Classificação de Digitos Manuscritos Trabalho feito para a disciplina FSI da Universidade de Brasília

1st Alberto Neto
Departamento de Ciência da Computação
Universidade de Brasília
Brasília, Brasil
albertotdneto@hotmail.com

***Abstract*—Foi utilizado o conjunto de dados MNIST e os algorítmos de aprendizagem KNN e LDA para realizar a tarefa de predição de números manuscritos. Além dos algorítmos em si, também foram analizados os features extraídos das imagens e como eles afetam os resultados, além de possíveis explicações para o melhor desempenho de um algorítmo de aprendizado sobre o outro.**

***Index Terms*—MNIST, KNN, LDA, Machine Learning**

I. Introdução

Com o avanço computacional das últimas décadas, muitos dos algorítmos computacionais que seriam impraticáveis hoje se tornam possíveis mesmo em computadores pessoais. Isso, juntamente com a grande disponibilização de dados, o Big Data, e de implementação de algorítmos, incluindo, mas não se limitando aos de aprendizado de máquina, É possível resolver uma grande sorte de problemas.

Inspirado por isto, serãp analizados dois algorítmos na resolução do problema de classificação de imagens de digitos manuscritos: o KNN (K nearest neighbors) e o LDA (linear discriminant analisys).

II. Problema

A definição de aprendizado de máquina de Tom M. Mitchell diz: "[...] um programa de computador aprende pela experiência E, com respeito a algum tipo de tarefa T e performance P, se sua performance P nas tarefas em T, na forma medida por P, melhoram com a experiência E" (referencia). Para melhor entendimento do problema, é importante decidir o que experiência, tarefa e performance significam dentro do contexto do problema(achar sinonimo de problema) a ser analizado.

A tarefa T é a classificação de números através de imagens, sendo que cada imagem contêm apenas um número manuscrito. As imagens utilizadas serão monocromáticas e de tamanho 28 por 28 pixels.

A experiência E se dará pelo conjunto de dados MNIST, que é uma coleção de 60000 imagens de treinamento e 10000 imagens de teste. Deve ser notado que essas imagens estão devidamente rotuladas.

A performance P será a porcentagem de acertos dos algorítmos utilizados com os dados de teste, estes não vistos pelo programa durante o treinamento.

III. Métodos

O método se dá por duas escolhas importantes: os algorítmos de aprendizagem a serem utilizados e às características (também chamadas de features) extraídas das imagens.

A. Algorítmos de Aprendizagem

Segue uma breve descrição dos algorítmos de classificação a serem utilizados:

- KNN: utiliza os pontos conhecidos próximos (a definição de distância pode variar, sendo a mais simples a distância euclideana) para categorizar um ponto de classe desconhecida. Dado um número K inteiro, a classe mais observada nos K pontos mais próximos do ponto que deseja-se prever é a saída do programa.
- LDA: assume que as classes possam ser separadas por uma função linear e divide os dados a partir da distância entre as médias amostrais. A função delimitadora então separa as predições de classes, sendo que quanto mais longe dos pontos da função, mais confiante é a predição. Importante notar que o LDA assume que as variáveis possuem distribuição normal.

Por ambos utilizarem dados rotulados para a aprendizagem, ou seja, os dados de treinamento possuem a saída (ou classe) desejada, são chamados de métodos de aprendizado supervisionados. Como as imagens do conjunto de dados MNIST possuem as saídas desejadas, a aprendizagem supervisionada é um método adequado e será utilizada.

Outra importante diferença é a suposição feita: LDA assume que as classes possam sem divididas por uma função linear e tenta encontrar os parâmetros dessa função, KNN não faz suposição alguma, portanto não há parâmetros a serem definidos. Logo, este é chamado de não paramétrico e aquele de paramétrico.

Essa diferença certamente terá um papel fundamental na performance final de cada um deles.

B. Features

Antes da extração das características em si é necessário um tratamento na imagem. Cada pixel é representado por um byte, sendo 0 a cor preta e 255 a cor branca. Porém, um pixel de valor 129 é branco ou preto? E um de valor 28? Devemos escolher um threshold para fazer essas definições.

Abaixo as características que foram extraídas:

- Centroide
- Área
- Pontos de extremo
- Ângulo da melhor elipse ajustada
- Centro do melhor círculo ajustado
- Raio do melhor círculo ajustado
- Eixos X e Y

IV. Resultados

Os resultados são divididos em 2: teste inicial e resultado final.

No teste inicial são feitas algumas considerações importantes, como a escolha do threshold e qual dos algorítmos de aprendizado se saem melhor naquele contexto. Por simplicidade, neste momento foram utilizados poucos features.

No resultado final serão analizados os resultados com todas as features escolhidas a fim de obter a melhor performance possível.

A. Teste inicial

Foram testados 3 valores de threshold, 127, que é metade da faixa dinâmica do byte, 63, um quarto, e 1. Foi notado que quanto maior o threshold, maior era a quantidade de contornos com área 0.

Para melhor entendimento, será testada a acurácia de predição com apenas a área do contorno como feature.

TABLE I
Features: Área

Threshold	**Área do contorno**	**Acurácia de predição**	
	Observações de Área 0	**KNN**	**LDA**
127	8	0.2465	0.2806
63	1	0.2533	0.2961
1	0	0.2656	0.3041

A partir da tabela I percebe-se que a acurácia dos algorítmos para a previsão aumenta conforme o threshold diminui. Porém, de 127 para 1, a quantidade de observações de área 0 vai de 1 para 0. Será que esta única observação causa este impacto de +-2%?

É intuitivo pensar que, quanto mais pontos forem decididos como 255 (branco), mais informação terá-se sobre o contorno que deseja-se classificar. Tome o centroide como exemplo. Ele é o "ponto de gravidade" do contorno, ou seja, numa imagem perfeitamente simétrica o ponto centroide estará no centro, mas numa imagem com maior concentração de pontos na esquerda do centro o centroide estará mais a esquerda do centro. Logo, existe a possibilidade de uma maior quantidade de pontos tornar o centroide mais expressivo.

Para testar esta hipótese, foi feito outro teste, mas no lugar de usar a área e o centroide, usará-se a área e a diferença do centro pro centroide.

Como o valor do ponto centroide é a coordenada x e y deste, duas figuras idênticas, mas uma transladada para o lado, teriam centroides diferentes. Então, para evitar este problema, a feature utilizada será a diferença do centro do círculo ajustado pelo centroide. Assim, essa diferença dependerá unicamente do objeto e não de sua posição relativa. Esta feature será chamada de centro ou centro relativo.

Utilizando essas duas features a tabela abaixo foi montada:

TABLE II
Features: Área e centro relativo

Threshold	**Área do contorno**	**Acurácia de predição**	
	Observações de Área 0	**KNN**	**LDA**
127	8	0.4592	0.4637
63	1	0.4146	0.4714
1	0	0.5001	0.5442

Utilizando esses dois features, vemos uma mudança bem mais expressiva na acurácia de predição do que quando foi utilizado apenas um feature. A LDA foi a que mais ganhou acurácia, aproximadamente 8%.

Considerando que uma observação é 1/60000 dos dados de teste, pode-se afirmar com maior certeza que o menor threshold aumenta a quantidade de informação que essas features traz da imagem.

Em ambas as tabelas I e II, o LDA teve melhor performance que o KNN. Uma das suposições do LDA é que a correlação entre as variáveis é baixa e a variância delas são próximas de iguais.

Calculando a tabela de correlação:

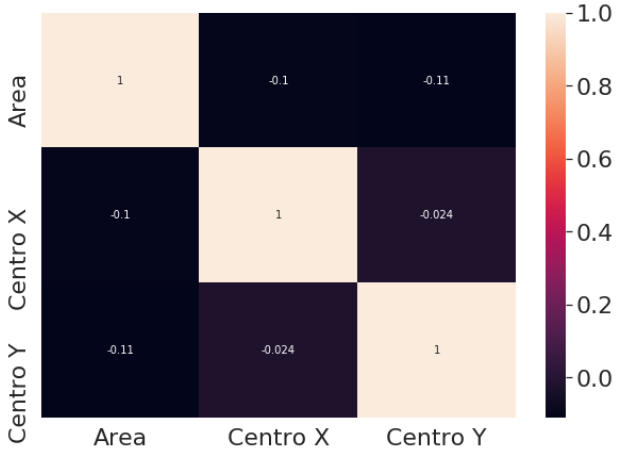

Fig. 1. Correlação entre as 3 variáveis

Da figura 1 nota-se que a correlação entre as variáveis (diferentes) é relativamente baixa. Isso é uma provável explicação

da melhor performance do LDA nessa situação, já que uma grande correlação pode diminuir a capavidade preditiva do algorítmo.

Então temos na figura 2 a matriz de confusão do melhor teste até agora, com threshold 1 e algorítmo LDA:

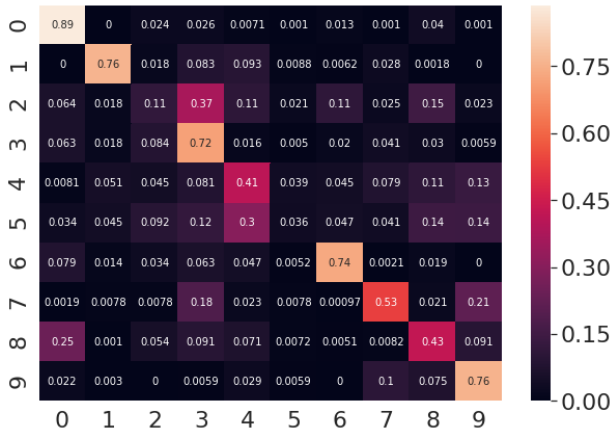

Fig. 2. Matriz de confusão

Apesar da acurácia ser de 54.42%, existem algums números que tem uma taxa de acerto muito baixa e outros com uma taxa muito alta.

É possível ver as distribuições das features através dos boxplots das figuras 3, 4 e 5

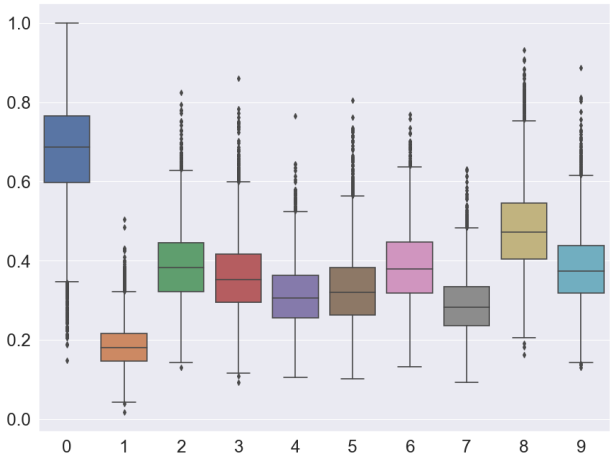

Fig. 3. Boxplot da feature área

Na figura 3 vemos que o centro da distribuição dos valores da área de 0 estão bem acima das demais, e as de 1 estão bem abaixo. Somente com esta feature podemos ver um grande potencial na diferenciação de 0 e 1 das demais.

Da mesma forma, na figura 5 o centro da distribuição de 6 está bem abaixo das outras.

Esses gráficos analisados explicam bem como os números 0, 1 e 6 tiveram ótimos resultados na predição.

Por outro lado, o 5 teve taxa de acerto de 3.6%, que é abaixo de uma taxa de acerto caso tentássemos predizer

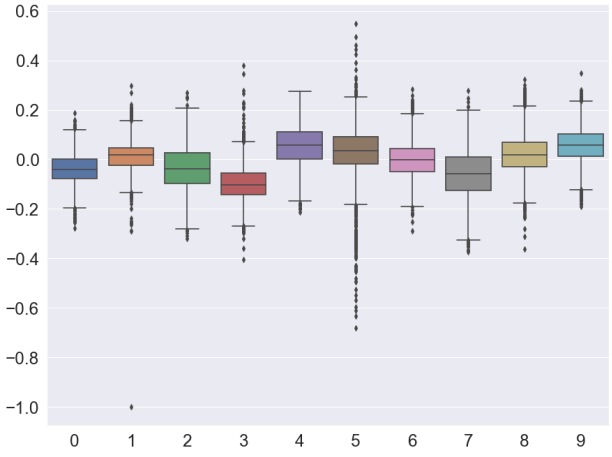

Fig. 4. Boxplot da feature centro em x

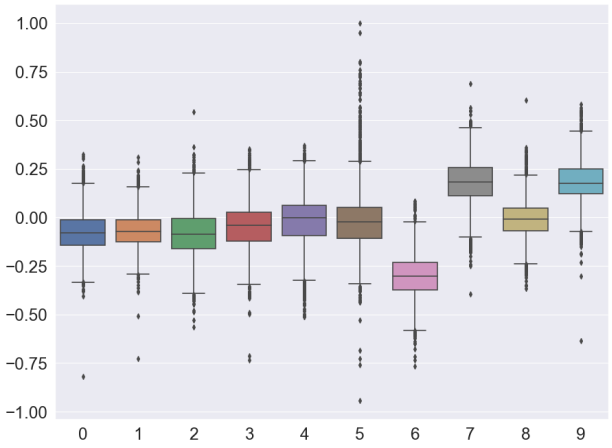

Fig. 5. Boxplot da feature centro em y

aleatoriamente. Olhando para os bloxplots das figuras 4 e 5, vemos que os valores de centro em ambos x e y estão extremamente dispersos, dificultando o encontro de padrões pelos algorítmos de aprendizagem.

B. Resultado final

Após obter o entendimento sobre o problema e tomar algumas decisões, o próximo passo é tentar aumentar a taxa de previsão adicionando outras features.

Além dos features da área e do centro relativo ao centroide, temos os pontos de extremo (também relativos ao centroide), o ângulo da melhor elipse ajustada e os eixos X e Y.

Os features foram escolhidos de modo que todos fossem invariantes a translação. Também foram escolhidos especificamente alguns que não fosse relativos a rotação e alguns que fossem. Tome o centro relativo como exemplo. O número 6 pode parecer como um 9 rotacionado 180 graus. Logo, o centro relativo pode ajudar a diferenciá-los pois varia com a rotação.

Verificamos na tabela III a invariância das features utilizadas.

Neste contexto, o resultado da acurácia foi:

TABLE III
INVARIÂNCIA DAS FEATURES UTILIZADAS

Feature	Invariância		
	Translação	**Rotação**	**Escala**
Centro relativo	Sim	Não	Não
Área	Sim	Sim	Não
Pts. de extremo relativos	Sim	Não	Não
Ângulo da elipse ajus.	Sim	Não	Sim
Raio do círculo ajus.	Sim	Sim	Não
Eixos X e Y	Sim	Não	Não

KNN: 0.7948 (79.48%)
LDA: 0.6894 (68.94%)

Diferentemente do teste inicial, onde o LDA teve melhor performance, com essas features o KNN se saiu melhor. Assim como anteriormente, podemos analisar a matriz de correlação em busca de uma possível explicação:

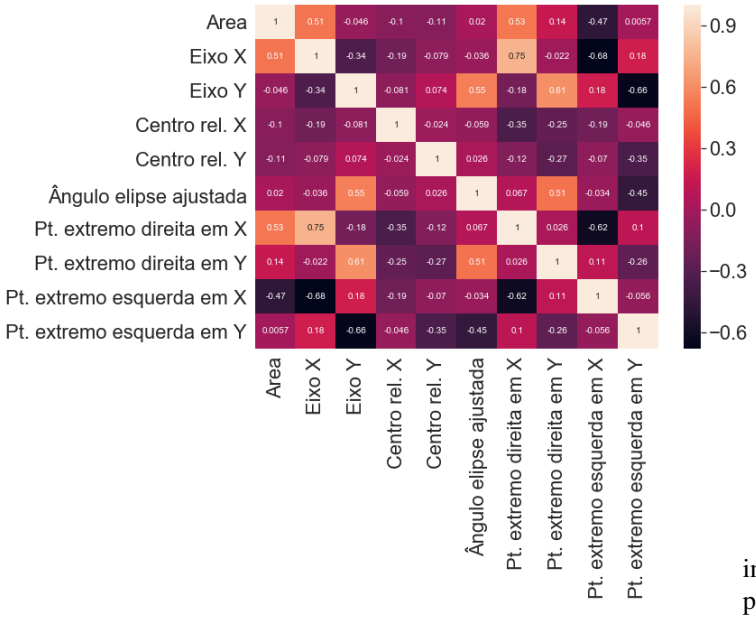

Fig. 6. Matriz de Correlação

Agora, a correlação entre as variáveis é bem mais significativa, impactando na performance de predição do LDA. Já o KNN é menos impactado por isso.

Por fim, a matriz de confusão de ambos KNN e LDA para comparação está nas figuras 7 e 8.

V. CONCLUSÃO

Após toda a análise feita, percebe-se que não há um algorítmo de aprendizado melhor que o outro, a performance depende do problema e dos features escolhidos. No início com poucas features o LDA se saiu melhor, mas quando mais features foram introduzidas e algumas tinham correlação forte, o poder de prediçao do LDA caiu e o KNN se destacou.

Uma possibilidade para melhorar a predição é de utilizar o QDA no lugar do LDA, pois este assume que as variâncias das classes são iguais, algo não muito provável.

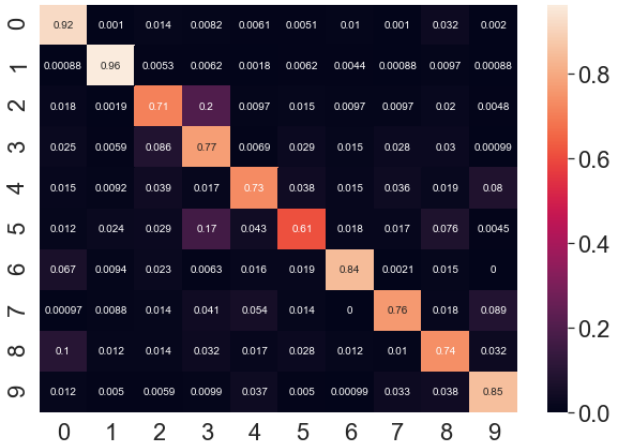

Fig. 7. Matriz de confusão KNN. 79.48% de acurácia

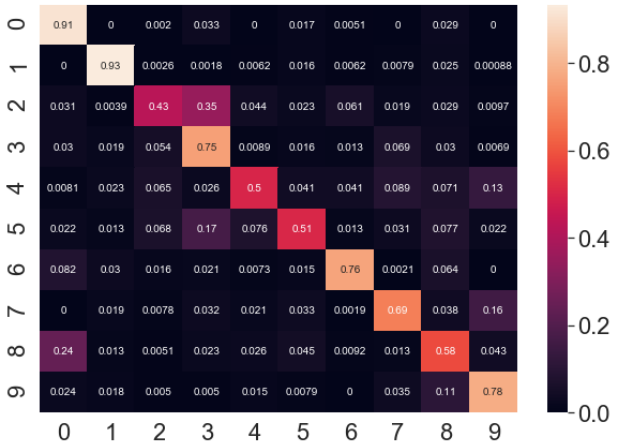

Fig. 8. Matriz de confusão LDA. 68.94% de acurácia

Não foram utilizadas momentos de Hu ou outras features invariantes a escala, rotação e translação. A utilização desses poderia aumentar significativamente a performance.