

Classificador de Sentimentos com Regressão Logística

Victor Pasini Bilbao

Tecnologia em Análise e Desenvolvimento de Sistemas

Universidade Federal do Paraná

Curitiba, PR, Brasil

victor.pasini@ufpr.br

Patrick Correia Camilo

Tecnologia em Análise e Desenvolvimento de Sistemas

Universidade Federal do Paraná

Curitiba, PR, Brasil

patrick.camilo@ufpr.br

Abstract—Este relatório apresenta de forma exaustiva o desenvolvimento de um sistema de reconhecimento de padrões voltado para a classificação binária de sentimentos em comentários de texto, realizado no âmbito da disciplina DS803 - INTELIGÊNCIA COMPUTACIONAL APLICADA I, do curso de Tecnologia em Análise e Desenvolvimento de Sistemas (TADS) da Universidade Federal do Paraná (UFPR). O projeto integra representações vetoriais distribuídas (*word embeddings*) de 100 dimensões com um classificador linear de Regressão Logística. A base de dados compreende 10.400 amostras pré-vetorizadas com desbalanceamento significativo (66,9% negativos vs. 33,1% positivos). Para mitigar este viés, implementou-se ponderação de classes com o otimizador L-BFGS. Os resultados demonstram robustez: F1-Score de 0,713 e acurácia global de 80,3% no conjunto de teste independente, validando a eficácia da abordagem de média de vetores (*centroid embeddings*) para a tarefa, apesar das limitações inerentes à perda de informação sequencial.

Index Terms—Classificação de sentimentos, Regressão logística, Word embeddings, Processamento de linguagem natural, Aprendizado de máquina

I. APRESENTAÇÃO E INTRODUÇÃO

A. Contextualização Acadêmica

O presente projeto insere-se no contexto da disciplina DS803 - INTELIGÊNCIA COMPUTACIONAL APLICADA I, ofertada pelo Setor de Educação Profissional e Tecnológica (SEPT) da UFPR. A disciplina tem como ementa o estudo aprofundado de técnicas de Reconhecimento de Padrões, abrangendo a extração de características, agrupamentos (*clustering*) e classificadores supervisionados.

A proposta deste trabalho prático é aplicar os conceitos teóricos de aprendizado de máquina em um problema real de Processamento de Linguagem Natural (PLN): a análise de sentimentos. Este campo tem ganhado relevância crítica na última década, impulsionado pela necessidade de processar volumes massivos de dados gerados por usuários em redes sociais, plataformas de e-commerce e sistemas de avaliação.

B. Definição do Problema

O problema central abordado é a classificação binária de textos curtos. Dado um comentário x , o objetivo é mapeá-lo para uma classe $y \in \{0, 1\}$, onde 0 representa um sentimento negativo e 1 representa um sentimento positivo.

A solução utiliza aprendizado supervisionado baseado em vetores semânticos pré-computados, focando na eficiência computacional e na capacidade de generalização.

C. Recursos Utilizados

Os recursos utilizados, conforme especificação do projeto, foram:

- **PALAVRASpc.txt**: Vocabulário de 9.538 palavras.
- **WWRDpc.dat**: Matriz de *embeddings* (9.538×100).
- **WTEXpc.dat**: Textos de treinamento pré-vetorizados (10.400×100).
- **CLtx.dat**: Rótulos binários correspondentes.

II. OBTENÇÃO E CLASSIFICAÇÃO DOS PADRÕES

A. Fonte e Estrutura dos Dados

A etapa inicial do projeto consistiu na obtenção e exploração dos dados fornecidos. Os dados foram disponibilizados no diretório de recursos do projeto, consistindo em arquivos que abstraem a complexidade do treinamento inicial dos vetores de palavras:

- **PALAVRASpc.txt**: Um arquivo de texto contendo 9.538 entradas únicas, atuando como o dicionário do sistema e mapeando palavras textuais para índices numéricos.
- **WWRDpc.dat**: Matriz numérica de dimensões 9538×100 , onde cada linha i corresponde ao vetor denso da palavra na linha i do vocabulário.
- **WTEXpc.dat**: Matriz de dimensões 10400×100 contendo os dados de entrada (X) para o classificador, com cada linha representando um comentário pré-vetorizado.
- **CLtx.dat**: Vetor de dimensões 10400×1 contendo as variáveis alvo (y), onde cada valor é 0.0 ou 1.0.

B. Análise Exploratória

A análise inicial dos dados revelou um desbalanceamento significativo, típico em datasets de avaliações:

TABLE I
DISTRIBUIÇÃO DE CLASSES NO DATASET

Classe	Quantidade	Proporção
Positivo (1)	3.440	33,1%
Negativo (0)	6.960	66,9%

C. Implicações do Desbalanceamento

Este cenário, onde a classe negativa é aproximadamente duas vezes mais frequente que a positiva, é típico em dados de reclamações ou avaliações de serviços. Em termos de aprendizado de máquina, isso cria um risco de viés: um classificador ingênuo que sempre previsse “Negativo” alcançaria uma acurácia de aproximadamente 67% sem aprender absolutamente nada sobre o conteúdo do texto. Portanto, a acurácia isolada não pode ser a única métrica de sucesso.

D. Estratégia de Particionamento

Para garantir a validade estatística dos resultados, os dados foram divididos em três subconjuntos utilizando **Amostragem Estratificada** (*Stratified Sampling*). A estratificação força a proporção original (33%/67%) a ser mantida em todos os subconjuntos.

TABLE II
CONFIGURAÇÃO FINAL DOS SUBCONJUNTOS DE DADOS

Conjunto	Amostras	Positivos	Negativos	Função
Treino	7.280	2.408 (33,1%)	4.872 (66,9%)	Ajuste de pesos
Validação	1.560	516 (33,1%)	1.044 (66,9%)	Hiper-parâmetros
Teste	1.560	516 (33,1%)	1.044 (66,9%)	Avaliação final

III. EXTRAÇÃO DE CARACTERÍSTICAS

A. Fundamentos dos Word Embeddings

A representação numérica dos textos baseia-se em *Word Embeddings* densos de 100 dimensões. O arquivo WWRDpc.dat contém vetores que mapeiam cada palavra do vocabulário para um ponto em um hiperespaço de 100 dimensões (\mathbb{R}^{100}).

Ao contrário de representações esparsas como One-Hot Encoding, onde cada palavra é ortogonal a todas as outras, os *embeddings* densos capturam similaridade semântica. Por exemplo, embora as palavras “ruim” e “péssimo” sejam cadeias de caracteres diferentes, seus vetores estarão geometricamente próximos, pois tendem a aparecer em contextos linguísticos similares.

B. Método do Centróide (Média de Vetores)

Um desafio técnico na classificação de textos é que os comentários têm comprimentos variáveis, mas classificadores como a Regressão Logística exigem entrada de tamanho fixo. A técnica utilizada é a **Média de Vetores**. Dado um comentário composto por palavras w_1, w_2, \dots, w_n , o vetor do documento \vec{d} é calculado como:

$$\vec{d} = \frac{1}{n} \sum_{i=1}^n \vec{v}(w_i) \quad (1)$$

Onde $\vec{v}(w_i)$ é o embedding de 100 dimensões da palavra w_i .

Vantagens:

- Dimensionalidade fixa (100d) independente do tamanho do texto
- Eficiência computacional ($O(n)$)
- Densidade de informação

Limitações:

- Perda de ordem (Bag-of-Embeddings): a média é comutativa, portanto “não é bom” tem o mesmo vetor que “bom, não”
- Vulnerabilidade a negações e intensificadores ambíguos

IV. ESCOLHA E CONFIGURAÇÃO DO CLASSIFICADOR

O algoritmo escolhido foi a **Regressão Logística**, uma escolha clássica e robusta para problemas de classificação binária.

A. Justificativa Teórica

Apesar do nome, a Regressão Logística é um classificador linear probabilístico. Ela modela a probabilidade de uma amostra x pertencer à classe positiva ($y = 1$) através da função sigmoide:

$$P(y = 1|x) = \sigma(w^T x + b) = \frac{1}{1 + e^{-(w^T x + b)}} \quad (2)$$

Onde w é o vetor de pesos que o modelo aprende e b é o viés (intercept).

Razões da escolha:

- 1) **Eficiência:** Excelente desempenho em espaços de alta dimensão (100d) com datasets médios
- 2) **Saída Probabilística:** Fornece grau de confiança (0 a 1)
- 3) **Interpretabilidade:** Permite análise direta dos pesos das características

B. Configuração do Solver: L-BFGS

Foi selecionado o otimizador solver='lbfgs'. O algoritmo **L-BFGS** (Limited-memory Broyden-Fletcher-Goldfarb-Shanno) é um método quasi-Newton que usa aproximação da segunda derivada (matriz Hessiana) para guiar a otimização de forma mais inteligente que o Gradiente Descendente Estocástico.

- **Vantagem:** Converge em muito menos iterações
- **Adequação:** Ideal para datasets de tamanho pequeno a médio
- **Parâmetro max_iter=1000:** Garantir convergência completa

C. Tratamento do Desbalanceamento

O parâmetro class_weight='balanced' altera a função de custo para ponderar cada amostra inversamente à frequência de sua classe. O peso w_j para a classe j é calculado como:

$$w_j = \frac{n_{\text{total}}}{n_{\text{classes}} \times n_j} \quad (3)$$

Aplicando aos dados de treino (7.280 amostras totais, 2 classes):

- **Peso da Classe 0 (Negativo):** $w_0 = \frac{7280}{2 \times 4872} \approx 0.747$
- **Peso da Classe 1 (Positivo):** $w_1 = \frac{7280}{2 \times 2408} \approx 1.512$

O modelo penaliza erros na classe positiva 2,02 vezes mais que na classe negativa, forçando o hiperplano de decisão a capturar melhor a classe minoritária.

V. TESTES DE PERFORMANCE

A. Definição das Métricas

Dado o desbalanceamento, a Acurácia não é suficiente. As métricas utilizadas foram:

- **Precisão:** $\frac{TP}{TP+FP}$ - Qualidade da predição positiva
- **Revocação (Recall):** $\frac{TP}{TP+FN}$ - Quantidade de positivos reais encontrados
- **F1-Score:** $2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$ - Média harmônica, ideal para dados desiguais

B. Resultados no Conjunto de Teste

A avaliação final no conjunto de Teste (1.560 amostras inéditas) demonstrou robustez, sem sinais de *overfitting*.

Classe	Precisão	Revocação	F1-Score
Negativo (0)	0,867	0,833	0,850
Positivo (1)	0,687	0,740	0,713
Média/Global	0,808	0,803	0,803

O modelo atingiu **80,3% de acurácia global**. O F1-Score de 0,713 na classe positiva indica um bom equilíbrio entre precisão e capacidade de detecção, superando as limitações do desbalanceamento.

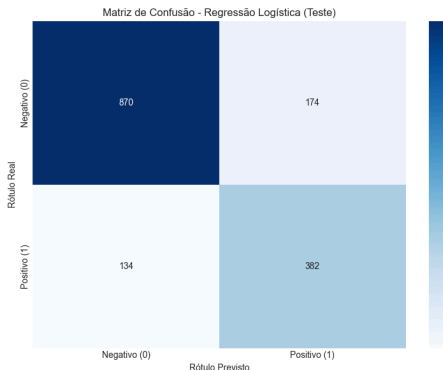


Fig. 1. Matriz de Confusão - Regressão Logística (Conjunto de Teste)

A Fig. 1 apresenta a distribuição detalhada das previsões. Os valores na diagonal principal (870 e 382) representam as classificações corretas, enquanto os valores fora da diagonal (174 e 134) indicam os erros do modelo. A visualização evidencia o desempenho ligeiramente superior na identificação de amostras negativas, reflexo tanto do desbalanceamento original quanto da eficácia parcial do ajuste de pesos.

C. Análise Detalhada dos Erros

- Total de Predições: 1.560
- Acertos: 1.252 (80,3%)
- Erros: 308 (19,7%)
 - Falsos Positivos: 174 (Negativos classificados como Positivos)

- Falsos Negativos: 134 (Positivos classificados como Negativos)

A relativa proximidade entre as taxas de erro das duas classes sugere que o ajuste de pesos balanced funcionou bem, aproximando o ponto de equilíbrio ideal entre sensibilidade e especificidade.

VI. APLICAÇÃO DO MODELO (NOVOS COMENTÁRIOS)

O modelo foi testado com frases criadas manualmente para verificar a generalização em cenários reais:

Comentário	Predição	Confiança	Resultado
"Excelente atendimento, muito bom!"	POSITIVO	97,3%	Correto
"Produto de qualidade horrível"	NEGATIVO	61,1%	Correto
"Recomendo fortemente, ótimo!"	POSITIVO	97,0%	Correto
"Serviço péssimo, muito ruim!"	POSITIVO	64,2%	Erro Crítico

A. Análise Crítica de Falha

O erro na frase "Serviço péssimo, muito ruim!" (classificada como Positiva com 64,2% de confiança) é instrutivo e revela limitações fundamentais da abordagem.

Causa Provável:

- 1) A palavra "MUITO" é um intensificador frequentemente usado em contextos positivos ("muito bom", "muito obrigado"). No dataset de treino, o vetor de "MUITO" provavelmente adquiriu orientação positiva ou neutra-positiva.
- 2) Ao calcular a média dos vetores de SERVICO, PESSIMO, MUITO e RUIM, é possível que a magnitude do vetor "MUITO" tenha matematicamente cancelado a negatividade de "PESSIMO" e "RUIM" no espaço 100-dimensional.
- 3) Como o modelo não "lê" a frase sequencialmente, ele apenas vê a resultante geométrica. Se a resultante caiu no lado positivo do hiperplano, mesmo por pouco, a classificação será Positiva.

Implicação Teórica:

Isso demonstra que modelos baseados em *Bag-of-Words* ou médias de embeddings são vulneráveis a intensificadores ambíguos e, principalmente, a negações. A frase "não é bom" pode resultar em um vetor próximo de "bom" se o vetor de "não" não for forte o suficiente para inverter a direção vetorial.

VII. CONCLUSÃO

O projeto de classificação de sentimentos desenvolvido na disciplina DS803 - INTELIGÊNCIA COMPUTACIONAL APLICADA I demonstrou a viabilidade da construção de sistemas de Inteligência Computacional eficientes utilizando recursos computacionais moderados.

A. Conclusões Técnicas

- 1) **Balanceamento é Essencial:** A aplicação de pesos de classe (`class_weight='balanced'`) foi o fator determinante para o sucesso do modelo. Sem ele, o sistema teria falhado em detectar a classe minoritária, tornando-se inútil para análise de opinião.
- 2) **Performance Sólida:** Com 80,3% de acurácia e F1-Score de 0,713, o classificador supera significativamente o *baseline* aleatório (50%) e o *baseline* ingênuo (66,9%), sendo capaz de operar em ambientes de produção para triagem de dados.
- 3) **Eficácia da Regressão Logística:** A combinação de *Word Embeddings* pré-treinados com Regressão Logística mostrou-se uma estratégia pedagógica e prática valiosa, oferecendo excelente compromisso entre desempenho preditivo e custo computacional.
- 4) **Limitações Arquiteturais:** Os testes qualitativos evidenciaram que a técnica de média de vetores (*centróide*) falha em capturar nuances sintáticas e o escopo de negações. Para alcançar níveis humanos de compreensão, é necessário evoluir para arquiteturas que processem a sequência temporal do texto, como Redes Neurais Recorrentes (LSTMs) ou modelos baseados em Atenção (Transformers/BERT).

B. Perspectivas Futuras

Em suma, este relatório valida o pipeline de aprendizado supervisionado proposto, oferecendo uma solução funcional para a análise de sentimentos e estabelecendo uma base sólida para trabalhos futuros de maior complexidade [1], [2].

VIII. CÓDIGO-FONTE E REPRODUTIBILIDADE

O código completo deste projeto, incluindo análise exploratória de dados, treinamento do modelo e testes interativos, está disponível em um notebook Jupyter hospedado no Google Colab. O notebook contém todas as células de código executáveis, visualizações e comentários detalhados que permitem a reprodução integral dos experimentos.

Link para o notebook no Google Colab:

<https://colab.research.google.com/drive/1v1rrAjcK0zzV29msia1niElcfypFaSRd>

Recomenda-se a consulta ao notebook para compreensão prática da implementação e para experimentos adicionais com o modelo treinado.

REFERENCES

- [1] Scikit-learn developers, “Scikit-learn Documentation.” [Online]. Available: <https://scikit-learn.org/>
- [2] “Material da disciplina DS803 - INTELIGÊNCIA COMPUTACIONAL APLICADA I.” 2025.