Name:- Swapna Vaidya ***Class:-*** MSDS – 422

COLAB NOTEBOOK LINK :- https://colab.research.google.com/drive/1ObtyGVgKO0VRwIIG5Vczrk8RUwSAfYsX

Principal Component Regression Models

(1) ***A summary and problem definition for management***

MNIST ("Modified National Institute of Standards and Technology") is the de facto “hello world” dataset of computer vision. Since its release in 1999, this classic dataset of handwritten images has served as the basis for benchmarking classification algorithms. As new machine learning techniques emerge, MNIST remains a reliable resource for researchers and learners alike. In this competition, the goal is to correctly identify digits from a dataset of tens of thousands of handwritten images.

From a management perspective, the predictive accuracy of models must be weighed against the costs of model development and implementation. After the experimentation, a recommendation must be made to the management using PCA as preliminary to machine learning classification

(2) ***Discussion of the research design, measurement and statistical methods, traditional and machine learning methods employed***

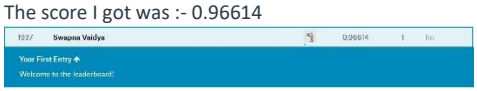
Based on the exploratory data analysis that was done the survey data, below were some of the findings:-

- There are 2 sets of data provided:- Training dataset and Test dataset.
- In the training data set there are 42000 instances with 784 features and 1 label in the data
- In the test data set there are 42000 instances with no label
- There are no missing values in any of the columns in the data set.

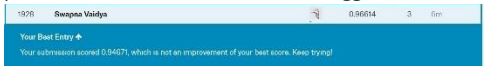
(3) ***Overview of programming work***

This assignment was done in Google Colab, using 4 main libraries (Pandas, numpy, matpotlib, Seaborn and Skylearn).

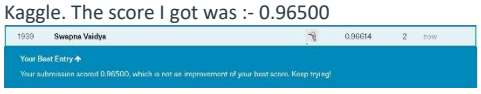
- ***Ingestion:-*** Using pandas the training and test data was first loaded from Google OnDrive.
- ***Exploration:-*** of the data was done using methods like describe(), info() and corr().
- ***Data Cleaning and Transformation:-*** No special data cleaning and transformation steps were taken on this data set.
- ***Modeling:-*** The train dataset was broken into features and label by removing the column 1 which is the label. The first modeling I did was a Random Forest model with grid search. The grid search included parameters for n_estimators, max_features, bootstrap and max_depth. The time it took to fit this model was 33mins 46 secs. When I run the same model as a default it takes 3 secs. This model was then used to predict on the test data. The predictions were then submitted on Kaggle. The score I got was :- 0.96614



The second model that was tried was with PCA using the random forest classifer. For a 95% variance, the PCA was fit on a train feature set. This PCA fit model was used to transform the train features and the test features. Next a Random Forest with grid search for parameters n_estimators, max_features, bootstrap and max_depth was fit on the train PCA features and train label data. There were initially 784 features but after the PCA with 95% variance the features were dropped to 154 features. This Random Forest with PCA model was used to predict the digits on the test data. The time it took to identify the PCA components was 7 secs. The time it took to run the grid search random forest with PCA components took 2hrs and 4 secs. The predictions were then submitted on Kaggle. The score I got was :- 0.94671

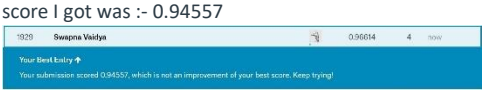


In order to fix the flaw the train data was split into train and test data. The third model Random Forest using the grid search included parameters for n_estimators, max_features, bootstrap and max_depth was then fit on the split train data. The fitted the model was then used to predict results using the split test data. The calculated RMSE error was 0.79 on test data while on the train data it was 0%. This indicates the model is slightly overfit. The model was then fit on the actual Test data provided by Kaggle. The time it took to run the grid search random forest was 29mins and 18 secs. The predictions were then submitted on Kaggle. The score I got was :- 0.96500



Finally a 4th model was enabled on the split train data using PCA and Random Forest Classifier. For the PCA a 95% variance, the PCA was fit on a train feature set. This PCA fit model was used to transform the train features and the split train test features

and then the actual Test data provided by Kaggle. Next a Random Forest with grid search for parameters n_estimators, max_features, bootstrap and max_depth was fit on the train PCA features and train label data. There were initially 784 features but after the PCA with 95% variance the features were dropped to 154 features. This Random Forest with PCA model was used to predict the digits on the split test data. The calculated RMSE error was 0.97 on test data. The model was then fit on the actual Test data from Kaggle. The time it took to identify the PCA components was 8 secs. The time it took to run the grid search random forest with PCA components took 2hrs and 4 secs. The predictions were then submitted on Kaggle. The score I got was :- 0.94557



Review of results with recommendations for management

The processing time and score for all the above models is as follows:-

Model	***Processing Time***	***Kaggle Score***	***Score Variance***	***Time Variance***
Grid Search Random Forest Model Before the Flaw	33.46	0.96614	2%	-292%
Grid Search With PCA Random Forest Model Before the Flaw	131.0	0.94671		
Grid Search Random Forest Model After the Flaw	29.18	0.96500	2%	-312%
Grid Search With PCA Random Forest Model After the Flaw	120.12	0.94557		

Based on the above scores it can be seen that a model without PCA has a higher score. While the model with PCA the accuracy drops by 2% which would be expected especially with 95% variance. But then on the other hand the time to process the PCA goes up by 292% which is mind blowing. So to reduce the processing time, I would try to increase the processing power by increasing my compute cores. Going with online cloud services like Azure Compute, if my company decides with a memory optimized and a high memory to core ratio compute cores, it will cost minimum $0.126/hour to my company. So to run one iteration of the model with PCA would cost $16/- minimum.

So finally, as a Data Science manager for my organization when I look at the visual below and compare the processing cost, the score and the $ cost , I would not recommend PCA as a preliminary to machine learning classification. The score reduces by 2% plus the processing time is too high. To reduce the processing time, if I had to increase the power, it would be very expensive in monetary terms for my organization. Also, PCA is suitable for scenario where we want to reduce multicollinearity between features or for cluster analysis to provide insight regarding the patterns. In case of classification of images none of the key PCA scenarios apply. Hence as a Data Science Manager, I will look at other machine learning classification algorithms that are cost efficient and yet increase the accuracy with reduced processing time for image classification.