Notebook Jupyter 3_classificarAplicação

Classificação dos modelos de motocicleta a partir da descrição

A grande dificuldade na tarefa de análise de valores compatíveis na importação de peças de motocicletas, em especial dos kits de transmissão, se dá no fato de que milhares de importadores adquirem essas peças no exterior e informam sua descrição em um campo texto livre.
Nem mesmo a utilização da classificação fiscal normatizada no Mercosul, chamada de Nomenclatura Comum do Mercosul – NCM, ajuda nesse caso específico, tendo em vista que grande parte das peças de motocicletas e todos os kits de transmissão são classificados em uma mesma posição na tabela da NCM.
Para que se possa tratar corretamente o dataset obtido na nossa etapa de processamento e tratamento de dados, e permitir o futuro aprendizado de máquina, com predições do modelo de motocicleta que aquele item se aplica, é preciso que primeiro se proceda a uma classificação de aplicações que futuramente será utilizado em um aprendizado supervisionado.
A ideia é se aplicar uma busca na descrição da mercadoria pelos termos conhecidos de aplicações e se buscar a qual aplicação aquela descrição se refere, fazendo desse modo a primeira classificação.
Posteriormente será utilizado um algorítimo de aprendizado de máquina para aprender com o próprio texto da descrição da aplicação e fazer a classificação utilizando a descrição já limpa de stopwords e outros termos desnecessários.
A interseção dos dois conjuntos de classificação será o dataset utilizado para fazer o treinamento do classificador, que será o primeiro passo antes da análise de valor do item importado.

Importa as bibliotecas necessárias

In [1]:
```
import pandas as pd, numpy as np
import re, time
# stopwords
from nltk.corpus import stopwords
# wordcloud
from wordcloud import WordCloud
# plotagem do gráfico
import matplotlib.pyplot as plt
```

In [2]:
```
# Data e hora da execução do script
initot=time.time()
print(f'Código executado em {time.strftime("%d/%m/%Y às %H:%M", time.localtime(tim
e.time()))}')
```

```
Código executado em 25/01/2022 às 12:39
```

Importa os dados já tratados

In [3]:
```
# Importa base de dados para um dataframe
df = pd.read_excel(r'./bases/dataframe.xlsx')
```

In [4]:
```
# Verifica o tamnanho do dataframe
df.shape
```

Out[4]:
```
(18276, 3)
```

In [5]:
```
# Mostra linhas de exemplo do dataframe
df.sample(5)
```

Out[5]:

	PAIS DE ORIGEM	DESCRICAO DO PRODUTO	VALOR UN.PROD.DOLAR
674	CHINA, REPUBLICA POP	KIT DE TRANSMISSAO , MARCA RIFFEL, TITANIUM (1...	6.728654
14354	CHINA, REPUBLICA POP	24730 - 71790 - KIT DE TRANSMISSAO PARA MOTOCI...	4.702000
4844	CHINA, REPUBLICA POP	10530021 IN KIT TRANSMISSÃO P/MOTOCICLETAS(COR...	3.573000
10699	CHINA, REPUBLICA POP	ITEM 14893 - KIT TRANSMISSÃO PARA MOTOCICLETA ...	38.019403
14683	CHINA, REPUBLICA POP	KIT DE TRANSMISSAO, COMPOSTO DE CORRENTE, CORO...	7.841000

In [6]:
```
df['DESCRICAO DO PRODUTO'][5]
```

Out[6]:
```
'80348 KIT DE TRANSMISSÃO, COMPOSTO DE CORRENTE, COROA E PINHÃO PARA MOTOCICLETA C
BX 250 TWISTER, MARCA ALLEN.'
```

In [7]:
```
type(df['DESCRICAO DO PRODUTO'][1])
```

Out[7]:
```
str
```

Importa as stopwords da língua portuguesa

In [8]:
```
# Importar lista de Stopwords
stopwords = set(stopwords.words('portuguese'))
```

In [9]:
```
# Mostra tamanho da lista de stopwords
len(stopwords)
```

Out[9]:
```
204
```

In [10]:
```
# Mostra toda a lista de stopwords
swtemp = list(stopwords)
swtemp.sort()
print(swtemp)
```

```
['a', 'ao', 'aos', 'aquela', 'aquelas', 'aquele', 'aqueles', 'aquilo', 'as', 'at
é', 'com', 'como', 'da', 'das', 'de', 'dela', 'delas', 'dele', 'deles', 'depois',
'do', 'dos', 'e', 'ela', 'elas', 'ele', 'eles', 'em', 'entre', 'era', 'eram', 'ess
a', 'essas', 'esse', 'esses', 'esta', 'estamos', 'estas', 'estava', 'estavam', 'es
te', 'esteja', 'estejam', 'estejamos', 'estes', 'esteve', 'estive', 'estivemos',
'estiver', 'estivera', 'estiveram', 'estiverem', 'estivermos', 'estivesse', 'estiv
essem', 'estivéramos', 'estivéssemos', 'estou', 'está', 'estávamos', 'estão', 'e
u', 'foi', 'fomos', 'for', 'fora', 'foram', 'forem', 'formos', 'fosse', 'fossem',
'fui', 'fôramos', 'fôssemos', 'haja', 'hajam', 'hajamos', 'havemos', 'hei', 'houv
e', 'houvemos', 'houver', 'houvera', 'houveram', 'houverei', 'houverem', 'houverem
os', 'houveria', 'houveriam', 'houvermos', 'houverá', 'houverão', 'houveríamos',
'houvesse', 'houvessem', 'houvéramos', 'houvéssemos', 'há', 'hão', 'isso', 'isto',
'já', 'lhe', 'lhes', 'mais', 'mas', 'me', 'mesmo', 'meu', 'meus', 'minha', 'minha
s', 'muito', 'na', 'nas', 'nem', 'no', 'nos', 'nossa', 'nossas', 'nosso', 'nosso
s', 'num', 'numa', 'não', 'nós', 'o', 'os', 'ou', 'para', 'pela', 'pelas', 'pelo',
'pelos', 'por', 'qual', 'quando', 'que', 'quem', 'se', 'seja', 'sejam', 'sejamos',
'sem', 'serei', 'seremos', 'seria', 'seriam', 'será', 'serão', 'seríamos', 'seu',
'seus', 'somos', 'sou', 'sua', 'suas', 'são', 'só', 'também', 'te', 'tem', 'temo
s', 'tenha', 'tenham', 'tenhamos', 'tenho', 'terei', 'teremos', 'teria', 'teriam',
'terá', 'terão', 'teríamos', 'teu', 'teus', 'teve', 'tinha', 'tinham', 'tive', 'ti
vemos', 'tiver', 'tivera', 'tiveram', 'tiverem', 'tivermos', 'tivesse', 'tivesse
m', 'tivéramos', 'tivéssemos', 'tu', 'tua', 'tuas', 'tém', 'tínhamos', 'um', 'um
a', 'você', 'vocês', 'vos', 'à', 'às', 'é', 'éramos']
```

Atualiza a lista de stopwords

In [11]:
```
# Palavras a adicionar na lista de stopwords estão contidas em um arquivo csv exter
no
dfsw = pd.read_csv('./bases/stopwords.csv', encoding='ISO-8859-1')
stopwords_df=sorted(list(dfsw['stopword']))
swtemp = list(stopwords_df)
swtemp.sort()
print(swtemp)
```

```
['abaixo', 'acessorios', 'acessórios', 'aco', 'acondicionados', 'adaptavel', 'adap
tável', 'allen', 'almas', 'alta', 'am', 'anel', 'ano', 'aplicacao', 'aplication',
'aplicavel', 'aplicação', 'aplicável', 'application', 'ate', 'atitanium', 'aç', 'a
ço', 'bicicleta', 'bicycle', 'bike', 'bravo', 'cada', 'caixa', 'caixas', 'cambio',
'carbono', 'certificado', 'cever', 'chain', 'chh', 'china', 'ciclomotor', 'ciclomo
tores', 'cilindrada', 'cilindradas', 'cod', 'code', 'codigo', 'comando', 'combustã
o', 'comercial', 'comercialmente', 'commodity', 'compativel', 'compatível', 'comp
l', 'completo', 'completos', 'composto', 'composto', 'compostopor', 'compostpo',
'comum', 'condicao', 'condicoes', 'condição', 'condições', 'confeccionado', 'confo
rmidade', 'conhecido', 'conj', 'conjunto', 'conjuntos', 'constituido', 'constituti
vo', 'constituído', 'contendo', 'coposto', 'coroa', 'coroaes', 'corr', 'corrent',
'corrente', 'correntee', 'correntes', 'corrnte', 'cx', 'câmbio', 'cãmbio', 'códig
o', 'decreto', 'denominada', 'dente', 'dentes', 'descricao', 'descrição', 'descriç
ao', 'descrição', 'destaque', 'destaque', 'destaques', 'detransmissão', 'diante',
'dimensao', 'dimensoes', 'dimensão', 'dimensões', 'diverso', 'diversos', 'dominad
o', 'durabilidade', 'elo', 'elos', 'embalagem', 'engine', 'engrenagem', 'engrenage
ns', 'epinhao', 'epinhão', 'especifico', 'específico', 'espessura', 'evol', 'exclu
sivo', 'fabr', 'fabri', 'fabricada', 'fabricado', 'final', 'foiproduzida', 'formad
o', 'funcao', 'funcao', 'função', 'função', 'funçao', 'função', 'gtin', 'hardt',
'ho', 'hp', 'imetro', 'in', 'incluindo', 'incluso', 'indicado', 'ingles', 'inmetr
o', 'inv', 'invoice', 'iron', 'item', 'jc', 'ki', 'kif', 'kit', 'kitr', 'kittr',
'kittr', 'kmc', 'ligacoes', 'ligaçoes', 'ligaçoes', 'ligações', 'marca', 'mark',
'match', 'material', 'materialdo', 'maxx', 'medida', 'medidas', 'medindo', 'meta
l', 'mini', 'mod', 'modelo', 'modelos', 'moto', 'motociclet', 'motocicleta', 'moto
cicletas', 'motoneta', 'motonetas', 'motoparts', 'motor', 'motorcicleta', 'motos',
'motos', 'movimento', 'nbsp', 'ncm', 'nome', 'nopinh', 'normais', 'nova', 'novo',
'nr', 'numero', 'número', 'onde', 'or', 'origem', 'oring', 'ox', 'papelao', 'papel
ão', 'part', 'parte', 'partes', 'parts', 'pc', 'pc-coroa', 'pc-corrente', 'pc-pinh
ao', 'pcs', 'pec', 'pecas', 'perfil', 'peç', 'peças', 'pinh', 'pinhao', 'pinhão',
'posição', 'premium', 'procedencia', 'procedência', 'prodepe', 'produto', 'própri
o', 'pç', 'qdes', 'qtd', 'qtds', 'qty', 'quadriciclo', 'quadriciclos', 'quantida
d', 'quantidade', 're', 'ref', 'reforcada', 'reforçada', 'registro', 'rel', 'relac
ao', 'relação', 'reposicao', 'resp', 'respo', 'respons', 'responsa', 'responsav',
'responsave', 'responsavel', 'responsáv', 'responsável', 'ret', 'retalho', 'retent
or', 'riffel', 'ring', 'roda', 'sae', 'scud', 'semi', 'semi-', 'semi-kit', 'send
o', 'serve', 'set', 'shipping', 'sistema', 'sm', 'sprocket', 'standard', 'standar
t', 'standartt', 'std', 'stdmodelo', 'steel', 'tambem', 'também', 'tec', 'temp',
'temperado', 'tipo', 'tipos', 'titanio', 'titaniu', 'titanium', 'titaniun', 'tr',
'tracao', 'tracão', 'trans', 'transmis', 'transmisao', 'transmiss', 'transmissa',
'transmissao', 'transmission', 'transmissão', 'transmitir', 'traseira', 'traçao',
'tração', 'und', 'unds', 'unid', 'unidade', 'unidade', 'unidades', 'unifort', 'u
o', 'uso', 'utilizada', 'utilizadas', 'utilizado', 'utilizados', 'utilização', 've
m', 'venda', 'with', 'xy', 'year']
```

In [12]:
```
# Atualizar stopwords
stopwords.update(stopwords_df)
```

In [13]:
```
# Mostra toda a lista de stopwords
swtemp = list(stopwords)
swtemp.sort()
print(swtemp)
```

```
['a', 'abaixo', 'acessorios', 'acessórios', 'aco', 'acondicionados', 'adaptavel',
'adaptável', 'allen', 'almas', 'alta', 'am', 'anel', 'ano', 'ao', 'aos', 'aplicaca
o', 'aplication', 'aplicavel', 'aplicação', 'aplicável', 'application', 'aquela',
'aquelas', 'aquele', 'aqueles', 'aquilo', 'as', 'ate', 'atitanium', 'até', 'aç',
'aço', 'bicicleta', 'bicycle', 'bike', 'bravo', 'cada', 'caixa', 'caixas', 'cambi
o', 'carbono', 'certificado', 'cever', 'chain', 'chh', 'china', 'ciclomotor', 'cic
lomotores', 'cilindrada', 'cilindradas', 'cod', 'code', 'codigo', 'com', 'comand
o', 'combustão', 'comercial', 'comercialmente', 'commodity', 'como', 'compativel',
'compatível', 'compl', 'completo', 'completos', 'composto', 'compostopor', 'compos
tpo', 'comum', 'condicao', 'condicoes', 'condição', 'condições', 'confeccionado',
'conformidade', 'conhecido', 'conj', 'conjunto', 'conjuntos', 'constituido', 'cons
titutivo', 'constituído', 'contendo', 'coposto', 'coroa', 'coroaes', 'corr', 'corr
ent', 'corrente', 'correntee', 'correntes', 'corrnte', 'cx', 'câmbio', 'cãmbio',
'código', 'da', 'das', 'de', 'decreto', 'dela', 'delas', 'dele', 'deles', 'denomin
ada', 'dente', 'dentes', 'depois', 'descricao', 'descrição', 'descriçao', 'descriç
ão', 'destaque', 'destaques', 'detransmissão', 'diante', 'dimensao', 'dimensoes',
'dimensão', 'dimensões', 'diverso', 'diversos', 'do', 'dominado', 'dos', 'durabili
dade', 'e', 'ela', 'elas', 'ele', 'eles', 'elo', 'elos', 'em', 'embalagem', 'engin
e', 'engrenagem', 'engrenagens', 'entre', 'epinhao', 'epinhão', 'era', 'eram', 'es
pecifico', 'específico', 'espessura', 'essa', 'essas', 'esse', 'esses', 'esta', 'e
stamos', 'estas', 'estava', 'estavam', 'este', 'esteja', 'estejam', 'estejamos',
'estes', 'esteve', 'estive', 'estivemos', 'estiver', 'estivera', 'estiveram', 'est
iverem', 'estivermos', 'estivesse', 'estivessem', 'estivéramos', 'estivéssemos',
'estou', 'está', 'estávamos', 'estão', 'eu', 'evol', 'exclusivo', 'fabr', 'fabri',
'fabricada', 'fabricado', 'final', 'foi', 'foiproduzida', 'fomos', 'for', 'fora',
'foram', 'forem', 'formado', 'formos', 'fosse', 'fossem', 'fui', 'funcao', 'funcã
o', 'funçao', 'função', 'fôramos', 'fôssemos', 'gtin', 'haja', 'hajam', 'hajamos',
'hardt', 'havemos', 'hei', 'ho', 'houve', 'houvemos', 'houver', 'houvera', 'houver
am', 'houverei', 'houverem', 'houveremos', 'houveria', 'houveriam', 'houvermos',
'houverá', 'houverão', 'houveríamos', 'houvesse', 'houvessem', 'houvéramos', 'houv
éssemos', 'hp', 'há', 'hão', 'imetro', 'in', 'incluindo', 'incluso', 'indicado',
'ingles', 'inmetro', 'inv', 'invoice', 'iron', 'isso', 'isto', 'item', 'jc', 'já',
'ki', 'kif', 'kit', 'kitr', 'kittr', 'kmc', 'lhe', 'lhes', 'ligacoes', 'ligaçoes',
'ligações', 'mais', 'marca', 'mark', 'mas', 'match', 'material', 'materialdo', 'ma
xx', 'me', 'medida', 'medidas', 'medindo', 'mesmo', 'metal', 'meu', 'meus', 'minh
a', 'minhas', 'mini', 'mod', 'modelo', 'modelos', 'moto', 'motociclet', 'motocicle
ta', 'motocicletas', 'motoneta', 'motonetas', 'motoparts', 'motor', 'motorciclet
a', 'motos', 'movimento', 'muito', 'na', 'nas', 'nbsp', 'ncm', 'nem', 'no', 'nom
e', 'nopinh', 'normais', 'nos', 'nossa', 'nossas', 'nosso', 'nossos', 'nova', 'nov
o', 'nr', 'num', 'numa', 'numero', 'não', 'nós', 'número', 'o', 'onde', 'or', 'ori
gem', 'oring', 'os', 'ou', 'ox', 'papelao', 'papelão', 'para', 'part', 'parte', 'p
artes', 'parts', 'pc', 'pc-coroa', 'pc-corrente', 'pc-pinhao', 'pcs', 'pec', 'peca
s', 'pela', 'pelas', 'pelo', 'pelos', 'perfil', 'peç', 'peças', 'pinh', 'pinhao',
'pinhão', 'por', 'posição', 'premium', 'procedencia', 'procedência', 'prodepe', 'p
roduto', 'próprio', 'pç', 'qdes', 'qtd', 'qtds', 'qty', 'quadriciclo', 'quadricicl
os', 'qual', 'quando', 'quantidad', 'quantidade', 'que', 'quem', 're', 'ref', 'ref
orcada', 'reforçada', 'registro', 'rel', 'relacao', 'relação', 'reposicao', 'res
p', 'respo', 'respons', 'responsa', 'responsav', 'responsave', 'responsavel', 'res
ponsáv', 'responsável', 'ret', 'retalho', 'retentor', 'riffel', 'ring', 'roda', 's
ae', 'scud', 'se', 'seja', 'sejam', 'sejamos', 'sem', 'semi', 'semi-', 'semi-kit',
'sendo', 'serei', 'seremos', 'seria', 'seriam', 'serve', 'será', 'serão', 'seríamo
s', 'set', 'seu', 'seus', 'shipping', 'sistema', 'sm', 'somos', 'sou', 'sprocket',
'standard', 'standart', 'standartt', 'std', 'stdmodelo', 'steel', 'sua', 'suas',
'são', 'só', 'tambem', 'também', 'te', 'tec', 'tem', 'temos', 'temp', 'temperado',
'tenha', 'tenham', 'tenhamos', 'tenho', 'terei', 'teremos', 'teria', 'teriam', 'te
rá', 'terão', 'teríamos', 'teu', 'teus', 'teve', 'tinha', 'tinham', 'tipo', 'tipo
s', 'titanio', 'titaniu', 'titanium', 'titaniun', 'tive', 'tivemos', 'tiver', 'tiv
era', 'tiveram', 'tiverem', 'tivermos', 'tivesse', 'tivessem', 'tivéramos', 'tivés
semos', 'tr', 'tracao', 'tração', 'trans', 'transmis', 'transmisao', 'transmiss',
'transmissa', 'transmissao', 'transmission', 'transmissão', 'transmitir', 'traseir
a', 'traçao', 'tração', 'tu', 'tua', 'tuas', 'tém', 'tínhamos', 'um', 'uma', 'un
d', 'unds', 'unid', 'unidade', 'unidades', 'unifort', 'uo', 'uso', 'utilizada', 'u
tilizadas', 'utilizado', 'utilizados', 'utilização', 'vem', 'venda', 'você', 'você
s', 'vos', 'with', 'xy', 'year', 'à', 'às', 'é', 'éramos']
```

In [14]:
```
len(stopwords)
```

Out[14]:
```
518
```

Carrega lista de aplicações

In [15]:
```
# carrega a lista de marcas de motos do arquivo
dftemp=pd.read_csv('./bases/Aplicacoes.csv')
```

In [16]:
```
dftemp.head()
```

Out[16]:

	APLICACOES
0	ACELLERA ACX 250F 250
1	ACELLERA FRONTLANDER 500
2	ACELLERA FRONTLANDER 800 EFI
3	ACELLERA HOTZOO SPORT 90
4	ACELLERA QUADRILANDER 300

Cria a lista de Palavras Chave das Aplicações

In [17]:
```
# remove caracteres especiais ou soltos e termos duplicados, salvando na lista
palavrasChave=sorted(set(re.sub(r"\b \w \b",
                                ' ',
                                re.sub(r"[/<>()|\+\$%&#@\'\"]+",
                                       " ",
                                       " ".join(dftemp['APLICACOES'].tolist()))).sp
lit()))
```

In [18]:
```
len(palavrasChave)
```

Out[18]:
```
894
```

In [19]:
```
# amostra de palavrasChave
print(palavrasChave[:20],' ... ', palavrasChave[-20:])
```

```
['1000', '1000F', '1000R', '1000V', '1098', '1100', '1100XX', '110S', '1125', '119
0', '1198', '1200', '1200Z', '125', '125R', '130', '1300', '135', '1400', '150']
...  ['YFS', 'YS150', 'YS250', 'YZ', 'YZF', 'YZR', 'Z1000', 'Z750', 'Z800', 'ZAC
H', 'ZANELLA', 'ZENITH', 'ZIG', 'ZING', 'ZIP', 'ZONGSHEN', 'ZR', 'ZRX', 'ZS', 'ZZ
R']
```

In [20]:
```
'RT' in palavrasChave
```

Out[20]:
```
True
```

Limpeza e criação da coluna DESCRICAO

Função de limpeza de dados irrelevantes para a classificação e remoção de stopwords

In [21]:

```python
def limpaDescricao(descricao): #
    descricao=descricao.lower() #transformar em minúsculas
    # remove top (1045) e variantes
    descricao=re.sub(r'\b[ (-]*top \( *1045 *\)[-) ]*\b',' ',descricao)
    # remove códigos numéricos entre parênteses com -*/
    descricao=re.sub(r"\( *\d*[\/\*\-\d]*\d* *\)", ' ', descricao)
    # remove a ocorrência de "código e etc." e o termo seguinte começado com número
    # att: (alguns tem hífen ou asterisco) (colocar antes de remover pontuação)
    descricao=re.sub(r"\b(invoice|código|codigo|cod|cód|(certificado|cert)( no|nr|)|ref)[0-9a-z/\-\*\.\:]* *\d[^ ]+", ' ', descricao)
    # remove identificação de referência de engrenagens dos kits (antes da pontuação)
    #descricao=re.sub(r"([^a-z]|)(ho|uo|h|l|t|ktd|sm|m|d| x|elos )\d{1,}[ x\-\/,);.]|[ x\-\/()*\d{1,}(ho|uo|h|l|z|t|ktd|m|d|x| dentes| elos)[ \-\/,;)]", ' ', descricao) # 00h
    descricao=re.sub(r"\d*(ho|uo|h|l|t|ktd|sm|m|d|elos )\d{1,}[ \-\/,);.]|[ \-\/()*\d{1,}(ho|uo|h|l|z|t|ktd|m|d| dentes| elos)", ' ', descricao) # 00h
    # substitui os termos "s/re" e "s/ret" por "sem retentor"
    descricao=re.sub(r"\b(s\/re|s\/ret)\b", 'sem retentor', descricao)
    # substitui os termos "c/re" ou "c/ret" por "com retentor"
    descricao=re.sub(r"\b(c\/re|c\/ret)\b", 'com retentor', descricao)
    # substitui o termo "aplicação" e "modelo" emendado com outro
    descricao=re.sub(r"aplicacao", "aplicacao ", descricao)
    descricao=re.sub(r"modelo", "modelo ", descricao)
    # remove códigos no início da descrição
    descricao=re.sub(r"^\b\d{2,}[^ ]*\b", ' ', descricao)
    descricao=re.sub(r"^k[^ ]+", ' ', descricao)
    descricao=re.sub(r"- | -|[\\\+,.:;!?/]+", ' ', descricao) #remover pontuação (att: "- " ou " -")
    #correção de erros de digitação comuns
    termos={'titan': ['titian','tita','tintan','tit'],
            'honda': ['hond','hnda','hon'],
            'twister': ['twist', 'twiste'],
            'dafra kansas': ['dafra kan'],
            'tenere': ['tener','tenerre'],
            'broz': ['bros','bross'],
            'titan 150': ['titan150'],
            'broz 150': ['bross125.','bros125.','broz125','bross150.','bros150.','broz150'],
            'pop 100':['pop100'],
            'phoenix':['phoeni','phenix'],
            'c100': ['c 100']}
    for termo in termos:
        for termoerrado in termos[termo]:
            descricao=re.sub(r"\b"+termoerrado+r"\b", termo, descricao)
    descricao=re.sub(r"[/<>()|\+\\$%&#@\'\"]+", ' ', descricao) #remover carcteres especiais
    # remove a ocorrência de medidas tipo 00x000x00 ou 000x0000
    descricao=re.sub(r"\b\d{1,}(x|\*)\d{1,}(x|\*)\d{1,}|\d{1,}(x|\*)\d{1,}\b", ' ', descricao)
    # remove identificação de quantidades, unidades, peças e conjuntos
    descricao=re.sub(r"\b(\d* *(conj|und|uni|pc|pç|pec|peç)( \w|\w)+?)\b", ' ', descricao)
    # remove identificação de mais de 4 dígitos com ou sem letras no início e no final
    descricao=re.sub(r"\w+\d{4,}\w+", ' ', descricao)
    # remove números de 4 dígitos ou mais começados de 2 a 9
    descricao=re.sub(r"\b[02-9]\d{3,}\b", ' ', descricao)
    # remove identificação de termos começados por zero
    descricao=re.sub(r"\b0\d*\w+?(?=\b)", ' ', descricao)
    # remove a ocorrência de "marca " e o termo na lista até o próximo espaço
    for marca in ['kmc *gold','am *gold','king','bravo *racing','riffel *top']:
        descricao=re.sub(r"\bmarca[ :\./]*"+str(marca)+r"[^ ]*", ' ', descricao) # colocar antes das stopwords
    descricao=re.sub(r"marca[ :\./]*\w+", ' ', descricao)
    descricao=re.sub(r"(^-| -|- )", ' ', descricao)
    # remove stopwords mantendo a ordem original da descrição
    descricao=list(dict.fromkeys(descricao.split())) # cria lista com termos únicos
    descricao=" ".join([x for x in descricao if x not in set(stopwords)]) # exclui stopwords
```

```
    # limpa os número que não estão na lista de aplicações (colocar depois das stop words)
    desc=descricao.upper().split() # quebra a descrição
    dif=list(set(descricao.upper().split()).difference(palavrasChave)) # pega os termos diferentes de palavrasChave
    [desc.remove(x) for x in desc if (x in dif and x.isnumeric())] # exclui de desc os termos numéricos diferentes
    descricao=" ".join(desc).lower() # volta para texto
    #remover hífen, letras ou números soltos (deixar duplicado mesmo)
    descricao=re.sub(r"(^-| -|- |\b\w\b)", ' ', descricao)
    descricao=re.sub(r"(^-| -|- |\b\w\b)", ' ', descricao)
    #substitui remove o i das cilindradas: ex.: 125i por 125
    termos=re.findall(r"\d{1,}i\b",descricao)
    if termos:
        for termo in termos:descricao=descricao.replace(termo,termo[:-1])
    # remove espaços em excesso (colocar no final)
    descricao=re.sub(r" {2,}", ' ', descricao)
    descricao=descricao.strip()
    # retorna a descricao como saída da função
    return descricao # retorna a descrição
```

Exemplo de execução da função

In [22]:
```
linha=345
```

In [23]:
```
df.iloc[linha]['DESCRICAO DO PRODUTO']
```

Out[23]:
```
'880374 - KIT DE TRANSMISSÃO, COMPOSTO DE CORRENTE, COROA E PINHÃO, UTILIZADO NAS MOTOS SHINERAY XY50CC - MARCA ALLEN'
```

In [24]:
```
limpaDescricao(df.iloc[linha]['DESCRICAO DO PRODUTO'])
```

Out[24]:
```
'shineray xy50cc'
```

Execução da função para criação da coluna DESCRICAO limpa

In [25]:
```
ini=time.time()
df['DESCRICAO']=df['DESCRICAO DO PRODUTO'].apply(limpaDescricao)
fim=time.time()
print(f'Tempo de execução: {fim-ini:.2} segundos.')
```

```
Tempo de execução: 8.0 segundos.
```

In [26]:
```
df.sample(5)
```

Out[26]:

	PAIS DE ORIGEM	DESCRICAO DO PRODUTO	VALOR UN.PROD.DOLAR	DESCRICAO
12676	CHINA, REPUBLICA POP	KIT DE TRANSMISSAO , MARCA RIFFEL, TITANIUM (1...	3.97000	cg 125 cargo fan
15550	CHINA, REPUBLICA POP	KIT TRANSMISSÃO, COMPOSTO DE CORRENTE, COROA E...	4.32200	titan 125
12994	CHINA, REPUBLICA POP	KIT DE TRANSMISSÃO EM AÇO 1045 PARA USO EM MOT...	3.41413	biz 125
2028	CHINA, REPUBLICA POP	COD.: 0119 - KIT DE TRANSMISSAO, MARCA BRANDY,...	3.73800	honda cg reg z43 z14 excl
7869	MALASIA	21010048 KIT TRANSMISSAO P/MOTOCICLETAS(COROA,...	7.21000	xre300 rk

In [27]:
```
df['DESCRICAO'].iloc[linha]
```

Out[27]:
```
'shineray xy50cc'
```

Criação de colunas Modelo

Função de determinação de palavras chave na coluna Modelo

In [28]:
```
def achaPalavraChave(descricao):
    palavras=[]
    descricao=descricao.upper()
    desc=descricao.split()
    for palavra in palavrasChave:
        if palavra in desc:
            palavras.append(palavra)
        else:
            if palavra.isnumeric():
                pat=r"[0-9]*"+str(palavra)+r"[0-9]*"
            elif palavra.isalpha():
                pat=r"[A-Z]*"+str(palavra)+r"[A-Z]*"
            else:
                pat=r"\b"+palavra+r"\b"
            a = re.findall(pat,descricao)
            if len(a)>0:
                 # adiciona resultado nas palavras se o resultado estiver em palavr
asChave
                palavras+=[a[i] for i in range(len(a)) if a[i] in palavrasChave]
    palavras=list(set(palavras)) # remove duplicados
    palavras=" ".join(palavras) # converte para string
    return palavras.lower()
```

In [29]:
```
achaPalavraChave(limpaDescricao(df['DESCRICAO DO PRODUTO'].iloc[linha]))
```

Out[29]:
```
'xy 50 shineray'
```

Função para acrescentar a marca da motocicleta

In [30]:
```
# termos que iniciam item da descrição correspondem a marca
# As que começam com espaço devem permanecer assim, pois há outros modelos com o me
smo final
Marcas = {'HONDA': ['CG','CD','CBX','CB','CBR','CRF','BIZ','BROS','BROZ','XL',' FA
N','XR','XRE'
                    'DREAM','TITAN','TODAY','TWIN','POP','NX','NXR','TWISTER', 'HOR
NET',
                    'AMERICA','BOLDOR','DUTY','FIREBLADE','FURY','WING','LEAD','MAG
NA','NL',
                    ' NC','NSR','NC','NXR','PACIFIC','COAST','SHADOW',' STRADA','ST
UNNER','HAWK',
                    'SUPERBLACKBIRD','TORNADO','TURUNA','XRV','AFRICA','VALKYRIE',
'VARADERO',
                    'VFR','VLR','VTR','VTX','TRANSALP'],
          'YAMAHA': ['AEROX','ALBA','AXIS','BWS','DRAG ','DT','FZ','FJ',' RD','TENE
RE',
                     'MT','XF','XJ','XS','XT','XZ','YF','YZ','LANDER','GLADIATOR',
'GRIZZLY',
                     'YBR','YZ','VIRAGO','FACTOR','EC','CRYPTON','FAZER','JOG',' LA
NDER',
                     'FROG','LIBERO','MAJESTY','MEST','MIDNIGHT','MORPH','NEO','PAS
SOL'],
          'DAFRA': ['APACHE','CITYCOM','KANSAS','LASER','NEXT','RIVA','ROADWIN','ZI
G','SPEED'],
          'SUZUKI': ['KATANA','YES','INTRUDER'],
          'ZONGSHEN': ['ZS'],
          'KASINSKI': ['COMET','MIRAGE'],
          'POLARIS': ['SPORTSMAN','RZR','RANGER'],
          'KAWASAKI': ['NINJA','VERSYS','VOYAGER','GTR','KDX','KL','KX','KZ','ZR',
'ZZ','ER6N','ER6F'],
          'DAYANG': ['DY1','DY2','DY5'],
          'SUNDOWN': ['WEB','FIFITY','PALIO','PGO','STX','VBLADE','EVO', 'HUNTER MA
X'],
          'SHINERAY': ['BIKE','BRAVO','DISCOVER','EAGLE','INDIANAPOLIS','JET','NEW'
,'WAVE',
                       'STRONG','SUPER SMART','VENICE',' XY']}
```

In [31]:
```
# Função para pegar a chave pelo valor, dado que valor é único.
def pegaChave(v, dict):
    for chave, valores in dict.items():
        if type(valores)!=type([1,2]):
            valores=[valores]
        for valor in valores:
            if v == valor:
                return chave
    return "Não existe chave para esse valor."
```

In [32]:
```
def acrescentaMarca(descricao):
    for marca in Marcas:
        if re.search(marca,descricao.upper()):
            descricao += " "+marca
        for termo in Marcas[marca]:
            t1=termo.split()
            if len(t1)>1:
                pat=r"(?:"+t1[0]+r"|"+t1[1]+r").*(?:"+t1[0]+r"|"+t1[1]+r")"
            elif len(termo)<3:
                pat=termo+r"([0-9]{1,}|\b)"
            else:
                pat=termo
            resultados = re.findall(pat,descricao.upper())
            if resultados:
                descricao += " "+marca
                descricao += " "+" ".join(resultados)
                descricao += " "+termo
    descricao=" ".join(sorted(set(descricao.lower().split())))
    return descricao
```

In [33]:
```
acrescentaMarca(achaPalavraChave(limpaDescricao(df['DESCRICAO DO PRODUTO'].iloc[lin
ha])))
```

Out[33]:
```
'50 shineray xy'
```

Aplica as funções

Tenha paciência, demora cerca de 1 minuto para cada mil registros.

In [34]:
```
# cria as colunas
df=df.assign(Modelo=df['DESCRICAO'])
df.iloc[linha]['DESCRICAO']
```

Out[34]:
```
'shineray xy50cc'
```

In [35]:
```
df.iloc[:,-2:].sample(5)
```

Out[35]:

	DESCRICAO	Modelo
17057		
772	xre 300 top	xre 300 top
2376	darom xtz lander 250 tenere	darom xtz lander 250 tenere
6919	fazer 250	fazer 250
4605	darom biz 13	darom biz 13

In [36]:
```
# aplica as funções a cada coluna
ini=time.time()
now = time.strftime("%H:%M", time.localtime(time.time()))
print("Hora de início:" + now)
print(f"Tempo estimado de execução: {df.shape[0]//1000} minutos.") # 1000 registros
por minuto

print('\nBuscando palavras chave... Aguarde...')
df['Modelo']=df['Modelo'].apply(achaPalavraChave)

print('\nBuscando marcas... Aguarde...')
df['Modelo']=df['Modelo'].apply(acrescentaMarca)

now = time.strftime("%H:%M", time.localtime(time.time()))
fim=time.time()
print("\nHora de término:" + str(now))
print("Tempo decorrido: " + str(round((fim-ini)/60,2)) + " minutos.")
```

```
Hora de início:12:40
Tempo estimado de execução: 18 minutos.

Buscando palavras chave... Aguarde...

Buscando marcas... Aguarde...

Hora de término:12:57
Tempo decorrido: 17.16 minutos.
```

In [37]:
```
df['DESCRICAO DO PRODUTO'].iloc[linha]
```

Out[37]:
```
'880374 - KIT DE TRANSMISSÃO, COMPOSTO DE CORRENTE, COROA E PINHÃO, UTILIZADO NAS
MOTOS SHINERAY XY50CC - MARCA ALLEN'
```

In [38]:
```
df[['DESCRICAO DO PRODUTO','DESCRICAO','Modelo']].sample(5)
```

Out[38]:

	DESCRICAO DO PRODUTO	DESCRICAO	Modelo
3909	item .09;Partes e peças para Motocicletas,Dest...	gsr 150 11-17	150 gsr
6859	ITEM 1000235WR - KIT TRANSMISSÃO COMPOSTO DE C...	ybr 125 factor	125 factor yamaha ybr
16674	91234 - KIT DE TRANSMISSÃO PARA MOTOCICLETAS (...	crypton	crypton yamaha
16182	10540005 IN KIT TRANSMISSAO P/MOTOCICLETAS(COR...	titan 150	150 honda titan
13461	10540028 IN KIT TRANSMISSAO P/MOTOCICLETAS(COR...	fan 125	125 fan honda

In [39]:
```
df_sem_modelo = df[df['Modelo']=='']
df_sem_modelo['DESCRICAO'].to_excel("./bases/sem_modelo.xlsx")
```

In [40]:
```
df_sem_modelo[['DESCRICAO DO PRODUTO','DESCRICAO','Modelo']].sample(10)
```

Out[40]:

	DESCRICAO DO PRODUTO	DESCRICAO	Modelo
8177	KIT TRANSMISSÃO AÇO (1045), COMPOSTO DE CORREN...		
12722	KIT TRANSMISSÃO AÇO (1045), COMPOSTO DE CORREN...		
5607	TRANSMISSAO PARA USO EM MOTOCICLETA COMPOSTO D...		
1385	KIT TRANSMISSÃO AÇO (1045), COMPOSTO DE CORREN...		
10794	TRANSMISSAO PARA USO EM MOTOCICLETA COMPOSTO D...		
10770	TRANSMISSAO PARA USO EM MOTOCICLETA COMPOSTO D...		
13086	KIT TRANSMISSÃO AÇO (1045), COMPOSTO DE CORREN...		
6395	TRANSMISSAO PARA USO EM MOTOCICLETA COMPOSTO D...		
13594	KIT TRANSMISSÃO AÇO (1045), COMPOSTO DE CORREN...		
10296	-35T14T/428H106L - KIT TRANSMISSAO EM ACO COMP...		

In [41]:
```
df_sem_modelo.shape
```

Out[41]:
```
(792, 5)
```

In [42]:
```
df_sem_modelo.reset_index(drop=True, inplace=True)
```

In [43]:
```
print(f'Número de registros sem aplicação contida na descrição: {df_sem_modelo.shape[0]}')
```

```
Número de registros sem aplicação contida na descrição: 792
```

Exclusão dos registros sem aplicação contida na descrição

Neste momento é necessário tomar uma decisão sobre o que fazer com os registros que permaneceram sem nenhuma extração na coluna **Modelo**.
Para tal decisão foi necessário observar cada um desses registros no arquivo "sem_modelo.xls" exportado e constatar que nenhum dos registros possui realmente qualquer alusão à aplicação do item descrito.

In [44]:
```
df=df[df['Modelo']!='']
```

In [45]:
```
df.reset_index(drop=True, inplace=True)
```

In [46]:
```
df.shape
```

Out[46]:
```
(17484, 5)
```

Funcão final que transforma a DESCRICAO DO PRODUTO em Modelo para classificar

In [47]:
```
def criaModelo(descricao):
    descricao=limpaDescricao(descricao)
    descricao=achaPalavraChave(descricao)
    descricao=acrescentaMarca(descricao)
    return descricao
```

In [48]:
```
criaModelo(df.iloc[linha]['DESCRICAO DO PRODUTO'])
```

Out[48]:
```
'150 broz honda nxr xr'
```

Exportando DataFrame com Modelos de aplicação

In [49]:
```
df.to_excel(r'./bases/dataframe_modelos.xlsx', index = False, header = True)
```

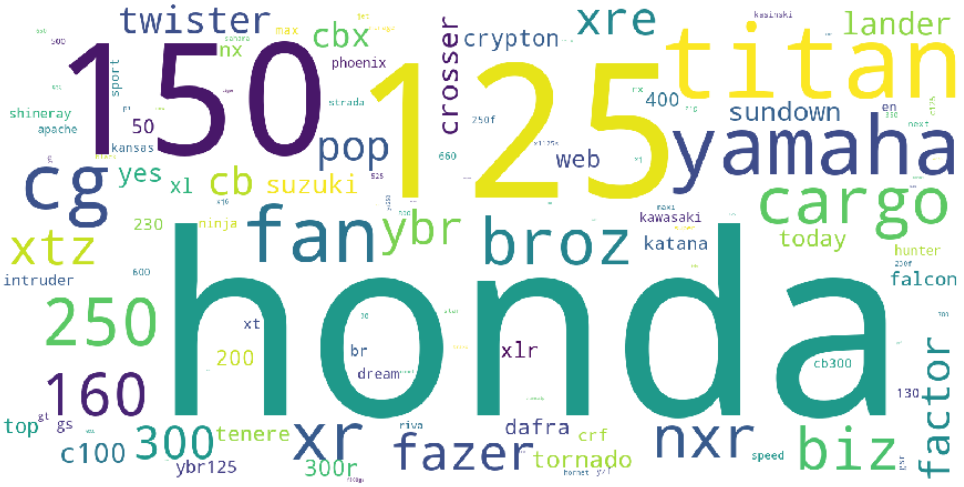
Gerando a WordCloud com o campo Modelo

In [50]:
```
# Mescla todas as descrições como uma string usado espaço como separador
descricoes = " ".join(df['Modelo']).lower()
```

In [51]:
```
# Define e gera a wordcloud para um máximo de 400 palavras de tamanho mínimo 2, sem termos duplos
wordcloud = WordCloud(stopwords=stopwords,
                      background_color="white",
                      width=1600, height=800,
                      max_words=400,
                      min_word_length=2,
                      collocations=False,
                      include_numbers=True).generate(descricoes)
```

In [52]:
```
# Exibe a imagem da nova WordCloud gerada
fig, ax = plt.subplots(figsize=(20,8))
ax.imshow(wordcloud, interpolation='bilinear')
ax.set_axis_off()
plt.imshow(wordcloud)
```

Out[52]:
```
<matplotlib.image.AxesImage at 0x1611e682a58>
```



In [53]:
```
# Exporta para um aquivo
wordcloud.to_file("./imagens/wordcloud_descricoes_final.png")
```

Out[53]:
```
<wordcloud.wordcloud.WordCloud at 0x1611ef53ac8>
```

In [54]:
```
tempotot=time.time()-initot
if tempotot>60:
    print(f'Tempo total de execução: {tempotot/60:.2f} minutos.')
else:
    print(f'Tempo total de execução: {tempotot:.2f} segundos.')
```

```
Tempo total de execução: 17.40 minutos.
```