In [1]:
```
import pandas as pd
```

In [2]:
```
titanic_data = pd.read_csv('https://raw.githubusercontent.com/zekelabs/data-science-complete-tutorial/master/Data/ti
tanic-train.csv.txt')
```

In [3]:
```
titanic_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
PassengerId    891 non-null int64
Survived       891 non-null int64
Pclass         891 non-null int64
Name           891 non-null object
Sex            891 non-null object
Age            714 non-null float64
SibSp          891 non-null int64
Parch          891 non-null int64
Ticket         891 non-null object
Fare           891 non-null float64
Cabin          204 non-null object
Embarked       889 non-null object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.6+ KB
```

In [23]:
```
titanic_data.describe()
```

Out[23]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

- Splitting feature data & target data
- Dropping less impotant columns

In [4]:
```
feature_data = titanic_data.drop(['Survived'], axis=1)
```

In [5]:
```
target_data = titanic_data.Survived
```

In [6]:
```
feature_data.drop(['Name','Cabin','PassengerId'], axis=1, inplace=True)
```

In [9]:
```
feature_data.drop(['Ticket'],axis=1,inplace=True)
```

In [10]:
```
feature_data.head()
```

Out[10]:

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	3	male	22.0	1	0	7.2500	S
1	1	female	38.0	1	0	71.2833	C
2	3	female	26.0	0	0	7.9250	S
3	1	female	35.0	1	0	53.1000	S
4	3	male	35.0	0	0	8.0500	S

In [11]:
```
feature_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 7 columns):
Pclass      891 non-null int64
Sex         891 non-null object
Age         714 non-null float64
SibSp       891 non-null int64
Parch       891 non-null int64
Fare        891 non-null float64
Embarked    889 non-null object
dtypes: float64(2), int64(3), object(2)
memory usage: 48.8+ KB
```

In [14]:
```
feature_data.Embarked.value_counts()
```

Out[14]:
```
S    644
C    168
Q     77
Name: Embarked, dtype: int64
```

- Seperating features based on types

In [13]:
```
cat_cols = list(feature_data.select_dtypes(include=['object']))
num_cols = list(feature_data.select_dtypes(exclude=['object']))
```

- Realizing missing values, create imputers

In [12]:
```
from sklearn.impute import SimpleImputer
```

In [15]:
```
si_cat_cols = SimpleImputer(strategy='constant', fill_value='S')
si_num_cols = SimpleImputer(strategy='median')
```

In [17]:
```
num_data = si_num_cols.fit_transform(feature_data[num_cols])
```

In [18]:
```
cat_data = si_cat_cols.fit_transform(feature_data[cat_cols])
```

In [19]:
```
from sklearn.preprocessing import OneHotEncoder
```

In [20]:
```
ohe = OneHotEncoder()
```

In [22]:
```
cat_data_ohe = ohe.fit_transform(cat_data).toarray()
```

Combine transformed data

In [24]:
```
import numpy as np
feature_data_tf = np.hstack([num_data, cat_data_ohe])
```

In [25]:
```
from sklearn.model_selection import train_test_split
```

In [26]:
```
trainx, testx, trainy, testy = train_test_split(feature_data_tf, target_data)
```

In [30]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
```

In [34]:
```
models = [ LogisticRegression(), DecisionTreeClassifier(), RandomForestClassifier()]
trained_models = []

for model in models:
    model.fit(trainx,trainy)
    trained_models.append(model)
```

```
/home/awantik/anaconda3/lib/python3.7/site-packages/sklearn/linear_model/logistic.py:433: FutureWarning: Default solv
er will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
/home/awantik/anaconda3/lib/python3.7/site-packages/sklearn/ensemble/forest.py:246: FutureWarning: The default value
of n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
```

In [35]:
```
for model in trained_models:
    print (model.score(testx,testy))
```

```
0.7354260089686099
0.7892376681614349
0.7847533632286996
```

In [ ]:
```

```