ECE 20875

Python for Data Science

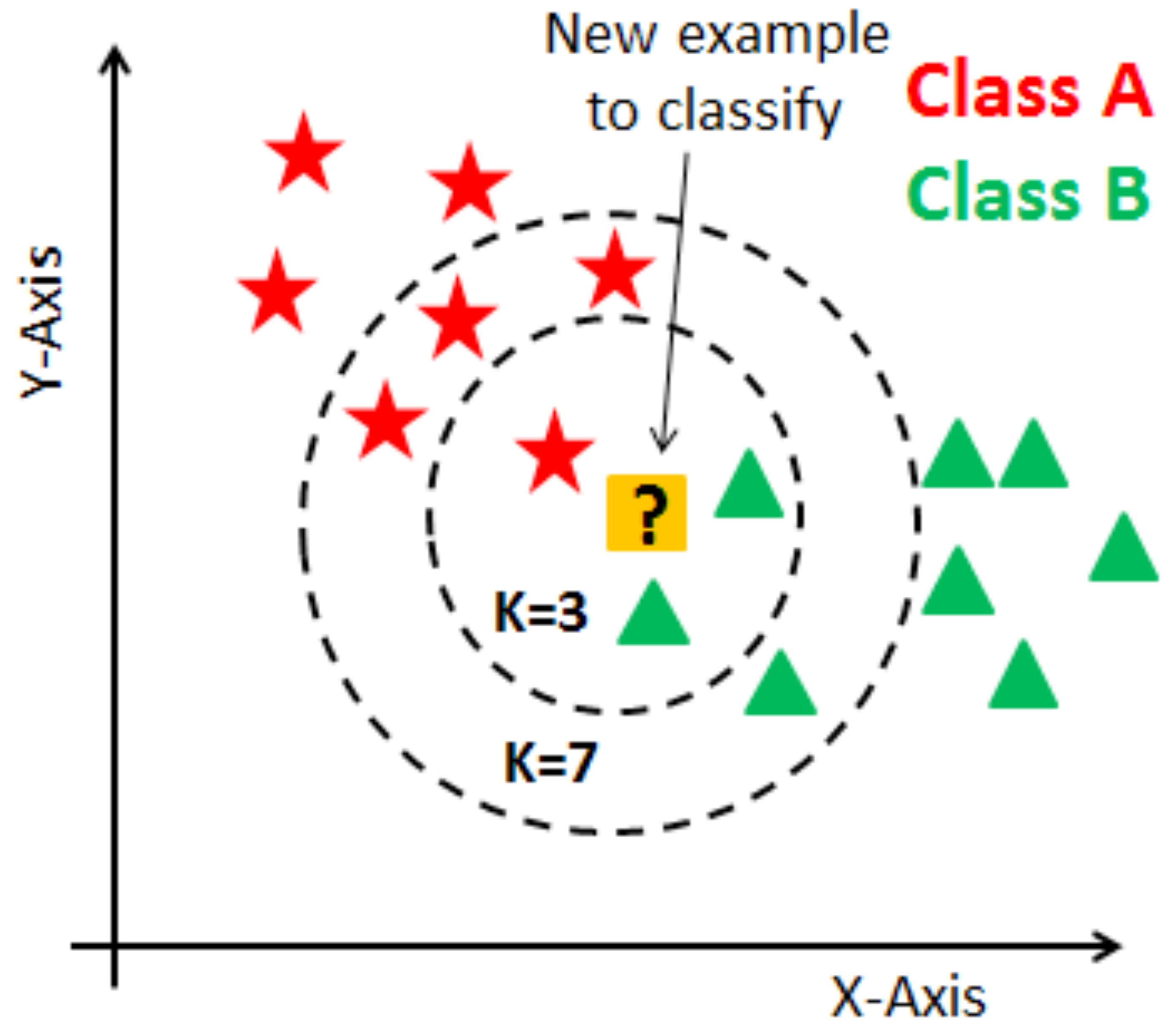
Chris Brinton and Qiang Qiu

(Adapted from material developed by Profs. Milind Kulkarni, Stanley Chan, Chris Brinton, David Inouye)

classification: k-nearest neighbor

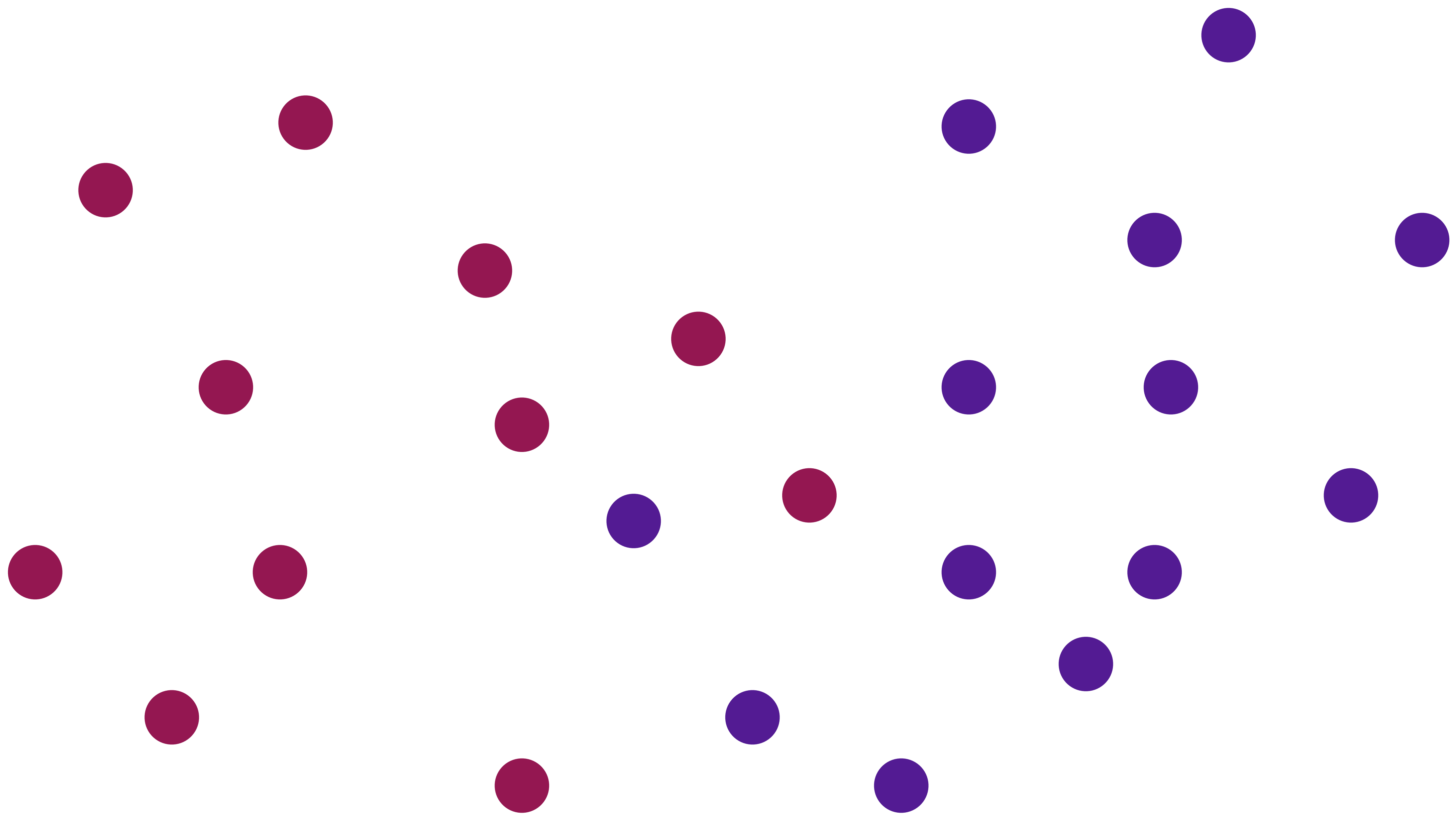
k-nearest neighbor

- Naïve Bayes is a nice classifier, but it is *parametric*
  - We must have a model of the data in mind, and some prior knowledge, to use it effectively.
- What if we don't have any such knowledge? What if all we have is our input data, and it does not seem to fit any existing distribution well?
- **k-nearest neighbor** (**kNN**) is a classifier that requires no assumptions about the data:
  - Look at the classes of the k-nearest points and pick the most frequent one



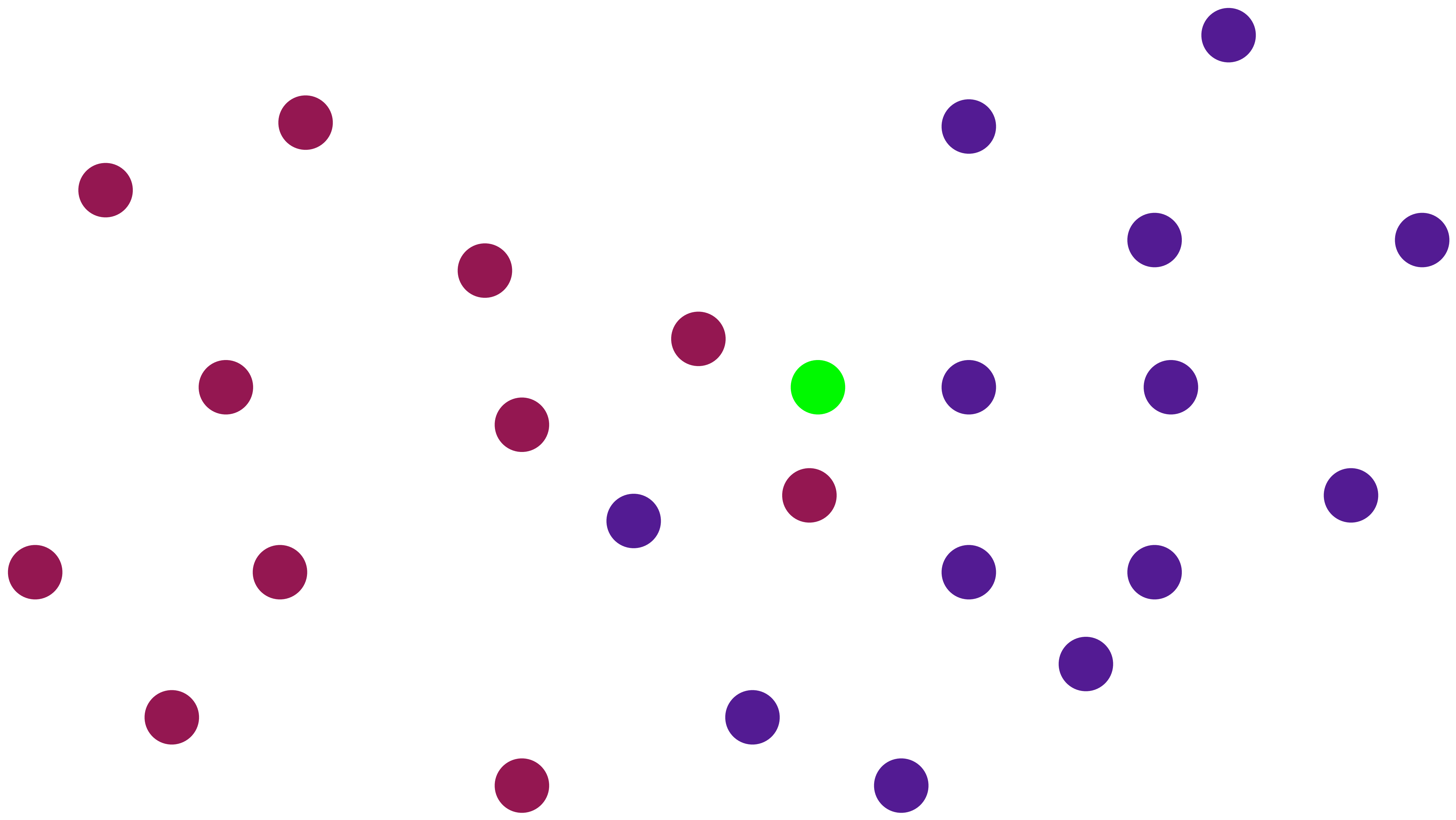
kNN algorithm

- Start with labeled training data, just like naïve Bayes



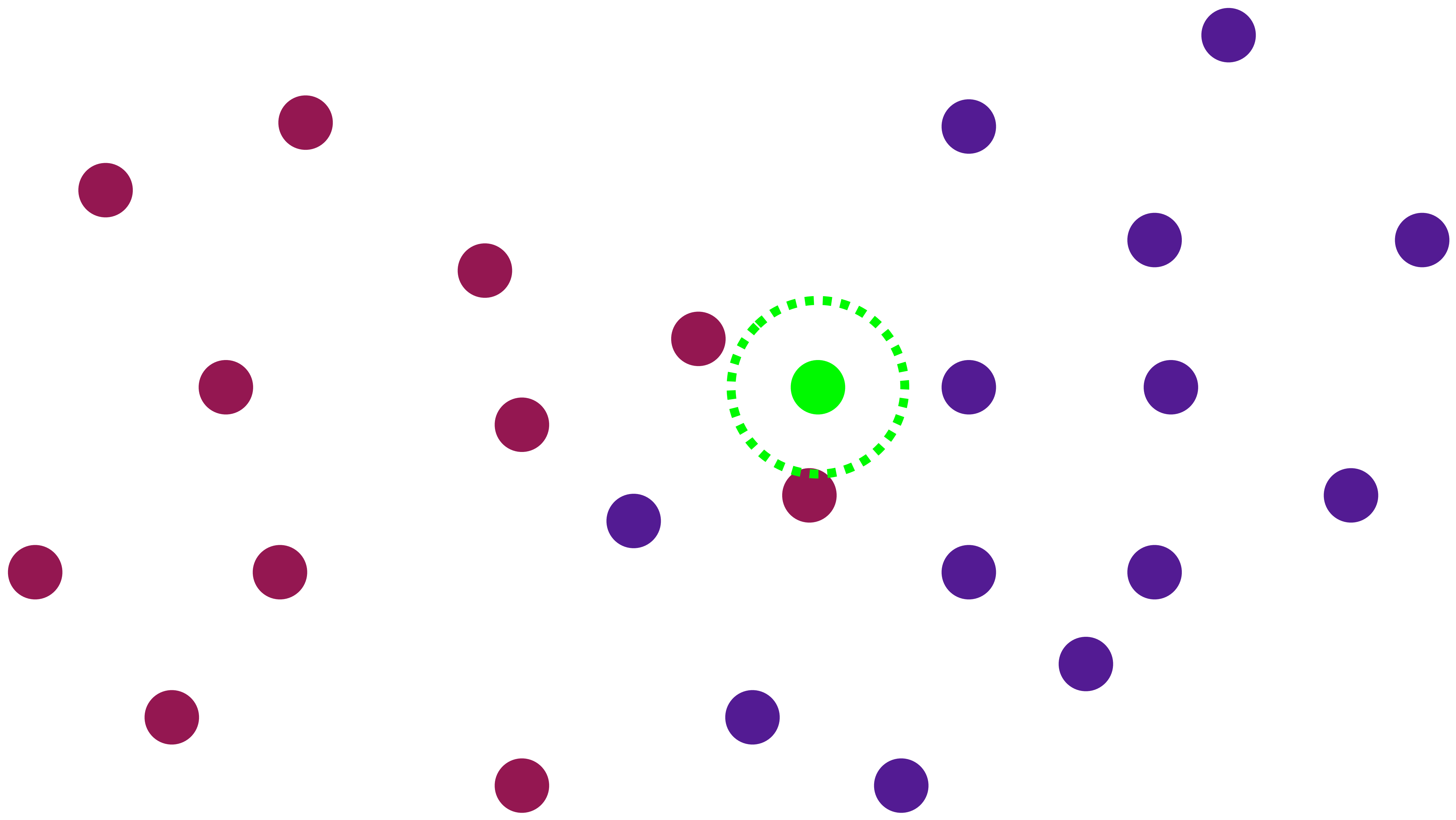
kNN algorithm

- Take new data point



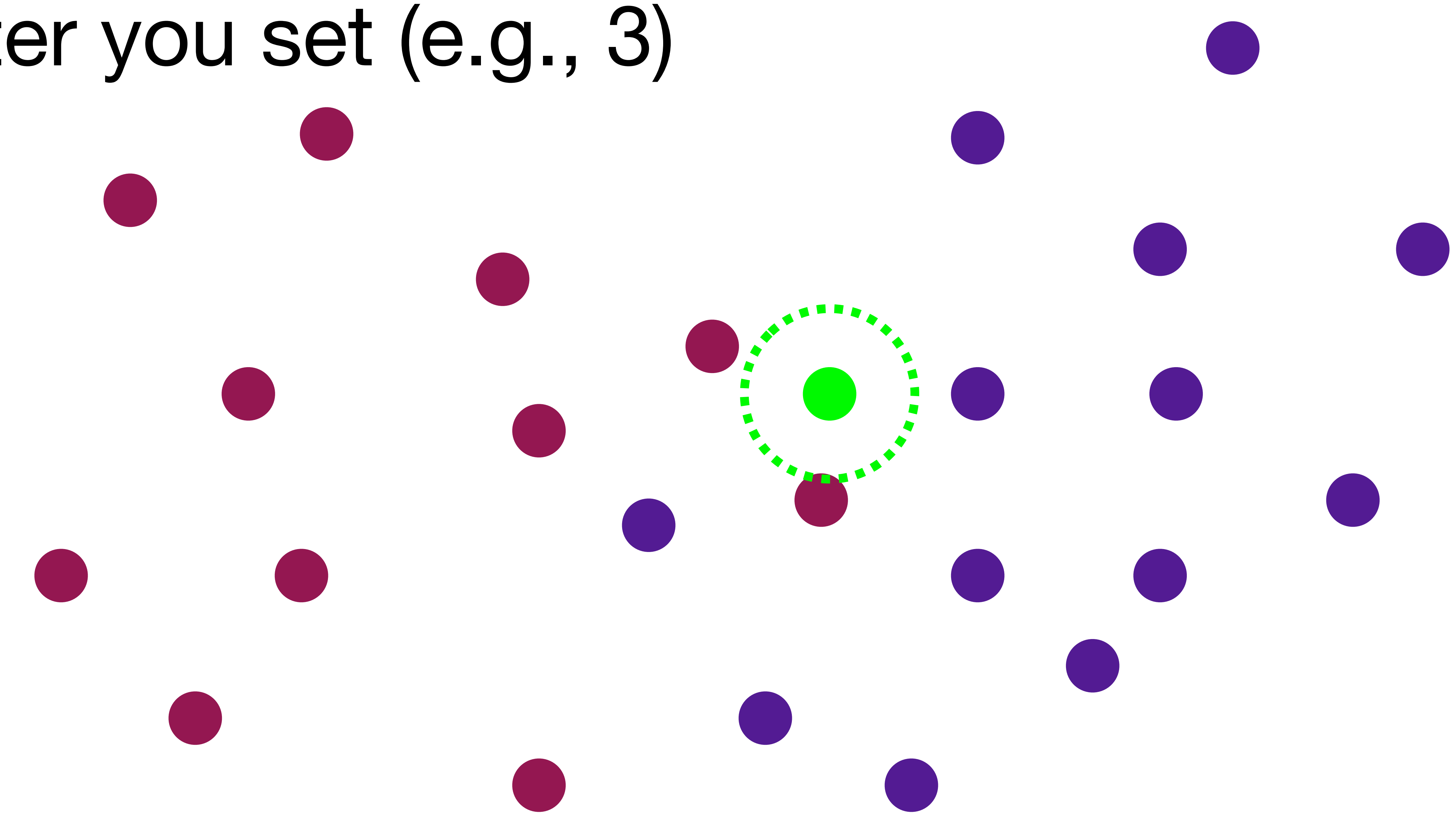
kNN algorithm

- Draw a circle around it



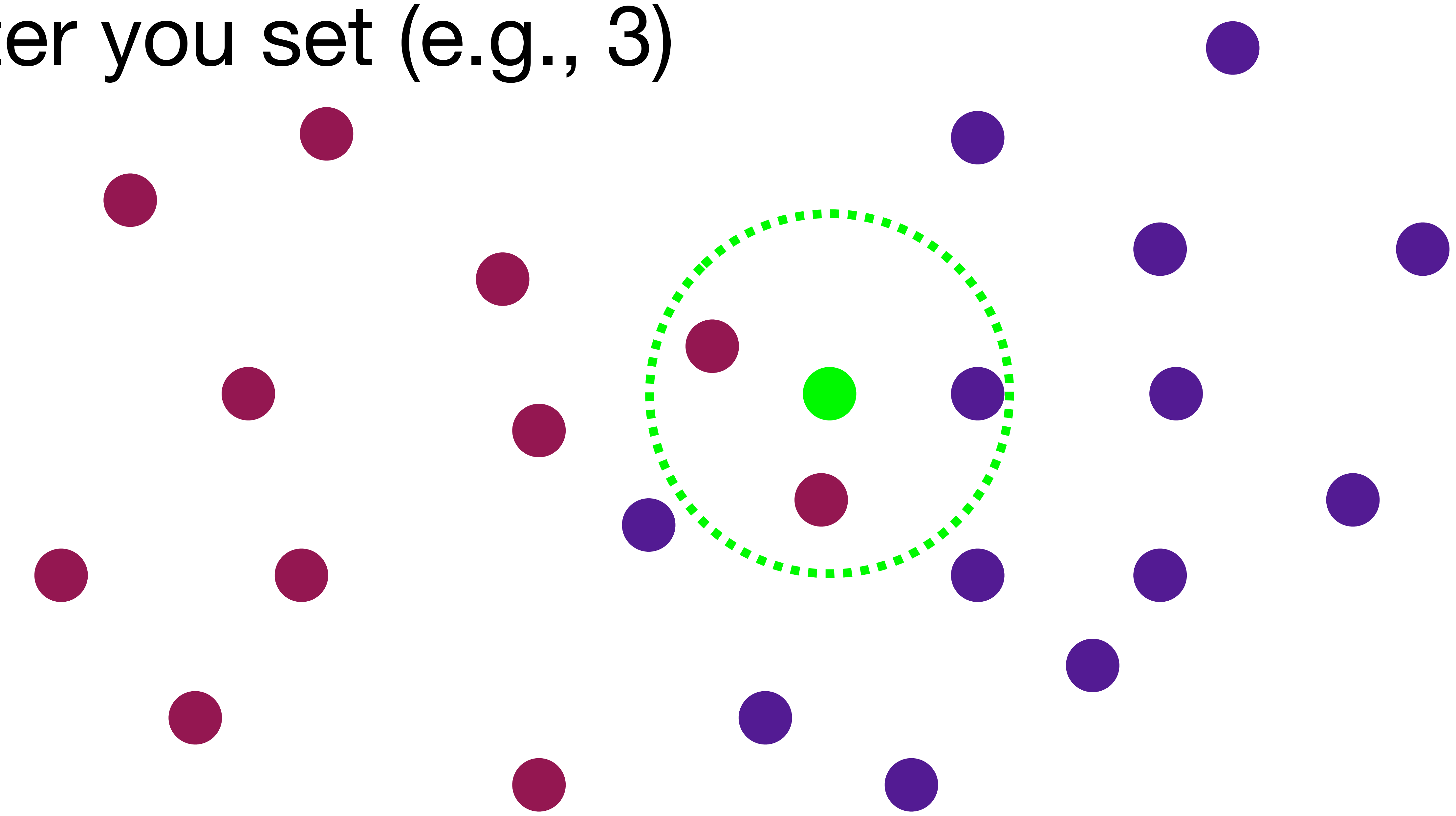
kNN algorithm

- Grow the circle until it has *k* other points in it
  - *k* is a parameter you set (e.g., 3)



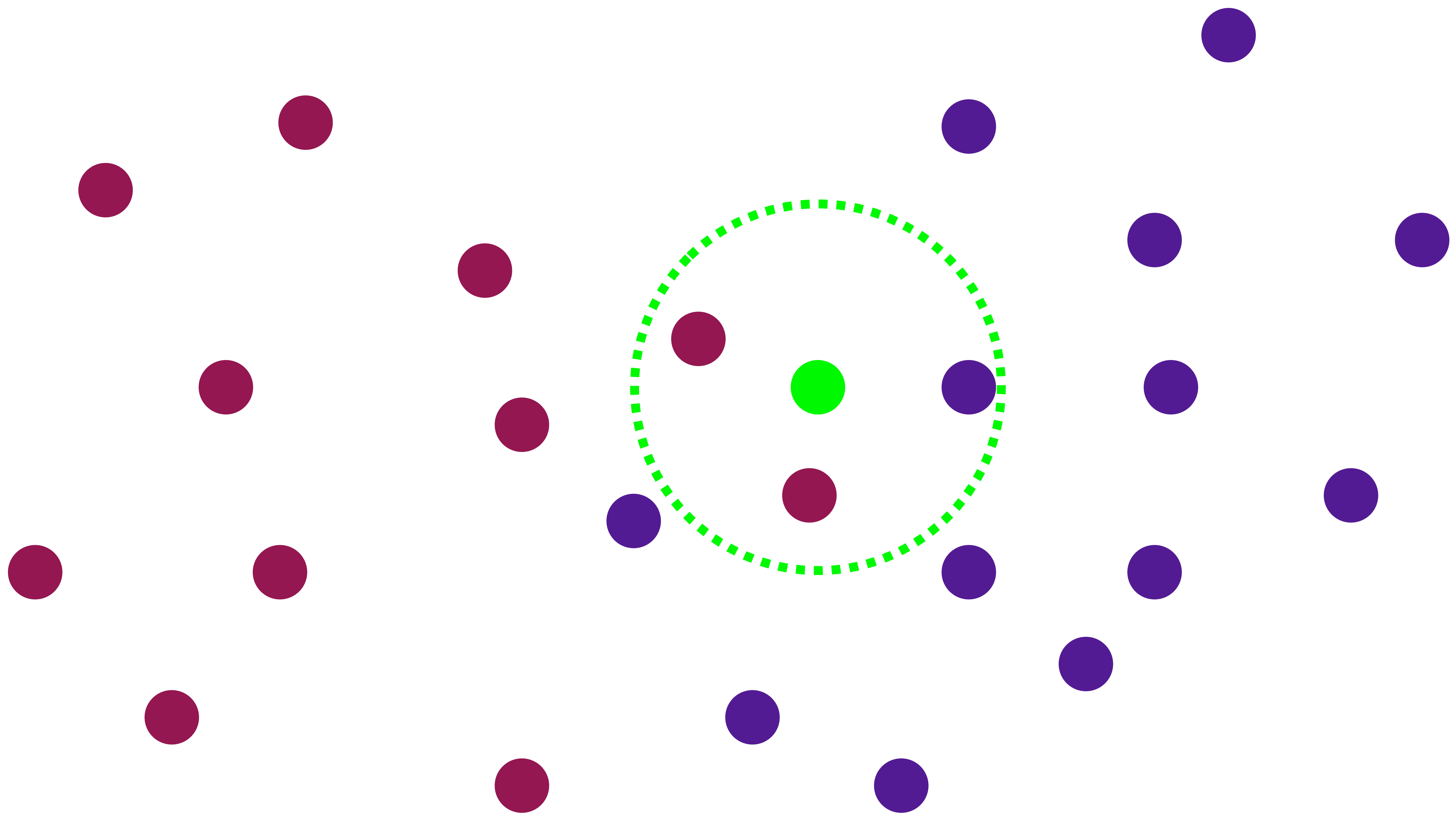
kNN algorithm

- Grow the circle until it has *k* other points in it
    - *k* is a parameter you set (e.g., 3)



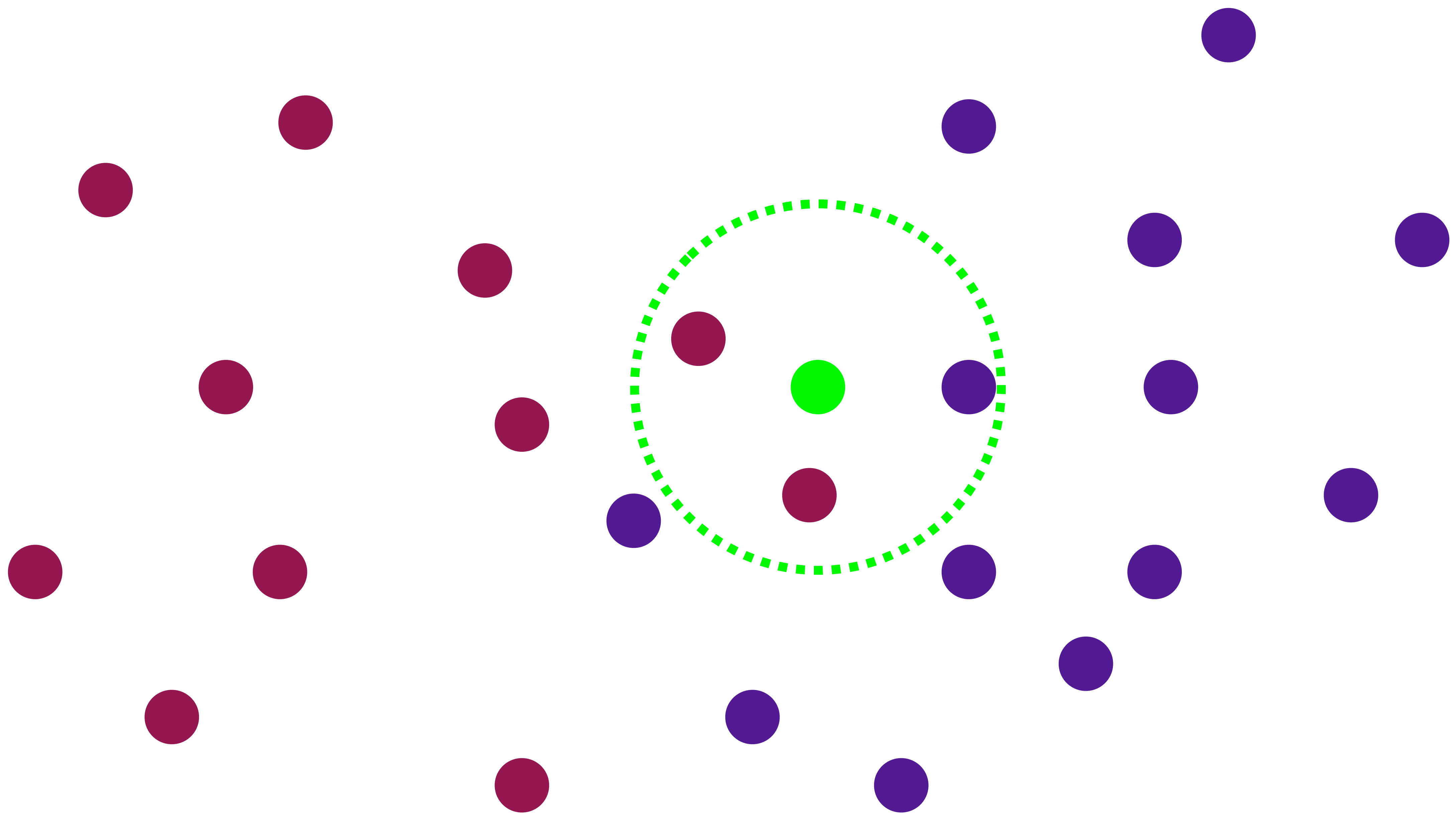
kNN algorithm

- Count how many points from **class 1** are in the circle and how many from **class 2**
  - **Majority wins**



kNN algorithm

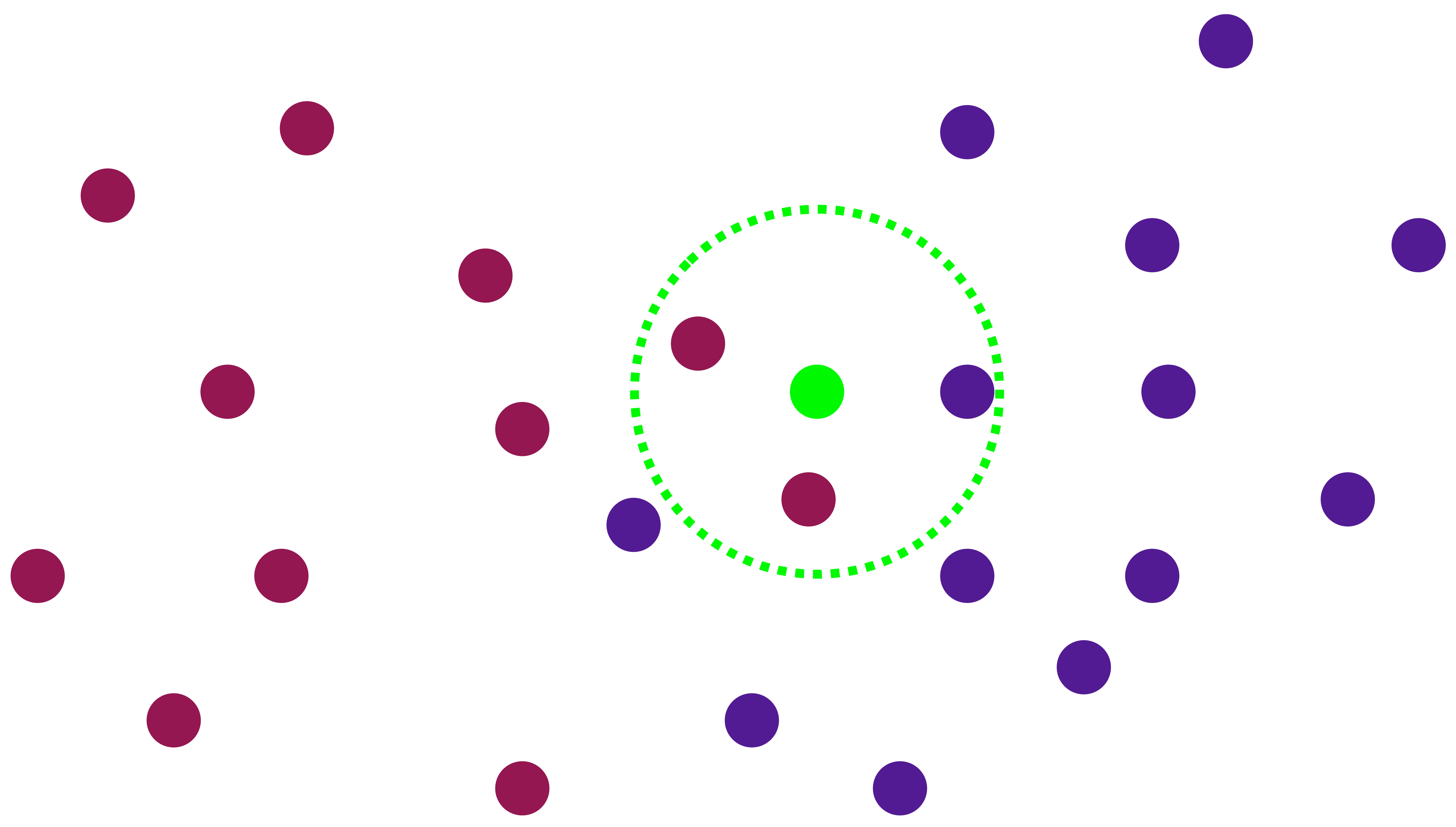
- Count how many points from **class 1** are in the circle and how many from **class 2**
    - **Majority wins**



- How to choose k?
    - Larger k means we are less sensitive to outliers, but also less sensitive to possibly informative (very near) neighbors
    - Cross validation!

formal algorithm and python

- Algorithmic interpretation:
  - Find the distance $||x - x_0||$ from **new point** x_0 to every other point x
  - Sort by distance, pick closest k points
  - Predicted class is the one with the most “votes” from these k
- In Python
  - `from sklearn.neighbors import KNeighborsClassifier`
  - https://scikit-learn.org/stable/modules/classes.html#module-sklearn.neighbors



pros vs cons

+ Simple concept for classifier

+ No models or prior knowledge required

- Expensive to use model (compute distances from all other points)

- Does not help with missing data (classifier is only as good as labeled training data)

- The intuition and usefulness can breakdown in high dimensions (what does it mean to “near” in 1000 dimensions?)

binary evaluation metrics

- With regression, we used MSE (and r^2) as our evaluation metrics
- In classification, these are not valid, because our predictions are either right or wrong
- For binary classification, we typically report several metrics (on a test set), based on a **confusion matrix** (shown to the right).
  The most common three are:
  - **Accuracy**: Fraction of correct predictions
  - **Precision**: Fraction of correct predictions in the predicted positive class
  - **Recall** (or **sensitivity**): Fraction of correct predictions in the actual positive class

see `sklearn.metrics` in Python

		Predicted Class: Positive	Predicted Class: Negative	
Actual Class	Positive	True Positive (TP)	False Negative (FN) Type II Error	Sensitivity $\frac{TP}{(TP+FN)}$
	Negative	False Positive (FP) Type I Error	True Negative (TN)	Specificity $\frac{TN}{(TN+FP)}$
		Precision $\frac{TP}{(TP+FP)}$	Negative Predictive Value $\frac{TN}{(TN+FN)}$	Accuracy $\frac{TP+TN}{(TP+TN+FP+FN)}$

composite binary metrics

- In regression problems, MSE is convenient: Single number that indicates quality
- With classification problems, none of these confusion table metrics tell the whole story:
  - If there is significant **class imbalance**, accuracy can look very good even if the classifier is not
  - For example, suppose 90% of cars are minivans and 10% are sports cars. If we always predict minivan, we will have 90% accuracy!
- There are two composite metrics that can be useful:
  - **F1 score**: Harmonic mean between precision and recall (both need to be high for the F1 score to be high)
  - **AUROC**: Area under true/false positive curve from varying decision threshold from 0 (predict all negatives) and 1 (predict all positives)