2020-0701 IST 707 Data Analytics

Final Project - Image Classification Using Google Landmarks Data

# 

Daphne Chang

Sathish Kumar Rajendiran

Sharat Sripada

2020-09-08

Table of Contents

[Task Summary 2](#_Toc50652279)

[Dataset Preparation 3](#_Toc50652280)

[Decision Trees (DT) 11](#_Toc50652281)

[Supply Vector Machine (SVM) 15](#_Toc50652282)

[k-Nearest Neighbors (kNN) 18](#_Toc50652283)

[Random Forest 19](#_Toc50652284)

[Summary 20](#_Toc50652285)

[Conclusion 20](#_Toc50652286)

## **Task Summary**

The data set comes from the Kaggle Image retrieval competition. The goal is to recognize and classify images in respective categories. Images are converted to 28 \*28 pixel with total of 784 pixels. Objective is to predict the images correctly by label using various conventional supervised machine learning models (Decision tree, kNN, SVM and Random Forest). In addition, tune their parameters to get the best model (measured by cross validation) and compare which algorithms provide better model for this task.

1. Dataset Preparation
   1. Data extraction & Split train & test dataset
   2. Principal Component Analysis (PCA)
2. Build Decision Tree model
   1. Run decision tree models with and without PCA
3. Build k-Nearest Neighbors (kNN) model
   1. Build and Predict using kNN model
4. Build Supply Vector Machine (SVM) model
   1. Run with linear, radial and polynomial kernel settings
5. Random Forest
   1. Build and Predict using Random forest model
6. Summary
   1. Summarize and compare the models

## **Dataset Preparation**

In order to build the classifier models, first step is to understand the data and prepare training and test data sets. It involves the following steps,

* 1. Data extraction - import necessary packages for data extraction.

A screenshot of a cell phone

Description automatically generated

A screenshot of text

Description automatically generated

A screenshot of a cell phone

Description automatically generated

Function to convert the images into grayscale 28 \*28 pixels matrix with

Training data has 42,000 rows ,784 predictor variables and 1 dependent variable (label) columns. Change the first column (label) as factor and make sure the remaining columns are numeric.

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

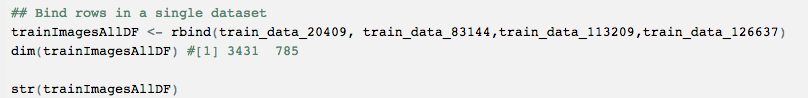
A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

Let’s, combine the datasets into one dataframe.



A screenshot of a cell phone

Description automatically generated

Let’s split the Training data into train\_set and test\_set with 80:20 split.

A screenshot of a social media post

Description automatically generated

Preview and datatype of the dataset below,

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

* 1. Principal Component Analysis (PCA)

As the predictor variables are more – it would be wiser to look at other options such as dimensionality reduction to improve the model performance. Principal component analysis help reduce number of dimensions to optimize the model efficiency. In this case, reduce the predictor variable to 30 and rename the columns.

A screenshot of a cell phone

Description automatically generated

A close up of a map

Description automatically generated

A close up of a map

Description automatically generated

A screenshot of a social media post

Description automatically generated

next step is to prepare train and test data set based on the PCA dataset.

A screenshot of a social media post

Description automatically generated

## **Decision Trees (DT)**

Next step is to build and tune machine learning models.

1. Build DT model with control measures (cp, minsplit and maxdepth)

rpart library – from R Studio is used for building the decision trees. Initial case let’s use the original data set split with 784 predictor variables and parameters (cp=0, minsplit=1 and maxdepth=5). This process involves, building the tree, plot and summarize it with cross validation results.

A screenshot of a cell phone

Description automatically generated

A close up of a map

Description automatically generatedA close up of a map

Description automatically generated

A close up of a map

Description automatically generated

A close up of text on a black background

Description automatically generated

A screenshot of a cell phone

Description automatically generated

A close up of text on a white background

Description automatically generated

From above, Decision tree with original dataset - Accuracy is at 68.43%

1. Tune Decision tree model with PCA & control measures (cp, minsplit and maxdepth)

Let’s build the tree but with train set prepared through PCA. i.e. with 30 predictor variables and analyze the model performance

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generatedA screenshot of a map

Description automatically generated

A close up of text on a white background

Description automatically generated

A screenshot of a cell phone

Description automatically generatedA close up of text on a white background

Description automatically generated

A screenshot of a cell phone

Description automatically generatedA picture containing large

Description automatically generated

Prepare the output file for kaggle

A screenshot of a cell phone

Description automatically generated

A close up of a building

Description automatically generated A screenshot of a cell phone

Description automatically generated

From above, Decision tree on PCA dataset with control measures (cp=0, minsplit=1 and maxdepth=10) - Accuracy is at 81.46%

## **Supply Vector Machine (SVM)**

* SVM model on the PCA dataset with “linear” kernel and cost factor at 10

A screenshot of a social media post

Description automatically generated

A screenshot of a cell phone

Description automatically generated

A close up of text on a white background

Description automatically generated

From above, SVM model on PCA dataset with control measures (cost = 10, kernel=” linear”) - Accuracy is at 91.96%. Better than either of decision tree models.

* SVM model on the PCA dataset with “polynomial” kernel and cost factor at 100

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

From above, SVM model on PCA dataset with control measures (cost = 100, kernel=” polynomial”) - Accuracy is at 92%. Better than either of decision tree models, slightly better than SVM linear model

* SVM model on the PCA dataset with “radial” kernel and cost factor at 100

A screenshot of a social media post

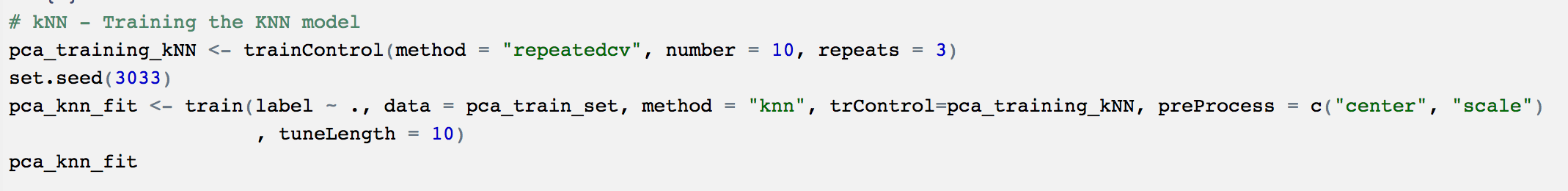
Description automatically generated

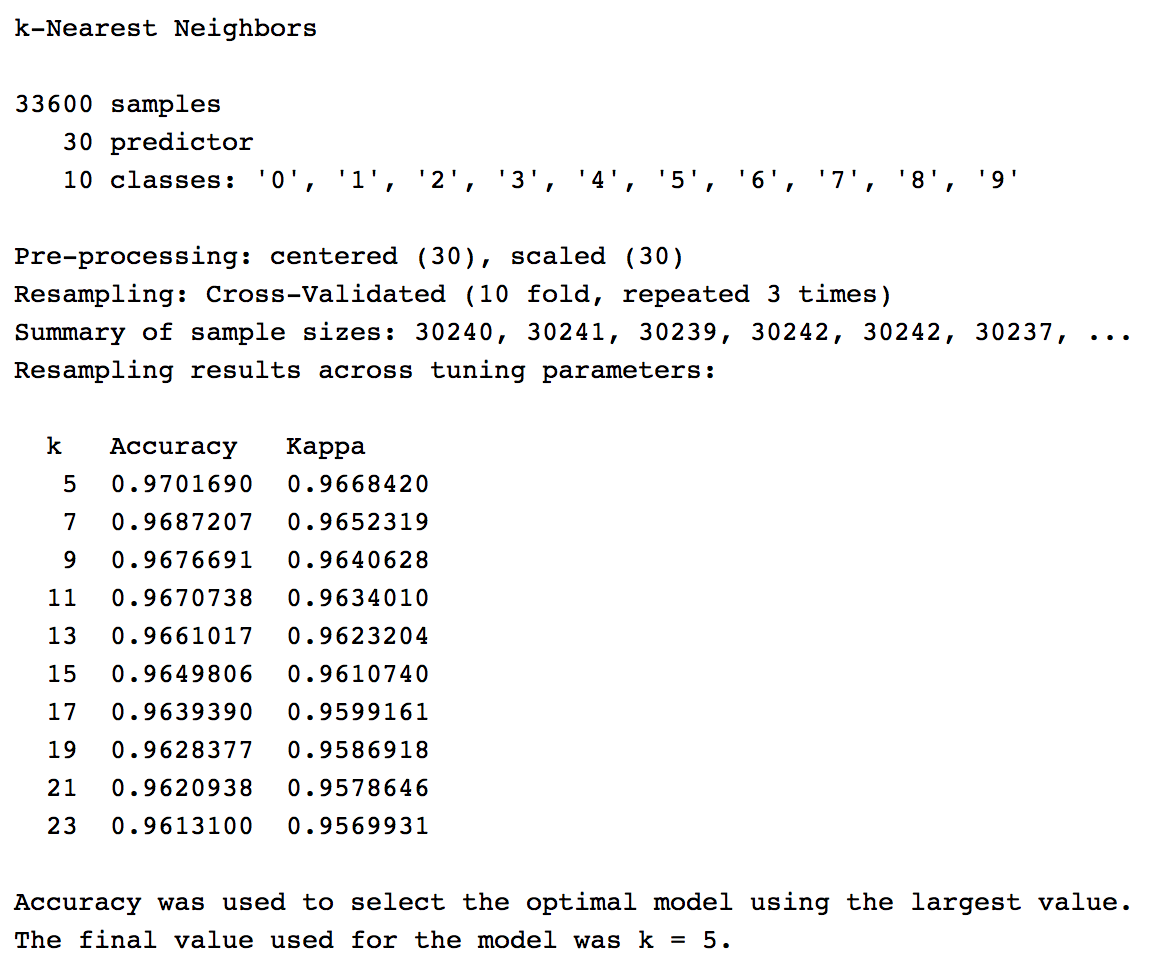
A screenshot of a cell phone

Description automatically generated

From above, SVM model on PCA dataset with control measures (cost = 100, kernel=” radial”) - Accuracy is at 94.88%. Better than either of decision tree models, SVM linear & polynomial models.

## **k-Nearest Neighbors (kNN)**





A screenshot of a cell phone

Description automatically generated

From above, kNN model on PCA dataset with control measures (repeats = 3 and k= 5) Accuracy is at 97.01%. Better than either of decision tree, all the SVM models.

## **Random Forest**

A screenshot of a cell phone

Description automatically generated

A close up of a sign

Description automatically generated

From above, Random forest model on actual train dataset with control measures (ntrees=50) had an accuracy is at 95.28%. Better than decision tree, all the SVM models, and naïve bayes models.

## **Summary**

* Based on above analysis
  + dimensionality reduction –number of predictor variables were reduced to 30 with principal component analysis.
  + Decision Tree models – decision tree with actual train and test data sets yielded an accuracy of 68.43%; Whereas, with PCA dataset the accuracy has increased to 81.46%
  + Supply Vector Machine (SVM) models– 3 versions of SVM model built on PCA dataset had slightly better results than DT models. SVM with linear Kernel performed better than DT models at 91.06% accuracy followed by SVM (polynomial – kernel) with 92% and SVM (radial -kernel) at 94.88% accuracy results.
  + k-Nearest Neighbors (KNN) – on PCA dataset had better results at k value 5 as 97.01%.
  + Naïve Bayes – This model predicted at 86.02% with PCA dataset
  + Random Forest model predicted with 95.28% on the actual dataset with number of trees set as 50

## **Conclusion**

In comparison with all the classification models, PCA really helped drive the values much better by reducing the dimensions from 784 to 30 keeping pixels only had impact on the image classification.

“Overall all the models preformed significantly over 80% accuracy with kNN (k =5) emerging as the winner with 97.01% followed by Random Forest with 95.28% accuracy and SVM (kernel=” radial”) at 94.88% accuracy.

Very good exercise to understand the behavior of various classification models and to highlight the significance of dimensionality reduction when dealing with huge predictor set.