Universidade Federal do Paraná
Setor de Ciências Exatas
Departamento de Estatística
Programa de Especialização em *Data Science* e *Big Data*

Cristian Juliano Vivan

# Aplicação de Modelos de Machine Learning para Classificação de Registros de Reclamação de Consumidores

**Curitiba**
**2020**

Cristian Juliano Vivan

# Aplicação de Modelos de Machine Learning para Classificação de Registros de Reclamação de Consumidores

Monografia apresentada ao Programa de Especialização em Data Science e Big Data da Universidade Federal do Paraná como requisito parcial para a obtenção do grau de especialista.

Orientador: Prof. Walmes Zeviani

Curitiba

2020

Especialização em Data Science e Big Data
Universidade Federal do Paraná
*dsbd.leg.ufpr.br*

# Aplicação de Modelos de Machine Learning para Classificação de Registros de Reclamação de Consumidores

Cristian Juliano Vivan[1]
Prof. Walmes Zeviani[2]

## Resumo

Um estudo de caso para avaliar as potencialidades de aplicação, de ferramentas de Aprendizado de Máquina, combinadas à técnicas de Mineração de texto, para classificar em categorias de defeitos, dados por registros históricos de reclamações de consumidores, de uma indústria de bens de consumo. Estudo prático performado pela avaliação de duas técnicas de definição de termos chave, associado ao comparativo de performance de classificadores lineares e não lineares, na modelagem do problema
**Palavras-chave**: aprendizado de máquina, mineração de texto, classificação, categorização, métodos lineares, métodos não lineares, reclamação de consumidores.

## Abstract

*A case to evaluate potential applications of Machine Learning tools, combining with Text Mining techniques, to clustering in defect categories, several consumer complaints historical records, from a consumer goods industry. A practical study performed by the evaluation of two different approaches for key terms definitions, combined to a model performance comparison, for linear and non linear classifiers.*
***Keywrods**: machine learning, text mining, clustering, categorization.*

## 1 Introdução

Atualmente muitos autores defendem a existência de uma quarta onda da Revolução Industrial, também conhecida como Industria 4.0. Um modelo de integração horizontal de Sistemas Inteligentes, capazes de processar dados, informações e de interagir com objetos do mundo externo - Internet das Coisas [1]. Neste contexto, uma nova dinâmica de relacionamento com o consumidor, adiciona outros aspectos aos processos produtivos tradicionais, como sistemas em Aprendizado de Máquina, técnicas de simulações e processamento em Big Data [2]. Conceitualmente definem-se estes modelos, como uma série de práticas para soluções de problemas, obtidos através de modelagens numéricas de um conjunto finito de fatos, e estes aplicados na identificação de padrões, agrupamentos e correlações, que descrevam propriamente objetos, processos, situações ou ambientes [3]. Ou ainda modelos numéricos capazes de aprender de forma adaptativa, por meio de aprendizado supervisionado, não supervisionado, de forma ativa ou passiva, por meio de correções externas ou por suas próprias relações [4].

O presente estudo visa demonstrar, através de um caso prático, as potencialidades da aplicação de modelos 4.0, no âmbito de Qualidade Industrial e Relacionamento com Consumidores. Dispondo para o estudo, uma base histórica do período de janeiro/2018 a março/2020, totalizando mais de 85.000 registros individuais, previamente classificados em categorias de defeitos. Por motivos de confidencialidade, a designação das categorias, será dada por: Cat.A, B, C, D, E.

Em se tratando de um processo totalmente manual, o procedimento atual para classificação das reclamações, apresenta um nível elevado de erro, na ordem de 10%, portanto, propõem-se neste trabalho, desenvolver um modelo automático, que classifique os registros em uma das cinco categorias de defeito, utilizando para isto, a transcrição textual da reclamação, também conhecida como Verbatim.

A apresentação deste trabalho divide-se em 3 seções principais: 1) Uma revisão conceitual de métodos e técnicas a serem utilizados no estudos, iniciando-se com recursos aplicados no Processamento de Linguagem Natural, onde serão exploradas técnicas para organização do formato textual em verbetes utilizáveis, seguido dos modelos de representação vetorial e seleção de Características principais, finalizando de uma breve descrição dos principais modelos de Classificadores. 2) Apresentação da modelagem do problema, suposições e medidas comparativas, bem como, resultados obtidos ao final do estudo. 3) Conclusões a partir dos resultados e considerações finais. Toda análise foi feita em Python versão 3.7.3, utilizando principalmente, as bibliotecas NLTK [5] e Scikit-learn [6]

[1]Aluno do programa de Especialização em Data Science & Big Data, cjvivan@uol.com.br.
[2]Professor do Departamento de Estatística - DEST/UFPR.

# 2 Revisão de Métodos e Conceitos

## 2.1 Processamento de Linguagem Natural

O problema de compreensão da linguagem natural é um exercício intrinsicamente complexo, por requerer entendimento sobre objetivos das comunicações, inferências do locutor, contexto da interação dos mediadores, e a necessidade de desconstruir a linguagem em níveis distínto de análise, desde a morfológica, semântica e regras de sintaxe [7]. As etapas principais, relacionadas em um problema de classificação por Machine Learning, envolvem [8]:

1. Pré-processamento de texto e criação de um dicionário de termos;
2. Mapeamento dos documentos e disposição em forma representativa;
3. Identificação de padrões que possam distinguir as categorias;
4. Avaliação da solução ótima, minimizando erros de classificação.

Em termos de processamento computacional, a estrutura de texto deve ser considerada como dado bruto de entrada, a ser manipulado, para conferir um caráter interpretativo aos programas ou sistemas. Para tanto uma série de técnicas e ferramentas podem ser utilizadas, com objetivos de compilar, indexar, promover refinamento e facilitar a seleção de palavras chaves, que melhor caracterizem o texto em análise [9]:

### 2.1.1 Normalização:

Atividade inicial para padronização do texto de entrada, aplica-se correção da capitalização (maiúsculas e minúsculas), remoção de pontuação, hifenização e espaçamentos. Aplica-se ainda as técnicas de *Unicode*, no qual caracteres especiais e acentuações de diversas línguas nativas, são removidos e substituídos por uma formatação padrão, dos caracteres ASCII [9].

### 2.1.2 Representação simplificada:

Uma das técnicas de simplificação da lista de palavras, é conhecida como *Stemming*, onde apenas o radical do vocábulo é extraído, eliminado assim, variações de flexões nominais e verbais, que em termos de caracterização, representam a mesma função[9]. A biblioteca NLKT dispõe de dois pacotes de Stemmers, com funções capazes de processar textos em língua portuguesa - o pacote *Snowball* e o pacote *RSLP* [5].

### 2.1.3 Refinamento da seleção:

Há ainda a possibilidade de retirada de conectores textuais, conhecidos como *Stopwords*, palavras que apresentam função gramatical como artigo ou advérbio, e que não constituem elementos de interesse, na caracterização das classificações. A retirada de Stopwords, oferece a diminuição nos comprimentos das listas finais, e por consequências, uma maior exatidão na análise da frequência dos termos [10]. Comumente a lista de termos característico é também conhecida como *Bag of Words*

## 2.2 Matriz de Documentos e Termos

A necessidade da representação da informação, em linguagem interpretativa para Aprendizado de Máquina, requer para seu uso uma codificação adequada destas informações. Em um espaço contendo k Documentos, sendo estes associados a uma lista de termos - para cada vocábulo indexado na lista, avalia-se o nível de representatividade do termo na caracterização individual, dos documentos do Corpus. Esta medida de representatividade é codificada em valor numérico através de um peso, ponderado pela importância relativa do termo no documento.

Em forma matricial, todos os Documentos contidos no espaço de dimensão k, são representados por um vetor t-dimensional, na forma: $D_i = (w_{i,1}, w_{i,2}...w_{i,t})$. Onde $t$ é o número de termos indexados, e $w_{i,t}$ o peso ponderado [11]. Finalmente o próprio Corpus de documentos representa-se em forma matricial, de dimensões k x t, denominada como Matriz de Documentos e Termos (MDT) - Fig.1.

$$\begin{array}{c} \text{Termos} \\ t_1 \quad \cdots \quad t_t \\ \downarrow \end{array}$$

$$\text{Documentos} \quad \begin{matrix} D_1 \\ \vdots \\ D_k \end{matrix} \rightarrow MDT_{k\,x\,t} = \begin{bmatrix} w_{11} & \cdots & w_{1t} \\ \vdots & \ddots & \vdots \\ w_{k1} & \cdots & w_{kt} \end{bmatrix}$$

Figura 1: Matriz Documentos e termos. (Fonte: Próprio Autor).

Uma forma simplificada de cálculo dos pesos da $MDT$, se dá pela utilização de lógica Booleana, atribuindo 1 quando um termo está presente, e 0 quando está ausente no documento [8]. A vetorização da matriz, pode então, assumir a forma binária, com variáveis *Dummy* do tipo:

$$MDT_{i,j} = \begin{cases} 1, & \text{se } t_{i,j} \in D_i \\ 0, & \text{se } t_{i,j} \notin D_i \end{cases} \quad (1)$$

### 2.2.1 Dicionário de termos:

Um recurso para obter a seleção de termos, que melhor representam as características, identificadas em um documento, é conhecido com método TF-IDF.

TF-IDF é a designação do termo (*term frequency - inverse document frequency*), utilizado para determinar a importância relativa de uma palavra, na representação

uma característica específica. Tal termo tem por função, oferecer parâmetros que permitem a identificação de um documento, em grupos, categorias ou famílias específicas. Desta forma palavras com uma alta frequência em um Documento, mas com um baixa ocorrência nos demais, formam a lista de termos mais importantes na representação vetorial da matriz $MDT$ [12].

A frequência de um termo (TF), é calculada pelo número de ocorrências de um termo $T_j$ em um mesmo $documento_i$, dividido pelo total de palavras contidas neste documento dado por $C_i$:

$$TF_{i,j} = \frac{(\sum T_j \in D_i)}{(C_i \in D_i)} \tag{2}$$

A frequência inversa (IDF), avalia a raridade de uma palavra em exposição. Se uma palavra é comum em todos documentos, ou se ela caracteriza um documento específico [13]. Seu valor é dado por:

$$IDF_j = \log\left(\frac{N}{\sum(T_j \in N)}\right) \tag{3}$$

onde $N$ é o conjunto de todos os documentos disponíveis. O peso final de um termo, e consequentemente o seu nível de importância na lista final, é calculado por:

$$W_{i,j} = TF_{i,j} * IDF_j \tag{4}$$

Uma segunda abordagem disponível, dá-se através de criação prévia de um glossário de palavras, selecionado de forma manual e intuitiva, apoiada por critérios de frequência nos documentos. O dicionário de termos será considerado Universal - quando aplicável a todas as categorias, ou Local - quando cada categoria define seu dicionário apropriado [8].

## 2.3 Modelos de Classificadores

Classificação de textos é uma técnica empregada para organizar classes de documentos, a partir de uma base de elementos categóricos rotulados, associados a um conjunto de termos característicos [14]. Os modelos de Aprendizado de Máquina - Supervisionados, são alimentados por informações conhecidas de uma base de treino, com as quais, o algoritimo atualiza suas variáveis internas, criando assim um modelo de transformação, que, uma vez aplicado a novos documentos, decide com base no aprendizado anterior, qual a melhor categoria de classificação, deste novo documento.

Na Fig. 2 verifica-se um diagrama em árvore, que sugestiona o melhor Classificador conforme a natureza do problema. Por este diagrama, identifica-se métodos lineares: *Support Vector Machine* e *Regressão Logística*, denotados nas funções *Linear SVC* e *SGD Classifier* respectivamente. Também identifica-se o modelo *Naive Bayes*, associado aos métodos não lineares [6]. Por fim, considera-se ainda uma abordagens clássicas, não referenciada no diagrama, mas de ampla aplicação em problemas de classificação - *Redes Neurais* [15].

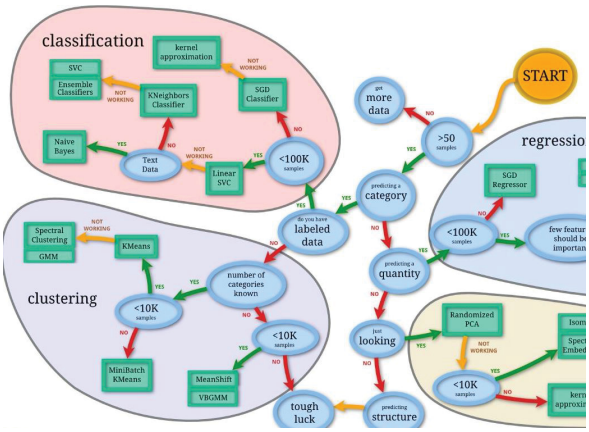


Figura 2: Diagrama para seleção de métodos. (Fonte: Sckiti-learn [6]).

### 2.3.1 Support Vector Machine:

Define-se o método *SVM - Linear* como um modelo capaz de classificar dados de forma linearmente separável - a aplicação linear do SVM é uma particularização do método, pela parametrização de Kernel linear. O método admite um hiperplano de n dimensões dado pelo função:

$$f(x) = w \cdot x + b = 0 \tag{5}$$

Esta função separa completamente os dados de treinamento em $1\ se\ f(x_i) > 0$ e $-1\ se\ f(x_i) < 0$.

O problema consiste então, em encontrar um vetor normal ao hiperplano $w$, e uma constante de deslocamento $b$. Para a solução final não se define um plano único, não separável, mas limites de margens obtido a partir da resolução de um problema de otimização quadrática, definida pelas equações [16]:

$$\begin{aligned} min/max : F(x) &= \frac{1}{2} w \cdot w \\ S.a : y_i(w \cdot x_i + b) &\geq 1\ onde :\ i = 1, \ldots m \end{aligned} \tag{6}$$

A Fig. 3 exemplifica o modelo SVM, com a região do hiperplano, as linhas de margem, e a disposição das variáveis na seleção do classificador:

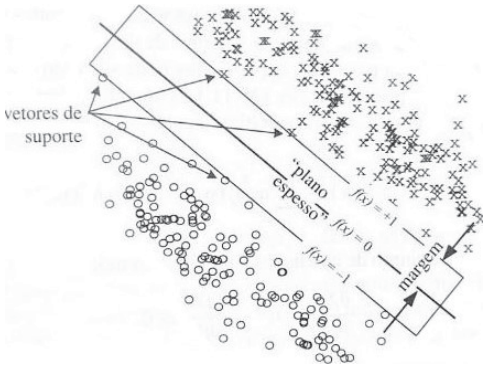


Figura 3: Modelo de Classificação SVM. (Fonte: Métodos Numéricos Aplicados [16]).

### 2.3.2 Regressão Logística:

O método de classificação utilizando Regressão Logística, avalia, via função de achatamento (sigmoide), a probabilidade de um elemento $x$ pertencer a uma determinada classe $K$, de tal forma que a soma de todos os termos da função linear, esteja no intervalo (0,1) [17].

A probabilidade do elemento $X$ representar uma classe $k$, é dada por [18]:

$$P(X_k|x1...xn) = \frac{1}{1+e^{-z}} \tag{7}$$

onde Z é a função linear dada por:

$$z = \beta 0 + \beta 1.x1 + ... + \beta n.xn \tag{8}$$

O modelo permite avaliar a probabilidade de um novo elemento pertencer a uma classe, a partir da inferência dos coeficientes $\beta$, previamente calculados com os dados de treino.

### 2.3.3 Naive Bayes:

Naive Bayes é um método de aprendizagem e classificação, baseado nos princípios da estatística Bayesiana, do qual decorre que a probabilidade a posteriore, de um elemento $x$ pertencer a uma classe $Y$, é dado por:

$$P(Y|x) = P(Y).\frac{P(x|Y)}{P(x)} \tag{9}$$

Pela suposição, de que todos os termos da matriz, são independentes, deriva-se a equação final, no qual a classe selecionada, será o argumento que maximiza o produtório da probabilidade condicional P(x | Y)[18]:

$$\dot{Y} = argmax\ P(Y).\prod_{i=1}^{n} P(x_i|Y) \tag{10}$$

### 2.3.4 Redes Neurais:

Redes Neurais são algoritmos de aprendizado, cuja função é atualizar seus pesos sinápticos, em resposta aos dados de entrada, corrigindo dinamicamente os erros de saída, conforme o rótulo de dados previamente conhecido. Tal processo sináptico é ilustrado na Fig 4.

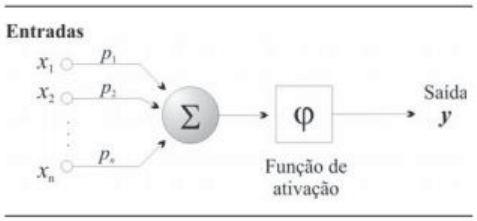


Figura 4: Modelo de Rede Neural.(Fonte: Redes neurais e sua aplicação em sistemas de recuperação de informação [19]).

No qual o vetor X é ponderado, pelos pesos na entrada, é transformado pela função de ativação, na variável de saída.

Redes do tipo Multi-Layer Perceptron (MLP), são estruturas organizadas em múltiplas camadas, capazes de aprender padrões, e identificar classificações de saída em qualquer modelo multivariado [15].A Fig 5 ilustra a configuração de uma rede em multicamadas e suas conexões sinápticas.

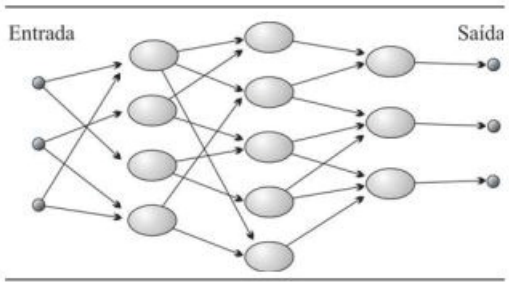


Figura 5: Multi-Layer Perceptron.(Fonte: Redes neurais e sua aplicação em sistemas de recuperação de informação [19]).

### 2.3.5 Medidas de desempenho:

A medida primária de performance dos Classificadores é a Acurácia, que avalia a proporção de erros e acertos obtidos com aplicação do modelo. Os tipos de erro possíveis em uma etapa de classificação, podem ser do tipo Falso Positivo ($fp$), ocorre quando uma categoria Negativa é classificada como Positiva e Falso Negativo ($fn$), ocorre quando uma categoria Positiva é classificada como Negativa, naturalmente os casos de acerto, Verdadeiro Positivo ($tp$) e Verdadeiro Negativo ($tn$) (Fig. 6).

| | **Resultado no Classificador** | |
|---|---|---|
| Categoria | Positivo | Negativo |
| Positivos | **tp** | **fn** |
| Negativo | **fp** | **tn** |

Figura 6: Tipos de erros (Fonte: Próprio Autor).

Assim, a medida de acurácia é dada por [20]:

$$Acurcia = \frac{tp+tn}{(tp+fp)+(tn+fn)} \tag{11}$$

Outra métrica de avaliação, chamada F1-Score, é definida como a média harmônica, entre os parâmetros de Precisão e Recall [18]. Os cálculos para estas métricas de performance, são dados pelas seguintes equações [20]:

$$F1 - Score = \frac{(\beta^2+1).(preciso * recall)}{(\beta^2 * preciso) + recall} \tag{12}$$

$$Preciso = \frac{tp}{(tp+fp)} \tag{13}$$

$$Recall = \frac{tp}{(tp+fn)} \tag{14}$$

Considerando o parâmetro $\beta = 1$, temos como resposta uma medida $F1 - Score$ balanceada entre precisão e recall, caso $\beta > 1$ o resultado pondera uma importância maior para os falsos positivos, e se $\beta < 1$, considera-se maior a importância para os falsos negativos.

#### 2.3.6 Validação Cruzada:

Validação cruzada é aplicada, principalmente, em casos de escassez de dados para dispor de duas bases teste e treino. Também, em casos que se deseja evitar o sobreajuste com os dados de treino.

Pelo método, divide-se o conjunto de dados em $K$ subconjuntos, O treinamento é realizado aplicando $K - 1$ elementos para o treino do classificador, e utilizando o subconjunto restante como base de teste. O procedimento é repetido K vezes, variando o subconjunto de teste em todas as iterações [15]. O esquema de avaliação por validação cruzada, segue apresentado na Fig 7.

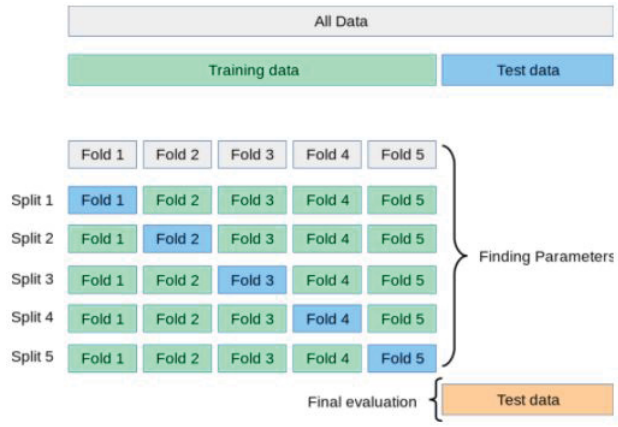


Figura 7: Validação Cruzada (Fonte: Sckiti-learn [6]).

## 3 Resultados e Discussões

Segue-se agora, apresentação dos resultados de todas análises, através da aplicação dos métodos e técnicas, previamente definidos, à avaliação do problema de classificação de reclamações de consumidores.

### 3.1 Preparação dos dados

A base de dados disponível, consta do registro histórico de reclamações, do período de janeiro/2018 a março/2020, em um total de 88.670 registros, todos previamente rotulados em uma de cinco categorias de reclamações apresentadas na introdução do artigo:

A mesma base de dados foi revisada, e corrigida. Um total, 10% das reclamações haviam sido classificadas de forma incorreta, portanto, para efeito de comparação, os modelos Classificadores utilizados no estudo, deverão apresentar uma acurácia superior a 90%, para que seu desempenho, seja considerado superior ao atual processo manual.

Inicia-se à análise com carregamento dos dados brutos, separados de forma aleatória em duas bases distintas: 70% dos dados aplicados para treino dos classificadores, 30% reservado como base de teste. A seleção aleatória dos dados, manteve a proporcionalidade das categorias, e eliminou efeitos de eventuais tendências sazonais, no período utilizado. A distribuição dos dados é apresentada na tabela 1.

| Categoria | Registros | %Total | Treino | Teste |
|---|---|---|---|---|
| Cat. A | 41.904 | 47.3% | 70% | 30% |
| Cat. B | 26.041 | 29.4% | 71% | 29% |
| Cat. C | 10.305 | 11.6% | 70% | 30% |
| Cat. D | 7.100 | 8.0% | 71% | 29% |
| Cat. E | 3.320 | 3.7% | 70% | 30% |

Tabela 1: Conjunto de dados - Reclamação de Consumidores

No pré-processamento de texto, todos os dados de entrada foram normalizados, a capitalização corrigida, retirada a pontuação e caracteres especiais (unicode). Ao final, foram suprimidas as Stopwords, disponível no conjunto do pacote *RSLP* [5], seguido do processo de Stemming para separação dos radicais das palavras. Na Fig. 8, fica exemplificada a disposição do texto extraído no Verbatim Nativo, comparado ao resultado após as transformações. Optou-se por manter a estrutura original na formatação do texto de saída.

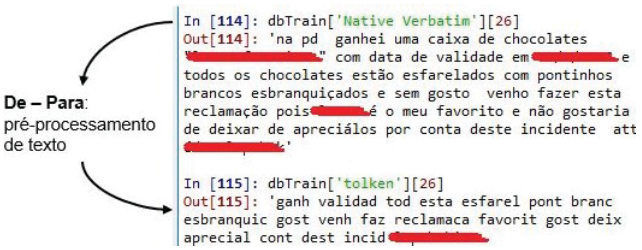


Figura 8: Exemplo de pré-processamento de texto (Fonte: Próprio Autor).

### 3.2 Matriz de documentos e termos

A matriz de documentos e termos (MDT), apresenta um formato k.t, onde k são todos os registros (Corpus $D_i$) contidos na base de treino, e t os termos ($t_{i,j}$), obtidos pelo Método TF-IDF, selecionados em função da ponderação $w_{i,j} = TF_{i,j}xIDF_j$. Este modelo de MDT é denominado, nas referências subsequentes, como modelo TF_IDF, ou simplesmente TF_IDF.

Para formação da lista de termos, buscou-se uma seleção de palavras, que sejam representativas das Categorias de reclamações. Para isso, ao invés de utilizar cada registro de reclamação como documento, o Corpus foi reagrupado em cinco documentos individuais, um para cada categoria representada, contendo todos os verbetes do subconjunto desta categoria. Desta maneira, a frequência relativa ($TF$), observou palavras de maior

incidência nos registros da categoria, e o IDF, avaliou se o termo, ainda que frequente, é comum em todos documentos. O formato de arranjo do corpus é ilustrado na Fig. 9, para avaliação do TF-IDF foi utilizada a função *TfidfTransformer()* [6].

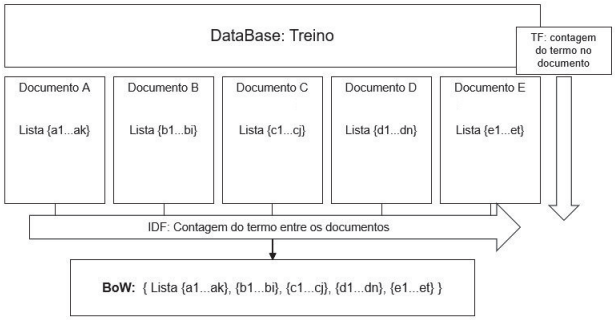


Figura 9: Arranjo dos documentos na avaliação TF-IDF (Fonte: Próprio Autor).

Finalmente, todos os termos dos Documentos avaliados, foram ordenados - pelos pesos ponderados $w_{i,j}$ - de forma decrescente, em um único frame de dados. O comprimento final da lista foi obtido por um critério limite de corte, $w_i \geq C$, onde $C$ é valor que otimiza o resultado de acuracidade da classificação. Para efeito de avaliação, utilizou-se o modelo *SVM*, através da função: $svm.LinearSVC(random_state = 0, tol = 1e-5)$ [6]. Como medidas de desempenho, em cada simulação, foram calculados a acurácia na classificação da base de teste, e a acurácia média por validação cruzada na base de treino, com $K - fold = 5$.

Pelo resultado apresentado na Fig. 10 verifica-se o aumento exponencial na quantidade de termos do *Bag of Word (BoW)*, na variação do critério de corte $w_i$.

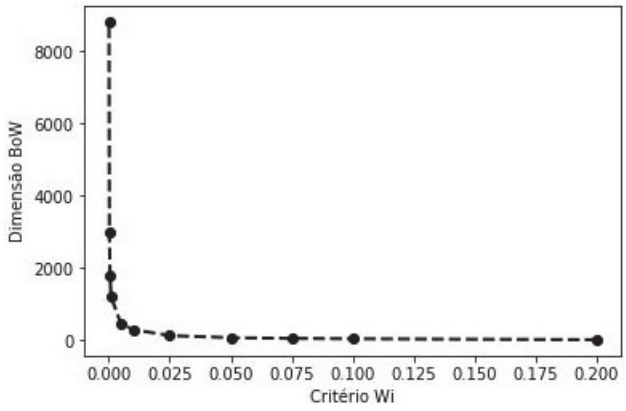


Figura 10: Dimensões do BoW em função do critério $w_i$ (Fonte: Próprio Autor).

Verifica-se (Fig. 11) um ponto de máxima performance $Ac = 0.883$, onde $w_i = 0.001$ e a dimensão do BoW igual a 1.191 termos.

Como saída, obteve-se uma MDT com dimensões $A_{[62.188x1.191]}$

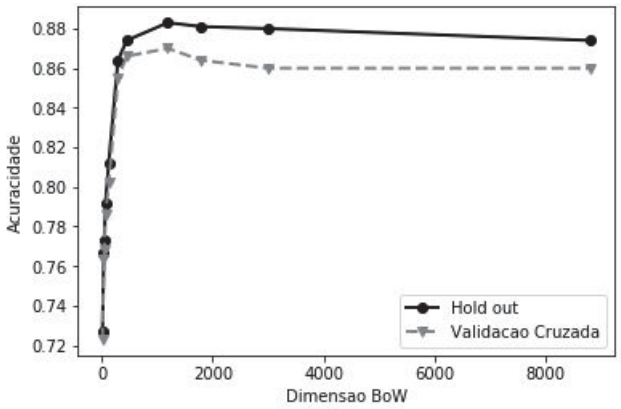


Figura 11: Acurácia na classificação pela dimensão BoW (Fonte: Próprio Autor).

## 3.3 Dicionário de termos

Utilizando a segunda abordagem definida para construção da matriz de documentos e termos, optou-se pela criação de um dicionário universal. Construído a partir da lista de termos, que resultou na melhor solução do classificador - $Acurcia = 0.883, BoW = 1191 termos$ - Fez-se a exclusão de palavras relacionadas a procedimentos internos, palavras que descrevem produtos, nomes e logradouros, selecionando apenas, os verbetes que melhor adjetivam as categorias.

Adicionalmente, foram somadas a lista de termos, um conjunto de bigramas - par de palavras, que são avaliadas quanto ao significado de forma conjunta [7] - com o objetivo de destacar diferenças semânticas em categorias que compartilham muitos termos comuns.

Ao final, selecionaram-se 189 termos livres e 12 bigramas, para a formação do Dicionário, pela utilização deste, a dimensão finais da matriz MDT, ficou em $A_{[62.188x201]}$. Para as referências subsequentes, este modelo de MDT é denominado como modelo Dicionário de termos, Dicionário, ou Dicionário de termos.

## 3.4 Performance de modelos e métodos

Inicialmente, comparando as duas abordagens para definição da MDT. Aplicando-se a mesma abordagem, descrita no item 3.2, seleciona-se para comparação a MDT gerada a partir do método TF-IDF, e a MDT gerada com uso do Dicionário de termos, descrito no item 3.3.

A Fig. 12 apresenta os resultados de performance, obtidos com a classificação da base de testes, tanto por categorias como resultado médio do modelo.

A matriz confusão, apresentada graficamente na Fig. 13, demonstra que, quanto maior a exatidão do modelo de classificação, maior será a concentração de dados na diagonal principal, valores plotados em outras regiões do gráfico, denotam haver confundimento com outras categorias.

Performado também, a validação cruzada nos dados da base de treinamento, utilizando um modelo K-fold, com $k = 10$. Apresentado graficamente (Fig. 14), a

Modelo TF-IDF

| | f1-score | precision | recall | support |
|---|---|---|---|---|
| Cat. (A) | 0.902035 | 0.881770 | 0.923254 | 12626.0 |
| Cat. (E) | 0.840442 | 0.891620 | 0.794821 | 1004.0 |
| Cat. (C) | 0.957856 | 0.964867 | 0.950946 | 3119.0 |
| Cat. (D) | 0.684279 | 0.738566 | 0.637427 | 2052.0 |
| Cat. (B) | 0.874136 | 0.883378 | 0.865087 | 7679.0 |
| micro avg | 0.882628 | 0.882628 | 0.882628 | 26480.0 |
| macro avg | 0.851750 | 0.872040 | 0.834307 | 26480.0 |
| weighted avg | 0.881310 | 0.881300 | 0.882628 | 26480.0 |

Modelo Dicionario de Termos

| | f1-score | precision | recall | support |
|---|---|---|---|---|
| Cat. (A) | 0.884848 | 0.849778 | 0.922937 | 12626.0 |
| Cat. (E) | 0.822633 | 0.906475 | 0.752988 | 1004.0 |
| Cat. (C) | 0.940121 | 0.957143 | 0.923693 | 3119.0 |
| Cat. (D) | 0.650785 | 0.732034 | 0.585770 | 2052.0 |
| Cat. (B) | 0.863503 | 0.887103 | 0.841125 | 7679.0 |
| micro avg | 0.866730 | 0.866730 | 0.866730 | 26480.0 |
| macro avg | 0.832378 | 0.866507 | 0.805303 | 26480.0 |
| weighted avg | 0.864671 | 0.866273 | 0.866730 | 26480.0 |

Figura 12: Comparativo de métricas (Fonte: Próprio Autor).

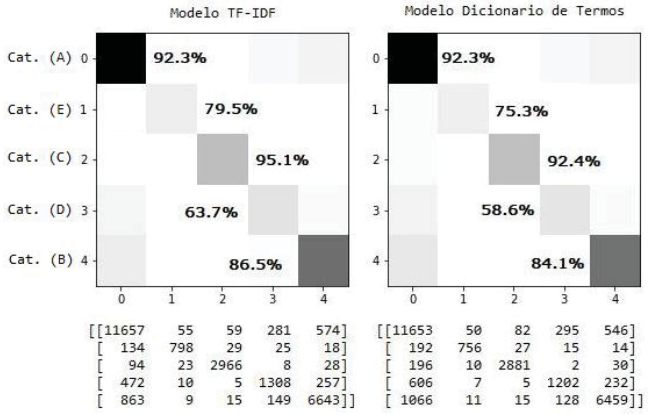


Figura 13: Matriz Confusão (Fonte: Próprio Autor).

dispersão dos resultados em cada ciclo da validação.

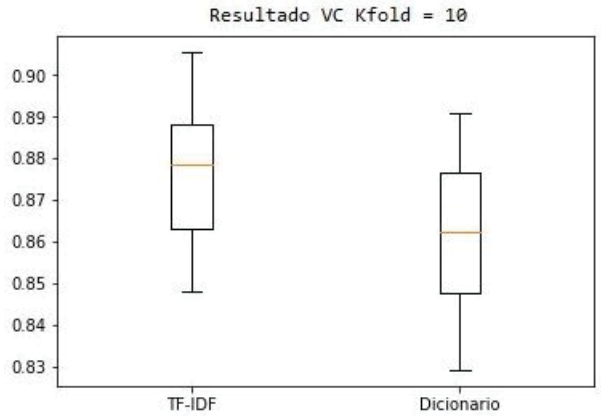


Figura 14: Validação Cruzada: Kfold = 10 (Fonte: Próprio Autor).

Em termos médios, os resultados para acurácia são apresentados na tabela 2.

| Modelo | Hold out | Kfold =10 | Desv Pad |
|---|---|---|---|
| TF-IDF | 0.883 | 0.876 | 0.017 |
| Dicionário | 0.867 | 0.862 | 0.018 |

Tabela 2: Medidas de Acurácia média

Finalmente, compara-se a performance, obtida com diferentes métodos de Classificação: Lineares - SVM e Regressão Logística, e não lineares: Naive Bayes e Redes Neurais. Para modelagem foram utilizadas as funções [6]:

- $LogisticRegression()$;
- $svm.LinearSVC()$;
- $MultinomialNB()$;
- $MLPClassifier()$.

Utilizando-se das mesmas medidas de avaliação, faz-se a comparação de performance dos Classificadores, quando submetidos a mesma base de treino, teste e Matrizes MDT.

Os resultados da análise, são apresentados na tabela 3 para dados do modelo TF-IDF:

| Métrica | R.Log | SVM | NB | MLP |
|---|---|---|---|---|
| Acurácia | 0.884 | 0.883 | 0.780 | 0.863 |
| F1-Score | 0.882 | 0.881 | 0.781 | 0.862 |
| Kfold(5) | 0.872 | 0.869 | 0.750 | 0.838 |
| Desv P. | 0.014 | 0.013 | 0.020 | 0.013 |
| Tempo(s) | 102 | 214 | 5 | 1010 |

Tabela 3: Comparativo de Classificadores (TF-IDF)

Os resultados com Modelo Dicionário de termos, são apresentados na tabela 4

| Métrica | R.Log | SVM | NB | MLP |
|---|---|---|---|---|
| Acurácia | 0.868 | 0.867 | 0.836 | 0.850 |
| F1-Score | 0.867 | 0.865 | 0.840 | 0.850 |
| Kfold(5) | 0.863 | 0.861 | 0.831 | 0.830 |
| Desv P. | 0.015 | 0.014 | 0.013 | 0.010 |
| Tempo(s) | 18 | 64 | 2 | 915 |

Tabela 4: Comparativo de Classificadores (Dicionário)

A performance média, de cada classificador pelo modelo de MDT, é representada na Fig.15.

## 3.5 Discussões dos resultados

Com base nos resultados obtidos e apresentados no capítulo anterior, é possível inferir a alta performance e estabilidade demonstrada em todos os resultados dos Classificadores. Ainda que o melhor resultado de acurácia obtido - $Ac = 88.4\%$ não tenha sido superior à referência do método manual, a performance em vários dos

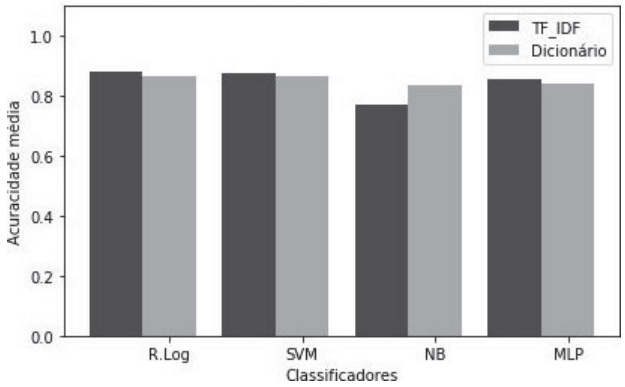


Figura 15: Performance média dos Classificador (Fonte: Próprio Autor).

resultados, se aproximam em muito do valor objetivo (90%). Apesar de não ser possível a conclusão, de que os modelos apresentados, possam substituir um processo inteiramente manual, fica evidenciado que tais modelos, atuando como um sistema de suporte a decisão, podem conferir uma melhoria significativa, na redução do nível de erro no processo atual.

Quanto a metodologia, as duas técnicas para criação da Matriz de Documentos e Termos (MDT), tiveram performances satisfatórias, sendo a abordagem por TF-IDF, superior em todos Classificadores, exceto para o Naive Bayes. No entanto, mesmo apresentando um resultado - em média - de 2 pontos percentuais abaixo em performance, a técnica do Dicionário de termos pode ser vantajosa em muitos aspectos, por permitir uma avaliação interpretativa das variáveis e por ser uma abordagem mais compreensível ao público não especializado, uma vez que todos os termos são diretamente relacionáveis às categorias, facilitando o entendimento dos mecanismos do método. A mesma técnica, demonstra uma simplicidade de implementação, aplicável a uma estrutura menos complexa de sistemas, sendo facilmente editável e atualizável para inserção de novas categorias.

As categorias de forma individual, tiveram um bom desempenho como demonstrado no índice F1-Score, Cat. A e Cat. C, com performance superior a 90% e as demais, apresentando uma acurácia média na ordem de 87%.

A exceção encontrada, está na categoria D, dado que os melhores resultados não foram superiores a 70%. Neste caso, uma explicação possível, reside no fato de que muitos dos registros, associados a ela, tratam de mais do que um objeto de reclamação, em um mesmo contato, um consumidor manifesta insatisfação quanto à multiplos aspectos, neste caso o mesmo registro apresentará termos designados para duas ou mais categorias simultaneamente, favorecendo assim, o confundimento não pela falha do método, mas pela falta de especificidade na descrição do problema.

Quanto aos classificadores, os modelos lineares - Regressão Logística e SVM, tiveram os melhores desempenhos para todas as métricas. O método Naive Bayes teve sua pior performance, quando da classificação, por um número maior de termos.

# 4 Conclusões

Conclui-se, considerando a questão inicial, sobre avaliação da potencialidade de aplicação de modelos com base em Aprendizado de Máquinas, que os métodos apresentados são plenamente factíveis de utilização em casos reais, acompanhando as necessidade de Negócio e novos processos industriais, bem como, plenamente flexíveis, para serem incorporados em diferentes panoramas, que não necessariamente os cenários explorados neste estudo.

# Agradecimentos

Ao Professor Walmes, pela orientação e dedicação prestada ao presente artigo. Bem como a todo corpo doscente representados no Programa de especialização.

A minha esposa e fílha, pelo suporte e constante apoio, na superação deste desafio.

Aos estimados colegas, que de muitas maneiras, apoiaram e enriqueceram este trabalho final.

# Referências

[1] Christoph Jan Bartodziej. *The Concept Industry 4.0*. Springer Gabler Berlin, 2017.

[2] Rodic Blaz. *Industry 4.0 and the New Simulation Modelling Paradigm*. Organizacija, Volume 50, Number3, August 2017, Liubliana, 2017.

[3] R. Prati. *Novas abordagens em aprendizado de máquina para geração de regras, classes desbalanceadas e ordenação de casos*. USP, São Carlos, 2006.

[4] Shai Shalev Shwartz and Shai Ben-David. *Understanding Machine Learning. From Theory to Algorithms*. Cambridge University Press, 2014.

[5] NLTK. *NLTK 3.5 documentation*. Disponível em https://www.nltk.org/, 2020.

[6] scikit learn. *scikit-learn*. Disponível em https://scikit-learn.org/stable/, 2020.

[7] George F Luger. *Inteligência Artificial*. Pearson Education do Brasil, 2013.

[8] F. Wess S. H. Apté, C. Damerau. *Automated learning of decision rules for text categorization*. ACM Transactions on Information Systems, v.12, n. 3, pg 233-251 , Jul, 1994, 1994.

[9] Ewan. Loper Edward Bird, Steven. Klein. *Natural Language Processing with Python*. O'Reilly Media, 2009.

[10] Lucas Moreira Gomes. *Clusterização de texto de reclamação não supervisionada usando K-means com python*. https://lamfo-unb.github.io/2019/09/02/clustertexto/, 2020.

[11] A. Yang C. S Salton, G. Wong. *A Vector Space Model for Automatic Indexing*. Association for Computing Machinery, Cornell University, New York, 1975, 1975.

[12] Ramsha Qaiser, Shahzad. Ali. *Text Mining: Use of TF-IDF to Examine the Relevance of Words to Documents*. International Journal of Computer Applications (0975 – 8887), Volume 181 – No.1, July 2018, 2018.

[13] Cai. Liang Yi Chen, Jie. Chen. *Optimized TF-IDF Algorithm with the Adaptive Weight of Position of Word*. Advances in Intelligent Systems Research, volume 133 - 2nd International Conference on Artificial Intelligence and Industrial Engineering, 2016, 2016.

[14] Fabrizio Sebastiani. *Machine Learning in Automated Text Categorization*. ACM Computing Surveys, Vol. 34, No. 1, March 2002, pp. 1–47, 2002.

[15] Simon Haykin. *Redes Neurais princípios e práticas*. Bookman, Porto Alegre, 2004.

[16] Saul A. Vetterling William T. Flannery Brian P Press, William H. Teukolsky. *Métodos Numericos Aplicados Rotinas em C++*. Bookman, Porto Alegre, 2011.

[17] Robert. Friedman Jerome Hastie, Trevor. Tibshirani. *The Elements of Statistical Learning Data Mining, Inference, and Prediction*. Springer, 2009.

[18] Claude Webb, Geoffrey I. Sammut. *Encyclopedia of Machine Learning*. Springer, 2011.

[19] Edberto Ferneda. *Redes neurais e sua aplicação em sistemas de recuperação de informação*. Ci. Inf., Brasília, v. 35, n. 1, p. 25-30, jan./abr. 2006, 2006.

[20] Nathalie. Szpakowicz Stan Sokolova, Mariana. Japkowicz. *Beyond Accuracy, F-score and ROC: a Family of Discriminant Measures for Performance Evaluation*. AI 2006, LNAI 4304, pp. 1015–1021, 2006, 2006.