Titanic

This is an attempt at it after studying some basics of Data Science.

Titanic is a discrete problem, hence it needs to be solved using classification.

We have the test set and the train set, and we need to make a model that categorizes the survivability of the passengers of the infamous Titanic disaster. So to do so, we are going to be looking at the dimensions of the data provided to us.

In [1]:

```
import pandas

train = pandas.read_csv("./data/train.csv")
train
```

Out[1]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500
...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500

So the first thing we wis to figure out is the behaviour and type of data given.

Data Description

We describe the data as the following:

Column	Definition	Keys if any	Numbers of keys
PassengerId	Key for identifying individual passengers		
Survived	Tells if a person survived or not	0 for dead 1 for alive	2
Pclass	Class of ticket purchased	n for nth class	3
Name	Name of the passenger		
Sex	The gender of the passenger		
Age	The age of the person		
SibSp	Number of siblings/spouses aboard the titanic		
Parch	Number of parents/children aboard the titanic		
Ticket	Ticket Number		
Fare	The fare for the ticket		
Cabin	The cabin numbers for people with a cabin		
Embarked	Port of embarkation	C = Cherbourg, Q = Queenstown, S = Southampton	3

Next object of interest is the correlation of the various fields, but in order to do so, we need to change certain categorical information into discrete numericals.

In [2]:
```
# categorical transformations

train['Sex'] = train['Sex'].replace(['female', 'male'], [0, 1])
train['Embarked'] = train['Embarked'].replace(['C', 'Q', 'S'], [0, 1, 2])
```

In [3]:
```
train.corr()['Survived']
```

Out[3]:
```
PassengerId   -0.005007
Survived       1.000000
Pclass        -0.338481
Sex           -0.543351
Age           -0.077221
SibSp         -0.035322
Parch          0.081629
Fare           0.257307
Embarked      -0.169718
Name: Survived, dtype: float64
```

Some columns we readily exempt from our analysis.

1. Fare, since Pclass is a much better indicator
2. PassengerId
3. Parch and Sibsp are reflective in the sense that if there was a sibling of the passenger onboard for a passenger, it would reflect the same for the person. This in turn is highly specific and a model around this cannot be built.

In [4]:
```
attribs_required = ['Pclass', 'Sex', 'Embarked', 'Age']
# Pclass, Sex and Embarked are discrete
# Age is continuous(relative to the dimensionality of the other attribute
```

Pclass is questionable, to show why, let us see the value counts of the data.

In [5]:
```
train.groupby("Survived")['Pclass'].value_counts()
```

Out[5]:
```
Survived  Pclass
0         3         372
          2          97
          1          80
1         1         136
          3         119
          2          87
Name: Pclass, dtype: int64
```

This shows us that while it has a high correlation, it might prove to be create a further non-deterministic category.

Next, we need to check what we need to impute, for training to work.

In [6]:
```
train.isna().sum()
```

Out[6]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

Age has 177 null values out of 891 values. This is significant to cause a change in the distribution if averaged, or imputed based on other factors.
We should still try to see if there is some quick method to do so....

In [7]:
```
required = train[train['Age'].isnull()]
required
```

Out[7]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Ca
5	6	0	3	Moran, Mr. James	1	NaN	0	0	330877	8.4583	N
17	18	1	2	Williams, Mr. Charles Eugene	1	NaN	0	0	244373	13.0000	N
19	20	1	3	Masselmani, Mrs. Fatima	0	NaN	0	0	2649	7.2250	N

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Ca
26	27	0	3	Emir, Mr. Farred Chehab	1	NaN	0	0	2631	7.2250	N
28	29	1	3	O'Dwyer, Miss. Ellen "Nellie"	0	NaN	0	0	330959	7.8792	N
...	...	...	...	...	...	...	...	...	...	...	
859	860	0	3	Razi, Mr. Raihed	1	NaN	0	0	2629	7.2292	N
863	864	0	3	Sage, Miss. Dorothy Edith "Dolly"	0	NaN	8	2	CA. 2343	69.5500	N
868	869	0	3	van Melkebeke, Mr. Philemon	1	NaN	0	0	345777	9.5000	N
878	879	0	3	Laleff, Mr. Kristo	1	NaN	0	0	349217	7.8958	N
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	0	NaN	1	2	W./C. 6607	23.4500	N

In [8]:

```python
underage_indicators = ['M\.', 'Ms\.', 'Miss\.', 'Master\.']
train[train['Name'].str.contains('|'.join(underage_indicators))]
```

Out[8]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	C
2	3	1	3	Heikkinen, Miss. Laina	0	26.0	0	0	STON/O2. 3101282	7.9250	
7	8	0	3	Palsson, Master. Gosta Leonard	1	2.0	3	1	349909	21.0750	
10	11	1	3	Sandstrom, Miss. Marguerite Rut	0	4.0	1	1	PP 9549	16.7000	
11	12	1	1	Bonnell, Miss. Elizabeth	0	58.0	0	0	113783	26.5500	
14	15	0	3	Vestrom, Miss. Hulda Amanda Adolfina	0	14.0	0	0	350406	7.8542	
...	...	...	...	...	...	...	...	...	...	...	
869	870	1	3	Johnson, Master. Harold Theodor	1	4.0	1	1	347742	11.1333	

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare
875	876	1	3	Najib, Miss. Adele Kiamie "Jane"	0	15.0	0	0	2667	7.2250
882	883	0	3	Dahlberg, Miss. Gerda Ulrika	0	22.0	0	0	7552	10.5167
887	888	1	1	Graham, Miss. Margaret Edith	0	19.0	0	0	112053	30.0000
888	889	0	3	Johnston, Miss. Catherine Helen	0	NaN	1	2	W./C. 6607	23.4500

In [9]:
```python
train.describe()
```

Out[9]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	
count	891.000000	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	8
mean	446.000000	0.383838	2.308642	0.647587	29.699118	0.523008	0.381594	
std	257.353842	0.486592	0.836071	0.477990	14.526497	1.102743	0.806057	
min	1.000000	0.000000	1.000000	0.000000	0.420000	0.000000	0.000000	
25%	223.500000	0.000000	2.000000	0.000000	20.125000	0.000000	0.000000	
50%	446.000000	0.000000	3.000000	1.000000	28.000000	0.000000	0.000000	
75%	668.500000	1.000000	3.000000	1.000000	38.000000	1.000000	0.000000	
max	891.000000	1.000000	3.000000	1.000000	80.000000	8.000000	6.000000	5

So we see that if we attempt to impute the age based on pre-existent data and the honorific, we'll have a deviation of 14, which is unacceptable, and would serve to increase issue in the prediction model.

In [10]:
```python
train.isna().sum()
```

Out[10]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

In [11]:
```python
train['Embarked'].value_counts()
```

Out[11]:
```
2.0    644
0.0    168
```

```
1.0      77
```

In [12]:

```python
train['Embarked'] = train['Embarked'].fillna(2)
```

In [13]:

```python
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split

y = train['Survived']
# x_train, x_valid, y_train, y_valid = train_test_split(train[attribs_req

model = RandomForestClassifier(n_estimators=100, max_depth=3, random_stat
model.fit(train[attribs_required[:-1]], y)
# val_predictions = model.predict(x_valid)

# accuracy_score(val_predictions, y_valid)
```

Out[13]:

```
RandomForestClassifier(max_depth=3, random_state=3)
```

In [14]:

```python
test_data = pandas.read_csv('./data/test.csv')
test_data
# val_predictions = model.predict(test_data)
```

Out[14]:

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
0	892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN
1	893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN
2	894	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN
3	895	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN
4	896	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN
...	...	...	...	...	...	...	...	...	...	...
413	1305	3	Spector, Mr. Woolf	male	NaN	0	0	A.5. 3236	8.0500	NaN
414	1306	1	Oliva y Ocana, Dona. Fermina	female	39.0	0	0	PC 17758	108.9000	C105
415	1307	3	Saether, Mr. Simon Sivertsen	male	38.5	0	0	SOTON/O.Q. 3101262	7.2500	NaN
416	1308	3	Ware, Mr. Frederick	male	NaN	0	0	359309	8.0500	NaN

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
417	1309	3	Peter, Master.	male	NaN	1	1	2668	22.3583	NaN

In [15]:

```
# transform test_data based on the predefined transform logic
test_data['Sex'] = test_data['Sex'].replace(['female', 'male'], [0, 1])
test_data['Embarked'] = test_data['Embarked'].replace(['C', 'Q', 'S'], [0
```

In [16]:

```
test_data.isna().sum()
```

Out[16]:

```
PassengerId      0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
dtype: int64
```

In [17]:

```
test_data['Survived'] = model.predict(test_data[attribs_required[:-1]])
```

In [18]:

```
answer_set = pandas.read_csv('./data/actual_result.csv')
answer_set
```

Out[18]:

	PassengerId	Survived
0	892	0
1	893	1
2	894	0
3	895	0
4	896	1
...	...	...
413	1305	0
414	1306	1
415	1307	0
416	1308	0
417	1309	1

418 rows × 2 columns

In [19]:

```
check = (test_data[['PassengerId', 'Survived']] == answer_set)
counts = check['Survived'].value_counts()
accuracy = counts[True]  * 100 / check['Survived'].size
accuracy
```

Out[19]:

```
77.7511961722488
```