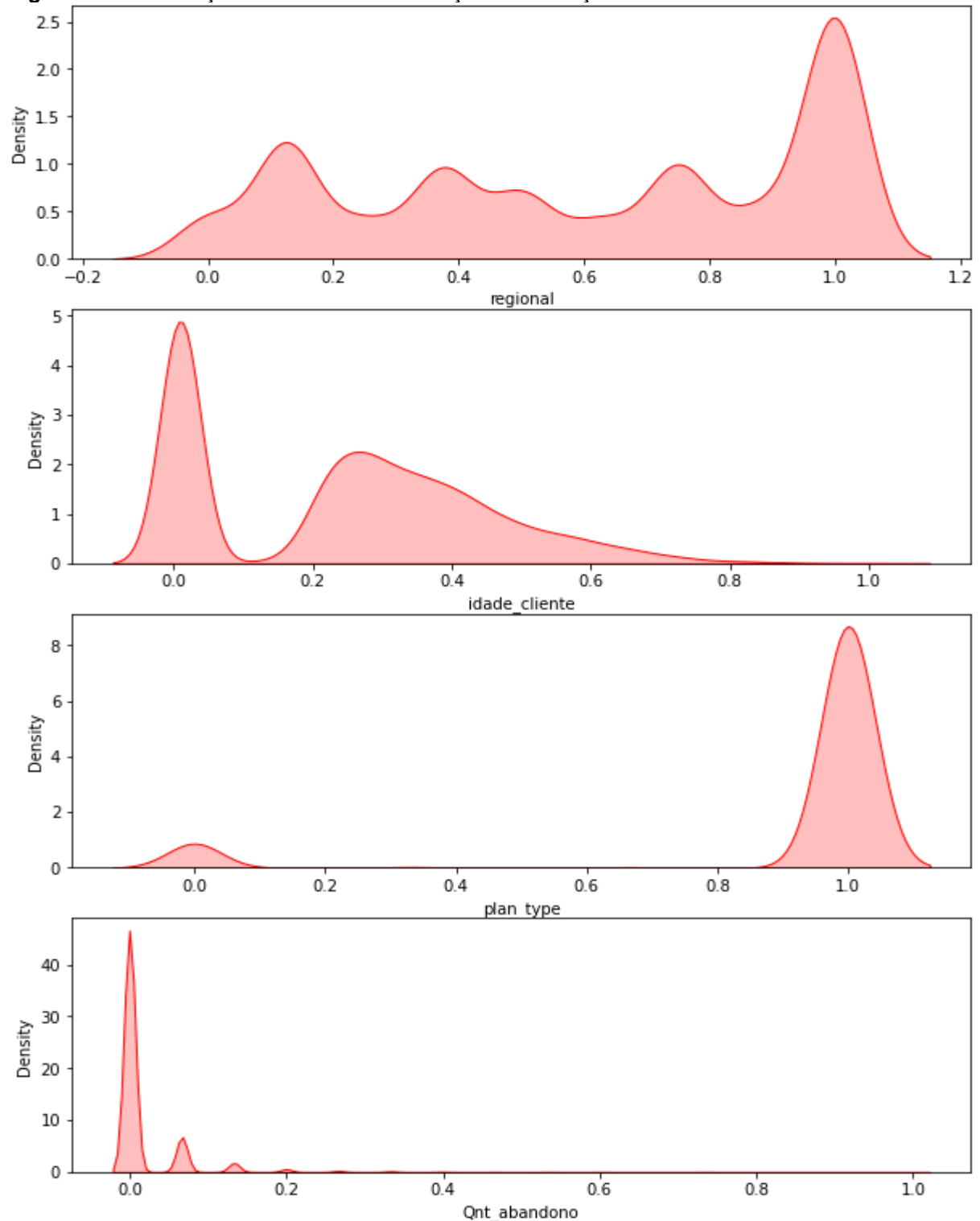


Figura 90 – Distribuição das variáveis *Lead* com redução de dimensionalidade *MinMaxScaler*

Para conclusão das análises e seleção das características, vamos utilizar um aliado para identificar as *features* mais importantes e confrontar com as análises e exploração dos dados realizados anteriormente.

Vamos utilizar as variáveis preditoras e a variável alvo no validador cruzado, nas Figuras 91 e 93, para realizar o treinamento do modelo Figura 95 e plotar o gráfico na Figura 97.

O modelo classificador utilizado no `feature_importance_` será o *RandomForestClassifier* (Floresta aleatória) Figura 95, o algoritmo seleciona a importância das variáveis preditoras Figura 96 a partir dos seus coeficientes.

```
# def X and Y
Y=np.array(data.venda.tolist())
df=data.drop('venda', axis=1)
X=np.array(df.to_numpy())
seed=42
```

```
1 X.shape
(16376, 45)
```

```
1 Y.shape
(16376,)
```

Figura 91 – X - Preditoras Perguntas e Y - Classes Resposta

O Validador cruzado de dobras K estratificadas, fornece índices de treinamento/teste para dividir dados em conjuntos de treinamento/teste. Este objeto de validação cruzada é uma variação do *k - fold* que retorna dobras estratificadas. As dobras são feitas preservando a porcentagem de amostras para cada classe.

A medida de desempenho relatada pela validação cruzada *k - fold* é então a média dos valores calculados no *loop*. Essa abordagem pode ser computacionalmente cara, mas não desperdiça muitos dados (como é o caso da fixação de um conjunto de validação arbitrário), o que é uma grande vantagem em problemas como inferência inversa, onde o número de amostras é muito pequeno.

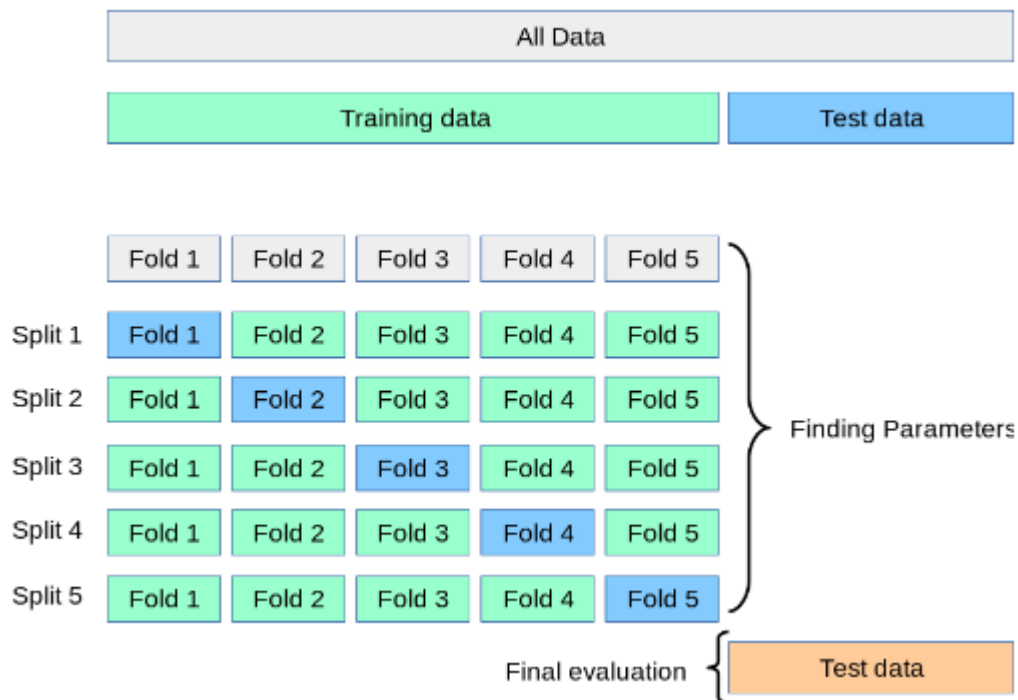


Figura 92 – StratifiedKFold

```

1 # StratifiedKFold k = 5
2 skf=StratifiedKFold(n_splits=5,
3                     shuffle=True,
4                     random_state=seed)
5 for train_index, test_index in skf.split(X, Y):
6     print("TRAIN:", train_index, "TEST:", test_index)
7     X_train, X_test = X[train_index], X[test_index]
8     Y_train, Y_test = Y[train_index], Y[test_index]

```

```

TRAIN: [  0  1  2 ... 16371 16373 16374] TEST: [ 11 12 17 ... 16370 16372 16375]
TRAIN: [  0  1  4 ... 16371 16372 16375] TEST: [  2  3  5 ... 16368 16373 16374]
TRAIN: [  0  2  3 ... 16373 16374 16375] TEST: [  1  4 10 ... 16352 16365 16371]
TRAIN: [  0  1  2 ... 16373 16374 16375] TEST: [  6 13 16 ... 16361 16363 16367]
TRAIN: [  1  2  3 ... 16373 16374 16375] TEST: [  0  7  9 ... 16360 16364 16366]

```

```

1 X_train.shape
(13101, 45)

```

```

1 X_test.shape
(3275, 45)

```

```

1 Y_train.shape
(13101,)

```

```

1 Y_test.shape
(3275,)

```

Figura 93 – StratifiedKFold


```
# Função para plotar gráfico feature importance
def plot_feature_importance(model):
    tmp=pd.DataFrame({'Feature': predictors, 'Feature importance': model.feature_importances_})
    tmp=tmp.sort_values(by='Feature importance',ascending=False)
    plt.figure(figsize = (10,18))
    plt.title('Features importance',fontsize=20)
    s=sns.barplot(y='Feature',x='Feature importance',data=tmp)
    s.set_yticklabels(s.get_yticklabels(),rotation=360)
    plt.show()
```

Figura 94 – Função de ajuste do gráfico *Feature importance*

```
# Ajustar o modelo usando X como dados de treinamento e y como valores de destino
rf_clf=RandomForestClassifier(n_estimators=100, random_state=seed, n_jobs=-1)
rf_clf=rf_clf.fit(X_train, Y_train)
```

Figura 95 – Modelo classificador utilizado

```
# Variaveis preditoras
predictors=['regional', 'idade_cliente', 'plan_type', 'Qnt_abandono',
            'sum_recharge', 'recharge_frequency', 'rec_online_10',
            'rec_online_35_b5', 'rec_online_15', 'sos_rec_5', 'rec_online_20_b2',
            'chip_pre_rec_10', 'chip_pre_rec_20', 'rec_online_13',
            'rec_online_50_b8', 'rec_online_30_b4', 'rec_online_40_b6',
            'pct_rec_1190', 'pct_rec_690', 'rec_online_100_b18', 'pct_rec_sos_5',
            'sos_rec_3', 'rec_online_8', 'sum_services', 'services_frequency',
            'inter_avulsa', 'antivirus', 'app_educacao', 'app_emprego', 'app_saude',
            'clube', 'pre_mix_giga', 'entretenimento', 'games',
            'pct_internet_mensal', 'prezao_diario', 'prezao_mensal',
            'prezao_quinzenal', 'prezao_semanal', 'recarga_sos',
            'servicos_operadora', 'sms_cobrar', 'sms_internacional',
            'transf_entre_regionais', 'truecaller']

# Plote feature importance
plot_feature_importance(rf_clf)
```

Figura 96 – Variáveis preditoras

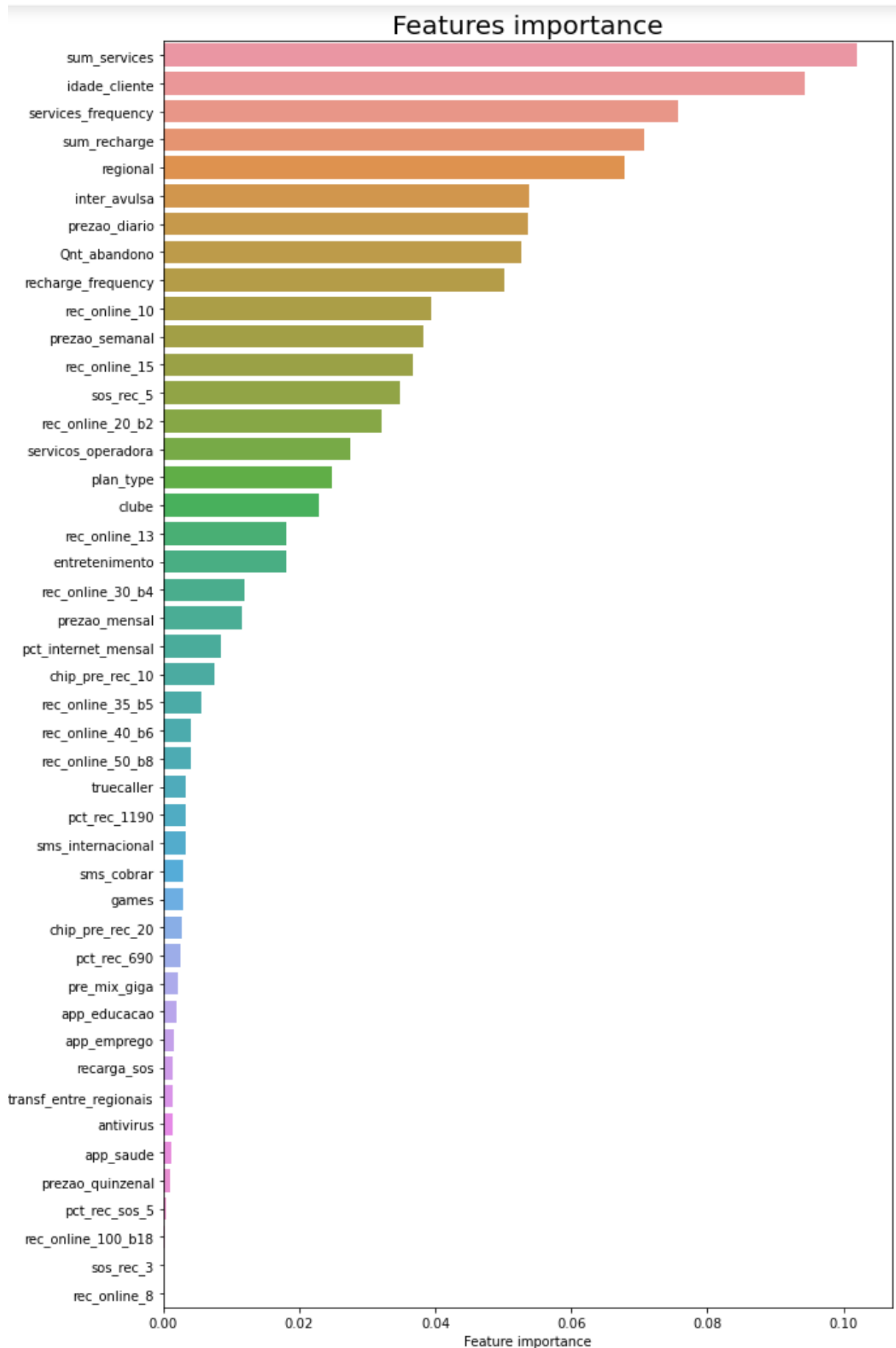


Figura 97 – Gráfico de barras contendo a importância das variáveis preditoras para o modelo

O `feature_importance_` retorna a Figura 97, onde cada elemento é uma variável que foi carregada no presente estudo.

A Figura 97, irá dizer, em proporções, quão importante aquela variável é para o modelo, **onde quanto maior o valor, mais importante a variável é para o modelo**. O modelo *RandomForestClassifier* (Floresta aleatória) Figura 95, é o modelo utilizado para a avaliação das variáveis.

Concluindo a análise, vamos entrar na seleção das variáveis, podemos observar e confrontar os estudos da matriz de correlação com `feature_importance_` as importâncias das variáveis preditoras dado variável alvo com a matriz de correlação.

Podemos observar que nos dois estudos, as variáveis preditoras que não tem relevância ou está com coeficiente inválido são respectivamente iguais.

As variáveis que serão eliminadas do estudo, são:

```
# Drop variáveis
to_drop=['rec_online_8', 'sos_rec_3', 'rec_online_100_b18', 'pct_rec_sos_5',
        'prezao_quinzenal', 'antivirus', 'app_saude', 'app_emprego', 'recarga_sos',
        'transf_entre_regionais', 'pre_mix_giga', 'app_educacao', 'pct_rec_690',
        'chip_pre_rec_20', 'games', 'sms_cobrar', 'pct_rec_1190', 'truecaller',
        'sms_internacional']
data.drop(to_drop, axis=1, inplace=True)
```

Figura 98 – Drop variáveis preditoras que não serão utilizadas

```
# Get .csv
data.to_csv('data.csv', sep=';', index=False)
```

Figura 99 – Gerando .csv para modelagem

5. Criação de Modelos de *Machine Learning*

Após todo processo de *data mining* (Mineração de dados), temos um *dataset* final para utilizarmos em treino e teste.

Todas as bibliotecas para implementação da estratificação, redução de dimensionalidade, modelos e métricas, são bibliotecas encontradas no site *Scikit-Learn*.

O estimador *MinMaxScaler*, dimensiona e traduz cada variável individualmente de tal forma que esteja no intervalo dado no conjunto de treinamento, por exemplo, entre -1 e 1. Conforme no tratamento de outliers na Figura 98, essa transformação é frequentemente utilizada como alternativa ao escalonamento de média zero e variância unitária, o que vemos muito nas estáticas simples das variáveis de recargas e serviços.

Para criação dos modelos, vamos utilizar o *dataset* final para definir X (Variáveis preditoras) e Y (Variável resposta).

A estratificação dos dados utilizando validador cruzado de K dobras estratificadas (*StratifiedKFold*) que fornece índices de treinamento/Teste, dividindo os dados em conjuntos de treinamento/teste. O melhor desempenho encontrado no *StratifiedKFold* é quando o parâmetro *n_splits* está definido com o número de subdivisões iguais $K = 5$, onde o *Fold* significa cada um dos blocos de cada K.

O próximo passo é a criação dos modelos e avaliação dos resultados de aprendizado de cada modelo para escolhermos um modelo final e atendermos as perguntas do problema proposto.

Entre os modelos, temos o *Support Vector Machines* (svm) que escolhe o limite de decisão que maximiza a distância dos pontos de dados mais próximos de todas as classes, ou seja, o limite de decisão mais ótimo, o limite ótimo tem margem máxima dos pontos mais próximos de todas as classes, a distância entre o limite de decisão e os pontos são chamados de vetores de suporte, como visto na Figura 100.

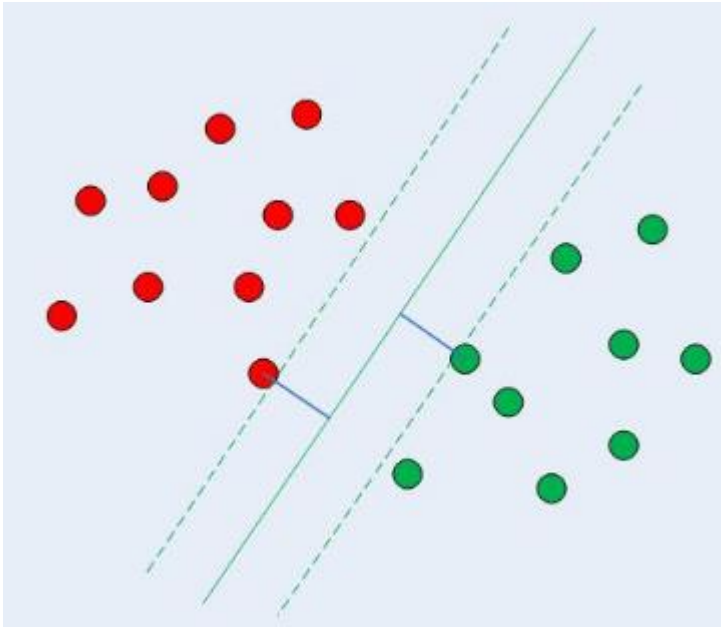


Figura 100 – Limite de Decisão com Vetores de Suporte

Para o estudo inicial com o modelo *Support Vector Machines* (svm), vamos utilizar os parâmetros *gamma*: definido como *auto* (coeficiente de *kernel* para 'rbf', 'poli' e 'sigmoid'), *probability*: definido como *True* (calcular probabilidades de resultados possíveis para amostras em *X*), *random_state*: definido com a semente (*seed*) 42 e demais parâmetros serão *default*, conforme definido na Figura 102.

O *Gaussian Naive Bayes* (gnb) é um modelo de classificação probabilística básico, mas eficaz em aprendizado de máquina. O teorema de *Bayes* é uma fórmula que oferece uma probabilidade condicional de um evento *A* ocorrer, dado que outro evento *B* ocorreu anteriormente.

Para o estudo inicial com o modelo *Gaussian Naive Bayes* (gnb), não definimos nenhum parâmetro, o modelo irá utilizar os parâmetros *default*, conforme definido na Figura 102.

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

Figura 101 – Fórmula matemática Bayes

RandomForestClassifier (rfc) é um meta-estimador que se encaixa em vários classificadores de árvore de decisão em várias subamostras do conjunto de dados e usa a média para melhorar a precisão preditiva e controlar o ajuste excessivo.

O tamanho da subamostra é sempre o mesmo que o tamanho da amostra de entrada original, mas as amostras são extraídas com substituição se *bootstrap=True* (padrão) e os demais parâmetros utilizados inicialmente no modelo *RandomForestClassifier* (rfc), temos o *n_estimators*: definido com o número de 100 árvores na floresta, *random_state*: definido com a semente (*seed*) 42 e demais parâmetros *default*, conforme definido na Figura 102.

```
clfs={  
    'svm': SVC(gamma='auto', probability=True, random_state=seed),  
    'gnb': GaussianNB(),  
    'rfc': RandomForestClassifier(n_estimators=100, random_state=seed),  
}
```

Figura 102 – Modelos e parâmetros

1. Como principal métrica, temos a Matriz de confusão (*confusion_matrix*), que é uma forma simples de visualizar a performance de um modelo de classificação. A matriz indica quantos exemplos existem em cada grupo:
 - falso positivo (FP): indica a quantidade de registros que foram classificados como positivos de maneira incorreta, ou seja, a resposta do classificador foi que a venda foi realizada, mas a venda não havia sido realizada.
 - falso negativo (FN): indica a quantidade de registros que foram classificados como negativos de maneira incorreta, ou seja, a resposta do classificador foi que a venda não foi realizada, mas a venda havia sido realizada.

- verdadeiro positivo (TP): indica a quantidade de registros de venda que foram classificados como venda corretamente, ou seja, a resposta do classificador foi que a venda havia sido realizada e a venda realmente foi realizada.
- verdadeiro negativo (TN): indica a quantidade de registros de não venda que foram classificados como não venda de maneira correta, ou seja, a resposta do classificador foi que não havia venda e realmente não tiveram a venda.

		P R E D I T O	
		👍 POSITIVO	👎 NEGATIVO
R E A L	👍 POSITIVO	✅ 👍 TP verdadeiro positivo	❌ 👎 FN falso negativo
	👎 NEGATIVO	❌ 👍 FP falso positivo	✅ 👎 TN verdadeiro negativo

Figura 103 – Composição da matriz de confusão

```

accuracy_scores=dict()
precision_scores=dict()
recall_scores=dict()
f1_scores=dict()
aucs=dict()
for clf_name in clfs:
    print(clf_name)
    clf=clfs[clf_name]
    clf.fit(X_train, Y_train)
    # Modelos prevendo os valores para o conjunto de teste
    Y_pred=clf.predict(X_test)
    # Prever as probabilidades de classe para o conjunto de teste
    Y_score=clf.predict_proba(X_test)[: , 1]
    accuracy_scores[clf_name]=accuracy_score(Y_test, Y_pred)
    precision_scores[clf_name]=precision_score(Y_test, Y_pred)
    recall_scores[clf_name]=recall_score(Y_test, Y_pred)
    f1_scores[clf_name]=f1_score(Y_test, Y_pred)
    aucs[clf_name]=roc_auc_score(Y_test, Y_score)

```

Figura 104 – Métricas para avaliação dos modelos

2. Acurácia (*accuracy_score*) é o indicador mais simples de se calcular. Ele é simplesmente a divisão entre todos os acertos pelo total.

$$\text{Acurácia} = \frac{\checkmark \text{👍 TP} + \checkmark \text{👎 TN}}{\checkmark \text{👍 TP} + \checkmark \text{👎 TN} + \text{✗} \text{👍 FP} + \text{✗} \text{👎 FN}}$$

Figura 105 – Composição da acurácia

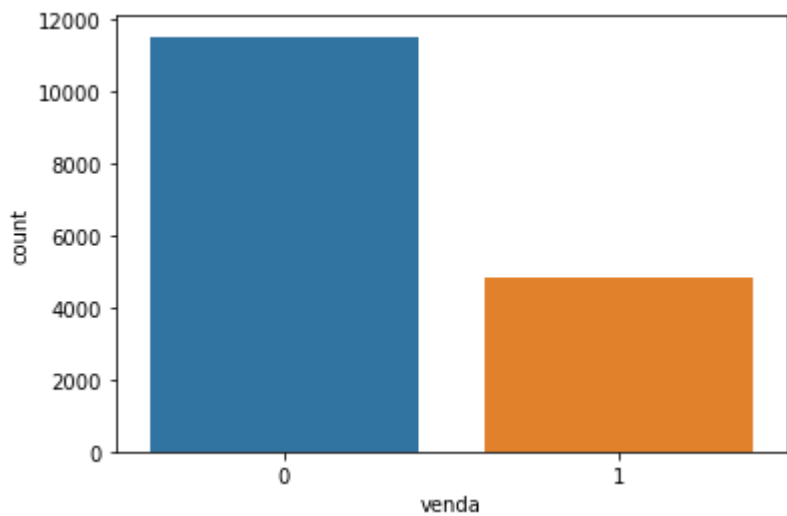
Para o estudo proposto, esse indicador não deve ser considerado, devido ao desbalanceamento das classes da variável venda, ou seja, temos 70% de registros que não há venda e os modelos utilizados no estudo, tem uma *Accuracy* de 70% ou próximo dos 70%, parece um número bem interessante. Mas, evidentemente, não é um bom teste, o que significa que não deveríamos colocar muita crença na acurácia bruta. É comum considerar a combinação de precisão e revocação, que combinados, temos *f1-score*.

Uma estratégia que não utilizamos no presente projeto e tivemos êxito, foi o de data *imputation*, que contribuiria ainda mais no desequilíbrio das classes de vendas.

Value	Count	Frequency (%)
0	40490	76.0%
1	12767	24.0%

Figura 106 – Classes de vendas antes da eliminação das linhas com valores ausentes

A Figura 106, mostra as classes “0” (Não fraude) e “1” (Fraude) da variável alvo vendas antes da eliminação das linhas contendo valores ausentes.



```
% of normal transacation      : 0.7048119198827553
Number of normal transaction   : 11542
% of fraud transacation       : 29.518808011724474
Number of fraud transaction    : 4834
```

Figura 107 – Informações sobre venda e não venda

A Figura 107, mostra que o balaceamento das classes da variável venda no *dataset* atual teve um equilíbrio maior em relação as classes da variável venda do dataset inicial, que continham os valores ausentes, mas não podemos afirmar que isso contibui na predição, pois como visto anteriormente nos graficos de densidades das variáveis preditoras, elas não separam muito bem as classes pertinetes a variável alvo venda.

```
{'svm': 0.7114503816793893,
'gnb': 0.6665648854961832,
'rfc': 0.7261068702290077}
```

Figura 108 – Resultado da acurácia dos modelos

3. Precisão (*precision_score*) é o número de exemplos classificados como pertencentes a uma classe, que realmente são daquela classe (positivos verdadeiros), dividido pela soma entre este número, e o número de exemplos classificados nesta classe, mas que pertencem a outras (falsos positivos).

$$\text{Precisão} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Figura 109 – Composição da precisão

```
{'svm': 0.5339805825242718,
'gnb': 0.4303097345132743,
'rfc': 0.5775280898876405}
```

Figura 110 – Resultado da precisão dos modelos

4. Revocação (*recall_score*) é o número de exemplos classificados como pertencentes a uma classe, que realmente são daquela classe, dividido pela quantidade total de exemplos que pertencem a esta classe, mesmo que sejam classificados em outra. No caso binário, positivos verdadeiros divididos por total de positivos.

$$\text{Revocação} = \frac{\text{✅👍 TP}}{\text{✅👍 TP} + \text{❌👎 FN}}$$

Figura 111 – Composição da revocação

```
{'svm': 0.17080745341614906,
'gnb': 0.4026915113871636,
'rfc': 0.2660455486542443}
```

Figura 112 – Resultado da revocação dos modelos

5. F1 (*f1_score*) é uma média harmônica entre precisão (que, apesar de ter o mesmo nome, não é a mesma citada acima) e *recall*. Veja abaixo as definições destes dois termos. Ela é muito boa quando você possui um *dataset* com classes desproporcionais, e o seu modelo não emite probabilidades. Isso não significa que não possa ser usada com modelos que emitem probabilidades, tudo depende do objetivo de sua tarefa de *machine learning*. Em geral, quanto maior o F1, melhor.

$$F_1 = 2 * \frac{\text{precisão} * \text{recall}}{\text{precisão} + \text{recall}}$$

Figura 113 – Composição do F1 score

```
{'svm': 0.25882352941176473,
'gnb': 0.4160427807486631,
'rfc': 0.36428065201984405}
```

Figura 114 – Resultado da F1 score dos modelos

6. AUC (*roc_auc_score*) é uma métrica interessante para tarefas com classes desproporcionais. Nela, mede-se a área sob uma curva formada pelo gráfico entre a taxa de exemplos positivos, que realmente são positivos, e a taxa de falsos positivos.

Uma das vantagens em relação ao F1, é que ela mede o desempenho do modelo em vários pontos de corte, não necessariamente atribuindo exemplos com probabilidade maior que 50% para a classe positiva, e menor, para a classe negativa.

Em sistemas que se interessam apenas pela classe, e não pela probabilidade, ela pode ser utilizada para definir o melhor ponto de corte para atribuir uma ou outra classe a um exemplo.

Este ponto de corte normalmente é o ponto que se localiza mais à esquerda, e para o alto, no gráfico, mas depende bastante do custo do erro na previsão de uma determinada classe.

$$\text{TPR} = \frac{\text{✓👍 TP}}{\text{✓👍 TP} + \text{✗👎 FN}} \quad \text{FPR} = \frac{\text{✗👍 FP}}{\text{✗👍 FP} + \text{✓👎 TN}}$$

Figura 115 – Composição da AUC

```
{'svm': 0.6593117937102724,
'gnb': 0.6352193729281495,
'rfc': 0.6947366816498945}
```

Figura 116 – Resultado AUC dos modelos

Prosseguindo o estudo com implementação do ajuste de *hiperparâmetros* *GridSearchCV* do *Scikit-Learn*, vamos realizar o *tuning* nas configurações dos

modelos para otimizar o desempenho e realizar uma nova avaliação e validação das métricas e escolha do modelo final.

O passo inicial é definir os parâmetros que serão otimizados para cada modelo, vamos treinar o modelo, passar como parâmetro para o *GridSearchCV* os parâmetros do modelo que serão otimizados e o treinamento do modelo seguido da estratégia para avaliar o desempenho do modelo de validação cruzada no conjunto de testes e ainda utilizando o *GridSearchCV*, um novo treinamento para executar o ajuste com todos os conjuntos de parâmetros.

➤ *GridSearchCV e Suport Vector Machine:*

Para o estudo *tuning* do modelo *Support Vector Machines* (svm) utilizando *GridSearchCV* com estratégia voltada para a revocação (*recall_score*), número de exemplos classificados como pertencentes a uma classe, que realmente são daquela classe, vamos utilizar no *param_grid_svm*, os parâmetros 'C': [0.1, 1, 10, 100] (parâmetro de regularização), 'gamma': [1, 0.1, 0.01, 0.001], 'kernel': ['rbf', 'poly', 'sigmoid', 'linear'] (Especifica o tipo de *kernel* a ser usado no *algoritmo*), conforme Figura 117.

```
# Best parameters grid
param_grid_svm={
    'C': [0.1, 1, 10, 100],
    'gamma': [1, 0.1, 0.01, 0.001],
    'kernel': ['rbf', 'poly', 'sigmoid', 'linear']
}
```

Figura 117 – Os melhores parâmetros *GridSearchCV* e *Suport Vector Machine*

Para *GridSearchCV* e *Suport Vector Machine* (svm), os parâmetros do modelo que serão otimizados e o treinamento do modelo, são seguidos da estratégia utilizada para avaliar o desempenho do modelo de validação cruzada no conjunto de testes e ainda utilizando o *GridSearchCV*, um novo treinamento para executar o ajuste com todos os conjuntos de parâmetros, conforme definido na Figura 118.

```
# Ajustar o modelo usando X como dados de treinamento e y como valores de destino
svm=SVC(gamma='auto', random_state=seed)
svm=svm.fit(X_train, Y_train)

# Best parameters SVM
gs_svm=GridSearchCV(estimator=svm, param_grid=param_grid_svm, scoring='recall',
                    verbose=10, n_jobs=-1)
gs_svm=gs_svm.fit(X_train, Y_train)
best_parameters=gs_svm.best_params_
print("The best parameters for using this model SVM is", best_parameters)
Fitting 5 folds for each of 64 candidates, totalling 320 fits
The best parameters for using this model SVM is {'C': 100, 'gamma': 1, 'kernel': 'sigmoid'}
```

Figura 118 – Os melhores parâmetros SVM

➤ *GridSearchCV e Gaussian Naive Bayes:*

Para o estudo *tuning* do modelo *Gaussian Naive Bayes* (gnb) utilizando *GridSearchCV* com estratégia voltada para a revocação (*recall_score*), número de exemplos classificados como pertencentes a uma classe, que realmente são daquela classe, vamos utilizar no *param_grid_gnb*, o parâmetro utilizado no *tuning Gaussian Naive Bayes* é o '*var_smoothing*': *np.logspace(0,-9, num=100)* (parte da maior variância de todos os recursos que é adicionada às variâncias para estabilidade de cálculo), conforme definido na Figura 119.

```
# Best parameters grid
param_grid_gnb={
    'var_smoothing': np.logspace(0,-9, num=100)
}
```

Figura 119 – Os melhores parâmetros *GridSearchCV* e *Gaussian Naive Bayes*

Para *GridSearchCV* e *Gaussian Naive Bayes*, os parâmetros do modelo que serão otimizados e o treinamento do modelo, são seguidos da estratégia utilizada para avaliar o desempenho do modelo de validação cruzada no conjunto de testes e ainda utilizando o *GridSearchCV*, um novo treinamento para executar o ajuste com todos os conjuntos de parâmetros, conforme definido na Figura 120.

```
# Ajustar o modelo usando X como dados de treinamento e y como valores de destino
gnb=GaussianNB ()
gnb=gnb.fit(X_train, Y_train)

# Best parameters GNB
gs_gnb=GridSearchCV(estimator=gnb, param_grid=param_grid_gnb, scoring='recall',
                    verbose=10, n_jobs=-1)
gs_gnb=gs_gnb.fit(X_train, Y_train)
best_parameters=gs_gnb.best_params_
print("The best parameters for using this model GNB is", best_parameters)
Fitting 5 folds for each of 100 candidates, totalling 500 fits
The best parameters for using this model GNB is {'var_smoothing': 5.3366992312063123e-05}
```

Figura 120 – Os melhores parâmetros GNB

➤ *GridSearchCV e RandomForestClassifier.*

Para o estudo *tuning* do modelo *RandomForestClassifier* (rfc) utilizando *GridSearchCV* com estratégia voltada para a revocação (*recall_score*), número de exemplos classificados como pertencentes a uma classe, que realmente são daquela classe, vamos utilizar no *param_grid_rfc*, os parâmetros utilizados no *RandomForestClassifier* são os '*n_estimators*': [100, 150, 200] (número de árvores na floresta), '*max_features*': ['auto', 'sqrt', 'log2'] (número de recursos a serem considerados ao procurar a melhor divisão), '*min_samples_leaf*': [2, 5, 10, 20] (número mínimo de amostras necessárias para estar em um nó folha), '*min_samples_split*': [2, 5, 10, 20] (número mínimo de amostras necessárias para dividir um nó interno), '*max_depth*': [4,5,6,7,8] (profundidade máxima da árvore), '*criterion*':['gini', 'entropy'] (função para medir a qualidade de uma divisão, suportados são "gini" para a impureza Gini e "entropia" para o ganho de informação) e '*class_weight*': ['balanced', 'balanced_subsample'] (pesos associados a classes no formulário, o modo "balanceado" usa os valores de y para ajustar automaticamente pesos inversamente proporcionais às frequências de classe nos dados de entrada como `n_samples / (n_classes * np.bincount(y))`, já o modo "balanced_subsample" é o mesmo que "balanced", exceto que os pesos são calculados com base na amostra *bootstrap* para cada árvore cultivada), conforme definido na Figura 121.

```
# Best parameters grid
param_grid_rfc={
    'n_estimators': [100, 150, 200],
    'max_features': ['auto', 'sqrt', 'log2'],
    'min_samples_leaf': [2, 5, 10, 20],
    'min_samples_split': [2, 5, 10, 20],
    'max_depth': [4,5,6,7,8],
    'criterion': ['gini', 'entropy'],
    'class_weight': ['balanced', 'balanced_subsample']
}
```

Figura 121 – Os melhores parâmetros *GridSearchCV* e *RandomForestClassifier*

Para *GridSearchCV* e *RandomForestClassifier* (rfc), os parâmetros do modelo que serão otimizados e o treinamento do modelo, são seguidos da estratégia utilizada para avaliar o desempenho do modelo de validação cruzada no conjunto de testes e ainda utilizando o *GridSearchCV*, um novo treinamento para executar o ajuste com todos os conjuntos de parâmetros, conforme definido na Figura 122.

```
# Ajustar o modelo usando X como dados de treinamento e y como valores de destino
rfc=RandomForestClassifier(n_estimators=100, random_state=seed)
rfc=rfc.fit(X_train, Y_train)

# Best parameters RFC
gs_rfc=GridSearchCV(estimator=rfc, param_grid=param_grid_rfc, scoring='recall',
                    verbose=10, n_jobs=-1)
gs_rfc=gs_rfc.fit(X_train, Y_train)
best_parameters = gs_rfc.best_params_
print("The best parameters for using this model RFC is", best_parameters)

Fitting 5 folds for each of 2880 candidates, totalling 14400 fits
The best parameters for using this model RFC is {'class_weight': 'balanced_subsample', 'criterion': 'entropy', 'max_depth': 5, 'max_features': 'log2', 'min_samples_leaf': 2, 'min_samples_split': 5, 'n_estimators': 200}
```

Figura 122 – Os melhores parâmetros RFC

➤ Avaliação do *tuning* dos melhores parâmetros para usar nos modelos:

```
clfs={
    'svm_best_param': SVC(C=100, gamma=1, kernel='sigmoid', probability=True, random_state=seed),
    'gnb_best_param': GaussianNB(var_smoothing=5.3366992312063123e-05),
    'rfc_best_param': RandomForestClassifier(class_weight='balanced_subsample', criterion='entropy',
                                           max_depth=5, max_features='log2', min_samples_leaf=2,
                                           min_samples_split=5, n_estimators=200, random_state=seed,
                                           n_jobs=-1),
}
```

Figura 123 – Modelos e melhores parâmetros


```

accuracy_scores=dict()
precision_scores=dict()
recall_scores=dict()
f1_scores=dict()
aucs=dict()
for clf_name in clfs:
    print(clf_name)
    clf=clfs[clf_name]
    clf.fit(X_train, Y_train)
    # Modelos prevendo os valores para o conjunto de teste
    Y_pred=clf.predict(X_test)
    # Prever as probabilidades de classe para o conjunto de teste
    Y_score=clf.predict_proba(X_test)[: , 1]
    accuracy_scores[clf_name]=accuracy_score(Y_test, Y_pred)
    precision_scores[clf_name]=precision_score(Y_test, Y_pred)
    recall_scores[clf_name]=recall_score(Y_test, Y_pred)
    f1_scores[clf_name]=f1_score(Y_test, Y_pred)
    aucs[clf_name]=roc_auc_score(Y_test, Y_score)

```

Figura 124 – Métricas para avaliação dos modelos

```

{'svm_best_param': 0.636030534351145,
 'gnb_best_param': 0.6665648854961832,
 'rfc_best_param': 0.6516030534351145}

```

Figura 125 – Resultado da acurácia dos modelos

```

{'svm_best_param': 0.37901498929336186,
 'gnb_best_param': 0.43,
 'rfc_best_param': 0.4417055296469021}

```

Figura 126 – Resultado da precisão dos modelos

```

{'svm_best_param': 0.36645962732919257,
 'gnb_best_param': 0.40062111801242234,
 'rfc_best_param': 0.6863354037267081}

```

Figura 127 – Resultado da revocação dos modelos

```

{'svm_best_param': 0.37263157894736837,
 'gnb_best_param': 0.41479099678456594,
 'rfc_best_param': 0.5374949331171464}

```

Figura 128 – Resultado da F1 score dos modelos

```

{'svm_best_param': 0.4351556202348,
 'gnb_best_param': 0.6349678591379309,
 'rfc_best_param': 0.7075544699963328}

```

Figura 129 – Resultado AUC dos modelos

Avaliando as métricas, após *tuning* dos modelos, podemos dizer que o modelo *Support Vector Machines* (svm) teve um resultado melhor, se comparado ao

desempenho sem o *tuning*, o modelo generalizou melhor, e teve os resultados otimizados na precisão Figura 126, revocação Figura 127 e F1 Figura 128, porem perdeu poder de aprendizado conforme visto na Figura 129.

Para o modelo *Gaussian Naive Bayes* (gnb), não tivemos quase nenhum movimento nas métricas, o modelo permaneceu com os valores das métricas, similares ao modelo inicial sem *tuning*.

O modelo *RandomForestClassifier* (rfc), por sua vez, teve as suas métricas potencializadas com o *tuning*, tendo ganhos na precisão Figura 126, revocação Figura 127, F1 Figura 128 e no aprendizado AUC Figura 129.

Avaliando os resultados das métricas do *tuning* dos modelos de classificação utilizados, para a escolha do modelo final, podemos observar que a estratégia voltada para a revocação (*recall_score*) Figura 127, onde a revocação indica o percentual de todas as fraudes que realmente foram fraudes corretamente.

O resultado AUC (*roc_auc_score*) Figura 129, que é uma métrica interessante para tarefas com classes desproporcionais, medindo o desempenho dos modelos, o modelo selecionado com o melhor desempenho nas duas métricas, revocação (*recall_score*) Figura 127 e AUC (*roc_auc_score*) Figura 129, qual seja, o *RandomForestClassifier* (Floresta Aleatória), é o modelo que vamos utilizar para dar continuidade no estudo.

No modelo escolhido *RandomForestClassifier* (Floresta Aleatória), tivemos bons resultados também na precisão Figura 126 e consequentemente no F1 Figura 128, se comparado com os modelos *Support Vector Machine* (Máquina de vetor de suporte) e *Gaussian Naive Bayes*.

6. Interpretação dos Resultados

O entendimento da correlação linear entre variáveis preditoras e alvo, avaliação da frequência das variáveis e seus coeficientes, descrição estatística das variáveis preditoras, avaliação da distribuição desequilibrada das classes que compõem a

variável alvo, avaliação da dimensionalidade das variáveis preditoras e a importância das variáveis para o modelo dado o resultado do `feature_importance_`, contribuíram para a geração do dataset final que utilizamos para realização da predição dos *Leads*.

Vamos analisar a correlação linear entre as variáveis preditoras e a variável venda do *dataset* final.

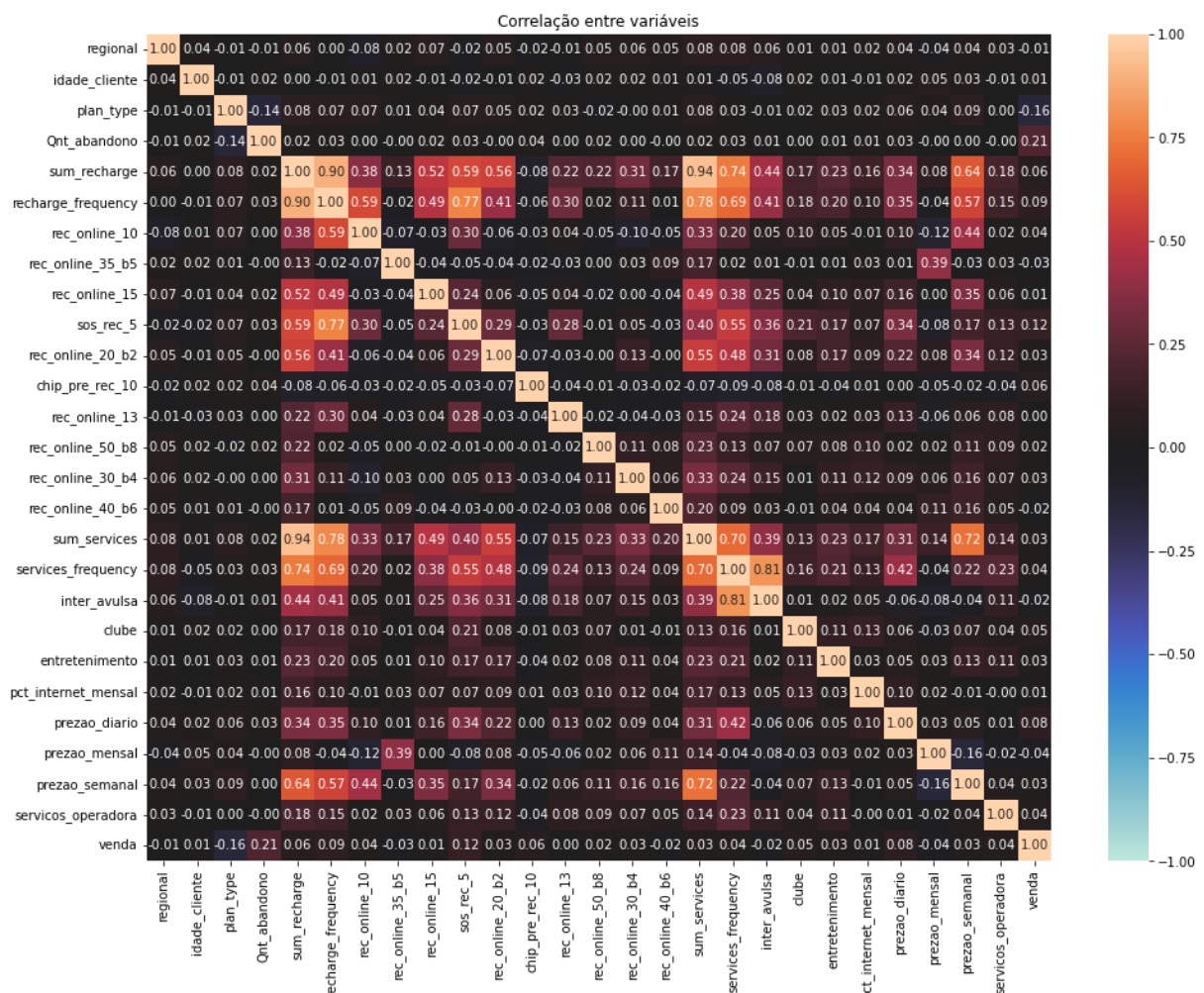


Figura 130 – Matriz de correlação linear das variáveis preditoras e variável alvo venda

Com o *dataset* final orquestrado e com a dimensionalidade reduzida, podemos observar na Figura 130, as variáveis **sum_recharge** e **recharge_frequency**, tem correlação linear positiva forte com as variáveis **sum_services**, **services_frequency** e **prezao_semanal**, indicando que os *Leads* que realizam recargas utilizam os serviços com certa frequência e conforme os valores de recargas e a fre-

quência das recargas aumentam, o serviço **prezao_semanal**, segue o mesmo movimento de correlação linear, conforme Figura 131.

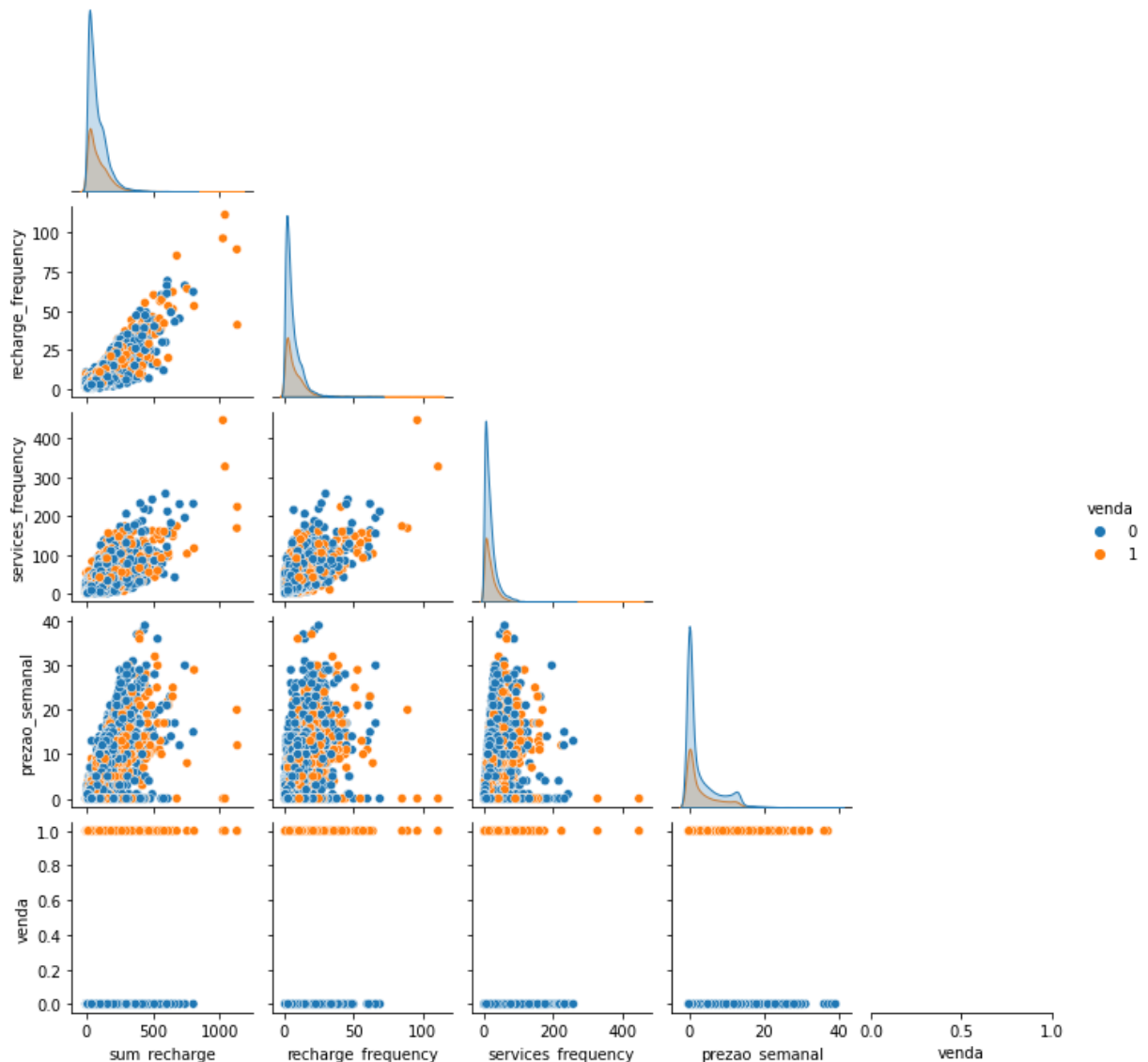


Figura 131 – Estimativas de densidades das variáveis com alguma correlação linear com a variável alvo

Ainda na Figura 131, podemos observar nos gráficos de densidade, que não temos uma boa separação na variável alvo **venda**, nas classes não venda (Sem fraude “0”) e venda (Fraude “1”).

As demais variáveis de serviços **inter_avulsa**, **clube**, **entretenimento**, **prezao_diario** e **servicos_operadora**, estão com correlação linear positiva baixa com a variável **sum_recharge**, indicando que os *Leads* que realizam recargas, em algum momento utilizam estes serviços.

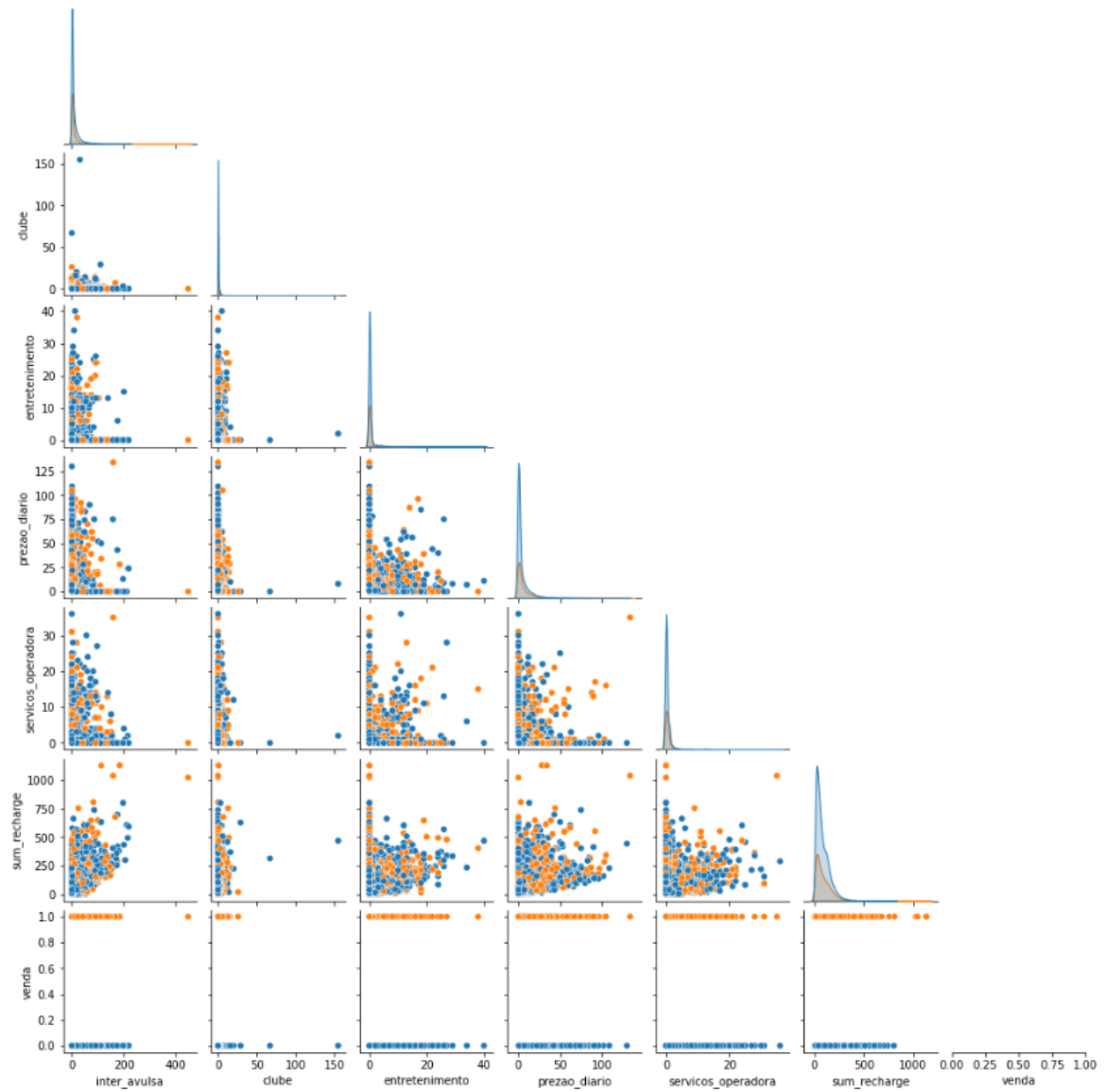


Figura 132 – Estimativas de densidades das variáveis com alguma correlação linear com a variável alvo

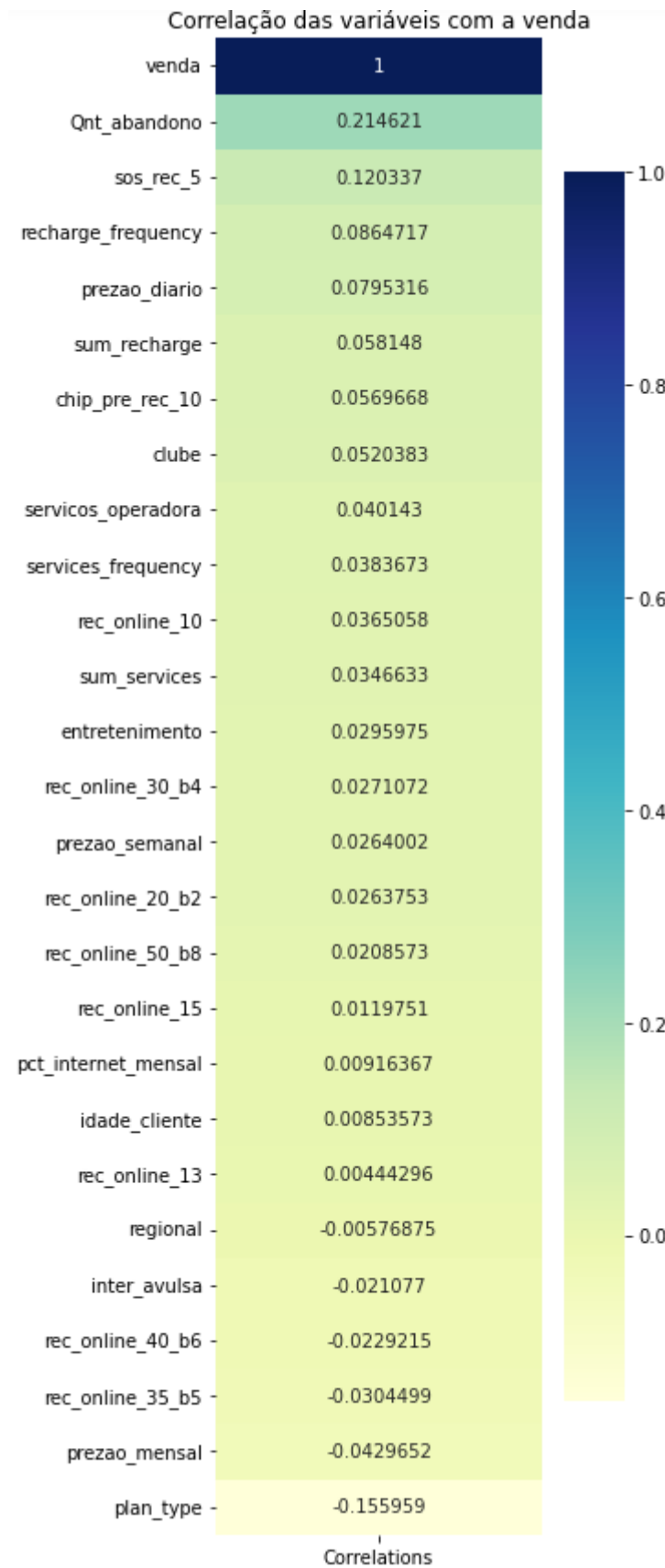


Figura 133 – Matriz de correlação linear das variáveis preditoras com a variável alvo venda

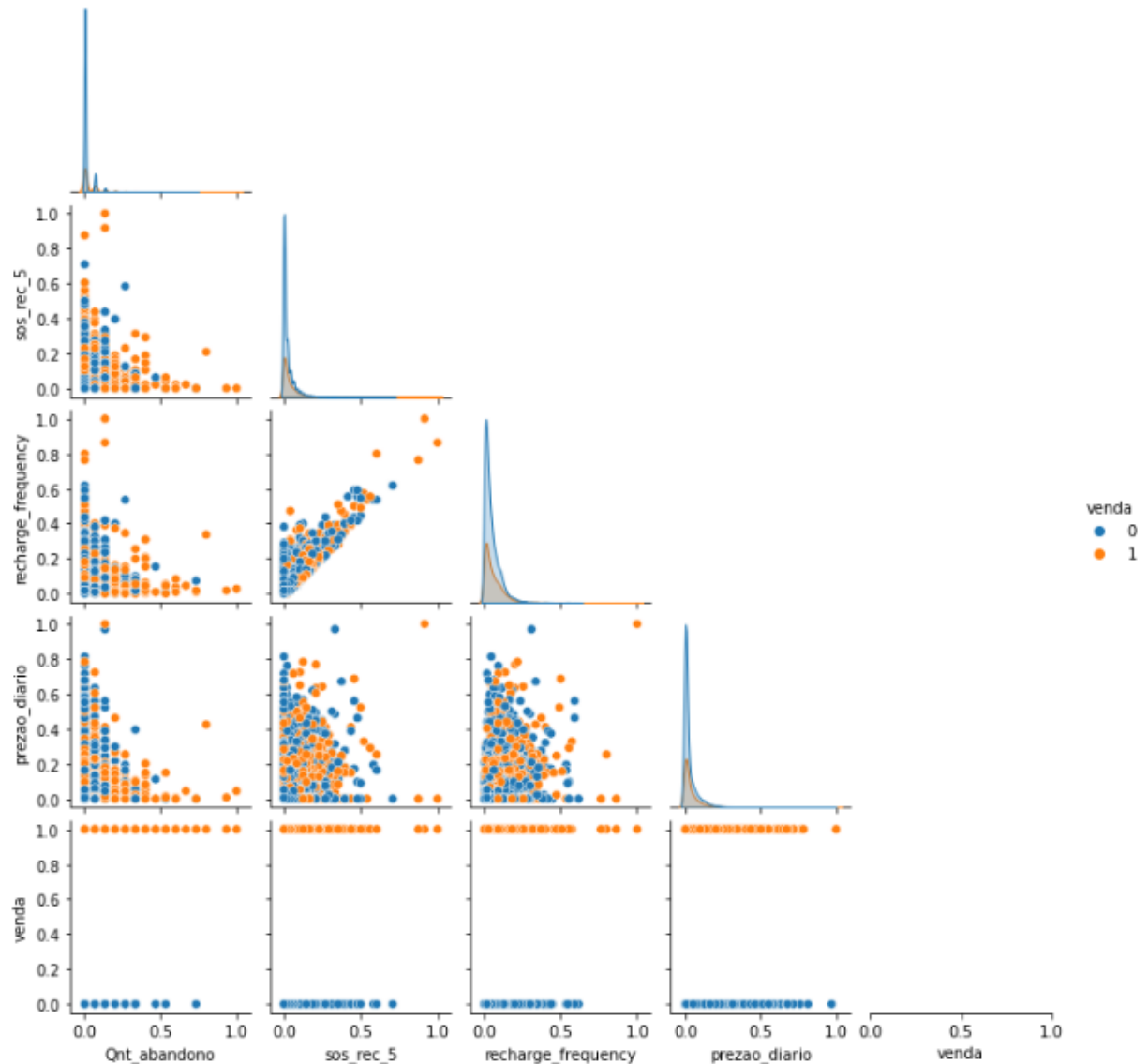


Figura 134 – Estimativas de densidades das variáveis com alguma correlação linear com a variável alvo

Na Figura 134, podemos observar que as variáveis **Qnt_abandono**, **sos_rec_5**, **recharge frequency** e **prezao_diario**, que possuem alguma correlação mais forte com a variável alvo **venda**, mas podemos observar o movimento de dispersão de algumas venda (Classe fraudulenta “1”) e que a maioria das variáveis não separam bem as classes não venda (Sem fraude “0”) da classe venda (Fraude “1”).

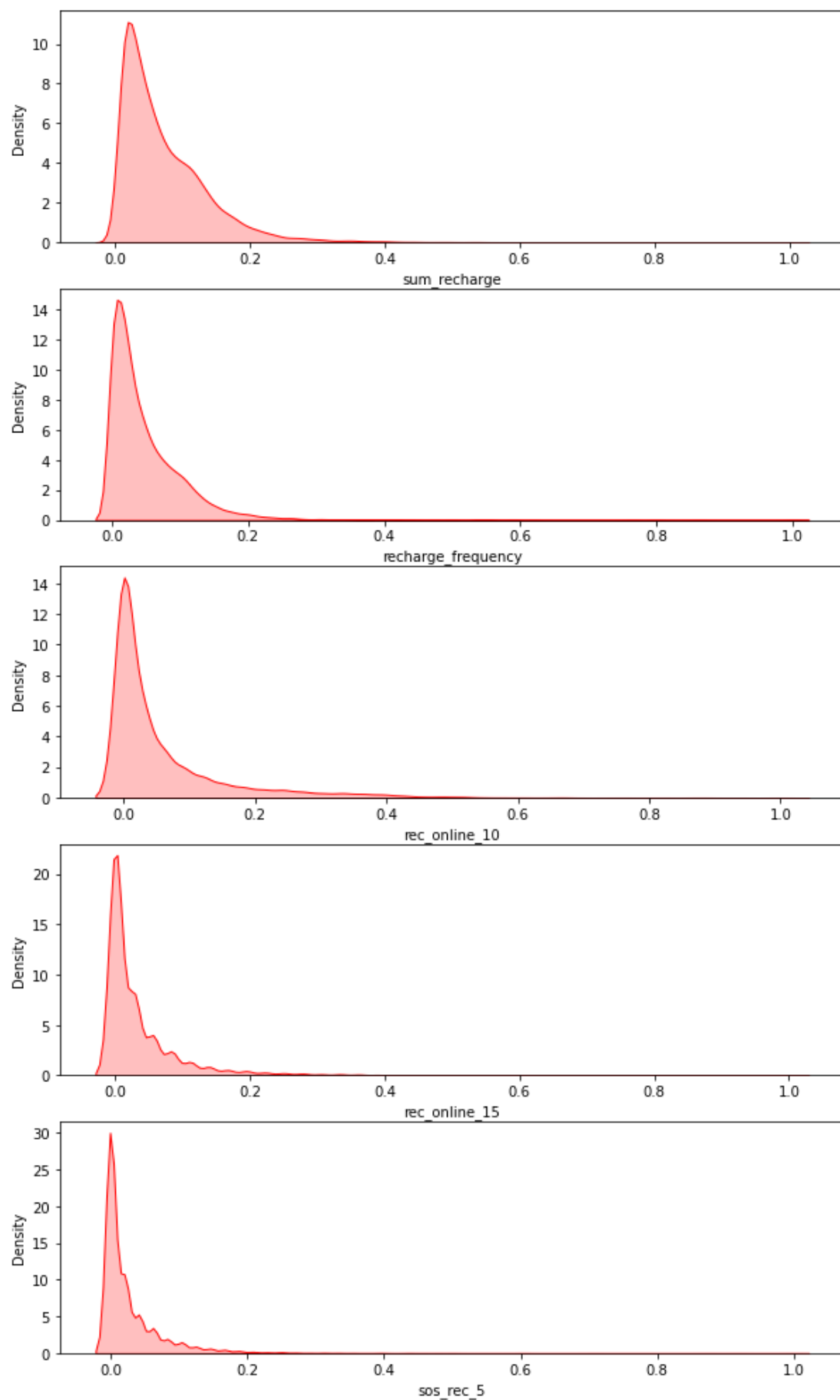


Figura 135 – Distribuição das variáveis de recargas com redução de dimensionalidade *MinMaxScaler*

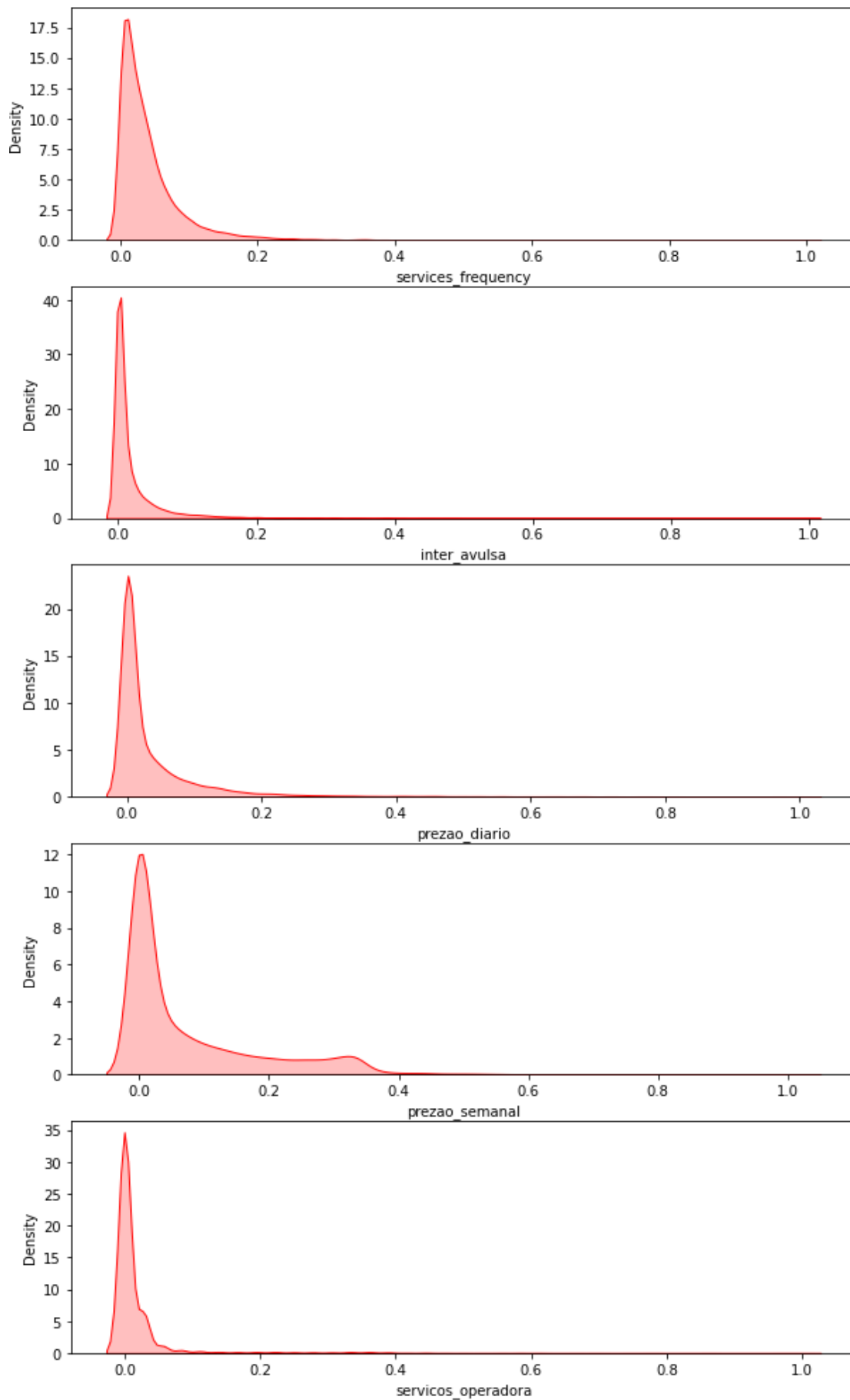


Figura 136 – Distribuição das variáveis de serviços com redução de dimensionalidade *MinMaxScaler*

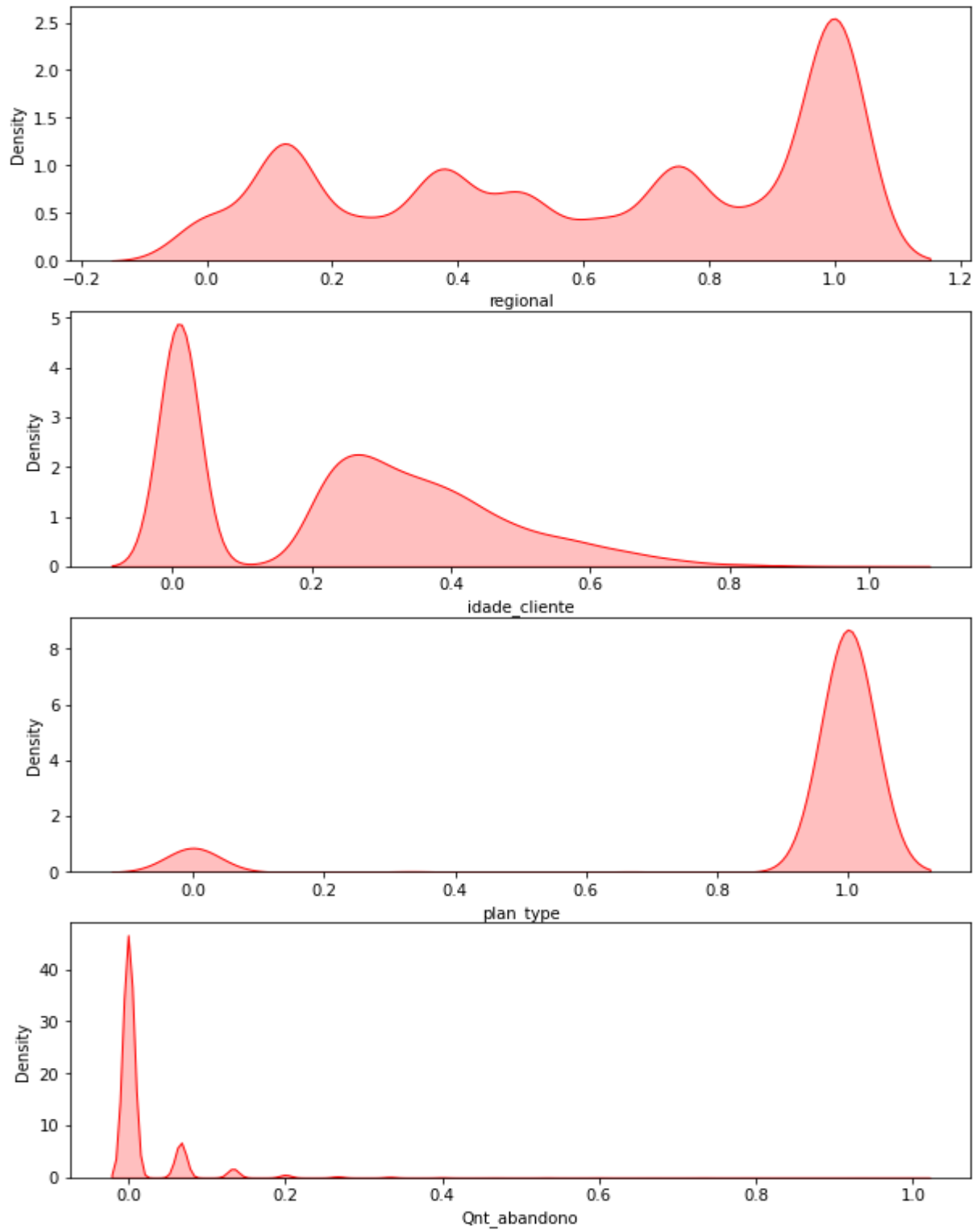


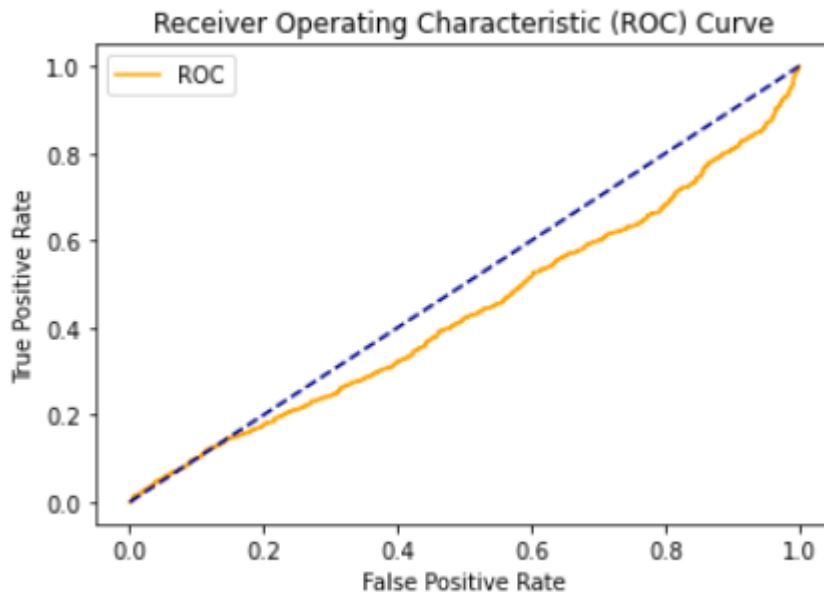
Figura 137 – Distribuição das variáveis Lead com redução de dimensionalidade *MinMaxScaler*

	count	mean	std	min	25%	50%	75%	max
regional	16376.0	4.868527	2.818845	0.00	2.00	5.00	8.0000	8.00
idade_cliente	16376.0	22.050623	20.190853	-2.00	-1.00	24.00	36.0000	98.00
plan_type	16376.0	2.720445	0.865847	0.00	3.00	3.00	3.0000	3.00
Qnt_abandono	16376.0	1.261053	0.740497	1.00	1.00	1.00	1.0000	16.00
sum_recharge	16376.0	82.675696	77.137051	0.00	30.00	60.00	118.0000	1133.00
recharge_frequency	16376.0	6.333842	6.238281	1.00	2.00	4.00	9.0000	111.00
rec_online_10	16376.0	1.913166	3.299767	0.00	0.00	1.00	2.0000	33.00
rec_online_35_b5	16376.0	0.078041	0.436381	0.00	0.00	0.00	0.0000	9.00
rec_online_15	16376.0	1.295249	2.456976	0.00	0.00	0.00	2.0000	36.00
sos_rec_5	16376.0	1.305264	2.427992	0.00	0.00	0.00	2.0000	48.00
rec_online_20_b2	16376.0	1.071385	1.933577	0.00	0.00	0.00	1.0000	29.00
chip_pre_rec_10	16376.0	0.040730	0.197671	0.00	0.00	0.00	0.0000	1.00
rec_online_13	16376.0	0.305997	1.109285	0.00	0.00	0.00	0.0000	42.00
rec_online_50_b8	16376.0	0.034929	0.315505	0.00	0.00	0.00	0.0000	11.00
rec_online_30_b4	16376.0	0.157181	0.633103	0.00	0.00	0.00	0.0000	12.00
rec_online_40_b6	16376.0	0.047386	0.333314	0.00	0.00	0.00	0.0000	10.00
sum_services	16376.0	68.549771	68.741957	0.01	19.98	46.18	100.1125	1202.26
services_frequency	16376.0	18.651380	21.593588	1.00	5.00	12.00	24.0000	448.00
inter_avulsa	16376.0	8.404433	18.052572	0.00	0.00	1.00	8.0000	447.00
clube	16376.0	0.376099	1.715855	0.00	0.00	0.00	0.0000	155.00
entretenimento	16376.0	0.595078	2.344888	0.00	0.00	0.00	0.0000	40.00
pct_internet_mensal	16376.0	0.094529	0.541283	0.00	0.00	0.00	0.0000	15.00
prezao_diario	16376.0	4.655532	9.634188	0.00	0.00	0.00	5.0000	134.00
prezao_mensal	16376.0	0.253786	0.806435	0.00	0.00	0.00	0.0000	18.00
prezao_semanal	16376.0	3.090620	4.614504	0.00	0.00	1.00	5.0000	39.00
servicos_operadora	16376.0	0.651136	2.245891	0.00	0.00	0.00	0.0000	36.00
venda	16376.0	0.295188	0.456141	0.00	0.00	0.00	1.0000	1.00

Figura 138 – Describe variáveis do dataset

➤ Modelo *Support Vector Machine* Tuning:

Plot ROC curve SVM



AUC: 0.44

Figura 139 – Roc curve SVM

A área coberta pela curva é a área entre a linha laranja (ROC) e o eixo. Esta área coberta é AUC. Quanto maior a área coberta, melhor os modelos de aprendizado de máquina distinguem as classes dadas.

O valor ideal para AUC é 1, mas um bom classificador terá AUC acima dos 0.90% e AUC para SVM é de 0.44%.

Interpretando a Figura 139, o *True Positive Rate* (TPR), até os primeiros 15%, aumenta também o *False Positive Rate* (FPR) 15% (a estimativa com é prevista corretamente e temos as classes verdadeiros positivos (não venda) e falsos positivos (venda)), mas depois incorre em um FPR de 40% para atingir um TPR de abaixo dos 30%.

Nas curvas ROC, a taxa de verdadeiros positivos (TPR, eixo y) é plotada em relação à taxa de falsos positivos (FPR, eixo x). Essas quantidades são definidas da seguinte forma, Figura 140:

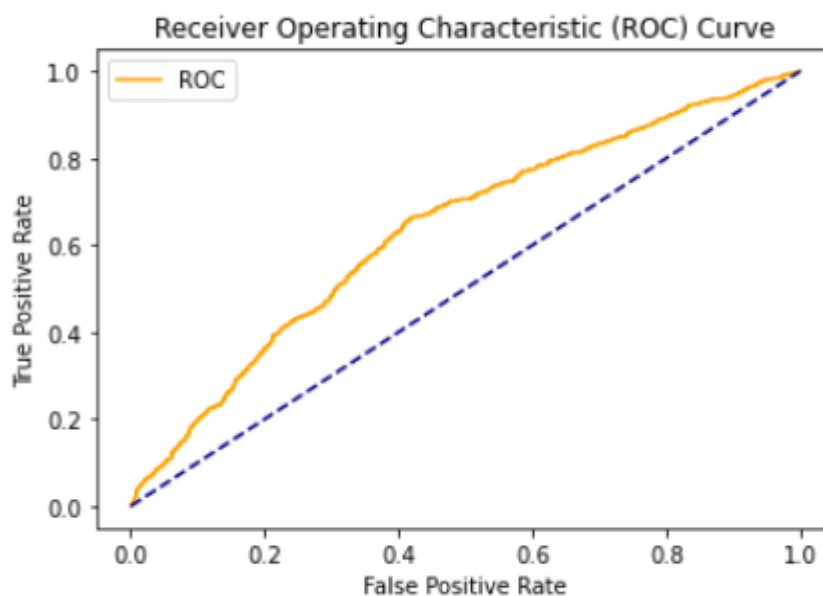
$$TPR = \frac{TP}{TP + FN}$$

$$FPR = \frac{FP}{FP + TN}$$

Figura 140 – Definição curva ROC

➤ Modelo *Gaussian Naive Bayes Tuning*:

Plot ROC curve GNB



AUC: 0.63

Figura 141 – Roc curve GNB

A área coberta pela curva é a área entre a linha laranja (ROC) e o eixo. Esta área coberta é AUC. Quanto maior a área coberta, melhor os modelos de aprendizado de máquina distinguem as classes dadas.

O valor ideal para AUC é 1, mas um bom classificador terá AUC acima dos 0.90% e AUC para GNB é de 0.63%.

Interpretando a Figura 141, o *True Positive Rate* (TPR), até os primeiros 30%, aumenta também o *False Positive Rate* (FPR) 20% (a estimativa com é prevista corretamente e temos as classes verdadeiros positivos (não venda) e falsos

positivos (venda)), mas depois incorre em um FPR de 50% para atingir um TPR de abaixo dos 70%.

- *RandomForestClassifier*, modelo que manteve o melhor resultado no *tuning* de parâmetros:

```
Confusion matrix:
[[1471  838]
 [ 303  663]]
```

Figura 142 – Resultado matriz de confusão RFC *Tuning*

Interpretando a matriz de confusão da Figura 142, temos um suporte de 2309 não venda, onde o modelo *RandomForestClassifier* conseguiu prever corretamente 1471 não venda (TP) e os demais 838 previu como venda (FN).

Para a venda, temos um suporte de 966, onde o modelo *RandomForestClassifier* conseguiu prever 303 como não venda (FP) e mais 663 como venda (TN).

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

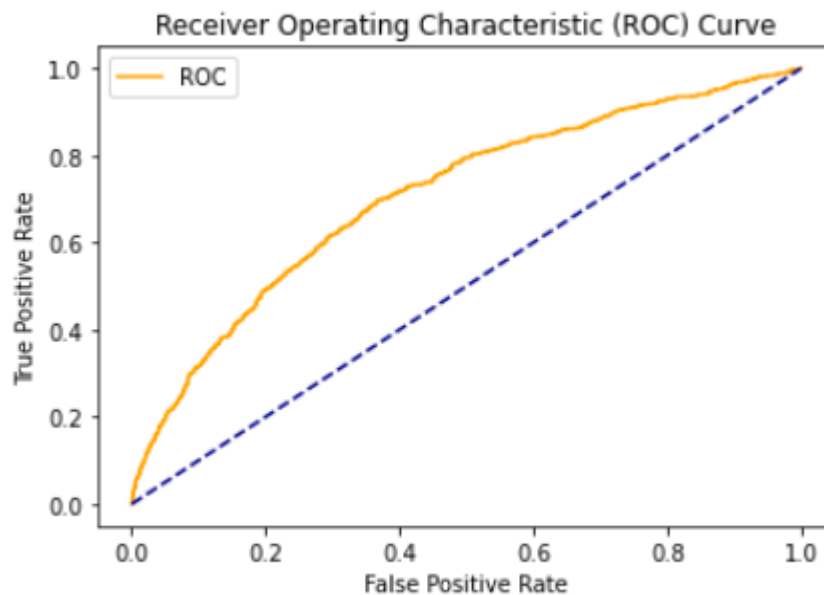
     0       0.83         0.64         0.72       2309
     1       0.44         0.69         0.54         966

 accuracy                   0.65       3275
 macro avg              0.64         0.66         0.63       3275
 weighted avg           0.71         0.65         0.67       3275
```

Figura 143 – Resultado métricas RFC *Tuning*

Para auxiliar no entendimento da matriz de confusão, podemos observar o *classification report* na Figura 143, que nos confirma toda interpretação acima, realizada sobre a predição do modelo *RandomForestClassifier* para as classes da variável venda, onde a não venda é (Classe sem fraude “0”) e venda é (Classe fraudulenta “1”).

Plot ROC curve RFC



AUC: 0.71

Figura 144 – Roc curve RFC

A área coberta pela curva é a área entre a linha laranja (ROC) e o eixo. Esta área coberta é AUC. Quanto maior a área coberta, melhor os modelos de aprendizado de máquina distinguem as classes dadas.

O valor ideal para AUC é 1, mas um bom classificador terá AUC acima dos 0.90% e AUC para RFC é de 0.71%.

Interpretando a Figura 144, o *True Positive Rate* (TPR), até os primeiros 50%, aumenta também o *False Positive Rate* (FPR) 20% (a estimativa com é prevista corretamente e temos as classes verdadeiros positivos (não venda) e falsos positivos (venda)), mas depois incorre em um FPR de 60% para atingir um TPR de abaixo dos 80%.

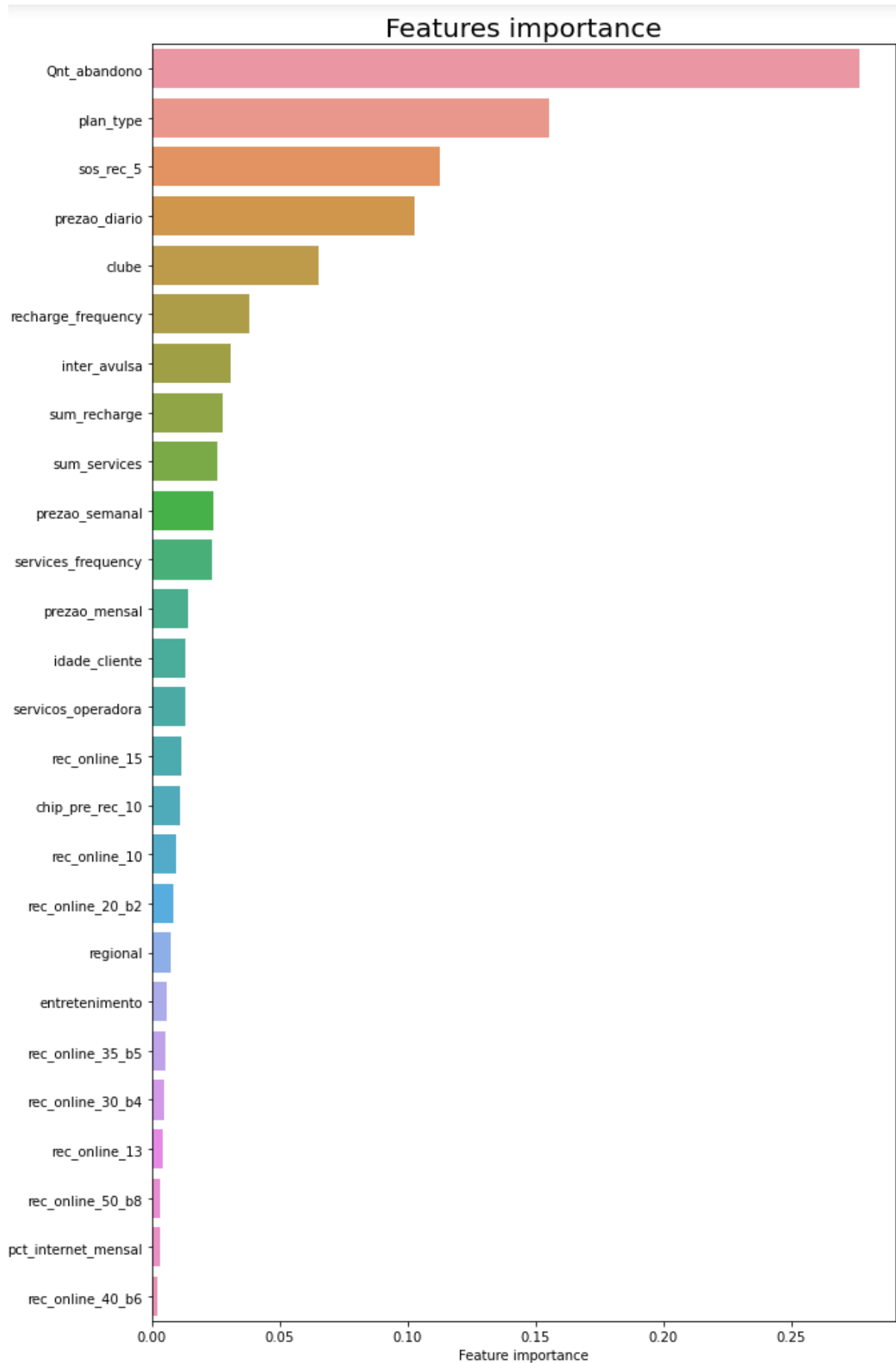


Figura 145 – Importância das *features* do *dataset* final

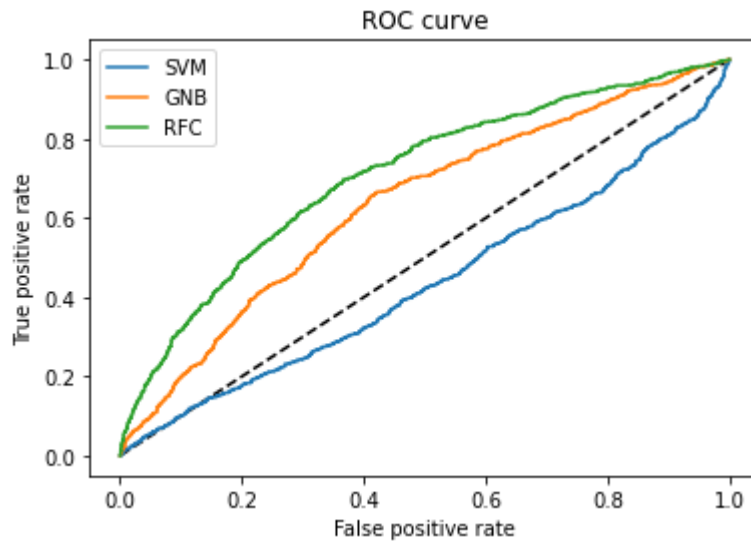


Figura 146 – Compare ROC Curve (all models)

A área coberta pela curva é a área entre as linhas laranja (GNB), verde (RFC) e azul (SVM) (ROC) e o eixo. Esta área coberta é AUC. Quanto maior a área coberta, melhor os modelos de aprendizado de máquina distinguem as classes dadas.

O valor ideal para AUC é 1 e AUC, mas um bom classificador terá AUC acima dos 0.90% ou próximo e AUC para os modelos SVM está com valor 0.44, GNB com 0.63 e o RFC é de 0.71.

O *RandomForestClassifier* é o classificador que chega mais próximo de um bom classificador para o nosso estudo.

7. Apresentação dos Resultados











<div><div>PREDICTION TASK</div><div></div><div>Score de predição da venda para Leads.</div></div>	<div><div>DECISIONS</div><div></div><div>O resultado do modelo é a predição da venda. Essa predição possibilita as equipes de vendas, planejamento e marketing priorizar os Leads com maior interesse nos pacotes de internet móvel.</div></div>	<div><div>VALUE PROPOSITION</div><div></div><div>O score da predição da venda é atribuído ao seu respectivo Lead em um document database NoSQL, onde as equipes de vendas, planejamento e marketing irão consumir as informações na estratégia omnichannel.</div></div>	<div><div>DATA COLLECTION</div><div></div><div>O conjunto de treinamento inicial é derivado de um projeto piloto que pode sofrer muitas alterações no seu enriquecimento nos próximos meses, alimentando a ideia de uma atualização continua nos estudos aplicados. A entrega da funcionalidade completa teve a duração de uma Sprint. O acompanhamento dos resultados, serão acompanhados por stakeholders das equipes de DS, planejamento e marketing.</div></div>	<div><div>DATA SOURCES</div><div></div><div>Todas as informações foram obtidas do banco de dados relacional da organização.</div></div>
<div><div>IMPACT SIMULATION</div><div></div><div>Modelo entregue e implantado, não foram gerados impactos de custo/impedimentos registrados pela equipe de DS.</div></div>	<div><div>MAKING PREDICTIONS</div><div></div><div>O tempo gasto da predição é de 528 ms ± 9.39 ms per loop. Todo processamento dos dados, foram realizados de forma rápida.</div></div>	<div><div>BUILDING MODELS</div><div></div><div>Temos um modelo serializado para o ambiente de produção e foram três os modelos avaliados no estudo. 1. Suport Vector Mahine (svm) 2. Gaussian Naive Bayes (gnb) 3. RandomForestClassifier (rfc)</div></div>		<div><div>FEATURES</div><div></div><div>Variáveis preditoras de recargas, serviços (build SQL) e informações dos Leads e variável resposta venda.</div></div>
<div><div>MONITORING</div><div></div><div>As técnicas de validação cruzada e redução de dimensionalidade, bem como as métricas, foram utilizadas para a avaliação e validação dos modelos na continuidade do estudo. StratifiedKFold, MinMaxScale, Matriz de confusão, Acurácia, Precisão, Recall, F1 – Score e AUC.</div></div>				

Tabela 2 – Machine Learning Canvas

8. Links

Link para o vídeo: <https://youtu.be/DknwV1jVFdo>

Repositório: <https://github.com/adrianofonseca1000/PucDataScienceTcc>

REFERÊNCIAS

FOSTER, P.; TOM, F. Data Science para negócios. Rio de Janeiro: ALTA BOOKS, 2016