PCA-Based image recognition to classify Volcanoes on Venus

**Name: Dheeraj Kumar Tatipamula**

**MavID:1002003485**

**Subject: Machine Learning (CSE6363)**

**Professor: Manfred Huber**

We are using PCA based features to calculate the KNN and find nearest neighbours for the given data which we got it from UCI website.

Dataset:

The original dataset was created in part by the California Institute of Technology's Jet Propulsion Laboratory, under contract with the National Aeronautics and Space Administration, and with the help of the UCI Machine Learning Repository (http://archive.ics.uci.edu/ml).

We give the data in two parts: training and testing. Photos of one channel 110x110, pixels from 0 to 255, where each image is one row of 12100 columns (all 110 rows of 110 columns), this image can contain more than one volcano or none. We also give the label data, or "ground truth," (trainlabels.csv and testlabels.csv), which is divided into four columns as follows:

Volcano? if in the image there are volcanoes (Main target), 1 or 0.

for Volcano? =0 these three next features are Nan

The data is uneven, which must be considered; the number of volcanoes is less than zero.

Incomplete Values

Blank patches appear in certain photos because to pauses in the Magellan acquisition or communication operations. These areas can usually be overlooked.

Steps:

1. First, we have load train images and train labels to the Notebook environment.
2. Then plot the graph with number of images with volcanoes and number of images without volcanoes and set the graph to the different dimension as we require.

Chart, histogram

Description automatically generated

1. We can see different images with volcanoes and without volcanoes in the given dataset print those images.
2. Plotted images using Jupiter Notebook in the attached code.
3. Then reshape the images with using sizes we require
4. The next important step is to scale the data to different set where we have lower and higher bound for the values, we can use scaler or min max scaler for scaling the data of different variations to bound in min max of range, so it is used for easier calculations.

Table

Description automatically generated

1. After scaling the data, the next important step is to do principal component analysis for given data.

Principal Component Analysis:

Let's say we have a dataset with two variables and 10 data points. We can readily visualize the data points if we are asked to do so. The result is also quite interpretable.

When the number of variables is increased, it becomes nearly hard to envisage a dimension higher than three dimensions. The "curse of dimensionality" is the challenge we confront while evaluating higher-dimensional dataset

Principal Component Analysis (PCA) lowers high-dimensional data to smaller dimensions while preserving the dataset's maximum variability. PCA is most used for data visualization. PCA can also be used to speed up algorithm training by lowering the amount of data dimensions.

he PCA algorithm is going to standardize the input data frame, calculate the covariance matrix of the features.

Steps to implement PCA in Python:

1. Subtract the mean of each variable
2. Calculate the Covariance Matrix
3. Compute the Eigenvalues and Eigenvectors
4. Sort Eigenvalues in descending order
5. Select a subset from the rearranged Eigenvalue matrix
6. Transform the data

We choose PCA number of components as 9 for 95% variance for threshold this gives us best solution.

Chart, line chart

Description automatically generated

Referred article: https://www.mikulskibartosz.name/pca-how-to-choose-the-number-of-components/

Then we implement KNN for calculating the neighbours and finding the volcano present in it or not.

We implemented algorithm for calculating the 5 nearest neighbours and 9 nearest neighbours and 15 nearest neighbours and calculating the accuracy for each value

For K=5:

We calculated 5 nearest neighbours for each point in test data set and match it with given accurate test labels and compare the values.

Find the accuracy for each k value to find best fit for the algorithm:

We get accuracy as 0.8258961228968544

For K=9:

We calculated 5 nearest neighbours for each point in test data set and match it with given accurate test labels and compare the values.

Find the accuracy for each k value to find best fit for the algorithm:

We get accuracy as 0.8280907095830286

For K=15:

We calculated 15 nearest neighbours for each point in test data set and match it with given accurate test labels and compare the values.

Find the accuracy for each k value to find best fit for the algorithm:

We get accuracy as 0.832114118507681

If we increase K value much higher than 20 our dataset going to lose the accuracy and overfit for the algorithm because

We have uneven data ratio of volcanoes in images to no volcano is 1:6. If we try to find more nearest neighbours because of uneven data our algorithm wrongly predicts the no volcano in volcanic image because of high domination of no volcano data in the dataset.

So, it is very important to choose less K value for getting the accurate value for the test images.

Here we show images we correctly predicted of volcanoes in test data where it is matching the test labels.

A picture containing chart

Description automatically generated

Thank you, I paste all references below:

<https://www.askpython.com/python/examples/principal-component-analysis>

<https://www.kaggle.com/code/max398434434/volcanos-on-venus-acc-94-1/notebook>

<https://www.mikulskibartosz.name/pca-how-to-choose-the-number-of-components/>

<https://www.kaggle.com/datasets/fmena14/volcanoesvenus>

<https://youtu.be/QdBy02ExhGI>