In [3]:
```
import pandas as pd
import numpy as np

df = pd.read_csv('/Users/juanrquilesjr/Downloads/UCI_MachineLearningDataSets/connect-4.data', names = ['a1', 'a2
       'b5', 'b6', 'c1', 'c2', 'c3', 'c4', 'c5', 'c6', 'd1', 'd2',
       'd3', 'd4', 'd5', 'd6', 'e1', 'e2', 'e3', 'e4', 'e5',
       'e6', 'f1', 'f2', 'f3', 'f4', 'f5', 'fg', 'g1', 'g2',
       'g3', 'g4', 'g5', 'g6', 'Target'])
pd.set_option('display.max_columns', 45)

df.index = df.index + 1
df.head(5)
```

Out[3]:

	a1	a2	a3	a4	a5	a6	b1	b2	b3	b4	b5	b6	c1	c2	c3	c4	c5	c6	d1	d2	d3	d4	d5	d6	e1	e2	e3	e4	e5	e6	f1	f2	f3	f4	f5	f
1	b	b	b	b	b	b	b	b	b	b	b	b	x	o	b	b	b	b	x	o	x	o	x	o	b	b	b	b	b	b	b	b	b	b	b	
2	b	b	b	b	b	b	b	b	b	b	b	b	x	b	b	b	b	b	x	o	x	o	x	o	o	b	b	b	b	b	b	b	b	b	b	
3	b	b	b	b	b	b	o	b	b	b	b	b	x	b	b	b	b	b	x	o	x	o	x	o	b	b	b	b	b	b	b	b	b	b	b	
4	b	b	b	b	b	b	b	b	b	b	b	b	x	b	b	b	b	b	x	o	x	o	x	o	b	b	b	b	b	b	o	b	b	b	b	
5	o	b	b	b	b	b	b	b	b	b	b	b	x	b	b	b	b	b	x	o	x	o	x	o	b	b	b	b	b	b	b	b	b	b	b	

In [4]:
```
# importing libraries

from sklearn.tree import DecisionTreeClassifier # Import Decision Tree Classifier
from sklearn.model_selection import train_test_split # Import train_test_split function
from sklearn import metrics #Import scikit-learn metrics module for accuracy calculation
from sklearn import preprocessing #Import label encoder
```

In [5]:
```
#initializing label encoder and encoding df string values to numeric

le = preprocessing.LabelEncoder()
df_encoded = df.apply(le.fit_transform)

#splitting data into feature and target sets
features = df_encoded.drop(['Target'], axis = 1).values # drop target columns to create the features data set
target = df_encoded['Target'].values
```

In [6]:
```
print(features.shape)
print(target.shape)
```

```
(67557, 42)
(67557,)
```

In [7]:
```
#splitting data into training and test sets

X_train, X_test, y_train, y_test = train_test_split(features, target, test_size=0.3, random_state=1)
```

In [8]:
```
#Create Decision Tree classifer object (Information Gain)
clf = DecisionTreeClassifier(criterion = 'entropy', splitter = 'random', max_depth = 6)

# Train Decision Tree Classifer
clf = clf.fit(X_train,y_train)
```

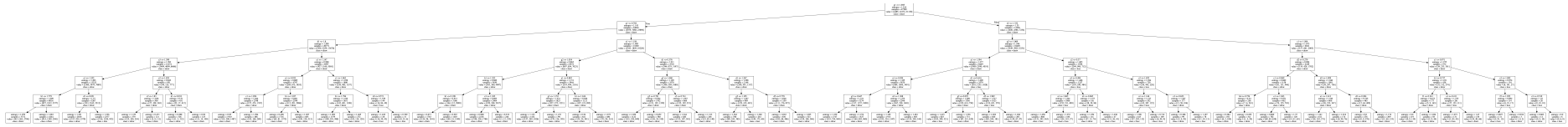
In [9]:
```
#Predict the response for test dataset
y_pred = clf.predict(X_test)
```

In [10]:
```
#Model metrics

from sklearn.metrics import classification_report, confusion_matrix

print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred, target_names= ['win', 'loss', 'draw']))
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
[[    0   341  1623]
 [    0  1559  3402]
 [    0   978 12365]]
              precision    recall  f1-score   support

         win       0.00      0.00      0.00      1964
        loss       0.54      0.31      0.40      4961
        draw       0.71      0.93      0.80     13343

   micro avg       0.69      0.69      0.69     20268
   macro avg       0.42      0.41      0.40     20268
weighted avg       0.60      0.69      0.63     20268

Accuracy: 0.6869942766923228

/Users/juanrquilesjr/anaconda3/lib/python3.6/site-packages/sklearn/metrics/classification.py:1143: UndefinedMe
tricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples.
  'precision', 'predicted', average, warn_for)
```

In [11]:
```
#Visualization of Decision Tree

from sklearn.tree import export_graphviz
from sklearn import tree
import pydotplus
from IPython.display import Image

feature_names = ['a1', 'a2', 'a3', 'a4', 'a5', 'a6', 'b1', 'b2', 'b3', 'b4',
       'b5', 'b6', 'c1', 'c2', 'c3', 'c4', 'c5', 'c6', 'd1', 'd2',
       'd3', 'd4', 'd5', 'd6', 'e1', 'e2', 'e3', 'e4', 'e5',
       'e6', 'f1', 'f2', 'f3', 'f4', 'f5', 'fg', 'g1', 'g2',
       'g3', 'g4', 'g5', 'g6']

# Create DOT data
dot_data = tree.export_graphviz(clf, out_file=None,
                                feature_names= feature_names,
                                class_names= ['win', 'loss', 'draw'])

# Draw graph
graph = pydotplus.graph_from_dot_data(dot_data)

# Show graph
Image(graph.create_png())
```

Out[11]:



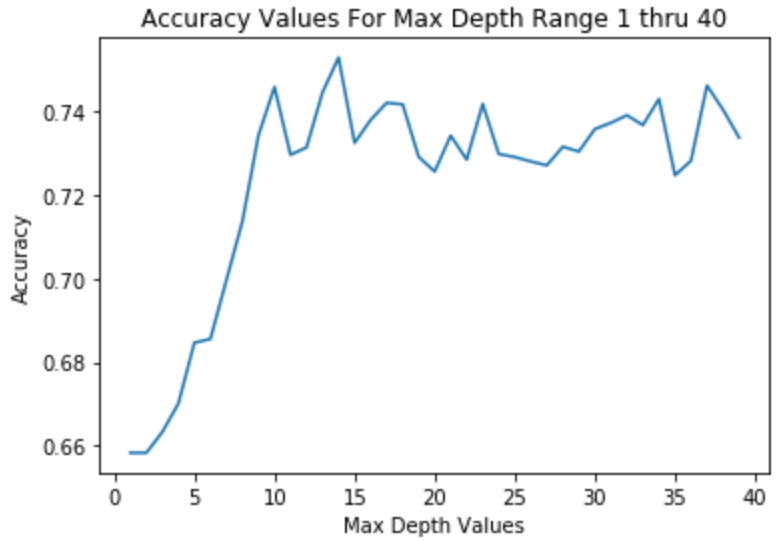
Identifying Max Depth

In [18]:
```
# Creating list of Accuracy scores for range of max depth values

k_range = range(1,40)
accuracy = []

for k in k_range:
    clf = DecisionTreeClassifier(criterion = 'entropy', splitter = 'random', max_depth = k)
    clf = clf.fit(X_train,y_train)
    y_pred = clf.predict(X_test)
    accuracy_scores = metrics.accuracy_score(y_test, y_pred)
    accuracy.append(accuracy_scores)

print(accuracy)
```

```
[0.6583283994474047, 0.6583283994474047, 0.6633116242352477, 0.6701203868166568, 0.6846753503059009, 0.6855634
497730413, 0.6996743635287153, 0.7137852772843892, 0.7343102427471877, 0.7458555358200119, 0.7296230511150582,
0.7313992500493388, 0.7445727254785869, 0.7529109926978488, 0.7324847049536215, 0.7379613183343201, 0.74205644
36550227, 0.7417110716400237, 0.7291296625222025, 0.7256266035129267, 0.7342115650286165, 0.72848825735149, 0.
7418097493585948, 0.7298204065522005, 0.7290803236629169, 0.7280442076179199, 0.727106769291494, 0.73154726662
71956, 0.7304124728636274, 0.7357904085257548, 0.7372705743043221, 0.7390961120978883, 0.7367278468521807, 0.7
429938819814486, 0.7247385040457864, 0.7281922241957766, 0.7462009078350108, 0.7404282612985987, 0.73376751529
50464]
```

In [20]:
```
import matplotlib.pyplot as plt
%matplotlib inline

plt.plot(k_range, accuracy)
plt.title('Accuracy Values For Max Depth Range 1 thru 40')
plt.xlabel('Max Depth Values')
plt.ylabel('Accuracy')
```

Out[20]:
```
Text(0, 0.5, 'Accuracy')
```



Gini Decision Tree Classifier

In [71]:
```
#Create Decision Tree classifer object (Gini)

clf = DecisionTreeClassifier(criterion = 'gini', splitter = 'random', max_depth = 6)
clf = clf.fit(X_train, y_train)
```

In [72]:
```
y_pred = clf.predict(X_test)
```

In [75]:
```
#Model metrics

from sklearn.metrics import classification_report, confusion_matrix

print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred, target_names= ['win', 'loss', 'draw']))
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
[[    0   314  1650]
 [    0  1574  3387]
 [    0   903 12440]]
              precision    recall  f1-score   support

         win       0.00      0.00      0.00      1964
        loss       0.56      0.32      0.41      4961
        draw       0.71      0.93      0.81     13343

   micro avg       0.69      0.69      0.69     20268
   macro avg       0.43      0.42      0.40     20268
weighted avg       0.61      0.69      0.63     20268

Accuracy: 0.6914347740280244
```