Project2_Investigate_Dataset_Titanic_v2

October 8, 2016

1 1. Dataset information and research questions

1.0.1 1.1. Read me

For our project, we will analyze Titanic Data. The Royal Mail Ship (RMS) Titanic was a British passenger liner that sank in the North Atlantic Ocean in the early morning of 15 April 1912, after colliding with an iceberg during her maiden voyage from Southampton to New York City (source: Wikipedia). The available dataset was obtained from the kaggle website by Udacity's training staff and instructors. It mainly contains demographics and passenger information from 891 of the 2224 passengers and crew on board of the Titanic.

1.0.2 1.2. Variable descriptions

The dataset contains the following information:
- Survival (survival: 0 = No; 1 = Yes)
- Passenger Class (pclass: 1 = 1st; 2 = 2nd; 3 = 3rd)
- Name (name)
- Sex (sex)
- Age (age)
- Number of Siblings/Spouses Aboard (sibsp)
- Number of Parents/Children Aboard (parch)
- Ticket Number (ticket)
- Passenger Fare (fare)
- Cabin (cabin)
- Port of Embarkation (embarked: C = Cherbourg; Q = Queenstown; S = Southampton)

1.0.3 1.3. Investigation question

A BBC Magazine article reporting converging opinions of various experts and investigators sustains that on the RMS Titanic "women and children survived in greater numbers across all classes as they were given priority on the lifeboats".
To assess the veracity of this affirmation, we will address whether and how the presence of siblings and other first-degree relatives aboard affected the proportion of Titanic passengers surviving, on the assumption that children were travelling (and saved) with their parents.

---

2 2. Data wrangling

2.1 2.1. Assessment of data quality

In [1]:
```
# The following code reads all the Titanic data into
# Pandas DataFrames.

import pandas as pd
import numpy as np

titanic_df = pd.read_csv('titanic_data.csv')
```

In [2]:
```
# The following code displays Titanic data first 5 lines
titanic_df.head()
```

Out[2]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

In [3]:
```
# Checking data types of our data to assess whether the parameters we are i
# are available and usable

titanic_df.dtypes
```

Out[3]:
```
PassengerId      int64
Survived         int64
Pclass           int64
Name            object
Sex             object
Age            float64
SibSp            int64
Parch            int64
```

```
Ticket          object
Fare           float64
Cabin           object
Embarked        object
dtype: object
```

```
In [4]: # Assessing the presence of missing data (NaN)

        titanic_df.isnull().sum()
```

```
Out[4]: 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
```

2.2 2.2. Data quality report

All our parameters seem available and usable: - survival of Titanic passengers ('Survived'), int64 with values 0 = No; 1 = Yes - passengers' age ('Age'), float64 - number of siblings aboard ('SibSp'), int64

177 values are missing from the Age column.
Rows containing these missing data will be dropped when the Age column will be handled

2.3 2.3. Droping of columns with information not relevant for the study

```
In [5]: del titanic_df['Ticket']
        del titanic_df['Fare']
        del titanic_df['Cabin']
        del titanic_df['Name']
        del titanic_df['Pclass']
        del titanic_df['Sex']
        del titanic_df['Parch']
        del titanic_df['Embarked']

        titanic_df.head()
```

```
Out[5]:    PassengerId  Survived   Age  SibSp
        0            1         0  22.0      1
        1            2         1  38.0      1
        2            3         1  26.0      0
```

```
3            4         1   35.0      1
4            5         0   35.0      0
```

---

3 3. Study of the correlation between the number of siblings aboard and survival

3.1. Data extraction

```
In [6]: # Total number of passengers who survived or died
        titanic_df.groupby('Survived')['PassengerId'].count()

Out[6]: Survived
        0    549
        1    342
        Name: PassengerId, dtype: int64

In [7]: # Count of passengers who survived and died grouped according to their numb
        D = titanic_df.groupby(['Survived','SibSp'], as_index=False)['PassengerId']
        print D
```

	Survived	SibSp	PassengerId
0	0	0	398
1	0	1	97
2	0	2	15
3	0	3	12
4	0	4	15
5	0	5	5
6	0	8	7
7	1	0	210
8	1	1	112
9	1	2	13
10	1	3	4
11	1	4	3

```
In [8]: # Creating a DataFrame including information on the survival of the proport

        dead_numpy_array = np.array(D['PassengerId'][0:7]) # remove the index
        dead_pd_series = pd.Series(dead_numpy_array, index = ['0','1','2','3','4','

        survivors_numpy_array = np.array(D['PassengerId'][7:12]) # removes the inde
        survivors_numpy_array = np.append(survivors_numpy_array,[0,0]) # add the va
        survivors_pd_series = pd.Series(survivors_numpy_array, index = ['0','1','2'

        siblings_number_df = pd.concat([dead_pd_series, survivors_pd_series], axis=
        siblings_number_df.columns = ['died', 'survived']
```

```
        siblings_proportion_df = siblings_number_df.div(siblings_number_df.sum())
        print 100*siblings_proportion_df
```

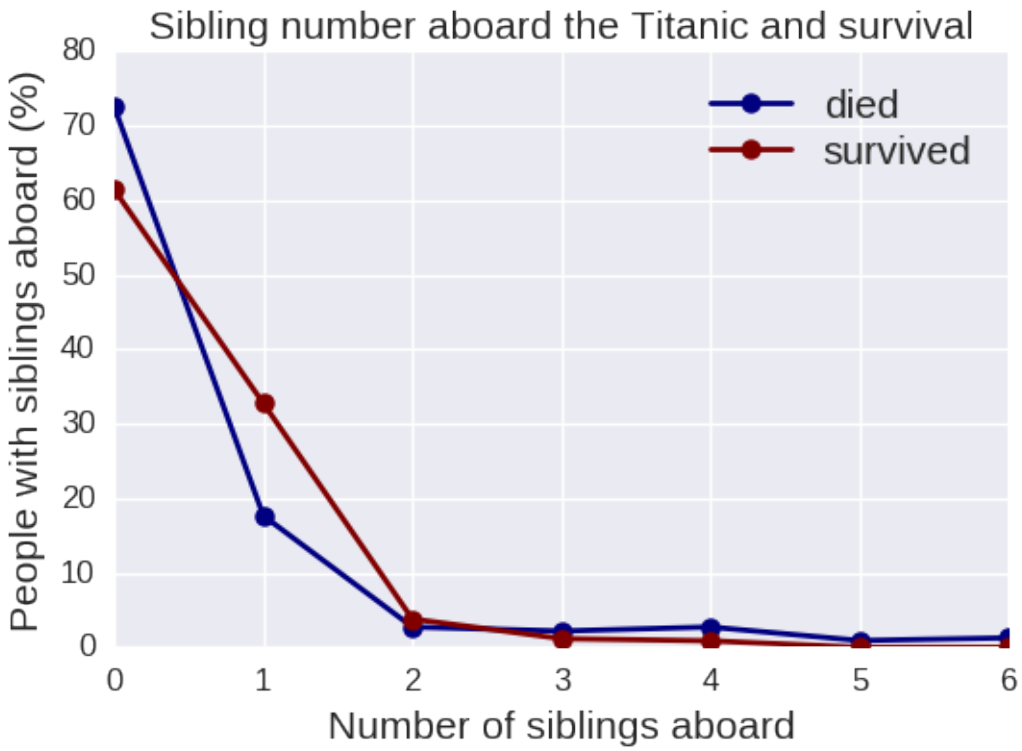
```
        died   survived
0  72.495446  61.403509
1  17.668488  32.748538
2   2.732240   3.801170
3   2.185792   1.169591
4   2.732240   0.877193
5   0.910747   0.000000
6   1.275046   0.000000
```

3.1 3.2. Plot allowing data visualization for data-based predictions/hypotheses

In [9]:
```
        # Plotting the Dataframe for preliminary observations and data-based hypoth
        import matplotlib.pyplot as plt
        import seaborn as sns

        % pylab inline
        ax =(100*siblings_proportion_df).plot(lw=2,colormap='jet',marker='.',marker

        ax.set_title('Sibling number aboard the Titanic and survival', fontsize=15)
        ax.set_xlabel('Number of siblings aboard', fontsize=15)
        ax.set_ylabel('People with siblings aboard (%)', fontsize=15)
        legend(labelspacing=0.25, fontsize = 15)
```

```
Populating the interactive namespace from numpy and matplotlib
```

Out[9]:
```
<matplotlib.legend.Legend at 0x7f2644d68c50>
```



3.2 3.3. Observations

1) Most people on the Titanic did not have siblings on board.There seem to be no difference on survival in this category

2) The number of dead and survivors decreases with the number of sibllings, becoming very low from 2 siblings. People with two or more siblings on board are to be grouped to obtain a larger groups

3) There seem to be no difference between the number of dead and survivors with 1 sibling on board.

3.3 3.4. Re-organizing data to group people with two siblings or more

```
In [10]: # Creating a DataFrame including information on the survival of the propor

         dead_numpy_array_2_groups = np.array(D['PassengerId'][0:2]) # remove the i
         dead_numpy_array_1_group = (D['PassengerId'][2:6]).sum() # summing up pass
         dead_numpy_array_3_groups = np.append(dead_numpy_array_2_groups, dead_nump
         dead_pd_series_3_groups = pd.Series(dead_numpy_array_3_groups) # attribute

         survivors_numpy_array_2_groups = np.array(D['PassengerId'][7:9]) # removes
         survivors_numpy_array_1_group = (D['PassengerId'][9:12]).sum() # summing u
```

```
survivors_numpy_array_3_groups = np.append(survivors_numpy_array_2_groups,
survivors_pd_series_3_groups = pd.Series(survivors_numpy_array_3_groups)

siblings_number_df_3_groups = pd.concat([dead_pd_series_3_groups, survivor
siblings_number_df_3_groups.columns = ['died', 'survived']

siblings_percent_df = 100*siblings_number_df_3_groups.div(siblings_number_

print siblings_percent_df
```

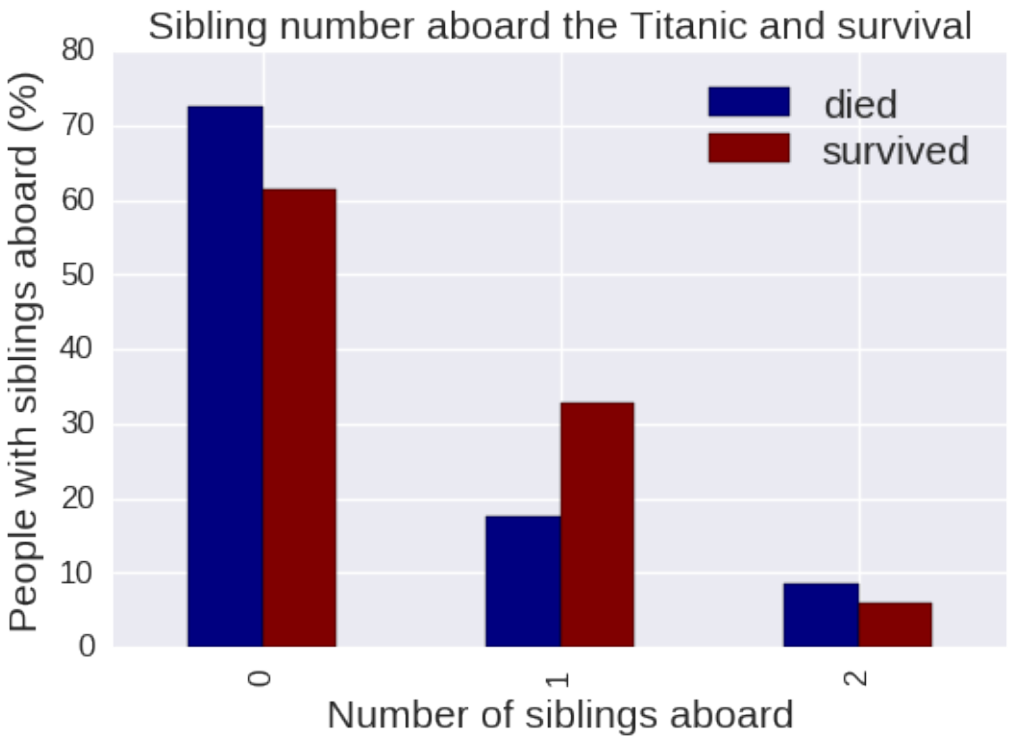
```
        died   survived
0  72.495446  61.403509
1  17.668488  32.748538
2   8.561020   5.847953
```

3.4 3.5. Additional plotting for current data visualization

In [11]:
```
# Plotting the Dataframe for preliminary observations and data-based hypot
import matplotlib.pyplot as plt
import seaborn as sns

% pylab inline
ax =(siblings_percent_df).plot.bar(colormap= 'jet', fontsize=12)

ax.set_title('Sibling number aboard the Titanic and survival', fontsize=15
ax.set_xlabel('Number of siblings aboard', fontsize=15)
ax.set_ylabel('People with siblings aboard (%)', fontsize=15)
legend(labelspacing=0.25, fontsize = 15)
```

```
Populating the interactive namespace from numpy and matplotlib
```

Out[11]:
```
<matplotlib.legend.Legend at 0x7f2642592c90>
```



3.6. Statistical analysis - Chi-square for 2 proportions Assessement of the significance of differences in proportions between dead people and survivors on the basis of the number of siblings aboard the Titanic

```
In [12]: # This function compares two proportions given two values in the same cate
         from scipy import stats
         def proportion_comparator_for_one_df_row(given_np_row):
             matrix_builder = np.array([[dead_pd_series_3_groups.sum(),given_np_row
                                        [survivors_pd_series_3_groups.sum(),given_np_row
             return  stats.chi2_contingency(matrix_builder)

In [13]: # This function compares proportions row-wise along the dataframe
         def proportion_comparator(siblings_number_df):
             return siblings_number_df.apply(proportion_comparator_for_one_df_row,

In [14]: # Running the functions
         print proportion_comparator(siblings_number_df_3_groups)

0    (2.48450912713, 0.11497221412, 1, [[556.943699...
1    (14.8477778645, 0.000116545296539, 1, [[516.81...
2    (1.70517804084, 0.191612192179, 1, [[547.50368...
dtype: object
```

3.5 3.7. Findings

- passengers with one sibling aboard: the proportion of passenger with 1 sibling aboard who survived was significantly higher than the proportion that died (32.75% vs. 17.67%, p = 0.0001).

- instead, no significant difference was observed in the other groups (passengers without siblings aboard and those with more than 1 sibling aboard)

3.6 3.8. Emerging hypothesis for further studies

- No clear trend in the difference of proportion between the survivors and the dead passengers was observed based on the presence of siblings (or other first-degree relative) aboard (higher proportion of survivors was observed only in the category of passengers with 1 sibling aboard while in the other extremities, i.e. no siblings or more than 2 siblings, the proportion of survivors was lower than the proportion of dead passengers).
- These observations indicate that the improvement in survival odds of passengers with 1 sibling aboard (conversely, the decrease in the proportion of survivors in the other categories) was the consequence of another factor (for instance parents with one child are faster, thus more likely to join lifeboats, than parents with 2 or more). Notably, in the BBC Magazine article on the Titanic, experts also argued that "women and children survived in greater numbers across all classes".

- In this regard, in the second part of our project, we will address whether the age of passengers with siblings aboard participated to the improved survivors proportion in the category of passengers with 1 sibling aboard the Titanic.

---

4 4. Study of the correlation between the passengers' sibling number aboard, the age and survival

4.1. Data wrangling

```
In [15]: # Verification of the availability of the data of interest
         print titanic_df.head()
         print ''
         print titanic_df.isnull().sum()
```

```
   PassengerId  Survived   Age  SibSp
0            1         0  22.0      1
1            2         1  38.0      1
2            3         1  26.0      0
3            4         1  35.0      1
4            5         0  35.0      0

PassengerId      0
Survived         0
```

```
Age            177
SibSp            0
dtype: int64
```

In [16]:
```
# Copying data to avoid affecting titanic_df eventually
# titanic_df kept for fast comparison, should the necessity emerge
age_df = titanic_df.copy(deep=True)

# Dropping axis labels with missing data
age_df_no_nan = age_df.dropna()

# Assessessment of the success of missing data removal
print age_df_no_nan.isnull().sum()
print ''
age_df_no_nan.head()
```

```
PassengerId    0
Survived       0
Age            0
SibSp          0
dtype: int64
```

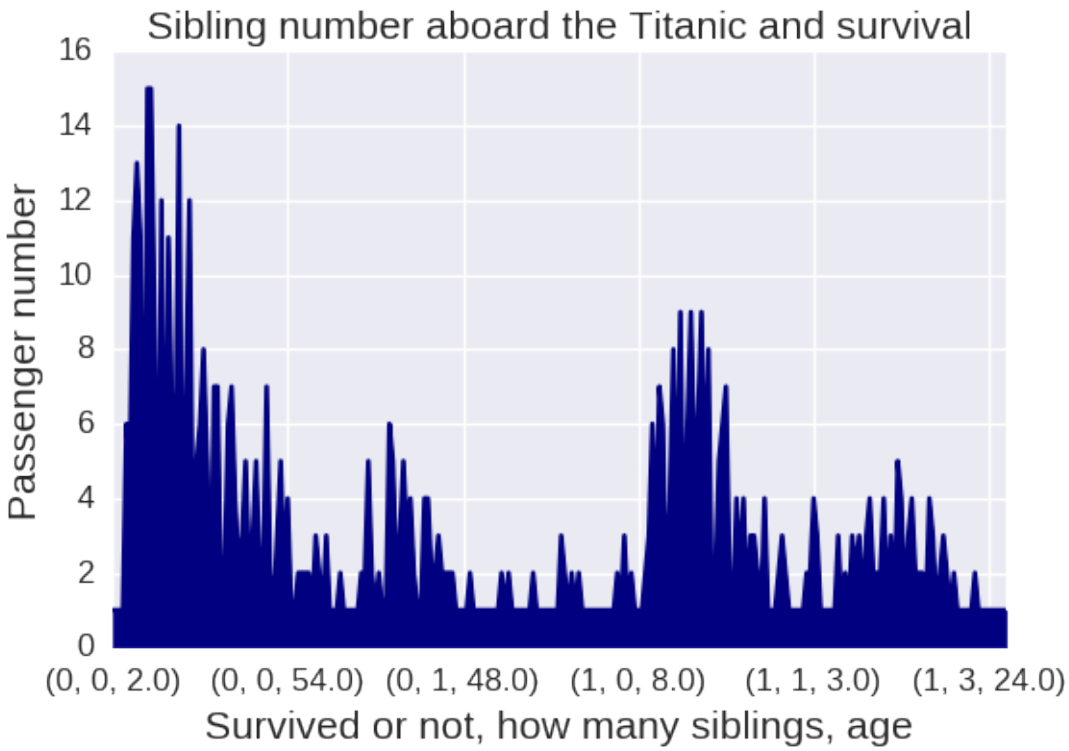
Out[16]:

	PassengerId	Survived	Age	SibSp
0	1	0	22.0	1
1	2	1	38.0	1
2	3	1	26.0	0
3	4	1	35.0	1
4	5	0	35.0	0

In [17]:
```
# Average number of passengers who survived or died grouped according to t
age_no_nan_grouped = age_df_no_nan.groupby(['Survived','SibSp','Age'], as_

import matplotlib.pyplot as plt
import seaborn as sns

% pylab inline
ax = age_no_nan_grouped.plot.area(colormap= 'jet', fontsize=12)

ax.set_title('Sibling number aboard the Titanic and survival', fontsize=15
ax.set_xlabel('Survived or not, how many siblings, age', fontsize=15)
ax.set_ylabel('Passenger number', fontsize=15)
```

```
Populating the interactive namespace from numpy and matplotlib
```

Out[17]: <matplotlib.text.Text at 0x7f26424c5950>



4.1 4.2. Further assessment of data quality

Assessing the age values between 0 and one

```
In [18]: # Resetting row index
         age_df_no_nan = age_df_no_nan.reset_index(drop=True)

         # Age equal to zero
         age_df_no_nan[(age_df_no_nan['Age'] == 0)]
```

```
Out[18]: Empty DataFrame
         Columns: [PassengerId, Survived, Age, SibSp]
         Index: []
```

```
In [19]: # Age between 0 and 1
         age_df_no_nan[(age_df_no_nan['Age'] < 1) & (age_df_no_nan['Age'] > 0)]
```

```
Out[19]:      PassengerId  Survived   Age  SibSp
         59            79         1  0.83      0
         243          306         1  0.92      1
         374          470         1  0.75      2
         509          645         1  0.75      2
         602          756         1  0.67      1
         640          804         1  0.42      0
         664          832         1  0.83      1
```

4.1.1 4.2.1. Observations and course of action

Some passenger are aged between 0 and 1, probably typos.
Data will be replaced by the value without the 0. e.g. 0.83 will become 83

```
In [20]: # Replacing typos 0.x with their value x
         # this method also add value 100 to Siblings column. Will be nullified in
         # instead of == 1 while selecting data (see age_data_parser_survivors func
         age_df_no_nan[(age_df_no_nan['Age'] < 1) & (age_df_no_nan['Age'] > 0)] = 1

         # Verifying that typos were removed
         print age_df_no_nan[(age_df_no_nan['Age'] < 1) & (age_df_no_nan['Age'] > 0
         print ''
         print age_df_no_nan['Age'].iloc[59]
         print age_df_no_nan['Age'].iloc[243]
         print age_df_no_nan['Age'].iloc[374]
```

```
Empty DataFrame
Columns: [PassengerId, Survived, Age, SibSp]
Index: []

83.0
92.0
75.0
```

```
In [21]: def checking_typos (age_df_no_nan):
             for i in range (len(age_df_no_nan)):
                                 return age_df_no_nan[age_df_no_nan[(age_df_no_nan['
```

```
In [22]: # Assessing the number of typos remaining (assuming that age in years is a
         (checking_typos (age_df_no_nan).isnull().sum()[['Age']]) - len(age_df_no_n
```

```
Out[22]: Age    0
         dtype: int64
```

4.1.2 4.2.2. Report on current data status

No additional typo detected

4.2 4.3. Data extraction

```
In [23]: # Getting the age of survivors and the number of siblings
         age_no_nan_survived = age_df_no_nan[age_df_no_nan['Survived'] > 0]#
         # Resetting row index
         age_no_nan_survived = age_no_nan_survived.reset_index(drop=True)

         # Getting the age of survivors and the number of siblings
         age_no_nan_died = age_df_no_nan[age_df_no_nan['Survived'] == 0]
         # Resetting row index
         age_no_nan_died = age_no_nan_died.reset_index(drop=True)
```

```
In [24]: # Making sure than both groups have a high number of values
         print age_no_nan_survived.head()
         print ''
         print 'number of survivor values available:', age_no_nan_survived['Age'].c
         print 'number of dead values available:', age_no_nan_died['Age'].count()
```

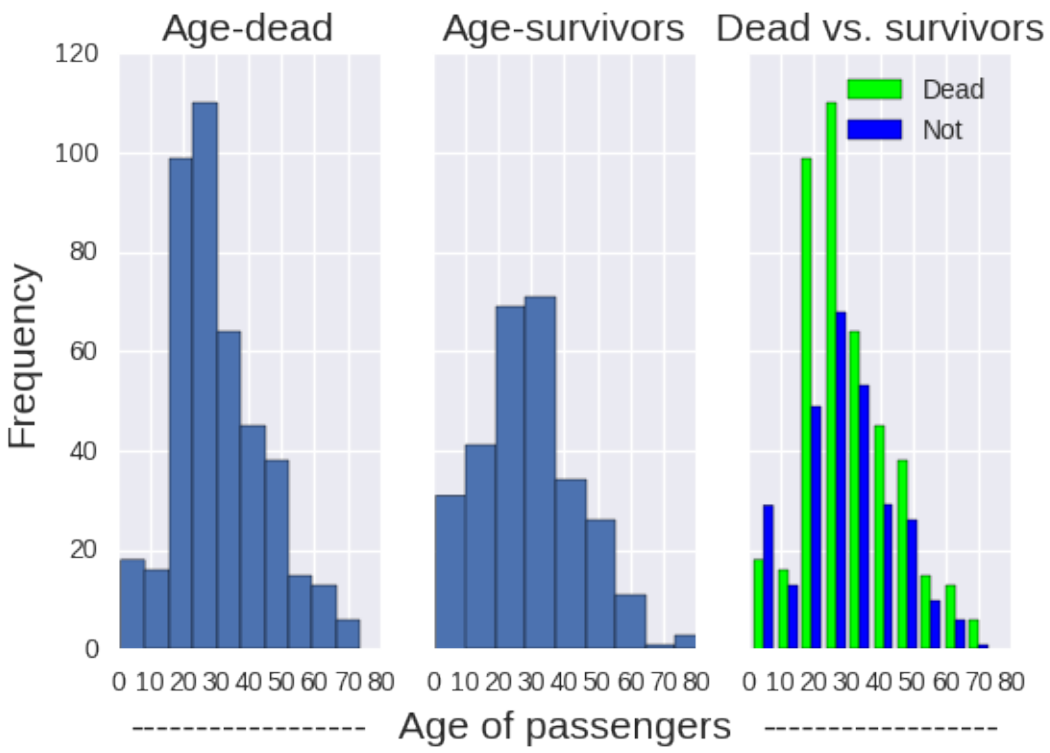
```
   PassengerId  Survived   Age  SibSp
0            2         1  38.0      1
1            3         1  26.0      0
2            4         1  35.0      1
3            9         1  27.0      0
4           10         1  14.0      1

number of survivor values available: 290
number of dead values available: 424
```

4.3 4.4. Age distribution among survivors and dead passengers

4.3.1 4.4.1. Histograms

```
In [25]: # Plotting age distribution of passengers who died or survived

         # Formatting variables for plotting
         a = age_no_nan_died['Age']
         b = age_no_nan_survived['Age']
         c = pd.concat([age_no_nan_died['Age'], age_no_nan_survived['Age']], axis=1
         d = np.array(c)
         n_bins = 10

         fig, axes = plt.subplots(nrows=1, ncols=3, sharex=True , sharey=True)
         ax0, ax1, ax2 = axes.flat

         ax0.hist(a, n_bins, normed=0, histtype='bar')
         ax0.set_title('Age-dead', fontsize=15)
         ax0.set_fontsize=12
         ax0.set_xlabel('------------------', fontsize=15)
         ax0.set_ylabel('Frequency', fontsize=15)
         ax0.set_ylim([0,120])
         ax0.set_xlim([0,80])

         ax1.hist(b, n_bins, normed=0, histtype='bar')
         ax1.set_title('Age-survivors', fontsize=15)
         ax1.set_fontsize=12
         ax1.set_xlabel('Age of passengers', fontsize=15)

         colors = ['lime', 'blue']
         ax2.hist(d, n_bins, normed=0, histtype='bar', color=colors, label=['Dead',
         ax2.legend(prop={'size': 10})
```

```
ax2.set_title('Dead vs. survivors', fontsize=15)
ax2.set_xlabel('------------------', fontsize=15)
ax2.set_fontsize=12

fig.subplots_adjust(hspace=0.7)
```



4.3.2 4.4.2. Observations

Both the distributions of survivors and dead passengers ages are normal, with the mode between 30 and 40 for the dead and between 20 and 30 for the survivors.

4.4 4.5. Study of sub-groups (ages of passengers grouped by the number of siblings)

```
In [26]: # Returns the age of people with a given number of siblings, who survived
         def age_data_parser_survivors (low, high, df):
             prser = df[(df['Survived'] > 0) & (df['SibSp'] > low) & (df['SibSp'] <
             return prser.reset_index(drop=True)

In [27]: # Returns the age of people with a given number of siblings, who died
         def age_data_parser_dead (low, high, df):
             prser = df[(df['Survived'] == 0) & (df['SibSp'] > low) & (df['SibSp']
             return prser.reset_index(drop=True)
```

In [28]:
```
# Siblings: 0
df = age_no_nan_survived
low = -1
high = 1
age_survived_0 = age_data_parser_survivors (low, high, df)

df = age_no_nan_died
age_died_0 = age_data_parser_dead (low, high, df)

print age_survived_0['Age'].describe()
print ''
print age_died_0['Age'].describe()
```

```
count    175.00000
mean      29.94000
std       13.95984
min        1.00000
25%       21.50000
50%       29.00000
75%       36.00000
max       83.00000
Name: Age, dtype: float64

count    296.000000
mean      32.677365
std       13.486835
min        2.000000
25%       22.000000
50%       29.000000
75%       40.125000
max       74.000000
Name: Age, dtype: float64
```

In [29]:
```
# Siblings: 1
df = age_no_nan_survived
low = 0
high = 2
age_survived_1 = age_data_parser_survivors (low, high, df)

df = age_no_nan_died
age_died_1 = age_data_parser_dead (low, high, df)

print age_survived_1['Age'].describe()
print ''
print age_died_1['Age'].describe()
```

```
count    94.000000
mean     29.414894
```

```
std      15.872076
min       1.000000
25%      19.000000
50%      31.000000
75%      39.000000
max      63.000000
Name: Age, dtype: float64

count    86.000000
mean     31.848837
std      12.235212
min       2.000000
25%      25.000000
50%      30.000000
75%      39.750000
max      70.000000
Name: Age, dtype: float64
```

```
In [30]: # Siblings: 2+
         df = age_no_nan_survived
         low = 1
         high = (age_df_no_nan['SibSp'].max())+1
         age_survived_2 = age_data_parser_survivors (low, high, df)

         df = age_no_nan_died
         age_died_2 = age_data_parser_dead (low, high, df)

         print age_survived_2['Age'].describe()
         print ''
         print age_died_2['Age'].describe()
```

```
count    21.000000
mean     34.619048
std      28.817488
min       1.000000
25%      17.000000
50%      24.000000
75%      53.000000
max      92.000000
Name: Age, dtype: float64

count    42.000000
mean     13.666667
std      11.176340
min       1.000000
25%       4.500000
50%       9.500000
```

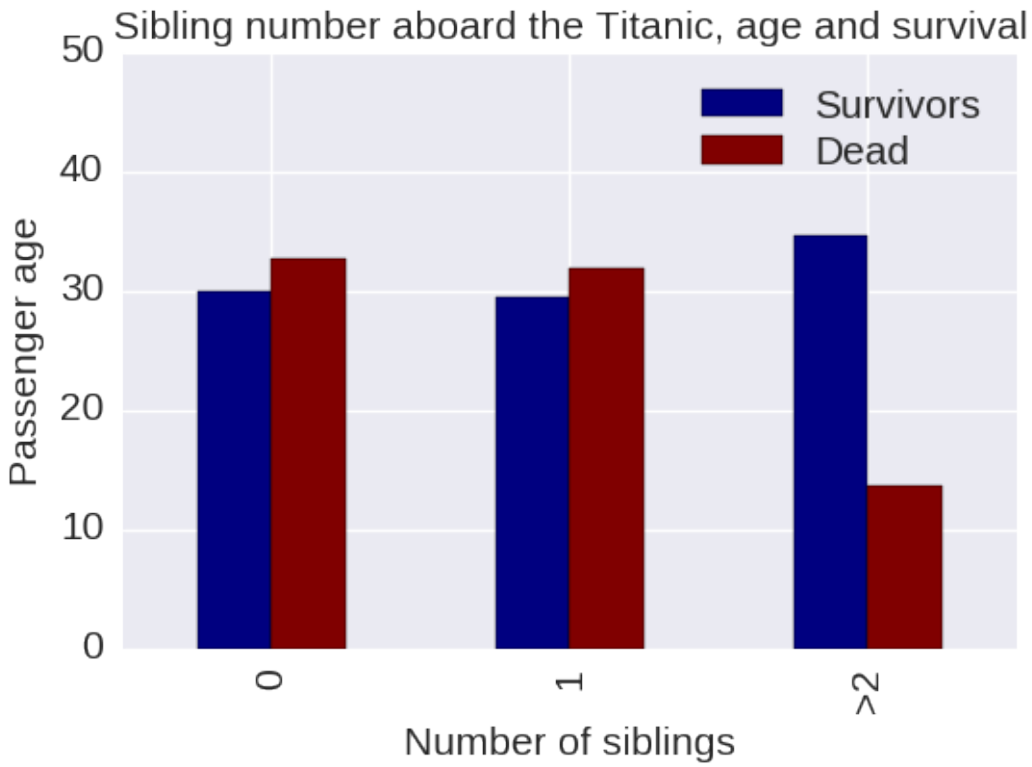
```
75%       20.500000
max       44.000000
Name: Age, dtype: float64
```

4.5 4.6. Qualitative observation on the age of survivors and dead passengers in function of sibling number

In [31]:
```
# Building the survival, in function of average age and number of siblings

survival_siblings_age_df = pd.DataFrame(
    data=[[(age_survived_0['Age'].mean()), (age_died_0['Age'].mean())],
          [(age_survived_1['Age'].mean()), (age_died_1['Age'].mean())],
          [(age_survived_2['Age'].mean()), (age_died_2['Age'].mean())]],
    index=['0','1','>2'],
    columns=['Survivors','Dead']
)
```

In [32]:
```
# Plotting the survivors and dead passengers age in function of sibling nu
import matplotlib.pyplot as plt
import seaborn as sns

#% pylab inline

ax = survival_siblings_age_df.plot.bar(colormap='jet', fontsize=15)

ax.set_title('Sibling number aboard the Titanic, age and survival', fontsi
ax.set_xlabel('Number of siblings', fontsize=15)
ax.set_ylabel('Passenger age', fontsize=15)
ax.set_ylim([0,50])
legend(labelspacing=0.25, fontsize = 15)
```

Out[32]:
```
<matplotlib.legend.Legend at 0x7f2642622190>
```



4.6 4.7. Observations and hypothesis - Statistical analysis (independent Student t-test)

It seems that most people with 2 sibling aboard who survived where older (about 35 in average) than those who died (about 14 in average)

4.6.1 4.7.1. Intergroup comparisons - Dead vs. survivors in groups with same number of siblings aboard

4.7.1.1. No sibling aboard

```
In [33]: import pandas as pd
         import scipy.stats

         # Assessing the statistical significance of diffences in age between survi
         print '(survivor mean, standard deviation):', (age_survived_0['Age'].mean
         print ''
         print '(dead passenger mean, standard deviation):', (age_died_0['Age'].mea
         print ''
         print 'Statistical analysis:', scipy.stats.ttest_ind(age_survived_0['Age']
         print ''
```

```
(survivor mean, standard deviation): (29.94, 13.959839771171502)

(dead passenger mean, standard deviation): (32.67736486486486, 13.486834552735605)

Statistical analysis: Ttest_indResult(statistic=-2.0823281395485731, pvalue=0.03802
```

4.7.1.2. One sibling aboard

```
In [34]: # Assessing the statistical significance of diffences in age between survi

         print '(survivor mean, standard deviation):', (age_survived_1['Age'].mean
         print ''
         print '(dead passenger mean, standard deviation):', (age_died_1['Age'].mea
         print ''
         print 'Statistical analysis:', scipy.stats.ttest_ind(age_survived_1['Age']
         print ''
```

```
(survivor mean, standard deviation): (29.414893617021278, 15.872075684915968)

(dead passenger mean, standard deviation): (31.848837209302324, 12.23521190649009)

Statistical analysis: Ttest_indResult(statistic=-1.157613041816119, pvalue=0.248616
```

4.7.1.3. Two or more siblings aboard

```
In [35]: # Assessing the statistical significance of diffences in age between survi

         print '(survivor mean, standard deviation):', (age_survived_2['Age'].mean
         print ''
         print '(dead passenger mean, standard deviation):', (age_died_2['Age'].mea
         print ''
         print 'Statistical analysis:', scipy.stats.ttest_ind(age_survived_2['Age']
         print ''
```

```
(survivor mean, standard deviation): (34.61904761904762, 28.817488076645734)

(dead passenger mean, standard deviation): (13.666666666666666, 11.1763397007111)

Statistical analysis: Ttest_indResult(statistic=3.2132233296387049, pvalue=0.003846
```

4.7.1.4. Findings:

- passengers with no sibling aboard: those who survived where younger than those who died (29.94±13.96 vs. 32.68±13.49, p= 0.038).

- passengers with one sibling aboard: no statistically significant difference in the age was observed (p = 0.24).

- passengers with more than two siblings aboard: those who died were significantly (about two-fold) younger than those who survived (13.67±11.18 vs. 34.62±28.82, p=0.0038)

4.6.2 4.7.2. Intra-group comparisons - Groups of survivors with same number of siblings aboard

4.7.2.1. Survivors - One sibling against no sibling

```
In [36]: # Assessing the statistical significance of diffences in age between survi
         print '(No siblings mean, standard deviation):', (age_survived_0['Age'].me
         print ''
         print '(One sibling passenger mean, standard deviation):', (age_survived_1
         print ''
         print 'Statistical analysis:', scipy.stats.ttest_ind(age_survived_0['Age']
         print ''
```

```
(No siblings mean, standard deviation): (29.94, 13.959839771171502)

(One sibling passenger mean, standard deviation): (29.414893617021278, 15.872075684

Statistical analysis: Ttest_indResult(statistic=0.26960054896153096, pvalue=0.78779
```

4.7.2.2. Survivors - Two siblings against no sibling

```
In [37]: # Assessing the statistical significance of diffences in age between survi
         print '(No siblings mean, standard deviation):', (age_survived_0['Age'].me
         print ''
         print '(Two sibling passenger mean, standard deviation):', (age_survived_2
         print ''
         print 'Statistical analysis:', scipy.stats.ttest_ind(age_survived_0['Age']
         print ''
```

```
(No siblings mean, standard deviation): (29.94, 13.959839771171502)

(Two sibling passenger mean, standard deviation): (34.61904761904762, 28.8174880766

Statistical analysis: Ttest_indResult(statistic=-0.73380501659565489, pvalue=0.4711
```

4.7.2.3. Survivors - Two siblings against one

```
In [38]: # Assessing the statistical significance of diffences in age between survi
         print '(One sibling mean, standard deviation):', (age_survived_1['Age'].me
         print ''
         print '(Two siblings passenger mean, standard deviation):', (age_survived_
         print ''
         print 'Statistical analysis:', scipy.stats.ttest_ind(age_survived_1['Age']
         print ''
```

```
(One sibling mean, standard deviation): (29.414893617021278, 15.872075684915968)

(Two siblings passenger mean, standard deviation): (34.61904761904762, 28.817488076

Statistical analysis: Ttest_indResult(statistic=-0.80087459401018068, pvalue=0.4314
```

4.7.2.4. Findings - Survivors No significant difference was observed between ages of survivors with more than 2, only one or no siblings aboard the Titanic (ages were around 30 in all groups)

4.6.3 4.7.3. Intra-group comparisons - Groups of dead passengers with same number of siblings aboard

4.7.3.1. Dead - One sibling against no sibling

```
In [39]: # Assessing the statistical significance of diffences in age between dead
         print '(No siblings mean, standard deviation):', (age_died_0['Age'].mean()
         print ''
         print '(One sibling passenger mean, standard deviation):', (age_died_1['Ag
         print ''
         print 'Statistical analysis:', scipy.stats.ttest_ind(age_died_0['Age'], ag
         print ''
```

```
(No siblings mean, standard deviation): (32.67736486486486, 13.486834552735605)

(One sibling passenger mean, standard deviation): (31.848837209302324, 12.235211906

Statistical analysis: Ttest_indResult(statistic=0.53987354244308416, pvalue=0.59008
```

4.7.3.2. Dead - Two siblings against no sibling

```
In [40]: # Assessing the statistical significance of diffences in age between dead
         print '(No siblings mean, standard deviation):', (age_died_0['Age'].mean()
         print ''
         print '(Two sibling passenger mean, standard deviation):', (age_died_2['Ag
         print ''
```

```
        print 'Statistical analysis:', scipy.stats.ttest_ind(age_died_0['Age'], ag
        print ''
```

```
(No siblings mean, standard deviation): (32.67736486486486, 13.486834552735605)

(Two sibling passenger mean, standard deviation): (13.666666666666666, 11.17633970

Statistical analysis: Ttest_indResult(statistic=10.035461227542809, pvalue=2.11228
```

4.7.3.3. Dead - Two siblings against one

```
In [41]: # Assessing the statistical significance of diffences in age between dead
        print '(One sibling mean, standard deviation):', (age_died_1['Age'].mean()
        print ''
        print '(Two siblings passenger mean, standard deviation):', (age_died_2['A
        print ''
        print 'Statistical analysis:', scipy.stats.ttest_ind(age_died_1['Age'], ag
        print ''
```

```
(One sibling mean, standard deviation): (31.848837209302324, 12.23521190649009)

(Two siblings passenger mean, standard deviation): (13.666666666666666, 11.1763397

Statistical analysis: Ttest_indResult(statistic=8.373668527332816, pvalue=7.718624
```

4.7.3.4. Findings - Dead passengers

- No significant difference was observed in terms of age between people without siblings (32.678±13.49 years) and people with only one sibling aboard (31.85±12.26)
- People with two sibling aboard were two-fold younger than the other groups (13.67±11.18, p= 2.11e-14 against passenger without siblings and p= 7.72e-13 against passenger with one sibling aboard)

---

5 5. Tentative conclusions and Hypothesis for future studies

- No significant difference was observed between survivors and the dead passengers in the category of passengers with one sibling (or other first-degree relative) aboard. Therefore, our data do not support that the priority of boarding for children in lifeboats and the better motility of a parent with one child compared to one with more than 2 children participated to the higher proportion of survivors in the category of passengers with 1 sibling aboard the Titanic.

- Nonetheless, the second part of the study revealed that among passengers with no sibling aboard, those who survived where significantly younger than those who died (29.94±13.96 vs. 32.68±13.49, $p < 0.05$). Although our data cannot be used to draw strong conclusions, the present finding is in agreement with experts arguing that there was no orderly evacuation, i.e. survival of the strongest (healthy young adults) was the only rule.
- Not surprisingly and in alignment with this hypothesis (and contrary to BBC hypothesis of women and children priority for boarding and Royal navy ethical codes), passengers who died were two-fold younger than those who survived (13.67±11.18 vs. 34.62±28.82, $p < 0.01$). Actually, with an average age of about 14 years old, it appears that passengers with more than 2 siblings aboard who died were mainly children. No other category considered in our study was as young (ages were around 30 in other sibling-based groups).
- For future studies: *(i)* considering that "third class passengers had to find their way through a maze of corridors and staircases to reach the boat deck", and that "less than one third of steerage passengers survived" BBC", it could be interesting to assess the impact of the socio-economic condition (as revealed by the class in which passengers were traveling) on the proportion of passengers, and mainly children, who survived or died; *(ii)* It could also be interesting to assess the impact of sex, notably the number of women surviving (as they were also supposed to have priority) on the proportion of survivors (or survival odds).

---

6 6. Code sources

https://sourceforge.net/p/ipython/discussion/markdown_syntax
http://github.com/adam-p/markdown-here/wiki/Markdown-Here-Cheatsheet
http://pandas.pydata.org/pandas-docs/stable/missing_data.html
http://stackoverflow.com/questions/13411544/delete-column-from-pandas-dataframe
http://pandas.pydata.org/pandas-docs/stable/groupby.html
http://docs.scipy.org/doc/numpy/reference/generated/numpy.append.html#numpy.append
http://stackoverflow.com/questions/18062135/combining-two-series-into-a-dataframe-in-pandas
http://stackoverflow.com/questions/11346283/renaming-columns-in-pandas
http://stackoverflow.com/posts/13337376/revisions
http://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.apply.html
http://stackoverflow.com/questions/21487329/add-x-and-y-labels-to-a-pandas-plot
http://stackoverflow.com/questions/7125009/how-to-change-legend-size-with-matplotlib-pyplot
http://manishamde.github.io/blog/2013/03/07/pandas-and-python-top-10/
http://stackoverflow.com/questions/13842088/set-value-for-particular-cell-in-pandas-dataframe
http://stackoverflow.com/questions/20490274/how-to-reset-index-in-a-pandas-data-frame
http://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.plot.bar.html
http://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.plot.bar.html
http://pandas.pydata.org/pandas-docs/stable/visualization.html
http://docs.scipy.org/doc/scipy-0.15.1/reference/generated/scipy.stats.ttest_ind.html

http://stackoverflow.com/questions/20675534/t-test-on-pandas-dataframes-and-make-a-new-matrix-of-resulting-p-values
http://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.ttest_ind.html
http://docs.scipy.org/doc/scipy-0.15.1/reference/generated/scipy.stats.ttest_ind.html
http://stackoverflow.com/questions/10038543/tracking-down-the-assumptions-made-by-scipys-ttest-ind-function
http://stackoverflow.com/questions/2849286/python-matplotlib-subplot-how-to-set-the-axis-range
http://matplotlib.org/examples/pylab_examples/subplots_demo.html
http://matplotlib.org/examples/statistics/histogram_demo_multihist.html