Lab-03

January 25, 2026

0.0.1 Lab-03

1 KNN as Classifier [As Regressor in another program]

```python
[1]: #Assigning Feature and local variables
     #First Feature
     weather = ['Sunny','Sunny',
     'Overcast','Rainy','Rainy','Rainy','Overcast','Sunny','Sunny','Rainy',
     'Sunny','Overcast','Overcast','Rainy']
```

```python
[2]: #Second Feature
     temp =␣
      ↪['Hot','Hot','Hot','Mild','Cool','Cool','Cool','Mild','Cool','Mild','Mild','Mild','Hot','Mi
```

```python
[3]: # Label or Target Variable
     play = ['No','No','Yes','Yes','Yes','No','Yes','No','Yes','Yes','Yes','Yes',
     'Yes','No' ]
```

```python
[4]: # import Label Encoder
     from sklearn import preprocessing
```

```python
[5]: #creating Label Encoder
     le = preprocessing.LabelEncoder()
```

```python
[6]: #converting Strigns Labels to numbers
     weather_encoded = le.fit_transform(weather)
```

```python
[7]: weather_encoded
```

```
[7]: array([2, 2, 0, 1, 1, 1, 0, 2, 2, 1, 2, 0, 0, 1])
```

```python
[8]: #converting Strigns Labels to numbers
     temp_encoded = le.fit_transform(temp)
     label_col = le.fit_transform(play)
     temp_encoded
```

```
[8]: array([1, 1, 1, 2, 0, 0, 0, 2, 0, 2, 2, 2, 1, 2])
```

```python
[9]: #combining weather and temp into single list of tuples
     features= list(zip(weather_encoded,temp_encoded))
     features
```

```
[9]: [(np.int64(2), np.int64(1)),
      (np.int64(2), np.int64(1)),
      (np.int64(0), np.int64(1)),
      (np.int64(1), np.int64(2)),
      (np.int64(1), np.int64(0)),
      (np.int64(1), np.int64(0)),
      (np.int64(0), np.int64(0)),
      (np.int64(2), np.int64(2)),
      (np.int64(2), np.int64(0)),
      (np.int64(1), np.int64(2)),
      (np.int64(2), np.int64(2)),
      (np.int64(0), np.int64(2)),
      (np.int64(0), np.int64(1)),
      (np.int64(1), np.int64(2))]
```

```python
[10]: from sklearn.neighbors import KNeighborsClassifier
      model = KNeighborsClassifier (n_neighbors=3)
      model.fit(features,label_col)
      KNeighborsClassifier(n_neighbors=3)
```

```
[10]: KNeighborsClassifier(n_neighbors=3)
```

```python
[11]: #predict output
      predicted = model.predict([[1,0]])
      predicted
```

```
[11]: array([1])
```

```python
[12]: print(predicted)
```

```
[1]
```