In [ ]:
```
import numpy as np
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree

import matplotlib.pyplot as plt
from sklearn.tree import export_graphviz
from IPython.display import Image
from subprocess import call
```

In [ ]:
```
iris = load_iris()

x = iris.data[:, ]
y = iris.target

r = 4
X_train, X_test, y_train, y_test = train_test_split(x, y,test_size=0.25, random_state=r)
```

In [ ]:
```
unpruned = DecisionTreeClassifier(max_depth=None, random_state=r)
unpruned.fit(X_train, y_train)
print('performance without', unpruned.score(X_test, y_test))
print('at depth', unpruned.tree_.max_depth)
```

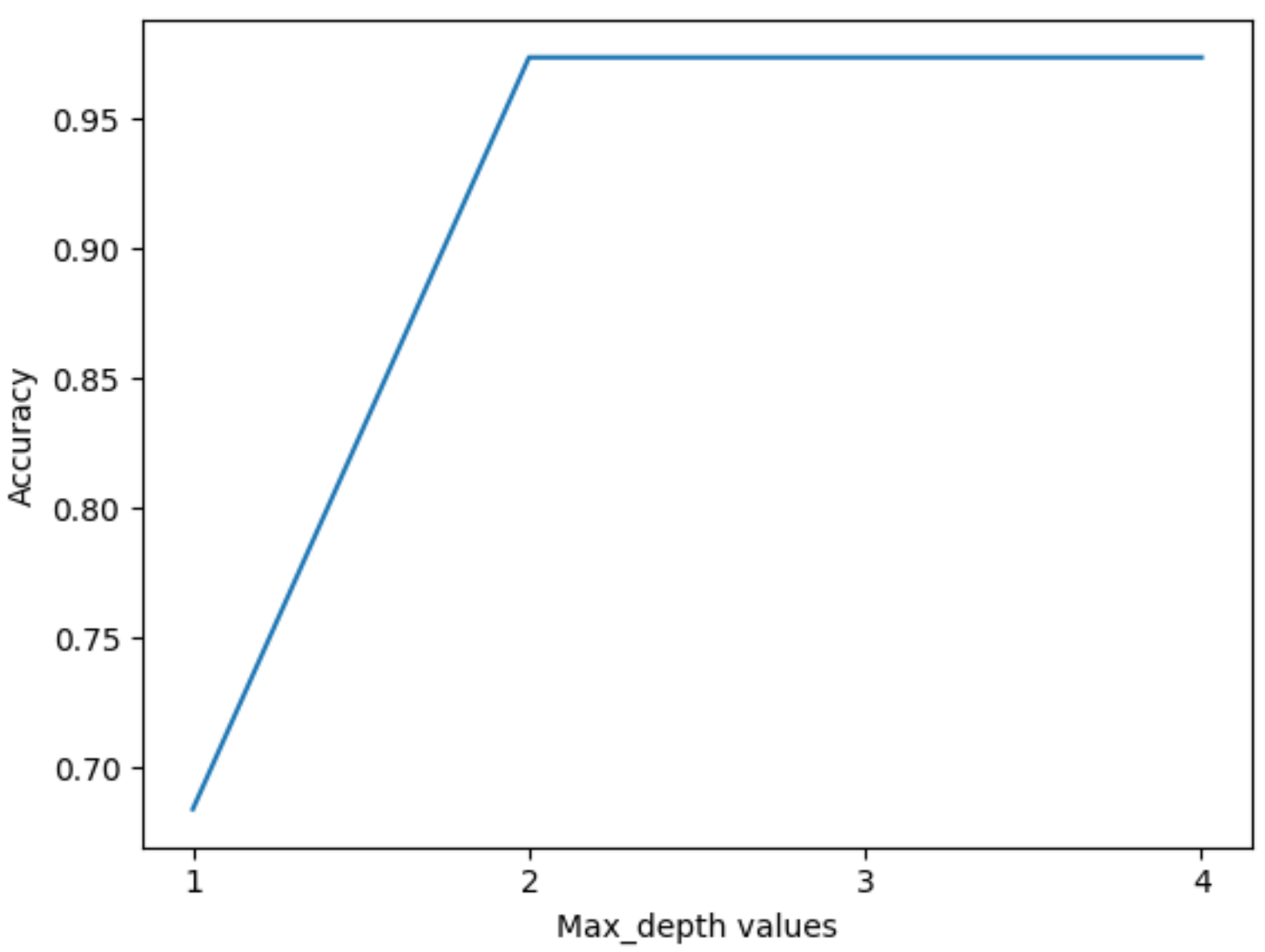
```
performance without 0.9736842105263158
at depth 4
```

In [ ]:
```
pruned_depths = range(1, unpruned.tree_.max_depth+1)

pruned_scores = []
for d in pruned_depths:
    clf = DecisionTreeClassifier(max_depth=d, random_state=r)
    clf.fit(X_train, y_train)  # Change 'fir' to 'fit'
    score = clf.score(X_test, y_test)
    pruned_scores.append(score)


fig, ax = plt.subplots()
ax.plot(pruned_depths, pruned_scores)
plt.xlabel('Max_depth values')
plt.ylabel('Accuracy')
ax.xaxis.set_ticks(pruned_depths)
plt.show
```

Out[ ]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



Best depth found at layer 2, lets prune at layer 2!

Lowest index of the highest score

In [ ]:
```
print(pruned_scores)
best_depth = pruned_depths[np.argmax(pruned_scores)]
print('Best performance reached at depth of:', best_depth)

pruned = DecisionTreeClassifier(max_depth=best_depth)
pruned.fit(x, y)
```

```
[0.6842105263157895, 0.9736842105263158, 0.9736842105263158, 0.9736842105263158]
Best performance reached at depth of: 2
```

Out[ ]:
```
DecisionTreeClassifier(max_depth=2)
```

In [ ]: