```
# coding: utf-8

# Develop a program to implement Random Forest classifier model and analyze the model using confusion matrix

# In[65]:


import pandas as pd
import numpy as np
import seaborn as sn
import matplotlib.pyplot as plt
iris_df = pd.read_csv('Iris_data_sample.csv',header =None)
iris_df.info()


# In[66]:


iris_df.head()


# In[67]:


iris_df = iris_df.replace(to_replace =['??', '###'],value =None)


# In[68]:


iris_df.dropna(axis = 0, how='any',inplace = True)


# In[69]:


iris_df.info()


# In[70]:


iris_df.iloc[:,5].value_counts()


# In[71]:


X_features = iris_df.iloc[:,1:5]

X_features


# In[72]:


Y_features = iris_df.iloc[:,5]
```

```
Y_features


# In[73]:


from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X_features, Y_features,test_size =
0.25,random_state = 42 )


# In[74]:


from sklearn.ensemble import RandomForestClassifier
clf = RandomForestClassifier(max_depth=15,n_estimators=20,max_features = 'auto')


# In[75]:


clf.fit(X_train,y_train)


# In[76]:


y_pred=clf.predict(X_test)


# In[77]:


from sklearn import metrics
# Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))


# In[82]:


def draw_cm( actual, predicted ):
    ## Cret
    cm = metrics.confusion_matrix( actual, predicted)
    sn.heatmap(cm, annot=True, fmt='.2f' )
    #plt.ylabel('True label')
    #plt.xlabel('Predicted label')
    plt.show()


# In[83]:



draw_cm( y_test, y_pred )


# In[85]:
```

```
print( metrics.classification_report( y_test, y_pred ) )
```