Relatório: Métodos Estatísticos e KNN

João Victor de Souza Albuquerque

[1]Instituto Federal de Educação, Ciência e Tecnologia do Ceará - Campus Maracanaú (IFCE)
Av.Parque Central, 1315 - Distrito Industrial I, Maracanaú - CE, 61939-140

Resumo. *Este relatório documenta o experimento da lista 4 de reconhecimento de padrões.*

1. Base Teórica

1.1. Análise de Discriminante Gaussiano e Naive Bayes Gaussiano

A até o momento tinha sido trabalhado com modelos discriminantes que estimam parâmetros para as fronteiras de decisão entre classes a partir dos dados de treino, agora será desenvolvido modelos generativos que modelam a distribuição das entradas associadas a cada classe. Foram montados dois classificadores Bayesianos diferentes, o primeiro foi o Análise de Discriminante Gaussiano que é modelo seguido a seguinte função classificação:

$$\hat{y}_k = \arg\max_k \left[\log p(C_k) - \frac{1}{2}\log|\Sigma_k| - \frac{1}{2}(x_* - \mu_k)^T \Sigma_k^{-1}(x_* - \mu_k) \right] \tag{1}$$

O segundo modelo, Naive Bayes Gaussiano, foi modelo seguindo a seguinte função de matemática:

$$\hat{y}_k = \arg\max_k \left[\log p(C_k) - \frac{1}{2}\sum_{d=1}^{D} \log 2\pi\sigma_{dk}^2 - \frac{1}{2}\sum_{d=1}^{D} \frac{(x_{*,d} - \mu_{dk})^2}{\sigma_{dk}^2} \right] \tag{2}$$

1.2. K Nearest Neighbors

Os modelos feitos nesse trabalho e nos trabalhos anteriores eram modelos do tipo paramétrico, isso se significa que eles tem um conjunto de variáveis que concentram o que foi aprendido com os dados de treino. Agora será a abordado um modelo não-paramétrico, isso significa que esse tipo de modelo não tem um conjunto de variáveis que concentram o aprendizado, em vez disso, eles admitem um conjunto ilimitado de parâmetros,ou seja, sua quantidade depende do número de padrões de treinamento, o modelo KNN que será usado nesse trabalho é um representante desses modelos.

O processo de classificação do KNN se basei na distancia de um ponto desconhecido num espaço de n dimensões para os pontos conhecidos desse espaço, portanto o modelo calcula a distancia do ponto desconhecido para todos os pontos conhecidos e pega os n pontos que tiveram a menor distancia calculada, e com base na classe majoritária desses n pontos, o ponto desconhecido tem sua classificação. Se tem diferentes equações para se calcular a distancia dos pontos, foi escolhido para esse trabalho utilizar a equação da distância Euclidiana:

$$\|x_i - x_j\|_2 = \sqrt{\sum_{d=1}^{D}(x_{id} - x_{jd})^2} \tag{3}$$

1.3. Normalização

As vezes os dados que queremos que o modelo se adapte estão em escalas muito diferentes o que pode acarretar em dificuldade do modelo em se adaptar a eles, por isso é recomendado deixar os dados numa escala comum, esse processo se chama de normalização, que pode ser colocar os dados em uma escala de [0,1] ou [-1,1] ou mesmo forção uma média e variância. Esse método de preprocessamento nos da maior controle dos valores dos parâmetros e facilita o ajuste dos hiperparâmetros

No trabalho foi utilizado a normalização dos dados na escala de [0,1], essa regularização seguir os seguintes passos:

Primeiro se calcula o valor máximo do vetor Y e matriz X, coluna a coluna:

$$y_{max} = \max(y), \quad [x_{max}]_d = \max([X]_d), \forall d.$$

Depois disso, calcule o menor valor do vetor Y e da matiz X, coluna a coluna:

$$y_{min} = \min(y), \quad [x_{min}]_d = \min([X]_d), \forall d.$$

E por fim se aplica a formula a seguir para finalizar a regularização:

$$y \leftarrow \frac{y - y_{min}}{y_{max} - y_{min}}, \quad [X]_d \leftarrow \frac{[X]_d - [x_{min}]_d}{[x_{max}]_d - [x_{min}]_d}, \forall d.$$

2. Metodologia

2.1. Dados

Importei os dados contidos no breast.csv e os dados da Iris dataset da biblioteca do sklearn, ambos os datasets foram salvos em dataframes do pandas. O dataset da breast não passou por nenhuma modificação nos seus dados.

2.2. Criação do modelo

Para a realização dessa lista foi necessário fazer os 3 modelos do zero seguindo as instruções matemáticas contidas nos slides das aulas, todas as formulas foram adaptadas para código python utilizando a biblioteca do numpy como apoio para manipulação matemática e a biblioteca pandas como apoio para obter algumas estatísticas dos dados.

2.3. Normalização

Para Implenetar a classe de normalização, preciso desenvolver 3 funções, a função fit(), normalize e desnormalize. A função fit salva os minemos e máximos do X e Y internamente na classe. A função de normalize pega os valores de máximos e minemos salvos na classe e utiliza na função de normalização já mostrada e que foi traduzida para código, e no final devolve os X e Y normalizados. A função desnormalize pega os valores mínimo e máximo salvos internamente na classe e as utiliza na inversa da função de normalização.

2.4. Questão 1

Primeiro separei os dados em conjunto de treino e teste, utilizando a função train_test_split() do sklearn com test_size de 0.2 e random_state de 42. Após isso, separei o conjuto de treino em train_x e train_y, e o conjunto de teste em test_x e test_y.

Com os conjuntos já em mãos, eu defini a classe de normalização e chamei sua função fit() para ajustar a escala aos dado. Em seguida chamei a função normalize para o conjunto de treino e teste, e salvei as saídas em variáveis.

Com os dados já normalizado, fiz a instanciação do modelo class DiscirminanteGaussianoClassifier, o treinei e fiz a predição com os dados de teste, após isso, fiz o calculo da quantidade de verdadeiros positivo, falsos negativo, falsos positivo, e verdadeiro negativo. Em seguida fiz os cálculos das estatísticas do modelo, acurácia, revocação, precisão e f1-score, por ultimo plotei a matriz de confusão, em seguida fiz um teste das estatísticas do modelo utilizando 10 folds, com o auxilio da função kfold() do sklearn, em cada rodada do fold eu fazia os mesmo passo a passo do primeiro teste do modelo com exceção em plotar a matriz de confusão e do split. No fim de todas as rodadas fiz uma media e desvio padrão de cada uma das estatísticas.

Finalizado o experimento como o primeiro modelo, eu repeti o mesmo passo a passo acima para o modelo NaiveBayesGaussianoClassifier, e para o modelo KNNClassifier, que o utilizei k = 3.

3. Resultado

3.1. Questão 1

O Modelo DiscirminanteGaussianoClassifier obteve as estatísticas no teste unitário presente na figura 1 e sua matriz de confusão da figura 2, já no teste dos 10 folds o modelo teve o resultado presente na figura 3.

Já o modelo NaiveBayesGaussianoClassifier obteve as seguintes estatísticas no teste unitário, figura 4, e obteve a segui-te matriz de confusão, figura 5. No experimento dos 10 folds o modelo teve os resultados presentes na figura 6.

Com o modelo KNNClassifier obteve os resultados da figura 7 no teste unitário e a matriz de confusão da figura 8, e nos 10 folds foi obtido os resultados da figura 9.

Como pode ser analisado pelos resultados dos modelos, todos eles foram ótimos para esse problema de classificação, e levando em consideração os resultados estatísticos qualquer um desse modelos poderiam ser escolhido para resolver esse problema, porem levando em consideração que o KNNClassifier facilmente escala sua complexidade tenderia a escolher para o problema de classificação o modelo NaiveBayesGaussianoClassifier ou DiscirminanteGaussianoClassifier por performaram de forma mais eficiente e não escalar com a facilidade a complexidade.

```
//////////////////////////////////////////////////
acuracia do modelo:  0.9473684210526315
revocação do modelo:  0.9436619718309859
precisao do modelo:  0.9710144927536232
f1-score do modelo:  0.9571428571428571
```

Figure 1. DiscirminanteGaussianoClassifier resultados

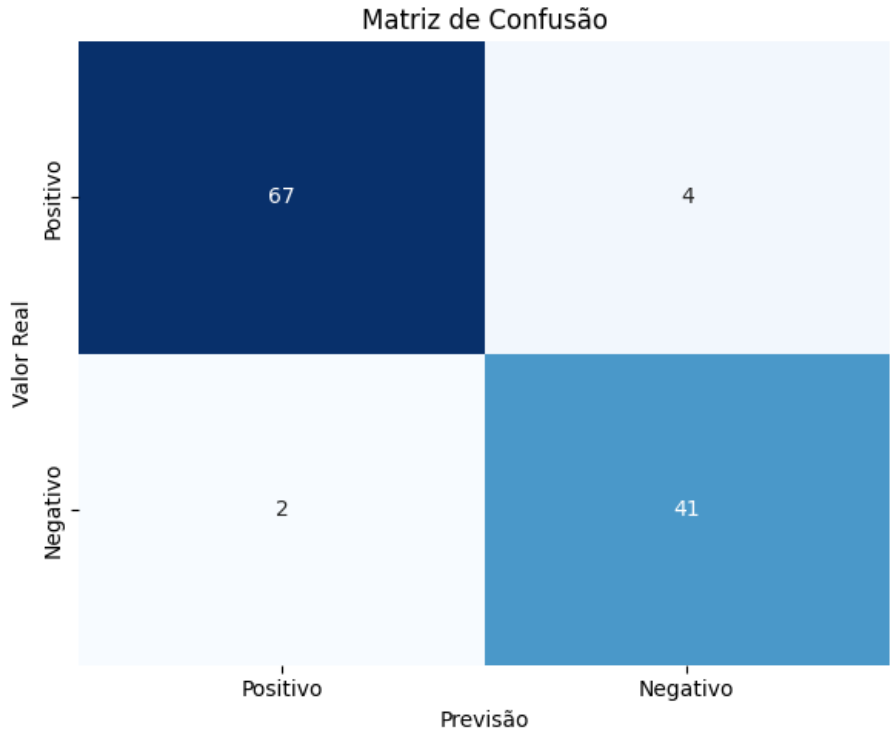

Figure 2. matriz de confusão DiscirminanteGaussianoClassifier

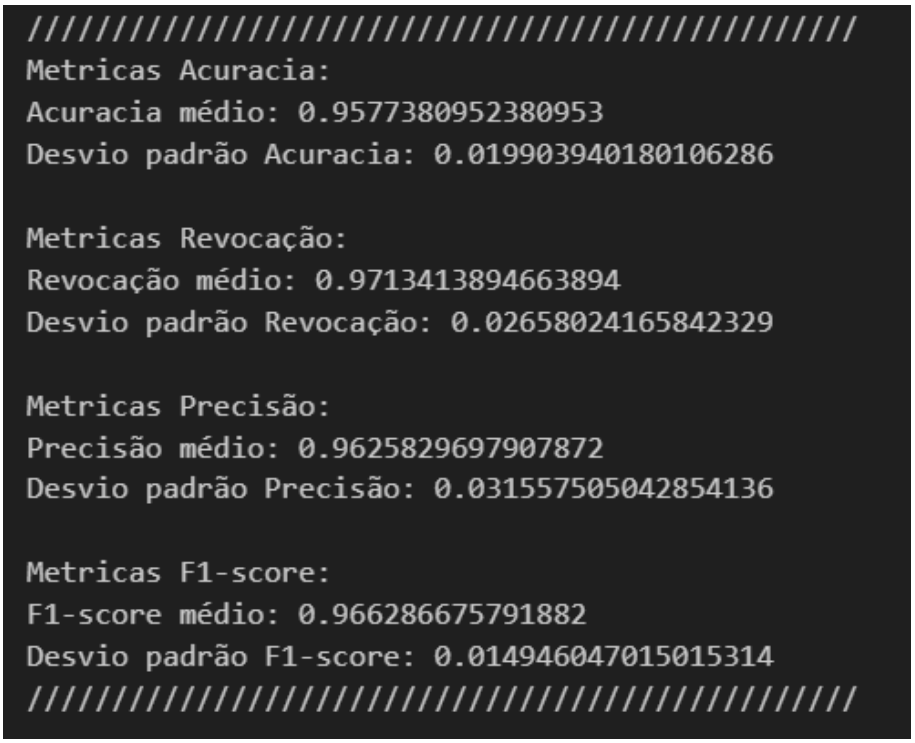

Figure 3. 10 k-folds DiscirminanteGaussianoClassifier

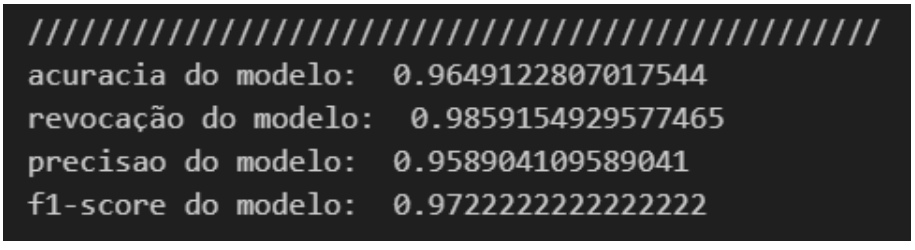

Figure 4. NaiveBayesGaussianoClassifier resultados

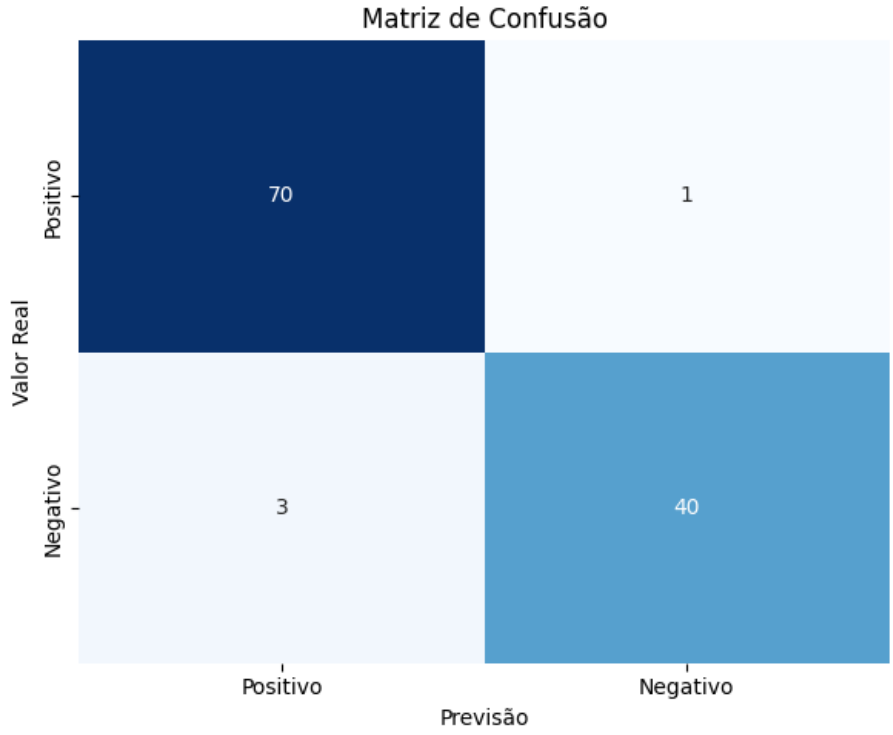

Figure 5. matriz de confusão NaiveBayesGaussianoClassifier

```
//////////////////////////////////////////////////
Metricas Acuracia:
Acuracia médio: 0.9313909774436089
Desvio padrão Acuracia: 0.033910212057259845

Metricas Revocação:
Revocação médio: 0.9515722453222454
Desvio padrão Revocação: 0.03959664059224107

Metricas Precisão:
Precisão médio: 0.9413150910494364
Desvio padrão Precisão: 0.025471197614335334

Metricas F1-score:
F1-score médio: 0.9458767856787155
Desvio padrão F1-score: 0.024640950800726227
//////////////////////////////////////////////////
```

Figure 6. 10 k-folds NaiveBayesGaussianoClassifier

```
////////////////////////////////////////////////
acuracia do modelo:  0.9649122807017544
revocação do modelo:  0.971830985915493
precisao do modelo:  0.971830985915493
f1-score do modelo:  0.971830985915493
```

Figure 7. KNNClassifier resultados

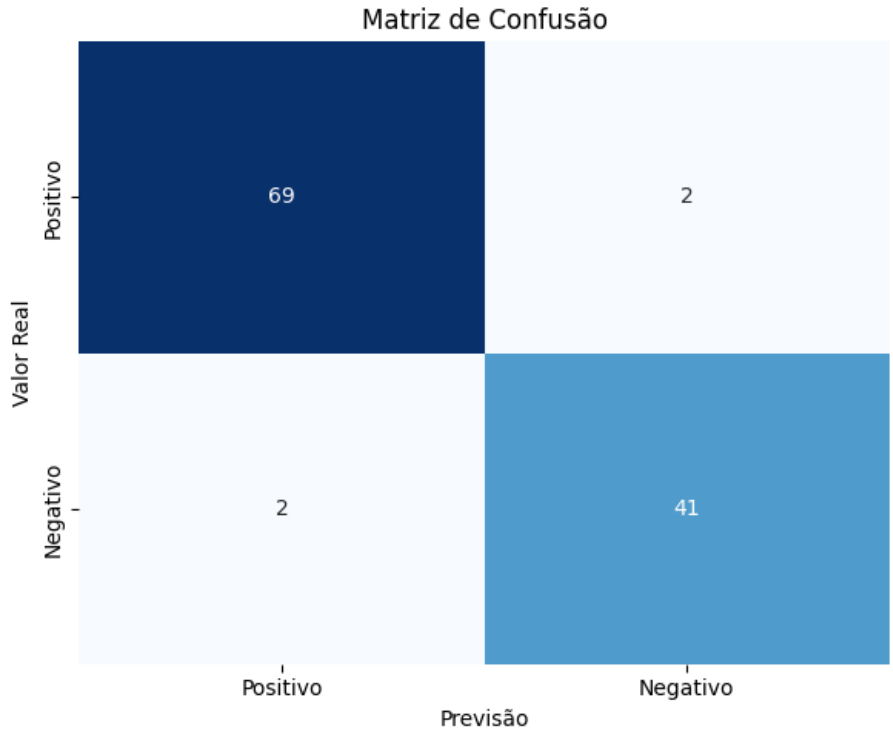

Figure 8. matriz de confusão KNNClassifier

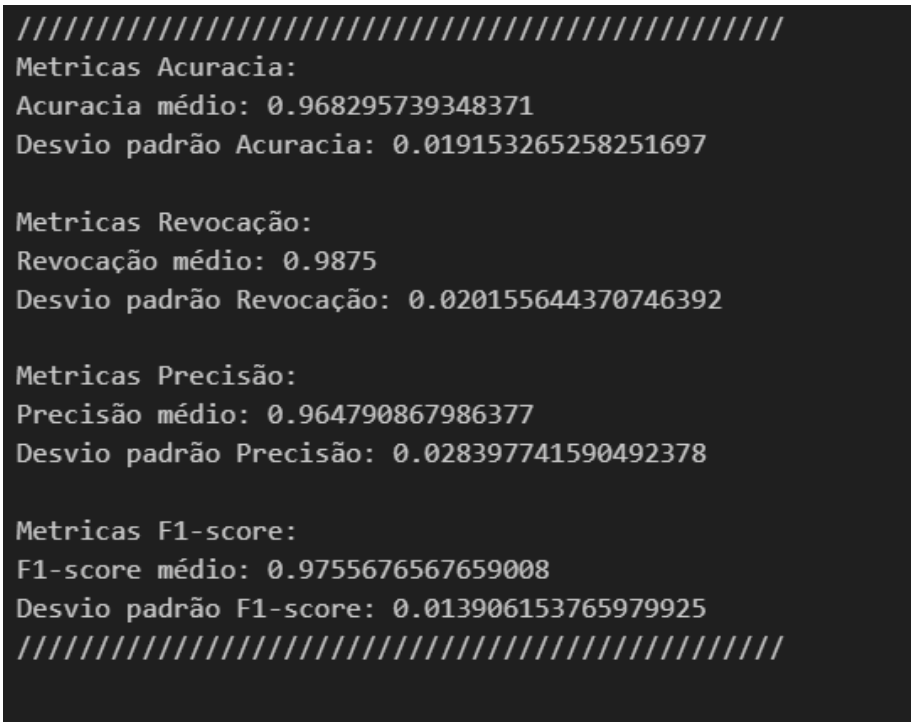

Figure 9. 10 k-folds KNNClassifier