Notebook Jupyter 8_predição Risco

Predição do Risco dada uma Importação

O objetivo agora é dada uma importação contendo **kit de transmissão de motocicletas** fazer a sua classificação e em seguida apresentar uma análise de risco visual baseada na estatística existente de outras importações.
É importante salientar que esta análise não substitui a acurada análise a ser realizada por um Auditor Fiscal, tendo em vista que o resultado será uma lista de parâmetros objetivos que indicarão graus de observação para o gerenciamento de risco. Desse modo, a entrada será o registro de importação contendo todos os campos. A função determinará a classificação e com base nos valores declarados de outras importações da mesma classificação, fará a comparação analítica com o valor da importação em análise.

Importando bibliotecas

In [1]:

```
import pandas as pd, time
import pickle
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches
```

In [2]:

```
# importa funções criadas no TCC e que precisam ser reutilizadas
from funcoesTCC import *
# Funções: criaModelo, limpaDescricao, achaPalavraChave, pegaChave, acresce
ntaMarca, retentorAux, classificaAplicacao
# Variávis: stopwords, palavrasChave, Marcas, cvt, tfi, clfsvc
# Datasets: dfaplicacoes (Aplicações)
```

In [3]:

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

```
Código executado em 20/01/2022 às 17:39
```

Carregamento do *dataset*

In [4]:

```
# Importa base de dados com os modelos já determinados para um dataframe
df = pd.read_excel(r'./bases/dataframe_modelos_classificado.xlsx')
```

Define dataset para gerar os relatórios

In [5]:

```
dfimp=df.sample(50)
dfimp.sample()
```

Out[5]:

	PAIS DE ORIGEM	DESCRICAO DO PRODUTO	VALOR UN.PROD.DOLAR	DESCRICAO	Modelo
12863	CHINA, REPUBLICA POP	20181/i45 - KIT DE TRANSMISSÃO EM AÇO 1045, MA...	3.56	ybr125	125 yamaha ybr ybr125

Teste do classificador

O classificador treinado e as funções de classificação foram importadas no módulo funcoesTCC

In [6]:

```
descricao = df.iloc[117]['DESCRICAO DO PRODUTO']
# verifica a descrição do produto
descricao
```

Out[6]:

```
'KIT DE TRANSMISSAO , MARCA RIFFEL, TITANIUM (1045) PARA MOTOC
ICLETAS XTZ 125, COMPOSTO DE CORRENTE 428 X M138 + COROA 41555
48Z + PINHAO 26571 14Z (CERTIFICADO NR. BR31512030) - ITEM: 71
817'
```

In [7]:

```
classificaAplicacao(descricao)
```

Out[7]:

```
'YAMAHA XTZ 125'
```

Importando a Tabela ABIMOTO

In [8]:

```
dfABIMOTO = pd.read_excel(r'./bases/dfABIMOTOv13.xlsx')
```

In [9]:

```
dfABIMOTO.sample()
```

Out[9]:

	PARTES E PEÇAS	VMLE	RETENTOR	APLICACAO
7	KIT TRANSMISSÃO 1045 TITAN 2000/04 14D+428HX11...	4.3	False	HONDA CG TIT TITAN 125 150 160

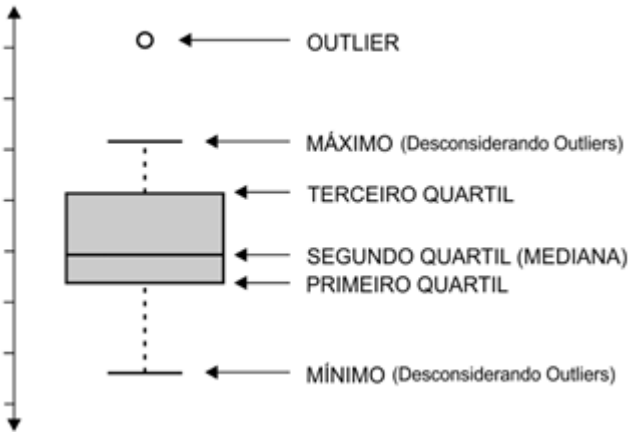
Função de busca do valor da Aplicação na Tabela ABIMOTO

In [10]:

```
def achaValorABIMOTO(aplicacao, retentor):
    # aplicação = string (aplicação da importação atual)
    # retentor = boolean (identificador da presença de retentor na importaç
ão atual)
    # retorna o valor na tabela ABIMOTO ou 0 se não existir a aplicação
    dfABtemp=dfABIMOTO[dfABIMOTO['APLICACAO']==aplicacao]
    dfABtemp=dfABtemp[dfABtemp['RETENTOR']==retentor]
    if dfABtemp.shape[0]==0:
        return 0
    return min(dfABtemp['VMLE'])
```

Função de determinação do risco

Uma observação muito importante a ser feita, foi quanto aos valores *outliers*, isto é, valores atípicos, posto que alguns itens eram discrepantes, fosse por ser uma sub ou sobrevaloração do item ou fosse porque alguns importadores insistem em, erroneamente, declarar caixas com vários kits em vez de unidades.
A identificação de *outliers*, distantes dos demais pontos da série, pode ser feita utilizando o intervalo interquartílico (IQR). Entende-se como *outlier*, valores menores que Q1-1,5*IQR ou valores maiores que Q3+1,5*IQR.
Estes valores também são representados no boxplot, mas como pontos acima ou abaixo da linha conectada à caixa. Vejamos um desenho de um boxplot com todos estes conceitos:



Apesar da bibliografia indicar o uso de uma vez e meia o intervalo entre o primeiro e o terceiro quartil para definir os *outliers*, optou-se nesse trabalho, por uma questão de segurança, em utilizar o valor de três vezes.

Função de plotagem da Previsão de Risco

In [11]:

```
def plotrisco(P, valor,valorABIMOTO=0):
    # P (lista) = percentis da base histórica da aplicação importada
    # valor (float) = valor declarado na importação analisada
    # valorABIMOTO (float) = valor de referência na tabela ABIMOTO (0, se n
ão exisitr)
    # cria o plot
    fig, ax = plt.subplots(figsize=(10, 1.25))

    # define os dados sequenciais, posição e cores dos percentis
    ax.broken_barh([(0,P[0]),(P[0],P[1]), (P[1],P[11]), (P[11],P[3]), (P[3
],P[4]), (P[4],P[5]),
                                (P[5],P[6]), (P[6],P[7]), (P[7],P[8]), (P[8],P
[9]), (P[9],P[10])],
                     [10, 9],
                    facecolors=('#c10000', 'r', 'r','#fdff01', '#00fe32', '#
02ff00',
                                '#02ff00','#02ff00', '#00fe32','#00fe32', '#
006dff'))

    # define limites dos eixos x e y
    ax.set_ylim(10,15)
    ax.set_xlim(P[0], P[10])

    # define os marcadores dos eixos
    ax.set_yticks([])
    ax.set_xticks([P[11],P[5],P[9]])
    ax.set_axisbelow(True)

    # define os textos
    # identificação dos percentis
    ax.text(P[1], 15.5, "P10%", fontsize=10, verticalalignment='bottom', ho
rizontalalignment='center',rotation='vertical')
    ax.text(P[1], 15, "|", fontsize=10, verticalalignment='center', horizon
talalignment='center')
    ax.text(P[11], 15.5, "P25%", fontsize=10, verticalalignment='bottom', h
orizontalalignment='center',rotation='vertical')
    ax.text(P[11], 15, "|", fontsize=10, verticalalignment='center', horizo
ntalalignment='center')
    ax.text(P[5], 15.5, "P50%", fontsize=10, verticalalignment='bottom', ho
rizontalalignment='center',rotation='vertical')
    ax.text(P[5], 15, "|", fontsize=10, verticalalignment='center', horizon
talalignment='center')
    ax.text(P[12], 15.5, "P75%", fontsize=10, verticalalignment='bottom', h
orizontalalignment='center',rotation='vertical')
    ax.text(P[12], 15, "|", fontsize=10, verticalalignment='center', horizo
ntalalignment='center')
    ax.text(P[9], 15.5, "P90%", fontsize=10, verticalalignment='bottom', ho
rizontalalignment='center',rotation='vertical')
    ax.text(P[9], 15, "|", fontsize=10, verticalalignment='center', horizon
talalignment='center')
    ax.text(P[10], 15.5, "P100%", fontsize=10, verticalalignment='bottom',
horizontalalignment='center',rotation='vertical')
    ax.text(P[10], 15, "|", fontsize=10, verticalalignment='center', horizo
ntalalignment='center')
```

```
    # o valor (P[10]-P[0])/40 relativiza a escala na hora de mover algo no
 gráfico.
    undrelativa = (P[10]-P[0])/40
    # identificação de mínimo, mediana e máximo
    ax.text(P[0]-1*undrelativa, 12.5, f"Min({P[0]})", fontsize=12, vertical
alignment='center', horizontalalignment='center',rotation='vertical')
    ax.text((P[5]), 8, "Mediana", fontsize=12, verticalalignment='center',
horizontalalignment='center')
    ax.text(P[10]+1*undrelativa, 12.5, f"Máx({P[10]})", fontsize=12, vertic
alalignment='center', horizontalalignment='center',rotation='vertical')

    # define alertas
    alerta=''
    # alerta quando valor fora dos limites máximo e mínimo
    if valor<P[0] or valor>P[10]:
        alerta+=f"ATENÇÃO: Valor declarado de USD{valor:.2f} fora dos limit
es do modelo (outlier)."
        alerta+='\n'
    if valorABIMOTO>valor:
        alerta+=f"ATENÇÃO: Valor declarado de USD{valor:.2f} menor que refe
rência mínima ABIMOTO: USD{valorABIMOTO:.2f}."
    ax.text((P[10]+P[0])/2, 6, alerta, fontsize=13, color='r', verticalalig
nment='center', horizontalalignment='center')

    # define o título
    fig.suptitle('Previsão de Risco', fontsize=13, y=1.5, horizontalalignme
nt='center')

    # define cores do ponteiro indicador
    cores=['#a20000','#ff6900', '#007d00', '#00007d'] # vermelho, amarelo,
 verde, azul
    # condições para determinar a cor do ponteiro
    if   valor>=P[9]: cor=cores[3]
    elif valor>=P[3]: cor=cores[2]
    elif valor>=P[11]: cor=cores[1]
    else:             cor=cores[0]

    # plota o ponteiro e o valor
    ax.scatter(x=valor, y=10.9, marker='v', c=cor, s=400)
    ax.text(valor, 12.2, valor, color='black', fontsize=14, verticalalignme
nt='center', horizontalalignment='center')
    ax.vlines(x=P[5], ymin=10, ymax=15)

    # se existir valor na tabela ABIMOTO coloca o indicador e valor
    if valorABIMOTO>0:
        ax.scatter(x=valorABIMOTO, y=10.7, marker='v', c='black', s=200)
        if valorABIMOTO>valor:
            ax.text(valorABIMOTO+1*undrelativa, 10.6, f'{valorABIMOTO:.2f}
(ABIMOTO)', color='black', fontsize=12, verticalalignment='center', horizon
talalignment='left')
        else:
            ax.text(valorABIMOTO-1*undrelativa, 10.6, f'(ABIMOTO){valorABIM
OTO:.2f}', color='black', fontsize=12, verticalalignment='center', horizont
alalignment='right')

    # mostra o gráfico
    plt.show()
```

Função que calcula o risco de valoração

In [12]:

```
def riscoValor(dfimp):
    # dfimp será uma série ou dataframe pandas contendo as importações a an
alisar
    # caso seja um dataframe e tenha mais de uma linha, será iterada sobre
 todas as linhas.
    if type(dfimp)==type(pd.Series([])):    tipo='s' # se for tipo series =
=> s
    if type(dfimp)==type(pd.DataFrame([])): tipo='d' # se for tipo datafram
e ==> d
    if not(tipo=='s' or tipo=='d'): # se o tipo não for séries ou dataframe
==> erro
        raise TypeError('É preciso entrar com um dataframe ou uma Series do
pandas.')
    if tipo=='d': # se for do tipo dataframe
        for i, linha in dfimp.iterrows(): # executa a função novamente para
cada linha (series)
            riscoValor(linha) # reexecuta para cada linha se fornecido um d
ataframe (recursividade)
    if tipo=='s':
        print('\x1b[1;31m'+'RELATÓRIO DE RISCO'+'\x1b[0m')
        print('\x1b[1;31m'+'\nDados da Declaração de Importação'+'\x1b[0m')
        descricao=dfimp['DESCRICAO DO PRODUTO']
        origem=dfimp['PAIS DE ORIGEM']
        retentor=retentorAux(dfimp['DESCRICAO DO PRODUTO'])
        retentortexto="com retentor" if retentor else "sem retentor"
        aplicacao=classificaAplicacao(descricao)
        valor=round(dfimp['VALOR UN.PROD.DOLAR'],2)
        valorABIMOTO=achaValorABIMOTO(aplicacao, retentor)
        print(f'Descrição: {descricao}\n'+
              f'Origem:    {origem}\n'+
              f'Retentor:  {retentortexto}\n'+
              f'Aplicação: {aplicacao}\n'+
              f'Valor DI:  USD {valor:.2f}')
        if valorABIMOTO>0: print(f'Tabela de Referência ABIMOTO\nValor:
USD {valorABIMOTO:.2f}\n')
        # filtra o df somente para os registros da aplicação classificda
        dfrisco=df[df['APLICACAO']==aplicacao] # filtra para a aplicação
        dfrisco=dfrisco[dfrisco['RETENTOR']==retentor] # filtra para a exis
tência de retentor
        # remove os valores discrepantes dos dados
        # filtra somente valores de Q25-discrepancia*(Q75-Q25) a Q75+*discr
epancia*(Q75-Q25)
        discrepancia=3 # valor padrão=1.5
        P25=np.percentile(dfrisco['VALOR UN.PROD.DOLAR'],25)
        P75=np.percentile(dfrisco['VALOR UN.PROD.DOLAR'],75)
        min=P25-discrepancia*(P75-P25)
        max=P75+discrepancia*(P75-P25)
        filtro=[min<=dfrisco['VALOR UN.PROD.DOLAR']] and [dfrisco['VALOR U
N.PROD.DOLAR']<=max]
        dfrisco=dfrisco[filtro[0]]
        # recálculo dos quartis 25 e 75 para df sem outliers
        P25=np.percentile(dfrisco['VALOR UN.PROD.DOLAR'],25)
        P75=np.percentile(dfrisco['VALOR UN.PROD.DOLAR'],75)
        # geração da lista com os quartis
```

```
        P=[round(np.percentile(dfrisco['VALOR UN.PROD.DOLAR'],x),2) for x i
n range(0,110,10)]+[round(P25,2),round(P75,2)]
        estatisticas=dfrisco.describe()['VALOR UN.PROD.DOLAR']
        estatisticas.rename(index={'count':'qtd de registros:','mean':'médi
a simples:','std':'desvio padrão:','min':'valor mínimo:','25%':'percentil 2
5%:','50%':'percentil 50%:','75%':'percentil 75%:','max':'valor máximo:'},
inplace=True)
        print('\x1b[1;31m'+f'Estatísticas:\n'+'\x1b[0m'+f'{estatisticas.to_
string()}\n')
        # fazer um plot indicando a posição do valor em uma barra variando
 em vermelho-amarelo-verde-azul
        quartil={}
        for x in range(0,10): quartil[10+x*10] = P[x]
        quartil[25]=P25
        quartil[75]=P75
        perc=[f'{chave}={quartil[chave]:.2f}' for chave in sorted(quartil.k
eys())]
        print('\x1b[1;31m'+f'Percentis:\n'+'\x1b[0m'+f'  {str(perc[:6])[1:-
1]}\n  {str(perc[6:])[1:-1]}')
        print('\x1b[1;31m'+f'\nHistograma:'+'\x1b[0m')
        plt.figure(figsize=(10,5),frameon=False)
        plt.hist(dfrisco['VALOR UN.PROD.DOLAR'],10,rwidth=0.9,range=(P[0],P
[10]))
        plt.show()
        print('\x1b[1;31m'+f'Previsão de Risco:'+'\x1b[0m')
        plotrisco(P,valor,valorABIMOTO)
        print('\n\n')
```

In [13]:

```
riscoValor(df.iloc[2103])
```

RELATÓRIO DE RISCO

Dados da Declaração de Importação

Descrição: 22317 - 91131 - KIT DE TRANSMISSÃO PARA MOTOCICLETA - MODELO C 100 BIZ (13-15) - CONTENDO COROA 34Z - PINHAO 14Z - C/CORRENTE 428H X 108L - TITANIUM (1045)
Origem: CHINA, REPUBLICA POP
Retentor: sem retentor
Aplicação: HONDA BIZ C100 125 C125
Valor DI: USD 4.09
Tabela de Referência ABIMOTO
Valor: USD 3.90

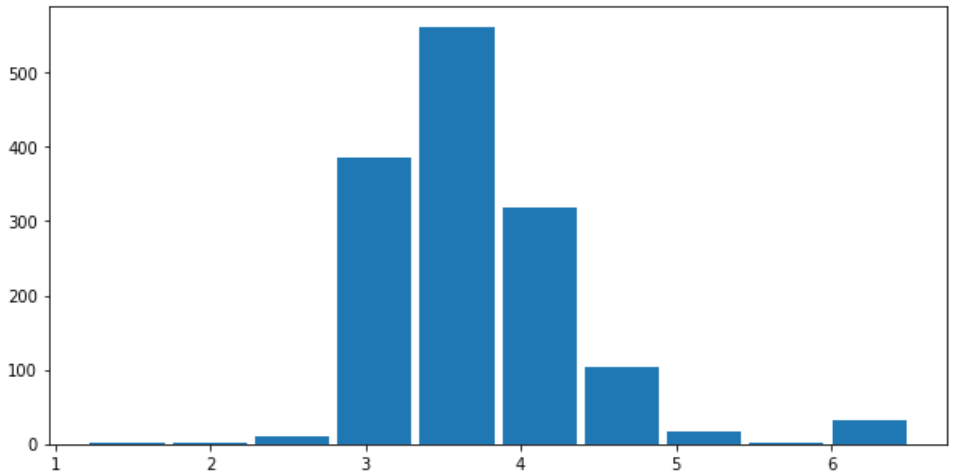
Estatísticas:

qtd de registros:	1434.000000
média simples:	3.742230
desvio padrão:	0.618322
valor mínimo:	1.202000
percentil 25%:	3.290000
percentil 50%:	3.713600
percentil 75%:	4.090000
valor máximo:	6.511000

Percentis:

'10=1.20', '20=3.06', '25=3.29', '30=3.23', '40=3.36', '50=3.55'
'60=3.71', '70=3.79', '75=4.09', '80=3.94', '90=4.13', '100=4.48'

Histograma:



Previsão de Risco: