*#Importing the Necessary Libraries*from sklearn.preprocessing import StandardScaler  
from sklearn.cluster import KMeans  
import pandas as pd  
import numpy as np  
from itertools import cycle, islice  
import matplotlib.pyplot as plt  
from pandas.plotting import parallel\_coordinates  
*#Creating a Pandas DataFrame from a CSV file*data = pd.read\_csv('C:/Users/Lenovo PC 16/Downloads/data/data/weather/minute\_weather.csv')  
print("Data Shape: ",data.shape)  
print("Sample Data: \n",data.head())  
*###Data Sampling #Lots of rows, so let us sample down by taking every 10th row.*sampled\_df = data[(data['rowID'] % 10) == 0]  
print("Sample Data Shape: ",sampled\_df.shape)  
*#Statistics*sampled\_df.describe().transpose()  
sampled\_df[sampled\_df['rain\_accumulation'] == 0].shape  
sampled\_df[sampled\_df['rain\_duration'] == 0].shape  
*#Drop all the Rows with Empty rain\_duration and rain\_accumulation*del sampled\_df['rain\_accumulation']  
del sampled\_df['rain\_duration']  
rows\_before = sampled\_df.shape[0]  
sampled\_df = sampled\_df.dropna()  
rows\_after = sampled\_df.shape[0]  
print("How many rows did we drop ? ",rows\_before - rows\_after)  
print("Columns in Sample Data: ",sampled\_df.columns)  
*#Select Features of Interest for Clustering*features = ['air\_pressure', 'air\_temp', 'avg\_wind\_direction', 'avg\_wind\_speed', 'max\_wind\_direction', 'max\_wind\_speed','relative\_humidity']  
select\_df = sampled\_df[features]  
select\_df.columns  
select\_df  
X = StandardScaler().fit\_transform(select\_df)  
print("X\n",X)  
*#Use k-Means Clustering*kmeans = KMeans(n\_clusters=12)  
model = kmeans.fit(X)  
print("model\n", model)  
*#What are the centers of 12 clusters we formed ?*centers = model.cluster\_centers\_  
print("centers: ", centers)  
*#Let us first create some utility functions which will help us in plotting graphs:*def pd\_centers(featuresUsed, centers):  
 colNames = list(featuresUsed)  
 colNames.append('prediction')  
 Z = [np.append(A, index) for index, A in enumerate(centers)]  
 centers\_df = pd.DataFrame(Z, columns=colNames)  
 centers\_df['prediction'] = centers\_df['prediction'].astype(int)  
  
 return centers\_df  
def parallel\_plot(data):  
 my\_colors = list(islice(cycle(['b', 'r', 'g', 'y', 'k']), None, len(data)))  
 plt.figure(figsize=(15,8)).gca().axes.set\_ylim([-3,+3])  
 parallel\_coordinates(data, 'prediction', color = my\_colors, marker='o')  
 plt.show()  
P = pd\_centers(features, centers)  
print("pd\_centers:\n",P)  
parallel\_plot(P)