ASumbaraju_wk6_TitanicCaseStudyPart1

July 13, 2021

0.1 DSC 550

0.2 Week 6

0.3 Exercise 6.2

0.4 Aditya Sumbaraju

0.5 07/13/2021

1 Titanic Tutorial Part 1 || Graphics Analysis

```python
[3]: import pandas as pd
     import yellowbrick
```

```python
[4]: class color:

         BOLD = '\033[1m'
         END = '\033[0m'
```

2 Step 1: Load data into a dataframe

```python
[15]: addr1 = "C:\BU\DSC550\wk6\week-6/train.csv"
      data = pd.read_csv(addr1)
      data
```

```
[15]:      PassengerId  Survived  Pclass  \
      0              1         0       3
      1              2         1       1
      2              3         1       3
      3              4         1       1
      4              5         0       3
      ..           ...       ...     ...
      886          887         0       2
      887          888         1       1
      888          889         0       3
      889          890         1       1
      890          891         0       3
```

```
                                            Name     Sex   Age  SibSp  \
0                          Braund, Mr. Owen Harris    male  22.0      1
1    Cumings, Mrs. John Bradley (Florence Briggs Th…  female  38.0      1
2                           Heikkinen, Miss. Laina  female  26.0      0
3      Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                         Allen, Mr. William Henry    male  35.0      0
..                                             …       …     …      …
886                        Montvila, Rev. Juozas    male  27.0      0
887                 Graham, Miss. Margaret Edith  female  19.0      0
888     Johnston, Miss. Catherine Helen "Carrie"  female   NaN      1
889                        Behr, Mr. Karl Howell    male  26.0      0
890                          Dooley, Mr. Patrick    male  32.0      0

     Parch            Ticket     Fare Cabin Embarked
0        0         A/5 21171   7.2500   NaN        S
1        0          PC 17599  71.2833   C85        C
2        0  STON/O2. 3101282   7.9250   NaN        S
3        0            113803  53.1000  C123        S
4        0            373450   8.0500   NaN        S
..     …               …       …     …        …
886      0            211536  13.0000   NaN        S
887      0            112053  30.0000   B42        S
888      2        W./C. 6607  23.4500   NaN        S
889      0            111369  30.0000  C148        C
890      0            370376   7.7500   NaN        Q

[891 rows x 12 columns]
```

3 Step 2: check the dimension of the table

```
[6]: print("The dimension of the table is: ", data.shape)
```

```
The dimension of the table is:  (891, 12)
```

4 Step 3: Look at the data

```
[7]: data.head(5)
```

```
[7]:    PassengerId  Survived  Pclass  \
0               1         0       3
1               2         1       1
2               3         1       3
3               4         1       1
4               5         0       3

                                Name     Sex   Age  SibSp  \
```

```
0                              Braund, Mr. Owen Harris    male  22.0      1
1   Cumings, Mrs. John Bradley (Florence Briggs Th…  female  38.0      1
2                               Heikkinen, Miss. Laina  female  26.0      0
3          Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                             Allen, Mr. William Henry    male  35.0      0

    Parch             Ticket     Fare Cabin Embarked
0       0          A/5 21171   7.2500   NaN        S
1       0           PC 17599  71.2833   C85        C
2       0  STON/O2. 3101282   7.9250   NaN        S
3       0             113803  53.1000  C123        S
4       0             373450   8.0500   NaN        S
```

5 Step 5: what type of variables are in the table

```python
print(color.BOLD + "Describe Data" + color.END)
print(data.describe())
print(color.BOLD + "Summarized Data" + color.END)
print(data.describe(include=['O']))
```

```
Describe Data
       PassengerId    Survived      Pclass         Age       SibSp  \
count   891.000000  891.000000  891.000000  714.000000  891.000000
mean    446.000000    0.383838    2.308642   29.699118    0.523008
std     257.353842    0.486592    0.836071   14.526497    1.102743
min       1.000000    0.000000    1.000000    0.420000    0.000000
25%     223.500000    0.000000    2.000000   20.125000    0.000000
50%     446.000000    0.000000    3.000000   28.000000    0.000000
75%     668.500000    1.000000    3.000000   38.000000    1.000000
max     891.000000    1.000000    3.000000   80.000000    8.000000

            Parch        Fare
count  891.000000  891.000000
mean     0.381594   32.204208
std      0.806057   49.693429
min      0.000000    0.000000
25%      0.000000    7.910400
50%      0.000000   14.454200
75%      0.000000   31.000000
max      6.000000  512.329200
Summarized Data
                               Name   Sex    Ticket Cabin Embarked
count                           891   891       891   204      889
unique                          891     2       681   147        3
top     Lemore, Mrs. (Amelia Milley)  male  CA. 2343    G6        S
freq                              1   577         7     4      644
```

6 Step 6: import visualization packages

7 draw histograms
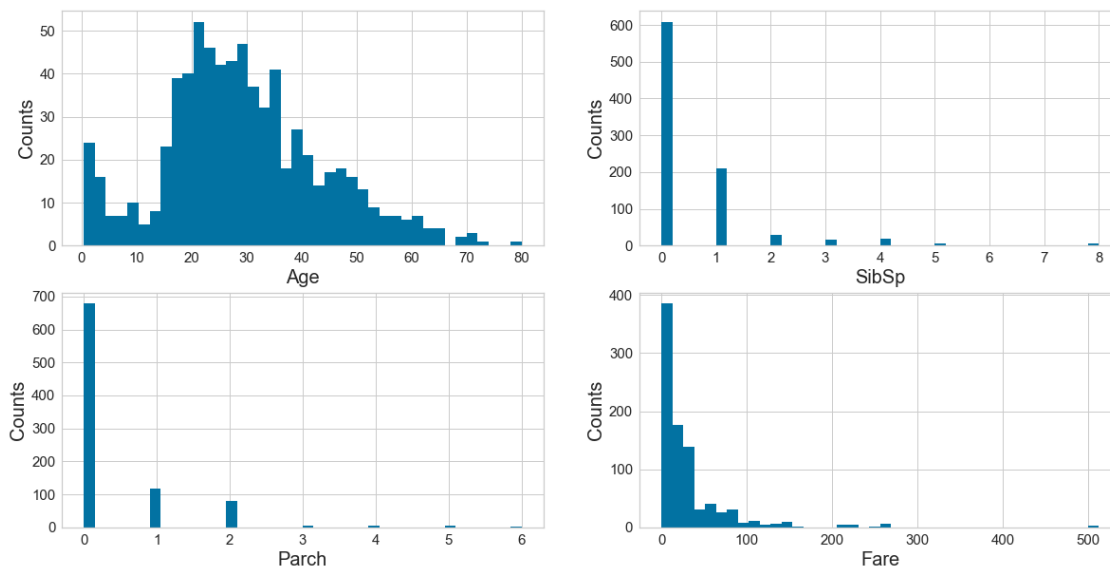
```
[9]:  import matplotlib.pyplot as plt
```

```
[10]:  # set up the figure size
       plt.rcParams['figure.figsize'] = (20, 10)

       # make subplots
       fig, axes = plt.subplots(nrows = 2, ncols = 2)

       # Specify the features of interest
       num_features = ['Age', 'SibSp', 'Parch', 'Fare']
       xaxes = num_features
       yaxes = ['Counts', 'Counts', 'Counts', 'Counts']

       # draw histograms
       axes = axes.ravel()
       for idx, ax in enumerate(axes):
           ax.hist(data[num_features[idx]].dropna(), bins=40)
           ax.set_xlabel(xaxes[idx], fontsize=20)
           ax.set_ylabel(yaxes[idx], fontsize=20)
           ax.tick_params(axis='both', labelsize=15)
       #plt.show()
```

8 Step 7: Barcharts: set up the figure size

```
[11]: #%matplotlib inline
      plt.rcParams['figure.figsize'] = (20, 10)

      # make subplots
      fig, axes = plt.subplots(nrows = 2, ncols = 2)

      # make the data read to feed into the visualizer
      X_Survived = data.replace({'Survived': {1: 'yes', 0: 'no'}}).
       ↪groupby('Survived').size().reset_index(name='Counts')['Survived']
      Y_Survived = data.replace({'Survived': {1: 'yes', 0: 'no'}}).
       ↪groupby('Survived').size().reset_index(name='Counts')['Counts']
      # make the bar plot
      axes[0, 0].bar(X_Survived, Y_Survived)
      axes[0, 0].set_title('Survived', fontsize=25)
      axes[0, 0].set_ylabel('Counts', fontsize=20)
      axes[0, 0].tick_params(axis='both', labelsize=15)

      # make the data read to feed into the visualizer
      X_Pclass = data.replace({'Pclass': {1: '1st', 2: '2nd', 3: '3rd'}}).
       ↪groupby('Pclass').size().reset_index(name='Counts')['Pclass']
      Y_Pclass = data.replace({'Pclass': {1: '1st', 2: '2nd', 3: '3rd'}}).
       ↪groupby('Pclass').size().reset_index(name='Counts')['Counts']
      # make the bar plot
      axes[0, 1].bar(X_Pclass, Y_Pclass)
      axes[0, 1].set_title('Pclass', fontsize=25)
      axes[0, 1].set_ylabel('Counts', fontsize=20)
      axes[0, 1].tick_params(axis='both', labelsize=15)

      # make the data read to feed into the visualizer
      X_Sex = data.groupby('Sex').size().reset_index(name='Counts')['Sex']
      Y_Sex = data.groupby('Sex').size().reset_index(name='Counts')['Counts']
      # make the bar plot
      axes[1, 0].bar(X_Sex, Y_Sex)
      axes[1, 0].set_title('Sex', fontsize=25)
      axes[1, 0].set_ylabel('Counts', fontsize=20)
      axes[1, 0].tick_params(axis='both', labelsize=15)

      # make the data read to feed into the visualizer
      X_Embarked = data.groupby('Embarked').size().
       ↪reset_index(name='Counts')['Embarked']
      Y_Embarked = data.groupby('Embarked').size().
       ↪reset_index(name='Counts')['Counts']
      # make the bar plot
      axes[1, 1].bar(X_Embarked, Y_Embarked)
      axes[1, 1].set_title('Embarked', fontsize=25)
```

```
axes[1, 1].set_ylabel('Counts', fontsize=20)
axes[1, 1].tick_params(axis='both', labelsize=15)
#plt.show()
```
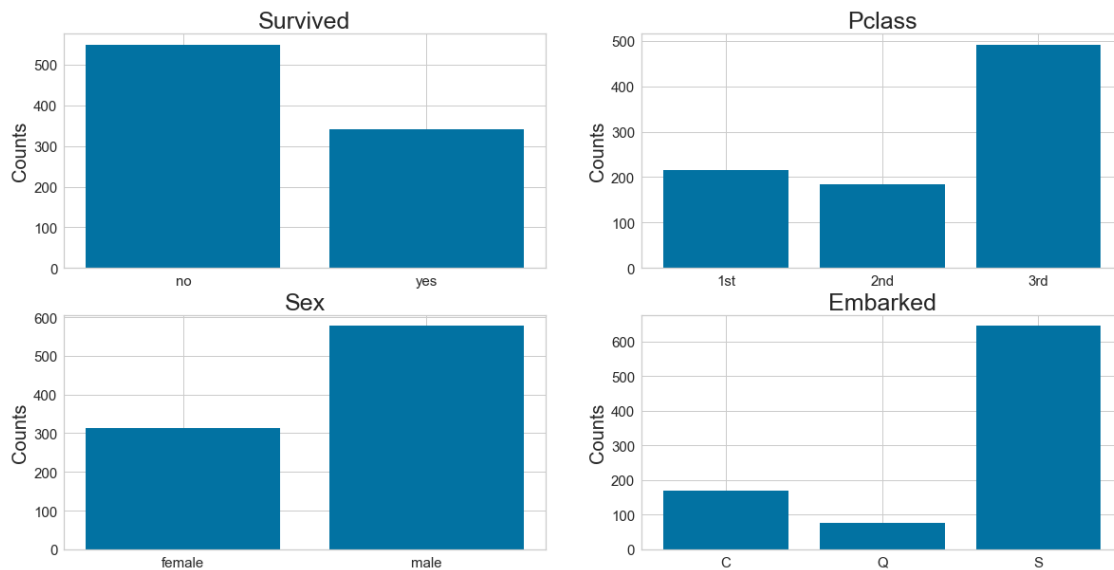

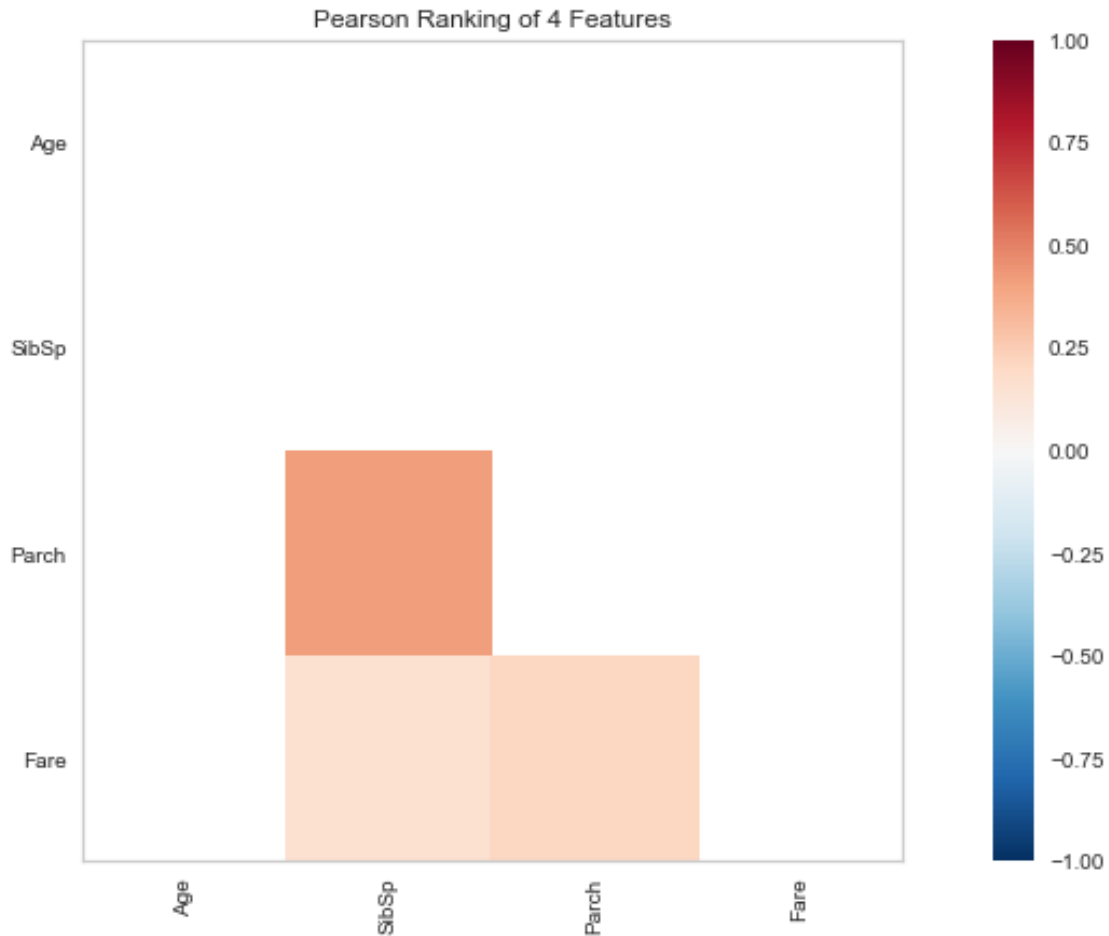# 9 Step 8: Pearson Ranking

```
[12]:  #set up the figure size
       #%matplotlib inline
       plt.rcParams['figure.figsize'] = (15, 7)

       # import the package for visulization of the correlation
       from yellowbrick.features import Rank2D

       # extract the numpy arrays from the data frame
       # X = data[num_features].as_matrix()
       X = data[num_features].values

       # instantiate the visualizer with the Covariance ranking algorithm
       visualizer = Rank2D(features=num_features, algorithm='pearson')
       visualizer.fit(X)                    # Fit the data to the visualizer
       visualizer.transform(X)              # Transform the data
       visualizer.poof(outpath="C:\BU\DSC550\wk6/pcoords1.png") # Draw/show/poof the
        ↪data
       #plt.show()
```

```
[12]:  <AxesSubplot:title={'center':'Pearson Ranking of 4 Features'}>
```

Pearson Ranking of 4 Features

10 Step 9: Compare variables against Survived and Not Survived

```
[13]: #set up the figure size
      #%matplotlib inline
      plt.rcParams['figure.figsize'] = (15, 7)
      plt.rcParams['font.size'] = 50

      # setup the color for yellowbrick visualizer
      from yellowbrick.style import set_palette
      set_palette('sns_bright')

      # import packages
      from yellowbrick.features import ParallelCoordinates
      # Specify the features of interest and the classes of the target
      classes = ['Not-survived', 'Survived']
      num_features = ['Age', 'SibSp', 'Parch', 'Fare']
```

```
# copy data to a new dataframe
data_norm = data.copy()
# normalize data to 0-1 range
for feature in num_features:
    data_norm[feature] = (data[feature] - data[feature].mean(skipna=True)) /␣
 ↪(data[feature].max(skipna=True) - data[feature].min(skipna=True))


# Extract the numpy arrays from the data frame
X = data_norm[num_features].values
y = data.Survived.values

# Instantiate the visualizer
# Instantiate the visualizer
visualizer = ParallelCoordinates(classes=classes, features=num_features)



visualizer.fit(X, y)       # Fit the data to the visualizer
visualizer.transform(X)    # Transform the data
visualizer.poof(outpath="C:\BU\DSC550\wk6/pcoords2.png") # Draw/show/poof the␣
 ↪data
plt.show();
```
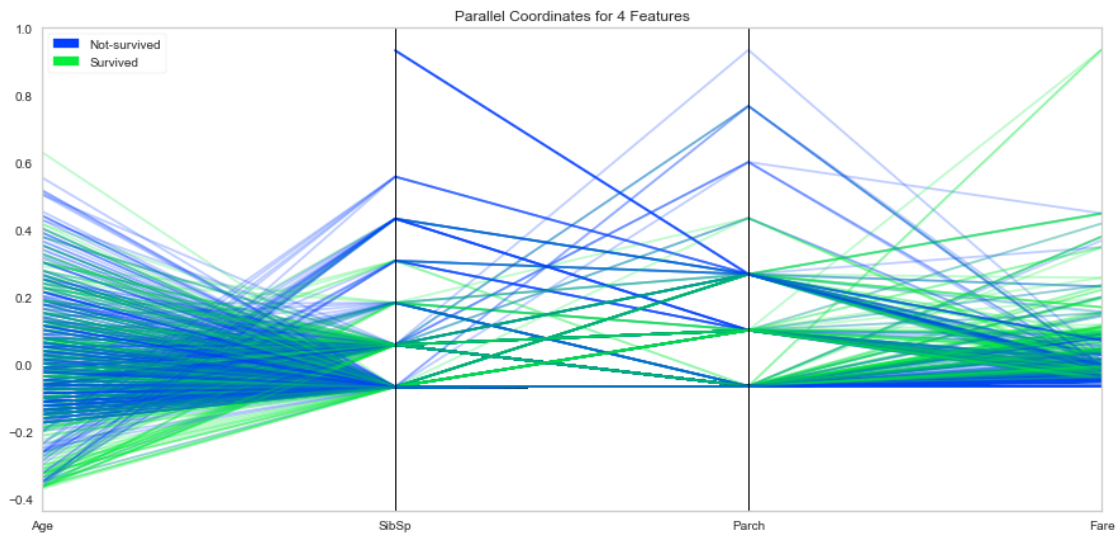

11 Step 10 - stacked bar charts to compare survived/not survived

```
[14]: #set up the figure size
      #%matplotlib inline
      plt.rcParams['figure.figsize'] = (20, 10)
```

```python
# make subplots
fig, axes = plt.subplots(nrows = 2, ncols = 2)

# make the data read to feed into the visulizer
Sex_survived = data.replace({'Survived': {1: 'Survived', 0:
 ↪'Not-survived'}})[data['Survived']==1]['Sex'].value_counts()
Sex_not_survived = data.replace({'Survived': {1: 'Survived', 0:
 ↪'Not-survived'}})[data['Survived']==0]['Sex'].value_counts()
Sex_not_survived = Sex_not_survived.reindex(index = Sex_survived.index)
# make the bar plot
p1 = axes[0, 0].bar(Sex_survived.index, Sex_survived.values)
p2 = axes[0, 0].bar(Sex_not_survived.index, Sex_not_survived.values,
 ↪bottom=Sex_survived.values)
axes[0, 0].set_title('Sex', fontsize=25)
axes[0, 0].set_ylabel('Counts', fontsize=20)
axes[0, 0].tick_params(axis='both', labelsize=15)
axes[0, 0].legend((p1[0], p2[0]), ('Survived', 'Not-survived'), fontsize = 15)

# make the data read to feed into the visualizer
Pclass_survived = data.replace({'Survived': {1: 'Survived', 0:
 ↪'Not-survived'}}).replace({'Pclass': {1: '1st', 2: '2nd', 3:
 ↪'3rd'}})[data['Survived']==1]['Pclass'].value_counts()
Pclass_not_survived = data.replace({'Survived': {1: 'Survived', 0:
 ↪'Not-survived'}}).replace({'Pclass': {1: '1st', 2: '2nd', 3:
 ↪'3rd'}})[data['Survived']==0]['Pclass'].value_counts()
Pclass_not_survived = Pclass_not_survived.reindex(index = Pclass_survived.index)
# make the bar plot
p3 = axes[0, 1].bar(Pclass_survived.index, Pclass_survived.values)
p4 = axes[0, 1].bar(Pclass_not_survived.index, Pclass_not_survived.values,
 ↪bottom=Pclass_survived.values)
axes[0, 1].set_title('Pclass', fontsize=25)
axes[0, 1].set_ylabel('Counts', fontsize=20)
axes[0, 1].tick_params(axis='both', labelsize=15)
axes[0, 1].legend((p3[0], p4[0]), ('Survived', 'Not-survived'), fontsize = 15)

# make the data read to feed into the visualizer
Embarked_survived = data.replace({'Survived': {1: 'Survived', 0:
 ↪'Not-survived'}})[data['Survived']==1]['Embarked'].value_counts()
Embarked_not_survived = data.replace({'Survived': {1: 'Survived', 0:
 ↪'Not-survived'}})[data['Survived']==0]['Embarked'].value_counts()
Embarked_not_survived = Embarked_not_survived.reindex(index = Embarked_survived.
 ↪index)
# make the bar plot
p5 = axes[1, 0].bar(Embarked_survived.index, Embarked_survived.values)
p6 = axes[1, 0].bar(Embarked_not_survived.index, Embarked_not_survived.values,
 ↪bottom=Embarked_survived.values)
```

```
axes[1, 0].set_title('Embarked', fontsize=25)
axes[1, 0].set_ylabel('Counts', fontsize=20)
axes[1, 0].tick_params(axis='both', labelsize=15)
axes[1, 0].legend((p5[0], p6[0]), ('Survived', 'Not-survived'), fontsize = 15)
#plt.show()
```

[14]: <matplotlib.legend.Legend at 0x22cb0520e50>