Gotchas!

By now, you're already familiar with the highlights of K-Neighbors. It's easy to apply and make sense of because it mimics our own understanding of reality. Even its implementation is straightforward, no curve balls or tricks to it. Unlike other algorithms, training is immediate because it simply stores all of your training data. This has the added benefit that if you were to come into contact with more labeled data in the future, you could introduce it into your model without having to do heavy computations to rebuild it. Because of this, K-Neighbors is sometimes referred to as a memory based, 'lazy' machine learning algorithm, as it delays doing calculations until you start classifying.

K-Neighbors being the first supervised learning model you've encountered in this course, you might not be aware of how the other models behave. But keep in the back of your mind that K-Neighbors is particularly useful when no other model fits your data well, as it is a parameter free approach to classification. So for example, you don't have to worry about things like your data being linearly separable or not.

Some of the caution-points to keep in mind while using K-Neighbors is that your data needs to be measurable. If there is no metric for discerning distance between your features, K-Neighbors cannot help you. As with all algorithms dependent on distance measures, it is also sensitive to feature scaling. K-Neighbors is also sensitive to perturbations and the local structure of your dataset, particularly at lower "K" values.

Lab	On the other hand, with large "K" values, you have to be more cautious of the overall class distribution of your samples. If 30% of your dataset is labeled **A** and 70% of labeled **B**, with high enough "K" values, you might experience K-Neighbors unjustly giving preference to **B** labeling, even in those localities of your dataset that should be properly classified as **A**.