In [27]:
```
pip install ipywidgets
```

```
Requirement already satisfied: ipywidgets in c:\users\acer\anaconda3\lib\site-package
s (7.6.5)
Requirement already satisfied: widgetsnbextension~=3.5.0 in c:\users\acer\anaconda3\l
ib\site-packages (from ipywidgets) (3.5.2)
Requirement already satisfied: jupyterlab-widgets>=1.0.0 in c:\users\acer\anaconda3\l
ib\site-packages (from ipywidgets) (1.0.0)
Requirement already satisfied: traitlets>=4.3.1 in c:\users\acer\anaconda3\lib\site-p
ackages (from ipywidgets) (5.1.1)
Requirement already satisfied: ipython-genutils~=0.2.0 in c:\users\acer\anaconda3\lib
\site-packages (from ipywidgets) (0.2.0)
Requirement already satisfied: ipykernel>=4.5.1 in c:\users\acer\anaconda3\lib\site-p
ackages (from ipywidgets) (6.15.2)
Requirement already satisfied: nbformat>=4.2.0 in c:\users\acer\anaconda3\lib\site-pa
ckages (from ipywidgets) (5.5.0)
Requirement already satisfied: ipython>=4.0.0 in c:\users\acer\anaconda3\lib\site-pac
kages (from ipywidgets) (7.31.1)
Requirement already satisfied: packaging in c:\users\acer\anaconda3\lib\site-packages
(from ipykernel>=4.5.1->ipywidgets) (21.3)
Requirement already satisfied: matplotlib-inline>=0.1 in c:\users\acer\anaconda3\lib
\site-packages (from ipykernel>=4.5.1->ipywidgets) (0.1.6)
Requirement already satisfied: jupyter-client>=6.1.12 in c:\users\acer\anaconda3\lib
\site-packages (from ipykernel>=4.5.1->ipywidgets) (7.3.4)
Requirement already satisfied: nest-asyncio in c:\users\acer\anaconda3\lib\site-packa
ges (from ipykernel>=4.5.1->ipywidgets) (1.5.5)
Requirement already satisfied: pyzmq>=17 in c:\users\acer\anaconda3\lib\site-packages
(from ipykernel>=4.5.1->ipywidgets) (23.2.0)
Requirement already satisfied: debugpy>=1.0 in c:\users\acer\anaconda3\lib\site-packa
ges (from ipykernel>=4.5.1->ipywidgets) (1.5.1)
Requirement already satisfied: psutil in c:\users\acer\anaconda3\lib\site-packages (f
rom ipykernel>=4.5.1->ipywidgets) (5.9.0)
Requirement already satisfied: tornado>=6.1 in c:\users\acer\anaconda3\lib\site-packa
ges (from ipykernel>=4.5.1->ipywidgets) (6.1)
Requirement already satisfied: prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0 in c:\use
rs\acer\anaconda3\lib\site-packages (from ipython>=4.0.0->ipywidgets) (3.0.20)
Requirement already satisfied: pygments in c:\users\acer\anaconda3\lib\site-packages
(from ipython>=4.0.0->ipywidgets) (2.11.2)
Requirement already satisfied: backcall in c:\users\acer\anaconda3\lib\site-packages
(from ipython>=4.0.0->ipywidgets) (0.2.0)
Requirement already satisfied: pickleshare in c:\users\acer\anaconda3\lib\site-packag
es (from ipython>=4.0.0->ipywidgets) (0.7.5)
Requirement already satisfied: decorator in c:\users\acer\anaconda3\lib\site-packages
(from ipython>=4.0.0->ipywidgets) (5.1.1)
Requirement already satisfied: jedi>=0.16 in c:\users\acer\anaconda3\lib\site-package
s (from ipython>=4.0.0->ipywidgets) (0.18.1)
Requirement already satisfied: setuptools>=18.5 in c:\users\acer\anaconda3\lib\site-p
ackages (from ipython>=4.0.0->ipywidgets) (63.4.1)
Requirement already satisfied: colorama in c:\users\acer\anaconda3\lib\site-packages
(from ipython>=4.0.0->ipywidgets) (0.4.5)
Requirement already satisfied: jsonschema>=2.6 in c:\users\acer\anaconda3\lib\site-pa
ckages (from nbformat>=4.2.0->ipywidgets) (4.16.0)
Requirement already satisfied: jupyter_core in c:\users\acer\anaconda3\lib\site-packa
ges (from nbformat>=4.2.0->ipywidgets) (4.11.1)
Requirement already satisfied: fastjsonschema in c:\users\acer\anaconda3\lib\site-pac
kages (from nbformat>=4.2.0->ipywidgets) (2.16.2)
Requirement already satisfied: notebook>=4.4.1 in c:\users\acer\anaconda3\lib\site-pa
ckages (from widgetsnbextension~=3.5.0->ipywidgets) (6.4.12)
Requirement already satisfied: parso<0.9.0,>=0.8.0 in c:\users\acer\anaconda3\lib\sit
e-packages (from jedi>=0.16->ipython>=4.0.0->ipywidgets) (0.8.3)
Requirement already satisfied: attrs>=17.4.0 in c:\users\acer\anaconda3\lib\site-pack
ages (from jsonschema>=2.6->nbformat>=4.2.0->ipywidgets) (21.4.0)
```

```
Requirement already satisfied: pyrsistent!=0.17.0,!=0.17.1,!=0.17.2,>=0.14.0 in c:\us
ers\acer\anaconda3\lib\site-packages (from jsonschema>=2.6->nbformat>=4.2.0->ipywidge
ts) (0.18.0)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\acer\anaconda3\lib
\site-packages (from jupyter-client>=6.1.12->ipykernel>=4.5.1->ipywidgets) (2.8.2)
Requirement already satisfied: entrypoints in c:\users\acer\anaconda3\lib\site-packag
es (from jupyter-client>=6.1.12->ipykernel>=4.5.1->ipywidgets) (0.4)
Requirement already satisfied: pywin32>=1.0 in c:\users\acer\anaconda3\lib\site-packa
ges (from jupyter_core->nbformat>=4.2.0->ipywidgets) (302)
Requirement already satisfied: nbconvert>=5 in c:\users\acer\anaconda3\lib\site-packa
ges (from notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets) (6.4.4)
Requirement already satisfied: jinja2 in c:\users\acer\anaconda3\lib\site-packages (f
rom notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets) (2.11.3)
Requirement already satisfied: argon2-cffi in c:\users\acer\anaconda3\lib\site-packag
es (from notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets) (21.3.0)
Requirement already satisfied: prometheus-client in c:\users\acer\anaconda3\lib\site-
packages (from notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets) (0.14.1)
Requirement already satisfied: terminado>=0.8.3 in c:\users\acer\anaconda3\lib\site-p
ackages (from notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets) (0.13.1)
Requirement already satisfied: Send2Trash>=1.8.0 in c:\users\acer\anaconda3\lib\site-
packages (from notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets) (1.8.0)
Requirement already satisfied: wcwidth in c:\users\acer\anaconda3\lib\site-packages
(from prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0->ipython>=4.0.0->ipywidgets) (0.2.
5)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in c:\users\acer\anaconda3\li
b\site-packages (from packaging->ipykernel>=4.5.1->ipywidgets) (3.0.9)
Requirement already satisfied: bleach in c:\users\acer\anaconda3\lib\site-packages (f
rom nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets) (4.1.0)
Requirement already satisfied: defusedxml in c:\users\acer\anaconda3\lib\site-package
s (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets) (0.7.1)
Requirement already satisfied: jupyterlab-pygments in c:\users\acer\anaconda3\lib\sit
e-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidget
s) (0.1.2)
Requirement already satisfied: nbclient<0.6.0,>=0.5.0 in c:\users\acer\anaconda3\lib
\site-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywid
gets) (0.5.13)
Requirement already satisfied: testpath in c:\users\acer\anaconda3\lib\site-packages
(from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets) (0.6.0)
Requirement already satisfied: mistune<2,>=0.8.1 in c:\users\acer\anaconda3\lib\site-
packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets)
(0.8.4)
Requirement already satisfied: pandocfilters>=1.4.1 in c:\users\acer\anaconda3\lib\si
te-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidget
s) (1.5.0)
Requirement already satisfied: beautifulsoup4 in c:\users\acer\anaconda3\lib\site-pac
kages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets) (4.
11.1)
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\acer\anaconda3\lib\site-p
ackages (from jinja2->notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets) (2.0.1)
Requirement already satisfied: six>=1.5 in c:\users\acer\anaconda3\lib\site-packages
(from python-dateutil>=2.8.2->jupyter-client>=6.1.12->ipykernel>=4.5.1->ipywidgets)
(1.16.0)
Requirement already satisfied: pywinpty>=1.1.0 in c:\users\acer\anaconda3\lib\site-pa
ckages (from terminado>=0.8.3->notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidget
s) (2.0.2)
Requirement already satisfied: argon2-cffi-bindings in c:\users\acer\anaconda3\lib\si
te-packages (from argon2-cffi->notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidget
s) (21.2.0)
Requirement already satisfied: cffi>=1.0.1 in c:\users\acer\anaconda3\lib\site-packag
es (from argon2-cffi-bindings->argon2-cffi->notebook>=4.4.1->widgetsnbextension~=3.5.
```

```
0->ipywidgets) (1.15.1)
Requirement already satisfied: soupsieve>1.2 in c:\users\acer\anaconda3\lib\site-pack
ages (from beautifulsoup4->nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.5.0->
ipywidgets) (2.3.1)
Requirement already satisfied: webencodings in c:\users\acer\anaconda3\lib\site-packa
ges (from bleach->nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidget
s) (0.5.1)
Requirement already satisfied: pycparser in c:\users\acer\anaconda3\lib\site-packages
(from cffi>=1.0.1->argon2-cffi-bindings->argon2-cffi->notebook>=4.4.1->widgetsnbexten
sion~=3.5.0->ipywidgets) (2.21)
Note: you may need to restart the kernel to use updated packages.
```

In [31]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from pandas.plotting import scatter_matrix
from sklearn import metrics
from sklearn.pipeline import Pipeline
from sklearn.feature_selection import SelectKBest, f_classif, RFE
from sklearn.preprocessing import MinMaxScaler, LabelEncoder
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.pipeline import make_pipeline
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from xgboost.sklearn import XGBClassifier
from sklearn.ensemble import GradientBoostingClassifier, RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
import ipywidgets as widgets
import warnings
warnings.filterwarnings('ignore')
PATH = './'
FILENAME = 'Crop_recommendation (1).csv'
data = pd.read_csv(PATH+FILENAME)
data.head()
data.info()
 #for which crops do we have data
crop_names = data['label'].unique()
print(crop_names)

# how many types of crops are there in the dataset
print(data['label'].unique().shape)
# how many data points do we have per crop
data['label'].value_counts()
# do we have missing data
data.isnull().sum()
# nope, all good!
# let's introduce more meaningful labels
data.rename(columns={'N':'nitrogen','P':'phosphorus','K':'potassium','label':'crop'},
data.head()
 #create variables that define what our dependent and independent variables are
features = ['nitrogen','phosphorus','potassium','temperature','humidity','ph','rainfal
target = ['crop']
# let's split the data up into features and labels
X = data[features]
y = data[target]
# test size defaults to 25% of whole dataset
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=.33)
```

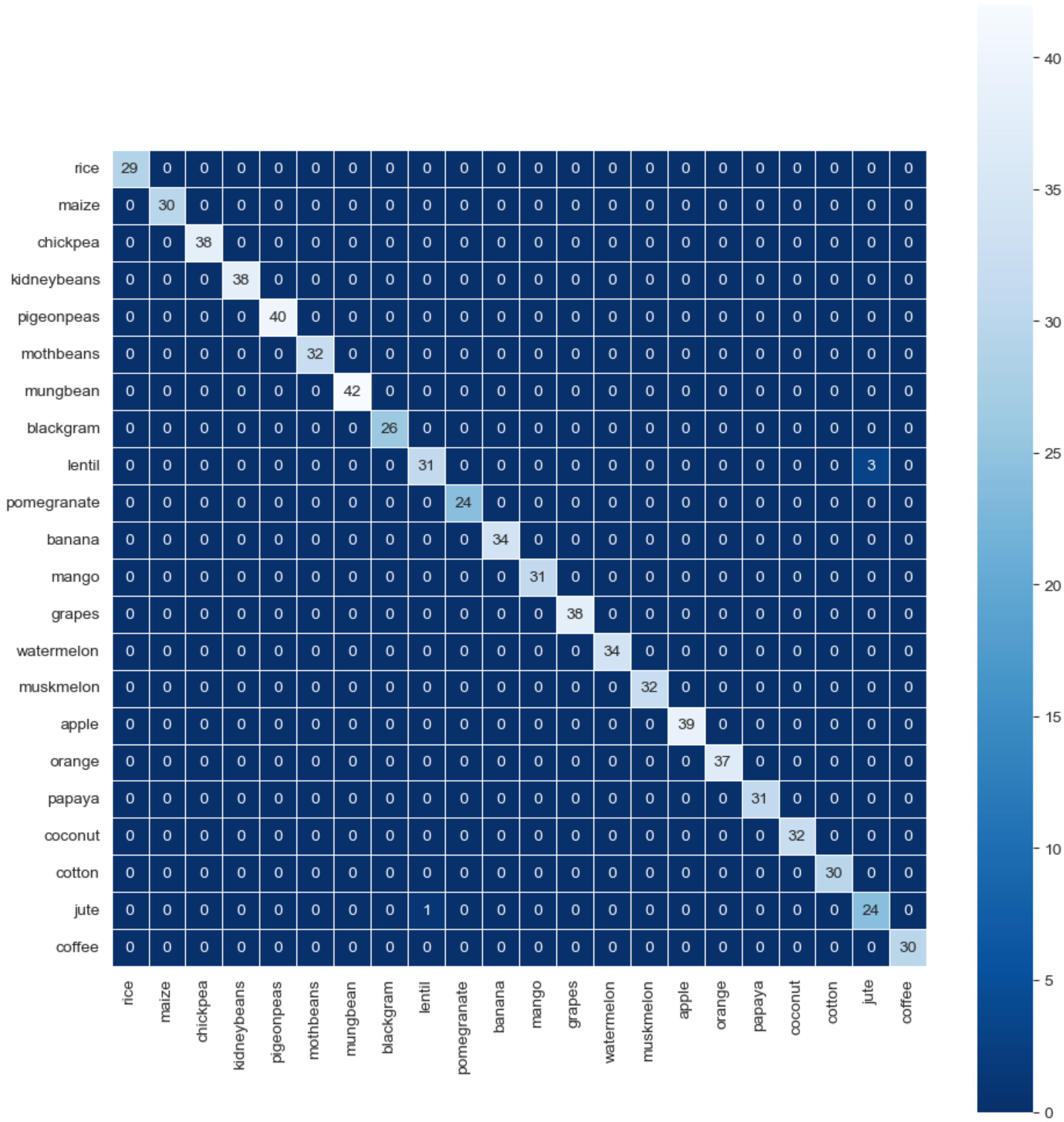
```
for ii, col in enumerate(features):
  print('{} (min,max): \t \t {:.2f} {:.2f}'.format(col,data[col].min(),data[col].max()
# scale inputs. It's important that we apply the scaling after splitting data into tra
# Otherwise, we would introduce a bias in the training, as the scaling would depend or
# in practice is not available during training
mmscaler = MinMaxScaler()
X_train = mmscaler.fit_transform(X_train)
X_test = mmscaler.transform(X_test)
# convert labels to numerical values
y_train = LabelEncoder().fit_transform(np.asarray(y_train).ravel())
y_test = LabelEncoder().fit_transform(np.asarray(y_test).ravel())
for ii, col in enumerate(features):
  print('{} (min,max): \t \t {:.2f} {:.2f}'.format(col,X_train[:,ii].min(),X_train[:,i
 #define our model. This is a one-liner, thanks to the powerful machine learning libra
model = RandomForestClassifier()
# fit the model to the training data
model.fit(X_train,y_train)
# get predictions on the test data
y_pred=model.predict(X_test)
# print training and test accuracy
print('Training Accuracy: {:.2f}%, Test Accuracy: {:.2f}%'.format(metrics.accuracy_sco
from sklearn.metrics import confusion_matrix
y_pred=model.predict(X_test)
metrics.accuracy_score(y_test,y_pred)
plt.figure(figsize=(10,10))
sns.heatmap(confusion_matrix(y_pred,y_test),square=True,cmap='Blues_r',annot=True,fmt=
ax = plt.gca()
_ = ax.set_xticklabels(crop_names,rotation='vertical')
_ = ax.set_yticklabels(crop_names,rotation='horizontal')
plt.tight_layout()
print(metrics.classification_report(y_pred,y_test))
from sklearn import tree

# fit a smaller forest with a maximum depth of 3 (this is how many consecutive
# decision the algorithm can make). As a consequence, the accuracy will be lower
# but it'll be easier to visualise it
small_rf = RandomForestClassifier(max_depth=3)
# fit the forest to the training data
small_rf.fit(X_train,y_train)
# get predictions on the test data
y_pred=small_rf.predict(X_test)
# print training and test accuracy
print('Training Accuracy: {:.2f}%, Test Accuracy: {:.2f}%'.format(metrics.accuracy_sco


# obtain list of decision trees
trees = small_rf.estimators_
# how many are there
print('there are {n} trees in the forest'.format(n=len(trees)))

# visualise the first tree
fig, axes = plt.subplots(nrows = 1,ncols = 1,figsize = (20,20),dpi=300)
tree.plot_tree(trees[0],
               feature_names = features,
               class_names=crop_names,
               filled = True);
import ipywidgets as widgets
def get_predictions(x1,x2,x3,x4,x5,x6,x7):
    feature = mmscaler.transform(np.asarray([x1,x2,x3,x4,x5,x6,x7]).reshape((1,-1)))
    croptoplant = crop_names[model.predict(feature).item()]
```

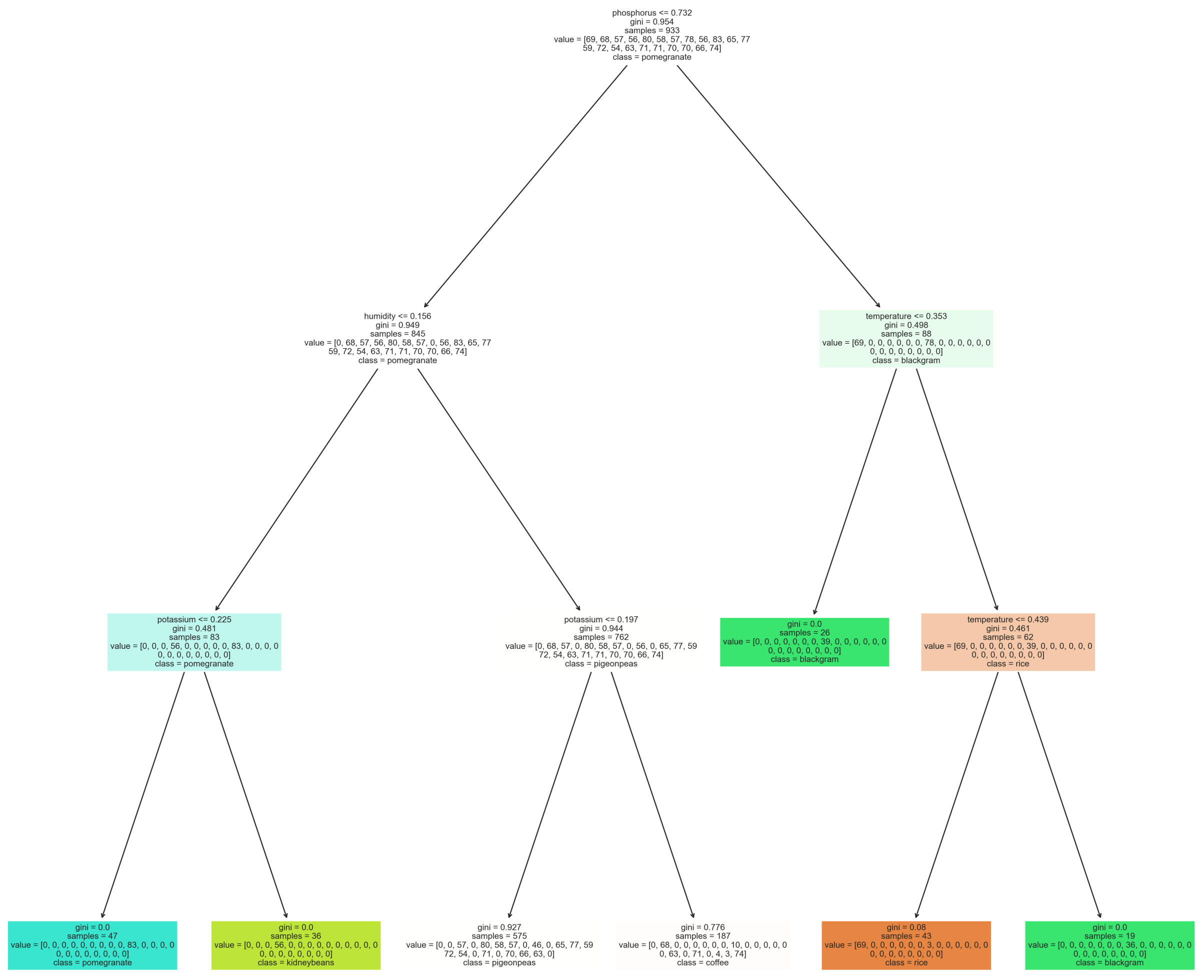
```
    print('{} should grow very well under these conditions'.format(croptoplant.upper()
N = widgets.FloatSlider(min=0.0, max=140.0, value=25.0, step=2.5, description="Nitroge
P = widgets.FloatSlider(min=5.0, max=145.0, value=25.0, step=2.5, description="Phospho
K = widgets.FloatSlider(min=5.0, max=205.0, value=25.0, step=2.5, description="Potassi
temp = widgets.FloatSlider(min=10.0, max=44.0, value=25.0, step=2.5, description="Temp
hum = widgets.FloatSlider(min=15.0, max=99.0, value=25.0, step=2.5, description="humid
ph = widgets.FloatSlider(min=3.5, max=9.9, value=5.0, step=.5, description="pH")
rain = widgets.FloatSlider(min=20.0, max=298.0, value=25.0, step=2.5, description="Rai

im = widgets.interact_manual(get_predictions,x1=N,x2=P,x3=K,x4=temp,x5=hum,x6=ph,x7=ra
_ = im.widget.children[-2].description = 'get prediction'
_ = im.widget.children[-2].style.button_color='lightgreen'

display(im)

selector = SelectKBest(score_func=f_classif,k='all')
X_train_kbest = selector.fit_transform(X_train,np.asarray(y_train).ravel())
scores = selector.scores_


X_test_kbest = selector.transform(X_test)
mask = selector.get_support() #list of booleans
new_features = []
scores = scores[mask==True]
for bool, feature in zip(mask, features):
    if bool:
        new_features.append(feature)

_ = sns.barplot(x=new_features,y=scores,log=True)
plt.ylabel('Univariate score')
plt.xlabel('predictor' )
_= plt.xticks(ticks=np.arange(X_train_kbest.shape[-1]),labels=new_features,rotation='v
selector = SelectKBest(score_func=f_classif,k=5)
X_train_kbest = selector.fit_transform(X_train,np.asarray(y_train).ravel())
scores = selector.scores_


X_test_kbest = selector.transform(X_test)
mask = selector.get_support() #list of booleans
new_features = []
scores = scores[mask==True]
for bool, feature in zip(mask, features):
    if bool:
        new_features.append(feature)

# check if it dropped temperature and ph
print(new_features)

# run training and test on reduced dataset
model_reduced = RandomForestClassifier()
model_reduced.fit(X_train[:,mask],y_train)

# print training and test accuracy
print('all features: Training Accuracy: {:.2f}%, Test Accuracy: {:.2f}%'.format(metric
print('fewer features: Training Accuracy: {:.2f}%, Test Accuracy: {:.2f}%'.format(metr
models = []
models.append(('LogisticRegression',LogisticRegression(max_iter=5000)))
models.append(('DecisionTreeClassifier',DecisionTreeClassifier()))
models.append(('XGBClassifier',XGBClassifier(use_label_encoder=False,eval_metric='mlog
models.append(('GradientBoostingClassifier',GradientBoostingClassifier()))
```

```
models.append(('RandomForestClassifier',RandomForestClassifier()))
models.append(('KNeighborsClassifier',KNeighborsClassifier()))
models.append(('GaussianNB',GaussianNB()))
models.append(('SVM',SVC()))

# same as above, but in cross-validation
nfolds = 5
print('{} fold cv'.format(nfolds))
X_cv = np.asarray(X)
y_cv = LabelEncoder().fit_transform(np.asarray(y).ravel())

for name,model in models:
    # apply transformation to each individual fold
    pipeline = Pipeline([('transformer', MinMaxScaler()), ('estimator', model)])
    scores = cross_val_score(pipeline, X_cv,y_cv , cv=nfolds)
    print(name, np.round(scores.mean(),3))
```

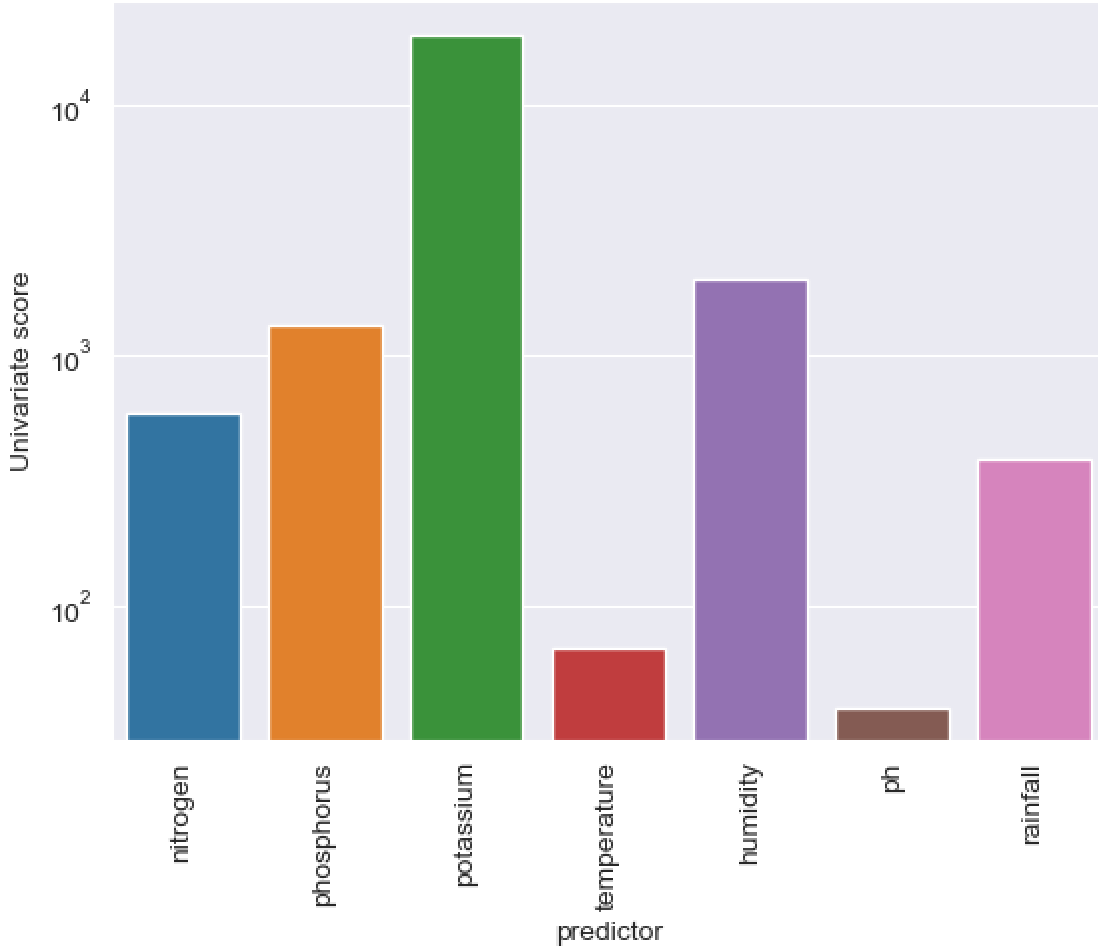
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2200 entries, 0 to 2199
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   N            2200 non-null   int64  
 1   P            2200 non-null   int64  
 2   K            2200 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
['rice' 'maize' 'chickpea' 'kidneybeans' 'pigeonpeas' 'mothbeans'
 'mungbean' 'blackgram' 'lentil' 'pomegranate' 'banana' 'mango' 'grapes'
 'watermelon' 'muskmelon' 'apple' 'orange' 'papaya' 'coconut' 'cotton'
 'jute' 'coffee']
(22,)
nitrogen (min,max):             0.00 140.00
phosphorus (min,max):           5.00 145.00
potassium (min,max):            5.00 205.00
temperature (min,max):          8.83 43.68
humidity (min,max):             14.26 99.98
ph (min,max):           3.50 9.94
rainfall (min,max):             20.21 298.56
nitrogen (min,max):             0.00 1.00
phosphorus (min,max):           0.00 1.00
potassium (min,max):            0.00 1.00
temperature (min,max):          0.00 1.00
humidity (min,max):             0.00 1.00
ph (min,max):           0.00 1.00
rainfall (min,max):             0.00 1.00
Training Accuracy: 100.00%, Test Accuracy: 99.45%
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        29
           1       1.00      1.00      1.00        30
           2       1.00      1.00      1.00        38
           3       1.00      1.00      1.00        38
           4       1.00      1.00      1.00        40
           5       1.00      1.00      1.00        32
           6       1.00      1.00      1.00        42
           7       1.00      1.00      1.00        26
           8       0.97      0.91      0.94        34
           9       1.00      1.00      1.00        24
          10       1.00      1.00      1.00        34
          11       1.00      1.00      1.00        31
          12       1.00      1.00      1.00        38
          13       1.00      1.00      1.00        34
          14       1.00      1.00      1.00        32
          15       1.00      1.00      1.00        39
          16       1.00      1.00      1.00        37
          17       1.00      1.00      1.00        31
          18       1.00      1.00      1.00        32
          19       1.00      1.00      1.00        30
          20       0.89      0.96      0.92        25
          21       1.00      1.00      1.00        30
```

```
    accuracy                           0.99       726
   macro avg       0.99      0.99      0.99       726
weighted avg       0.99      0.99      0.99       726

Training Accuracy: 90.91%, Test Accuracy: 90.22%
there are 100 trees in the forest
```






```
interactive(children=(FloatSlider(value=25.0, description='Nitrogen', max=140.0, step
=2.5), FloatSlider(value=…
<function __main__.get_predictions(x1, x2, x3, x4, x5, x6, x7)>
['nitrogen', 'phosphorus', 'potassium', 'humidity', 'rainfall']
all features: Training Accuracy: 100.00%, Test Accuracy: 99.45%
fewer features: Training Accuracy: 100.00%, Test Accuracy: 99.31%
5 fold cv
LogisticRegression 0.94
DecisionTreeClassifier 0.99
XGBClassifier 0.992
GradientBoostingClassifier 0.991
RandomForestClassifier 0.993
KNeighborsClassifier 0.98
GaussianNB 0.995
SVM 0.983
```


10^4
10^3
10^2
Univariate score
nitrogen
phosphorus
potassium
temperature
humidity
ph
rainfall
predictor

In [ ]: