CMPUT 466 Final Project Report
Dry Bean Dataset Classification

Shuxin Qiao

1. Introduction

Data Set Name: Dry Bean Dataset

Abstract:

Images of 13,611 grains of 7 different registered dry beans were taken with a high-resolution camera. A total of 16 features; 12 dimensions and 4 shape forms, were obtained from the grains. Applying Softmax Regression, Support Vector Machine, Neural Network three Machine Learning algorithms to classify dry beans and predict for future.

Background Information:

Seven different types of dry beans were used in this research, taking into account the features such as form, shape, type, and structure by the market situation. A computer vision system was developed to distinguish seven different registered varieties of dry beans with similar features in order to obtain uniform seed classification. For the classification model, images of 13,611 grains of 7 different registered dry beans were taken with a high-resolution camera. Bean images obtained by computer vision system were subjected to segmentation and feature extraction stages, and a total of 16 features; 12 dimensions and 4 shape forms, were obtained from the grains.

Source:

UCI Machine Learning Repository website
https://archive-beta.ics.uci.edu/ml/datasets/dry+bean+dataset

Dataset Author:

Murat KOKLU
Faculty of Technology,
Selcuk University,
TURKEY.
ORCID : 0000-0002-2737-2360
mkoklu@selcuk.edu.tr.

Ilker Ali OZKAN
Faculty of Technology,
Selcuk University,
TURKEY.
ORCID : 0000-0002-5715-1040
ilkerozkan@selcuk.edu.tr

2. Problem Formulation

Number of Instances (records in data set): 13611
Number of Attributes (fields within each record): 17
Data Type: Multivariate

Attributes Information:

1. **Area (A):** The area of a bean zone and the number of pixels within its boundaries.
2. **Perimeter (P):** Bean circumference is defined as the length of its border.
3. **Major axis length (L):** The distance between the ends of the longest line that can be drawn from a bean.
4. **Minor axis length (l):** The longest line that can be drawn from the bean while standing perpendicular to the main axis.
5. **Aspect ratio (K):** Defines the relationship between L and l.
6. **Eccentricity (Ec):** Eccentricity of the ellipse having the same moments as the region.
7. **Convex area (C):** Number of pixels in the smallest convex polygon that can contain the area of a bean seed.
8. **Equivalent diameter (Ed):** The diameter of a circle having the same area as a bean seed area.
9. **Extent (Ex):** The ratio of the pixels in the bounding box to the bean area.
10. **Solidity (S):** Also known as convexity. The ratio of the pixels in the convex shell to those found in beans.
11. **Roundness (R):** Calculated with the following formula: (4piA)/(P^2)
12. **Compactness (CO):** Measures the roundness of an object: Ed/L
13. **ShapeFactor1 (SF1)**
14. **ShapeFactor2 (SF2)**
15. **ShapeFactor3 (SF3)**
16. **ShapeFactor4 (SF4)**
17. **Class** (Seker, Barbunya, Bombay, Cali, Dermosan, Horoz and Sira)

Split 17. Class feature as the label. Concatenate all 1 bias as the new 17th feature.

Normalization

Before training on algorithms, we need to look through the data first.
These attributes are not sharing the same unit. Some of them have significantly larger value than others.
To obtain them all on a reasonable scale, we need to normalize them.
Here, I chose to treat variables as normal and subtract its mean and divided by its standard deviation.

$$new\ value = \frac{old\ value\ -\ mean}{Std.}$$

3. Algorithms

Data Summary:

Name	**Shape**	**Details**
Raw shape	**(13611, 17)**	**16 features + 1 class**
Input shape	**(13611, 17)**	**16 features + 1 bias**
Split	**Train : Validation : Test**	**11000 : 1000 : 1611**

Label	BARBUNYA	HOROZ	BOMBAY	SEKER
Number	1,322	1,928	522	2,027
Label	CALI	SIRA	DERMASON	
Number	1,630	2,636	3,546	

Infrastructure:

Training-Validation-Test, with hyperparameter tuning.

1. Baseline: Majority guess

Always guessing the most class.

Dermason has the most samples over the whole dataset, around 26%.

Since the dataset is unbalanced, the majority guess is better than $\frac{1}{number\ of\ class}$.

2. Softmax Regression:

Z = **W** * **X**

Y = softmax(**Z**)

Mini-batch Stochastic Gradient Descent

Cross-entropy loss

Hyperparameters:

- Learning rate: 0.1
- Batch size: 1100
- Epoch: 100

Batch size controls the convergence speed.

Learning rate controls the

3. Support Vector Machine:

Z = **W** * **X**

Y = argmax(**Z**)

Full-batch Gradient Descent

Max-margin loss

Hyperparameters:

- Regularization loss weight = 0.1
- Right classify score = 1
- Learning rate = 0.025

MaxEpoch = 100

Right classify score is safe when using 1, weights will adjust itself to adapt this range.
Regularization loss weight is the key to control the fitting power.

4. Neural Network:

Use library **tensorflow.keras**

Z = **W** * **X**
Y = softmax(**Z**) / sigmoid(**Z**)

Sequential model uses the **linear stack** of layers.
Dense layer uses the **full connection** between layers.

One layer with **64** neurons and **sigmoid** activation function.
Output layer with **7** neurons and **softmax** activation function.

Mini-batch Stochastic Gradient Descent
Cross-entropy loss
Use Keras validation split: Train : Validation = 9 : 1

Hyperparameters:
  Learning rate: 0.1
  Batch size: 100
  Epoch: 100

4. Evaluation

The measure of success is accuracy of test data.
Test data is split from the whole dataset randomly.

5. Results

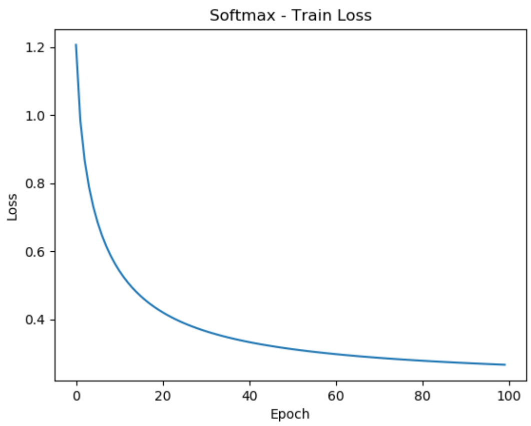
Model	Majority Guess	Softmax	SVM	NN
Test Accuracy	25.9466 %	91.8684 %	90.1303 %	91.8684 %

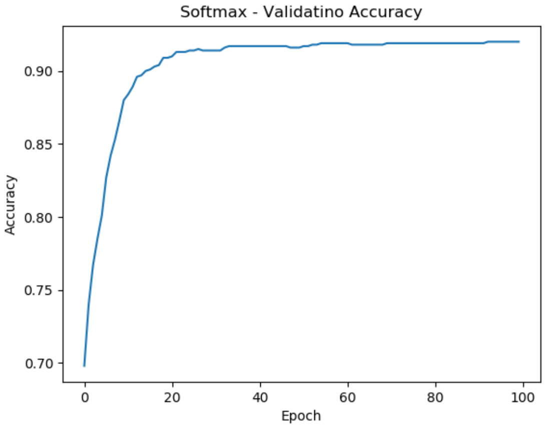
Three Machine Learning models all better than the majority guess.
Softmax Regression is a straightforward linear classifier. It produces the confidence in the classification in probability.
Support Vector Machine is a binary linear classifier but applying it to higher dimensions with some techniques. It produces the measure from the closet point to the classifier line. It separates the data with loss rather than some probability.
Neural Network is non-linear. We applied the sigmoid function as the first layer activation function. One layer seems sufficient for the duty. Output layer has the softmax activation function thus it produces the probability as softmax regression does.

According to the test accuracy results, softmax and NN have the same result while SVM performs a bit worse. But all of them have over 90% accuracy.

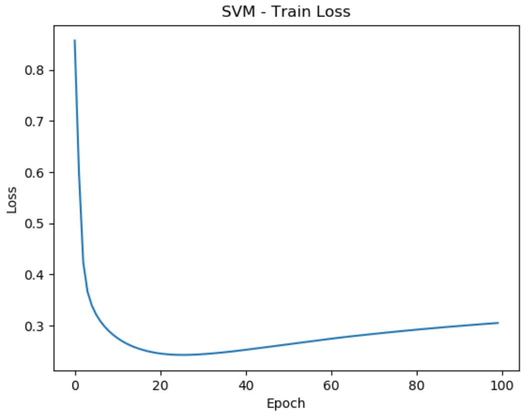
Training loss and Validation accuracy plots

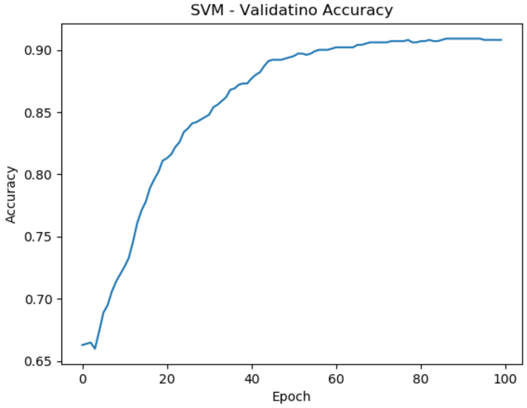
Softmax:


Softmax - Train Loss

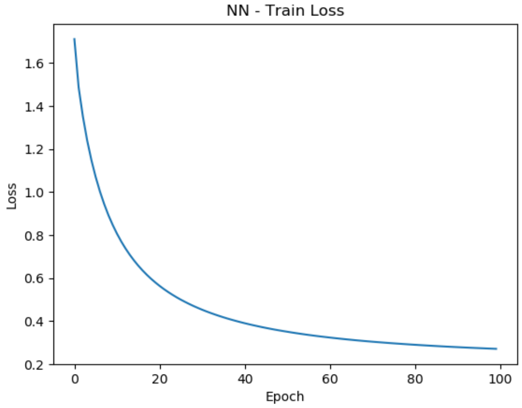

Softmax - Validatino Accuracy

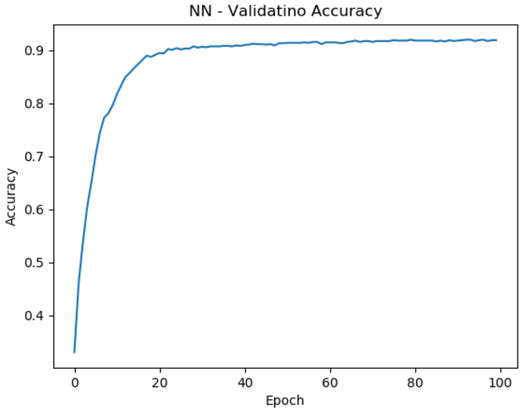
Support Vector Machine:


SVM - Train Loss


SVM - Validatino Accuracy

Neural Network:


NN - Train Loss


NN - Validatino Accuracy