Decision Trees and Random Forests in Python

This is the code for the lecture video which goes over tree methods in Python. Reference the video lecture for the full explanation of the code!

I also wrote a blog post explaining the general logic of decision trees and random forests which you can check out.

Import Libraries

In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

Get the Data

In [2]:

```
df = pd.read_csv('15 Decision Trees and Random Forests.csv')
```

In [3]:

```
df.head()
```

Out[3]:

	Kyphosis	Age	Number	Start
0	absent	71	3	5
1	absent	158	3	14
2	present	128	4	5
3	absent	2	5	1
4	absent	1	4	15

EDA

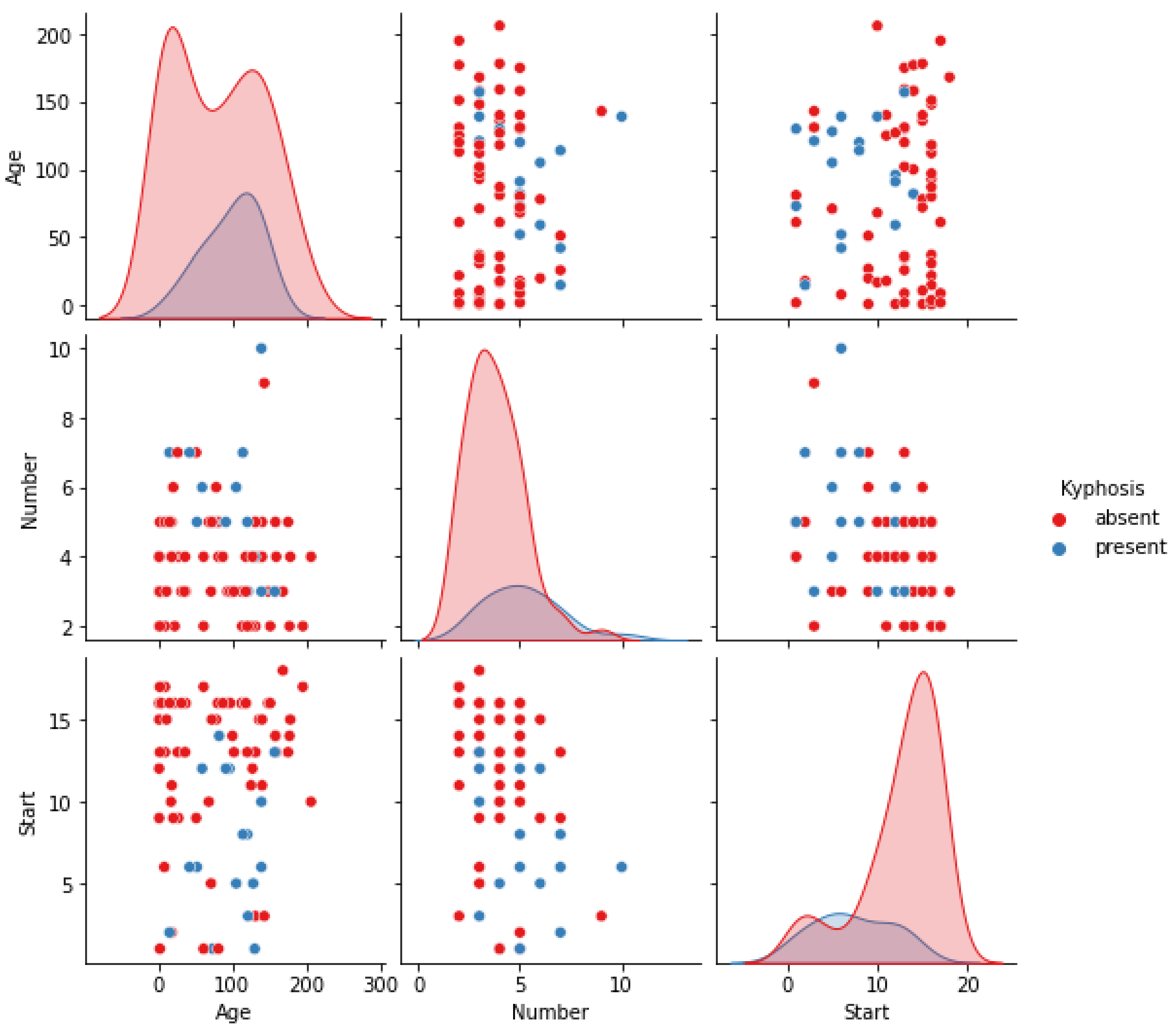
We'll just check out a simple pairplot for this small dataset.

In [4]:

```
sns.pairplot(df,hue='Kyphosis',palette='Set1')
```

Out[4]:

```
<seaborn.axisgrid.PairGrid at 0x25c5fc605e0>
```



Train Test Split

Let's split up the data into a training set and a test set!

In [5]:

```
from sklearn.model_selection import train_test_split
```

In [6]:

```
X = df.drop('Kyphosis',axis=1)
y = df['Kyphosis']
```

In [7]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30)
```

Decision Trees

We'll start just by training a single decision tree.

In [8]:

```
from sklearn.tree import DecisionTreeClassifier
```

In [9]:

```
dtree = DecisionTreeClassifier()
```

In [10]:

```
dtree.fit(X_train,y_train)
```

Out[10]:

```
DecisionTreeClassifier()
```

Prediction and Evaluation

Let's evaluate our decision tree.

In [11]:

```
predictions = dtree.predict(X_test)
```

In [12]:

```
from sklearn.metrics import classification_report,confusion_matrix
```

In [13]:

```
print(classification_report(y_test,predictions))
```

```
              precision    recall  f1-score   support

      absent       0.90      0.95      0.93        20
     present       0.75      0.60      0.67         5

    accuracy                           0.88        25
   macro avg       0.83      0.77      0.80        25
weighted avg       0.87      0.88      0.87        25
```

In [14]:

```
print(confusion_matrix(y_test,predictions))
```

```
[[19  1]
 [ 2  3]]
```

Random Forests

Now let's compare the decision tree model to a random forest.

In [15]:

```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier(n_estimators=100)
rfc.fit(X_train, y_train)
```

Out[15]:

```
RandomForestClassifier()
```

In [16]:

```
rfc_pred = rfc.predict(X_test)
```

In [17]:

```
print(confusion_matrix(y_test,rfc_pred))
```

```
[[20  0]
 [ 3  2]]
```

In [18]:

```
print(classification_report(y_test,rfc_pred))
```

```
              precision    recall  f1-score   support

      absent       0.87      1.00      0.93        20
     present       1.00      0.40      0.57         5

    accuracy                           0.88        25
   macro avg       0.93      0.70      0.75        25
weighted avg       0.90      0.88      0.86        25
```

Great Job!