Internship Project Report

DRY BEAN CLASSIFICATION

At

IC SOLUTIONS

SUBMITTED BY:

NAME: DEKSHITHA RAVIKUMAR,
HRUTHIK K Y,
SINDHU V S

USN: 1KS18CS075
4KM18CS016
1VI17IS091

EMAIL ID: dekshi17@gmail.com
sonuhruthik@gmail.com
sindhu.prema.v@gmail.com

INSTRUCTOR: ABHISHEK C

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task will be incomplete without the mention of the individuals, we are greatly indebted to, who through guidance and providing facilities have served as a beacon of light and crowned our efforts with success.

We would like to take this opportunity to express our sincere gratitude to IC Solutions for providing this Internship program on Machine Learning with python. It is with grateful heart that we thank the coordinators of IC Solutions.

We would like to thank our instructor Mr. Abhishek C for training, providing useful knowledge that can be applied on practical tasks, and providing resources. During the period of internship, we have learnt a lot. Truly thank him for his best efforts in making us understand the concepts clearly. We also thank him for his valuable guidance and support in order to complete this project.

We extend our gratitude and regards to everyone who helped us during our internship. Being a part of this internship was a great experience.

ABSTRACT

Seed classification is essential for both marketing and production to provide the principles of sustainable agricultural systems. Thus, a computer vision system was developed to distinguish different registered varieties of dry beans with similar features in order to obtain uniform seed classification. Given a dataset containing numerous rows and columns of specific details of many seeds, we have to utilize some Machine Learning Algorithms to predict the category/class, the grain belongs to. Here, accuracy score of different models are measured and compared. This kind of problem comes under Regression category so we utilize Regression Models or Algorithms to solve this problem

The Machine Learning models we utilized for this project are as follows :-

1. Support Vector Model
2. Linear Classification Model
3. Decision Tress Model
4. Random Forest Model

This is followed by Exploratory Data Analysis which depicts the relation between various parameters visually through Graphs.

ABOUT THE COMPANY

ICS is a digital service provider that aims to provide software, designing and marketing solutions to individuals and businesses. At ICS, we believe that service and quality is the key to success.

We provide all kinds of technological and designing solutions from Billing Software to Web Designs or any custom demand that you may have. Experience the service like none other!

Some of our services include:

Development - We develop responsive, functional and super-fast websites. We keep User Experience in mind while creating websites. A website should load quickly and should be accessible even on a small view-port and slow internet connection.

Mobile Application - We offer a wide range of professional android, iOS & Hybrid app development services for our global clients, from a start up to a large enterprise.

Design - We offer professional Graphic design, Brochure design & Logo design. We are experts in crafting visual content to convey the right message to the customers.

Consultancy - We are here to provide you with expert advice on your design and development requirement.

Videos - We create a polished professional video that impresses your audience.

INDEX

INTRODUCTION

There is a wide range of genetic diversity of dry bean which is the most produced one among the edible legume crops in the world. Seed quality is definitely influential in crop production. Therefore, seed classification is essential for both marketing and production to provide the principles of sustainable agricultural systems. The primary objective of this study is to provide a method for obtaining uniform seed varieties from crop production, which is in the form of population, so the seeds are not certified as a sole variety. Thus, 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.

With these results, the demands of the producers and the customers are largely met about obtaining uniform bean varieties.

PROBLEM STATEMENT AND OBJECTIVE

Problem Statement:

The given dataset- “Dry Bean Dataset” was obtained from images of 13,611 grains of seven different registered dry beans that were taken with a high-resolution camera. The dataset contains 16 features, 12 dimensions and 4 shape forms of the grains.

When such parameters are specified, the models need to predict the category/class, the grain belongs to. Uniform seed Classification should be performed.

Accuracy score of different models need to be measured and compared.

Exploratory Data Analysis need to be done and meaningful graphs has to be plotted. Relationship between various parameters need to be analysed.

Objective:

- The type of the model to use for the given problem statement should be identified. The dry beans should be classified into the respective classes as per the specified parameters.
- The given dataset contains missing values, so appropriate method to fill those values should be used.
- Clean the dataset and pass the dataset to an algorithm. The data should be scaled and then fit.
- Accuracy score is found out for that particular algorithm.
- Grid search could be used to get better accuracy score.

- Pass the various parameters into the algorithm and check if it is classifying it to its respective class correctly.
- Exploratory Data Analysis has to be done and graphs should be plotted. Necessary conclusion and relationship between various parameters in every graph should be obtained.

REQUIREMENT SPECIFICATION

Software Requirements:

- Data set containing values for features of dry beans such as form, shape, type, structure, etc.
- Python compiler as the code is written using python.
- Anaconda – Anaconda is a distribution of the Python and R programming languages for scientific computing, that aims to simplify package management and deployment. Jupyter Notebook can be installed from it.
- Alternate to jupyter notebook, Google colab can be used. Colaboratory or Colab is a product from Google Research which allows users to write and execute arbitrary python code through the browser.
- Libraries such as Numpy, Pandas, ScikitLearn, Seaborn, Matplotlib.

Hardware Requirements:

- Dell Inspiron 15 3593 Series
- 10th Generation Intel® Core™ i5-1035G1 Processor (6MB Cache, up to 3.6 GHz)
- 4GB, 1 x 4GB, DDR4, 2666Mhz

EXPLORATORY DATA ANALYSIS

Exploratory data analysis is an approach to analysing data sets to summarize their main characteristics, often using statistical graphics and other data visualization methods.

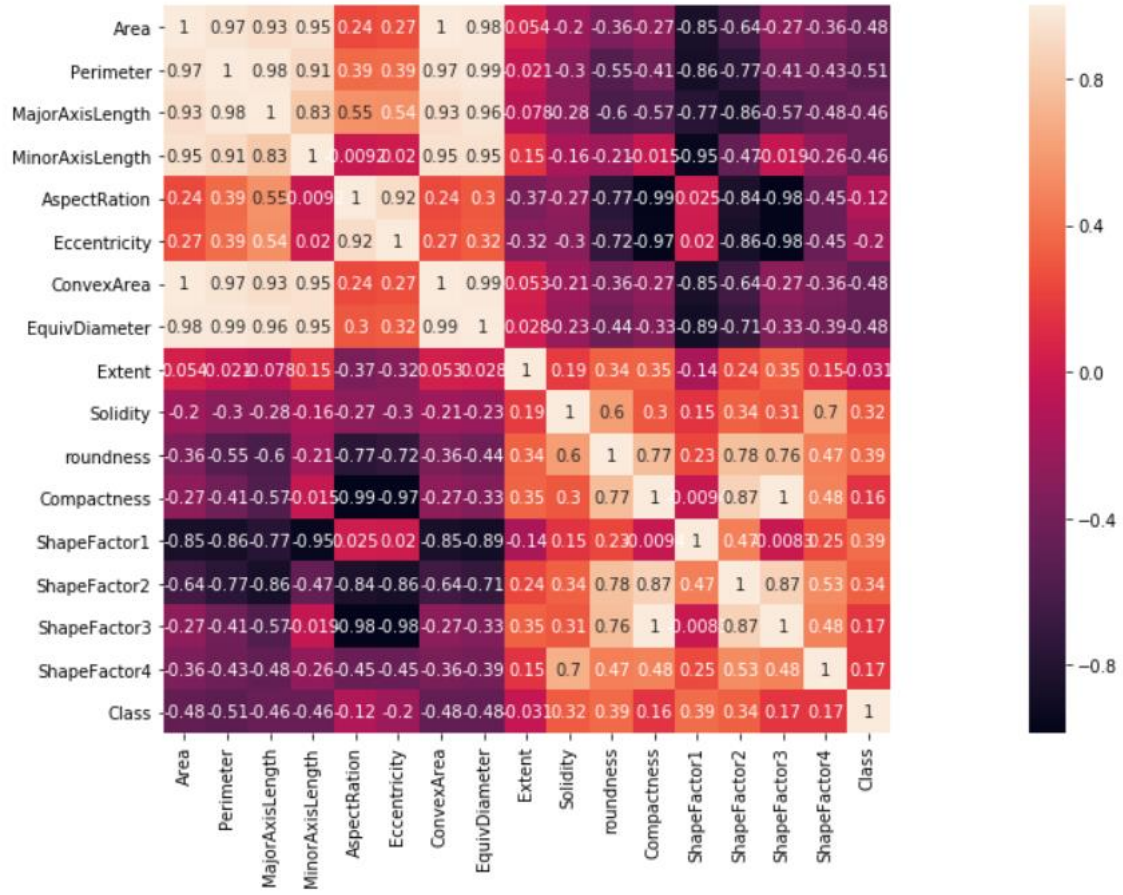
The main purpose of EDA is to help look at data before making any assumptions.

It is used to understand data, get some context regarding it, understand the variables and the relationships between them, and formulate hypotheses that could be useful when building predictive models.

1. PLOT 1 - CORRELATION HEATMAP

```
In [70]: #correlation
         plt.figure(figsize=(35,10))
         corrmat=df.corr()
         sns.heatmap(corrmat, square=True, annot=True)
```

```
Out[70]: <matplotlib.axes._subplots.AxesSubplot at 0x12ec999fba8>
```

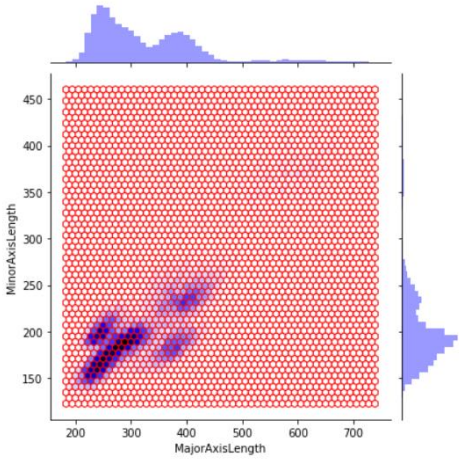


- Each square in the plot shows the correlation between the variables on each axis.
- Correlation coefficient ranges from -1 to +1. The scale on the right side shows the correlation coefficient.
- Values closer to zero means there is no linear trend between the two variables.

- Values close to 1 means the two variables are positively correlated that is, if one variable increases so does the other. Closer to 1 indicates strong relationship.
- Values closer to -1 is similar to +1 but one variable decreases as the other increases.
- The values can be interpreted as a slope of the two parameters which depicts the increase or decrease of a parameter in terms of magnitude.
- Area is strongly correlated to Perimeter, Major Axis Length, Minor Axis Length. When Area increases Perimeter, Major Axis Length, Minor Axis Length, Convex Area and Equiv Diameter increases and vice versa. These parameters are directly proportional to each other.
- For area and perimeter the correlation is +0.97, this indicates that if one increases, the other increases on a scale of 0.97.
- Perimeter is closely related to Equiv Diameter with correlation as +0.99. They are directly proportional.
- Area is strongly related to Shape Factor 1. But, when Area increases, shape factor 1 decreases and vice versa. These parameters are inversely proportional to each other.
- For Area and Shape Factor 1, correlation is -0.85, this indicates that if one increases, the other decreases on a scale of 0.85.
- All the diagonal elements are 1 as same parameters are being compared.

2. PLOT 2 – JOINT PLOT

```
In [81]: #jointplot
         sns.jointplot(x='MajorAxisLength',y='MinorAxisLength',data=df,kind='hex',dropna=True,color="blue",edgecolor="red",linew
Out[81]: <seaborn.axisgrid.JointGrid at 0x12ecb92e668>
```



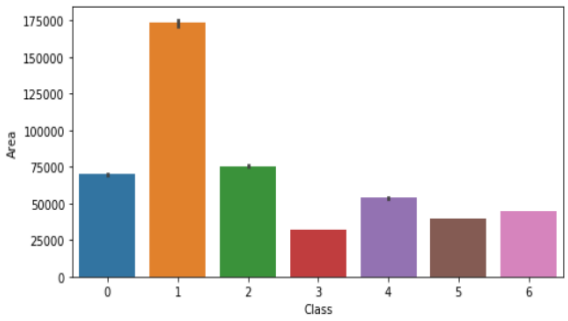
- Seaborn also provides a nice function called jointplot which will gives a scatter plot showing the relationship between two variables along with histograms of each variable in the margins also known as a marginal plot**.**
- Not only can we see the relationships between the two variables, but also how they are distributed individually.
- This particular joint plot shows relationship between Major Axis Length and Minor Axis Length and their individual distributions.
- Major Axis Length has dark points around 250-300 meaning high density region.
- On an average, dry beans have Major Axis Length in the range from 200-480.
- Minor Axis Length has dark points around 150-200 meaning high density region.
- On an average, dry beans have Minor Axis Length in the range from 150-250.
- This plot indicates that many dry beans having Major Axis Length around 250-300 has Minor Axis Length around 150-200.

- It also indicates that as Minor Axis Length increases, Minor Axis Length also increases.
- Major Axis Length is directly proportional to Minor Axis Length.

3. PLOT 3- BAR PLOT

```
In [56]: #bar
         plt.figure(figsize=(8,4))
         sns.barplot(x='Class',y='Area',data=df)
```

```
Out[56]: <matplotlib.axes._subplots.AxesSubplot at 0x12ec216cc18>
```

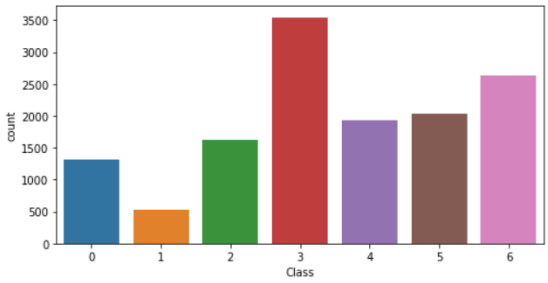


- A bar plot represents the category of data with rectangular bars with lengths and heights that is proportional to the values which they represent.
- A bar chart describes the comparisons between the discrete categories.
- One of the axis of the plot represents the specific categories or Class being compared, while the other axis represents the measured values corresponding to those categories.
- This particular bar plot has Class on X-axis and corresponding area on the Y-axis.
- There are 7 Categories of dry beans in the given data set.
  0 represents Class Barbunya , 1 represents Class Bombay, 2 represents Cali, 3 represents Dermason, 4 represents Horoz, 5 represents Seker and 6 represents Sira.
- Dry beans of ‘Bombay’ (Class 1) variety have the highest area. Area of ‘Bombay’ (Class 1) beans is on an average 170000.
- ‘Bombay’ seeds maybe be quite big.
- ‘Dermason’ (Class 3) beans have lowest area. On an average, area is 40000. It seems that this variety is small.

4. PLOT 4 – COUNT PLOT

```
In [52]: #count
         plt.figure(figsize=(8,4))
         sns.countplot(x='Class',data=df)
```

```
Out[52]: <matplotlib.axes._subplots.AxesSubplot at 0x229cf41c780>
```



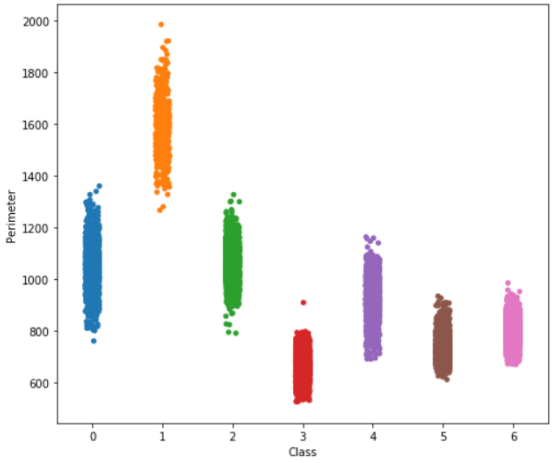
- Basically, a count plot is a graphical display to show the number of occurrences or frequency for each categorical data using bars.
- This particular plot shows the frequency of various Classes of dry beans.
- There are 7 Categories of dry beans in the given data set.

  0 represents Class Barbunya , 1 represents Class Bombay, 2 represents Cali, 3 represents Dermason, 4 represents Horoz, 5 represents Seker and 6 represents Sira.
- Class 3 – ‘Dermason’ occurs most number of times approximately 3500 times according to the graph.
- This can be verified from the data set, it has 3546 rows which belong to Class 3 – ‘Dermason’.
- Class 1 – ‘Bombay’ occurs least number of times approximately 500 times according to the graph
- This can be verified from the dataset, it has 522 rows which belong to Class 1 – ‘Bombay’.
- Thus, it seems that ‘Dermason’ is most popular Dry Bean variety.

5. PLOT 5 – STRIP PLOT

```
In [61]: #strip
         plt.figure(figsize=(8,7))
         sns.stripplot(x="Class", y="Perimeter", data=df)

Out[61]: <matplotlib.axes._subplots.AxesSubplot at 0x1f3e6fcffd0>
```



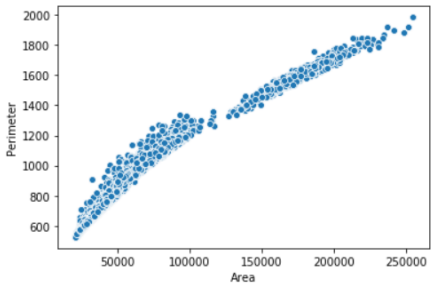
- A strip plot is a scatter plot where one of the variables is categorical.
- One of the axis of the plot represents a categorical variable, the markers of a specific category align in a straight line parallel to the other axis. It explicitly conveys that the markers for each category are just clouds of values.
- This particular plot represents Class on X-Axis and Perimeter on Y-axis.
- It represents the range of perimeter values of Dry beans of every Class.
- It gives better understanding of perimeter values.
- There are 7 Categories of dry beans in the given data set.

  0 represents Class Barbunya , 1 represents Class Bombay, 2 represents Cali, 3 represents Dermason, 4 represents Horoz, 5 represents Seker and 6 represents Sira.
- Class 0 has perimeter values in the range of 800-1390 approximately.
- Class 1 has perimeter values in the range of 1300-2000 approximately.
- Class 2 has perimeter values in the range of 800-1300 approximately.
- It depicts that Class 1 has wide range of perimeter values.
- This plot can ease manual Classification based on perimeter value ranges.

6. PLOT 6 – SCATTER PLOT

```
In [63]: #scatter
         sns.scatterplot(x=df['Area'],y=df['Perimeter'])
```

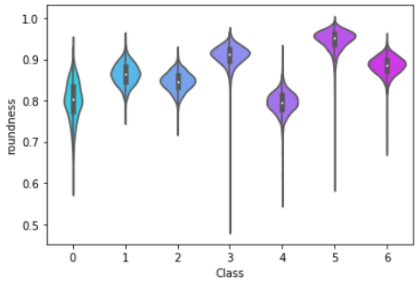
```
Out[63]: <matplotlib.axes._subplots.AxesSubplot at 0x1f3e727cbe0>
```



- Scatter Plot represents the relationship between two continuous values, respectively.
- It depicts how one data variable gets affected by the other data variable in every fraction of the value of the data set.
- The above plot shows the relationship between Area and Perimeter.
- Area and Perimeter are directly proportional to each other. If Area increases Perimeter also increases.
- When the Area of bean is 50000, its Perimeter is between 700-1000.
- When the Area is around 240000, its perimeter slighty decreased to 1790. This can be an anomaly.
- When Area is slightly beyond 250000, Perimeter is highest, close to 2000.

7. PLOT 7 – VIOLIN PLOT

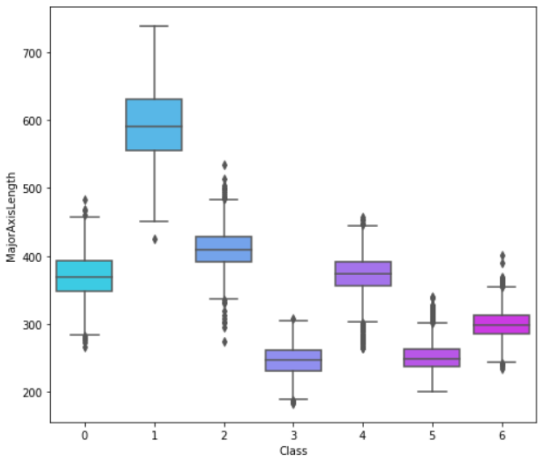
```
In [64]: #violin
         sns.violinplot(x="Class", y="roundness", data=df,palette='cool')
Out[64]: <matplotlib.axes._subplots.AxesSubplot at 0x1f3e734f828>
```



- It shows the distribution of quantitative data across several levels of one (or more) categorical variables such that those distributions can be compared.
- It is a combination of the box plot with a kernel density plot.
- This particular plot shows the distribution of roundness values of dry beans across various varieties/Classes.
- The median of roundness values of Class 5 – Seker is the highest (0.95). It can be understood that this kind of beans are more round in shape.
- The median of roundness values of Class 4 – Horoz is low (0.79). Maybe, this variety is not so round.
- The range of roundness values for Class 0 – Barbunya is around 0.58-0.95.

8. PLOT 8 – BOX PLOT

```
In [65]: #box
         plt.figure(figsize=(8,7))
         sns.boxplot(x="Class", y="MajorAxisLength",data=df, palette="cool")
Out[65]: <matplotlib.axes._subplots.AxesSubplot at 0x1f3e74e5eb8>
```

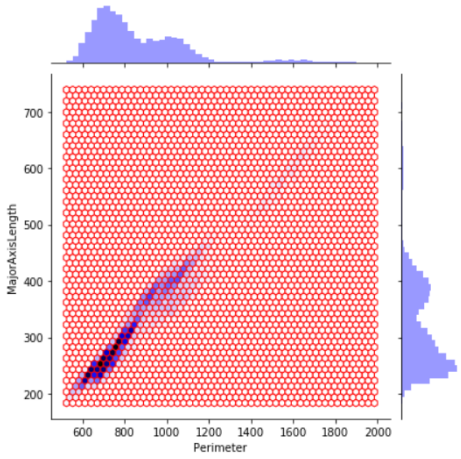


- Box plots visually show the distribution of numerical data and skewness through displaying the data quartiles (or percentiles) and averages.
- Box plots show the five-number summary of a set of data: including the minimum score, first (lower) quartile, median, third (upper) quartile, and maximum score.
- Class 1 - Bombay has maximum Major Axis Length, about 750.
- The median of Major Axis Length of Class 1 is about 590. Which means most beans of this variety have Major Axis Length around 590.
- Class 3 - Dermason have minimum Major Axis Length, less than 200.
- Median of Major Axis Length values of Class 3 is about 230.

9. PLOT 9 – JOINT PLOT

```
In [66]: #jointplot
         sns.jointplot(x='Perimeter',y='MajorAxisLength',data=df,kind='hex',dropna=True,color="blue",edgecolor="red")

Out[66]: <seaborn.axisgrid.JointGrid at 0x1f3e7617550>
```



- Seaborn also provides a nice function called jointplot which will gives a scatter plot showing the relationship between two variables along with histograms of each variable in the margins also known as a marginal plot**.**
- Not only can we see the relationships between the two variables, but also how they are distributed individually.
- This particular joint plot shows relationship between perimeter and Major Axis Length and their individual distributions.
- It depicts that Major Axis Length and Perimeter are directly proportional. As Perimeter increases, Major Axis Length also increases.
- Perimeter has dark points around 600-800 meaning high density region.
- Major Axis Length has dark points 200-300 meaning high density region.
- On an average, dry beans have Perimeter in the range from 550-1200.
- On an average, dry beans have Major Axis Length in the range from 200-480.
- It can be understood that, most beans having Perimeter around 600-800, has Major Axis Length around 200-300.

PREPARING ML MODEL

Since, we have to classify the dry beans among the seven varieties or Classes, we have used Classification Models.

The Machine Learning Algorithms we have used-

- Support Vector Model
- Logistic Classification
- Decision Tree
- Random forest

Train Test Split

- Data is prepared for training and testing. It is trained with different classification algorithms.

```
D=df.values
X=D[:,0:16]
y=D[:,16]
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2)
```

1. SUPPORT VECTOR MODEL

- TRAINING

```
from sklearn.svm import SVC
model1=SVC(C=10,kernel='rbf', gamma=0.1)
model1.fit(X_train,y_train)
```

- TESTING

```
predict=model1.predict(X_test)
print(accuracy_score(y_test,predict))
```

0.9302240176276166

```
print(confusion_matrix(y_test,predict))
```

```
[[243   0  13   0   0   1   2]
 [  0 112   0   0   0   0   0]
 [  8   0 303   0   9   0   0]
 [  0   0   0 655   0  15  31]
 [  3   0   2   3 372   0   8]
 [  1   0   0  10   1 385  15]
 [  2   0   0  55   6   5 463]]
```

```
print(classification_report(y_test,predict))
```

```
              precision    recall  f1-score   support

         0.0       0.95      0.94      0.94       259
         1.0       1.00      1.00      1.00       112
         2.0       0.95      0.95      0.95       320
         3.0       0.91      0.93      0.92       701
         4.0       0.96      0.96      0.96       388
         5.0       0.95      0.93      0.94       412
         6.0       0.89      0.87      0.88       531

    accuracy                           0.93      2723
   macro avg       0.94      0.94      0.94      2723
weighted avg       0.93      0.93      0.93      2723
```

2. LOGISTIC CLASSIFICATION

- TRAINING

```
from sklearn.linear_model import LogisticRegression
model=LogisticRegression(C=10)
model.fit(X_train,y_train)
```

- TESTING

```
predictions=model.predict(X_test)
accuracy_score(y_test,predictions)
```

0.9236136614028645

```
print(confusion_matrix(y_test,predictions))
```

```
[[264   0  16   0   1   1   8]
 [  0  97   0   0   0   0   0]
 [ 11   0 317   0   4   2   4]
 [  0   0   0 627   1  11  52]
 [  2   0   3   4 341   0  10]
 [  4   0   0   4   0 404  12]
 [  3   0   0  42   7   6 465]]
```

```
print(classification_report(y_test,predictions))
```

```
              precision    recall  f1-score   support

         0.0       0.93      0.91      0.92       290
         1.0       1.00      1.00      1.00        97
         2.0       0.94      0.94      0.94       338
         3.0       0.93      0.91      0.92       691
         4.0       0.96      0.95      0.96       360
         5.0       0.95      0.95      0.95       424
         6.0       0.84      0.89      0.87       523

    accuracy                           0.92      2723
   macro avg       0.94      0.93      0.94      2723
weighted avg       0.92      0.92      0.92      2723
```

3. DECISION TREE

- TRAINING

```
from sklearn.tree import DecisionTreeClassifier
dtree=DecisionTreeClassifier()
dtree.fit(X_train,y_train)
```

- TESTING

```
pred_dtree=dtree.predict(X_test)
accuracy_score(y_test,pred_dtree)
```

0.8894601542416453

```
print(classification_report(y_test, pred_dtree))
```

```
              precision    recall  f1-score   support

         0.0       0.88      0.92      0.90       273
         1.0       0.99      1.00      1.00       109
         2.0       0.93      0.92      0.92       333
         3.0       0.88      0.89      0.88       686
         4.0       0.93      0.92      0.92       403
         5.0       0.91      0.91      0.91       371
         6.0       0.82      0.80      0.81       548

    accuracy                           0.89      2723
   macro avg       0.91      0.91      0.91      2723
weighted avg       0.89      0.89      0.89      2723
```

```
print(confusion_matrix(y_test, pred_dtree))
```

```
[[250   1  11   0   1   3   7]
 [  0 109   0   0   0   0   0]
 [ 15   0 307   0   7   0   4]
 [  0   0   0 609   4  15  58]
 [  5   0  10   2 371   0  15]
 [  6   0   1  13   0 339  12]
 [  7   0   2  68  17  17 437]]
```

4. RANDOM FOREST

- TRAINING

```
from sklearn.ensemble import RandomForestClassifier
rfe=RandomForestClassifier()
rfe.fit(X_train,y_train)
```

- TESTING

```
pred_rfe=rfe.predict(X_test)
accuracy_score(y_test,pred_rfe)
```

0.9210429673154609

```
print(classification_report(y_test, pred_rfe))
```

	precision	recall	f1-score	support
0.0	0.95	0.94	0.94	273
1.0	0.99	1.00	1.00	109
2.0	0.94	0.95	0.94	333
3.0	0.90	0.93	0.92	686
4.0	0.95	0.94	0.95	403
5.0	0.93	0.94	0.93	371
6.0	0.87	0.84	0.86	548
accuracy			0.92	2723
macro avg	0.93	0.93	0.93	2723
weighted avg	0.92	0.92	0.92	2723

MACHINE LEARNING MODEL CHART

SERIAL NO.	ALGORITHM	ACCURACY SCORE
01	SUPPORT VECTOR MODEL	0.9302
02	LOGISTIC CLASSIFICATION	0.9236
03	RANDOM FOREST	0.9210
04	DECISION TREE	0.8895

CONCLUSION

Dry bean classification involves a high number of attributes that should be considered for accurate prediction. A major step in the prediction is processing of the data. In this project a few machine learning algorithms were used to process this data.

A user-friendly interface was designed using the MATLAB graphical user interface (GUI). Bean images obtained by computer vision system (CVS) were subjected to segmentation and feature extraction stages, and a total of 16 features; 12 dimension and 4 shape forms, were obtained from the grains. Support Vector Machine (SVM), logistic classification (LC), Decision Tree (DT), Random forest (RF) classification models were created and performance metrics were compared. It was observed that support vector had the best accuracy score.

The codes, graph/diagram and the insights gained from these helps in the prediction.

Outcomes of the different algorithms used are noted, compared and the better one is taken and hence goes the automobile price prediction.

BIBLIOGRAPHY

- https://www.sciencedirect.com/science/article/abs/pii/S0168169919311573
- https://seaborn.pydata.org/generated/seaborn.heatmap.html
- https://scikit-learn.org