In this lab, we will practice for Naive Bayes, Cross Validation, KNN, Entropy and Decision Tree

In [15]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import sklearn as skl
import seaborn as sb
from statistics import mean
```

We will use the dataset "Naive-Bayes-Classification-Data.csv" to practice Naive Bayes classifier in python. There are 3 columns in the dataset, glucose, blood pressure and whether have diabetes. You will use glucose and blood pressure to predict diabetes.

In [4]:
```
Diabetes = pd.read_csv(filepath_or_buffer='Naive-Bayes-Classification-Data.c
Diabetes.head(10)
```

Out[4]:

	glucose	bloodpressure	diabetes
0	40	85	0
1	40	92	0
2	45	63	1
3	45	80	0
4	40	73	1
5	45	82	0
6	40	85	0
7	30	63	1
8	65	65	1
9	45	82	0

1. Seperate the original dataset to training set(70%) and testing set(30%), and train the Naive Bayes model with training data and test with the testing data, print out the accuracy rate. (10)

In [11]:
```
#split the dataset
from sklearn.model_selection import train_test_split
x= Diabetes.drop(['diabetes'], axis=1)
y= Diabetes.diabetes.values
x_train,x_test,y_train,y_test = train_test_split(x,y, test_size=0.3)
```

In [12]:
```
from sklearn.naive_bayes import GaussianNB
from sklearn import metrics
model= GaussianNB()
model.fit(x_train,y_train)
y_pred= model.predict(x_test)
print("ACC:", metrics.accuracy_score(y_test, y_pred))
```

```
ACC: 0.9230769230769231
```

1. Instead of seperate training set and testing set, use K-fold cross validation to get the accuracy rate. Print all 10 accuracy score. (k = 10) (10)

In [18]:
```
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
k_fold = KFold(10)
print ("Average Acc: ", mean(cross_val_score(model,x , y, cv=k_fold)))
```

```
Average Acc:  0.9335252525252525
```

1. Explain whether we need to use Cross Validation to test the model.(10)

Answer here

1. Use the same dataset. Train a KNN model with the data.(k = 10). Print the ACC score with K-fold cross validation, k = 10.(10)

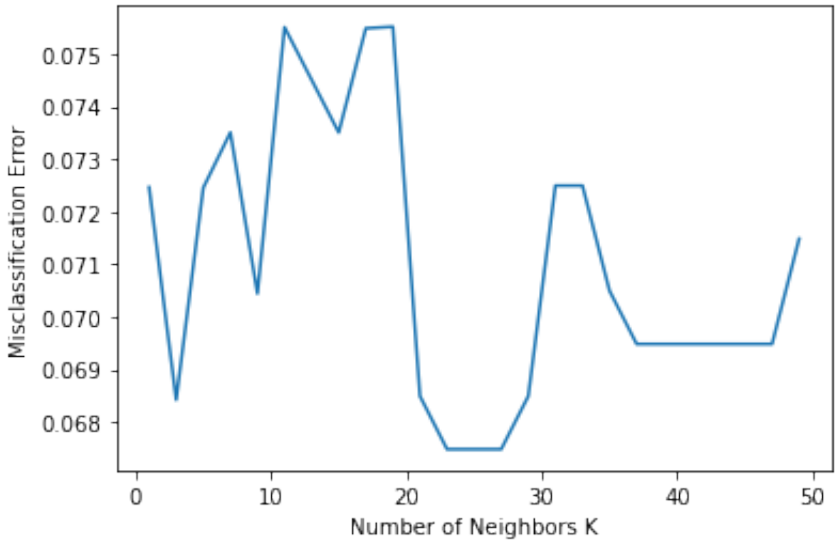
In [20]:
```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=10)
k_fold = KFold(10)
print ("Average Acc: ", mean(cross_val_score(knn,x, y, cv=k_fold)))
```

```
Average Acc:  0.9285353535353535
```

1. Next, plot misclassification error vs neighbors, and find the best k and print it's acc score.(10)

In [23]:
```
neighbors = list(range(1, 50, 2))
cv_scores = []
for k in neighbors:
    knn = KNeighborsClassifier(n_neighbors=k)
    scores = cross_val_score(knn, x, y, cv=10,scoring='accuracy')
    cv_scores.append(scores.mean())
mse = [1 - x for x in cv_scores]
optimal_k = neighbors[mse.index(min(mse))]
print("The optimal number of neighbors is {}".format(optimal_k),
      "Its ACC score is: {}".format(cv_scores[mse.index(min(mse))]))
plt.plot(neighbors, mse)
plt.xlabel("Number of Neighbors K")
plt.ylabel("Misclassification Error")
plt.show()
```

```
The optimal number of neighbors is 23 Its ACC score is: 0.9325151515151514
```



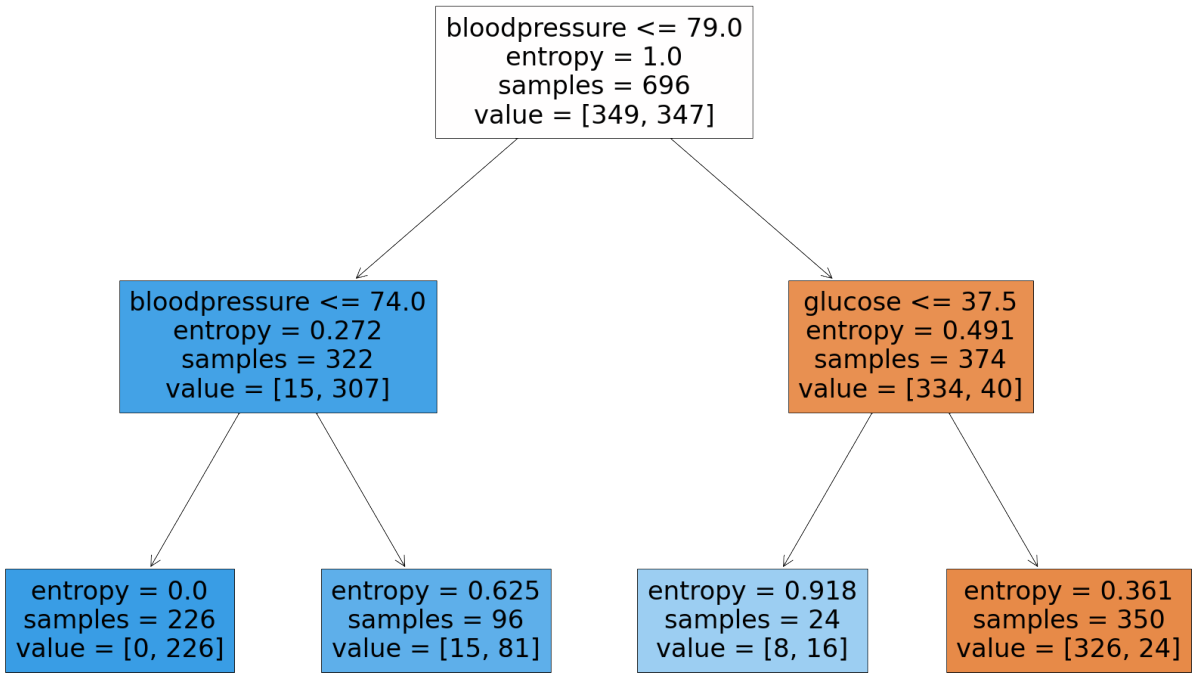
1. Apply decision tree to the data, choose entropy as the criterion, max_depth = 2 (Use training set and testing set.) and print the ACC score. (10)

In [87]:
```
from sklearn.tree import DecisionTreeClassifier
DTC = DecisionTreeClassifier(criterion = "entropy", max_depth = 2)
DTC.fit(x_train,y_train)
y_pred= DTC.predict(x_test)
print("ACC:", metrics.accuracy_score(y_test, y_pred))
```

```
ACC: 0.9230769230769231
```

1. Print the tree.(10)

In [88]:
```
from sklearn.tree import plot_tree
fig = plt.figure(figsize=(30,20))
_ =plot_tree(DTC,
                    feature_names=list(x.columns),
                    filled=True)
```



1. Calculate the root's entropy and check whether your result is the same as the tree plot(use the equation from class only use numpy to do the calculation).(10)

In [114…
```
from scipy.stats import entropy
base = 2  # work in units of bits
pk = np.array([349/(349+347), 347/(349+347)])
H = entropy(pk, base=base)
H
```

Out[114]:
```
0.9999940435616232
```

1. Calculate given blood pressure <= 79, what is the entropy for pressure <= 79, what is the IG, (use the equation from class only use numpy to do the calculation). (10)

In [115…
```
from scipy.stats import entropy
base = 2  # work in units of bits
pk = np.array([15/(322), 307/(322)])
H_2 = entropy(pk, base=base)
print("Entropy = ", H_2)
print("IG = ", H-H_2)
```

```
Entropy =  0.2717042182923411
IG =  0.7282898252692822
```

1. Find the best max_depth in Decision Tree for entopy and Gini. If it's different, explain why.(10)

In [113…
```
ACCs_Gini = []
for max_depth in range(1,11):
    DTC_gini = DecisionTreeClassifier(max_depth = max_depth)
    DTC_gini.fit(x_train,y_train)
    y_pred= DTC_gini.predict(x_test)
    ACCs_Gini.append(metrics.accuracy_score(y_test, y_pred))
print("ACCs:", ACCs_Gini)
```

```
ACCs: [0.9130434782608695, 0.9163879598662207, 0.9163879598662207, 0.9163879
598662207, 0.9264214046822743, 0.9230769230769231, 0.9230769230769231, 0.923
0769230769231, 0.9230769230769231, 0.9230769230769231]
```

In [ ]:
```
```