Análise Comparativa de Algoritmos de Aprendizado Supervisionado para Classificação de Cães e Gatos a partir de Descritores Visuais

Janderson Soares Mendes, João Gilberto Neves Saraiva

Instituto Instituto Metrópole Digital – Universidade Federal do Rio Grande do Norte (UFRN)

{jgilbertons,jandersonsoaresmendes}@gmail.com

Abstract *Image recognition, specifically the classification between dogs and cats, is a classic challenge in computer vision. This work conducts a comparative analysis of the performance of five classes of supervised learning algorithms (k-NN, Decision Tree, Naive Bayes, MLP Neural Networks, and Classifier Ensembles) in identifying images of canines (Miniature Pinscher, English Setter) and felines (Birman, Ragdoll). The data was pre-processed using the feature descriptors HOG (Histogram of Oriented Gradients) and LBP (Local Binary Patterns), in addition to dimensionality reduction techniques such as PCA. The models were evaluated using the weighted F1-Score metric and statistically compared using the Friedman test. The results indicate that the use of a neural network with one hidden layer of 150 neurons and a maximum of 500 iterations achieved the best performance considering its accuracy rate and computational cost, and that the LBP descriptor proved to be statistically superior.*

Resumo. *O reconhecimento de imagens, especificamente a classificação entre cães e gatos, é um desafio clássico em visão computacional. Este trabalho realiza uma análise comparativa do desempenho de cinco classes de algoritmos de aprendizado supervisionado (k-NN, Árvore de Decisão, Naive Bayes, Redes Neurais MLP e Comitês de Classificadores) na identificação de imagens de caninos (Miniature Pinscher, English Setter) e felinos (Birman, Ragdoll). Os dados foram pré-processados utilizando os descritores de características HOG (Histograma de Gradientes Orientados) e LBP (Local Binary Patterns), além de técnicas de redução de dimensionalidade como o PCA. Os modelos foram avaliados utilizando a métrica F1-Score equilibrada e comparados estatisticamente através do teste de Friedman. Os resultados indicam que a utilização de uma rede neural com uma camada oculta de 150 neurônios e máximo de 500 iterações obteve o melhor desempenho considerando taxa de acerto e custo computacional, e que a escolha do descritor LBP mostrou-se estatisticamente superior.*

1. Introdução

A classificação de imagens é uma das tarefas fundamentais da Inteligência Artificial, com aplicações que vão desde o diagnóstico médico até a operação de veículos autônomos. Um problema clássico e desafiador nesta área é a distinção entre imagens de cães e gatos

A relevância deste problema não reside em si na classificação de animais de estimação, mas em sua função como um benchmark canônico. O desafio "cão vs. gato" é notório pela sua dificuldade em aprendizado de máquina clássico devido a dois fatores: alta variância intra-classe (raças de cães, como Pinscher e Setter, são visualmente muito distintas entre si) e baixa variância inter-classe (um gato Ragdoll e um cão Setter podem compartilhar texturas e posturas semelhantes)

Resolver este problema de forma eficaz demonstra a robustez de um *pipeline* metodológico. Um *pipeline* (descritor + classificador) que obtém sucesso aqui pode ser adaptado para resolver problemas práticos em outros contextos, como controle de qualidade industrial (identificação de peças defeituosas vs. não defeituosas) ou análise de imagens médicas (tecido saudável vs. tecido doente), onde os desafios de variância são igualmente complexos.

Embora modelos de Aprendizado Profundo (*Deep Learning*) tenham mostrado alta performance, eles exigem vastos *datasets* e um custo computacional elevado. Os métodos clássicos de Aprendizado de Máquina (ML) supervisionado, quando pareados com descritores de características robustos, ainda são uma alternativa viável, oferecendo vantagens como menor custo computacional, maior interpretabilidade e eficácia em *datasets* de tamanho moderado.

Para que algoritmos clássicos de ML possam processar imagens, é necessário realizar a etapa de extração de características (feature extraction). Descritores como HOG (que captura forma) e LBP (que captura textura) são amplamente utilizados. Este trabalho tem como objetivo comparar sistematicamente o desempenho de diferentes classes de algoritmos de ML na tarefa "cão vs. gato", utilizando *features* extraídas por HOG e LBP, para identificar o *pipeline* de melhor custo-benefício

2. Metodologia

A metodologia deste trabalho seguiu o fluxo padrão de desenvolvimento de um projeto de Machine Learning, compreendendo a definição do *dataset*, pré-processamento, definição do ambiente experimental, treinamento dos modelos e avaliação estatística.

A biblioteca de Aprendizado de Máquina empregada foi a Scikit-learn (sklearn). A escolha desta biblioteca justifica-se por ser o padrão da indústria para Machine Learning em Python, oferecendo uma API unificada e eficiente para a implementação de todos os modelos de classificação utilizados neste trabalho (k-NN, Árvore de Decisão, Naive Bayes, MLP e diversos Comitês de Classificadores), bem como as rotinas de pré-processamento (scaling) e métricas de avaliação (F1-Score) (PEDREGOSA, 2011).

2.1. Pré-processamento

O conjunto de dados utilizado neste trabalho foi construído a partir de imagens de cães e gatos, totalizando 800 amostras do Oxford-IIIT Pet Dataset (*VISUAL GEOMETRY GROUP, 2012)*. O *dataset* foi balanceado, contendo duas classes principais: Cão (rótulo 0) e Gato (rótulo 1), com cerca de 400 imagens por classe. Para garantir a diversidade intra-classe, foram selecionadas duas raças distintas para cada classe, cada uma contendo cerca de duas centenas de imagens:

- **Classe Cão (0):** *Miniature Pinscher* e *English Setter*
- **Classe Gato (1):** *Birman* e *Ragdoll*

Para que os algoritmos de ML pudessem processar as imagens, foram utilizados descritores de características para gerar as bases de dados. O LBP (Local Binary Patterns) é um descritor de textura em tons de cinza. Ele funciona resumindo a estrutura local ao comparar cada pixel com sua vizinhança, gerando um número binário. Sua principal vantagem é a robustez a variações de iluminação. Para capturar padrões de textura em diferentes escalas, foram geradas bases variando o parâmetro de raio igual a 3, 6 e 12 pixels. O HOG (Histogram of Oriented Gradient) é um descritor de características utilizado para detecção de objetos em visão Computacional. A técnica contabiliza as ocorrências de orientação no gradiente em porções localizadas da imagem, conhecidas como "região de interesse" (ROI). Ele consegue computar uma grade densa de células uniformemente espaçadas e usar normalização de contraste local com sobreposição, o que melhora sua acurácia (SILVA, 2017). Para avaliar o impacto da granularidade deste descritor, foram geradas bases de dados variando o tamanho da célula: 8x8, 16x16, 20x20 e 32x32 pixels.

Antes da extração, as imagens foram redimensionadas para 128x128 e 256x256 pixels. Isso foi necessário porque os descritores de características necessitam de entradas de tamanho fixo. O uso de dois tamanhos distintos permitiu analisar o *trade-off* entre custo computacional e informação. O tamanho 128x128 é mais rápido de processar, mas pode perder detalhes finos, enquanto o 256x256 preserva esses detalhes ao custo de gerar vetores de características bem maiores. O objetivo foi ter resoluções diferentes para aplicar os classificadores e mensurar os resultados.

Adicionalmente, foram geradas bases de dados utilizando a Análise de Componentes Principais (PCA). O objetivo desta etapa foi avaliar o impacto da redução de dimensionalidade, uma técnica que ao remover características correlacionadas ou redundantes, visa: a) reduzir o custo computacional do treinamento, b) mitigar o risco de sobreajuste (*overfitting*) ao remover características correlacionadas ou redundantes. Foram definidos dois limiares de variância explicada para esta análise: 90% (uma compressão moderada) e 75% (um limiar mais agressivo).

Os vetores de características extraídos foram submetidos ao escalonamento (scaling). Esta etapa é crucial pois algoritmos como k-NN e MLP são sensíveis à distância e podem ser enviesados por atributos em escalas maiores.

2.3 Abordagem de Validação Dupla (Holdout e K-Fold)

Foram aplicadas **duas estratégias de validação** distintas para todos cada base, cujos resultados são apresentados lado a lado em todas as tabelas de experimentos: Holdout (70/30) e 10-fold Cross-Validation (k-fold).

1. **Holdout (70/30):** uma avaliação rápida que consiste em uma única divisão estática dos dados, onde 70% são usados para treino e 30% para teste. Embora seja computacionalmente rápido, seu resultado pode variar significativamente dependendo de quais amostras caíram aleatoriamente no conjunto de teste. No contexto deste trabalho, foi usado para a seleção preliminar de hiperparâmetros (como na filtragem do GridSearch do MLP) por ser mais veloz de ser implementado.

2. **10-fold Cross-Validation (k-fold):** uma avaliação mais robusta e confiável. O *dataset* é dividido em 10 partes ("folds"). O modelo é então treinado e testado 10 vezes; a cada vez, 9 partes são usadas para treino e 1 parte é usada para teste. O resultado final é a média do F1-Score das 10 execuções. Este método é de execução mais demorada, no entanto, minimiza o viés de uma única divisão aleatória e garante que todas as amostras do *dataset* sejam usadas tanto para treino quanto para teste.

2.4. Modelos de Classificação

Para abordar o problema, foi selecionado um portfólio de algoritmos que representa um espectro completo das principais abordagens de aprendizado de máquina supervisionado. A intenção não foi apenas encontrar um vencedor, mas comparar como diferentes formas de classificação lidam com os dados. A seleção inclui: um modelo baseado em instância (k-NN), um modelo probabilístico (Naive Bayes), um modelo baseado em regras (Árvore de Decisão), um modelo não-linear complexo (Rede Neural MLP) e, finalmente, métodos de Comitês (*Ensembles*) que combinam os anteriores. Esta seleção permite avaliar o *trade-off* entre simplicidade, interpretabilidade, custo computacional e poder preditivo.

Foram implementados cinco tipos de classificadores supervisionados para esta análise comparativa:

- **k-Nearest Neighbors (k-NN):** Um classificador baseado em instância (ou *lazy learner*) que rotula um novo dado com base no voto majoritário de seus "k" vizinhos mais próximos no espaço de características.

- **Árvore de Decisão (DT):** Um modelo *white-box* (interpretável) que aprende regras de decisão (sim/não) a partir dos dados, criando uma estrutura de fluxo hierárquica para chegar a uma classificação final.

- **Naive Bayes (NB):** Um classificador probabilístico rápido, baseado no Teorema de Bayes, que assume (ingenuamente) que todas as características (*features*) são independentes entre si ao calcular a probabilidade de uma amostra pertencer a

uma classe.

- **Rede Neural (MLP):** O Perceptron de Múltiplas Camadas (MLP) é um modelo não-linear inspirado no cérebro humano, composto por camadas de "neurônios" que aprendem padrões complexos nos dados através de um processo de otimização chamado *backpropagation*.

- **Comitês de Classificadores (Ensembles):** Métodos que combinam múltiplos modelos para obter uma predição mais robusta e reduzir o sobreajuste. Os comitês específicos avaliados neste trabalho foram:
    - **Bagging:** Treina múltiplos classificadores (do mesmo tipo) em diferentes subconjuntos de dados (criados por amostragem com reposição) e combina suas previsões por votação para reduzir a variância.
    - **Random Forest:** Uma especialização do Bagging que constrói múltiplas Árvores de Decisão, introduzindo aleatoriedade também na seleção de *features* em cada nó para aumentar a diversidade.
    - **Voting:**Combina as previsões de diferentes tipos de classificadores (ex: k-NN, NB, DT). Para este trabalho, foi utilizada a estratégia "soft voting**"**. Diferente do "hard voting" (que usa apenas a votação majoritária da classe prevista), o "soft voting" calcula a média das *probabilidades* de previsão de cada modelo. Esta abordagem é geralmente considerada mais robusta, pois permite que modelos que estão mais "confiantes" em suas previsões (ex: 95% de chance) tenham um peso maior na decisão final, em vez de tratar todos os votos com o mesmo peso.
    - **Stacking:** Um comitê de dois níveis onde um "meta-classificador" aprende a combinar de forma inteligente as previsões de um conjunto de classificadores base.

2.5. Avaliação Experimental e Estatística

A avaliação dos modelos utilizou a métrica **F1-Score (weighted)**. Embora o *dataset* seja balanceado, o F1-Score foi preferido em relação à Acurácia. Ela mede os acertos totais, já o F1-Score exige que o modelo tenha um bom desempenho equilibrado em *ambas* as classes (Cão e Gato), penalizando modelos que acertam muito uma classe, mas falham na outra. O modificador weighted garante uma média geral justa, ponderada pelo número de amostras em cada classe. Para comparar o desempenho entre os múltiplos algoritmos e bases de dados, foram usados testes estatísticos não-paramétricos:

1. **Teste de Friedman:** Utilizado para comparar os *rankings* de três ou mais algoritmos em múltiplos *datasets*. Testa a hipótese nula de que todos os algoritmos têm desempenho estatisticamente equivalente.

2. **Teste Post-hoc de Nemenyi:** Aplicado apenas quando o Teste de Friedman indicou uma diferença significativa (p-valor < 0.05). O Nemenyi realiza comparações par-a-par para identificar *quais* pares de modelos são significativamente diferentes.

A regra de desempate, caso não haja diferença estatística entre os melhores modelos, é escolher o modelo com a partir de uma avaliação de menor complexidade e custo computacional versus melhor desempenho médio.

Dado o grande número de *datasets* candidatos gerados (combinando resoluções 128/256, descritores HOG/LBP com múltiplos parâmetros, e PCA), uma etapa de seleção de bases foi necessária para focar a análise nos *datasets* mais promissores. Para isso, o algoritmo k-Nearest Neighbors (k-NN) foi empregado como um *baseline* rápido e de baixo custo computacional. Todas as bases geradas foram avaliadas com o k-NN (como visto na Figura 1), e as 12 bases que apresentaram o melhor desempenho (F1-Score weight foram selecionadas. Este conjunto final de 12 *datasets* foi, então, utilizado como o universo de dados padrão para todos os experimentos subsequentes com os demais classificadores.

Uma observação metodológica crucial desta etapa (baseada nos dados de todas as bases) foi a clara superioridade do descritor LBP; as bases HOG apresentaram desempenho pífio no *baseline* (como HOG_128_20x20_PCA-75 com F1 de 0.112), e, por isso, nenhuma base HOG de alta dimensionalidade se qualificou para a análise final.

3. Resultados e Discussão

A seguir, são apresentadas as análises estatísticas para cada classe de algoritmo e a comparação final.

3.1. k-Nearest Neighbors (k-NN)

Foram definidas de modo empírico, dez configurações de k-NN (k=1 a 10) para testar as bases. O Teste de Friedman para estas 10 configurações resultou em um p-valor = 0.014535. Como $p < 0.05$, a hipótese nula foi rejeitada, indicando que existe uma diferença estatística significativa entre os modelos. Embora o teste pós-hoc de Nemenyi não tenha isolado um par estatisticamente diferente (todos os p-valores > 0.05), o resultado do Friedman, que confirmou a diferença, nos leva a selecionar o modelo com o melhor desempenho médio. Portanto, o modelo k-NN com k=2 foi escolhido como o melhor, por apresentar o maior F1-Score médio (0.662). O fato de k=2 ter superado k=1 (média de 0.615) sugere que o modelo k=1 era muito sensível a ruídos ou *outliers* no *dataset*; a escolha de k=2 representa um equilíbrio entre simplicidade e maior robustez contra amostras mal rotuladas

kNN												
Base	Treinamento / Teste	1k	2k	3k	4k	5k	6k	7k	8k	9k	10k	Média
	k-fold10	0,739	0,731	0,776	0,775	0,789	0,780	0,801	0,798	0,801	0,788	0,790
LBP_256_6r.csv (50 atributos)	70/30	0,778	0,800	0,808	0,811	0,813	0,795	0,807	0,805	0,803	0,810	
	k-fold10	0,735	0,720	0,782	0,772	0,776	0,776	0,783	0,793	0,793	0,791	0,785
LBP_256_6r_PCA-90.csv (4 atributos)	70/30	0,693	0,751	0,793	0,808	0,807	0,821	0,812	0,832	0,829	0,843	
	k-fold10	0,734	0,743	0,767	0,771	0,777	0,773	0,772	0,773	0,780	0,771	0,761
LBP_256_12r_PCA-90.csv (9 atributos)	70/30	0,729	0,744	0,748	0,767	0,760	0,789	0,762	0,778	0,735	0,751	
	k-fold10	0,733	0,705	0,738	0,731	0,763	0,747	0,777	0,757	0,770	0,767	0,753
LBP_256_12r.csv (98 atributos)	70/30	0,756	0,761	0,736	0,767	0,757	0,756	0,752	0,765	0,764	0,754	
	k-fold10	0,680	0,679	0,744	0,730	0,756	0,748	0,746	0,738	0,747	0,748	0,729
LBP_256_3r.csv (26 atributos)	70/30	0,664	0,713	0,739	0,729	0,701	0,735	0,724	0,769	0,733	0,754	
	k-fold10	0,673	0,682	0,707	0,720	0,726	0,744	0,738	0,735	0,744	0,747	0,734
LBP_256_3r_PCA-90.csv (3 atributos)	70/30	0,675	0,741	0,723	0,746	0,753	0,763	0,756	0,766	0,754	0,780	
	k-fold10	0,600	0,642	0,591	0,642	0,594	0,628	0,562	0,604	0,559	0,588	0,583
HOG_128_32x32_PCA-75.csv (32 atributos)	70/30	0,603	0,664	0,599	0,638	0,532	0,616	0,444	0,565	0,472	0,512	
	k-fold10	0,605	0,660	0,563	0,624	0,549	0,604	0,536	0,569	0,507	0,541	0,536
HOG_128_32x32.csv (324 atributos)	70/30	0,601	0,661	0,476	0,582	0,436	0,532	0,382	0,460	0,376	0,459	
	k-fold10	0,559	0,610	0,511	0,575	0,467	0,514	0,430	0,464	0,428	0,449	0,411
HOG_128_20x20.csv (900 atributos)	70/30	0,412	0,528	0,356	0,479	0,275	0,340	0,200	0,252	0,185	0,185	
	k-fold10	0,518	0,568	0,497	0,519	0,446	0,471	0,434	0,469	0,411	0,431	0,400
HOG_256_32x32.csv (1764 atributos)	70/30	0,398	0,536	0,364	0,424	0,268	0,340	0,234	0,252	0,183	0,235	
	k-fold10	0,556	0,609	0,481	0,516	0,437	0,475	0,399	0,431	0,374	0,396	0,390
HOG_128_32x32_PCA-90.csv (64 atributos)	70/30	0,415	0,612	0,297	0,486	0,204	0,327	0,183	0,235	0,155	0,211	
	k-fold10	0,526	0,573	0,475	0,538	0,440	0,481	0,401	0,431	0,370	0,404	0,356
HOG_128_20x20_PCA-75.csv (71 atributos)	70/30	0,369	0,452	0,292	0,424	0,195	0,284	0,097	0,183	0,081	0,112	
MÉDIA =>		0,615	0,662	0,607	0,649	0,584	0,618	0,564	0,593	0,557	0,576	
DESVIO PADRÃO =>		0,124	0,087	0,170	0,127	0,208	0,176	0,234	0,213	0,245	0,231	

Figura 1. Tabela de acurácia do k-NN

```
Teste de Friedman - kNN:
p-valor: 0.014535
qui-quadrado: 20.602930
```

Figura 2. Friedman no k-NN

```
Teste Nemenyi (pós-hoc) - kNN:
             1k           2k           3k           4k           5k  \
1k   1.0000000000  0.8612982174  0.9999972552  0.3069317663  0.9999101856
2k   0.8612982174  1.0000000000  0.9737781924  0.9975805785  0.9922721387
3k   0.9999972552  0.9737781924  1.0000000000  0.5632961317  0.9999999941
4k   0.3069317663  0.9975805785  0.5632961317  1.0000000000  0.6967791716
5k   0.9999101856  0.9922721387  0.9999999941  0.6967791716  1.0000000000
6k   0.6311932810  0.9999972552  0.8612982174  0.9999809846  0.9328453346
7k   0.9998244085  0.4615596102  0.9922721387  0.0695928920  0.9737781924
8k   0.8612982174  1.0000000000  0.9737781924  0.9975805785  0.9922721387
9k   0.9997598138  0.4450004575  0.9908296072  0.0649848170  0.9700986180
10k  0.9999953418  0.9771048015  1.0000000000  0.5803746626  0.9999999982

             6k           7k           8k           9k           10k
1k   0.6311932810  0.9998244085  0.8612982174  0.9997598138  0.9999953418
2k   0.9999972552  0.4615596102  1.0000000000  0.4450004575  0.9771048015
3k   0.8612982174  0.9922721387  0.9737781924  0.9908296072  1.0000000000
4k   0.9999809846  0.0695928920  0.9975805785  0.0649848170  0.5803746626
5k   0.9328453346  0.9737781924  0.9922721387  0.9700986180  0.9999999982
6k   1.0000000000  0.2297649077  0.9999972552  0.2182420150  0.8720408291
7k   0.2297649077  1.0000000000  0.4615596102  1.0000000000  0.9908296072
8k   0.9999972552  0.4615596102  1.0000000000  0.4450004575  0.9771048015
9k   0.2182420150  1.0000000000  0.4450004575  1.0000000000  0.9891795289
10k  0.8720408291  0.9908296072  0.9771048015  0.9891795289  1.0000000000
```

Figura 3. Nemenyi em k-NN

3.2. Árvore de Decisão (DT)

Inicialmente foram definidas 10 configurações com melhores resultados a partir do dataset com melhores resultados no kNN, a base LBP_256_6r. Para a Árvore de

Decisão, foram definidas dez configurações distintas, variando os parâmetros de criterion (como 'gini', 'entropy' e 'log_loss') e max_depth (com profundidades de 2, 3, 4, 5 e 6). O Teste de Friedman, aplicado aos resultados (F1-Score) destas dez configurações nos 12 *datasets*, resultou em um p-valor = 0.289825.

Como p > 0.05, conclui-se que não há diferença estatística entre o desempenho de nenhuma das configurações. Seguindo a regra de desempate (complexidade vs. desempenho), o modelo da Conf. 10 (configuração gini com max_depth=2) foi escolhido como o melhor. Esta configuração representa a árvore menos complexa, pois utiliza a menor profundidade máxima. O fato de uma árvore tão rasa (`max_depth=2`) ter um desempenho estatisticamente idêntico ao de árvores mais complexas (com 5 ou 6 níveis) é uma limitação importante; sugere que, com estes descritores, o problema não se beneficia de regras de decisão mais profundas ou complexas.

Árvore de Decisão

Base	Treinamento / Teste	Conf. 1	Conf. 2	Conf. 3	Conf. 4	Conf. 5	Conf. 6	Conf. 7	Conf. 8	Conf. 9	Conf. 10	Média
	k-fold10	0,793	0,795	0,796	0,789	0,782	0,779	0,774	0,774	0,795	0,760	0,796
LBP_256_6r.csv (50 atributos)	70/30	0,821	0,821	0,816	0,813	0,804	0,804	0,796	0,796	0,821	0,787	
	k-fold10	0,793	0,779	0,755	0,749	0,786	0,797	0,794	0,794	0,779	0,795	0,784
LBP_256_6r_PCA-90.csv (4 atributos)	70/30	0,804	0,774	0,754	0,752	0,804	0,804	0,804	0,804	0,774	0,775	
	k-fold10	0,767	0,765	0,758	0,761	0,712	0,762	0,762	0,762	0,765	0,731	0,755
LBP_256_12r_PCA-90.csv (9 atributos)	70/30	0,778	0,765	0,755	0,764	0,712	0,762	0,762	0,762	0,765	0,731	
	k-fold10	0,765	0,737	0,717	0,741	0,774	0,762	0,762	0,762	0,737	0,769	0,758
LBP_256_12r.csv (98 atributos)	70/30	0,779	0,741	0,762	0,779	0,774	0,765	0,765	0,765	0,741	0,769	
	k-fold10	0,716	0,709	0,726	0,727	0,714	0,701	0,702	0,702	0,709	0,717	0,714
LBP_256_3r.csv (26 atributos)	70/30	0,708	0,697	0,754	0,741	0,712	0,704	0,704	0,704	0,697	0,733	
	k-fold10	0,738	0,750	0,746	0,754	0,748	0,746	0,746	0,746	0,750	0,705	0,743
LBP_256_3r_PCA-90.csv (3 atributos)	70/30	0,738	0,750	0,746	0,754	0,748	0,746	0,746	0,746	0,750	0,705	
	k-fold10	0,650	0,662	0,640	0,640	0,667	0,642	0,642	0,642	0,662	0,649	0,650
HOG_128_32x32_PCA-75.csv (32 atributos)	70/30	0,650	0,662	0,640	0,640	0,667	0,642	0,642	0,642	0,662	0,649	
	k-fold10	0,628	0,629	0,643	0,637	0,608	0,607	0,607	0,607	0,629	0,634	0,623
HOG_128_32x32.csv (324 atributos)	70/30	0,628	0,629	0,645	0,637	0,608	0,607	0,607	0,607	0,629	0,634	
	k-fold10	0,600	0,607	0,618	0,620	0,612	0,623	0,623	0,623	0,607	0,617	0,617
HOG_128_20x20.csv (900 atributos)	70/30	0,600	0,615	0,610	0,620	0,638	0,624	0,624	0,624	0,615	0,612	
	k-fold10	0,569	0,577	0,598	0,573	0,624	0,561	0,561	0,561	0,577	0,612	0,581
HOG_256_32x32.csv (1764 atributos)	70/30	0,569	0,577	0,598	0,573	0,624	0,561	0,561	0,561	0,577	0,612	
	k-fold10	0,616	0,626	0,618	0,614	0,637	0,650	0,650	0,650	0,626	0,642	0,635
HOG_128_32x32_PCA-90.csv (64 atributos)	70/30	0,616	0,626	0,618	0,614	0,675	0,650	0,650	0,650	0,626	0,649	
	k-fold10	0,594	0,552	0,584	0,600	0,616	0,612	0,612	0,612	0,552	0,617	0,595
HOG_128_20x20_PCA-75.csv (71 atributos)	70/30	0,594	0,552	0,584	0,600	0,616	0,612	0,612	0,612	0,552	0,617	
MÉDIA =>		0,688	0,683	0,687	0,687	0,694	0,688	0,688	0,688	0,683	0,688	
DESVIO PADRÃO =>		0,086	0,083	0,076	0,078	0,070	0,080	0,080	0,080	0,083	0,066	

Figura 4. Acurácias com Árvore de Decisão

```
••• Teste de Friedman - Árvore de Decisão:
    p-valor: 0.289825
    qui-quadrado: 9.658394
```

Figura 5. Friedman na Árvore de Decisão

3.3. Naive Bayes (NB)

Para a análise do Naive Bayes, foram testadas três variações do algoritmo (GaussianNB, MultinomialNB e ComplementNB). O Teste de Friedman para estas três variações resultou em um p-valor = 0.000772, confirmando que a escolha tem um impacto estatístico significativo no resultado. O teste pós-hoc de Nemenyi mostrou que o GaussianNB era estatisticamente superior ao MultinomialNB (p=0.0005) e estatisticamente equivalente ao ComplementNB (p=0.318). Dentro do grupo de melhor desempenho, o GaussianNB foi escolhido como o melhor, por ter a maior média de F1-Score (0.717). O melhor desempenho do GaussianNB não é surpreendente porque ele foi projetado para features contínuas (como os valores HOG/LBP), enquanto

MultinomialNB e ComplementNB é mais utilizado para trabalhar com para features discretas SCIKIT-LEARN DEVELOPERS (2025).

Naive Bayes

Base	Treinamento / Teste	GaugasianNB	MultinomialNB	ComplementNB	Média
	k-fold10	0,775	0,675	0,692	0,697
LBP_256_6r.csv (50 atributos)	70/30	0,708	0,689	0,643	
	k-fold10	0,824	0,713	0,775	0,741
LBP_256_6r_PCA-90.csv (4 atributos)	70/30	0,825	0,570	0,740	
	k-fold10	0,770	0,690	0,752	0,711
LBP_256_12r_PCA-90.csv (9 atributos)	70/30	0,765	0,578	0,711	
	k-fold10	0,739	0,583	0,588	0,647
LBP_256_12r.csv (98 atributos)	70/30	0,708	0,678	0,584	
	k-fold10	0,730	0,641	0,670	0,646
LBP_256_3r.csv (26 atributos)	70/30	0,675	0,550	0,612	
	k-fold10	0,751	0,687	0,711	0,699
LBP_256_3r_PCA-90.csv (3 atributos)	70/30	0,738	0,629	0,675	
	k-fold10	0,707	0,687	0,712	0,694
HOG_128_32x32_PCA-75.csv (32 atributos)	70/30	0,740	0,614	0,706	
	k-fold10	0,696	0,656	0,660	0,671
HOG_128_32x32.csv (324 atributos)	70/30	0,648	0,685	0,678	
	k-fold10	0,704	0,685	0,687	0,687
HOG_128_20x20.csv (900 atributos)	70/30	0,672	0,690	0,681	
	k-fold10	0,686	0,684	0,681	0,664
HOG_256_32x32.csv (1764 atributos)	70/30	0,603	0,666	0,662	
	k-fold10	0,690	0,704	0,716	0,696
HOG_128_32x32_PCA-90.csv (64 atributos)	70/30	0,691	0,633	0,741	
	k-fold10	0,687	0,672	0,698	0,678
HOG_128_20x20_PCA-75.csv (71 atributos)	70/30	0,678	0,639	0,694	
MÉDIA =>		0,717	0,654	0,686	
DESVIO PADRÃO =>		0,052	0,046	0,047	

Figura 6. Acurácias com Naive Bayes

```
Teste de Friedman - Naive Baynes:
p-valor: 0.000772
qui-quadrado: 14.333333
```

Figura 7. Friedman em Naive Bayes

```
Teste Nemenyi (pós-hoc) - Naive Baynes:
                GaugasianNB  MultinomialNB  ComplementNB
GaugasianNB    1.0000000000   0.0005130108  0.3186450818
MultinomialNB  0.0005130108   1.0000000000  0.0545006301
ComplementNB   0.3186450818   0.0545006301  1.0000000000
```

Figura 8. Nemenyi em Naive Bayes

3.4. Redes Neurais (MLP)

Para a análise das Redes Neurais (MLP), a metodologia seguiu as instruções da atividade prática. Inicialmente, um GridSearch abrangente (com 768 combinações de parâmetros como activation, hidden_layer_sizes, max_iter e solver) foi executado na melhor base de dados (selecionada na etapa do k-NN) usando a estratégia *Holdout (70/30)*.

```
param_grid = {
    'hidden_layer_sizes': [(50), (100), (120), (150), (70,70), (100,50)],
    'activation': ['identity', 'logistic', 'tanh', 'relu'],
    'solver': ['sgd', 'adam'],
    'learning_rate_init': [0.0001, 0.001, 0.01, 0.1],
    'max_iter': [500, 1000, 1500, 2000]
}
```

Figura 9. GridSearch

As 10 melhores configurações (em F1-Score) resultantes deste GridSearch foram selecionadas para a análise final. Estas configurações foram então executadas em todos os 12 *datasets*. O Teste de Friedman para estas dez configurações resultou em um p-valor = 0.199580. Como $p > 0.05$, a hipótese nula não foi rejeitada, indicando que não há diferença estatística significativa entre o desempenho destes 10 melhores modelos. Partimos então para uma análise de complexidade. Os modelos Conf. 2 e Conf. 5 foram identificados como os menos complexos do grupo, pois ambos exigem o menor número de iterações (max_iter=500). Entre estes dois, a Conf. 5 foi escolhida como a melhor, por apresentar o maior F1-Score médio (0.746186) e possuir uma arquitetura mais simples (uma camada de 150 neurônios). Os valores dela foram activation: 'relu', hidden_layer_sizes: (150,), max_iter': 500 e solver: 'adam'.

O GridSearch com 768 combinações representou o maior custo computacional na etapa de análise de modelos individuais, uma dificuldade prática significativa. A análise do comportamento dos parâmetros foi reveladora: o fato de max_iter=500 ter sido escolhido sugere que o modelo convergiu rapidamente, e iterações adicionais (como 1000, 1500 ou 2000) não trouxeram benefício estatístico, apenas aumentaram o custo de treinamento.

Redes Neurais												
Base	Treinamento / Teste	Conf. 1	Conf. 2	Conf. 3	Conf. 4	Conf. 5	Conf. 6	Conf. 7	Conf. 8	Conf. 9	Conf. 10	Média
	k-fold10	0,796211	0,796211	0,796211	0,796211	0,814198	0,814198	0,814198	0,814198	0,816773	0,816773	0,817698
LBP_256_6r.csv (50 atributos)	70/30	0,829125	0,829125	0,829125	0,829125	0,827498	0,827498	0,827498	0,827498	0,826139	0,826139	
	k-fold10	0,816761	0,816761	0,816761	0,816761	0,794224	0,794224	0,794224	0,794224	0,773097	0,773097	0,796534
LBP_256_6r_PCA-90.csv (4 atributos)	70/30	0,812081	0,812081	0,812081	0,812081	0,784860	0,784860	0,784860	0,784860	0,776395	0,776395	
	k-fold10	0,803252	0,803252	0,803252	0,803252	0,767204	0,767204	0,767204	0,767204	0,761155	0,761155	0,778895
LBP_256_12r_PCA-90.csv (9 atributos)	70/30	0,812220	0,812220	0,812220	0,812220	0,743966	0,743966	0,743966	0,743966	0,774513	0,774513	
	k-fold10	0,808163	0,808163	0,808163	0,808163	0,810574	0,810574	0,810574	0,810574	0,816687	0,816687	0,805553
LBP_256_12r.csv (98 atributos)	70/30	0,798575	0,798575	0,798575	0,798575	0,806275	0,806275	0,806275	0,806275	0,791669	0,791669	
	k-fold10	0,757202	0,757202	0,757202	0,757202	0,776249	0,776249	0,776249	0,776249	0,785260	0,785260	0,772661
LBP_256_3r.csv (26 atributos)	70/30	0,773356	0,773356	0,773356	0,773356	0,776931	0,776931	0,776931	0,776931	0,773869	0,773869	
	k-fold10	0,757457	0,757457	0,757457	0,757457	0,752478	0,752478	0,752478	0,752478	0,742126	0,742126	0,760170
LBP_256_3r_PCA-90.csv (3 atributos)	70/30	0,762144	0,762144	0,762144	0,762144	0,781183	0,781183	0,781183	0,781183	0,753048	0,753048	
	k-fold10	0,718421	0,718421	0,718421	0,718421	0,714337	0,714337	0,714337	0,714337	0,738403	0,738403	0,712442
HOG_128_32x32_PCA-75.csv (32 atributos)	70/30	0,715646	0,715646	0,715646	0,715646	0,699422	0,699422	0,699422	0,699422	0,690368	0,690368	
	k-fold10	0,702736	0,702736	0,702736	0,702736	0,728342	0,728342	0,728342	0,728342	0,719423	0,719423	0,697038
HOG_128_32x32.csv (324 atributos)	70/30	0,662330	0,662330	0,662330	0,662330	0,680528	0,680528	0,680528	0,680528	0,703084	0,703084	
	k-fold10	0,675082	0,675082	0,675082	0,675082	0,697658	0,697658	0,697658	0,697658	0,716876	0,716876	0,676602
HOG_128_20x20.csv (900 atributos)	70/30	0,646250	0,646250	0,646250	0,646250	0,660707	0,660707	0,660707	0,660707	0,689745	0,689745	
	k-fold10	0,667084	0,667084	0,667084	0,667084	0,672637	0,672637	0,672637	0,672637	0,677689	0,677689	0,676927
HOG_256_32x32.csv (1764 atributos)	70/30	0,655923	0,655923	0,655923	0,655923	0,707488	0,707488	0,707488	0,707488	0,685320	0,685320	
	k-fold10	0,729726	0,729726	0,729726	0,729726	0,729038	0,729038	0,729038	0,729038	0,708676	0,708676	0,718147
HOG_128_32x32_PCA-90.csv (64 atributos)	70/30	0,701468	0,701468	0,701468	0,701468	0,717459	0,717459	0,717459	0,717459	0,717410	0,717410	
	k-fold10	0,697729	0,697729	0,697729	0,697729	0,740586	0,740586	0,740586	0,740586	0,710645	0,710645	0,714935
HOG_128_20x20_PCA-75.csv (71 atributos)	70/30	0,702889	0,702889	0,702889	0,702889	0,724615	0,724615	0,724615	0,724615	0,707063	0,707063	
MÉDIA =>		0,741743	0,741743	0,741743	0,741743	0,746186	0,746186	0,746186	0,746186	0,743976	0,743976	
DESVIO PADRÃO =>		0,059	0,059	0,059	0,059	0,048	0,048	0,048	0,048	0,045	0,045	

Figura 10. Acurácias com MLP

```
Teste de Friedman - Redes Neurais:
p-valor: 0.199580
qui-quadrado: 12.250000
```

Figura 11. Friedman em MLP

3.5. Comitês de Classificadores

Os Comitês de Classificadores representam a abordagem mais complexa e com maior custo computacional deste estudo. A principal dificuldade encontrada nesta etapa foi o alto custo de processamento (mencionado na conclusão), pois cada comitê exige o treinamento de múltiplos modelos base (de 5 a 100). A limitação teórica é que, se os classificadores base forem fracos ou seus erros forem correlacionados, o comitê pode falhar em produzir um resultado melhor.

Os Comitês de Classificadores, também conhecidos como *Ensemble Methods*, são uma técnica avançada em aprendizado de máquina supervisionado que, em vez de depender de um único modelo, combina as previsões de múltiplos classificadores base (como k-NN, Árvores de Decisão ou MLPs). O objetivo fundamental é obter um resultado final que seja mais preciso, robusto e com menor sobreajuste (*overfitting*) do que qualquer um dos modelos individuais. A predição final é geralmente determinada por um sistema de votação (majoritária ou ponderada), onde os erros de um classificador podem ser compensados pelos acertos dos outros, aproveitando a "sabedoria do coletivo" para tomar uma decisão mais inteligente.

3.5.1 Bagging

Para a análise do Bagging, foram testadas 12 configurações, variando o estimador base (k-NN, DT, NB, MLP) e o número de estimadores (10, 15, 20 ou 30). O Teste de Friedman resultou em um p-valor = 0.000...5696, confirmando que existe uma diferença estatística significativa entre os modelos. O teste pós-hoc de Nemenyi revelou que os

modelos DT, NB e MLP formavam um grupo de desempenho superior, estatisticamente equivalentes entre si. O modelo MLP(10) foi escolhido como o melhor dado ter o maior F1-Score médio (0.7472) de todos os modelos e, ao mesmo tempo, ser o modelo menos complexo (10 estimadores) dentro da família de melhor desempenho (MLP). O comportamento deste parâmetro sugere que o estimador base (MLP) já era muito forte sozinho; aumentar o número de estimadores para 20 ou 30 não adicionou diversidade suficiente para melhorar a média geral, apenas aumentou o custo computacional.

Bagging														
Base	Treinamento / Teste	kNN(10)	kNN(20)	kNN(30)	DT(10)	DT(20)	DT(30)	NB(10)	NB(20)	NB(30)	MLP(10)	MLP(20)	MLP(30)	Média
	k-fold10	0,7937	0,7926	0,7951	0,8199	0,8198	0,8172	0,7657	0,7631	0,7656	0,8122	0,8147	0,8135	0,7791
LBP_256_6r.csv (50 atributos)	70/30	0,7667	0,7583	0,7624	0,7959	0,8043	0,8043	0,7082	0,7082	0,7123	0,8165	0,8165	0,8165	
	k-fold10	0,7927	0,7990	0,7953	0,7995	0,8044	0,7958	0,8235	0,8260	0,8297	0,8123	0,8111	0,8111	0,8045
LBP_256_6r_PCA-90.csv (4 atributos)	70/30	0,7959	0,7959	0,7960	0,7959	0,8082	0,8039	0,8041	0,8081	0,8250	0,7788	0,7788	0,7829	
	k-fold10	0,7786	0,7760	0,7785	0,7793	0,7793	0,7767	0,7663	0,7737	0,7699	0,8120	0,8083	0,8083	0,7783
LBP_256_12r_PCA-90.csv (9 atributos)	70/30	0,7833	0,7959	0,7959	0,7541	0,7751	0,7751	0,7574	0,7698	0,7698	0,7998	0,7916	0,7914	
	k-fold10	0,7826	0,7774	0,7697	0,8007	0,8108	0,8107	0,7369	0,7407	0,7418	0,8159	0,8097	0,8109	0,7649
LBP_256_12r.csv (98 atributos)	70/30	0,7497	0,7499	0,7497	0,7960	0,7751	0,7875	0,7037	0,7078	0,7078	0,7831	0,7873	0,7790	
	k-fold10	0,7341	0,7331	0,7325	0,7398	0,7535	0,7398	0,7314	0,7340	0,7315	0,7810	0,7848	0,7835	0,7326
LBP_256_3r.csv (26 atributos)	70/30	0,7292	0,7459	0,7456	0,7293	0,7460	0,7418	0,6751	0,6835	0,6877	0,7581	0,7621	0,7621	
	k-fold10	0,7409	0,7434	0,7447	0,7469	0,7410	0,7473	0,7592	0,7641	0,7679	0,7572	0,7522	0,7548	0,7423
LBP_256_3r_PCA-90.csv (3 atributos)	70/30	0,7332	0,7334	0,7374	0,7335	0,7293	0,7376	0,7335	0,7377	0,7377	0,7209	0,7251	0,7209	
	k-fold10	0,5580	0,5554	0,5534	0,6832	0,6970	0,7020	0,7160	0,7159	0,7107	0,7248	0,7208	0,7247	0,6685
HOG_128_32x32_PCA-75.csv (32 atributos)	70/30	0,5685	0,6012	0,5714	0,6960	0,6835	0,6918	0,7450	0,7363	0,7407	0,7199	0,7112	0,7112	
	k-fold10	0,5135	0,4950	0,4922	0,6554	0,6735	0,6749	0,6817	0,6839	0,6827	0,7157	0,7181	0,7102	0,6205
HOG_128_32x32.csv (324 atributos)	70/30	0,5010	0,5122	0,5082	0,6467	0,6433	0,6798	0,6523	0,6485	0,6400	0,7090	0,7056	0,7184	
	k-fold10	0,4176	0,3986	0,3934	0,6602	0,6833	0,6913	0,7072	0,7086	0,7047	0,6867	0,6831	0,6790	0,5954
HOG_128_20x20.csv (900 atributos)	70/30	0,4563	0,4482	0,4482	0,5812	0,5955	0,5968	0,6817	0,6769	0,6724	0,6993	0,6940	0,6908	
	k-fold10	0,4103	0,4051	0,4103	0,6497	0,6877	0,7002	0,6796	0,6808	0,6845	0,6716	0,6735	0,6773	0,5887
HOG_256_32x32.csv (1764 atributos)	70/30	0,4485	0,4689	0,4613	0,6359	0,6367	0,6068	0,6108	0,6231	0,6192	0,6830	0,7002	0,6792	
	k-fold10	0,3760	0,3607	0,3601	0,6888	0,6972	0,6983	0,6997	0,7010	0,7023	0,7335	0,7298	0,7283	0,6039
HOG_128_32x32_PCA-90.csv (64 atributos)	70/30	0,3952	0,3997	0,3867	0,6833	0,6794	0,6752	0,7035	0,7035	0,7035	0,7303	0,7016	0,7050	
	k-fold10	0,3704	0,3629	0,3628	0,6666	0,6667	0,6700	0,6921	0,6832	0,6905	0,6944	0,7067	0,7004	0,5918
HOG_128_20x20_PCA-75.csv (71 atributos)	70/30	0,4064	0,3977	0,3977	0,6748	0,6824	0,6701	0,6744	0,6781	0,6781	0,7159	0,7242	0,7112	
MÉDIA =>		0,6084	0,6086	0,6062	0,7172	0,7239	0,7248	0,7170	0,7190	0,7198	0,7472	0,7463	0,7446	
DESVIO PADRÃO =>		0,168	0,171	0,172	0,066	0,064	0,063	0,048	0,048	0,051	0,048	0,047	0,049	

Figura 12. Acurácias com Bagging

```
Teste de Friedman - Bagging:
p-valor: 0.00000000000005696455800094591761
qui-quadrado: 87.272142
```

Figura 13. Friedman em Bagging

3.5.2 Random Forest

Nesta análise foram testadas 8 configurações, variando o critério ('gini', 'entropy') e o número de estimadores (10, 20, 30 e 100). O Teste de Friedman resultou em um p-valor = 0.000...08, confirmando que existe uma diferença estatística significativa entre os modelos. O teste pós-hoc de Nemenyi revelou que os modelos com 30 e 100 estimadores (GINI(30), ENTROPY(30), GINI(100), ENTROPY(100)) formavam um grupo de desempenho superior, estatisticamente equivalentes entre si (ex: ENTROPY(100) vs ENTROPY(30), p=0.262). O modelo ENTROPY(30) foi escolhido como o melhor, pois, embora o ENTROPY(100) tivesse a maior média numérica (0.7442), o ENTROPY(30) oferece um desempenho estatisticamente idêntico com uma complexidade muito menor (30 estimadores vs. 100).

Random Forest										
Base	Treinamento / Teste	GINI(10)	GINI(20)	GINI(30)	GINI(100)	ENTROPY(10)	ENTROPY(20)	ENTROPY(30)	ENTROPY(100)	Média
	k-fold10	0,8004	0,8168	0,8158	0,8234	0,8169	0,8245	0,8345	0,8346	0,8180
LBP_256_6r.csv (50 atributos)	70/30	0,7917	0,8209	0,8333	0,8125	0,8085	0,8251	0,8167	0,8125	
	k-fold10	0,7768	0,7897	0,7870	0,7971	0,7742	0,7870	0,7970	0,7960	0,7843
LBP_256_6r_PCA-90.csv (4 atributos)	70/30	0,7751	0,7538	0,7749	0,7955	0,7917	0,7708	0,7916	0,7913	
	k-fold10	0,7638	0,7959	0,7893	0,7919	0,7742	0,7805	0,7868	0,8005	0,7763
LBP_256_12r_PCA-90.csv (9 atributos)	70/30	0,7626	0,7876	0,7543	0,7709	0,7624	0,7540	0,7668	0,7792	
	k-fold10	0,7992	0,8120	0,8072	0,8035	0,7980	0,7995	0,8033	0,8122	0,8020
LBP_256_12r.csv (98 atributos)	70/30	0,7876	0,8043	0,8043	0,7959	0,8041	0,7876	0,8126	0,8001	
	k-fold10	0,7412	0,7326	0,7494	0,7557	0,7259	0,7396	0,7641	0,7730	0,7535
LBP_256_3r.csv (26 atributos)	70/30	0,7501	0,7581	0,7582	0,7747	0,7335	0,7668	0,7585	0,7751	
	k-fold10	0,7061	0,7191	0,7269	0,7320	0,7082	0,7126	0,7263	0,7316	0,7199
LBP_256_3r_PCA-90.csv (3 atributos)	70/30	0,6708	0,7168	0,7334	0,7210	0,7167	0,7376	0,7292	0,7293	
	k-fold10	0,6816	0,6967	0,7170	0,7458	0,6571	0,6994	0,7015	0,7454	0,6987
HOG_128_32x32_PCA-75.csv (32 atributos)	70/30	0,6407	0,6585	0,6832	0,6908	0,6901	0,7290	0,7252	0,7167	
	k-fold10	0,6580	0,6763	0,7045	0,7205	0,6712	0,7067	0,7104	0,7370	0,6922
HOG_128_32x32.csv (324 atributos)	70/30	0,6498	0,6752	0,6794	0,7328	0,6918	0,6543	0,6911	0,7161	
	k-fold10	0,6090	0,6648	0,6856	0,7085	0,6242	0,6615	0,6698	0,7144	0,6651
HOG_128_20x20.csv (900 atributos)	70/30	0,6085	0,6710	0,6752	0,6876	0,6706	0,6701	0,6543	0,6660	
	k-fold10	0,6291	0,6389	0,6639	0,6932	0,6038	0,6627	0,6800	0,6994	0,6477
HOG_256_32x32.csv (1764 atributos)	70/30	0,5985	0,5909	0,6407	0,6787	0,6042	0,6377	0,6627	0,6794	
	k-fold10	0,6394	0,6603	0,6738	0,7023	0,6469	0,6869	0,6948	0,6931	0,6787
HOG_128_32x32_PCA-90.csv (64 atributos)	70/30	0,6344	0,6787	0,6827	0,7002	0,6830	0,6913	0,7167	0,6752	
	k-fold10	0,6446	0,6524	0,6755	0,6774	0,6144	0,6503	0,6779	0,6985	0,6606
HOG_128_20x20_PCA-75.csv (71 atributos)	70/30	0,6522	0,6496	0,6663	0,6746	0,6729	0,6211	0,6585	0,6835	

Figura 14. Acurácias com Random Forest

```
Teste de Friedman - Random Forest:
p-valor: 0.000000000000008
qui-quadrado: 81.105367793240575
```

Figura 15. Friedman em Random Forest

```
••• Teste Nemenyi (pós-hoc) Random Forest:
                 GINI(10)      GINI(20)      GINI(30)     GINI(100)  \
GINI(10)      1.0000000000  0.0637642198  0.0003949457  0.0000000012
GINI(20)      0.0637642198  1.0000000000  0.8645403532  0.0063086321
GINI(30)      0.0003949457  0.8645403532  1.0000000000  0.3110429385
GINI(100)     0.0000000012  0.0063086321  0.3110429385  1.0000000000
ENTROPY(10)   0.7733741319  0.8645403532  0.1027171486  0.0000129616
ENTROPY(20)   0.0346688578  0.9999992081  0.9381996495  0.0132067772
ENTROPY(30)   0.0000010010  0.1694969499  0.9381996495  0.9590595332
ENTROPY(100)  0.0000000003  0.0032182781  0.2194880343  0.9999997286

                ENTROPY(10)   ENTROPY(20)   ENTROPY(30)  ENTROPY(100)
GINI(10)      0.7733741319  0.0346688578  0.0000010010  0.0000000003
GINI(20)      0.8645403532  0.9999992081  0.1694969499  0.0032182781
GINI(30)      0.1027171486  0.9381996495  0.9381996495  0.2194880343
GINI(100)     0.0000129616  0.0132067772  0.9590595332  0.9999997286
ENTROPY(10)   1.0000000000  0.7559810383  0.0020142765  0.0000050924
ENTROPY(20)   0.7559810383  1.0000000000  0.2628395880  0.0070326299
ENTROPY(30)   0.0020142765  0.2628395880  1.0000000000  0.9110668846
ENTROPY(100)  0.0000050924  0.0070326299  0.9110668846  1.0000000000
```

Figura 16. Nemenyi em Random Forest

3.5.3 Voting

Para o comitê Voting foram testadas 4 configurações, variando o número de classificadores base (5, 10, 15 e 20) . O Teste de Friedman resultou em um p-valor = 0.008165. Como $p < 0.05$, a hipótese nula foi rejeitada, confirmando que existe uma diferença estatística significativa entre os modelos.

O teste pós-hoc de Nemenyi foi aplicado, mas não apontou um único grupo estatisticamente superior (a maioria dos pares, como (5) vs. (10) [p=0.64] e (5) vs. (20) [p=0.77], foram considerados equivalentes). O modelo Classificadores(5) foi escolhido como o melhor, pois apresentou o maior F1-Score médio (0.7317) e é, simultaneamente, o modelo menos complexo (5 classificadores**)**.

Voting						
Base	Treinamento / Teste	Classificadores(5)	Classificadores(10)	Classificadores(15)	Classificadores(20)	Média
	k-fold10	0,7995	0,7996	0,8071	0,8095	0,8041
LBP_256_6r.csv (50 atributos)	70/30	0,7876	0,8126	0,8084	0,8084	
	k-fold10	0,8095	0,8096	0,8045	0,8070	0,8075
LBP_256_6r_PCA-90.csv (4 atributos)	70/30	0,8085	0,8168	0,8042	0,8000	
	k-fold10	0,7896	0,7970	0,7907	0,8009	0,7856
LBP_256_12r_PCA-90.csv (9 atributos)	70/30	0,7750	0,7709	0,7749		
	k-fold10	0,7959	0,7871	0,7895	0,8021	0,7937
LBP_256_12r.csv (98 atributos)	70/30	0,7832	0,8001	0,7960	0,7960	
	k-fold10	0,7600	0,7511	0,7639	0,7628	0,7475
LBP_256_3r.csv (26 atributos)	70/30	0,7418	0,7292	0,7376	0,7334	
	k-fold10	0,7425	0,7462	0,7460	0,7489	0,7402
LBP_256_3r_PCA-90.csv (3 atributos)	70/30	0,7335	0,7334	0,7335	0,7377	
	k-fold10	0,6993	0,6846	0,6950	0,7096	0,7041
HOG_128_32x32_PCA-75.csv (32 atributos)	70/30	0,7025	0,7069	0,7021	0,7328	
	k-fold10	0,6942	0,6870	0,6820	0,6979	0,6752
HOG_128_32x32.csv (324 atributos)	70/30	0,6752	0,6651	0,6500	0,6500	
	k-fold10	0,6827	0,6787	0,6717	0,6721	0,6692
HOG_128_20x20.csv (900 atributos)	70/30	0,6630	0,6577	0,6496	0,6784	
	k-fold10	0,6840	0,6426	0,6308	0,6497	0,6456
HOG_256_32x32.csv (1764 atributos)	70/30	0,6344	0,6401	0,6355	0,6479	
	k-fold10	0,7020	0,6633	0,6588	0,6797	0,6764
HOG_128_32x32_PCA-90.csv (64 atributos)	70/30	0,6901	0,6772	0,6702	0,6702	
	k-fold10	0,6988	0,6638	0,6787	0,6808	0,6736
HOG_128_20x20_PCA-75.csv (71 atributos)	70/30	0,7087	0,6459	0,6480	0,6642	
MÉDIA =>		0,7317	0,7236	0,7220	0,7278	
DESVIO PADRÃO =>		0,052	0,063	0,064	0,060	

Figura 17. Acurácias com Voting

```
Teste de Friedman - Voting:
p-valor: 0.008165
qui-quadrado: 11.782979
```

Figura 18. Friedman em Voting

```
Teste Nemenyi (pós-hoc) - Voting:
                 Classificadores5  Classificadores10  Classificadores15  \
Classificadores5       1.0000000000       0.6433556312       0.1135810989
Classificadores10      0.6433556312       1.0000000000       0.7126204852
Classificadores15      0.1135810989       0.7126204852       1.0000000000
Classificadores20      0.7775705165       0.1453533877       0.0078586700

                 Classificadores20
Classificadores5       0.7775705165
Classificadores10      0.1453533877
Classificadores15      0.0078586700
Classificadores20      1.0000000000
```

Figura 19. Nemenyi em Voting

3.5.4 Stacking

Para a análise do Stacking, foram testadas 4 configurações, variando o número de classificadores base (5, 10, 15 e 20). O Teste de Friedman resultou em um p-valor = 0.080195. Como p > 0.05, a hipótese nula não foi rejeitada, indicando que NÃO há diferença estatística significativa entre os modelos. Seguindo a regra de desempate "complexidade versos desempenho médio", o modelo Classificadores(5) foi escolhido como o melhor. Embora o Classificadores(20) tenha a maior média numérica (0.7550), o Classificadores(5) é o modelo menos complexo e seu desempenho é estatisticamente idêntico ao dos modelos mais complexos.

Stacking						
Base	Treinamento / Teste	Classificadores(5)	Classificadores(10)	Classificadores(15)	Classificadores(20)	Média
	k-fold10	0,8171	0,8121	0,8108	0,8082	0,8076
LBP_256_6r.csv (50 atributos)	70/30	0,8043	0,8042	0,8083	0,7959	
	k-fold10	0,8210	0,8159	0,8134	0,8134	0,8142
LBP_256_6r_PCA-90.csv (4 atributos)	70/30	0,8084	0,8167	0,7999	0,8248	
	k-fold10	0,8085	0,8072	0,8061	0,8160	0,7968
LBP_256_12r_PCA-90.csv (9 atributos)	70/30	0,7791	0,7873	0,7873	0,7832	
	k-fold10	0,8048	0,8048	0,8008	0,8045	0,8029
LBP_256_12r.csv (98 atributos)	70/30	0,7914	0,7957	0,8126	0,8084	
	k-fold10	0,7680	0,7717	0,7820	0,7844	0,7663
LBP_256_3r.csv (26 atributos)	70/30	0,7542	0,7584	0,7664	0,7455	
	k-fold10	0,7506	0,7507	0,7619	0,7596	0,7440
LBP_256_3r_PCA-90.csv (3 atributos)	70/30	0,7293	0,7293	0,7332	0,7374	
	k-fold10	0,7432	0,7407	0,7409	0,7447	0,7329
HOG_128_32x32_PCA-75.csv (32 atributos)	70/30	0,7060	0,7224	0,7343	0,7312	
	k-fold10	0,7201	0,7255	0,7304	0,7383	0,7154
HOG_128_32x32.csv (324 atributos)	70/30	0,7011	0,7040	0,6954	0,7084	
	k-fold10	0,7144	0,7219	0,7280	0,7257	0,7189
HOG_128_20x20.csv (900 atributos)	70/30	0,6992	0,7179	0,7179	0,7258	
	k-fold10	0,6917	0,6952	0,6967	0,6849	0,6762
HOG_256_32x32.csv (1764 atributos)	70/30	0,6498	0,6580	0,6620	0,6709	
	k-fold10	0,7345	0,7270	0,7282	0,7280	0,7269
HOG_128_32x32_PCA-90.csv (64 atributos)	70/30	0,7236	0,7188	0,7232	0,7316	
	k-fold10	0,7110	0,7146	0,7235	0,7260	0,7241
HOG_128_20x20_PCA-75.csv (71 atributos)	70/30	0,7290	0,7283	0,7369	0,7232	
MÉDIA =>		0,7483	0,7512	0,7542	0,7550	
DESVIO PADRÃO =>		0,047	0,045	0,044	0,044	

Figura 20. Acurácias com Stacking

```
Teste de Friedman - Stacking:
p-valor: 0.080195
qui-quadrado: 6.753191
```

Figura 21. Friedman no Stacking

3.5.4 Melhores Comitês de Classificadores

Por último, foi realizada uma análise final comparando os quatro modelos vencedores de cada comitê: MLP(10), ENTROPY(30), Voting(5) e Stacking(5) - o nome dos dois últimos foram alterados para facilitar a compreensão de qual comitê se trata. O Teste de Friedman resultou em um p-valor = 0.000255, confirmando que existe uma diferença estatística significativa entre os melhores comitês.

O teste pós-hoc de Nemenyi revelou que MLP(10) e Stacking(5) formavam o grupo de desempenho superior, sendo estatisticamente equivalentes um ao outro (p=0.944) e estatisticamente superiores aos outros comitês (ex: Stacking(5) vs. Voting(5), p=0.0012). O modelo Stacking(5) foi escolhido como o melhor comitê geral.

A justificativa é que, dentro do grupo estatisticamente superior, ele possui o maior F1-Score médio (0.7483) e é o modelo menos complexo (5 classificadores) em comparação com o MLP com 10 estimadores.

Comitês de Classificadores						
Base	Treinamento / Teste	MLP(10)	ENTROPY(30)	Voting(5)	Stacking(5)	Média
	k-fold10	0,8122	0,8345	0,7995	0,8171	0,8111
LBP_256_6r.csv (50 atributos)	70/30	0,8165	0,8167	0,7876	0,8043	
	k-fold10	0,8123	0,7970	0,8095	0,8210	0,8034
LBP_256_6r_PCA-90.csv (4 atributos)	70/30	0,7788	0,7916	0,8085	0,8084	
	k-fold10	0,8120	0,7868	0,7896	0,8085	0,7897
LBP_256_12r_PCA-90.csv (9 atributos)	70/30	0,7998	0,7668	0,7750	0,7791	
	k-fold10	0,8159	0,8033	0,7959	0,8048	0,7988
LBP_256_12r.csv (98 atributos)	70/30	0,7831	0,8126	0,7832	0,7914	
	k-fold10	0,7810	0,7641	0,7600	0,7680	0,7607
LBP_256_3r.csv (26 atributos)	70/30	0,7581	0,7585	0,7418	0,7542	
	k-fold10	0,7572	0,7263	0,7425	0,7506	0,7362
LBP_256_3r_PCA-90.csv (3 atributos)	70/30	0,7209	0,7292	0,7335	0,7293	
	k-fold10	0,7248	0,7015	0,6993	0,7432	0,7153
HOG_128_32x32_PCA-75.csv (32 atributos)	70/30	0,7199	0,7252	0,7025	0,7060	
	k-fold10	0,7157	0,7104	0,6942	0,7201	0,7021
HOG_128_32x32.csv (324 atributos)	70/30	0,7090	0,6911	0,6752	0,7011	
	k-fold10	0,6867	0,6698	0,6827	0,7144	0,6837
HOG_128_20x20.csv (900 atributos)	70/30	0,6993	0,6543	0,6630	0,6992	
	k-fold10	0,6716	0,6800	0,6840	0,6917	0,6697
HOG_256_32x32.csv (1764 atributos)	70/30	0,6830	0,6627	0,6344	0,6498	
	k-fold10	0,7335	0,6948	0,7020	0,7345	0,7157
HOG_128_32x32_PCA-90.csv (64 atributos)	70/30	0,7303	0,7167	0,6901	0,7236	
	k-fold10	0,6944	0,6779	0,6988	0,7110	0,6993
HOG_128_20x20_PCA-75.csv (71 atributos)	70/30	0,7159	0,6585	0,7087	0,7290	
MÉDIA =>		0,7472	0,7346	0,7317	0,7483	
DESVIO PADRÃO =>		0,048	0,056	0,052	0,047	

Figura 22. Acurácias dos Classificadores

```
Teste Nemenyi (pós-hoc) - Comitês de Classificadores:
                MLP10   ENTROPY(30)       Voting5     Stacking5
MLP10      1.0000000000  0.0874663861  0.0094526283  0.9441109087
ENTROPY(30) 0.0874663861  1.0000000000  0.8623825783  0.0191296412
Voting5    0.0094526283  0.8623825783  1.0000000000  0.0012850014
Stacking5  0.9441109087  0.0191296412  0.0012850014  1.0000000000
```

Figura 23. Nemenyi nos Classificadores

```
Teste de Friedman - Comitês de Classificadores:
p-valor: 0.000255
qui-quadrado: 19.150000
```

Figura 24. Friedman nos Classificadores

3.6. Análise Comparativa dos Melhores Modelo

Para eleger o melhor modelo geral, os cinco vencedores das etapas anteriores (kNN(2), Árvore de Decisão (Conf. 10), Naive Baunes Gaussian NB, Rede Neural

(Conf. 5) e Comitê de Classificadores (Stacking 5)) foram comparados.

Melhores Modelos

Base	Treinamento / Teste	kNN(2)	DT Conf. 10	GaugasianNB	RedeNeural (Conf. 5)	Stacking(5)	Média
	k-fold10	0,731	0,760	0,775	0,814198	0,8171	0,775
LBP_256_6r.csv (50 atributos)	70/30	0,800	0,787	0,708	0,827498	0,8043	
	k-fold10	0,720	0,795	0,824	0,794224	0,8210	0,784
LBP_256_6r_PCA-90.csv (4 atributos)	70/30	0,751	0,775	0,825	0,784860	0,8084	
	k-fold10	0,743	0,731	0,770	0,767204	0,8085	0,749
LBP_256_12r_PCA-90.csv (9 atributos)	70/30	0,744	0,731	0,765	0,743966	0,7791	
	k-fold10	0,705	0,769	0,739	0,810574	0,8048	0,758
LBP_256_12r.csv (98 atributos)	70/30	0,761	0,769	0,708	0,806275	0,7914	
	k-fold10	0,679	0,717	0,730	0,776249	0,7680	0,725
LBP_256_3r.csv (26 atributos)	70/30	0,713	0,733	0,675	0,776931	0,7542	
	k-fold10	0,682	0,705	0,751	0,752478	0,7506	0,732
LBP_256_3r_PCA-90.csv (3 atributos)	70/30	0,741	0,705	0,738	0,781183	0,7293	
	k-fold10	0,642	0,649	0,707	0,714337	0,7432	0,683
HOG_128_32x32_PCA-75.csv (32 atributos)	70/30	0,664	0,649	0,740	0,699422	0,7060	
	k-fold10	0,660	0,634	0,696	0,728342	0,7201	0,668
HOG_128_32x32.csv (324 atributos)	70/30	0,661	0,634	0,648	0,680528	0,7011	
	k-fold10	0,610	0,617	0,704	0,697658	0,7144	0,638
HOG_128_20x20.csv (900 atributos)	70/30	0,528	0,612	0,672	0,660707	0,6992	
	k-fold10	0,568	0,612	0,686	0,672637	0,6917	0,625
HOG_256_32x32.csv (1764 atributos)	70/30	0,536	0,612	0,603	0,707488	0,6498	
	k-fold10	0,609	0,642	0,690	0,729038	0,7345	0,667
HOG_128_32x32_PCA-90.csv (64 atributos)	70/30	0,612	0,649	0,691	0,717459	0,7236	
	k-fold10	0,573	0,617	0,687	0,740586	0,7110	0,636
HOG_128_20x20_PCA-75.csv (71 atributos)	70/30	0,452	0,617	0,678	0,724615	0,7290	
MÉDIA =>		0,662	0,688	0,717	0,746	0,748	
DESVIO PADRÃO =>		0,087	0,066	0,052	0,048	0,047	

Figura 25: Tabela de F1-Score dos melhores modelos de cada classe.

O Teste de Friedman aplicado a estes 5 modelos resultou em um p-valor = 0.00000000016, rejeitando a hipótese nula e confirmando que há uma diferença estatística significativa entre eles. O teste pós-hoc de Nemenyi foi então aplicado:

```
Teste Nemenyi (pós-hoc) - Melhores Modelos:
                          kNN2      DTConf10   GaugasianNB  RedeNeuralConf5  \
kNN2             1.0000000000  0.9240765009  0.0065642119     0.0000011866
DTConf10         0.9240765009  1.0000000000  0.0787193525     0.0000754953
GaugasianNB      0.0065642119  0.0787193525  1.0000000000     0.3081170741
RedeNeuralConf5  0.0000011866  0.0000754953  0.3081170741     1.0000000000
Stacking5        0.0000000169  0.0000019458  0.0621207319     0.9495099579

                     Stacking5
kNN2            0.0000000169
DTConf10        0.0000019458
GaugasianNB     0.0621207319
RedeNeuralConf5 0.9495099579
Stacking5       1.0000000000
```

Figura 26: Resultado do Teste de Nemenyi para os 5 melhores modelos.

A análise da matriz de Nemenyi revelou um grupo com resultados superiores incluindo a Rede Neural e o Comitê de Classificadores. Eles são significativamente superiores aos demais e estaticamente equivalentes entre si. Embora as médias de F1-Score sejam quase idênticas (0.746 vs 0.748), a complexidade computacional é drasticamente diferente. A Rede Neural em questão é de um modelo com até 500 iterações, enquanto o Stacking (5) requer o treinamento de múltiplos modelos base mais um meta-classificador, tornando-o significativamente mais custoso. Portanto, a Rede Neural (Conf. 5) é considerada o melhor modelo geral testado neste trabalho.

4. Conclusão

Este trabalho realizou uma análise comparativa de cinco classes de algoritmos de aprendizado supervisionado (k-NN, DT, NB, MLP e Comitês) para a classificação de imagens de cães e gatos, utilizando descritores HOG e LBP. A rigorosa análise estatística permitiu extrair duas conclusões principais.

A primeira é que a escolha do descritor de características teve um impacto estatisticamente significativo no desempenho. Conforme evidenciado ao longo dos experimentos (Figura 23) , o descritor LBP (Local Binary Patterns) demonstrou ser consistentemente superior ao HOG para este *dataset*, com as seis bases de melhor desempenho sendo geradas a partir desta técnica.

A segunda e principal conclusão foi a determinação do modelo de classificação mais eficaz. A análise comparativa final (Seção 3.6) revelou que a Rede Neural (Conf. 5) e o comitê Stacking(5) formavam um grupo de desempenho superior, sendo estatisticamente equivalentes entre si (p=0.949) e significativamente melhores que as abordagens k-NN, Árvore de Decisão e Naive Bayes. Priorizando o menor custo computacional entre modelos estatisticamente idênticos, a Rede Neural (Conf. 5) foi selecionada como a melhor solução geral. O modelo vencedor—com uma camada oculta de 150 neurônios e máximo de 500 iterações —atingiu um F1-Score médio de 0.746.

O trabalho, portanto, conclui que para este problema, a combinação do descritor LBP com um modelo MLP relativamente simples oferece o melhor equilíbrio entre poder preditivo e eficiência computacional. Entre os desafios encontrados, destaca-se o alto custo de processamento para a geração dos 12 *datasets* e para a execução dos múltiplos testes de validação cruzada.

5. Referências

PEDREGOSA, F. et al. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, p. 2825-2830.

VISUAL GEOMETRY GROUP. (2012). Oxford-IIIT Pet Dataset. Disponível em: https://www.robots.ox.ac.uk/~vgg/data/pets/. Acesso em: 07 nov. 2025.).

SILVA, José Antônio da. (2017). *Detecção de Imagens Manipuladas utilizando Descritores Locais*. Trabalho de Conclusão de Curso (Bacharelado em Engenharia da Computação), Centro de Informática, Universidade Federal de Pernambuco, Recife.

SCIKIT-LEARN DEVELOPERS. (2025). 1.9. Naive Bayes. *Scikit-learn 1.5.2 documentation*. Disponível em: https://scikit-learn.org/stable/modules/naive_bayes.html#naive-bayes. Acesso em: 07 nov. 2025.