Educate Elevate Enlighten

JSS Mahavidyapeetha

JSS Science And Technology University

(Established Under JSS Science and Technology University Act No. 43 of 2013)

(Formerly Known as SJCE)

JSS SCIENCE AND TECHNOLOGY UNIVERSITY MYSURU

CS745 Pattern Recognition

Event - IV

Topic: Crop Prediction

Submitted to:

Dr. Srinath S

Associate Professor

Department of Computer Science

JSS S&TU, Mysuru

Submitted by:

Sl. No	USN	NAME	Roll No
1	**01JST18CS106**	**Rishitha S Ramesh**	**32**
2	**01JST18CS133**	**Shriya Mittapalli**	**39**

CS-C SECTION, 7^{th} Sem

1. Introduction

Pattern recognition is the automated recognition of patterns and regularities in data. It has applications in statistical data analysis, signal processing, image analysis, information retrieval, bioinformatics, data compression, computer graphics and machine learning. Pattern recognition uses various machine learning algorithms to find patterns in the data being operated on, classification is the process of predicting the class of a given set of data points. Classification predictive modelling is the task of approximating a mapping function, from input variables, X to discrete output variables, Y. Classification belongs to the category of supervised learning where the targets are also provided with the input data. Classification is the grouping of related facts into classes. Two of the commonly used classifiers are the Naive Bayes and the k-Nearest Neighbours classifiers.

I. K Nearest neighbours:

K-Nearest Neighbours is one of the most basic yet essential classification algorithms in Machine Learning. It belongs to the supervised learning domain and finds intense application in pattern recognition, data mining and intrusion detection.

It is an easily implementable algorithm, it stores n-dimensional training data and when classification has to be performed it compares the distance between the point that has to be classified and all the points in the dataset, the k nearest points are chosen and the majority class is assigned to the sample point.

Since the k-Nearest Neighbours classifier is a lazy learner, it does not need any training time at all. Since the classifier requires no training, new data can be easily added, which does not impact the accuracy of the classifier. Another advantage is that it is simple and easy to implement, which uses only two parameters, the value of k, and the distance function used. However, it suffers from large datasets, since it has to calculate the distance between the test sample and each point from the dataset, which requires a lot of time. The classifier also suffers when the dimensionality of the data is considerably high. The k-Nearest Neighbours classifier may also require feature scaling to provide fairly accurate results. The classifier is quite sensitive to noise in the dataset and requires manually imputing missing values and removing outliers.

II. Naive Bayes:

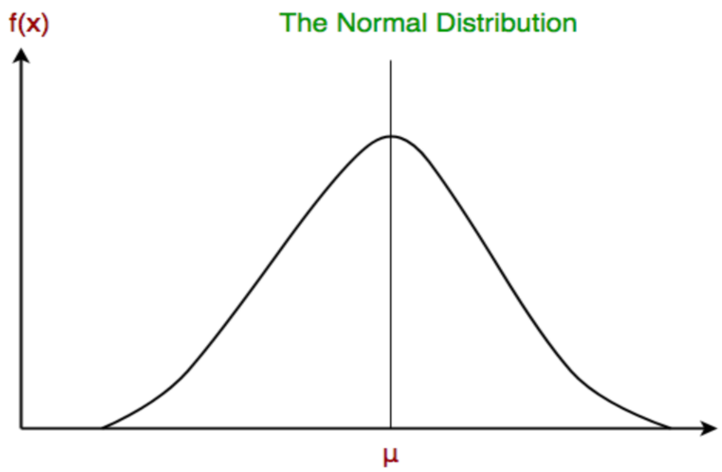
Naive Bayes is a probabilistic classification algorithm based on the Bayes theorem and the strong autonomy hypothesis.

Bayes' Theorem finds the probability of an event occurring given the probability of another event that has already occurred. Bayes' theorem is stated mathematically as the following equation:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

It is entirely dependent on the precise existence of the probability model. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other. In a supervised learning environment, this classification technique can be trained to a strong level. This algorithm has the advantage of only requiring a small amount of training data to evaluate constants such as variances and means of variables that are essential for a classifier.

In Gaussian Naive Bayes, continuous values associated with each feature are assumed to be distributed according to a Gaussian distribution. A Gaussian distribution is also called Normal distribution. When plotted, it gives a bell-shaped curve which is symmetric about the mean of the feature values as shown below:



2. Aim

To build a predictive model to recommend the most suitable crops to grow in a particular farm based on parameters.

3. Literature Survey

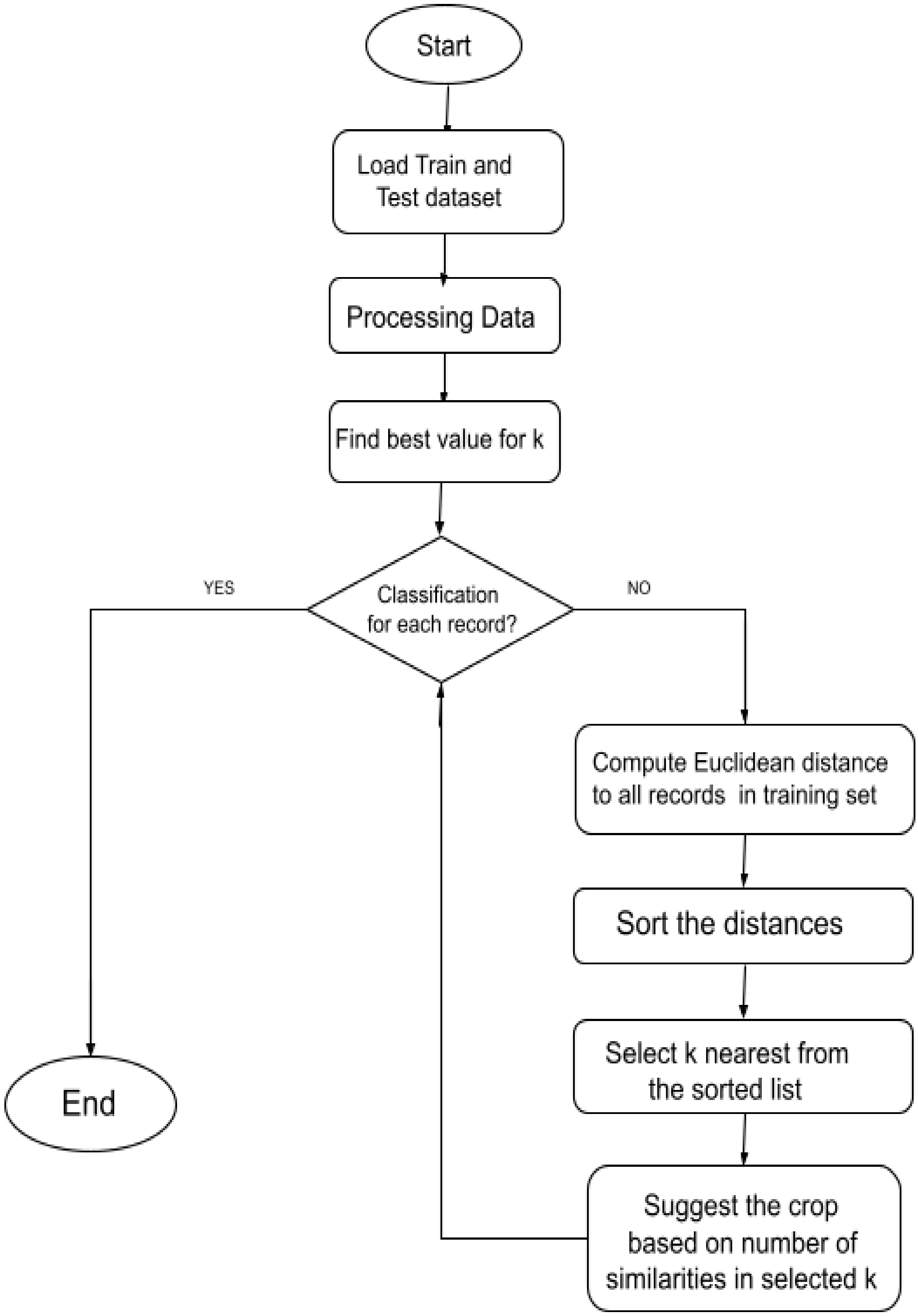
- Y. J. N. Kumar et al. [1], implemented a prediction system on crop production from the collecting of past data. Crop yield is estimated using data mining techniques. They used the Random Forest algorithm to forecast the highest yield crop as a product. Crops yield predictions are often appropriate in the agricultural sector. The higher the accuracy, the higher the benefit on the crop yield. Farmers will use the proposed technique to help them decide which crop to plant in their fields. Under this system would cover the widest range of crops possible. Farmers in India can benefit from accurate forecasting of various crops across various districts.

- [2] Reddy, D. Anantha, Bhagyashri Dadore, and Aarti Watekar. “Crop recommendation system to maximize crop yield in ramtek region using machine learning”. This proposed system worked on three parameters: soil characteristics, soil types and crop yield data collection based on these parameters suggesting the farmer suitable crop to be cultivated. This proposed system worked on different machine learning algorithms like random forest, CHAID, K-Nearest Neighbour and Naïve Bayes. By applied this proposed system we can predict particular crop under particular weather condition, state and district values. Thus our proposed work would help farmers in sowing the right seed based on soil requirements to increase productivity of the nation.”

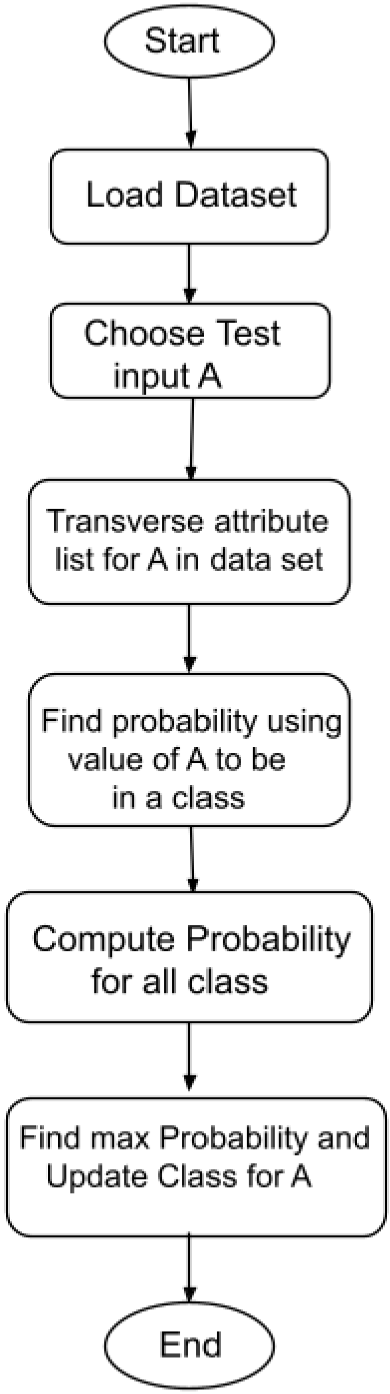
- [3] Kulkarni, Nidhi H., G. N. Srinivasan, B. M. Sagar, and N. K. Cauvery. “Improving Crop Productivity Through A Crop Recommendation System Using Ensembling Technique”. This proposed system is used for recommended the right crop based on the soil specific type and characteristics like average rainfall and the surface temperature with high accuracy. This proposed system worked on various machine learning algorithms like Random Forest, Naive Bayes, and Linear SVM. This crop recommendation system classified the input soil dataset into the recommendable crop type, Kharif and Rabi. By applying this proposed system achieved 99.91% accuracy result.”

4. Flowchart

I. K Nearest neighbours:



II. Naive Bayes:



5. Dataset

Crop Recommendation Dataset: It consists of 22 different crops each having 100 records and classified based on 7 features; Nitrogen, Phosphorus, Potassium, temperature, humidity ph, rainfall. This dataset was build by augmenting datasets of rainfall, climate and fertilizer data available for India.

- **Nitrogen -** ratio of Nitrogen content in soil
- **Phosphorus -** ratio of Phosphorous content in soil
- **Potassium -** ratio of Potassium content in soil
- **temperature -** temperature in degree Celsius
- **humidity -** relative humidity in %
- **ph -** ph value of the soil
- **rainfall -** rainfall in mm

6. Code and Result

Prediction of Agricultural Crops

Imports

In [1]:
```
import pandas as pd
import numpy as np
import math
import operator
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import cross_val_score
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore") #to remove unwanted warnings
```

Load Dataset

In [2]:
```
col=['Nitrogen','Phosphorus','Potassium ','Temperature','Humidity','ph','Rainfall','label']
crop=pd.read_csv("Crop_recommendation.csv",names=col)
print(crop.head())
```

```
   Nitrogen  Phosphorus  Potassium   Temperature   Humidity        ph  \
0        90          42          43    20.879744  82.002744  6.502985
1        85          58          41    21.770462  80.319644  7.038096
2        60          55          44    23.004459  82.320763  7.840207
3        74          35          40    26.491096  80.158363  6.980401
4        78          42          42    20.130175  81.604873  7.628473

     Rainfall label
0  202.935536  rice
1  226.655537  rice
2  263.964248  rice
3  242.864034  rice
4  262.717340  rice
```

Processing Data

In [3]:
```
print("shape:",crop.shape, "\n")
print("Size:",crop.size)
```

```
shape: (2200, 8)

Size: 17600
```

In [4]:
```
print("no of samples available for each type")
print(crop['label'].value_counts())
```

```
no of samples available for each type
rice           100
maize          100
jute           100
cotton         100
coconut        100
papaya         100
orange         100
apple          100
muskmelon      100
watermelon     100
grapes         100
mango          100
banana         100
pomegranate    100
lentil         100
blackgram      100
mungbean       100
mothbeans      100
pigeonpeas     100
kidneybeans    100
chickpea       100
coffee         100
Name: label, dtype: int64
```

In [5]:
```
print(crop.describe())
```

```
          Nitrogen   Phosphorus    Potassium  Temperature     Humidity  \
count  2200.000000  2200.000000  2200.000000  2200.000000  2200.000000
mean     50.551818    53.362727    48.149091    25.616244    71.481779
std      36.917334    32.985883    50.647931     5.063749    22.263812
min       0.000000     5.000000     5.000000     8.825675    14.258040
25%      21.000000    28.000000    20.000000    22.769375    60.261953
50%      37.000000    51.000000    32.000000    25.598693    80.473146
75%      84.250000    68.000000    49.000000    28.561654    89.948771
max     140.000000   145.000000   205.000000    43.675493    99.981876

                ph     Rainfall
count  2200.000000  2200.000000
mean      6.469480   103.463655
std       0.773938    54.958389
min       3.504752    20.211267
25%       5.971693    64.551686
50%       6.425045    94.867624
75%       6.923643   124.267508
max       9.935091   298.560117
```

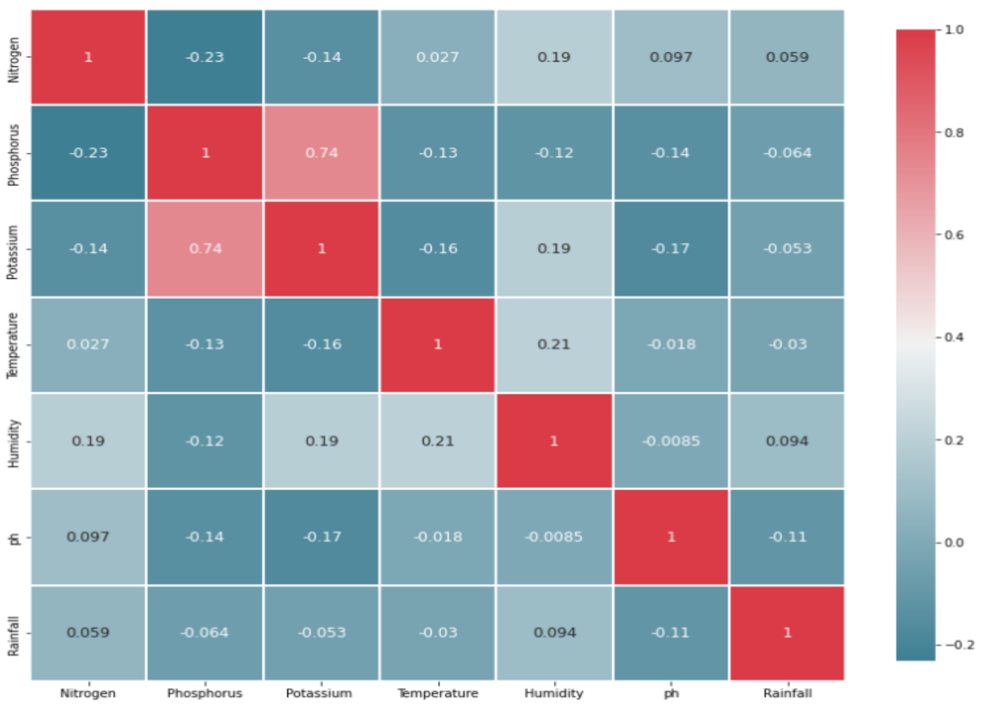
In [6]:
```
crop.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2200 entries, 0 to 2199
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Nitrogen     2200 non-null   int64
 1   Phosphorus   2200 non-null   int64
 2   Potassium    2200 non-null   int64
 3   Temperature  2200 non-null   float64
 4   Humidity     2200 non-null   float64
 5   ph           2200 non-null   float64
 6   Rainfall     2200 non-null   float64
 7   label        2200 non-null   object
dtypes: float64(4), int64(3), object(1)
memory usage: 137.6+ KB
```

```
In [7]: crop.isnull().sum()
```

```
Out[7]: Nitrogen       0
        Phosphorus     0
        Potassium      0
        Temperature    0
        Humidity       0
        ph             0
        Rainfall       0
        label          0
        dtype: int64
```

```
In [8]: def correlation(crop):
            _ , ax =plt.subplots(figsize =(14,12))
            colormap = sns.diverging_palette(220, 10, as_cmap = True)

            _ = sns.heatmap(
                crop.corr(),
                cmap = colormap,
                square = True,
                cbar_kws={'shrink':.9 },
                ax=ax,
                annot=True,
                linewidths=0.1, vmax=1.0, linecolor='white',
                annot_kws={'fontsize':12 }  )

            plt.title('Correlation of Features', y=1.05, size=15)

        correlation(crop)
```




```
In [9]: #Splitting the data
        X=crop.iloc[:,:7]#features
        y=crop.iloc[:,7:]#class labels
        X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2)
              #train=80% and test=20% data is randomly split
        print('Train Shape: {}'.format(X_train.shape))
        print('Test Shape: {}'.format(X_test.shape))
```

```
Train Shape: (1760, 7)
Test Shape: (440, 7)
```

Model

K-Nearest Neighbour Classifier

In [10]:

```
# KNN
from sklearn.neighbors import KNeighborsClassifier
```

In [11]:

```
# to find the best k using brute force
cv_scores = []
neighbors = list(np.arange(3,50,2))
for n in neighbors:
    knn = KNeighborsClassifier(n_neighbors = n,algorithm = 'brute')

    cross_val = cross_val_score(knn,X_train,y_train,cv = 5 , scoring = 'accuracy')
    cv_scores.append(cross_val.mean())

error = [1-x for x in cv_scores]
optimal_n = neighbors[ error.index(min(error)) ]
knn_optimal = KNeighborsClassifier(n_neighbors = optimal_n,algorithm = 'brute')
knn_optimal.fit(X_train,y_train)
pred = knn_optimal.predict(X_test)
acc = accuracy_score(y_test,pred)*100
print("The accuracy for optimal k = {0} using brute is {1}".format(optimal_n,acc))
```

```
The accuracy for optimal k = 7 using brute is 97.95454545454545
```

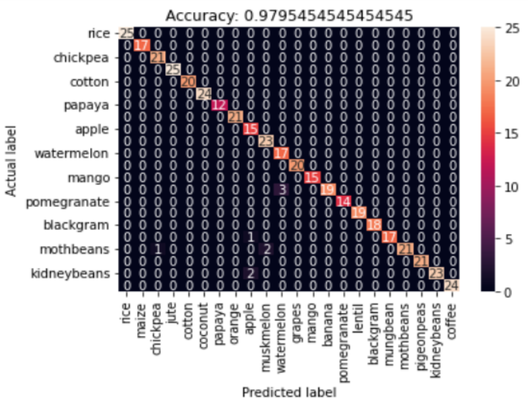
In [12]:

```
print(classification_report(y_test,pred))
```

```
              precision    recall  f1-score   support

       apple       1.00      1.00      1.00        25
      banana       1.00      1.00      1.00        17
   blackgram       0.95      1.00      0.98        21
    chickpea       1.00      1.00      1.00        25
     coconut       1.00      1.00      1.00        20
      coffee       1.00      1.00      1.00        24
      cotton       1.00      1.00      1.00        12
      grapes       1.00      1.00      1.00        21
        jute       0.83      1.00      0.91        15
 kidneybeans       0.92      1.00      0.96        23
      lentil       0.85      1.00      0.92        17
       maize       1.00      1.00      1.00        20
       mango       1.00      1.00      1.00        15
   mothbeans       1.00      0.86      0.93        22
    mungbean       1.00      1.00      1.00        14
   muskmelon       1.00      1.00      1.00        19
      orange       1.00      1.00      1.00        18
      papaya       1.00      0.94      0.97        18
  pigeonpeas       1.00      0.88      0.93        24
 pomegranate       1.00      1.00      1.00        21
        rice       1.00      0.92      0.96        25
  watermelon       1.00      1.00      1.00        24

    accuracy                           0.98       440
   macro avg       0.98      0.98      0.98       440
weighted avg       0.98      0.98      0.98       440
```

In [13]:

```
# Creates a confusion matrix
cm = confusion_matrix(y_test, pred)
# Transform to df for easier plotting
cm_df = pd.DataFrame(cm,
    index = ['rice','maize','chickpea','jute','cotton','coconut','papaya','orange','apple','muskmelon','watermelon',
 'grapes','mango','banana','pomegranate','lentil','blackgram','mungbean','mothbeans','pigeonpeas','kidneybeans','coffee'],
    columns = ['rice','maize','chickpea','jute','cotton','coconut','papaya','orange','apple','muskmelon','watermelon',
 'grapes','mango','banana','pomegranate','lentil','blackgram','mungbean','mothbeans','pigeonpeas','kidneybeans','coffee'])

sns.heatmap(cm_df, annot=True)
plt.title('Accuracy: {0}'.format(knn_optimal.score(X_test, y_test)))
plt.ylabel('Actual label')
plt.xlabel('Predicted label')
plt.show()
```



In [14]:

```
#Testing using KNN
testSet = [[85, 58, 41, 21.77046, 80.31964, 7.038096, 226.6555]]
test = pd.DataFrame(testSet)
print(test)
print("predicted:",knn_optimal.predict(test))
print("neighbors",knn_optimal.kneighbors(test))
```

```
    0   1   2         3         4         5         6
0  85  58  41  21.77046  80.31964  7.038096  226.6555
predicted: ['rice']
neighbors (array([[3.77635763e-05, 9.96382265e+00, 1.13803479e+01, 1.60337998e+01,
        1.74455770e+01, 1.77994944e+01, 1.83747400e+01]]), array([[ 919, 1010, 1080, 1312, 1389,  332,  358]], dtype=int64))
```

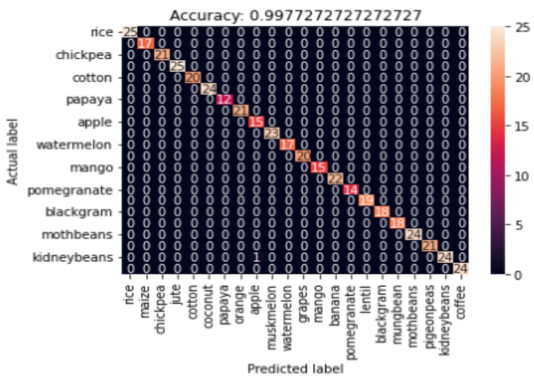
Naive Bayes Classifier

In [15]:

```
from sklearn.naive_bayes import GaussianNB
nb = GaussianNB()
nb.fit(X_train,y_train)
y_pred_nb = nb.predict(X_test)
```

In [16]:

```
# Creates a confusion matrix
cm = confusion_matrix(y_test, y_pred_nb)
# Transform to df for easier plotting
cm_df = pd.DataFrame(cm,
 index = ['rice','maize','chickpea','jute','cotton','coconut','papaya','orange','apple','muskmelon','watermelon','grapes',
        'mango','banana','pomegranate','lentil','blackgram','mungbean','mothbeans','pigeonpeas','kidneybeans','coffee'],
 columns = ['rice','maize','chickpea','jute','cotton','coconut','papaya','orange','apple','muskmelon','watermelon','grapes',
        'mango','banana','pomegranate','lentil','blackgram','mungbean','mothbeans','pigeonpeas','kidneybeans','coffee'])

sns.heatmap(cm_df, annot=True)
plt.title('Accuracy: {0}'.format(nb.score(X_test, y_test)))
plt.ylabel('Actual label')
plt.xlabel('Predicted label')
plt.show()
```



In [17]:

```
#Testing using Naive Bayes
testSet = [[11, 57, 23, 21.18857178, 19.61964599, 5.728038096, 136.9876435]]
test = pd.DataFrame(testSet)
print(test)
print("predicted:",nb.predict(test))
```

```
    0   1   2          3          4         5           6
0  11  57  23  21.188572  19.619646  5.728038  136.987643
predicted: ['kidneybeans']
```

7. Conclusion

The implementation of the system is to predict crop yield to help farmers choose the best seeds for plantation using the past data. Datasets are ordered in a well-structured manner. Two classifier algorithms namely K Nearest neighbours and Naive Bayes are used and the outcome of these techniques is compared based on accuracy. The result of the experiment showed that the Naive Bayes algorithm gets the highest accuracy value of 99.7727, while the accuracy for KNN is 97.9545. The future is bright for the implementation of machine learning algorithms in the field of crop production and using these techniques will improve the farmer's income level and the crop yield can be increased. Hence data mining techniques could be used in the agriculture sector for tacking better decisions.

8. References

[1] Kumar, Y. Jeevan Nagendra, V. Spandana, V. S. Vaishnavi, K. Neha, and V. G. R. R. Devi. "Supervised Machine learning Approach for Crop Yield Prediction in Agriculture Sector." In 2020 5th International Conference on Communication and Electronics Systems (ICCES), pp. 736-741. IEEE, 2020.

[2] Reddy, D. Anantha, Bhagyashri Dadore, and Aarti Watekar. "Crop recommendation system to maximize crop yield in ramtek region using machine learning." International Journal of Scientific Research in Science and Technology 6, no. 1 (2019): 485-489.

[3] Kulkarni, Nidhi H., G. N. Srinivasan, B. M. Sagar, and N. K. Cauvery. "Improving Crop Productivity Through A Crop Recommendation System Using Ensembling Technique." In 2018 3rd International Conference on Computational Systems and Information Technology for Sustainable Solutions (CSITSS), pp. 114-119. IEEE, 2018.

[4] https://www.kaggle.com/atharvaingle/crop-recommendation-dataset