```
#import libraries
import numpy as np
import pandas as pd


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

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

```
#list of symptoms(features)
l1 = ['back_pain', 'constipation', 'abdominal_pain', 'diarrhoea', 'mild_fever', 'yellow_urine
      'yellowing_of_eyes', 'acute_liver_failure', 'fluid_overload', 'swelling_of_stomach',
      'swelled_lymph_nodes', 'malaise', 'blurred_and_distorted_vision', 'phlegm', 'throat_irr
      'redness_of_eyes', 'sinus_pressure', 'runny_nose', 'congestion', 'chest_pain', 'weaknes
      'fast_heart_rate', 'pain_during_bowel_movements', 'pain_in_anal_region', 'bloody_stool'
      'irritation_in_anus', 'neck_pain', 'dizziness', 'cramps', 'bruising', 'obesity', 'swol
      'swollen_blood_vessels', 'puffy_face_and_eyes', 'enlarged_thyroid', 'brittle_nails',
      'swollen_extremeties', 'excessive_hunger', 'extra_marital_contacts', 'drying_and_tingli
      'slurred_speech', 'knee_pain', 'hip_joint_pain', 'muscle_weakness', 'stiff_neck', 'swel
      'movement_stiffness', 'spinning_movements', 'loss_of_balance', 'unsteadiness',
      'weakness_of_one_body_side', 'loss_of_smell', 'bladder_discomfort', 'foul_smell_of urir
      'continuous_feel_of_urine', 'passage_of_gases', 'internal_itching', 'toxic_look_(typhos
      'depression', 'irritability', 'muscle_pain', 'altered_sensorium', 'red_spots_over_body'
      'abnormal_menstruation', 'dischromic _patches', 'watering_from_eyes', 'increased_appeti
      'family_history', 'mucoid_sputum',
      'rusty_sputum', 'lack_of_concentration', 'visual_disturbances', 'receiving_blood_transf
      'receiving_unsterile_injections', 'coma', 'stomach_bleeding', 'distention_of_abdomen',
      'history_of_alcohol_consumption', 'fluid_overload', 'blood_in_sputum', 'prominent_veins
      'palpitations', 'painful_walking', 'pus_filled_pimples', 'blackheads', 'scurring', 'ski
      'silver_like_dusting', 'small_dents_in_nails', 'inflammatory_nails', 'blister', 'red_sc
      'yellow_crust_ooze']



#list of diseases(target)
disease = ['Fungal infection', 'Allergy', 'GERD', 'Chronic cholestasis', 'Drug Reaction',
           'Peptic ulcer diseae', 'AIDS', 'Diabetes', 'Gastroenteritis', 'Bronchial Asthma',
           ' Migraine', 'Cervical spondylosis',
           'Paralysis (brain hemorrhage)', 'Jaundice', 'Malaria', 'Chicken pox', 'Dengue', 'T
           'Hepatitis B', 'Hepatitis C', 'Hepatitis D', 'Hepatitis E', 'Alcoholic hepatitis',
           'Common Cold', 'Pneumonia', 'Dimorphic hemmorhoids(piles)',
           'Heartattack', 'Varicoseveins', 'Hypothyroidism', 'Hyperthyroidism', 'Hypoglycemia
           'Arthritis', '(vertigo) Paroymsal  Positional Vertigo', 'Acne', 'Urinary tract inf
           'Impetigo']


#append target variables
l2 = []
for x in range(0, len(l1)):
```

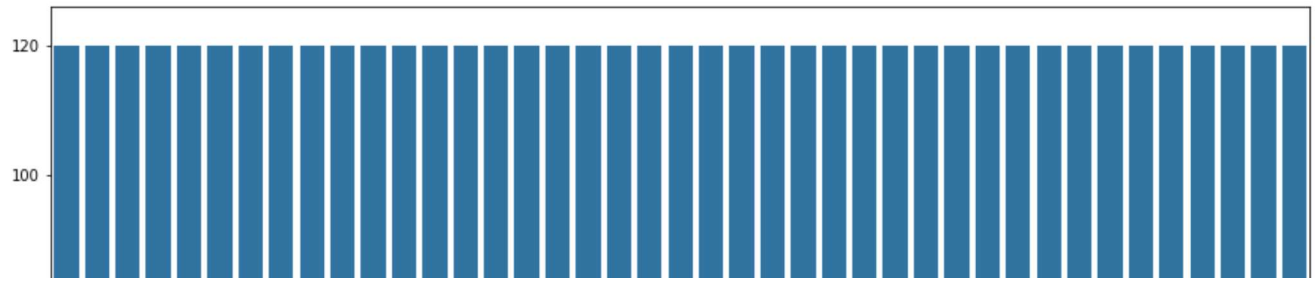
```
        l2.append(0)
    # adds data



    # TRAINING DATA df -------------------------------------------------------------------
    df = pd.read_csv("/content/drive/MyDrive/Disease-prediction-using-python-and-machine-learning



    import matplotlib.pyplot as plt
    import seaborn as sb
    #plot graph for count of each disease
    plt.figure(figsize=(15, 10))
    base_color = sb.color_palette()[0]
    sb.countplot(data=df, x='prognosis', color=base_color)
    plt.xticks(rotation=90);
```




```
#display unique target variables i.e no. of diseases
len(df['prognosis'].unique())
```

```
41
```

```
#Data cleaning
#drop null values
def drop_null_values(df):
    null_col = [col for col in df.columns if df[col].isnull().any()]
    df.drop(null_col, axis=1, inplace=True)
    return df
drop_null_values(df)
```

	itching	skin_rash	nodal_skin_eruptions	continuous_sneezing	shivering	cl
0	1	1	1	0	0	

```
#replace categorical data with numeric values
df.replace({'prognosis': {'Fungal infection': 0, 'Allergy': 1, 'GERD': 2, 'Chronic cholestasi
                          'Peptic ulcer diseae': 5, 'AIDS': 6, 'Diabetes ': 7, 'Gastroenterit
                          'Bronchial Asthma': 9, 'Hypertension ': 10,
                          'Migraine': 11, 'Cervical spondylosis': 12,
                          'Paralysis (brain hemorrhage)': 13, 'Jaundice': 14, 'Malaria': 15,
                          'Dengue': 17, 'Typhoid': 18, 'hepatitis A': 19,
                          'Hepatitis B': 20, 'Hepatitis C': 21, 'Hepatitis D': 22, 'Hepatitis
                          'Alcoholic hepatitis': 24, 'Tuberculosis': 25,
                          'Common Cold': 26, 'Pneumonia': 27, 'Dimorphic hemmorhoids(piles)':
                          'Varicose veins': 30, 'Hypothyroidism': 31,
                          'Hyperthyroidism': 32, 'Hypoglycemia': 33, 'Osteoarthristis': 34, '
                          '(vertigo) Paroymsal  Positional Vertigo': 36, 'Acne': 37, 'Urinary
                          'Psoriasis': 39,
                          'Impetigo': 40}}, inplace=True)
# inplace = True is used here to replace and save the data in place of prognosis in the trair
```

4917	0	0	0	0	0	

```
#import testing data
tr = pd.read_csv("/content/drive/MyDrive/Disease-prediction-using-python-and-machine-learning
```

4919	0	1	0	0	0	

```
#replace categorical data with numeric values
tr.replace({'prognosis': {'Fungal infection': 0, 'Allergy': 1, 'GERD': 2, 'Chronic cholestasi
                          'Peptic ulcer diseae': 5, 'AIDS': 6, 'Diabetes ': 7, 'Gastroenterit
                          'Bronchial Asthma': 9, 'Hypertension ': 10,
                          'Migraine': 11, 'Cervical spondylosis': 12,
                          'Paralysis (brain hemorrhage)': 13, 'Jaundice': 14, 'Malaria': 15,
                          'Dengue': 17, 'Typhoid': 18, 'hepatitis A': 19,
                          'Hepatitis B': 20, 'Hepatitis C': 21, 'Hepatitis D': 22, 'Hepatitis
                          'Alcoholic hepatitis': 24, 'Tuberculosis': 25,
                          'Common Cold': 26, 'Pneumonia': 27, 'Dimorphic hemmorhoids(piles)':
                          'Varicose veins': 30, 'Hypothyroidism': 31,
                          'Hyperthyroidism': 32, 'Hypoglycemia': 33, 'Osteoarthristis': 34, '
                          '(vertigo) Paroymsal  Positional Vertigo': 36, 'Acne': 37, 'Urinary
                          'Psoriasis': 39,
                          'Impetigo': 40}}, inplace=True)



#Training data
X = df[l1] #list of symptoms

y = df[["prognosis"]]   #list of diseases
```

```
np.ravel(y)
# ravel is used here to give the y array in 1D form
```

```
array([ 0,  0,  0, ..., 38, 39, 40])
```

```
#testing data
X_test = tr[l1]
y_test = tr[["prognosis"]]
np.ravel(y_test)
```

```
array([ 0,  0,  0,  0,  0,  0,  1,  1,  1,  1,  1,  2,  2,  2,  2,  2,  3,
        3,  3,  3,  3,  4,  4,  4,  4,  4,  5,  5,  5,  5,  5,  6,  6,  6,
        6,  6,  7,  7,  7,  7,  7,  8,  8,  8,  8,  8,  9,  9,  9,  9,  9,
       10, 10, 10, 10, 10, 11, 11, 11, 11, 11, 12, 12, 12, 12, 12, 13, 13,
       13, 13, 13, 14, 14, 14, 14, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16,
       17, 17, 17, 17, 17, 18, 18, 18, 18, 18, 19, 19, 19, 19, 19, 20, 20,
       20, 20, 20, 21, 21, 21, 21, 21, 22, 22, 22, 22, 22, 23, 23, 23, 23,
       23, 24, 24, 24, 24, 24, 25, 25, 25, 25, 25, 26, 26, 26, 26, 26, 27,
       27, 27, 27, 27, 28, 28, 28, 28, 28, 29, 29, 29, 29, 29, 30, 30, 30,
       30, 30, 31, 31, 32, 32, 32, 32, 33, 33, 33, 33, 33, 34, 34, 34, 34,
       34, 35, 35, 35, 35, 35, 36, 36, 36, 36, 36, 37, 37, 37, 37, 37, 38,
       38, 38, 38, 38, 39, 39, 39, 39, 39, 40, 40, 40, 40, 40])
```

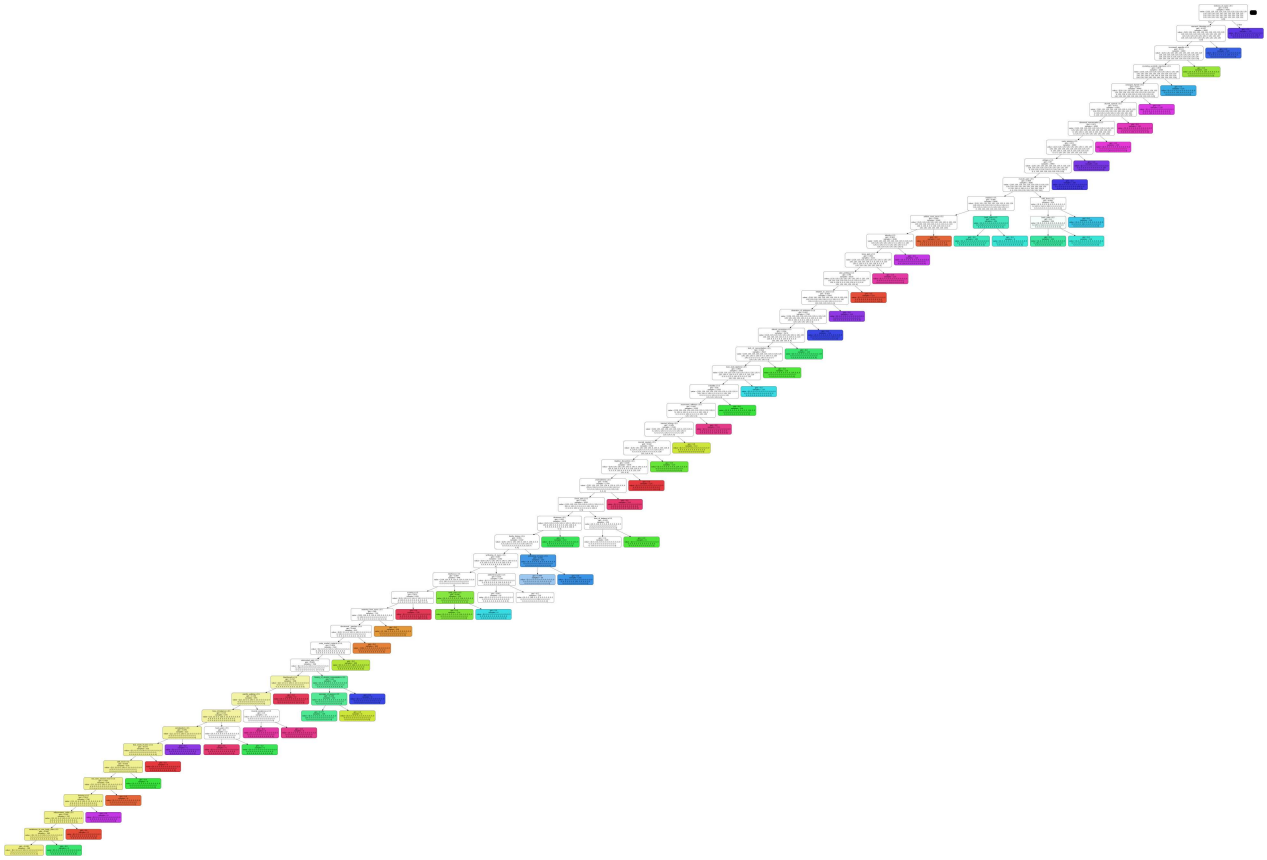
```
from sklearn.tree import DecisionTreeClassifier
#apply decision tree classifier
clf3 = DecisionTreeClassifier(criterion='gini', max_depth=50)
clf3 = clf3.fit(X, y)
y_pred = clf3.predict(X_test)
```

```
from sklearn.tree import export_graphviz
```

```
!pip install six
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/pub
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (1.15.0)
```

```
#visualize decision tree
from six import StringIO
from IPython.display import Image
import pydotplus
data = StringIO()
export_graphviz(clf3, out_file = data, filled = True, rounded = True, special_characters=True
graph = pydotplus.graph_from_dot_data(data.getvalue())
Image(graph.create_png())
```



```
# calculating accuracy
from sklearn.metrics import accuracy_score

accuracy = accuracy_score(y_test, y_pred)

print(accuracy)
```

```
0.945273631840796
```

```
#confusion matrix
from sklearn.metrics import confusion_matrix
cf_matrix=confusion_matrix(y_test,y_pred)
print(cf_matrix)
```

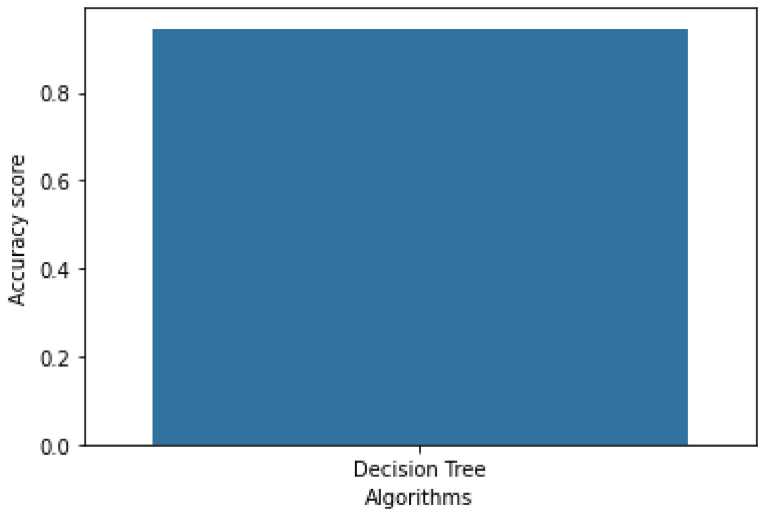
```
[[6 0 0 ... 0 0 0]
```

```
[0 5 0 ... 0 0 0]
[0 0 5 ... 0 0 0]
...
[0 0 0 ... 5 0 0]
[0 0 0 ... 0 5 0]
[0 0 0 ... 0 0 5]]
```

```
import seaborn as sns
```

```
#plot of accuracy
score = [accuracy]
algorithms = ["Decision Tree"]
plt.xlabel("Algorithms")
plt.ylabel("Accuracy score")
sns.barplot(algorithms,score)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: P
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7fa52382bd90>
```



✓ 0s completed at 9:01 AM