**ARCHITECTURE**

**Wheat Kernel Classification**

Revision Number – 1.0

Last Date of Revision – 16-05-2023

**SHAHIN ANJUM**

**K M SUMANTH**

**Liabaries Requirment**

Flask

seaborn

pandas

numpy

scikit-learn

klib

**Content**

# **Abstract -----------------------------------------------------------------------------------4**

**Introduction ---------------------------------------------------------------------------- 4**

**Why the Architecture Design Documents? ------------------------------ 4**

**1.Architecture ------------------------------------------------------------------------ 4**

**2.Architecture Design ------------------------------------------------------------- 5**

**2.1 Data gathering from main source --------------------------------------- 5**

**2.2 Data description -------------------------------------------------------------- 5**

**2.3 Upload data inti Cassandra ----------------------------------------------- 5**

**2.4 Export data from database ----------------------------------------------- 5**

**2.5 Data pre-processing ------------------------------------------------------- 5**

**2.6 EDA ------------------------------------------------------------------------------ 6**

**2.7 Modelling ---------------------------------------------------------------------- 6**

**2.8 UI integration ---------------------------------------------------------------- 6**

**2.9 Data from user --------------------------------------------------------------- 6**

**2.10 Data validation -------------------------------------------------------------- 6**

**2.11 Rendering the results ---------------------------------------------------- 6**

**2.12 Deployment ------------------------------------------------------------------ 6**

**Abstract**

­ For this Internship, I examine the Wheat kernel classification dataset available at the UC Irvine Machine Learning Repository and Kaggle. I aim to build a model for find the variety (classification) of wheat after harvest on the bases of given attributes.

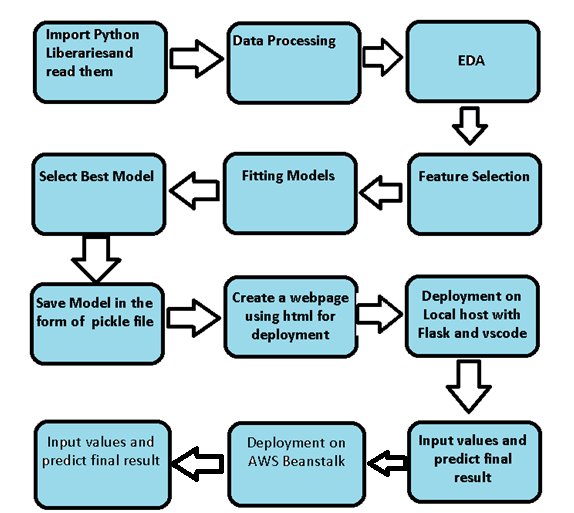
**Introduction**

Classification of wheat after harvest is a challenging task. Several techniques are prevalent for measuring the geometrical parameters of the wheat and for scrutinizing the wheat seeds. Parameters used for the classifier includes area, perimeter, compactness, length of kernel, width of kernel, asymmetry coefficient, length of kernel groove. Neural network pattern recognizer is used for the classification of the wheat seeds in this paper. The database for the Wheat seed is downloaded from the UCI machine learning repository. MSE, Accuracy Score Calculated as performance measure. Results show that Decision Tree classifier give most accuracy 95%.

Why this Architecture Design documentation?

The main objective of the Architecture design documentation is to provide the internal logic understanding prediction code. The Architecture design documentation is designed in such a way that the programmer can directly code after reading each module description in the documentation.

1 Architecture



**2 Architecture design**

This project is to create an interface for the user to know the variety(classification) of wheat with the help of given attributes.



**2.1 Data gathering from main source**

The data for the current project is being gathered from UC Irvine Machine learning Repository, the link to the data is:

https://archive.ics.uci.edu/dataset/236/seeds

**2.2 Data description**

The data was collected from UCI Machine learning repository. Soft X-ray technique was used for getting the kernel structure. 13 cm × 18 cm Xray KODAK plates were used for placing the wheat for the recording the images. The images were collected in the Institute of Agrophysics of Polish Academy of Science in Lublin. Seven geometric attributes of the wheat kernel image such as area, perimeter, compactness, kernel length, kernel width, asymmetry coefficient, kernel groove length was computed.

**2.3 Import data into Cassandra**

Created an api for the upload of the data into the Cassandra database, steps performed are:

* Connection is made with the database.
* Created a database with name Income Prediction.
* Cqlsh command is written for creating the data table with required parameters.
* And finally, a cqlsh command is written for uploading the dataset into data table by bulk insertion.
* .

2.4 Export data from database

In the above created api, the download url is also being created, which downloads the data into a csv file format.

2.5 Data pre-processing

Steps performed in pre-processing are:

* First rename all columns
* The data types are being checked and found all columns are of type integer.
* Checked for null values as there are no null values.
* Checking outliers as looking box plot, there are 5 outliers present. Deleted all outliers
* Scaling is performed for required data.

And the data is ready for passing to the machine learning algorithm.

2.6 EDA

I first plot bar graph of targets columns with all independent columns. By these graphs I concluded many things which is written in EDA Jupyter notebook.

**2.7 Modelling**

The pre-processed data is then visualized, and all the required insights are being drawn. Although from the drawn insights, the data is randomly spread but still modelling is performed with different machine learning algorithms to make sure we cover all the possibilities. And finally, as expected decision classifier performed well.

**2.8 UI integration**

Both CSS and HTML files are being created and are being integrated with the created machine learning model. All the required files are then integrated to the app.py file and tested locally. Note I copy the code from internet and make the CSS and HTML File.

**2.9 Data from user**

The data from the user is retrieved from the created HTML web page.

**2.10 Data validation**

The data provided by the user is then being processed by app.py file and validated. The validated data is then sent for the prediction.

**2.11 Rendering the results**

The data sent for the prediction is then rendered to the web page.

**3 Deployment**

I deployed on AWS beanstalk. This is link:-

6

**ScreenShot of the App Interface which I will deploy**



A white background with black text

Description automatically generated with low confidence