In [1]:
```
import pandas as pd
raw = pd.read_csv(r"D:\NIRALI\Nirali College DSBA\bank mkt task 3.csv", header=N
data = raw[0].str.split(";", expand=True)
data = data.applymap(lambda x: x.strip('"'))
header = data.iloc[0]
data = data[1:]
data.columns = header
print(data.head())
print(data.columns)
```

```
C:\Users\ADMIN\AppData\Local\Temp\ipykernel_16276\3684155092.py:4: FutureWarning:
DataFrame.applymap has been deprecated. Use DataFrame.map instead.
  data = data.applymap(lambda x: x.strip('"'))
0 age           job  marital  education default balance housing loan  contact  \
1  58    management  married   tertiary      no    2143     yes   no  unknown
2  44    technician   single  secondary      no      29     yes   no  unknown
3  33  entrepreneur  married  secondary      no       2     yes  yes  unknown
4  47   blue-collar  married    unknown      no    1506     yes   no  unknown
5  33       unknown   single    unknown      no       1      no   no  unknown

0 day month duration campaign pdays previous poutcome   y
1   5   may      261        1    -1        0  unknown  no
2   5   may      151        1    -1        0  unknown  no
3   5   may       76        1    -1        0  unknown  no
4   5   may       92        1    -1        0  unknown  no
5   5   may      198        1    -1        0  unknown  no
Index(['age', 'job', 'marital', 'education', 'default', 'balance', 'housing',
       'loan', 'contact', 'day', 'month', 'duration', 'campaign', 'pdays',
       'previous', 'poutcome', 'y'],
      dtype='object', name=0)
```

In [4]:
```
# Separate features and target
X = data.drop("y", axis=1)
y = data["y"].map({"yes": 1, "no": 0})

# One-hot encode categorical variables
X = pd.get_dummies(X, drop_first=True)
```

In [8]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split( X, y, test_size=0.3, random
```

In [9]:
```
from sklearn.tree import DecisionTreeClassifier

clf = DecisionTreeClassifier(criterion="entropy",max_depth=5,random_state=42)
clf.fit(X_train, y_train)
```

Out[9]:

DecisionTreeClassifier

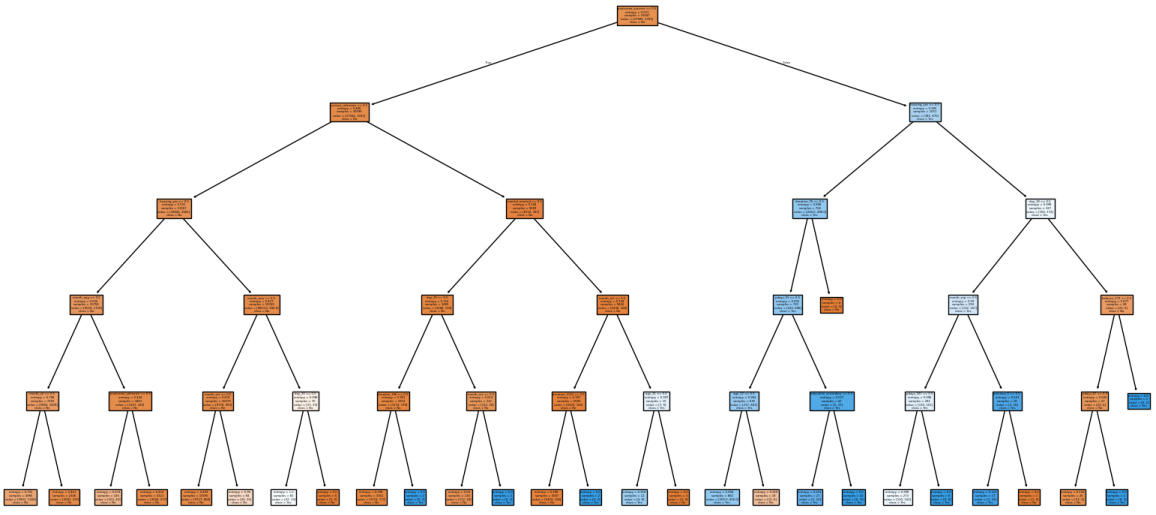
```
DecisionTreeClassifier(criterion='entropy', max_depth=5, random_state=4
2)
```

In [10]:
```
from sklearn.metrics import accuracy_score, confusion_matrix, classification_rep

y_pred = clf.predict(X_test)
```

```
print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
```

```
Accuracy: 0.8939103509289296

Confusion Matrix:
 [[11810   167]
 [ 1272   315]]

Classification Report:
              precision    recall  f1-score   support

           0       0.90      0.99      0.94     11977
           1       0.65      0.20      0.30      1587

    accuracy                           0.89     13564
   macro avg       0.78      0.59      0.62     13564
weighted avg       0.87      0.89      0.87     13564
```

In [11]:
```
from sklearn.tree import plot_tree
import matplotlib.pyplot as plt

plt.figure(figsize=(20,10))
plot_tree(clf, feature_names=list(X.columns), class_names=["No","Yes"], filled=T
plt.show()
```



In [ ]: