Project summary for kmeanseval

# Background and motivation

K-means is the most common unsupervised learning algorithm used for clustering today. When clustering data using k-means, one of the most important decisions to make is how many clusters to use ("k"). This is a somewhat subjective decision, representing a tradeoff between how simply the clusters summarize the data (small k) vs. how well the clusters represent the data (large k). Picking a k value is typically done by comparing within-cluster-sum of squared errors ("WSS") and/or silhouette scores ("SS") at different values of k. This process often involves creating a chart to visually inspect how WSS and/or SS change at different values of k. This means that every k-means analysis involves some boilerplate code to loop through the creation of multiple k-means models at different values of k, calculate WSS or SS for them, and plot these results.

# Project idea

The goal of this package is to reduce the amount of boilerplate code that goes into selecting a value for k when using k-means. Rather than creating loops to capture WSS and SS and plotting them from scratch, this can all be handled by simple functions that wrap around the scikit-learn k-means interface.

# Design considerations

The overarching goal for the design of kmeanseval is to create a simple, lightweight package that can satisfy the majority of use cases for selecting k in a k-means analysis. Most of the design considerations in this project relate to the tradeoff between simplicity and functionality.

* kmeanseval will only support k-means clustering. This does limit the usefulness of the package, however k-means is by far the most popular clustering method. Other clustering methods like GMM, DBSCAN, etc. would add complexity to the code and the interface while only serving more niche use cases.
* kmeanseval will only support within-cluster-sum of squared errors and silhouette scores as evaluation metrics. Similar to the decision to only support k-means clustering, this simplifies the scope of the project while still providing the needed functionality for the most common use cases.
  + However, one of the design goals for this project is to write code with enough abstraction that additional evaluation metrics for k-means could be added relatively easily at a later point.
* kmeanseval wraps around the scikit-learn k-means functions. This will allow any user familiar with using the k-means interface in scikit-learn to easily pick up the interface for kmeanseval.
* Plots for WSS and SS in kmeanseval will be kept simple and not allow for much user customization. Users can get the list of WSS/SS values to make their own more complex plots if desired.

# Usage

The example workflows below both get a list of WSS for k = 1 through 10 and plot them. When using kmeanseval there are fewer lines of boilerplate code.

**Workflow using kmeanseval:**

from kmeanseval import getmetrics, plotmetrics

import pandas as pd

data = pd.read\_csv('data.csv', sep=',')

wss = getmetrics(data, method = 'wss', range = [1, 10])

plotmetrics(data, method = 'wss', range = [1, 10])

**Workflow using scikit-learn and matplotlib:**

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.cluster import KMeans

data = pd.read\_csv('data.csv', sep=',')

wss = []

K = range(1, 10)

for k in K:

model = KMeans(n\_clusters=k)

model.fit(data)

wss.append(model.inertia\_)

plt.figure(figsize=(20,10))

plt.plot(K, wss)

plt.xlabel('k')

plt.ylabel('Within-cluster-sum of squares')

plt.title('WSS for K = 1 through 10')

plt.show()

# Comparison: clusteval

Link to clusteval: <https://pypi.org/project/clusteval/>

clusteval as described on PyPi: *"clusteval is a Python package for unsupervised cluster evaluation. Three evaluation methods are implemented that can be used to evalute clusterings; silhouette, dbindex, and derivative. Four clustering methods can be used: agglomerative, kmeans, dbscan and hdbscan."*

Similar to kmeanseval, clusteval is a tool for calculating evaluation metrics in clustering. However, the two packages are functionally very different:

* clusteval supports k-means, agglomerative clustering, dbscan, and hdbscan while kmeanseval only supports k-means