2º Trabalho Computacional

T10097 - Introdução ao Reconhecimento de Padrões

Aluno: Hubert Luz de Miranda
Matrícula: 552798

Fortaleza, 5 de agosto de 2025

Professor Responsável: Prof. Guilherme de Alencar Barreto
Instituição: Universidade Federal do Ceará (UFC)

Objetivo do trabalho

Desenvolver as habilidades de síntese de classificadores de padrões estudados na disciplina em problemas de processamento e reconhecimento de pessoas a partir de imagens da face.

1 Atividade 1

Nesta atividade, foi realizada uma adaptação do arquivo em Octave/Matlab presente no diretório "Codigos_Octave": **face_preprocessing_column.m** sem aplicar o PCA para a linguagem de programação Python que pode ser encontrada nas duas primeiras células, após a importação das bibliotecas, presentes no diretório "Resolucao"no arquivo **TC02.ipynb**, onde são realizados o carregamento e o pré-processamento das imagens, que inclui o redimensionamento para 20x20 pixels e a vetorização por coluna, gerando o arquivo de dados **recfaces.dat**.

2 Atividade 2: Classificação sem PCA

Nesta atividade, o desenvolvimento de funções para cada classificador foi necessário e, para isso, o material **classificacao-turmaUFC-2025.pdf** foi utilizado como base para o desenvolvimento das funções do classificador quadrático e suas variantes.

Lógica de cada função de classificador:

- **Quadrática:** A função discriminante a ser minimizada é dada pela Eq. (167):

$$g_i^*(x) = Q_i(x) + \ln(|C_i|) - 2\ln(f(\omega_i))$$

  onde $Q_i(x)$ é a distância de Mahalanobis da Eq. (166), e m_i e C_i são o vetor médio e a matriz de covariância da classe, respectivamente, calculados pela Eq. (165).

- **Variante 1:** A função discriminante é dada pela Eq. (177):

$$g_i^*(x) = Q_i^{(\lambda)}(x) + \ln(|C_i(\lambda)|) - 2\ln(f(\omega_i))$$

onde a distância de Mahalanobis $Q_i^{(\lambda)}(x)$ (Eq. 176) utiliza a matriz de covariância regularizada $C_i(\lambda) = C_i + \lambda I_p$ (Eq. 175).

- **Variante 2:** Utiliza a função discriminante da Eq. (201):

$$g_i^*(x) = Q_{pool_i}(x) - 2\ln(f(\omega_i))$$

A distância de Mahalanobis aqui utiliza a inversa da matriz de covariância agregada (C_{pool}), que é a média ponderada das covariâncias individuais, conforme a Eq. (197).

- **Variante 3:** A função discriminante é dada pela Eq. (204):

$$g_i^*(x) = Q_i^{(\lambda)}(x) + \ln(|C_i(\lambda)|) - 2\ln(f(\omega_i))$$

A distância $Q_i^{(\lambda)}(x)$ (Eq. 203) é calculada com a matriz regularizada de Friedman $C_i(\lambda)$ da Eq. (202).

- **Variante 4:** A função discriminante é dada pela Eq. (209):

$$g_i^*(x) = Q_{diag_i}(x) + \ln(|C_{i,diag}|) - 2\ln(f(\omega_i))$$

A distância $Q_{diag_i}(x)$ (Eq. 208) é calculada com a matriz de covariância diagonal $C_{i,diag}$ da Eq. (207).

- **OBS:** Foi implementado um tratamento de exceção robusto: nos casos em que todas as matrizes de covariância de uma rodada se mostraram singulares, a taxa de acerto para aquela rodada foi explicitamente definida como zero. Essa medida foi tomada a fim de evitar um comportamento de fallback da lógica de predição, que resultaria em uma acurácia artificial e não representativa do real desempenho do modelo.

Já para os classificadores MaxCorr, DMC e 1-NN, foi realizada uma adaptação dos códigos em Octave/Matlab: **Maxcorr.m**, **myDMC.m** e **myKNN.m** para a linguagem Python assim como foi realizado na atividade 1. A regra de decisão para o 1-NN busca o índice do vetor de treino mais próximo, conforme a Eq. (9):

$$i^* = \arg\min_{i=1,\ldots,N} \text{dist}(x_{new}, x_i)$$

E a regra para o Máxima Correlação busca a classe com maior produto escalar entre os vetores normalizados, conforme a Eq. (64):

$$\tilde{m}_{i^*}^T \tilde{x}_{new} > \tilde{m}_i^T \tilde{x}_{new}, \quad \forall i \neq i^*$$

Além disso, é importante destacar que os métodos de normalização propostos, sendo eles o z-score e por mudança de escala [0,+1] ou [-1,+1] comparados juntamente com os classificadores sem normalização, foram utilizados durante todo o restante atividade para encontrar a maior taxa de acerto para os classificadores MaxCorr, DMC e 1-NN.

Posteriormente, os classificadores foram treinados e avaliados utilizando os dados das imagens redimensionadas para 20x20 pixels (400 atributos), conforme nossas escolhas na primeira atividade, sem qualquer transformação via PCA. Para os classificadores 1-NN, DMC e MaxCorr, foi realizada uma busca pela melhor técnica de normalização (nenhuma, Z-Score, ou Min-Max), e o melhor resultado foi incluído na tabela. Toda a lógica do código também está presente no diretório "Resolucao"no arquivo **TC02.ipynb**

Tabela 1: Tabela de resultados sem a aplicação de PCA.

Classificador	Média	Mínimo	Máximo	Mediana	Desvio Padrão	Tempo de execução
Quadrático	0.0000	0.0000	0.0000	0.0000	0.0000	23.1205
Variante 1	79.5758	66.6667	93.9394	78.7879	6.7012	49.5954
Variante 2	3.5758	0.0000	15.1515	3.0303	4.0466	6.0291
Variante 3	6.5455	6.0606	9.0909	6.0606	1.1222	63.3541
Variante 4	4.3030	0.0000	9.0909	6.0606	3.0666	18.9394
MaxCorr	77.7576	66.6667	96.9697	75.7576	6.7235	0.0854
DMC	76.9091	63.6364	96.9697	75.7576	6.8963	0.1557
1-NN	77.8182	63.6364	90.9091	78.7879	6.3162	0.2022

2.1 Análise das Questões 1, 2 e 3

Questão 1: O que se pode concluir sobre os desempenhos dos classificadores avaliados?
Resposta: Dentre os classificadores quadráticos, apenas a variante 1 teve um bom desempenho, enquanto o quadrático gaussiano não teve efetividade nenhuma e os outros tiveram uma baixa acurácia. Já os outros classificadores obtiveram um bom desempenho em comparação ao restante, se aproximando da variante 1.

Questão 2: Qual deles teve o melhor desempenho em relação à taxa de acerto? E em relação ao tempo?
Resposta: O melhor desempenho em relação à taxa de acerto foi da variante 1 do classificador quadrático, com uma média de 79.5758% de acurácia. Já o melhor desempenho em relação ao tempo de execução foi do classificador de máxima correlação, que foi executado em $0.0854s$.

Questão 3: Houve problemas de inversão das matrizes de covariância? Se sim, para quais classificadores? Este problema foi contornado por alguma das variantes avaliadas?
Resposta: Sim, houve problemas de inversão de matrizes de covariância. Os problemas atingiram notadamente os classificadores quadrático gaussiano e a variante 4, já para as variantes 2 e 3, o impacto foi menor, com poucas matrizes singulares, a quantidade de matrizes singulares para cada classificador quadrático está presente no arquivo **TC02.ipynb**. O problema foi contornado pela variante de número 1, devido à sua técnica de regularização, que consiste na regularização de Tikhonov, onde uma matriz identidade multiplicada por um pequeno escalar λ é somada à matriz de covariância singular como foi explicado anteriormente, garantindo que a matriz resultante seja sempre invertível. Também é válido destacar que as técnicas de regularização das variantes 2 e 3 mitigaram este problema, apenas não foram eficientes como a variante 1.

3 Atividade 3

Para resolver esta atividade, foi necessário adicionar uma célula ao arquivo **TC02.ipynb** com a aplicação do PCA para um $q = 400$ conforme a dimensão de redução que foi escolhida anteriormente, entretanto, o nosso q verdadeiro vai ser limitado, pois o número de amostras ($N = 165$) é menor que o número de atributos ($p = 400$), o que limita o posto (rank) da matriz de covariância. Logo, só podemos obter no máximo $N - 1 = 164$

autovalores com valores significativos (não nulos), portanto limitamos nosso q efetivo a 164 componentes, que corresponde a 100% da variância contida nos dados.

Segue o gráfico da variância explicada:

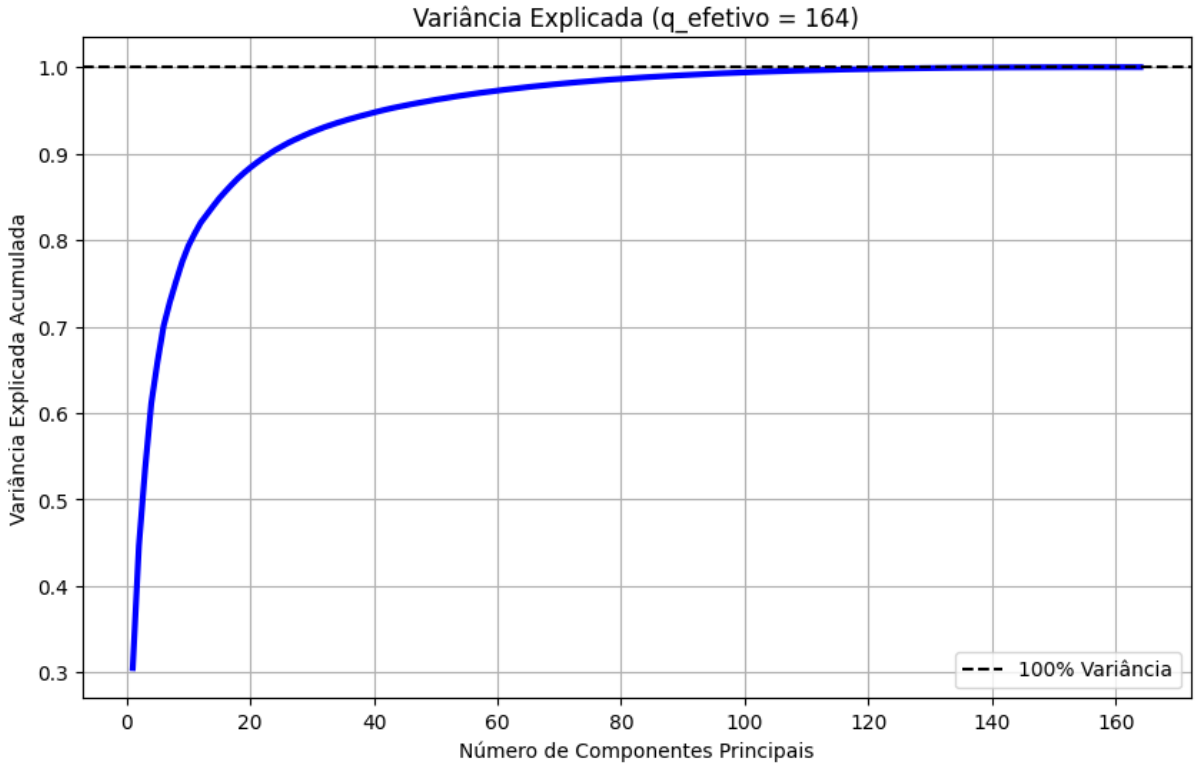

Figura 1: Gráfico da variância explicada para q=400

4 Atividade 4: Classificação com PCA (sem redução)

Nesta atividade, seguimos o mesmo passo a passo da atividade 2, mas com a aplicação do PCA.

Tabela 2: Tabela de resultados com PCA sem redução de dimensionalidade (q=164).

Classificador	**Média**	**Mínimo**	**Máximo**	**Mediana**	**Desvio Padrão**	**Tempo de execução**
Quadrático	0.0000	0.0000	0.0000	0.0000	0.0000	29.2530
Variante 1	80.1212	69.6970	93.9394	80.3030	6.2781	80.4130
Variante 2	0.0000	0.0000	0.0000	0.0000	0.0000	8.7477
Variante 3	0.0000	0.0000	0.0000	0.0000	0.0000	48.5713
Variante 4	0.0000	0.0000	0.0000	0.0000	0.0000	31.4270
MaxCorr	78.7879	66.6667	96.9697	77.2727	6.8993	0.0764
DMC	77.9394	66.6667	96.9697	77.2727	6.9275	0.1149
1-NN	78.5455	63.6364	90.9091	78.7879	6.5321	0.1789

4.1 Análise da Questão 4

Questão 4: (i) O que se pode concluir sobre os desempenhos? Houve alguma mudança? (ii) O classificador quadrático e a Variante 4 foram teoricamente equivalentes?

Resposta: Os desempenhos, no geral, pioraram, pois as variantes 2,3 e 4 passaram a não ter nenhuma efetividade, tal qual o classificador quadrático gaussiano, enquanto o desempenho dos outros classificadores se manteve similar, o que pode ser explicado por conta do fato de que a aplicação do PCA, sem uma redução de dimensionalidade significativa (mantendo q=164) não resolve o problema fundamental de $p > N_i$. A matriz de covariância para cada classe ainda é estimada com muito poucos exemplos ($N_i \approx 9$) para o grande número de atributos (p=164), fazendo com que elas permaneçam singulares e os classificadores que dependem delas continuem ineficazes. Já no que diz respeito à equivalência entre os classificadores quadrático gaussiano e sua variante 4, podemos inferir que eles não se mostraram equivalentes na prática, e a premissa teórica do enunciado não se aplica diretamente. A razão para isso é que o PCA garante a descorrelação das features ao diagonalizar a matriz de covariância **global** (calculada sobre todo o conjunto de dados). No entanto, os classificadores Quadrático e Variante 4 operam com base nas matrizes de covariância **de cada classe** (C_i). A diagonalização global não garante que as matrizes C_i individuais também serão diagonais no novo espaço de features. Como o classificador Quadrático utiliza a C_i completa (com termos fora da diagonal) e a Variante 4 força essa matriz a ser diagonal, seus modelos matemáticos são diferentes. Na prática, para este experimento com q=164, ambos falharam completamente devido ao problema de $p > N_i$, resultando em um desempenho igualmente ineficaz.

5 Atividade 5

Nesta atividade, foi preciso calcular, a partir do gráfico gerado, a quantidade de componentes(q) que preservasse pelo menos 98% da variância. O valor foi encontrado computacionalmente e o gráfico com sua identificação foi o seguinte:

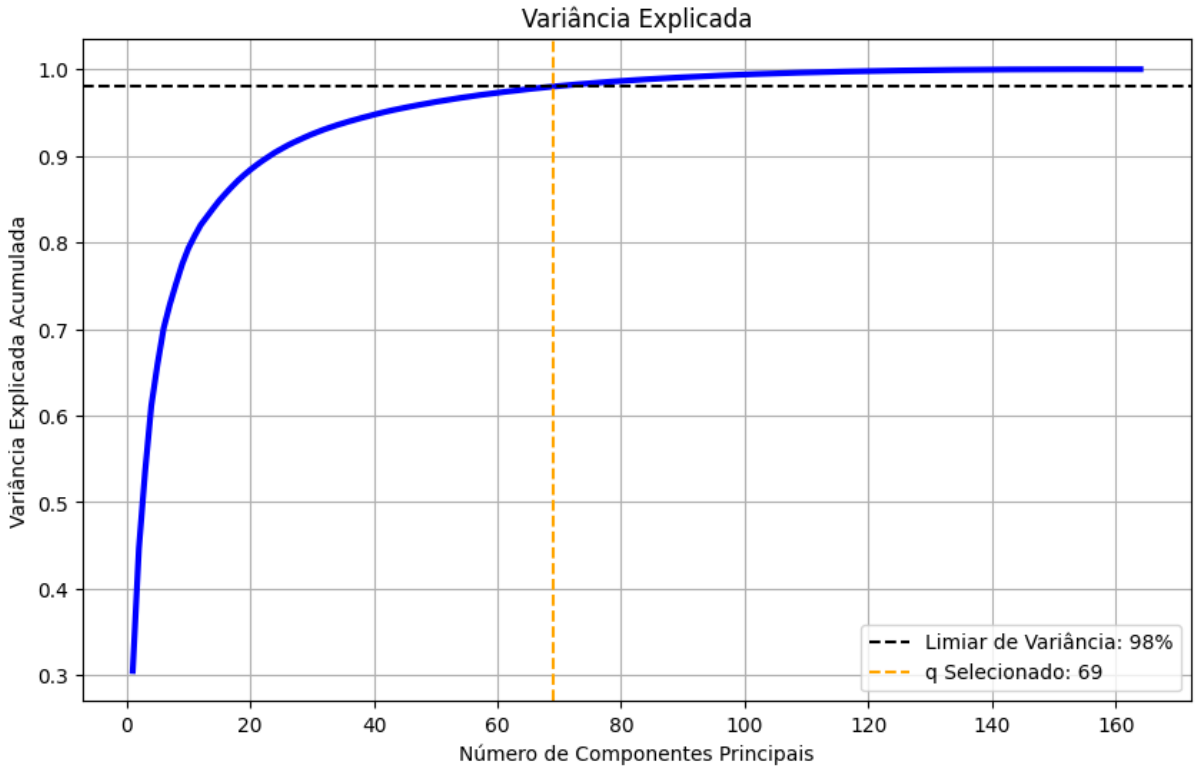

Figura 2: Gráfico da variância explicada para q=69

5.1 Análise da Questão 5

Questão 5: Qual foi a dimensão de redução q escolhida, de modo a preservar 98% da informação do conjunto de dados original?
Resposta: O valor de q escolhido para preservar 98% da variância, após a correção da vetorização para o padrão de colunas, foi de $q = 69$.

6 Atividade 6: Classificação com PCA (com redução)

Aqui, o PCA foi utilizado para reduzir a dimensionalidade do problema, a partir das informações encontradas na atividade 5.

Tabela 3: Tabela de resultados com PCA e redução de dimensionalidade (q=69).

Classificador	**Média**	**Mínimo**	**Máximo**	**Mediana**	**Desvio Padrão**	**Tempo de execução**
Quadrático	6.4242	6.0606	9.0909	6.0606	0.9947	0.2537
Variante 1	79.5152	69.6970	93.9394	78.7879	6.3788	0.2609
Variante 2	95.8788	84.8485	100.0000	96.9697	3.6610	0.1997
Variante 3	95.8182	84.8485	100.0000	96.9697	3.8179	0.2523
Variante 4	77.5758	60.6061	90.9091	78.7879	7.0604	0.2185
MaxCorr	94.6667	84.8485	100.0000	93.9394	3.5466	0.0957
DMC	92.6667	81.8182	100.0000	93.9394	5.0887	0.1197
1-NN	88.5455	75.7576	100.0000	87.8788	5.2768	0.0672

6.1 Análise da Questão 6

Questão 6: O que se pode concluir sobre os desempenhos? Quais classificadores melhoraram ou pioraram?
Resposta: Todos os classificadores apresentaram melhoras, exceto a variante 1 do classificador quadrático que apresentou uma leve piora. Pode-se destacar a grande melhora das variantes 2 e 3 do classificador quadrático, o que pode ser explicado pela drástica redução de dimensionalidade de 164 para 69 atributos. Essa redução, além de remover ruído contido nos componentes de menor variância, mitigou significativamente o problema de $p > N_i$. Com menos atributos, a estimativa da matriz de covariância agregada (C_{pool}) tornou-se muito mais estável e robusta, pois é calculada com um número de amostras ($N \approx 132$) bem maior que o número de *features* ($p = 69$). Como as variantes 2 e 3 dependem dessa matriz agregada, seu desempenho melhorou drasticamente, tornando-se os classificadores mais precisos nesta etapa.

7 Atividade 7: Classificação com Box-Cox e PCA

Nesta etapa final, a transformação de Box-Cox foi aplicada aos dados originais antes do PCA e da classificação, implicando em uma alteração do q escolhido para manter 98% da variância.

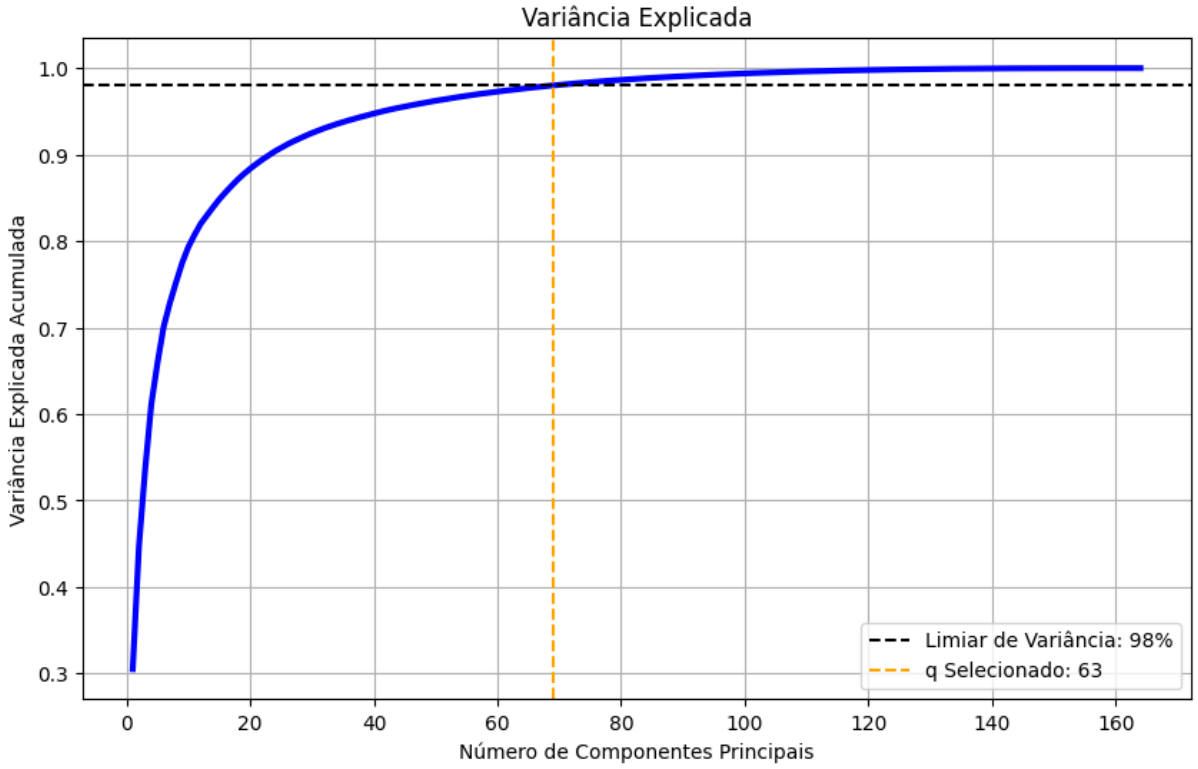

Figura 3: Gráfico da variância explicada para q=63

Tabela 4: Tabela de resultados com Box-Cox + PCA.

Classificador	**Média**	**Mínimo**	**Máximo**	**Mediana**	**Desvio Padrão**	**Tempo de execução**
Quadrático	6.4848	6.0606	9.0909	6.0606	1.0622	0.2998
Variante 1	88.2424	78.7879	96.9697	87.8788	4.7674	0.3280
Variante 2	97.0909	90.9091	100.0000	96.9697	2.8689	0.2147
Variante 3	97.2727	90.9091	100.0000	96.9697	2.7550	0.3250
Variante 4	73.9394	60.6061	90.9091	75.7576	6.3328	0.3073
MaxCorr	93.8182	84.8485	100.0000	93.9394	4.0127	0.1070
DMC	93.9394	81.8182	100.0000	93.9394	3.5698	0.1354
1-NN	89.8788	72.7273	100.0000	90.9091	4.3121	0.0681

7.1 Análise da Questão 7

Questão 7: Houve alguma mudança (melhora ou piora) nos desempenhos dos classificadores em relação à Atividade 6?
Resposta: Em geral, houve melhoras dos classificadores, com um grande destaque para a variante 1 do classificador quadrático, que aumentou sua média de taxa de acerto em mais de 8%. Ademais, o único classificador que sofreu uma piora significativa foi a variante 4 do classificador quadrático. Este comportamento pode ser explicado por o fato de a transformação de Box-Cox ter tornado a distribuição de probabilidade de cada atributo mais próxima de uma distribuição Gaussiana. Os classificadores quadráticos, em especial a Variante 1, baseiam-se fortemente na premissa de que os dados de cada classe seguem uma distribuição normal multivariada. Ao pré-processar os dados com Box-Cox, essa condição

foi melhor satisfeita, permitindo que a Variante 1 modelasse a estrutura de covariância de cada classe com mais precisão, resultando no ganho expressivo de desempenho.

Por outro lado, a piora da Variante 4 (Naive Bayes) também é explicada por essa transformação. O Naive Bayes assume que os atributos são condicionalmente independentes. A transformação de Box-Cox, ao ajustar a distribuição de cada atributo individualmente, pode ter, como efeito colateral, fortalecido as correlações existentes entre os atributos. Com a violação da premissa de independência se tornando mais acentuada, sua performance foi degradada.

8 Atividade 8

Para resolver esta atividade, os classificadores DMC e Mahalanobis foram adaptados para uma aplicação de controle de acesso. A lógica de classificação foi modificada para uma de verificação, baseada em limiares de distância. Primeiramente, o dataset foi expandido para incluir 11 imagens de um indivíduo "intruso"(classe 16).

O processo de treinamento, realizado apenas com os 15 usuários válidos, consistiu em calcular o protótipo de cada classe (centroide) e um limiar de decisão. Este limiar foi determinado de forma não-paramétrica, utilizando o percentil 95% ($\alpha = 0.05$) das distâncias das amostras de treino em relação ao seu próprio centroide.

Na fase de teste, para uma nova amostra, o sistema primeiro identifica a classe válida mais próxima e então compara a distância a esta classe com o seu limiar específico. Se a distância for menor que o limiar, o acesso é permitido; caso contrário, a amostra é classificada como "intruso"e o acesso é negado. Foram implementados dois modelos:

- **Modelo 1:** Utiliza a distância de Mahalanobis (com regularização de Tikhonov para garantir a invertibilidade das matrizes de covariância) sobre os dados brutos vetorizados.
- **Modelo 2:** Utiliza a distância Euclidiana sobre os dados após a aplicação de PCA (com 98% da variância) e normalização Z-Score.

8.1 Análise da Questão 8

Questão 8: Calcule os seguintes índices de desempenho para os classificadores implementados: acurácia, taxa de falsos negativos (proporção de pessoas às quais acesso foi permitido incorretamente) e taxa de falsos positivos (pessoas às quais acesso não foi permitido incorretamente), sensibilidade e precisão. Os valores devem ser médios com inclusão de medida de dispersão para 50 rodadas.

Tabela 5: Métricas de desempenho para o Modelo 1 (Mahalanobis).

Métrica	**Média**	**Mínimo**	**Máximo**	**Mediana**	**Desvio Padrão**
Acurácia (%)	14.89	5.56	22.22	13.89	3.92
Taxa de Falsos Positivos (%)	90.64	84.85	100.00	91.18	3.86
Taxa de Falsos Negativos (%)	0.00	0.00	0.00	0.00	0.00
Sensibilidade (%)	100.00	100.00	100.00	100.00	0.00
Precisão (%)	6.72	5.56	9.68	6.25	1.31

Tabela 6: Métricas de desempenho para o Modelo 2 (Euclidiano + PCA + Z-Score).

Métrica	**Média**	**Mínimo**	**Máximo**	**Mediana**	**Desvio Padrão**
Acurácia (%)	69.82	48.57	83.33	71.01	7.42
Taxa de Falsos Positivos (%)	27.15	11.76	50.00	26.47	7.64
Taxa de Falsos Negativos (%)	76.33	0.00	100.00	100.00	28.59
Sensibilidade (%)	23.67	0.00	100.00	0.00	28.59
Precisão (%)	5.64	0.00	25.00	0.00	6.98

Resposta: As tabelas 5 e 6 apresentam as métricas de desempenho para os dois modelos de controle de acesso. A análise dos resultados revela um claro trade-off entre segurança e conveniência.

O **Modelo 1 (Mahalanobis)** demonstrou ser um sistema extremamente seguro, mas impraticável. Sua sensibilidade de 100% e taxa de falsos negativos de 0% indicam que ele foi perfeito em barrar todos os intrusos. Contudo, isso veio a um custo altíssimo: uma taxa de falsos positivos de 90.64%, significando que ele negou o acesso a quase todos os usuários válidos. Isso o torna um sistema muito seguro, porém inútil na prática, o que é refletido em sua baixíssima acurácia (14.89%).

O **Modelo 2 (Euclidiano + PCA)**, por outro lado, apresentou um comportamento mais equilibrado, embora tendendo a ser excessivamente permissivo. A taxa de falsos positivos foi drasticamente menor (27.15%), tornando o sistema muito mais conveniente para os usuários válidos. No entanto, essa conveniência comprometeu a segurança: a sensibilidade de apenas 23.67% (e uma taxa de falsos negativos de 76.33%) mostra que o modelo falhou em detectar a grande maioria dos intrusos. Sua acurácia mais alta (69.82%) é majoritariamente devida ao acerto na aceitação dos usuários válidos, que compõem a maior parte do dataset.

Conclui-se que o Modelo 1 é seguro demais, e o Modelo 2 é inseguro demais.