**Computer Engineering Department National University of Technology Islamabad, Pakistan**

**Introduction to Data Mining**

**Practice Exercise 03**

****

Name: Muhammad Sami Uddin Rafay

Roll Number: F18604013

Submitted To: Dr. Kamran Javed

Date: 27 November 2020

**Practice Exercise 03**

**Understand and Analyse Weather Data.**

**Objective:**

* Feature extraction and selection from time series- temperature data for complete year (365).

**Equipment/Software Required:**

* Python (Spyder 4.0 Anaconda Distribution)

**Background:**

**Feature Extraction:**

Feature Extraction aims to reduce the number of features in a dataset by creating new features

from the existing ones (and then discarding the original features).

It is nowadays becoming quite common to be working with datasets of hundreds (or even

thousands) of features. If the number of features becomes similar (or even bigger!) than the

number of observations stored in a dataset then this can most likely lead to a Machine Learning

model suffering from overfitting. To avoid this type of problem, it is necessary to apply either

regularization or dimensionality reduction techniques (Feature Extraction).

* Feature Extraction techniques have many advantages such as:
* Accuracy improvements.
* Overfitting risk reduction.
* Speed up in training.
* Improved Data Visualization.
* Increase in explain-ability of our model.

**Tasks:**

* Generate hourly random temperature for each day between range 0 -50 C for 365 days.

(hint: import random.uniform(low, high) function in python) and plot temperature for 365 days.

* Extract mean temperature per day (hint: 365 values of mean temperature) and plot mean feature.
* Calculate the variance of temperature per day (hint:365 values of variance use np.var() function in Python) and plot variance feature.
* Select the days that had highest variance of temperatures and print days for which variance> threshold.

**Code:**

**# Feature extraction from Time Series Data**

**# importing neccessary python pakages**

import matplotlib.pyplot as plt

import numpy as np

import random

import statistics

import pandas as pd

**# defining function for generating random temperature values**

def random\_floats(low, high, size):

return [random.uniform(low, high) for \_ in range(size)]

print('\n')

**# defining array for storing respective values**

Temperature\_yearly=np.array([])

Temperature=np.array([])

Temperature\_mean=np.array([])

Temperature\_var=np.array([])

**# generating Time Series Temperature, Mean per day, Variance per day**

for i in range(365):

Temperature=random\_floats(0, 50, 24)

Temperature\_yearly=np.append(Temperature\_yearly,Temperature)

Temperature\_mean=np.append(Temperature\_mean ,np.mean(Temperature))

Temperature\_var=np.append(Temperature\_var, statistics.variance(Temperature))

**# printing shape of respective arrays to confirm the total values**

print(Temperature\_yearly.shape)

print(Temperature\_mean.shape)

print(Temperature\_var.shape)

**# intializing figure 1 for plotting Temperature values**

plt.figure(1, figsize=(10,8))

**# plotting temperature per year**

plt.subplot(111)

plt.plot(Temperature\_yearly, color='purple')

plt.xlabel("Time (One Year)")

plt.ylabel("Temperature")

plt.grid()

**# intializing figure 2 for plotting Mean Temperature values**

plt.figure(2, figsize=(8,6))

**# plotting Mean temperature per day**

plt.subplot(111)

plt.plot(Temperature\_mean, color='brown')

plt.xlabel("Time (365 Days)")

plt.ylabel("Mean(average) Temperature per Day")

plt.grid()

**# intializing figure 2 for plotting Mean Temperature values**

plt.figure(3, figsize=(6,4))

**# intializing figure 3 for plotting Variance Temperature values**

plt.subplot(111)

plt.plot(Temperature\_var, color='gray')

plt.xlabel("Time (365 Days)")

plt.ylabel("Variance Temperature per Day")

plt.grid()

**# showing all graphs**

plt.show()

**# Selecting any random value of variances as threshold**

threshold=random.choice(Temperature\_var)

print("\n")

print("Threshold : ")

print("\n")

print(threshold)

**# defining an array for saving variance values greater than threshold**

days\_above\_threshold=np.array([])

**# designing algorithm to sort out the day having variance values greater than Threshold**

for i in Temperature\_var:

days\_above\_threshold=np.append(days\_above\_threshold,np.where(Temperature\_var>threshold))

**# printing variances above threshold**

print("\n","Days above Threshold : ","\n")

print(days\_above\_threshold)

**# converting days\_above\_threshold in DataFrame**

day\_above\_DataFrame=pd.DataFrame(days\_above\_threshold)

**# exporting days\_above\_threshold in .csv**

day\_above\_csv=day\_above\_DataFrame.to\_csv("D:\day\_above\_DataFrame.csv")

**Output:**

(8760,)

(365,)

(365,)

**Threshold:**

149.48455629372353

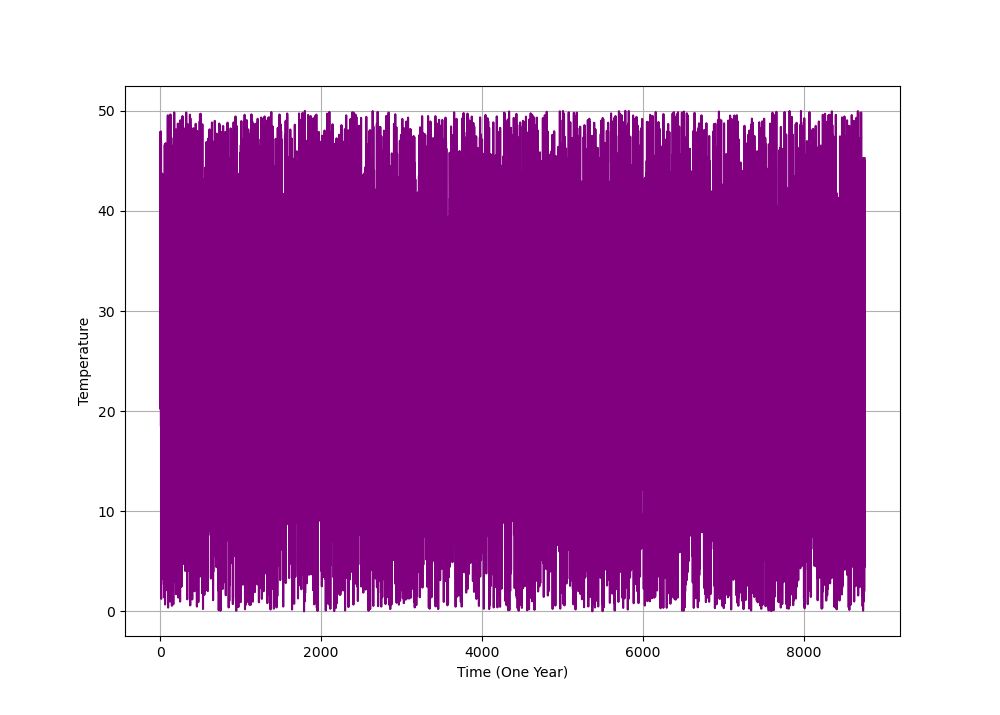
**Days above Threshold:**

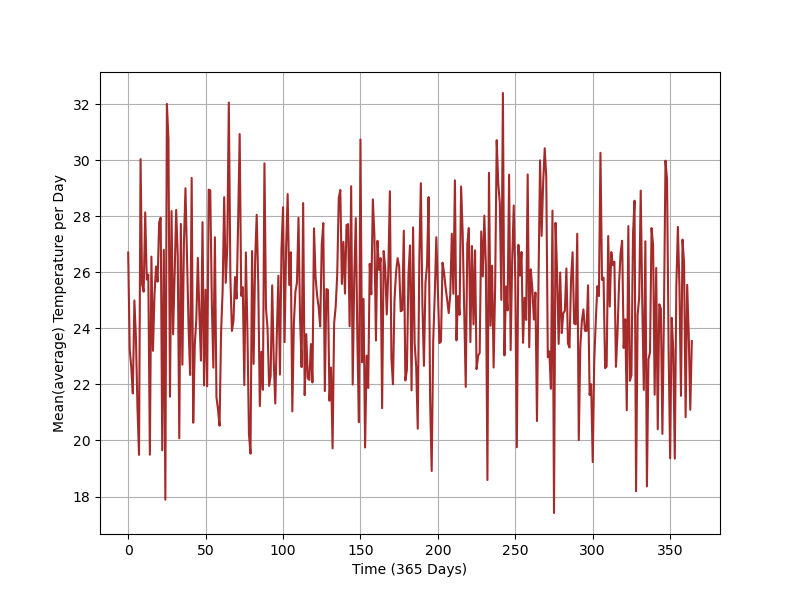
[ 1. 3. 23. ... 353. 357. 361.]

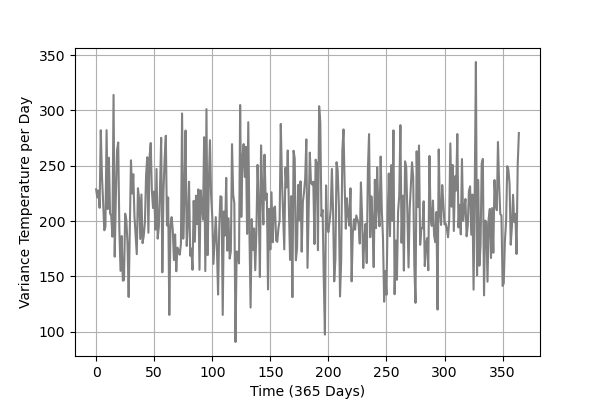
Note: as the array is so large so I am exporting **day\_above\_threshold** in **.csv** file and giving google drive link below:

<https://drive.google.com/file/d/126vCq2nIpDdOhfrlKDRC-CqrdogWvL48/view?usp=sharing>

please go through this link and acknowledge my complete results of array.





****

**Results and Discussions:**

In this practical I learned to. Extract features from a random time series temperature data by classification using threshold value. And extract the days which are having variances above to the threshold. As we are choosing days and values randomly so the results are changing at each run of program. The main issue I faced in this practical is designing the algorithms for feature extraction. After completing this practical I realize that this could be very helpful in the field of time series big data manipulation.

The python packages I used in this practical: -

* NumPy
* SciPy
* matplotlib
* statistics
* pandas
* random

**Conclusion:**

**Need of Feature Selection:**

As discussed in class when we give our raw data to any machine learning algorithm so the outcome trained model results with very low efficiency because when we exquisite the real-world data so that data has so many irrelative and useless features which are not related with the objective we want to achieve by trained model. So that data actually called noise which can never let a model to achieve the accuracy we expect from model. We generally use to solve this problem by Feature Extraction Techniques.

In Feature Extraction method, we extract/select the optimal feature from our raw data by sub setting a group of features to give algorithm to achieve our expected task with high accuracy.