Performance Analysis of K-nearest Neighbor Algorithms

Project Presentation

COMP 5704: Parallel Algorithms and Applications in Data Science

Fall 2022

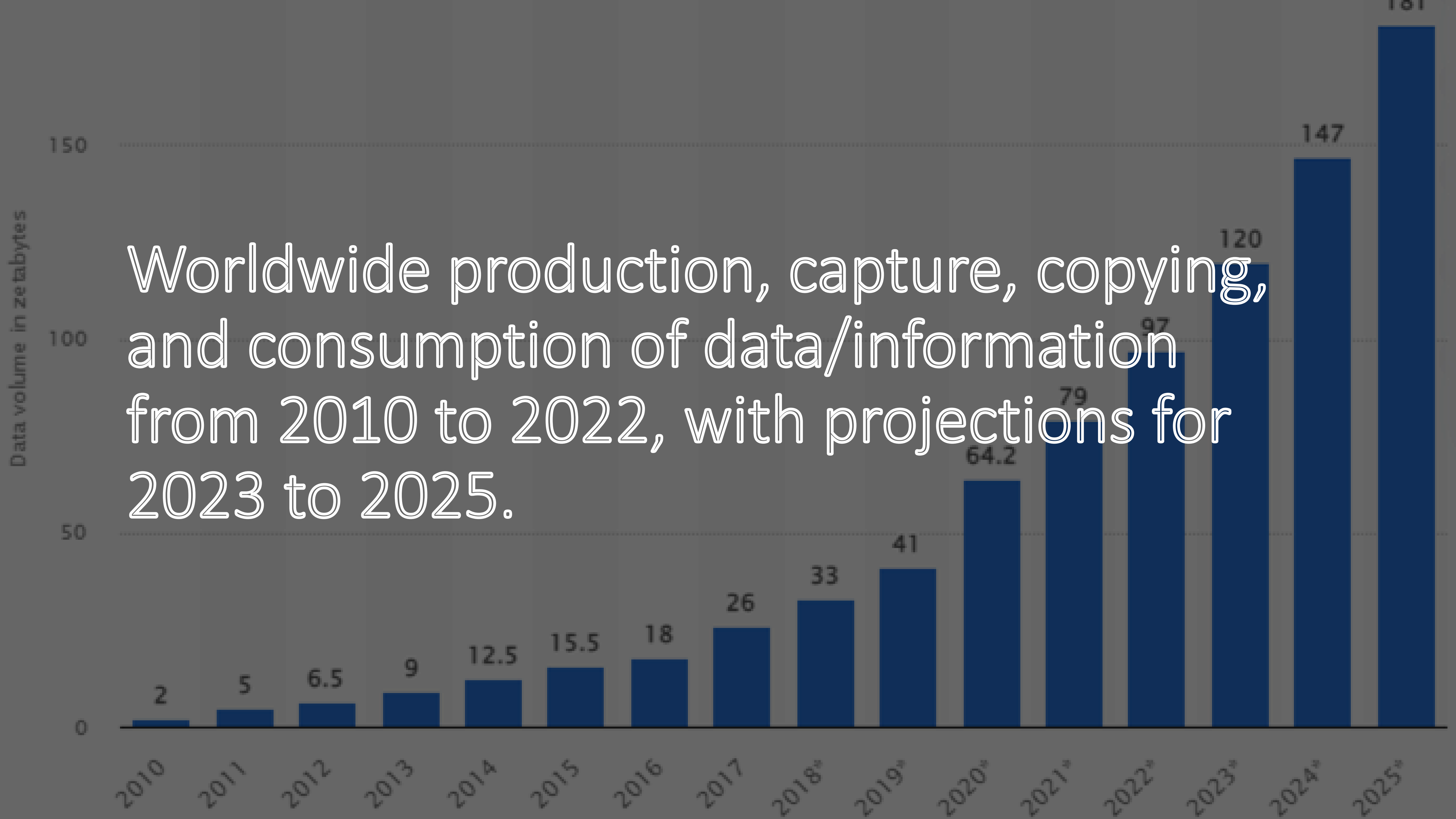
Prepared by: Heli A. Patel Professor: Prof. Frank Dehne

Carleton University

Outline

- Introduction
- K-Nearest neighbor
- Need of Parallel Processing
- Parallel KNN
- Experiment
- Evaluation
- Results
- Conclusion
- Discussion Questions
- References

Worldwide production, capture, copying, and consumption of data/information from 2010 to 2022, with projections for 2023 to 2025.



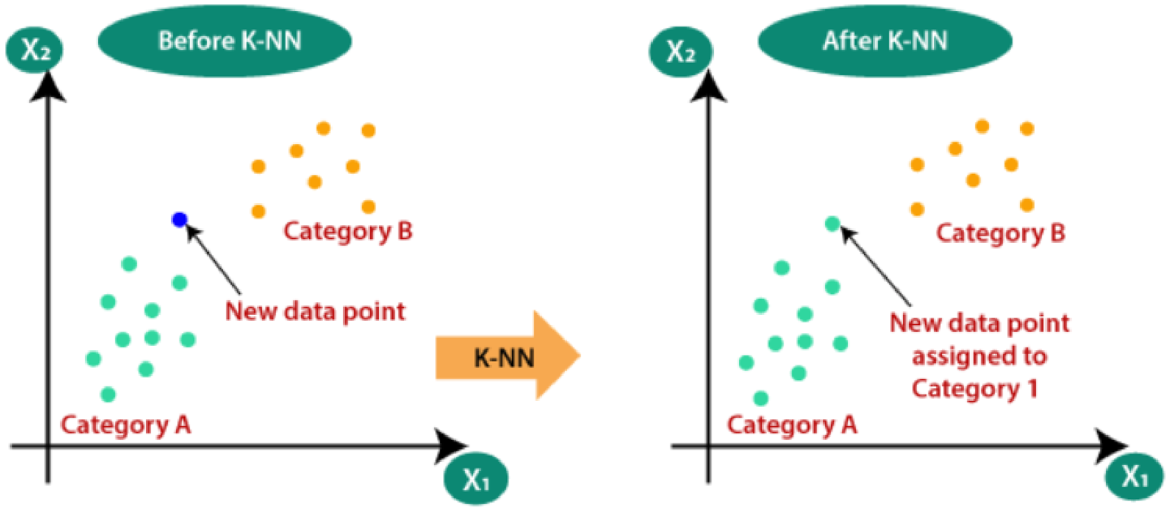
Data/Information is the central in how we live.

Data mining, which uncovers patterns in huge databases.

Classification is one of the core techniques in data mining.

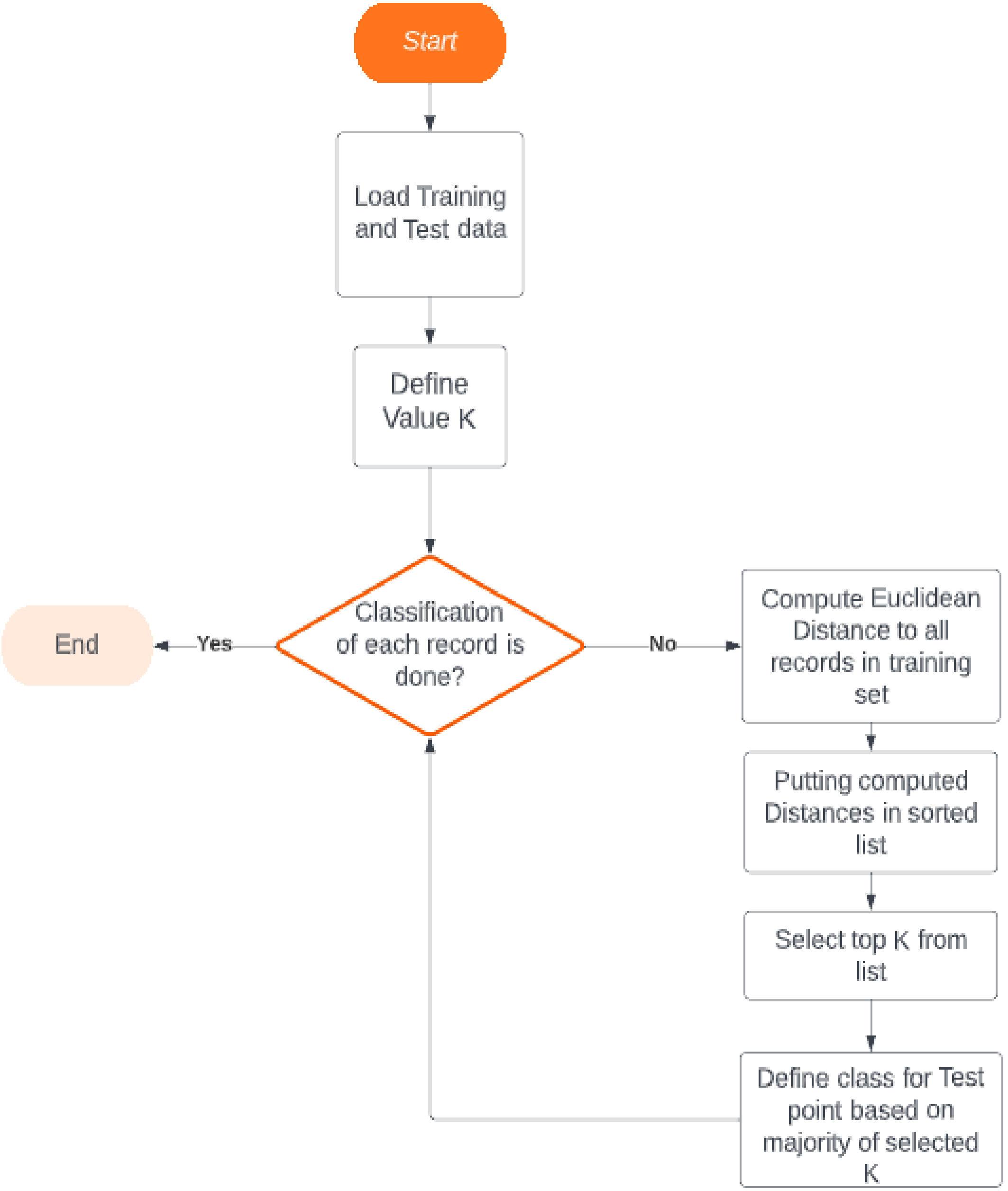
K-Nearest Neighbor

- The most popular classification method is K-Nearest Neighbor (KNN).
- One of the top ten most important and commonly applied algorithms.
- Using the attributes and training dataset, this algorithm aims to characterize new objects.



KNN Overview [5]

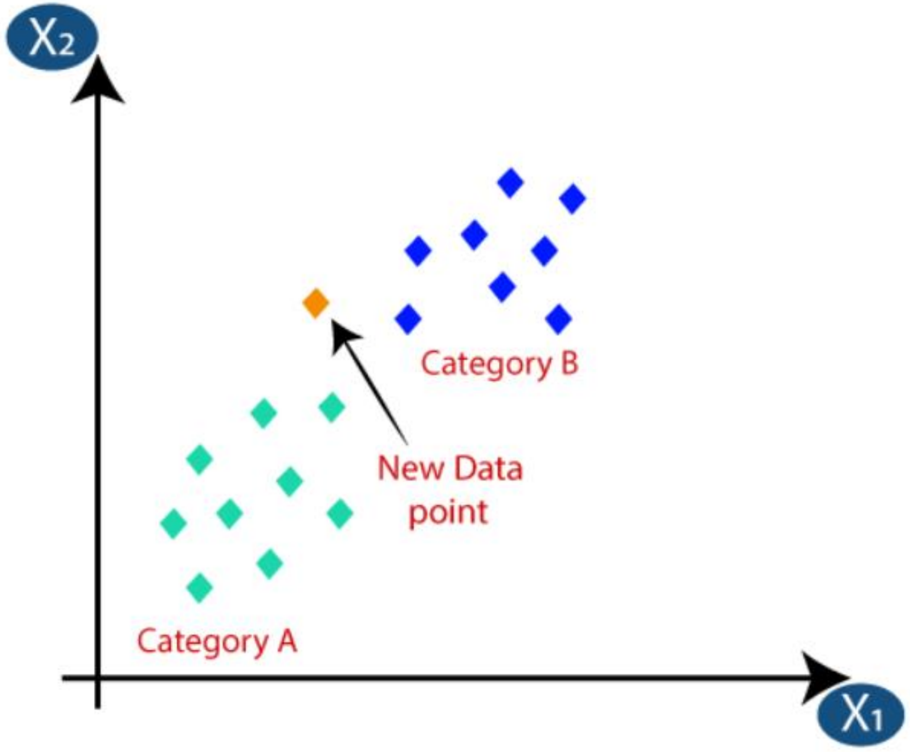
K Nearest Neighbor



KNN Algorithm

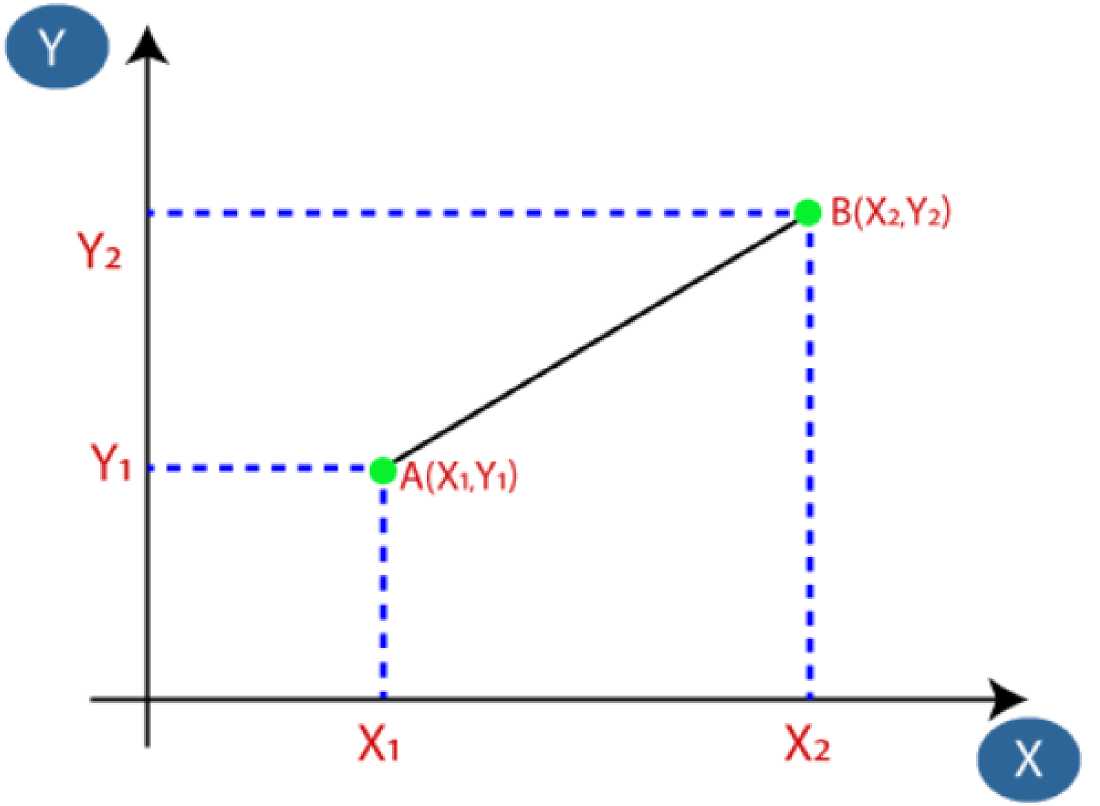
K-Nearest Neighbor

- Suppose, we have a new data point, and we need to put it in the required category

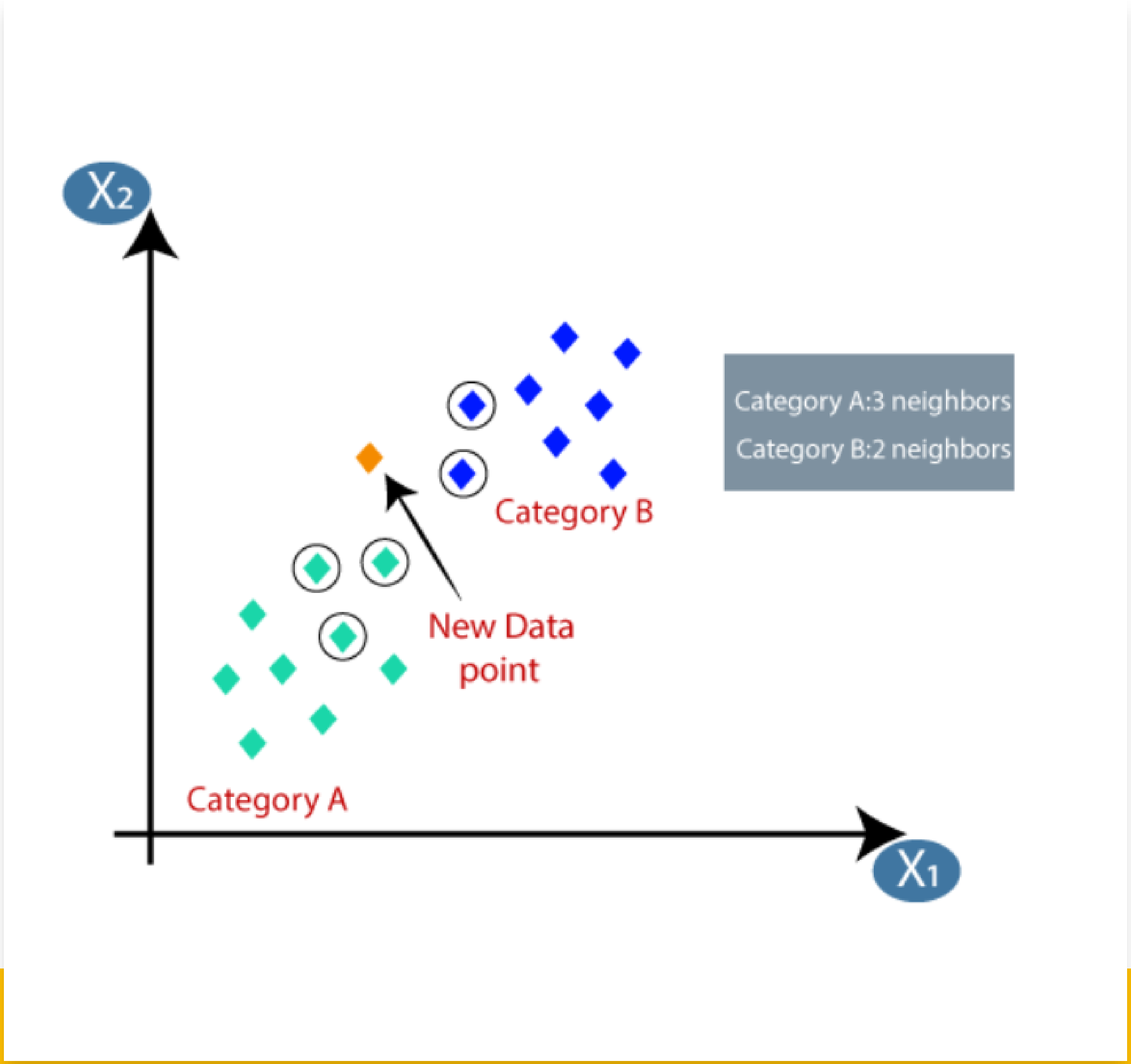


KNN-Euclidean Distance

- Firstly, we will choose the number of neighbors, so we will choose the k=5.
- Next, we will calculate the **Euclidean distance** between the data points.



Euclidean Distance between A_1 and $B_2 = \sqrt{(X_2-X_1)^2+(Y_2-Y_1)^2}$

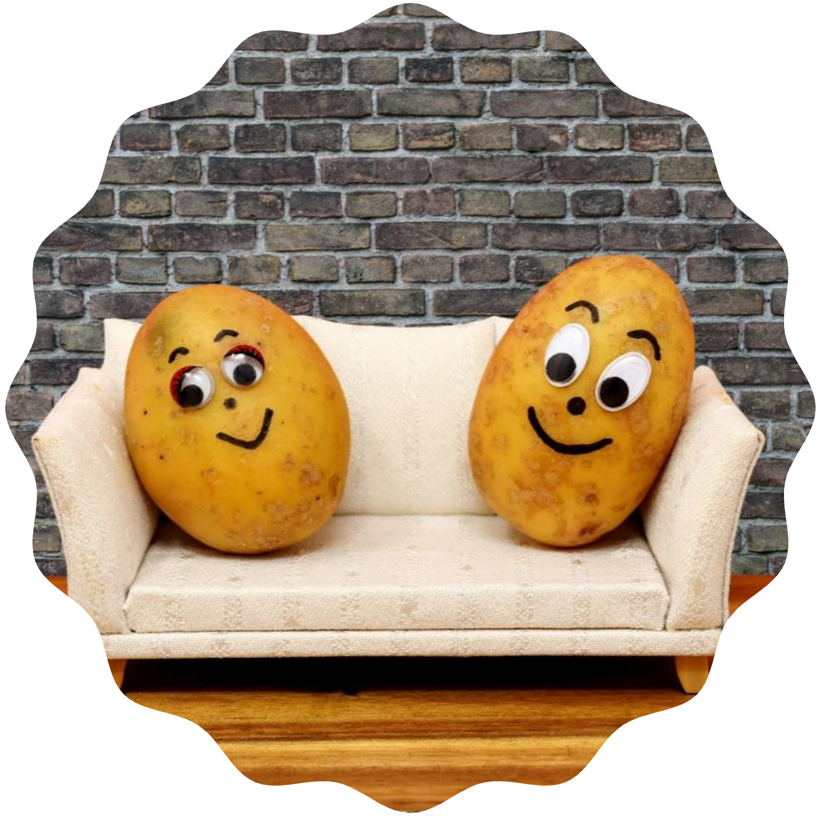


KNN- Euclidean Distance

- By calculating the Euclidean distance, we got the nearest neighbors, as three nearest neighbors in category A and two nearest neighbors in category B.
- we can see the 3 nearest neighbors are from category A, hence this new data point must belong to category A.

Laziness of KNN

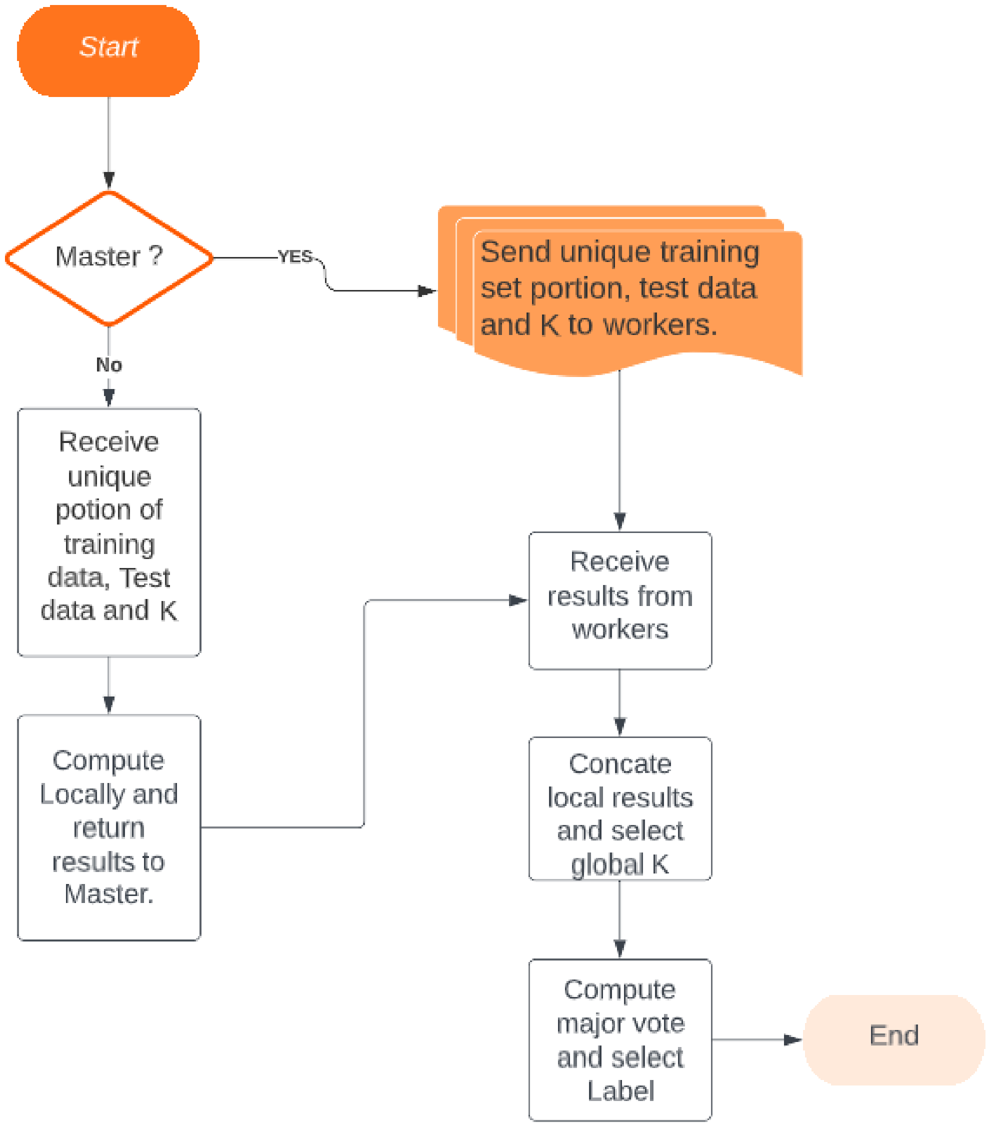
- KNN classifier is a model-free lazily learning algorithm.
- KNN classifier requires to store all the training instances in memory in order to locate all K nearest neighbors for a test sample.
  - A large memory requirement
  - A slow processing speed
  - The computation cost is high because of calculating the distance between the data points for all the training samples.



Lazy Learner [6]

The requirement for Parallel Processing

- To address the problem the computational complexity.
- Time complexity can be significantly reduced with the use of parallel processing.
- Utilize KNN to tackle Big data problems.
- Utilized Message Passing Interface(MPI) to parallelize KNN algorithm.



Parallel K Nearest Neighbor

Experiments

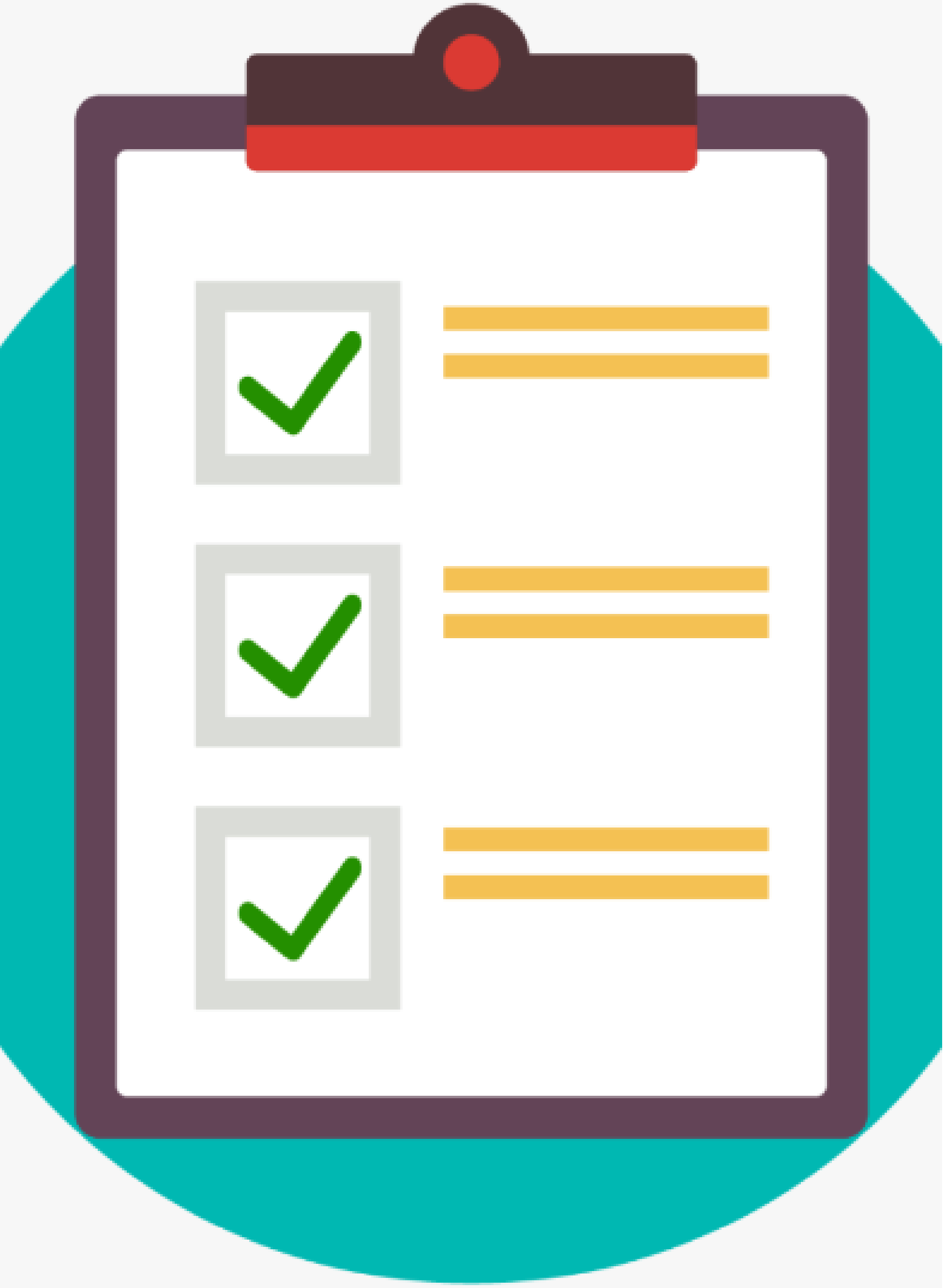
Scenarios

K = 5 was specifically chosen as the number of closest neighbors, and various **training dataset sizes (ranging from 1000 to over 200k records)** and **multiprocessor counts were examined(Processor counts=2,4 and 7).**

On a training dataset of 10,000 records, examine with **various values of k from 5 to 25 nearest neighbors,** using various **multiprocessor counts(Processor counts=2, 4 and 7).**

Data

- Credit fraud detection dataset from Kaggle [3].
- Data divided into training data and test data in prior.
- ~200k training entries and ~56k test entries
- Credit fraud ('1') or not ('0').

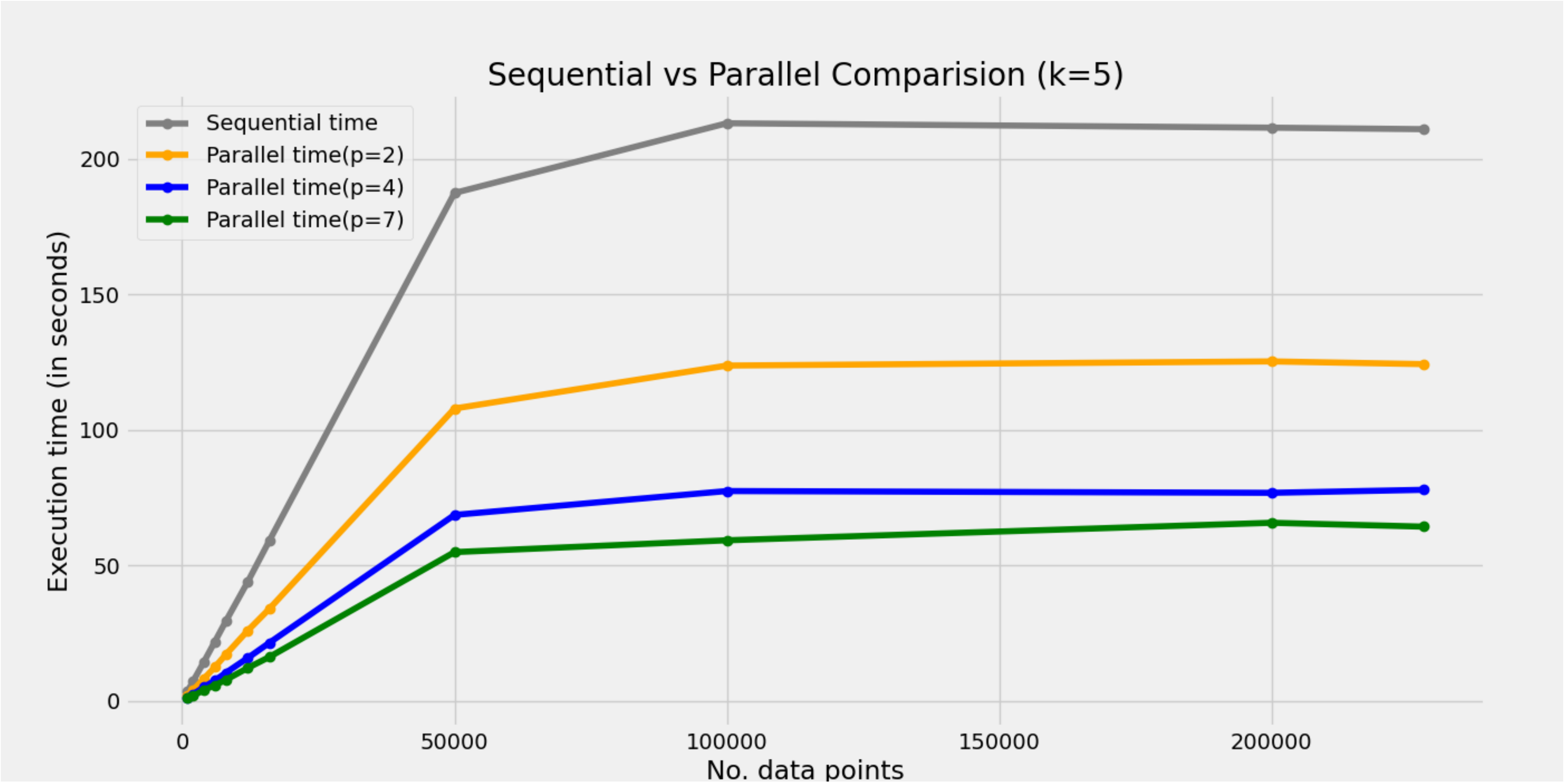


Evaluation

1. Speedup:
   - The ratio of the time needed by Sequential algorithm to solve a problem on one processing element (TS) to the time needed by the parallel algorithm (Tp) to solve the same problem on p identical processing element.
   - S = Ts/ Tp
2. Efficiency:
   - Efficiency is the ratio of speedup to the number of processors (p).
   - E = S/p = Ts/(p*Tp)

Results

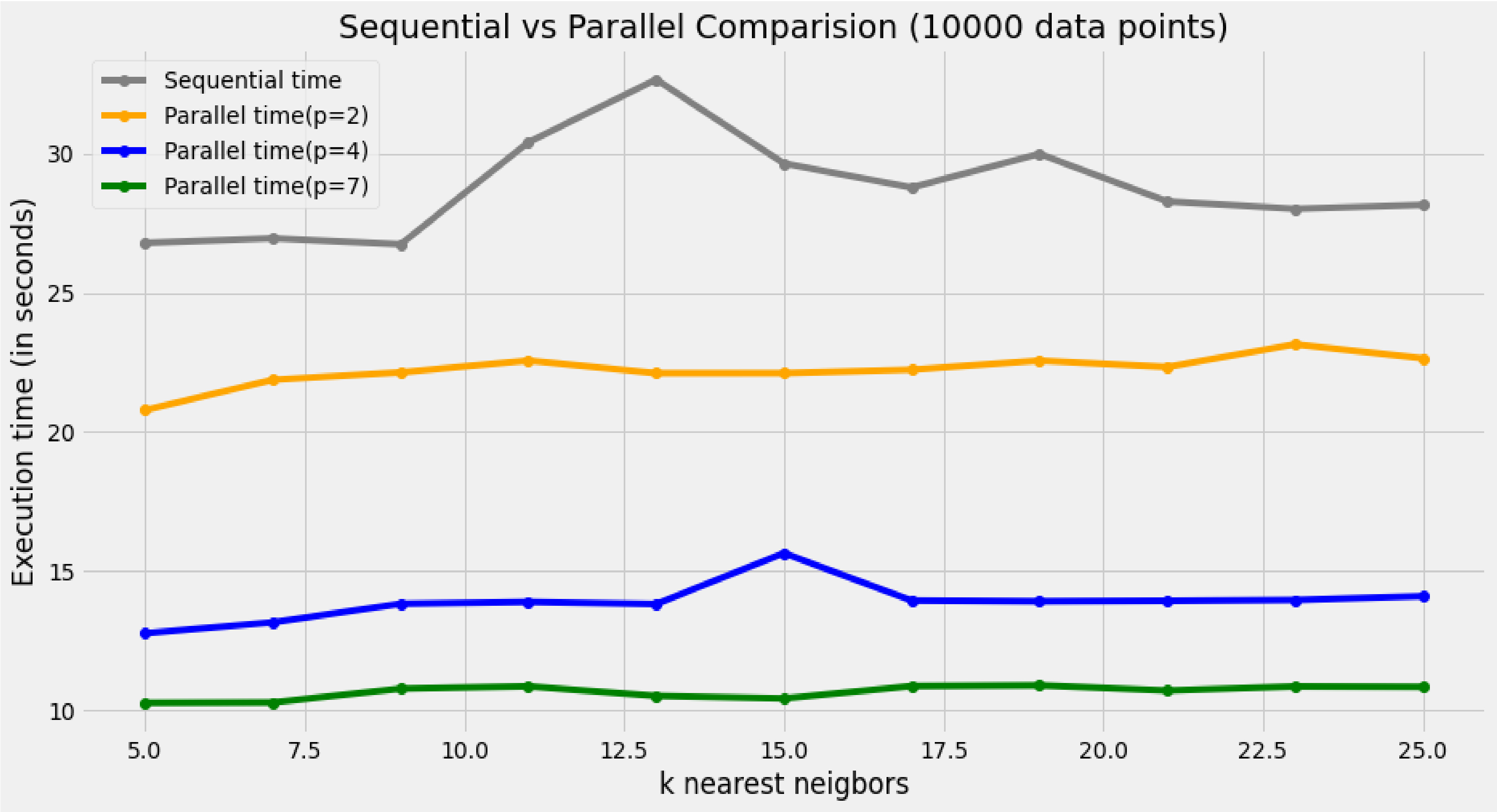
Scenario 1: K = 5, training dataset sizes (ranging from 1000 to over 200k records) and multiprocessor counts were examined(Processor counts=2,4 and 7).



	No. data	Sequential time	Parallel time(p=2)	Parallel time(p=4)	Parallel time(p=7)	Speedup(k=5)	Efficiency(p=2)	Efficiency(p=4)	Efficiency(p=7)
0	1000	3.605	2.100	1.303	1.036	1.717	0.858	0.692	0.497
1	2000	7.233	4.012	2.483	1.930	1.803	0.901	0.728	0.535
2	4000	14.445	8.228	5.122	3.982	1.755	0.878	0.705	0.518
3	6000	21.846	12.459	7.539	5.804	1.753	0.877	0.724	0.538
4	8000	29.333	17.068	10.363	7.731	1.719	0.859	0.708	0.542
5	12000	43.783	25.853	15.793	12.125	1.694	0.847	0.693	0.516
6	16000	59.085	33.964	21.430	16.205	1.740	0.870	0.689	0.521
7	50000	187.378	107.869	68.620	54.896	1.737	0.869	0.683	0.488
8	100000	213.089	123.710	77.422	59.254	1.722	0.861	0.688	0.514
9	200000	211.454	125.240	76.763	65.716	1.688	0.844	0.689	0.460
10	227844	210.906	124.238	77.874	64.293	1.698	0.849	0.677	0.469

- We see noticeable performance difference when Data records increase in size.
- Parallel KNN outperforms Sequential KNN when data records size increases.

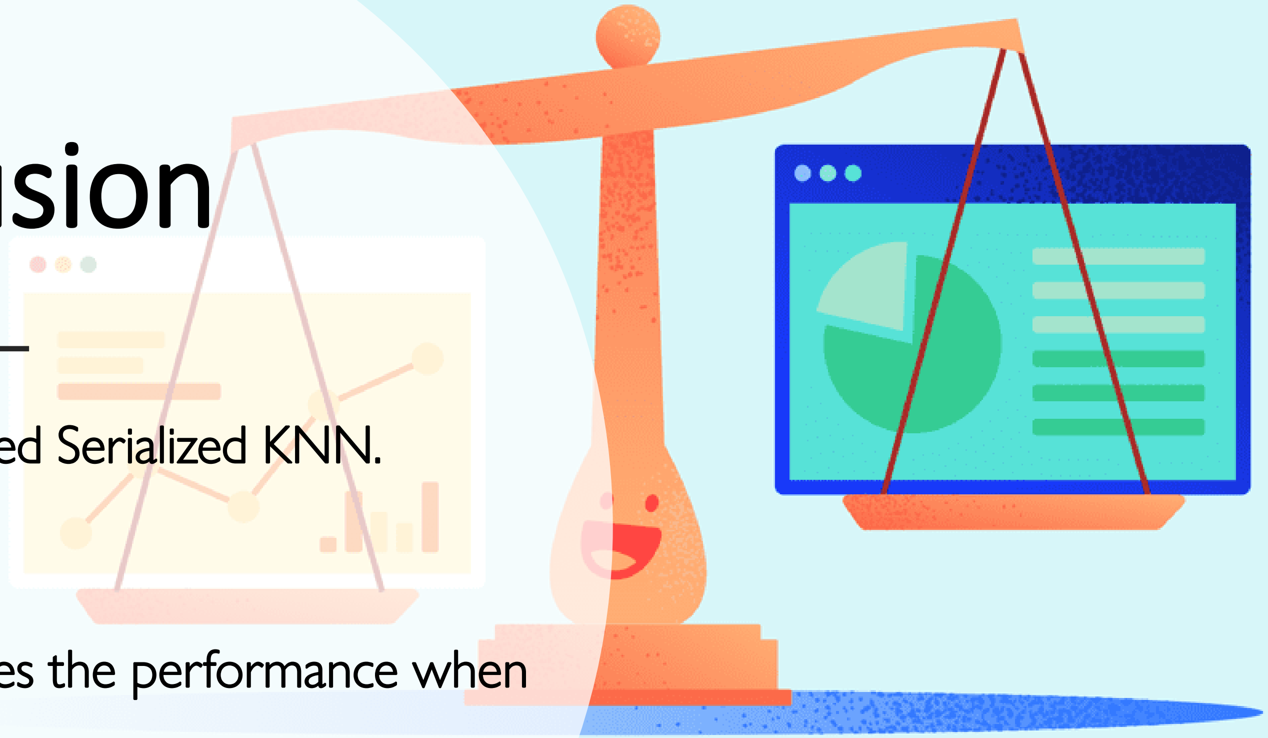
Scenario 2: Training dataset of 10,000 records, examine with **various values of k from 5 to 25 nearest neighbors**, using various **multiprocessor counts(Processor counts=2, 4 and 7).**



	Value k	Sequential time	Parallel time(p=2)	Parallel time(p=4)	Parallel time(p=7)	Speedup(10000 records)	Efficiency(p=2)	Efficiency(p=4)	Efficiency(p=7)
0	5	26.790	20.781	12.761	10.262	1.289	0.645	0.525	0.373
1	7	26.955	21.870	13.155	10.275	1.232	0.616	0.512	0.375
2	9	26.739	22.137	13.820	10.770	1.208	0.604	0.484	0.355
3	11	30.407	22.561	13.886	10.852	1.348	0.674	0.547	0.400
4	13	32.662	22.109	13.811	10.511	1.477	0.739	0.591	0.444
5	15	29.645	22.110	15.641	10.417	1.341	0.670	0.474	0.407
6	17	28.783	22.233	13.937	10.862	1.295	0.647	0.516	0.379
7	19	29.987	22.560	13.910	10.887	1.329	0.665	0.539	0.393
8	21	28.283	22.329	13.928	10.703	1.267	0.633	0.508	0.377
9	23	28.014	23.145	13.958	10.847	1.210	0.605	0.502	0.369
10	25	28.152	22.644	14.093	10.833	1.243	0.622	0.499	0.371

- We see noticeable performance difference as value of K increases.
- Parallel KNN is performing way better than Sequential KNN when computational complexity increases.

Conclusion



- Parallel KNN outperformed Serialized KNN.
- Analysis:
  - Parallel KNN improves the performance when exposed to good amount of data and computational complexity.

Discussion Questions:

1. Did you get the idea how K nearest neighbor work?
2. Why we need Parallel Processing?
3. What is Efficiency and Speedup metrics?

References

1. Reynaldo Gil-Garc´ıa, Jos´e Manuel Bad´ıa-Contelles, and Aurora Pons-Porrata. Parallel nearest neighbour algorithms for text categorization. In Anne-Marie Kermarrec, Luc Boug´e, and Thierry Priol, editors, Euro-Par 2007 Parallel Processing, pages 328–337,Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.
2. Parallel k-nearest neighbor. https://alitarhini.wordpress.com/2011/02/26/parallel-k-nearest-neighbor/
3. Credit card fraud detection. https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud.
4. How to evaluate the performance of a parallel program. https://subscription.packtpub.com/book/application-development/9781785289583/1/ch01lvl1sec14/how-to-evaluate-the-performance-of-a-parallel-program/
5. https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning
6. https://hackerbits.com/data/k-nearest-neighbor-knn-data-mining-algorithm/
7. https://blog.saleslayer.com/differences-between-pim-erp-systems
8. https://www.statista.com/statistics/871513/worldwide-data-created/
9. https://blog.inkjetwholesale.com.au/marketing-advertising/6-effective-social-media-marketing-experiments-that-you-can-try/
10. https://www.pngitem.com/middle/TohwiJw_pre-qualification-icon-png-transparent-png/
11. https://dataschools.education/resource/useful-datasets-for-data-education-in-schools/
12. https://www.searchenginejournal.com/google-serp-study-which-rich-results-get-the-most-clicks/382445/

Thank you!