**K- Means**

* **Unsupervised Learning**

**Unsupervised Learning**

Up until now we only looked at models that require training data in the form of features and labels. In other words, for each example, we need to have the correct answer, too.

Usually, such training data is hard to obtain and requires many hours of manual work done by humans (yes, we’re already serving “The Terminators). Can we skip all that?

Yes, at least for some problems we can use example data without knowing the correct answer for it. One such problem is *clustering*.

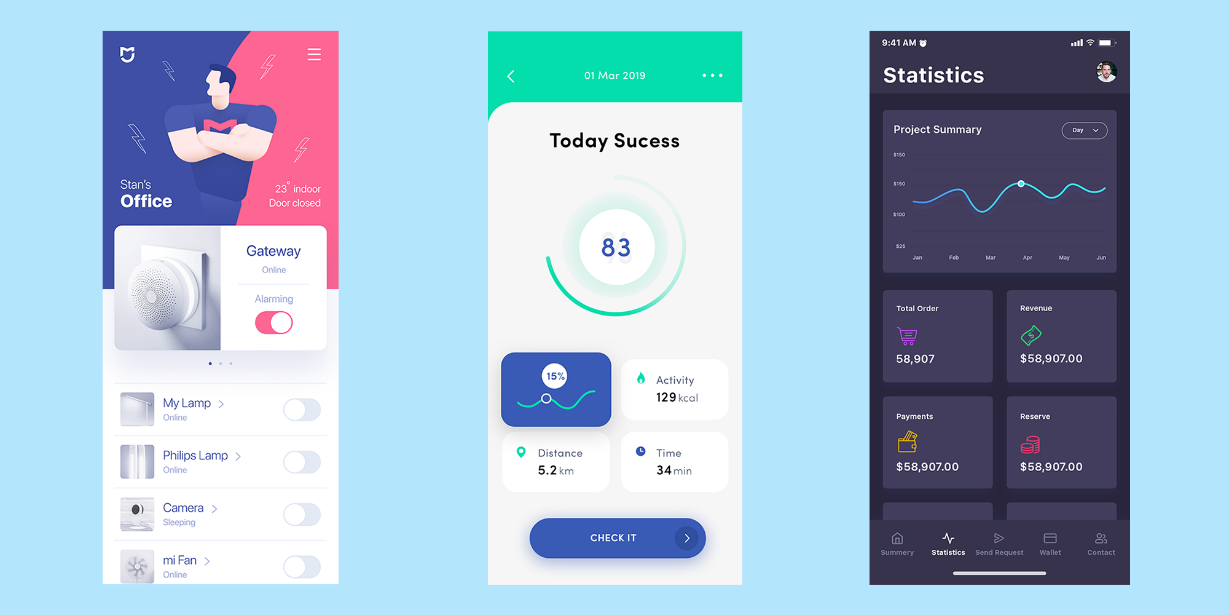
**What is clustering?**

Given some vector of data points *X*, clustering methods allow you to put each point in a group. In other words, you can categorize a set of entities, based on their properties, automatically.

Yes, that is very useful in practice. Usually, you run the algorithm on a bunch of data points and specify how much groups you want. For example, your inbox contains two main groups of e-mail: spam and not-spam (were you waiting for something else?). You can let the clustering algorithm create two groups from your emails and use your beautiful brain to classify which is which.

More applications of clustering algorithms:

* *Customer segmentation* — find groups of users that spend/behave the same way
* *Fraudulent transactions* — find bank transactions that belong to different clusters and identify them as fraudulent
* *Document analysis* — group similar documents



*source: various authors on*[*https://www.uplabs.com/*](https://www.uplabs.com/)

This time, our data doesn’t come from some predefined or well-known dataset. Since Unsupervised learning does not require labeled data, the Internet can be your oyster.

Here, we’ll use 3 mobile UI designs from various authors. Our model will run on each shot and try to extract the color palette for each.

**What is K-Means Clustering?**

K-Means Clustering is defined by [Wikipedia](https://en.wikipedia.org/wiki/K-means_clustering) as:

k-means clustering is a method of vector quantization, originally from signal processing, that is popular for cluster analysis in data mining. k-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. This results in a partitioning of the data space into Voronoi cells.

Wikipedia also tells us that solving K-Means clustering is difficult (in fact, [NP-hard](https://en.wikipedia.org/wiki/NP-hardness)) but we can find local optimum solutions using some heuristics.

But how do *K-Means Clustering* works?

Let’s say you have some vector *X* that contains data points. Running our algorithm consists of the following steps:

1. Take random *k* points (called *centroids*) from *X*
2. Assign every point to the closest centroid. The newly formed bunch of points is called *cluster*.
3. For each cluster, find new centroid by calculating a new center from the points
4. Repeat steps 2–3 until centroids stop changing

Let’s see how can we use it to extract color palettes from mobile UI screenshots.

**Data Preprocessing**

Given our data is stored in raw pixels (called images), we need a way to convert it to points that our clustering algorithm can use.

Let’s first define two classes that represent a point and cluster:

Our Point is just a holder to the coordinates for each dimension in our space.

The Cluster is defined by its center and all other points it contains.

Given a path to image file we can create the points as follows:

A couple of things are happening here:

* load the image into memory
* resize it to smaller image (mobile UX requires big elements on the screen, so we aren’t losing much color information)
* drop the alpha (transparency) information

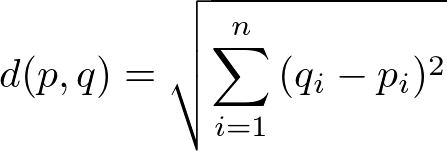
Note that we’re creating a Point for each pixel in our image.

Alright! You can now extract points from an image. But how can we calculate the distance between points in our clusters?

**Distance function**

Similar to the cost function in our supervised algorithm examples, we need a function that tells us how well we’re doing. The objective of our algorithm is to minimize the distance between the points in each centroid.

One of the simplest distance functions we can use is the Euclidean distance, defined by:



where and are two points from our space.

Note that while Euclidean distance is simple to implement it might not be the best way to calculate the [color difference](https://en.wikipedia.org/wiki/Color_difference).

Here is the Python implementation:

**Implementing K-Means clustering**

Now that you have all pieces of the puzzle you can implement the K-Means clustering algorithm. Let’s start with the method that finds the center for a set of points:

To find the center of a set of points, we add the values for each dimension and divide by the number of points.

Now for finding the actual clusters:

The implementation follows the description of the algorithm given above. Note that we exit the training loop when the color difference is lower than the one set by us.

**Evaluation**

Now that you have an implementation of K-Means clustering you can use it on the UI screenshots. We need a little more glue code to make it easier to extract color palettes:

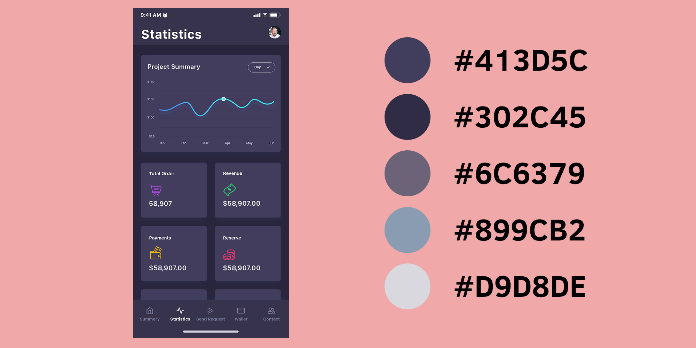
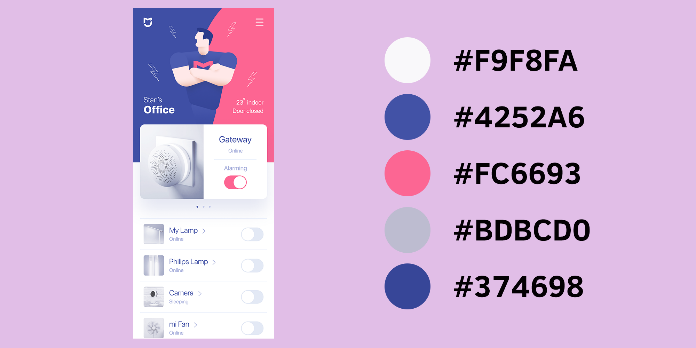
The get\_colors function takes a path to an image file and the number of colors you want to extract from the image. We sort the clusters obtained from our algorithm based on the points in each (in descending order). Finally, we convert the RGB colors to hexadecimal values.

Let’s extract the palette for the first UI screenshot from our data:

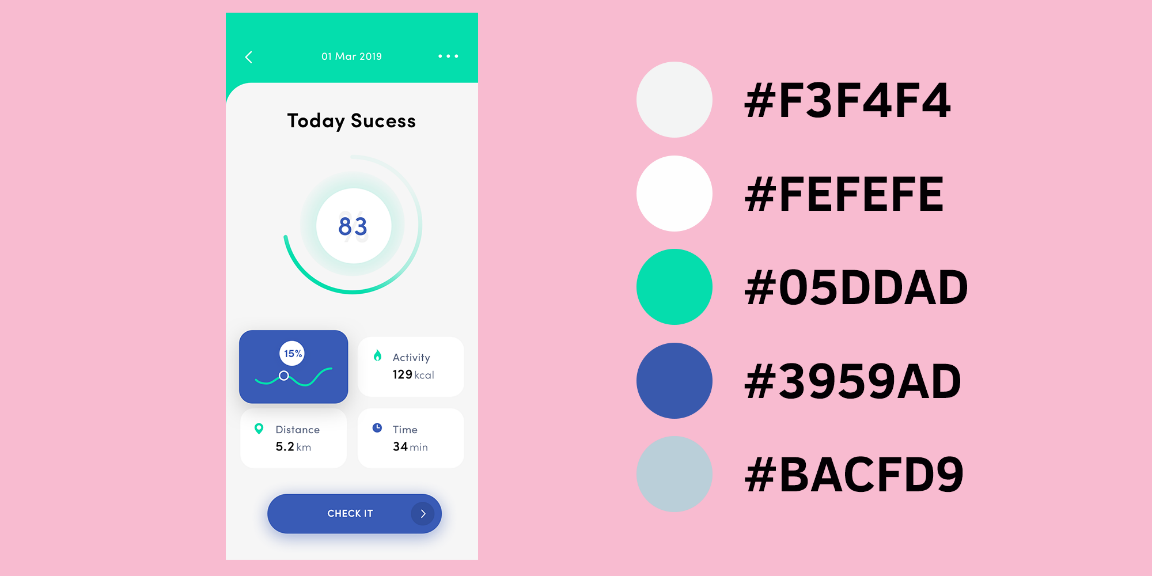
and here are the hexadecimal color values:

#f9f8fa, #4252a6, #fc6693, #bdbcd0, #374698

Running the clustering on the next two images, we obtain the following:



The last screenshot:



Well, that looks cool, right? Go on try it on your own images!

**Conclusion**

Congratulations, you just implemented your first unsupervised algorithm from scratch! It also appears that it obtains good results when trying to extract a color palette from images.