**Name : Sai Venkata Krishnaveni Devarakonda**

**Student Id : 1002003125**

**Project : PCA-Based image recognition to classify Volcanoes on Venus**

**Project Description:**

PCA-Based image recognition to classify Volcanoes on Venus: using PCA-based features and a k-Nearest Neighbor classifier to classify the certainty that there is a volcanoes in the picture from the Volcanoes on UCI Venus image dataset. To make this problem easier, I did pre-processing for images by cropping them around the indicated center of the entity into a smaller size.

**Data Description:**

The following two folders from UCI Venus image dataset are used to build this project.

1. **Images:**

This folder contains img#.sdt and img#.spr files for 134 images. Mainly img#.sdt files are used in this project. These .sdt files are binary files of images. Each image is 1024 X 1024 X (unsigned) 8-bit.

1. **Ground Truths:**

This foldercontainsimg#.lxyr and img#.jtri files for 134 images. Mainly img#.lxyr files are used in this project. There is no absolute ground truth provided for this data set. The labels provide some measure of subjective uncertainty (1 = definitely a volcano, 2 = probably, 3 = possibly, 4 = only a pit is visible).

**Steps followed:**

* Load the images from UCI Venus image dataset
* Load lxyr (label, center of volcano(x,y),radius of the volcano) from UCI Venus image dataset.
* Crop the images by using volcano center(x,y) and its radius.
* Reduce the dimensionalities for images by using PCA
* Split data into train and test
* Use KNN classifier for classification of certainty of volcano in the image.
* Use performance measure to calculate performance of the model
* Conduct below experiments
  + Vary k value in KNN classifier to identify the difference in performance
  + Vary PCA components to identify the difference in performance
  + Vary train data and test data split to identify the difference in performance

|  |  |
| --- | --- |
| Illustration of volcanoes and corresponding radii | |
| **Graphical user interface  Description automatically generated** | **Chart, scatter chart  Description automatically generated** |

1. Overlay circles on the grayscale image showing volcanoes spatial presence on two example images.
2. Plot showing distribution of radius of the volcanoes

Chart, histogram

Description automatically generated

**Observation:** The distribution of radii for all volcanoes shows bimodal distribution with two of its peaks centered around ~5 and ~25. Here we chose 25x25 as the resizing shape for all volcanic image patches.

1. Histogram showing distribution of labels of volcanic image patches

Chart, bar chart

Description automatically generated

**Observation:** Class labels are highly imbalanced (1 = definitely a volcano, 2 = probably, 3 = possibly, 4 = only a pit is visible). Train and Test splits are created by stratifying the distribution of class labels in all splits.

1. Plot showing percent variance explained versus number of principal components.

Chart

Description automatically generated

Observation: About 90% of variance in the data is explained when we use more than first 40 principal components

**Data preprocessing:** A total of 134 images of each size 1024x1024 are used for this study. Intensity values are in the range of [0 255] following uint8 datatype. I had extracted images patches of size (radiusxradius) from the volcano center (x,y). Each extracted image patch is resized to a common size of 25x25. This size is used as it is one of the peaks (mean) of radius distribution. I used **scikit-image** function resize for performing image resize. PCA is performed on the flattened image patches using **np.linalg.eig** after z-score normalization and construction of variance covariance matrix. Each image patch is flattened and reduced to lower dimensions using PCA.

**Test-Train Split:** Train and Test data folds are split by ensuring uniform label distribution identical to the label distribution of entire dataset. 5-fold cross validation is used to compute mean performance of KNN classifier.

**Model Training:** KNN classifier is used to perform prediction of class label. Neighbors are weighted by the inverse of the class size to address class imbalance. This step provides us the fraction of class present in the neighbors. Number of neighbors is tuned manually using a grid search of ‘k’ with values [1,3,5,9,15,19,39].

**Addressing class imbalance:** KNN classifier is modified to address class imbalance in the training dataset. While computing the majority voting, class size is used as normalizing factor (weighted by inverse of class size) to get a fraction of class values in the selected ‘k’ neighbors.

**Performance evaluation:** Multiple holdout train and test sets are computed using 5-fold CV. Each split is ensured to have identical label distribution. Accuracy is computed on outer test fold for each k-fold. Mean test accuracy is reported as the performance measure for KNN classifier. Accuracy for each label is computed to check the performance on each label category (1,2,3,4).

Plotting first two principal components (scatterplot)

Chart, scatter chart

Description automatically generated[red = 1, green = 2, blue = 3, pink = 4]

**Observations**: Label 4 data is separable and had better prediction accuracy, 1,2,3, are not readily separable in two dimensions.

**Tables of experimental results:**

**1.Number of principle components = 200, image cropping size = (25,25), with one train and test split (75%,25%)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Num of neighbors (K) | Accuracy (label=1) | Accuracy(label=2) | Accuracy(label=3) | Accuracy (label=4) |
| 1 | Train:100  Test:27.78 | Train:100  Test:26.39 | Train:100  Test:53.17 | Train:100  Test:92.30 |
| 2 | Train:100  Test:36.11 | Train:80.55  Test:38.89 | Train:60.74  Test:38.09 | Train:93.47  Test:86.71 |
| 3 | Train:72.64  Test:22.22 | Train:50.0  Test:20.83 | Train:77.45  Test:56.35 | Train:95.34  Test:92.31 |
| 4 | Train:63.2  Test:19.44 | Train:56.94  Test:25.0 | Train:72.41  Test:58.73 | Train:94.64  Test:91.61 |
| 5 | Train:59.43  Test:25.0 | Train:46.76  Test:25.0 | Train:76.13  Test:64.29 | Train:93.94  Test:90.01 |
| 6 | Train:48.11  Test:25.0 | Train:41.67  Test:19.44 | Train:79.84  Test:71.43 | Train:93.24  Test:90.21 |
| 7 | Train:45.28  Test:16.67 | Train:39.35  Test:22.22 | Train:83.55  Test:70.63 | Train:93.70  Test:90.91 |

**2.Number of principle components = 200, image cropping size = (25,25), with 5-fold(imbalance classes) cross validation**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Num of neighbors (K) | Accuracy (label=1) | Accuracy(label=2) | Accuracy(label=3) | Accuracy (label=4) |
| 1 | Train:100  Test:22.14 | Train:100  Test:22.81 | Train:100  Test:54.8 | Train:100  Test:94.21 |
| 2 | Train:100  Test:36.43 | Train:81.14  Test:37.54 | Train:64.25  Test:38.8 | Train:94.78  Test:88.95 |
| 3 | Train:68.57  Test:30.71 | Train:53.42  Test:17.54 | Train:79.85  Test:58.8 | Train:96.71  Test:93.16 |
| 4 | Train:56.61  Test:15.71 | Train:61.40  Test:22.46 | Train:75.95  Test:60.6 | Train:94.96  Test:91.23 |
| 5 | Train:51.79  Test:17.86 | Train:51.67  Test:23.16 | Train:79.0  Test:65.4 | Train:94.74  Test:91.23 |
| 6 | Train:41.25  Test:18.57 | Train:47.46  Test:19.29 | Train:80.35  Test:68.4 | Train:93.99  Test:90.88 |
| 7 | Train:36.43  Test:12.14 | Train:44.29  Test:19.65 | Train:82.6  Test:70.6 | Train:93.73  Test:90.35 |

**3.Number of principle components = 200, image cropping size = (25,25), with 5-fold(equal sizes) cross validation**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Num of neighbors (K) | Accuracy (label=1) | Accuracy(label=2) | Accuracy(label=3) | Accuracy (label=4) |
| 1 | Train:100  Test:40.71 | Train:100  Test:32.14 | Train:100  Test:37.14 | Train:100  Test:86.43 |
| 2 | Train:100  Test:60.71 | Train:71.07  Test:26.43 | Train:55.18  Test:20.71 | Train:90.54  Test:80.0 |
| 3 | Train:81.43  Test:45.71 | Train:63.39  Test:25.0 | Train:66.07  Test:32.86 | Train:92.5  Test:84.28 |
| 4 | Train:78.03  Test:45.0 | Train:68.39  Test:29.29 | Train:61.61  Test:32.14 | Train:91.79  Test:82.14 |
| 5 | Train:73.93  Test:45.71 | Train:60.71  Test:27.14 | Train:60.0  Test:32.86 | Train:91.78  Test:82.86 |
| 6 | Train:70.71  Test:40.0 | Train:61.25  Test:25.0 | Train:60.54  Test:37.86 | Train:90.71  Test:84.28 |
| 7 | Train:64.64  Test:40.0 | Train:59.46  Test:27.14 | Train:60.53  Test:42.86 | Train:89.82  Test:82.86 |

**4.K value in KNN classifier = 2, image cropping size = (25,25), with 5-fold(equal sizes) cross validation**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PCA components | Accuracy (label=1) | Accuracy(label=2) | Accuracy(label=3) | Accuracy (label=4) |
| 2 (%var explained = 69.48) | Train:100  Test:56.43 | Train:70.71  Test:17.86 | Train:41.79  Test:17.14 | Train:80  Test:74.28 |
| 5(%var explained =79.18) | Train:100  Test:51.43 | Train:70.18  Test:23.57 | Train:58.39  Test:17.14 | Train:83.57  Test:77.14 |
| 15(%var explained = 88.01) | Train:100  Test:56.43 | Train:70.18  Test:25 | Train:58.39  Test:22.14 | Train:83.57  Test:75.71 |
| 25(%var explained =90.61 ) | Train:100  Test:57.86 | Train:73.04  Test:30 | Train:55.18  Test:20 | Train:84.62  Test:77.86 |
| 35(%var explained = 92.09) | Train:100  Test:59.28 | Train:71.61  Test:31.43 | Train:54.11  Test:17.14 | Train:85  Test:75.71 |
| 45(%var explained = 93.09) | Train:100  Test:57.86 | Train:74.11  Test:30.0 | Train:56.07  Test:17.14 | Train:85.89  Test:76.43 |
| 70(%var explained = 94.71) | Train:100  Test:61.42 | Train:73.39  Test:31.43 | Train:54.82  Test:15.71 | Train:87.14  Test:77.14 |
| 200(%varexplained= 98.11) | Train:100  Test:60.71 | Train:71.07  Test:26.43 | Train:55.18  Test:20.71 | Train:90.54  Test:80 |

**Conclusion:**

I have implemented PCA based image patch classification for a multiclass problem using KNN classifier to predict the uncertainty in class labels for volcanic image paths in UCI-JARTOOL dataset. For a single holdout test train split with 75% of data in train and 25% of data in test split. For unbiased performance evaluation I have implemented a 5-fold cross validation by stratifying the distribution of labels in each fold. I have observed that Increasing k (number of neighbors) in KNN classifier has lead to a decrease in prediction accuracy for labels 1 and 2.This might be due to these labels being not separable in reduced PCA space. Also, number of samples in labels 1 and 2 are much lower compared to number of labels 3 and 4. PCA plot in 2 components shows the same. I have addressed the class imbalance by following 2 methods: 1. Weighting neighbor counts by inverse of class labels. 2.uniform sampling of observations from each label (n=140).These techniques are resulted in increased accuracy for label 1.Overall label 4 can be distinguished with high accuracy. This might be because of distinct image features for label 4. Using convolution Neural Networks(CNN), addressing intensity confounds might lead to better accuracy.