Load dateset

```
import pandas as pd
df = pd.read_csv("https://raw.githubusercontent.com/dataprofessor/data/master/weather-nominal-weka.csv")
df.head()
```

	outlook	temperature	humidity	windy	play
0	sunny	hot	high	False	no
1	sunny	hot	high	True	no
2	overcast	hot	high	False	yes
3	rainy	mild	high	False	yes
4	rainy	cool	normal	False	yes

Data prepration

Seperate data into x and y

```
#X=df.drop(columns="Species")
Y=df["play"]
Y
```

```
0      no
1      no
2     yes
3     yes
4     yes
5      no
6     yes
7      no
8     yes
9     yes
10    yes
11    yes
12    yes
13     no
Name: play, dtype: object
```

```
X = df.drop(["play"], axis=1)
X
```

	outlook	temperature	humidity	windy
0	sunny	hot	high	False
1	sunny	hot	high	True
2	overcast	hot	high	False
3	rainy	mild	high	False
4	rainy	cool	normal	False
5	rainy	cool	normal	True
6	overcast	cool	normal	True
7	sunny	mild	high	False
8	sunny	cool	normal	False
9	rainy	mild	normal	False
10	sunny	mild	normal	True
11	overcast	mild	high	True
12	overcast	hot	normal	False
13	rainy	mild	high	True

Encodering

```
from sklearn.preprocessing import LabelEncoder

label_encoder = LabelEncoder()
for column in df.columns[:-1]:  # Exclude the target column 'play'
    df[column] = label_encoder.fit_transform(df[column])
```

Seperate data into x and y

```
# Separate the features (X) and target variable (Y)
X = df.drop('play', axis=1)
Y = label_encoder.fit_transform(df['play'])
```

Date spliting

```
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test= train_test_split(X,Y, train_size=0.8 , random_state=100)
```

```
x_train
```

	outlook	temperature	humidity	windy
1	2	1	0	1
9	1	2	1	0
4	1	0	1	0
6	0	0	1	1
2	0	1	0	0
0	2	1	0	0
10	2	2	1	1
7	2	2	0	0
3	1	2	0	0
13	1	2	0	1
8	2	0	1	0

```
x_test
```

	outlook	temperature	humidity	windy
11	0	2	0	1
12	0	1	1	0
5	1	0	1	1

Modle training

Classifion

Train the model

```
from sklearn import tree
model=tree.DecisionTreeClassifier()
model.fit(x_train, y_train)
```

```
DecisionTreeClassifier
DecisionTreeClassifier()
```

Apply the model for prediction

```
y_train_pred=model.predict(x_train)
y_test_pred=model.predict(x_test)
```

```
y_train_pred
```

```
array([0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1])
```

```
y_test_pred
```

```
array([0, 1, 1])
```

Evaluate the model

```
from sklearn.metrics import mean_squared_error,r2_score
lr_train_MSE=mean_squared_error(y_train,y_train_pred)
lr_train_r2=r2_score(y_train,y_train_pred)

lr_test_MSE=mean_squared_error(y_test,y_test_pred)
lr_test_r2=r2_score(y_test,y_test_pred)
```

```
print('LR MSE(Train):',lr_train_MSE)
print('LR r2(Train):',lr_train_r2)
print('LR MSE(Test):',lr_test_MSE)
print('LR r2(Test):',lr_test_r2)
```

```
LR MSE(Train): 0.0
LR r2(Train): 1.0
LR MSE(Test): 0.6666666666666666
LR r2(Test): -1.9999999999999996
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
import pickle
# save the model to disk
filename = 'Weather_dataset.sav'
pickle.dump(model, open(filename, 'wb'))
```

```
#calling saved model
loaded_model = pickle.load(open(filename, 'rb'))
```

```
result = loaded_model.score(x_test, y_test)
print(result)
```

```
0.3333333333333333
```