K-Means Clustering Documentation

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import random
from mpl_toolkits.mplot3d import Axes3D
from preprocessing import NormalScaler

class KMeans:
    def __init__(self, X, k):
        self.assignments = pd.Series([random.randint(0,k-1) for i in range(X.shape[0])])
        self.centers = X.iloc[[random.randint(0,X.shape[0]-1) for i in range(k)]]
        self.k = k
        self.cost_arr = []

    def get_dist(self,a,b):
        return np.sum(np.square(a-b),axis=1)

    def get_cost(self, X):
        cost = 0
        for i in range(self.k):
            cost += np.sum(self.get_dist(X[self.assignments==i], self.centers.iloc[i]))
        self.cost_arr.append(cost)
        return cost
```

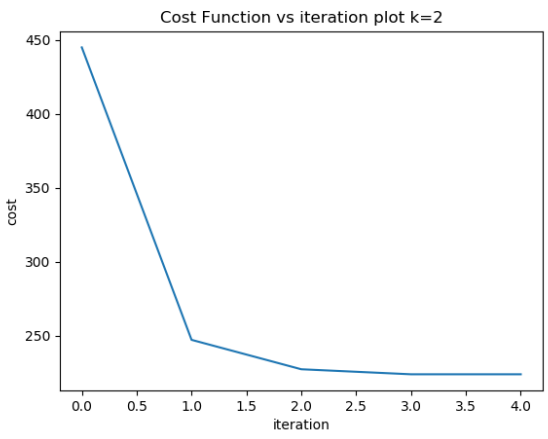
```
    def run(self, max_iter=100):
        for _ in range(max_iter):
            # assigning points to their nearest center
            for i in range(X.shape[0]):
                nearest_center_i = np.sum(np.square(X.iloc[i]-model.centers), axis=1).values.argmin()
                self.assignments[i] = nearest_center_i
            self.get_cost(X)
            # calculating mean of each cluster
            for i in range(self.k):
                self.centers.iloc[i] = np.mean(X[self.assignments==i])
```

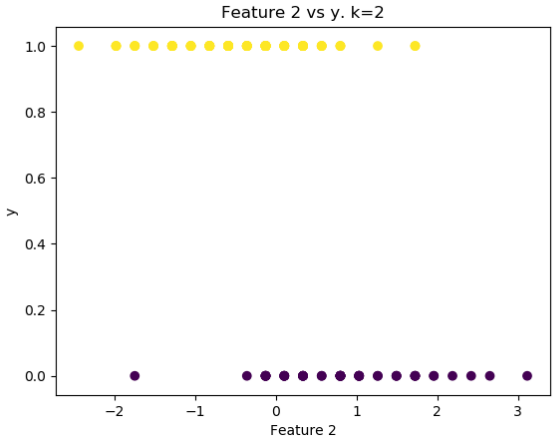
```
if __name__ == "__main__":
    # data input
    data = pd.read_excel("./data2.xlsx",header=None)
    X = data.copy()
    # data Normalization
    mscaler = NormalScaler()
    for j in range(X.shape[1]):
        mscaler.fit(X[j])
        X[j] = mscaler.transform(X[j])

    # initializing k-means with k value
    model = KMeans(X,2)
    # running k-means algorithm
    model.run(10)
    y = model.assignments
    plt.plot(model.cost_arr)
    plt.show()
    plt.scatter(X[0],y,c=model.assignments)
    plt.show()
    plt.scatter(X[1],y,c=model.assignments)
    plt.show()
    plt.scatter(X[2],y,c=model.assignments)
    plt.show()
    plt.scatter(X[3],y,c=model.assignments)
    plt.show()
```

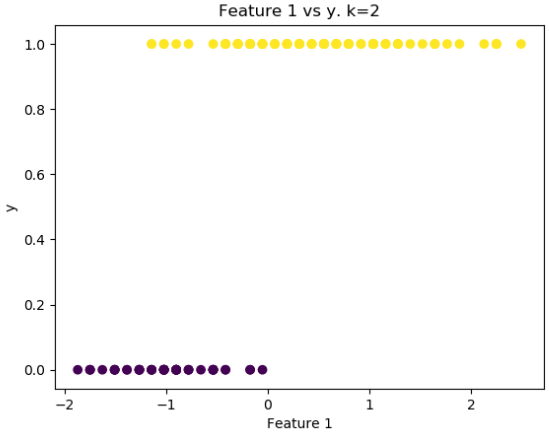
Results:

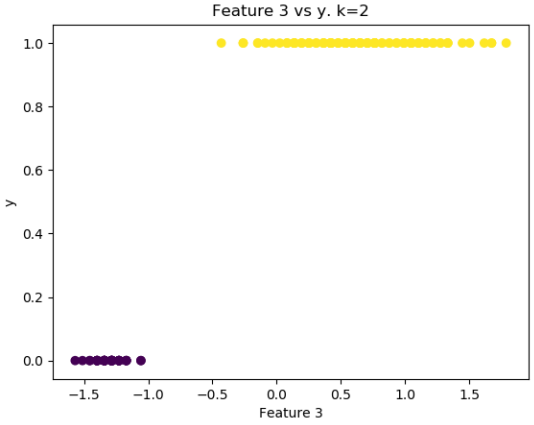
K = 2

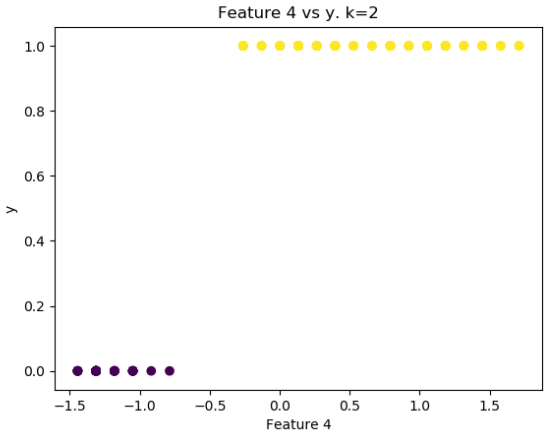
Total minimum cost = 223.7320057






Feature 1 vs y. k=2
y
Feature 1
1.0
0.8
0.6
0.4
0.2
0.0
−2
−1
0
1
2


Feature 3 vs y. k=2
y
Feature 3
1.0
0.8
0.6
0.4
0.2
0.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5


Feature 4 vs y. k=2
y
Feature 4
1.0
0.8
0.6
0.4
0.2
0.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5