UNIVERSIDADE FEDERAL DO ABC

Tópicos em Inteligência Artificial: Machine Learning

PROJETO 1: Implementação e análise do algoritmo KNN

WILLIAM DE SOUZA GOMES

PROF. DR. André Kazuo Takahata

Santo André

Março

1 Resumo

Neste trabalho analisa-se o algoritmo de classificação KNN, passando por sua implementação e após a implementação estuda-se como o algoritmo se comporta para diferentes datasets, tanto computacionalmente quando nas suas métricas. O trabalho é separado em 3 partes. Parte I discuti-se os resultados do KNN para um conjunto de dataset mais simples com algumas amostras. Parte II procura-se discutir com um conjunto de dataset gerado por funções na qual tem caracterpisticas bem definidas e por fim, a parte III discuti-se sobre o dataset The Olivetti Face.

Sumário

2 Introdução

Algoritmos de classificação são algoritmos na qual dado um conjunto de dados e seus rótulos o algoritmo aprende com as características desses dados e depois generaliza para novos dados retornando seus rótulos. Tais algoritmos são os mais variados, numa complexidade de dados não seria viável, atualmente, um único tipo de algoritmo na qual se encaixaria bem para todos os tipos de problemas. Cada tipo de algoritmo tem suas forças e fraquezas e no que diz respeito a forças há pelo menos um tipo de problema na qual esse algoritmo se sairá bem. Como poderíamos classificar um conjunto de dados na qual sua distribuição se dá num plano cartesiano, como mostra a Figura 1. Para esse caso seria interessante dividir o espaço cartesiano em um fronteira de decisão, talvez essa não seja a melhor abordagem, e se, escolhêssemos o vizinho mais próximo, já seria uma melhor abordagem, mas pode-se melhorar um pouco mais, ao invés do vizinho mais próximo, escolha-se um número de vizinhos mais próximos, no caso, k vizinhos mais próximos.

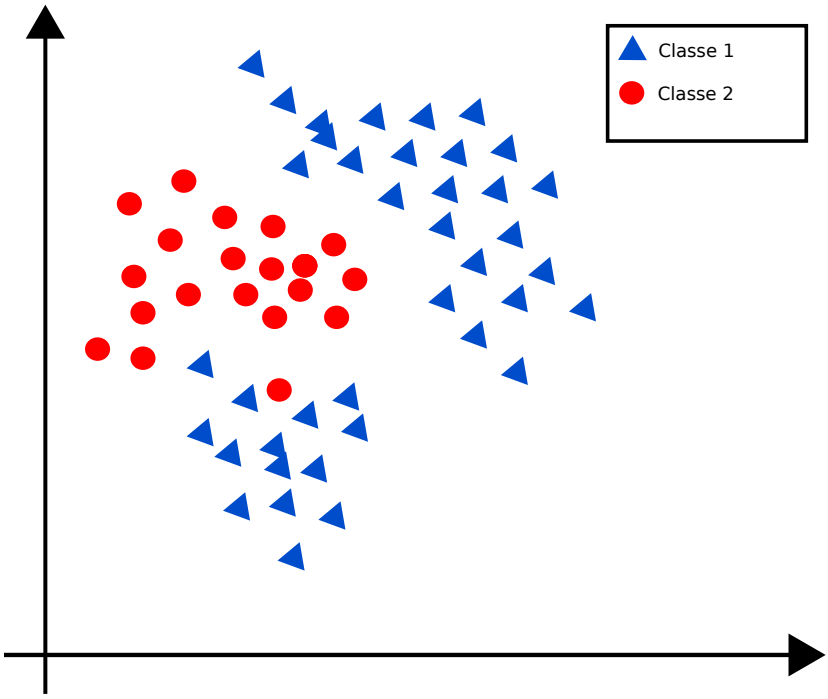

Figura 1: Representação 2d de um conjunto de dados

E nesse método que se baseia o k-Nearest Neighbors, conhecido como KNN [1]. Dado uma amostra interessa-se saber qual seria sua classe pertencente, para tal o KNN calculará a distância dessa amostra em relação a todos as outras amostras presente no

conjunto e após feito isso, escolhe-se os k mais próximos vizinhos e tira-se a moda e por fim a classe na qual pertence essa moda. O cálculo da distância mais usual é a distância euclidiana, então dado um vetor com n dimensões a expressão é dada na Equação 1

$$d = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + (p_3 - q_3)^2 + ... + (q_n + -p_n)^2} \quad (1)$$

3 Metodologia

3.1 Parte I

A parte I é feita a análise do algoritmo KNN com o total de seis datasets, na qual, sua distribuição espacial é dada pela Figura 2. Os datasets são compostos por 3 características e o número de amostras é pequena. Fora treinado para cada um desses dataset um número de vizinhos de 1 à 10 para valores ímpares. Para o treinamento é usado duas implementações do KNN, a primeira pela autor desse trabalho e a segunda pelo Sckit-Learn. Após feito o treinamento é computado os valores de acurácia, precisão, revocação e o tempo de execução para cada k, aplica-se para ambos os algoritmos.

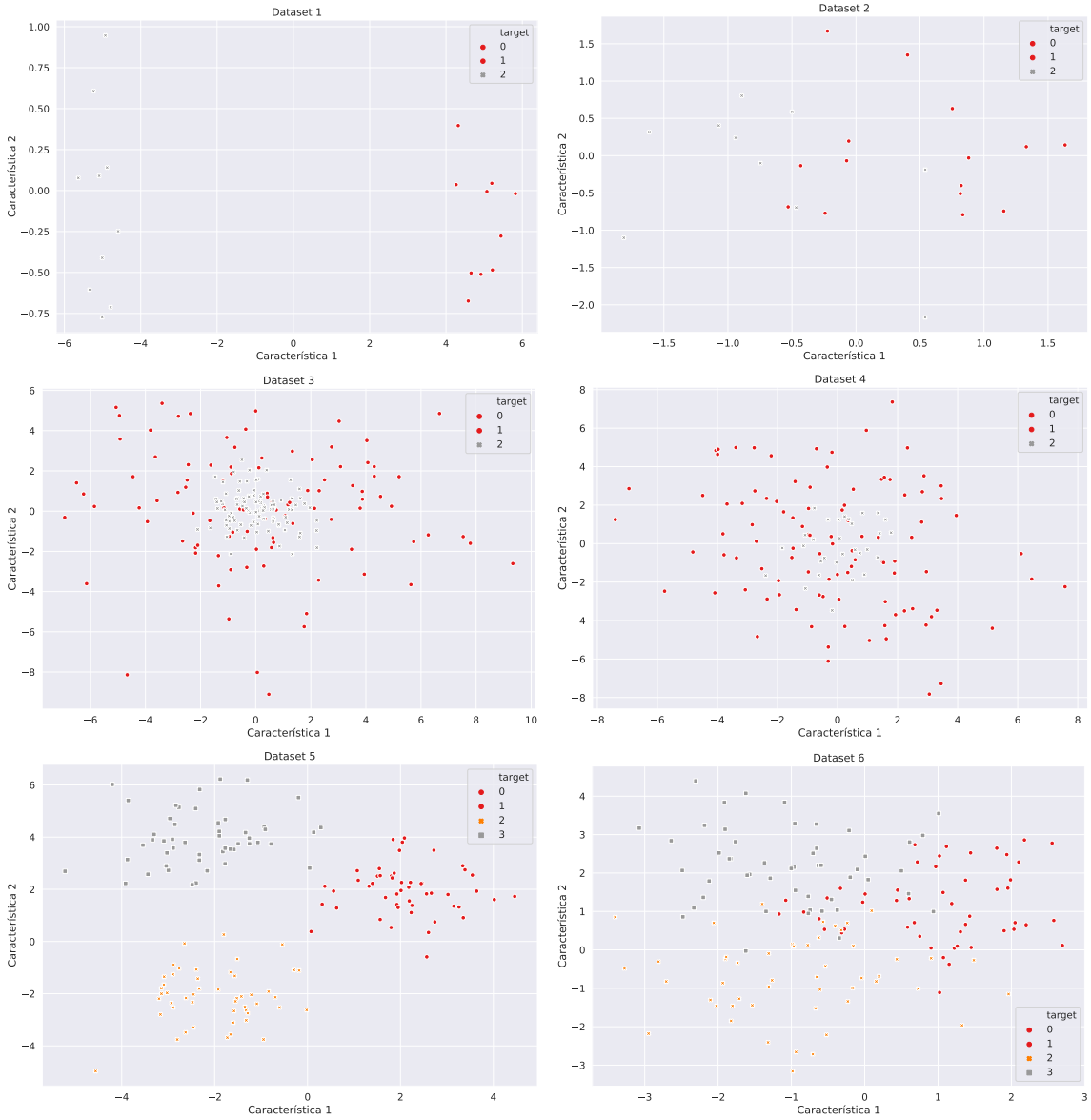

Figura 2: Distribuição dos dados para os 6 conjunto de dados parte I

3.2 Parte II

A segunda parte da avaliação do algoritmo KNN é feito usando um conjunto de dataset com características bem marcantes, suas fronteiras são bem definidas, como mostra a Figura 3. O número de características se mantém em duas dimensões, o número de classes são 2, 2, 4, 2, 2 e 4, respectivamente. O número de amostras sobe consideravelmente num intervalo de 600 à 2000 amostras.

Para esses conjuntos de dataset é feito o treinamento utilizando o hold-out com separação de 25% para teste e o número de vizinhos segue um intervalo de 1 à 20 para valores ímpares e é utilizado duas implementações do KNN, uma feita pelo autor do

relatório e o segundo implementado pela biblioteca em python Scikit-Learn. Os resultados obtidos ao final do treino são a acurácia, precisão, revocação e o tempo de execução para cada k.

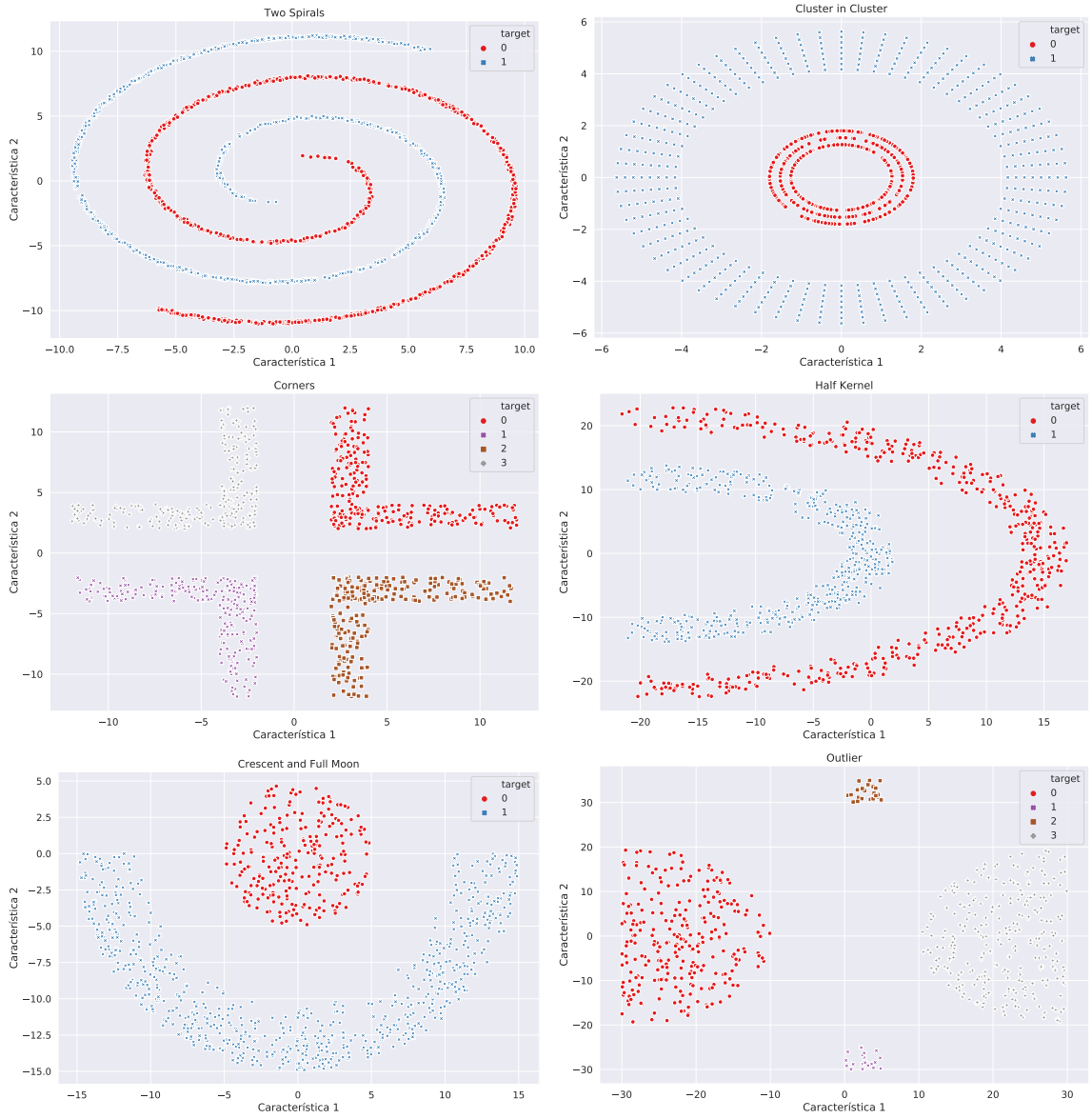

Figura 3: Distribuição dos dados para os 6 conjunto de dados parte II

3.3 Parte III

O último conjunto de testes é feito com o dataset Olivetti Faces [2]. É um dataset composto de imagens de face de pessoas feita entre 1992 e 1994 nos laboratórios AT&T, são um total de 400 amostras, 4096 características e 40 classes.

Para esses conjuntos de dataset é feito o treinamento utilizando o hold-out com separação de 25% para teste e o número de vizinhos segue um intervalo de 1 à 20 para

valores ímpares e é utilizado duas implementações do KNN, uma feita pelo autor do relatório e o segundo implementado pela biblioteca em python Scikit-Learn. Os resultados obtidos ao final do treino são a acurácia, precisão, revocação e o tempo de execução.

4 Resultados

4.1 Parte I

O primeiro conjunto de dados são de complexidade computacional menor e contém poucas amostras. O dataset 1 e 5 tem suas classes bem segmentadas e para todos os valores de k usados suas métricas alcançaram 100% como pode ser visto na Figura 4.

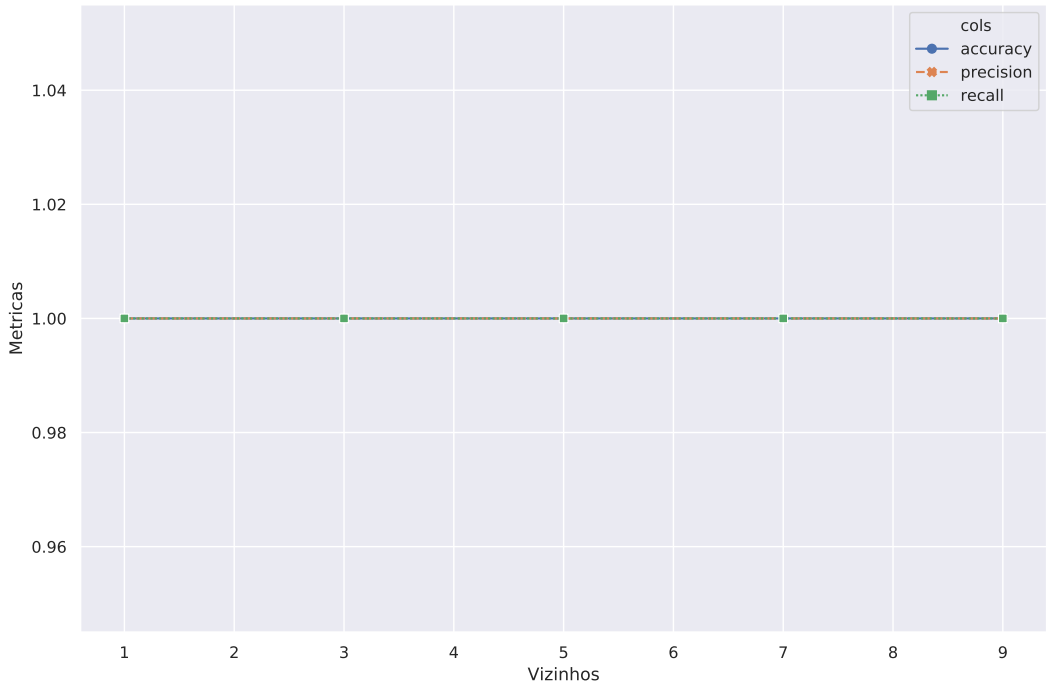

Figura 4: Resultados das métricas em função do K para o Dataset 1

Os resultados da acurácia, precisão e revocação dos dataset 2, 3, 4 e 6 podem ser visto na Figura 5, respectivamente. O dataset 3 e 6 tem um resultado interessante, o dataset 3 as métricas seguem uma tendência de aumento para os k vizinhos enquanto o dataset 6 segue uma tendência de queda dos resultados.

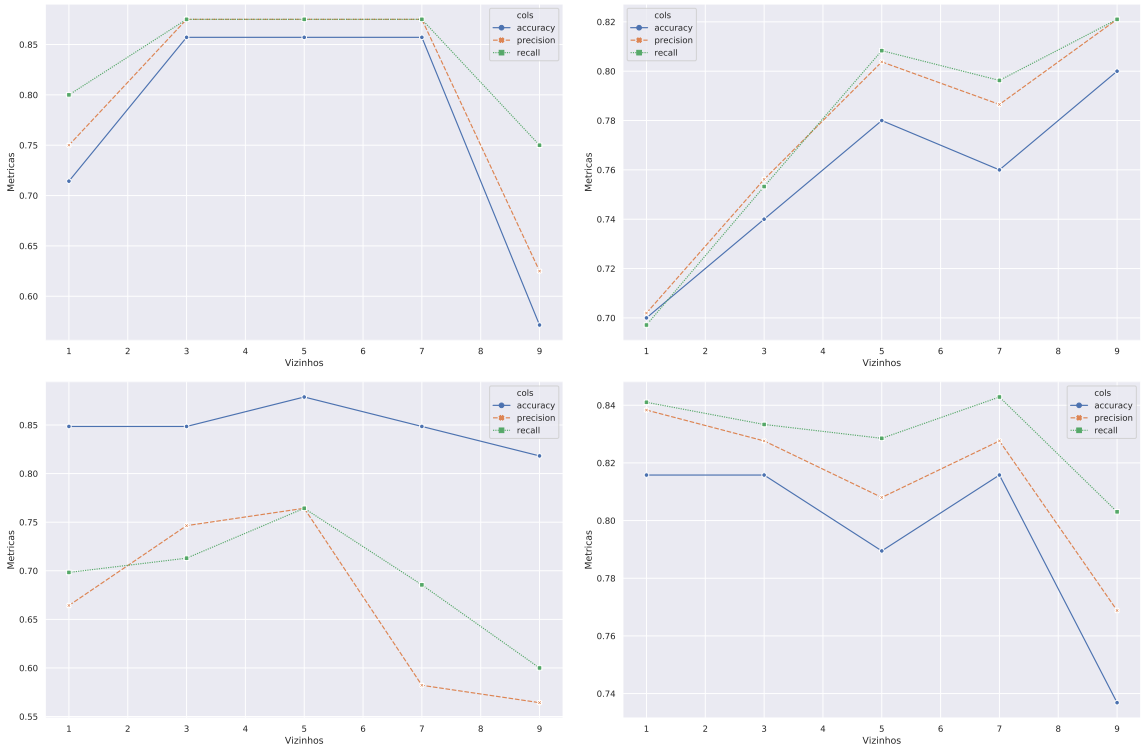

Figura 5: Distribuição dos dados para os 6 conjunto de dados parte I

O dataset 2 apresenta uma melhora nos resultados de suas métricas, se mantém constante por alguns K vizinho e quando atinge k = 9 cai acentuadamente, principalmente para a acurácia, ficando abaixo dos 60%. Por último, o dataset 4 apresenta um comportamento interessante, enquanto sua acurácia se mantém aproximadamente constante para a precisão e revocação tem uma queda vertiginosa em seus resutados, assim, indicando uma grande dispersão dos dados e de fato, ao olhar sua distribuição espacial isso se confirma

4.2 Parte II

O segundo conjunto de datasets são gerados por funções e tem seus segmentos bem definidos e não só isso, seus dados tem baixíssima dispersão dada sua classe, a priori, podemos dizer que para valores mais baixos e médios de k o algoritmo terá um ótimo desempenho. E de fato, após rodada as simulações esse comportamento é comprovado, como pode-se observar na Figura 6. Todas as três métricas se manteram em 100% para todos os K rodado pelo algoritmo.

Figura 6: Distribuição dos dados para os 6 conjunto de dados parte II

Outro aspecto importante é sobre o desempenho do algoritmo. Os testes foram rodados tanto na implementação do autor desse relatório, quanto na implementação no sckit-learning e como os dataset são maiores nesse conjunto II o tempo de execução aumenta consideravelmente, como mostra a Figura 7. E vale observar a Figura a7 e a Figura 8 que o tempo de execução para a própria implementação aumenta vertiginosamente enquanto para a implementação pelo Scikit-learn o tempo contínua abaixo do 1 segundo. Vale-se dizer que o Sckit-learn usa todas as threads do processador.

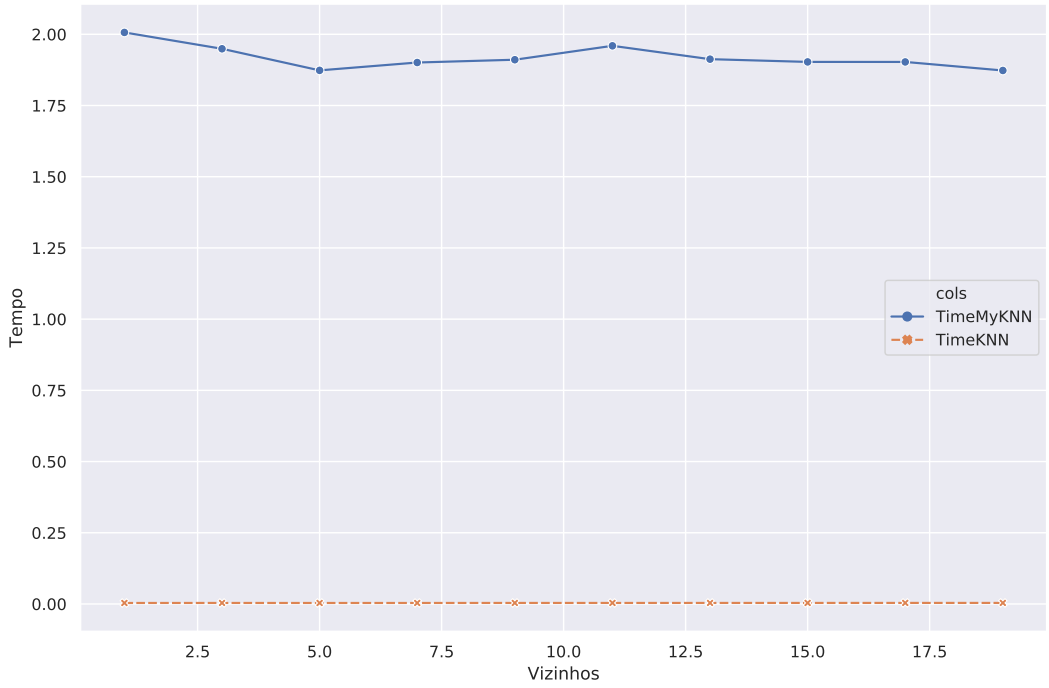

Figura 7: Comparação de tempo entre diferentes implementações do KNN para o Outlier Dataset

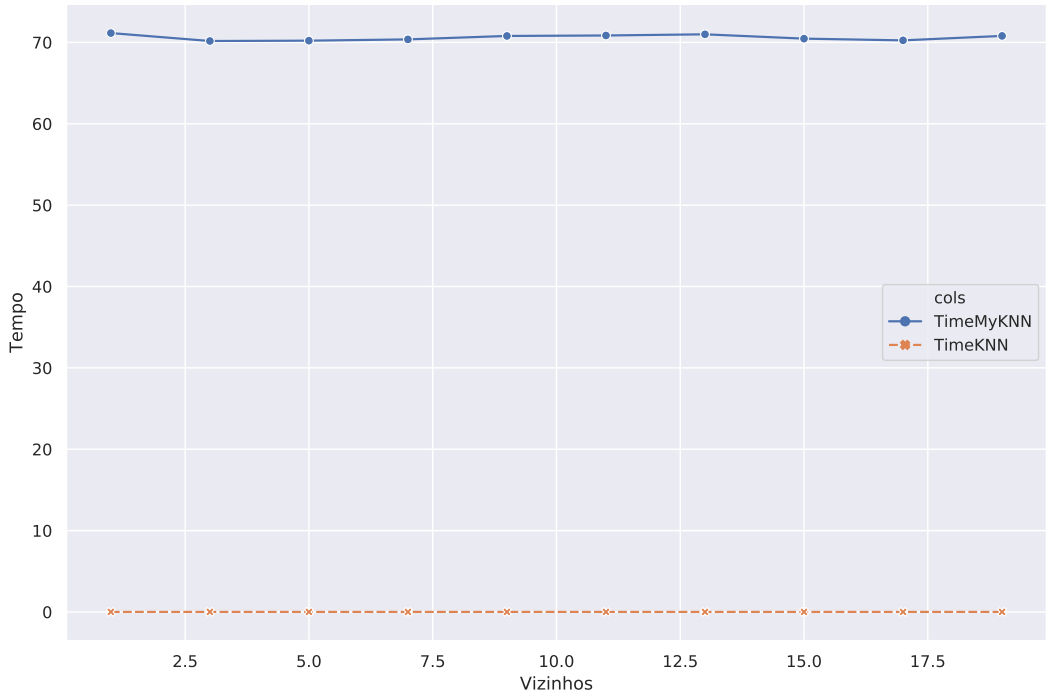

Figura 8: Comparação de tempo entre diferentes implementações do KNN para o TwoSpirals Dataset

4.3 Parte III

Por último, foi testado o algoritmo com um dataset de imagens, bem famoso por servir como exemplo de classificação de imagens, Olivetti Faces. Esse é um dataset bem

mais complexo, são um total de 400 amostras, 4096 características e 40 classes, resultando numa matriz de mais de 1,6 milhões de dados.

Com tamanha complexidade em relação aos outros grupos de datasets testado anteriormente, o tempo de execução aumentou muitíssimo, como pode ser visto na Figura 9. O tempo para a implementação prórpia ficou em mais de 40 minutos e para o Sckit-Learn ficou ainda sim muitíssimo rápido abaixo do 1 minuto.

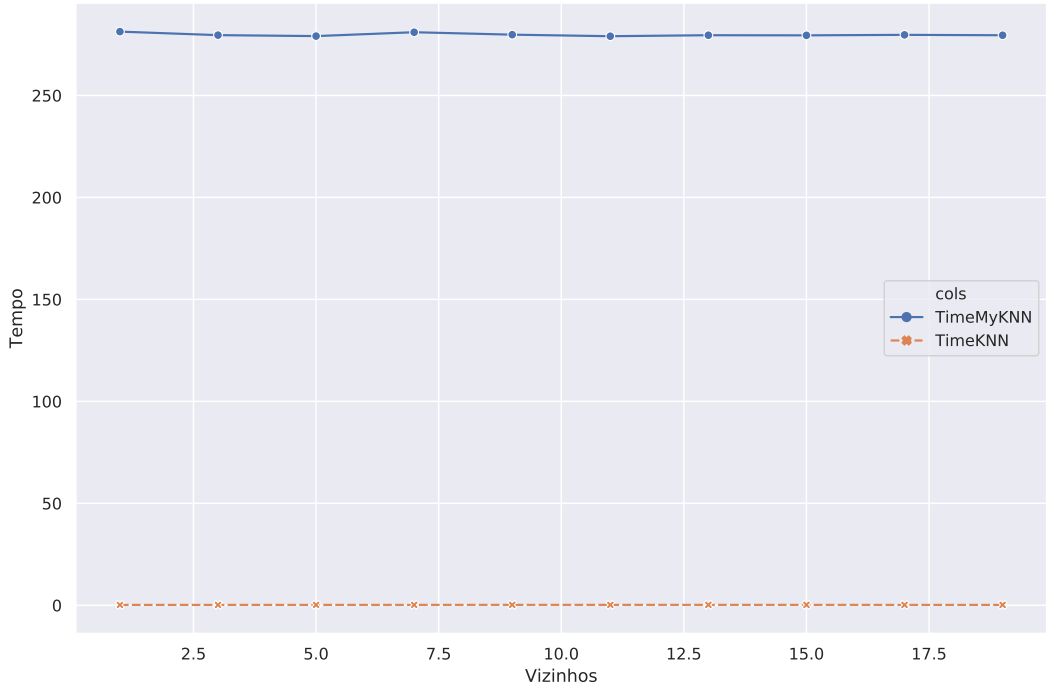

Figura 9: Tempo de execução do KNN com o Olivetti Faces para diferentes implementações

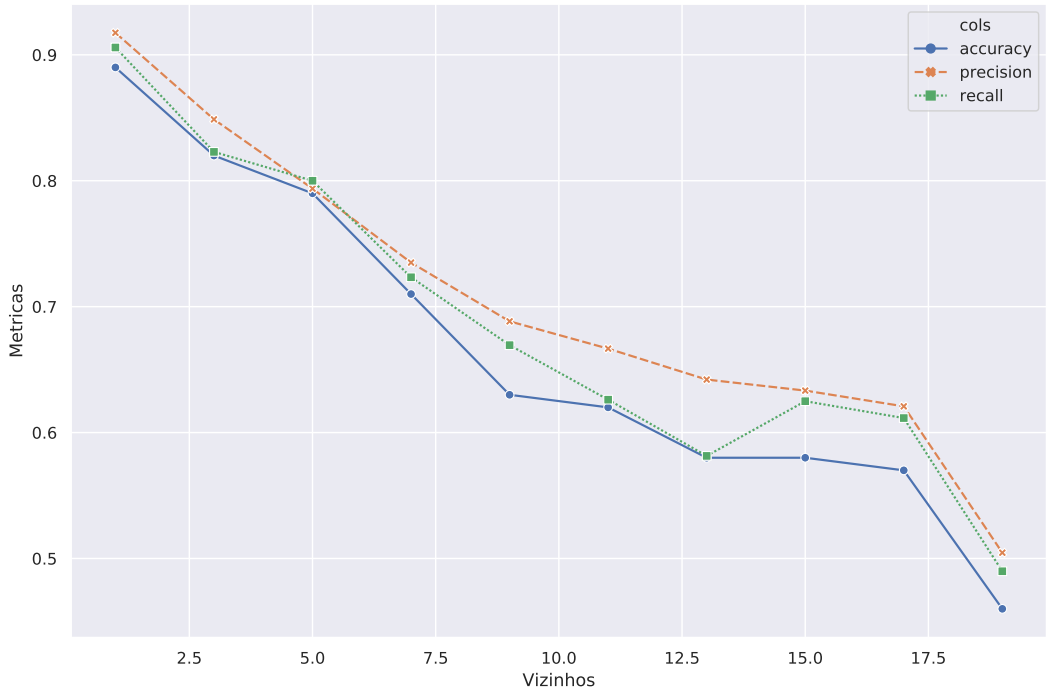

Figura 10: Métricas em função do k para o Olivetti Faces

Para as métricas do dataset, as métricas seguem uma tendência de queda para os K vizinhos. Para o k =19 todas as três métricas ficam abaixo dos 50%

Uma maneira muito intuitivida de visualizar os resultado é através da matriz de confusão, como se pode observar pelas Figuras 11 e 12 as colunas representam as classes enquanto as linhas representam os valores preditos. Como se oberva na figura 12 é escolhido o melhor resultado de k e como é esperado os valores ficam na diagonal principal mostrando assim que as classes preditas batem com as classes verdadeiras. Já a Figura 11 é escolhido a matriz de confusão para o pior resultado do KNN que é para o K = 19 e observa-se que muitos pontos de calor estão fora da diagonal principal, o que indica que o classificador não acertou as classes para muitos casos. O matriz de confusão é uma ótima maneira de visualizar os resultados de seu classificador.

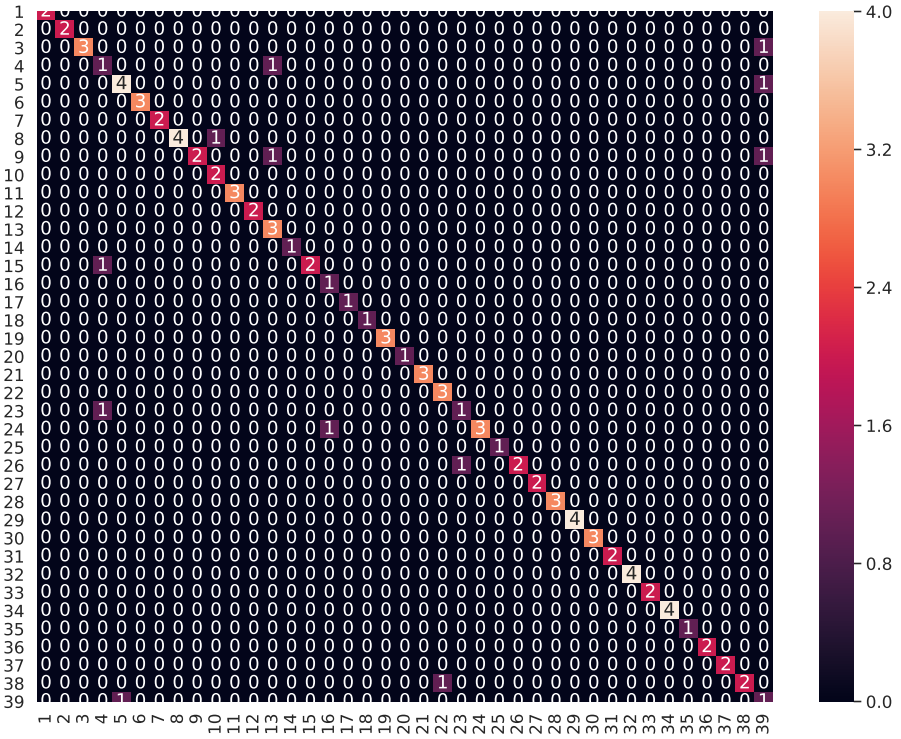

Figura 11: Matriz de Confusão para K = 1

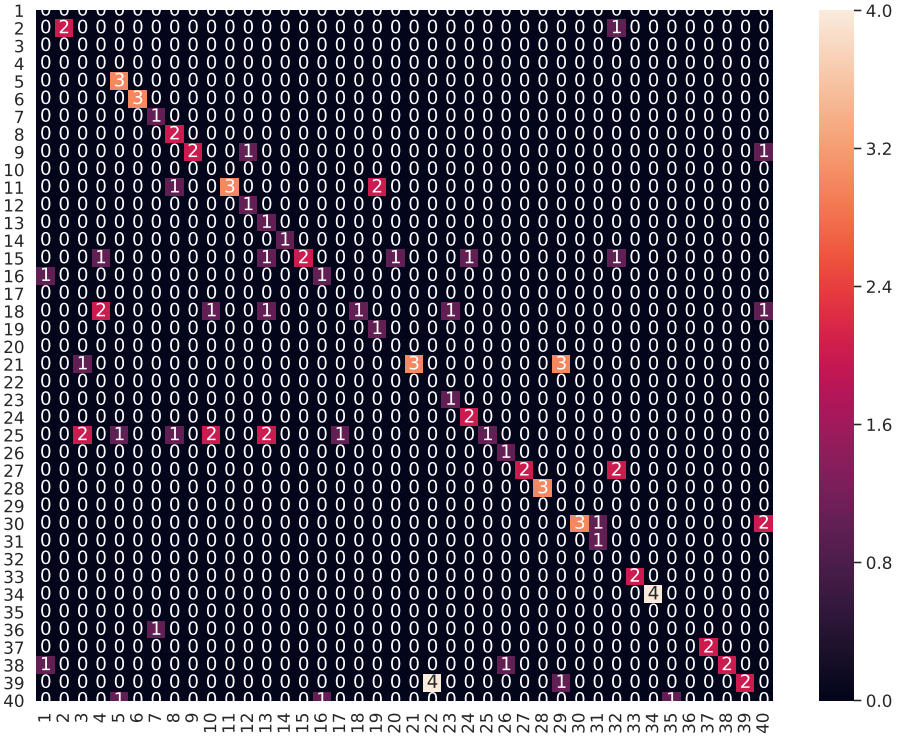

Figura 12: Matriz de Confusão para K = 19

5 Considerações Finais

O knn é um algoritmo muito simples de ser implementado e sua lógica de funcionamento também o que levante uma dúvida quando sua qualidade em classificar e como pudemos ver ele se saiu muito bem para muitos casos o que levante a conclusão de que a escolha de um classificador não deve ser avaliado apenas sua robustez, mas olhar para o dataset e em como os dados estão distribuídos, quantas classes há e etc. Ajuda bastante na escolha do classificador. Com o KNN observou-se que para dados bem segmentados e com pouca dispersão o KNN se mostrou extremamente bom alcançando por muitas vezes valores acima de 95%. Mas há um preço a se pagar por isso, quanto maior o dataset mais custoso computacionalmente o classificador fica e o tempo aumenta bastante.

Vale apena também separar muito bem quais métrica há a necessidade de usar e qual seu significado dentro de um problemas, no trabalho pode-se observar que mesmo para uma acurácia mantendo-se aproximadamente constante, outras métricas como a precisão e revocação caíram bastante quando se aumentava o número de vizinhos. Ou seja, para cada tipo de problema é importante definir no que se pretende chegar e qual fator é mais importante para tais escolhas.

Mais uma vez técnicas de paralelização se mostraram eficientes quando o tempo de execução de algoritmos mais complexos caíam vertiginosamente mostrando que uma boa implementação é de extrema importância para rodar datasets maiores e mais complexos e por útltimo a escolha dos classificadores é uma decisão baseada também nos dados, como os dados se comportam e para o KNN dados segmentados e poucos dispersos o KNN é um candidato ideal

6 Referências

[1] P. e. S. D. Duda, R.O. e Hart, *Pattern Classification*. Wiley, 2012. [Online]. Available: https://books.google.com.br/books?id=Br33IRC3PkQC

[2] "fetch olivetti faces," Sckit-Learn. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.datasets.fetch_olivetti_faces.html#sklearn.datasets.fetch_olivetti_faces